Troubleshooting Macintosh Networks

Covers AppleTalk and System 7

Kurt VanderSluis and Amr Eissa

Learn to

- Identify and correct network errors
- Troubleshoot printer problems
- Diagnose and correct router problems

DISK INCLUDED

Disk contains AppleTalk Reference and PacketSend, two hypercard stacks especially useful for network troubleshooters.

M&T BOOKS

Troubleshooting Macintosh Networks

Troubleshooting Macintosh Networks

A comprehensive guide to troubleshooting and debugging Macintosh networks.

M&T BOOKS

Kurt VanderSluis
and Amr Eissa

M&T Books
A Division of MIS:Press
A Subsidiary of Henry Holt and Company, Inc.
115 West 18th Street
New York, New York 10011

Printed in the United States of America

Limits of Liability and Disclaimer of Warranty
The Author and Publisher of this book have used their best efforts in preparing the book and the programs contained in it. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness.

The Author and Publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The Author and Publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

Library of Congress Cataloging-in-Publication Data (to be assigned)

VanderSluis, Kurt and Amr Eissa
Troubleshooting Macintosh Networks / Kurt VanderSluis and Amr Eissa
p. cm.
Includes index.
ISBN 1-55851-315-9

96 95 94 93 4 3 2 1

Project Editor: Mike Welch
Technical Editor: Scott Smith
Copy Editor: Anne Kandra
Cover Design: Lauren Smith Design

Contents

Why This Book Is for You

If you're a network supervisor, consultant, or installer responsible for maintaining a productive Macintosh network, this book is for you. It contains detailed information on troubleshooting AppleTalk and System 7, plus dozens of examples that provide practical methods for troubleshooting network problems.

- If you're new to troubleshooting networks, you'll appreciate the easy-to-understand explanations. The authors present technical details in a friendly, informative tone that clearly conveys their troubleshooting experience and techniques. You'll also benefit from specific, organized instructions on what to try when troubleshooting network problems.

- Experienced network troubleshooters will find in-depth discussions of key network troubleshooting topics. You'll learn how to choose the most effective troubleshooting mode: Random, Hunch, Set, Layer, or Process. You'll also learn how to correct printing problems, identify and solve router errors, and more.

The companion disk contains two HyperCard stacks, AppleTalk Reference and PacketSend, filled with information you'll find helpful when troubleshooting your Macintosh network.

Introduction

Network troubleshooting is difficult for two reasons: Networks are hard to understand and troubleshooting is hard to perform. We believe that when you face a difficult task, the best course of action is to admit that it's difficult, discover why it's difficult, and do what you can to overcome the difficulties. It certainly isn't constructive to pretend that the difficulty doesn't exist, because then you'll do all the easy things that you can think of and then give up, shaking your head and muttering that you can't understand why the measures you took weren't effective.

Sometimes, to overcome a problem, you need to change your normal approach. For instance, mathematicians know many kinds of equations and have elegant solutions to lots of them. But other equations—transcendental equations, for example—are solved only by using brute force. Still others require that you combine elegance, to get near to the solution, and brute computational force, to home in on the exact answer. It's important to have more than one approach to a situation.

A World of Abstraction

Networks are difficult to understand because they require you to create mental linkages between several abstract concepts. This chain of abstractions is analogous to the child's song "leaf on the twig on the branch on the limb on the stump in the hole of the bottom of the sea." Some people see the leaf poking out from the water and can't imagine anything else. Troubleshooters must conceive the following:

A screen projects *images*, which are controlled by an
application, which performs a specific set of computing tasks under the guidance of an
operating system, which works in cooperation with a set of
network protocols to manage the sending and receiving of data between systems using
network hardware to provide electronic signal generation and reception over a
physical wiring plant that is interconnected by a configuration of repeaters, bridges, routers, and gateways.

Understanding the linkage is incredibly important. Networks are difficult to troubleshoot and manage because they consist of a great number of different kinds of components (perhaps acquired from different vendors) interacting together. Small changes in one component in one part of the system can produce enormous changes in the operation of another component far away in the network.

Using Your "Pre-Computer" Expertise

Many people who work in the computer industry today came to it from some other field. In fact, your "pre-computer" background can help you understand computer concepts.

If you grew up on a farm, for example, you can easily borrow ideas from your knowledge of agriculture. You know that before you can plant a field, the ground must be prepared a certain way: the soil must have the right ingredients, and it must be properly irrigated and weeded to produce a healthy crop. Networks need that same kind of preparation and care to produce good results.

Similarly, if you're trained as a nurse, you know that a person has needs, both physical and emotional, that must be met for that person to remain healthy. You know how to perform actions that minister to these needs to promote a person's health. You can relate these actions to the actions you would take on a network: inoculating the network against viruses or scanning the network's vital signs to get a picture of its condition. You might have been an athlete, a car mechanic, a visual artist, a real estate agent, an accountant, or a community organizer. Or maybe you liked just watching ants. Whatever your background, look for ways you can apply the knowledge you already have to the task of becoming a computer or network expert.

Read and Learn Both Practice and Theory

In the computer field, it's impossible to make progress or even maintain your level of competence without frequent and aggressive reading and learning. In computing, everything is constantly changing, so the computer experts must always be learning something new.

This is both our blessing and our curse. It seems like a paradox, but the more you know, the faster you must learn just to maintain that same level of knowledge. That's why there's so much written material about every aspect of computing: artificial intelligence (AI), operating systems, connectivity and telecommunications, graphics and publishing, hypermedia and multimedia, and so on.

It's now almost to the point where the publishing industry isn't fast enough to keep up with the rapid changes taking place in the world of computers. That's why it's so important to learn the theory behind your area of expertise. Although someone in metallurgy might invent a new method for the protective thermal spray of a metal this year, the principles governing the diffusion of one metal into another remain the same as they ever were.

In Macintosh computing, it's very important to read weekly magazines such as *MacWeek* to learn about the latest network gadgets and inventions. But it's equally important that you clearly understand the principles of networking so that each new gadget and invention can fit into a larger context than simply "what does it do?" The better you understand fundamental networking principles, the better you'll be able to cope with new products and new ways of accomplishing the same old thing—moving ideas between people.

In this book, we've tried to provide the right combination of theory and practice. The goal is to include enough practice so that you'll be able to use the book to help solve practical problems, and enough theory so that even five years from now (the working equivalent of a generation in networking), some of the advice and methods will still apply even though the products will be different.

Many books on the market are more heavily slanted toward theory, and some of these are very worthwhile reading. For example, one of our favorite networking books, Andrew Tannenbaum's *Computer Networks* (Prentice Hall, 1988), is virtually all theory, using examples only to illustrate how the theory takes form. An advantage of this book is that it explains the process of designing a protocol and points out how each design decision makes certain things more feasible and other things more difficult.

Other books are more slanted toward practice. For example, Farallon's (LocalTalk) StarController manual is probably the best book we've seen on the subject of building a LocalTalk network. The simple fact is, you need both theory and practical information to design, build, and maintain an effective network.

Can You Be Both Impulsive and Methodical?

Some people are very impulsive. Although they may often get themselves in hot water, they also gain a lot of experience about many things. Other people are very methodical, and while the kinds of experience they gain cover less territory, the methodical types usually develop a very deep understanding of the subjects that interest them. Both approaches—the impulsive and the methodical—can be valuable in troubleshooting.

When troubleshooters arrive on a crisis scene, they frequently plunge almost immediately into action, trying this and that until they've exhausted all their ideas. At that point, they begin a more careful, methodical approach to the problem.

This is much like the way some people behave when they misplace their car keys. As soon as they notice that their keys are missing, they begin looking in all the places they usually keep the keys. Not until they exhaust all of these possibilities do they begin a more methodical search procedure, asking themselves, "Where did I last see my car keys?" or "Who was the last person to use the car?" At that point, they begin exploring the house or office in a methodical fashion, closing doors to rooms that have already been thoroughly searched.

Impulsive methods are sometimes the best way to proceed; sometimes they locate the car keys or fix the network. But impulsive fixes can also prove ineffective or even cause more problems than the ones that existed before troubleshooting began. Then you must use a more methodical strategy.

Methodical approaches are more likely to find subtle or hidden causes of trouble, but sometimes the narrowness of the method can lead you away from the cause of the problem. For the methodical approach to work, you must already be on the right track. For example, if you go out to start your car in the morning and the car emits no sound when you turn the key, a methodical troubleshooting of the fuel system will not identify the problem.

Using This Book

This book contains detailed information about specific AppleTalk troubleshooting problems. It also offers conceptual information about general types of problems and how to solve them.

While these pages provide some specific solutions, the book is primarily concerned with the methodologies, knowledge, and sensibilities you need to master for effective troubleshooting. We have good reason for taking this approach. The specific problems we face as troubleshooters change rapidly as new software and hardware continue to flow into the marketplace. At the same time, the principles of troubleshooting change more slowly.

We've tried to include both methodical approaches and impulsive approaches in this book because each kind is sometimes appropriate. In Chapter 1, be sure to read the section outlining the five kinds of troubleshooting algorithms, or modes. You'll begin to understand the situations in which each mode of troubleshooting is useful and also the situations in which a mode might be harmful.

Two ways to read the book

Some of you who buy this book will take it home, sit down in a comfortable chair, and read it in its entirety. Others will quickly scan through it and place it on a bookshelf until they're actually involved in a troubleshooting problem, then try to find a section that deals with the situation they find themselves in.

If you're one of those reading the book in a comfortable chair, reading it in the order of its chapters is probably as good a method as any. The concepts in Chapter 2 will be new to some of you and a review for others, but we hope that you'll find some interesting information there, regardless of your current level of expertise. Later on, some of the investigative techniques you'll read about, such as the Progressive Echo Test (PET) described in Chapter 6, are fairly easy to understand. You'll probably be able to perform a PET after reading through the description just once. Other techniques, such as the router configuration troubleshooting detailed in "Troubleshooting a Router Problem" in Chapter 9, are considerably more involved and will probably require frequent reference to the book while you're dealing with the problem.

For those of you still in your first couple of years in networking, we've included information that might be slightly tangential for a troubleshooting book but that might be difficult to come by any other way. A good example of this is the discussion of transmission theory in Chapter 2 and other references to it throughout the book. While the principles of transmission theory are not central to troubleshooting network problems, at times they can help you determine whether to investigate a particular theory.

For example, if you're troubleshooting a LocalTalk network that's only 25 feet long, transmission theory tells you that improper termination is unlikely to be the cause of a problem because the network is too short to be affected by the reflection of a wave with a wavelength on the order of 3,000 feet (120 times as long as the network). While there are excellent books on transmission theory, all of them are designed to be textbooks for graduate-level courses in electrical engineering. They're roughly half text and half differential equations. We've included the important concepts of transmission theory in this book and left the differential equations to more erudite books.

If you're not reading this book in a comfortable chair but are instead using the book "live,"curled up with some cables under someone's desk, skip ahead to Chapter 3. We hope that you'll become acquainted with the concept of the five modes detailed later in this chapter, but that can wait for now. Chapter 3 will help you get the ball rolling in asking the questions and exploring the boundaries of the situation.

During this phase of the job, you can also try a few randomly selected remedies. Even novice network managers have a few random methods that they try during this period, such as restarting the workstation, reinstalling software, or wiggling cable connectors. Once you understand the nature of the problem, start testing your hunches. When your hunches are exhausted, you need to switch from hunch mode to one of the three analytical modes. Chapter 3 helps you choose the right mode. Then you can jump to the appropriate chapter later in the book, depending on which mode you choose. Don't forget to look in the glossaries (packet and text definitions) or to the concepts in Chapter 2 if you have a question.

What if this book doesn't have the specific example I need?

This book is a guide to the process of troubleshooting, not a catalog of troubleshooting solutions. Many specific topics are missing from this book.

For example, we offer virtually no discussion about the problems of electronic mail. Despite this chosen omission, you can still use the concepts in this book to help you troubleshoot electronic mail problems or problems involving other kinds of network applications not directly addressed in this book. The concepts and diagnostic methods discussed in Chapter 3 apply to any AppleTalk troubleshooting problem.

Likewise, the principles discussed in the Setup and Layer Troubleshooting chapters are just as applicable to mail setups and functions. For process troubleshooting, you'll find many similarities between mail server and file server processes. There will still be session maintenance tasks, notification packets, and the transfer of file information. The information contained in the Setup and Layer chapters still applies to troubleshooting mail server problems, and the descriptions of file server processes have many parallels in e-mail operations.

If you focus on the principles of the book as you use it, you will probably find the information you need to apply to your specific problem.

The layout of the chapters

Chapter 1 is an introductory chapter that tells you how to orient yourself to network problem solving. It contains, among other things, an organization of troubleshooting methods into five modes. Understanding these modes—Random, Hunch, Set, Layer, and Process—will help you use the book more effectively because you will learn how to choose the most effective mode for your problem.

Chapter 2 is a reference chapter that outlines some of the large concepts of networking. We included this knowledge in a troubleshooting book because we've always found it helpful to understand the big picture before beginning to work on the smaller ones.

Chapter 3 describes how to begin the process of troubleshooting. Gaining a clear understanding of the problem is the first critical step toward solving it; the second critical step is to choose the right investigative approach to discover its cause. Both of these steps are covered in Chapter 3.

Setup troubleshooting, covered in Chapter 4, works best when you can't even initiate a connection or begin a session with another device. In setup troubleshooting, you attempt to verify the continuity between the components until you find the place where the continuity is disrupted.

Chapters 5 and 6 are concerned with layer-mode troubleshooting, where we test for the presence of all the network's functions using the OSI Seven-Layer Model as a reference. Layer-mode troubleshooting is best used when you're finding it difficult to understand the problem. A good rule of thumb is that if you can't explain the problem in one sentence, you should use the layer method. Chapter 5 is written primarily for people who don't have a detailed understanding of the functions of the seven layers. It provides the theoretical basis for Chapter 6, which concerns itself with actually performing the troubleshooting.

Chapter 7 is a set of process descriptions for several basic network operations. We choose a number of representative processes and detail what occurs in the devices and on the networks in each one.

Chapter 8 describes some useful troubleshooting tips and techniques.

Chapter 9 is a case study about solving a printing problem. It illustrates the flow of a troubleshooting session from beginning to end. In addition to providing information about printing to a LaserWriter, it can serve as a model to help you deal with many other kinds of troubleshooting situations.

The appendices contain reference material that includes a review of LocalTalk and Ethernet parameters. If you are new to protocol analysis, we've included a packet glossary as well as a standard text glossary.

We've also included a bibliography for people who want to continue their self-education. It lists only those books that we've found useful in our own AppleTalk troubleshooting.

You're on your way

Good luck in your troubleshooting. It's a unique learning experience. Problems almost always place you in new situations—and fixing a problem gives you a clear objective and a way of knowing whether you've attained your objective. On the one hand, we hope that you'll view troubleshooting as this kind of learning opportunity and come to enjoy it. But on the other hand, we hope that as you become better and better at it, you'll have to do it less often.

CHAPTER ONE

Troubleshooting Preview

Troubleshooting is, by definition, a process performed when there is trouble. In other words, something isn't behaving the way it should. The purpose of troubleshooting is to find the problem and correct it, restoring the network to its normal condition—trouble-free.

During the process of troubleshooting, you follow these basic steps:

1. Compare the network in its troubled state to the network in its normal state
2. Hypothesize about the cause of the difference
3. Apply corrective measures until the trouble is eliminated

Troubleshooting Requirement 1: The Normal Network

For the troubleshooting process to solve problems, certain requirements must be met. The first is that there must be a condition called "normal" during which the network operates in a trouble-free manner. The first step in troubleshooting is a comparison to this normal state. If there is no normal state, no useful comparison is possible. Further, the normal state should be something that you understand and document well, because the normal network isn't available for comparison during the troubleshooting process.

The better you understand the normal state, the more easily you can describe the trouble. That's why doctors check your blood pressure when you're healthy. If you're feeling sick, they can take your blood pressure and compare it to the normal values in their files. If doctors don't know what your normal blood pressure is, your "sick" blood pressure has less meaning to them and is less useful in their diagnosis. They can compare your "sick" blood pressure to the national average for people of your age, weight, height, and gender, but the reading is more meaningful when they can compare it to your own normal blood pressure.

You also need to understand the normal state so you know when you've corrected the problem and you're finished troubleshooting. To continue the medical analogy, doctors need to know when you're cured—when your blood pressure and other measurable aspects of your body have returned to their normal states. If your normal blood pressure is different from the national average, your doctor must know that in order to determine when to stop prescribing medicine.

Troubleshooting should be a rare event. If you find yourself troubleshooting frequently, the network has a design problem, not a troubleshooting problem. As a general rule, a network design should be reliable enough so that any major subsystem requires troubleshooting, on average, a *maximum* of once a month.

For example, if the network consists of LocalTalk networks routed to Ethernet, LocalTalk is a major subsystem and should require troubleshooting once a month or less. Ethernet and its routers should also require the same low level of troubleshooting. Think of once a month as the absolute limit for the frequency of troubleshooting. A good target frequency for troubleshooting should be twice a year.

If troubleshooting occurs more frequently than this, you need to change the network design so that reliability, rather than trouble, becomes the norm. Network design includes the selection, arrangement, and configuration of network components, administrative procedures, and user support. A change in any one of these might be the design change needed to bring the network to maximum reliability.

Troubleshooting Requirement 2: Tools

The second requirement for the troubleshooting process is that you must have tools to quantify, characterize, and adjust the network. Some of these tools are as simple as a wristwatch with a second hand. If you develop some benchmarks—for example, you might have a word processing document that you know takes four minutes to print when the network is healthy—they can provide a way to quantify the difference between normal and trouble. Other tools are more complex. You can use an oscilloscope to see the shape of the network signal as it travels along the wire, or a network analyzer to examine the data being transmitted at a higher level. Each tool tells its own story and is useful for different purposes and comparisons.

AppleTalk network management currently requires us to have a large collection of tools that measure, display, and adjust various aspects of the network. The all-in-one network tool hasn't yet been invented. Networks are complex and involve many interdependent components, both hardware and software. Network managers must have tools that deal with all these components.

We can't stress strongly enough the concept that before trouble occurs, you should use *all* the tools on the normal network to provide baseline data for comparison when the network is in trouble. Data that you gather on the normal network should be saved or printed for reference when the network breaks down. This approach might seem obvious, but people sometimes don't think about the fact that when the network is in trouble, the normal network isn't available, and thus preexisting normal data must be available for a meaningful comparison.

Troubleshooting Requirement 3: Understanding

Most of the current generation of tools gather and display information. A few of them suggest answers or implement solutions, but no tool provides reliable answers and solutions for all possible situations. That ability requires a level of intelligence not currently available in troubleshooting tools. Even a very smart tool, like the Norton disk utilities, runs into problems that it can't diagnose or fix.

This is true because the creator of a tool can't possibly conceive of all the possible problems that could occur in a system. While a tool can be useful for a variety of problems, it can't invent a new method for dealing with situations for which it wasn't explicitly programmed. All tools are made with the basic program: In this situation, perform this procedure. In unprogrammed situations, the tool is powerless.

Network managers have fewer intelligent tools than are available to other kinds of computer support personnel. While there are many good disk or virus analysis and repair tools, there are virtually none for network troubleshooting. This is because the structure of a hard disk and the problems it can encounter are more predictable than the possible structures and problems in a network. Networks have many different kinds of components from many different manufacturers interacting in complex ways.

Luckily, humans possess the ability to invent new diagnostic methods and repair procedures if they have enough understanding of network processes and interactions. Troubleshooting tools can gather and display the information needed to make the assessment, but it takes a human troubleshooter to make the diagnosis and perform the repair, particularly in completely new situations. New situations occur regularly in networking because of the pace with which our industry is developing and the pace with which we implement new products and technologies.

Understanding doesn't develop overnight; we all start out as beginners and gradually develop understanding and expertise through activities, reading, and training. More advanced troubleshooters, in addition to having more specific information stored in their brains, have more methods of diagnosis available to them. Because of the way it focuses your attention, the act of troubleshooting itself helps you expand your understanding of networks. Reading a book such as this one in your armchair is one way of developing your knowledge, but when it's midnight and you must get your database back on line before the users come back to work in six hours, you become a more focused reader.

This book is specifically designed for you to use during the troubleshooting process. While you should read this introductory chapter and the next chapter about troubleshooting in advance, you can use the rest of the book "live."

Troubleshooting Algorithms

Five algorithms, or modes, of troubleshooting are useful in different kinds of situations. All five modes proceed by the three steps outlined previously: *compare*, *hypothesize*, and *apply*. A natural progression in troubleshooting ability occurs as you develop knowledge and experience. The more experienced you become, the more modes you have at your disposal.

Arranged from elementary to advanced, the five modes are:

1. *Random.* Try something and see whether the trouble goes away.
2. *Hunch mode.* Try something that was previously useful in a similar situation.
3. *Setup-oriented.* Verify the continuity of the components and connections.
4. *Layer mode.* Verify that the network functions according to the seven-layer model.
5. *Process mode.* Compare the process that is occurring with the normal process.

Random mode troubleshooting

This is the most elementary method of troubleshooting. In it, you simply try a number of random methods to eliminate the trouble and proceed until one of the methods succeeds or the methods are exhausted. Methods might include restarting a system, reinstalling software, removing INITs, or reconfiguring the hardware or software. Although these activities can occur in one of the more advanced troubleshooting methods, in random troubleshooting they happen before a formal analysis is derived.

The goal of random troubleshooting is to make the problem go away, not to understand the problem or to prevent it from happening in the future. There's a fairly standard list of a dozen or so things that a network troubleshooter can do to make many problems go away.

Some network managers give their users a subset of these things to try and request that the users try all the items on the list before logging a trouble call. These procedures are usually mild cures, such as reselecting a device in the Chooser or wiggling connectors, but if network managers have properly constructed the list, having the users run through it can greatly reduce the volume of trouble calls.

Random troubleshooting is best used by new network managers or their assistants who are still unfamiliar with network basics. The method is also used by experienced troubleshooters when they're in a hurry or are otherwise unable or unwilling to perform a more thorough analysis.

Hunch mode troubleshooting

Hunch mode troubleshooting involves applying previous experience to the current situation. In other words, you recognize a similarity between this troubleshooting experience and a previous one. Sometimes the similarity is obvious and sometimes it's obscure, but either way you apply the corrective methods that have previously been effective.

If a beginning user complains about network trouble, experienced network managers usually begin with the assumption that the user isn't performing a procedure correctly. In such a situation, you should observe the user's procedure to determine whether the user is making a mistake. New users, of course, frequently make mistakes or have misconceptions about network operations. If it turns out that the user is performing the procedure correctly but the operation still doesn't complete successfully, the next step might be to check the Chooser configuration because many users misunderstand the Chooser.

If the Chooser configuration is correct, you might have a hunch to make sure that the network connector is firmly in place, that the Network Control Panel is properly configured, or that a terminator is properly placed.

Hunch mode troubleshooting is similar to random troubleshooting, except that in hunch mode troubleshooting, experience forms the basis for deciding which procedure to try. Like random troubleshooting, hunch mode troubleshooting places more emphasis on curing the problem than on understanding it, although the process of linking similar situations and cures can lead your intuition to suspect or discover the cause of the problem.

In the preceding example, if adjusting the Chooser configuration is frequently an effective cure you might begin to suspect a system software configuration problem. This thought might lead to a more thorough examination of the version, structure, and location of Chooser extensions, preference files, and other system documents, and subsequently to the discovery of the problem's cause.

Another possibility is that if the Chooser is frequently misunderstood, the naming scheme for network services might be inadequate or confusing. You might suggest more user training, reference documents, or a restructuring of the naming scheme. Administrative aspects of the network must always be analyzed with the same close scrutiny as hardware components, because they just as frequently cause trouble for the user.

A powerful tool in developing a hunch is to consider what changes have occurred, either in the network or in the user's system, between the last time the network worked and the time someone noticed the problem. These changes are often responsible for the failure of the network to function properly. Reverting to the state of the network before the change was made sometimes makes the problem go away.

It's sometimes difficult to obtain information about these changes because users generally don't remember when they made which structural changes to their systems. For example, by questioning the user, you might determine that a network operation worked perfectly until a week earlier. If you probe until you ask the right question, you might prompt the user to remember that co-workers added a screen saver that they got from "somewhere" a few days ago. Also, more than one change might have been implemented between the time a process worked and the time it stopped working, making it more difficult to figure out which change caused the problem.

Don't overuse this method of examining changes, however, because you can sometimes develop short-sighted, negative attitudes toward a product or procedure. Let's say that you buy a new network device and install it; afterwards, you can no longer print. When you remove the device, you can print again, so you conclude that the device is preventing you from printing. But the problem might be that the device wasn't installed properly, or that its presence caused other changes in the network. By simply removing the device and examining the problem no further, you might develop a negative attitude toward the device even though it wasn't the actual cause of the trouble.

Setup-oriented troubleshooting

This approach is a version of the finger-bone-connected-to-the-hand bone type of analysis. It's best used when a connection between two components can't be established at all. For example, your screen might be displaying a dialog box with "Looking for the Microsoft Mail Server" and "Give Up." Using the setup-oriented method, you would proceed by mentally breaking down the end-to-end connection into a series of connections between components whose continuity can be verified independently. By testing the overall connection as a series of smaller connections, you can more easily find the break in the continuity.

A prerequisite for setup-oriented troubleshooting is, of course, an understanding of the setup. A user or beginning network manager might understand the setup for printing, as shown in Figure 1-1:

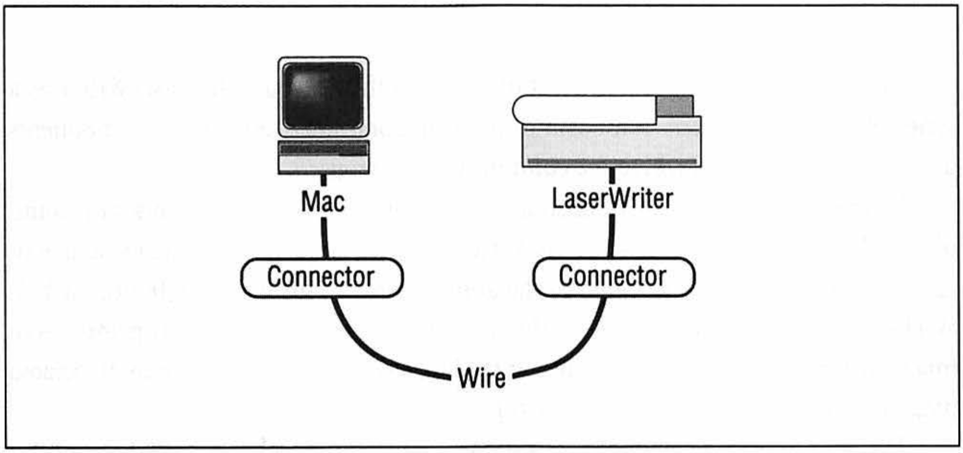


Figure 1-1. A simple way of viewing the Mac-LaserWriter setup.

A more experienced troubleshooter however, might conceptualize the printing setup as shown in Figure 1-2:

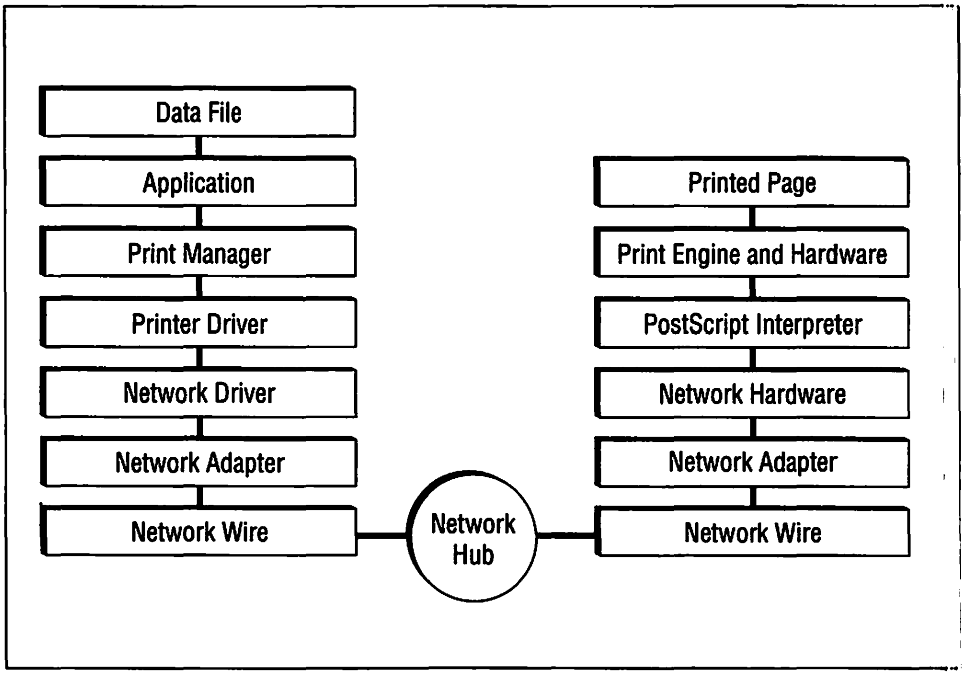


Figure 1-2. A more detailed view of the Mac-LaserWriter setup.

After conceptualizing the connection between the Mac and the LaserWriter as a series of components, you can begin to perform continuity tests on the components until you find the place where the continuity is interrupted.

You verify continuity by conducting tests on subgroups of components. For example, in WYSIWYG (What You See Is What You Get) applications, the existence of screen data verifies the link between the application and the data file. In non-WYSIWYG applications, you can verify this link by printing to a different printer—an ImageWriter, for example. You might test the link from PostScript interpreter to printed page by having the printer print a startup page.

This kind of testing and analysis proceeds until you either establish the continuity of the entire setup or discover the point of discontinuity so you can replace or repair the defective component.

Layer mode troubleshooting

In layer mode troubleshooting, we see the network as a collection of interdependent functions. This approach is called layer mode because it proceeds according to concepts established by the seven-layer model of the International Standards Organization (ISO), which served as a conceptual model for the AppleTalk Protocol family.

In the ISO seven-layer model, each of the seven layers has a specific set of network functions to carry out. By methodically checking the presence of each individual function, you can isolate the faulty aspect of the network. This method is best used in locating the source of a problem that isn't easily defined, occurs intermittently, or just seems "odd."

When a problem seems odd, it means that your instincts are leading you in the wrong direction. For example, you see services in the Chooser dropping in and out and you suspect that your router might be malfunctioning. You check the router and everything seems to be functioning properly. The problem, therefore, seems odd. You might have tried the hunch mode method and completely exhausted your hunches. Layer mode troubleshooting can help you identify the aspect of the network that is causing the problem.

For layer mode troubleshooting, you use tools to verify that the network's functions are being carried out. Each layer has a set of functions that can be verified. If the functions at that layer are not verified, you check the layer below. On the other hand, if the layer's functions check out, the analysis proceeds to the next higher layer.

Layer mode troubleshooting is similar to the child's game Chutes and Ladders. At each protocol layer there is a question, and there are tools to discover the answer to the question. If the answer is yes, you can take the ladder to the next layer. If the answer is no, you slide down the chute to the layer below.

For example, one function of the data link layer is to gain access to the network and transmit the information that was given to it by higher protocol layers. You can verify that particular function of the data link layer function by watching the traffic lights blink on the network hub when the device sends data. If the lights don't blink, the analysis drops to the physical layer. If the lights do blink at the appropriate times and other data link layer functions also check out, the analysis proceeds to the network layer.

The question at the data link layer is, "Are all the devices logically visible on the network?" In addition to the blinking lights on the network hub, Inter•Poll (among other tools) is a useful tool to answer this question. By looking at the device list in Inter•Poll, you can see whether all the network devices you're accustomed to seeing are present. Inter•Poll also provides an echo test that can quantify the level of reliability for the connection.

Process mode troubleshooting

In process mode troubleshooting, we view the network as a series of micro events. These micro events are captured by a protocol analyzer as a series of packets that contain the commands and information that a process such as printing needs to be successful.

While the setup-oriented method is best used for troubleshooting when the connection can't be made at all, process mode troubleshooting is best used when a network process behaves abnormally. For example, you might be able to communicate with the printer, but it seems very slow or it quits partway through the job. Process mode troubleshooting uses the information contained in the packets to examine the process at its smallest level of detail. The packet "trace" of the troubled process is compared to a previously captured trace of a normal process.

Process mode and layer mode troubleshooting are effective for many of the same situations. Process mode troubleshooting is almost always more effective and usually faster than layer mode troubleshooting, but it requires much more knowledge on your part.

By observing the process and comparing it with normal operation, you find out where the process bogs down or terminates. The best tool for carrying out process mode troubleshooting is a protocol analyzer, because it allows you to watch each tiny step in the process. Protocol analyzers are difficult to use at first. While they provide a great deal of detailed information, it's difficult for beginners to know exactly what information to look for. Protocol analyzers become easier to use as your experience and knowledge of the network develops.

For example, think of the process of printing to a LaserWriter as a series of events. The first four events are:

1. The Mac resolves the printer's name, stored in the LaserWriter file, to an internet socket address.
2. The Mac asks the printer for permission to print.
3. When permission is granted, the next step in the process verifies that the LaserWriter and the Mac are using the same version of the user dictionary for Macintosh-specific drawing commands. The Mac reinitializes the printer if the versions are different.
4. The Mac asks the LaserWriter which fonts it has loaded in memory and downloads any needed fonts not already in the LaserWriter's memory.

Of course, the LaserWriter printing process has many more events. We simplified them here to explain the method. Again, the process mode troubleshooting method checks the continuity between events. Each event is verified in sequence until the discontinuity is found.

Process mode troubleshooting is a very advanced technique. There are some good training companies that offer advanced classes in AppleTalk protocols. Whether or not you can attend these classes, *Inside AppleTalk* (Addison-Wesley, 1990) is invaluable when you're performing process mode troubleshooting. Also invaluable is a previously captured protocol analyzer trace of a similar process that completed satisfactorily. Like all troubleshooting tools, protocol analyzers should be used on the normal network before trouble arises. It's worthwhile to have a catalog of saved files for many common network processes, including printing, file transfer operations, electronic mail, or any other network services available on your network.

By comparing the file containing the normal process with the troubled process, you can spot the place where the two processes differ. This comparison gives you the clues you need to diagnose the trouble.

Choosing the best troubleshooting mode

The first two modes, random troubleshooting and hunch mode troubleshooting, are useful when you need a quick solution. While they don't usually yield useful information about the cause of the problem, they frequently provide at least a temporary cure. These methods are especially useful to beginning network managers or their assistants as well as to network users. Many experienced network managers use one of these two methods as a first line of attack when they arrive on the scene or as an attempt at troubleshooting over the phone. (If you try a series of solutions, however, you'd be well advised to document your changes to the network.) When all the hunches and random methods are exhausted, you must proceed to one of the more formal analyses found in the remaining three methods.

Setup-oriented troubleshooting, layer mode troubleshooting, and process mode troubleshooting are similar in that the network connection is conceptualized as a series of "things," and the continuity of those things is methodically examined in sequence. The difference between the three modes is that each uses different sets of things. Setup-oriented troubleshooting sees the network as a series of connections between components; layer mode troubleshooting examines the network as a series of connections between functions; process mode troubleshooting approaches the network as a series of connections between events.

You should choose an analysis method based on how the trouble seems to manifest itself. If the connection can't be made at all, use setup-oriented troubleshooting. If you can't figure out where the problem is occurring, use layer mode troubleshooting. If the process begins, but doesn't go well, use process mode troubleshooting.

Summary

Each of the five troubleshooting modes is useful in different circumstances. All five modes proceed through the same three steps of troubleshooting:

1. *Compare* the network in its troubled state to the normal network.
2. *Hypothesize* about the cause of the difference.
3. *Apply* corrective measures, one at a time, until the problem goes away.

The modes differ from each other mainly in the first step—the comparison process. In random troubleshooting, the difference is noted but not analyzed. At the other end of the spectrum, process mode troubleshooting, the comparison and analysis process is often highly detailed. Each troubleshooting method depends on remembering—and better yet, documenting—the characteristics of the normal network for comparison when the network is troubled. Since you'll use network troubleshooting and management tools for the comparison process, use them on the normal network and save or print the resulting data for a baseline.

CHAPTER TWO

Network Concepts for Troubleshooters

In the beginning of AppleTalk—1986 and before—the terminology was very simple. We used AppleTalk protocols over AppleTalk cables to communicate with AppleTalk devices. AppleTalk and Ethernet were incompatible networks. AppleTalk was referred to as a "peripheral sharing bus" and not really as a network; in fact, the only peripheral that AppleTalk shared back then was an Apple LaserWriter.

Unknown to most network managers at that time, Apple had enormous plans for AppleTalk as a network system. A network system includes the entire architecture of communication: hardware, software, protocols, and philosophy.

AppleTalk has grown through the years with the addition of many new protocols and services. AppleTalk Filing Protocol (AFP) was introduced, along with the underlying AppleTalk Transaction Protocol (ATP) and AppleTalk Session Protocol (ASP), and has steadily been improved. AppleTalk Data Stream Protocol (ADSP) has made more flexibility possible in the AppleTalk session layer, and major enhancements have been made to Routing Table Maintenance Protocol (RTMP) and Zone Information Protocol (ZIP). At this writing, we have new protocols for routing, asynchronous network access, and network management, with more on the horizon.

The AppleTalk Network System designers continue to dream large. It's very unusual for a company that makes only microcomputers to design and promote a network architecture at all, let alone one as ambitious as AppleTalk. Typically, complete network systems are designed only by manufacturers of larger, more powerful computers, like Digital's DECnet and IBM's Systems Network Architecture (SNA), or by international committees of network engineers. The success of AppleTalk, actually something of an anomaly, is a tribute to the vision of its founders and designers.

Establishing official protocols and incorporating those protocols into the formal, documented framework of the AppleTalk Network System has many benefits. The most important benefit for troubleshooters is that having the documentation for a protocol makes troubleshooting much easier because you can refer to a document that explains what all the bits and bytes mean and how all the information mechanisms work together.

A second important benefit of a standard protocol is that interoperability between products with similar functions is much more likely. For example, CE Software's QuickMail and Microsoft Mail are currently the most popular AppleTalk-based mail packages, but they can't communicate with each other directly because each product uses its own proprietary system for storing, transferring, and announcing mail messages. An AppleTalk mail protocol defined by Apple and followed by CE Software and Microsoft might someday allow network managers to construct integrated mail systems using both products. Individual mail users could then choose between the two mail systems, based on secondary features such as user interface or storage and retrieval capabilities.

AppleTalk has become a mainstream protocol because Apple designed a good framework, published AppleTalk's specifications, worked with developers, and consistently promoted both the technology and the philosophy of AppleTalk's user-centered design.

Along with these improvements, the terminology of AppleTalk has become much more complicated. The term "AppleTalk" now refers to the entire network system, not to any specific piece of that system. There is no longer any such thing as AppleTalk cable. What we used to call an AppleTalk network (sharing peripherals on cable and connectors made by Apple), we now refer to as LocalTalk. LocalTalk is one possible physical implementation of the AppleTalk Network System.

LocalTalk also refers to alternate cabling schemes such as Farallon's PhoneNET System, and components such as the Focus TurboStar and the Tribe LocalSwitch. LocalTalk is a physical network, and is only one of the physical means for hosting an AppleTalk Network System. LocalTalk is specified by a set of properties that we associate with the bottom two layers of the OSI Seven-Layer Model—that is, the physical and data link layers: twisted-pair wiring, 230.4 Kbps, collision avoidance, and so on.

AppleTalk networks can be constructed on the shoulders of other physical networks, such as Ethernet, Token Ring, FDDI, and Arcnet. Like LocalTalk, all these networks are defined by their physical layer characteristics. AppleTalk operates on these networks because there's an AppleTalk data link protocol for each of the physical networks: LocalTalk Link Access Protocol (LLAP) is the data link protocol for AppleTalk networks running on LocalTalk. Similarly, EtherTalk, TokenTalk, and

ArcTalk link access protocols are the data link protocols for AppleTalk running on Ethernet, Token-Ring, and ARCnet networks, respectively. Starting with the network layer, the protocols operate without regard for the choice of physical network.

The old terminology lingers on, however. You still hear people say things like "AppleTalk is slow," when what they really mean is, "LocalTalk is a slow network."

It's not just AppleTalk terminology that has changed. The terminology of all networking has gone through enormous changes. We have different definitions for such basic network components as bridges, routers, and gateways than we did in 1986. We also have network switches, which weren't considered network devices then, and products like brouters, which perform combined functions that were once performed separately.

AppleTalk is not without its problems, however. Because it was originally designed around the performance parameters of LocalTalk, some of the design choices made in AppleTalk's early years are now causing problems as AppleTalk networks become larger and faster and as they provide more services. Some of these problems were addressed as Apple "rolled out" and prodded the incorporation of AppleTalk Phase 2. AppleTalk protocols, even after Phase 2, continue to have shortcomings with routing, scalability, and timing, among other parameters. Thankfully, these problems are painful only in the largest networks, and Apple is very aware of the problems. For example, scalability problems in very large networks are currently being addressed by the introduction of the AppleTalk Update-Based Routing Protocol (AURP).

Computer networking is changing at an incredible pace. Overall, AppleTalk is keeping up and in some ways, setting that pace. We believe that AppleTalk will continue to play a major role in networking because of its intelligent design, its rich set of user services, and the loyalty and evangelism of the developers, network managers, and users who make up its community.

Layered Protocols

In human terms, a protocol is a set of rules and conventions that governs social interaction. To participate in a classroom setting, for example, all the students must know the proper way to ask a question. That's part of the classroom protocol.

Networking protocols have the same purpose: to provide an agreed-upon system for communicating. In a network protocol, a set of algorithms and data structures constitutes this system.

When you refer to the AppleTalk protocols, you're referring to a family of protocols—a protocol suite. Protocol families such as AppleTalk govern the process that starts with the user clicking the mouse or making a menu choice, and ends with 0s and 1s being signaled on a copper wire. Along the way, individual protocols within the protocol family handle certain portions of the process.

The overall process of network communication is complex, with many subprocesses and functions. A traditional way to cope with this complexity is to create a structural framework that organizes and categorizes the subprocesses and functions. The subprocesses become more understandable and easier to grasp when placed in their categories.

There are two conceptual frameworks for network protocols. One, created by a team of network engineers in the early 1970s under the auspices of the U.S. Department of Defense, is referred to as the DoD Model. The protocol suite referred to as TCP/IP (Transmission Control Protocol/Internet Protocol) is organized according to the DoD Model.

The other conceptual model, developed later by the International Standards Organization (ISO), is steadily becoming the preferred model for network protocols. The new model, called the OSI Seven-Layer Model (OSI stands for Open Systems Interconnect), forms the basis for organizing the AppleTalk protocol family, among others.

Think of the Seven-Layer Model as a "building code" for protocols. Just as your house was constructed according to your local building codes (probably with some small variances), AppleTalk was designed according to the Seven-Layer Model, with some minor variances.

The Seven-Layer Model, as its name implies, organizes the subprocesses into seven distinct categories, or layers. Each layer has a specific set of tasks to perform. Protocol designers, such as the people who designed AppleTalk, construct individual protocols for each of the layers that accomplish the layer's set of tasks. We think of individual protocols as software processes because of the way they work. An individual protocol receives data from another protocol process, performs a task on that data, and passes along the results to the next protocol in the chain.

In addition to defining rules of construction and operation, organizing a protocol family into a framework like the Seven-Layer Model also provides a way for new technology to be integrated as it is invented. To extend the functionality of AppleTalk to include, say, the transfer of live video, new protocols can be built to supplement the existing AppleTalk protocols and to be placed in the AppleTalk layer structure.

As long as the new protocols perform the same basic tasks (like representing data in a digitally encoded manner according to an agreed-upon format) and communicate with existent protocol processes using legitimate AppleTalk data structures, the incorporation of a new technology can happen quickly and smoothly.

Inside AppleTalk, a book written by the designers of AppleTalk and published by Addison-Wesley (second edition, 1990), is Apple's official specification of the AppleTalk protocols. It's supplemented periodically by documents as new protocols are invented and existing protocols are modified. These supplemental documents are available from APDA, the Apple Programmers and Developers Association.

Functions of the layers

One way to understand the layers is to think of them as the group of functions that are set in motion when a user gives a command to the computer. The seven layers of protocols interpret that command, transform it into computer code, package it with the information necessary to guide the command to its destination, and place the package on a copper wire in the form of a voltage wave. That voltage wave travels down the wire, is received somewhere else by another computer, and a set of symmetrical processes occurs in reverse.

The layers that perform these tasks in the Seven-Layer Model are as follows:

Application Allows the user to interact with the computer, giving it commands and receiving the results of those commands.

Presentation Takes the user's commands and data and represents them in a format recognizable to a computer.

Session Manages the computing resources used by two devices as they remain in relationship with each other. The session layer handles such activities as establishing the relationship, allocating resources (such as memory and CPU time), exchanging password and privilege information, ending the relationship, and deallocating the resources.

Transport Manages the delivery of data through the internet and copes with problems such as information getting lost along the way or changes in the structure of the internet.

Network Delivers data from one process to another across the internet.

Data Link Manages the tasks of packaging the information for transmission, gaining access to the network, and screening incoming information to make sure that it's readable. In AppleTalk, the data link layer also makes sure that each device has a unique address.

Physical Performs transmission under the guidance of the data link layer—creates outgoing signals and decodes incoming signals.

A useful way to remember the names and placement of the seven layers from top to bottom is the sentence "All People Seem To Need Data Processing." An alternative to this mnemonic device goes from the bottom up: "Pretty Darn Near The Silliest Possible Arrangement."

Network Addressing

Humans use addresses to communicate. For example, in verbal communications, we typically use some combination of names to distinguish one person from another. If three of your fellow workers are named Robert Stevens, Robert Hastings, and Robert Wallace, the people in your office will probably develop an addressing system that clearly distinguishes the three Roberts. For example, you could refer to them as Robert, Hastings, and Bob. The point is that the addressing system should clearly and unmistakably distinguish one entity from another.

Any entity can have numerous identifiers. For instance, people in our culture usually have a social security number, a driver's license number, and a telephone number. Each identifier signals something about a different aspect of the person's existence and is useful for a special kind of communication.

Computer networks need addressing because networks are populated with many devices. They need multiple identifiers because a device can simultaneously be a member of an Ethernet network, an AppleTalk node, a TCP/IP node, and a DECnet node.

When a device places a data packet on the network wire, that packet will be received electronically by many devices, even if it's intended for only a single device. Therefore, most networking systems require that the first few bytes of the packets contain the address of the device to which the packet is being sent. In that way, all other devices can quickly discover that the packet isn't directed to them. The devices can then ignore the rest of the data in the packet and spare their CPUs the task of processing useless data.

The device to which the packet is sent reads its own address in the front of the packet and continues reading and processing the packet. The packet usually contains the address of the device that sent it in case a reply is necessary.

As in human communication, a single device can have more than one kind of address at a time. It might have an address that refers to it as a hardware device or an address that refers to a particular software process it has operating inside itself. It might also respond to an address that refers to it as a member of a particular class of devices, and to another address that refers to all devices regardless of class, hardware, software, or individual address.

Group and individual addressing

Imagine yourself standing in a large lunchroom filled with people. You want to convey an important message to a number of individuals scattered throughout the room. To get their attention, you want to shout something that will make them look up. You can shout:

- "Hey everybody, listen to this!" and everyone will look up. After hearing the topic of your information, the people who don't need the information will quickly go back to what they were doing and ignore the rest of what you say.

- "Hey, members of Professor Olmstead's 10 o'clock economics class!" and only members of that group will listen to your announcement.

- "Hey, Mark Fletcher! Economics class is canceled today because Professor Olmstead is sick." Of course, you'd have to repeat this message for every class member.

These examples represent three different kinds of addressing used in network systems to communicate with a particular entity. The first kind ("Hey, everybody!") is called *broadcasting*. The second kind ("Hey, members of a particular group!") is called *multicasting*. The third kind ("Hey, Mark Fletcher!") is called *directed transmission*.

AppleTalk networking supports all three kinds of addressing. AppleTalk devices interpret and respond to packets sent to their specific address, to a group address that refers to all AppleTalk devices, and to a broadcast address that refers to all devices, regardless of the protocol they use.

On LocalTalk, every device is by definition an AppleTalk device. The multicast address of all AppleTalk devices is identical to the broadcast address of all devices. On other networks, such as Ethernet and Token-Ring, it's normal to have many different kinds of computers running a variety of protocols simultaneously. These general-purpose networks act as a delivery system for any protocol as long as the protocol information can be placed inside their standard packet format. In these environments, the multicast address has a different meaning from the broadcast address.

Because of the potential complexity, multiprotocol networks like Ethernet must make more distinctions in their addressing systems than do networks like LocalTalk. To make these distinctions, the network uses two addressing systems: one that refers to the network hardware, and one that refers to protocols running inside the device.

Hardware and protocol addresses

The distinction between hardware addresses and protocol addresses is simple: Hardware addresses refer to a device independently of what software or protocol it runs. Protocol addresses refer to a particular protocol component regardless of what physical network hardware it uses.

Hardware addresses are established when the network interface, such as an Ethernet adapter, is manufactured. They stay with a device regardless of which network it's placed on or in which company the device is installed.

Protocol addresses are established when a protocol suite is initialized, typically at startup, and are assigned either by the network administrator or (in the case of AppleTalk) dynamically by a software process inside the device. Most physical networks, including Ethernet and Token Ring, require that the hardware address of the intended receiver be placed in the first few bytes of the packet. The sender places an indicator of the packet's protocol type and the protocol address deeper in the packet. It's the hardware address that determines whether the device processes the packet.

After the packet is properly received by the hardware, the protocol type determines which protocol suite the packet is intended for; the protocol address determines whether the suite continues, in fact, to process the packet.

For example, any Macintosh placed on an Ethernet network is assigned the Ethernet address that refers to the Ethernet card that provides the connection. That Ethernet address is assigned by the manufacturer when the card is made. You can tell which manufacturer made any Ethernet device by looking at the first three bytes of the Ethernet address. For example, any Ethernet address that begins with 08:00:07 belongs to a device manufactured by Apple; an address beginning with 00:00:89 belongs to a device made by Cayman Systems. Appendix C contains a more complete list of Ethernet manufacturer designations.

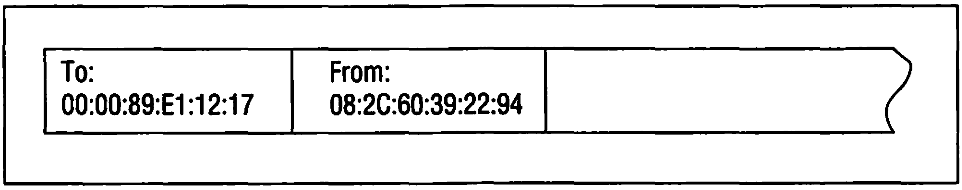


Figure 2-1. Ethernet and other kinds of multiprotocol networks require that all packets use Ethernet hardware addresses regardless of which kind of protocol information they might carry.

An AppleTalk address is a protocol address. An AppleTalk address, identifying a particular software address on a particular node on a particular network, is established independently from the hardware address. When a device activates the AppleTalk protocols, its AppleTalk address is established through a dynamic process that we'll describe in "Device Startup," found in Chapter 7.

When a device uses AppleTalk to send information on a network such as Ethernet, which uses hardware addresses, it encodes the AppleTalk address of the destination software process inside the body of the packet. When the hardware adapter receives the message, the device can determine which software process needs the information by reading the encoded protocol address inside.

Having an independent addressing system gives AppleTalk the ability to operate on many different kinds of physical networks. The only requirement is that each network in the electronic link between sender and receiver must allow AppleTalk information inside its packets.

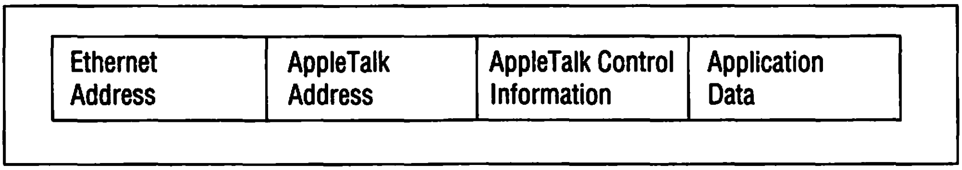
AppleTalk communicates well on a variety of physical hardware and network types: on LocalTalk, Arcnet, Ethernet, and IBM Token Ring networks, over synchronous and asynchronous modems, through X.25 links and over Fiber Data Distributed Interface (FDDI) links, as well as over the Integrated Services Digital Network (ISDN). In each case, AppleTalk takes the information from the sending process and places it in an appropriately formatted data frame according to the rules of the underlying physical network. The most common example is AppleTalk protocols on Ethernet.

AppleTalk addresses and Ethernet addresses

When a packet is sent to a software process in an AppleTalk device that is on Ethernet, the packet must be sent to the Ethernet address of the Ethernet network adapter. The AppleTalk address of the software processes that are sending and receiving are embedded inside the packet, but the packet must first be delivered to the Ethernet adapter. AppleTalk packets sent over Ethernet (EtherTalk) have five parts:

1. The Ethernet addresses of the source and destination devices
2. A protocol type field, which indicates that the packet is an AppleTalk packet
3. The AppleTalk addresses of the source and destination devices
4. Protocol control information that tells the software process how to interpret the data
5. The data itself

The first address in the packet is the Ethernet hardware address, which cues the Ethernet card to capture and decode the packet. The Ethernet address is stripped away, the protocol type is examined, accepted, stripped, and the rest of the packet is handed over to the AppleTalk protocol suite. In the remaining portion of the packet, the AppleTalk protocol address identifies the recipient software process that uses the control information to determine how to process the packet. Following the control information is the actual data—the "payload" of the packet.



Ethernet Address	AppleTalk Address	AppleTalk Control Information	Application Data

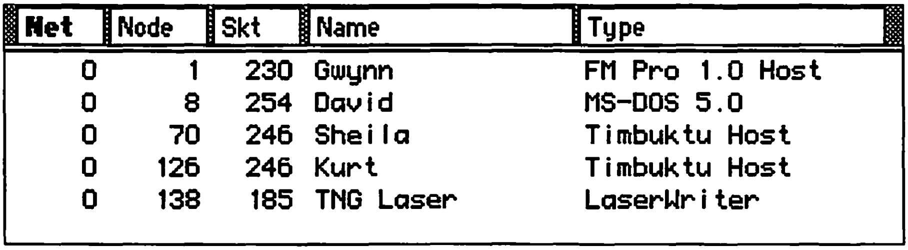
Figure 2-2. When an application uses AppleTalk protocols to carry data on Ethernet, the packets must carry Ethernet addressing information for delivery to the right Ethernet adapter, AppleTalk addressing information so the data can be sent to the right application, AppleTalk control information so the process knows how to interpret the data, and, finally, the data itself.

AppleTalk Address Resolution Protocol (AARP) handles the address mapping between AppleTalk addresses and the hardware addressing system of the network. AARP creates and manages a table of addresses, the Address Management Table (AMT). The AMT has two columns per entry, one for the AppleTalk addresses of particular nodes and one for the corresponding hardware addresses. To allow the other AppleTalk protocols to communicate in terms of AppleTalk addresses, AARP's responsibility is to supply the necessary hardware addresses.

Because LocalTalk can carry only one protocol (AppleTalk), it differs from Ethernet and TokenTalk in that it doesn't use hardware addresses and has no need of an AARP process. Packets on LocalTalk are delivered to the AppleTalk node address; there is no hardware address.

Address numbering systems

Most networks and protocol systems use numerical addresses. In general, these numbering systems are separated into fields that describe different portions of the address. We've already mentioned that the first three numbers (bytes) of an Ethernet address indicate the manufacturer of the card. AppleTalk addresses identify individual software processes and consist of three fields: a network number, a node number, and a software process number (also known as the socket number). The combination of network number and node number specifies a particular AppleTalk device, such as a Macintosh, LaserWriter, or router. The socket number specifies a particular process within that device.



Net	Node	Skt	Name	Type
0	1	230	Gwynn	FM Pro 1.0 Host
0	8	254	David	MS-DOS 5.0
0	70	246	Sheila	Timbuktu Host
0	126	246	Kurt	Timbuktu Host
0	138	185	TNG Laser	LaserWriter

Figure 2-3. AppleTalk addresses identify a software process in terms of network number, node number, and socket number.

Address fields

In most numerical addressing systems, a specific number of bits is designated to hold each field of the address. The numerical value of the address in each field has a potential range of zero to the highest number that can be represented by the number of bits designated for that field.

For example, an address element that is designated by 8 bits has a range of 256 potential addresses, and the range of that address element is typically 0–255. AppleTalk node numbers use this system. They are designated by 8 bits, so all nodes take node numbers between 0 and 255.

The number of bits used by the entire address of the numbering system determines how many entities can be specified. Because AppleTalk uses 32 bits to specify a software process, the total (theoretical) number of software processes that can simultaneously use AppleTalk to communicate in a single internet is 2^{32}, or 4.32 billion addresses. Ethernet addresses use 48 bits, allowing 2^{48} addresses. This number—281 trillion—is roughly equal to the number of pennies in the federal deficit (in summer, 1991). Since the first 24 bits designate the manufacturer and the last 24 bits designate the device, the Ethernet addressing system can support millions of manufacturers who can each build millions of devices.

Broadcast and multicast addresses

The highest number in the range of an address field usually has the meaning "all entities of this type." Node number 255, the highest number in the node address range, refers to all nodes. Sending a packet with a destination network number of 0 and a node number of 255 translates to sending the packet to all nodes on this network. This is the broadcast address and it signals all nodes to interpret and process the packet.

Likewise, an Ethernet broadcast address of FF:FF:FF:FF:FF:FF means to send to all Ethernet adapters. With some entities, this designation convention has little meaning. For example, the Ethernet address of 00:00:00:00:00:00 means "this Ethernet adapter," so you should never see an Ethernet packet with this address as the Ethernet destination.

AppleTalk also has an Ethernet multicast address that refers to all devices that use AppleTalk protocols. But this address can be used only for AppleTalk Phase 2 devices. When an AppleTalk device sends this on Ethernet, it sets the Ethernet address to 09:00:07:FF:FF:FF. Inside the packet, the AppleTalk network and node address are 0.255. The AppleTalk socket address is set according to the nature of the communication. For example, socket 2 on every AppleTalk device is designated as the Names Information Socket. It's the address of a software process that manages name tables using the Name Binding Protocol (NBP). When you search for AppleShare servers with the Chooser, you must send an NBP packet to every AppleTalk device's Name Information Socket and ask whether it has a name of the type "AFPServer" registered. The Ethernet address of this packet is the Ethernet multicast address 09:00:07:FF:FF:FF, and the AppleTalk address inside the packet is 0.255.2.

AppleTalk Addressing

Every device on an AppleTalk network must have a unique address—one that distinguishes that node from every other node in the internet. This unique address is comparable to how the U.S. Post Office uniquely identifies every building in the United States through the combination of the building's ZIP code and street address. On an AppleTalk network, each network has a unique identifier, and within each network, each node has a unique identifier. The combination of network address and node address that uniquely identifies a device is referred to as the Internet Node Address.

Inside an AppleTalk device there might be several processes that need to communicate over the network, just as in a single building there might be many people who need to receive letters and packages. People in the building are distinguished from each other by the use of a third identifier: their names. Software applications inside a device, such as FileMaker Pro or a QuickMail process, distinguish themselves by opening a unique "mailbox number" called the socket address.

The Internet Node Address combined with an application's socket address is called the Internet Socket Address of the application. When data is sent to an application, it goes to the application's Internet Socket Address.

Network address

Network managers assign network numbers when they configure AppleTalk routers (such as a Shiva FastPath, Cayman GatorBox, or Farallon Liaison). Other nodes learn their network address from routers at startup time. Each network must have a network address that is different from every other network. The network address can be a single number, or on AppleTalk Phase 2 networks, a range of numbers, such as 1–10. However, all the numbers within the range must be assigned to only that network.

Figure 2-4 shows two routers creating three networks. Each network has a unique network address defined by the router. The network in the middle of the figure is an example of an extended network.

Extended networks

AppleTalk Phase 2 introduced the concept of the "extended network." In AppleTalk Phase 1, a network could have only a single network number. Because of the limitations of node addressing (described later), one network number would allow only 254 devices to attach to the network. Because extended networks can have a range of network numbers rather than a single network number, they can contain more devices. Each network number in the range can contain a full set of node addresses (253 in Phase 2). A network identified with a number range of 1–10, like the one shown above, can contain 2530 devices.

AppleTalk requires that the network numbers in the range be continuous. Ranges such as 1–10 or 24000–24500 are valid, but designating network numbers of 3, 37, and 104 to a single network is not valid. LocalTalk networks can only be assigned a single network number and are by definition nonextended networks.

Network numbers

Network addresses use 16 bits. This setup allows network addresses of 0–65,535. Network managers can assign only network numbers between 1 and 65,279, however. Network "0" is reserved to describe a Phase 1 network with no routers. The network address of 65,535 is not used; it would refer to all networks and routers won't pass that. Network numbers between 65,280 and 65,534 are reserved for a startup range. The startup range is used during the node address acquisition process (explained in the following section).

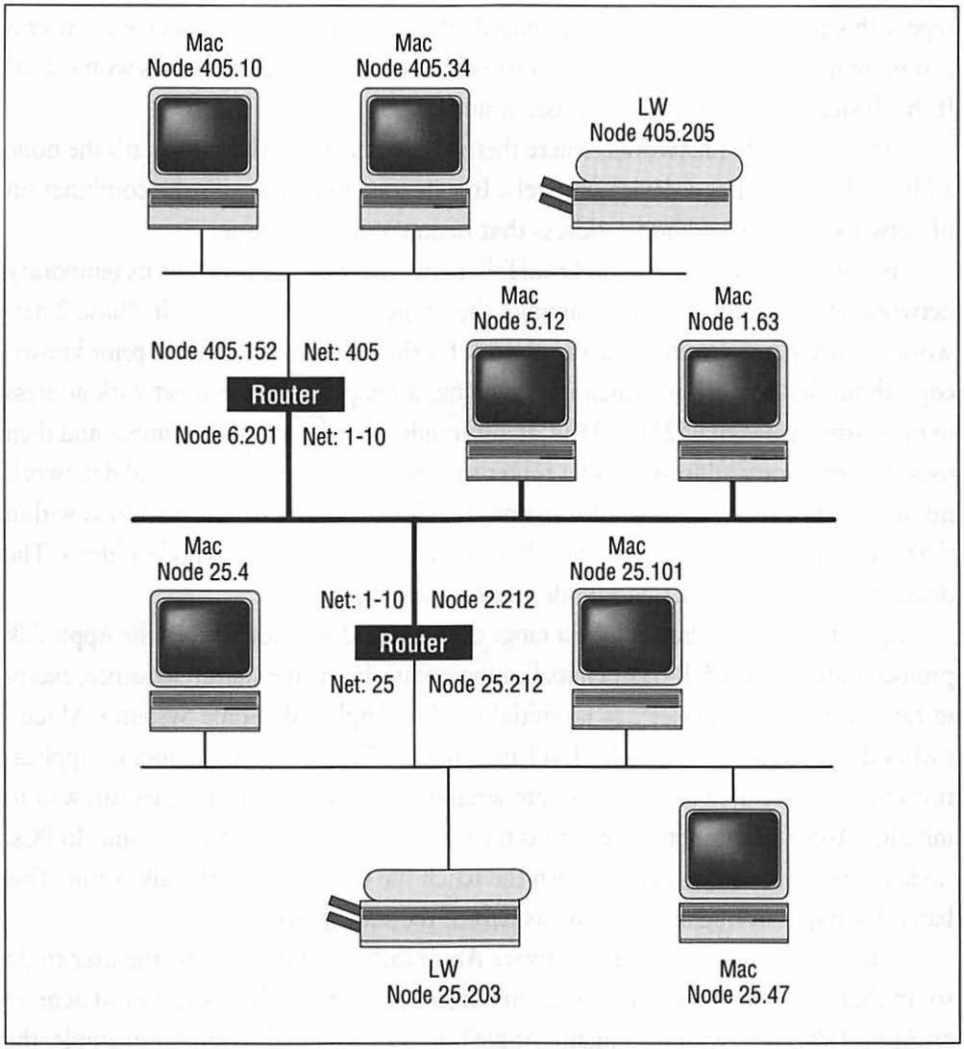


Figure 2-4. Every AppleTalk device must have a unique Internet Node Address. Routers define the valid network numbers for each network. Devices such as Macintosh workstations discover their network address from the router. All nodes must establish their own node address during startup.

Node address

When they start up, AppleTalk devices pick a node address on their own, checking first to make sure that no other device has their chosen address. If they find another device already using the address they selected, they repeat the process by randomly choosing a new address and checking the new address for uniqueness. A device must repeat this process until it finds an unused address, but some devices give up after a certain number of tries. In such (very rare) cases, the AppleTalk processes won't start. If the device is a workstation, the user is notified.

On nonextended networks, where there's only one network number, it's the node address that identifies a device uniquely. In extended networks, it's the combination of network address and node address that makes a device unique.

In AppleTalk Phase 1 and on LocalTalk networks, a device uses 0 as its temporary network address until the router supplies the proper network number. In Phase 2 networks, when a device comes onto a network for the first time and has no prior knowledge about the network to which it's attaching, it temporarily uses a network address in the startup range of 65280–65534. It also randomly selects a node number and then uses this temporary address to ask a router to supply it with the range of valid network numbers. After receiving this information, the device takes a network address within the range supplied to it by the router, along with a randomly selected node address. The device then checks this network/node address for uniqueness.

AppleTalk node addresses have a range of 0–255 and are chosen when the AppleTalk protocols are initialized. Protocol initialization occurs during the startup sequence, except in rare cases when a device has no initial need of AppleTalk. Some System 6 Macintoshes don't have the Responder INIT or other INITs that register names or applications to check the network for duplicate serial numbers; these Macintoshes can wait to initialize AppleTalk until the user opens the Chooser or prints for the first time. In PCs, node address acquisition occurs when the batch file that loads AppleTalk is run. This batch file might or might not be run as part of the startup process.

Sometimes a device might deactivate AppleTalk or be directed by the user to do so. In that case, when the machine again becomes an AppleTalk node, it must acquire an AppleTalk address all over again. AppleTalk is deactivated when, for example, the node is switched to a different network in the Network Control Panel or when AppleTalk is turned off in the Chooser.

Deactivation and reinitialization can also occur when certain software processes use the AppleTalk network adapter for a non-AppleTalk purpose and temporarily disable AppleTalk. When this software process terminates, it might reinitialize AppleTalk and repeat the node address acquisition process. Applications that do this are usually network management tools, such as TrafficWatch and LocalPeek.

Instead of using a random number as an initial guess as a node number, Macs and many other network devices recover the node address that they used during their most recent connection from nonvolatile memory (PRAM, for example) and use that previous address as the first address to check. This stored address is known as a "hint." Other devices have a "favorite number" that they use for the first address bid. In either case, if the first address is already taken, the device uses randomly generated numbers for its next attempts.

The mechanics of the node acquisition process are specific to the type of network and are described explicitly in Chapter 7 for LocalTalk and Ethernet networks.

Node address numbers

Node addresses use 8 bits, allowing node addresses from 0–255. Node address 0 is not used, and node address 255 is the broadcast address; packets sent with a destination address of 255 must be read by all devices.

In LocalTalk, all numbers in the 1–254 range can be used for node numbers, but there's a distinction between personal computers and shared devices. The lower half of this range, 1–127, is typically used by personal computers; the upper range is typically used by shared devices, such as printers, routers, file servers, and shared modems. The rule that divides the address range in half has never been very firm, and was intended to avoid a potential problem, described in the next section.

Duplicate node addresses

The node address acquisition process is highly successful in avoiding duplicate addresses, but it can be defeated if users turns on their computers before plugging them into the network. This often happens when users bring their computers back from a trip or bring them back from using them at home for the weekend. If the node acquisition process occurs while the network isn't connected, the first address attempted will be successful. Because there are no devices connected, no other device is able to indicate that it already has the address being checked. When a user discovers that he or she forgot to plug a computer into the network and do so after the computer is running, there's a chance that the addresses the computer is using will duplicate another device's address.

If two devices have the same address, communication is difficult for both computers, although some sporadic communication usually gets through. AppleTalk currently has no way of automatically detecting and correcting this situation. You can locate the problem by seeing duplicate listings for a single address in Inter•Poll, but you can remedy the problem only by turning AppleTalk off in one of the devices and then reactivating the AppleTalk protocols.

A network manager can also cause duplicate addresses by joining two networks with a bridge or a repeater, or by removing a router and joining two existing networks into a single network without turning the devices off. This might sound farfetched, but it happens more often than you'd expect.

When the node address range is divided in half—lower range for personal computers, upper range for servers—users who start their computers and then later connect them to the network don't cause problems with one of the shared devices, only with another person's workstation. On Ethernet and Token-Ring networks, this precaution is less necessary because in most cases, the AppleTalk protocols won't initialize unless the device is connected to a network. In these networks all devices, personal or shared, choose from the entire range of node addresses.

In Phase 2 networks, the node address of 254 is also a reserved address, although Apple hasn't announced its purpose. All devices can choose node addresses from the range of 1–253.

Node addresses in extended networks

In extended networks, because there's a range of network numbers to choose from, the combination of network and node number must be tested for uniqueness. In Figure 2-4, the extended network in the middle of the figures has a range of 1–10, so devices choose a network address in that range and select a random node address. The device then checks to see whether the network/node number combination is unique.

Socket addresses

After the device has secured its node address, individual software processes within the device can open socket addresses as "mailbox numbers" for themselves. A node can have many sockets open simultaneously, each socket a "mailbox" for a particular software process. All data sent through AppleTalk is sent to socket addresses. For

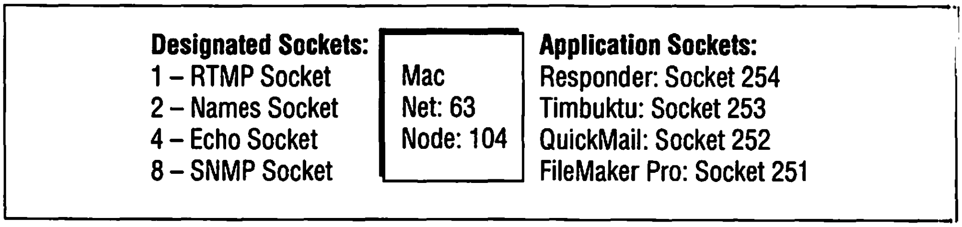


Figure 2-5. Each process in an AppleTalk device that needs to communicate over the network establishes a socket address for itself.

example, when devices log in to a server, the device and the server inform each other which sockets to use for further communication.

Socket numbers

AppleTalk sockets use 8 bits and have a range of 0–255. Socket number 0 is not used, and neither is socket number 255. Sockets in the range of 1–127 are referred to as statically assigned, meaning that a software process opens a particular one of these sockets as its mailbox because that socket has a specially designated purpose.

Typically, commercial applications like FileMaker Pro and Microsoft Mail use the dynamically assigned sockets in the range of 128–254. In this case, the application asks the AppleTalk protocol process for a socket. The protocol process then chooses one of the unused sockets in this range and tells the application the socket's number.

In the statically assigned range of 1–127, Apple has reserved numbers 1–63 and has designated five of these sockets for special purposes. The other numbers are unavailable for use and continue to be reserved for some future use. The statically assigned sockets in the range of 64–128 can be used, but only experimentally—not for commercially released products. Although Apple has asked programmers to avoid using these sockets, some programmers have chosen to use them anyway. An example is Farallon, whose product Timbuktu sends broadcast packets (all nodes) to socket 64. Devices belonging to Timbuktu users process this packet; other devices reject the packet.

The four sockets that Apple has designated are the mailboxes for five AppleTalk Protocol functions. Any information about routing is sent to socket 1, name information requests are sent to socket 2, echo packets go to socket 4, zone information goes to socket 6, and SNMP (Simple Network Management Protocol) information goes to socket 8.

Network Relationships

In troubleshooting, conceptualizing the relationship between two computers is the basis for forming hunches and hypotheses. The critical aspects of computer relationships that you need to understand are:

1. How the course of the relationship is controlled
2. How the elements of computing—data storage, application code storage, processing, and display—are distributed
3. The factors that affect the flow of the relationship

4. The kinds of conditions that typically cause slowness and loss of connections
5. The effect of network errors on the relationship
6. How the relationship can be abandoned and what happens as a result

There is great diversity in these factors among the commercial products available.

Control of the relationship: Client-server and peer-to-peer

Several models describe the relationship between two computers. These models are roughly divided into two categories: *client-server* and *peer-to-peer.* The main distinction between the two is in how the course of the relationship is controlled. However, the distinctions between the categories of client-server and peer-to-peer are not always crystal clear. Sometimes there are elements of both in a relationship, and the choice of category is somewhat arbitrary.

In the *client-server* model one device—the client—controls the course of the relationship. The client initiates the relationship, asks all the questions, gives all the commands, and ends the relationship when it's finished. The server in the client-server model only reacts to the input that it receives from the client. Examples of client-server relationships include:

- The Mac-LaserWriter connection
- The relationship between a mail or file server and a client workstation
- A FileMaker database being used over the network

In a *peer-to-peer* relationship, the rights, responsibilities, and abilities to control the course of the relationship are more symmetrically distributed. Either device can initiate or break connections at will, give commands, and ask questions. Examples of peer-to-peer relationships include:

- Data Club's virtual volume consisting of disk space from many computers
- A Publish and Subscribe link between two users
- Two applications in communication using Apple Events

Networks typically include numerous examples of both kinds of relationships. Client-server relationships are more common, but peer-to-peer relationships seem to be gaining ground as the cost of computing power drops and the traditional way of categorizing computers—into mainframes, minis, workstations, and micros—continues to break down.

Configuration of computing elements

In the client-server category, there are five models of relationships: *terminal emulation, diskless workstation, file server, compute server,* and *process server.* Each model is distinguished not only by the way it distributes the computing elements, but also by the information that gets passed between client and server

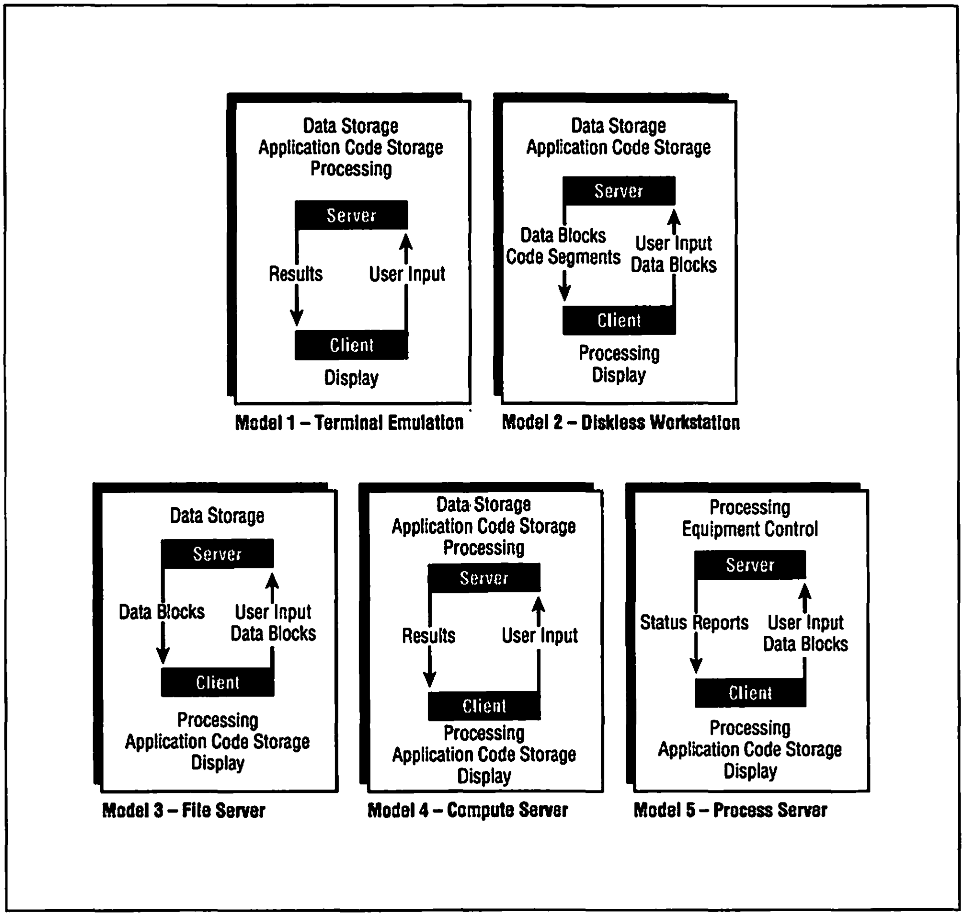


Figure 2-6. Models of client-server interaction.

Peer-to-peer relationships are more symmetrical and are characterized by simultaneous flow of control, data, and replies in both directions between the two peers.

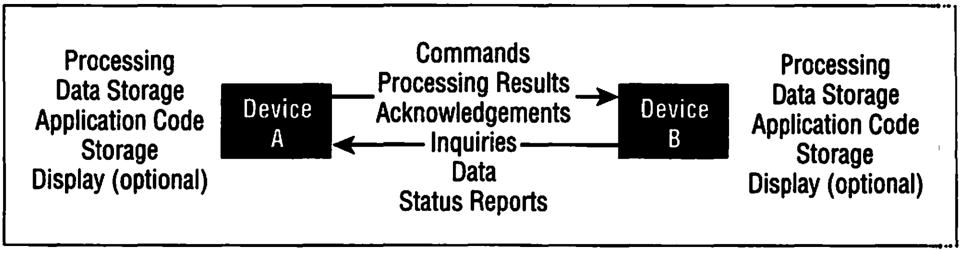


Figure 2-7. Peer-to-peer interaction.

Of the two types of interaction, client-server relationships are somewhat easier to troubleshoot because you can usually think of the devices one at a time. When you think in terms of the client, the server can usually be considered a "black box," and vice versa. Also, commands and replies, questions and answers, and so forth, usually occur sequentially or nearly sequentially.

In peer-to-peer relationships, however, because actions can be initiated by any device at any time, you must keep both devices in mind simultaneously. Troubleshooting peer-to-peer is also more difficult because the synchronization of events is not always sequential.

Flow limitations: Finding the bottleneck

A useful way to consider the limitation on work flow is to think in terms of a bottleneck. For example, the bottleneck in printing to an eight-page-per-minute LaserWriter over LocalTalk can be one of two things: the printer's ability to process the PostScript code it receives, or the printer's ability to mechanically print the page. Which one of these is the bottleneck depends on the complexity of the PostScript code.

The bottleneck concept is simplistic; flow relationships among multiple components are much more complex than this. But the image is useful when you're asking yourself, "What factors make this process go faster or slower?" If the user complains about the slowness of a process, you must determine whether, given the current network configuration, the process is already proceeding as quickly as possible, or something is preventing the equipment from working optimally.

The first case exemplifies a network management problem. If the network is working as fast as it can with the current setup, you need to make some decisions: will the cost of upgrading the equipment be offset by the projected productivity gains from a faster process? Identifying the bottleneck helps you determine which components need to be upgraded.

The second situation is a troubleshooting problem. Obviously, you must conduct an investigation to find the cause of the slowness. Identifying the bottleneck helps you locate the source of slowness. For example, you can analyze the LaserWriter bottleneck identified above by asking such questions as:

- Is the PostScript complexity necessary?
- Is the LaserWriter receiving information as soon as it asks for more?
- Are fonts being downloaded every time they're used, or only at the beginning of the print job?
- Could ROM-resident fonts be used in place of the fonts in the documents?

By looking at the factors contributing to the bottleneck and the processes immediately surrounding the bottleneck, you might be able to suggest ways to alleviate the user's complaint. You might, for instance, notice complicated graphic objects in the document that might be replaced with objects that the LaserWriter can draw more quickly.

Some graphics applications provide a wide range of fill patterns for objects. Some of these take longer than others for the LaserWriter to draw. You might then experiment a little to find a faster pattern fill type that produces a similar image but reduces printing time by 75 percent. The user can then decide whether to keep the gradient fill and tolerate the slowness or change the object.

The bottleneck isn't always easy to identify. It could be any of the following:

- The speed of the processor/the amount of processing required
- The speed of the disk/the amount of disk processing required
- The speed of the computer's bus (SCSI, NuBus, and so on)
- The speed of the network/network hardware
- RAM speed/size limitations

Speed factors

In addition to the bottleneck, speed can be further affected by the following factors:

- Other processes in one of the devices competing for resources
- The overhead processing requirements of the server process
- The efficiency inherent in the structure of the relationship; for example, the amount of data that must be exchanged to accomplish the process, the sheer number of transactions that must be completed, the sequential nature of the interaction, and the efficiency of memory and cache management
- Disk or memory fragmentation
- The number of routers/networks between devices

In multiuser situations, speed might also be affected by:

- The overhead of session management for simultaneous clients
- The sheer number of simultaneous users
- The workload caused by requests from other users
- The way in which the server processes concurrent requests; for example, sequentially: complete request A, then complete B, and so on, or "round-robin": do a little of A, then a little of B, and so on

Effect of errors

The following errors can also slow things down:

- Lost or garbled packets caused by bad wiring
- Lost connections with other users, forcing the server to look for them
- A stuck process on a network device, slowing other processes
- Disk errors: bad blocks, corrupted directories, invalid pointers, and so on

The effect of a lost or garbled packet varies, depending on the type of packet. Almost all packets have an associated "retry timer." After a device sends a packet, it usually expects a packet in return: an acknowledgment, data, a command, or an answer to a query, for example. The duration of a packet's retry timer is the amount of time the device lets pass before investigating whether its partner-device has received the packet. The duration of the retry timer is dependent on many things but is usually slightly longer than the time for a typical response. Retry timers can be as short as a few milliseconds and as long as 30 seconds. If a packet with a long retry timer gets lost or garbled, there can be a greater effect on the process speed than if a packet with a short retry timer gets lost.

Loss of connection

Connections get lost and dropped when one of the devices in a relationship discovers that the other device is no longer available. There can be many reasons why a device makes this assessment, but in each case, the symptom that the device perceives is that it is no longer receiving packets from its partner. Possible causes can include:

- The other device crashed
- The other device discontinued offering the service
- The network link was broken in some way, such as the cabling broke or was disconnected, an intermediate network device (for example, a router or hub) crashed an intermediate network device was removed from service, or network reliability deteriorated to the point that the percentage of successful transmission wasn't high enough to support the connection

Many network relationships are "bursty" in nature, characterized by periods of relatively high usage alternating with periods of relative idleness. During the idle periods, the devices typically notify each other that the connection is still healthy by sending what are known as "tickle packets."

A tickle packet usually has a "tickle timer" associated with it; that is, the amount of time that passes before a tickle is sent or expected to be received. Tickle timers range from a few seconds to as much as five minutes.

Along with defining the tickle timer, the programmer also places a limit on the number of tickles that can be missed before a connection problem is suspected. This number is typically in the 1–5 range. When a problem in the connection is suspected, the device usually initiates a search for its partner that lasts until the connection timer expires. The connection timer is the maximum amount of time that can pass before the connections are considered lost; the search for the partner is terminated and the device initiates the process of closing down the connection.

When closing down a connection, the device makes every effort to recover and return to a normal state. It deallocates any resources, such as memory and sockets, that were being consumed by the connection, closes any open files, and terminates any processes used. The difficulty of closing the connection depends on the state of the relationship at the time of loss, and can be easy, difficult, or impossible.

In some cases the device cannot recover at all; it requires restarting, or data is corrupted. This is most typical of connections that are lost in the middle of a write operation, such as saving a file to a remote device or modifying the record in a multiuser database. The worst case occurs when one device crashes and sets off a domino effect, freezing or crashing other devices around the network in its wake.

Here's an example of an extreme, yet possible, domino scenario:

- One device crashes
- Devices in relationship with it initiate searches
- The searches consume major amounts of bandwidth
- Because the network is so busy, other connections slow down
- Some of the connections reach their tickle timer and begin searches
- The network "freezes" while devices flood the network searching for partners
- Connections are dropped, searches are terminated, and some devices crash because they can't recover from the loss of connection
- The high level of traffic subsides and the network returns to normal
- Devices are restarted and connections are reinitiated

This case is admittedly extreme, but it happens occasionally, particularly in networks with reliability problems. Why is poor reliability a factor? Because lost packets make the expiration of the tickle timer more likely, which makes searches and lost connections occur more frequently. Also, the frequent need to resend packets in an unreliable network inflates the utilization, making it easier for the devices to max out the network's bandwidth.

An Example of a Client-Server Relationship

A Mac printing to a LaserWriter represents a client-server relationship in which the Mac is always the client. The relationship is asymmetrical because the Mac client and the LaserWriter have very different roles, rights, and responsibilities. The Laser-

Writer is a process server; it accepts commands, controls equipment, and reports its status as the job proceeds. Processing occurs in both devices; the Mac turns the data file of an application into PostScript; the LaserWriter interprets that PostScript and controls the mechanical, electrical, and optical components of the print engine.

The flow of the relationship is read-driven rather than write-driven. Neither the Mac nor the LaserWriter can send data until the other device specifically invites it to send. The need for controlling flow in this way comes from the fact that the LaserWriter can't print as fast as LocalTalk can deliver data; the LaserWriter is the bottleneck. The LaserWriter limits flow by delaying the "send data" invitation until it has cleared enough buffer memory to accept another burst of data from the Mac.

The LaserWriter can accept only one client at a time. However, it can report its current state to other prospective clients on request.

Printer Access Protocol, which manages the relationship, has a tickle timer of one minute and a connection timer of two minutes. The loss of one tickle packet initiates a vigorous search for the other device that lasts until the connection timer is reached. The search typically is initiated by a device when the tickle packet is late by approximately 20 seconds. A Mac searching for a lost LaserWriter has noticeably slower performance during the search.

In most cases, both devices fully recover from a lost connection. In some cases, the LaserWriter might need attention to clear itself and become available for the next client. In rare cases, the Mac might hang or crash.

An Example of a Peer-to-Peer Relationship

An example of a nonsequential peer-to-peer process is Novell's Data Club. Data Club creates a virtual volume consisting of specially designated files from "member" Macs. Although the virtual volume behaves like a dedicated server, the contents of the virtual volume are physically distributed throughout a zone. A copy of the desktop information—the icons, filenames, and directory structure of the volume—is kept locally on every member's station. Through keeping multiple copies of the desktop information, Data Club has the unique feature of allowing users to see and manipulate files and folders whose contents are off-line. Off-line files and directories have a 0 file size, however, to let you know they're not currently available. For comparison, on a dedicated server, the server keeps the master desktop information, which it supplies to its users at login time.

The Macs all coordinate changes in desktop information by broadcasting (to the "experimental" socket 130) a checksum of the desktop information. When the checksum changes, the other Macs know that a change in the desktop information has taken place. The process is not always sequential; multiple Macs can simultaneously change the volume's directory structure, prompting several Macs to coordinate changes in the desktop information concurrently. Because of the independence that Data Club gives each member Mac, there might be multiple versions of the desktop information. The Data Club software coordinates these changes until the information is identical on all Macs.

As you can imagine, the dynamics of this process are considerably harder to follow during troubleshooting than are the dynamics of the client-server process. One way to follow them is to watch the proceedings with a protocol analyzer, making a list of each Mac's changes. You can "check off" other Macs as they incorporate that change. Then you can see which changes are complete and which are incomplete.

> **Disclaimer:** It's a little silly to be troubleshooting a commercial product at this level; if Data Club hadn't coordinated the changes properly, there's really nothing we can do to help it out. We've never needed to troubleshoot Data Club—it's always worked fine—we were just curious to see how it worked. We've included it here because it's a good example of the peer-to-peer process.

Data Club's complexity is state-of-the-art as networking applications go, with "fuzzy consistency" between independent stations, but this type of complexity is swiftly becoming the norm as we enter the world of cross-platform process-to-process communication. The use of broadcasts to a Data-Club-specific, "experimental" socket is not troublesome as long as no other applications try to use that same socket; it's probably the most efficient way to intercommunicate.

Signaling and Transmission

Because you've been speaking and listening to sounds for many years, you have a good feel for the relationship between distance and loudness. You're aware that many factors alter that relationship. For example, in a noisy room, you compensate for the noise by speaking loudly even at short distances. Similarly, you know that listeners who are farther away have a harder time understanding your words because the sound of your voice grows softer with distance. To speak to people far away, you must speak more loudly to compensate for the distance. From your experience, you've developed an understanding of the physics of sound transmission.

Although a physicist can express the relationships between distance, amplitude, ambient noise, frequency, and media characteristics mathematically, you already have a feel for the basic principles.

Network Transmission Theory

Books on transmission theory treat the subject as a mathematical exercise. While math is useful for specifying the exact nature of physical relationships, the math used in transmission theory is very complicated. The principles for transmitting an electronic signal over a copper wire are simple, sharing many similarities to the principles of sound transmission that you already know.

In this section, we'll discuss the principles of network transmission theory. When you troubleshoot, your understanding of these principles can help you formulate more realistic hunches and hypotheses. You might wonder, "Does this look like a reflection problem?" Knowing more about what causes reflections helps you consider that possibility.

Signals decay over distance

After a signal is created, it travels over some kind of media. In speech, the signal is a pressure wave that uses air as media. In networks, the signal is a voltage wave that travels (in most cases) over copper wire. In both systems, as in any transmission system, the signal interacts with the media and decays as a result. Its amplitude (loudness) get smaller and it becomes distorted by the interaction. For a listener to understand the message conveyed in the signal, the signal must be of high enough quality and loudness to be interpreted properly.

Most digital computer networks use square waves to encode the 0's and 1s that make up the communication. Square waves, pictured in Figure 2-8, are useful for encoding digital information because square waves themselves are digital: they oscillate between two voltage states.

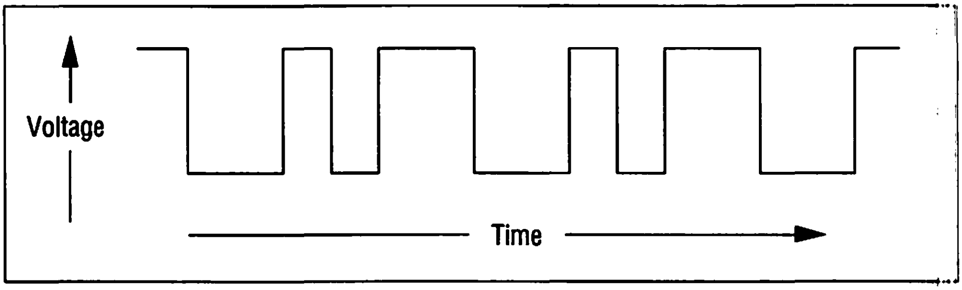


Figure 2-8. Square waves easily represent digital information because they oscillate between two voltage states.

On twisted-pair wire, square waves decay in four ways: through *loss of signal power*, from *frequency distortion* that degrades signal quality, from *reflections*, and as a result of *interference* from electrical noise.

Loss of power

Like sound waves, electronic signals get weaker as they travel along the media, but there is another way in which electronic signals get weaker that has no analogy in sound transmission. Passive junctions in networks (like the center of passive stars) weaken the signal by splitting the power of the signal among the number of branches at the junction. When a device on one branch of a four-branch passive star sends a signal, the signal travels to the center of the star, where it divides and distributes its signaling power among the other three branches. If the amplitude of the original signal was 6 volts, the split signals each have an amplitude of about 2 volts. This type of power loss is linear with (not affected by) the frequency of the signal and is sometimes referred to as DC Loss.

Frequency distortion

Frequency is a measure of how fast a signal is changing. Square waves change very rapidly in their corners (high frequency), but very slowly in their flat portions (low frequency). Because high frequencies decay faster than low frequencies, the corners of the square wave get rounded off as they travel down the wire. Because the spacing of the corners encodes the 1s and 0s, the signal becomes harder to interpret the farther it has to travel. Frequency distortion is sometimes referred to as AC Loss. This distortion is the same effect that causes bass sounds in music to travel farther through walls than high frequency sounds.

Reflections

You've probably heard an echo when standing at a distance from a wall or some other object. The sound of your voice travels through the air, hits the wall, and some of the energy of the sound wave is reflected back to you. Echoes (*reflections* in network terms) occur when a wave encounters a change in the media that carries it. Because the wall has different transmission characteristics than the air, some of the energy of your voice wave echoes. The amount of echo depends on the difference between the transmission characteristics of the two media.

Impedance is the transmission characteristic that describes a media's resistance to the motion of a wave. Plain network wire has a characteristic impedance. At any place where the wire changes, the impedance changes and some of the signal reflects, or echoes, when the signal encounters that change.

Stations on a network with reflections hear more than one signal at a time: the "incident" signal that the sending device is creating *and* the reflections of that signal as it bounces off various reflecting points. As you know from experience, echoes make it harder to understand what's being said.

Just placing a network adapter on the wire creates a reflecting point because the adapter electrically loads the wire at the connection point, locally changing the wire's impedance. Many network specifications include limits on how many network adapters can share a single wire. In the 10BASE-2 Ethernet specification, for example, a limit of 30 devices per network segment keeps the level of reflections to a reasonable level. Here are some other examples of wiring situations that cause reflections:

- Any junction—a punch-down block, patch panel, or wiring block
- Changes in wire thickness, quality, or insulating materials
- Any place the wire changes from shielded to unshielded

Reflections are a part of any wiring system, and reflections caused by the preceding conditions are tolerable in moderation. Although manufacturers set limits on the number, kind, and spacing of some kinds of reflections, the ways in which reflections interact is very complex. This complexity makes it difficult for a manufacturer to set limits that apply in all situations. Normally, small reflections don't cause problems unless there are many of them and they add together in the right ways. You can find more information about reflections in "Troubleshooting by the Layers: Physical Layer," in Chapter 6.

By far the biggest reflections that can occur are at the end of the wire, but you can easily prevent them by proper termination. Terminating a line means placing a resistor between the two conductors at the ends of the wire. The resistor's resistance should be equal to or slightly higher than the nominal impedance of the wire. LocalTalk resistors are 120 Ω.

In coaxial Ethernet wiring, lack of termination prevents the network from working at all. In LocalTalk systems, lack of termination or improper termination can have no effect at all to a very serious effect. In LocalTalk, however, even with the worst termination scheme, some of the nodes can communicate sporadically with some other nodes. LocalTalk termination problems don't produce catastrophic failure, as improper termination does in Ethernet.

Electrical noise

Electronic signals can suffer interference from electrical noise under certain conditions. Electromagnetic Interference (EMI) is the general term for this phenomenon. There are two forms of EMI. *Ambient noise* refers to disruptive noise from electrical equipment or fluorescent lights; *crosstalk* refers to disruptions that come from nearby wires carrying similar signals.

A signal's tendency to be adversely affected by electrical noise depends on many factors. Those factors are what you must examine to see whether EMI is a problem. Like reflections, EMI in both forms is ubiquitous; it can be minimized, but not eliminated. The likelihood that a signal will be negatively affected by electrical noise is influenced by the factors discussed in the following sections.

Twisting of the wire. Twisted wire is used because the twisting of the conductors causes electrical noise to affect both wires evenly. Because the wires are affected evenly, the relative voltage of the wires with respect to each other remains constant, even though the absolute voltage of the wires can change. Higher quality wire, with more twists per foot, is more effective at averaging the noise signals and more effective at resisting noise.

Electrical noise can come from outside the wire (ambient noise) or from other signals on other conductors inside the same wire sheath (crosstalk). Crosstalk is prevented only when the pairs of conductors in a wire are twisted relative to each other.

Noise strength. Stronger noise signals have greater influence and are more likely to produce a negative effect. If there's a strong noise source, you might need to alter one of the other factors to compensate; for example, moving the network wires away from the noise source or using wire that has more twists.

Similarity between noise and signal. Noise that has similar characteristics, particularly a similar frequency, to the normal signal is more likely to cause disturbance. That's why the wires and connectors carrying normal 60 Hz electrical power, a very strong signal, typically don't affect the quality of a network signal. Fluorescent lights give off a very wide spectrum of radiation and always contain some frequencies that are similar to the network signal. Any device that uses an electric motor, such as a copy machine or a dot matrix printer, can be suspect as well.

A most extreme case of signal similarity is in 10BASE-T networks, where two pairs of wire in a single jacket carry the receive signals and transmit signals, respectively. If the twist isn't adequate, a signal on one of the pairs causes the other pair to resonate (crosstalk). The transceiver interprets this as a collision and renders the station connected to this wire inoperable. That's why higher grades of wire are required for 10BASE-T ; they have the greater amount of twist necessary to prevent crosstalk between the two pairs of wires.

Shielding. Shielded cable is insensitive to EMI, but shielding causes low-speed signals, like LocalTalk, to suffer signal power loss faster than unshielded wire. At Ethernet's speed, however, the benefits that shielding holds for the signal are approximately equal to its drawbacks. At high speeds—for example, ten times faster than Ethernet—shielding is probably necessary.

Proximity. The closer the noise is to the signal, the greater the probability of harmful interaction.

In crosstalk, the distance of separation between the two wires is, of course, very small. The longer the two signals travel together, the greater the chance that they'll interact with each other. That's why 10BASE-T signals can be sent over voice grade wire *only* if the wire distance is very short.

Network Bandwidth and Utilization

In analog communication technology, *bandwidth* is a measure of the range of frequencies that can be sent by a transmitter. The width of the frequency range is related to how much information can be sent by the transmitter; higher bandwidth signals carry more information. While the term bandwidth doesn't strictly apply to digital networks like LocalTalk or Ethernet, it's often used to describe the information-carrying capacity of these networks. It's usually expressed qualitatively, such as "That's because of the low bandwidth of LocalTalk." When expressed numerically, it's expressed as a number of bytes per second that the network can carry. Utilization is the portion of bandwidth, usually expressed in percentages, in use at a given time.

NetStats and TrafficWatch are two tools that can show the level of utilization on the network over time. In troubleshooting, the primary importance of the utilization graph is to help you get a qualitative feel for a particular network process. Is the network busy right now? How is the network affected if users back up their hard disks to the server in the middle of the day? How do system variables like CPU type and speed, RAM size, and system and application versions affect performance? How do concurrent network processes affect each other? How long does a file transfer take to complete? How is a complicated process like printing coming along? By looking at a graph of utilization versus time, a network manager can formulate answers to these questions.

In troubleshooting, utilization graphs are best used for comparison to a normal situation. If you know that a process—printing, for example—has a certain utilization profile under normal circumstances, looking at the graph of utilization during a problem can provide an important clue to locate the trouble. Utilization information can also be useful even without comparison data; it can help you, for example, estimate how long it might take for a long process to complete or discover why a certain database query takes longer than another. The information is useful because it shows the volume of information transferred over the network to fulfill the queries.

Bandwidth is a useful way of characterizing a network, and utilization is a useful way of characterizing the way that a network is being used. But bandwidth and utilization are widely misunderstood, difficult to specify, and not easily measured. It's important to understand the concepts, however, to make good judgments based on the data that you gather. To better explain exactly what the utilization data means, we'll discuss the major issues surrounding these concepts, using a file transfer over LocalTalk as the basis for the discussion.

Calculating bandwidth

The speed of LocalTalk is 230.4 kilobits per second (Kbps). That's simply the rate at which bits of data are transmitted along the wire. You can't just divide that speed by 8 bits/byte and say that the bandwidth of LocalTalk is 28.8 kilo*bytes* per second. The reason is that every network has, designed into its specifications, a certain amount of unusable transmission time, or "overhead." Overhead exists because networks need to provide control and order as well as to send data. The time that the network devotes to maintaining control and order is time that it can't spend sending data. Due to this overhead, bandwidth is always less than the speed of a network.

A useful analogy is a highway. For safety reasons, drivers leave gaps in between their cars and the next cars. The carrying capacity of the highway is diminished because of these unusable spaces.

LocalTalk's specifications require some dead time where no communication occurs. Just like the gaps between cars, the network is required to leave gaps between the packets. This dead time is necessary to avoid certain kinds of network errors, primarily collisions. LocalTalk is a collision avoidance network, and most of its required dead time results from the way that it avoids collisions. All networks have some dead time in their design. We can call this *hardware overhead*, because it arises from the needs of the network and not from the needs of the protocol using the network.

In addition to the hardware overhead introduced by LocalTalk's need to have orderly communication, AppleTalk protocols also contribute some software overhead. Each packet sent on the network contains a certain amount of AppleTalk protocol control information in addition to the data it carries. This extra information is necessary to guide the packet through the network and provide certain kinds of control. The extra protocol bytes might identify the sender and receiver of the information, indicate the sequence of the packet in the overall conversation, or tell how to interpret the data being sent, but they also consume some of the time available for transmission.

Returning to our highway analogy, the purpose of the highway is to carry people and goods—but people and goods must ride in vehicles. In addition to the gaps in between the vehicles, the vehicle itself takes up some space. This space is also inefficient because it can't be used to carry the payload.

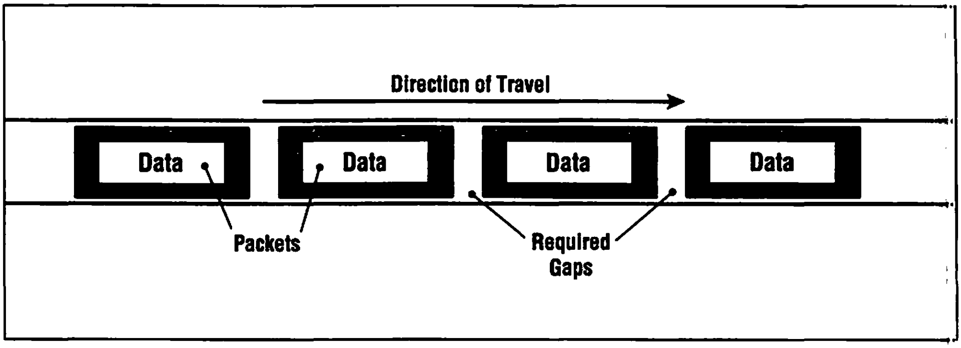


Figure 2-9. Just like passenger cars on a highway, there's a network carrying capacity that isn't usable for data due to the needs of the network and the protocol using the network.

The actual bandwidth of a network is always less than the speed of the network due to the needs of the network and the needs of the protocols on that network. When you calculate the ability of the network to carry information (the bandwidth), the number that you get depends on whether you take into account the overhead. If you measure the utilization of a file transfer over LocalTalk at 40 percent, does that mean the network is sending data at 40 percent of LocalTalk's speed,40 percent of LocalTalk's bandwidth, or 40 percent of AppleTalk's ability to use LocalTalk as a conduit for data? How does the value of 40 percent relate to how many bytes are being sent per second and how long does it take the file transfer to complete?

Bandwidth calculations and utilization measurements frequently take into account the hardware overhead but ignore the overhead introduced by the protocol. Some calculations ignore both causes. Also, both hardware and software overhead are made up of many contributing factors that exert varying degrees of influence on the amount of time that each overhead consumes; the number that you calculate as the bandwidth depends on how many of those contributing factors you use in your model.

Because different troubleshooting tools use different methods to show utilization, you need to be aware of the differences if you want to compare results from different tools or use the results of one tool for a calculation or estimate you want to make. To illustrate the differences, let's look at the quantitative effect of some of these variables and the effect on the calculation of LocalTalk bandwidth. Then we'll compare LocalTalk utilization measurements from a stopwatch, from a HyperCard tool we've invented called PacketCrunch, and from Farallon's NetStats.

The effect of packet size

For any network, there's a limit to the amount of information that can be sent in a single packet. Because the size of the required gap is independent of the size of the packet, obviously the network is more efficient and can carry more data if it uses larger packets. This concept is graphically shown in Figure 2-10.

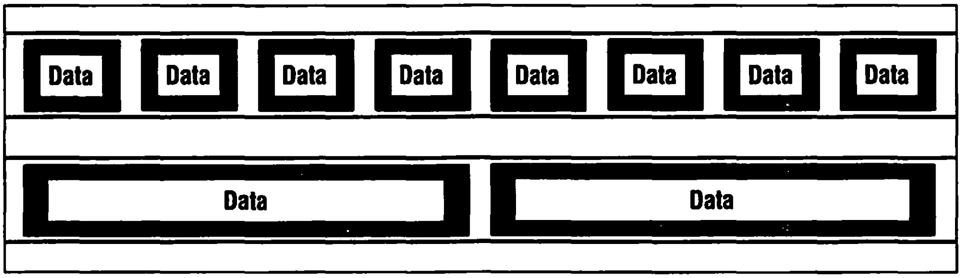


Figure 2-10. When the network uses larger packet sizes, it can send more data more efficiently because the effect of the gap in between the packets is minimized. In the figure, the bottom network is sending twice as much data as the top network in the same amount of time because it's using larger packets.

One way to calculate the bandwidth of the network is to take the time needed to send the number of bytes in the maximum size packet and divide it by the total time necessary to send that packet. This takes into account the hardware overhead but ignores the software overhead. For LocalTalk, the gap between packets is about .001 seconds and the longest packet allowable is about 600 bytes, which takes about .021 seconds to send. The efficiency of a 600-byte packet is therefore about 95 percent—the time spent sending data divided by the time needed to send the data. Using this efficiency rating, you can calculate that the bandwidth of LocalTalk is the speed of the network times the efficiency rating.

LocalTalk bandwidth: Calculation #1

(This calculation takes into account the hardware overhead for large packets.)

$$\begin{aligned}\text{Effective Bandwidth of LocalTalk} &= \text{Speed of LocalTalk} \times \text{Efficiency of LocalTalk}\\ &= 28.8\text{ Kbps} \times 95\%\\ &= 27.4\text{ Kbps}\end{aligned}$$

Of course, this bandwidth calculation is only good for the maximum size packet of 600 bytes. Efficiency, and therefore bandwidth, varies with packet size. If you perform the calculation for different size packets and then plot the results, you get a relationship between bandwidth and packet size that peaks at the maximum packet size of LocalTalk of 600 bytes. Notice that this is the maximum packet size and the bandwidth is the same as calculated above.

LocalTalk bandwidth: Calculation #2

(This calculation takes into account the hardware overhead of any size packet.)

$$\text{Bandwidth} = \text{Speed} \times \text{Efficiency}$$

Efficiency is a function of packet size.

Efficiency and bandwidth calculation is performed for various packet sizes and plotted.

$$\begin{aligned}\text{Efficiency} &= \frac{\text{time needed to send data in packet}}{\text{total time of packet}}\\ &= \frac{\text{number of bytes in packet} \times \text{time needed to send one byte}}{\text{time to send data} + 0.001 \text{ seconds}}\\ &= \frac{N \text{ bytes} \times 0.0000347 \text{ seconds/byte}}{0.0000347\,N + 0.001 \text{ seconds}}\end{aligned}$$

As expected, long packets can support a higher bandwidth. Because of this factor, the bandwidth can't be simply stated as a certain number of bytes per second; bandwidth must be stated as a number in relationship to the packet size used.

Some network "conversations" naturally take place in shorter packets. An example is a remote terminal session, where each keystroke is sent to the host and echoed back—two one-byte packets for each keystroke. Other types of communication, like file transfer and printing, naturally take place in longer packets, but there are some number of shorter packets used in the process for control purposes.

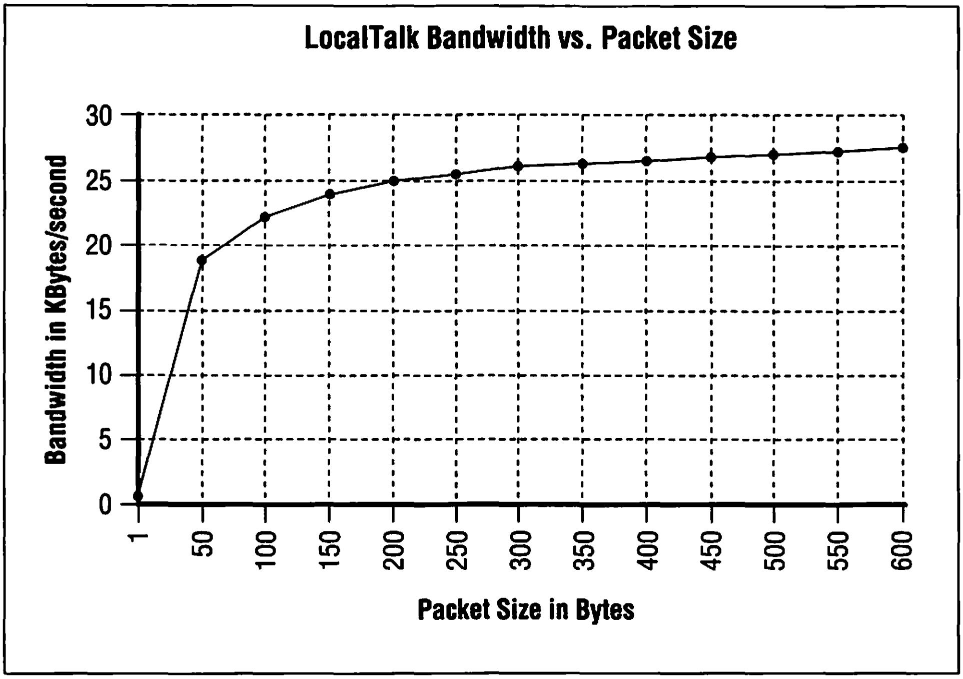


Figure 2-11. The bandwidth of LocalTalk varies according to the size of the packets used to carry the information. At packet sizes less than 150 bytes, the efficiency of the network drops quickly.

Now let's explore the effect of the factor of packet size and the influence it has on the utilization measurements of a particular process. The process we'll use as the example is the transfer of the application Apple File Exchange over LocalTalk from a Mac Portable to a Mac SE, both using unaccelerated MC68000 processors—two relatively slow machines. The transfer application we'll use is the ADSP mechanism in Farallon' s Timbuktu 4.0 file exchange module. The file size is 245K, but remember that K in this case means 1,024 bytes, not 1,000. The actual file size is 250,009 bytes; you can get that from the Get Info window. We'll figure this as the number of kilobytes and for the rest of this section, one kilobyte (K) means 1,000 bytes.

Utilization: Using a stopwatch

Using a stopwatch, you can calculate the utilization during a file transfer. Because utilization is the portion of bandwidth used, you can divide the transfer rate of the file by the bandwidth and call that result the utilization. To determine the transfer rate, simply divide the number of bytes transferred by the time taken to transfer them. Then to calculate the utilization, divide that transfer rate by the bandwidth.

In the test, the 250K file took 24 seconds to transfer, so the transfer rate was 250K/24 seconds or 10.4K/second. This shows utilization to be 10.4/27.4, or about 38 percent. Here is a summary of the calculation:

Utilization: Calculation 1

(This calculation uses a stopwatch to time the transfer of 250K.)

$$\text{Utilization} = \frac{\text{Transfer Rate}}{\text{Bandwidth}}$$

$$= \frac{\text{250 Kbytes / 24 seconds}}{\text{27.4 Kbytes / second}}$$

$$= \frac{\text{10.4 Kbytes / second}}{\text{27.4 Kbytes / second}}$$

$$= 38\%$$

Utilization with PacketCrunch

To refine the calculation, you can take into account that the file transfer didn't exclusively use full-size packets. While much of the data was sent in large packets, there were also many smaller packets. To take into account the effect of the smaller packets, you can refine the calculation so that the utilization of the entire process is the cumulative of the momentary utilization of each packet in the process. To calculate this momentary utilization on a second-by-second basis, you need a protocol analyzer to record the size and time of transmission of each packet. Then you can calculate the overall utilization as the summation of these momentary utilizations. This is the method PacketCrunch uses to determine utilization.

PacketCrunch shows the size distribution of the 1,127 packets and the network utilization during the file transfer as shown in Figure 2-12.

As you can see, during most of the file transfer, PacketCrunch reported a utilization higher than the measurement calculation of 38 percent. During a few of the seconds the utilization was over 50 percent, but was mostly in the mid-40s. There are two reasons PacketCrunch gives a higher number than the stopwatch calculation. The first is that when smaller packets are used, the bandwidth is lower, so a given amount

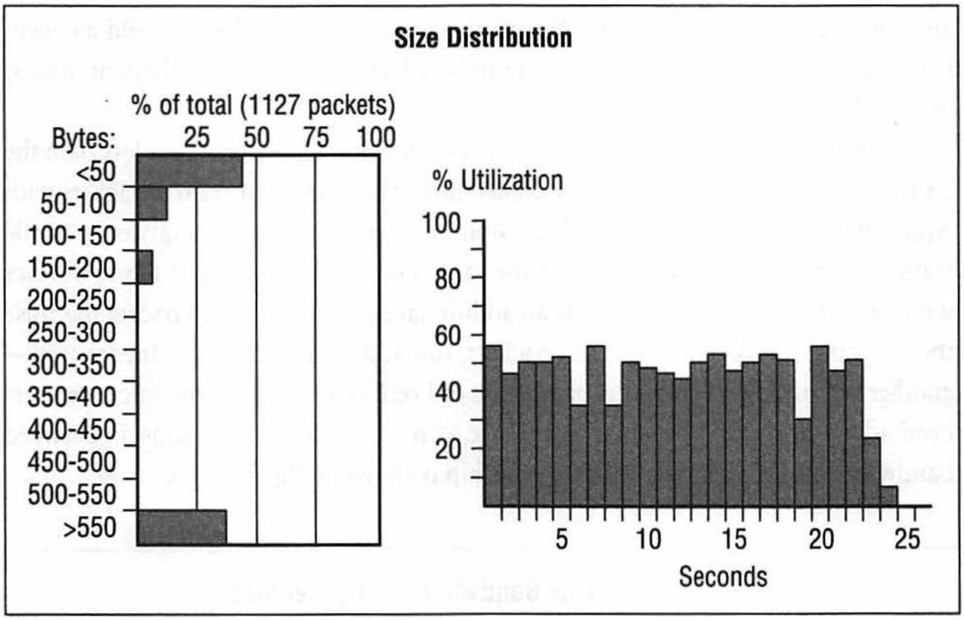


Figure 2-12. PacketCrunch shows a utilization higher than the stopwatch method because the network sends more data over the wire than just the contents of the file, and because many short packets were used.

of data moving through the network in small packets has a higher utilization in small packets than in large packets.

The second reason that the stopwatch calculation was low is that our 250K was sent along with a lot of AppleTalk protocol information. Timbuktu 4.0 uses ADSP for the file transfer mechanism; each ADSP packet on LocalTalk carries with it 21 bytes of protocol overhead. Additionally, the receiving Mac also sent many control packets to the sending Mac to inform the sender of the progress and accurate receipt of the packets.

The actual amount of data that Timbuktu transferred for this 250K file was 265,680 bytes from sender to receiver and 8,580 bytes from receiver to sender, for a grand total of 274,260 bytes. If you use the actual number of bytes transferred in those 24 seconds for the utilization calculation, you get a transfer rate of 11.43 Kbps and about 42 percent utilization. This is much closer to the answer given by PacketCrunch.

The reason that the stopwatch calculation was accurate at all is that the rate of utilization during a file transfer is fairly consistent throughout the process and the number of control bytes was not overwhelming. If a process is characterized by stops and starts, like printing to a LaserWriter, this crude technique doesn't yield an accurate measurement because it can't accurately reflect the peaks and valleys of utilization in the process.

ADSP's contribution of 21 bytes per packet for protocol overhead is less than the 28 bytes per packet that AppleShare contributes. The number of control packets with AppleShare is also much higher. Those numbers of control bytes apply only to file transfers sent between two Macs on the same LocalTalk network. If the two Macs were on different networks, there is an additional eight bytes in each packet. Because the file transfer took place in 1,127 packets, this adds nearly 9K to the file transfer—another second. If you graph the bandwidth of LocalTalk, including the software overhead added by ADSP in a single LocalTalk, you get a different relationship between bandwidth and packet size. This relationship is shown in Figure 2-13.

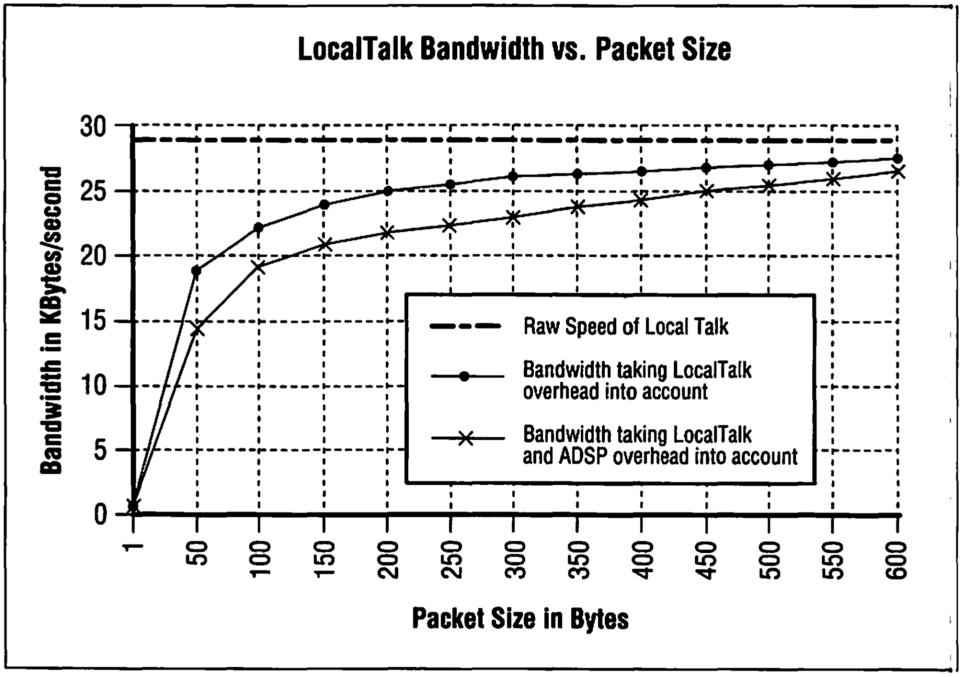


Figure 2-13. Timbuktu 4.0, which uses ADSP as the protocol control mechanism, adds 21 bytes to every packet sent on LocalTalk. This software overhead further reduces the bandwidth of LocalTalk, as shown.

Utilization with NetStats

Let's compare the calculations we just made with those of NetStats, which is part of Farallon's Network Manager's Pack. This tool takes a more statistical approach to utilization: it samples the LocalTalk port to see whether there's a signal on it. It makes a utilization calculation measurement based on the percentage of samples taken at a time when there was activity. It makes this calculation aside from any of the considerations of number of bytes or packet sizes; it reflects only the percentage of time that there is a signal on the line. This is an entirely different method—but how does its information compare to the numbers we developed previously?

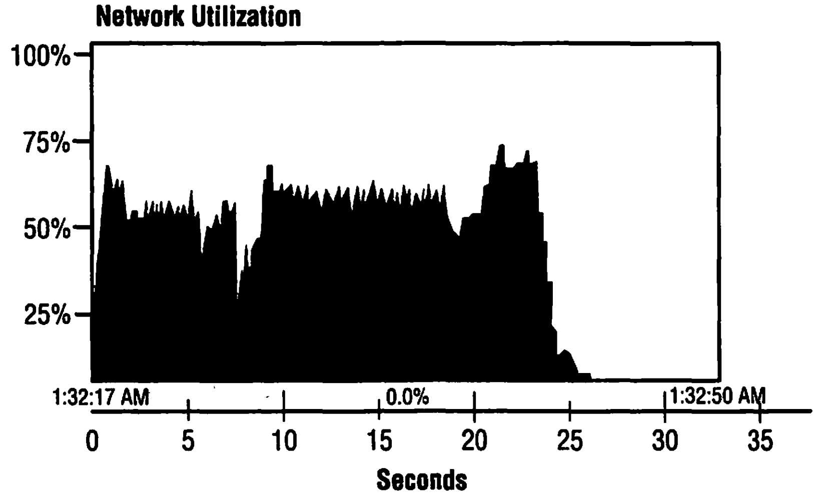


Figure 2-14. Farallon's NetStats uses a statistical means of measuring utilization. It tests whether the LocalTalk network has a signal on it and reports the percentage of time the port showed activity. We've added a time scale in seconds for convenience.

As shown in Figure 2-14, NetStats shows a utilization slightly higher than either PacketCrunch or the stopwatch calculations. Utilization hovers in the 50 percent range, and you'll notice similarities to the PacketCrunch graph in the general shape of the curve. In terms of hardware and software overhead, NetStats takes neither into account, and while it should provide the lowest bandwidth measurement, it actually provides the highest. This discrepancy comes from the way NetStats makes its measurements. It samples the LocalTalk port to see whether it's in synchronization with a signal. Because this method of sampling can be affected by many system variables, such as the amount of CPU time that NetStats receives and whether the port is reading a broadcast packet, it isn't very reliable for accurate measurements.

Despite this, NetStats remains a useful troubleshooting tool to give you a feel for the network and its processes. For example, you can use it to gauge how fast the printer is accepting data by looking at the time delay between the spikes of activity. The longer the delay, the longer it's taking for the printer to turn the PostScript information into an image.

Summary

For file transfer, the utilization rate that we determined was completely dependent on the method of measuring used. The values ranged from a low of 38 percent using a stopwatch to a high of 50 percent using NetStats. Utilization measurements and graphs are useful troubleshooting tools, although you must be aware of the way in which the numbers are determined and the manner in which the theoretical bandwidth of the network is calculated. These factors significantly affect the values taken by the measuring tool.

CHAPTER THREE

Beginning the Troubleshooting Process

So, in spite of all your best efforts to create a network that is well-designed, well-documented, and smoothly running, your network has developed a problem. In all likelihood, it has been reported to you by one of the users of the network, who almost certainly has work that needs to get done, and who would prefer that the problem be solved yesterday or, failing that, immediately. Where are you going to start?

"Well Begun is Half Done"

No serious networking problem is likely to be solved before it is at least to some degree understood. Random troubleshooting is essentially the process of trying remedies without analyzing the nature of the difficulty. Frequently you'll wiggle cables, restart systems or give the offending component an "impact adjustment" in the hope that the problem will go away, and in many cases it does. But when it doesn't, you must start looking a little deeper. Before you can develop a diagnosis, you must examine the patient. The obvious place to begin your investigation is with the person who brought the problem to your attention.

The user is your main source of information in the initial stages of examination. Frequently, however, your primary source of data can have a difficult time describing to you precisely what's happening in terms that are useful to you. It's usually necessary to focus on the aspects of the problem that begin to lead you to a solution.

Try to keep in mind that keeping the network operating correctly is *not* the user's job. The user's job is *using* the network to accomplish whatever their *real* job is. To the extent that the real work can't get done, the network is not fulfilling its role. Getting it to fulfill its role is *your* job.

An important part of your job is translating what the user is saying into pertinent information that you can act on. The user almost certainly won't be familiar with the intimate details of how file sharing works or of the importance of proper configuration in setting up internetwork routers. It would be unreasonable for you to expect otherwise. What you need to do is a combination of eliciting specific information from the user and examining the problem itself by reproducing the user's actions based on the answers to your questions.

By questioning the user and walking through the problem, you can develop a sense of the nature of the problem and the circumstances under which it's occurring. Can it be reproduced? Can it be isolated down to a problem in a specific hardware component or a particular piece of software? Most importantly, what's changed since the last time things were working right?

Start by eliminating the obvious possibilities. A document can't be printed: is the printer selected, or is the output ending up someplace unexpected? A server can't be mounted: is it running and on-line?

There are some methods you'll probably want to try almost every time you look at a problem. These might include such things as bringing up the Chooser or running Inter•Poll. These tactics generally require a minimal amount of time to execute and give you a baseline of information from which you can proceed.

There are also a number of random troubleshooting responses you might consider trying. If your Macintosh computer appears to be the source of the networking problem, use the Chooser to turn AppleTalk off and then back on again, or more drastically, restart the computer. If you have more than one network driver, try switching to a different one. For example, if you're connected simultaneously to Ethernet and LocalTalk networks, you can momentarily choose the alternative network with the Network Control Panel and then switch back to your regular network choice. This forces your Mac to join the first network all over again and re-alert other devices of its presence.

If you see that an entire group of users is affected, you may suspect that the problem may be elsewhere on the network. You may try cycling the power on some of the network devices, but you must be very careful not to sever any healthy connections that other users may have with or through those devices. For example, if you turn off your hub while another user on that hub is in the middle of a file transfer, the other user will experience a "blip" in their file transfer—or something worse. If the power to the hub is only off for five or ten seconds, his or her Mac will easily resume the transfer when the power is restored. If you leave the power off for more than one minute, then the Mac (and the workstations of any other users who are connected through the hub to a file server) will become "nervous," and begin frantically searching for the file server in an effort to save the connection. After two minutes, the Macs will give up and sever the connection; any interrupted actions will not be completed. Further, any of the server's files that the user has opened on his or her workstation will be "stranded." If the memory allocation of this application is large and the size of the server's file is small, it's possible that the user will have the entire file is RAM. If that's the case, the user can do a "Save As..." and save the file along with all of its edits. On the other hand, if the server's file is bigger than the RAM space, only a part of the file will be in the workstation's memory. After the connection to the server is reestablished, it's very important that the user not replace the original file on the server with the portion of the file that exists in RAM—it's not the complete file. Any portion of the file not in the workstation's memory will be lost, probably for good.

Resetting a router is even more dangerous than resetting a hub because routers typically require a longer warm-up time before re-entering service. Hubs usually require five seconds or less to warm up and begin operation. Depending on the make and model of a router (and, to some degree, the size of your internet), a router's warm-up period could last anywhere from 15 seconds to two minutes. Again, you must be careful not to make any healthy connections expire in the meantime.

Turn off all your INITs with Aask or INITPicker, or remove them all from your System Folder, and restart your Mac. In System 7, you can temporarily turn off all INITs by holding down the shift key while the system starts up. INIT conflicts are tricky and unpredictable, and few people understand why they cause problems. If the problem goes away after you've turned off the INITs and restarted, it is probably an INIT conflict of some kind. Try turning the INITs back on, one by one; this can help you discover which one is the culprit. If you find the problem INIT and you really must use it, try renaming it so that it loads in a different order (for example, adding a tilde in front of the name forces it to load last). Be careful, though; some INITs don't work properly if they're renamed. In System 7, this is typically less of a problem because the memory management of the system heap space is greatly improved, although still not perfect. The biggest change is that the system can allocate more memory on the fly. In System 6, once the system heap memory is allocated, it cannot be expanded.

A number of pieces of important network-related information are stored in the Macintosh's nonvolatile parameter RAM, also known as the "PRAM." Among these are the following:

- Hints for network and node numbers
- The link access protocol that should be used, whether LocalTalk, EtherTalk, TokenTalk or some other type
- The name of the node's default zone

If the information in the PRAM becomes corrupted, odd results can frequently follow. Problems of this type can be straightened out by clearing, or "zapping," the parameter RAM. Under System 7, this can be accomplished by holding down the command, option, P and R keys while the machine starts. You can tell that the parameter RAM has been cleared when the machine restarts for a second time before it comes up completely. Release the keys, and the machine starts normally, selecting fresh network and node numbers, and using LocalTalk as the link access protocol. If the Mac needs to use EtherTalk or some other link access protocol, you must select it again using the Network Control Panel. Clearing parameter RAM on system releases prior to System 7 is best accomplished using one of several desk accessories or applications designed specifically for this purpose. A good one is PRAM Slammer, which also allows you to inspect and modify the values currently stored in PRAM.

Check for viruses on the file servers and on your Mac. If you use a commercial virus checker such as Symantec's SAM or Mainstay's Virex, make sure you have the latest version and the latest virus information. If you have a shareware or freeware checker such as Disinfectant, check the bulletin boards or commercial on-line services to find out about your favorite's latest release. Viruses don't usually cause network problems (although they can be transmitted over a network), but it's worth a try.

Reload the software in network devices such as hubs, routers, gateways, and servers. Reenter the configuration information, making sure it is exactly what you want and is compatible with the other devices in your internet. It's possible that a colleague changed the software or configuration tables without your knowledge and misconfigured them, or perhaps you've misconfigured them yourself.

Swap out the various devices, cables, connectors, and network cards on your machine and across the network. Do this one component at a time, and see whether anything changes. Usually if cables or connectors are bad, they are bad right out of the box, but sometimes they go bad while in service. Maybe a forklift ran over your network wires, lightning struck the building, or something wiggled loose.

If you use Ethernet or a token-ring configuration, reinstall your network software or reconfigure your workstation network information. If you have an older Mac, you might need to patch its ROM by running a newer version of AppleTalk, or if you are running under System 6, you might need to put the ADSP (AppleTalk Data Stream Protocol) patch file into your System Folder (this file comes with some network products).

Remove your password from the server, or reconfigure your user information with the administrator software that comes with your server. Maybe you typed the password incorrectly. Also, passwords are usually case-sensitive, so be sure you're not typing in your password with the Caps Lock key down.

Try adding more RAM to your machine (or the server) or using the Finder instead of MultiFinder. Use CE Software's HeapFixer to increase the size of your system's heap space. Do the same for the server if necessary. If your central workstation is running more than one network service (such as file-server, mail-server, and print-spooler software), try placing the services on different machines or disabling one of them temporarily.

Try reading the manual, even if you've read it before. It's strange, but sometimes while you're having a problem, you could open the manual and see an obscure passage you've never noticed before or didn't understand. This has happened to us more than once.

We're not recommending the random method as your usual troubleshooting approach, but there are times when it does the job. However, if you're having network problems more than twice a month, you should investigate further, using a more systematic approach and more traditional troubleshooting tools.

A drawback of random troubleshooting is more subtle. Restarting a system, for instance, might sometimes appear to resolve the immediate problem, at which point you will probably be tempted to congratulate yourself on a job well done. But you might well discover later that the same problem, or a variant, has resurfaced. It might even recur on a different workstation than the one where it was originally observed. This is a strong indication that the real underlying problem has not actually been addressed yet.

Network problems can stem from a wide variety of causes, and the relationship between the basic cause and the effects you can see isn't always obvious. Networks are complex and built from many individual pieces, some hardware and some software, all of them interrelated. Any one of them can conceivably develop a problem on its own. Software can become corrupted, and hardware can break. A problem involving a failure of a single specific component is relatively easy to solve: replace or repair the component that's causing trouble and the trouble goes away.

On the other hand, the difficulty might be caused by an interaction between two components that are individually working within tolerance. An example is a LocalTalk network constructed from a variety of cables with widely differing impedances. While each of the individual cables might be fine in isolation, when they're connected they could cause serious reflections on the various segments of the network, which would make any transmitted signals unintelligible.

A good initial approach can be to see whether the difficulty can be isolated in some way to a particular network, device, software application, or hardware component. You might try running through the same sequence of events on a second machine. If you suspect a difficulty in hardware, a time-tested strategy is to "swap out" particular components to see whether another component of the same kind also manifests the problem.

The most important aspect of isolating a problem is determining what is different between a machine that manifests the problem and a machine that doesn't. This might entail finding another machine with an identical configuration, or comparing the setup of the nonworking machine with its setup when it last functioned correctly.

It is important to gather enough information to be able to choose a specific troubleshooting strategy intelligently. Random troubleshooting is probably worth trying for a number of basic and common problems, such as checking for loose connectors, but relying exclusively on such an approach would be unproductive and wasteful. The same can be said of exclusively using a network analyzer to examine every problem that is brought to your attention. Sifting through the volume of information produced by an analyzer might help you find a misconfigured router, but it isn't the best way to find a marginal LocalTalk cable.

Different troubleshooting strategies lend themselves most effectively to particular kinds of situations. Each situation has a different set of requirements and constraints. Your usual starting point will be some hunch troubleshooting; you can gather data, come up with theories, and test hypotheses. Frequently, however, this approach won't be sufficient to solve the problem within a reasonable time frame. At that point, you need to consider the nature of the problem and see which of the troubleshooting approaches best matches the situation.

Setup mode troubleshooting tests the electrical continuity of the physical components that make up the network. It's frequently most useful when a workstation is completely unable to access the network or when a given task cannot even be started. You'll need to use electrical test tools such as a tone generator or an ohmmeter; in addition, you'll need to be able to access the physical components of the network. Setup troubleshooting is most useful in resolving hardware-related configuration problems such as bad wiring or broken network interfaces.

Layer mode troubleshooting examines the functional continuity of the network by examining the interactions between the various layers of the AppleTalk protocol suite. You'll find it most useful when the problem is difficult to describe. Network management software running at the network's hub and utilities such as Inter•Poll are especially useful tools for this type of troubleshooting. You'll need a good overall understanding of the AppleTalk protocols; plan to invest in a copy of *Inside AppleTalk*. Layer troubleshooting is best used to find and resolve configuration problems affecting the network software as well as problems with the network itself, such as misconfigured routers.

Process mode troubleshooting looks at the events that make up a particular network process, such as mounting a volume on a file server or printing a document to a LaserWriter. This strategy is most helpful in cases where a task can be initiated but can't be completed in a satisfactory fashion. The most useful tool for troubleshooting a network process is a protocol analyzer, such as Network General's Sniffer or EtherPeek from the AG Group. To successfully perform process troubleshooting, you'll need to have a detailed understanding of the mechanics of the process you're attempting to troubleshoot. Several standard processes are detailed in Chapter 7; others can be usefully observed, again with a protocol analyzer, when they are working correctly. This kind of educational process is critical to successful process-mode troubleshooting. Process troubleshooting can be applied to a wide range of problems. It can, however, be time consuming to perform, as it involves sifting through a large volume of data, much of which might not be relevant to the problem at hand.

The characteristics of the various troubleshooting strategies are detailed in Table 3-1.

Table 3-1. Characteristics of the different problem-solving strategies

Start your investigation in:	**Hunch Mode** Gather clues, develop theories, try solutions		
If necessary, switch to:	**Setup Mode**	**Layer Mode**	**Process Mode**
Which tests the continuity of:	Components	Functions	Events
Is most useful when:	You can't even begin the task	The problem is hard to describe	The task begins, but results are unsatisfactory
Makes use of:	Electrical tools	Network management software utilities	Protocol analyzers
Requires:	Physical access to network components	General understanding of AppleTalk protocols	Detailed understanding of process mechanics
Usually finds:	Hardware configuration problems	Network and software configuration problems	All problems

Examining the Problem

"Listen, do you have a minute? I can't get this thing to work."

This is frequently where things start. The difficulty can be virtually anything. From the user's point of view, the problem is easy to describe: things don't work. The process of troubleshooting begins with the journey from the general to the specific. What is "this thing," and in what way isn't it working? What exactly are the symptoms?

"I can't find the server with last month's sales figures."

Well, now you know that you're discussing a problem with a file server. But what does it mean to say that you "can't find" a server? More often than not, you start the process of mounting a server volume by opening the Chooser. Is the problem that the server is failing to show up in the Chooser's list?

> *"Um, I don't know; I wasn't using the Chooser. I was mounting it with an alias."*

It's never too early to ask a core question: can the problem be reproduced? A problem that can be made to happen 100 percent of the time is in many ways the best of all possible worlds. If you can get the difficulty to happen whenever you'd like it to, you have the opportunity to choose your tools and plan your strategy. Many times, however, it might not be immediately apparent what the set of circumstances are that allow you to reproduce the error. Some problems might be transient and cannot be reproduced reliably at all. These are invariably the most difficult situations to address. This one, fortunately, appears to be fairly consistent.

> *"Yes, I've tried it a bunch of times, but the same thing happens. Here, look. See? It just chugs along for a minute, and then that message comes up."*

Here's your first data point from the network itself: an error dialog containing the text "The volume . . . can't be mounted because it cannot be found on the network." In the context of mounting a server with an alias, this generally indicates that your node has sent out some number of NBP LookUp frames in an attempt to locate a device answering the description of "Device Name:Device Type@Zone Name," and that it has not received an answer in the form of an NBP LookUpReply within the timeout period. On the other hand, if you *were* able to get a response but ran into some other problem, you would see a different error message, perhaps one indicating that your username or password had not been recognized by the server.

At this point, you might try out a bit of random troubleshooting. The alias clearly isn't working. There might be a number of causes for this: the volume might have been renamed, the server might have been renamed or moved into a different zone, or the alias itself might have somehow become corrupted. As a simple cross-check, it is probably worthwhile to try to find the server by looking for it with the Chooser.

In this case, the Chooser does not, in fact, show a server of the name the user has mentioned where the user is expecting to find it. It does, however, show a number of *other* servers in the zone, so the problem doesn't seem to be in the AppleShare workstation software.

Explore the Boundaries of the Problem

As a check on this assumption, you might try to mount and access one of the other servers. You do, and it seems to be working fine. So whatever the problem is, it seems at the moment to be specific to this particular server.

You would probably, as a matter of course, bring up Inter•Poll at this point to perform an exhaustive search of the zone in question to see whether the Chooser itself might be having a problem. Just as bringing up the Chooser allowed you to cross-check the correct operation of the alias, Inter•Poll allows you to cross-check the Chooser. It would be optimistic, to say the least, to expect radically different results from Inter•Poll, but it's always a good idea to check your results in several different ways. Searches performed in different ways might produce different results, and any inconsistencies might be very revealing.

Let's start by eliminating the obvious problems. Since both the Chooser and Inter•Poll search for devices only on a zone-by-zone basis, the easiest explanation might be that you're simply not looking in the right place for it. More often than not, these out-of-the-chute possibilities won't pan out, but when they do, they can be tremendous timesavers and the problem can be resolved immediately. So you ask the user if they're looking in the right zone.

> *"I think so. That server has always been in the Administration zone whenever I've accessed it."*

No luck. Is it possible that the name of the server has been changed? You might call up the administrator of the server to find out. You can also ask whether the server might have gone down or been taken off-line for some reason.

However, a more expedient approach might be to walk into another office down the hall and see whether things might look any different from there. This allows you to accomplish a couple of things: you can check whether the server in fact exists under the name and in the place you expect, and you can see whether the problem is peculiar to the specific workstation you've been looking at or whether other nodes might also be sharing this problem.

> *"Nope, I can't see it from here, either."*

So it seems that the problem is affecting at least one more workstation than the Mac on which it was originally reported. It's beginning to look as though the server might not be on-line. Now might be a good time to make that call to the server's administrator.

Sadly, the results of your phone call seem only to further muddy the waters. The server is running and on-line. Its name has not been changed, nor has it been moved into another zone. Moreover, the administrator informs you that there are several users who currently have various volumes on the server mounted and who don't seem to be experiencing anything unusual.

Two standard questions you might want to ask the user are when was the last time this procedure worked, and whether anything has changed since then. This is part of the general approach of problem isolation, an important concept in troubleshooting.

> *"Well, I used it last Friday, and it worked fine. I haven't done anything to my Mac since then."*

At this point, after a little head-scratching, you might decide that it would be interesting to have a look at one of the workstations that is capable of accessing the server. The administrator is kind enough to provide you with the name of one of these users, and you walk over to have a look.

The indicated Mac is clearly able to access the server; in fact, it has the server volume mounted as you inspect it. You can dismount and remount the volume with no apparent difficulties. What's more, Inter•Poll can see the server without any problems whatsoever.

A cursory inspection doesn't reveal any obvious differences between the Mac that can access the server and those that can't. The system software is the same version, and the general hardware configurations are comparable. You might have been hoping that there would be some clear difference that might give you a clue, but you're not going to be that lucky today.

So what might have changed between the last time the server was accessible from the problem Mac and now?

Let's step back for a moment and note that you have been proceeding so far by employing a mixture of random troubleshooting, as when you brought up the Chooser as a cross-check on the validity of the alias, and hunch-based troubleshooting, as when you checked on the status of the server with its administrator.

Stop Periodically to Examine Your Progress

You might productively spend a few moments now thinking about what kind of strategy you want to use to proceed. Let's recap what you've learned so far:

- The server can't be seen in the Chooser from at least two workstations on the network.
- The problem doesn't seem to be with the AppleShare workstation software, at least not in any generic sense. You can mount other servers and access them.
- Some Macs, on the other hand, can not only see the server that you are unable to locate, they can also access it and get useful work done.
- The results from Inter•Poll are consistent: it also can't see the server from either of the problem workstations. It does, predictably, show the server from the workstation that you know can access the server.
- The Macs that can see the server don't show any obvious differences in configuration compared to the Macs that can't locate it.

Now you can try to find any common factors between the Macs that can't access the server. Perhaps they are on the same network and no node on that network can access the network that the server is on.

A look at the documentation of your network layout is not encouraging on this score, either. The two problem Macs are not, in fact, on the same wire. One is on EtherTalk network number 12000; the other is on EtherTalk network 13000. The server, oddly enough, is also on network 13000. The Mac that was accessing the server is on EtherTalk network 14000.

So, in fact, you know quite a bit more about what's going on here than you did initially. The facts, as you currently understand them, are detailed in Figure 3-1.

At this point, the problem is fairly mystifying. How can it be that a workstation on the same network as the server fails to locate the server when a Mac operating across a router is able to access it without any apparent problems?

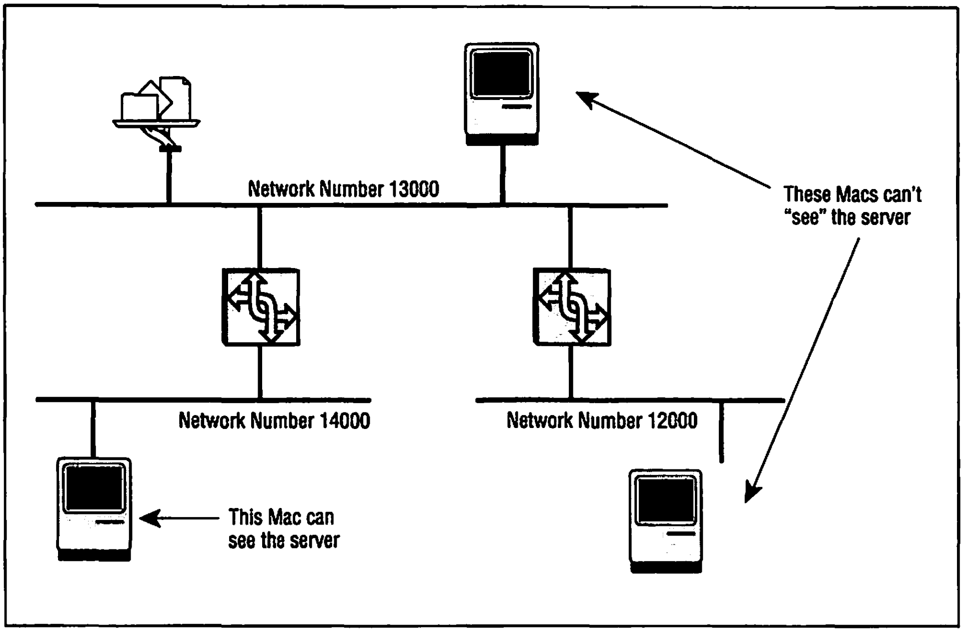

Network Number 13000
These Macs can't "see" the server
Network Number 14000
Network Number 12000
This Mac can see the server

Figure 3-1. The problem as you probably understand it now.

What is the best way to proceed? Random troubleshooting is almost certainly not going to be useful. You clearly have a fairly complex problem going on here, and the causes are far from obvious. You've pretty much run out of hunches at this point—unless you come up with further information, hunch-based troubleshooting is not likely to be very helpful.

Ask yourself just what, in simplest terms, is wrong here. A particular set of workstations is unable to mount a specific server. Another way of saying this is that something is breaking down in the process of mounting a server volume. Process-mode troubleshooting, a close examination of the sequence of events involved in locating and mounting the server, might well be in order here.

Before you proceed, you might collect a few more data points. At this point, you might ask whether the two Macs that cannot "see" the server can see one another. One of the easiest ways to check this (assuming that at least one of the Macs is running System 7 or later) is to turn on filesharing on one of the workstations and determine whether it can be seen by the others. You do this, and in fact, the one Mac can see the other.

Oddly enough, however, it cannot be seen by the Mac that was able to mount the server! Checking there, either with the Chooser or with Inter•Poll, produces the same result: no file sharing Mac is visible from this machine.

What have you accomplished so far? On the one hand, you've gathered a good deal of information about the nature of the problem, but only in external terms. You started off with information from the user about the nature of the problem from his point of view. You took a couple of stabs at random troubleshooting, using the Chooser and Inter•Poll. You also attempted some hunch-based troubleshooting by getting information on the status of the server from its administrator.

You next tried to see whether you could isolate the problem by inspecting the behavior of other workstations on the network and comparing their ability to locate and access the server with the problem workstation.

Admittedly, your results so far have been inconclusive. Clearly the problem you're dealing with here is neither peculiar to a specific workstation nor, apparently, to a particular network. Evidently, you're going to need to dig a little deeper into the nature of the difficulty, and to do that, you'll need to collect some tools. Experience suggests that our next step is to break out the protocol analyzer and examine in detail what is transpiring on the network. By comparing what you see on the wire with a "normal" case, you should be able to find some clues to help you better understand what is or isn't happening.

Let's put aside your examination of this problem for the moment, but you'll pick it up again in the second section of Chapter 9, "Troubleshooting a Router Problem."

Causes of Network Problems

We have said several times already that network problems can be difficult to resolve. As complicated as modern hardware and software are, networks are even more so. Even simple networks consist of many individual pieces: computers, printers, network interfaces, cables, operating systems, application programs, networking protocol processes, and many other entities. For the network to function correctly and usefully, all of these pieces must operate together smoothly and reliably. A breakdown, however small, can initiate a cascading series of effects, culminating in the user's call for help.

Any of the discrete elements in a network are capable of causing a variety of difficulties, some of them obvious and some less so. Any given symptom can be caused by a wide variety of underlying difficulties. For example, a missing zone display in the Chooser, as shown in Figure 3-2, can be caused by selecting the wrong link access protocol in the Network Control Panel, a router that has gone down, a cable that has become disconnected, or a number of other things.

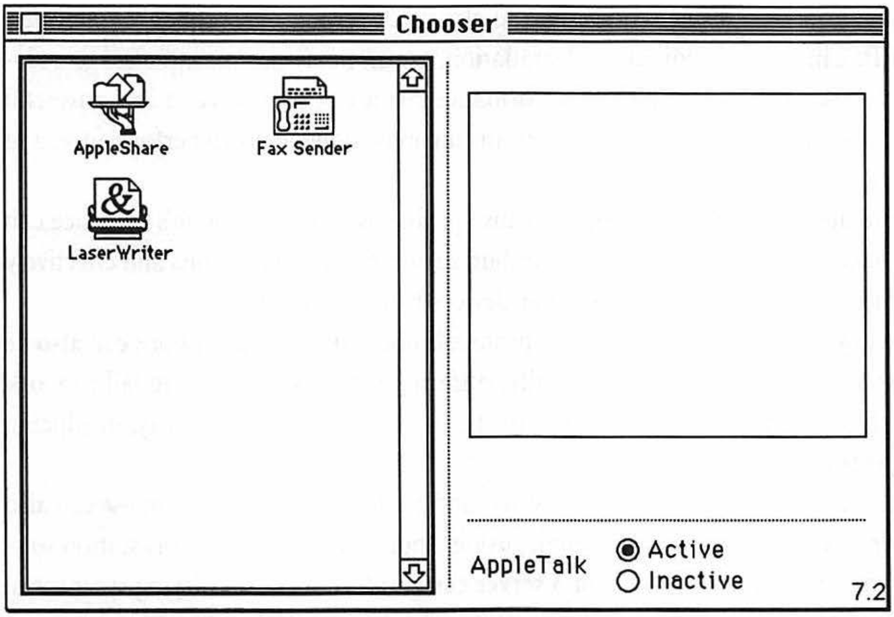


Figure 3-2. No zone display! What's the problem here? Several items might be at fault.

The physical layer is susceptible to a variety of difficulties. Cables can break or be of poor quality. Connectors can become disconnected, and hubs can be miswired. Cables at a punchdown block can be improperly connected and cords at a patch panel can be misrouted. Debugging problems on the physical layer can be approached either in setup mode, by checking the electrical continuity of the components, or in layer mode, by examining error statistics collected at the network hub or by performing tests using the AppleTalk Echo protocol.

Further, hardware elements that operate correctly individually can fail to work correctly in combination with one another. Every component operates within a range of tolerances. Two components that are operating close to the extremes of their tolerances can fall outside the range when used together. Equipment made by different manufacturers can also fail to work properly together.

Even if the underlying hardware is operating correctly, the network itself can create difficulties. The routers responsible for getting data from one piece of cable to another can be misconfigured, inhibiting the ability of various devices to communicate with one another.

Conditions on the network itself can also cause errors. Unreasonably high levels of traffic can cause performance degradation, as can many devices attached to a single network. A sufficiently high performance client can overpower a less powerful server, causing dropped packets and retransmissions, slowing down performance, and chewing up bandwidth.

The devices on the network can cause problems as well. A node's interface can start "jabbering," transmitting an unending stream of meaningless bits and effectively jamming the network, inhibiting other devices from transmitting.

Interactions between the various protocols operating on the network can also be the source of difficulties. Occasionally, differing versions of software fail to work together reliably. A protocol might also be used in a nonstandard way, producing unexpected results.

The operating system, utility software, and application programs you use can also be sources of difficulties. For example, using AppleShare version 2 workstation software with an AppleShare version 3 server can produce some mystifying symptoms. Software might not be designed to operate in the way you would like to use it: some programs are incapable of interacting across zone boundaries. Sometimes the only solution to a problem is to change your expectations or adjust the way you work.

So not only can individual elements of a network cause problems, but the interactions between the components can be sources of difficulty. On a network of any complexity, the number of potential interactions can rapidly become astronomical.

Clearly, it's completely impractical to approach troubleshooting with either a scattershot approach or a completely exhaustive one. To accomplish your work, you need theoretical knowledge, practical experience, pattern recognition, deductive reasoning, inductive reasoning, intuition, and luck.

Isolating the Problem

A manager, a programmer, and a technician are driving down the highway in their car when they suddenly become aware of a serious vibration coming from the vicinity of the right front fender and a general loss of control. They pull over and begin to discuss the situation. After some debate they come to a general agreement, based on prior experience, that the most likely cause of the observed symptoms is a flat tire. They then proceed to determine who among them is best equipped to address the problem. The manager immediately declares that as a manager, his best response is to delegate the authority to fix the car to the programmer and the technician. The programmer then turns to the technician and says, "Well, it looks like it's your responsibility. We're clearly talking about a hardware problem here. Where are you going to start?" After a little thought, the technician replies, "I guess I'll start by swapping around the tires to see which one's bad..."

Troubleshooting is often done in a high-pressure situation with people watching and impatiently waiting, with incomplete or inaccurate information, or sometimes with a lack of the proper tools and inadequate training. Also, computer networks are very complex and involve a great number of hardware and software components from different manufacturers. To make matters worse, these components change rapidly as new versions, products, network segments, users, and services are incorporated, sometimes without the troubleshooter's knowledge or consent.

In spite of the slippery texture of troubleshooting, a few solid principles apply. Let's explore these in terms of the classic troubleshooting model below:

Step 0 The system works normally.
Step 1 Something about the system changes.
Step 2 The change is noticed.
Step 3 The change is characterized in terms of a symptom.
Step 4 The symptoms are hypothesized to have a cause.
Step 5 A candidate cause is removed or altered.
Step 6 If the system does not return to normal, repeat steps 3, 4, and 5 as needed.

A couple of key concepts apply here. First, every effect has a specific cause or set of causes. People frequently talk about unreproducible errors, but in fact there are no such things. Some causes might be much more difficult to call up at any given moment, but computers are deterministic, not random. If a given problem situation can be reproduced identically, the problem will occur every time. Reproducing the required conditions, however, might be difficult or even impossible.

If you can reduce the problem situation to a minimum set of conditions, you stand a much greater likelihood of getting to the crux of the matter. The process of pruning away extraneous conditions is known as *problem isolation.*

The classical scientific method consists of taking a given situation, altering a single parameter, and observing how the total system changes as a result. Isolating a network problem is, in many senses, an analogous approach. After examining the situation, you might be able to hypothesize that a particular component is at fault. You can then test our theory by replacing the suspect component with one known to be operating properly and observe the effect.

Frequently, you might not have an immediate sense of where the cause of the problem is located. From the previous example, you know by talking to the user that at least a couple of specific devices are involved. You saw that at least two workstations are affected by the problem. At the point you left off, what you didn't know was what other devices might be affected as well.

In isolating a problem, it is useful to begin by looking at the internet as a whole and identifying which of the devices on the internet show similar symptoms. This is where an up-to-date and complete set of network documentation comes in handy. A map of our internet, detailing the devices and connections, can be a great help in determining where the problem lies.

For example, suppose a user is complaining of being unable to access a specific printer. You might begin by checking several other workstations on the same network segment to see whether any of them can access the printer in question.

If other nodes on the segment can access the printer successfully, that is a good indicator that the problem probably lies with the workstation on which the problem was originally reported. You can then proceed to examine the setup and configuration of that particular node to see whether you can come up with further . The difficulty here might well be a bad LocalTalk connector or a loose cable between the LocalTalk connector and the workstation.

If, on the other hand, none of the nodes on that segment can access the printer, you might want to look at nodes on another segment to see whether they can use the printer. If they can, you would suspect a problem affecting only the original network, such as a router that has gone down or a loose connector from the network to the router. If nodes on another network are also unable to access the printer, you might start wondering about the printer itself.

Now, suppose that some of the nodes on the original network can access the printer, while others cannot. This is the situation diagrammed in Figure 3-3. The Macs filled in with gray cannot print to the LaserWriter on the remote network, while the remaining Macs can access it without a problem. What do you suspect is going on here?

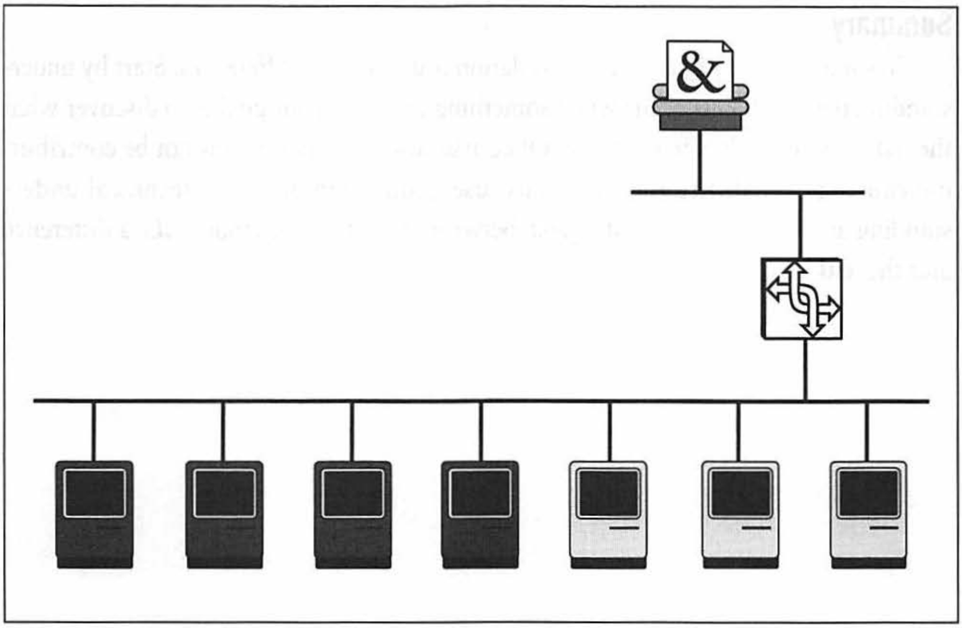


Figure 3-3. The Macs patterned in gray on the lower network segment can't access the LaserWriter on the upper network. The remaining Macs can use the printer normally.

The most likely possibility is a break in the network cable between the Macs that are working correctly and those that aren't.

The preceding examples show the standard pattern in problem isolation: examine the problem from several different angles, and determine the things that are common and those that are different. In most cases, this will give you some good ideas as to the origin of the difficulty.

This approach can be used within the context of a single workstation as well. For example, the Mac can't access the printer. Can it access a different printer? Can it access a file server? If the workstation can't print successfully to any printer on the internet but *can* access AppleShare servers, you might well suspect a corrupted printer driver. If, on the other hand, the workstation cannot access any services on the network at all, clearly you're dealing with a more global problem. You might check the LocalTalk connector for this Mac, or the wires connecting this Mac to the rest of the network at the punchdown block. You might, for that matter, bring up the Chooser to see whether AppleTalk is turned off!

Summary

In summary, the key to problem isolation is discovering differences. Start by understanding that problems occur when something changes; your goal is to discover what the nature of that change might be. Of course, some changes might not be contributing causes to the difficulties. You must use good common sense, technical understanding and experience to distinguish between the differences that *make* a difference and those that don't.

CHAPTER FOUR

Setup Troubleshooting

Use Setup Mode when you believe that one of the components along the network path is defective or improperly installed. The goal of Setup Mode is to identify that component and either repair or replace it. The beginning of this chapter discussed various types of network paths and their unique characteristics. The rest of this chapter is devoted to the types of problems characteristic of the many components that make up these paths.

LocalTalk is still a useful network in 1993—but no one knows how much longer this will be true. Many people say that LocalTalk is obsolete (or soon will be) because it doesn't provide the data transfer rate needed to support the types of networks that our computers and applications are (or soon will be) capable of running on.

But although this allegation might be correct, predictions of the death of a technology are notoriously unreliable. After all, thousands of people still make their living writing and maintaining programs in COBOL, a programming language whose imminent death was predicted before 1975.

LocalTalk is an extraordinarily inexpensive, efficient, and reliable network. Yes, it's slow, but it's faster than today's LaserWriters, faster than a remote terminal application, and faster than a networked modem—indeed, it's faster than most network transactions. LocalTalk's speed is the bottleneck for file transfer and for some kinds of databases, but that's about it. For all other applications, operation on higher speed networks like Ethernet occurs at the same speed as, or only slightly faster than, LocalTalk.

Network managers often can't justify the cost of making the transition to Ethernet based on the type of network computing their users typically perform and the performance gains they would get from Ethernet or Token-Ring. The assertion that LocalTalk is often as good as, or better than, a faster network technology like Ethernet will become less true as time passes. The speed of the networked components continues to increase (approximately doubling every two years), more and more services and resources are offered over the network, and our concepts about how to design and build distributed computing systems continues to evolve. But right now, LocalTalk is often a logical choice.

Troubleshooting Setup: General Information

Twisted-pair networks are very popular for a number of good reasons. Twisted-pair wire is inexpensive, easy to work with, and installed everywhere due to its use in telephony. Another result of its use in telephony is that many people have extensive experience designing, building, and maintaining twisted-pair wiring plants. Twisted pair is also very versatile and can be used for nearly all of the data communications systems that a company might employ, including LocalTalk, Ethernet, Token Ring, ARCnet, 3270 and other terminal lines, voice and modem lines, and so on.

Twisted-pair networks typically consist of an electronic hub that acts as a repeater. Incoming signals are processed—amplified and conditioned—by the hub and sent out to the other devices. In a network with a hub, devices are connected in a physical *star topology,* although the logical topology might be something other than a star. Token Ring networks (ring topology) and Ethernet networks (bus topology) are frequently wired in a physical star topology as well.

Between the device and the star are three kinds of components: wire, connection devices, and network adapters. There might be several types and changes of wire, including distribution wire, riser cables, patch cord, and cross-connect wire and several connection devices, including wall outlets, patch panels, punch-down blocks, and modular connectors, before the signal arrives at the network adapter.

Many problems in twisted pair networks result from the use of twisted pair wire installed for purposes less demanding than high-speed data networks. The wire quality might not be sufficient or the number of connections and wire changes between hub and device might simply be too great to reliably carry high-speed digital signals.

Besides the possibility of a single component or connection being faulty, some networks also have problems because they simply contain too many components and wire changes. Each of the components and connections in the network path has a small negative effect on the integrity of the signal.

Troubleshooting the setup of LocalTalk networks is in some ways easier than troubleshooting other networks. Less equipment is in the network path, and the electronics are less complicated than in most other systems. On the other hand, LocalTalk can also be difficult to troubleshoot because of two phenomena peculiar to LocalTalk networks: network managers routinely under-design LocalTalk networks, exceeding the manufacturer's guidelines and the boundaries of common sense; and users routinely modify LocalTalk networks without informing the network manager.

Both factors are much less evident in Ethernet and Token-Ring networks. Network managers generally don't under-design such networks because they know (or fear) that the technology is less forgiving than LocalTalk technology. Users modify the network less because they don't commonly have the knowledge and tools required to do it.

Since the nature of troubleshooting is to compare *what is* to *what should be*, the LocalTalk phenomena of under-design and user alteration can make diagnosing LocalTalk problems difficult. Ideally, *what should be* is a healthy, well-constructed network. A good troubleshooter has well-developed expectations for how the network should behave under all kinds of conditions. When the behavior of the network varies from *what should be*, you must be able to explicitly and quantitatively define the way in which it is different.

When a network is under-designed, however, *what should be* is already less than adequate. And when a user modifies the network, *what should be* changes; it might no longer correspond to the network manager's expectations. Getting rid of these two impediments to network health—under-design and user modification—is key to both solving many organization's network woes and making troubleshooting easier and faster.

The relationship between design and troubleshooting

The reason network managers underdesign is usually either due to economics or the inertia of tradition: That's the way we've always done it. Sometimes, of course, network managers can take advantage of LocalTalk's forgiving nature and underdesign their network without compromising reliability. For most networks, however, this decision eventually leads to a troubleshooting nightmare. As the network matures, more users are added, users are moved, the mix of services changes, and the principles of underdesign become entrenched in the minds of the network staff.

In any network, the frequency, severity, and disruption of troubleshooting helps you assess the validity of a network's design. A good network designer will create and observe a *design standard* that specifies the ways in which the network can be built. The design standard is an important document, even if it specifies a network that exceeds the manufacturers' guidelines.

At the same time you create a design standard that sets limits for acceptable network construction, you should also set limits for an acceptable level of troubleshooting.

While your design standard will specify the number of feet and the number of branches and the number of nodes per port, your troubleshooting standard will set limits on the frequency, severity, and duration of your troubleshooting adventures.

For example, you might decide that troubleshooting any subsystem of a network more than once a month is excessive. You might also specify other limits, such as a maximum allowable downtime of 10 minutes per troubleshooting call or a maximum 30 minutes per month. Set your troubleshooting limits to whatever you feel is reasonable and appropriate. When you set a troubleshooting limit, you create the criteria by which to judge the validity of your network's design standard. You create a yardstick to measure whether or not your design is fulfilling your objectives.

For example, in a LocalTalk network using active star topology, you might decide to set a maximum of four branches per port, with a maximum of four users per branch. While this specification isn't strictly beyond the manufacturers' guidelines, we don't consider it a wise design standard because there are too many opportunities for trouble. Also, the looseness of this specification makes troubleshooting more difficult; there are simply too many variables for easy troubleshooting. But suppose you choose this specification as your design standard. If you're troubleshooting only once every two months and typically have the problem corrected in less than 20 minutes, the situation falls within the limits previously set and network design can be considered acceptable.

If your design standard fails the test, however, you'll know that you need to cut back the limits of your network design—perhaps to three branches with three users, or two branches with two maximum users—until you fall within the guidelines you set. By establishing the guidelines independently of the design, you automatically create the justification for change; it's not guesswork anymore.

(*A terminology note*: Two categories of LocalTalk connectors are in use today. Apple's LocalTalk connector is the only example in the grounded LocalTalk connector category. All other manufacturers use ungrounded circuits, for which Farallon now holds a patent.)

More network managers use connectors based on Farallon's circuitry rather than Apple's LocalTalk connectors and cabling, using shielded twisted pair. Ungrounded connectors have several advantages, including lower cost and a wider range of configuration options. The rest of this section concerns Farallon's ungrounded connectors, which we'll refer to simply as LocalTalk connectors. We'll refer to Apple's connectors as Apple's LocalTalk connectors.

Because LocalTalk wiring is so simple, users sometimes modify the wiring of the network without notifying the network manager. They might extend a stub with extra line cord, put a splitter in the line, and make a passive star at the wall outlet. Or worse, they might create a loop by connecting together two Macs on different ports of the hub. In rare cases, they might even sneak into the wiring closet and modify the wiring by adding new branches. Because users often only partially understand what they're doing, this sort of alteration creates a great risk.

Of course, the network manager might also make a mistake during installation. The following kinds of problems can result from improper network modifications:

- Added cable that's longer than allowable
- Reflections created by mismatched cabling
- A change in the network termination
- The creation of a remote star on a port that already has more than one branch
- The addition of a device that's already running, causing a duplicate node address
- The addition of a defective, "jabbering" device
- A loop in the wiring between devices in the same network
- The addition of a router that's improperly configured
- A loop in the wiring between devices in different networks

Conceptualizing the LocalTalk Setup

Setup troubleshooting involves verifying the continuity and quality of the connection components. Figure 4-1 shows the LocalTalk components for a typical active star topology.

The following sections will trace the segment of the connection path that's external to the computing device, starting with a discussion of the LocalTalk connector and ending with a discussion of the LocalTalk hub. The material in these sections accompanies material in Chapter 6, Physical Layer Troubleshooting 102, which deals with this external path as a single entity. Here the emphasis is on how the components work, both individually and together.

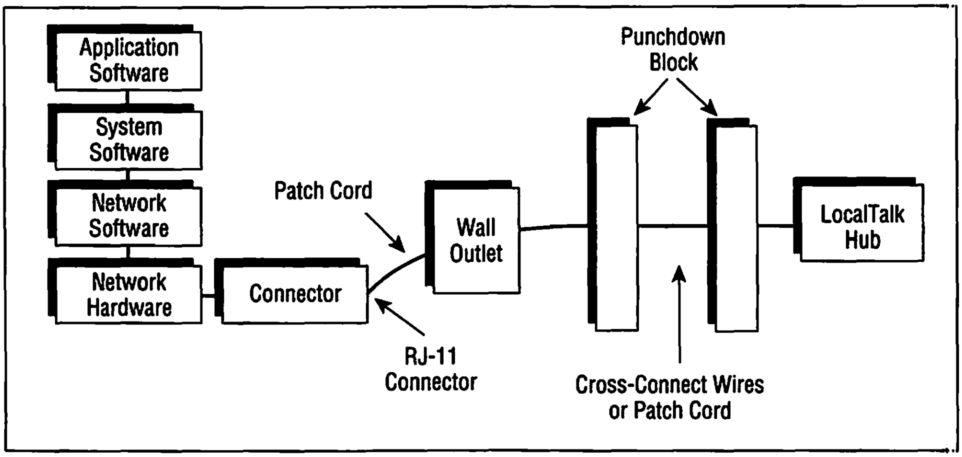


Figure 4-1. The setup for hub networks is shown in schematic form. Each component is discussed in greatei detail in this chapter.

After tracing the external connection route, we'll return to the LocalTalk connector and trace the internal connection path inside the computing device.

The LocalTalk connector

A critical element in all LocalTalk connections is the LocalTalk connector itself. The connector transforms the Mac's DC signals into the LocalTalk network signal. The two most common problems with these connectors are that they're faulty or have become damaged.

There are many suppliers of LocalTalk connectors. Some network managers (or their non-technical purchasing departments) buy from the smaller manufacturers because they can save a few dollars per connector. This is frequently a bad decision; most of these connectors are inferior. Here are some thoughts about what to look for to determine if a particular connector is of high enough quality.

First, one LocalTalk connector quality problem involves internal faults resulting from manufacturing errors or from bad components. For all manufacturers, the number of connectors that are faulty upon delivery naturally fluctuates as changes are made in manufacturing processes, personnel, tooling, inspection techniques, component suppliers, or subcontractors. Of the major manufacturers, Farallon has consistently had one of the lowest rejection rates. Based on our own experience and that of many others, we estimate the rejection rate at less than one bad connector in 250. Almost 100 percent of faulty connectors are discovered during their initial installation; however, connectors rarely become faulty during service, unless they're damaged. Because of the chance of faulty connectors, it's always a good idea to have a few extra connectors on hand for spares.

Some connectors are also poorly designed. The main component in a LocalTalk connector is the transformer, which must faithfully reproduce the computing device's DC voltage impulses on the network wire in a crisp, square wave. Some manufacturer's transformers are better than others, and the best ones are used in the Farallon PhoneNET connectors and the Focus (formerly NuvoTech) TurboNET connectors. You can tell whether or not your connector has a good transformer by observing the wave it produces in an oscilloscope. Crisp, sharp edges on the corners of the square wave on the network side of the connector show a good transformer. Inferior transformers produce inferior signals. Remember that the signal will decay as it travels over the wire, so you should start out with the best signal possible.

Good connectors also have a variable resistor that acts as a surge protector. Because the patch cord used in the LocalTalk connector looks much like a telephone cord, users might unsuspectingly plug their LocalTalk connectors (and therefore their computer's motherboard) into a live telephone circuit. The voltage that drives the ringing signal of a telephone is typically 90 volts—high enough to do serious damage to a computer's motherboard. A good surge protector will absorb the ringing voltage so that it doesn't damage the motherboard.

The final component of the connector is a large resistor. It allows the static charges that naturally build up on the other components to dissipate to ground (even though the ground potential isn't used as a voltage reference for the signal).

If you're thinking of buying a connector other than the Farallon PhoneNET connector or the TurboNET, check the quality of the connector by making sure the following factors are true:

1. The square wave it produces is of high quality
2. The circuit board is masked
3. The solder joints are small, neat, and symmetrical
4. The variable resistors and the fixed resistors (two of each) are also present on the circuit board

If you don't know how to perform these tests, ask the manufacturer to confirm the quality of the product's design and manufacturing. If you plan to buy a large number of connectors, a prospective supplier should also be able to give you a photograph of an oscilloscope trace showing the quality of its connector's signal.

Also, we strongly recommend against the use of the TurboNET ST connector. This is a "self-terminating" connector with LEDs that blink when there is network traffic. These connectors have a clear case to allow the viewing of LEDs in the circuit that light up when there is traffic. Instead of using conventional termination, the ST provides a kind of termination by *dampening* the signal, or reducing its amplitude, as it passes through the connector.

While the signal dampening is equivalent to conventional termination in many typical LocalTalk configurations, it doesn't work well in other configurations. Since the purpose of a self-terminating connector is to free the network manager from the worry of termination, and since the ST can't effectively eliminate that worry because it doesn't work in all configurations, we don't consider it a worthwhile alternative to the standard connectors. The regular TurboNET connector, however, is on a par with the PhoneNET connector.

LocalTalk connectors can also become damaged in service. The most vulnerable parts of the connector are its external electrical contact points, which are located both in the DIN-8 (or DB-9) plug that connects to the computing device and in the RJ-11 jack that receives the patch cord. In both the plug and the RJ-11 jack are contact pins that can become bent or broken through repeated rough use. We find that a large pair of tweezers is a good tool for straightening out the contacts in the LocalTalk connector. In the RJ-11 connector, it's usually best to simply clip off the connector and install a new one.

The pins can also become corroded. Because the contact pins are treated to resist corrosion in normal environments, corrosion generally happens only in unusually aggressive environments. Consider corrosion a possibility primarily in networks located *very* close to the ocean or in the vicinity of harsh industrial chemicals. If the pins have become corroded, there's nothing you can do except replace the connector.

A more common occurrence is that the contacts become dirty. Usually you can clean them with a good strong puff of breath or with a shot from one of those compressed gas canisters used by electronics hobbyists.

Even when the connector hasn't been damaged, network problems can be caused by an ill-fitting contact, whether in the plug-in to the computing device or in the RJ-11 jack. Be sure that both connections are firmly and squarely in place. If you find no damage but still believe the problem is a faulty LocalTalk connector, try swapping connectors with a node that's working.

In rare cases, the connector will stop working for no apparent reason. Most reputable manufacturers will replace connectors that you think are defective or have stopped working. They want to diagnose the connector's health as part of their quality control plan.

Problems with the patch cord

In addition to ensuring that the RJ-11 connectors on the patch cord fit snugly into their jacks, check that the components of the patch cord are the right kind and have good physical integrity. A LocalTalk patch cord consists of only three components: two RJ-11 plugs at either end and the patch cord itself. In all configurations except the backbone, the wire can be either twisted or untwisted wire; untwisted wire (flat cable) is more common. In the backbone configuration, the patch cord must be *un*twisted. Patch cord is also known as line cord, silver-satin, tinsel, and modular extension cable.

The cabling and connectors used in LocalTalk's patch cord wire are identical to a common telephone patch cord (the cord that runs between a telephone handset and a wall outlet). Therefore, premade cables are readily available from a variety of sources, as are the components and tools for making one's own patch cord from bulk supplies. Some of these cables, components, and tools are of much higher quality than others. Like the LocalTalk connectors, patch cords cause problems when they're faulty or damaged.

When checking for faulty cables, make sure the patch cord has four conductors. The four conductors are typically colored black, red, green, and yellow. Some patch cord with only two conductors is designed strictly for use with telephones. Two-conductor patch cord is sometimes surrounded by a clear jacket so no one will mistake it for four-conductor patch cord. A normal single-line telephone like the one in your home typically uses only the two conductors in the center, usually colored green and red. With an RJ-11 connection, LocalTalk always uses the outer two conductors, usually colored black and yellow.

Because patch cord isn't expensive, some network managers are careless when making it, or they make it from unusual materials. We've seen patch cord made by cutting off the end of an RJ-45 patch or using another equally inappropriate cable and force-fitting an RJ-11 plug onto the wire.

Equally foolish is the practice of using solid conductor wire with RJ-11 crimping components designed for stranded wire stock. Almost all patch cord is made from stranded wire. Such irregular practices usually result in bad patch cord due to poorly made crimps.

LocalTalk uses a signal that isn't polarity sensitive. This means that you don't have to worry about which way the yellow and black conductors are oriented in the plug as long as all four conductors are present and the two outer pins in the RJ-11 plugs at either end are connected all the way through the patch cord. This is especially important to check if the patch cord is from twisted-pair stock; it's much more difficult to line up the right conductors with the right pins using twisted-pair wire. Be sure that the same pair of conductors is connected to the outer pins of the RJ-11 plug on both ends of the patch cord.

Problems with the RJ-11 connector

One of the best features of patch cord is that it's easy to crimp the RJ-11 plug onto the end of the cord with an inexpensive tool. Many network managers buy their own components and make their own patch cord during the installation process to the length required for the location.

While it may be tempting to buy the cheapest components and tools, the money you'll lose to downtime and troubleshooting will quickly dwarf the small amount of money saved. To make sure that you're getting good components, buy them from a reputable telephone equipment dealer where professional telephone technicians buy their gear, such as Anixter Bros. or Graybar; don't shop at a neighborhood electronics hobby shop or a general-purpose hardware store.

Whether you crimp your own RJ-11 plugs or use pre-made cables, check the quality of the crimp to make sure the connection is solid. Here are the points to look for (the following numbers correspond to the numbers shown in Figure 4-2):

1. Make sure the portion of cord that isn't stripped is pushed all the way into the plug's chamber. You shouldn't see any of the colored conductors outside the RJ-11 plug.

2. Make sure that the stripped conductors completely fill up their chambers, going all the way to the end of the plug. The conductors in the outside positions should be black and yellow.

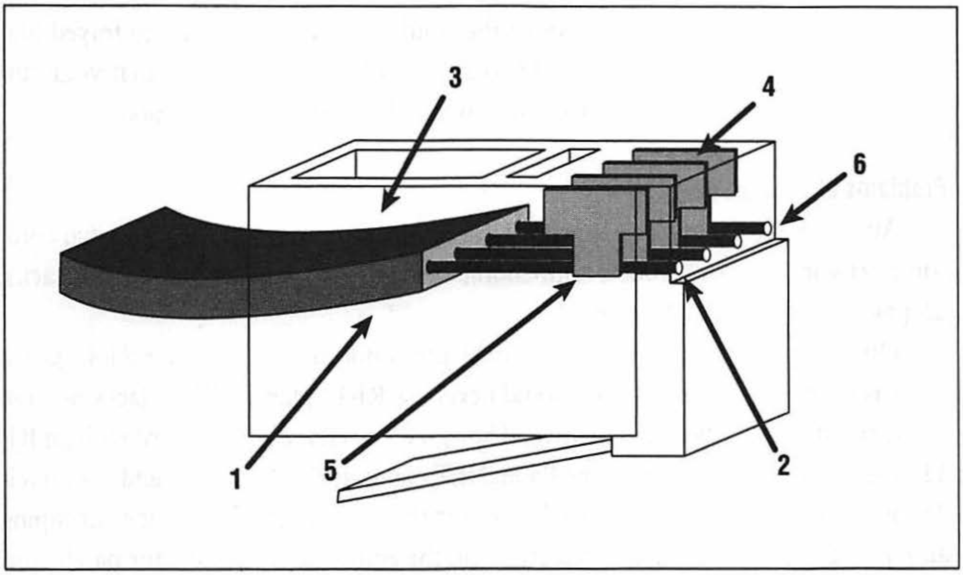


Figure 4-2. Check the quality of the RJ-11 crimp.

3. The crimping action should disturb the portion of cord inside the plug enough to provide strain relief against sudden jerks during service. Look through the clear plastic of the plug and make sure that the cord inside the plug at (1) is visibly bent or mechanically disturbed; give a firm tug on the cord to make sure it's wedged in place and won't slip out.

4. Make sure that the contact edges of the four metal plates are seated evenly and deeply in their slots. Before the crimp is made, these pins should be riding high above their slots; the crimp pushes them down and forces the sharpened edge on the other end of the metal plate through the jacket of the stripped conductors so that the metal plate can nestle among the filaments of the individual conductors. If the exposed contact edges are not deeply and evenly seated, you either have a poor-quality crimping tool or its die is out of alignment.

5. A good firm crimp pushes the knife edge of the plates into the conductors and slightly beyond. Through the plastic, you should be able to see the tips of these knife edges protruding slightly through the conductors.

6. Make sure that the ends of the conductors are neat, with no frayed filament ends sticking out. Frayed ends, which usually mean that your cutters are dull, can cause short circuits between the conductors.

Problems at the wall outlet

Although the LocalTalk connectors are designed to work with RJ-11 patch cord, some networks use wall outlets with another style of jack, such as IBM cabling jacks, 25-pair line taps, or RJ-45 plugs.

Opinion is divided about how to make the connection when using RJ-45 jacks. Some network managers say you should never use RJ-11 plugs in RJ-45 jacks because of the size difference between the two. They prefer to create a patch cord with an RJ-11 plug at one end to plug into the PhoneNET connector's RJ-11 jack, and use an RJ-45 plug at the other end of the patch cord for the wall outlet's RJ-45 jack. Crimping an RJ-45 plug, with its eight conductors, on the end of a four-conductor patch cord can be a tricky operation. You must make sure that the four stripped conductors go into their proper positions in the plug. If you decide to do this, carefully check each connector after you crimp it.

Other network managers claim that RJ-11 plugs work reliably in RJ-45 jacks because the size difference is in the width of the connector only, not in its height or depth. Their reasoning says that since the RJ-11 plug is narrower than the RJ-45 jack and since the plug is automatically centered by its locking tab, there's no problem whatsoever with using an RJ-11 plug in an RJ-45 jack.

We've found small variances in the dimensions of jacks and plugs from different manufacturers. The small size differences can sometimes affect the reliability of the mismatched connectors. Usually, you can get a feel for the reliability of the connections when you plug the patch cord into the jack. If the fit feels snug and secure and you hear a satisfying snap when the retaining tab locks into place, you probably have a good connection. If you feel unsure of the connection, you can perform a simple continuity check with an ohm-meter, a technique described in Chapter 6, Physical Layer Troubleshooting. If your connection doesn't pass the ohm-meter test, you should consider the trickier procedure of crimping an RJ-45 plug onto the patch cord.

With IBM cabling, you must use a balun that converts the jack type of the outlet from the IBM Cable Tap (commonly called a "Boy George connector" because of the lack of distinction between plug and jack) to an RJ-11. Besides converting connector styles, baluns sometimes provide impedance matching as well. For example, if your

LocalTalk distribution cables are shielded (as some types of IBM Cabling are) and your patch cord is unshielded, there's an impedance mismatch at the point where they join together. The balun is constructed in a way that minimizes the effect of this mismatch.

However, if you're using shielded wiring, even with a good balun the impedance mismatch at the IBM Cable Tap can cause a significant reflection of the network signal. It's therefore wise to wire your network very conservatively. We recommend one run of wire per port of the wiring hub only, with no passive star wiring. Also, you shouldn't mix ungrounded LocalTalk connectors with grounded LocalTalk connectors if you're using shielded twisted-pair for your distribution cables. Use ungrounded LocalTalk connectors exclusively.

With 25-pair line taps, most troubleshooting problems result from simple installation mistakes, such as wiring the wrong pair of the 25-pair cable to the line tap's jack. Another mistake is to wire the right pair of the 25-pair distribution cable, but to the wrong pair of pins on the wires. Because of the potential for mistakes, you should individually check each connection to 25-pair line taps after installation or following wiring changes, with either a tone test or an ohm-meter, both described in Chapter 6.

Like the pins in the LocalTalk connector jack, the pin contacts in the wall jack can become dirty or corroded over time. You should check them, also, in the way described earlier.

Problems with the distribution cabling

There are two network problems associated with distribution cabling, although they aren't common in LocalTalk. The first is when the cabling is not of high enough quality for the signal. The second is when the distribution cabling must pass through an overabundance of connections—intermediate punch-down blocks, patch panels, connecting blocks, changes in wire type, and baluns—on its way from the LocalTalk hub to the wall outlet.

These problems are rare in LocalTalk because the slow speed of the LocalTalk signal isn't conducive to the types of problems that occur. As mentioned in the section on Signaling and Transmission in Chapter 2, higher-speed signals require higher-quality transmission media and are therefore more prone to the problems of crosstalk, reflections, and frequency distortion.

Although LocalTalk's slowness makes it naturally resistant to these types of problems, they can occur if the wiring conditions are unusually bad. This is sometimes the case in buildings built before 1955 or in buildings built or wired by an untrained or otherwise incompetent wiring technician.

Low-quality distribution cabling

One thing to look out for is the use of voice-grade cabling—that is, cabling with a low number of twists per foot and poor electrical properties. In LocalTalk, voice-grade cabling is usually a problem only with long runs of cable with 25 or more pairs in a single sheath when multiple pairs within the cable are carrying LocalTalk signals.

An easy way to check your cabling is to slice it open and count or estimate the number of twists per linear foot. Each pair of conductors in the cable should be wrapped around each other in a helix. LocalTalk-quality 4-pair cabling will have more than five twists per linear foot; 25-pair cabling of quality high enough for LocalTalk will have more than three twists per linear foot.

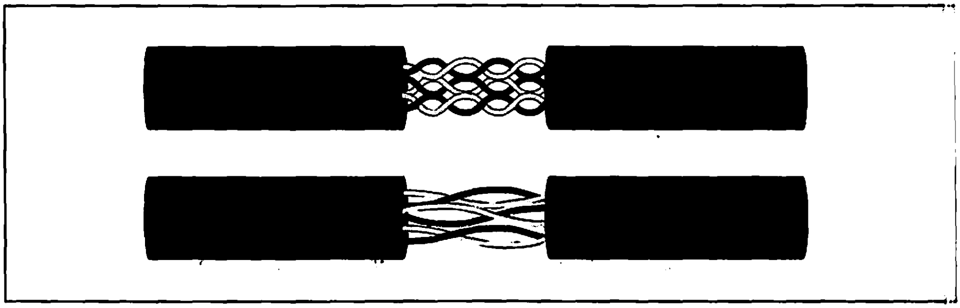


Figure 4-3. Data-grade cabling has a higher number of twists per foot than voice-grade cabling. Also, the number of twists per foot within a grade usually declines as the number of pairs in the cable increases.

A more reliable method for determining whether your cabling is suitable for LocalTalk (or any other data application) is to check the manufacturer's guidelines. Manufacturers often state in their literature what types of applications are suitable for which kinds of cable. Although LocalTalk isn't usually mentioned, most unshielded twisted-pair cable listed as suitable for RS-232, 1BASE5, Digital Telephone, or IBM 3270 (or faster signals such as Ethernet or Token-Ring) is suitable for LocalTalk. Each wiring manufacturer has its own classification system, but the manufacturers' literature might reference other standard classification systems. These systems might be slightly confusing because of the many aspects of cabling that they classify, but you can sort out the classifications according to the term used to describe the cabling.

Class. Refers to a cable's conformance to the National Electric Code, which is primarily concerned with fire safety. All of the codes for analog phone cables are of the format CL2_; data cables (low voltage) are specified by CM_. The suffix on the end of codes indicates the intended application and electrical code requirements. They include the following:

Table 4-1. Suffix Types

(No suffix)	General use	Not fire-retardant. Must be enclosed in conduit in air plenums or vertical shafts.
X	Limited use	Open work areas—10-foot length maximum.
R	Riser cable	Fire retardant jacket. Must be enclosed in conduit in air plenums or vertical shafts because it produces noxious fumes when burned.
P	Plenum cable	Low-flame, low-fume jacket. Conduit not required.

According to this system, then, cable of the class CMR is data cable with a fire-retardant jacket that produces noxious fumes and must be placed in conduit of non-flammable wireways when run through air plenums in vertical shafts such as elevators.

Type. Refers to the cable's physical characteristics and electrical properties. The IBM Cabling System originated this classification system in 1984. Many manufacturers specify their cabling as compatible with a specific IBM cable type. Any cabling specified as compatible with IBM Type 3 is suitable for LocalTalk. IBM Type 3 cabling refers to two-, three-, or four-pair unshielded twisted-pair wire with a DC resistance of less than 28.6 ohms/1,000 feet, a characteristic impedance in the range of 84–113 ohms (measured at 1 MHz), and a maximum signal attenuation of less than 8.0 dB in 1,000 feet (also with a 1 MHz signal). This is more than adequate for LocalTalk. IBM Type 2 cabling has both shielded and unshielded twisted pairs in the same jacket. While either the unshielded or shielded pairs are suitable for LocalTalk networks, the unshielded pair is the better choice.

You can also use Type 1 cabling (two shielded twisted pairs). When using shielded cabling, be sure to use the proper balun to minimize the harmful effect of the impedance discontinuity at the wall outlet. Also, the network distance limitations for shielded cables are approximately one-third of the distance limits for unshielded wire at LocalTalk speeds.

Grade or *Level.* Refers to the cable's transmission properties only, and is sometimes referred to as a cable's *implementation number*. All three of these terms—grade, level, and implementation number—refer to nearly identical systems; the term used depends on the affiliation of the speaker. For example, Anixter employees always use the term Level.

There is a formal system and an informal system for specifying grades. The formal system specifies seven grades. Grades 1 through 5 indicate progressively higher quality (supporting faster data rates) twisted-pair wire. Grade 6 specifies coaxial cabling for both thin and thick Ethernet, and Grade 7 is used for optical cable that can support data rates of up to 100 GB/sec (gigabits per second). Some requirements for the five twisted-pair grades are shown in Table 4-2:

Table 4-2. Informal System for Specifying Cabling Grades

Grade	Maximum Impedance	Typical applications
Grade 1 (Unshielded)	None	Analog telephone, RS-232
Grade 2 (Unshielded)	8 dB/1000' @ 1 MHz	Digital telephone, LocalTalk IBM 3270, 1BASE5
Grade 3 (Unshielded)	30 dB/1000' @ 10 MHz	T1, 10BASE-T, ISDN, ARCnet
Grade 4 (Shielded) - & -	7.5 dB/1000' @ 1 MHz 25 dB/1000' @ 10 MHz	≤ 100 MB/sec. applications including 10BASE-T, 16 MB Token Ring
Grade 4 (Unshielded)	4.9 dB/1000' @ 1 MHz 16 dB/1000' @ 10 MHz	≤ 40 MB/sec. applications including 10BASE-T, 16 MB Token Ring
Grade 5 (Shielded)	10.6 dB/1000' @ 10 MHz 30 dB/1000' @ 100 MHz	≤ 100 MB/sec. applications

Note: IBM Type 3 Cabling has electrical qualities equivalent to Grade 3.

The informal system refers to cabling as either *voice grade* or *data grade*. In general, Grade 3 and higher are always referred to as data grade and Grade 1 is always considered voice grade. Grade 2 is called either voice grade or data grade depending on the speaker's background. Grade 2 or higher is always adequate for LocalTalk, and Grade 1 is usually adequate. If you have Grade 1 wire (or voice grade or ungraded wire) you can check its quality by counting the twists per foot, as described earlier.

Too many wiring changes and connections

Sometimes many small factors can gang up to become one large problem. Figure 4-4 shows a LocalTalk setup of marginal design. While you wouldn't want to design a LocalTalk network this way, many LocalTalk networks are built like this because their network manager had no other choices.

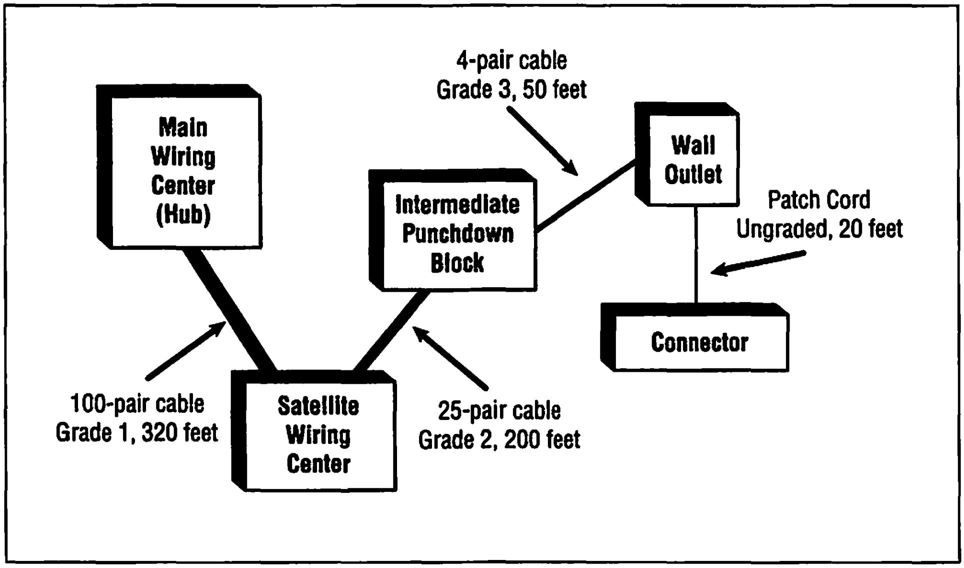


Figure 4-4. While this network is within all of the LocalTalk hub manufacturer's guidelines, it's of marginal design.

Several factors could cause problems in this design. The first is the 100-pair cable that connects the hub with the satellite wiring center. While voice grade cabling is not a problem in and of itself, this particular cable exceeds the recommended maximum of 25 pairs for its grade. Crosstalk on this particular link could be a problem, particularly if it runs through electrically "noisy" areas (that is, near fluorescent lights, copying machines, motors and so forth).

LocalTalk hubs and the signals they send can be affected by certain kinds of electrical noise, sometimes because the signals travel through the voice grade wire already installed in the building. In some cases, crosstalk will occur between LocalTalk signals and digital phone circuits (64 Kb) traveling in the same sheath of voice grade wire, although this is generally more disruptive to the digital phone signals than to LocalTalk signals. Analog phone circuits don't exhibit this problem.

Multiple LocalTalk signals shouldn't be sent over long distances (more than 100 feet) within the same voice grade cable because the signals can crosstalk with each other.

Be careful when using a hub from one manufacturer with a patch panel from another. Different manufacturers have different numbers of ports, and they assign them to different conductors. An inactive pair on one patch panel might be active on another. Worse, two ports might be wired into the same connection jack. The installer might unwittingly create loops or other types of connection problems because of the differences.

Most LocalTalk hubs tolerate wavering electrical power conditions fairly well. Routers usually exhibit problems as a result of bad power before LocalTalk hubs do. If you suspect that you have bad line power (surges, brownouts, voltage variations, and so forth), you might want to rent a recording power monitor. The electric company, even if presented with documentation of a power problem, might not move very quickly to remedy the situation. Still, the monitoring equipment will let you either rule out or identify bad power as a cause of your network problems.

If it turns out that poor-quality power is causing a problem, some uninterruptible power supplies also act as power conditioners. These are relatively expensive, though, so you'll probably want to be sure the power is actually the cause before you purchase one of these devices.

Sometimes network designers under-design and over-install the network to save money. In these networks, the hub might not work properly because of too much wire, too many devices, or too many branches per port. Your design standard must place a reasonable limit on the maximum number of branches per port and the maximum number of devices per port. Well-designed LocalTalk networks rarely have problems that require troubleshooting.

Typically, a design will have a target configuration and a maximum allowable configuration. Here are some recommendations for the target (to be balanced against economy):

Best:	1 port = 1 branch with 1 device
Good:	1 port = 1 branch with 2 or 3 devices
Fair:	1 port = 2 or 3 branches with 1 device per branch
Poor:	1 port = multiple branches with multiple devices per branch

Remember, network downtime and the time spent troubleshooting quickly spends the money saved by skimping on equipment.

The patch panel

Punch-down blocks and patch panels are simple devices that allow easy connection between two wires. Distribution wires are normally run between a wall outlet at the user location and one of these two kinds of devices located in a wiring closet or data closet. Punch-down blocks are usually preferred for telephones and are frequently used for data connections as well. The use of patch panels for data connections has grown in popularity as computer technicians without extensive telecommunications experience are made responsible for maintenance of the wiring plant.

Punch-down blocks, also known as type 66 blocks, usually contain 50 rows of four pins each. The first two contacts in each row are electrically connected to one another, as are the second two. A wire is attached to a contact by placing it into the retaining clip on the contact and then using a special device called a *punch-down tool* to force the wire into the contact.

In contrast, a patch panel uses RJ-11 jacks, typically in groups of four, which are electrically connected to one another. Groups of jacks can be connected together by running a patch cable between them.

When you use a punch-down block or a patch panel to wire together a LocalTalk network's transmit lines and receive lines, you create a *passive star*. Passive stars are simple to set up, but they can handle only a small number of nodes. The reason for this is that the strength of the signal becomes divided among the various wires at the point of connection. If the signal strength is reduced enough, it can no longer be reliably distinguished by the LocalTalk hardware.

The electrical connection made by either of these devices is mechanical in nature—the wires to be connected are each held in physical contact with a third component that is electrically conductive—and is therefore subject to the mechanical problems of wear, misalignment, or just plain bad installation practices. Also, you might find such irregularities as distribution cables connected to different wall outlets than their identifications indicate.

The most common problem with punch-down blocks comes from the fact that the tines of the connection pin get spread apart after frequent use. If they are worn, they might not grip the wire hard enough to strip the insulation back during the punching action. Even if properly installed, worn pins might not hold the wire firmly enough to suffer routine mechanical disturbances during use.

Patch panels are mechanically quite reliable. The sort of problems seen here are typically things like poor quality patch cables, dirty connections, or, most frequently, misconnected cables. The latter, fortunately, is easy to resolve once it has been identified.

The cross-connect wiring

Connections between the pins of one punch-down block and another are called cross-connects or jumpers. If these connections are poorly made or become damaged, the nodes whose signals are transmitted through these rows will be unable to access the network. On a patch panel, short patch cords are used to connect together groups of RJ-11 jacks or RJ-45 jacks. These patch cords are functionally identical to the cables running from the LocalTalk connector to the wall outlet.

The LocalTalk hub

Some networks still use either the traditional bus or a passive star, where a small number of wires are physically joined in a central location, but these can have reliability problems if they support more than a few nodes. To address this problem, most LocalTalk networks use an active star configuration, with a wiring hub like Farallon's StarController, NuvoTech's TurboStar, or Tribe's LocalSwitch at the center of the network. The function of the hub is largely to amplify incoming signals before transmitting them to the other connected nodes. Some network managers put off buying a hub at the expense of reliability. While the troubleshooting tips in this section might be helpful, these managers really need to get a hub. If you think you might need a hub, here's a test:

A. You're not using a LocalTalk hub

AND

B. you have more than eight computing devices

OR

you have more than three locations in your building to network

AND

C. you troubleshoot your LocalTalk network more than once a month

If A, B, and C are all true, you need a hub. It should solve almost all of your reliability problems.

While LocalTalk doesn't require a hub, most LocalTalk network managers with more than eight devices find it desirable to include a hub in their network design. Most of the current crop of hubs are multiport repeaters. These repeaters have 8, 12, 16, or 24 ports that are physically isolated from each other but electrically connected through the hub's internal amplifier. One hub, the Tribe LocalSwitch, is a 16-port switching hub.

Each available LocalTalk hub has distinct advantages. The StarController has the most reliable signal processing, the TurboStar has slightly more useful management features, and the LocalSwitch lets you attach many more users to a single LocalTalk network without routers.

Hubs divide the LocalTalk bus into segments and repeat or switch incoming signals. Besides simple amplification, some hubs process the signal in an attempt to restore it in some way. This might include regenerating the signal (Tribe LocalSwitch), retiming the signal (PhoneNET Repeater), or filtering out errors caused by distortion and reflections (Farallon StarController, NuvoTech TurboStar). Some hubs (StarController, TurboStar) have the ability to detect "jamming" devices (devices that are continuously sending a signal) on a port and automatically deactivate that port.

Hubs and repeaters are usually used to enhance a network's design qualities. These qualities include:

- *Reach.* By amplifying the signal, hubs extend the distance over which the signal can be sent.

- *Reliability.* By distributing network devices among the ports of a hub, each device will typically have less wire associated with it. The chance of signaling error increases with distance.

- *Fault tolerance.* Ports are physically isolated from each other; a problem on one port usually doesn't affect devices on other ports.

- *Flexibility.* When a user moves, the network manager quickly changes a connection in the wiring closet, and the move is accomplished with no network downtime.

- *Ease of troubleshooting.* Troubleshooting can be performed on smaller segments, with fewer devices per segment. Also, most hubs have management software to aid you, as well as LEDs on the hardware that indicate power and network activity.

- *Expandability.* The wise network manager will leave a hub underpopulated, so that a sudden influx of devices can easily be accommodated and users won't have to wait for new equipment to be installed.

- *Modularity.* LocalTalk hub design is flexible enough that one or two design standards can accommodate the needs of a very large company. The standard can then be replicated as needed instead of being redesigned each time. Besides saving design time, troubleshooting is easier when you can make certain assumptions about the network that are true regardless of location.

- *Simplicity.* Structured wiring using a single design can result in networks that are easy to conceptualize, document, and maintain.

But you'll have more to deal with than the normal problems of unplugged connectors and unpowered devices. Here are some of the kinds of problems you might see in LocalTalk hub networks.

Although all of the major manufacturers subject each unit to a burn-in test prior to shipping, a small percentage of hubs fail during use. Typically, one or two ports discontinue functioning, but in some cases, the entire unit will fail. If just one port is bad, you'll know because switching the network wires to a different port will make the problem go away. If the entire hub fails, you can tell because devices on the same port will be able to communicate with each other, but devices on different ports won't.

Typically, though, an entire hub will fail through its power supply, and the fact that all of its LEDs are dark will be a pretty big clue.

Many LocalTalk hub manufacturers offer next-day "swap" replacements at no charge; however, spare units are very handy in such cases. A spare unit is also useful, of course, to swap in when you suspect that the hub might be at fault.

Be sure to check the LEDs for power and activity, as well as the Amphenol cable connections between the hub and its wire distribution device (patch panel, punch-down block, and so forth).

Special Considerations for Troubleshooting Ethernet

Ethernet for the Macintosh is a special animal because of Apple's proliferation of bus architectures and its addition of the AAUI port. A look at the product sheet of any of the vendors who make the so-called full line of Mac Ethernet cards will show how many combinations of Macintosh bus types (6) and connector types (4) are necessary for a vendor to claim that it carries a full line of products.

Besides changes in technology, the other major change has been the incredible drop in the price of Ethernet components since the finalization of the 10BASE-T standard. The least expensive hub at that time was around $250 per port and the least expensive Ethernet card for the Macintosh was around $400. A recent advertisement in *MacWeek* offered cards for $160 and an 8-port hub for $299, less than $38 per port. Ethernet has become a price-driven commodity.

The effect of these trends on troubleshooters is that Ethernet networks often combine a multitude of different products from several vendors. While Ethernet standards such as 10BASE-T and 10BASE-5 are fairly exact, there is still the chance of incompatibility between two vendors' implementation of the same standard. For example, one vendor's initial implementation of its 10BASE-T hub had an AAUI port that was incompatible with many other vendors' repeaters. Recently, we could not mix Ethernet cards from different vendors in the same Macintosh (to make a router, for example) because the Ethernet drivers are vendor-specific. Since the introduction of AppleTalk version 56, Apple has implemented a multivendor driver architecture which has largely resolved this issue.

Besides the real, technical incompatibilities between vendors and components, there is also the problem of imagined incompatibilities. It's easy for a vendor to blame the incompatibility on someone else's product, and the level of confusion that results from finger-pointing rises exponentially as the number of vendors increases.

Sometimes the incompatibility will have a cause that you will not be able to directly determine without very specialized test equipment, like a very minute fluctuation in the frequency of a signal. In setup troubleshooting, you might have to rely on inference and circumstantial evidence to single out the cause of an incompatibility. Component swapping between vendors is a powerful method for this kind of indirect determination, but it should be done scientifically—swapping one component at a time, recording all results, and checking all possible combinations—to avoid an erroneous conclusion.

CHAPTER FIVE

Troubleshooting by the Layers—The Theory

To illustrate how AppleTalk protocols work and the functions they perform, let's consider a typical user action on an AppleTalk network and the way in which the protocols respond and accomplish their tasks. The user action is double-clicking a folder icon from a server volume to receive a listing of the contents of that directory on the screen. We'll discuss the functions of the AppleTalk protocols in each layer and the types of problems that can occur in the layers.

Application Layer

Function	Allows the user to interact with the computer, giving it commands and receiving the results of those commands.

For users to communicate with the computer, they must be able to give commands that indicate specific computer functions. The computer, in return, must display data in a way that makes sense to a human. The Application Layer protocol establishes the meanings of the actions users can take (mouseclicks, keystrokes, menu choices, and so on) as well as the meaning of the data that the computer displays (icons, windows, text).

The programmer of every Macintosh application that uses AppleTalk must create such an Application Layer protocol. *Human Interface Guidelines* (Addison-Wesley, second edition, 1993) acts as a guideline for creating Application Layer protocols by promoting and suggesting the principles of good Macintosh user interface design.

Each application has its own set of meanings for the windows, icons, and menus it uses to display information to the user. In turn, the application also has a set of meanings for the various mouseclicks and keystrokes that the user employs to communicate to the application. In the Finder, the icon corresponds to a subdirectory on the server volume, and the act of double-clicking that icon communicates the user's desire to see the contents of the subdirectory. The Finder responds by setting in motion the chain of events to retrieve that data from the server. When the data returns, the Finder creates a new window on the user's monitor and places objects inside that window. The window represents the subdirectory and the objects inside the window represent the contents of the subdirectory.

Application Layer problems

Because the Application Layer represents the relationships between desires and actions and between images and data, a problem in the Application Layer occurs when the user cannot understand or consistently remember the relationships. Application Layer troubleshooting involves minimizing or removing the misunderstanding and its effects. The troubleshooter tries to understand exactly why the user has difficulty with the interface by asking questions and observing the user. Depending on the nature of the misunderstanding, the network manager can either change the command procedure so that that user understands it, educate the user, or create reference information.

Although the Macintosh is generally considered easy to learn and use, it can be confusing to some users. The metaphors the Macintosh uses (such as the filing cabinet, folder, and document metaphor) are better suited to some users than others. For example, an inexperienced Mac user might hesitate at first to eject a floppy disk by dragging it to the Trash, believing that this might cause the disk to be erased.

The best Mac users are people who understand the current state of their computer, the state they want it to attain, and how to make the change. These users appreciate the fact that you can open a file by double-clicking on the file icon, double-clicking the alias of the file icon, opening it from an application, or selecting it from the Apple Menu after placing either the file or its alias inside the Apple Menu Items Folder. By contrast, users who are more "procedurally oriented" prefer to follow a precisely defined series of steps.

Networks invariably contain both kinds of users, and both need to carry out their operations without confusion or annoyance. To some extent, the network manager can help users customize their computing environment according to their individual preferences through utilities such as System 7 aliasing features, CE Software's QuicKeys, Now Software's Now Utilities, or UserLand's Frontier. These utilities can help users extend and customize their environment in such a way that operations are performed in a more personal style. For example, you might use QuicKeys to customize a procedurally oriented user's Macintosh to provide the following reference guide (on paper):

Action	Purpose
1. Press F15	Switches Mac to the Finder
2. Press F10	Mounts a server volume using QuicKeys mounting extension
3. Press F7	Opens a spreadsheet application
4. Press Command-O	Brings up the Open dialog box
5. Press Command-2	Sets the default directory to the template folder
6. Type "exp"	Selects the locked expense report template
7. Press Return	Opens the expense report template
8. Modify the expense report by adding your expenses in the appropriate columns	Adds the new information of your expenses
9. Press Command-S	Invokes the "Save As" command, because the file is locked
10. Press Command-3	Sets the default directory to a designated drop folder
11. Type your name and the date (example: "FMiller5/19/92")	Sets the file name
12. Press Return	Saves the file
13. Press Command-Option-Q	Quits application and dismounts server

You might know of some users who could benefit from having a very systematic, documented way of accomplishing a routine job like this one.

Presentation Layer

Function Takes the user's commands and data and represents them in a format recognizable to a computer.

At the Presentation Layer, the user's command, "Show me the contents of this subdirectory," is transformed into the proper computer commands to carry out the user's wishes. Because the folder icon represents a folder on the server, the information must come from the server. When the Finder gives the command to carry out the user's wishes, the AppleShare client software in the Macintosh intercepts the command and translates it from a Macintosh-specific file command into the equivalent command(s) in AppleTalk Filing Protocol (AFP). AFP consists of a set of commands and data structures that represent a file system and its commands in a device-neutral way.

Double-clicking on a server-based folder is different from double-clicking on a folder icon that represents a folder on your own hard disk. That action generates Mac-specific commands that manipulate the Mac's own file system, HFS (Hierarchical File System). The AFP translation is needed only for operations on storage volumes that are located elsewhere on the network.

The server has a symmetrical AFP process that transforms the command from the AFP language into a set of commands that correspond to its own native filing system, even if the server is also a Macintosh. AFP is referred to as a file "metalanguage" because it is a language that is used to represent another language—the language of the native filing systems on the clients and servers on the network. In this way, the differences between the filing system are resolved and the user can easily and transparently store and retrieve data on different types of computers, as long as they can provide the translation between AFP and their native file language. AFP translation software is currently available for DOS, VAX/VMS, and many varieties of UNIX systems, as well as some proprietary network operating systems used by Novell, Microsoft, and Banyan.

Applications are almost completely isolated from this translation. The distant device, server or printer, is treated like a local device as far as the application is concerned. That is why any application that can open a file can open it from a server volume as easily as from a local volume.

PostScript is another example of a Presentation Layer metalanguage. Just as AFP represents file commands and operations in a device-neutral way, PostScript can describe drawing commands and image data in a device-neutral way. When you print, your printer driver transforms the Mac's own imaging system, QuickDraw, into the equivalent PostScript commands and sends them to the printer. The printer then translates the PostScript commands into its own native imaging language to print the image.

Presentation Layer problems

A metalanguage such as AFP masks the differences between file systems and computers. Like most standards, AFP is comprised of a set of functions chosen for their universality—a "lowest common denominator" of file systems—and might not reflect the richness of individual file systems. For example, the VAX/VMS operating system has a rich set of backup utilities, but these are not part of the AFP language because backup utilities are not native to all operating and file systems (including the Mac's). As a result, there is no AFP command to communicate to the VAX that you would like to use these utilities.

A second problem is that the imposition of a metalanguage between file systems causes a slight reduction in performance. Because the translation of native file operations to and from a file metalanguage consumes CPU cycles and other resources, it adds some extra processing that a local file operation does not have. That is part of the reason why a VAX that is much faster than a Mac in native mode is a slow AFP server. Also, because the metalanguage must be built so that it is flexible enough to represent the many types of file systems that exist, AFP sometimes sacrifices speed for the sake of flexibility.

In the Presentation Layer, one of the biggest sources of problems is when the versions of the application, the Chooser Extension (which typically performs the translation to the metalanguage), and the server software are less than 100 percent compatible. When this is true, the translations of commands and operations from native language to metalanguage and back to native language can cause problems. A second, more subtle, concern is that the local application does not ask for an operation that has no easy translation for the server device. For example, an application might want to open a file and a duplicate copy of the file at the same time. Perhaps this is a normal occurrence for files on the local volume but represents a function not allowed on the server device. In the gap between what the application wants and what the server allows, problems can occur.

Session Layer

Function Manages the computing resources used by two devices as they remain in relation with each other. Session Layer handles such activities as establishing the relationship, allocating resources (such as memory and CPU time), exchanging authentication and privilege information, ending the relationship, and de-allocating the resources.

The relationship between two computers consumes resources, and the job of the Session Layer is to manage those resources. In the example of double-clicking on the folder icon, the client's workstation must have enough memory to buffer the incoming directory data until it is ready to be displayed. The Session Layer protocol in both the client and the server allocates buffer space for incoming and outgoing packets when the connection is established. The Session Layer protocol on each device also establishes a session identifier, creates processes to communicate, and logs the client into the server by exchanging authentication information (for instance, account name and password). When the user dismounts the server, all of those resources are returned to both computers.

The Session Layer also has the responsibility of detecting the premature loss of connection. Session Layer protocols expect to receive data on a continuing basis. Although there might be periods of time during a session when no data needs to be sent, the Session Layer protocol in each device has some kind of mechanism to determine whether its counterpart in the other device is still available over the network. A common technique is to send and receive "tickle" packets to indicate that the session is still alive and well.

If the Session protocol on one of the devices does not hear these reassuring tickle packets from the other device, it might try to locate its partner device. If it can't locate the other device, it independently closes down its end of the connection. This might happen if the other device crashes or is turned off suddenly, or if a network device between them crashes. By closing down the dead connection, the Session Layer protocol can recover the resources it allocated for the session and return them to general use.

After the session is closed, even if both devices become available again within a short period of time, the session is generally not restorable. While a new session can be initiated, the old session cannot be rejoined in progress. This is true both for AppleShare Session Protocol (ASP), which manages the resources for AFP Server sessions, and for Printer Access Protocol (PAP), which manages the resources for sessions between an AppleTalk workstation and LaserWriters.

Sessions between devices that are managed by ASP or PAP tend to be very structured and asymmetrical. For example, there is a limited number of commands that you can send to an AppleShare File Server, and virtually all of the communication is of the command-reply or question-answer format. A Mac's relationship with a printer or a file server is asymmetrical because there is a significant distinction between client and server, and each device has a different set of prerogatives and requirements.

Besides ASP and PAP, which have been used for many years, there a relatively new Session Layer protocol named AppleTalk Data Stream Protocol (ADSP). ADSP has a different nature than ASP and AFP. ADSP provides a flexible, efficient method for managing resources—it has the ability to adjust its allocation of resources mid-session. As its name implies, it is most useful for managing data stream relationships. In a data stream relationship, there is not always such a great distinction between server and client. The two devices might in fact be more like two peers. Also, ADSP allows other communication formats besides simple question/answer or command/reply. Because of ADSP's special features, programmers are beginning to use the protocol for an increasingly diverse variety of applications.

Because ADSP is a Session Layer protocol, besides managing resources it also handles log-in and authentication. But ADSP also performs some protocol functions that are normally considered the province of the Transport Layer. In fact, ADSP completely bypasses the Transport Layer and communicates directly with the Network Layer. Because ADSP performs all of the tasks of both Session and Transport Layers, it breaks some of the rules of the OSI Seven-Layer model. It's normal for real-world protocols to break the OSI rules once in a while.

In addition to its Session Layer duties, ADSP also performs the Transport Layer functions that manage the reliability of the data and provide a sequencing mechanism for the packets. These aspects of ADSP are discussed later in the Transport Layer section.

As you can see, there can be more than one protocol available at any layer. An application programmer can choose any of the Apple-supplied protocols in any layer. The programmer chooses protocol based on its suitability for the kinds of communication that would normally occur during the use of the application. At the Session Layer, ASP and PAP are well suited to the jobs they perform, but a programmer could also use them for a different purpose than using a file server (ASP) or communicating with a printer (PAP). Alternatively, the programmer can invent a Session Layer protocol of their own. This is fine, as long as the protocol works within the structure of the rest of the AppleTalk protocol stack. Examples of applications that use proprietary Session Layer protocols are Novell's Data Club, Sassafras Software's KeyServer, and Farallon's Timbuktu when it controls a device over the network.

Programmers invent their own protocols when they feel that the standard protocols are not efficient or fast enough to handle the communication needs of his or her program. Because the standard AppleTalk protocols must handle a wide variety of tasks, they are designed to handle a broad range of requirements, while proprietary protocols tend to be optimized specifically for the communication needed for the application. The fact that an application uses its own proprietary protocols at any of the AppleTalk layers is not usually a problem, except that the proprietary protocols are not documented in the way that AppleTalk protocols. Because they are not documented, they are harder to understand and troubleshoot.

Another Session Layer protocol that does not exactly fit into the mold of the Seven-Layer Model is Zone Information Protocol (ZIP). ZIP provides zone names to network number mapping, which is more similar to a Transport Layer function than a Session Layer function, but it is nonetheless positioned in the Session Layer. ZIP is discussed in the Transport Layer section because it works so closely with Routing Table Maintenance Protocol, a Transport Layer protocol.

Session Layer problems

Although the ability to close down "half-open" sessions is important, under certain circumstances the Session Layer can cause trouble when it closes down its half of the session independently of the other device. Premature session shutdowns can be a problem on dedicated servers when the network link is faulty or interrupted. If the network link between the two devices is inoperable long enough, it can trigger the automatic shutdown even if both devices are healthy. This creates more of a problem for the client than for the server.

When the session is terminated, only the portion of the file currently in the client's RAM remains. From the point of view of the application managing the data in the file, this piece of the file is now the entire file. When the network link is restored and the server is available again, users can reestablish contact with the server by logging in for a new session. However, users might not be aware that the file they are working with is only a piece of the original file. If they decide to save their changes to the file on the server, they replace the older complete file on the server with the newer, incomplete file in their workstation's RAM. The Save operation should warn users by asking "Replace existing file?"

AppleShare servers running on a dedicated Mac with no auxiliary processes mounted (such as mail servers or routers) are very stable and can run for years without crashing. If a client workstation crashes or is suddenly unavailable during the session, the server can usually close all open files with minimal damage, if the files are damaged at all. An exception to this is when the user's workstation crashes in the middle of a Save operation. Because saving the changes to a Macintosh file automatically replaces its previous version, a crash during this operation can cause significant damage or even total loss of the file being saved as well as its previous version.

Non-dedicated servers such as System 7's File Sharing have a reliability problem that results from the fact that the server is also a user's workstation. If the server crashes due to a user process, irreversible file damage can occur. Another threat is that a user whose workstation is being used as a server might end the session abruptly by shutting the Macintosh off. Users should be trained to avoid these kinds of problems in a file sharing environment and know which operations have an adverse effect on file-sharing clients. These operations include turning the workstation off, turning off file sharing (which is necessary in System 7 to eject SCSI volumes on removable media), switching the Data Link connection, and so on. Users should also know how to determine the identity of their clients with the File Sharing Monitor Control Panel.

One of the newer features implemented in AppleTalk-accessible servers is the ability to set and adjust timers, like the connection timer mentioned above, based on local network conditions. This feature is necessary because AppleTalk is used over so many types of data links—fast, slow, reliable, and unreliable—and one set of timing parameters is not appropriate for all situations. Although this technology is important, it is still emerging, and the ability to adjust timers has not yet been developed for optimum effect.

Transport Layer

Function Manages the delivery of data through the internet and copes with problems such as information getting lost along the way or changes in the structure of the internet.

The Transport Layer's job is to manage and assist the Network Layer's delivery of data through the internet. The Transport Layer is teamed with the Network Layer for the task of data delivery. The Network Layer performs the delivery and the Transport Layer makes sure that it is done correctly.

The primary function of the Transport Layer protocols is to manage the reliability of the data delivery. There are many reasons why a particular packet might become lost in transit or arrive out of sequence. Transport Layer protocols must be able to handle these situations as well as cope with changes in the structure and design of the internet as new workstations, hubs, routers, networks and sites are added and old ones are removed. The Transport Layer must also be able to handle extreme cases, such as when a device is removed from service through a crash or sudden loss of electrical power.

In the example of double-clicking on the folder icon, the functions of reliability management and packet sequencing are handled by AppleTalk Transaction Protocol (ATP). ATP transactions consist of one machine initiating the transaction by asking a question or giving a command—in this case issuing a request for the contents of a subdirectory.

The ATP transaction request has other features besides just the question itself. First, it is numbered to provide a link between request and response. Second, it informs the responding device of how many packets (up to a maximum of eight) it can hold in memory for the answer. The responding device answers the request with a series of response packets that each carry, in addition to the answer, the identifying number of the transaction and the sequence number of the packets within the transaction. The series of response packets is sent until the requester's memory space is full or the answer is complete, whichever happens first. In the case that eight packets are not enough to hold the entire answer, the requester can immediately initiate a new transaction request to receive the rest of the answer.

The sequence numbers also provide a way to ask which packets have been received. The server has the ability to ask, "What packets in transaction #242 have you received so far?" The client can respond, "I've received packets 1, 3, 4, and 5 of transaction #242 so far. I'm still waiting for packets 2, 6, 7, and 8."

To review our example up to this point, AFP translates the file command into the appropriate syntax for the network, ASP provides the management of resources, and ATP provides a way of linking the question and the answer through the transaction format and provides reliability through a method of checking and acknowledging what has been delivered.

Another important function of the Transport Layer is to cope with changes in the internet as new routers and networks are added and old ones removed. When you bring a router into service, you supply it with the identity of the networks and zones to which it will be directly attached. As it enters service, it exchanges this information with other routers in an effort to learn the location of every network in the internet, the zone name associated with that network, and the best path to reach that network. The purpose of learning the internet is to be able to provide accurate zone information to workstations on request and to provide a routing service for packets when the sending device and receiving device are located in different networks.

Learning the internet structure and providing the services are the duties of the internet routers, using the algorithms of Routing Table Maintenance Protocol (RTMP) and Zone Information Protocol (ZIP). RTMP and ZIP are the most troublesome protocols in the AppleTalk family both in their design and their implementation in the current crop of internet routers. RTMP and ZIP processes, their common problems, and the techniques used to diagnose and fix their problems, are described in great detail in Chapters 7 and 8.

AppleTalk uses names to refer to services that are available over the network. A software process that needs to be contacted by other software processes can use Name Binding Protocol (NBP) to establish a name for itself that other devices can locate. An example of this is the AppleShare server of the example described earlier. It established a service name for the server and "bound" or linked it to an Internet Socket Address. The service name includes the name, type, and zone of the server, and the Internet Socket Address includes the net, node, and socket of the server. The Internet Socket Address is thus established as the address to send information when you need to contact the service; the service name is viewable by the user in the Chooser. An example of this is VOP VAX:AFP Server@VOP Zone (server name "VOP VAX," an AFP Server located in the "VOP Zone") linked to 3.145.251 (network 3, node 145, socket 251).

The ability to name and contact services accomplishes two purposes for AppleTalk. First, it gives the user something more descriptive and meaningful to use than a numerical address. The second reason that named services are necessary comes from the dynamic addressing characteristics of AppleTalk (discussed later in Data Link Layer section). Because AppleTalk addresses nodes dynamically, node (and socket) numbers can change arbitrarily from one day to the next. Names, however, change only when a network manager uses software to deliberately change the name of a device. Whenever the Mac invokes a service, NBP searches for the service by name, finds out its address, and the rest of the communication can be carried out by address only. Using a name lookup instead of address-based lookup ensures that the service can be contacted even if its address changes periodically.

We mentioned earlier that ADSP performs some functions of a Transport Layer protocol, specifically the management of delivery reliability and sequencing. ADSP performs these functions very differently than ATP. The most striking difference is that ASDP does not use the concept of a transaction to segment and structure the communication during a session. In ADSP, the connection consists of two simultaneous and continuous flows of data—one flow going from client to server, and the other from server to client. Questions and answers, commands and replies are just part of the stream; data is identified only by its position in the stream. The bytes that make up each stream are numbered beginning with Byte 0, and every packet in the stream is identified by the number of the first byte in the packet. In ADSP, if device A has already sent 432 bytes of information to device B, the packet includes the number 432 to indicate its position relative to the data previously sent.

Every ADSP packet also tells the receiving node how many bytes the sending node has received and how much memory it has allocated for further reception.

Transport Layer problems

The biggest problems in the Transport Layer come from the routing and zone functions. (The reliability management and sequencing functions rarely cause problems.) The most common sources of routing and zone problems are:

1. Bugs in router software and hardware that cause errors and crashes
2. Incompatibilities between routers from different vendors
3. Unintentional misconfiguration of router
4. Introduction of an inappropriately configured router into the internet by someone other than an authorized network manager

These problems are discussed at length elsewhere. For more information, refer to:

Device Processes	Router Startup on Ethernet
Device Processes	Maintaining Routing Tables and Zone Tables
Techniques	Finding and Correcting Router Configuration Problems

Network Layer

Function	Performs the delivery that Transport Layer manages.

Datagram Delivery Protocol (DDP) is the sole protocol in AppleTalk's Network Layer, and is used by all of the higher level protocols to carry information between two software processes through the internet.

In the sending device, DDP takes the information passed down from the higher layer protocols and packages it in a DDP Datagram, which includes the Internet Socket Addresses (net.node.socket) of the source and destination processes. DDP then determines whether the packet must be forwarded through a router or can be sent directly to the destination node by checking to see whether the source and destination processes are on different networks.

If the source and destination processes are on the same network *and* that network is non-extended, then DDP can use a special short format. In this instance, DDP can use the short format because the network addresses are exactly the same (because non-extended networks can only have a single network number) and because the DDP source node and destination nodes are the same as the Data Link source and destination node addresses. In this short format, DDP can simply supplement the Data Link information with the socket addresses of the source and destination processes.

In all other cases, DDP must use the long format, which gives the complete Internet Socket addresses. The long format also includes a 2-byte field where a checksum of the datagram can be inserted. While each link layer CRC will guarantee integrity as the datagram traverses each link along its path through the internet, the DDP checksum in the datagram is meant to protect the data integrity from start to finish across the internet.

The DDP checksum function is almost never used in AppleTalk, however. The DDP checksum is considered redundant because each of the link types that AppleTalk uses has its own built-in integrity check. Also, requiring devices to compute checksums for all of their datagrams would be time-consuming. While link layer CRC's are almost always calculated in hardware, checksums would have to be calculated by most devices (including Macs) in software, which would further slow down all network communications.

This reasoning is not entirely sound, however, because there is one source of data corruption that is left undefended, the router's buffer memory. A router will check an incoming packet's CRC during reception and store the packet in buffer memory. Then, when it sends the packet on the next leg of its journey, it calculates a new CRC for the packet. If the packet's data is corrupted inside the router's buffer memory, the router will not be aware of the corruption. The link layer CRC that the router's hardware calculates, then, will be based on corrupt data.

Does this cause network errors to remain undetected? Yes. Perhaps you have spotted these network errors when transferring compressed files through the internet. On the destination machine, if you tried to decompress the file and got an error message saying that the file was corrupted, it may be due to errors during file transfer. Compression programs typically compute an integrity flag (similar to a checksum or CRC) for the files they compress. The file integrity corruption might not always come from a network error. Perhaps the file type or creator type changed for some reason; restoring these to their original settings should make the file whole again. In other cases, however, the file integrity flag is showing you that the transfer process has introduced errors into the file.

Certain routers have been known to have memory corruption problems, including the Shiva FastPath 4 and the Novell NetWare router (v3.0). Shiva referred to this euphemistically as "memory leakage". Although this kind of data corruption happens only very rarely and only with routers that have memory corruption problems, if you're extremely worried about data corruption, you may consider compressing your files prior to transfer using a compression program like Alladin's Stuffit Deluxe, which provides an integrity check mechanism for the file. This mechanism, of course, can also be used to check your internet for this particular kind of corruption problem.

Network Layer problems

Besides the weakness in the Network Layer resulting from Apple's decision to make the DDP checksum optional, the only other problem in the Network Layer hat is not a result of a problem elsewhere is when a user receives the dialog box that begins with "Access to your internet has changed..." and is instructed to use the Network Control Panel to rejoin the network. This dialog box appears when a non-routing node (on extended networks only) reads an RTMP packet that changes its perception of which network it is on. A typical example is when a router appears on an EtherTalk or TokenTalk network that previously had no routers. RTMP packets are sent by routers every 10 seconds and non-routing nodes retain a small portion of the information in the RTMP packet known as the RTMP stub. The RTMP stub holds 2 pieces of information for a non-routing node: the network's address (in Inside AppleTalk, this piece of information is referred to as "THIS-NET") and the node address of the router that sent the RTMP packet ("A-ROUTER"). Non-routers keep these two pieces of information and ignore the rest of the information contained in these RTMP packets, which is interesting only to any other routers that may be connected to the network.

On a network with more than one router, it's normal for a non-routing node's value of A-ROUTER to change as it reads the RTMP packets coming from the various routers.

Despite the fact that A-ROUTER may be changing constantly, the value of THIS-NET should remain constant because all the routers on the network should agree on the identity of the network's address. If the value of THIS-NET changes from "Undefined" (no routers on the network) to a value or from one value to another, the previously mentioned dialog box will appear on the Macintosh screen. (Note: The dialog box does not appear when THIS-NET changes from a positive value to "Undefined", which occurs approximately one minute after a node stops receiving RTMP packets.)

For some reason, many users choose to ignore this dialog box and don't use the Network Control Panel as instructed. Depending on the nature of the change in THIS-NET and the actions of their fellow users, they may continue enjoying some of the network's services despite their inaction. To regain the full use of the internet services, however, they must follow the instructions to use the Network Control panel and re-establish the network services. Of course, this is also true for dedicated and possibly unattended server devices, some of which may lack a Network Control Panel and may require a restart to accomplish the re-establishment of their network services.

Data Link Layer

Function Manages the tasks of packaging the information for transmission, gaining access to the network, and screening incoming information to make sure that it is readable. In AppleTalk, the Data Link Layer also makes sure that each device has a unique address.

Just as the Transport Layer manages the activities of the Network Layer, the Data Link Layer manages the activities of the Physical Layer. The Physical Layer is responsible for sending and receiving packets. The Data Link Layer makes sure that this job is performed correctly.

AppleTalk has a Data Link protocol for every network type on which it can operate. The Data Link Layer protocols are named for the network system they use. The LocalTalk Link Access Protocol (LLAP) runs on LocalTalk, EtherTalk LAP on Ethernet, and the TokenTalk LAP on Token Ring and so on. Apple has stated publicly that it intends to expand this list to include all the major networking systems currently in use.

The Data Link protocol takes the AppleTalk information that DDP packages and places it inside a data frame that is formatted for the kind of physical network it manages. The term "data frame" refers to the packet and all of the auxiliary information needed to send it. The data frame includes information that is necessary for the networking hardware but is not used by the protocol in any way.

For example, at the beginning of a data frame, the network hardware typically requires a few special bits that synchronize the hardware clock of the receiving device with the hardware clock of the sending device. This synchronization field is placed before any addressing information. At the end of the packet, the network hardware might require an error checking field or some type of special bit pattern to signal the end of the frame.

Besides framing the protocol data, Data Link Layer manages the access to the network. All networks have some kind of method of Media Access Control (MAC) that lets devices know when they can send data and when they can't. LocalTalk, EtherTalk, and TokenTalk all use different methods of MAC. Regardless of the method used for MAC, the purpose is the same—to make sure that only one device transmits information at a time.

LocalTalk and EtherTalk are both examples of contention networks, which means that if a device wants to send a packet, it must first sample the network wire to see if there's a signal already on it—another node sending a packet. If there is a signal, the device waits until a short time after the signal passes by, and at this point LocalTalk and EtherTalk diverge.

LocalTalk's MAC is called Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) or simply *collision avoidance*. EtherTalk uses CSMA/CD or *collision detection*. A collision occurs when two devices are simultaneously transmitting, in which case neither signal is understandable by any device.

In addition to building the frame and gaining network access properly, the Data Link protocol is also responsible for checking incoming packets for physical errors. Which types of errors Data Link protocol checks for depends on network type, but the purpose of checking is to make sure that the received packet is identical to the packet that was sent. This process includes checking to see that the packet is whole and that the packet remained in synchronization with the receiver's clock. The checking procedure can also use a mathematical check sequence to verify the integrity of the data inside the packet.

The error checking that Data Link Layer performs differs from the error checking performed by the Transport Layer. The Data Link Layer's protocol checks that the physical network hardware sent and received a packet with physical integrity. The Transport Layer makes sure that the entire delivery was made and that all the data arrived, An analogy can be made to a person ordering equipment from a mail-order house. When the packages arrive, the receiving clerk is like the Data Link layer He makes sure that the package is not damaged, that it has a shipping manifest, that the packing seal is not broken, and that it is addressed to the name of a real employee. If the package fails any of these tests, he rejects the package. If it does pass all of the tests, he forwards the package to the employee who ordered the equipment. The employee performs tests like the Transport Layer before she considers the goods "delivered." The employee opens the package and compares the contents of the package to the order that she placed. If there are missing items, she calls the mail order house and requests that the missing items be sent in a new package.

In this analogy, the receiving clerk can only perform his checks on packages that arrive at the loading dock. If packages get lost in transit, he has no way of knowing that anything is missing. Only the employee expects an arrival. The Data Link Layer checks the packets it receives and rejects damaged packets. It does not request retransmission. The Transport Layer has the sole responsibility of requesting packets to be resent if they are not received by the Transport Layer protocols.

Data Link Layer problems

The chief concern in Data Link layer troubleshooting is that the link can reliably transmit packets between any two of its nodes. Several pieces of testing software can determine whether this concern is being met. Examples include Dartmouth's MacPing, Caravelle NetWORKS, and Apple's Inter•Poll. If the link is unreliable, the concern then shifts to the reasons for the unreliability. The most effective first step is to pin down where the network link is reliable and where it is not. Of the three tools mentioned, MacPing is the most useful because it can provide a graphical display of the reliability between the device running the MacPing application and every other node on the link.

Another good source of information to help in determining the origin of the problem is the statistics in the network hub, although these can be difficult to interpret at first. These troubleshooting processes are described in the Physical and Data Link sections of the next chapter. Sometimes these techniques will actually find the cause of your problems; other times they will only indicate the area in which to concentrate your search. For example, you may determine that your trouble spot is on port 11 of a particular hub, but still not know which device or component on that port is responsible for the problem. Then, the techniques of Setup mode troubleshooting explained in Chapter 4 can be used to zero in on the cause.

A second concern of the Data Link Layer in AppleTalk is that every node *really has* a unique node address. Although every data link specification in AppleTalk has mechanisms to insure this, these mechanisms occasionally fail. On LocalTalk networks, users may duplicate existing node addresses if they turn their Macintosh on or wake their PowerBook from sleep when it is not plugged into the network. Network managers may inadvertently create duplicate node addresses in LocalTalk by switching a user from one network to another in the data closet without having the user restart his workstation. On large, bridged Ethernet and Token Ring networks, devices may inadvertently take duplicate node addresses when there are too many and too slow bridges for packets to traverse during the two-second period that a node checks its address for uniqueness at start-up. To resolve this problem, the network manager can either reduce the number of bridges, replace the bridges with routers. If these methods are not possible, another method is to drastically increase the size of the network range, which reduces the statistical likelihood of address duplication.

When two user nodes have the same node address, they can generally accomplish most network tasks, but with diminished performance. They will not, however, be able to communicate with each other directly. If server nodes are affected by the address duplication, the problem can be more serious and affect more users. On Ethernet and Token Ring networks, duplicate addresses are also accompanied by a larger than normal amount of AppleTalk Address resolution Protocol (AARP) traffic. A careful scan of the device list in Inter•Poll can usually spot the duplicate address problem; two devices with different names will have a common address. If one or both of them are restarted, the problem will usually clear after a short period of time.

A related problem that can affect any type of network link results when a device is unable to find a unique node address when it joins the network. This generally happens only when the number of available node addresses is very low and the pseudorandom number generator does not pick one of the few addresses remaining in its allotted number of tries. This problem can be resolved by either reducing the number of devices on the link or by increasing the size of the network range (in EtherTalk or TokenTalk). A good rule of thumb for extended networks is that there should be at least five times as many node addresses as there are nodes.

Physical Layer

Function	Performs transmission under the guidance of Data Link Layer—creates outgoing signals and decodes incoming signals.

The Physical Layer "protocols" are actually specifications that govern the network and signaling hardware and specify the rules for connecting them. Physical Layer specifications govern such network parameters as cable length and type, the maximum number and spacing of repeaters on a network, the maximum number of nodes and the maximum distance between nodes, the characteristics of the signaling method used, the tolerance for signal errors, and the maximum signal propagation delay between stations.

For a committee-produced standard like IEEE 802.3, the governing committee has an official document that spells out the details of the standard. For LocalTalk, a network invented by Apple, the specifications for the Physical Layer are published in *Inside AppleTalk*. No one uses these specifications for building a real network, however, because the purpose of these documents is to act as a specification for hardware designers. The specifications used by the people who build networks are typically written by the manufacturers of the network products.

An example of a more usable Physical Layer specification is contained in the manuals that come with products in Farallon's PhoneNET System. These manuals contain guidelines, rules, suggestions, and tips and are more practically oriented than the actual LocalTalk specification.

Building a good network is still something of an art, in spite of the voluminous documentation from standards committees, hardware manufacturers, cable manufacturers, and trade publications. These documents are sometimes more stringent than necessary, but it's almost always best to follow the manufacturer's guidelines to the letter if reliability is a concern.

Physical Layer problems

When the Physical Layer is working well, every signal sent is electrically clean, travels smoothly over the wire, and is received in error-free condition. When the Physical Layer has problems, the symptom is that the signals have errors. Signaling problems are easily conceptualized—either the signal was not generated properly, developed errors during transmission, or was not received properly.

Most of these problems are caused by improper installation of the network. Sometimes network installers deliberately install the network improperly in an attempt to cut costs. This happens more often in LocalTalk networks because of LocalTalk's "forgiving" nature. The network installer, encouraged by the success of networks with small deviances from the manufacturer's recommended limits, deviates more and more from specification. The networks grow as more users are added and begins to experience more and more lost and damaged packets. Before long, all of the money saved by cutting corners on the installation is lost to the high cost of maintaining the patchwork network. We refer to this kind of networking as "guerrilla net."

In other cases, the mistakes are unintentional or result from mistaken notions of the network's physical requirements. Examples might include a bad choice of wire or connector jacks due to unfamiliarity with the available choices. When a network manager makes the transition from LocalTalk to Ethernet, he or she might be surprised to find the wiring that worked well for LocalTalk signals performs only marginally or not at all for Ethernet. Another common mistake is a misunderstanding about termination. This happens more frequently with LocalTalk than Ethernet. Although both networks need proper termination, a badly terminated Ethernet simply does not work. A badly terminated LocalTalk works, but not well.

Problems also result from physical damage to the network cabling while in service. This kind of problem can be very subtle and hard to find, like a slightly kinked coaxial cable or a slightly loose connector.

Some problems are caused by unexpected incompatibilities between supposedly compatible hardware products. For example, a transceiver from Vendor A might be incompatible with a card from Vendor B when used with software from Vendor C, even if all were designed according to the same specification.

Some network hardware can also go bad after some time of good service. A memorable example of this was the first series of Apple LaserWriters, some of which would begin to "jabber" after a year or two. ("Jabbering" is the condition where a device spontaneously transmits nonsensical signals on the wire.)

In addition to problems caused by improper installation, electrical problems that can cause signal errors are crosstalk, electromagnetic interference from fluorescent lighting or electric motors, and signal reflections caused by local impedance variations.

The problems associated with the Physical Layer and Data Link Layer are mechanical and electrical in nature, and the Setup mode of troubleshooting is almost always the right choice.

Summary

Layer mode troubleshooting locates network problem by determining which aspects of the network are working and which are not. The OSI Seven-Layer Model provides

vides the conceptual framework for the various functions of the network and serves as our troubleshooting guide. The functions provided in each of the layers are checked separately as we make our way through the model. Starting with Data Link Layer, we check to make sure that the functions provided by each layer are present and functioning properly. When the functions of a layer are shown to be healthy, we rise to check the functions in the next layer. When a layer's functions are unhealthy, we try to ascertain what obstacle is blocking those functions. If we cannot find the problem, we turn our attention to the layer below, because the functions in any layer can only take place if the functions in the layers below are working properly.

CHAPTER SIX

Troubleshooting by the Layers—The Practice

Many physical layer problems are simple. Common problems include the breakdown of the physical network because of a loose or unplugged cable, the placement of a plug in the wrong jack, or the lack of a terminator—things that a visual inspection can find and a simple procedure can correct. Visual inspection should be the first check in troubleshooting the physical layer. Once you've performed a visual inspection and corrected simple problems, you can use a more analytical approach.

The quality and integrity of a network's physical layer depends on three simultaneous conditions:

1. A continuous electrical path must exist between senders and receivers.
2. The cable path must be of high enough quality to reliably and accurately carry the network signal.
3. The hardware components along the path must be well-matched and of high quality so they don't degrade the signal quality.

These conditions are listed in the order in which they're usually tested. The first condition is the criterion of the telephone installer: Can you hear a dial tone between the two points? We'll look at a number of ways of establishing electrical continuity, including tone testing, ohm-meter testing, and the use of Time Domain Reflectometers (TDRs).

While electrical continuity is a necessary condition for data networks, it isn't a sufficient condition. Data networks require a higher quality cable than telephones for two reasons. First, network data signals use much higher frequency signals than telephone systems. Higher frequency signals decay more quickly and interact more readily with their environment than low-frequency signals. Wire is classified voice grade and data grade because of these frequency-based requirements.

Many network managers experience the effect of this frequency-based phenomena when they convert from a LocalTalk network to a twisted-pair Ethernet network. Ethernet, of course, uses much higher frequency signals than does LocalTalk. In many cases, the wire that reliably carries the LocalTalk signal isn't suitable for the 10BASE-T signal and some additional wiring is necessary.

The second reason that data circuits require higher quality wiring than telephone circuits is that a computer's ability to use poor-quality network signals is far inferior to a human's ability to interpret poor-quality voice signals. You may be able to understand someone talking to you over the telephone despite an echo on the line or the presence of static, although you might find these conditions annoying. Computers have a very limited ability to accurately interpret a network signal under such conditions.

There are two ways to examine the quality of the signal path. The first way is to use an electrical measurement tool that directly measures the quality of the path. For example, you can examine the signal with an oscilloscope, which will show you exactly how the signal has been distorted. With some of the better scopes, you can even measure the distortion. The second way to test the path quality is by gauging the effectiveness of the path by an indirect method—by seeing what percentage of echo packets get lost during a round trip between a sender and receiver, for example. We'll discuss these indirect tests in Data Link Layer Troubleshooting later in this chapter.

Indirect tests are generally faster at helping you judge whether there's a problem with the path; direct examinations are better at determining the nature of a physical layer problem. That's why we suggest that you start the "troubleshooting by the layers" method with the data link layer. Using indirect tests in the data link layer, you can establish whether or not there's a problem. With the direct tests of the cabling and components outlined in physical layer troubleshooting, you can find the nature of the problem.

The last condition necessary for a reliable physical layer is that the physical network components must be well matched and high quality. This is less of a concern with Ethernet components than with LocalTalk components because Ethernet components are manufactured to be compliant with one of several specifications: 10BASE-T, 10BASE-2, and so forth. If a manufacturer claims compliance with one of these specifications, barring a manufacturing problem, the component should work well and should also work with any other components that comply with the same specification, even if they're from different manufacturers.

In LocalTalk, howeve, the specifications for the physical layer are very loose and interpreted differently by various hardware manufacturers.

The most widely used wiring "standard" in LocalTalk networks is the set of construction rules published in Farallon's PhoneNET System. These rules aren't a published specification at all, but a system of guidelines. In the absence of a strict specification, some configurations of LocalTalk components from different manufacturers do work reliably. We'll explain these configurations later in this section

All of these criteria assume that the network hardware has no bugs. Bugs, of both the hardware and software varieties, are a fact of life for network managers. If a manufacturer has a large installed base, howeve, the bug may be known and the manufacturer's tech support department may tell you about it or another known incompatibility. Of course, some vendors are more candid about known bugs in their products than others.

One of the goals of the Apple Network Manager's Association (ANMA) is to create a society of network managers that can exchange candid knowledge of products, including bugs and incompatibilities as well as features and abilities.

Testing electrical continuity

The following section describes three tests of electrical continuity: a tone test, an ohm-meter test, and a Time Domain Reflectometer (TDR) test. Check electrical continuity when you suspect that something is physically wrong with your cabling, for example, when nothing appears in the Chooser or Inter•Poll lists.

All of the tests require some equipment; each can examine other aspects of your cabling besides its continuity. For example, the tone test lets you check that you've connected the proper conductors to your network wiring apparatus, and an ohmmeter can help you determine the length of the cable as well as the presence of shorts and opens. If you're fortunate enough to have a TDR, you can check all of these functions and much more.

The tone test for continuity. In twisted-pair wiring, a tone test can easily and quickly verify the electrical continuity of a circuit. A tone test set consists of two components: a tone generator and a tone receive. Tone test sets are inexpensive, usually around $100, and are a standard tool for telephone technicians. Some shopping advice: get a tone generator with a recessed on/off switch. If the switch isn't recessed, you may accidentally turn the tone generator on while rummaging through your toolkit.

There's no sound unless the amplifier is on, so you probably won't notice that the tone generator is on, but within a few days, the battery will lose its charge.

The tone generator is attached to the conductors at one end of the wire; you listen for the tone at the other end of the wire, usually using a battery-powered inductive amplifier or a lineman's "butt set," a telephone handset built explicitly for line testing. The only advantage of a butt set for a network manager is that the tone is heard through an earpiece instead through the speaker of the hand-held inductive amplifier. (The sound of the tone could be annoying in some office environments.) Most network managers purchase the simple inductive amplifier, rather than the more expensive butt set, for the receiver.

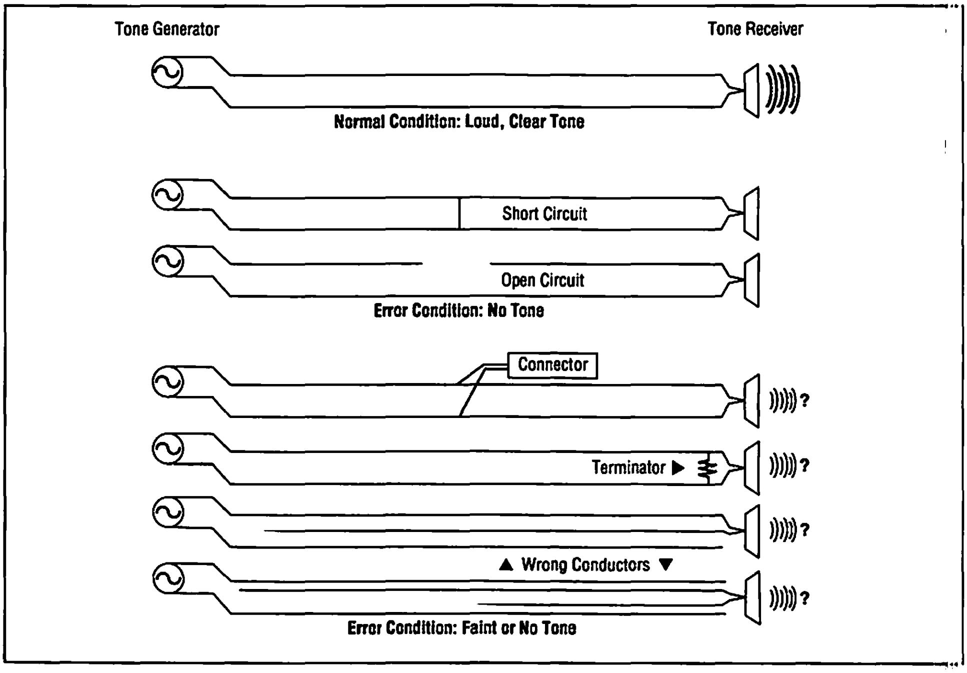


Figure 6-1. The tone test verifies electrical continuity.

Before you perform tone testing, remove all of the network equipment and conduct the tone test on a single pair of conductors with nothing attached to them. At one end of the cable, attach the tone generator to the conductors. Usually the tone is activated at the user's workstation and heard in the wiring closet. We find it useful to use a modular adapter, a tool that plugs into an RJ-11 or RJ-45 jack and allows you to select any two of the jack's conductors. In any case, be sure that you have solid contact with a bare metal surface, not to any insulation. Turn on the tone generator and verify that it's generating a tone by holding the receiver near the contact.

Next, go to the other end of the cable and hold the receiver so that the probe is very close to or touching the two conductors. A loud, clear tone heard through the receiver proves simple continuity. A faint tone or no tone can indicate many error conditions, as shown in Figure 6-1.

The ohm-meter test for continuity. You can conduct another test of continuity by using an ohm-meter. The ohm-meter test has advantages over the tone test because you can use it on either coax or twisted-pair wire; it can also detect the presence of a terminator or a network adapter.

As with the tone test set, you should prepare for the ohm-meter test by removing all of the network equipment. In the user area, either short the conductors (by connecting them together) or place a terminator at the end of the cable. The ohm-meter test will then take place in the wire closet.

Some shopping advice: you can purchase a digital ohm-meter that is adequate for the testing described in this section for about $35. The more expensive meters offer a higher level of accuracy and precision than necessary for the tests described here. If you have a little extra money, you may find it more useful to buy an extra set of probes. (We like the probes with a claw that firmly grabs onto the wire.)

A little math is involved in using an ohm-meter, and we've included some sample calculations below. The resistance of a wire is proportional to its length, but when you have a terminator on the line, you must also take into account its resistance. You must perform the math to determine whether your ohm-meter's reading is normal or not.

The first thing you should know is what kind of wire you are using and what its resistance is. If you are using solid copper wire, you can use the resistance values given below. The number shown for each gauge is the number of ohms that a wire 1,000 feet long should have. Because the relationship between distance and resistance is known, the length of the cable can be determined with an ohm-meter. Keep in mind, though, that since normal variations in manufacturing may affect the values you get, your determinations are approximate. Wire made by manufacturers in the United States is among the highest quality in the world, so if you're using wire from a U.S. manufacturer, your calculations will probably be within 5 percent of the actual value. The resistance values you can expect are as follows:

- For solid copper cabling at 68 degrees Fahrenheit, the resistance of 1000 feet of wire is 15.14 Ω for 22 AWG (American Wire Gauge), 25.67 for 24 AWG, and 40.81 for 26 AWG.

- The resistance values for Ethernet coax cabling varies with manufacturer and product number, but is generally 8 to 10 Ω for thin coax cable (10BASE-2) and around 1.5 Ω per 1,000 feet for thick coax (10BASE-5).

- The values for stranded cables varies even more greatly by product. The most variance is seen in voice grade telephone patch cord (not twisted), which can have resistance values that range from 100 Ω to 1000 Ω per 1000 feet.

- If you need the standard resistance values for the type of wire that you use, ask your cable supplier or get a copy of the manufacturer's technical specifications. You can also determine a working value for a particular cable by measuring the resistance of a known length of that cable.

When you know the standard resistance value for your wire, test its continuity and measure its length by shorting two conductors at one end and measuring the resistance between conductors at the other end of the wire. The resistance value that you measure is used for the length calculation.

An example of the length calculation is in order: Suppose you short two conductors of a 24 AWG wire and measure 10.7 Ω between the conductors at the other end of the wire (see Figure 6-2). This is shown schematically below. Since you know that 1000 feet of 24 AWG wire would have 25.67 Ω of resistance, you can calculate the wire's (round-trip) length by dividing the measurement by the standard value.

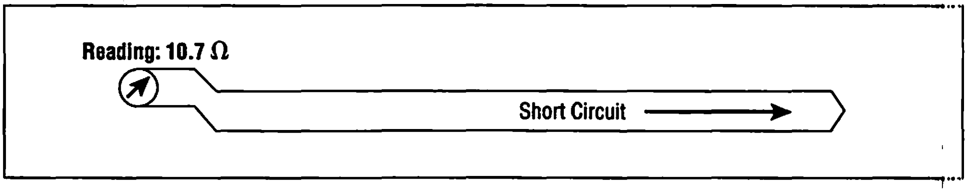


Figure 6-2. Measuring the resistance of shorted cable

In this example, 10.7 divided by 25.67, or .417, is the portion of the standard 1000 foot length that is in your cable. Multiplying .417 times 1000, you get 417 feet, which is twice the length of your cable. It's twice the length because you measured the resistance "there and back." The cable in this example is therefore approximately 210 feet long.

Here is the equation for wire length when you short the conductors:

$$\text{Distance} = (\text{Measured Resistance}/\text{Standard Resistance}) \times 500$$

If you have a terminator instead of a short circuit at one end of the conductors, you must take into account the resistance value of the terminator. An easy way to do this is to measure the resistance at both ends of the wire and calculate the distance of the wire from the difference between the two resistance measurements.

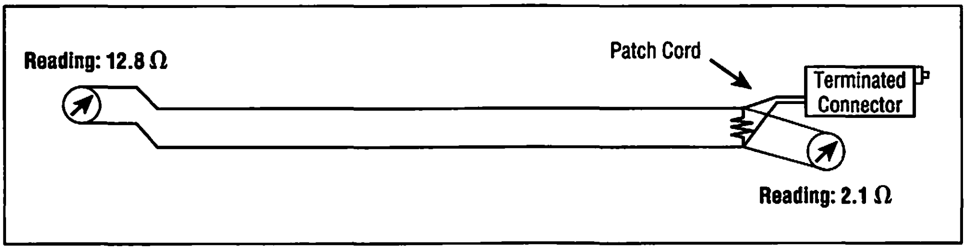


Figure 6-3. Measuring the resistance of terminated cable

If you attach a 120 Ω (nominal value) resistor as a terminator on one end of the wire used in Figure 6-3, the difference between the readings will still be 10.7 Ω, which is the value used to calculate the length of the wire.

The equation for wire length on a terminated cable is as follows:

$$\text{Distance} = \frac{(\text{Resistance at Open End} - \text{Resistance at Terminated End}) \times 500}{\text{Standard Resistance}}$$

In PhoneNET (LocalTalk) networks, the ohm-meter test is normally performed on circuits with the connectors and patch cord removed. Another approach is to leave them on the circuit. As in the previous example, the distance is calculated from the difference of the resistances. For our sample circuit, the differences between the resistance measurements should be about the same, 10.7 Ω, but the values measured will be much lower.

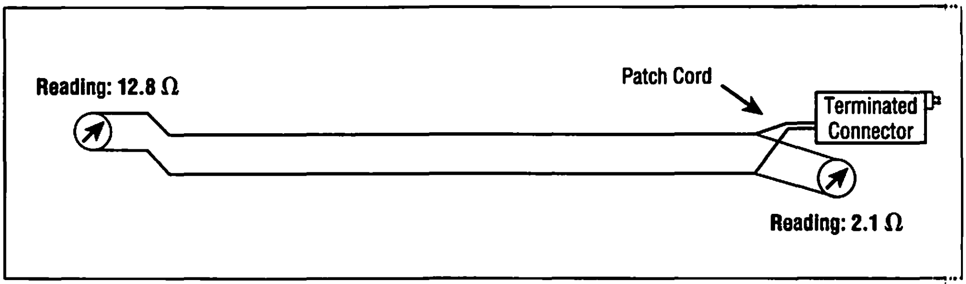


Figure 6-4. Measuring the resistance of a LocalTalk circuit with a connector

Looking at Figure 6-4, you may wonder why the mounted terminating resistor in the connector doesn't create a higher resistance on the line. The same would be true if you have a terminating resistor in your wall outlet. The reason is that a PhoneNET connector has such a low resistance that the terminator has no effect on the measurement. What you're measuring is the resistance across the poles of a transformer inside the connector, along with the line resistance of the patch cord. This is unfortunate, because it means that the ohm-meter test can't discern whether a terminator is in the circuit or not; you'll have to rely on your eyes for that.

TDR testing of data circuits. A good TDR is much more expensive than either an ohm-meter or a tone test set, but it also does much more. For example, Microtest makes a line of TDRs with a number of attachments for various testing situations. For general-purpose cable testing, they make a product called the NEXT Scanner, which costs about $1500. More expensive is a TDR that includes special features for 10BASE-T or Token-Ring networks. Some TDRs are highly automated and can check all of the pertinent cable functions for a specific use or automatically print the test results, taking most of the guesswork out of cable testing.

The first advantage of a TDR you'll probably notice is that it tells you, in English on an LED screen, all the numbers that you calculated above. The screen might say that your cable is "Open at 182 feet," for example. You won't have to make two resistance measurements or know the standard resistance values or perform any calculations to determine the length of your cable. Most TDRs also report the presence of shorts and terminators as well.

Although you won't have to worry about calculation errors giving you false length values, TDRs are slightly less accurate than ohm-meter tests at determining the length of solid conductor, twisted-pair wire. The reason is that a TDR calculates the length of a wire by measuring the round-trip time of an electrical pulse placed on one end of the cable. In twisted-pair wire, the velocity of electrical signals varies slightly from cable to cable, just like resistance values, but most TDRs use an approximate value that ignores these slight variances. (Some can be calibrated by entering the specific velocity of your cable, which may be available from the manufacturer.) For coaxial cable and stranded cable, however, the TDR is more accurate than the ohm-meter test for length measurement.

In addition to providing length measurement and telling you about shorts, opens, and terminators. TDRs (depending on the model) can also measure line noise, crosstalk, and signal loss, and, plot the impedance profile of the cable. For 10BASE-T networking, a TDR like the PairScanner can tell after just a couple of tests whether or not your circuit is suitable for the 10BASE-T signal—a real bonus when your cabling is less than first rate.

Because the capabilities and testing procedures differ on various TDR models, we'll omit procedural descriptions and let the preceding description of the instruments' capabilities suffice.

Check the error statistics. One advantage of having a wiring hub is that it provides some additional management and troubleshooting capabilities such as reporting error statistics on the packets it has processed. Not all these error statistics are available on all hubs, but you may find that some other network monitor, such as a protocol analyzer or traffic monitor, can provide these statistics if your hub can't. In addition, some routers and bridges also track packet errors.

> Note: Strictly speaking, error statistics are a function of the data link layer, but we've included them here because they directly indicate problems in the physical layer. As you read this section, you may notice frequent use of words like "usually," "sometimes," and "normally." That's because the relationship between packet errors and their causes isn't straightforward. If you notice a high level of packet errors, it can at best serve to direct your investigation in a certain direction. Don't immediately assume that a high level of errors is a sure indication of a physi-

cal layer problem; rather, use it as an indication of what kinds of problems to watch for.

In LocalTalk, the level of CRC (cyclic redundancy check) errors is the statistic most indicative of wiring problems. Normally, you should have a very low level of CRC errors—fewer than 5 in 30,000 packets—if you have any CRCs at all. If you have more CRC errors than this, you should probably check your wiring.

Usually CRC errors indicate termination problems, but CRCs can also be caused by other wiring conditions that create severe signal reflections, such as changes from shielded to unshielded wire or too many wire type changes, or a circuit with an intolerably high number of wire connecting devices (such as punch-down blocks and patch panels) between the hub and the last node.

In LocalTalk, the second most indicative statistic is the number of framing errors—overrun and underrun errors. These can sometimes be misleading, however, because they can be artificially created when the device noting the errors is running near the top of its processing capacity. For example, if you're running Farallon's TrafficWatch, which counts framing errors, on a Mac IIsi with all of TrafficWatch's display windows open and constantly updating, you're likely to have an artificially high number of framing errors (particularly overruns). The IIsi *falsely perceives* framing errors because its own processor can't stay in synchronization with the packet. There may or may not be anything wrong with the packet. Of course, the framing error could also be created by the same mechanism if the sending node is near the top end of its CPU capacity. In either of these scenarios, there's nothing wrong with the network, despite the fact that the monitor is showing a high level of errors.

Other, less indicative statistics in LocalTalk include a high level of packet fragments, sometimes caused by noise on the line, and "jabbers," sometimes caused by faulty hardware. All devices are capable of jabbering if they have a problem, but devices known to occasionally jabber are the LaserWriter Plus, LaserWriter IIg, and LaserWriter IIf. When a device is jabbering, you can usually clear the problem by restarting it. If the problem persists, however, it's usually necessary to replace the motherboard, and in rare cases, the entire device.

If you can't attribute framing errors to a shortage of processing power and your monitor shows that more than 3 percent of the total packets have framing errors (overruns and underruns), look for the same kinds of wiring problems as in CRC errors.

Generally, the wiring problems will be less severe than when the framing errors are accompanied by CRCs.

In Ethernet, the first thing to check is the Link Status LED on your Ethernet Transceiver. If you're in the wiring closet, there is typically a corresponding Link Status LED on the front panel of the hub. If the Link Status LED isn't lit at either end, it means that the hub and the transceiver are not in communication. This is a sure indication of a wiring problem. Make sure that all the plugs are firmly in their jacks and that all the conductors are connected to the right pins—you may have some crossed wires.

If the Link Status light is on, other physical layer problems may be indicated if any of the following packet errors are reported greater than 1 percent of the total number of packets: CRC, Loss of Phase Lock Loop, Fragments, or Missing Start Frame Delimiter.

Collisions—normal occurrences in Ethernet networks—are usually proportional to the level of traffic on the network, and are typically not the result of cabling problems. To keep collisions to a reasonable level, a practical rule of thumb is to keep your Ethernet traffic level no higher than 15 percent of capacity during the busiest hour of the day. An extremely high number of collisions can indicate an improperly terminated cable.

Gathering and reporting the wiring statistics mentioned above is a basic feature in most active wiring hubs. Some hubs have the useful ability to search the cable on a particular port and identify which nodes are connected. Perhaps the cable integrity was compromised by users adding or removing nodes without the assistance of the network manager. Sometimes the error statistics mentioned above are reported on a "per port" basis. This is also very helpful because it narrows your search for the problem.

Data Link Layer Troubleshooting

The data link layer's responsibility is to build a proper frame for the data that has been provided by the higher layers in the protocol stack. This frame is a kind of "package" for the data as it travels along the link, and it must be built according to the rules of whichever data link—Ethernet, LocalTalk, or IBM Token-Ring, for example—the node is connected to. The data link is also responsible for managing the physical layer as it delivers these frames from node to node. This includes managing the physical layer functions of signaling and receiving, gaining link access, and checking the physical integrity of incoming packets.

The question to ask at the data link layer is, can the data link layer reliably deliver packets from node to node?

Two kinds of tests

You can perform two kinds of tests to test the reliability of data link layer functions. The more practical of the two tests provides an empirical value of how well the packets are being transmitted based on how reliably packets are shuttled back and forth between various nodes on the data link. Like most network protocols, AppleTalk has an echo (or "ping") protocol that lets you test a point-to-point connection by sending echo packets to a node. When a node receives one of these echo packets, it simply returns (echoes) the packet back to the sender. The sender records whether or not the echo packet was returned and how long it took to receive the echo, then sends the next echo packet. We'll explain more about how to conduct the echo test toward the end of this section.

The second kind of test is more theoretical in nature. Instead of measuring the effectiveness of the data link is at carrying packets under a simulated network condition, it looks at the number of packet errors that the data link layer encounters. In some ways, this is a more thorough test of the link because some anomalies can be present on a link but are not always revealed by an echo test.

For example, a node may be intermittently transmitting nonsensical signals and wasting network bandwidth. Echo testing is a spot check; you don't normally test all the devices on a link. In the case of the jabbering device, unless you select it as the echo recipient for one of your tests, you probably won't notice that this device is faulty.

Monitoring data link errors

Several network management tools monitor and report data link errors. Among them are Farallon's TrafficWatch, protocol analyzers, and the network management software that runs on many wiring hubs. TrafficWatch and Mac-based protocol analyzers use the Mac's network hardware in "promiscuous mode." In promiscuous mode, the Mac's network hardware accepts and attempts to process every signal on the wire. Instead of rejecting signals with errors, a Mac in promiscuous mode keeps statistics on the errors. If there are signals on the wire that even remotely look like a packet, these tools will try to synchronize with the signal and read its data. Since promiscuous mode is contrary to the normal functions of the data link layer, a Mac running one of these tools is typically not responsive to AppleTalk while the testing is in progress.

Although the results of this kind of testing can be very useful, you must qualify the results before you can interpret them properly. Many variables affect the quality of the test and the meaning of the results. The requirement to read every signal on the link is simultaneously the strength and the weakness of this test. The strength is that the test may see signals that the Mac would otherwise reject and would be "invisible" unless you were using an oscilloscope. The main weakness of the test is that the Mac is barely capable of this rather intensive processing task and may show errors that don't really exist. Also, some types of data link errors are simply not sensible by a computer's network adapter (NIC) or can't be read because they occur during a time when the NIC isn't watching for signals.

In addition to a simple shortage of processing power and measurement precision, other factors may cause erroneous results. One such factor is that because the different tools that perform this test work differently, the tools may report different results when monitoring the same network. Some tools that report these data link values are, in our opinion, virtually worthless for this task.

The most reliable results are usually those reported by network hubs. In most cases, they have special hardware and circuitry for this task. For measuring LocalTalk errors with Mac-based software, the most reliable tool is also the oldest—Apple's AppleTalk Peek (we prefer version 3.1). Unfortunately, AppleTalk Peek isn't for sale (nor is it supported). It's available only on certain CD-ROMs sent to developers and consultants.

Next in order of usefulness and reliability are the Mac-based protocol analyzers, but then only when running on one of the faster Macintoshes (IIcx or faster).

Because of the inherent uncertainty of data link error data, network experts and manufacturers disagree over what is an acceptable level of errors. All networks, no matter how well built, occasionally experience some errant signals and damaged packets. The difficult part is to determine what rate of errors (number of errors/number of total packets) indicates a network problem.

In the next few sections we'll cover which errors are important to watch, what level of errors may indicate the *possibility* of a problem, and what kind of problem might be indicated, if there is one at all. You should view data link layer error statistics as a pointer for the direction of your investigation, not as proof of a problem.

Having noted these qualifications and disclaimers, let's look at some ways in which you can use data link layer error results to diagnose network problems.

Data link layer error results in LocalTalk

In LocalTalk, only one kind of data link error is meaningful for troubleshooting: the CRC error. A high rate of CRC errors implies that the received packet was altered in some way as it traveled along the wire.

The LocalTalk CRC itself is a 16-bit number that is calculated using the data in the packet as the seed of a mathematical formula. This mathematical formula is calculated by the node that sends the packet, and the CRC that results is affixed to the end of the packet. When the packet is received, the receiving node performs the same mathematical calculation on the incoming data and compares the resulting CRC to the CRC affixed to the end of the packet.

If the CRC calculated by the receiving node exactly matches the CRC that was sent on the end of the packet, the data received is probably correct. In fact, the probability is only 1 in 2^{16} (65,536) that two packets with different data could have the same CRC. If the CRCs don't match, then a CRC error is recorded.

Normally, the data link layer rejects packets with CRC errors. Later, when one of the higher layers notices that some of the data it expects is missing, it will ask its partner node to resend the missing data. The data link layer in LocalTalk, as in most network systems, is incapable of spontaneously requesting retransmissions.

Of the more popular LocalTalk hubs, the TurboStar (formerly made by NuvoTech, now made by Focus) makes the most useful measurements. The TurboStar tracks the number of CRCs for each of the 16 ports as well as the number of packets transmitted by the devices on the port.

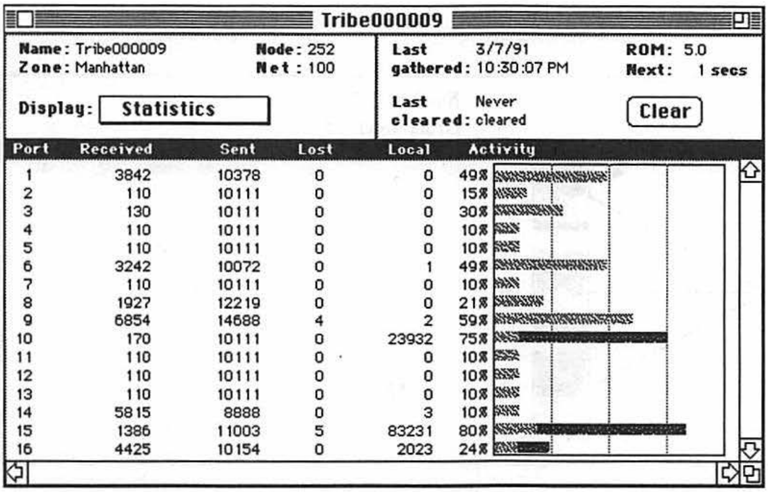
In LocalTalk, a high ratio of CRC errors to total packets typically means bad wiring conditions, most often improper termination. Non-CRS LocalTalk errors—overruns, underruns, length errors, and so forth—may result from causes other than an unhealthy network, and are not very useful for diagnosis.

The Tribe LocalSwitch and the Farallon StarController also track the number of errors per port, but they don't distinguish between the different types of LocalTalk errors. A high number of errors on a particular port may not necessarily indicate a problem because the errors reported may not be CRC errors. Still, it may be worthwhile to check those ports for wiring problems, either by using a Mac-based tool to check for CRC errors or with the echo testing methods (both techniques are described later in this chapter).

A rule of thumb for LocalTalk networks is that a wiring problem, usually improper termination, may be indicated when the error statistics of a network hub or AppleTalk Peek show a rate higher than five CRC errors per 10,000 packets.

If you see a high rate of CRC errors on a particular port, try to determine whether there truly is a wiring problem, and if so, what the problem is and where it is located. If you can't measure CRC errors per port, or are not using a hub, you can still use CRC errors as an indicator if you have AppleTalk Peek.

If you have Tribe's LocalSwitch, which doesn't distinguish between the different kinds of LocalTalk errors, investigate a port when you see more than five "lost" packets per 1,000 packets received. On Farallon's StarController, there's no useful rule of thumb, but watch for abnormally high values on a particular port.


Tribe000009

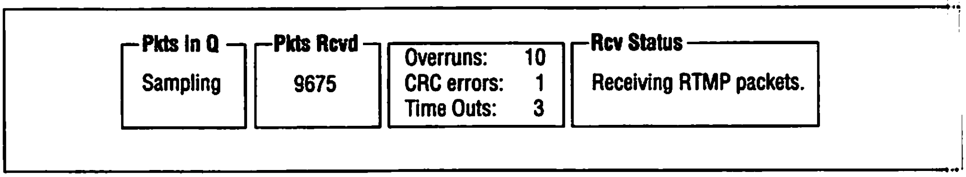
Name: Tribe000009
Zone: Manhattan
Node: 252
Net : 100
Last gathered: 3/7/91 10:30:07 PM
ROM: 5.0
Next: 1 secs
Display: Statistics
Last cleared: Never cleared
Clear

Port	Received	Sent	Lost	Local	Activity
1	3842	10378	0	0	49%
2	110	10111	0	0	15%
3	130	10111	0	0	30%
4	110	10111	0	0	10%
5	110	10111	0	0	10%
6	3242	10072	0	1	49%
7	110	10111	0	0	10%
8	1927	12219	0	0	21%
9	6854	14688	4	2	59%
10	170	10111	0	23932	75%
11	110	10111	0	0	10%
12	110	10111	0	0	10%
13	110	10111	0	0	10%
14	5815	8888	0	3	10%
15	1386	11003	5	83231	80%
16	4425	10154	0	2023	24%

Figure 6-5. The Tribe LocalSwitch's port statistics tell you how many packets were lost due to errors on the line. If you see more than five "lost" packets per 1000 packets received, you should investigate the wiring on that port. Port 15 is running very close to this limit.

A useful way to use AppleTalk Peek is to run it simultaneously on several nodes and compare each node's results to its geographical location. Just start AppleTalk Peek and let it capture the normal network traffic for a few minutes or until you've captured several thousand packets. If you're in a hurry, perform an action that will create a lot of traffic—for example, a file transfer. The AppleTalk Peek window will tell you how many CRC errors it found.

AppleTalk Peek gives you the option of also capturing LLAP (LocalTalk Link Access Protocol) Control packets. If you choose to capture LLAP Control packets, make sure that all Macs running this test are set to capture these packets.



Pkts In Q
Sampling

Pkts Rcvd
9675

Overruns: 10
CRC errors: 1
Time Outs: 3

Rcv Status
Receiving RTMP packets.

Figure 6-6. AppleTalk Peek's statistics tell you how many packets it received and how many of them had CRC errors. The values shown are an acceptable error rate.

Below are some examples of what kinds of results you might see and how you might interpret them. In each example, the CRCs shown are sample data from 10,000 packets.

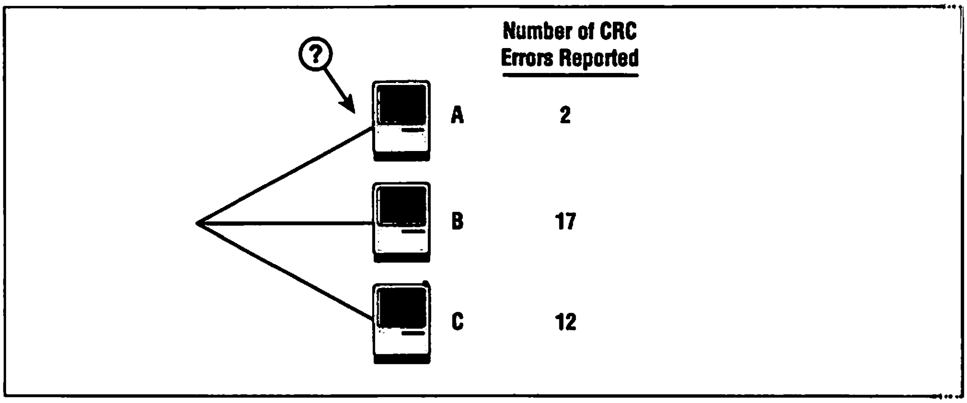


Figure 6-7. In this passive star topology, you would check whether Node A was properly terminated. Reflections from this node may be causing signal problems for Nodes B and C.

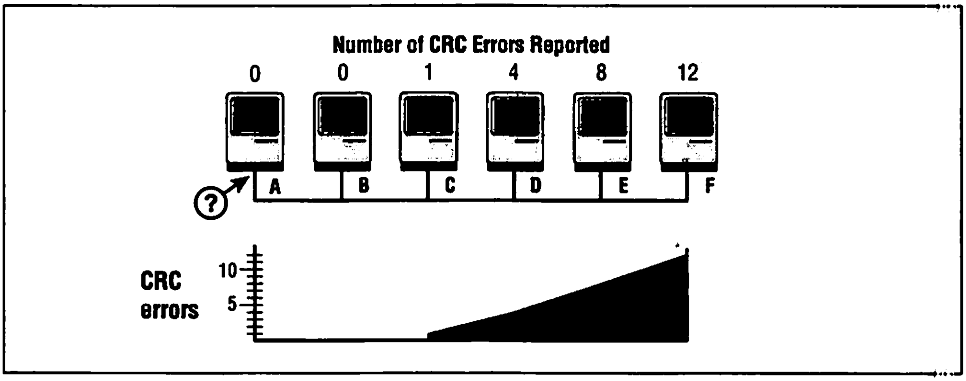


Figure 6-8. In this bus topology, you might suspect a missing terminator at the end of the bus near Node A, which could be causing the signal problems that grow worse as you approach the other end of the bus.

Sometimes the results are not so clear. In Figure 6-8, also a bus topology, only one of the nodes, Node D, shows an unacceptable level of CRC errors. In this case, you might suspect wiring problems in the vicinity of this node. Possible causes of the high error level might be bad connections, untwisted wire near electrical noise, changes in the wire gauge or type, or an excessive number of connections through patch panels or punch-down blocks. You might also have missing terminators at both ends of the bus. In this case, though, you would expect to see *some* CRCs at every node.

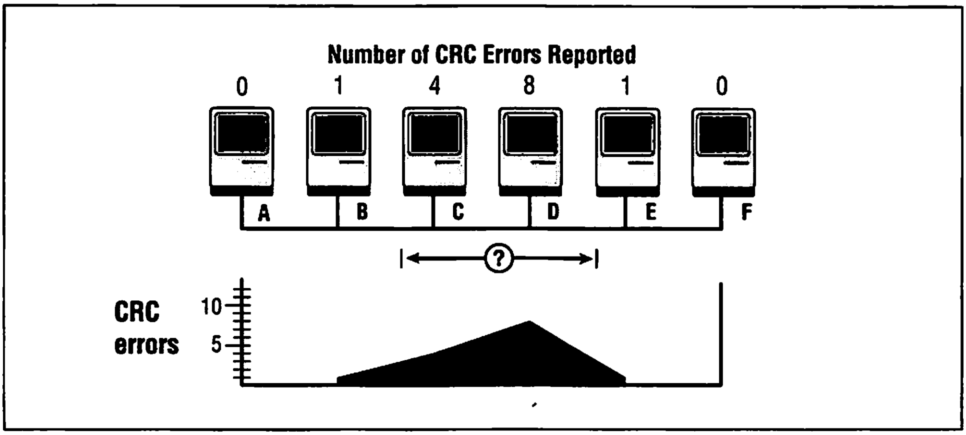


Figure 6-9. In this LocalTalk bus, you might suspect some bad wiring conditions near Node D, which reported a high level of CRC errors.

Measuring data link layer errors in Ethernet

In contrast to LocalTalk, when you measure the number of data link errors on Ethernet links you get more clear-cut meanings of errors. In addition, the number of errors reported is less subject to the kinds of variables we've discussed. But that doesn't necessarily make the interpretation of the test data any easier. Some of the errors clearly indicate a problem; others you can tolerate if they occur infrequently, like the level of CRC errors tolerated in LocalTalk.

On a coaxial Ethernet, such as a 10BASE-5 or 10BASE-2 Ethernet, every node on the segment can see an error that occurs. Protocol analyzers monitoring the segment can monitor and record many of the errors listed in the following section; some bridges and repeaters also have the ability to track errors.

On 10BASE-T networks, you must rely on the management features of the hub, since a protocol analyzer connected to a 10BASE-T port will see only the errors experienced by that one port. There is great variety in the management capabilities (and the cost) of hubs currently available. While 10BASE-T requires all hubs to notice most of these errors listed below, the 10BASE-T specification does not require hubs to report error statistics. This is typically done by the hub's optional, and sometimes very expensive, network management software.

Some 10BASE-T hubs, such as those made by Asanté and Dayna, are designed for minimum cost. They can have a price per port that is one-third of the price per port of a top-of-the-line hub decked out with all of its optional management software. To achieve this low cost, some of these hubs provide no management information except for the LEDs on the body of the repeater. Others provide just enough management software to include something about the importance of network management in their marketing literature.

These products are often good from a signaling point of view (they must meet the strict requirements of 10BASE-T), but they're not very helpful for diagnosis. With these hubs, you'll have to use some other means to diagnose the problem. If you're having trouble on a network with one of these hubs, you may have to resort to a "Random Mode" technique; simply cycle the power to the hub (turn it off and turn it back on almost immediately). This is often a good technique to try if you having trouble getting packets between the 10BASE-T ports of the hub and the AUI and/or BNC ports. Remember, these hubs can't tell you when there's something wrong with them.

Ethernet error types and their meaning

If you have a way to view the statistics for Ethernet data link errors, you can use this information to point your investigation in a fruitful direction. Here is a list of the error types most commonly reported, along with an explanation of their meanings and possible causes.

Link Integrity. Most 10BASE-T hubs and external transceivers, as well as many 10BASE-T cards, have link integrity indicators, typically LEDs. Both 10BASE-T hubs and transceivers send a link integrity pulse to each other every five seconds. The light on the transceiver indicates that the hub's link integrity pulse is being received by the transceiver, and vice versa. If basic electrical continuity exists (which can be confirmed by some other test, such as an ohm-meter or tone test) but the light isn't lit, something is preventing the transceiver and the hub from communicating. Make sure that the card's jumpers are properly set, the proper EtherTalk Network Extension is selected in the Network Control Panel, and that AppleTalk is turned on in the Chooser.

Also check the polarity of the wires. If the hub's link indicator is lit but the transceiver's is not, check the polarity of the transceiver's receive wires (pins 1 and 2 of the RJ-45 connector). If the transceiver's link indicator is lit but the hub's is not, check the polarity of the transceiver's transmit wires (pins 3 and 6 of the RJ-45 connector).

Transceiver Power. If your transceiver or card has a power indicator, it should be on if the transceiver is receiving power from your Mac. The checks are the same as previously described for Link Integrity.

Polarity. If you reverse the polarity of either the transmit lines or the receive lines, your Ethernet device will have difficulty communicating with other devices. Some hubs can automatically compensate for reversed polarity some hubs can spot it but can't correct it and some hubs can't even detect it. If the LEDs on the hub or transceiver indicate that the device is sending packets, look at those packets with a protocol analyzer. If they contain completely nonsensical data, then the transmit lines are probably reversed and your hub can't correct it. If the device is sending packets but seems to be ignoring packets that it receives, use that device or the wall outlet it's using to run a Mac-based protocol analyzer. If the receive lines are reversed, all of the packets that are displayed will contain nonsensical data.

Jabbers. Jabbers occur when a device's network hardware malfunctions and the device sends a continuous stream of bits. Ethernet repeaters, including 10BASE-T hubs, are required to detect a continuous transmission and shut off the port that is jabbering.

Auto-Partition. 10BASE-T hubs have the ability to automatically shut off a port if they sense a serious error condition on. One such condition occurs when a port experiences more than 30 consecutive transmit collisions or when a collisions occurs over a long period of time.

Some hubs have the ability to report auto-partitions through the use of their management software, or can indicate the auto-partition with an LED. This may require a Setup Mode check of all of the components as indicated in Chapter 4. Check wire quality, polarity and length, transceiver health and settings (particularly the SQE setting), the jumper settings on the Ethernet card, the health of the Ethernet card (using software), the Network Control Panel settings, the EtherTalk drivers, the AppleTalk version number, and the system software configuration.

Transmit Collisions. These happen when a 10BASE-T hub is sending a signal and senses a collision on one of its ports. In Ethernet, a tiny percentage of packets are expected to experience collisions. It's difficult to set a maximum collision rate because collision rate is related to utilization—that is, as the utilization of the link increases, not only is the *number* of collisions expected to rise, but the *rate* of collisions (the ratio of collisions to the total number of packets) is expected to rise as well. The expected level of collisions can be expressed only in terms of probability, and the probability must be related to the traffic level. When measured over an hour-long period, an Ethernet utilization of more than 15 percent is considered a high traffic density. When measured over a period of one second, utilization of more than 45 percent is considered high.

While there are no accepted limits for collision rates, you may want to investigate when the rate seems greater than normal. If you can't correlate it to a high traffic level, check for wiring problems that could result in excessive collisions, such as the coupling phenomena mentioned above or transient signals on the wire from other electrical devices—in other words, short events.

Receive collisions. These occur when the hub is *not* transmitting and receives a jam signal from a transceiver that is experiencing a collision. This should never happen on a 10BASE-T port, since there may only one device per port and a collision requires two devices transmitting simultaneously on the same wire.

If you do see Receive Collisions on a 10BASE-T port, it could mean that port is connected to more than one device. For example, it could be connected to another 10BASE-T hub that is sending a jam signal, or to a 10BASE-T to thin-net repeater where the thin-net contains several devices. If the port showing receive collisions is a pure 10BASE-T port with only one device, the transceiver on the user's end somehow sensed a collision and sent the jam signal. Check the wiring for coupling or other kinds of electrical noise.

Late collisions (also called Out of window collisions). Late collisions occur when a device receives a jam signal immediately after sending a packet. Ethernet specifications are structured to prevent late collisions. The Ethernet parameters that affect late collisions include the minimum Ethernet packet size (512 bits), maximum separation distance between any two nodes (2.6 km), maximum number of repeaters between nodes (4) and the maximum acceptable propagation delay through a repeater (about 4 bits). Late collisions usually suggest that one of these limits has been violated. They could also suggest defective hardware.

Runts (also called Fragments). Runts are usually the result of collisions, so the concepts stated for transmit collisions also apply to runts. If there are significantly more runts than collisions, then apply the following concepts for short events.

Short events (also called pygmy packets). These are signals that are shorter than 74 bits (7.4 microseconds) in duration. These might not be packets at all, but electrical noise on the wire resulting from fluorescent lights or an electric motor in a copy machine. It could also be coming from defective network hardware. Just as some kinds of hardware defects result in a jabbering node (continuous, meaningless signals), some kinds of hardware defects result in a node that periodically sends short signal bursts, which can be interpreted as a pygmy packet. Again, a few of these are tolerable, but if you are getting more than one short event per 1000 packets, you may want to test for electrical noise.

Data rate mismatch (also called Elasticity buffer error or buffer error). This kind of error indicates that the device sending the signal to the hub is not sending at the correct Ethernet speed. It's typically caused by defective hardware.

CRC errors. CRC errors are very similar to LocalTalk CRC errors. They indicate that the packet's data has become corrupted as it traveled along the network link. One CRC error per 1000 packets is tolerable, but if more than that occur you should investigate the physical layer. In coaxial Ethernets, check for violations of Ethernet's strict construction rules (number of nodes/segment, maximum segment distance, and so forth) as well as kinked or tightly coiled cable. In 10BASE-T, check for bad wiring or too many connection points between the hub and the transceiver.

Alignment errors. Alignment errors occur when the receiving node (or hub) suspects that it may have lost synchronization with the packet, either as a result of corruption in the packet or due to its own internal causes. The node or hub makes this assessment if the number of bits it receives is not evenly divisible by 8. Again, check for faulty wiring if there is more than one alignment error per 1000 packets.

Phase lock errors. These errors occur when the receiving node can't achieve synchronization with the incoming signal. Either the signal was irreparably damaged when it left the node (similar to a jabber), it was damaged during transmission (faulty wiring), or it isn't really an Ethernet packet at all (line noise). Check for these conditions if you're getting more than one per 1000 packets.

Using the echo test

Compared to looking at error statistics and correlating them to particular kinds of link level problems, the echo test is a very practical test; it tests whether a packet can reliably be sent between two points. We'll describe how the echo test can be used in other ways later, but data link layer troubleshooting is concerned only with the reliability of transmission between two points on the same link. On a good data link, for each echo packet sent to a distant node, one will be returned from that node. Echo testing tells you whether your network line works all the time, some of the time, or not at all.

If the echo test indicates that your network line works all the time, you can move your investigation up to the network layer. The drawback to the echo test, however, is that when your line works only some of the time or not at all, the echo test doesn't tell you much about the nature of what the problem might be. For that you'll have to rely on the methods concerning data link errors outlined above or the troubleshooting methods of the physical layer described in the previous section.

The echo test is built into several software tools, but we find that the most convenient tool is Apple's Inter•Poll (despite the fact that it hasn't been updated for many years). Inter•Poll uses AppleTalk Echo Protocol, which is incorporated into the system software of all Macs running System 6 or higher and is present in many other devices that use the AppleTalk protocols. Inter•Poll records how many packets were lost en route and keeps statistics concerning the round-trip delay time of the packets.

Inter•Poll gives you the ability to control four variables: the number of packets sent, the delay between the packets, the length of time to wait for the returning echo packet (timeout), and whether to send short or long echo packets. We use the same variables all the time for consistency: 100 long echo packets with no delay and a one-second timeout. This places about a 30 percent utilization on a LocalTalk link, depending on how fast the other node can turn the packet around and send it back. Some devices don't respond to echo packets because Echo Protocol is not implemented in their network software. An example is Apple's LaserWriter II series. For these devices, Inter•Poll allows you to send printer status packets instead of AppleTalk Echo packets. For other devices, you may not be able to perform this kind of testing at all.

Interpreting echo test results

The data points we rely on most heavily in the echo test are the number of packets that get returned and the average time of the round trip. Since we send out 100 packets, the number of packets successfully returned is the point-to-point reliability expressed as a percentage. The average round-trip time is a measure of the speed of the other device and the amount of processing time that it has available to answer the echo packets. Inter•Poll tells you the minimum, average, and maximum round-trip delays expressed to the nearest hundredth of a second. In some tests, there may be a considerable disparity among these three times. Generally, a wide range between minimum and maximum round trip delays indicates that the device is very busy.

On LocalTalk, you should get 100 percent of the echo packets returned every time. If you get 99 percent, repeat the test a few more times to see if the value of 99 was a one-time fluke or is consistent. If you get a high return percentage, but not 100, check to see if the other device is being overtaxed by its other processing demands. If it is, use a protocol analyzer to verify that the echo packets sent were in fact good.

If you can verify that a good echo packet traveled from the test node to the busy recipient node but no echo packet returned, you may be able to forgive a missing packet or two. If you see the opposite—a good echo packet being returned but not being counted by your testing device—then you may question whether your testing node is being overtaxed by its other processing chores. If so, you might try the test again on a faster Mac or one without a lot of extra processes running on it.

In circumstances where you cannot attribute lost packets to very busy devices, on LocalTalk or TokenTalk links, 100 percent of your echo packets should be returned every time. If this is not happening, check your wiring; there's a good chance that something is wrong. Over Ethernet or remote access links, use 98 percent as the good/investigate threshold.

Interpreting the echo round trip time

When an echo packet takes a long time to be returned, it's usually because the other machine had other tasks to accomplish before it could get around to returning the echo packet. Some applications, including Claris' FileMaker, may place such heavy demands on the device that echo packets may be held for a considerable length of time before they are returned. Generally, on a LocalTalk link, you should expect the average round trip time to be between 0.03 (on a fast Mac) and 0.07 seconds (on a very slow Mac like a Mac Plus). The maximum time is generally less than 0.15 seconds, unless the machine is busy, in which case the maximum time can stretch to as long as 0.75 seconds. Even then, only one or two packets out of 100 should take such a long time.

If you're echo testing a link using a very busy node as your recipient, you may get less than perfect test results when returned packets exceed the recommended one-second timeout value. In such cases, increase the timeout value.

Using the Tribe LocalSwitch in your network will cause a slight delay as the packet is switched from one port to another. With no other traffic on the network but echo testing, the LocalSwitch will typically require about one-half of a millisecond to perform the switching action, during which time the first several bytes of the packet are buffered inside the switch. This delay, however, is an order of magnitude less than the precision of Inter•Poll (which measures time in hundredths of a second) and is also slightly below the level of precision of a protocol analyzer.

On a LocalSwitch network with other traffic on it, the LocalSwitch might buffer the entire packet if the port to which the echo packet must switch is busy with a packet of its own. In this case, there might be an additional time delay because the LocalSwitch stands by, holding the packet in memory until the port is clear. This can add approximately 0.02 seconds to the transmission, in either direction—the outgoing or the returning echo packet—or in both directions. This time delay, probably a rare occurrence, is within the measuring ability of Inter•Poll and will affect the test results.

A very busy node can also affect the test in one other way: its data link layer may reject some packets if it perceives that a packet is damaged because one of the two nodes involved in the test didn't have enough processing time to adequately read the packet when it arrived.

You'll certainly want to increase the timeout value if you're testing a remote access link or across a low-speed link (less than LocalTalk speed) between distant sites. A value of three seconds should be sufficient if there are no routers between the testing node and the recipient node. The time value you get will depend mostly on the speed of the remote link.

A special consideration for LocalTalk

One of LocalTalk's chief design goals was to be a very inexpensive system. All its electronic components were relatively unsophisticated and inexpensive, even in 1983 when they were selected for the design of the original Macintosh. As a result, LocalTalk is a very loose specification compared to most other Data Link protocols.

LocalTalk uses a collision avoidance technique to arbitrate the use of the link. Part of this mechanism is that each data packet on LocalTalk is preceded by two LLAP Control packets that establish whether the sender should place its data packet on LocalTalk. Just prior to sending the data packet, the sender sends a Request to Send (RTS) packet to its intended receiver and expects, in return, a Clear to Send (CTS) packet back within a very short time; 200 microseconds (µsecs) is the maximum that a device is required to wait for the CTS to be returned. For most devices, 40 µsecs is a typical time.

When the sender receives the CTS, it assumes that the LocalTalk link is continuous—that the intended node is on the link and ready to receive the packet. Only then does the sender place the data packet on the wire.

This type of exchange can be compared to everyday speech:

Joe:	*Hey, Mike!*	(A Request to Send)
Mike:	*What, Joe?*	(A Clear to Send)
Joe:	*There's a meeting at two o'clock.*	(The data)

In this exchange, Joe is confirming that Mike can hear him and that Mike is ready to hear what Joe has to say. If Mike's Clear to Send occurs an hour after Joe's Request to Send, however, Joe may have given up on talking to Mike.

Although there is some tolerance for variations from the 200-microsecond limit, some LocalTalk devices take a very long time to return the CTS and may exceed the sender's patience. The sender might give up and try again later by sending a new RTS, waiting again, giving up, and so on. Since the time necessary to return the CTS in all devices varies, some of the tardy device's CTS packets may make it back in time and some may not. This certainly affects the apparent performance of the device and can also affect the echo test results.

The RTS/CTS mechanism precedes the echo packets as they travel in both directions across a LocalTalk link. If you're testing the link to a device that takes a long time to return a CTS, you may get poor echo test results in spite of the fact that the link is good.

Identifying this special timing problem in a device is difficult because the times involved are shorter than the precision of any protocol analyzer or computer-based network management tool. While a protocol analyzer can see that a CTS was returned and no data packet followed it, no protocol analyzer can tell you whether there was a 200- or 700-microsecond gap between the RTS and CTS. That type of information is available only by measuring the time between packets with an oscilloscope.

If you don't have a scope, then create a three-node LocalTalk daisy chain with the testing Mac, a protocol analyzer, and the device to be tested. If you make this daisy chain with only a few feet between nodes, you effectively eliminate the possibility of a bad link. Run the echo test and capture the packets, setting your filters to capture LLAP Control packets as well as data packets. If you have missing echo packets, look through the trace to see whether or not a CTS was returned.

Network Layer Troubleshooting

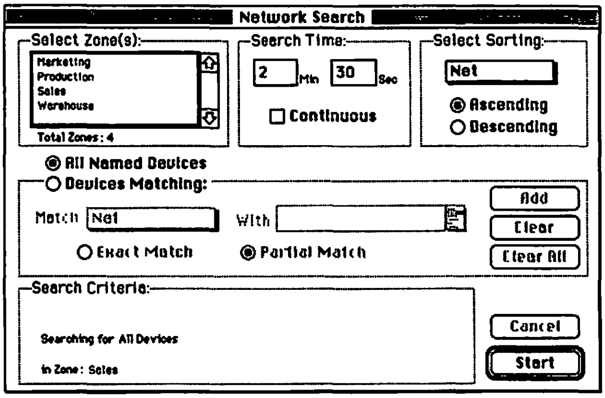
At the network layer, you must determine if the routers are working properly. The tests that we'll run at the network layer deal with two main questions: Are all the routers that you expect to see in your internet functioning? and Do all of the routers have the proper routing and zone information?

Both of these questions are easy to state. In a small internet of four or five routers, they're also fairly easy to answer. But in a large internet with fifty or more routers, neither is particularly easy to answer directly or conclusively. There are tests, however, that can gather the circumstantial evidence to help you make a nearly sure determination.

You can answer the first question, concerning the existence of the routers, by testing to see whether all the networks and zones expected in the internet actually do exist. This testing involves looking for named services in those networks and zones. You can answer the second question, concerning router configuration information, by using various network management tools such as RouterCheck and PacketSend to actively investigate the information held in the routers network and zone tables.

Verifying the existence of zones

A very simple way to check for zones is to open your Chooser to see if you have a zone list at all. This zone list verifies that there's at least one router on your network and that you can communicate with this router. In a small internet, it's easy to count the small number of zones in this list and check for misspelled or duplicate zone names. Be aware, however, that this is only one router's opinion of what zones are available in your internet. Other routers could have different opinions, so you may want to open the Chooser in other networks to check the other routers.


Network Search

Select Zone(s):
Marketing
Production
Sales
Warehouse
Total Zones : 4

Search Time: 2 Min 30 Sec
□ Continuous

Select Sorting: Net
● Ascending
○ Descending

● All Named Devices
○ Devices Matching:
Match Net With
○ Exact Match ● Partial Match

Search Criteria:
Searching for All Devices
in Zone : Sales

Figure 6-10. Inter•Poll's Network Search window tells you how many zones are in your internet and lists their names.

If you have long list of zones, the Chooser is not a very effective tool. It doesn't tell you how many zones exist, and counting a long list of zones in the Chooser isn't easy. A better method of verifying your zones is to run Apple's Inter•Poll.

When you first enter the program, you'll see the Network Search window, where you select which zones you want to monitor. In this window, as shown in Figure 6-10. Inter•Poll also tells you how many zones are in your internet and lists their names. The zone list is printable, for easy comparison to previously printed lists. To print the zone list, make sure that the Zone List checkbox is checked in the Print dialog box.

Verifying the existence of networks

Verifying the existence of networks is a little harder than verifying the existence of zones using Inter•Poll. Here, if you know that the zone Sales consists of networks 3, 146, and 20503, you can search the Sales zone for services. Then check to make sure that you see devices from all of these networks in Inter•Poll's resulting list. If you do this for all of the zones, you can verify all of the networks, as long as all of those networks have at least one service that Inter•Poll can locate.

Neon Software's RouterCheck provides a much easier way to perform this task. After getting your initial router list, double-click on one of the routers to bring up the Query window. In the Query window, ask the router for its list of routes (networks). RouterCheck will list all networks the router has in its Routing Table. It will also tell you how many networks are on the list. Perform this check on several of the routers in your network to verify that they all provide the same answer.

You may also want to check the log kept by some of your routers. These may include entries such as "Network 23 went down at 12:23:46 AM on Saturday 2/17/91." Different routers keep different information in their logs (if they keep logs at all), but this information may provide some additional helpful information.

Once you've verified the existence of the zones and networks in your internet, you're ready to move on and investigate your routers' configuration information. We discuss this matter in great detail in Chapter 8, under "Finding and Correcting Router Configuration Problems."

Transport Layer Troubleshooting

The question to answer at the transport layer is, Are the messages getting through the internet? Testing the data link layer's function of sending packets from one node to another on the same link, echo testing can also test the transport layer's ability to manage the reliable delivery of packets through the internet.

The method that we'll use is called the Progressive Echo Test (PET), a series of echo tests to devices at different locations around the network. A network manager can use PET to see how the reliability and speed of packet delivery varies with location. Figure 6-11 shows a typical PET scenario.

Next we'll look at an automated approach to monitoring the "reachability" of devices and services using a "sentinel" network management tool like the AG Group's NetWatchMan or Caravelle's NetWORKS.

Using the progressive echo test

Suppose a user is complaining about the poor reliability of his or her connection to an AppleShare server located in another network. He or she might report that the connection is dropping occasionally or seems very slow at times.

Dropped connections are usually caused by unreliable packet delivery. AppleShare servers and clients don't always have data to send to each other, and in the absence of data, each side will periodically send packets to each other to verify that

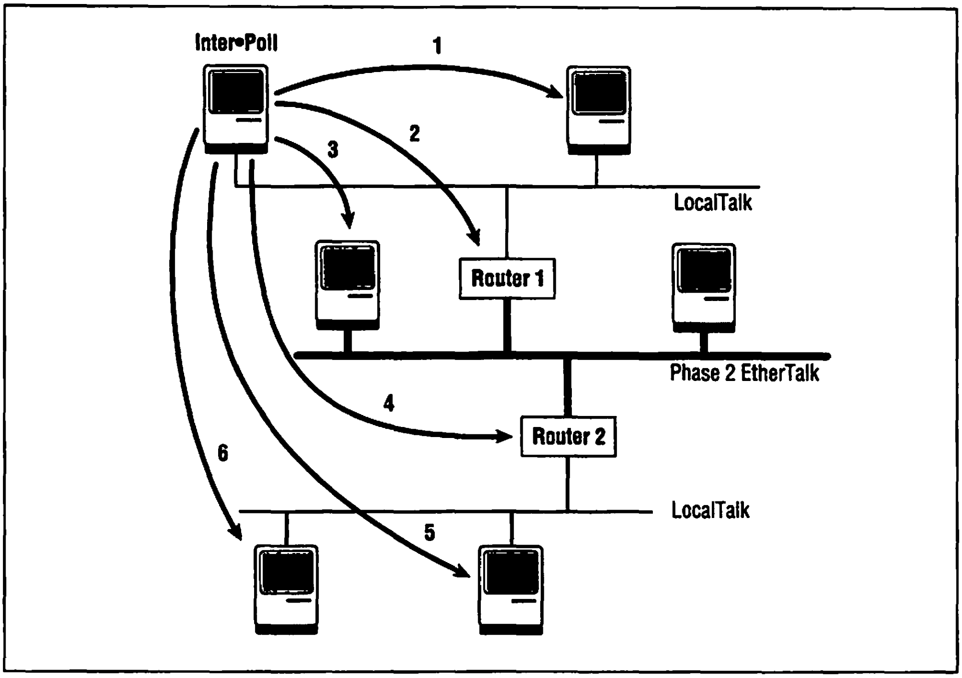


Figure 6-11. The Progressive Echo Test is a series of echo tests to test the reliability and speed of packets sent through the internet.

they're both still on-line and that the connection between them still works. When either side stops receiving these packets for a period of time, the connection is dropped. If both devices continue to function, a dropped connection usually means that the packets weren't getting through the network, at least not for a time long enough to cause the connection to drop.

Slow performance indicates slow communication between the two devices. Slow communication can have many causes. Your investigation must try to discover what conditions in the internet are causing the slowness and/or poor reliability. Here are some possibilities:

- The link that the user is on is unreliable or very busy.
- The link that the server is on is unreliable or very busy.
- One of the intermediate network links along the path is unreliable or very busy.
- One of the routers along the path is unreliable or slow.

- The internet routers have unreliable routing information.
- The server is very slow due to some cause that has nothing to do with the network.

A useful way to begin your investigation of this problem would be to conduct a PET from the complaining user's Mac to the server. Your testing should include all the pertinent intermediate points between the user and the server. Look for variance in the reliability of delivery and the time necessary for the echo packets to make their round trip among the tests; this process will usually give you some insight as to why the connection is being dropped and how best to continue your investigation. For each of the test points chosen, record the reliability, average time, and maximum time as indicated by Inter•Poll.

Use the same values for the variables of number and type of packets, delay, and timeout. Remember to clear the results of each test before beginning the next test. As mentioned earlier, we always send 100 long echo packets with no delay between packets and a timeout of one second. See the information earlier in this chapter under Data Link Troubleshooting, for more information about the echo test.

It may sometimes help to record the number of hops that the return packet took on its way back to the Mac running the echo test. The number of hops is the number of routers that the packet was routed through. Look for any variation in the hop count: that is, did some of the return packets show three hops and others only two?

Interpreting the PET hop count results

Before we begin discussing the reliability and time numbers, let's discuss the hop count. Normally, the hop count isn't a particularly interesting aspect of the test results. In most cases, the hop count will be constant for every packet in the test. In Figure 6-11, for example, you would expect a hop count of 0 for every packet in Tests 1 and 2, a hop count of 1 for every packet in Tests 3 and 4, and a hop count of 2 for every packet in Tests 5 and 6. If you get anything else, you need to do some investigation; the variation may show something wrong with the configuration of your internet routers. Even with router problems, however, a hop count discrepancy is rare. The following is a valid rule about the hop count in the echo test:

In any one echo test, the number of hops should be constant for every returned echo packet if:

A. the echo responder is on a network with only one router, or
B. the echo responder is running a Phase 2 network driver.

If A is true and the hop count shows variation, then you may suspect that there is some kind of routing problem that affects more than one router. In Figure 6-11, this sort of situation occur if the returning echo packets in Test 5 showed both two hops and three hops. The echo responder is handing its packets to Router 2 (first hop), which is sometimes sending the return echo packets to Router 1 (second hop) and at other times sending them to some router (second hop) that forwards them to Router 1 (third hop). In other words, Router 2 isn't sure about the best way to get back to the network that the Inter•Poll node is on. If that's true, then you probably have serious router configuration problems and should thoroughly check your routers as described in Chapter 8.

If B is true and the hop count shows variation, there may not be a serious error, but it's still worth investigating. Test 3 in Figure 6-11 corresponds to Condition B, where the echo responder is on a Phase 2 EtherTalk network and should be using a Phase 2 network driver. Nodes running Phase 2 network drivers keep a "Best Router" table in their memory that tells them, for each network they communicate with, the best router to use to send packets to that network. The node determines which router is the "Best Router" by remembering which router forwarded the packet to it when it received packets from other networks.

On receiving a packet from Network 23, for example, the node would remember that the packet had been forwarded by Router 8. Whenever this node needed to send a packet to Network 23, it would ask Router 8 to forward the packet along the correct path. In AppleTalk Phase 1, there was no best router table, and nodes sending a packet to a different network would simply use the last router from which they received any packet.

If Test 3 is to an AppleTalk Phase 2 node there should only be one hop, because that node should have a Best Router Table listing Router 1 as the best way to get back to the Inter•Poll node's network. If any of the returned echo packets come back with a hop count other than 1, you'd know that the Best Router algorithm isn't working correctly. At this point, you'd need to figure out why.

The best way to determine that is with a protocol analyzer connected to the Phase 2 EtherTalk network. Run Test 3 again and capture all Echo packets (filter on DDP Type 4) and watch the path that the packets take to and from the echo responder to see where else they're being sent to pick up the extra hop. Carefully check the configuration for the router that accounts for the extra hop; see if it's in line with the rest of the internet's routers.

Interpreting the PET time and reliability results

Once you've compiled the test results, you'll find it useful to graph the results to show the trends in the data. The interesting portion of the data is where the graph suddenly changes. In the data in Figure 6-12, for example, the round trip time graph changes very dramatically, nearly doubling between Test 5 and Test 6. The reliability data changes between Test 2 and Test 3, changing from perfect reliability to partial reliability.

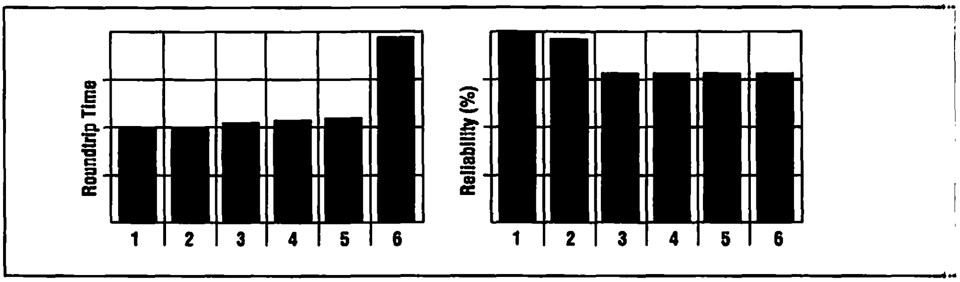


Figure 6-12. Graphing the data from a PET helps you see where the data changes suddenly.

The time data shows that the echo packets in Test 6 took a very long time to return. The two most likely causes are either a high level of activity on the server or a congestion of packets on the link. Test 5, which involves the same links as Test 6, shows that the speed of the links is not the controlling factor, so it must be that the server is slow, right?

Before you jump to that conclusion, repeat Test 5 and Test 6 to see if you get the same results. There's always the chance with the echo test that a temporary condition on the network will greatly affect the test. Every time you think you have the "Aha!" data, it's a good idea to repeat the test just to confirm your conclusions.

If your second batch of tests confirms the data in the first, you could conclude that the server is burdened with a lot of tasks and takes longer to respond than other nodes. If, in the second set of tests, Test 5 and Test 6 take about the same time to return packets, you can conclude that the server's activity is sporadic and you'll want to repeat Test 5 and Test 6 several times to get a feel for how often the server is busy and what effect the server's workload has on its responsiveness. Since some kinds of servers can tell you how busy they are, you may be able to correlate your data to the server's own assessment of its activity.

In the PET data graphed in Figure 6-12, the dramatic change in reliability occurred between Test 2 and Test 3. There are two possible causes for this change: the router is unreliable, or the Ethernet is unreliable.

There are many ways to find out which element is at fault. First, look at the statistics in the router's log to see if it dropped any packets due to data link errors or because its buffers were full. Different routers have different abilities to gather and display statistics, but this kind of information can sometimes tell you whether the router is the culprit.

You should also check whatever Ethernet statistics you have access to, whether taken directly from a hub or gathered using a protocol analyzer. You may be experiencing high numbers of errors. The error statistics would have to be somewhere in the range of your echo tests (5 percent to 10 percent errors) to indicate Ethernet reliability as the culprit. However, remember that the time over which you gather the Ethernet statistics and the time over which you perform the echo test should be the same if you want to correlate them to each other. Hub statistics might show the data accumulated for several days or weeks, and the unreliable conditions you're noticing may be transitory.

If you don't have access to Ethernet statistics, you can stick with the PET method and perform a seventh test, as shown in Figure 6-13. Here, you perform the echo test between the nodes on the Ethernet to get a reading on the reliability of the Ethernet alone.

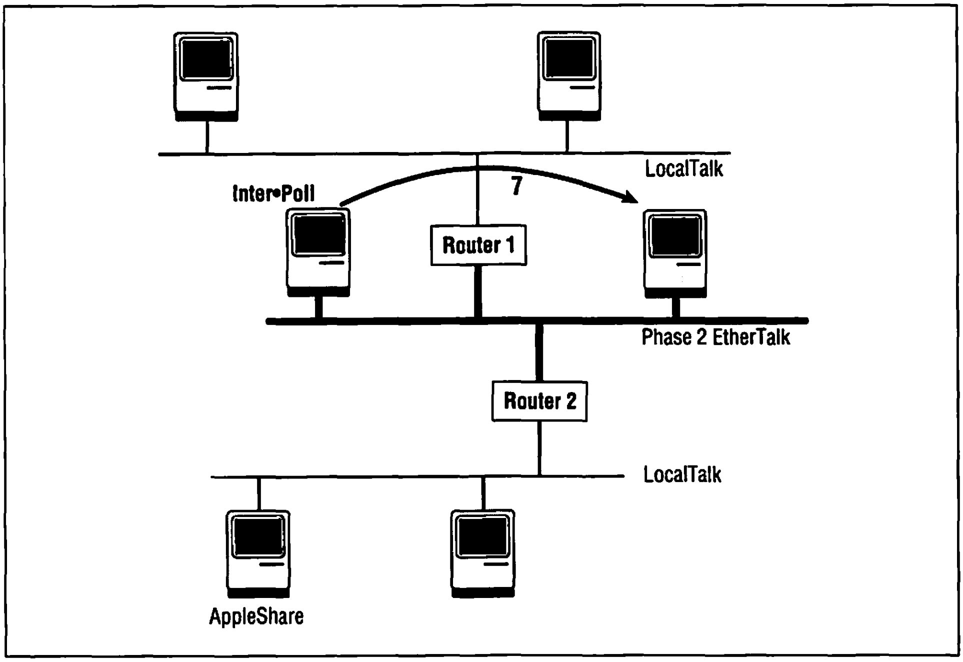


Figure 6-13. Test 7, when added to the data from previous testing, will show whether Router 1 or the Ethernet is unreliable.

If Test 7 shows 99 percent reliability or better and the data from Tests 2 and 3 are repeatable, then you can conclude that your router is having reliability problems. If the Ethernet shows reliability in the low 90s, then you can conclude that the router is fine and that the Ethernet needs some investigation.

Automating reliability checks

NetWatchMan from the AG Group and NetWORKS from Caravelle can continuously monitor the "reachability" of named services in your internet. You use these products by building a list of the named services available in your internet and then selecting the ones that you want to monitor.

For each service you select for monitoring, you can specify how frequently you want to check its reachability and what actions you'd like the program to take if it can't find the service. Your choices include playing an appropriate sound, making a log notation, displaying a dialog box, calling your beeper, or sending e-mail to the appropriate persons (assuming that the electronic mail still works at this point). You can also have these software tools monitor the appearance and disappearance of zones, among other things.

Both of these programs make their reachability determinations from a simple yes/no viewpoint. They provide no statistical data other than when they first notice the service is missing and when they first notice it is back on line. While this infor-

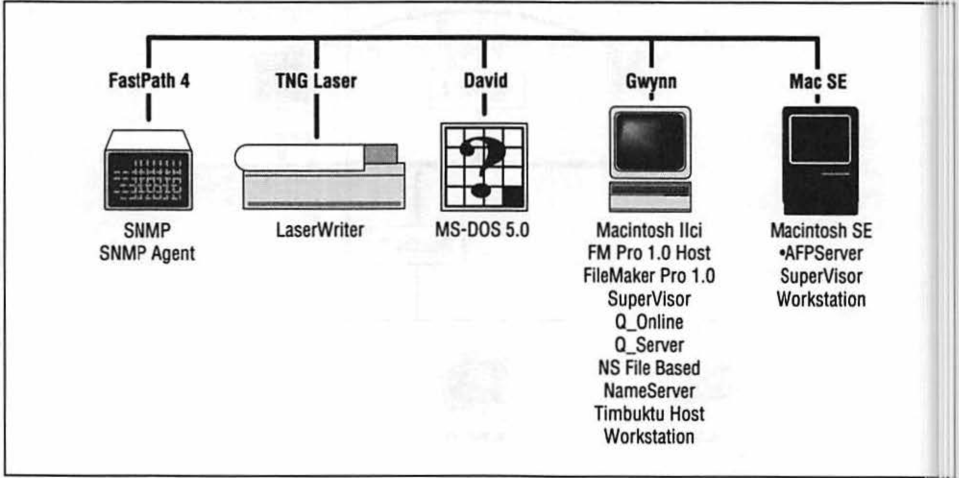


Figure 6-14. NetWatchMan, made by the AG Group, monitors the critical services in your internet and notifies you whether they are reachable or not. The darkness of the Mac SE indicates that it isn't currently reachable.

mation won't usually help you to diagnose why the node wasn't reachable, these programs are useful as an early warning system for network problems. In the best cases, you may be able to notice problems before your users do. When these programs start making their sounds or calling your beeper, you can immediately begin your diagnosis and apply your remedies.

Session Layer Troubleshooting

The purpose of the session layer is to manage the resources necessary to maintain a relationship between two computing devices. This includes establishing or allocating those resources at login, providing login and logout services, managing the mechanics of passwords and privileges, and providing any required session maintenance, such as verifying the connection between the client and server.

The question to ask at the session layer is, Does the server have the correct configuration? Session layer problems usually stem from a server-client incompatibility in which one expects the other to have some resource or characteristic that isn't actually present.

Symptoms of session layer problems include:

- some users can log into and use the service but others can't
- the server doesn't allow services that it should allow
- the server crashes or hangs unpredictably
- the server is slow at times
- the server drops connections arbitrarily

Typical session layer causes of those symptoms include:

- the server has software and/or hardware conflicts
- the server and client system software may not be compatible
- the server software may be in conflict with the client's application software
- the user having trouble isn't properly authorized to use the server

Session layer troubleshooting proceeds through a checklist of questions that verify various aspects of session layer functions. You may have considered some of these questions already when you were trying to discover the cause of the problem (in Hunch Mode, perhaps), but now you need to ask them in a methodical way. Here is the checklist:

1. Make sure the service is advertised.
2. Make sure the service can communicate.
3. Make sure the server's version is correct.
4. Make sure the server's system configuration is all right.
5. Make sure the server allows the access you need.
6. Make sure no previous users have damaged the server.

Special considerations of the session layer

The term *server,* as it's used in this section, refers in a general sense to any device that offers a resource or data service to other devices through a network-based connection. Basic server concepts are explained in more detail in Chapter 7, "Network Process Descriptions." Examples of servers are LaserWriters, AppleShare servers, shared modems, SNA gateways, a Timbuktu host, multiuser databases, and remote access servers, to name a few. *Client* refers to a device or software process attempting to use the server.

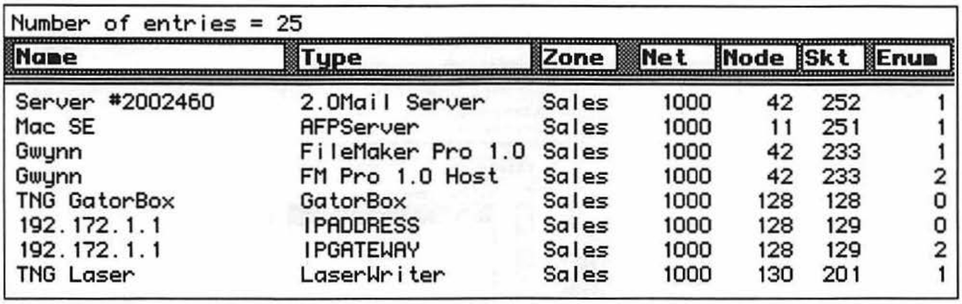
For each device in the preceding list of examples, there exists a group of potential problems peculiar to the server function that it performs. In fact, each type of server has its own list of common problems, mistakes, and misconceptions. In addition, each of several manufacturers and models for each of the server functions in the list possesses its own set of idiosyncrasies. Layer-oriented troubleshooting, which ignores much of the individuality of the service, is still useful, in part because it ignores the idiosyncrasies and concentrates on the functions.

Session layer troubleshooting primarily involves running through the items in the checklist, asking yourself various questions and verifying that certain conditions were met and certain questions answered. Occasionally in this section, we'll use techniques borrowed from process-oriented troubleshooting; we might peek inside a packet or two to ascertain a software version or to find out what methods of user authentication the server offers.

Checklist item #1: Make sure the service is advertised

If you've been following the procedures of layer-oriented troubleshooting, you should have already established that the server is on-line, that it's reachable, and that it responds properly to echo packets both from its own link and from other points in the internet. Since some routers hide services in one direction and not in the other, you should also make sure that the user isn't trapped by a router's security mechanism.

Still, take time to verify that it's advertising the correct service type name, using Inter•Poll (see Figure 6-15) or any similar tool. AppleShare servers should be advertising AFPServer, LaserWriters should be advertising LaserWriter, and so forth. It's possible, even if the node is on and on-line and can send and receive echo packets, that it isn't currently offering the service that you desire.



Number of entries = 25

Name	Type	Zone	Net	Node	Skt	Enum
Server #2002460	2.0Mail Server	Sales	1000	42	252	1
Mac SE	AFPServer	Sales	1000	11	251	1
Gwynn	FileMaker Pro 1.0	Sales	1000	42	233	1
Gwynn	FM Pro 1.0 Host	Sales	1000	42	233	2
TNG GatorBox	GatorBox	Sales	1000	128	128	0
192.172.1.1	IPADDRESS	Sales	1000	128	129	0
192.172.1.1	IPGATEWAY	Sales	1000	128	129	2
TNG Laser	LaserWriter	Sales	1000	130	201	1

Figure 6-15. Verify that your desired service is advertised. This CheckNet list, sorted by type, shows that Mac SE, an AFPServer, is advertised properly.

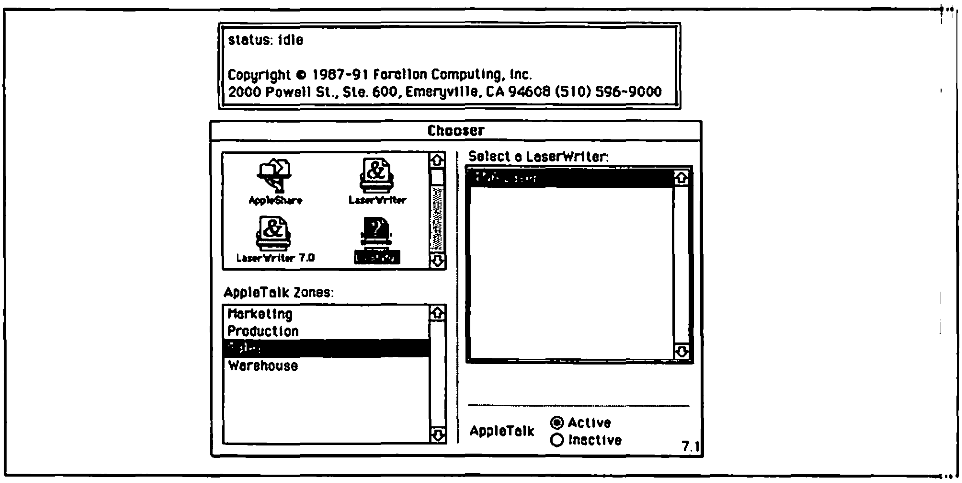
If you don't see the service you want but the node itself is on-line, find out why it isn't advertising the service. In dedicated AppleShare servers, perhaps the server program isn't started up. For System 7 File Sharing, perhaps Sharing has been turned off in the Sharing Setup Control Panel. Perhaps your LaserWriter has been given the Exit-Server command, such as a user might perform to reset the LaserWriter or change the configuration of its permanent RAM. Perhaps the Timbuktu host with which you wish to connect has Guest Access turned off.

If you have a nondedicated server running on someone's Mac, maybe that person has done something to cause the server to stop. For example, if a user running a Microsoft or QuickMail server switches the network connection from LocalTalk to EtherTalk, the server will shut down. It won't start up again after the switch is made, nor will it start up again if the user switches back to LocalTalk. It will only start up again when the user restarts his or her Mac.

Checklist item #2: Make sure the service can communicate

Even after you've verified that the service is being advertised, check to see if the service can communicate. There are several ways to do this. The simplest method is to find out if other users are currently using or can use the service. You might try the service from another workstation or ask other users if they've used the service recently or are currently using it.

Some servers—LaserWriters, for example—can give status reports over the network, even to devices that aren't users of their service. Apple's LaserWriter Font Utility and Farallon's LWStatus have the ability to ask LaserWriters for a status report and display the results.



status: idle

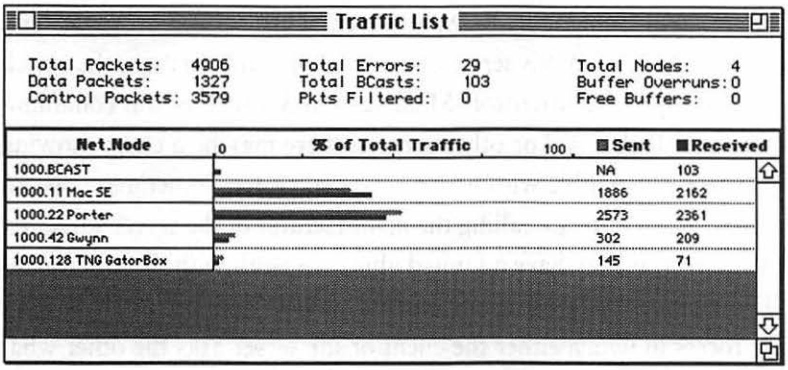
Copyright © 1987-91 Farallon Computing, Inc.
2000 Powell St., Ste. 600, Emeryville, CA 94608 (510) 596-9000

Figure 6-16. Farallon's LWStatus can display the current status of a LaserWriter. This readout proves that the service is not only available, but is also able to communicate.

You can also use a protocol analyzer to verify that the service is communicating. Watch a user as he or she tries to log in and see whether the device responds at all.

If you're not comfortable with a protocol analyzer, try using a traffic analyzer like Farallon's TrafficWatch to see whether other devices are communicating with the node you're testing. Although you can't be sure that the traffic you're seeing is coming from the service you want, you can tell that some communication is occurring and with whom. With either a protocol analyzer or a traffic analyzer, it's best to be connected to the same physical network as the service that you're trying to troubleshoot.

Some servers allow a limited number of users (perhaps only one) at a time to access their services. You can also use a traffic analyzer like TrafficWatch to find the identities of the users currently enjoying the server's attention.

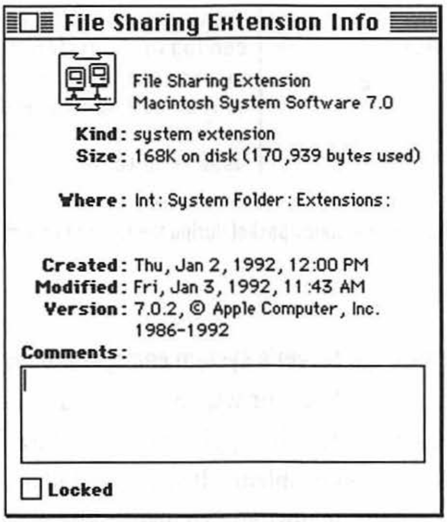


```
Traffic List

Total Packets:    4906    Total Errors:    29    Total Nodes:      4
Data Packets:     1327    Total BCasts:    103   Buffer Overruns:0
Control Packets: 3579    Pkts Filtered:  0     Free Buffers:    0

Net.Node               0   % of Total Traffic   100   Sent    Received
1000.BCAST                                              NA      103
1000.11 Mac SE                                          1886    2162
1000.22 Porter                                          2573    2361
1000.42 Gwynn                                           302     209
1000.128 TNG GatorBox                                   145     71
```

Figure 6-17. TrafficWatch shows that Mac SE is communicating heavily with Porter and somewhat less with Gwynn.

Checklist item #3: Make sure the server's version is correct

Verify that the version of the client's software is compatible with the version of the server's software. If the service runs on a workstation, you can check the software version directly. Figure 6-18 shows the Get Info window from the File Sharing Extension which provides the AppleShare server functionality for System 7 Macintoshes. LaserWriters indicate which version of PostScript they're using on their Start Page.



File Sharing Extension Info

File Sharing Extension
Macintosh System Software 7.0

Kind: system extension
Size: 168K on disk (170,939 bytes used)

Where: Int: System Folder: Extensions:

Created: Thu, Jan 2, 1992, 12:00 PM
Modified: Fri, Jan 3, 1992, 11:43 AM
Version: 7.0.2, © Apple Computer, Inc. 1986-1992

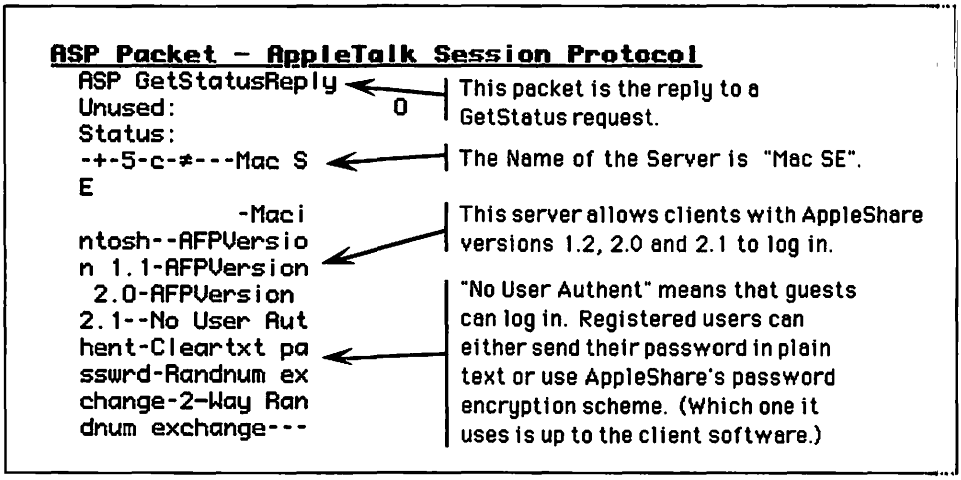
Comments:

☐ **Locked**

Figure 6-18. Services that run on a workstation may easily reveal which software version they are running.

It may not be apparent from Figure 6-18, however, which version of AppleShare client software is compatible with this server software. For many services, the match must be exact. For example, for Microsoft Mail, version 3.0 servers will communicate only with version 3.0 clients. For other servers, there may be a chart showing which client software is compatible with which server software. Sometimes you can find this out by trial-and-error or by calling the manufacturer of the server software.

Some clients and servers may have a limited ability to work with dissimilar software versions. If you happen to be using one of these services, there is usually a stage during the login process in which either the client or the server asks the other what software version it's using or is capable of interacting with. This query is often visible in one of the packets. Figure 6-19 shows many aspects of the server in a status packet that it sent to the client during login.



```
ASP Packet - AppleTalk Session Protocol
  ASP GetStatusReply
  Unused:                       0
  Status:
  -+-5-c-*---Mac S
  E
                -Maci
  ntosh--AFPVersio
  n 1.1-AFPVersion
   2.0-AFPVersion
  2.1--No User Aut
  hent-Cleartxt pa
  sswrd-Randnum ex
  change-2-Way Ran
  dnum exchange---
```

This packet is the reply to a GetStatus request.

The Name of the Server is "Mac SE".

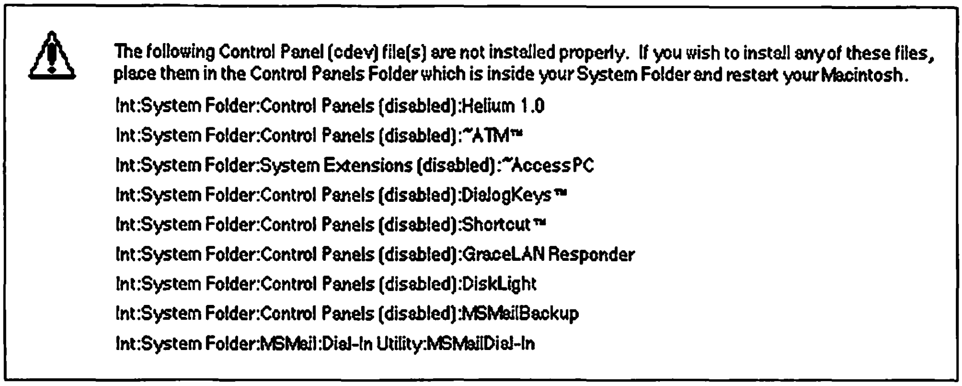
This server allows clients with AppleShare versions 1.2, 2.0 and 2.1 to log in.

"No User Authent" means that guests can log in. Registered users can either send their password in plain text or use AppleShare's password encryption scheme. (Which one it uses is up to the client software.)

Figure 6-19. AppleShare servers send a status packet during the login to inform the client of its login options.

Checklist item #4: Make sure the server's system configuration is okay

Check the server device to find out whether it has any potentially troublesome incompatibilities. On Macintoshes, the application Help! by Teknosys is very useful in spotting system configuration problems. It scans a Macintosh's complete configuration and checks what it finds against an exhaustive knowledge base of known conflicts and bugs. The figure below shows a small excerpt from a Help! analysis. You might also run the various virus checkers and disk analyzer programs available.



The following Control Panel (cdev) file(s) are not installed properly. If you wish to install any of these files, place them in the Control Panels Folder which is inside your System Folder and restart your Macintosh.

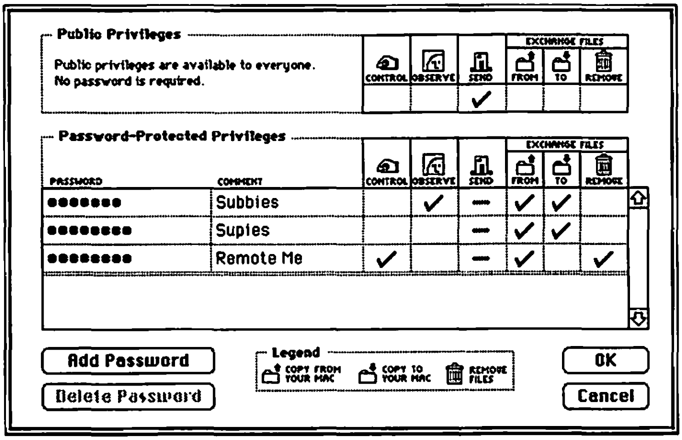
```
Int:System Folder:Control Panels (disabled):Helium 1.0
Int:System Folder:Control Panels (disabled):~ATM™
Int:System Folder:System Extensions (disabled):~AccessPC
Int:System Folder:Control Panels (disabled):DialogKeys™
Int:System Folder:Control Panels (disabled):Shortcut™
Int:System Folder:Control Panels (disabled):GraceLAN Responder
Int:System Folder:Control Panels (disabled):DiskLight
Int:System Folder:Control Panels (disabled):MSMailBackup
Int:System Folder:MSMail:Dial-In Utility:MSMailDial-In
```

Figure 6-20. Help! lets you know what configuration errors may be present in a Macintosh.

Checklist item #5: Make sure the server allows the access you need

AppleTalk services, more than those of any other network system, allow all users the same opportunities. Everyone using a LaserWriter, for example, has equal status with regard to privileges and priorities; it's a simple first-come, first-served device. Only a few services actually check to see who the user is and allow access according to the user's identity.

Even if the server meets all the previous tests, it may not offer all its services to all users. Check the server's rights and privileges configuration. Figure 6-21 shows the configuration for Farallon's Timbuktu.



Public Privileges

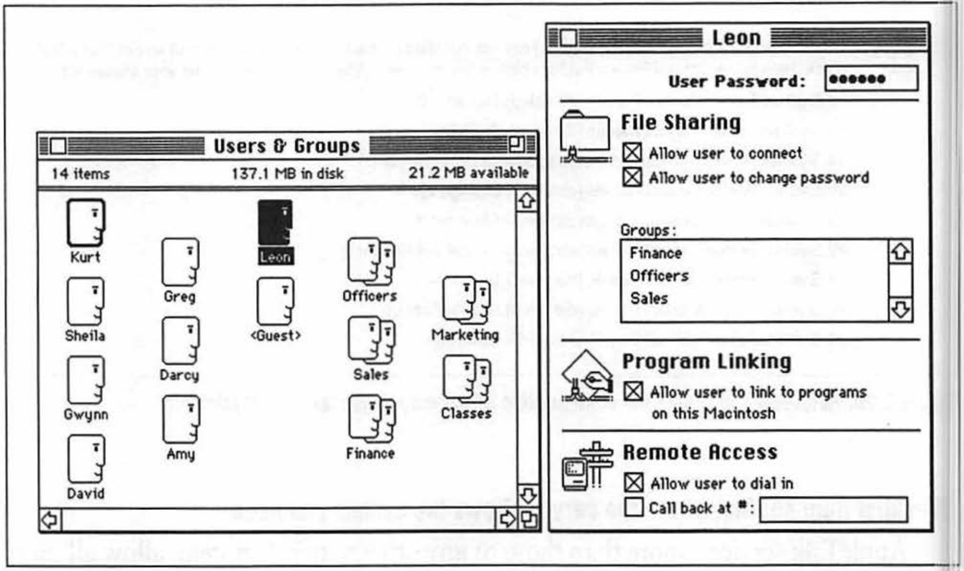
Public privileges are available to everyone. No password is required.

CONTROL	OBSERVE	SEND	EXCHANGE FILES FROM	EXCHANGE FILES TO	EXCHANGE FILES REMOVE
		✓			

Password-Protected Privileges

PASSWORD	COMMENT	CONTROL	OBSERVE	SEND	EXCHANGE FILES FROM	EXCHANGE FILES TO	EXCHANGE FILES REMOVE
●●●●●●●	Subbies		✓	—	✓	✓	
●●●●●●●●	Supies			—	✓	✓	
●●●●●●●●	Remote Me	✓		—	✓		✓

Figure 6-21. Farallon's Timbuktu allows its users to set different access privileges for different classes of guests.



14 items 137.1 MB in disk 21.2 MB available

Leon

User Password: ●●●●●●

File Sharing

☒ Allow user to connect

☒ Allow user to change password

Groups:
Finance
Officers
Sales

Program Linking

☒ Allow user to link to programs on this Macintosh

Remote Access

☒ Allow user to dial in

☐ Call back at #:

Figure 6-22. System 7 File Sharing lets users define who can use their Macs and for what purpose.

Checklist item #6: Make sure no previous users have damaged the server

Some servers may be rendered useless by a user action that the server wasn't prepared to handle; for example, when a user logs out improperly when using a shared gateway. A gateway, by definition, performs a translation between AppleTalk and another protocol at the session layer. A gateway typically has an AppleTalk session between itself and a client workstation, and a session using a different network protocol, such as SNA, TCP/IP or LAT, between itself and a (non-AppleTalk) host.

An example of this type of arrangement is a gateway to an SNA environment. Communication between a Mac workstation and the SNA gateway it's using is carried through the network by AppleTalk protocols. The SNA gateway also has, on the Macintosh user's behalf, an SNA connection to a mainframe. The gateway translates the Mac user's actions into mainframe commands and the mainframe data into Mac screen commands.

A difficult circumstance arises if the Macintosh user suddenly terminates the AppleTalk half of that arrangement without closing down the gateway's connection to the mainframe. Under certain circumstances, the gateway-mainframe connection may remain intact (but unusable) and may block future Mac clients from using that gateway. Although the user may be able to successfully make the Mac-gateway part of the connection, the gateway-mainframe part of the connection is unusable because of the previous user's action.

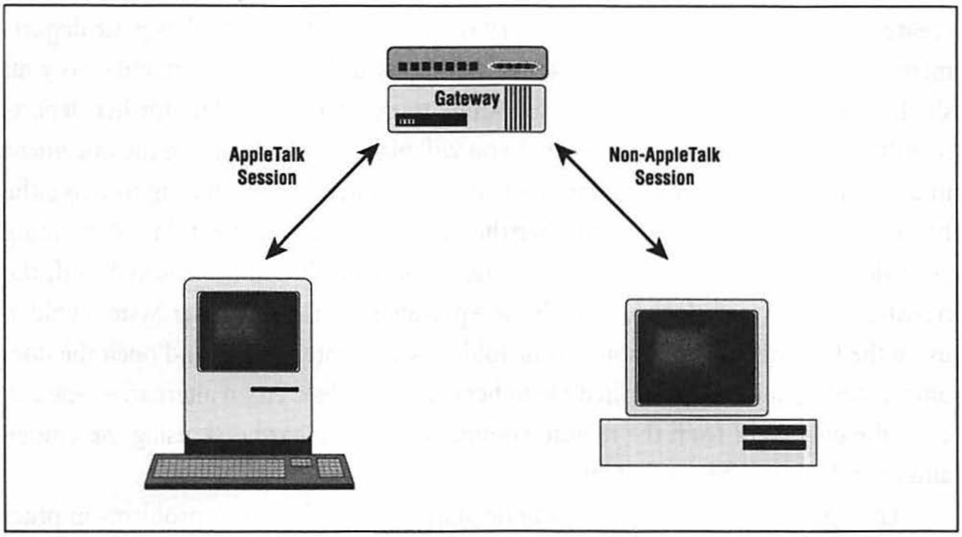


Figure 6-23. Gateways are particularly susceptible to unusual user actions that may impair their ability to function for subsequent users.

In a circumstance like this, there are typically two alternatives. Using another path to the mainframe, you may be able to forcefully terminate the gateway-mainframe session and open it again for a new user. The second alternative is to restart the gateway, which also forcefully closes its connection to the mainframe.

Presentation and Application Layer Troubleshooting

The presentation layer *presents* the data and commands of a specific computer system in a manner that can be understood by other devices on the network. This can be either easy or complex, depending on the nature of the types of data, applications, files, and file systems involved. Because this section focuses on troubleshooting the presentation layer, it assumes that you have already established that the layers below have sufficient vitality and reliability; you can move bits between the system and establish and maintain sessions between the devices. The task that now remains is to make sure the data is accurately and completely represented when it arrives. The question to ask at the presentation layer is, Do the two devices agree on how the data is represented?

The least complicated communication between two systems generally occurs when the systems are very similar. The highest degree of similarity is when a device running an application wants to communicate or trade information with another device of the same type running another copy of the same application. For example, suppose you create a document with Microsoft Word on your Mac that describes your department's monthly activities. A coworker in a different department sees it and calls you; she likes your report so much that she wants to use it as a template for her department's activity report. You tell her that you will place a locked copy of the document in a particular folder on your hard disk and use System 7 File Sharing to make the file available to her. You then give her the necessary access to the folder containing your document. If your coworker also uses a Macintosh and Microsoft Word, the transfer is very straightforward. With the AppleShare Extension in her System Folder, using the Chooser she can mount your folder as a remote volume and open the document, edit it, and save the edited file to her own hard disk. As an alternative, she can copy the document from the remote volume to her own hard disk using the Finder, and then do what she wants with it.

The application layer is easy to understand and rarely causes problems in practice from the network troubleshooter's point of view. It is simply the software that is used directly by the user. When a file is copied from a server to a local hard disk, the Finder is the Application layer. When a Microsoft Word file residing on a server is edited, Word is the application layer. The most common problem encountered at the application layer is user misunderstanding as to how the software works.

Operations at the presentation layer

Using the scenario above, let's look at how the presentation layer allows communication to take place. First, both machines must have appropriate configuration of software and hardware. The proper extensions must be loaded into each Mac, and there must be sufficient memory to allow the system, application, and extensions to operate. These extensions must also not conflict with other system additions on either Mac that might hinder the exchange.

Second, the two devices must have a way of representing files in the language of a common file system. With the devices properly configured, your folder and its contents can be represented to your coworker's Mac as a remote volume with the AppleTalk Filing Protocol (AFP) as the file language. Your Mac's native file system, the Hierarchical File System (HFS), its data structures, and command language are translated into their AFP equivalents and sent across the network, then are immediately translated back into HFS once the file arrives at your coworker's Mac, as shown in Figure 6-24.

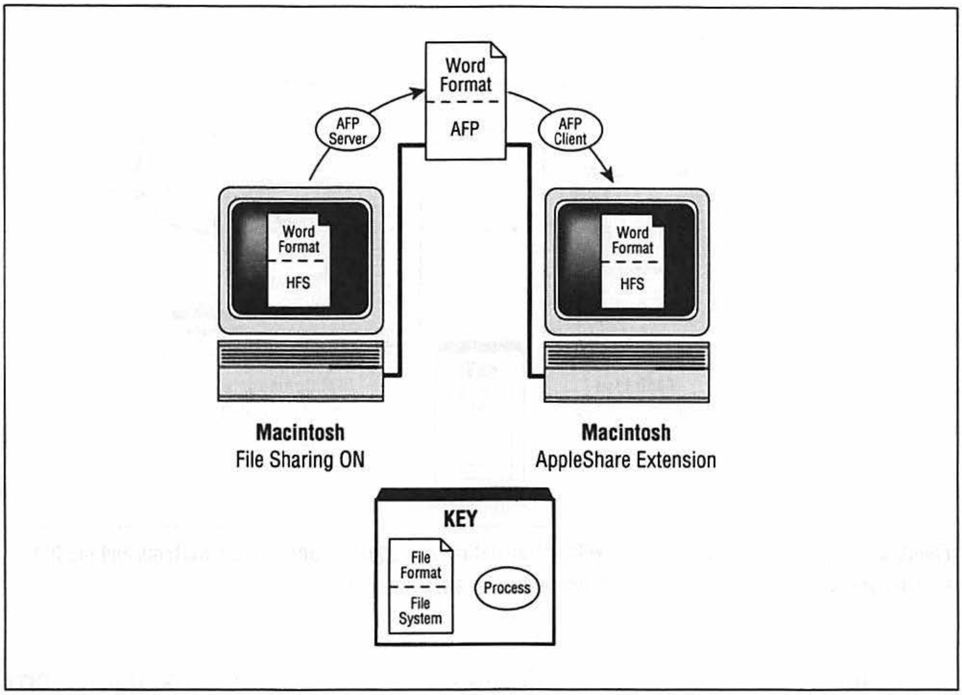


Figure 6-24. When two Macintoshes share a file, the Macs use Apple's network file system, AFP. Because both are running Microsoft Word, no file format translation is needed.

Third, the applications used on the two devices must be able to find a common file format. In this case, since your coworker is editing the file using Microsoft Word, the same application that you used to create it, there's no need to change the file format. If your coworker was using WordPerfect on the Mac, you could use WordPerfect's import capability to read the Word file. If your coworker was using WordPerfect on a DOS machine, you might make the transfer by exporting your Word file to RTF (Rich Text Format), transferring it to the PC, then importing into WordPerfect. Finding a common file format becomes more complicated as you move away from mainstream computer types and applications. You might find it difficult if your coworker is using a Tandy TRS-80, for example. Figure 6-25 shows file sharing with several layers of translation.

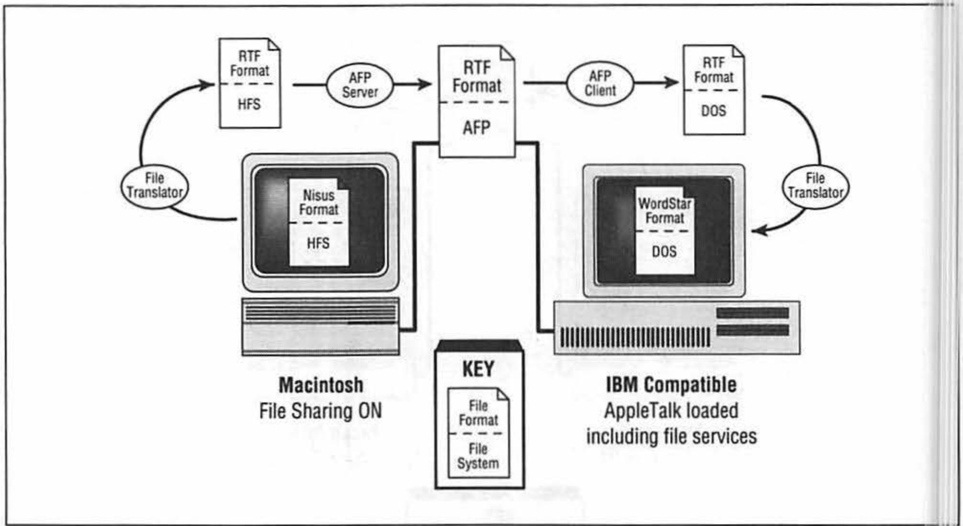


Figure 6-25. In this example, several levels of translation are required. Both the file systems and the file formats are dissimilar, and a common denominator for each must be found.

Although the file is now physically present on your coworker's hard disk, there might yet be difficulties in the way the data is represented that might keep your coworker from using the file successfully. The two devices must also have in common the various components that are implicit within the data. For example, suppose that because you enjoy working with graphics, you used Deneba's Canvas to render your company's logo with your department name incorporated into the design. If your coworker does not have Canvas or another application that can edit this graphic, she'll have trouble modifying your design to indicate the name of her department. And since your rendition of the company logo uses the Futura font (with a point size of 45), unless your coworker has an outline font of Futura in her system, the logo might look quite different when printed from his or her Mac.

Suppose that one of the things that your coworker especially liked about your report was that you included an appendix listing all your company's current activity codes with a short description of each code. Because this information changes frequently, your appendix is actually a subscription to a report from an Omnis database that gets updated automatically each time you open and print the document. Up until now, it hasn't mattered whether your coworker was using System 7, but if she wants to use this live link to the report, she'll need access to the folder containing the name of publisher of the subscription.

When we introduced this scenario, remember that we said that this was a simple case because you and your coworker are using the same kind of system and the same application. As you can see, however, it can quickly become complicated depending on the nature of the data. If your coworker were a DOS or UNIX user, your problems would be even greater, because there would be more chance for disagreement among the various elements.

Troubleshooting system configurations

Troubleshooting system configurations is as complex an activity as troubleshooting network problems. While it's not feasible to provide a comprehensive guide to this subject in the context of this book, several aspects deserve mention because network managers are often called upon to troubleshoot computer systems in the course of troubleshooting network problems.

One of the aspects of the Mac that its users like is its ability to be personalized and enhanced. This is accomplished through the use of files known as System Additions, which include System Extensions, Control Panels, Start-Up Documents, Fonts, and Sounds. Although these additions can be very useful, they modify the "pure" Mac system configuration in various ways and can potentially create problems for other applications and additions. For example, a modification by one addition may conflict with the modification made by a different addition or with the needs of an application. If programmers never made mistakes, these conflicts would never arise, but in this world conflicts between system additions occur from time to time. The DOS equivalent of an addition is called a TSR (terminate-and-stay-resident) program, and much of the following information applies to them as well.

Some additions are made to extend the Mac's capability. For example, the MacTCP System Extension gives a Mac the ability to use the TCP/IP protocol suite, and DAL gives a Mac the ability to access databases that speak this particular query language. Hardware drivers like those used for CD-ROM drives and pressure-sensitive graphic tablets are also in this category.

Other additions, particularly Control Panels, give the user some control over Macintosh functions. Some of these control critical aspects of the Mac, such as the Memory Control Panel, which allows the user to select either 24-bit or 32-bit addressing, change the size of the RAM cache and select whether he or she wants to use virtual memory. Other Control Panels, such as Views, Labels, or Color, control aspects of the Mac's display elements. Some of these Control Panels are frivolous, like Sound-

Master, which among other features allows you to assign a "belching" sound to accompany the activity of emptying the trash on the desktop.

Other additions are designed to increase a user's productivity. Our favorites in this category are CE Software's QuicKeys, a macro utility; Fifth Generation's AutoDoubler, an automatic file compression manager; and Now Software's SuperBoomerang, which remembers which files and folders you use and can recall them when you open or save documents. Users can become very attached to these.

One rule of thumb about system additions is that you should not have two additions trying to modify the same resource or provide the same function. Users should choose between Berkeley Systems' After Dark and Fifth Generation's Pyro! screen savers, for example, or between SuperBoomerang and Shortcut. If they use one, they should not use the other. It's appropriate for network managers to exert some control over the user's system configuration if they are expected to support the user and troubleshoot problems, especially if license agreements are involved. There is currently a wide range of network manager policies toward system additions, ranging from complete control to no control.

Another rule of thumb is that commercial, shrink-wrapped applications are more stable and have better quality control with regard to conflicts than public domain or shareware additions. This is generally, but not always, true. Some network managers have instituted a "no-shareware" policy as a result. In addition, network managers should pay attention to version control, since version *x* of an addition might cause a conflict with a particular piece of software, while version *y* does not.

Some of the network administration tools designed for asset management can help you track system additions. They include TechWorks' GraceLAN tools, CSG's Network SuperVisor, MacVonk's NetOctopus, On Technology's Status*Mac, and Sonic System's Radar. All these applications are multipurpose network management tools. Although none were designed specifically for the needs of a network troubleshooter, all of them have a few functions that can be used for troubleshooting, and you'll definitely want to use them.

When tracking system additions in a troubleshooting scenario, the three most important questions are as follows:

- Which system additions do the users have loaded on their computers?
- Which versions of those additions are they using?
- What have they added or removed recently?

It's a good idea to establish what has changed recently because you are often called on to troubleshoot a system that "used to work." In such situations, looking at what aspects of the system configuration have changed since the last time it worked often provides a good clue.

If you are currently trying to decide which of these products to invest your time and money in, consider using a broad range of purchasing criteria. How well each of these products performs with respect to one particular troubleshooting task is not necessarily a good predictor of how well it will perform for a broad variety of tasks. Nevertheless, let's rate the effectiveness of the products mentioned in terms of how effectively they can help you answer the questions posed above.

Status*Mac isn't particularly useful for this kind of troubleshooting because it can't scan systems on demand. It's oriented toward preventive maintenance and resource planning. Likewise, NetOctopus has many useful features, but is lacking in this particular category and is not well-suited to this particular troubleshooting task.

GraceLAN comes in three modules that are available separately. You can use the Network Manager module to gather the data from systems connected to the network and either view this data on screen or export it a file. If you have the GraceLAN Asset Manager module, you can import the saved data and take advantage of the Asset Manager's database functionality. In troubleshooting, however, speed is often a consideration, and this two-step process might be slower than you'd like if you must find an answer very quickly. If you're not in a great hurry, a wonderful feature of the Asset Manager's import process is that it keeps a journal that tells you what has changed since the last time you performed a scan. You can also look at the data you exported from the Network Manager module with a spreadsheet or word processor program. Unfortunately, with regard to system additions, GraceLAN tracks only the filename and type and does not look at file size, heap size, or version, among other aspects. As a result, its effectiveness at troubleshooting system configurations is limited. However, GraceLAN does many other things well and is the only one of these products with any DOS support.

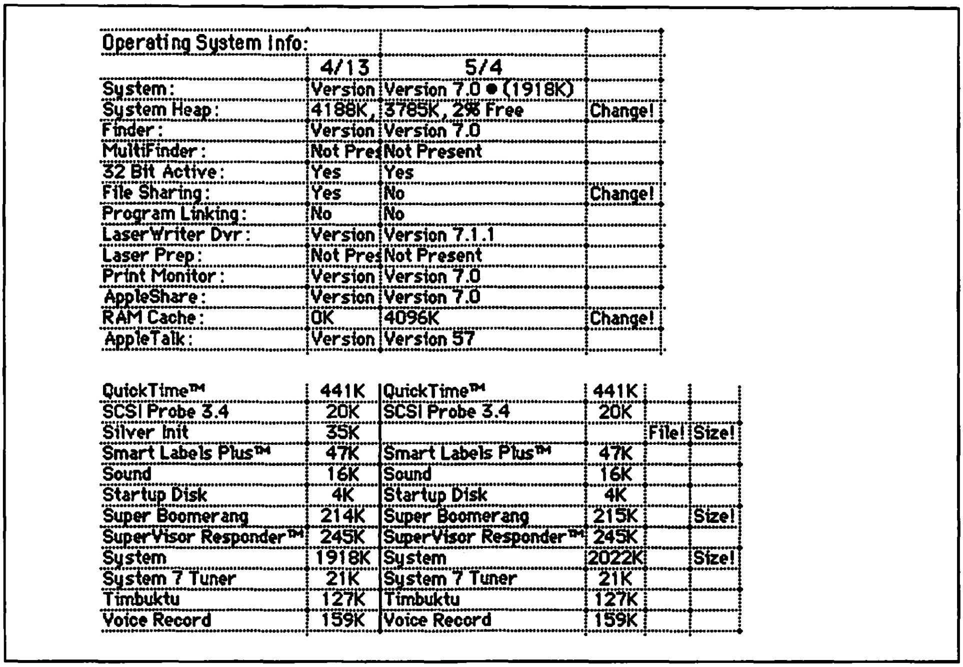
Both Network SuperVisor and Radar can scan the network for system additions and tell you the version numbers, but only Network SuperVisor can tell you how much memory an extension is using and what other processes the extension might have started when it was activated. Radar has an unattractive interface when viewed on a monochrome monitor, which usually isn't a problem, but in Radar's case, it detracts from its usability. You can't tell, for example, whether a button is active or inactive in monochrome. It also has no database functions, but it has some printer maintenance features found in the LaserWriter utility and includes alarms similar to the AG Group's NetWatchMan.

The bottom line is that none of these products has everything you want in a troubleshooting tool. The best of the current crop for troubleshooting is Network SuperVisor, which, in addition to the features already mentioned, has a self-scan feature built into the Control Panel which you install on each user's machine. Using the Network SuperVisor Responder alone (without the database application), you can get much of the information you need for troubleshooting because you can save the scan information to a tabular format and open it in a spreadsheet. That's a nice feature because when you're out in a user area and away from your own Mac and you don't have the Network SuperVisor application or the database files present, you still have a tool to use.

Figure 6-26 shows a sample spreadsheet analysis of the saved output from Network SuperVisor Responder on the same Mac on two different days. Besides the two columns of information, one for each day, we made a simple function in a third column to automatically compare them. This is helpful because it's easy to miss small differences among all this text data.

Finding information about potential conflicts between system additions can be difficult. Sometimes it's included in a product's user's guide. The server installation guide for QuickMail 2.5, for example, warns that the server process is not compatible with System 7's File Sharing Extension. In some cases, a company's tech support staff is aware of a conflict. For example, both Now Software and Berkeley Systems knew of the conflicts between Now Utilities 3.0 and After Dark 2.0u, advised users to upgrade to Now Utilities 3.0.2 and After Dark 2.0.v. Fortunately, both companies made an upgrade utility available for downloading on CompuServe and other public on-line systems, and provided a disk-based upgrade for a nominal charge.

In other cases, you'll have to use trial-and-error to find which system addition is causing the incompatibility. The way to accomplish this is to selectively add and remove system additions until you find the combination that is causing the conflict. This can be quite tedious, especially if you are looking for the definitive cause of the conflict, because there could be a great many combinations. Now Utilities includes a StartUp Manager control panel that simplifies the process somewhat by allowing you to selectively enable or disable extensions and control panels.



Operating System Info:			
	4/13	5/4	
System:	Version	Version 7.0 ● (1918K)	
System Heap:	4188K,	3785K, 2% Free	Change!
Finder:	Version	Version 7.0	
MultiFinder:	Not Pres	Not Present	
32 Bit Active:	Yes	Yes	
File Sharing:	Yes	No	Change!
Program Linking:	No	No	
LaserWriter Dvr:	Version	Version 7.1.1	
Laser Prep:	Not Pres	Not Present	
Print Monitor:	Version	Version 7.0	
AppleShare:	Version	Version 7.0	
RAM Cache:	0K	4096K	Change!
AppleTalk:	Version	Version 57	

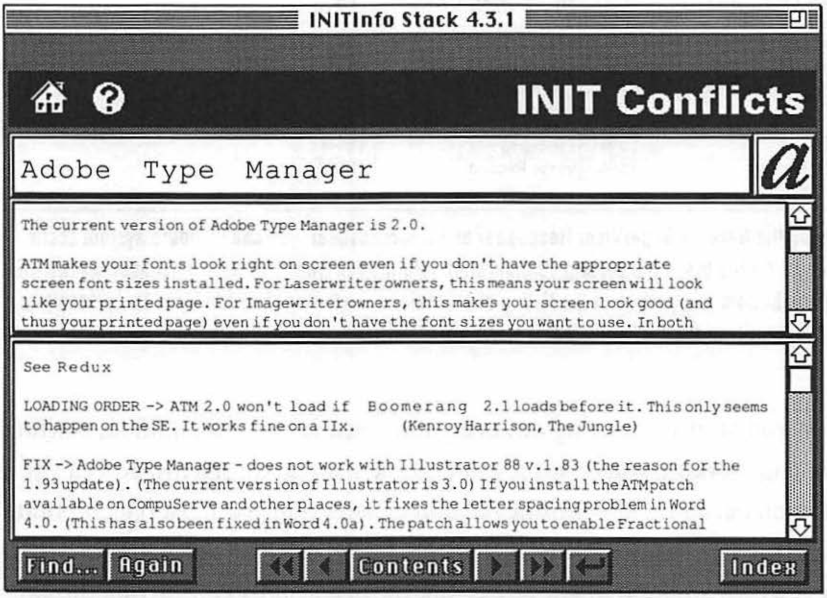
QuickTime™	441K	QuickTime™	441K		
SCSI Probe 3.4	20K	SCSI Probe 3.4	20K		
Silver Init	35K			File!	Size!
Smart Labels Plus™	47K	Smart Labels Plus™	47K		
Sound	16K	Sound	16K		
Startup Disk	4K	Startup Disk	4K		
Super Boomerang	214K	Super Boomerang	215K		Size!
SuperVisor Responder™	245K	SuperVisor Responder™	245K		
System	1918K	System	2022K		Size!
System 7 Tuner	21K	System 7 Tuner	21K		
Timbuktu	127K	Timbuktu	127K		
Voice Record	159K	Voice Record	159K		

Figure 6-26. Using the Network SuperVisor Responder and a spreadsheet, you can compare system scans from different days. On the top is the system configuration information from the same Macintosh on two different days. On the bottom is a portion of the listing of extensions, including their file size. The extensions heap size and the identity of the INIT code that it starts are also listed, but not shown.

Typically, you start by clearing out every nonessential system addition, starting the machine, and working with it for a while. If there are no problems, you can add a system addition back in. The scientific method dictates that you must reincorporate only one addition at a time, but very few network managers actually do this. Often you can guess which addition might be causing the problem by considering the condition under which the problem occurred. For example, if the Mac crashes only when it's unattended, you'd naturally suspect a process that occurs during idle system activity, such as a screen saver like Pyro! or After Dark, an indexing utility like On Location or a file compression utility like AutoDoubler.

If the problem happens when you have an Open or Save Dialog Box on the screen, you should first suspect an addition that changes the appearance of these kinds of dialog boxes; you can check the ShortCut, SuperBoomerang, or Directory Assistance II system additions first. If your problem occurs only during a complex analysis or computation, you might suspect a bug in the program or a lack of memory. This kind of analysis may or may not help. The key is to notice the common threads in how and when the problem occurs, consider the resources involved in the processes that are occurring at that time, and look at the system additions that deal with those resources.

On-line services can also provide you with good information both in their discussion sections and in files uploaded by users and user groups who track such conflicts. Figure 6-27 shows a screen shot from INITInfo Stack 4.3.1 by Glenn Brown and Gary Ouellet, downloaded from America On-line. It's a shareware stack ($15) that is frequently updated and contains conflict information for over 140 system additions and applications, as well as for hardware problems that are known to exist in all the Macintosh models.



The current version of Adobe Type Manager is 2.0.

ATM makes your fonts look right on screen even if you don't have the appropriate screen font sizes installed. For Laserwriter owners, this means your screen will look like your printed page. For Imagewriter owners, this makes your screen look good (and thus your printed page) even if you don't have the font sizes you want to use. In both

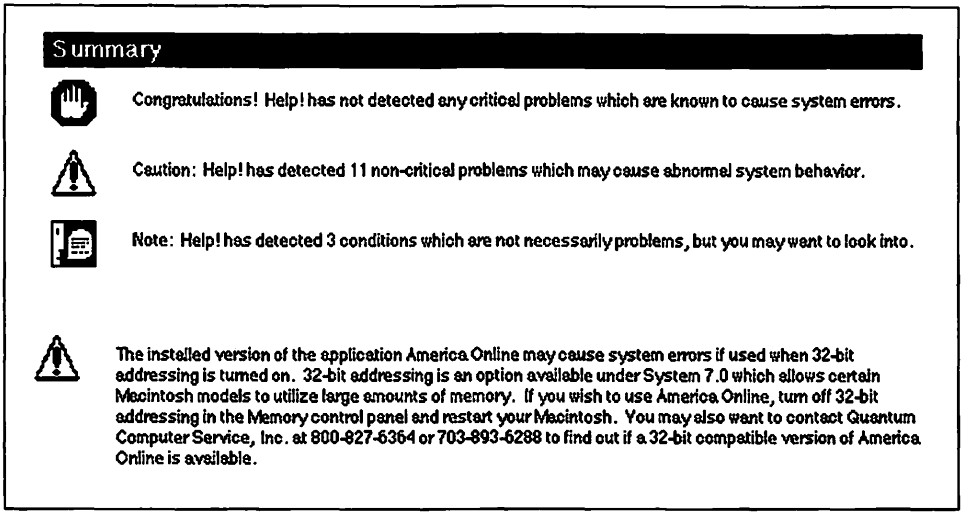
See Redux

LOADING ORDER -> ATM 2.0 won't load if Boomerang 2.1 loads before it. This only seems to happen on the SE. It works fine on a IIx. (Kenroy Harrison, The Jungle)

FIX -> Adobe Type Manager - does not work with Illustrator 88 v.1.83 (the reason for the 1.93 update). (The current version of Illustrator is 3.0) If you install the ATM patch available on CompuServe and other places, it fixes the letter spacing problem in Word 4.0. (This has also been fixed in Word 4.0a). The patch allows you to enable Fractional

Figure 6-27. The INITInfo stack, a shareware HyperCard stack, is available on America On line, CompuServe, and GEnie, and contains a wealth of information about system addition conflicts and incompatibilities.

Another useful utility is Help!, which performs a complete scan of your hardware and software configuration, checks the configuration with a knowledge base of over 25,000 configuration "rules," and reports the discrepancies to you. The knowledge base is updated quarterly by subscription. Figure 6-28 shows the summary at the beginning of 22 pages of documentation and analysis and the listing of one of the 11 noncritical errors that it detected.



Summary

Congratulations! Help! has not detected any critical problems which are known to cause system errors.

Caution: Help! has detected 11 non-critical problems which may cause abnormal system behavior.

Note: Help! has detected 3 conditions which are not necessarily problems, but you may want to look into.

The installed version of the application America Online may cause system errors if used when 32-bit addressing is turned on. 32-bit addressing is an option available under System 7.0 which allows certain Macintosh models to utilize large amounts of memory. If you wish to use America Online, turn off 32-bit addressing in the Memory control panel and restart your Macintosh. You may also want to contact Quantum Computer Service, Inc. at 800-827-6364 or 703-893-6288 to find out if a 32-bit compatible version of America Online is available.

Figure 6-28. Help! analyzes a Mac's system configuration and reports any anomalies. The summary of the analysis and the listing for one of the noncritical problems is shown.

Of course, you should also check for viruses and problems with the user's disk drive. Even though you might want to recommend only one disk analysis program for your users to purchase, it's worthwhile for you to have them all in your own toolkit. Our favorite in this class of utilities is the Norton Utilities for the Macintosh 2.0, which works for 98 percent of the problem cases. For that remaining 2 percent of your problems, have Apple's Disk FirstAid, Central Point Software's MacTools Plus, the Fifth Generation utilities, and La Cie's SilverLining waiting in the wings. You never know what might work.

If your system is crashing, it might give you a numerical code that indicates the conditions that caused the crash. Some of these codes indicate a specific problem. For example, an error code of –34 indicates a full disk. Other error codes can indicate a tremendous range of problems, such as a bus error (Error Code 1). While it's safe to say that the bus error resulted from a software problem, you can determine nothing more specific than that.

A great source for interpreting Macintosh error codes is the appendixes of *MacsBug Reference and Debugging Guide.* This book is written by Apple and published by Addison-Wesley. Some on-line services also have these error codes in their file libraries. A favorite of ours, available on America On-line, is called System Errors Guide 7.0.1 by "Dr. Pete" Corless of Apple Computer, Inc.

You can sometimes eliminate a configuration problem by reinstalling the software. The installer utility checks for all the necessary components and installs them if they do not exist or are out of date. Help!, for all its abilities, checks only for the presence of configuration errors; it does not look for the absence of components required to perform a desired activity.

When you reinstall software, there are two levels of rigor. In the first, a "dirty" reinstallation, you simply run the installer with the software you are trying to install already in place. Depending on how its script is programmed, the installer can check for the presence of old software, check the version and date, and either ask you if you want the current files replaced, replace them automatically, or leave them untouched. You usually won't know for sure which of these happens. In a "clean" reinstallation, you first remove all the software, then install it again as if it were new.

Establishing a common file system

If you are sharing files between Macintoshes, there are typically no problems that result from the way that the file systems make their translations from HFS to AFP and back to HFS again (see Figure 6-28). Even complicated file operations, such as allowing multiple concurrent access to the same file in a database, are typically trouble-free because AFP was designed specifically for the needs of HFS and generally for the needs of other native file systems such as DOS. The operations available within AFP are detailed in Table 6-1:

Table 6-1. The AppleTalk File Protocol (AFP) supports a comprehensive set of operations to meet the needs of Apple's Hierarchical File System (HFS).

FPAddAPPL	Adds an association between a specific creator and a particular application to the Desktop database.
FPAddComment	Associates a comment string with a file or directory in the Desktop database.
FPAddIcon	Adds an association between a file type and creator and an icon to the volume's Desktop database.
FPByteRangeLock	Locks or unlocks a range of bytes in an open fork.
FPChangePassword	Changes the log-in password for the current username.
FPCloseDir	Closes a directory.
FPCloseDT	Closes a Desktop database.

Table 6-1. *Continued*

FPCloseFork	Closes a file's fork.
FPCloseVol	Closes a volume.
FPCopyFile	Copies a file from one place to another on the same AFP server volume.
FPCreateDir	Creates a new directory.
FPCreateFile	Creates a new file.
FPDelete	Deletes a file or directory.
FPEnumerate	Lists contents of a directory.
FPFlush	Writes modified volume data back to disk.
FPFlushFork	Writes buffered data associated with a file fork to disk.
FPGetAPPL	Retrieves the application associated with a specific creator from the Desktop database.
FPGetComment	Retrieves a comment associated with a file or directory.
FPGetFileDirParms	Retrieves file or directory parameters.
FPGetForkParms	Retrieves parameters for a particular file's data or resource fork.
FPGetIcon	Retrieves an icon bit map from the Desktop database.
FPGetIconInfo	Retrieves information about an icon on the Desktop database.
FPGetSrvrInfo	Retrieves descriptive information about an AFP server.
FPGetSrvrParms	Retrieves information about volumes on a particular AFP server.
FPGetUserInfo	Retrieves information about a user registered with an AFP server.
FPGetVolParms	Retrieves information about a specific volume.
FPLogin	Establishes a session with an AFP server.
FPLoginCont	Continue the log-in process. This call is necessary for some types of authentication.
FPLogout	Ends a session with an AFP server.
FPMapID	Maps a numeric user ID to a username, or a group ID to a group name.
FPMapName	Maps a username to a numeric user ID, or a group name to a group ID (the inverse of FPMapID).
FPMoveAndRename	Moves a directory or file to another location on the same volume, and optionally renames it in the process.
FPOpenDir	Opens a directory.
FPOpenDT	Opens the Desktop database on a particular volume.
FPOpenFork	Opens a resource or data fork on a specified file.
FPOpenVol	Opens a volume on an AFP server.

Table 6-1. *Continued*

FPRead	Reads a block of data from an opened fork.
FPRemoveAPPL	Removes ancreator/application mapping from a Desktop database.
FPRemoveComment	Removes a comment from a Desktop database.
FPRename	Renames a directory or file.
FPSetDirParms	Sets parameters for a specified directory.
FPSetFileDirParms	Sets parameters for a file or directory. (This call sets those parameters that are common to both files and directories.)
FPSetFileParms	Sets parameters for a specified file.
FPSetForkParms	Sets the fork length for a specified open fork.
FPSetVolParms	Sets the backup date for a volume.
FPWrite	Writes a block of data to an opened fork.

When using Mac files on a PC, however, there can be problems because of the way that the PC application might need to lock a file while it's in use. For example, an AppleShare-capable PC running Lotus 1-2-3 for Windows 1.0 cannot open a spreadsheet file that resides on a Mac using the Mac's File Sharing option because Lotus 1-2-3 cannot reserve the file in a way that it finds satisfactory. Lotus 1-2-3 tells you that it has a "file contention" problem because the Mac server's file system seems to have control of the file in a way that blocks access by Lotus 1-2-3. In this situation, you'll have to transfer the file over to the PC and work with it while it resides on the PC's own native volume.

The more different the two native file systems are, the more likely that problems like these can occur. This is because the way that the native file system accomplishes an operation such as reserving a file for use might not have a suitable equivalent in the Mac's native file system, HFS, or in AppleTalk's network file system, AFP.

Typically, transferring the file from a network volume to a native volume cures these problems, although it might reduce the user's ability to share information. Another remedy is to share the file when it is on a dedicated AppleShare server (which, in the case of the Lotus 1-2-3 file, still won't work) or on another type of file server that is mountable by both DOS and Mac platforms, such as a Novell file server (which, in the example, works). Even though all these file servers make use of AFP, they might do so in ways that are different enough to alleviate the source of the problem. (In the 1-2-3 example, you should probably notify Lotus' technical support department of the problem.)

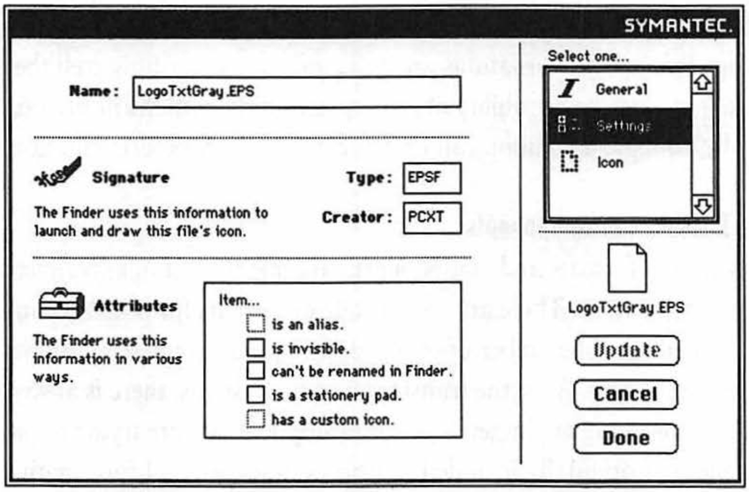
When you use a gateway such as GatorShare to access an NFS-based server, the translation passes through yet another step, and so the potential for problems is even greater. Here the operations on the Mac's HFS representation of the file system are translated to AFP commands by the AppleShare extension, and GatorShare transforms the AFP commands into an NFS operation, which is finally translated (typically) into operations on a UNIX-style file system. Mapping functions between AFP and NFS can be difficult, as the function set supported by NFS is not nearly as rich as that supported by AFP. Typically, several NFS calls must be made to support a given AFP operation. When this many translations are going on, no matter how well they are performed, there is always the possibility of error, particularly with the more complicated ones. Usually, though, a solution can be achieved through experimentation.

Troubleshooting file formats and components

When dealing with file formats and components, you might need to experiment a bit to get the right combination. There are a multitude of formats for both text and graphic files and an equally great number of public domain and commercial utilities to convert files between the two. As in the translation of file systems, there is always the risk that a bit of the meaning might get lost. For example, if you are trying to use a word processing file that originally included footnotes in its original form with a word processor that does not have a footnote feature, it's difficult to predict how the footnotes will be displayed, if they are displayed at all. In translating spreadsheet programs from one file format to another, there might be times when you can't translate your macros or charts.

Some programs are especially good at working with a wide variety of files. An example is Deneba's Canvas, which can work with a wide variety of both Mac and PC graphics file formats. Some network managers buy Canvas specifically to help users perform file format conversion, and have no other reason to use the program. Another popular utility is the Apple File Exchange program, which, like most programs published by Claris, can be extended with translators such as those from DataViz.

One problem you might encounter when you try to create or export a Mac file while working on another type of computer is that the program might not have a way of giving the Mac file the file attributes it needs, such as the appropriate type or creator. For example, suppose you want to create an EPS (Encapsulated PostScript) File from a graphic created on a PC so you can use the image in an Aldus Freehand drawing on the Mac. When you try to open the file in Freehand, the filename will not appear in the Open dialog box unless the file's attributes indicate that it is an EPS file by having a file type of EPSF. Some common file types, such as TEXT or PICT, might be familiar to you. You can determine appropriate values for others by examining files that work with utilities like ResEdit, DiskTop, or Norton Utilities. There are many ways to set or change the type of a file, including the use of ResEdit. An example using Norton Utilities for the Macintosh 2.0 is shown in Figure 6-29.


Name: LogoTxtGray.EPS

Signature

The Finder uses this information to launch and draw this file's icon.

Type: EPSF

Creator: PCXT

Attributes

The Finder uses this information in various ways.

Item...

□ is an alias.
□ is invisible.
□ can't be renamed in Finder.
□ is a stationery pad.
□ has a custom icon.

Figure 6-29. With the FastFind utility in Norton Utilities for the Macintosh 2.0, you can easily change the type of a Mac file.

Troubleshooting the application layer

Finally, you might occasionally run into problems at the application layer itself. The application layer is the user's interface to the rest of the system. The difficulties encountered here usually fall into two categories: "cockpit error," or misunderstandings on the part of the user as to how the program works; and actual bugs with the application software.

The first category is most easily dealt with through education. You might watch the user recreate the problem and determine the cause of confusion. This can either be done at the user's workstation or remotely with a utility program such as Timbuktu. Once the user's error has been identified, you can demonstrate the correct way to address the situation.

Actual bugs with application programs also crop up from time to time. While most software publishers do a pretty good job of quality control, there are occasionally "corner cases," or peculiar combinations of circumstances that manage to slip by the publisher's testing group.

You really have only two courses of action in this sort of situation: you can get in touch with the publisher's technical support department, report the bug, and hope for a timely resolution; or you can attempt to find some kind of workaround, either in the form of an alternative procedure or another application with comparable capabilities.

Summary

Layer-mode troubleshooting locates problems by systematically checking each of the functions of a network according to the order prescribed by the OSI Seven-Layer model. Starting at the Data Link Layer, we test each of the functions in the layer. When we are satisfied that all functions are healthy, we go up to the next higher layer and begin again. If the functions are not healthy in our current layer, we find out why by concentrating our search for the cause in the workings of that layer and in the layer just below.

CHAPTER SEVEN

Network Process Descriptions

Process mode is the most revealing and the most challenging mode of network troubleshooting. It's most revealing because you look at exactly what happened—the record of network events—in a protocol analyzer. It's most challenging because it requires the greatest knowledge. You get to look at the chain of events in a process and look at where the process begins to unravel. Network processes involve information exchange events. Devices trade questions and answers, commands and replies; they ask for and receive status reports; they ask for, accept, and refuse connections; and they send and receive data. In most processes, there is a definable flow of events that leads to a successful conclusion for the user. A deviance from the prescribed flow of events often leaves the user with less than satisfactory results.

Process-mode network analysis, however, is more than simply spotting the required events in the protocol analyzer and ticking them off a checklist. The critical aspect of network process analysis is that you must understand the interdependencies of the events. You must conceptualize each device as an information agent: "The Gonkulater performed Event A to gain the information necessary to perform Event B, which sets the stage for Events C and D. If Event A does not get a valid answer, then the Gonkulater will perform this other chain of events to get the information it desired from Event A." When you view any event that a device initiates or responds to, your main concerns are:

1. What information does the device need to know to perform this event?
2. How did it find out that information?
3. What information is gained by performing this event?
4. What events will the device be capable of as a result of gaining this information?
5. If the event is unsuccessful, what alternatives does the device have?

You must, in a way, personally identify with the device. Protocol analyzers provide mountains of information; your task is to sift through those mountains for the nuggets of pertinent data that lead to an understanding of the problem.

This chapter describes the flow of events for many common AppleTalk processes. If you study these descriptions, keeping in mind the questions above, you should also gain the kind of understanding that willl help you troubleshoot processes not included in this book.

Device Startup

When a networked device powers up, sooner or later it needs to use the network. Usually, the first need for the network is during startup when one of the device's software processes needs to register an NBP (Name Binding Protocol) name for itself.

Before this can happen, the AppleTalk protocol processes must be initialized. This is a complex series of hardware and software events: software processes are started, computing resources are allocated, hardware drivers are engaged, and settings and preferences are retrieved. Then the device sends out its first packets and becomes a member of the network. Some of these processes, like memory allocation and driver initialization, are invisible to you. But you can see other events on the network by using a protocol analyzer.

Watching a workstation initialize its protocols will tell you what software processes it intends to run and how those software processes make their presence known on the network. It may also give you insight into why the device is or is not making a connection.

This section gives a very detailed account of how two devices, a Mac and a router, start their AppleTalk processes on both LocalTalk and Ethernet. Routers and non-routing nodes like Macintosh workstations have very different responsibilities in AppleTalk. Routers are responsible for knowing everything about how the internet is built: where all the networks are located and how the internet's zones correspond to its networks. Non-routing nodes know very little about the internet or even about the network and zone in which they're placed. Because the burden of routing is placed almost exclusively on the internet routers, routers behave differently during startup than nonrouters.

The two kinds of devices do have some similar tasks during startup, because both devices must become legitimate AppleTalk nodes on the network to which they're attached. Any AppleTalk node must choose a valid, unique AppleTalk address. After securing an address, both kinds of nodes then register their service names, first making sure that no other node in their zone already has registered that name.

Routers pursue these goals much more rigorously than nonrouters. It's very important that a router have correct information because routers define the network numbers and zone identities. Also, non-routing nodes depend on the routers to make the internet function properly.

Other parameters that affect startup behavior are the type of network to which a node is connected and the existence of a router already on the network when the node boots up. All of these scenarios—routers and nonrouters, LocalTalk and EtherTalk, and the presence or absence of a preexisting router—are covered in this section.

Watching the startup procedure often provides important clues that can lead you along a fruitful troubleshooting path. In addition, detailed descriptions of startup procedures provide a good way of explaining key AppleTalk concepts—not only the *what* of the events, but the *how* and the *why*.

Startup on LocalTalk with no router

Macs normally initialize their AppleTalk processes during startup when a System Extension (INIT) such as Responder or Timbuktu needs to register an NBP service name on the network. The first network-visible event is that the Mac performs an address bid to secure an AppleTalk node address for itself.

The Mac does this by choosing a tentative address and then checking to make sure that no other device already has that address. If the tentative address is unique, the Mac adopts the address as its permanent node address. If the address is already in use, the Mac randomly picks another address and checks it for uniqueness. The process continues until a unique address is found or the Mac determines that a unique address can't be found.

One of the elements that Macs store in Parameter RAM (PRAM) is their node address. When they're turned off and then later restarted, they first use this stored address, the "hint" address, as their first tentative address. We'll use 63 as the hint address of the hypothetical Mac that's booting up. The Macs send a large number (640) of LLAP (LocalTalk Link Access Protocol) LAPEnq packets to the hint address, node 63. The purpose of a LAPEnq packet is to ask, "Is any device already using 63 for its node address?" If another LocalTalk device is already using the hint address, it receives this packet and quickly responds by sending a LAPAck packet back to the booting Mac, in effect saying, "Yes, I'm using 63 for our node address."

If the booting Mac receives a LAPAck, it immediately stops sending LAPEnq packets to Node 63. It then quickly (in less than .001 of a second) chooses a random number between 1 and 127 for its next tentative node address and sends LAPEnq packets to this new number. This process will be repeated as necessary, but the Mac will eventually abandon the process if it fails to turn up a unique address. In these extremely rare cases, the Mac that fails to find a unique address will alert the user, "Can't Open AppleTalk Driver." An address bid on LocalTalk usually takes slightly over a second to complete.

In this section, we'll assume the node address 63 has successfully checked and been adopted by the device. This is the node address portion of the AppleTalk address only; at this time, the Mac still doesn't know whether or not there's a router on the network, or whether or not the network has a network address assigned to it.

Because LocalTalk can have only a single network number, it isn't necessary to know the network address to perform an address bid. This Mac now has an Internet Node Address of 0.63. Macs and other nonrouters on LocalTalk assume a network address of 0 when they enter service. The address 0 means "this network."

Registering an NBP service name

Following a successful address bid, the name registration process begins. First, the System Extension (we'll use Timbuktu as the example) uses NBP (Name Binding Protocol) to search for the name it intends to register. Like the node address, the Mac wants to make sure that the service name it registers will be unique.

Before it can send any NBP packets, however, the Mac must find out whether there are any routers on the network. NBP names are registered on a zone-wide basis, and zones can include more than one network. If the network on which the Mac resides is part of a larger zone, then the Mac must use the router to search for names in all of the networks that make up the zone.

There are two ways to determine whether a router is on the network. The first method is to wait for a few seconds, listening for Routing Table Maintenance Protocol (RTMP) packets. Routers are required to broadcast an RTMP Data packet every ten seconds. The second method, which Macs use, is to actively search for routers by broadcasting RTMP Request packets, inviting any router that may be on the network to immediately send back an RTMP Response packet. Only routers can send RTMP Data or Response packets, and these packets contain the network identity.

In this network, however, there are no routers, so there will be no RTMP Response packet. The Mac will continue to use 0.63 as its Internet Node address until such time as a router comes on line and begins sending RTMP Data packets.

This simplifies things, because now the Mac can do a zone-wide search for its intended service name by simply broadcasting a packet to the network it's on. It uses the default network number, 0 (this network) Without routers, "this zone" is synonymous with "this network." On a network with a router, the Mac always assumes that the zone is larger than the network.

The Timbuktu software process engages the NBP process to search the network/zone for the name it intends to register by specifying the number of NBP LookUps to be sent and the name to search for. Service names have three parts: an *object name,* a *service type,* and a *zone.* Timbuktu might ask NBP to search three times for the object name 040567 with a service type TB2 Serial in the "*" zone. The form of this query is expressed as 040567:TB2 Serial@*. NBP will respond by sending three NBP LookUps for 040567:TB2 Serial@*. The packets are broadcast to the Names Information Socket, socket 2, on every node. The destination address, then, is 0.255.2—the name socket on all nodes in this network.

The NBP query format bears some similarity to a database query in a structured language. In fact, NBP conceptualizes the network as a distributed database; each node remembers only the sockets that it has registered and can be queried using the NBP query format. An AppleTalk device responds when a query that it receives "hits," or matches one of its registered sockets by sending the full address of the socket back to the requesting node.

Like any structured query language, the NBP query has wildcard characters. An asterisk (*) in the zone field means "this zone" and matches any zone name. While there's no wildcard character that means "all zones," an equal sign (=) in the object or type fields means "all object names" and "all types," respectively.

After establishing that the serial number isn't already in use, Timbuktu then registers the name that others users will see when they start up their copies of Timbuktu and look for available hosts. In our example, the three LookUps for 040567:TB2 Serial@* will be followed by three more NBP LookUp broadcasts for George:TB2 Host@*. The English translation of the second trio of packets is, "Does any device on this network have a socket registered with the name of George and the type of TB2 Host?"

Besides the AppleTalk destination address of the LookUp packet—0.255.2—the LookUp must also include the AppleTalk address to which responding devices should send their replies. This requires that the NBP search process designate a socket for the return packets. While some system processes designate the Names Socket—socket 2—to receive replies, name searches for extensions like Timbuktu usually open one of the dynamically allocated socket numbers near the top of the socket address range.

In our example, we'll use socket 254 as the source socket of the NBP LookUp. Any devices on the network that have one of these two service names in use would send an NBP Reply packet to node 63, socket 254. Socket 254, however, isn't necessarily the socket that Timbuktu will take when it registers its name; it's merely the socket used for searching.

The NBP LookUps are usually sent to ensure that the name a process wants to register isn't already in use. The first trio of search packets is actually a search for a device using the same Timbuktu serial number. Any device with a registered service name of 040567:TB2 Serial@* would be a device with a duplicate serial number, which would violate the license agreement. If the Mac making this search received an NBP Reply, the user would be informed that Timbuktu couldn't open because of a duplicate serial number.

The second trio (for George...) is a search for the name that Timbuktu will use as its service name when other devices scan the network for Timbuktu hosts. Timbuktu has a selection window that when open, asks NBP to periodically search for =:TB2 Host@*. The "=", remember, is the wildcard that matches any object name. Any device running Timbuktu will have a socket registered with the type TB2 Host—but only if the user has Timbuktu's Guest Access turned on.

The Mac repeats the NBP process for every extension or system process that needs a name. Besides NBP activity, other AppleTalk processes that may occur during startup are logins to any mail and file servers that the user selected for automatic login during a previous session. (Login processes are discussed later in this section.) Performing three searches per name is the most common practice, but some software processes use more than three LookUps, and some use fewer. The PPCToolBox, for example, sends 18, while a QuickMail server uses only two searches for each of the five names it registers.

Usually the last name registration to take place in a System 7 Mac workstation is the registration of the device as a file server (type AFP Server), because file sharing takes a while to get started. When the setup is complete, the search is performed.

All LocalTalk devices perform these two processes: an address bid using LAPEnqs, followed by NBP name searches. Each device is programmed to send a large number of LAPEnqs to make sure that their address is unique. The Mac ROM tells the Mac that a test of 640 packets without a LAPAck is sufficient to establish uniqueness. Other devices' programs may use a different number of LAPEnqs as the test for uniqueness; some send many more LAPEnqs than the Mac's 640. Since we've never seen this take more than three packets, even 640 LAPEnqs seems like overkill, but the address bid on a Mac only lasts a second or so and consumes less than 30 percent of the bandwidth during that second.

A device that's started without first being plugged into the network will go ahead and perform its address bid and name searches in spite of the fact that it isn't connected. The LocalTalk signaling hardware has no method for checking whether or not it's connected to a "live" network. In such cases, all searches for addresses and names will naturally reveal no duplicates. Later, if the device is connected to the network, its address and service names could possibly duplicate those of other devices. All extensions that require name service perform LookUps of this type. Most extensions send three LookUps per name, but some, such as the PPCToolBox, send more than three if someone wants a more rigorous search.

More about name registration

Using NBP to search for names isn't the same as registering a name. The actual NBP name registration is a silent process (no packets) that occurs after the search is completed and no duplicates are found. As mentioned earlier, a software process usually registers the service on a different socket than the one it used for searching. The whole purpose of NBP is to match service names, which are easy for users to work with, to the AppleTalk internet addresses that devices must use to address the packets they send. Just as the LookUp packet provides the address for replies, the Reply packet provides the return address for whatever is the next step in the communication.

Software processes take the trouble to register sockets only if they want to provide a way for other software processes to be able to locate them later. There are three reasons a software process might want to be contacted.

The first reason is illustrated by the Timbuktu serial number; NBP registration is a way of enforcing license agreements. The second reason is illustrated by the "TB2 Host" registration; the name provides a way for other devices to locate the service so that a connection can be established. The third reason is that some network management programs use service names to build device maps or lists of the network or to provide a service access point for other network management functions. The extensions for GraceLAN, StatusMac, or Network SuperVisor are examples.

In serial number searches like the one described above, finding a duplicate serial number results in some negative action on the part of the extension. Serialized extensions normally respond to name duplication by aborting the load process, not registering any name, and in some cases, wiping out the registration information entered when the application was first run. Sometimes the user is presented with a dialog box in which to enter a new serial number; the user may also be notified of the name of the user with the duplicate serial number.

A common question about the Name Binding Protocol is, "If NBP searches for serial numbers are only conducted in a zone, does that mean we could buy one zone's worth of licenses and then duplicate serial numbers from one zone to the next?"

Aside from any legal or ethical considerations, the answer is, sometimes. During startup, it's true that the name registration only takes place with one zone, but some applications later go snooping in other zones looking for duplicates. An example of an application that does this is Farallon's Liaison software router. Timbuktu doesn't do this, but two users with the same serial number, even in different zones, can't connect with each other.

Serial number duplications usually result in a negative action on the part of the software process. But for duplications of names that don't involve serial numbers, a more typical response is for the extension to simply modify its object name and repeat the LookUp process. For example, it may send out a LookUp for George1... if the name George is already taken. If George1 is taken, the extension may try George2, and so on. When it finds an available name, it then registers that name along with the type and zone.

Although this name modification process happens in Macintoshes, a more significant example of name duplication occurs with shared devices such as LaserWriters. A common scenario is that an organization will buy two LaserWriters of the same type (for example, two LaserWriter II NTXs). The default name (straight out of the box) of a LaserWriter II NTX is LaserWriter II NTX. Both LaserWriters will want to register the name LaserWriter II NTX:LaserWriter@*. The first LaserWriter on the network will successfully get that name; the second LaserWriter will get the name LaserWriter II NTX1.

Both of these names will show in the Chooser window of Macs searching that zone for LaserWriters. When a user selects one of these LaserWriters, the name of the

LaserWriter will be written into their System File and LaserWriter Driver. It's conceivable that at some future point the two LaserWriters could be turned off concurrently and then powered up in the reverse order. In this case, their names would switch and users would unwittingly print to the wrong LaserWriter. Of course, this problem can be avoided by using the Namer software to give the LaserWriters unique names.

Some hardware devices include their serial numbers within their default NBP name—not for serial registration, but to avoid duplication. An example is the GatorBox CS, which has a default name like CS023468:GatorBox, where CS023468 is the serial number. If more than one GatorBox is attached to the same network, the default NBP name of each GatorBox will be unique.

Startup on LocalTalk with a router

As mentioned earlier, a Mac checks the network for routers by sending two RTMP Request packets before it performs any name searches. If there are routers on the network, they'll announce their presence to the Mac by sending RTMP Response packets in reply.

The Response packet is a modified version of an RTMP Data packet. Unlike the data packet, which is broadcast, the Response packet is sent only to the requesting node. This packet tells the Mac what network the Mac is connected to. This information is referred to as Our_Net_No and is recorded in a piece of memory called the RTMP Stub. The source node address of the router's Response packet is also recorded in the RTMP Stub as A_Router. Whenever the Mac needs a router, it gets the network number and address of the router from the RTMP Stub.

The requesting Mac thereby learns its network number—but more importantly, it learns the node address of the router. It needs this information because the Mac must rely on the router to perform the name searches in order to be sure that the entire zone is searched for duplicate names.

Because there's a router, the Mac doesn't broadcast its required NBP searches using LookUp packets, but instead asks the router (whose A_Router node address is in the RTMP Stub) to perform the search. The Mac does this by sending the router an NBP Broadcast-Request packet. The format of the query will be the same, however: Name:Type@*. For Timbuktu, the packet would translate, "Router, please ask any devices in this zone with the name of George and the type of TB2 Host to tell me their address." The router would then turn that Broadcast-Request into the required LookUp broadcasts and send them to all of the networks that make up the zone.

Notice that the Mac doesn't know and doesn't need to know what zone it's in. The Mac can still think in terms of "this zone."

The router translates the "*" into a list of network numbers by first using its Zone Information Table to determine the zone name of the network that the Mac is on, then determinine which other networks are also part of that zone. Having acquired a list of networks, the router then consults its Routing Table to find out how to send packets to those networks. Then it sends the required LookUp broadcast packets—one to each network that constitutes the zone.

When the router searches a network to which it's directly attached, it simply broadcasts an NBP LookUp packet that supplies the address of the Mac that requested the name search—"If there is any device on this network that has a socket registered with the name of George and the type of TB2 Host, please report your address to node 63, socket 254 on network 145."

If the router must search a network to which it isn't directly connected, it sends an NBP Forward-Request to the router that its routing table shows is the next link in the chain on the best path to that network. The next router will examine the destination address of the Forward-Request packet and perform the appropriate action; it will either pass it along to the next router along the path or broadcast a LookUp if the destination network is directly connected.

One interesting aspect of the NBP search process in routed internets is that while the Mac uses "*" in the zone field of the Broadcast-Request packet, the router replaces that "*" with a specific zone name in the corresponding LookUp packet. Then, if the LookUp is answered, almost every software processes on a LocalTalk network will send an NBP Reply packet with "*" in the zone field, saying, "I have a registered socket with the name of George and the type of TB2 Host in this zone."

This happens because that's the way the application registered the name—simply in "this zone." The fact that the zone field of the LookUp packet contains a specific zone name is irrelevant to the software process. It sends a Reply based simply on the fact that both object name and type match. "This zone" will match any value in the zone field.

This unusual feature is actually quite fortunate in light of a situation that commonly occurs when the zone name of a LocalTalk network changes while the devices are active. If the network manager renames the zone or replaces the router while the devices on that network remain on, other software processes will still be able to locate the devices because of the "*" zone name.

Some applications on LocalTalk networks, however, do find it necessary to register a name into a specific zone. An example is a QuickMail Name Server. After the name searches are performed, the Mac will ask the router for the name of the zone (using a ZIP GetMyZone packet) just prior to registration so that a specific zone name will be registered instead of "*."

What happens when you turn off AppleTalk?

At times, a user or a network manager might temporarily turn AppleTalk off in the Chooser while the Mac remains on. When AppleTalk is deactivated, all of the network processes are terminated and all of the sockets are closed. Other ways that AppleTalk can become deactivated is when a Mac Portable or PowerBook is put "to sleep," when some network management tools are used, or when the network is switched from one data link to another—for example, from LocalTalk to Ethernet.

For whatever reason AppleTalk was deactivated, when AppleTalk is started up again without a system startup—for example, when AppleTalk is reactivated in the Chooser—only some of the network processes that initialize at system startup will reinitialize themselves and become active again. A few others can be jump-started into action again, but most are not recoverable unless you restart the computer.

System 7 file sharing is an example of a process that will recover automatically, although any sessions active at the moment when AppleTalk was deactivated will be abruptly terminated. This can be a serious problem, which we will describe in more depth in "Using an AFP Server," later in this chapter.

Timbuktu is an example of a process that can be jump-started—in this case, simply by opening the Timbuktu desk accessory. In other applications, there may be a menu choice to start the process up.

QuickMail, Microsoft Mail, and Liaison are examples of processes that can be started only when the Mac boots up. The majority of network applications also fall into this category.

Always take care when disrupting AppleTalk for any reason, but especially when communication between applications is in progress. Users can become very angry when they lose their work. Since System 7 provides the opportunity for every Mac to be a file server, users need to be aware of the consequences of deactivating AppleTalk. Teach them to first check the File Sharing Monitor to determine whether any users are currently logged into their Mac as a file server. As interdependence among networked computers becomes more widespread, this step becomes even more important.

Router startup on LocalTalk

Routers, of course, are also AppleTalk nodes, and their startup processes share many characteristics with non-routing nodes. Like nonrouters, they also must acquire a node address and register their NBP names. Typically, however, routers perform more thorough checks for duplicate addresses than Macs.

There are also some small differences in the startup sequences of routers, depending on the manufacturer. The router used as an example in this section is the Cayman GatorBox CS, which has a fairly representative startup procedure among the routers in its class (less than $1500 per port). The example assumes that no other router is already present on the LocalTalk network, as is true in most situations.

The GatorBox sends out 3840 LAPAcks over a 20-second period and then sends an RTMP Response packet with only one tuple for the local network. (A tuple is a pair of data values: the network address and the number of hops.) Then it looks for the names of the services it wants to register on the network. It registers two services with the same object name but two different types: SNMP and GatorBox. The object name that it uses is a factory default name unless the network manager has renamed the router, using the router's configuration software. The GatorBox's default name is similar to CS101467, which includes the serial number of the device, although often the router's default name is simply the product name.

Perhaps because the GatorBox expects to have a unique name in either case—default or named—it performs only two name lookups for the GatorBox type and one for the SNMP type.

After these two events, about 25 seconds after sending its first LAPAck, it achieves steady state as an AppleTalk node, broadcasting its RTMP information every ten seconds.

Among the more expensive multiprotocol routers made by manufacturers such as Cisco, Wellfleet, and Ungermann-Bass, there may be significant differences not only in the startup procedures but in other situations as well. These differences arise because the designers of those routers approach AppleTalk after considerable experience with other routing protocols. Typically, their design philosophies differ slightly from those of designers who began their design careers with AppleTalk.

One example is the way the Ungermann-Bass MaxTalk router handles the situation when its tentative node address is already in use. Unlike more traditional AppleTalk routers (such as FastPath, GatorBox, Ether•Route, or Liaison), MaxTalk routers don't use a random address selection process when their tentative address is already in use. They simply choose the next highest numerical value for the next tentative address.

Startup on EtherTalk: Non-routing nodes

The actions that an Ethernet node performs on startup are similar to those of a LocalTalk node. The node needs to first verify that its AppleTalk address is unique and valid, then register its sockets. Phase 2 EtherTalk nodes must also be sure that they are assigned to specific zones. On startup, an EtherTalk node must:

1. Make sure it has a unique address that's valid for the network to which it's attached.
2. Make sure it has a valid zone designation.
3. Be informed of its zone-specific multicast address.
4. Register any sockets, checking the names first to verify their uniqueness.

The mechanics of this process are only slightly different from those of LocalTalk. Instead of the LAPEnq packet used in LocalTalk, the Ethernet node uses the AARP Probe packet. The message of the AARP Probe packet is the same as the LAPEnq packet: "Does any node have this address?"

Returning to a familiar network

In AppleTalk Phase 2, if a node boots up with Ethernet selected as the designated network (this can happen only when Ethernet was chosen during a previous session), it sends AARP Probes to the AppleTalk address it used when it was last on the network—its hint address. Thus, if the node's previous address was network 6, node 35, it sends AARP Probes to 6.35. This is shown in the packet below:

AARP Packet—AppleTalk Address Resolution Protocol

Hardware Type:	1 Ethernet
Protocol Type:	0x809b AppleTalk
Hardware Address Length:	6
Protocol Address Length:	4
AARP Function:	3 Probe
Source Hardware Address:	02:60:8c:83:26:13
Source AppleTalk Address:	6.35
Unused:	00:00:00:00:00:00
Dest. AppleTalk Address:	6.35

All of the AARP Probes are sent a number of times to make sure that the address in question truly is unique. The standard number of transmissions is 10 AARP Probes in 0.2-second intervals. Some nodes—routers, for example—may send more AARP Probes if a greater certainty is required for the unique address.

If the address bid is successful (no AARP Replies are received), the node then verifies its saved zone name. In this example, we'll use Ether Zone as the zone name used by the Mac when it was last on this network. The Mac will broadcast a ZIP GetNetInfo (GNI) packet, meaning, "Is Ether Zone a valid zone name for my network?" If there's no reply, the Mac will send out three identical ZIP GNIs for this zone name. If it receives a reply to the first ZIP GNI, it won't send any more. Only routers can respond to ZIP GNI's. The ZIP GNI Reply packet will tell the Mac the following information:

1. whether the zone name is valid or not
2. the default zone name for the network if the zone name supplied was invalid
3. the network numbers that are valid for the network
4. the zone multicast address for the supplied zone name (or default zone).

When the Mac has a valid zone name, it will then register its sockets, as described below.

Starting up on an unfamiliar network

If the node doesn't know which network numbers are valid, it sends out AARP Probes for an address in the EtherTalk startup range. The startup range is a set of reserved network addresses between 65280 and 65534. The purpose of the startup range is to allow nodes to take temporary addresses that will let them contact a router. The node then asks the router for the range of valid network numbers and then bids for an address in the valid range. The AARP Probe for the startup range would be as follows:

```
AARP Packet—AppleTalk Address Resolution Protocol
Hardware Type:                  1 Ethernet
Protocol Type:                  0x809b AppleTalk
Hardware Address Length:        6
Protocol Address Length:        4
```

AARP Function:	3 Probe
Source Hardware Address:	02:60:8c:83:26:13 Phase 2 Mac
Source AppleTalk Address:	65529.35
Unused:	00:00:00:00:00:00
Dest. AppleTalk Address:	65529.35

In some situations a Mac may not know which network numbers are valid. The most common is when a user switches their network connection from LocalTalk to Ethernet in mid-session. In these cases, the Mac will use AARP Probes to gain a node address using a network number in the startup range.

After the Mac gains an address in the startup range, it broadcasts a ZIP GNI. If the Mac has a saved zone name, it will verify that zone name in the ZIP GNI. If the Mac has no saved zone name, the ZIP GNI will be broadcast with a blank zone name.

On networks with more than one router, the Mac uses the data in the ZIP GNI Reply of the first router to respond to the broadcast packet. Typically, higher-speed routers (an APT ComTalk, for example) respond more quickly to a ZIP GNI than lower-speed routers, such as the Shiva EtherGate. If the routers are misconfigured and have different opinions concerning what zone names are valid, users might be informed that their home zone is invalid at this point in the boot process. This error message occurs because when a user selects a home zone using a ZIP GetLocalZones packet, the Mac doesn't broadcast the packet, but sends it to the address of the router in the RTMP Stub. This router is typically the last router heard from—either the last router to send that Mac a packet, or the last router to broadcast an RTMP packet.

There's a certain degree of randomness to the router that gets chosen to supply the zone list, and a lack of randomness for the router that's first to respond to the ZIP GNI broadcast. Because of this, the randomly chosen router may supply a zone name that's considered invalid by the router asked to confirm the zone name. This will happen only periodically, and although the users are notified that they're being placed in the default zone, many users don't report this occurrence; they simply reselect their home zone from the Network Control Panel and continue working. Only after it happens a few times might they bother to inform the network manager of the strange behavior. In most cases, it simply means that one of the zone names was mistyped into one of the routers.

Checking for duplicate service names on EtherTalk

After the node has gained a valid, unique address and confirmed its zone name, it will check to see if the service names it intends to register are unique in the zone to which it belongs. As in LocalTalk, it sends an NBP Broadcast-Request to the last router it heard from for this purpose. An interesting sidelight is that when a device switches its home zone mid-session, the sockets that the node has registered will switch zones along with the node. When the registered sockets join the new zone, however, they do so without the normal check for duplicate names. Although this usually has little consequence, depending on the importance of having a unique service name, it could cause a problem.

Router startup on EtherTalk

Router startup procedures on EtherTalk can vary a great deal. As in the case of the non-router, the device uses a combination of AARP, ZIP GNI packets, and NBP packets to determine that it has a unique and valid network address and that its service names are not duplicated elsewhere. Every router has a different way of doing this.

Beyond meeting the necessary requirements for being an EtherTalk node, a router must fulfill its routing mission: supplying network and zone information on request and forwarding packets through the internet at the request of non-routing nodes. We'll cover the details of this process later in this chapter.

To the extent that the router needs to cooperate and work with other routers, each seed router (one having network and zone information preconfigured) tries to determine whether its seed information is consistent with the information of other routers on the network. Different routers accomplish this task in different ways, depending on manufacturer, model, and version. The process mainly involves sending ZIP GNI broadcasts to verify zone names and examining the RTMP packets (routing table maintenance protocol) of other routers before commencing data exchange with other routers on the network.

The most important thing to know about your routers regarding this process is what they do when they encounter inconsistent configurations on the network. The router can make three basic choices:

1. Refuse to come on-line as a router, log the error, and do nothing.

2. Disregard the seed information and come on-line as a non-seed router. This is a relatively new approach. The router is then referred to as a "soft seed" router.
3. Ignore the discrepancies and come on-line anyway.

Routers that make the first response include Cisco and Wellfleet routers. The Cayman GatorBox can be configured for either response 1 or response 2. The Webster MultiPort and Shiva's FastPath 4 use response 3.

Maintaining Routing Tables and Zone Tables

Routers separate AppleTalk networks from each other and act as doorkeepers for the information that passes from network to network. Besides passing information, routers give a network its definition—its network address and zone names. When a network manager configures a router (usually off-line), he or she assigns network addresses and zone names only for the networks that will be directly connected to the router. The router must learn all other networks and zones from other routers already on the network. When the newly configured router is attached to the network, it exchanges its network definitions with the definitions that other routers hold and eventually learns the entire internet structure—the (logical) location of every network and zone as well as the method for sending packets to those networks and zones.

Some routers, called *non-seed* routers, learn from other routers not only about distant networks, but also about the local network. A non-seed router is configured without any information about the network to which it will be attached.

The method for data exchange among routers is handled by two separate processes working together: the RTMP Process and the ZIP Process. Every router, regardless of manufacturer, has both of these processes running inside it. The RTMP Process is guided by the algorithms defined in the Routing Table Maintenance Protocol, and the ZIP Process by the algorithms in the Zone Information Protocol. The ZIP Process learns the internet's network/zone mapping and the RTMP Process learns the locations of networks and the paths to them. Both processes store their knowledge in tables—a Routing Table and a Zone Information Table (ZIT).

> Note: The terminology may seem a little confusing because there is a *protocol,* a *process,* and a *table* for both the routing and zone functions. In each case, the *protocol* provides the rules and formats for the exchange

of data, the *process* does the computing and communication necessary to accomplish the exchange, and the *table* stores the information learned by the process. Another potentially confusing bit of terminology is the difference between the use of Ethernet and EtherTalk. We'll use Ethernet to indicate the physical network to which the router is attached, and, EtherTalk to indicate the AppleTalk network created on that Ethernet.

The RTMP Process always goes first. It constantly monitors the network. When the RTMP Process learns about a new network or a change to an existing network, it stores this new knowledge in the Routing Table. The ZIP Process doesn't monitor the network, but it does monitor the Routing Table. When it notices a change in the Routing Table, the ZIP Process makes a change in its table, the ZIT.

Protocol problems

RTMP and ZIP are two of the most troublesome protocols in the AppleTalk protocol family. The first problem is that since the protocols are optimized for smaller networks, they don't scale well to large internets. There's also considerable disagreement among vendors about how to interpret some of the rules outlined in the specifications for the RTMP and ZIP protocols (and in some cases, the rules are underspecified). The result is that in a few key situations, routers from different manufacturers react differently to identical information.

Because of the troublesome nature of these protocols and the products that implement them, it's doubly important that network troubleshooters understand the details of the interactions between routers as well as the interactions between Macs and routers.

Router configuration

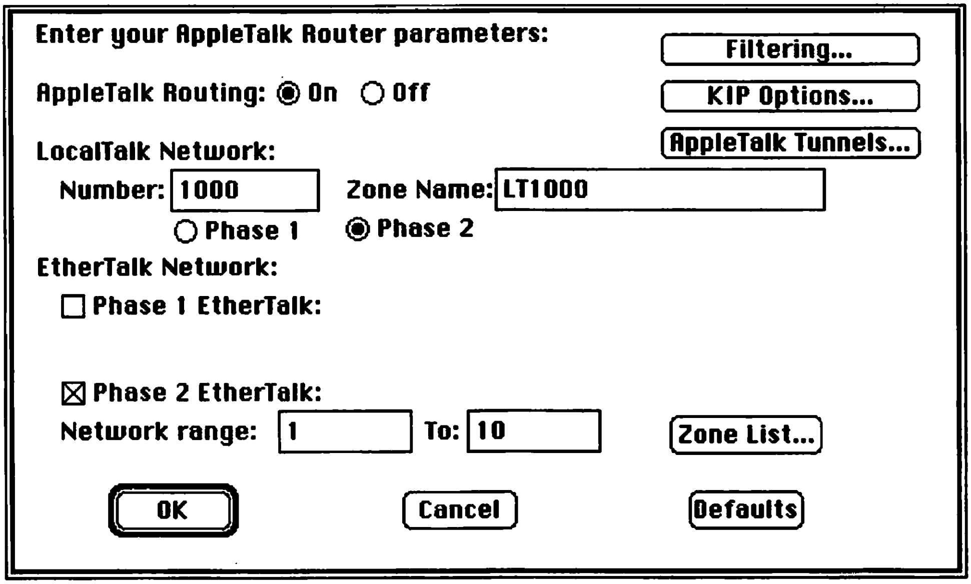
Routers are configured by means of bundled software. Every router has its own configuration management software; it can't be managed by other manufacturer's software. To describe configuration, we'll use GatorKeeper 2.0 configuration software and the GatorBox CS as an example. In addition, we'll confine the discussion to AppleTalk Phase 2 protocols to keep the flow of this section as linear as possible. (Besides, although a few networks still use Phase 1 protocols, their numbers are rapidly diminishing.)

The GatorBox, like many routers, has the ability to route data that uses other protocols besides AppleTalk (TCP/IP and DECnet), but it routes only AppleTalk data in its default configuration. Each protocol must be activated and configured in order to function.

You can enter AppleTalk information by using a configuration window that has one entry for each of the GatorBox's two ports: Ethernet and LocalTalk. Routers with more than two ports have a separate entry for each port.

You must enter network address and zone information for both ports. If you want one of the ports to be non-seeded, the network address entry for that port should be 0 and the zone list should be left blank. Router ports should be non-seeded only if there are other routers already on that network that can supply them with information.

Some routers are *self-configuring*: they generate network addresses and zone names automatically if none are supplied. This process typically includes some aspects of the non-seed algorithm in that it may check the network for other routers and get network and zone information from them, but in the absence of other routers, the self-configuring router will fill in its own default values. A non-seed router (that is, a router that has only non-seeded ports) must have another router to supply it with information in order to work. In the absence of other routers, a non-seed router that isn't self-configuring won't acquire network address and zone information.



Enter your AppleTalk Router parameters:

AppleTalk Routing: ● On ○ Off

LocalTalk Network:

Number: 1000 Zone Name: LT1000

○ Phase 1 ● Phase 2

EtherTalk Network:

□ Phase 1 EtherTalk:

☒ Phase 2 EtherTalk:

Network range: 1 To: 10

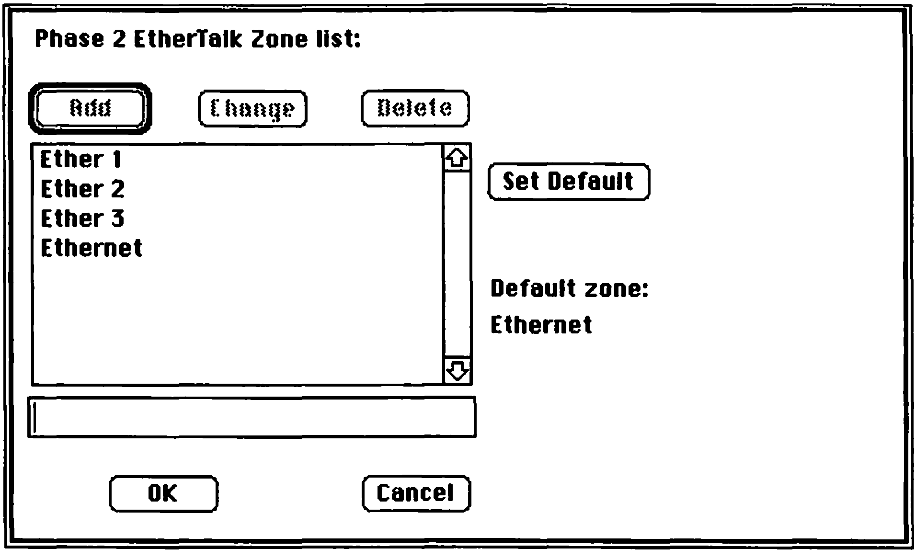
Figure 7-1. The GatorBox configuration window.

In the GatorBox example (shown in Figure 7-1), the LocalTalk network is assigned the network address of 1000 and the zone name of LT1000. The Ethernet network is assigned the network address range of 1–10, and the zone list for the EtherTalk network contains four zone names: Ethernet, Ether 1, Ether 2, and Ether 3. These names are entered into the window made visible when the Zone List... button is clicked (shown in Figure 7-2).

Routers are generally configured off-line, powered off, connected to the live network, and then powered back on.

Router startup

A router, like any network node, must take an AppleTalk address and register its service name. Both ports need an address as each is a member of a different network. This process was described more thoroughly in "Device Startup," near the beginning of this chapter.



Phase 2 EtherTalk Zone list:

Ether 1
Ether 2
Ether 3
Ethernet

Default zone:
Ethernet

Figure 7-2. Entering the Zone list for EtherTalk.

The Routing Table

The RTMP Process keeps a routing table that lists all of the networks that the router knows about. The router will begin its service knowing about only the two networks described by the network manager when he or she configured the router.

A router's routing table lists, for every network it knows, the network address, its hop distance (the number of routers situated between the router keeping the table and that network), the port that the router uses to send a packet to that network, the next router in the path to get to that network, and the router's rating of the reliability of the path to that network. The Routing Table for the GatorBox shown in Figure 7-3 reflects its knowledge at the moment that the GatorBox enters service. Of course, the table will grow as the GatorBox learns of other networks.

Network Number	Port	Hop distance	Next Router	Reliability
1000	LocalTalk	0	—	Good
1-10	Ethernet	0	—	Good

Figure 7-3. The GatorBox Routing Table at startup.

Because both networks that the GatorBox currently knows are local (directly attached) networks, the hop distance to these networks is 0 and the Next Router entry is not applicable. The reliability rating reflects how frequently the GatorBox has heard from the Next Router in the path by which a network is reachable. With directly attached networks, however, the value will always be Good. For networks not directly attached, the reliability entry will start out Good but can be reduced to Suspect or Bad if the Next Router doesn't regularly report the network's reachability.

The Zone Information Table

The ZIP Process keeps a table that lists each network in the Routing Table and the zone name or list associated with that network. For the moment the GatorBox enters service, the table contains only the entries for the EtherTalk and LocalTalk networks, as shown in Figure 7-4.

Network Number(s)	Zone Name(s)
1000	LT1000
1-10	Ethernet, Ether 1, Ether 2, Ether 3

Figure 7-4. The GatorBox Zone Information Table at startup.

As previously mentioned, the ZIP Process monitors the Routing Table. When the Routing Table changes, the ZIP Process makes the appropriate change in the ZIT.

Route information broadcasts

Every AppleTalk router is required to broadcast, every ten seconds, an RTMP Response packet that lists the networks that it knows about along with the number of hops to those networks—one broadcast sent out each port every ten seconds. The hop distance is the number of routers between a router and a network.

Suppose the GatorBox, in order to send a packet to network 2305, must send the packet to Router A, which in turn sends it to Router B, which in turn sends to Router C, which is directly connected to network 2305. Router C will send the packet directly to the destination device on network 2305. Network 2305 is zero hops away for Router C, one hop for Router B, two hops for Router A, and three hops for the GatorBox.

In the beginning of a router's service, however, it knows only about the networks to which it's directly connected (networks with a hop distance of 0), so these are the only networks that will be listed in the first RTMP Response Packet it sends.

When the GatorBox first comes on-line, it broadcasts an RTMP Response packet out each port that lists two tuples: one describes the LocalTalk network and the other describes the Ethernet network. (A tuple is a pair of data values: the network address and the number of hops.)

> Note: We'll use (1000,0 ^ 1–10,0) to express the information in this packet and we'll use this shorthand consistently to describe RTMP Response Packets: within a tuple, a network and its hop distance will be separated by a comma; tuples will be separated from each other by the " ^ " character. Translated to English, the RTMP Response Packets shown above says, "Network 1000 is 0 hops away from me and network 1 through 10 is 0 hops away from me."

In Phase 2, when routers send their RTMP Response packets, they perform *split horizon broadcasting*. Instead of listing every network they know about (as in Phase 1), the routers broadcast to each port the network on that port, followed by a listing of all networks available on their other ports. Each network is described in a tuple. Most routers list the network they're sending to in the first tuple, but this isn't required.

In the case of the GatorBox, if there are other routers on the Ethernet that report the presence of other networks, these networks will be added to the GatorBox's Routing Table and will be listed in the tuples of the RTMP Response Packets sent to the LocalTalk port. According to the rules of split horizon broadcasting, however, they won't be described in tuples sent out the GatorBox's EtherTalk port. Unless there are other routers and other network connected to the GatorBox through its LocalTalk network, the EtherTalk tuple will remain (1000,0 ^ 1–10,0).

The reliability rating in a router's Routing Table is tied to the regularity of the receipt of RTMP Response Packets. Each network entry in the Routing Table has an associated "aging timer" that indicates how long it has been since the router received an RTMP tuple describing that network. The aging timer is refreshed every time the router receives the tuple, and the reliability remains Good. If 20 seconds lapse after a tuple, the reliability is demoted to Suspect. At 60 seconds, the reliability rating is reduced again to Bad. Some routers will eventually delete a Bad entry, but deletion isn't required.

The RTMP process

Rather than describe the RTMP Process occurring as other routers receive the GatorBox's RTMP Response packets, we'll focus on what the GatorBox does to build and maintain its tables as it receives packets from other routers. You should be aware, however, that just as the GatorBox changes its tables as it learns about networks from other routers, the other routers change their tables based on information that the GatorBox supplies. Network 1000, with the zone name LT1000, is a new network as far as the other routers are concerned; that's how they'll enter it into their tables.

Besides sending out its first RTMP Response Packet, the GatorBox will also receive RTMP Response Packets from other routers on the Ethernet. The destination address of these packets is 0.255.1, which means, "the routing socket on every node on this network." Socket 1 is a Statically Allocated Socket specially designated for routing functions.

> Note: AppleTalk devices that are not routers (each user's Macintosh, for example) have a routing process on socket 1 as well, but it has only enough functionality to capture the network address designation and the address of the router. It keeps these values in its RTMP Stub.

For each RTMP Response packet that a router receives, the GatorBox's RTMP Process will examine each tuple in the packet to see if a change in its routing table is warranted. There are three situations in which the router would make a change:

1. On hearing about a network for the first time, it adds a new entry in the Routing Table. It lists the network number advertised in the tuple and the identity of the port that received the packet, adds one to the hop count shown in the packet and uses the result for the Hop Distance, lists the router that sent the packet as the Next Router, and sets the reliability rating to Good.

2. On hearing about a network that's already known, the RTMP Process compares the hop count in the tuple (first adding one more hop to the number) to the hop distance in its table. If the hop count in the tuple indicates that the router sending this packet has a longer or equal path, the RTMP Process simply refreshes the reliability rating. If the hop count shows a better path to the network (a lower number of hops), the RTMP Process changes the Hop Distance and Next Router data for that entry to reflect the improvement. It also sets the reliability entry to Good if the entry is currently at a different setting. Note that RTMP uses hop distance alone as the basis for routing decisions. It doesn't consider link speed, reliability (based on experience), or the link's level of activity.

3. On not hearing about a network in its table, it can downgrade that network's reliability rating. The process of downgrading the reliability rating (and eventually removing the entry), not well defined in AppleTalk, is one aspect of this process that varies as different manufacturers implement RTMP in their router products.

The first RTMP Packet is received

After the GatorBox completes its startup activity, its view of the world looks like Figure 7-5 (also reflected in the routing and zone tables shown earlier).

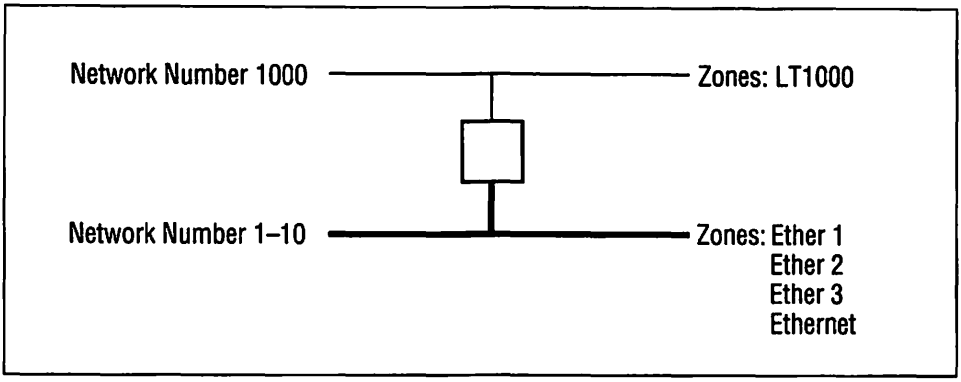


Figure 7-5. What the GatorBox knows at startup.

Then it receives its first RTMP Response Packet from an Ethernet node:

From: 3.147.1 To: 0.255.1
Routing Tuples: (1–10,0; 375,0)

Checking the first tuple, the GatorBox's RTMP Process sees that Router 3.147 has no better way to get to network 1–10 than the way it currently knows. In the second tuple, network 375 is a network that the GatorBox doesn't know about, so it creates a new entry in its Routing Table, which appears as shown in Figure 7-6.

Network Number	Port	Hop distance	Next Router	Reliability
1000	LocalTalk	0	—	Good
1-10	Ethernet	0	—	Good
375	Ethernet	1	3.147	Good

Figure 7-6. The Routing Table after the first RTMP Packet is received.

Because the ZIP Process is monitoring the Routing Table, it notices the addition of the new network and makes a new entry in the Zone Information Table. It doesn't, however, have any zone information for network 375 because RTMP Response Packets don't contain zone information, just network information. Nonetheless, it will modify its ZIT, placing NIL in the zone name column.

Network Number(s)	Zone Name(s)
1000	LT1000
1-10	Ethernet, Ether 1, Ether 2, Ether 3
375	NIL

Figure 7-7. ZIP makes a change in the ZIT when the Routing Table changes.

The NIL value in the zone name is the cue that prompts the ZIP Process to inquire about the zone information for network 375. To do so, it asks the Next Router in the Routing Table, node 3.147. The packet that it sends is a ZIP Query Packet, whose English translation is, "What zone names are associated with network 375?" Node 3.147 responds by saying "Network 375 has the zone name 'Eastern Operations'" in the ZIP Reply Packet.

> Note: ZIP Queries and Replies are sent only when new routers and networks come into existence. You shouldn't see them in steady state networks.

After receiving the ZIP Reply packet, the GatorBox can update its Zone Table, as shown in Figure 7-8.

Network Number(s)	Zone Name(s)
1000	LT1000
1-10	Ethernet, Ether 1, Ether 2, Ether 3
375	Eastern Operations

Figure 7-8. The ZIP Reply Packet tells the ZIP Process the name of the zone.

The GatorBox will now see the Internet as shown in Figure 7-9.

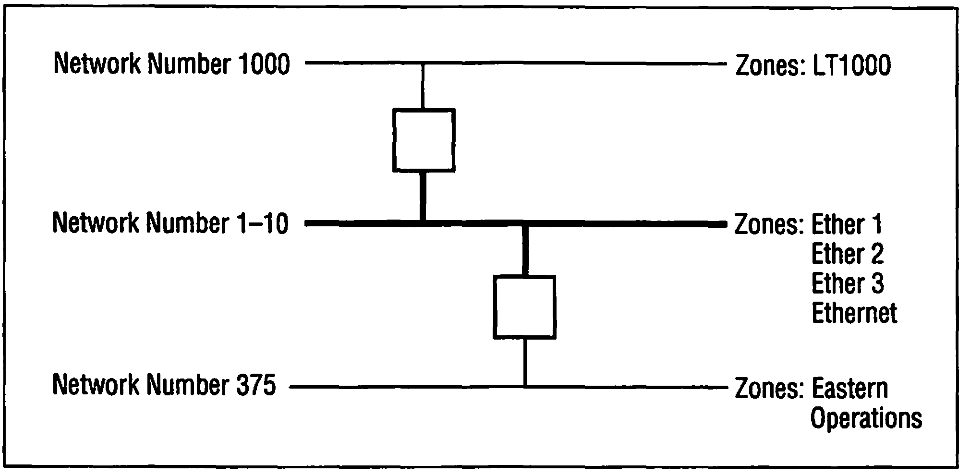


Figure 7-9. The GatorBox has learned its first new network.

Notice, however, that the GatorBox never asks router 3.147 for the zone names that went with the EtherTalk network, network 1–10. That's because network 1–10 is already known. This makes some sense because it cuts down on the amount of communication necessary, but if the two routers have different zone lists because of a configuration mistake, the routers would never discover the discrepancy.

The process continues

The process of building the Routing Table and the ZIT continues until the GatorBox has learned every network in the internet. It then sends a ZIP Query and receives a ZIP Reply from every router on the Ethernet as it learns of their networks and zones. It also receives a ZIP Query and responds with a ZIP Reply as those routers learn of the GatorBox's LocalTalk network.

Changing zone names

Because of some of the idiosyncrasies of AppleTalk, there are special considerations when the network manager wants to change a zone's name. We'll consider two such changes. The first involves changing the zone name of the GatorBox's LocalTalk zone from LT1000 to Western Operations. The second change is to add the zone name Operations to the EtherTalk zone list.

You'll need to change the name of the GatorBox's LocalTalk network in such a way that the other routers will ask for the new zone name after the change is made. If the other routers don't ask for the new zone name, Western Operations will never become part of their zone lists. But routers ask zone names only when the routing table shows a new network, so you must make the other routers consider LocalTalk network 1000 a new network. There are two ways to accomplish this: change the network number along with the zone name, or wait long enough so that the other routers will have "forgotten" about network 1000.

The safest way of accomplishing the name change is to change the network number to 1001. After the change, the GatorBox will be asked by all the other routers for the zone name network 1001. Another advantage of this method is that the change happens as quickly as the GatorBox can reboot.

The other way to change the GatorBox's LocalTalk zone name is to wait for all of the other routers to age out network 1000 before you bring the GatorBox back on-line. That way, when the GatorBox begins reporting network 1000 again, the other routers will treat network 1000 like a new network and will send a ZIP Query to get the zone name. However, if you don't wait long enough, they'll simply refresh the reliability rating and use the old zone name.

How long should you wait to make sure all the routers have aged out a network? This question is a little like asking, how long after a meal you should wait before swimming There's disagreement over the definitive answer. The answer depends on the manufacturer, model, and version of the other routers, but most experts agree that 20 minutes should always be enough time and 10 minutes is probably enough time. In any event, it's a good idea to check the routers after performing this change to make sure they're all in accord.

The second zone name change you might want to accomplish is to add a zone name to the list of zones on Ethernet. Unfortunately, this is much more difficult, because you'll have to reconfigure and restart every single seed router and then restart every non-seed router—a tremendous amount of work in a large internet. Any routers connected downstream from the routers on the Ethernet would have to be handled as described above: either wait long enough for them to age out network 1–10, or change the network number. As with any internet configuration change, you should confirm this change by checking router configurations, using one of the router techniques outlined in Chapter 6.

Note: Apple has defined a way to change zone names automatically by using the ZIP Notify Packet, but at the time of this writing no router has implemented the ZIP Notify function. The ZIP Notify Packet says to a router, "For network <net address>, where you used to use the zone list of <old zone list>, begin using <new zone list> instead." Until this function is implemented, however, zone name changes must be handled as described above.

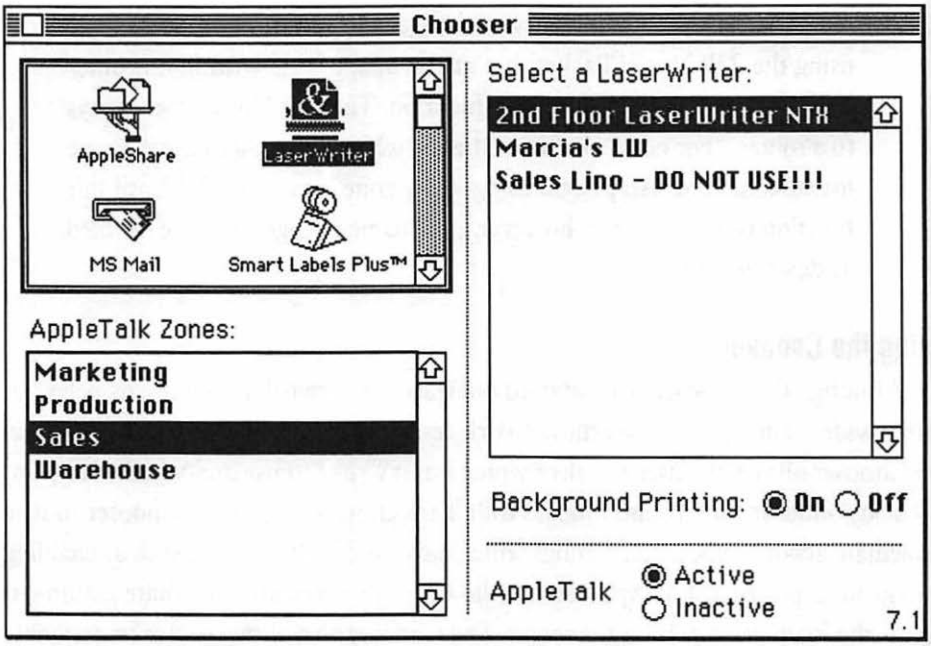
Using the Chooser

Although the Chooser is a standard interface for network resources, its behavior varies widely among the different network resources it serves. For the LaserWriter, the Chooser allows the user to select which LaserWriter to use and whether to print in background or foreground mode. Not all selections made in the Chooser induce immediate action; some affect settings which have visible effects later, such as enabling background printing. For AppleShare, the Chooser takes an immediate action—it begins the login process for a file server. The Chooser also allows a user to designate a volume for automatic mounting at startup.

The Chooser is the connection utility that allows users to locate and invoke network resources such as LaserWriters and AppleShare file servers. The Chooser works with a file called a Chooser Extension, which is a software driver for a particular network resource. Chooser Extensions are special system files placed in the user's Extensions Folder (or in the System Folder in System 6). The Chooser also selects and manages non-network resources connected to either of your Mac's two serial ports.

Figure 7-10 shows the icons for the Chooser Extensions LaserWriter, AppleShare, MS Mail, and Smart Labels Plus. In the Chooser interface, users select which kind of service they would like to locate and the zone in which they would like to search. With the settings shown in the figure, the Chooser searches for all LaserWriters in the Sales zone. It broadcasts to each network comprising the Sales zone, instructing any device with an active process of the type LaserWriter to respond to this Chooser. The Chooser then displays the names of the processes that respond.

Some other network applications use proprietary utilities similar to the Chooser to find and invoke resources. Examples are Farallon's Timbuktu (see Figure 7-11), Claris FileMaker Pro, and Sitka's TOPS. These Chooser-like utilities share many traits with the Chooser; many processes described in this chapter therefore apply to them as well.



Chooser

Select a LaserWriter:

2nd Floor LaserWriter NTX
Marcia's LW
Sales Lino - DO NOT USE!!!

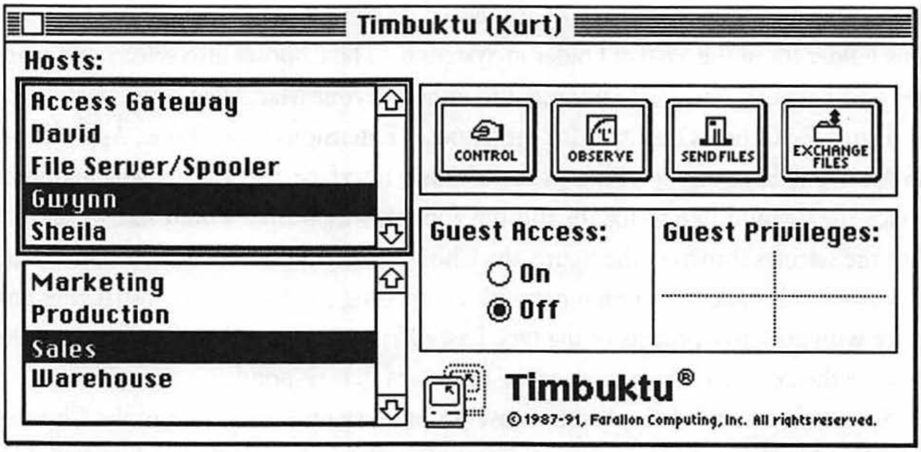
AppleTalk Zones:

Marketing
Production
Sales
Warehouse

Background Printing: ● On ○ Off

AppleTalk ● Active ○ Inactive

Figure 7-10. AppleTalk network resources are uniquely identified by a three-component service name: Name, Type, and Zone.



Timbuktu (Kurt)

Hosts:

Access Gateway
David
File Server/Spooler
Gwynn
Sheila

Marketing
Production
Sales
Warehouse

Guest Access: ○ On ● Off

Guest Privileges:

Figure 7-11. Farallon's Timbuktu has a Chooser-like utility to select network resources of the type "Timbuktu Host." The zone and name components of the service names appear in the utility.

The beauty of the Chooser and utilities like it is that they allow users to navigate the resources in the internet while remaining completely ignorant of the complexities of the AppleTalk addressing system. Users need only be concerned with service names and zone names; they can be completely unaware of network, node, or socket addresses.

This is made possible through the use of Name Binding Protocol (NBP), which establishes a unique link between a character-based service name (name, type, and zone) and a numeric AppleTalk address (network, node, and socket). NBP is discussed earlier in this chapter, where we described how software processes establish their service names and addresses on startup. In this section, we will focus on how NBP is used to locate and invoke processes after they're in service.

Although the focus of this section is on the underlying mechanics of the Chooser and its functions, it's also a good place to mention a few things we've noticed about how users use (or misuse) the Chooser. Because the Chooser's characteristics and behavior vary depending on which extension is selected, the Chooser is often confusing to novice users. They often don't know which aspects of a resource's operation they can control, and when they do know, they sometimes don't remember which aspects of a network resource they control by using the Chooser, which they control by using a Control Panel, and which they control by menu choices within an application.

The list-oriented nature of the Chooser also contributes to the confusion, particularly when network resources don't have names that have an immediate meaning for the user. Sometimes you can solve this problem by creating some simple reference documents for the users that tell them what options they can control and how to set those options.

Another way you can make life simpler for your users is by programming in a few macros to let users select services by keystrokes. For example, you might give users the ability to switch between a LaserWriter and a Linotronics printer, or the ability to mount a file server by pressing a particular function keys, using a macro utility like CE Software's QuicKeys or UserLand's Frontier. You can then place labels on the users' keyboards identifying which function key performs which function.

Basic Chooser operation

When the user opens the Chooser on a network with routers, the Chooser displays a zone list with the user's home (default) zone highlighted. The Chooser also displays a list of icons, one for each of the Chooser Extensions in the user's Extensions Folder, as shown in Figure 7-12.

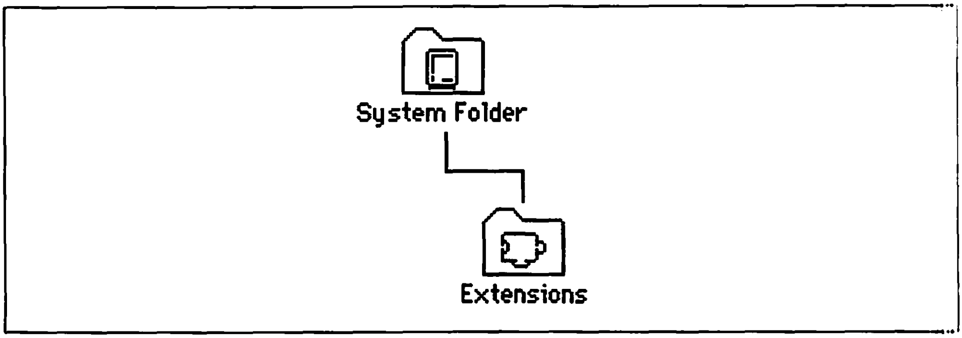
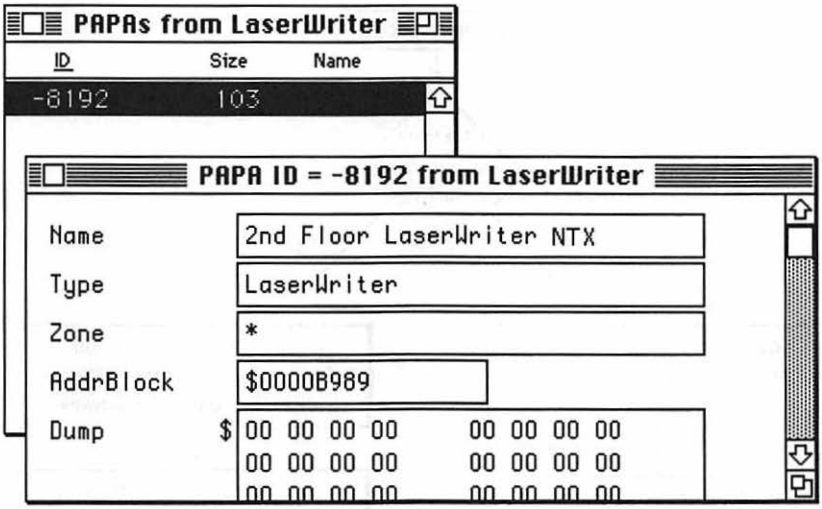


Figure 7-12. In System 7, Chooser Extensions become active when they are placed in the Extensions Folder, which is inside the System Folder.

When the user selects the icon for the type of service he or she wants to work with, the Chooser displays all available services of that type within the highlighted zone. One exception occurs when the user selects the LaserWriter Chooser Extension; the Chooser then displays all LaserWriter services in the zone of the previously selected LaserWriter.

From the displayed list of names, the user selects the service that he or she wants to use, along with any options that the Chooser may allow. The selection and options are then stored for later use. For some extensions, the Chooser stores the user's choice in a resource in the extension file. In System 7, for example, the Chooser stores the name, type, and zone name of the user's preferred LaserWriter in the PAPA resource in the LaserWriter (see Figure 7-13). When the user prints from an application, the LaserWriter Extension will locate and use the device indicated in the PAPA resource. The choice of printer driver (for example, LaserWriter, ImageWriter, or LabelWriter) is stored in the System file.

Other Chooser Extensions store the user's Chooser selections in other, separate files. For example, when a user uses the AppleShare Extension to mount an AppleTalk Filing Protocol Server (AFP Server) volume, the Chooser asks the user if he or she would like to automatically mount this volume on future startups. If the user answers yes, that preference is stored in the BMLS resource in the AppleShare Prep file, which is located in the Preferences Folder. The user may also store his or her name and (encrypted) password in this file. When the system boots, the AppleShare Chooser Extension checks the AppleShare Prep file and mounts the volumes specified in this resource if they're available.



PAPAs from LaserWriter

ID	Size	Name
-8192	103	

PAPA ID = -8192 from LaserWriter

Name: 2nd Floor LaserWriter NTX

Type: LaserWriter

Zone: *

AddrBlock: $0000B989

```
Dump   $ 00 00 00 00     00 00 00 00
         00 00 00 00     00 00 00 00
         00 00 00 00     00 00 00 00
```

Figure 7-13. The PAPA resource in the LaserWriter Chooser Extension contains the name of the LaserWriter chosen by the user.

In System 7, the system will create an AppleShare Prep file if the user doesn't already have one. This is a great way to cure the problem of when the AppleShare Prep file is corrupted or contains undesirable values—simply delete the old file and let the system create a new one. This doesn't work in System 6, where you must install a fresh AppleShare Prep file from an original system disk.

The role of routers in using the Chooser

When it's opened, the Chooser must ask a router for a list of zones before it can display a zone list (see Figure 7-14). On a network without routers, the Chooser window doesn't have a zone list. The Chooser can tell whether or not there's a router on the network by examining the RTMP Stub All AppleTalk devices, including Macintoshes, that are not routers maintain an RTMP Stub that holds the network and node address of a router. This RTMP Stub is created and refreshed by the periodic RTMP Packets that all routers are required to broadcast at 10-second intervals.

Although non-routing devices ignore most of the information in these packets, they store in their RTMP Stub the network and node address of the router that sent the RTMP Packet. On a network with more than one router, the identity of the router stored in the

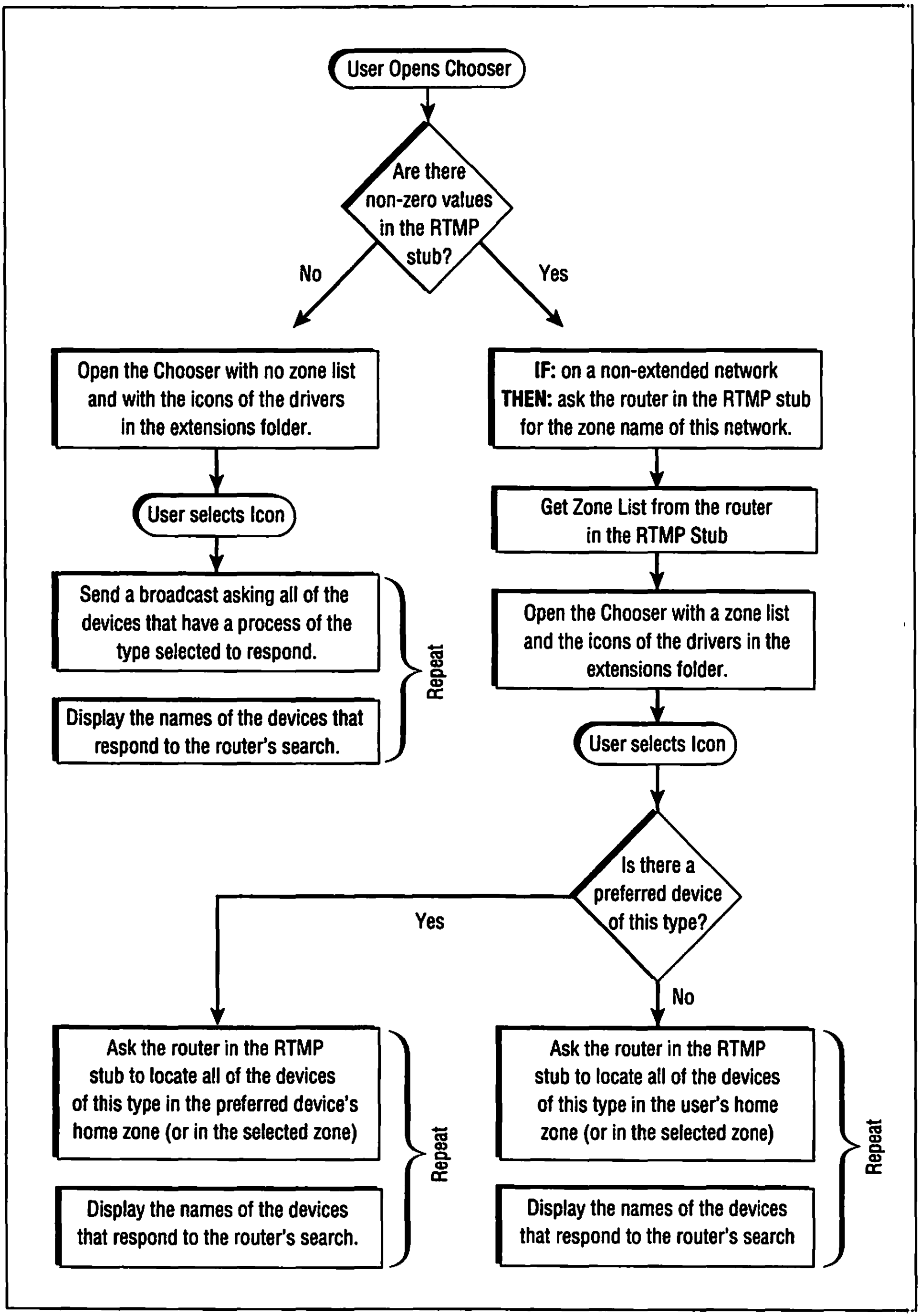

User Opens Chooser
Are there non-zero values in the RTMP stub?
No
Yes
Open the Chooser with no zone list and with the icons of the drivers in the extensions folder.
User selects Icon
Send a broadcast asking all of the devices that have a process of the type selected to respond.
Display the names of the devices that respond to the router's search.
Repeat
IF: on a non-extended network THEN: ask the router in the RTMP stub for the zone name of this network.
Get Zone List from the router in the RTMP Stub
Open the Chooser with a zone list and the icons of the drivers in the extensions folder.
User selects Icon
Is there a preferred device of this type?
Yes
No
Ask the router in the RTMP stub to locate all of the devices of this type in the preferred device's home zone (or in the selected zone)
Display the names of the devices that respond to the router's search.
Repeat
Ask the router in the RTMP stub to locate all of the devices of this type in the user's home zone (or in the selected zone)
Display the names of the devices that respond to the router's search
Repeat

Figure 7-14. Chooser process flow.

RTMP Stub will change every time one of the routers sends an RTMP Packet. Whichever router address is in the RTMP Stub at the time the Chooser is opened will be the router the Mac will use to get the zone list. If the RTMP Stub has no values in it when the Chooser is opened, the Chooser will open without a zone list.

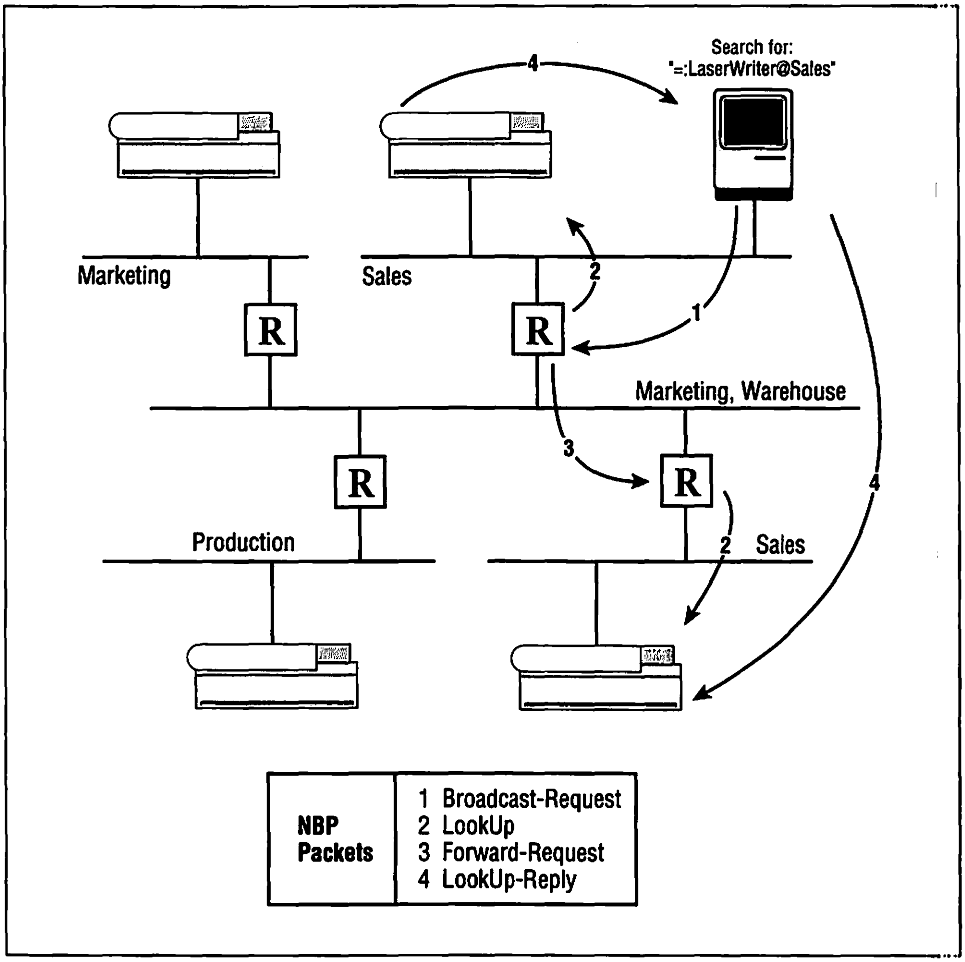
If no router is present (indicated by a null value in the RTMP Stub), the Mac will broadcast an NBP LookUp Packet to its network. Any software process that has a service name matching the query will respond with an NBP LookUp Reply Packet, which supplies the Internet Socket Address (ISA) of the process as well as its complete service name. The Object Name portion of all of the Reply Packets is then displayed in the Chooser window. The LookUp and Reply process will repeat as long as the icon is selected.

NBP Search mechanics

On a network with routers, the Chooser will ask the router currently in the RTMP Stub to perform the search. If the Mac asks the router to search the Sales zone for LaserWriters, the router will then consult its Zone Information Table to determine which networks are included in the Sales zone. It will then consult its Routing Table to determine how to send broadcast packets to those networks. Even if the Mac is searching in its home zone, a router must be involved since the home zone may also include other networks located elsewhere on the internet. If a router must use other routers to reach distant networks, it will use the NBP Forward-Request packet (Fwd-Req), shown in Figure 7-15.

The Chooser will send an NBP Broadcast-Request (Br-Req) packet to the router that lists the service type it's looking for and the zone it wants to search. The router will then send an NBP LookUp Packet to its network, asking all devices that have a service registered with that type to respond. The Internet Socket Address (ISA) of the Chooser performing the search is recorded inside the LookUp packet so that the responding node will know where to send any replies.

In NBP terminology, the LookUp Packet is a "query" to the distributed database of the service names held collectively by the nodes. Each node receiving the LookUp broadcast will examine its own list of service names for a match on the LookUp criteria and send an NBP Reply Packet to the ISA contained in the LookUp. The NBP Reply Packet carries the ISA of the service as well as the complete service name. A Chooser's query for LaserWriters appears as =:LaserWriter@*, or in English, "A LaserWriter with any name in this zone." (The equal sign in the name field means "any name" and the asterisk in the zone field means "this zone." On a network with no routers, of course, "this zone," "this network," and "this cable" all have identical meaning, so a simple broadcast is sufficient.) The Chooser then displays the names of all of the processes that respond.



NBP Packets	1 Broadcast-Request 2 LookUp 3 Forward-Request 4 LookUp-Reply

Figure 7-15. All four kinds of NBP packets may be required for a zone-wide search for LaserWriters.

If the RTMP Stub does contain a router's address, that router is asked to supply a zone list. When the user selects an icon, a router must also search for services because the Chooser wants to find all of the services within a zone, regardless of whether they happen to be on the same network as the node making the request.

Also, since the Chooser searches for and displays a list of all devices of a certain type within a zone, a network router must be involved in the search because Macs don't understand the relationship between zones and networks. If the Mac asks the router to search the Sales zone for LaserWriters, the router will consult its Zone Information Table to determine which networks are included in the Sales zone. It will then consult its Routing Table to determine how to send broadcast packets to those networks. Even if the Mac is searching in its home zone, a router must be involved since the home zone may also include other networks located elsewhere on the internet.

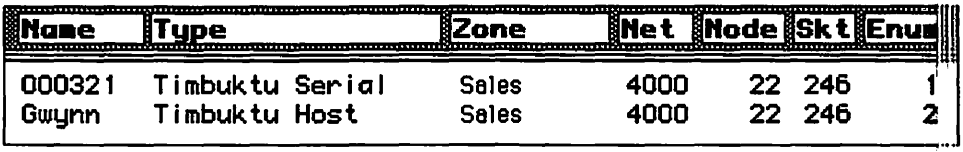
Named services

To understand the Chooser it's important to understand the concept of *named services*. Named services are software processes that have used NBP to associate (bind) a service name to their AppleTalk Internet Socket Address (ISA). A process uses NBP to establish a name when it needs to make contact with another process that doesn't know its numeric address. When the contact is made, the two processes exchange their ISAs and no longer need the names.

The notation generally used to refer to the service name of a process is Name:Type@Zone, while the notation for an ISA is Net.Node.Socket. Using this notation, for example, you'd say that the service Fred's LaserWriter:LaserWriter@Sales is bound to ISA 12.185.128. Some services also use an enumerator, which is necessary when more than one service name is bound to a particular socket.

For example, in the excerpt from CheckNet shown in Figure 7-16, Timbuktu shows two service names, both registered to the same ISA: 4000.22.246. The first name in the list is part of Timbuktu's license agreement enforcement. The Object Name portion of the service name comprises the first six digits of Timbuktu's 12-digit serial number. Timbuktu registers this service name when the Timbuktu Extension loads during startup. The second service name, Gwynn:Timbuktu Host@Sales, is registered only if Gwynn (the user) is currently allowing guest access to her Mac. The Timbuktu Chooser-like utility (shown in Figure 7-11) looks for a service type Timbuktu Host and displays the object names of the processes that reply.

There's something misleading in Figure 7-16. Timbuktu isn't actually registered in the Sales zone. But CheckNET displays the zone name Sales in this list, despite the fact that Timbuktu is registered with a zone name "*," because CheckNET knew that it was searching the Sales zone when the reply was received.



Name	Type	Zone	Net	Node	Skt	Enu
000321	Timbuktu Serial	Sales	4000	22	246	1
Gwynn	Timbuktu Host	Sales	4000	22	246	2

Figure 7-16. Software processes can register more than one name on a socket. This is an excerpt from a display using Farallon's CheckNET.

Although you can't tell from the figure or from CheckNET, network 4000 is a LocalTalk network. As a rule, processes on a LocalTalk network, Timbuktu included, don't register themselves with a specific zone name, but rather with the zone name "*," which means "this zone." LocalTalk devices generally don't know which zone they're in. At the time the Chooser is opened, the Mac asks the router which zone it's in so it will know which zone to highlight in the zone list (ZIP GetMyZone). Consequently, to most processes on LocalTalk, all zone names match "this zone." A notable exception is the White Pages service running on a (CE Software) QuickMail server, which performs a ZIP GetMyZone before registering its name and *does* register in a particular zone.

Let's look at how this affects the functioning of the Chooser.

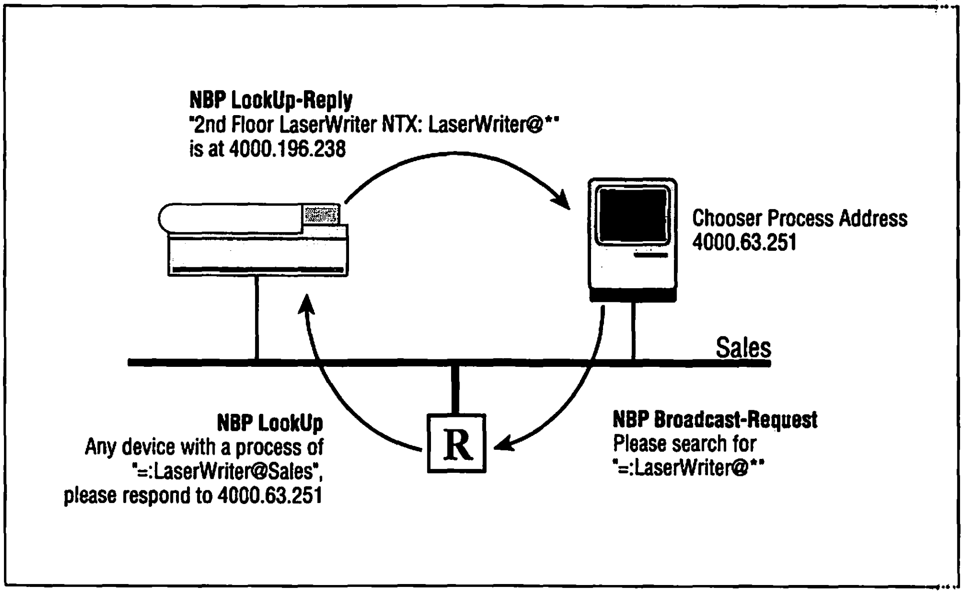


Figure 7-17. A Chooser searches its own zone for LaserWriters. Although the LaserWriter is registered in the "*" zone, the Chooser shows it in the Sales zone.

When a LocalTalk Mac's Chooser searches its own zone for LaserWriters (see Figure 7-17), it asks the router to find all LaserWriters in this zone—the "*" zone. The router translates this into the actual zone name in the LookUp, but the LaserWriter responds with the "*" zone in its name.

Chooser troubleshooting example

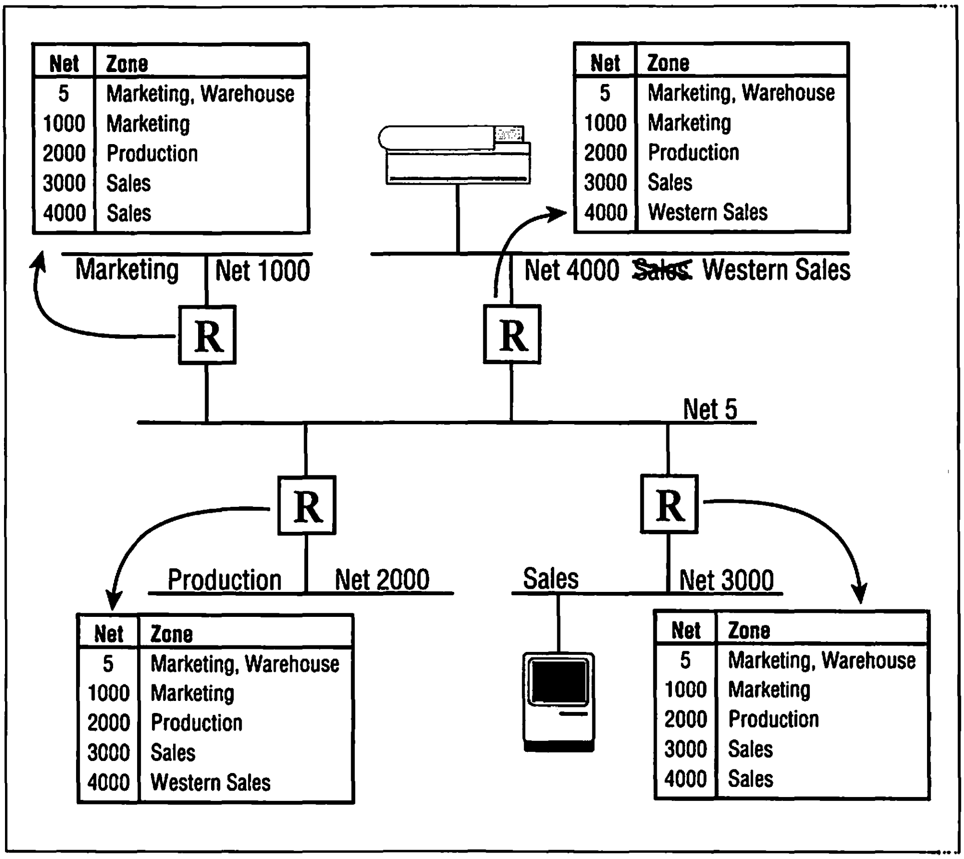
The fact that a service will respond with "*" rather than a zone name can have rather large implications, depending on the situation. You need to keep it in mind when troubleshooting, especially when considering NBP searches like the ones the Chooser performs.

For example, imagine an internet where one of the routers was reconfigured because the network manager wanted to change the LocalTalk network's zone name from Sales to Western Sales. A rule of thumb in this instance says that you should leave this router off-line for ten minutes times the number of hops away to the farthest router. This gives all routers plenty of time to "forget" about the network, so that when the router comes back on-line, all the routers will think there is a new network and ask for its zone name. If you don't leave the router off-line long enough, some of the routers may not notice that it went off-line and will continue to use the old zone name.

In the internet shown in Figure 7-18, the time zone of the router in the upper-right network was changed. The farthest router is only one hop away, so the rule of thumb says that you should wait ten minutes before activating the router again. In the example, however, the network manager failed to wait long enough, and only one other router learned of the zone name change. The Zone Tables of all routers are shown.

If you were troubleshooting printing problems on this internet, which networks do you think would be able to print to the LaserWriter in network 4000? You'd probably guess that users would be able to print to this LaserWriter from network 4000 and network 2000, but you may be surprised to learn that users could print to this LaserWriter from the other LocalTalk networks as well. The key is that the LaserWriter registers its name with the "*" zone.

If your Mac were attached to network 5, however, you would have only sporadic success printing to this LaserWriter. When you open the Chooser, you'd have a 50 percent chance of seeing Western Sales in the zone list because two of the routers know about that zone while two routers haven't heard of it.



Net	Zone
5	Marketing, Warehouse
1000	Marketing
2000	Production
3000	Sales
4000	Sales

Net	Zone
5	Marketing, Warehouse
1000	Marketing
2000	Production
3000	Sales
4000	Western Sales

Net	Zone
5	Marketing, Warehouse
1000	Marketing
2000	Production
3000	Sales
4000	Western Sales

Net	Zone
5	Marketing, Warehouse
1000	Marketing
2000	Production
3000	Sales
4000	Sales

Figure 7-18. A change in the zone name of one of the routers was not noticed by one of the other routers on the internet.

When you select the LaserWriter icon, you will again use one of the routers to perform the NBP search, and will again have a 50 percent chance of getting a router that knows about Western Sales. Whether or not the Chooser can locate the LaserWriter depends on whether the router you use for the zone list and the router you use for the search agree on the existence of Western Sales. Whether they think it does or doesn't exist, if they agree, the Chooser will locate that LaserWriter and write its full service name into the PAPA resource in the LaserWriter Chooser Extension. However, the zone name can be either Sales or Western Sales depending on which routers you use. Later, when you print, you'll again have a 50 percent chance of locating the LaserWriter.

The Chooser as a troubleshooting tool

In System 6, the Chooser repeats its search every 1.5 seconds as long as the Chooser Extension icon is selected. In System 7, this interval decreases as time passes. Each time the search is repeated, the list of names in the Chooser is refreshed. Ideally, the name list should always show the same names since each search should yield the same results. Sometimes, however, the list will change while you watch it. How the name list changes can give you a clue to what may be wrong with your network.

Symptom #1. The names jump around wildly in the list. This happens only in System 6, and it means that you have too many services of a given type in the zone. The System 6 Chooser can hold just under 500 characters in the name list, and a maximum of 16 names. When the Chooser receives replies in excess of these limits, the result is this rather entertaining symptom. There are two cures: either use System 7 or reduce the number of services of that type in that zone.

Symptom #2. The names appear and disappear, with no jumping around. This oddity typically occurs when you have wiring problems. This symptom implies that the NBP search isn't consistently successful. Either the LookUp packets are getting lost on their way to the network service or the reply packets are getting lost on their way back to the Chooser. In either case, you should check the wiring by using the Echo Test or the Progressive Echo Test.

A less common reason for this symptom is that the routers are in disagreement. To check this possibility, use Neon Software's RouterCheck or one of the methods discussed in Chapter 6.

Printing from an Application

Applications allow users to input, manipulate, and display data. Macintosh applications create a series of QuickDraw commands to display an image. QuickDraw, the Mac's native imaging language, is a flexible language that can describe a rich set of text objects, graphic objects, and bitmaps. When you see an image on the screen, it appears because the application has taken the user's data, generated the appropriate QuickDraw commands, and sent them to the appropriate display software and hardware.

Printing happens in much the same way, except that the QuickDraw commands are sent to a group of software processes collectively called the Printing Manager (see

Figure 7-19). The application structures the QuickDraw commands it sends to the Printing Manager according to the user's formatting and Page Setup choices. The Printing Manager handles and forwards the incoming QuickDraw commands according to the user's Chooser selections—which printer driver the user selected and whether the user chose foreground or background printing.

If the user selected foreground printing, the Printing Manager forwards the QuickDraw commands directly to the appropriate printer driver, which translates the QuickDraw commands into PostScript information and contacts the desired printer to begin the information transfer process. The application, Printing Manager, and the printer driver remain engaged in the process until printing is complete, all of these events occurring in a locked series, somewhat like a bucket brigade.

If the user selected background printing, the Printing Manager creates a spool file that contains the application's QuickDraw commands and the user's Chooser selections and places the file in the PrintMonitor Documents folder. The application then disengages from the process after the spool file is created. The presence of a spool file in this folder will activate the PrintMonitor application, which then works with the printer driver to create the PostScript information. In Figure 7-19 and in the rest of this description, we'll assume that the user selected the LaserWriter printer driver. The choice of printer driver and the name of the desired printer, as well as whether the file is printed in foreground or background mode, is selected in the Chooser and stored in the System file.

Overview of the network process

After the appropriate PostScript instructions have been constructed, the LaserWriter Driver must accomplish the following tasks:

1. Locate the LaserWriter by name and discover the LaserWriter's Internet Socket Address
2. Establish a connection with the LaserWriter
3. Configure the LaserWriter (Laser Prep version, fonts, dictionaries, and so forth)
4. Describe the printing job (page characteristics, number of copies, paper tray, and so forth)
5. Send the PostScript instructions for the printed output
6. Close the connection with the LaserWriter

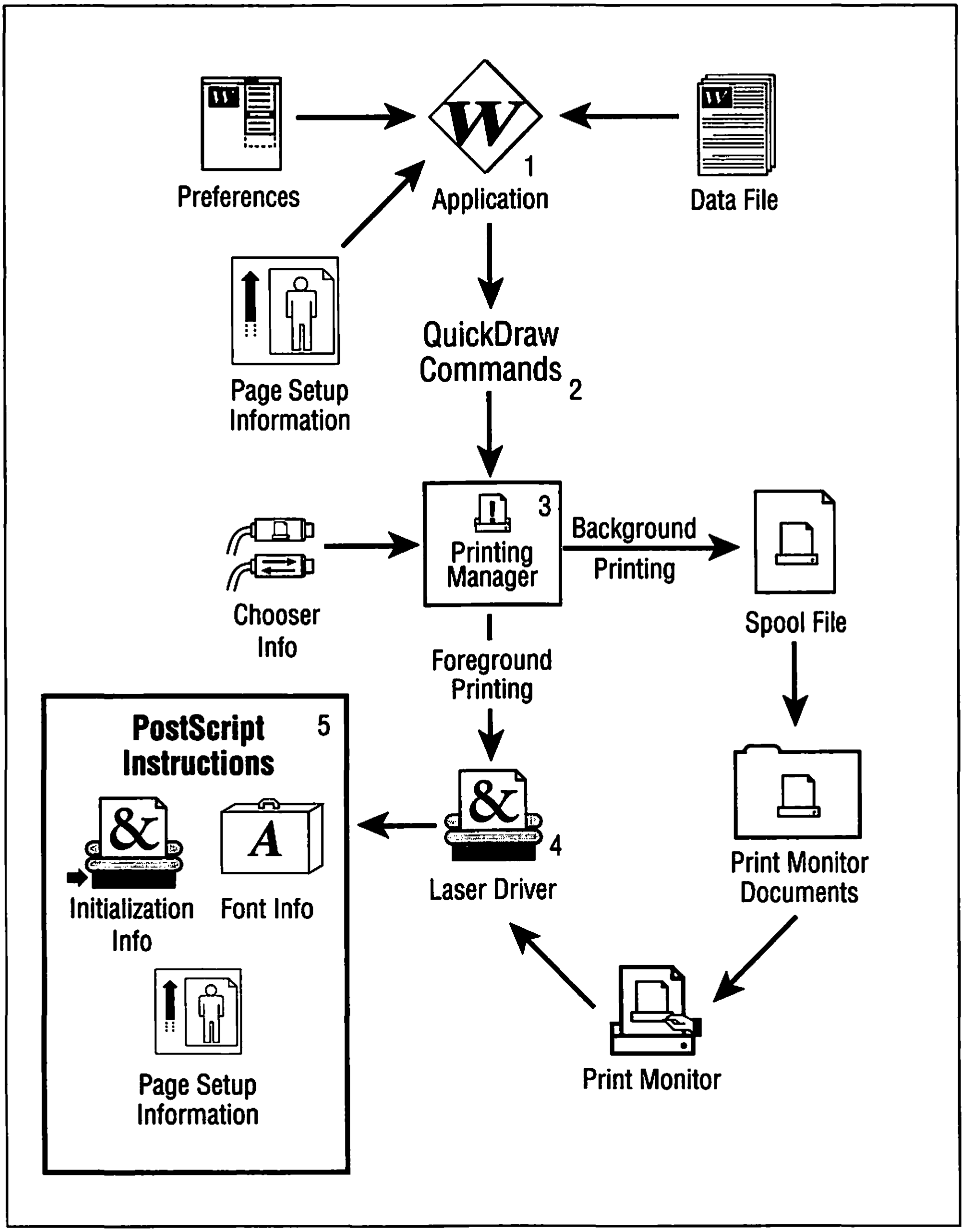


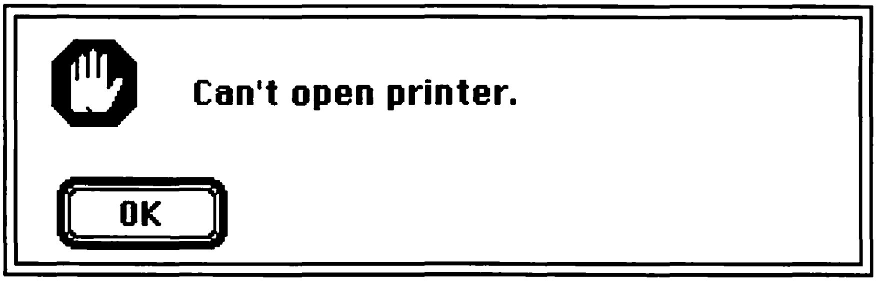
Figure 7-19. The Printing Manager accepts QuickDraw commands from the application and forwards them according to the user's Chooser selections.

Locating the LaserWriter by name

Because the LaserWriter Driver stores the service name of the chosen LaserWriter (name, type, and zone), it must first discover the LaserWriter's Internet Socket Address (network, node, and socket address) before it can start sending packets to the printer. In a network without a router, the Mac broadcasts a Name Binding Protocol (NBP) LookUp packet, querying for the LaserWriter's service name (for example, "Charlie:LaserWriter@Sales"). In a network that does contain a router, the Mac asks the router to locate the printer. The Mac sends an NBP Broadcast-Request packet to the router, and the router converts that Broadcast-Request into a series of LookUps; one LookUp is broadcast to every network that's part of the zone.

In either case, "Charlie" sends an NBP Reply packet directly back to the Mac that's initiating the search. The Reply packet contains the Internet Socket Address of the LaserWriter's Session Listening Socket (the socket used to make initial contact with the LaserWriter). At the time that the LaserWriter accepts the user's request to print, the LaserWriter requests that the Mac use a different socket to send the actual printer data. This leaves the Session Listening Socket free to answer NBP packets coming from other Macs that may try to print during the print session.

If "Charlie" doesn't answer after a certain length of time, the LaserWriter Driver displays the alert shown in Figure 7-20. In background printing, the Printing Manager would engage the Notification Manager to ask the user to bring PrintMonitor to the foreground so that PrintMonitor can display a similar message.



Can't open printer.

Figure 7-20. If Name Binding Protocol can't locate the user's chosen LaserWriter, the LaserWriter Driver displays this alert.

> Note: When the user uses the Chooser to select a printer, the Chooser could just as easily store the printer's Internet Socket Address as its service name. The service name is stored because AppleTalk uses dynamic

addressing. Every time the printer is restarted, it has a chance of having a different AppleTalk address. Since the name of the printer is assigned by the network administrator, it isn't subject to random changes. The name of the LaserWriter will change only if someone deliberately changes it or if more than one device is trying to use exactly the same name.

Establishing the LaserWriter connection

Once the Mac learns the identity of LaserWriter's Session Listening Socket, it attempts to establish a Printer Access Protocol (PAP) session with the LaserWriter to begin the printing process. To accomplish this, it sends a PAP Open Connection Request to the LaserWriter. The Open Connection Request includes the length of time that the Mac has been waiting to connect.

The LaserWriter responds to the Open Connection Request with an Open Connection Reply. If the LaserWriter is idle, the OpenConn Reply contains an error code of 0, which indicates that the Mac should begin the connection. If the LaserWriter is servicing another user, it returns an error code of 65,535 and a status string that gives the user information about why the LaserWriter is currently unavailable. This is the message that appears in the alert box on the user's screen. The status string includes such information as the name of the current user, the name of the document, and the name of the application; the string's exact format and contents are determined by the manufacturer. Other LaserWriter activities and conditions that might be reported are a paper jam, a missing paper tray, or that the LaserWriter is currently reinitializing.

The Open Connection Request is reissued by the LD every two seconds. When the LaserWriter finishes printing a job, it listens for approximately four seconds (twice the request interval) and accepts the user who reports the longest waiting time as the next customer.

Initializing the LaserWriter connection

At the time the session is initiated, the two devices inform each other of the socket ID where they want to receive information, the amount of memory they have to receive data, and the Connection ID of the session between them. The LaserWriter's name socket, or Session Listening Socket, isn't used for data transfer once the connection is established. The Session Listening Socket's only purposes are for the NBP process and the initial connection request (including the return of status strings).

Both ends of the PAP connection—the LaserWriter and the LD—then initialize their tickle timer. Tickle packets are sent through the connection when a device has no actual data to send but wants to assure its connection partner, "I'm still here." The receipt of the tickle packet verifies the integrity of the connection as well as the presence of the device sending the tickle packet. Tickle mechanisms are useful in printing, because if the LaserWriter fills up its memory with instruction data, it may take a while to process before it can ask for more. Tickling provides a way of letting the Mac know that the LaserWriter is still on the job and the network still works.

The tickle timer is one minute long—half the duration of the connection timer. When a device's tickle timer expires, the device sends (and expects to receive) a tickle packet to verify the connection. If the tickle isn't received, the device begins searching the network for its connection partner and continues the search until the connection timer expires.

In addition to the tickle timer, there's also a connection timer. The connection timer is two minutes long and is reset to 0 whenever data or tickles are received from the other end. When either the Mac's or the LaserWriter's connection timer reaches its two-minute limit, the device shuts down its half of the connection. The purpose of this mechanism is to allow the LaserWriter or the Mac to recover from a loss of its connection partner due to a crash, shutdown, or loss of the network connection. This allows the device to stay in service and recover the resources (memory, CPU time, and so forth) being consumed by the connection.

PAP is a session layer protocol, which means that its job is to establish and maintain the relationship (or "session") between two devices and manage the resources used by the devices during the connection. LaserWriters have a special characteristic for which PAP is especially designed: LaserWriters can receive data much faster than they can process it, even over a slow network like LocalTalk. Since the nature of printing is that the user's workstation initiates and drives the interaction, the session management protocol needs to have a mechanism that prevents the workstation from sending data to the LaserWriter when the LaserWriter can't receive it.

Sending data in a read-driven connection

PAP handles this need by having a communication structure that is read-driven. In a PAP connection, each device must signal the other when it's ready to receive by sending an invitation to send data. At the beginning of the session, after the tickle timer is initialized, each device issues this invitation.

When you first start looking at printer packets, this signaling process can be a little confusing. PAP information is carried inside ATP packets. The basis of ATP is a transaction that consists of a Request, a Response, and, in Exactly Once Transactions (XO), a Release that indicates that the transaction has been concluded. Normally in ATP, the Request asks a question or gives a command and the Response gives the answer or the status of the command. The Release is simply a notification from the requester to the responder that the transaction is complete.

In a PAP session, however, the Transaction Request carries the Send Data invitation. Questions and commands, as well as answers and command replies, are carried in the Response Packets. Each device has its own succession of transaction requests in which it gives the invitation to Send Data. The Response Packets it sends could either be questions or answers to questions.

After both sides have given the PAP Send Data invitation, the Mac asks the LaserWriter a question (see Figure 7-21). That uses up the Mac's invitation to send. The LaserWriter then sends the Release for the transaction, using up its invitation to send the answer. Both devices must then renew their "send" invitations to each other before any more data can be sent. This structure of invitations and data, requests and releases continues throughout the session.

Figures 7-21 and 7-22 illustrate the difference between PAP sessions and more typical ATP sessions. Figure 7-21 shows an ATP transaction model in which one end

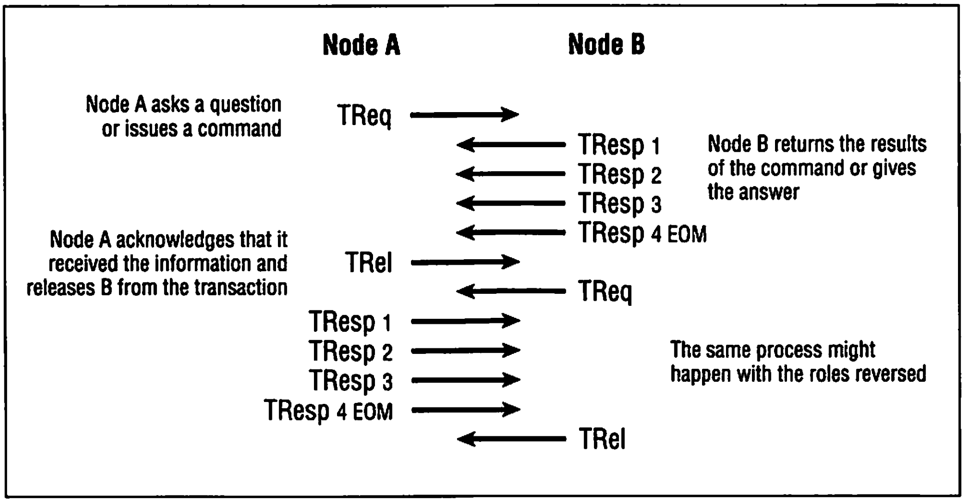

Node A
Node B
Node A asks a question or issues a command
TReq
TResp 1
TResp 2
TResp 3
TResp 4 EOM
Node B returns the results of the command or gives the answer
Node A acknowledges that it received the information and releases B from the transaction
TRel
TReq
TResp 1
TResp 2
TResp 3
TResp 4 EOM
The same process might happen with the roles reversed
TRel

Figure 7-21. This figure shows a typical client-server transaction model for AppleTalk Transaction Protocol (ATP). The two transactions shown above contain two question-answer pairs.

asks a question, receives a response from the other end, and then acknowledges receipt. (Either side can initiate the transaction.) Questions or commands are contained in the Transaction Request; results or answers are contained in the Transaction Response(s).

Because PAP is read-driven and each side must invite the other to send data, ATP works somewhat differently.

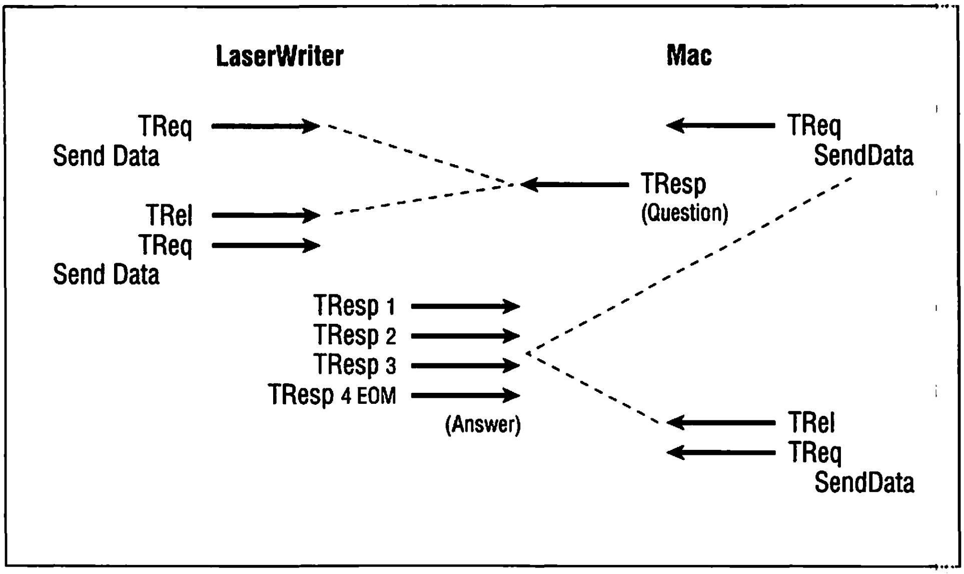


Figure 7-22. PAP sessions have two interleaved series of transactions, one set initiated by the Mac and the other by the LaserWriter. Each transaction request contains a PAP SendData command. Transaction Responses may contain either a question or an answer. The dotted lines link the three components of a transaction: Request, Response, and Release.

Overview of the print session

Most print sessions are conducted in three stages. In the first stage, the LaserWriter Driver asks the LaserWriter if it's using the same version of PostScript definitions that the Mac is using. In most networks, the answer to this question is yes because network managers usually prefer to have one standard version of printer driver loaded on all stations. We've described below what happens when the answer is something other than yes.

In the second stage, the Mac asks what fonts the LaserWriter has in its memory. The Mac compares the LaserWriter's font list to the list of fonts used in the document. If the document contains fonts not in the LaserWriter's memory, the LaserWriter Driver knows that it must provide the definitions for those fonts before the font is used.

In the third stage, the PostScript instructions for the printed output are sent to the LaserWriter. First, the LaserWriter is given the non-printing information that describes the job and the document and the user's Page Setup options, which include data such as paper size, margin information, and any PostScript definitions that the LaserWriter may need. Then the PostScript code for the printed output is sent. As soon as the LaserWriter Driver has successfully delivered all of the PostScript code, the Mac sends a PAP Close Connection Request and closes down the connection.

Querying the LaserWriter for laser prep version

The LaserWriter Driver asks the LaserWriter if it's running version *xx* of the Apple Dictionary, which is called "md." (The wording of the exchange is slightly different in System 7 because the Laser Prep file no longer exists.) The "md" dictionary is contained in the LaserWriter Driver file.

Mac's Question: *Are you running version xx of the Laser Prep dictionary?*

Query Format for LaserWriter 5.2 and 6.0.2:

```
Version 68 = Laser Prep 5.2, Version 70 = Laser Prep 6.0.1
%!PS-Adobe-2.0 Query
%%Title: Query for Laser Prep, the Apple PostScript Dictionary
%%?BeginProcSetQuery: "(AppleDict md)" 68 0
/md where{/md get /av get cvi 68 eq{(1)}{(2)}ifelse}{(0)}ifelse = flush
%%?EndProcSetQuery: unknown
```

Query Format for LaserWriter 7.0:
Version 71 is embedded in the System 7 LaserWriter Driver

```
%!PS-Adobe-2.0 Query
%%Title: Query for PatchPrep
%%?BeginProcSetQuery: "(AppleDict md)" 71 0
userdict/PV known{userdict begin PV 1 ge{(1)}{(2)}ifelse end}{/md
where{pop(2)}{(0-continuation- )}ifelse}ifelse = flush
%%?EndProcSetQuery: unknown
```

LaserWriter's Answer *(Three possible responses):*

Response	Meaning	LaserWriter Driver:
0	No dictionary is loaded	Downloads Laser Prep information Downloads Apple Dictionary
1	That version is loaded	System 6: Downloads Laser Prep instructions Downloads Apple Dictionary System 7: Downloads Apple Dictionary
2	Different version is loaded	Checks that the device is not a spooler and asks the user if he or she wants to re-initialize the printer

If the LaserWriter's response is 2, the Mac closes the connection and asks the user whether he or she wants to reinitialize the LaserWriter. If the user wants to reinitialize the LaserWriter, the LaserWriter Driver establishes a new connection and performs this function. When the reinitialization is complete, the Mac will again break the connection, establish a new connection, and print the document.

Querying the LaserWriter for font information

The LaserWriter Driver then finds out which fonts are loaded in the LaserWriter. It compares the LaserWriter's font list to the fonts required for the document so it will know which fonts to download. While dictionary definitions must be downloaded prior to the actual printing information, font definitions are often given just prior to their use in the body of the printing instructions.

The LaserWriter responds by sending the Mac a list of fonts that are in memory. On LaserWriters with an attached hard disk, the fonts on the hard disk are also reported.

As described earlier, this is the second stage of the normal print job. If you're trying to locate the beginning and end of these stages in the packet trace of a printing session, you can search for certain text patterns. At the beginning of each stage, you'll find the text string %!PS-Adobe-2.0 and at the end of each stage you'll find the text string %%EOF.

Sending the document instructions

The last stage of the print job is sending the PostScript instructions that define the printed output. In the packet trace, you can tell where this section begins because if all is going well, the transactions become very regular. Figure 7-23 shows some of the things to watch for in your troubleshooting.

Figure 7-23 shows a small portion of a typical trace from a print session that occurred within a single LocalTalk network. The large packets of 530 bytes contain the PostScript instructions, and the small packets of 18 bytes contain the control packets that initiate the transaction, issue the SendData invitation, and release the transaction. The LaserWriter Driver includes 512 bytes of PostScript in each of the large packets, plus the overhead needed to complete the packet (PAP and ATP Control information, the DDP Header, and the LAP frame information).

```
PKT.  FROM         TO           TYPE       SIZE    TIME
 46   LaserWriter  The Mac      ATP  TRel   18     0:00:13.419
 47   LaserWriter  The Mac      ATP  TReq   18     0:00:13.435
 48   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.470 |
 49   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.482 |
 50   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.494 |
 51   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.513 | ①
 52   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.525 |
 53   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.537 |
 54   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.549 |
 55   The Mac      LaserWriter  ATP  TRsp  530     0:00:13.561 |
 56   LaserWriter  The Mac      ATP  TRel   18     0:00:13.569 | ②
 57   LaserWriter  The Mac      ATP  TReq   18     0:00:14.012 |
 58   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.015
 59   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.025
 60   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.035
 61   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.045
 62   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.055
 63   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.065
 64   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.075
 65   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.085
 66   LaserWriter  The Mac      ATP  TRel   18     0:00:14.093
 67   LaserWriter  The Mac      ATP  TReq   18     0:00:14.654
 68   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.657
 69   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.668
 70   The Mac      LaserWriter  ATP  TRsp  530     0:00:14.677
```

Figure 7-23. In a print session, the transactions should flow smoothly and regularly in the third stage of sending the PostScript instructions.

If you capture a session on Ethernet or a print session on LocalTalk that's being printed through a router to a different network, the size of the packets will be slightly different, but the pattern will be the same: a long succession of transactions with the same size packets again and again. In this sample trace, we've filtered out all the other network traffic that was occurring at the time.

Losing packets due to network or wiring problems is always a concern, but in the packet trace of a print session, the missing packets are relatively easy to spot. At (1) in Figure 7-23, notice how the transaction response from the Mac to the LaserWriter includes eight packets. In this trace, which has no lost packets, the transactions are moving along in regular batches of eight responses. If packets were getting lost, you'd see retransmissions. A retransmission appears as a packet of 530 bytes sitting all by itself surrounded by smaller packets.

If you see fewer than the usual number of packets in the transaction, it may be that the packet was sent, but the protocol analyzer for some reason didn't capture it. The other possibility is that the sending station didn't elect to use all of the packets in the transaction. All of the packets in the transaction response are numbered; if the sender elects not to use packet #8, for example, it sends an End of Message notifier in packet #6. This isn't the end of the job; it simply means that the sender didn't want to use all of the allotted packets. You can quickly check the sequence to find out why packets are missing.

The LaserWriter decides how many packets (with a maximum of eight) to include in a single transaction, based on how much memory it has free when the session begins. It notifies the Mac of how many packets it can accept in one transaction when the connection is opened. This is called the *Flow Quantum*. Many LaserWriters use four packets for their Flow Quantum instead of eight, but the interactions are the same.

Within one transaction, the Mac is supplying the responses at a rate of approximately one packet per ten milliseconds. This is a reasonable rate for a Macintosh portable. A large LocalTalk packet like this takes a little more than two milliseconds to transmit.

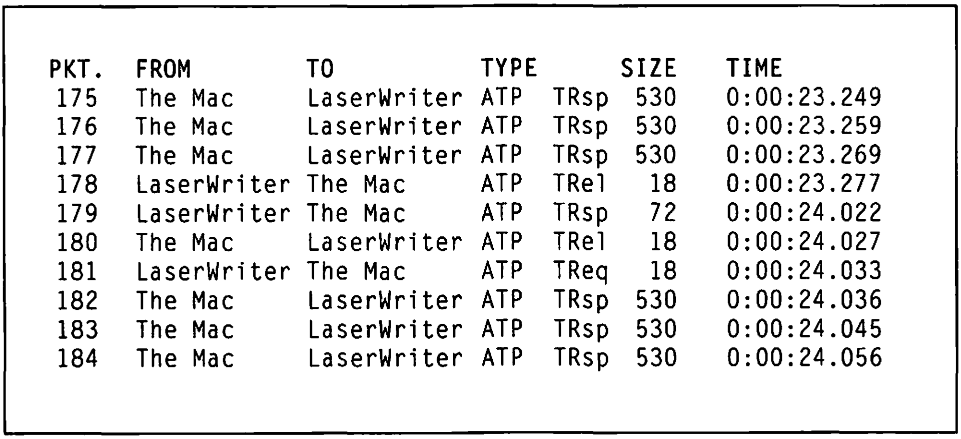
At (2) in Figure 7-23, the LaserWriter releases the Mac from the previous transaction, then issues the SendData command that allows the Mac to send the next batch of eight packets. At this point, look at the time period between the Release and the next Request. If there's a gap, it's because the LaserWriter is busy processing the information it already has and its buffer is full.

Also, look at the time between the SendData command (in the transaction Request) and the first packet in the following transaction Response. In Figure 7-23, both sides are working at high speed; as soon as a transaction is completed, the LaserWriter issues the next SendData command. In addition, as soon as the SendData command is issued, the Mac supplies the LaserWriter with more information.

If your print job is slow, looking at these time gaps can help you determine which end—the Mac or the LaserWriter—is causing the slowness. If it's the LaserWriter, perhaps you're sending particularly complex graphics. Images can be rendered in different ways, and some are less efficient for the LaserWriter to draw. You might also try sending the image from a similar application made by a different manufacturer or restructuring the graphic to be less complex.

You might also try reducing the number of fonts involved, or using PostScript fonts instead of TrueType fonts. Other things to check are the Page Setup and Print Options settings. The Unlimited Downloadable fonts option can also slow down your printing considerably because each time you change fonts a new font definition is sent to the LaserWriter.

The Mac's response time depends on such system variables as processor type and speed, the number of other applications running, and the activities of the user during the print session. You might also see time gaps between the packets within a single transaction. In background printing, the Mac is doing something else in the foreground, so less processing power is available for the printing process. The gap in Figure 7-23, which is only three milliseconds, represents the speed of foreground printing. In background printing, the gap will be somewhat larger.



```
PKT.  FROM         TO           TYPE       SIZE    TIME
 175  The Mac      LaserWriter  ATP  TRsp   530    0:00:23.249
 176  The Mac      LaserWriter  ATP  TRsp   530    0:00:23.259
 177  The Mac      LaserWriter  ATP  TRsp   530    0:00:23.269
 178  LaserWriter  The Mac      ATP  TRel    18    0:00:23.277
 179  LaserWriter  The Mac      ATP  TRsp    72    0:00:24.022
 180  The Mac      LaserWriter  ATP  TRel    18    0:00:24.027
 181  LaserWriter  The Mac      ATP  TReq    18    0:00:24.033
 182  The Mac      LaserWriter  ATP  TRsp   530    0:00:24.036
 183  The Mac      LaserWriter  ATP  TRsp   530    0:00:24.045
 184  The Mac      LaserWriter  ATP  TRsp   530    0:00:24.056
```

Figure 7-24. This figure shows a portion of the trace later in the print session.

In Figure 7-24, the flow of the transactions is interrupted by a status packet—the 72-byte packet between the transactions. It's normal for the LaserWriter to periodically update the Mac with a status report. (These status reports are the messages you see on your screen: "Preparing data," "9 pages to print," and so forth.)

Notice that the Mac didn't specifically request this status report; the LaserWriter spontaneously sent it. Had the Mac requested a status report, there would be a transaction Request from the Mac to the LaserWriter preceding the 72-byte packet containing the PAP SendStatus command. In other words, the Mac simultaneously has two outstanding transactions with the LaserWriter: a SendData invitation and a SendStatus Request. We mentioned earlier that filtering out Requests and Responses would make the trace easier to follow. The downside of that simplicity gain is that because the control packets are not visible, you wouldn't be able to distinguish between status messages sent spontaneously and status messages sent due to a SendStatus Request.

In this case, the LaserWriter sent a spontaneous status message, using an outstanding SendData Request issued many packets earlier. A Mac will ask for a status report if it's unsure of what's happening at the LaserWriter. A typical example is a case in which the LaserWriter is busy and hasn't issued an immediate SendData after releasing the previous transaction. After several seconds, the Mac may ask the LaserWriter, "What's going on?" with a SendStatus command.

The end of the job is easy to spot because the last transaction contains a small packet (less than 530 bytes) to conclude the instructions. The Mac will immediately close the connection, even though the print job hasn't finished processing in the LaserWriter. If there's an unrecoverable error at the end of the print job and the LaserWriter discovers it after the connection is closed, the Mac won't be notified of the error. Many LaserWriters have the ability to terminate a job on their own in such a case; the only way that you'll know that there was an error is that some of your output won't be in the tray.

File Servers—AppleShare and FileShare

The Macintosh has a rich and full-featured filing system, with many unique characteristics. Whereas in most other filing systems files on disk typically consist of a simple stream of bytes, files on the Macintosh can contain structured *resources* such as windows, menus, icons, and dialog boxes. Furthermore, the Macintosh file system stores a great deal of information about the file itself, including creation, modification, and backup dates; signature information used to link files to icons and documents to their applications; and information used by the Finder to display the file on the desktop.

File systems and file servers

Prior to the introduction of the Macintosh Plus, the Mac used a *flat file system* known as MFS (Macintosh File System), shown in Figure 7-25, in which all files were effectively in their disk's root directory. The Finder allowed the user to create folders and place files in them, but these folders were strictly ornamental. The Open File dialog under MFS showed all the files on the specified disk, regardless of which folder they had been placed in. This scheme worked fairly well for floppy disks, which were capable of holding only a limited amount of data.

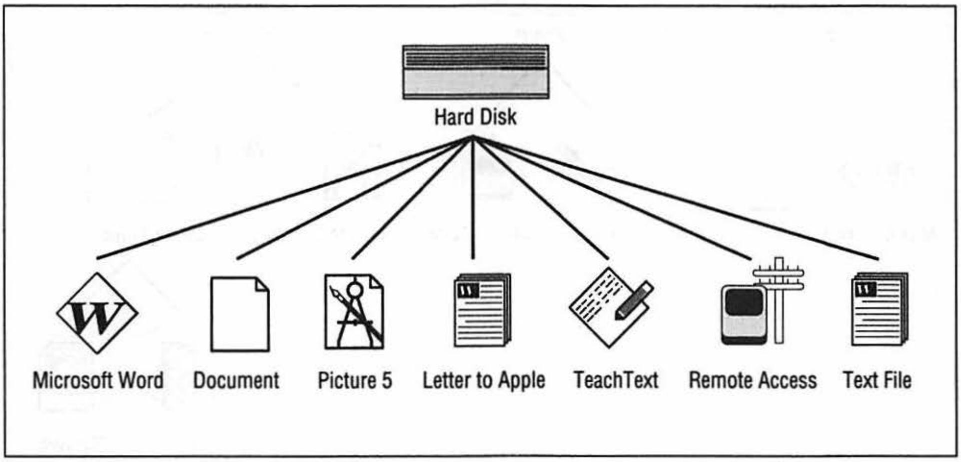


Figure 7-25. A flat file system keeps all its files in the root directory of the disk.

With the advent of larger Macintosh hard disks, MFS soon became unmanageable. When a user stored thousands of files on a single disk, scrolling through the entire list to select a single file became difficult and annoying. In addition, each file had to have a unique name, because files were not actually segregated into directories. To address these issues, Apple introduced a new file system called HFS (Hierarchical Filing System), shown in Figure 7-26.

As its name suggests, HFS is *hierarchical*, meaning that information on a disk can be represented in the form of a tree. At the base of the tree is the *root directory*, which contains the files and folders that you see when you initially click the disk's icon. The folders in the root can themselves contain other files and folders, with as many levels as the user desires. File names need only to be unique within a given folder, and the Open File and Save File dialogs accurately display the structure of the disk, showing only the folders and files within a particular directory at any given moment.

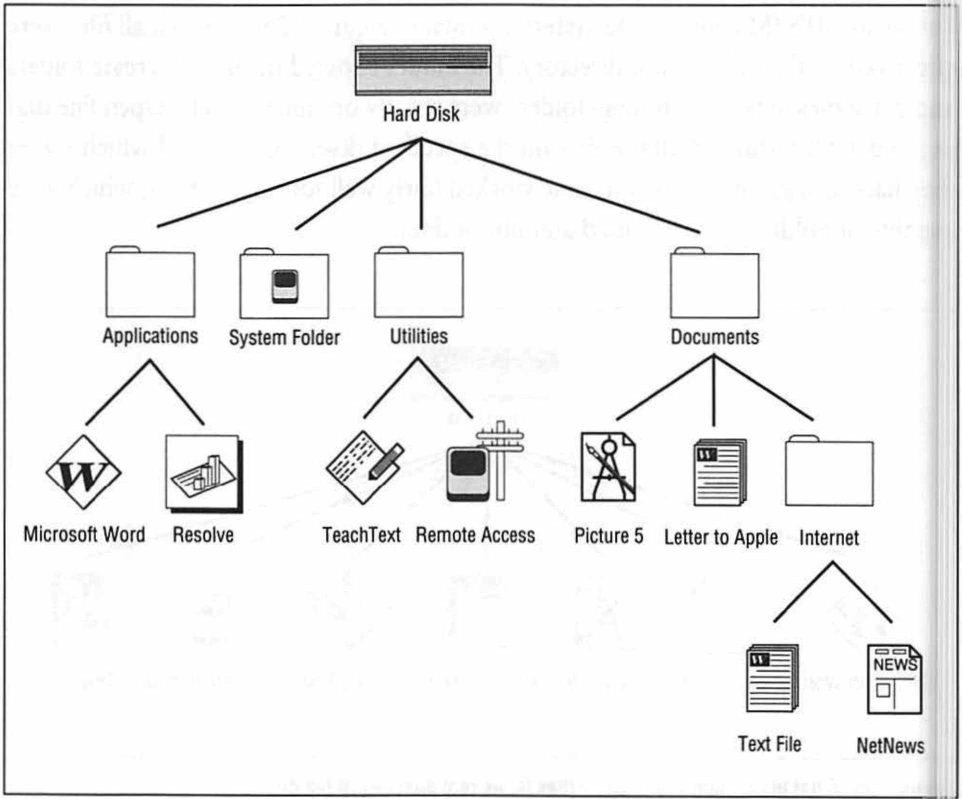


Figure 7-26. A hierarchical file system allows its users to segregate files into folders, allowing greater organization and ease of access.

HFS is supported within the Mac's operating system by a large set of basic operations, comprising well over 100 unique functions. Specific functions exist to perform the following tasks:

- Create new files and new folders
- Open existing files
- Delete files or folders

- Rename files or folders
- Enumerate the contents of a folder
- Get or set information about files or folders
- Read and write data to files

The operating system calls that support the Macintosh file system are exhaustively documented in *Inside Macintosh*. To understand how file servers operate, you must have a good understanding of how the underlying file system operates locally, and we strongly recommend that you study the relevant chapters if you need to troubleshoot server operations.

Several factors led to the development of modern file servers. Large hard disks became cheaper, more reliable, and more widely available. This provided managers with an opportunity to realize economies of scale by having their users share a single large disk rather than buying each individual workstation its own smaller disk. The necessity to share data among users became more prevalent as well. To address these requirements, in 1988 Apple introduced the AppleShare File Server and the network protocol that supports its operation.

Data-sharing among many users requires additional capabilities to those offered in HFS. The need to authenticate users, limit access to files by particular users or groups of users, and maintain data integrity when a single file is accessed by multiple users have no equivalent operations within the set of operations defined for HFS, and they are not meaningful in a single-user context. The Apple File Protocol (AFP) addresses these needs and others as well.

The operations available in AFP must also satisfy another set of constraints. While AFP was originally developed to be used in the context of a file server running on a Macintosh and accessed by Macintoshes, there are many third-party implementations that turn other types of systems into AppleShare-compliant servers. PacerShare performs this function on Digital Equipment VAX systems, and Novell markets a module that allows Macintoshes to access a NetWare server through AFP. In addition, there is a public domain implementation of AFP, originally developed at Columbia University, for UNIX systems. Conversely, Farallon offers products that allow MS-DOS PCs to access AFP servers.

AFP was structured to allow easy application of its functionality to servers using other file systems than HFS; it was also designed to allow non-Macintosh computers to use functions that are meaningful in the context of their native file systems.

AFP also contains commands that allow the server to optimize certain operations that would be very costly to do over the network. For example, the FPCopyFile operation allows the server to duplicate a file without needing to transfer the data down to the client and then back to the server again in a new location. Similarly, the FPMoveAndRename call allows the server to place a file in a new location, optionally renaming it as well, without further client intervention.

AFP is one of the most utilized file server protocols currently available. Its popularity has been dramatically increased by Apple's introduction of FileShare with the release of System 7. FileShare allows *any* Macintosh to operate as a file server. The only distinction between FileShare and AppleShare is that AppleShare supports a larger number of concurrent users and has somewhat higher performance. Both implementations use AFP in the same way.

The process in the Macintosh

Accessing a network server volume from a client Macintosh takes advantage of a programmatic "hook" built into the Macintosh file system. Basically, the Mac's native operating system processes application calls to the file system to the point of recognizing whether the volume being accessed is locally mounted on the Macintosh or remotely over the network.

If it determines that the volume isn't local, the file manager hands off further processing of the call to a piece of code called an *external file system.* It is the responsibility of this piece of code to translate the native HFS request into whatever form is required to interact, in an equivalent way, with the actual owner of the volume. In the case of AFP servers, this amounts to converting the HFS request to one or more AFP commands, transmitting them to the server, collecting the server's responses, and communicating them back to the Mac's file system.

In fact, external file systems are used in a number of other ways on the Macintosh. Other types of network file servers, such as NFS servers, are implemented as external file systems. CD-ROM disks are accessed by the Macintosh through the same mechanism.

The process on the network

In our earlier examination of the Presentation Layer, we discussed, in a fairly cursory way, how file system operations on a Macintosh are translated into AFP commands. We can now look into these functions in more detail.

In general, using a server proceeds as follows:

1. The client locates the server by name and gets the internet address of the server's Listening Socket.
2. The client opens a session to the server using the AppleTalk Session Protocol.
3. The client uses AFP to log in to the server. Authentication information such as a user name and a password are exchanged at this time.
4. The client queries the server for a list of available volumes.
5. The client selects one of the volumes and requests that the server open it for use.
6. The client sends AFP requests to the server to perform various file system operations. The server, in response, sends information that allows the client to represent the results of those operations using the normal mechanisms of the Macintosh.
7. The client logs out of the server.
8. The session connection is closed.

Before they can be used, AFP servers must be located. This is accomplished by precisely the same operation of locating a LaserWriter. The user opens the Chooser, and rather than selecting LaserWriter, selects AppleShare. In response, the Chooser sends out NBP LookUp frame to search for =:AFPServer@*. All nodes intended to offer AFP-based services registered an object of type AFPServer when they were initialized, and respond with the object's name and the network address of the server's Listening Socket. The Chooser collects the responses and displays a list of the names of the responding devices. The user selects one and clicks the OK button, initiating access to the server.

Sharing files between a Macintosh client and an AFP server requires a dedicated connection between the two. To accomplish this, the AppleTalk Session Protocol (ASP) is used. ASP, which operates at the Session Layer, is a client of the AppleTalk Transactions Protocol and also manages an ongoing, dedicated exchange of messages between two nodes.

The only request an unconnected client can make of a server is to query the server's status. In the case of an AFP server, this returns a frame containing information specific to the kind of service that is using ASP. This exchange, between a client Macintosh and an AFP server, is illustrated in the following two packets.

Client Request

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 1 TReq
 Control Information: %000 ALO
 TRel Timeout Indicator: %000 30 seconds
 Bitmap: %00000001 Need Packet(s) 0
 Transaction ID: 2685
 User Bytes: 0x03000000
ASP Header - AppleTalk Session Protocol
 Function Code: 3 GetStatus
```

Server Reply

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 2 TResp
 Control Information: %010 ALO EOM
 TRel Timeout Indicator: %000 30 seconds
 Sequence Number: 0 Here is Packet 0
 Transaction ID: 2685
 User Bytes: 0x00000000
 ATP Data:
 -+-5-c-ù---Padma 00 2b 00 35 00 63 00 9d 00 03 0d 50 61 64 6d 61
 sambhava........ 73 61 6d 62 68 61 76 61 20 20 20 20 20 20 20 20
 ...........-Maci 20 20 20 20 20 20 20 20 20 20 20 09 4d 61 63 69
 ntosh--AFPVersio 6e 74 6f 73 68 03 0e 41 46 50 56 65 72 73 69 6f
 n 1.1-AFPVersion 6e 20 31 2e 31 0e 41 46 50 56 65 72 73 69 6f 6e
  2.0-AFPVersion  20 32 2e 30 0e 41 46 50 56 65 72 73 69 6f 6e 20
 2.1--Cleartxt pa 32 2e 31 03 10 43 6c 65 61 72 74 78 74 20 70 61
 sswrd-Randnum ex 73 73 77 72 64 10 52 61 6e 64 6e 75 6d 20 65 78
 change-2-Way Ran 63 68 61 6e 67 65 16 32 2d 57 61 79 20 52 61 6e
 dnum exchange--- 64 6e 75 6d 20 65 78 63 68 61 6e 67 65 00 00 00
 -----------ü---P 00 00 00 00 00 00 01 00 00 00 02 9f e0 00 04 50
 0--0(---<-†----Ñ 30 00 08 30 28 00 10 10 3c 07 a0 08 04 18 7f 84
 ---Ç---Å---Ç---Ñ 04 10 00 82 04 10 00 81 04 10 00 82 04 10 00 84
 ---à---ê---∞---- 04 10 00 88 04 10 00 90 04 10 00 b0 04 10 00 d0
 -----@---?------ 04 ff ff ff ff 40 00 00 02 3f ff ff fc 00 00 07
```

```
---------------- 00 00 00 05 00 00 00 05 00 00 00 05 00 00 00 0f
Ä---Ä---Ä---Ä--- 80 00 00 08 80 00 00 08 80 00 00 0f 80 00 00 0a
Äø--t----ø------ 80 bf ff f2 74 00 00 05 00 bf ff f8 f4 00 00 00
-----------ü---- 00 00 00 00 00 00 01 00 00 00 03 9f e0 00 07 df
----------ø----- f0 00 0f ff f8 00 1f ff fc 07 bf ff fc 1f ff ff
---------------- fc 1f ff ff fc 1f ff ff fc 1f ff ff fc 1f ff ff
---------------- fc 1f ff ff fc 1f ff ff fc 1f ff ff fc 1f ff ff
---------?------ fc ff ff ff ff 7f ff ff fe 3f ff ff fc 00 00 07
---------------- 00 00 00 07 00 00 00 07 00 00 00 07 00 00 00 0f
Ä---Ä---Ä---Ä--- 80 00 00 0f 80 00 00 0f 80 00 00 0f 80 00 00 0f
Äø---ø---ø--- 80 bf ff ff f4 bf ff fd f4 bf ff f8 f4
```

Although the contents of the AFP server's response to the GetStatus query have not been documented by Apple, some things are recognizable. At the beginning of the response, you can see the name of the server, Padmasambhava; the versions of AFP it supports, 1.1, 2.0, and 2.1; and the methods of authentication it can use. These pieces of information are used by the client later, in the actual login to the file server.

The client then establishes a session with the server by sending an ASP OpenSession request frame. This packet specifies the socket number that the client wants to use for the session, as well as the version of the ASP protocol it supports. The server responds with an OpenSessionReply frame containing a server socket number to be used for subsequent interactions and a session ID used to identify further packets belonging to this particular session.

Client Request

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 1 TReq
 Control Information: %100 XO
 TRel Timeout Indicator: %000 30 seconds
 Bitmap: %00000001 Need Packet(s) 0
 Transaction ID: 2686
ASP Header - AppleTalk Session Protocol
 Function code 4 OpenSession
 Session socket: 252
 ASP Version: 0x0100 ASP Version 1.0
```

Server Reply

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 2 TResp
 Control Information: %000 ALO
 TRel Timeout Indicator: %000 30 seconds
 Sequence Number: 0 Here is Packet 0
 Transaction ID: 2686
ASP Header - AppleTalk Session Protocol
 Session Socket: 248
 Session ID: 5
 Error Code: 0
```

With the connection established, the client and the server can exchange several types of frames. Most of these are ASP Command and CommandReply frames; it is these that the AFP uses to send and receive file operations. An exception to this is the case of a client writing data (for example, copying a file) to a server. Because ASP is an asymmetrical protocol, making a distinction between a client and a server, this operation is handled differently. (The operation is described later in this chapter.) Client and server also exchange ASP Tickle frames periodically, to ensure that the session connection is still active. If either side does not receive a Tickle frame within a certain amount of time, it closes the connection. Finally, either side can close the connection by sending an ASP CloseSession packet.

AFP is vastly more complex than PAP. You can get a good sense of what is happening over a PAP connection by simply examining a listing of the packet types and sizes, but this does not suffice with AFP. The actual operation of AFP is best understood by closely examining interactions between a client and a server.

AFP, in total, contains over 45 individual commands. Examining each one in detail takes a tremendous amount of space and might not give a good sense of its actual use. A particular HFS call can translate into a sequence of several AFP calls, and as you will see, a user operation such as dragging a file to the trash and emptying the trash can require many HFS calls to implement.

The following section examines in detail several common user operations. To really understand AFP however, you must observe it in operation. This is most easily accomplished through the use of a network monitor such as EtherPeek or a network analyzer such as a Sniffer.

Logging in to a server

The login process actually begins when the user sends an FPLogin command to the server. This frame contains the version of AFP and the authentication method the client wants to use, the user name with which to log in, and possibly a password. The password is sent in the FPLogin frame if Cleartxt Passwrd has been selected as the authentication method. This method is not widely used, however, as it is inherently insecure. Anyone with access to a network monitor can collect passwords with no trouble whatsoever!

A more typical authentication method is random number exchange, which never allows the actual password to be sent over the network. Instead, the user provides only a username in the FPLogin command. The server responds with a reply frame that contains an error code indicating that the client must provide further information before the login can be completed. The reply frame also contains an 8-byte random number. The client uses the password supplied by the user as an encryption key, encodes the random number, and transmits it back to the server in an FPLoginCont frame. The server then encrypts the same random number with what it believes is the user's password, and compares the two results. If they match, the server allows the login. The method used by AFP for encrypting the random number is called the Digital Encryption Standard and is generally considered unbreakable by anyone possessing fewer resources than the National Security Agency.

The next step in the login process is to query the server to determine what volumes it has available. The client sends the server an FPGetSrvrParms frame and the server replies with the time indicated by its system clock, the number of available volumes, and the names of those volumes.

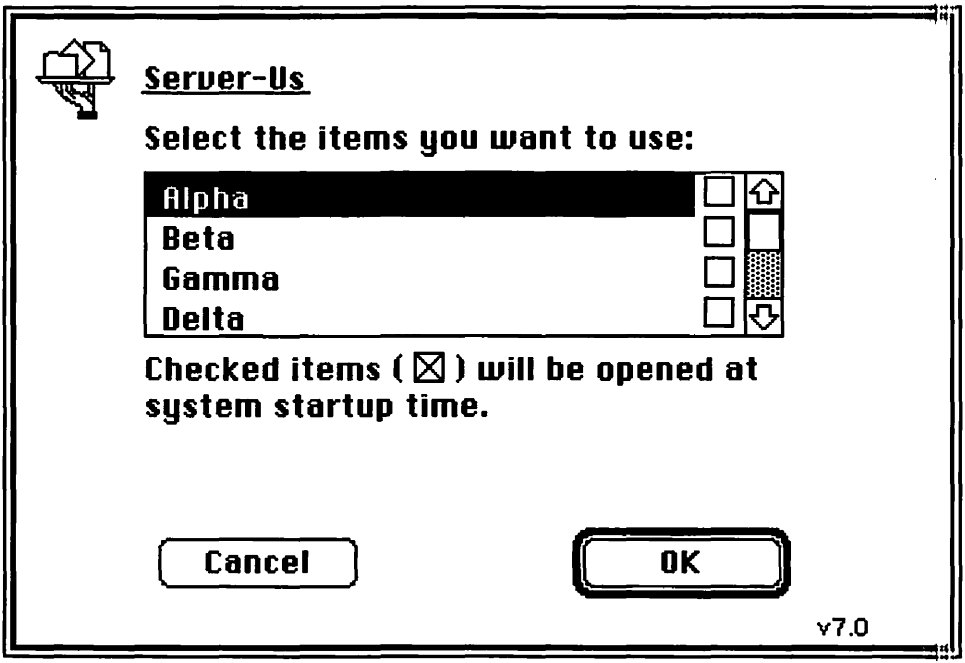
Client Request

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 1 TReq
 Control Information: %100 XO
 TRel Timeout Indicator: %000 30 seconds
 Bitmap: %00000111 Need Packet(s) 0 1 2
 Transaction ID: 2690
ASP Header - AppleTalk Session Protocol
 Function Code: 2 (Command)
 Session ID: 5
 Sequence Number: 2
AFP - AppleTalk Filing Protocol
 Function: 0x10 FPGetSrvrParms
```

Server Reply

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 2 TResp
 Control Information: %010 ALO EOM
 TRel Timeout Indicator: %000 30 seconds
 Sequence Number: 0 Here is Packet 0
 Transaction ID: 2690
 User Bytes: 0x00000000
AFP FPGetSrvrParms Response
 Server Time: 0xf2eebe76
 NumVols: 4
 Volume #1: Alpha (HasConfigInfo)
 Volume #2: Beta
 Volume #3: Gamme
 Volume #4 Delta
```

The client can then display a list of the volumes to the user, who can select one based on the server's reply to an FPGetSrvrParms request (see Figure 7-27).



Server-Us

Select the items you want to use:

Alpha	□
Beta	□
Gamma	□
Delta	□

Checked items (☒) will be opened at system startup time.

Figure 7-27. The client displays this dialog to allow the user to select a particular volume.

Mounting a server volume

Once the user has selected a particular volume, the client must request that the server grant access to it. The client does this by sending an FPOpenVol request to the server. This call supplies the name of the specified volume, flags indicating which of several pieces of information about the volume should be returned, and, optionally, an additional password to gain access to the volume. In response, the server returns a copy of the flags and the requested information.

Client Request

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 1 TReq
 Control Information: %100 XO
 TRel Timeout Indicator: %000 30 seconds
 Bitmap: %00000001 Need Packet(s) 0
 Transaction ID: 2691
ASP Header - AppleTalk Session Protocol
 Function Code: 2 (Command)
 Session ID: 5
 Sequence Number: 3
AFP - AppleTalk Filing Protocol
 Function: 0x18 FPOpenVol
 Flags: 0x0020 (Volume ID)
 Volume Name: Alpha
```

Server Reply

```
PATP Header - AppleTalk Transaction Protocol
 Function Code: 2 TResp
 Control Information: %010 ALO EOM
 TRel Timeout Indicator: %000 30 seconds
 Sequence Number: 0 Here is Packet 0
 Transaction ID: 2691
 User Bytes: 0x00000000
AFP FPOpenVol Response
 Flags: 0x0020 (Volume ID)
 Volume ID: 0xffff
```

From the server's point of view, this is all that is required to access the volume. The volume is then inserted into the list of disks available for access by the client. The

Finder on the Macintosh client, however, needs much more information before it can display the volume to the user. At this point, a flurry of requests and responses passes between the client and the server. Some of the operations performed include opening the Desktop database that contains icons, file comments, and server location of the applications that can access particular documents. Another operation is requesting information about the volume itself, such as size, free space, and creation and modification dates. Next, the client enumerates the contents of the volume's root directory, determining the type of each file, the name of the corresponding application, the size of the file, its Finder-specific information, its icon, and, in the case of folders, a directory ID that can be used to refer to that folder in subsequent calls.

Only when all this information has been collected, which can require the exchange of several hundred commands between client and server, does the Finder open a disk window and display the contents of the volume's root.

Listing the contents of a server directory

A common operation between the client and the file server is obtaining a listing of all the files and folders within a given directory. This must be done in order to display a directory in the Finder, to allow an application to open or save a file, to calculate the size of a folder, and in many other contexts. AFP uses the FPEnumerate command to perform this function. The client transmits the command to the server, along with the volume and directory identifiers, two sets of flags indicating what information should be returned for files and for directories, the maximum number of entries to be returned, the maximum number of bytes of information to be returned, a directory path to use (relative to the supplied directory ID), and an index. The index tells the server which item in the specified directory should be returned first.

Since directories can contain large numbers of entries, it's possible—even likely—that a full listing exceeds the maximum number of bytes that ASP can return in a single command response. To continue the listing, the client makes another FPEnumerate call, increasing the index to one greater than the index of the last entry returned. For example, if the first FPEnumerate is issued with an index of one and returns ten entries, a second FPEnumerate must be issued with an index of eleven to retrieve additional entries.

An extract of an FPEnumerate command and response is displayed in the following packet. Although the full response contained thirteen entries, only three are displayed here. This command was followed by a second FPEnumerate, using an index of fourteen to get information about additional files and folders. Further FPEnumerate commands are issued until the server returns an error code indicating that there is no additional information to be returned.

Client Request

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 1 TReq
 Control Information: %100 XO
 TRel Timeout Indicator: %000 30 seconds
 Bitmap: %00000011 Need Packet(s) 0 1
 Transaction ID: 2716
ASP Header - AppleTalk Session Protocol
 Function Code: 2 (Command)
 Session ID: 5
 Sequence Number: 0x1c
AFP - AppleTalk Filing Protocol
 Function: 0x09 FPEnummerate
 Volume ID: 0xffff
 Directory ID: 0x00000002
 File Flags: 0x077f
 Directory Flags: 0x137f
 Max Entries Requested: 64
 Start Index: 1
 Max Reply Size: 1152
 Path Type: 2 (Long Path Name)
 Path Name: NIL
```

Server Reply

```
ATP Header - AppleTalk Transaction Protocol
 Function Code: 2 TResp
 Control Information: %000 ALO
 TRel Timeout Indicator: %000 30 seconds
 Sequence Number: 0 Here is Packet 0
 Transaction ID: 2716
 User Bytes: 0x00000000
AFP FPEnumerate Response
 File Bitmap: 0x077f
 Directory Bitmap: 0x137f
 Actual Count: 13
Entry #1 (Directory)
```

Server Reply *(continued)*

```
 Attributes: BackupNeeded, Invisible
 Parent Directory ID: 0x00000002
 Creation Date: 0xf2ee9108
 Modification Date: 0xf2ee9108
 Backup Date: 0x80000000
 Finder Info: 00000000000000004000000000000000
 00000000000000000000000000000000
 Long Name: Move&Rename
 Directory ID: 0x00009990
 Offspring Count: 0
 Access Rights:
 User: Read, Write, Search
 World: None
 Group: None
 Owner: None
Entry #2 (File)
 Attributes: BackupNeeded, ResForkOpen, DataForkOpen, SysFile,
Invisible
 Parent Directory ID: 0x00000002
 Creation Date: 0xf1bd0370
 Modification Date: 0xf2ee9280
 Backup Date: 0x80000000
 Finder Info: 4254464c706473207000000000000000
 00000000000000000000000000000000
 Long Name: AppleShare PDS
 File Number: 0x00002567
 Data Fork Length: 106496
 Resource Fork Length: 836
Entry #3 (Directory)
 Attributes: BackupNeeded
 Parent Directory ID: 0x00000002
 Creation Date: 0xf19a87f4
 Modification Date: 0xf29a48cf
 Backup Date: 0x80000000
 Finder Info: 013d01050229028d0100003400000200
 fff8fff00000000200008000000000002
 Long Name: Applications
 Directory ID: 0x00000081
 Offspring Count: 48
```

```
Access Rights:
User: Owner, Read, Write, Search
World: None
Group: None
Owner: Read, Write, Search
```

As mentioned, the FPEnumerate call does not by itself return all the information the Finder needs to display a disk or folder window. What additional information the Finder requires depends on the file view (by name, size, kind, or date). Icon and small icon views need only the appropriate icon for each file's signature. This information is retrieved from the Desktop database by issuing an FPGetIcon call for the corresponding view. In the case of document files, the various views need the corresponding application's name. This is also obtained from the Desktop database through the FPGetAPPL call. File comments are also stored in the Desktop database and retrieved by the FPGetComment call.

Copying a file to the server

An operation that seems seamless to someone using the Finder can easily translate into the exchange of several hundred discrete commands between the client and the server. Copying a file to the server, a simple matter of clicking an icon, dragging it to the desired location, and releasing it, is a complex affair when examined over the network.

The exchange typically begins with an FPEnumerate call to the server for the target folder. If this reveals an existing file of the same name, the familiar dialog asking whether the user wants to replace the file is shown. If the user replies Replace, or if there's no existing file with the designated name, the client issues an FPCreateFile command. This creates an empty file of the specified name in the target directory on the server. Next, an FPSetFileDirParms call is sent to the server to initialize the file's Finder information, owner and group IDs, and access rights.

The client then opens the file's data fork using an FPOpenFork command and issues the FPWrite call to transmit the data to the server.

As mentioned earlier, the sequence of events when writing a file from the client to the server is unusual compared to the fairly simple request/response structure of other AFP interactions. There are several reasons for this.

ASP is an inherently asymmetrical protocol since it makes a distinction between client and server. With the single exception of the ASPCloseSession call, all interactions between client and server are initiated by the client. ASP is a client protocol of ATP, which means that at the ATP level, the client is issuing TReq frames and the server is answering them with TResp frames.

ATP, in turn, is structured so that relatively small amounts of data, up to 578 bytes, can be passed with a TReq. Eight times as much information, 4624 bytes, can be returned in a TResp, however. ASP takes advantage of this feature to optimize the performance of writing to a server by structuring the operations as a pair of interleaved commands.

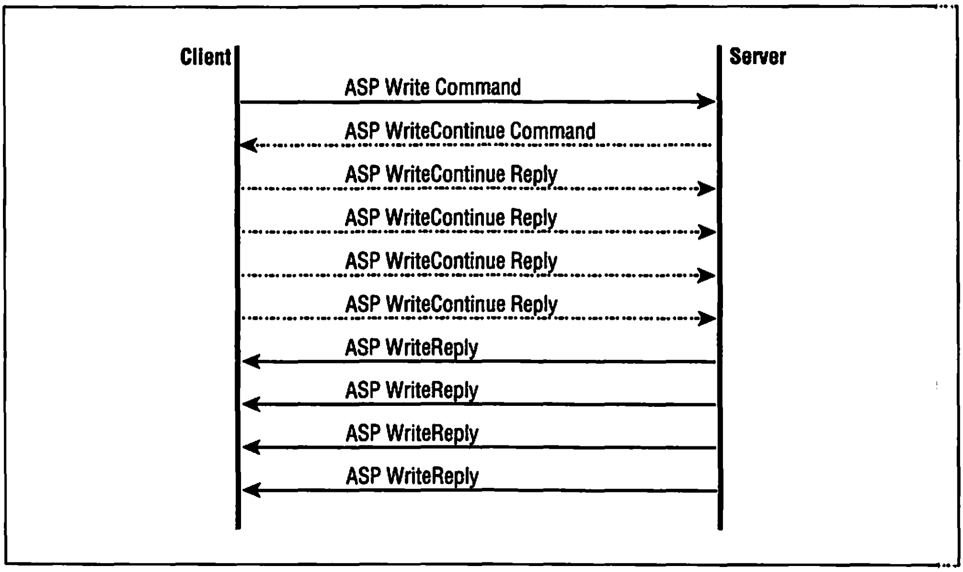


Figure 7-28. Interaction of the SPWrite command and the SPWriteContinue command.

The client issues an SPWrite call containing a command block to be used by the higher-level client protocol of ASP (see Figure 7-28). The server, without replying to the SPWrite, sends the client an SPWriteContinue call, with the number of bytes of available buffer space. The client replies to the SPWriteContinue call with one to eight TResp frames containing the data to be written to the server. Finally, the server replies to the original SPWrite with up to eight response frames of its own. Figure 7-28 shows the interaction between the SPWrite command, which initiates the operation, and the SPWriteContinue command, in which the data to be written is actually transmitted.

When AFP performs this operation, a third operation (this time an actual AFP call) is "wrapped around" the two ASP operations that actually do the write. The client transmits an FPWrite command containing a reference number for the open file fork that is being written, a requested count of bytes to be written, an offset at where the data should be written, and a flag indicating whether that offset should be measured from the beginning or the end of the file. The SPWrite and SPWriteContinue exchange takes place, and the server finally replies to the FPWrite with a response containing the number of the last byte of the fork to which data was written.

Once one fork of the file has been written, it is flushed to disk with an FPFlush call and closed with an FPClose call. The process is then repeated for the resource fork. Finally, the Desktop database must be updated to reflect the presence of the new file. If it contains icons, they are added with a series of FPAddIcon calls. If the file has a Finder comment (that is, one shown by the Finder's Get Info... command) it is added with an FPAddComment call. Finally, if the file is an application, its signature and directory location are added with an FPAddAPPL call.

Copying a file from the server

Compared to writing a file to a server, reading one from a server is relatively straightforward. The client begins by collecting information about the file with an FPGetFileDirParms call and an FPGetComment call. Each fork of the file on the server is then opened and FPGetForkParms calls are made to return the length of each fork.

The client then makes a series of FPRead commands to transfer the file's data from the server. Because movement of data from the server to the client fits in more naturally with the way ASP is structured, the data is simply moved in the SPCommandReply frames that the server issues in response to the FPRead command. Each FPRead, then, can move a maximum of 4624 bytes. The client issues to the server as many FPReads as necessary to move all the data.

Once the data transfer is completed, the file's forks are closed on the server.

Deleting a file from the server

Once again, the seemingly simple operation of deleting a file balloons into an exchange of over 100 commands when it happens on a file server. The client begins by looking for a directory on the server titled Network Trash Folder. If no such folder is found, the client creates and initializes it.

Another directory is then created, if necessary, within the Network Trash Folder called Trash Can #*x*, where *x* is the user's ID number. All files that the user places in the Trash are stored in this folder.

Next, the client transfers the file to the appropriate trash folder. Ideally, this is performed with an FPMoveAndRename call, which allows the server to move the file from one folder to another and optionally change its name in a single operation without further data transfer between the server and the client. Not all servers support this command, however. If necessary, the client creates a new file in the trash folder, reads the original files data from the server, writes the data back to the new file on the server, and deletes the original file. The file stays in this special folder until the user actually deletes it by emptying the Trash.

When this happens, the client sends an FPRemoveComment to the server to take any Finder comment for the file being deleted out of the Desktop database. If the file is an application, an FPRemoveAPPL call is also made to remove any application mapping from the Desktop database. Finally, the file is actually removed from the server.

Conclusion

AFP is the richest and most complex protocol in the AppleTalk suite. Understanding it fully is a challenge and requires familiarity with a broad range of information. Ideally, you should understand the native Macintosh Hierarchical File System in detail and have access to a network analyzer, such as a Sniffer, to observe the various operations that take place between the client and the server.

The sections on AFP operations were intended to serve as only a general overview. A complete exposition of AFP could take up an entire book in its own right. The full set of AFP commands and their responses is detailed in *Inside AppleTalk*.

Interapplication Communication and the PPC Toolbox

A new protocol, introduced by Apple with System 7, allows applications to communicate information directly to one another over the network in a variety of useful ways. The basic mechanism that allows this is called Program-to-Program Communication (PPC).

The PPC Toolbox builds on the AppleTalk Data Stream protocol (ADSP). ADSP is an efficient, full-duplex, connection-oriented stream protocol that was originally implemented as an add-on to AppleTalk, packaged in the form of a start-up file or INIT. With System 7, ADSP has become a regular part of the full AppleTalk protocol suite.

Like ASP, ADSP is connection-oriented. This means that before any true communication can occur between nodes, they must first set up communications with one another. One node starts this process by sending an ADSP OpenConnection request frame that contains a connection ID (which is used throughout the life of the connection), a sequence number for the next byte of data to be transmitted, and the amount of buffer space the initiating node has available for incoming data.

The remote node responds with a combined OpenConnection and acknowledgment frame, repeating the information provided by the first node and adding its own sequence numbers and buffer space figures. The original node completes the process of opening the connection by sending an OpenConnection acknowledgment frame. At this point, the two nodes are in communication until a CloseConnection frame is transmitted by either side.

ADSP differs from higher-level AppleTalk protocols in that it is a symmetrical peer-to-peer protocol. Unlike ATP, there is no distinction between requester and responder or between server and client. Each side of the connection is on an equal footing with the other and each side can transmit essentially unlimited amounts of data to the other.

ADSP is a stream-oriented protocol. The information passed through ADSP between the two sides of the connection is simply an unstructured sequence of bytes. Any further interpretation of the data passed is left to higher-level client protocols of ADSP, such as PPC.

ADSP uses the byte sequence numbers embedded in each frame exchanged to determine whether any intervening data was lost. If the sequence number corresponding to the first byte in the frame does not gibe with what the receiver expects it to be, the receiving node sends out a control frame to request retransmission of the missing data.

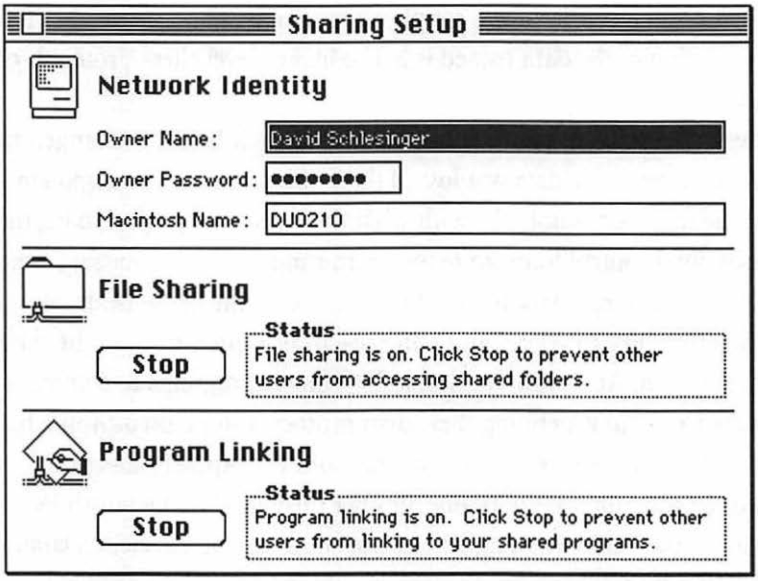
PPC uses ADSP to exchange data between two nodes on the network. It adds a further layer of functionality to ADSP and enforces an interpretation on the data passed over the connection. At its simplest level, PPC allows programs to communicate with one another privately, defining their own protocols to be used. Apple has defined two subprotocols to work within the context of PPC. AppleEvents provides a way for programs to communicate with one another through the Macintosh Event Manager. For example, the Finder running on a remote node can be directed to launch a particular application by receiving an AppleEvent of type 'oapp.' A specific document can be opened on the remote node by sending an 'odoc' AppleEvent, or printed by sending a 'pdoc' AppleEvent.

A second use of PPC on the Mac is "Publish and Subscribe" functionality, also known as the Macintosh Edition Manager. This allows a user of Macintosh programs that support this capability to share selected data from a document with other users on the network by *publishing* it. The published data is saved in a special file called an *edi-*

tion, which can be accessed by other Macintoshes who *subscribe* to it. Subscribing Macs access the data within the edition by using the same AFP operations described earlier. The Edition Manager also allows a publisher to alert its subscribers that the edition has been modified so that subscribing documents can be updated without user intervention.

The process on the Macintosh

All interprocess communication is based on program linking, introduced with System 7. Program linking is turned on separately from file sharing. For a Macintosh to send and receive AppleEvents or other PPC-based services, the user must turn on program linking in the Sharing Setup control panel. Before editions can be published to other users on the network, file sharing must also be turned on because data in editions is moved between Macintoshes with the AFP protocol. The Edition Manager uses PPC to announce to subscribers the availability of updated editions.



Sharing Setup

Network Identity

Owner Name: David Schlesinger

Owner Password: ●●●●●●●●

Macintosh Name: DU0210

File Sharing

Status

File sharing is on. Click Stop to prevent other users from accessing shared folders.

Program Linking

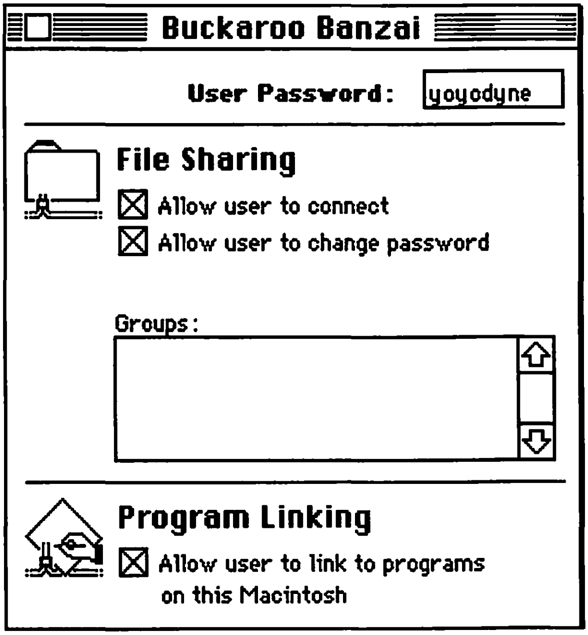
Status

Program linking is on. Click Stop to prevent other users from linking to your shared programs.

Figure 7-29. The Sharing Setup control panel.

But this only allows the Mac the capability of using PPC. For a user to access PPC-based services, he or she must be given authorization. This means that the user of the Mac providing those services must set up a user record for the user who wants to access them.

The Users & Groups control panel is used to create new user records. A password should always be specified. Again, program linking and file sharing capabilities are turned on separately. For PPC-only services, such as AppleEvents, only program linking must be enabled. If the user subscribes to editions, file sharing must also be turned on.



Buckaroo Banzai

User Password: yoyodyne

File Sharing

☒ Allow user to connect

☒ Allow user to change password

Groups:

Program Linking

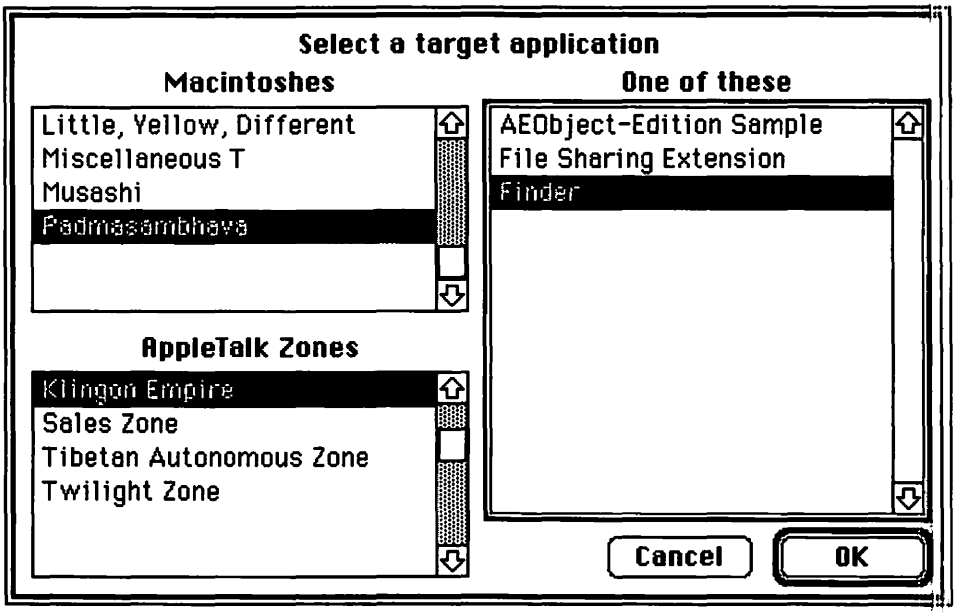
☒ Allow user to link to programs on this Macintosh

Figure 7-30. An example of the Users & Groups control panel.

Assuming that everything is configured appropriately on the Macintosh providing PPC services, the user of those services must first *establish a link* to the service provider. Typically, an application that intends to use program linking brings up a Chooser-like dialog called the *PPC Browser* that allows the selection of a particular PPC service on a specific Macintosh.

Note that the PPC Browser works in a way that is somewhat the reverse of the normal operation of the Chooser. In the Chooser, a type of service—such as AppleShare, LaserWriter, or FAXSender—is initially selected. Next, the Chooser displays a list of devices that are advertising that service. In the PPC Browser, a particular device (generally a Macintosh) is chosen and a list of PPC services are offered to the user. An authorization dialog, identical to that used in AppleShare and FileShare, is also shown to allow the user to log in to the service selected.

Program linking is a very useful capability that developers are only beginning to utilize. Although a growing number of programs are offering Publish and Subscribe capabilities, a few, most notably UserLand's Frontier scripting package and CE Software's QuicKeys, are using AppleEvents and PPC in significant ways. The latest version of Apple's Macintosh Programmer's Workshop also uses PPC and AppleEvents to implement a product called ToolServer, which allows a Mac to be used as a dedicated server for compiling, linking, and building programs.



Select a target application

Macintoshes

Little, Yellow, Different
Miscellaneous T
Musashi
Padmasambhava

AppleTalk Zones

Klingon Empire
Sales Zone
Tibetan Autonomous Zone
Twilight Zone

One of these

AEObject-Edition Sample
File Sharing Extension
Finder

Figure 7-31. The PPC Browser dialog box.

The process on the network

When Terry, the owner of the Macintosh called Terry's Mac, opens the Sharing Setup control panel and turns on program linking, the Mac registers an object with the name Terry's Mac and a type of PPCToolbox on the network with NBP. This informs other workstations that a PPC service of some type might be available on that Mac.

A Mac that opens a PPC Browser sends out a series of NBP LookUps for any objects of type PPCToolbox in the zone the user has selected in the PPC Browser.

Workstations with program linking enabled reply with their names and full network addresses. When a particular workstation is selected in the PPC Browser, the Macintosh that is doing the browsing opens an ADSP connection with the advertised socket on the specified node.

Once the connection is open, the Browser sends its first actual PPC message. This is a List Ports, or LPRT request. Specific PPC services on a given Mac are called ports, and each might use a different socket address to provide its own services. The request contains a starting index to indicate which PPC port should be the first one returned, a count of the maximum number of port names to be returned, and a specific name and type of port that should be returned. Typically, the name and type values are both set to =, indicating that all names and types should be returned. The start index is used in the same way as an index in the FPEnumerate call, which is described earlier in this chapter. If the first LPRT call returns four port names, a second LPRT can be issued with an index of five to return subsequent ports.

The response to the LPRT request is broken into two ADSP messages. The first contains a list of port specifiers, including the name of the registering application for display in the PPC Browser, whether authentication is required to use that port, and a signature string to identify the service provided by that port. Typically, the signature consists of eight bytes, four identifying the application's creator code (for example, 'MSWD' for Microsoft Word, or 'MACS' for the Finder) and four identifying the port type. The port type most commonly seen is 'ep01,' which indicates that the port accepts standard AppleEvents.

The second block of the response begins with the ASCII letters LRSP (for "List Response") and contains the number of ports about which information was returned.

When the user selects a specific program in the PPC Browser, a PPC session is started. The PPC Browser sends out a block beginning with the ASCII letters SREQ. This is a PPC Session Request block, and it contains specifications for the local and remote PPC ports to be linked, the location of the local service described as a machine name, type (again PPCToolbox), and zone name. Before the remote PPC port replies to the Session Request, it attempts to authenticate the user.

The block used to perform user authentication begins with the characters ACNT, for Authorization Continue. The block contains an 8-byte random number, used as data for a random number exchange as done for AppleShare authentication. The user's password is used as an encryption key against the random number supplied. The result is returned to the remote PPC system in an Authentication Response block, beginning with the letters ARSP, followed by the encrypted random number. The remote node, having encrypted the same random number with what it believes the user's password to be, compares the two. If they match, the service provider replies with a Session Accept block, beginning with the letters SAPT. If they do not match, a block beginning with the letters SREJ is sent to reject the session.

At this point, PPC data blocks begin to be exchanged between the two nodes. These are ADSP blocks passed between the two sockets that began communications and that begin with specifiers other than LPRT, SREQ, ACNT, ARSP, SAPT, SREJ, or UREJ. These specifiers are referred to as *event classes*. Common event classes are aevt, FNDR, and sect. The aevt class is used for so-called "core" AppleEvents. The FNDR class indicates events that can be accepted by the Finder, and the sect event class is used by the Edition Manager to indicate changes in the status of published editions.

The four bytes following the event class specify the type of event. Their meaning varies depending on the event class.

PPC and AppleEvents

Apple has defined a small number of "required" AppleEvent types that should be implemented by all applications that take advantage of AppleEvents. These are:

1. The oapp event—open a particular application on the target system.
2. The odoc event—open a specified document on the target system, starting up the appropriate application, if necessary.
3. The pdoc event—print a specified document on the target system.
4. The quit event—quit from the specified application on the target system.

These AppleEvents typically pass between the Finder and a given application, but their use is not limited to this context. All AppleEvent-aware applications are expected to implement these capabilities and can extend them by adding optional parameters.

The required AppleEvents are generally used locally on a single Mac, but this is not always the case. The "drag and drop" capability supported under System 7, for example, can be implemented over networks by using the required AppleEvents. Dragging a document on a local hard disk on top of the icon for an already running application on remote server sends an 'odoc' event to the target application.

PPC and "Publish and Subscribe"

The other main use of PPC on the Macintosh is to help implement of the Publish and Subscribe functions of the Edition Manager, also introduced with System 7. The Edition Manager uses two AppleTalk-based protocols to publish and subscribe applications' editions. AFP is used to transfer the data stored in edition files, and PPC AppleEvents are used by the publisher to advise subscribers of edition changes.

Edition files serve as a sort of network-based clipboard. These files can be identified fairly easily in the Finder or in the Open File... dialog box. By convention, edition files display an icon in the Finder that is surrounded by a gray border. The icons of edition files belonging to various applications are shown in Figure 7-32.

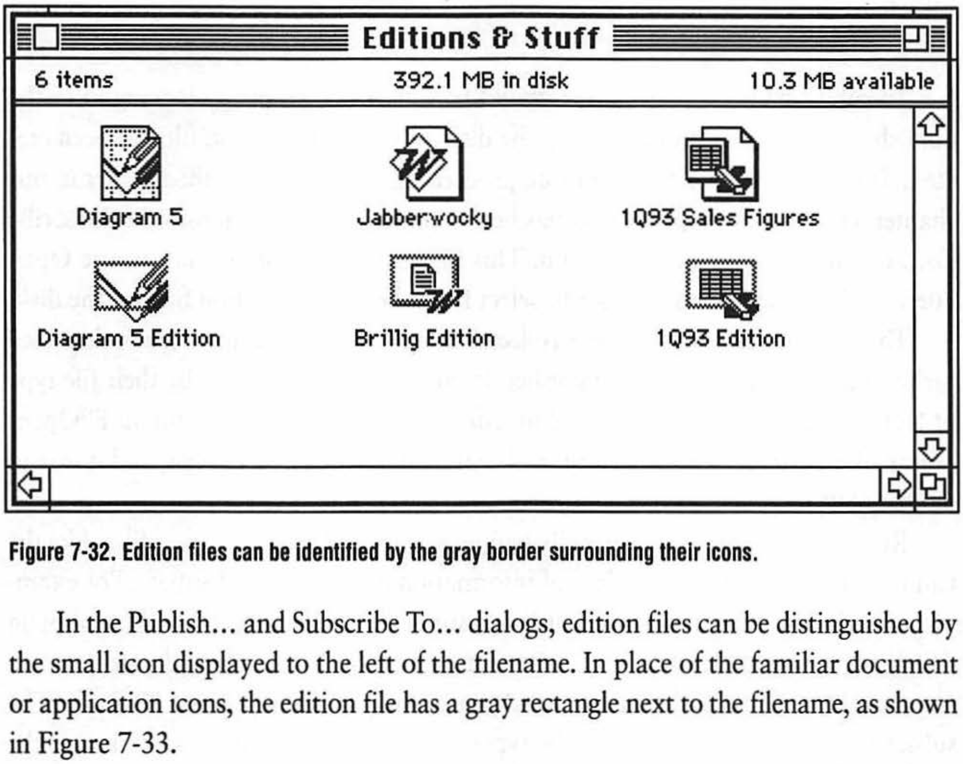
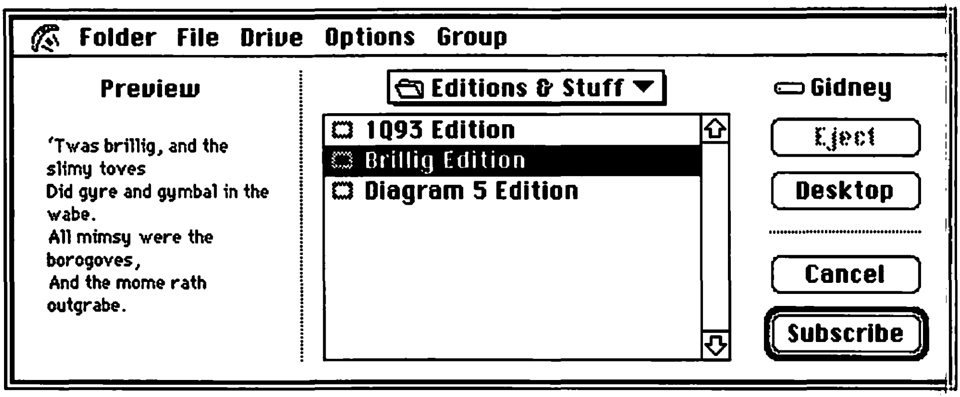


Figure 7-32. Edition files can be identified by the gray border surrounding their icons.

In the Publish... and Subscribe To... dialogs, edition files can be distinguished by the small icon displayed to the left of the filename. In place of the familiar document or application icons, the edition file has a gray rectangle next to the filename, as shown in Figure 7-33.

To publish an edition on the Macintosh, the user chooses Create Publisher... from the application's Edit menu. This creates an edition file on the hard disk with a name specified by the user. No special network interaction accompanies the creation of an edition; for other network users to access it, however, file sharing must be enabled and the disk containing the edition file must be set up to be shared on the network.



'Twas brillig, and the slimy toves
Did gyre and gymbal in the wabe.
All mimsy were the borogoves,
And the mome rath outgrabe.

Figure 7-33. Edition files have their own small icons to distinguish them from other documents in the "Subscribe To..." dialog.

To subscribe to an edition, the would-be subscriber begins by logging in to the publisher's Macintosh and mounting the disk on which the edition file has been created. This is strictly an AFP operation, proceeding exactly as described earlier in this chapter. Once the appropriate disk has been mounted, the user chooses the Subscribe To... command from the Edit menu. This brings up a dialog, similar to the Open File... dialog, that allows the user to select from the various edition files on the disk.

The information in this dialog is collected using the FPEnumerate call, also described earlier. Edition files can be distinguished from ordinary documents by their file type of 'edtt.' Once the user has selected an edition, the file is opened with an FPOpenFork call, its parameters are determined with an FPGetForkParms call, and it is read in using FPRead calls.

Reading the edition file normally happens in several stages. Edition files, like the Clipboard, can store a given piece of information in a variety of formats. For example, a publisher created in Microsoft Word stores published text in the form of plain ASCII text, in Rich Text Format, a format created by Microsoft and used to communicate between several different word processing packages, and as a PICT file. The subscribing application examines the types of publication available and chooses the one best suited to its needs.

So far, PPC has not entered into the picture at all. For the initial acts of publishing or subscribing to an edition, PPC is not utilized by the Macintosh. Instead, the Edition Manager uses AFP in exactly the same ways as described earlier in this chapter to find the edition file and access its data.

When the publisher updates the data, however, PPC is used to inform subscribers that the edition has been changed. To do this, the publisher transmits a PPC data block with an event class of 'sect' and an event type of 'read.'

An Edition Manager 'read' event indicates to the subscriber that the edition has been updated by the publisher, and the information contained in the edition file should be reread so that the document that has subscribed to the edition can be updated appropriately.

Summary

The use of PPC is remarkably difficult to inspect in practice. While the basic protocol is not complicated, its many uses make it as complex as AFP, but it is not nearly as well documented. To further complicate matters, PPC and the AppleEvent protocol that uses it are designed to be extended by third-party developers to suit their own needs.

PPC is an extremely powerful mechanism, with great flexibility. Its capabilities are just beginning to be used by Macintosh applications. You are sure to see more widespread use of the protocol in the future.

CHAPTER EIGHT

Troubleshooting Techniques

When you go to the doctor's office for a checkup, the assistant usually gives you a few quick tests while you're waiting for the doctor. The assistant might check your weight, your blood pressure, your heartbeat rate, and your reflexes, then draw blood and urine samples for routine analysis. The assistant also might ask you a few general questions like, "How have you been feeling lately? Have you had any headaches or felt tired?"

The results of these tests and questions are written on your chart and given to the doctor when he or she arrives. The doctor looks over the chart, checking for results that are unusual in general or just unusual for you. It's a way to begin an investigation of your health. The medical profession has standardized certain preliminary checks because they're quick and easy to perform and because they might lead the doctor to conduct more specific tests if the simple tests show anything suspicious.

With the quantitative data, the doctor will refer to your previous records to find out if your slightly high blood pressure is new or something seen before. It's not only the values of the numbers the doctor's looking at, but how they compare with previously collected data.

Performing a Network Health Scan

The purpose of a quick network health scan is to give you an answer to the general question, "How is the old network feeling today?" rather than to find specific problems. If any of the tests show values that are worrisome, then you can conduct more specific tests to diagnose the problem—if there is one.

Instead of checking blood pressure as a doctor does, check the network utilization level. In place of a blood sample, take a look at the nature of the traffic and find out what services people are using. Instead of measuring the heartbeat rate, check the page count on the LaserWriters or the message count on the mail servers. For the network equivalent of checking reflexes, launch a couple of benchmark jobs to see what response times people are getting. Instead of analyzing a urine sample, look for "waste" packets in the error statistics of your network's hubs. You also might ask the users how they've experienced the network recently. Sometimes questions such as "Any server crashes lately?" and "Have you experienced any unusually slow service?" help you spot a problem before your users notice it. If you run these few simple checks, examine the results, and compare them to previous data, you can get a quick feel for how your network is doing. Unusual results can prompt you to begin an investigation that uncovers a problem before it has highly noticeable symptoms. Even if you don't spot a problem, you should probably run through these two or three times a year anyway; the information you gather can help you with other aspects of network administration besides troubleshooting—or example, resource planning.

By keeping records, you'll get a feel for how your network is changing over time. For example, you'd expect network utilization to increase gradually as users become more familiar with network services and make more use of them. You may also be adding more users and new services. If the network utilization isn't increasing, or if it's actually dropping, try to figure out why. It might be just a normal statistical fluctuation, but it might also uncover a trend that you should monitor for a while. Perhaps people don't like the new version of electronic mail and are using it less, or perhaps the manufacturer has redesigned the new version so that it sends less traffic for the same level of usage. If the network utilization has risen dramatically, more than you'd expect, you might try to analyze that as well. Perhaps people are doing backups during the day or are storing all their personal data files on the server. After examining the traffic in more detail, you may also decide to reexamine the placement of your routers and bridges to make sure that your network's configuration is at an optimum.

You may not have an assistant, but you can still have these measurements taken for you if you're clever with macros and sequences. Nearly all of the checks suggested in the next few sections can be automated. Our favorite automation tools are CE Software's QuicKeys, Userland's Frontier, and HyperCard, all of which can run sequences and exchange Apple Events. Farallon's MediaTracks can also be useful for saving dynamic data, although you have to be careful about using MediaTracks unattended. The size of the MediaTracks data file can grow very quickly, depending on what application you're monitoring and your monitor characteristics.

Keeping records

The success of the network health scan procedures that you develop for your network depends heavily on keeping records of previous testing. Keep your records simple so that you'll be sure to maintain them. Well-kept records are key to the value of this process, since it relies so heavily on comparisons to previous data. Imagine what you would think if your doctor didn't keep records of your visits!

We find that a loose-leaf notebook works best for keeping numerical data, with each page devoted to a single measurement. For graphical data such as a NetStats screen shot, a collection of printed records in a notebook works very well. Later, we may transfer the numerical data to a computer for analysis, but the paper records always form the master data.

Each test you develop for your health scan should have its own definition that defines the tasks and configuration variables. You should also keep auxiliary records that can help you reconstruct the exact conditions of the test should you notice differences during subsequent health scans. These errant values may be caused by subtle configuration changes either in the network or the systems involved. This reconstruction is impossible to do perfectly because there are so many factors that can affect performance.

You need specify only a reasonable number of major factors. For example, if one of your benchmark tests is to upload a file from an AppleShare server on Ethernet to a client Mac on LocalTalk, your definition of this test may look like this:

Benchmark Test Definition

- Task Definition

  Upload the file Aldus SuperPaint 3.0 from Dawn's Macintosh to the Safety Server File Size is 1032KB. Completion time is measured from the release of the mouse button to the return of the pointer cursor after the file transfer.

- Nodes/Protocols/Services Involved

  Source Folder: Dawn::Tiger:Apps:Pictures:SuperPaint—User Macintosh
  Destination Folder: Safety Server::Archives:Test Folder—AppleShare Server

- System Configuration Information

  Dawn's Mac: Mac IIcx, 5MB RAM, 80MB Quantum Drive with Apple 8-bit video card and 13" RGB Monitor. Hard disk has approximately 45MB used, with a fragmentation index of 0.03. System Version is 7.0.1, Apple Share 7.0. Dawn's system profile at the time the baseline data was generated on the Admin Server in Managers:Benchmarks:Test Configs:File Transfer:Dawn.

The Safety Server is a Mac IIfx with 20MB RAM, a 210MB Hard Drive server. Server's system profile at the time the baseline data was generated is on the Admin Server in Managers:Benchmarks:Test Configs:File Transfer:Safety Server.

- Network Configuration Information
  Dawn is on LocalTalk Network: 2035, Zone: Safety Zone
  Safety Server is on Ethernet Network: [10-15] Zone: Safety Zone
  Intermediate Devices: GatorBox CS. TCP/IP: ON, GatorShare: Not Loaded, No Tunnels

Check the network utilization

The Media Access Control methods of both Ethernet (collision detection) and LocalTalk (collision avoidance) are referred to as *contention networks*. These networks offer advantages when the traffic is "bursty" and strict control over the timing of delivery is not critical.

By contrast, Token-Ring networks have advantages when traffic is consistently heavy and process-to-process communication relies on predictable and timely data delivery, such as is necessary for distributed processing or real-time control. In reality, however, most networks with DEC or UNIX histories are usually Ethernet; on the other hand, networks with a long tradition of IBM mainframes and minicomputers are oriented towards Token-Ring.

Bursty traffic is characterized by sporadic use of the network by individual nodes. Typical users on such a network may download the data file of a drawing they've been working on from the file server and then not need the network for 10 or 20 minutes. Then they'll retrieve their mail messages and stay off the network for a while, working on a drawing or chatting on the phone. Later, they'll finish their changes to the drawing, save the file back on the file server, and send the file to the printer on they way to lunch.

Both Ethernet and LocalTalk are well suited to this kind of traffic. Generally, users are not encouraged to keep documents open on a server because an unexpected loss of connection can be disastrous.

Characterizing bursty traffic

Bursty traffic, by its nature, fluctuates with time. A measurement of network utilization is only a snapshot of a very particular time; it's likely to be quite different a short time later. Network statistics are usually given as an average over a specific time period rather than as network measurements.

For example, you might characterize the utilization of a particular Ethernet by saying that during the busiest hour of the day its utilization averaged 4 percent; during the busiest minute of the day, it averaged 26 percent; and during the busiest second of the day, network utilization averaged 49 percent. You can compare this information to previously captured data and then compare it against an ideal maximum.

If you haven't yet developed an ideal maximum for Ethernet, reasonable numbers to use are 15, 40, and 75 percent for the busiest hour, minute, and second, respectively. If your network utilization is above this level, you should probably examine the traffic and reexamine your network design.

The reason you should keep utilization of Ethernet at such low levels is that the rate of Ethernet collision errors increases as the utilization increases. By keeping the utilization rate low, you keep the error rate low. With LocalTalk, in contrast, you can use a higher rate of utilization without a corresponding rise in error rate, although the apparent performance of your services may fall as the network becomes busier and stations have more difficulty gaining access to the network. In LocalTalk, you might use 50 percent utilization (averaged over an hour) as the threshold for further investigation for reexamining your design.

Before you rush to the drawing board, however, try to ascertain that the level of traffic you're seeing is normal, caused by legitimate users and devices going about their usual daily tasks. This isn't always the case; you should also consider the possibility that a device on your network (or a specific group of devices) is malfunctioning and creating an artificially high traffic load.

One of the most documented (and most feared) errors is called a *storm*. In a storm, a confused device starts spewing out incredibly large numbers of packets as it tries to resolve its confusion. In AppleTalk, we've experienced AARP storms (most of these were caused by a particular router problem in 1987 and 1988) and ZIP storms (caused routers again, in 1991). Protocols besides AppleTalk can also have storms. (If you'd like to experience a storm for yourself, give the identical TCP/IP address to two VAX computers that are running both DECnet and TCP/IP. They must be on unconnected networks while they boot up. Then join the networks together with a router, grab your umbrella, and watch the clouds gather.)

Storms are characterized by a large number of one type of packet coming from one particular device or from one particular *kind* of device (often routers). Usually the storm results from either a bug in the software of the device or the device's inability to handle a particular kind of situation. ZIP storms (which generally result from corrupt routing tables) are described in more detail in "Troubleshooting Router Configurations," later in this chapter. The AARP storms we've seen in the past resulted when a FastPath 2 or 3 tried to resolve the Ethernet address of a device with an AppleTalk node address of 0, a nonsensical situation for which the router software was unprepared.

Traffic monitoring tools

To measure the utilization of LocalTalk over a short period of time, use Farallon NetStats. NetStats is very simple to use; you simply watch a moving graph of utilization vs. time. The only way to save or print NetStats data is by doing a screen capture. NetStats' only setting is for sensitivity of the time scale. Your choices impact the duration of the scale. On the most sensitive setting—10 samples per second—duration is around 42 seconds. On the least sensitive setting—one sample per second—you can see about seven minutes of activity at one time.

An interesting way to use NetStats is to create a recurring QuicKey sequence that calls your screen capture utility and saves the screen shot to disk at regular intervals. You can use NetStats on the least sensitive time scale and tell your QuicKey sequence to take a screen capture every seven minutes. For the screen capture, we use Flash-It, a shareware program ($15). We like Flash-It because once you specify all the options Flash-It can do its job without needing you to respond to a dialog box.

For example, when the Caps Lock key is down, Flash-It takes a screen shot of the active window only (NetStats, in this example). It can then save the screen to any of several destinations specified beforehand. The destination can then by select by QuicKey sequence. (We typically specify a ScrapBook file as the destination for this procedure.)

One tip: set the monitor for two colors only. There's no advantage to having more colors for NetStats, and the screen shot sequence runs faster.

There's a bug in NetStats, though, that may occasionally ruin your testing. Occasionally, NetStats will "peg", or indicate a network utilization of 100 percent that is clearly not an accurate measure of network traffic. It doesn't usually recover from this error, and if this happens during your automated testing you may come back to a useless set of results.

There are some very powerful tools in some of the more expensive Ethernet hubs available, and high-end protocol analyzers like Network General's Sniffer also do a fairly good job of recording and displaying utilization data. Both the AG Group's Skyline/E and Dayna's NetScope (neither of which is shipping at this writing) also appear potentially useful for traffic characterization, but only for Ethernet networks in their first release.

If you spot irregularities in your data, or would like to read more about traffic analysis, more suggestions for traffic analysis are provided in "Analyzing Network Traffic," later in this chapter.

Check the service activity

Several kinds of servers have ways of telling you how much they've been used, and these activity monitors may also help measure network health. A great example is a Remote Access server, which keeps a log of its activity.

In QuickMail, for example, you can check the Mail Log to see who has sent messages to whom. Although it doesn't provide you with a statistical breakdown, you can get a very good idea of how heavily your mail services are being used. You can count how many messages are sent per day.

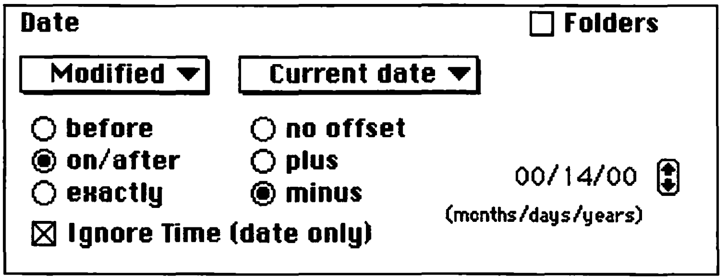
QuickMail has other utilities as well. By looking at the quantity of undeliverable mail, you can get an idea of how far out of date your users' address books are and whether they need to update them. If you use a different package, such as Microsoft Mail or WordPerfect Office, there are probably other ways to characterize your mail server's usage. Develop a way to measure the usage and begin to track it. You may also discover a way to automate the reporting procedure.

For LaserWriters, an easy measure of usage is the page count, printed on the startup page; you can also obtain it through Apple's LaserWriter Utility. If you use AppleShare Print Server or any other spooler that keeps a log of activity, you'll be able to see which users printed exactly how many pages and when they printed them.

While AppleShare 3.0 still doesn't have a good set of statistics to show you how much the server has been used, you can determine the percentage of the file system that changes during a certain period of time. With Dantz's Retrospect, you can see how many files have been modified during a certain time period by means of the Choose... criteria in the Selection Window.

Figure 8-1 shows the settings to select files that have been changed in the last 14 days. Retrospect tells you how many files you've selected for backup and how many bytes they'll consume on your destination volume. To get an idea of the level of change, first select the folders of interest and write down what Retrospect tells you about the number of files and their collective size. Then use the Chooser criteria to select only those files that have changed. Using this method on a server volume of 164MB and 2704 files, Retrospect tells us that there are 82 files that had changed in the last two weeks, with a collective size of 5.6MB. This represents about a 3.5 percent change in a two-week period, a relatively light amount of activity.

This method of measurement has its weaknesses; an obvious one is that, in the absence of a more direct method, a file that has changed 1000 times and a file that has changed once will appear the same. Still, this method has its merit for certain situations.



Date □ Folders

Modified ▼ Current date ▼

○ before ○ no offset
● on/after ○ plus
○ exactly ● minus

00/14/00
(months/days/years)

☒ Ignore Time (date only)

Figure 8-1. Using Dantz' Retrospect, you can find the number of files on your file server that have been modified in the last two weeks. This may be the best available measure of server activity.

Run benchmark operations

Another quick check is to run a couple of benchmark operations that use your network services. In order for a benchmark to be useful, you'll need to be familiar with the range of values it can have in normal operation. Start by running the benchmark when no one is using the network or any of its services. Next, when the network is known to be healthy, run the benchmark several times during the day to see how it varies from this standard time of completion. It's a good idea to set a limit on how great the difference between your scan value and the benchmark standard can be before you try to figure out what's causing the difference.

If you've never developed a benchmark operation, now is a good time to develop one. Good benchmarks are real-world operations, performed to see how long they take to execute under variable conditions. When you develop a benchmark, you should try to keep all of the variables of the operation constant except the variable you're testing.

For our network health scan, we're interested in understanding how the completion time varies by time of day during a normal workday. For example, we may want to measure how the responsiveness of a particular file server varies with time. Our benchmark could be the transfer of a particular file on the server to a particular place in the directory structure of a particular client. We'll perform this operation and measure the completion time at different times of the day, then plot the results to see the magnitude and characteristics of the range of data.

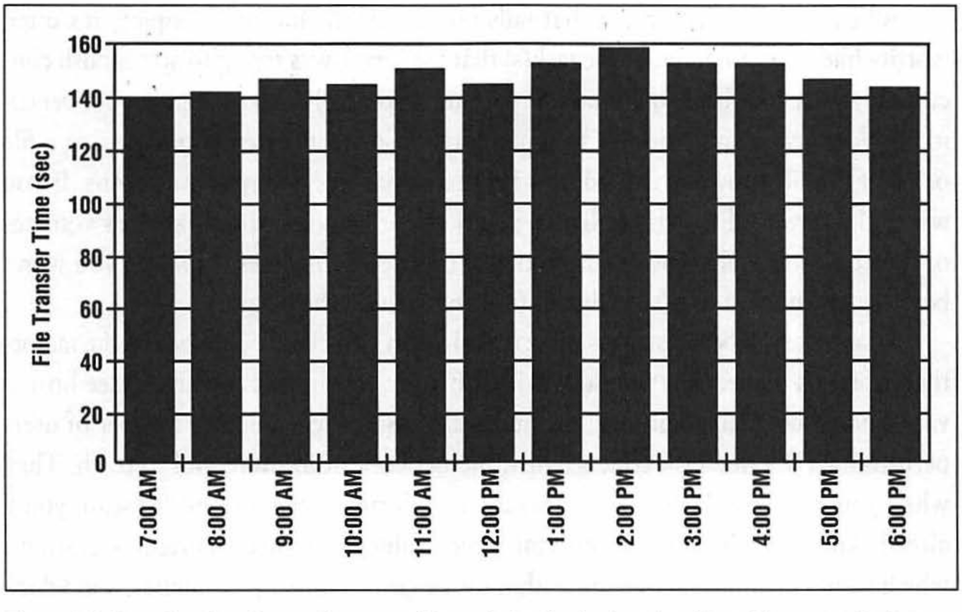


Figure 8-2. Run a benchmark operation several times during the day to get an idea of the range of values.

Figure 8-2 shows our benchmark file transfer results. The values range from 140 seconds when the file server and network are unloaded to a maximum of 158 seconds in mid-afternoon. One useful way of characterizing the data is to express the range of variation as a percentage of the average value, which for this data is slightly over 10 percent. This is a reflection of how heavily the server is taxed during its daily operations. The higher the percentage of variation, the greater the degree to which the server is taxed.

Two factors control the value of this measure: the *variation* in the number and kind of tasks that a server may be performing at the time of the test, and the *degree* to which those tasks consume the server's resources. In our testing, the dominant factor is the former, which is evident from the relatively even distribution of completion times seen. This kind of data indicates a server that can handle its tasks with relative ease but that receives a variable level of user transactions during the day. If the dominant factor is the latter (the server is highly stressed by its tasks) then you'd see much more dramatic variations in the data. While a good portion of the data may fall within a well-behaved range, individual values may vary as much as 100 percent or more.

Of course, this is a very small amount of testing, and it may well be that we need more tests to make this kind of characterization of the server. Perhaps another test on another day will take 300 seconds.

When you get a data point that falls far outside the normal grouping, it's often worthwhile to try to pinpoint the task(s) that the server was trying to accomplish concurrently with your benchmark task. If you run a protocol analyzer during your benchmark file transfer, you'll be able to spot another client concurrently transferring a file or see if the slowdown is caused by a lot of lost packets and retransmissions. If you watch the server's disk activity lights, you'll notice if another of the server's volumes or tape unit starts blinking during your file transfer. Sometimes, however, you won't be so lucky and you won't be able to tell what caused the delay.

In summary, a benchmark is most useful when you clearly understand the factors that affect its value. You should work with your benchmark a while to see how it varies under normal conditions: the number of users logged in, the number of users performing the same task concurrently, the network utilization, and so forth. Then when you take your benchmark measurement during a network health scan, you'll already know which values are normal, which values indicate concurrent operations, which values are unusual but still within the range of normal possibilities, and which values definitely indicate trouble.

Besides file transfer, here are some suggestions for benchmark tests:

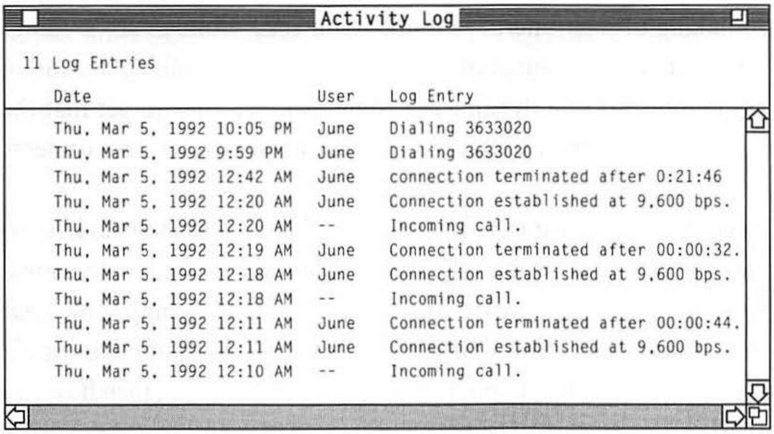
- The time required to print a particular word processing file with graphics
- The time required for a particular user to be notified that he or she has mail after it's sent, with sender and receiver on the same mail server
- The time required for a particular user to be notified that he or she has mail after it's sent when sender and receiver are on different mail servers
- The time required to type the contents of a particular text file on a host computer

- The time required to receive the results of a particular search on a network database
- The time required to contact a particular server and start up its administration program

Check error logs and statistics

Several servers keep error logs that can tell you when they encounter unusual conditions. Some hubs and routers also keep error logs. A quick check of these error logs on a regular basis can sometimes help you spot trouble. It may not be obvious at first which log notations indicate trouble and which do not, so you may also have to spend some time becoming familiar with the notations that these log files use.

Log files may help spot potential problems, even if they're not specifically error logs. The Activity Log that Remote Access maintains, for example, may provide some clues as to which users might need a little more training or which ones may be having equipment problems.



```
11 Log Entries
   Date                        User    Log Entry
   Thu, Mar 5, 1992 10:05 PM   June    Dialing 3633020
   Thu, Mar 5, 1992 9:59 PM    June    Dialing 3633020
   Thu, Mar 5, 1992 12:42 AM   June    connection terminated after 0:21:46
   Thu, Mar 5, 1992 12:20 AM   June    Connection established at 9,600 bps.
   Thu, Mar 5, 1992 12:20 AM   --      Incoming call.
   Thu, Mar 5, 1992 12:19 AM   June    Connection terminated after 00:00:32.
   Thu, Mar 5, 1992 12:18 AM   June    Connection established at 9,600 bps.
   Thu, Mar 5, 1992 12:18 AM   --      Incoming call.
   Thu, Mar 5, 1992 12:11 AM   June    Connection terminated after 00:00:44.
   Thu, Mar 5, 1992 12:11 AM   June    Connection established at 9,600 bps.
   Thu, Mar 5, 1992 12:10 AM   --      Incoming call.
```

Figure 8-3. The Remote Access Activity Log

Looking at the Remote Access log file in Figure 8-3, you can see that June had two connections that quickly terminated before she was able to make a longer connection. Seeing this information, you may ask June about this incident. She might say, "I couldn't get it to work until I turned off the hardware handshake." If you know that other people have this type of modem, you may want to alert them to the potential problem.

On your network's hubs, check the error statistics. You'll find more information on the errors that hubs report in Chapter 6, "Data Link Troubleshooting."

Finding and Correcting Router Configuration Problems

With the advent of Phase 2, AppleTalk internetworking has become more flexible than ever. Network managers have many more possibilities in assigning network numbers and zone names. Some managers of large Phase 2 networks have made extensive use of the ability to assign multiple zones to a single Ethernet, some even designating nearly a hundred zones to a single Ethernet. In doing so, many managers have discovered an old problem with data entry: the more data you have to enter, the greater the chance of error. When routers are configured improperly, either by design or by mistake, your network can be affected in unpredictable ways.

A special note about router diagnosis: in teaching people to diagnose router problems, we've found that it takes a while to become proficient with this kind of investigation. Two conceptual hurdles stand in their way. The first is that network managers are used to thinking of their internet's routing and zone tables as static things that get set up, turned on, and then function. They're not used to thinking of routing and zone tables as components of a dynamic, ongoing process. They forget that the routers are constantly exchanging information, comparing and making adjustments to their tables.

The second, and possibly more important, hurdle is that troubleshooters have to learn to be drawn to and informed by the abnormal conditions they find. In the introduction to this book, we discussed how troubleshooting begins by comparing *what is* to *what should be*. You must develop sharply defined expectations of *what should be* before you can notice eccentricity. If you don't know what to expect, then how can anything be unusual?

There's another side to that coin, however. When a person has highly developed expectations, the natural tendency is to reject information that falls outside the expectation.

For example, it's recorded that when George Vancouver and Peter Puget first sailed into what would become Puget Sound, they anchored within sight of an Indian village. Not only were the Indians not alarmed by the presence of a ship that was a hundred times larger than any boat they'd ever seen, for two days they didn't even know that it was a ship! They thought it was a cloud. A ship of that size that carried men from far away simply didn't fit into their concept of "what the world contains."

When you're troubleshooting, you'll see eccentricity; you'll even be on the lookout for it—unusual behaviors, unfamiliar images, and so forth. When you see it, you can't toss it aside. You must learn to focus on the eccentricity rather than disregard it. In diagnosing router problems, as in many other kinds of troubleshooting, those eccentricities are the keys to the success of the diagnosis-but not *because of* the unusual image or strange behavior. The eccentricity you notice is simply an *effect* of the error condition you're trying to locate. Sometimes it isn't even a primary effect, but the effect of an effect.

Diagnosis depends on making a conceptual leap backward from the effect of a problem condition to its cause. You must imagine what kind of condition would cause the eccentricity that you notice, or you'll go on thinking, "It's probably just a cloud."

How router problems start

The most common way for erroneous network and zone information to be introduced into an AppleTalk internet is through data entry errors. During the router configuration process, for example, the network manager might misspell a zone name, forget to perform one of the configuration procedures (such as entering the zone list), or forget to set one of the optional parameters. He or she might also enter an improper network number, duplicating a network number that exists elsewhere in the internet or contradicting the information held by other routers already in service.

Errors may also creep in when users other than network managers configure routers. Sometimes a user outside the jurisdiction of the network management staff but connected to a common network may purchase and install a router. This user may be completely ignorant of the existing internet's management policy, networking addressing rules, and zone naming conventions. These "rogue routers" are often configured with conflicting network and zone information and may infect the legitimate routers.

A more innocent scenario for the introduction of a rogue router (with similar destructive potential) occurs when a user purchases a remote dial-in product, such as Farallon Liaison or Shiva NetModem, that can also be configured as a router. The user, because of his lack of experience, may intend to configure the product for dial-in access only, but may inadvertently create an AppleTalk router with erroneous network and zone information.

A third way for erroneous routing information to enter the network occurs when a router's information tables become corrupted. Routing information can be corrupted when the router's software has to deal with an unforeseen situation or as a result of bugs in the router's hardware or software. An example of router corruption, the dreaded "ZIP storm," is a condition that happens only in very large internets (greater than 150 networks), but is detailed here because it illustrates the corruption mechanism well. Keep in mind that this condition is rare; there are many more mundane ways in which a router's memory can become corrupted.

A ZIP storm can occur when a router's Routing and Zone Tables contain so many network and zone entries that they exceed the router's available physical memory space (RAM). A router's software should, of course, prevent a memory overload from occurring—or at least gracefully shut down before nuking the network. Unfortunately, the programmers of some routers evidently failed to foresee this condition and created no procedure to prevent catastrophe.

A network manager sets the ZIP storm mechanism in motion when he or she adds a new router or a couple of extra networks or zones to an already large internet. Some routers in the internet, whose memory spaces are already completely used up, learn this new network and zone information and create new entries in their Zone and Routing Tables *on top of* memory already being used by another software process. Because of the memory overlap, the router's software processes become confused about which process is using which part of RAM.

What happens next depends on what type of router and whether the router is configured for AppleTalk Phase 1 or Phase 2. The RTMP process may, for example, begin to interpret the alphanumeric characters of the Zone Table as if those characters were Routing Table entries—network identities. The RTMP process would then treat these false networks (with random networks numbers and ranges) as if they were real networks.

In AppleTalk Phase 1, the next time the router sends out its RTMP information, it may advertise these phantom networks to all other routers in addition to the legitimate networks in its routing table. In Phase 2, this router would probably not advertise the phantom networks but send out a large number of ZIP Queries for both its legitimate and phantom networks.

In the Phase 1 scenario, the other routers that receive these RTMP packets record these phantom networks in *their* already full routing tables. They then make inquiries about what zone names are associated with the phantom networks, creating new entries in their Zone Table. The new entries for the phantom networks in the Routing Tables and Zone Tables, of course, create more memory overlap and the overloaded routers create a new batch of phantom networks. These routers then advertise the newly created phantom networks in *their* RTMP packets, which is followed by more ZIP traffic, more phantom networks, more memory overlap, and so on. The process may continue through a few more cycles, but soon the internet is completely flooded with ZIP and RTMP packets, all the AppleTalk routers roll over and die, and the AppleTalk internet experiences catastrophic failure. (For you Monty Python fans, this is similar to the result of having "one little mint" after a big dinner.)

In a Phase 2 scenario, you don't create such a spectacular failure, but the internet begins to act very strangely. Networks and zones don't seem to be as strongly linked to each other as you're used to.

Avoiding a ZIP storm is a simple exercise in supply and demand. The supply of memory space in each of your routers must be greater than the routing information requirements of the internet. The network manager may either use only routers that have enough memory to hold the required network and zone information, or keep the required network and zone information small enough to fit into the memory space of the router. The usual method of accomplishing this goal is to keep the zone names very short so that less memory is used per zone name or to restrict the number of networks and zones. Also, some routers allocate memory for extended networks in an amount proportional to the size of the address range. In internets that use these routers, network administrators should avoid using large network address ranges.

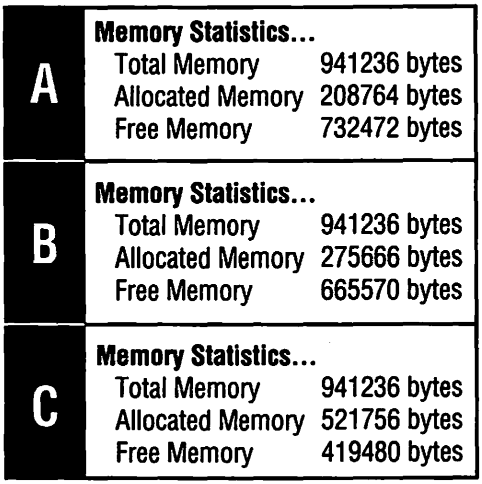
If you're on a large internet and you're now worried about being on the brink of this disaster, you can verify your concern or alleviate your worry by using your router configuration software (if it allows you to check how much memory is available) to find out how many bytes are free. Most routers don't currently offer this ability, but it will be a standard feature of the AppleTalk SNMP MIB.

Figure 8-4 shows the effect of the number of zones and networks on the memory of a GatorBox CS under various conditions.

- "A" shows the memory available in the GatorBox CS on an internet with 10 networks and 13 zones.
- "B" shows the memory report from the same GatorBox on an internet with 212 networks and 164 zones, where most zone names are 10 characters long.

- "C" shows the free memory with 455 networks (420 of them extended) and 235 zones (15 zones per extended network), using approximately 19 characters per zone name.

In each of the three conditions, the GatorBox is configured for Phase 2 AppleTalk only; it includes no additional routing or gateway services that would consume memory.



```
A  Memory Statistics...
     Total Memory      941236 bytes
     Allocated Memory  208764 bytes
     Free Memory       732472 bytes

B  Memory Statistics...
     Total Memory      941236 bytes
     Allocated Memory  275666 bytes
     Free Memory       665570 bytes

C  Memory Statistics...
     Total Memory      941236 bytes
     Allocated Memory  521756 bytes
     Free Memory       419480 bytes
```

Figure 8-4. The memory statistics from a Cayman GatorBox CS for three different internet configurations.

In each condition, there was no load on the router besides routing and zone information, and no other services were offered besides AppleTalk Phase 2 routing. As you can see, the GatorBox CS handled the situation with quite a bit of memory to spare. A FastPath 5, a FastPath 4 (512K), an APT ComTalk, and a Webster MultiGate were also on the internet. All routers could handle situation B) and all of the routers but the FastPath 4 were able to handle the huge routing and zone tables required for situation C).

Via these three avenues for erroneous data—errant data entry, the introduction of rogue routers, and the corruption of router memory—many AppleTalk networks, both Phase 1 and Phase 2, have developed router configuration problems. How the problem manifests itself—what symptoms it has, how much damage it creates, how long it takes before it can be seen, and so forth—depends not only on the nature of the error, but also on which manufacturer's routers are involved and which AppleTalk Phase is used.

Router configuration problems can either manifest themselves as catastrophic network problems or as minor, infrequent annoyances. Some router conflicts have no user symptoms for long periods of time until a small change in the internet creates the special situation in which the symptom can show itself. We often ask the local network manager if we can look for router configuration problems, even if the network manager does not suspect a router problem. In more than half of these cases, we have found router configuration errors.

AppleTalk's susceptibility to router configuration errors

Router problems are common in AppleTalk internets because of certain design aspects of AppleTalk's Network and Transport Layers. These include:

1. *A high degree of differentiation between the roles of routing and non-routing nodes.*

Routers are required to know everything about the structure of the internet; non-routing nodes know almost nothing about how the internet is built and are completely dependent on routers.

An example of the non-routing node's lack of knowledge is demonstrated when a Macintosh user opens the Chooser. The Mac, a non-routing node, needs a router to gain the zone list that the Chooser displays. On a network with more than one router, the Mac will select different routers on repeated openings of the Chooser. The reason for this is that the Mac will pick the router from which it most recently received a packet, and ask it for the zone list. In a multirouter network, where different routers have different information, users will experience inconsistent symptoms depending on which router their workstations choose for routing information. This makes users less likely to report symptoms.

In addition, most network services register the zone component of their service name as simply "*," or "this zone," which matches all zone names. Under most conditions, non-routing nodes will respond to an NBP LookUp that matches a service name and type fields regardless of the content of the zone field. In some cases this is a blessing, because it allows users to continue using services despite the fact that their router configurations are in conflict. However, it also masks the problem and makes detection more difficult.

2. *The lack of a standard network management protocol to detect and coordinate router inconsistencies.*

Each vendor has implemented a proprietary method for passing messages, gathering statistics, detecting errors, and reporting events. Although Apple has announced support for the SNMP standard, it will be a while before the SNMP definitions are finalized and vendors have uniformly implemented them and worked out bugs. We expect that SNMP management of AppleTalk routers will be universally implemented and useful before late 1993.

Further, because SNMP, in its present form, has no method for authenticating the network manager other than a password transmitted in clear text (not encrypted), Apple has decided not to implement the ability to change router configurations using the SNMP "set" command. SNMP, for the time being, will simply provide a universal way of checking router configurations, but not correcting them.

3. *A latitude in how vendors interpret key points of the RTMP and ZIP specifications.*

A great diversity of behavior exists in the current generation of AppleTalk routers that results from a looseness in the way the specifications of the RTMP and ZIP protocols are written. The most important way in which router behavior varies is in the way that seed routers handle network information conflicts. Seed routers are preconfigured with network and zone definitions for the networks to which they are directly connected. When they enter service, there are three basic strategies for how routers can handle conflicts between their network configuration information and the information contained held by routers already in service on the network. The first strategy is to ignore any conflicts that may exist, which is the strategy of the Shiva FastPath series, the Cayman GatorBox (version 2.0 and later) and the Webster MultiGate router. These routers don't even check to see if there are any conflicts. Routers that follow the other two strategies all check for conflicts, but handle the conflicts in different ways. One of these strategies is to simply refuse to enter service if network information conflicts exist, which is the method of the Apple Internet Router, as well as the Cisco, Wellfleet and Novell NetWare routers. The final way to handle conflicts is the so-called "soft seed" approach, which is to take on the configuration of other routers. The soft seed approach, however, is not defined at all by

Apple, so almost every manufacturer that implements a soft seed approach has their own particular algorithm. Soft seed routers include the APT ComTalk, the Compatible Systems routers and the Cayman GatorBox prior to version 2.0 (soft seeding is also a setting on versions 2.0 and later). Finally, of the routers that check for conflicts, some of them continue to check for conflicts while they are in service (NetWare, Cisco, Wellfleet), but most check only at start-up.

This is the most important way in which router behavior varies because it confounds the relationship between cause and effect. The symptoms that your network experiences will depend on the order that routers of different manufacturers enter their service life.

The second way router behavior varies is in the way they age out old network information and zone information. *Inside AppleTalk* prescribes a method, but this is followed very loosely by the current generation of routers. A router can take anywhere from 40 seconds to two minutes to age out network information if there are no redundant routes through the internet, and much, much longer if there are redundant routes.

4. *The way in which Apple has implemented a special class of routers, called non-seed routers.*

Non-seed routers learn their network and zone identity from pre-existing routers when they boot. In the event that a non-seed router boots on a network containing multiple routers with conflicting information, no definition exists of how the non-seed router should respond. Some non-seed routers seem to take information from the first router they hear; others seem to take the information from the last router they hear. Another problem is that non-seed routers, once established on their networks, are indistinguishable from seed routers. You can't designate a "super-seed" router that will provide seed information to non-seed routers as they boot. Any router that has completed its boot cycle will provide seed information.

5. *AppleTalk's simplistic way of determining the best route to a given network.*

In LocalTalk and other Phase 1 networks, a non-routing node with a packet to send to a distant network simply sends the packet to the last router from which it received a packet. That router then consults its routing tables to determine the path. The router's own routing algorithm to determine the best route is made solely on the basis of the shortest hop count. No considerations are given for the speed of the links that might be intermediate between the router and the distant network, how reliable the intermediate links might be, whether one link might be toll-free while another link may incur a monetary usage charge, or how busy the intermediate links might be.

While some attempts have been made by half-routers to artificially add hops to raise the "hop cost" of remote links, these efforts have been only partially successful. As a result, AppleTalk doesn't make very effective use of redundant paths. In the case of redundant routers, the order in which the routers are turned on is often the determining factor as to which route is chosen.

Because of these design aspects of AppleTalk, AppleTalk routers tend to:

- Accept bad routing information
- Keep bad routing information because they have no way of resolving conflicts
- Pass along bad routing information to other routers and non-routers
- Be hard to diagnose when they have bad information
- Be difficult to correct when the error is detected

When the routers are in conflict with their routing and zone information, there are characteristic patterns and symptoms that deviate from normal. As mentioned earlier, learning to spot the patterns and interpret the symptoms is the key to diagnosis and correction.

Easy ways to identify router configuration errors

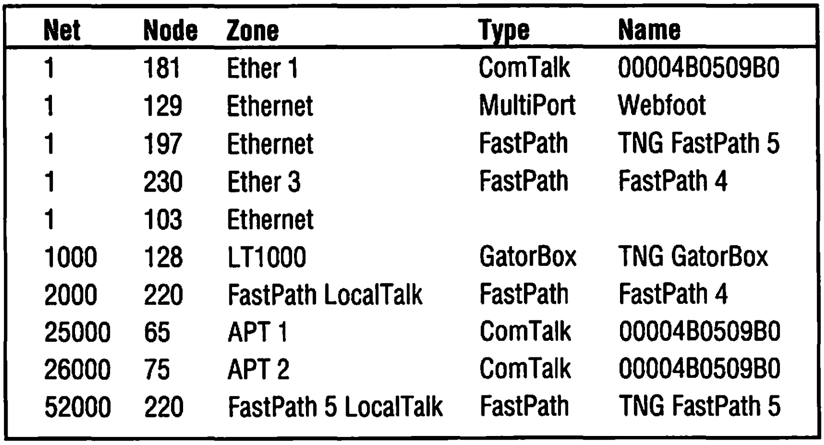
Many router problems are very subtle and complex, and the diagnosis methods presented in this chapter will rely almost entirely on packet analysis. Packet analysis, while difficult at first, offers the surest method of noticing router problems.

Other network management tools can also help you spot router problems. These tools, however, are not always effective because they are limited to the situations that their programmers conceived while creating the tools. Error conditions in routers are, as described earlier, unpredictable in both cause and effect. These tools can spot only the problems their programmers designed into the software. Such tools, however, can be effective at noticing a problem, even if they cannot tell you exactly what it is.

An example of a router management tool is Neon Software's RouterCheck (Version 1.0), which we've used in many situations. When there is a problem, RouterCheck is generally reliable at noticing it, but it is generally ineffective at determining exactly what the problem is. In addition, RouterCheck, even when set for very large memory sizes, crashes when analyzing large internets.

When you first open RouterCheck, it searches for the routers in your internet and displays a list of those that it finds. After it locates and lists the routers, you can perform several information-gathering and comparison functions. Of these, the most useful are the Conflicting Zones, Missing Zones, and Zone Locator routines. The first two routines ask each of the routers for a zone list and automatically compare the lists, noting any anomalies. Zone Locator makes a report that includes the zone list for each router so that network managers can compare the lists themselves.

Here's an example of using RouterCheck: The error situation is an internet with five Ethernet/LocalTalk routers, each connected to a common backbone network. The desired condition is that the zone list for the extended backbone network should include four names: Ethernet, Ether 1, Ether 2, and Ether 2. In one of the routers, however, an extra space was typed after Ether 2 during configuration. This is an easy error to make and will serve for the demonstration of RouterCheck.



Net	Node	Zone	Type	Name
1	181	Ether 1	ComTalk	00004B0509B0
1	129	Ethernet	MultiPort	Webfoot
1	197	Ethernet	FastPath	TNG FastPath 5
1	230	Ether 3	FastPath	FastPath 4
1	103	Ethernet		
1000	128	LT1000	GatorBox	TNG GatorBox
2000	220	FastPath LocalTalk	FastPath	FastPath 4
25000	65	APT 1	ComTalk	00004B0509B0
26000	75	APT 2	ComTalk	00004B0509B0
52000	220	FastPath 5 LocalTalk	FastPath	TNG FastPath 5

Figure 8-5. RouterCheck's router list

Figure 8-5 shows the router list from RouterCheck after a scan was done on the entire internet. The relationship between the number of routes and the number of entities in the list is not straightforward, however. There are five routers and ten entries in the list, but there are not two entries per router. Three of the routers (GatorBox, FastPath 4, and FastPath 5) have two ports each: one LocalTalk and one Ethernet.

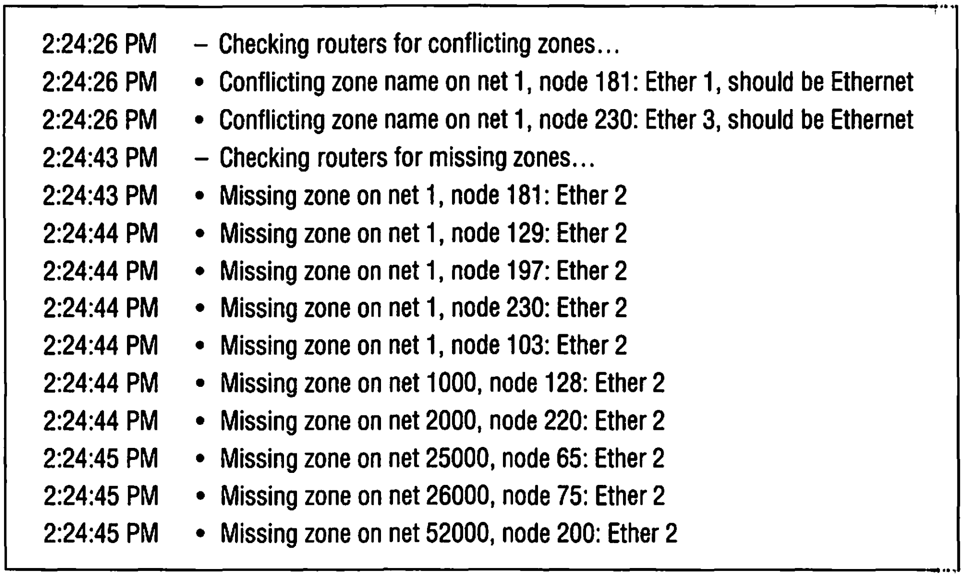
You can see each of these two-port routers listed twice below: once for their Ethernet port on network 1, and once for their LocalTalk port on networks 1000, 2000, and 52000, respectively. Node 1.103 is the Ethernet port of the GatorBox reported at 1000.128, but this pairing isn't noted in the list.

Routers 1.181, 25000.65, and 26000.75 are all ports on the same router; you can tell this because they all have the same name—the serial number for the APT ComTalk. The Webster router, however, is listed only once for its Ethernet port. It also has four LocalTalk ports, connected to networks 32000,32001, 32002, and 32002.

This list is incomplete because RouterCheck uses an NBP LookUp for router ports on the LocalTalk side. If the router registers an NBP name on the LocalTalk side, RouterCheck lists it. If it doesn't, the port won't appear in the list.

When you run the RouterCheck's Missing Zones and Conflicting Zones routines, the Session Log lists the anomalies given in Figure 8-6.

You can tell from the Session Log that there's a zone naming problem, and zone names Ether 1, Ether 2, and Ether 3 are mentioned. Every router port that RouterCheck found is listed as having an anomaly. The on-line help system advises you to reconfigure or restart every router that shows an anomaly, but that seems like a lot of work, especially since you know that only one router has a problem. That's the nature of automatic comparison: it doesn't necessarily yield intelligent results that are easy to understand.

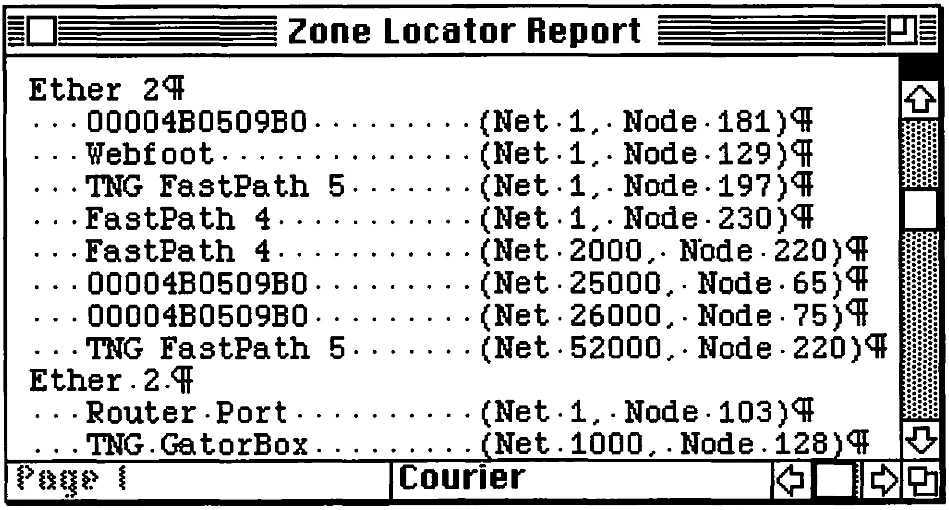


```
2:24:26 PM   – Checking routers for conflicting zones...
2:24:26 PM   • Conflicting zone name on net 1, node 181: Ether 1, should be Ethernet
2:24:26 PM   • Conflicting zone name on net 1, node 230: Ether 3, should be Ethernet
2:24:43 PM   – Checking routers for missing zones...
2:24:43 PM   • Missing zone on net 1, node 181: Ether 2
2:24:44 PM   • Missing zone on net 1, node 129: Ether 2
2:24:44 PM   • Missing zone on net 1, node 197: Ether 2
2:24:44 PM   • Missing zone on net 1, node 230: Ether 2
2:24:44 PM   • Missing zone on net 1, node 103: Ether 2
2:24:44 PM   • Missing zone on net 1000, node 128: Ether 2
2:24:44 PM   • Missing zone on net 2000, node 220: Ether 2
2:24:45 PM   • Missing zone on net 25000, node 65: Ether 2
2:24:45 PM   • Missing zone on net 26000, node 75: Ether 2
2:24:45 PM   • Missing zone on net 52000, node 200: Ether 2
```

Figure 8-6. RouterCheck's Session Log has determined that there is a zone name problem.

The best thing to do at this point is use the Zone Locator routine to make a list that you can compare yourself. It's a little more work than running the automatic comparison, but a person's ability to recognize patterns is more powerful than a computer's. For this particular error, however, RouterCheck's Zone Locator will be useful only if you first export the Zone Locator report to a text file and view it with a word processor. The reason for this is the same reason that you wouldn't be able to spot this error in the Chooser or Network Control Panel: the trailing space on the zone name isn't noticeable in any of RouterCheck's windows. You need to see the list with the spaces visible in order to find this particular error. If the error had been that one of the zones had Ether-2 instead of Ether 2, then it would have been possible to find it from one of RouterCheck's windows.

Figure 8-7 shows the portion of the Zone Locator Report (in Microsoft Word) that indicates the error. We've changed the text font to Courier, which is a fixed-space font. Fixed-space fonts are more useful when looking for spelling errors. Also, notice that the paragraph marks are left on as another visual aid.



```
Ether 2¶
···00004B0509B0·········(Net·1,·Node·181)¶
···Webfoot··············(Net·1,·Node·129)¶
···TNG FastPath 5·······(Net·1,·Node·197)¶
···FastPath 4···········(Net·1,·Node·230)¶
···FastPath 4···········(Net·2000,·Node·220)¶
···00004B0509B0·········(Net·25000,·Node·65)¶
···00004B0509B0·········(Net·26000,·Node·75)¶
···TNG FastPath 5·······(Net·52000,·Node·220)¶
Ether·2·¶
···Router·Port··········(Net·1,·Node·103)¶
···TNG·GatorBox·········(Net·1000,·Node·128)¶
```

Figure 8-7. Using Courier font and showing the spaces and paragraph marks, you can see that eight router ports report the zone name Ether 2 and two of the routers report the zone name Ether 2.

As mentioned previously, the two routers listed as having the extra space are actually different ports on the same router.

One special aspect of the above problem is that the symptoms are very subtle. Neither the Chooser nor Network Control Panel would show a problem because neither of those devices clearly indicates the extra spaces. Also, none of the routers have both Ether 2 and Ether 2, so the zone name won't appear twice in anyone's Chooser or Control Panel.

However, there will be symptoms, but only for users or services that have Ether 2 for their home zone. These users will occasionally get an error message during startup saying that their home zone is unavailable and they are being placed in the default zone. If such a user then used the Control Panel, he or she would be able to successfully reselect Ether 2 most of the time. Success would depend on which router was chosen to supply zone information, so results would be inconsistent. All other things being equal, the user would be successful 80 percent of the time, since 80 percent of the routers (4 out of 5) have the right information.

One class of network management tools that sometimes notices router configuration problems are those that build maps of the internet: for example, TechWorks' GraceLAN or Farallon's NetAtlas. Although not a fool-proof method, you can make a map of your internet and compare it to a map made previously. Besides looking for simple differences—such as new networks and zones—look for map features that make no sense. For example, you may see a router that appears as if it were two separate routers instead of one router with two ports. Another anomaly to look for is the note: THIS CABLE ALREADY MAPPED. This notation often indicates an unintended loop in the network structure.

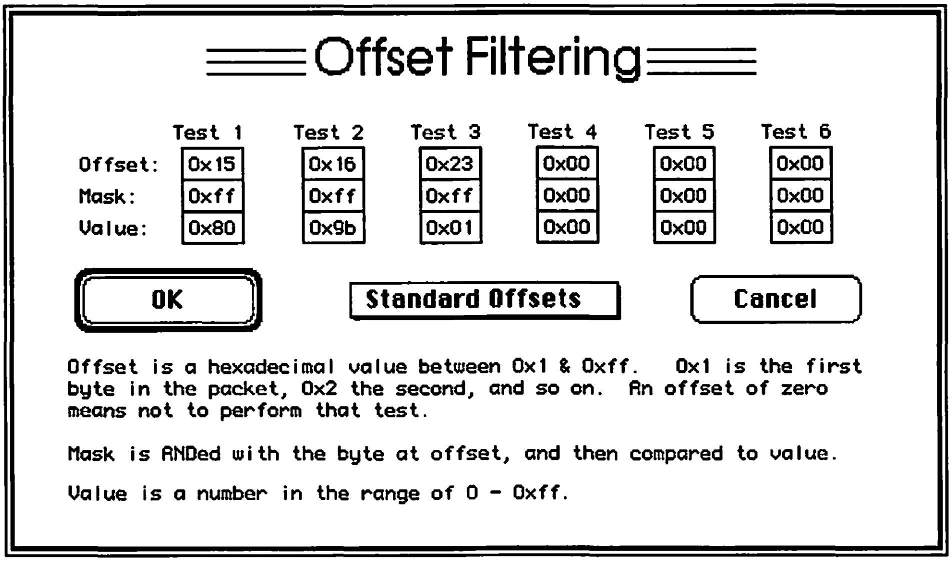
Even if you see an anomaly in NetAtlas, it doesn't necessarily mean that you have a router problem. For example, many legitimate router configurations can be represented strangely by NetAtlas. In some instances, NetAtlas's inability to accurately portray the network is related to weaknesses in the program's design. Other harmless anomalies in the NetAtlas map can be attributed to the lack of a standard, robust network management protocol. In lieu of a protocol, NetAtlas must make inferences from available information. Because different devices behave differently, NetAtlas sometimes makes incorrect inferences, resulting in an anomalies on the maps it draws.

As mentioned previously, packet analysis is your best bet. The procedures in this section are designed to help when you're troubleshooting a router problem, but it's probably wise to perform some of them once a month even if you don't suspect a problem. Some router problems can develop over long periods of time, slowly showing more and more symptoms as time passes. With preventive maintenance, you can often spot a router problem in an early stage and correct it before your users even notice the problem.

Counting the routers on your backbone network

If you can't diagnose the problem with RouterCheck or NetAtlas, or if your internet is so large that RouterCheck can't handle the amount of information, you'll have to use more manual methods to find your router problems.

When you begin your analysis, a good first step is to count how many routers are on your backbone network and compare that number to the number of routers you expect to see. To count the routers using a protocol analyzer, set your capture filters so that your protocol analyzer captures only RTMP Response packets (see Figure 8-8). RTMP Response packets are sent only by routers; every router should be sending one transmission every 10 seconds to every network to which it's connected. By examining the number of nodes sending packets that pass through this filter, you can determine how many routers are on your network.



Offset Filtering

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Offset:	0x15	0x16	0x23	0x00	0x00	0x00
Mask:	0xff	0xff	0xff	0x00	0x00	0x00
Value:	0x80	0x9b	0x01	0x00	0x00	0x00

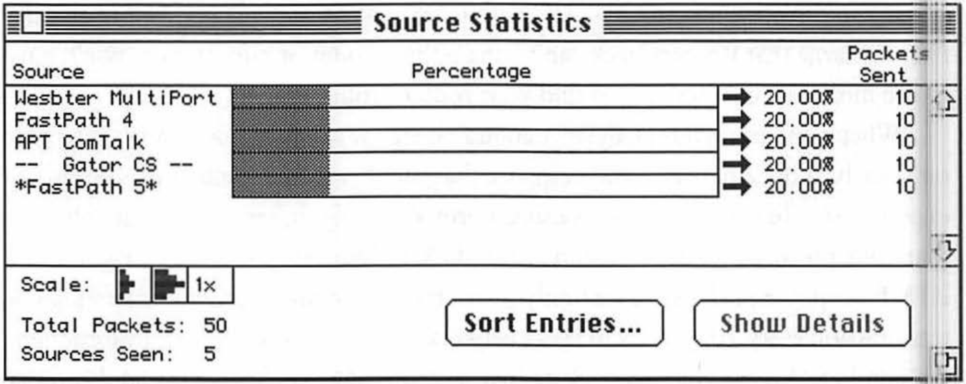
Offset is a hexadecimal value between 0x1 & 0xff. 0x1 is the first byte in the packet, 0x2 the second, and so on. An offset of zero means not to perform that test.

Mask is ANDed with the byte at offset, and then compared to value.

Value is a number in the range of 0 - 0xff.

Figure 8-8. EtherPeek's capture filter is set to accept AppleTalk Phase 2 RTMP packets. The first two tests establish that it is an AppleTalk Phase 2 packet. The third test establishes that the packet has 1 for its DDP type, which uniquely indicates RTMP Response packets. Filtering for RTMP packets is one of the standard filters available from the menu button in the center of the figure.

After you've let your protocol analyzer run for a minute or two, you can analyze the data. The first thing to look at is the number of source nodes in your packet trace. Figure 8-9 shows five routers on the network.



Source Statistics

Source	Percentage	Packets Sent
Wesbter MultiPort	20.00%	10
FastPath 4	20.00%	10
APT ComTalk	20.00%	10
-- Gator CS --	20.00%	10
FastPath 5	20.00%	10

Scale: 1x

Total Packets: 50

Sources Seen: 5

Figure 8-9. When you restrict the packets captured to RTMP Packets only, the source statistics show how many routers are on the network.

In Figure 8-9, you may notice that the name table of the protocol analyzer has a preassigned name for every router found on the network. If you take the time to build your name table, new routers are very easy to spot: they are listed in the statistics table with an Ethernet address instead of a name.

It's a good idea to sort this list by device name and print it when your network is healthy; then you'll have something to refer to when you're troubleshooting. If there are fewer routers on your network than you expected, you can consult your list to find which ones are missing and call the appropriate people to get them restarted (or find out why they've been removed from service).

If there are more routers on your backbone network than you expect, then your task is to find out where they are and who has placed them on the network. The following section deals with such a case, so if you found the correct number of routers and they all have names that are in your protocol analyzer's name table, you may want to skip this section for now.

Finding an unknown router

Sometimes a new router unexpectedly may appear on your network. The unknown router may or may not have incompatible routing and zone information, but under any circumstances, you'll want to discover the nature of the router. You'll also want to find out who put the router on the network so you can better coordinate future activities.

The first step is to determine which manufacturer made the Ethernet adapter (we'll assume it's an Ethernet network) for the unit. You can make this determination by comparing the first three bytes of the Ethernet address to the table of Ethernet addresses in Appendix B. If the manufacturer designation is 3Com, Asanté, TechWorks, or another manufacturer that makes Ethernet adapter cards but doesn't make a router, then you know that the router is actually a computer running router software. If the manufacturer is Cayman, Cisco, Wellfleet, or another manufacturer that makes routers but doesn't make Ethernet cards, that too is a clue. The third possibility is that the manufacturer is Shiva, Compatible Systems, or another manufacturer that makes both cards and routers. Knowing the adapter's manufacturer will help you determine where to look both on the network within the building. Also, if you have the router configuration software for that manufacturer, you may be able to correctly reconfigure the router.

If the router is also a computer workstation, then a network listing tool such as Inter•Poll may be able to learn enough about the workstation to locate it and the person responsible.

First, find out the AppleTalk address of the router. The RTMP packet will tell you the network and node address of the router that sent the packet. This information is found at the top of the RTMP header. In the example packet portion that follows, the router's address is 1.181.

```
RTMP Packet – Routing Table Maintenance Protocol
Router's Net: 1
ID Length: 8
Router's Node ID: 181
```

Next, use Inter•Poll or a similar network listing tool to scan the network and find out what Name Binding Protocol service names are associated with node 1.181.

If your backbone network has multiple zones assigned to it, you'll want to search all zones simultaneously. Tell Inter•Poll that you want to see the device list sorted by network number and you'll easily find the listing(s) for 1.181. In the best case, the name of one of the sockets listed for 1.181 will give you a clue that will lead you in the right direction. You may see, for example, that 1.181 has a Timbuktu socket with the name of Richard Sherburne or a QuickMail server with the name Room 204 Server. If you're not so lucky, then continue with the method described below.

If the Inter•Poll search of the backbone network does not help you, search the other zones served by the router for clues. First, examine the RTMP packets that the router is sending to find out what networks it's describing. Looking further into the packet, you can read the RTMP tuples to determine which networks are directly connected to the router. These directly connected networks will be listed with a hop distance of 0.

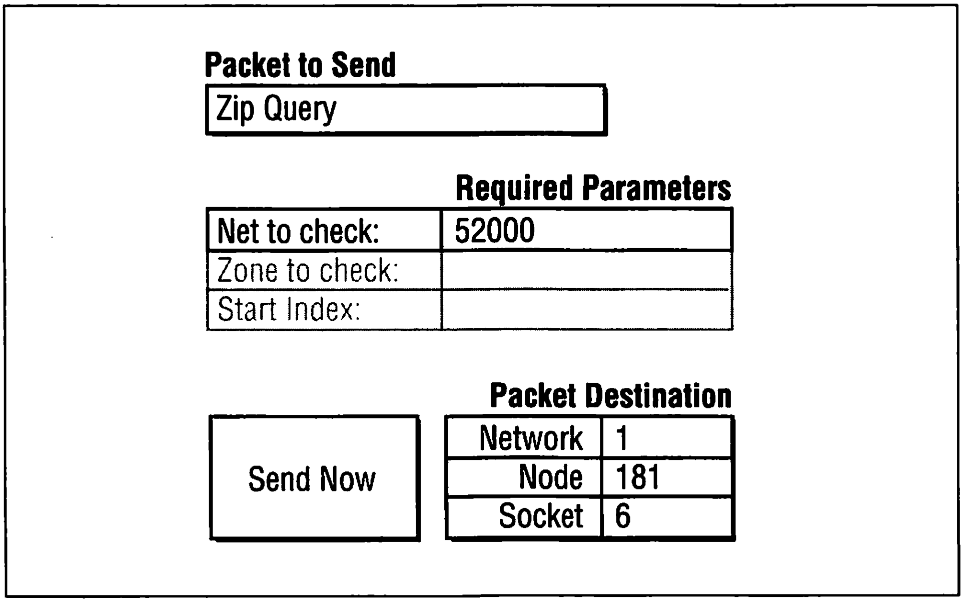
In the following packet, the router is indicating that it's directly connected to two networks: network 1–10 and network 52000. You can tell which of these was the network where we captured the packet by comparing the network addresses to the router's address. Since the router is node 1.181, this trace was obviously captured on network 1–10. The router has another port, directly connected to network 52000. In that network, another router is connected to network 52250. You know this because network 52250 is one hop away, and since this is Phase 2 (because tuple 1 describes an extended network), this router will describe only the network it's on and all the networks connected to (or through) its other ports. That's the nature of split horizon broadcasting, which is a feature of AppleTalk Phase 2.

```
RTMP Packet – Routing Table Maintenance Protocol
Router's Net: 1
ID Length: 8
Router's Node ID: 181

RTMP Tuple #1
Range Start: 1
Range Flag: %100 Extended
Distance: 0
Range End: 10
Version: 0x82
```

RTMP Tuple #2
Network Number: 52000
Range Flag: %000 Nonextended
Distance: 0

RTMP Tuple #3
Network Number: 52250
Range Flag: %000 Nonextended
Distance: 1

Next, use PacketSend, a shareware utility, and send a ZIP Query packet (see Figure 8-10) to node 1.181 to discover what zone name(s) goes with network 52000. Since this is a Ethernet/LocalTalk router, you already know that network 52000 is a LocalTalk network, and so it will have only one zone name.



Packet to Send
Zip Query

Required Parameters

Net to check:	52000
Zone to check:	
Start Index:	

Packet Destination

Network	1
Node	181
Socket	6

Figure 8-10. The ZIP Query packet will ask the unknown router what zone name is associated with the network the router 1.181 it is advertising, network 52000.

Once you've determined the zone name, you can use Inter•Poll to search for clues in the names of the services registered in network 52000.

In some cases, the person who installed the router may not have checked to see if the backbone network already had a number or range assigned to it. You may have assigned 1-10 as the network range of the backbone, and the person who installed the unknown router may have assigned 25-40 to the backbone network. In this case, the address range is completely incompatible with the official scheme; your routers may not even know about the unknown router (and vice versa). This poses a problem to an Inter•Poll-style search because Inter•Poll typically won't be able to reach the unknown router networks and zones.

A good technique for coping with this situation is to configure a spare router to be compatible with the unknown router and place it on your backbone network. In the example above, you would set up the spare router with a network range of 25-40 and connect it the backbone. This new router can recognize and exchange packets with the unknown router. Then you can connect the Mac running Inter•Poll to the LocalTalk side of this router and perform the search. Again, you're looking for clues in the names listed by Inter•Poll that will help you find the unknown router. In order to get the configuration to match, use PacketSend to discover the zone names assigned to the ports on the unknown router.

If Inter•Poll searching doesn't turn up a clue and you feel confident that you have the proper authority, you might try more invasive techniques to learn the identity of the router or the person responsible for it. Some of these techniques can border on maliciousness, so we suggest restraint. Use these techniques only as a last resort.

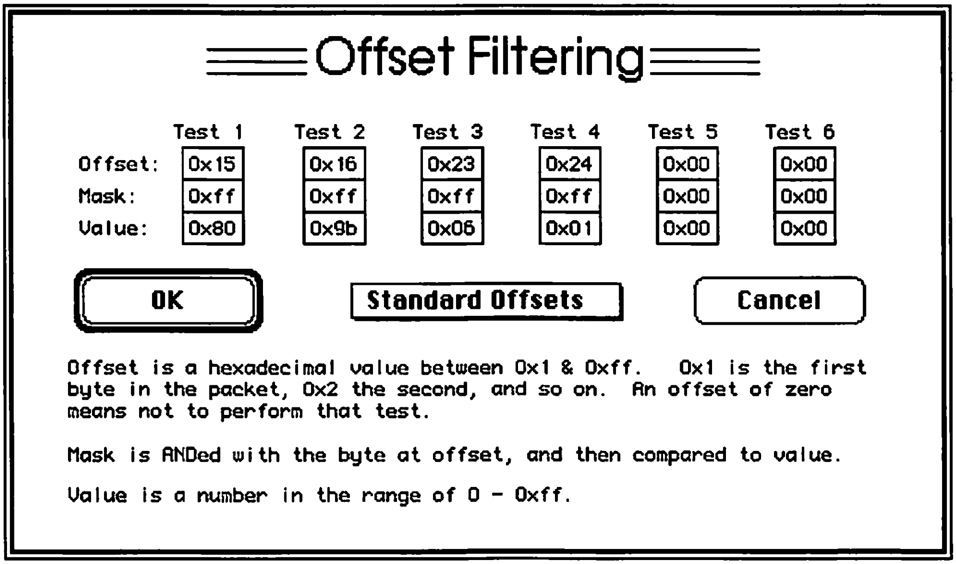
If your users are using QuickMail, for example, you can see the names of the registered users on the QuickMail server. And if they're using a QuickMail version earlier than v2.5, you can also send them messages. When you press the button on the QuickMail server that displays the names of the registered users, you can also see their passwords in clear text. You might consider logging into one of the servers and sending a message to the administrator to give you a call. You might try logging in to the AppleShare Server as a guest and creating a bunch of files named "Call John @ x2465" at the root directory to get users' attention.

Checking for ZIP Query Packets

After you've restarted any routers missing from the list and tracked down the unknown routers in the list of routers generated by filtering for RTMP packets (like we did in Figure 8-9), the next step in diagnosis is to look for ZIP Query packets. The presence of ZIP Query packets often indicates a router configuration error. ZIP

Query packets should appear only at router startup; if they show up at any other time, there's usually a problem, particularly if the ZIP Queries are sent repeatedly. (The absence of ZIP Queries, however, doesn't necessarily mean that you do *not* have a router problem.)

Set your filters to capture only ZIP Queries. The filter settings for ZIP Queries in EtherPeek are shown in Figure 8-11.



Offset Filtering

	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Offset:	0x15	0x16	0x23	0x24	0x00	0x00
Mask:	0xff	0xff	0xff	0xff	0x00	0x00
Value:	0x80	0x9b	0x06	0x01	0x00	0x00

Offset is a hexadecimal value between 0x1 & 0xff. 0x1 is the first byte in the packet, 0x2 the second, and so on. An offset of zero means not to perform that test.

Mask is ANDed with the byte at offset, and then compared to value.

Value is a number in the range of 0 - 0xff.

Figure 8-11. Offset filters in EtherPeek for ZIP Query packets. Test 3 checks for DDP type 6 (ZIP), Test 4 checks for ZIP function 1 (Query).

If you see ZIP Query Packets and you can't attribute them to a router startup, one or more of the routers probably has a problem. For example, if you forget to enter the zone name for a LocalTalk network when you configure a router, the other routers can't have a proper zone name for that network in their Zone Information Table. These other routers will periodically ask the improperly configured router (using ZIP Query) for the name of that zone. They should repeat the question approximately once every 10 seconds. The improperly configured router won't respond because it has no answer to give.

ZIP Queries and Replies indicate a problem, but a problem for which of the routers? In general, if repetitive Query packets are answered by valid Reply packets (Reply packets containing valid network and zone information) the router sending the Queries is probably the one with the problem. If the ZIP Replies don't contain valid information or there are no Replies, chances are good that the router receiving the ZIP Queries is the router with the problem. In the case above, for example, the Query packets were not being answered, and the router receiving the Queries had the configuration problem.

This first characteristic—whether the Queries are being answered with valid information—is the best indicator of which router is having the problem. A second, less conclusive, indicator is whether all of the routers on the network are sending ZIP Queries or just a router subgroup.

If all the routers are sending Queries, the problem is most likely in the router receiving the Queries. In the case above, all other routers on the network would be sending ZIP Queries to the one misconfigured router. This fact points to the conclusion that the router receiving the Queries is the one having the problem.

If a group of routers (but not all of the routers on the network) is sending the Queries, try to determine whether the subgroup has any identifying characteristics. For example, the AppleTalk Router built into NetWare 2.11 has a very short aging timer for Network Information. If only NetWare routers are sending the ZIP Queries, the router receiving the Queries may have gone down momentarily. The downtime may have been long enough for the NetWare routers to age out their zone information (and require a new ZIP Query), but not long enough for the rest of the routers on the network to age out their zone information.

In this case, the NetWare router would send ZIP Queries to the reappearing router, but the rest of the routers might not. You sometimes see this behavior when a half-router link goes down momentarily-the telephone line goes dead, and the half-router stops advertising the remote network in its RTMP packets. Moments later, the line is restored and the RTMP listing resumes. Depending on the time needed to restore the line, the routers with the shortest aging timers might send ZIP Queries and those with longer aging timers might not.

Another example of ZIP Query Packets indicating a problem is the pattern that you see when a router's Zone Information Table (ZIT) becomes corrupted. The corruption of a table can happen for a variety of reasons, but a corrupted router may send ZIP Queries to one or more of the other routers on the network in an attempt to bring order to its ZIT. In this case, it will usually receive valid ZIP Replies. If it repeatedly sends Queries to the same routers asking about the same networks, then you know that the router has a problem from which it can't recover. The best thing to do in this case is to turn the power off on the router, leave it off for 30 seconds, and turn it back on. This will cause the router to build its Routing and Zone Tables over from scratch. In extreme cases, the router may have corrupted its non-volatile memory. To cure this malady, remove the router's battery for 10 minutes, restart the router, and reload the router's software and configuration files.

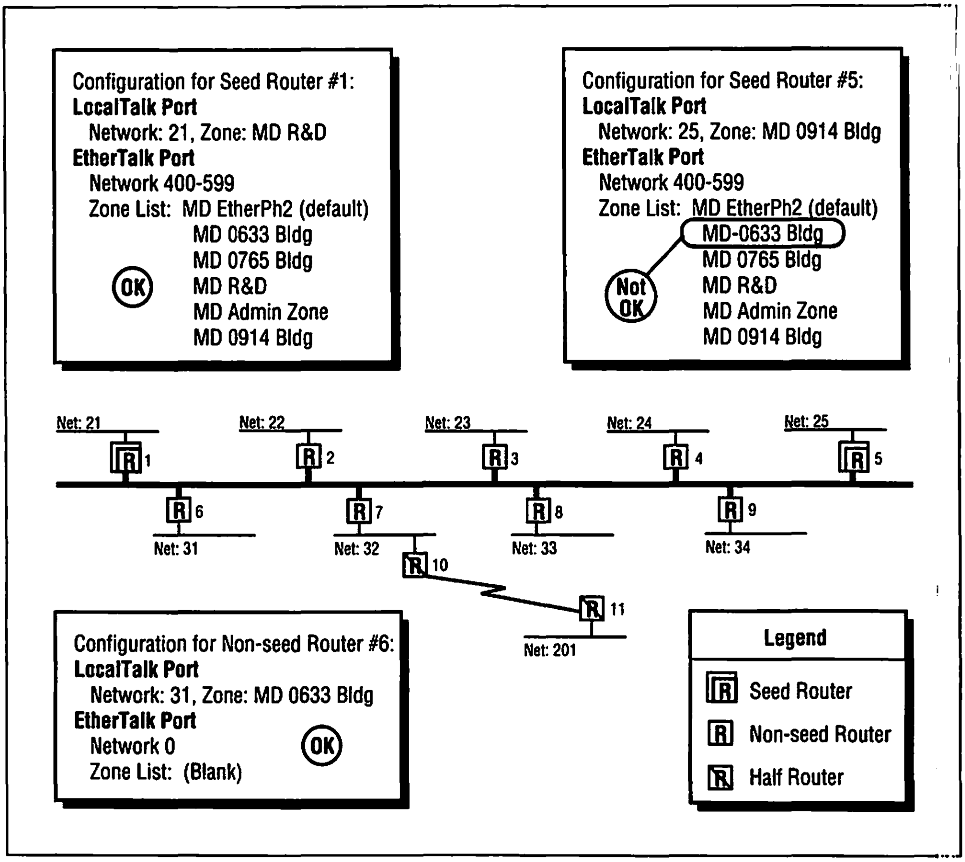
There's one other possible interpretation for the situation above: your network numbers might be in conflict. Different manufacturer's routers handle this situation differently, but it may be that while your backbone has the network range of 1–10, someone may have inadvertently assigned a network address within that range—8, for example—to a LocalTalk network elsewhere in the internet. That would also cause some routers to send this periodic ZIP Query. If the reset advised above doesn't cure the problem, you may have to look thoroughly at all the network numbers.

Mistyped zone names

While some kinds of router problems are accompanied by ZIP Queries, other router problems are not accompanied by obvious error signatures. This is true of the misconfigured network shown in Figure 8-12. The symptom in this network is not obvious and does not appear consistently.

This network has an Ethernet backbone with nine LocalTalk-EtherTalk routers; two are configured as seed routers and the other seven as non-seed routers. Six zone names are assigned to the Ethernet, and some zones on Ethernet also encompass zones on LocalTalk networks. An example of this is Seed Router #1. The LocalTalk Port is assigned a zone name that is also a valid Ethernet zone. One of the LocalTalk networks has a half router (non-seed) to a LocalTalk network at a remote site.

Let's examine how this problem spreads. Suppose that the incorrect seed router was the most recently configured; that means that every router but Seed Router #5 (SR5) has the correct information. A couple of weeks later, the building that contains Non-seed Router #9 (R9) experiences a power failure. When power is restored, the router boots up and takes its Ethernet configuration from an Ethernet router already in service. There are eight other Ethernet routers, all of which supply seed information, and the new, non-seed router randomly selects one of the eight to "believe." (Non-seed routers already in service propagate network and zone information the same as actual seed routers.) It has a 1-in-8 chance of select-



```
Configuration for Seed Router #1:
LocalTalk Port
  Network: 21, Zone: MD R&D
EtherTalk Port
  Network 400-599
  Zone List: MD EtherPh2 (default)
             MD 0633 Bldg
             MD 0765 Bldg
        OK   MD R&D
             MD Admin Zone
             MD 0914 Bldg
```

```
Configuration for Seed Router #5:
LocalTalk Port
  Network: 25, Zone: MD 0914 Bldg
EtherTalk Port
  Network 400-599
  Zone List: MD EtherPh2 (default)
             MD-0633 Bldg
             MD 0765 Bldg
      Not    MD R&D
      OK     MD Admin Zone
             MD 0914 Bldg
```

```
Configuration for Non-seed Router #6:
LocalTalk Port
  Network: 31, Zone: MD 0633 Bldg
EtherTalk Port
  Network 0                  OK
  Zone List: (Blank)
```

Figure 8-12. This internet has a problem with its zone names. One of its seed routers was configured incorrectly when a technician inadvertently typed a hyphen into one the zone names.

ing SR5 to supply it with information, and if it does, R9 will get the bad Ethernet zone list.

A few weeks later, when the telephone technician accidentally unplugs R3 from its power outlet and then plugs it back in, it also will have the chance of getting the bad zone list. As you can see, this problem may spread very slowly at first, and then propagate more quickly as the number of "bad routers" increases.

This type of problem will be noticed intermittently, if at all. We'll summarize two of the irregularities that would occur in this network. In describing these symptoms, we'll assume that the problem hasn't spread beyond the one incorrectly configured router.

Irregularity #1—Chooser Zone List. Users in LocalTalk Network #25 would see a zone name of MD 0633 as well as MD-0633, because R5 knows about both zones. For R5, the zone name with the hyphen would refer to the Ethernet; the zone name without the hyphen would refer to LocalTalk Network #31. Users in network #25 would be able to print or use any of the network services in network #31, but only if they choose the zone name without the hyphen. (This problem is even more confusing to resolve when the only difference between zone names is that one has an extra space at the end.)

Users in the other LocalTalk networks would see only the proper zone name—the name without the hyphen—which would refer collectively to devices on the Ethernet in zone MD 0633 Bldg as well as to the devices in Network 31. They would be able to use devices in any LocalTalk network as well as any device on the Ethernet in any zone, including MD 0633 Bldg.

On Ethernet, users' workstations usually display a zone list with the correct name, but once in a while a user will see both names in the Chooser. When a Mac on Ethernet opens the Chooser, it will pick whichever router it heard from most recently to supply the zone list. It will send a ZIP GetZoneList to this router and will have a 1-in-9 chance that the Chooser will pick R5, which carries both zones in its zone list.

Irregularity #2—Selecting a Home Zone on Ethernet. On Ethernet, a user may occasionally report a problem when selecting the problem zone name (of either variety). This symptom appears because of a checking mechanism built in to the process. When we perform our diagnostic tests, we'll use a modified form of this checking method to find the bad routers.

When selecting a home zone in the Network Control Panel, the user sees a list of available Ethernet zones. The Control Panel obtains this information by using the ZIP GetLocalZones packet to ask the router from which it most recently received a packet. On this Ethernet, the same 1-in-9 chance applies as above.

When the user selects his or her home zone, the Macintosh confirms the validity of the zone name by sending a ZIP GetNetInfo packet to the most recently heard router, which may very well be a different router than the one that supplied the local zone list moments ago. Here's where the user may experience the anomaly.

Imagine the following scenario:

1. The Mac picked a "good" router for the local zone list.
2. The user selected MD 0633 Bldg for the home zone.
3. The Mac picked the "bad" router, R5, for confirmation of the zone name MD 0633 Bldg.

Result: The user sees a dialog box saying that zone MD 0633 Bldg is not a valid zone name for this Ethernet and that the user will be placed in the default zone name MD EtherPh2.

R5 has to report an invalid zone because it believes that MD 0633 Bldg refers only to LocalTalk Network #31 and is not a zone on Ethernet. It supplies the default Ethernet zone name of MD EtherPh2 and the network range of 400–599.

You'll need to identify the group of routers that have the incorrect zone name, including both the seed router that started the error and the non-seed routers that picked up the erroneous name. After you identify these routers, you'll need to reconfigure the erroneous seed routers first, then reset all the non-seed routers to restore consensus.

Checking your backbone network's routers

The scenario described above illustrates how a data entry error can introduce a bad zone name into the internet, how the bad zone name can spread from router to router, and what symptoms can be noticed. But it's only one scenario. Even if you don't suspect a problem with zone names and you're not experiencing any symptoms, it's still a good idea to check your backbone routers periodically.

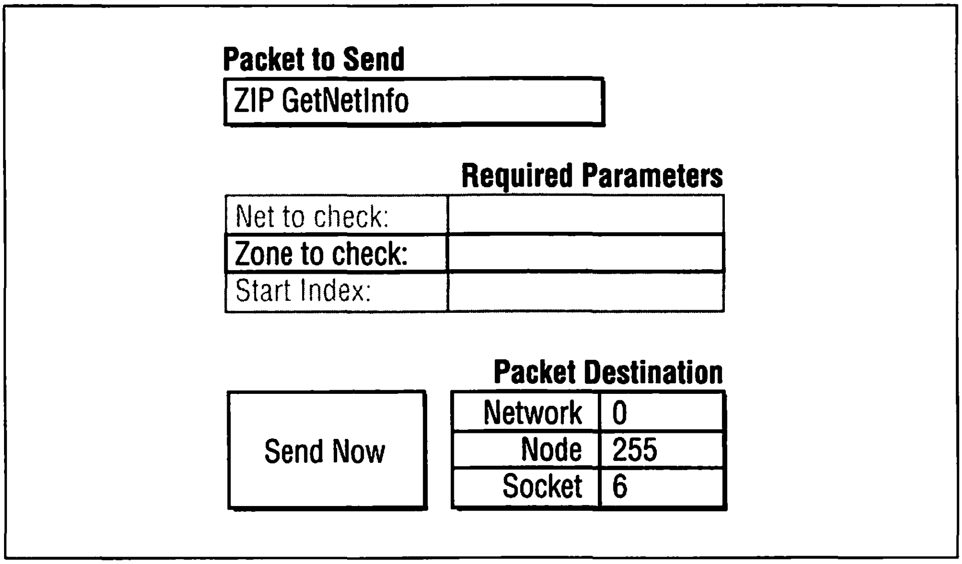
Earlier, we used RouterCheck to automatically check for anomalies; now we'll check the routers with PacketSend. The testing is far more laborious than using RouterCheck, but the labor required for 100 routers is no more than required for two. Also, you can use PacketSend on even the largest internets.

It's best to perform these tests from a backbone network-a central network that connects other networks together through routers. We'll assume that the test to determine the number of routers on the backbone network, outlined above, has already been performed and has yielded the expected number of familiar routers.

On an extended backbone network (remember that extended networks can have more than one zone name), the first task is to verify that each router on the backbone has the same zone and network data for the backbone network itself. Then we'll look outward to the other networks. In the sample tests, we'll be using only two routers—a FastPath 5 and a Gator CS—and in both cases the FastPath 5 will have the wrong information. (This is no reflection on the FastPath 5; we flipped a coin to determine which one would be the "wrong" router.)

To verify that all routers hold the same information for the backbone network, we need to verify that each router on the backbone network has the same network range assigned to the backbone network, the same number of zone names for the backbone, and the identical zone names assigned to the backbone.

We'll perform the three verifications in order. To make sure that each router has the same network address range, we'll broadcast a ZIP GetNetInfo (GNI) packet to all the routers (see Figure 8-13). Normally, non-routing nodes send the ZIP GetNetInfo to a particular router to verify a zone name, but the GetNetInfo Reply (GNIR) that the router returns also includes the network address range, so it's useful as a verification of that network address range.



Packet to Send

ZIP GetNetInfo

Required Parameters

Net to check:	
Zone to check:	
Start Index:	

Packet Destination

Network	0
Node	255
Socket	6

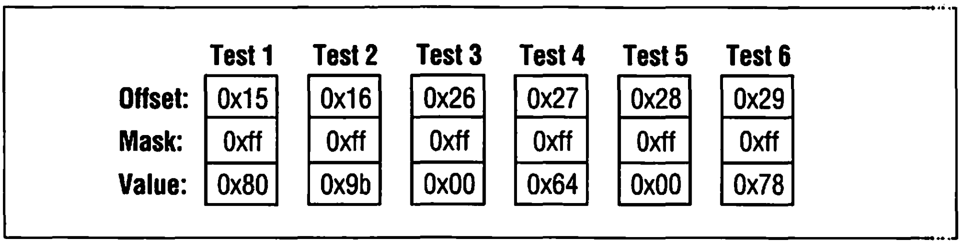
Figure 8-13. Broadcasting a GetNetInfo packet with the zone name left blank ensures that the network address range is in the same position in each GetNetInfo Reply packet.

When using the ZIP GNI to confirm the network address range, leave the zone name blank. That way, each router's reply will tell you that the zone name you asked about is invalid, and the network address range will be in the same offset position in every packet.

When you broadcast this packet, you should get a response from every router on the backbone. Since you've already counted them, you should know how many routers should be sending replies. If you're on a busy network, use address filtering to capture only the packets going to and from the node sending the GNI.

Once you've captured the GNI Replies, you can use your filters again to highlight only the replies that contain the correct network range. It's easiest if your protocol analyzer lets you filter on a text string; you can enter only the network range, but you'll have to enter the range in hex. For example, if the network range you expect is 100–120 ($0064–$0078), select the packets that have the hexadecimal text string 00640078. Remember that network numbers are two bytes long, so you'll have to enter the extra 0's.

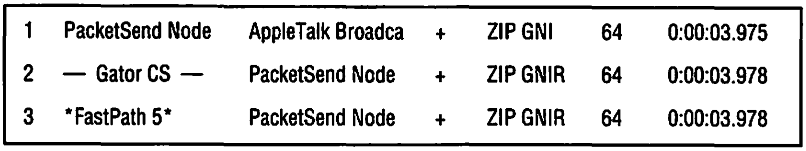
If you used the blank zone name, you can also use your offset filters. The four bytes of the network range occupy the four consecutive byte locations starting with offset $26 (counting from the beginning of the Ethernet destination address). For the network range of 100 to 120, your offset filters would look like Figure 8-14.



	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Offset:	0x15	0x16	0x26	0x27	0x28	0x29
Mask:	0xff	0xff	0xff	0xff	0xff	0xff
Value:	0x80	0x9b	0x00	0x64	0x00	0x78

Figure 8-14. Offset filters to select GNI replies with the network range 100 to 120.

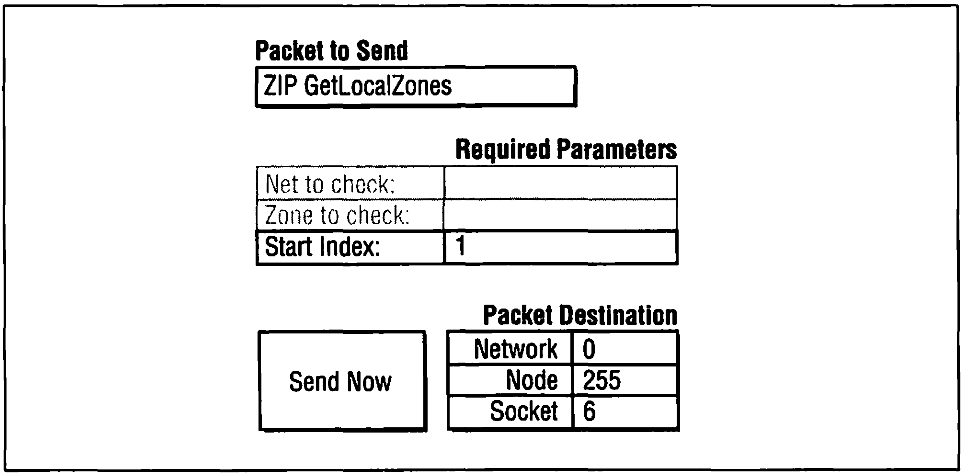
Regardless of your filter method, the result should be that the routers with the correct network range will be highlighted and the routers with a different network range will not (see Figure 8-15). If your protocol analyzer simply clears the packets instead of highlighting them, you'll need to reverse the criteria; have the protocol analyzer clear all of the replies that have the correct range and retain the replies that have an incorrect range.



1	PacketSend Node	AppleTalk Broadca	+	ZIP GNI	64	0:00:03.975
2	— Gator CS —	PacketSend Node	+	ZIP GNIR	64	0:00:03.978
3	*FastPath 5*	PacketSend Node	+	ZIP GNIR	64	0:00:03.978

Figure 8-15. After filtering, the selected packets will show which routers have the correct network range.

The next step in verifying the backbone routers is to make sure that each router has the same number of zones assigned to the backbone cable. For that, we'll broadcast the GetLocalZones (GLZ) packet (see Figure 8-16). If you have only a few zone names assigned to the backbone, you can leave the start index at 1. The start index tells the router which zone in its list to begin with in the GLZ Reply. There's a limit to how many zone name characters can fit into one packet, so if you have a lot of zones assigned to the backbone or if you have long zone names, you may want to set the start index to a number slightly less than the number of zones you expect. For reference purposes, if your zone names are 20 characters long, you can have 27 zones listed in one packet.



Packet to Send

ZIP GetLocalZones

Required Parameters

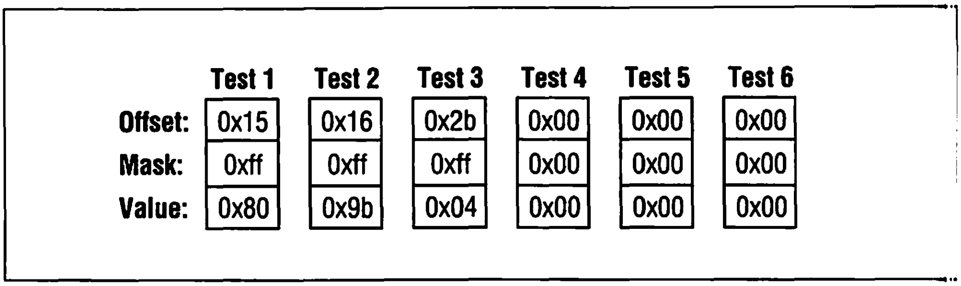
Net to check:	
Zone to check:	
Start Index:	1

Packet Destination

Network	0
Node	255
Socket	6

Figure 8-16. To check the number of zones assigned to the backbone, send the GetLocalZones packet.

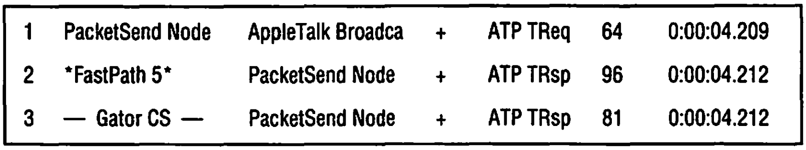
Once the GLZ Replies are captured, we'll verify that each router has the same number of zones assigned to the backbone by filtering on the byte (offset $2b) in the GLZ Reply that tells how many zones are listed. The filter criterion for this is in Figure 8-17. The filter is set for a reply with four zone names.



	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6
Offset:	0x15	0x16	0x2b	0x00	0x00	0x00
Mask:	0xff	0xff	0xff	0x00	0x00	0x00
Value:	0x80	0x9b	0x04	0x00	0x00	0x00

Figure 8-17. The filter checks byte $2b to see that there are four zones listed in the GLZ Reply.

After filtering, the routers with the correct number of zones will be highlighted, and the routers without the correct number will not be, as shown in Figure 8-18.



1	PacketSend Node	AppleTalk Broadca	+	ATP TReq	64	0:00:04.209
2	*FastPath 5*	PacketSend Node	+	ATP TRsp	96	0:00:04.212
3	— Gator CS —	PacketSend Node	+	ATP TRsp	81	0:00:04.212

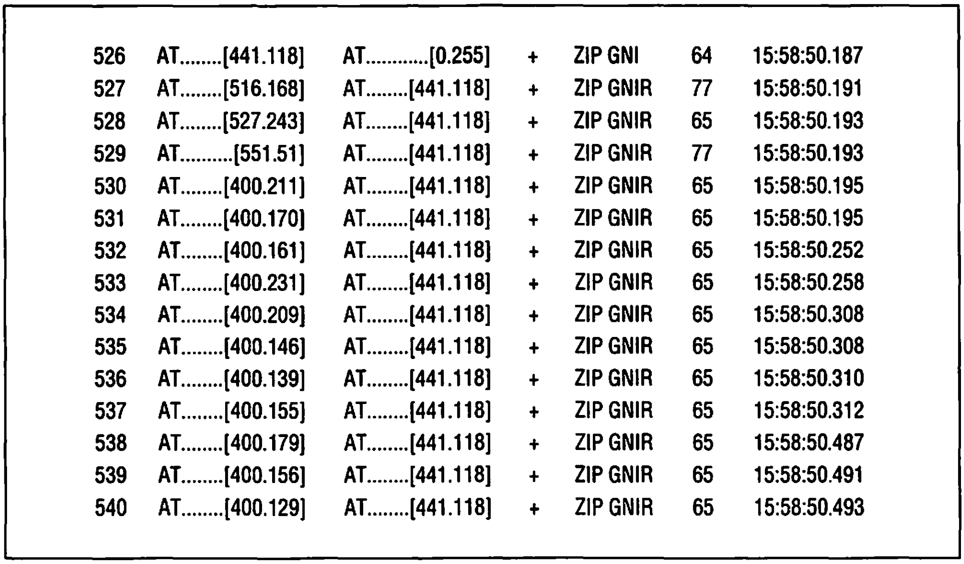
Figure 8-18. After filtering, the highlighted packets will show which packets have the correct number of zones.

You may notice that the FastPath's GLZ Reply (carried in an ATP Response packet) has a different number of bytes as well as a different number of zones. It's generally not helpful to use the packet length as the criterion for correctness, even though on this network any correct reply would be 81 bytes long. The packet length isn't useful for two reasons. First, it's possible that a router with an incorrect number of zones could have the same length as a reply with the correct number of zones. Second, if you're using a start index higher than 1, the routers won't necessarily list their zones in the same order. If there are 80 zones and you send a GLZ with a start index of 76, each router should reply with five zones (also listing their names), but each router will probably list a different group of five zone names. Thus, all the Reply packets might have the correct number of zones, but different lengths. Therefore, the only reliable test is to check for the number of zones contained in the packet using the offset filters, as shown in Figure 8-18.

Now that you've verified that each router has the same network range and the same number of zone names (and reconfigured the routers found to be in error), it's time to check the zone names one by one. You'll use the ZIP GNI again, but now you'll type in the name of each valid zone name for the cable, send the GNI, type in the next valid zone name, send it, and so forth.

Again, you should get a GNI Reply from every router to each of the GNIs you send. If you have a large, congested network, it would probably be best to send each GNI twice to make sure that you get a reply from every router. The replies should fall in blocks, one for each zone name, which can be distinguished by their having the same packet length. In this test, packet length is the best criterion because it's fast and reliable.

Let's look at a block of replies to a GNI that asks whether MD 0633 Bldg is a valid zone name. The GNI Replies in Figure 8-19 are 65 bytes long, with the exception of two routers that sent 77-byte replies. These two routers both reported that MD 0633 Bldg was not a valid zone name. These routers must be reconfigured if they're seed routers and restarted if they're non-seed routers, after all other configuration problems have been eliminated.



```
526  AT........[441.118]  AT............[0.255]  +  ZIP GNI   64  15:58:50.187
527  AT........[516.168]  AT........[441.118]    +  ZIP GNIR  77  15:58:50.191
528  AT........[527.243]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.193
529  AT..........[551.51]  AT........[441.118]   +  ZIP GNIR  77  15:58:50.193
530  AT........[400.211]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.195
531  AT........[400.170]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.195
532  AT........[400.161]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.252
533  AT........[400.231]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.258
534  AT........[400.209]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.308
535  AT........[400.146]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.308
536  AT........[400.139]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.310
537  AT........[400.155]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.312
538  AT........[400.179]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.487
539  AT........[400.156]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.491
540  AT........[400.129]  AT........[441.118]    +  ZIP GNIR  65  15:58:50.493
```

Figure 8-19. Routers that do not have the valid zone name will have a slightly longer packet.

Analyzing Network Traffic

We want to start off this section by saying that traffic analysis is very difficult to do well, particularly because the available tools to analyze traffic in AppleTalk are completely inadequate. While there are ways to accomplish the task, they are very tedious to perform. The guidelines for traffic analysis in this section are mostly given in case an adequate tool will someday be available.

What analysis of traffic is needed?

The purpose of traffic analysis is to understand the flow of traffic in a network. Once you understand the traffic flow on your network, you'll be better able to provide data for network design and resource planning. Some of the aspects of your network that you may want to quantify are:

- *Who are the power users of my network? What services are they using on the network and why are they using them?*

It's useful to understand what power network users do with the network because they are sometimes doing what many other users will begin to do in the near future. Sometimes you may want to restrain power users or suggest alternate ways for them to accomplish their computing goals, but in general, you need to understand the nature of power usage when planning for the future.

- *Who uses the network the least? Why are they using the network so little? Do they lack training? Do they lack a need for network services?*

You should consult these users as well. If they don't need the network services, are the services configured to provide the most useful service possible? Perhaps these users have a better way of accomplishing the same business functions.

- *What is the level of traffic and how does it change over time?*

You may notice that the level of traffic is going up from month to month. If you extrapolate this trend into the future, when will you have to change your network design due to traffic congestion?

- *Are there any times of the day (or week, month, or quarter) when the usage of any particular network resource is predictably high?*

If so, you may want to evaluate your resources in terms of these times or consider ways to reduce the usage. For example, if everyone needs to print enormous documents the last two days of the month, you may think about alternatives to the rush for the printer during those workdays. Perhaps you could contract the printing out to an agency or hire a temporary worker to come in evenings and print the documents.

You may notice fairly predictable cycles that will influence how you schedule network maintenance. For example, if you notice a tremendous amount of network activity on Thursday in the accounting department because of payroll activities, perhaps Thursday evening is a good time to schedule a full backup of the accounting server.

- *Do any users consistently use resources in other networks? Is their traffic going to just one other network or to several different networks? Would such users be better off on one of those other networks? Or would the service be better off if placed elsewhere?*

Remember, routers should be used to segregate traffic, not funnel it.

The inadequacy of the current generation of tools

The above questions—particularly the last one—expose a glaring weakness in the current development of AppleTalk network management tools. At the time of this writing, there is no tool that helps you comprehensively characterize AppleTalk network traffic. Some products are cumbersome, some don't run well on anything less than a 68030 Mac II, and some have major bugs in both design and implementation. Some are also prone to giving false error statistics, as discussed in Chapter 6, "Data Link Layer Troubleshooting." Moreover, if a product doesn't record the number of bytes sent between devices, telling you only the number of packets (which is meaningless unless you know the size of the packets), you have no useful information with which to characterize your traffic patterns.

Even if a product did all these things, it still couldn't tell you about traffic to other networks, since it looks only at the LAP portion of the packet. It could tell you that many packets were sent to the router, but not where the packets went beyond that. TrafficWatch II is an ambitious program, and the windows are fun to watch, but it's not very useful unless you want to pad reports to make them seem more impressive and well-researched. (In this capacity, we used it for this purpose once and have to admit that TrafficWatch really shines.)

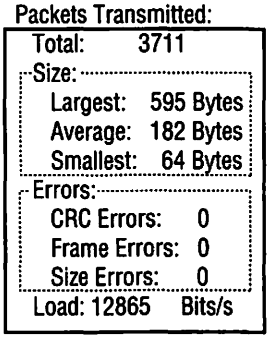
NetStats is infinitely more useful as a troubleshooting tool, but mostly because it dares to be simple in its approach. It gives you an instant look at the traffic level of a LocalTalk network—no frills like analysis of the traffic. You can't save it, you can't print it; all you can do is look at it (and do a screen capture). It tells you only what the LocalTalk utilization is right *now*.

On some servers the log file provides some good information. For example, the log from Apple's AppleShare Print Spooler gives you all the information you need to completely analyze your printing traffic if users always print using the spooler.

Traffic analysis with protocol analyzers

A protocol analyzer can also help you determine some of the questions posed above. The analysis is not necessarily easy to perform, however, because protocol analyzers are designed not to analyze traffic patterns, but to analyze the *content* of the traffic. Although the method is tedious, you can use some features in EtherPeek to figure out what percentage of your traffic is going to and from which networks and devices for any particular user.

First, capture the traffic coming from and going to a particular user and save it in a file all by itself. (Include only those packets that originate from or are destined for this user.) For example, suppose we let the protocol analyzer run for seven minutes and collect packets to and from a user named Mark. After stopping the capture and saving the file, you can open the Source Statistics window to get a breakdown of how many packets Mark sent to each node that he communicated with. Although it's helpful to know the identity of the nodes that Mark communicated with, the numerical information in this window isn't particularly useful for that purpose, since it gives only the number of packets (not bytes), just like TrafficWatch data. The interesting numbers in this window are in the summary section, where you can see the total number of packets that Mark sent and the average number of



```
Packets Transmitted:
 Total:      3711
 Size:
   Largest:   595 Bytes
   Average:   182 Bytes
   Smallest:   64 Bytes
 Errors:
   CRC Errors:     0
   Frame Errors:   0
   Size Errors:    0
 Load: 12865    Bits/s
```

Figure 8-20. With this summary information, you can calculate out how many bytes that a particular user sent during the time the data was captured.

bytes in those packets. Multiply those two numbers together to get the total number of bytes sent out by Mark.

With the summary information shown in Figure 8-20, you can calculate that this node transmitted a total of 3711 packets with an average size of 182 bytes, for a total of 675,402 bytes during the capture period of seven minutes. This works out to 1608 bytes per second (12,865 bits per second) or 0.13 percent of the nominal Ethernet bandwidth of 10 million bits per second.

Remember, that's only the calculation for Mark's outgoing packets. Figuring out the statistics for the incoming packets is somewhat harder, because you have to get the statistics for each of Mark's partner nodes separately.

Documenting a Network for Troubleshooting

As we've stated before, the essence of troubleshooting is comparing what is to what should be. The first criteria, therefore, is that troubleshooting documentation should provide a catalogue of what should be. Further, the usefulness of the documentation will depend on how clear, detailed, and up-to-date it is.

Each of the five modes of troubleshooting has a different orientation to the concept of what should be, and each mode has different documentation needs. This subsection is organized according to those five modes, and the documentation needs will be described for each mode.

Random mode documentation

In Random mode troubleshooting, the concept of what should be is a set of user expectations. You begin troubleshooting when you or the user notices something peculiar in the way a particular system is working. You try various measures to cure the problem and when the problem is gone, you are finished. You may never find out whether a network, hardware, or software problem caused the symptom.

Before you can notice something peculiar in the way a system behaves, you must have an expectation of what normal operation is. For example, normal operation for a particular Macintosh in your network might be eight zones listed in the Chooser, four LaserWriters in the Southern Ops zone, about 70 seconds to complete its startup cycle, about three minutes to print a particular file on its hard disk, and about one minute to download a particular file from the file server. In small internets, some of these expectations will not need documentation because you can remember them. The network manager of this small internet would know exactly how many LaserWriters and NetModems were installed in each zone.

In a larger internet, however, it is wise to make a list of the devices in each zone. You can make this list very easily with a network listing device such as Inter•Poll or CheckNET and save a printout of it for future reference. Other programs, such as the AG Group's NetWatchMan and Caravelle's NetWORKS, can not only build a list of the devices in each zone, they also have the ability to periodically check that each device is still reachable through the network and can report the status of the device on an on-going basis, visually with a graphical representation of the zone, with audible noises, and with a log entry. These programs simulate a user trying to reach these services, monitoring the success rate the user would have. NetWatchMan adds to this the ability to watch for new zones or the disappearance of existing zones, and NetWORKS has the additional ability to signal your beeper or send mail messages to the appropriate people.

To obtain network performance expectations, an easy method is to create and keep files that serve as benchmarks. These are files you printed or transferred when your network was healthy, noting the time the printing or file transfer took to complete. For printing, your benchmark file should be at least six pages long and contain a representative sampling of the types of graphics that your organization uses. For file transfer, the file should be at least 500K in size. A good file to use is one created in a large application that doesn't get upgraded very often, like Excel—the size of such a file typically does not change unless it is upgraded.

If you don't have previous test data that tells you what a "normal" time should be, sometimes you can estimate how long something should take. For example, suppose you were transferring a 500K file, and it took 19 seconds to transfer. Is that normal? Let's look at a way to perform this estimation.

In explaining the concepts of bandwidth and utilization, we developed a number for the bandwidth of LocalTalk: under the very best conditions, LocalTalk can carry about 20 Kbytes of information per second. On LocalTalk, a reasonable estimate is that a file transfer will proceed at a transfer rate between 40 percent to 80 percent of the network bandwidth, depending on what types of devices and transfer applications are involved. With these guidelines, you can estimate a range for a "normal" file transfer. At 40 percent utilization (5K/sec), the file transfer would take 100 seconds, at 80 percent it would take half that time, 50 seconds. "Normal," therefore, for a 500K file would be between 50 and 100 seconds. Clearly, 19 seconds is outside the normal range.

For printing, we usually take the printer manufacturer's claims (8 pages per minute, for example) and multiply it by 75 percent for straight text with few or no font changes in the document, 50 percent for text and graphics, and 25 percent for a document that is mostly graphics, then add 30 seconds. For example, we would estimate that a 16-page document of text and graphics on an 8-page per minute printer would print at 4 pages per minute. Adding 30 seconds, we would estimate the normal time to be 4½ minutes and wouldn't be suspicious of a printing time between 3 and 7 minutes.

Documentation for hunch-based troubleshooting

In hunch-based troubleshooting, you use your prior experience to form hunches about the cause of the problem and make adjustments to the system until the problem goes away. You examine the problem's "qualities" and explore its boundaries—where and when it happens with which software and systems—and make intelligent guesses about which system variables to manipulate. As in random-mode troubleshooting, you're finished when the problem is gone and you may not know what the problem was, but you'll probably have some guesses because you used your previous experience as the basis for the fix.

Two documents are required for hunch-based troubleshooting. First, the network management staff should keep an activity log that records what problems have been encountered and what procedures have been performed. Because the log compiles information over a length of time, the information recorded should also be useful in helping you see the "big picture." This can help you address more global concerns than a simple break in the cable, such as user training and references, network administration policy, and resource planning.

At minimum, the log should tell which device or network component had a problem, the time and date that it occurred, what the user was trying to accomplish, a "snapshot" of the device's hardware and software configuration when the problem occurred, the user's experience level (beginner, intermediate, advanced), and a record of the steps taken to fix the problem. The log should also record all system modifications such as software upgrades, hardware modifications and repairs, etc.

You may notice trends in the log. For example, you may notice that a particular kind of problem happens only to users of a certain level or that a certain problem happens only on Monday mornings. Keeping a log is always beneficial, but it's even more so when network management functions are performed by two or more people.

Log books are only useful when they are used, of course, so you should structure the logging system so that it will actually be used by the troubleshooters. A log entry should be relatively fast and easy to make, and the log book should be easily accessible to all. How you will physically establish a logging system will depend on the characteristics of your network—its geographic size, the number of troubleshooters on the staff, etc. For example, if you are geographically dispersed, you may find it convenient to establish an electronic mailbox and a custom mail form for the purpose of reporting network problems, or you may want to have a word processing template and electronically mail the completed forms to an administrator who compiles them into a database that is accessible to all.

If the entire network management staff works out of a single office, the electronic mail scheme outlined above would seem cumbersome, and perhaps simply keeping a notebook would be more appropriate. The point is that the logging system should be designed in such a way that everyone finds it convenient to add to include the information base on nearly every action taken.

The stated purpose of the log book is to record experience in a permanent form. This has two immediate benefits. The most obvious is that experience gained by one person can be used by another. The second benefit is that log books are often useful for building claims and justifications. If your wiring system is bad but management thinks it is too expensive to fix the problem by rewiring, your log book can play a crucial role in building the justification for the critical task of resource allocation. Telling your manager, "In the last 6 months, 154 trouble calls attributable to wiring problems were logged. In these trouble calls, 85 maintenance hours were consumed and 210 hours of user downtime was suffered," is a lot more convincing than, "Gee, boss, the other guys and I spend most of our time putting out all these little fires with our wiring system. It seems like we're always chasing halfway across the building because of some stupid wiring problem." Of course, this works in reverse as well. You may think that rewiring is necessary, but the log book may show that the time and energy put into fixing wiring problems would not justify the cost.

If you keep a detailed log, you might also notice patterns in the problems that you encounter, which would lead to changes in network or system management policy. You might find, for example, that every single problem with Application A is accompanied by Screen Saver B. From your experience, you might be able to discern "rules" of system configuration from the continuity of experience documented in the log.

The second crucial document of hunch-based troubleshooting can be called by many names: Escalation Chart, Tip Sheet, and Help Card are just a few. This is the document that should be given to every user to explain how to solve problems. Before distributing this, you should seriously consider how much latitude to give your users to solve problems. If you give them too little, you will unnecessarily burden the troubleshooting staff and create excessive downtime due to the waiting period between the time the help call is made and the time you arrive on the scene. If you give users too much latitude, their lack of experience might prompt them to try some wildly inappropriate corrective measures that could cause further problems. The best approach is to provide them with a help document that explicitly describes solution procedures for the most typical network problems.

The network managers must draw the line somewhere between the users changing the cross-connects in the wiring closet and changing fonts in their own documents.

For example, the document might say:

> **Printing Problems:**
>
> If you are having problems printing, *before* you call the Help Desk, perform these steps:
>
> 1. Go through the procedure on Page 4 of this document, titled "General Network Problems."
>
> 2. Make sure that your LaserWriter is on, it has a PhoneNET connector plugged into it, and it has paper in the tray.

3. If the LaserWriter display panel says "Off Line", press the "Line" button once. The display panel should now say, "Idle–On Line."

4. If the LaserWriter display panel says "Paper Jam," clear the jam and press the "Clear" button. The display panel should now say, "Idle–On Line."

5. Check your Chooser once again. If you cannot locate the LaserWriter in your Chooser, call the Help Desk.

6. If you *can* locate the LaserWriter, close the Chooser and attempt to print. If you cannot print, call the Help Desk.

Documentation for setup-mode troubleshooting

The main document for setup-mode troubleshooting is a network map. Setup-mode troubleshooting involves verifying the connections between the components, and the network map should define the expected connections. The best map for troubleshooting is different from the floor plan that you usually see, however. While a floor plan is beneficial for setup-mode troubleshooting because it tells you where the computers are situated and which users use the computers, you also need information about which pair of which circuit is serving the user, which port of which hub the user is connected to, and what kind of network adapter the user has.

There are three points of view to take when generating the map that specifies the connections between the network components. The first point of view is the user's. In this case, all you need to do is to augment the information on the traditional floor plan diagram with circuit, pair, port, and adapter information.

A listing might look like this:

Sarah Connor
Mac IIsi/80/5
LocalTalk Net 43, Zone: Skynet
Circuit 22L4, Pair 8 (Terminated)
Hub 6, Port 5

In the map, a line might be drawn to indicate the Sarah is also connected to a nearby LaserWriter II NT.

The second point of view is the wiring technician's. The map in this case consists of a list of all the pairs on all the circuits, stating the connection that is on each pair. All of the circuits in the phone room are first identified by the location of their wall outlet. This should be done using whatever method your company uses to identify location, grid location, room designation, etc. In many cases, a particular worker's name can also be linked to the wall outlet. After identifying the wall outlet, each of the pairs in the circuit are identified. If the pair is used for a phone, the extension number is listed. If the pair is used for a network adapter, the type of adapter is listed, and/or the port to which it is connected.

From the wiring technician's point of view, the sample listing above might look like this:

Circuit 22L4—Sarah Connor
Pair 1—Telephone ex: 2274
Pair 2—Telephone ex: 2285
Pair 3—
Pair 4—
Pair 5—
Pair 6—
Pair 7—
Pair 8—LocalTalk–Hub 5, Port 6

The third point of view is that of the hub. For each hub, each of the ports are listed, along with the circuit and devices on each port:

LocalTalk Hub 5, Port 6
22L4/8 Sarah Connor Mac IIsi/80/5—LaserWriter II NT

The ideal mapping system should be able to show you the network map in any of these three views, but few of us are willing or able to make a system that complete. The most important feature of the mapping system is not that it have a multiplicity of display modes, but that it is always up-to-date. Troubleshooting from out-of-date information can be very frustrating.

You may have purchased mapping software or have already developed extensive maps, and you may not be able to change your choice of map. You will probably be able to augment your current system to include the three viewpoints. If you can't change your system or augment it, then make a second, simple system. A very simple way to record the information you need for troubleshooting is to have a clipboard in the wiring closet for each hub. The clipboard contains a chart of all of the hubs ports, with spaces for notes about what's connected to the port—circuits, pairs, and users. When a wiring change is made, the technician can easily note the change by erasing the previous setting and penciling in the new condition. This system is simple, cheap, and places the mapping system's data entry inside the wiring closet, the place where the network structure is usually changed.

Documentation for layer-mode troubleshooting

Layer-mode troubleshooting uses a variety of network management programs to accomplish its tasks—list builders like Inter•Poll, survey builders like NetWork Super-Visor, configuration analyzers like Help!, etc. The network management tools survey or measure the availability, configuration, and reliability of the network functions, layer by layer. The documentation necessary in this mode is previously captured data from each of the management programs when run on healthy networks and systems. Capturing this documentation serves two purposes: it requires that the network manager gain familiarity with the features, idiosyncrasies, and limitations of each management applications; and it produces the "normal" data needed for comparison.

Documentation for process-mode troubleshooting

The primary tool for process analysis is the protocol analyzer. Anyone who has taught protocol analysis to hundreds of people as Kurt VanderSluis has, knows that learning to analyze network processes takes time.

The approach that we use to help network managers learn protocol analysis is to capture the packets from small network events, such as selecting a LaserWriter. Then we examine the packets one by one and become familiar with the workings of the protocols in each packet. Starting with the most elementary processes, we slowly build our understanding of increasingly larger and larger network processes until we can dissect a complex process like printing. All the while, we are developing expectations of what types of packets are produced in response to various kinds of user and system events. For each packet that we see in the trace, we try to discern what the device sending the packet is trying to learn or tell the other device. By developing an understanding of the tiny task of each individual packet, we see how the packets work together to accomplish a small task like establishing a network address or locating a named network service. You can use this same learning process on your own network by using a protocol analyzer and studying *Inside AppleTalk*, alone or combined with instruction. You may also find the process descriptions in Chapter 7 useful as a study aid. This learning process requires hard work, especially if AppleTalk is your first network protocol, but mastering AppleTalk protocols is definitely worth the effort. Protocol analysis is absolutely the single most effective technique in network troubleshooting.

The point of the learning process is to build an idea of what to expect in the network process. These expectations are the basis of process-mode troubleshooting—the "normal" reference point from which to compare the condition of the problem network.

In lieu of, or in addition to, establishing a mental set of expectations, you should capture and keep packet traces for ordinary network events. Not only will they help you learn the protocols when you sift through the traces, they will be an excellent reference for troubleshooting. When a printing problem occurs, you can pull out your printing reference trace and compare it side-by-side with the problem trace. When you find the place where the traces diverge, you have found the location of the problem.

Overall, your documentation should be easily accessible, up-to-date, and detailed enough to be helpful, but not so minutely detailed as to discourage people from using it.

The basic documentation set required for troubleshooting is:

1. A record of "normal network" expectations from a user's point of view.
2. A record of previous experiences and procedures.
3. A guide for your users to solve some of their own problems.
4. A network map that includes wiring information.
5. A record of the data from your network management applications.
6. Packet traces from normal network processes.

Building Your Tool Kit

This section focuses on the software tools that you need for troubleshooting. You probably have all of the hardware tools you need—all the crimpers, strippers, and screwdrivers that are required for your daily tasks. We're also assuming that you have all the configuration management software required for your routers and servers. If you use a GatorBox, you obviously need (and should have) GatorKeeper software. The information in this section is supplemental to other information in the book, particularly to Chapter 6, "Troubleshooting by the Layers," which mentions a number of network management tools and their usefulness for troubleshooting.

The organization of this section is similar to what you might see in a mountaineering guide. Some of these guides list the minimum equipment for different kinds of expeditions. For example, if you intend to go rock scrambling or "bouldering," there are just a few necessary items that you should carry with you—a knife, drinking water, a first aid kit, matches, and so on. If you intend to do any technical climbing, you should add other items to the basic kit—a rope, some carabineers, and a seat harness, to name a few. On glaciers, there are further requirements—crampons, an ice ax, and dark goggles, for example.

None of the lists in the guidebooks are complete; you can always take anything else that you like—a flask, your favorite bandanna, or a copy of *The Dubliners*—but only in addition to the basic necessities.

It's harder to prepare for network troubleshooting than mountaineering, however, because you don't always know what kind of terrain you're going to be working on. In mountaineering, when you reach a steep wall you aren't prepared for, you may have to call it quits and head back for camp. You have to tell yourself that the scenery, the camaraderie and the exercise make up for the fact that you didn't achieve your goal of reaching the top. Calling it quits in troubleshooting due to lack of preparation is much more frustrating, especially when there are users standing around asking, "Is it a serious problem?", or "When do you think you'll have it working again?"

With this in mind, if you can afford to buy all of the network management and troubleshooting software, then go ahead and buy it. All of it's useful for something, and because AppleTalk network management is still in its infancy (as is all of LAN management), different programs offer advantages in different situations. Also, something that is not appealing to us may very well be appealing to you. We have participated in group discussions where the merits of GraceLAN, Network SuperVisor, Status*Mac, and NetOctopus have been discussed, and we have listened to impassioned opinions on the clear superiority of each of these products from managers who have tried them all.

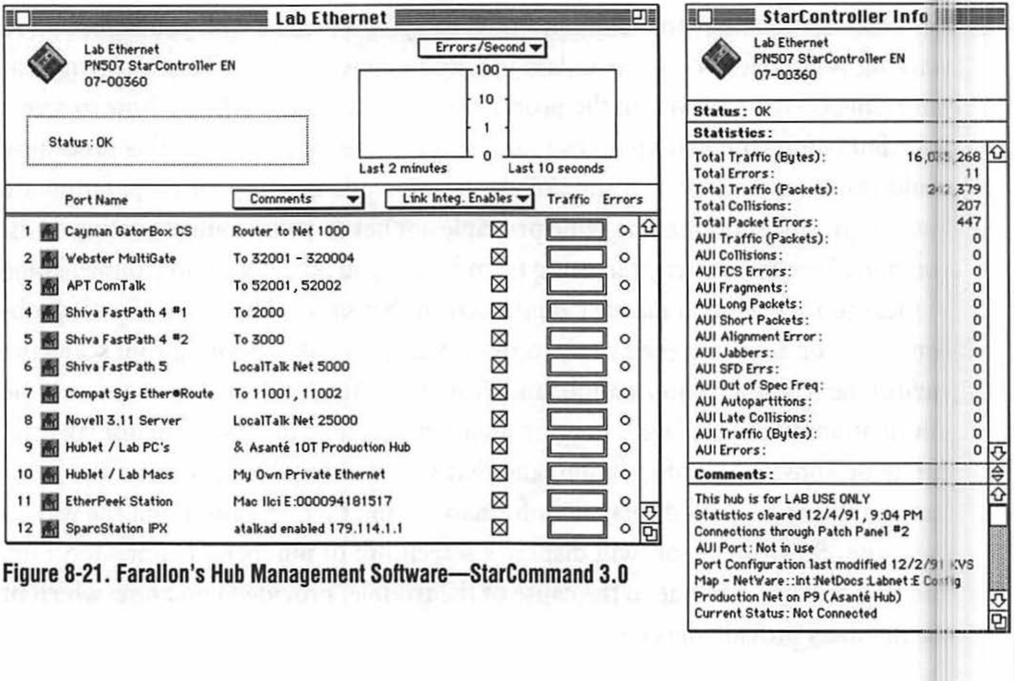
One more analogy to mountaineering: carrying an ice ax with you while you're walking is completely useless unless you know how to use it. That's why mountaineering classes drill you in the proper use of the ax and teach you how to arrest your fall before you end up in a situation where your ignorance of this procedure could result injury or even death. While the consequences of poor preparation are lower in troubleshooting, the same principle applies to your troubleshooting tools; you must become proficient in using them before you actually get in trouble. None of these tools produce a message on the screen that says, "The cause of your problem is...," or anything even close to that. A good troubleshooting tool scans the environment, gathers information, and displays it. Your job is to make sense of the information that is displayed. In error situations, some of the tools will display inaccurate or nonsensical information, and that will be your clue as to what the problem is. Other tools will display no information, and that will also point the way to the cause. Still other tools will display a screen full of numbers that are accurate, and that will be your clue to the cause of the trouble, provided you know which of the numbers provide the clues.

The basic tool kit

The basic tool kit includes:

- Your network hub's management software
- Apple's ancient, but still useful, Inter•Poll
- Timbuktu (or Carbon Copy)

Your hub's wiring management software is crucial, provided it can give you the kind of information you need about the health of the wire on a port. Most LocalTalk hubs should be able to tell you the names, or at least the node addresses, of the devices that are connected to each port (see Figure 8-21). That lets you verify the connectivity on a port by port basis. A good Ethernet hub will tell you the link status of each of its ports, also a basic piece of troubleshooting information. The hub statistics are also valuable when they can give you a clue as to what sort of problem—jabbering, termination, bad wiring—may be on that port. Some hub software goes beyond this, of course, and may alert you when it perceives a problem.



Lab Ethernet

Lab Ethernet
PN507 StarController EN
07-00360

Status: OK

	Port Name	Comments	Link Integ. Enables	Traffic	Errors
1	Cayman GatorBox CS	Router to Net 1000	☒		○
2	Webster MultiGate	To 32001 - 320004	☒		○
3	APT ComTalk	To 52001, 52002	☒		○
4	Shiva FastPath 4 #1	To 2000	☒		○
5	Shiva FastPath 4 #2	To 3000	☒		○
6	Shiva FastPath 5	LocalTalk Net 5000	☒		○
7	Compat Sys Ether•Route	To 11001, 11002	☒		○
8	Novell 3.11 Server	LocalTalk Net 25000	☒		○
9	Hublet / Lab PC's	& Asanté 10T Production Hub	☒		○
10	Hublet / Lab Macs	My Own Private Ethernet	☒		○
11	EtherPeek Station	Mac IIci E:000094181517	☒		○
12	SparcStation IPX	atalkad enabled 179.114.1.1	☒		○

StarController Info

Lab Ethernet
PN507 StarController EN
07-00360

Status: OK

Statistics:

Total Traffic (Bytes):	16,0[illegible]268
Total Errors:	11
Total Traffic (Packets):	2[illegible]379
Total Collisions:	207
Total Packet Errors:	447
AUI Traffic (Packets):	0
AUI Collisions:	0
AUI FCS Errors:	0
AUI Fragments:	0
AUI Long Packets:	0
AUI Short Packets:	0
AUI Alignment Error:	0
AUI Jabbers:	0
AUI SFD Errs:	0
AUI Out of Spec Freq:	0
AUI Autopartitions:	0
AUI Late Collisions:	0
AUI Traffic (Bytes):	0
AUI Errors:	0

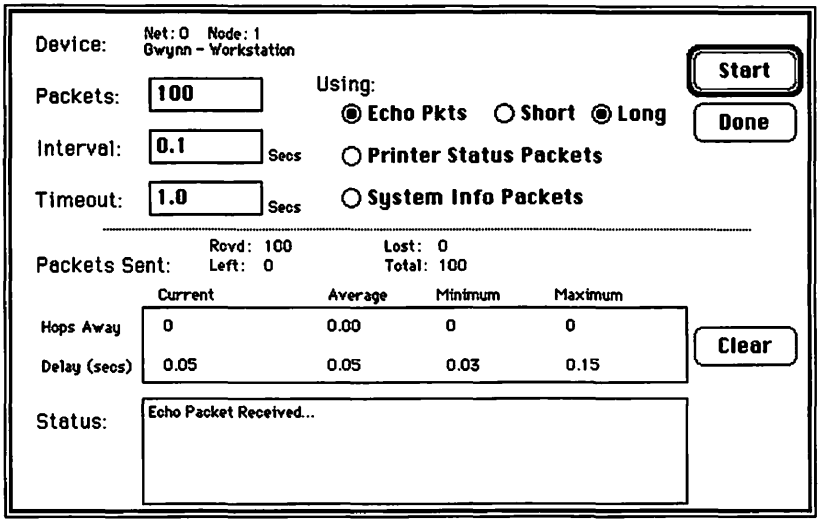
Comments:

This hub is for LAB USE ONLY
Statistics cleared 12/4/91, 9:04 PM
Connections through Patch Panel #2
AUI Port: Not in use
Port Configuration last modified 12/2/9[illegible] KVS
Map - NetWare::Int:NetDocs:Labnet:E Co[illegible]ig
Production Net on P9 (Asanté Hub)
Current Status: Not Connected

Figure 8-21. Farallon's Hub Management Software—StarCommand 3.0

Inter•Poll (see Figure 8-22) can quickly check the health of any network or zone in your internet. Despite the fact that it's old and it beeps at you when you start it up, it's still better at what it does than anything else. Inter•Poll's best three features are that it tells you how many zones are in your internet, lists the devices in a zone, and performs the echo test (described in "Using the Echo Test," found in Chapter 6).

Timbuktu (or Carbon Copy) is a great troubleshooting tool because it makes you more effective from more physical locations. We don't recommend that you purchase a copy for every device in your network unless your users have other uses for screen-sharing software, but you should have a copy loaded on all of your servers and routers that run on Macintoshes as well as on the network management staff's Macs. With this software, if you learn of a problem while you're away from your desk, you can often cope with it from where you are. With the programs beginning to support Microsoft Windows, these screen sharers will be even more useful for troubleshooting when you need to administer a Windows-based service or use a Windows-based troubleshooting tool. On this note, we expect to see a Windows-based SNMP console before we see a Mac-based console.



Device: Net: 0 Node: 1
Gwynn - Workstation

Packets: 100

Interval: 0.1 Secs

Timeout: 1.0 Secs

Using:
● Echo Pkts ○ Short ● Long
○ Printer Status Packets
○ System Info Packets

Packets Sent: Rcvd: 100 Left: 0 Lost: 0 Total: 100

	Current	Average	Minimum	Maximum
Hops Away	0	0.00	0	0
Delay (secs)	0.05	0.05	0.03	0.15

Status: Echo Packet Received...

Figure 8-22. Apple Computer's Inter•Poll—Echo Test

We also suggest that you carry disks with you that contain your favorite troubleshooting tools, including Inter•Poll and Timbuktu. A spare copy of Timbuktu is also useful for transferring other software over the network, so you might want to carry a few extra serial numbers in your wallet or day planner.

The intermediate tool kit

At the intermediate level, there are three types of network management tools that are especially useful:

1. A traffic monitor, such as NetStats
2. A network alert system, such as NetWatchMan or NetWORKS
3. A system management package, such as GraceLAN or Network SuperVisor

Two examples of traffic monitors are made by Farallon: TrafficWatch and NetStats. Of the two, NetStats is more useful for troubleshooting purposes. TrafficWatch, used thoughtfully, can be useful for other network management functions, like resource allocation and planning. Except for its ability to count LocalTalk CRC errors, how-

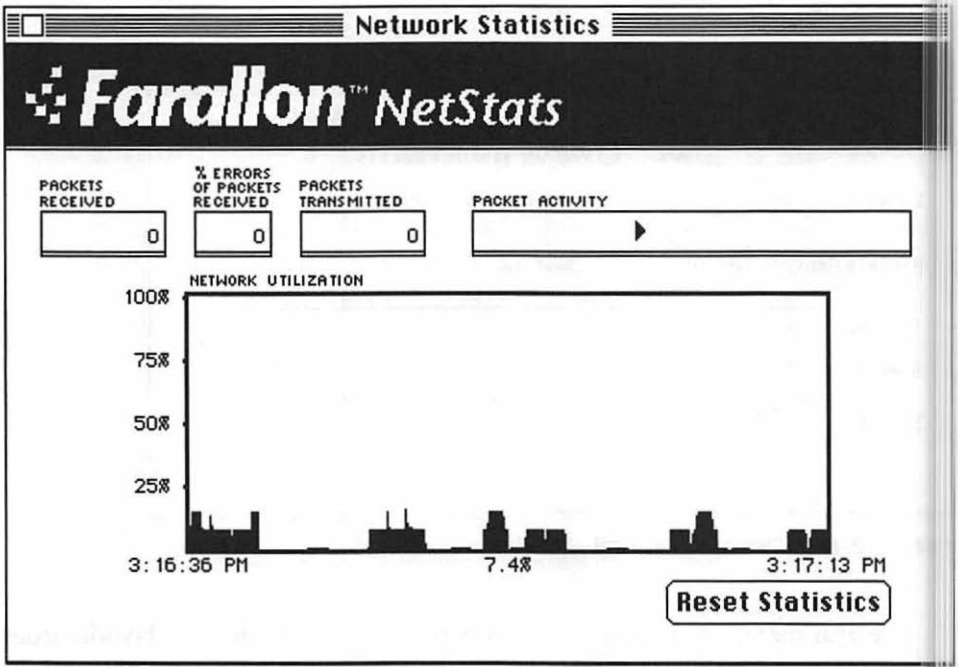


Figure 8-23. Farallon's NetStats

ever, TrafficWatch is not especially useful because its time scale is not fine enough for most troubleshooting purposes.

In the Network Utilization window, the finest resolution TrafficWatch can show is five seconds. NetStats (see Figure 8-23), which unfortunately works only on LocalTalk networks, samples 1,000 times per second and can show utilization values every ⅒th of a second. That's the kind of resolution you need for troubleshooting. You need to know if there's anything on the wire, and packets are on the wire only for a very short time.

The traffic level you see in NetStats can tell you if the LocalTalk network is busy and can show you the shape of the traffic utilization. After some experimentation and observation, you can use NetStats to determine different kinds of traffic on the network. The utilization profile of printing looks very different from the profile of file transfer or electronic mail. When you get to this level of familiarity, you'll also have an insight into the nature of those network processes useful for troubleshooting. For example, if user A is printing to LaserWriter A, and User B is printing to LaserWriter B, will they slow each other down by jamming up the network with traffic? How about when four users are printing to four LaserWriters? The answer to both of these questions is no, and this will be quite apparent after you become familiar with NetStats. This kind of familiarity is extraordinarily useful when you're making guesses about the nature of a problem.

Insights aside, sometimes you just want to answer the question, "Is the network busy right now?" On LocalTalk, NetStats will answer that question very quickly.

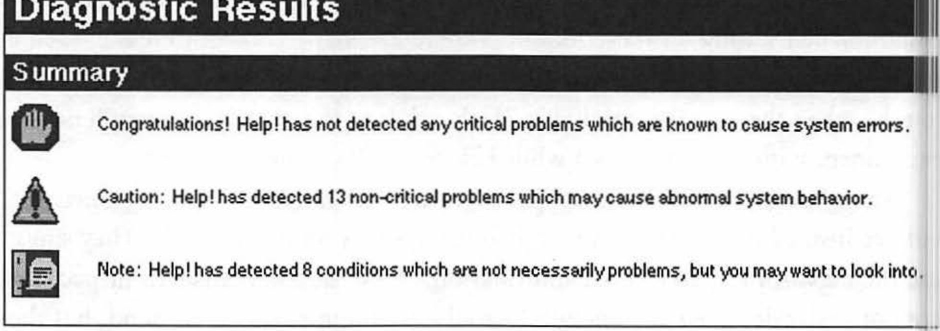
On Ethernet, traffic monitors that have a fine enough time scale for troubleshooting are very expensive. They are a part of DOS-based protocol analyzers like the Sniffer or LANalyzer and are found in the hub management software of the more expensive Ethernet hubs. In these devices, there is special hardware and firmware that is used to capture and simultaneously analyze the traffic at a fast enough rate to provide the necessary time resolution.

All traffic monitors have one basic limitation, however. They can tell you only about the traffic on the network to which they are attached. One nice feature of some hub management software is that you can view the traffic from more than one hub (and thus more than one network) simultaneously. Some even have alarms that you can set to go off when the traffic rises above a certain level.

Another type of intermediate level management program alerts you to network problems. With the right settings, you can sometimes even learn of network problems before your users notice them. The prime examples of this type of software for AppleTalk networks is the AG Group's NetWatchMan and Caravelle's NetWORKS. Both of these products watch named services through the internet and report on their reachability according to time intervals you can set. This can be very helpful, because if a service or network becomes unreachable, you can know before your phone begins to ring. NetWORKS can even call your beeper or send you e-mail. When setting up these monitors there are many choices to make, such as which services and zones to monitor, how often to check them, and what actions to take when they are not reachable. With either of these packages, you'll need to experiment with these options a while before you'll set them properly.

Some "network management" packages are not really for managing a network, but are instead for managing the individual systems on the network. They simply use the network to gather their information. These system management packages are not really designed with network troubleshooting purposes in mind, but they are useful nonetheless because of their ability to report information on the hardware, software, and peripherals used by the devices in your network. There is no clear winner in this category, but the players (for troubleshooting purposes) are CSG's Network SuperVisor, TechWork's GraceLAN, and MacVONK's Net Octopus because they can gather and display information in real time. ON Technology's Status*Mac may be the best tool in its class for systems administration aimed at preventing problems, but it is nearly useless for troubleshooting because it does not gather the information in real time. A wonderful feature for these programs would be the ability to perform automatic comparisons between two configurations and simply give you the results comparing, for example, two similar devices, all the devices within a class, a device and its standard configuration, or a device as it's configured today and the way it was configured at a previous time. These comparisons would be very useful for troubleshooting because they could help you answer such questions as "Why are these three devices behaving normally, while this fourth one of the same type is having trouble?", or "This device was working perfectly last Friday. Why is it not working today?" Unfortunately, while all of these comparisons are useful for analysis, they currently must be performed by the "software" in your brain. With today's software, you can use the system management packages only to gather the information, not to analyze it.

The only exception to this rule is Teknosys' Help! (see Figure 8-24), an analysis package that will analyze a single Macintosh's configuration for thousands of known configuration problems. Help! currently has no ability to work over the network or with data gathered by the other system management packages, but that will probably change before too long. For now, you'll have to run this on each Macintosh individually.



Diagnostic Results

Summary

Congratulations! Help! has not detected any critical problems which are known to cause system errors.

Caution: Help! has detected 13 non-critical problems which may cause abnormal system behavior.

Note: Help! has detected 8 conditions which are not necessarily problems, but you may want to look into.

Figure 8-24. The first few lines of Help!'s information and analysis of a Macintosh's hardware and software configuration.

The advanced tool kit

At the advanced level, you need a protocol analyzer to look at the packets. There are two basic types: software that runs on a Mac and costs less than $1000, and a software/hardware combination that runs on a DOS machine, which is always over $10,000 and usually over $20,000 before the device will provide all the capabilities you need. While the DOS-based protocol analyzers are very powerful, the Mac-based analyzers are easier to use. No one would ever buy a Mac-based protocol analyzer to analyze a network if AppleTalk was just one of the protocols they needed to analyze. No one should buy a DOS-based protocol analyzer just to troubleshoot AppleTalk.

If you do consider getting a DOS-based analyzer, make the decision based on how it handles a number of protocols, not just AppleTalk. If you buy a Mac-based analyzer, it's probably best to judge it solely by how it handles AppleTalk. We have used most of the major contenders in this area, but for 95 percent of the AppleTalk troubleshooting we do, we prefer to use one of the Mac-based troubleshooting tools. The DOS-based analyzers are best used in the rare times when you need to capture a mountain of packets or when you have an Ethernet that is overflowing with traffic.

The two main contenders for Mac-based analyzers are Neon's NetMinder Ethernet and the AG Group's EtherPeek (both companies also make a LocalTalk product that has similar features). While they are both less expensive than any of the DOS-based analyzers, they are still costly enough that you probably won't buy both. We're sure that you'll have equal success with either one. We happen to prefer the AG Group's products because overall we find them slightly easier to use, but we have colleagues whose opinions we greatly respect who prefer Neon's product.

A protocol analyzer lets you watch the network process. When you capture the packets, you see the information that the devices exchange. Watching the information exchange between two devices allows you to chart their progress as the devices conduct their network business, but the information they send-the commands they give, the questions they ask, the answers they give-also allows you to peer into each device's internal processes. Most expert troubleshooters actually endow the devices with animate characteristics and needs. They say things like, "Okay, so this router's just learned about network 23, but he still doesn't know what 23's zone name is, so he asks the router who told him about network 23 for the zone information with a ZIP Query. Since the ZIP Query wasn't answered, it means that the router that he asked (the router who advertised network 23 in the first place) doesn't know what zone goes with network 23, either."

Talking this way and thinking of the devices as animate helps you understand their roles in the network process. It also helps you remember which devices are which because you can get beyond the bloodless view of Ethernet addresses and CPU types. You can actually take the viewpoint of an individual device, placing yourself in its situation and asking questions such as, "What is being asked of me, what do I currently know about the network, what do I need to know in order to complete this task, where is that information kept, and how do I ask for this information?"

Protocol analyzers take longer to learn to use properly than any other tool in your kit. Once you become an expert at protocol analysis, however, it will frequently be the second software tool out of your bag (after Inter•Poll) because it makes all of the information on the network visible.

Summary

An expert troubleshooter requires the following tools:

- The network hub's management software
- Inter•Poll
- A screen-sharing program like Timbuktu or Carbon Copy
- A traffic monitor
- A network alert system like NetWatchMan or NetWORKS
- A system management package like GraceLAN or Network SuperVisor
- A protocol analyzer like EtherPeek or NetMinder Ethernet

When a network has a problem, your troubleshooting tools are useful only if you've used them before. Besides the expertise of their use that comes with familiarity, you also need baseline data to compare to the data you measure when the network is unhealthy.

In many areas of network management and troubleshooting, our current generation of tools are under-powered and our management options are limited. Because of this, we need to be clever and creative about how we approach troubleshooting. We often need a variety of tools and approaches because each tool or approach can only provide us a limited view of the problem and its environment. We often need a more global view that can be assembled from multiple views, even if the problem may sometimes seem to be seen as through a kaleidoscope.

Developing a deep understanding of the underlying protocols, technologies, and components is critical to our success as troubleshooters. On the other hand, everything changes in networking, so old understandings are constantly stretched and give way to new concepts. We are both blessed and cursed by working in a field that offers a never-ending supply of information to learn and incorporate.

CHAPTER NINE

Case Studies

Since in the process of printing, the Mac and the LaserWriter exchange Postscript commands, you can understand the details of the process by reading the packets. Troubleshooting a printing process can be intimidating at first because of all the Postscript code inside the packets. Most network managers are unfamiliar with the PostScript language, but as you will see, PostScript uses commands that are or are similar to English words. In addition, these commands are described in widely available reference manuals. (Some of these are listed in the bibliography.)

Because you can so easily follow the process, printing problems lend themselves particularly well to troubleshooting in process mode; however, troubleshooting almost always starts in hunch mode. As the following case study demonstrates, hunch mode helps you gather clues and develop theories when the user is stymied.

Troubleshooting a Printing Problem

In this particular case study, hunch mode even led to a solution, although not a great one. The analysis then switched to process mode to explore the boundaries of the problem and find a better solution.

A little help with terminology: The printer in this example is a GCC BLP IIS, which is similar to a LaserWriter NTX in performance and use. We'll refer to it as the BLP from now on. The term "LaserWriter" refers to LaserWriters as a class of devices. "LaserWriter driver" refers to the Chooser extension file (which Apple formerly called an RDEV) that drives all LaserWriters, including the BLP.

What happened

A user had a disk that contained, among many other files, a nine-page Microsoft Word 4.0 document consisting of text and graphics. The user printed it from a Mac Portable in background mode (using PrintMonitor) to the BLP.

The user's description: "After starting the print job, I walked away from the portable. When I returned, I saw the Notification Manager's blinking icon and a message saying that PrintMonitor needed my attention. When I brought PrintMonitor to the front, it said that the document was okay, but it couldn't be printed because of a PostScript error."

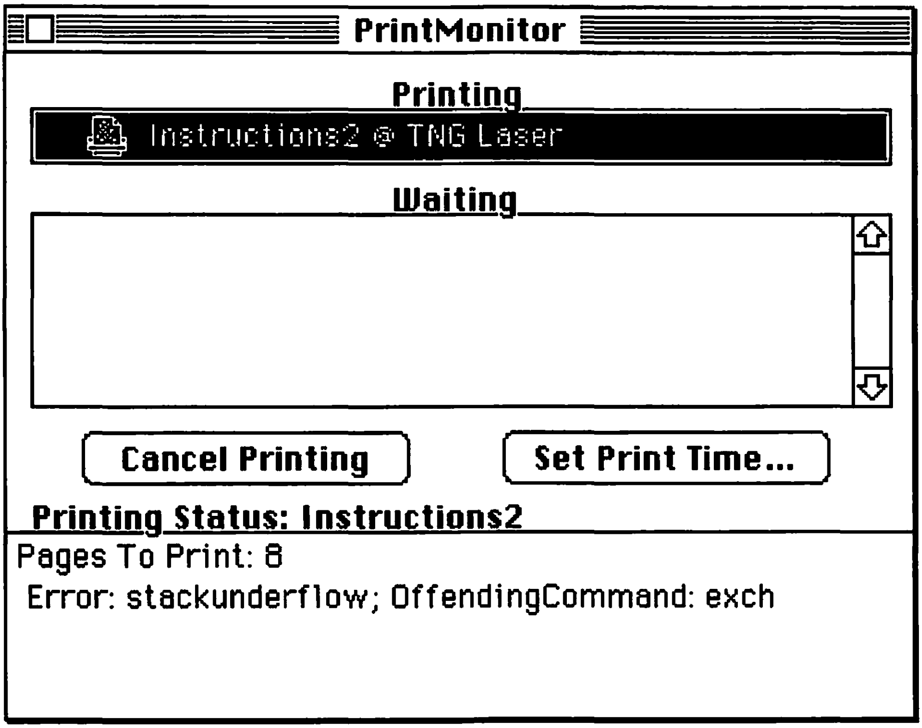
The first thing we did was repeat the user's procedure, with one exception: we printed the document with PrintMonitor in the foreground so that we could see all the status messages coming from the printer. The job proceeded normally at first:

1. Pages to Print: 9
   Looking for LaserWriter "TNG Laser."
2. Pages to Print: 9
   status: starting job
3. Pages to print: 9
   user: Amr; document: Instructions2; status: processing job
4. Pages to print: 9
   user: Amr; document: Instructions2; status: preparing data
5. Pages to print: 8
   user: Amr; document: Instructions2; status: preparing data

Then the screen message in Figure 9-1 appeared.

Since status messages in the PrintMonitor window come from the BLP and not from the Mac, whatever the nature of this error, it occurred as the BLP was trying to turn the Portable's PostScript instructions into a printed page.

First, we checked the BLP printer manual. It said that messages in this format are generated by the standard error handler, and we should check the *PostScript Language Reference Manual.* By checking through a few PostScript and LaserWriter reference manuals, we learned that the command "exch" asks the LaserWriter to exchange two variables in its current stack. If the stack doesn't contain valid variables in both stack positions, you get the "stack underflow" error. Since "exch" is a common PostScript command, there's no telling where this error occurs in the printing process without a little digging. We began by stating the problem.



Printing Status: Instructions2

```
Pages To Print: 8
 Error: stackunderflow; OffendingCommand: exch
```

Figure 9-1. During a printing session, this message appeared on the screen. The print session on this nine-page document terminated without a single page printed. Messages like this one that occur in the middle of the job appear only when you have PrintMonitor in the foreground.

> Problem Statement: *Something in the document causes PostScript errors on the BLP.*

Having stated the problem, hunch mode was a natural choice. We knew that this was some kind of software error because the problem occurred in PostScript. Next we needed to consider the software involved, including the application (Microsoft Word), the LaserWriter driver, the system software, and the PostScript interpreter in the BLP.

First pass—hunch mode

Since the portable was running on System 6.0.7 and the LaserWriter driver it used was from System 7, we thought that there might be an incompatibility problem between the driver and the system. Nothing we've ever read about using the System 7 driver on a System 6 machine would indicate this possibility, and the user had operated the portable in that configuration for quite some time, but the equipment and software used to create the document was from an unknown source. Because we had no knowledge of the document's contents other than what we'd seen on the screen, we made software incompatibility our first theory.

> Current Theory 1: *The System 7 LaserWriter driver, used in conjunction with System 6.0.7, is incompatible with something in the document.*

To test the incompatibility theory, we printed the same file from a Mac running System 7 and got the same error response. So we removed the System 7 LaserWriter driver and installed a System 6.0.7 LaserWriter driver and Laser Prep system documents on the portable. When we printed the file from the portable once again, the LaserWriter, after reinitializing itself, printed the document without error.

Despite the fact that hunch mode cured the problem—we were able to print the document—we didn't consider the cure completely acceptable. We couldn't keep System 6 on the portable to let the user edit and print the remaining documents at any time, because the rest of the office used System 7 drivers and we didn't want to have the printer engaging in time-consuming reinitializing over and over. We needed to find a way to print the documents from System 7 LaserWriter drivers.

At least we had the document to look at. We looked it over for any clues. It was nine pages of text and graphics, with two or three graphics per page. It appeared to have been created by a fairly advanced computer user, since it incorporated many features that beginners don't use, such as vertical lines, tables, and footnotes. Most of the graphics were screen dumps enhanced with text annotation and arrows. Although there didn't seem to be any color graphics, judging by the printed page and by what we saw on the screen of the System 7 print test, color graphics on LaserWriters or BLP's aren't usually a problem in the printing process (aside from the fact that the print quality of the graphics might not be very high).

Hunch mode had shown that something in the file was compatible with System 6 LaserWriter drivers but incompatible with the System 7 drivers. Since Microsoft Word 4.0 definitely *is* compatible with System 7, the most likely conclusion was that there was an object in the file created by an application that is not System 7-compatible—probably one of the graphics. Since Microsoft Word has no native graphics ability, any graphic in the document would have had to come from another application.

The next question was which graphic, and from which application? To answer this question, we needed to switch to process mode.

Second pass—process mode

First we thought about the available clues. Although you should constantly verbalize or write down your theories and clues, it's especially important when you switch troubleshooting modes.

> Current Theory 2: *Some graphic object in the file is incompatible with the System 7 LaserWriter driver.*
>
> Clues:
>
> - Since there were nine pages in the document and the error message showed eight pages left to print when the error occurred, the incompatibility was probably on page 2.
> - The offending command "exch" was discovered by the LaserWriter and not internally in our Mac. We knew this because the message text appeared in the status window of PrintMonitor. As mentioned before, these status messages come from the LaserWriter.

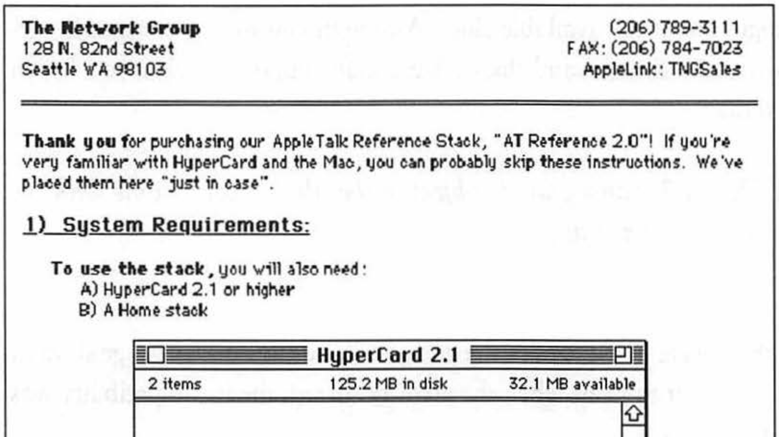
For reference, Figure 9-2 shows a 50 percent view of page 2 of the document we were trying to print.

Exploring the trouble process

The next step was to find the place in the file where the LaserWriter encountered the problem. We opened Word again and printed the file from the System 7 Mac, capturing the packets with LocalPeek. We also created a PostScript file for reference purposes. You can do this by using the destination checkbox in the Print dialog box. This PostScript file, created by the LaserWriter driver, contains all the information sent to the printer—and a bit more.

The PostScript file, readable with any word processor, contains all the information that a LaserWriter (or a BLP) could ever possibly need to print the document—all the font definitions, user dictionaries, and so forth. Some of this information, however, is already stored in the BLP, and the LaserWriter driver tries to avoid sending redundant information. When the LaserWriter driver (LD) handles a normal print job, it asks the

BLP what information it currently has in memory and compares it to what is needed for the document. For example, it asks which fonts are in the printer's memory and downloads only the fonts in the document that the printer doesn't already have.



The Network Group
128 N. 82nd Street
Seattle WA 98103

(206) 789-3111
FAX: (206) 784-7023
AppleLink: TNGSales

Thank you for purchasing our AppleTalk Reference Stack, "AT Reference 2.0"! If you're very familiar with HyperCard and the Mac, you can probably skip these instructions. We've placed them here "just in case".

1) System Requirements:

To use the stack, you will also need:
A) HyperCard 2.1 or higher
B) A Home stack

C) Approximately 1 MB free space on your hard disk.
D) Approximately 1 MB of free RAM, more is better.

2) Installing your AppleTalk Reference Stack:

To install this stack on your hard drive, insert the floppy and drag the file "AT Reference 2.0.SEA" into your HyperCard folder.

Then, eject the floppy disk by dragging the disk icon (not the file icon) into the trash. If you are unfamiliar with this procedure, your Macintosh Reference Manual has more details.

Figure 9-2. The current theory is that one of the graphic objects on this page is incompatible with the System 7 LaserWriter drivers.

The PostScript file (PSF), on the other hand, contains a font definition for every font in the file, so this file almost always contains more information than would normally be sent during printing. Whatever the offending object was, it would be contained in the PSF as well as in the packets. The PSF is easier to read because the PostScript instructions are all in one place rather than spread out over hundreds of packets.

To get a feel for the print job, we loaded the PSF into Word and numbered the lines (using Renumber..., found in the Utility Menu). There were 7840 lines of PostScript code. Then we replaced all instances of "exch" with "exch". This changes nothing, of course, but it let us know that there were 292 instances of "exch," the command that caused the stack underflow error. Then we got an idea of where the document pages fell in the PSF by searching for Page 1, Page 2, and so on. This search showed us where the information for a specific page began. The pages came out like this:

Instructions For: Started On: Instructions For: Started On: Page 1: Line: 525
Page 6: Line: 4469
Page 2: Line: 1210 Page 7: Line: 5547
Page 3: Line: 1764 Page 8: Line: 6627
Page 4: Line: 2934 Page 9: Line: 7810
Page 5: Line: 3733

Since we suspected that the problem was on page 2, we selected the lines 1210-1764 in the PSF and counted the number of instances of "exch." This count showed only seven instances. When we looked to see where "exch" was located in the description for page 2, we found that they all occurred in the selection below:

```
1237. % P2 Header - Version 2.0-14 - Copyright 1988 Silicon Beach
  Software, Inc.
1238. userdict/md known{currentdict md eq}{false}ifelse{bu}if
  currentdict/P2_d known not{/P2_b{P2_d
1239. begin}bind def/P2_d 26 dict def userdict/md known{currentdict
  md eq}{false}ifelse P2_b dup dup
1240. /mk exch def{md/pat known md/sg known md/gr known and
  and}{false}ifelse/pk exch def{md
1241. /setTxMode known}{false}ifelse/sk exch def/b{bind def}bind
  def/sa{matrix currentmatrix P2_tp
1242. concat aload pop}b/sb{matrix currentmatrix exch concat P2_tp
  matrix invertmatrix concat aload
1243. pop}b/se{matrix astore setmatrix}b/bb{gsave P2_tp concat
  newpath moveto}b/bc{curveto}b/bl
1244. {lineto}b/bx{closepath}b/bp{gsave eofill grestore}b/bf{scale
  1 setlinewidth stroke}b/be
1245. {grestore}b/p{/gf false def}b/g{/gf true def}b g pk{/_pat/pat
  load def/_gr/gr load def}{/_gr
1246. div setgray}b}ifelse sk{/_sTM/setTxMode load def}if/gx{/tg
  exch def}b 0 gx end P2_b pk
1247. end{/pat{P2_b gf{end pop sg}{/_pat load end exec}ifelse}bind
  def}{/pat{P2_b pop _gr end}bind
1248. def}ifelse P2_b sk end{/setTxMode{P2_b/_sTM load end
  exec P2_b tg/_gr load end exec}bind def}
1249. {/setTxMode{pop P2_b tg/_gr load end exec}bind def}ifelse}if
1250. 0 pen
1251. 526 gm
1252. 525 lin
1253. 159 1 index neg 1 index neg matrix translate 3 1 roll
1254. currentpoint 2 copy matrix translate 6 1 roll
1255. 575 gm
1256. 576 lin
1257. 207 currentpoint 1 index 6 index sub 4 index 9 index sub div
1258. index 6 index sub 4 index 9 index sub div
1259. matrix scale 11 1 roll
1260. [ 9 1 roll cleartomark
1261. 2 roll matrix concatmatrix
1262. exch matrix concatmatrix
1263. /P2_tp exch def P2_b mk end{bn}if
1264. 1 pen
1265. 215 gm
1266. 214 lin
1267. T 51 48.95046 525 150 14 107 103 T 1 db
1268. 0000000000000000000000000000
1269. 0000000000000000000000000000
1270. FFFFFFFFFFFFFFFFFC0000000000
1271. FFFFFFFFFFFFFFFFFC0000000000
1272. FFFFFFFFFFFFFFFFFC0000000000
```

In PostScript, the "%" sign begins a comment, and comments begin functional sections of the PostScript instructions. As you can see, line 1237 contains a comment that includes the words "Silicon Beach"—the name of the company that makes the graphics application SuperPaint. The hexadecimal data that begins on line 1268 is the beginning of a bitmap. From this, it is obvious that lines 1237–1267 and all the hexadecimal beyond are part of the graphic.

Checking the document that Apple ships along with System 7 upgrades—*System 7: Compatibility With Selected Hardware and Software*—we found that Silicon Beach's SuperPaint 2.0 is listed as mostly compatible while version 2.0a is listed as fully compatible. Time to update Current Theory 2.

> **Current Theory 3:** *A graphic object created by SuperPaint 2.0 is incompatible with the System 7 LW driver.*
>
> **Clues:**
> - The offending command "exch" occurs seven times in a graphic definition on page 2 that was created by SuperPaint.
> - SuperPaint 2.0 is listed as only "mostly compatible" with System 7.

Although we'd identified this graphic on page 2 as a problem, we wondered if there were other graphics that might also have a printing problem. To investigate, we looked through the PSF to find out if there were any other instances of "Silicon Beach." To our surprise, we found the first instance of "Silicon Beach" on page 1 at Line 660. Again, "Silicon Beach" was in the comment at the beginning of a graphic definition shown below.

```
660. % P2 Header - Version 2.0-14 - Copyright 1988 Silicon Beach
  Software, Inc.
661. userdict/md known{currentdict md eq}{false}ifelse{bu}if
  currentdict/P2_d known not{/P2_b{P2_d
662. begin}bind def/P2_d 26 dict def userdict/md known{currentdict
  md eq}{false}ifelse P2_b dup dup
663. /mk exch def{md/pat known md/sg known md/gr known and
  and}{false}ifelse/pk exch def{md
664. /setTxMode known}{false}ifelse/sk exch def/b{bind def}bind
  def/sa{matrix currentmatrix P2_tp
665. concat aload pop}b/sb{matrix currentmatrix exch concat P2_tp
  matrix invertmatrix concat aload
666. pop}b/se{matrix astore setmatrix}b/bb{gsave P2_tp concat
  newpath moveto}b/bc{curveto}b/bl
{lineto}b/bx{closepath}b/bp{gsave eofill grestore}b/bf{scale 1 setlinewidth stroke}b/be
{grestore}b/p{/gf false def}b/g{/gf true def}b g pk{/_pat/pat load def/_gr/gr load def}{/_gr
div setgray}b}ifelse sk{/_sTM/setTxMode load def}if/gx{/tg exch def}b 0 gx end P2_b pk
end{/pat{P2_b gf{end pop sg}{/_pat load end exec}ifelse}bind def}{/pat{P2_b pop _gr end}bind
def}ifelse P2_b sk end{/setTxMode{P2_b/_sTM load end exec P2_b tg/_gr load end exec}bind def}
{/setTxMode{pop P2_b tg/_gr load end exec}bind def}ifelse}if
0 pen
74 gm
73 lin
29 1 index neg 1 index neg matrix translate 3 1 roll
currentpoint 2 copy matrix translate 6 1 roll
104 gm
105 lin
71 currentpoint 1 index 6 index sub 4 index 9 index sub div
index 6 index sub 4 index 9 index sub div
matrix scale 11 1 roll
[ 9 1 roll cleartomark
2 roll matrix concatmatrix
exch matrix concatmatrix
/P2_tp exch def P2_b mk end{bn}if
1 pen
-257 gm
-258 lin
T 32 43.90908 73 150 6 33 46 T 1 db
000000000000
000000000000
000000000000
07FFFF000000
```

To our eye, this graphic definition from page 1 looked quite similar to the suspect graphic definition on page 2. Looking further, we found three more instances of a SuperPaint definition on page 1. Now we weren't sure about our SuperPaint theory. We'd suspected the error to be on page 2 because PrintMonitor's error message said there were eight pages (of nine) left to print. It seems we should have suspected that the error was on page 1, since *no* pages were printed. We made a mental note to base future hunches on the number of pages printed rather than on the number in the status window. Then we remembered that since the printer has a buffer, the printer may have discovered the error in page 1 while the Mac was loading the instructions for page 2.

Had we stopped here, we still wouldn't have been able to accomplish our goal solution. The user still would need a System 6 LaserWriter driver to print the documents, and we didn't want a mixed network. We knew that if we looked into the packets, we could discover more information that might help us create a better solution. To help us decide whether or not to continue, we made a list of all the things we didn't know.

The first unclear item was that our whole troubleshooting analysis was based on reading the PostScript file, which isn't necessarily what is sent to the printer. Remember, the LaserWriter driver can decide *not* to send information that appears in the PSF if the LaserWriter already has that information. Looking inside the packets would tell us more precisely where the offending "exch" occurred. We could find out whether all, only portions of all, just one, or none of the Silicon Beach definitions were sent.

The second unclear item is that we had no idea how much data had been delivered before the BLP discovered and reported the error. Looking in the packets, we could find the exact moment that the LaserWriter reported the PostScript error and then search through the packets immediately before that for instances of "exch."

Because there was still more to learn, we decided to continue. Perhaps in finding out what we didn't know, we'd also discover a better solution.

Examining the printing packets

When you first approach the printing packets, try to get a feel for the entire printing session in the same way that we got a feel for the PSF. There are several phases to a print job. While these are discussed in more detail in Chapter 7, "Processes: Printing Process Description," we'll repeat some of that information here to help you gain a feel for the print job.

The entire printing session is driven by the LD, which does the following:

1. Locates the printer from the Chooser name and finds its AppleTalk address
2. Establishes a session with the LaserWriter
3. Queries the LW to find which version of the Laser Prep dictionary is loaded
4. Queries the LW to find which fonts are loaded

5. Restarts LW or downloads additional dictionary information, depending on query results
6. Sends a description of the job (user name, application, document name, number of copies)
7. Sends the document characteristics
8. Downloads any additional fonts needed (this may occur in body of print info)
9. Sends the actual page descriptions
10. Closes the printing session

When we captured the packets, we used address filtering to capture only those packets sent by the BLP or the portable. If we hadn't done that during capture, we'd have to filter out all the other packets at this time to make the print session easier to follow.

The unsuccessful print session used 221 packets and lasted for 17 seconds. The table below shows where the different sections of the job started, the packet number, the time of the packet, and the search string used to find them in the print captures. The initialization information is shown as N/A, since it wasn't needed for this job. Also, the PostScript code for page 3 was never sent because of the error.

Section	Packet #	Time (sec)	Search String
Locate BLP	1	0	$74657201
Establish print session	5	2.1	$020000
Query prep version	11	2.6	ProcSetQuery
Query for fonts	23	3.6	FontListQuery
Font List Reply	26	3.7	Oblique
Restart LW	N/A	N/A	Restarting
Additional Prep Info	38	4.8	BeginProcSet:
Download Fonts	N/A	N/A	BeginFont
Doc Characteristics	122	9.9	DocumentSetup
Begin Print Info	123	9.9	Page 1
Page 2 begins	165	14.1	Page 2
PostScript error reported	183	16.0	Offending
Page 3 Begins	Canceled	N/A	Page 3
Session Ends	221	17.2	$070000

Since we knew the page breakdown, we catalogued the instances of "Silicon Beach" relative to the page numbers and also found the packet where the error was reported.

"Silicon Beach" Instance	Packet	Page
1	125	1
2	132	1
3	140	1
4	146	1
5	166	2
6	179	2
Error Reported	183	2

Looking at the Word file we were trying to print, we saw that there were four graphic objects in the instructions for page 1 and three graphic objects in the instructions for page 2. We concluded that a new "Silicon Beach" dictionary definition was sent for each graphic object and that the third definition on page 2 wasn't in the packet trace because the print job was canceled before it could be sent.

> **Question:** *Why did the BLP allow six SuperPaint graphics to be sent before reporting the error?*

It occurred to us that since the 2MB BLP has a buffer of some size, it may have taken in a good batch of data before it started to work on the PostScript code. It might have "choked" on the very first graphic and reported the error while the other five graphic objects were sitting in the buffer waiting to be processed. On the other hand, maybe each of the graphic objects had different characteristics, some of which were compatible with the System 7 LaserWriter driver and some of which were not.

We individually tested the seven graphics on the first two pages by copying them into a document by themselves and printing them one by one. Of the seven graphics, four could be printed and three could not. We examined their characteristics to discern a pattern. The pattern revolved around whether the graphic object was created in SuperPaint's Draw mode or its Paint mode. The results are as follows:

Page - Figure	Could it Print?	SuperPaint Mode
Page 1—Figure 1	Yes	Paint
Page 1—Figure 2	Yes	Paint
Page 1—Figure 3	Yes	Paint
Page 1—Figure 4	No	Draw
Page 2—Figure 1	Yes	Paint
Page 2—Figure 2	No	Draw
Page 2—Figure 3	No	Draw

The difference was obvious. The graphic objects that could print were all rendered in SuperPaint's Paint layer—they were bitmaps. The graphic objects that didn't print under System 7 were Draw objects. By using a screen capture utility (Flash-It 2.1), we took snapshots of all of the Draw objects and replaced them with their snapshots. The two pages then printed without an error.

The memory buffer did have an effect on the precision of the diagnostic method. The BLP choked on the fourth SuperPaint graphic, but allowed two more graphics to be sent before reporting the error.

Conclusion: It pays to keep going

The preceding situation is an example of continuing the troubleshooting process past the point of the first, most obvious solution. At two points in the investigation we could have called the job complete, but in both instances continuing the investigation refined the solution. These more refined solutions provided something more than just idle intellectual satisfaction; they allowed a wider latitude of choices for the user in coping with the situation.

Hunch mode offered the solution, "Print with the System 6 LaserWriter driver and everything will be fine." This is a workable solution in that it provides a way of printing the files, but it has the unfortunate side effect of creating a network with mixed LaserWriter drivers, which means that the LaserWriter will frequently need reinitializing. While some Laser printers reinitialize very quickly, most printers need at least a couple of minutes. Because the user's workstation is engaged in the process of reinitializing, this solution isn't very attractive; most users don't want to wait that long. Some network managers might at this point designate one Laser printer a System 6 device and another a System 7 device.

Continuing the investigation, we discovered the solution "Get an upgrade to SuperPaint 2.0a and everything will be fine." The problem with this approach is the time delay between ordering and receiving the upgrade, especially if users have files that need to be printed right away.

The final solution we discovered was, "Convert all your graphics to paint objects and everything will be fine." This too has an undesirable side effect in that paint objects are printed at lower resolution (72 dpi) than draw objects (the resolution of the printer), but now the user has a range of solutions available.

The solution we'd like to find is a way to print the graphics in these files—and any graphics like them—at the highest possible resolution before the SuperPaint upgrade arrives. We can continue to explore this incompatibility problem by trying other methods, such as copying the graphics to the Scrapbook, then to another draw program like Canvas or MacDraw, and then back into Word.

Many tools and references

We "threw the book" at this problem, using PostScript reference manuals, disk editing tools, packet analyzers, Apple manuals, and other sources of information. This kind of research is often necessary for a problem occurring in the higher layers of the protocol stack. PostScript is a presentation layer (Layer 6) protocol, dependent on many lower layers to function properly. Many printing problems are caused by lower layers. In this case, however, it was caused a PostScript language incompatibility.

As we mentioned in the Preface, networks involve many components in complex interaction with one another. No person can remember everything necessary to troubleshoot problems like this one, so it's important to have a range of books and resources available, like the ones mentioned in the bibliography. You don't need to read them all right away, but you should become familiar with the information and approach of each book.

Troubleshooting a Router Problem

We began looking into a very puzzling problem in Chapter 3, when we discussed how to get started troubleshooting network problems. Let's review what we discussed at that point.

We were confronted with a situation in which a particular Macintosh was unable to gain access to a specific server. The user initially attempted to mount the server volume using an alias that had worked without problems several times in the past but was now failing consistently.

Using the Chooser to double-check the proper operation of the alias file, we discovered that the server could not be found there either, at least not in the zone where we expected to find it. Other servers could be located and mounted correctly, but not the one we wanted to access. At the least, it seemed as though the AppleShare workstation software was working properly.

We got as far as using Inter•Poll to double-check the correct operation of the Chooser. The results we obtained here were identical to the previous ones. No server of the specified name could be found in the zone indicated.

Next, we checked to see whether we could see the server on a nearby Macintosh. That machine failed to locate the server as well. Clearly, the problem was not isolated to a single system on the network.

At this point, we called the administrator responsible for the server to see if it had been taken off-line for any reason. We discovered that not only was the server on-line and available, it was currently being accessed by several other users on the network. Moreover, none of the current users had reported any problems in using that server.

Upon inspecting one of the Macintoshes that *was* able to access the server, we noted nothing particularly unusual. It could access, dismount, and remount the server. Both the Chooser and Inter•Poll clearly showed the server as being available on the network. The Mac accessing the server showed no significant differences in either hardware or software configuration.

At this point we went back to look at the network layout documentation to see if we could come up with any obvious common factors that might explain why the two Macintoshes displaying the problem might be having difficulty accessing the server. For instance, we suspected that the two problem Macs might be on the same cable and the cable might be broken or improperly terminated.

Unfortunately, we found nothing on this score, either. The two Macs were on separate EtherTalk networks. One of them, as a matter of fact, was on the same EtherTalk network as the server!

As another cross-check, we enabled file sharing on each of the two Macs to see whether they were able to see each other. In fact, they could. Each Mac was able to access the other's hard disk with no apparent problems.

At this point, however, we noted a real peculiarity. We went back to the Macintosh that *was* able to access the server to try to log in to one of the problem Macintoshes and discovered that while it *can* access the original server, it *can't see either one of the problem Macintoshes!*

The situation, as we currently understand it, is displayed in Figure 9-3.

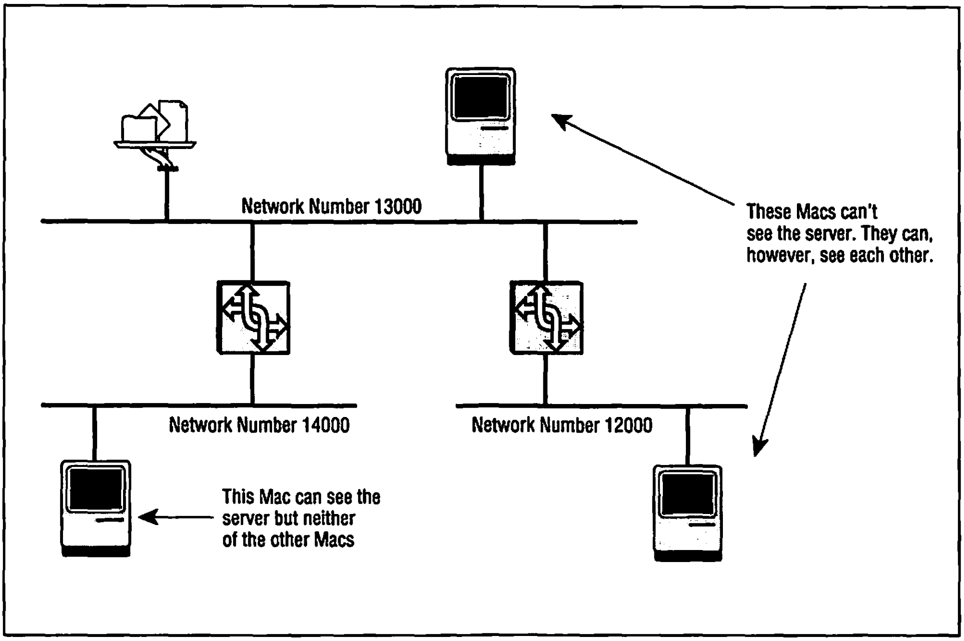

Network Number 13000
These Macs can't see the server. They can, however, see each other.
Network Number 14000
Network Number 12000
This Mac can see the server but neither of the other Macs

Figure 9-3. This figure summarizes what we know about our situation at the moment.

We verified this result with both the Chooser and Inter•Poll and in both cases came to the same conclusion: the Macs manifesting the problem were apparently invisible to the Macs that were not.

Getting into details

So where do we go from here? It's probably time to break out our network analyzer and see what is actually transpiring on the wire.

We can begin by capturing some NBP packets at various points on the network. This seems a reasonable way to proceed because our problem seems to be related to some vagaries in the way NBP is operating. In a normal situation, the Macintosh trying to access the file server sends out (either directly or through the mediation of a router) LookUp frames for AFPServers in the selected zone; the server responds with LookUpReply frames, returning its name and Internet Socket Address.

Apparently, this sequence of events is failing somewhere. We'd like to find out where things are breaking down.

We start the next phase of our investigation by setting up a Macintosh running EtherPeek on network number 12000, alongside the Macintosh that originally manifested the problem. We set EtherPeek up to filter all frames except those going to or from the problem Mac, using the Ethernet address of the Macintosh's network interface card. Further, we know that unlike AppleTalk addresses, Ethernet addresses are static and will not be changing. We start EtherPeek, go over to the problem Macintosh, and open the Chooser.

After a few minutes, we stop EtherPeek and inspect the results. We see that we have collected a number of NBP broadcast requests directed to the router between EtherTalk networks 12000 and 13000, but no responses are coming back. This corresponds to the symptoms we have been seeing, but it offers no insights into the nature of the difficulty.

We next set up EtherPeek on a Macintosh on network 13000, the same cable segment on which the server resides. Here, we configure it to filter all packets except those going to or coming from the router's port on this network, again using the Ethernet address. To cut down on the amount of data we might have to wade through, we also set it up to filter everything transmitted by the router other than NBP frames since these are what we are primarily interested in. This excludes RTMP and ZIP frames in particular, because these are not what we are immediately looking for.

Again, the results validate what we've seen previously, but they don't really help us understand the cause of the problem. The router is dutifully sending out LookUp frames on behalf of the original Macintosh. The server, however, is failing to respond to them.

We clearly have a case of a process not working here. Apparently, there is some sort of "defect" in the NBP Lookup frames the router is sending to the server. As we sit back down at the first Macintosh, looking glumly at the Chooser display, we realize something: the other Macintosh that cannot access the server *is* responding to the LookUp frames!

Of course, we knew this already, even if we did not focus on it. But now we have something to focus on, possibly to some advantage. The server responds to LookUps from the Mac on net 12000; the Mac on the same net does not. The Mac on the server's net, on the other hand, answers LookUps from the original problem Mac, while the server does not. It occurs to us at this point that by comparing the two

LookUp frames, we can possibly spot a difference between them. If we can, we might well have located the approximate cause of the problem.

The unsuccessful LookUp packet—the one to which the server fails to respond but which is answered by the other problem Macintosh—appears as follows:

```
Long DDP Header - Datagram Delivery Protocol
 Dest. Network: 1300
 Source Network: 1300
 Dest Node: 255
 Source Node: 252
 Dest. Socket: 2 NBP Socket
 Source Socket: 254
 DDP Type: 2 NBP
NBP Packet - Name Binding Protocol
 Function: 2 LkUp - LookUp Request
 Tuple Count: 1
 NBP ID: 5
NBP Tuple #1
 Node Address: 12000.245
 Socket Number:250
 Enumerator: 0
 Object: =
 Type: AFPServer
 Zone: Administration
```

This looks completely normal. But there must be something about it that the server doesn't like. Perhaps a comparison with the other Macintosh's LookUp frames will be enlightening.

The NBP Lookup frames sent on behalf of the Macintosh that *can* access the server, look like this:

```
Long DDP Header - Datagram Delivery Protocol
 Dest. Network: 13000
 Source Network: 13000
 Dest Node: 255
 Source Node: 248
 Dest. Socket: 2 NBP Socket
 Source Socket: 252
 DDP Type: 2 NBP
NBP Packet - Name Binding Protocol
 Function: 2 LkUp - LookUp Request
 Tuple Count: 1
 NBP ID: 5
NBP Tuple #1
 Node Address: 14000.238
 Socket Number:251
 Enumerator: 0
 Object: =
 Type: AFPServer
 Zone: Administration
```

Again, nothing here leaps out as being more correct than the previous frame. Yet there must be something!

And of course, there is. Embarrassingly enough, it took ten minutes and the better part of a can of Mountain Dew before we found the problem. The first Macintosh's LookUp frames were being directed to network number 1300, *not* network number 13000! Apparently, the router between network 12000 and the server's Ethernet segment was confused about the numbering of one of its ports.

To verify this, we repeated the collection of NBP LookUp frames, this time for the Macintosh on the server's net. The results we saw pretty much confirmed things:

```
Long DDP Header - Datagram Delivery Protocol
 Dest. Network: 1300
 Source Network: 1300
 Dest Node: 255
 Source Node: 252
 Dest. Socket: 2 NBP Socket
 Source Socket: 254
 DDP Type: 2 NBP
NBP Packet - Name Binding Protocol
 Function: 2 LkUp - LookUp Request
 Tuple Count: 1
 NBP ID: 5
NBP Tuple #1
 Node Address: 1300.234
 Socket Number:241
```

```
Enumerator: 0
Object: =
Type: AFPServer
Zone: Administration
```

No question about it. Not only did the router believe that its port into network 13000 was connected to network 1300, but the Macintosh on the same cable as the router thought it was on network number 1300 as well!

Armed with this information, we finally took a close look at the router that seemed to be the cause of the problem. Sure enough, it had been misconfigured. It was set up as a seed router with network number 1300 on the Ethernet port, when it should have been configured as network number 13000.

Down the oubliette

What we have uncovered here is a networking situation commonly referred to as a "black hole." By creating a discrepancy among the routers on what *should* be network 13000, we have in effect created two mutually disjointed networks on the same piece of cable. This situation, which is not at all healthy, is illustrated in the Figure 9-4.

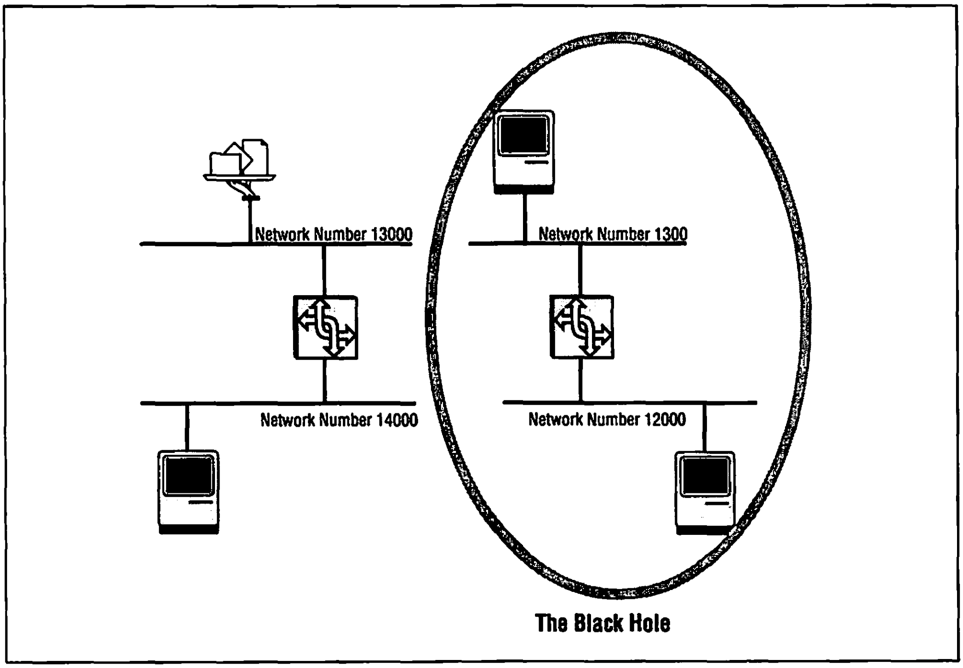


Figure 9-4. An illustration of our network's "black hole"

Finally, we are now in a position to really understand the sequence of events that led to our predicament:

1. The problem Macintosh, upon opening the Chooser and selecting the Administration zone, sends out Broadcast Request frames to its router.

2. The router, which is mistakenly configured to translate the Administration zone name to network number 1300, sends out a series of NBP LookUp frames to the broadcast address on the Ethernet, which is, in fact, network number 13000.

3. The server never receives these frames. Although the zone multicast address is correct, because it is based on the zone name and the Ethernet card on the server does receive the frames at the physical level, the frames are discarded by the time they reach the data link layer. DDP does not deliver the frames addressed to network 1300, because the server's address is in fact on network 13000.

4. The Mac on net 13000, which also believes itself to be on network 1300, *does* respond to the LookUp frames, because the address corresponds to the network number it believes it's on. When the Macintosh on network 14000 sends out its LookUps, however, this Mac discards them in the same way that the server discarded the other Mac's LookUps.

After a little thought, we realize that this situation also explains why the Macintosh that *can* access the server cannot access either of the two problem Macintoshes. In the case of the Mac that should be on network 13000, the explanation is easy: it thinks it's on net 1300 and discards the LookUps.

The case of the Macintosh on network 12000 is only slightly more complex. Every ten seconds, the misconfigured router is sending out RTMP frames advertising its mistaken belief that network 13000 is, in fact, network 1300. The other router, seeing the discrepancy of network number between its own configuration and the network number passed in the RTMP frame, discards the RTMP frame completely, ignoring any further routing information it might contain. As a result, it never collects a route to network 12000 and is unable to forward packets there.

Further ramifications

How, you might wonder, did the Macintosh on the same Ethernet as the server come to believe that it was on network 1300 as well? There is a fairly simple explanation for this. When a Macintosh starts up, it sends out a ZIP GetNetInfo request to validate its stored zone name and obtain a network number corresponding with that zone name. It accepts the first GetNetInfo reply it receives and ignores any subsequent replies. On startup, the first GetNetInfo response that this Macintosh saw must have been sent by the misconfigured router. This caused the Macintosh to fall into the "black hole."

If we had happened to restart this Macintosh we might have noticed that, upon restarting, it had accepted the GetNetInfo reply from the other server instead. If that had happened, the Macintosh would have begun to work perfectly for no immediately apparent reason.

This sort of situation can mask problems that can then hang around on the net for a long time. Everyone likes to indulge in a bit of random troubleshooting, and rebooting the machine is a normal thing to try when unexplained problems occur. As illustrated here, this can sometimes make the problem appear to "go away." Therefore, the user never reports it but the problem persists.

How did this problem arise? We know that a misconfigured router should refuse to start up. Unfortunately, several available ones don't. This one, however, was not one of those, but a Macintosh running Apple's AppleTalk Internet Router. Clearly, some further investigation is necessary.

Upon asking around, it turns out that the router in question had developed some problems late the previous Friday. Specifically, the hard disk had gone bad and needed to be replaced. A technician disconnected the router from the network and replaced the hard disk. Because this machine was operating solely as a router, specific backups of its hard disk weren't taken. Instead, the technician reinstalled the system and router software and reentered the router's configuration from a card attached to the machine. Unfortunately, when the technician reconfigured the router, he mistyped the network number for one of the Ethernet ports.

Normally, this would not have been a problem because the router would have refused to start up with its configuration incorrect, but the error was further compounded. When the router was put back in its place, the technician restarted it and realized, once it had started up, that he had neglected to connect the Ethernet cable to the back of the Macintosh. Without thinking, he simply connected the cable to the running Macintosh and left it.

Summary and Observations

This problem was fairly simple, once we figured out where the actual difficulty lay. Determining that occupied most of our time in troubleshooting this situation. Several conclusions can be drawn from our examination of this problem:

- Router configurations are rich grounds for causing all sorts of network problems. Frequently, the symptoms give us no reliable clues to the core cause of the problem. It is good practice to establish a procedure for logging any changes whatsoever in router configuration, so that a "sanity check" can be performed on the network immediately afterward.

- We almost certainly spent too much time looking at other issues before we checked the routers themselves. The error statistics collected by either of the routers would have been very revealing and might have allowed us to clear up the problem more quickly. Specifically, either one of the routers would have logged a Local Net Configuration Error on the affected Ethernet port whenever the other router sent out an RTMP frame containing information that contradicted its own understanding of the number assigned to that net. Similarly, use of a product like RouterCheck would have isolated the problem very quickly.

- Starting any AppleTalk device standalone and then attaching it to a network is a bad idea. Router configuration problems aside, this completely circumvents AppleTalk's scheme for dynamically allocating network node numbers and on crowded networks commonly results in two separate nodes acquiring the same network address.

- Again, it can be very difficult to tie the symptoms you observe back to their original cause. We spent a great deal of time examining the individual NBP LookUp packets themselves without comparing them to one another or considering how they were getting to where they were going. Focusing too early on a specific aspect of the troubleshooting situation without keeping the big picture in mind is an almost certain prescription for increasing the amount of time it takes to resolve a problem.

APPENDIX A

About the Disk

The disk included with this book contains three tools, an AppleTalk Reference Stack, the PacketSend router troubleshooting tool discussed in Chapter 8, and Watch, a protocol analyzer developed by Cayman Systems that is distributed as freeware.

AppleTalk Reference Stack

On the disk that accompanies this book, you'll find a self-expanding archive that, when decompressed, will become the AppleTalk reference stack. You'll need HyperCard (version 2.0 or later) and a Home Stack. There's a Read Me file on the disk if you have any questions about installing this stack.

The stack contains a list of vendors that make the AppleTalk products mentioned in the book (and many others) along with their addresses and phone numbers. This information is current as of April 1, 1993. Also in the stack are two dozen articles by one of the authors of this book, Kurt VanderSluis, as well as mini-reviews of other networking books and magazines.

PacketSend

In Chapter 8, we mentioned a HyperCard stack called PacketSend that would allow you to ask questions of your routers. It's included on the disk for your use.

Watch

Watch is a small freeware protocol analyzer for use on Ethernet networks. Although it doesn't have all the capabilities of a full-blown protocol analyzer such as EtherPeek, it is capable of performing some basic troubleshooting analysis, such as levels of traffic, router information, and so forth.

APPENDIX B

LocalTalk Parameters

This appendix gives the wiring guidelines for LocalTalk networks using the Farallon PhoneNET® System or its equivalent.

Daisy Chain

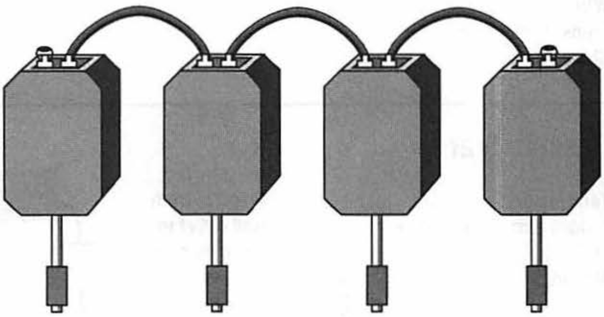
Terminators:
1 mounted terminator
at each end of daisy chain

Max Length:
2000 feet end to end

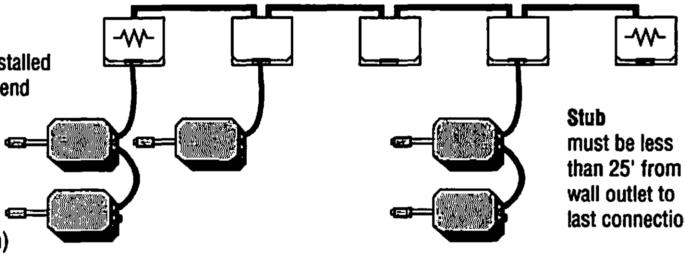
Comments:
Useful for small nets only, keep out of walkways, away from electrical noise

Budget:
22 AWG – 4500'
24 AWG – 3000' (most common)
26 AWG – 2000'

Wire:
Telephone Line Cord (also called flat wire, silver-satin, modular extension cable, etc.)



Backbone



Terminators:
1 loose terminator installed in wall outlet at each end of trunk

Max Length:
Trunk length cannot exceed Budget – (4 x total stub length)

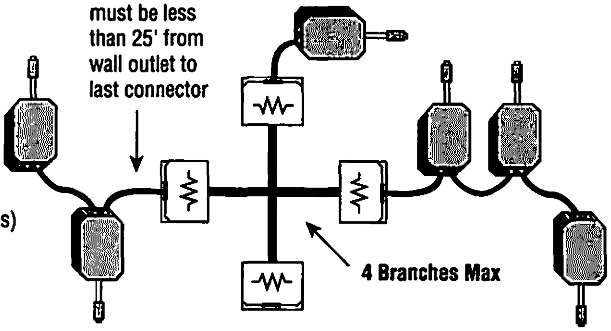
Stub
must be less than 25' from wall outlet to last connection

Comments:
Most useful when you can pre-install the wiring and you'll never need to change the trunk layout

Budget:
22 AWG – 4500'
24 AWG – 3000' (most common)
26 AWG – 2000'

Wire:
Trunk: Twisted Pair
Stubs: Line Cord

Passive Star



Stub
must be less than 25' from wall outlet to last connector

4 Branches Max

Terminators:
1 loose terminator installed in wall outlet at each end of branch

Max Length:
Each branch length cannot exceed: (Budget / # of branches) minus (4 x total stub length)

Comments:
Useful when you have a few devices in a couple of rooms, passive stars can easily upgrade to active stars

Budget:
22 AWG – 4500'
24 AWG – 3000' (most common)
26 AWG – 2000'

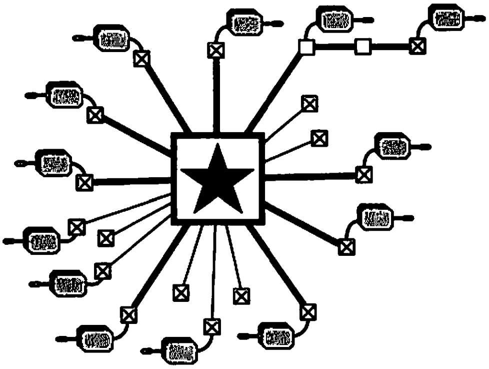
Wire:
Branch: Twisted Pair
Stubs: Line Cord

Active Star

Terminators:
1 loose terminator installed in wall outlet at end of each branch of each port

Max Length:
Treat each port separately, calculate for the configuration you use on that port with budget at right (different from passive budget)

Comments:
An active star topology is suitable for any network that needs reliability, flexibility, and growth potential



Budget:
22 AWG – 3000' per port
24 AWG – 2000' per port
26 AWG – 1500' per port

Wire:
Trunk: Twisted Pair
Stubs: Line Cord

APPENDIX C

Ethernet Parameters

Types of Ethernet Media

Specification	Type	Cabling	Connector
10BASE5	Thick Ethernet	Thick Coax	AUI
10BASE2	Thin Ethernet	Thin Coax	BNC
10BASE-T	Twisted-Pair Ethernet	Unshielded Twisted Pair	RJ-45

Pinouts

Pin	AUI	10BASE-T (RJ45)	AAUI(Apple Attachment Unit Interface)
1	Control In Shield	Receive (+)	Power (+12V @ 2.1W or +5V @ 1.9W)
2	Control In—Circuit A	Receive (–)	Data In Circuit A
3	Data Out Circuit A	Transmit (+)	Data In Circuit B
4	Data In—Shield		Voltage Common
5	Data In—Circuit A		Control In Circuit A
6	Voltage Common	Transmit (–)	Control In Circuit B
7	Control Out—Circuit A		+5V
8	Control Out—Shield		Secondary +5V
9	Control In—Circuit B		Data Out Circuit A
10	Data Out—Circuit B		Data Out Circuit B
11	Data Out—Shield		Secondary Voltage Common
12	Data In—Circuit B		Not Used
13	Voltage Plus		Not Used
14	Voltage Shield		Secondary +12V @ 2.1W or +5V @ 1.9W
15	Control Out—Circuit B		(AAUI has only14 pins)
Shell	Ground		Ground

Ethernet Manufacturer Designators

00-00-0F	NeXT
00-00-10	Sytek
00-00-18	Webster
00-00-1B	Novell
00-00-1D	Cabletron
00-00-20	DIAB (Data Intdustrier AB)
00-00-22	Visual Technology
00-00-2A	TRW
00-00-4B	APT
00-00-5A	S & Koch
00-00-5E	U.S. Department of Defense
00-00-65	Network General
00-00-6B	MIPS
00-00-77	MIPS
00-00-7A	Ardent
00-00-81	SynOptics
00-00-89	Cayman Systems
00-00-93	Proteon
00-00-94	Asante
00-00-9F	Ameristar Technology
00-00-A2	Wellfleet
00-00-A3	Network Application Technology
00-00-A6	Network General
00-00-A7	NCD (X-terminals)
00-00-A9	Network Systems
00-00-AA	Xerox
00-00-B3	CIMLinc
00-00-B7	Dove (Fastnet)
00-00-BB	Tri-Data
00-00-BC	Allen-Bradley
00-00-C0	Western Digital
00-00-C5	Farallon
00-00-C6	HP Intelligent Networks Operation
00-00-C8	Altos

Ethernet Manufacturer Designators, *continued*

00-00-C9	Emulex (Terminal Servers)
00-00-D7	Dartmouth College (NED Router)
00-00-DD	Gould
00-00-DE	Unigraph
00-00-E2	Acer Counterpoint
00-00-EF	Alantec
00-01-02	BBN
00-17-00	Kabel
00-80-06	Nuvotech
00-80-19	Novell/FastPath
00-80-2D	Xylogics
00-80-35	Technology Works
00-80-8C	Frontier Software Development
00-80-D3	Shiva/FastPath
00-AA-00	Intel
00-DD-00	Ungermann-Bass
00-DD-01	Ungermann-Bass
02-60-86	Satelcom MegaPac (UK)
02-60-8C	3Com (IBM PC; Imagen; Valid; Cisco; Apple)
AA-00-04	DecNet
02-CF-1F	CMC (Masscomp; Silicon Graphics; Prime EXL)
03-54-67	MICOM/Interlan
03-72-90	BBN internal usage (not registered)
08-00-02	Bridge
08-00-03	ACC (Advanced Computer Communications)
08-00-05	Symbolics
08-00-07	Apple
08-00-08	BBN
08-00-09	Hewlett-Packard
08-00-0A	Nestar Systems
08-00-0B	Unisys
08-00-10	AT&T
08-00-11	Tektronix
08-00-14	Excelan (BBN Butterfly, Masscomp, Silicon Graphics)

Ethernet Manufacturer Designators, *continued*

08-00-17	NSC
08-00-1A	Data General
08-00-1B	Data General
08-00-1E	Apollo
08-00-20	Sun
08-00-22	NBI
08-00-25	CDC
08-00-26	Norsk Data
08-00-27	PCS Computer Systems GmbHw
08-00-28	TI Explorer
08-00-2B	Digital Equipment
08-00-2E	Metaphor
08-00-2F	Prime Computer Prime 50-Series LHC300
08-00-36	Intergraph CAE stations
08-00-37	Fujitsu-Xerox
08-00-38	Bull
08-00-39	Spider Systems
08-00-41	DCA Digital Communications Associates
08-00-46	Sony
08-00-47	Sequent
08-00-49	Univation
08-00-4C	Encore
08-00-4E	BECC
08-00-56	Stanford University
08-00-5A	IBM
08-00-67	Comdesign
08-00-68	Ridge
08-00-69	Silicon Graphics
08-00-6A	AT&T
08-00-6E	Excelan
08-00-7C	DDE (Danish Data Elektronik A/S)
08-00-7C	Vitalink TransLAN III
08-00-80	XIOS
08-00-86	Imagen/QMS

Ethernet Manufacturer Designators, *continued*

08-00-87	Xyplex
08-00-89	Kinetics
08-00-8B	Pyramid
08-00-8D	XyVision
08-00-90	Retix Inc. Bridges

Ethernet Protocol Designators

Designator	Protocol
0000–05FF	IEEE802.3 Length Field
0101–01FF	Experimental
0200	Xerox PUP (Conflicts with 802.3 length field range)
0201	Xerox PUP Address Translation (conflicts...)
0600	Xerox NS IDP (XNS)
0800	DOD Internet Protocol (IP)
0801	X.75 Internet
0802	NBS Internet
0803	ECMA Internet
0804	CHAOSnet
0805	X.25 Level 3
0806	Address Resolution Protocol (ARP) (used by IP and CHAOS)
0807	Xerox NS (XNS) Compatibility
081C	Symbolics Private
0888–088A	Xyplex
0900	Ungermann-Bass network debugger
0A00	Xerox IEEE802.3 PUP (was 0200, see above)
0A01	Xerox IEEE802.3 PUP Address Translation (was 0201, see above)
0BAD	Banyan Systems
1000	Berkeley Trailer negotiation
1001–100F	BerkeleyTrailer encapsulation for IP
1600	Valid System protocol
4242	PCS Basic Block Protocol
5208	BBN Simnet Private

Ethernet Protocol Designators, *continued*

Designator	Protocol
6000	DEC unassigned, experimental
6001	DEC Maintenance Operation Protocol (MOP) Dump/Load Assistance
6002	DEC Maintenance Operation Protocol (MOP) Remote Console
6003	DECNET Phase IV, DNA Routing
6004	DEC Local Area Transport
6005	DEC diagnostic protocol
6006	DEC customer protocol
6007	DEC Local Area VAX Cluster (LAVC), System Communication Architecture (SCA)
6008–6009	DEC unassigned
6010–6014	3 Com Corporation
7000	Ungermann-Bass download
7002	Ungermann-Bass diagnostic/loopback
7020–7029	LRT
7030	Proteon
7034	Cabletron
8003	Cronus VLN
8004	Cronus Direct
8005	HP Probe protocol
8006	Nestar
8008	AT&T
8010	Excelan
8013	Silicon Graphics diagnostic
8014	Silicon Graphics network games
8015	Silicon Graphics reserved
8016	Silicon Graphics Xerox NS (XNS) NameServer, bounce server
8019	Apollo DOMAIN
802E	Tymshare
802F	Tign, INC.
8035	Reverse Address Resolution Protocol (RARP)
8036	Aconic Systems
8038	DEC LanBridge Management
8039–803C	DEC unassigned

Ethernet Protocol Designators, *continued*

Designator	Protocol
803D	DEC Ethernet SCMA/CD Encryption Protocol
803E	DEC unassigned
803F	DEC LAN Traffic Monitor Protocol (LTM)
8040–8042	DEC unassigned
8044	Planning Research Corp.
8046–8047	AT&T
8049	ExperData
805B	Stanford V Kernel, experimental
805C	Stanford V Kernel, production
805D	Evans & Sutherland
8060	Little Machines
8062	Counterpoint Computers
8065–8066	University of Massachusetts at Amherst
8067	Veeco Integrated Automation
8068	General Dynamics
8069	AT&T
806A	Autophon
806C	ComDesign
806D	Compugraphic Corporation
806E–8077	Landmark Graphics Corporation
807A	Matra
807B	Dansk Data Elektronik A/S
807C	Merit Internodal (University of Michigan)
807D–807F	Vitalink Communications
8080	Vitalink TransLAN III Management
8081–8083	Counterpoint Computers
809B	EtherTalk (AppleTalk over Ethernet)
809C–809E	Datability
809F	Spider Systems Ltd.
80A3	Nixdorf Computers
80A4–80B3	Siemens Gammasonics Inc.
80C0–80C3	Digital Communications Associates Inc. (DCA)
80C6	Pacer Software

Ethernet Protocol Designators, *continued*

Designator	Protocol
80C7	Applitek Corporation
80C8–80CC	Intergraph Corporation
80CD–80CE	Harris Corporation
80CF–80D2	Taylor Instrument
80D3–80D4	Rosemount Corporation
80D5	IBM SNA Services over Ethernet
80DD	Varian Associates
80DE–80DF	Integrated Solutions Transparent Remote File System (TRFS)
80E0–80E3	Allen-Bradley
80E4–80F0	Datability
80F2	Retix
80F3	AppleTalk Address Resolution Protocol (AARP)
80F4–80F5	Kinetics
80F7	Apollo Computer
80FF–8103	Wellfleet Communications
8107	Symbolics Private
8108	Symbolics Private
8109	Symbolics Private
8130	Waterloo Microsystems Inc.
8131	VG Laboratory systems
8137	Novell, Inc.
8139–813D	KTI
814C	SNP over Ethernet
9000	Loopback (Configuration Test Protocol)
9001	Bridge Communications Xerox NS (XNS) Systems Management
9002	Bridge Communications TCP/IP Systems Management
9003	Bridge Communications
FF00	BBN VITAL-LanBridge cache wakeups

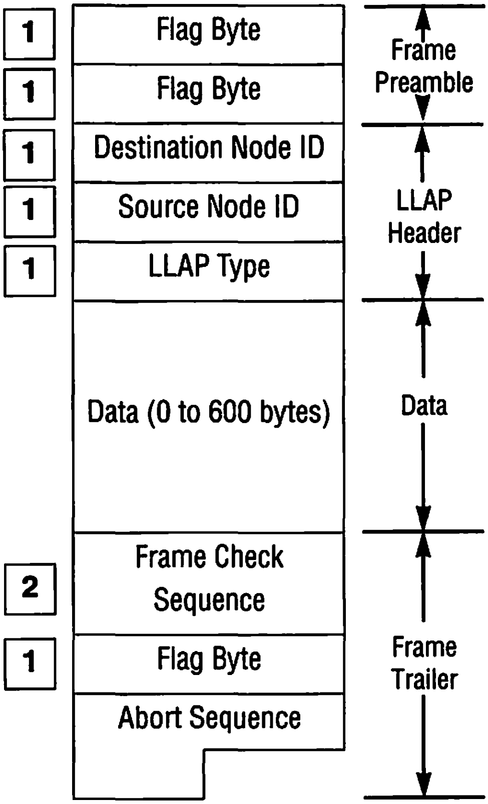
APPENDIX D

Packet Glossary

LAP Headers

LocalTalk Link Access Protocol (LLAP), EtherTalk Link Access Protocol (ELAP)

LAP headers specify the source and destination devices on a single network.

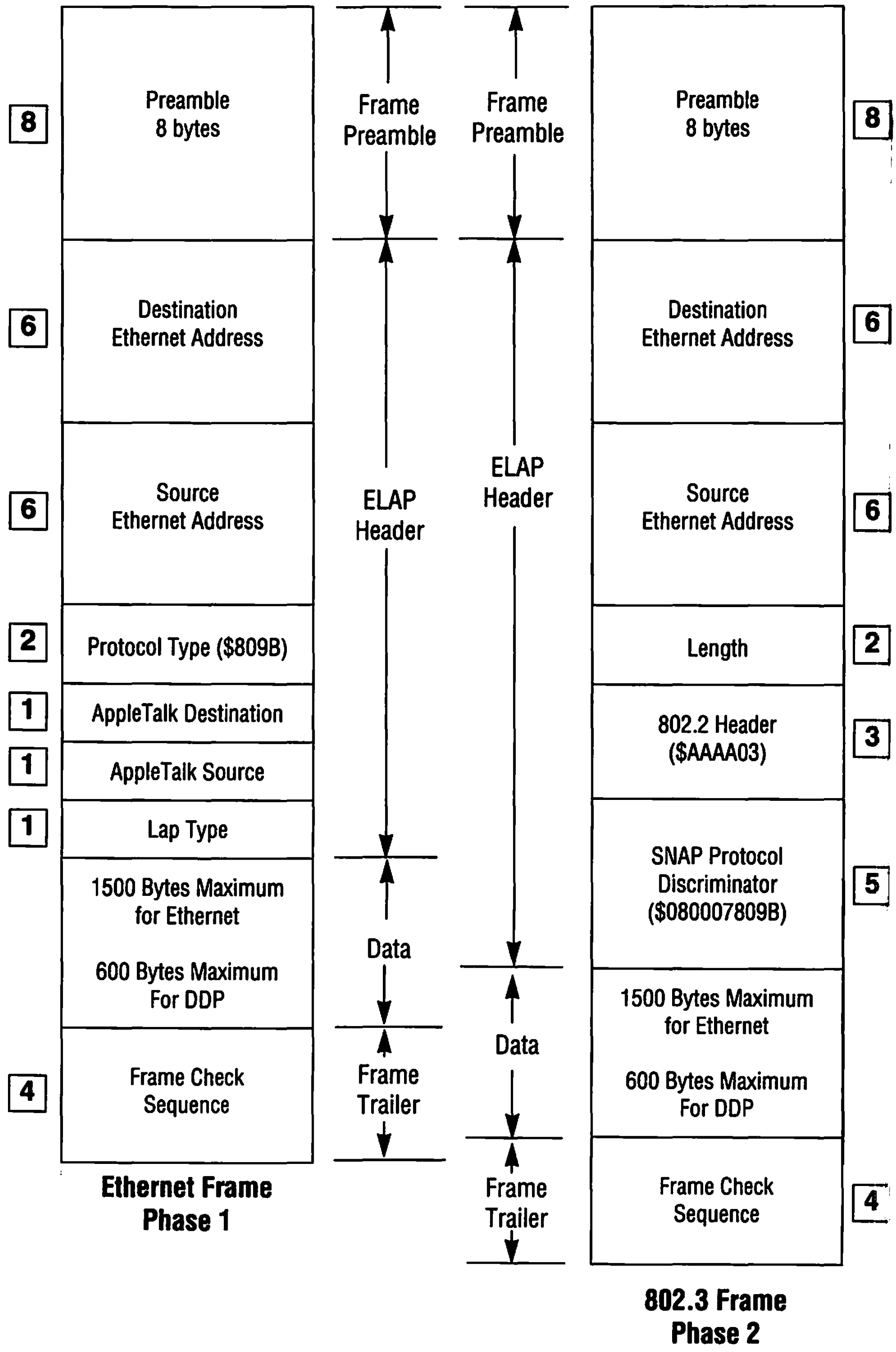


LocalTalk Frame

Troubleshooting Tips

1. LAP Errors can be useful for diagnostic purposes. Refer to Chapter 6, Data Link Troubleshooting, for more details.

2. All traffic analysis software for AppleTalk examines only the LAP addresses, which are the source and destination nodes on this particular link. The ultimate source and destination devices are identified in the DDP portion of the AppleTalk packet.


8
Preamble
8 bytes
6
Destination
Ethernet Address
6
Source
Ethernet Address
2
Protocol Type ($809B)
1
AppleTalk Destination
1
AppleTalk Source
1
Lap Type
1500 Bytes Maximum
for Ethernet
600 Bytes Maximum
For DDP
4
Frame Check
Sequence
Ethernet Frame
Phase 1
Frame
Preamble
ELAP
Header
Data
Frame
Trailer
Frame
Preamble
ELAP
Header
Data
Frame
Trailer
Preamble
8 bytes
8
Destination
Ethernet Address
6
Source
Ethernet Address
6
Length
2
802.2 Header
($AAAA03)
3
SNAP Protocol
Discriminator
($080007809B)
5
1500 Bytes Maximum
for Ethernet
600 Bytes Maximum
For DDP
Frame Check
Sequence
4
802.3 Frame
Phase 2

AppleTalk Address Resolution Protocol (AARP)

AARP maps two different address systems to each other. Example: AppleTalk and Ethernet.

	Frame Header
2	Hardware Type Ether = 1, Token = 2
2	Protocol Type AppleTalk = $809B
1	Hardware Length = 6
1	Protocol Length = 4
2	AARP Function Req = 1, Rsp = 2, Prb = 3
6	Source Hardware Address
4	Source AppleTalk Address
6	Destination Hardware Address
4	Destination AppleTalk Address

Troubleshooting Tips

1. Requests and Responses are seen in pairs and are sent just before the requesting node has something to ask or tell the other device.

2. Unanswered Probes are normal; unanswered Requests sometimes indicate trouble—a crashed device, for example, which occurs when the requesting device wants to send a packet to an AppleTalk address that no longer exists.

3. While AARP Requests and Responses are common (especially when new devices come onto the network), if a particular device is sending a large number of AARP Requests that receive Responses, ask the manufacturer of the device sending the AARP Requests whether the AARP cache parameters can be modified by either cache size (how many AARP addresses the AARP cache holds) or AARP timer (how long AARP entries remain in the cache). If the device is sending a large number of AARP Requests that do not receive Responses, there is probably a bug in its software. If the device sending the large number of Requests is a router, try to determine whether it is attempting

to locate a device on behalf of a node in another network. If so, that node probably has a bug in its software. If not, the bug is probably in the router's software.

4. A nonrouting device typically sends 10 Probes over a two-second period when it starts up. On most Ethernet networks, this is entirely adequate. On bridged Ethernets with slow links between segments, this may not be long enough to search for duplicate addresses. Look for an AARP Response that arrives after the two-second period.

Table D-1. AARP Functions

Code	Name	English Translation	Usage
1	Request	Q: Who has this AppleTalk Address?	When a node needs to send to an AT address that is not in its AARP cache
2	Response	A: I have this AppleTalk Address.	Response to either a Request or a Probe
3	Probe	Q: Does anyone have this AppleTalk Address?	When a node boots up to test its address for uniqueness; equivalent to LLAP NodeEnq packet

Datagram Delivery Protocol (DDP)

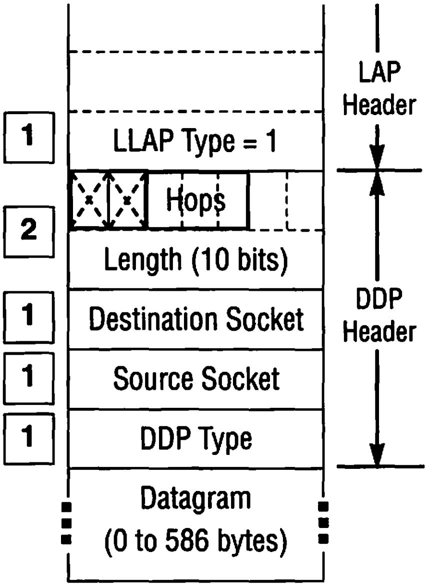
DDP specifies full source and destination addresses—network, node, and socket.

Troubleshooting Tips

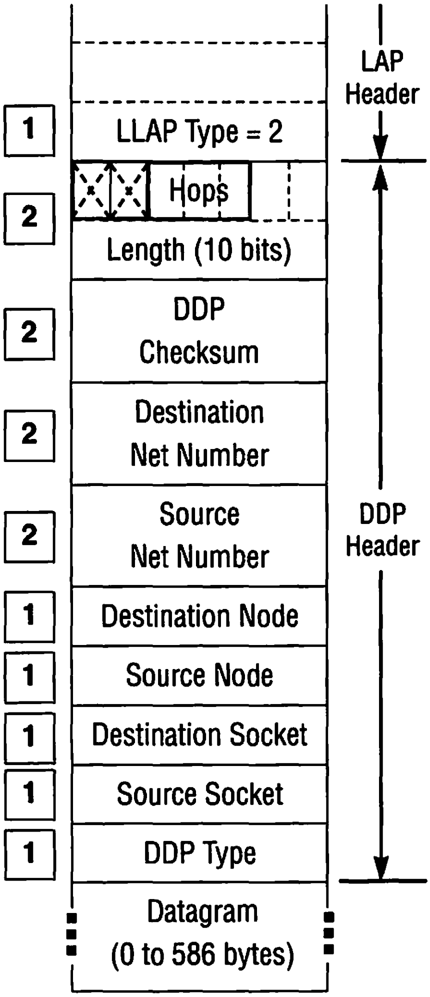
1. The DDP part of a packet tells the ultimate source and destination of the packet; the LAP portion tells just the source and destination on a particular link.

2. High hop counts (higher than the hop width of your network) indicate routing configuration problems.

3. The DDP Checksum is often 0, which means "not used;" see discussion of Network Layer problems in Chapter 5.



Short Header destination and source share same network



Long Header destination and source on different networks

Table D-2. DDP Socket Designations

1 – 127	Static Sockets	Apple Designated
1 – 63	Reserved by Apple	1 – Routing
64 – 127	Experimental Use Only	2 – Name Info
128 – 254	Dynamic Sockets for applications and other system extensions	4 – Echo Protocol
		6 – Zone Info
		8 – SNMP

Table D-3. DDP Types

Type	Protocol	Uses
1	RTMP	Routing Data
2	NBP	All Name Functions—Br-Req, LkUp, Reply, Fwd-Req
3	ATP	AppleShare, Printing, Database, some ZIP functions
4	EP	Echo Testing, to set custom connection timers
5	RTMP	RTMP Requests
6	ZIP	ZIP Queries and Replies
7	ADSP	Meeting Maker, Timbuktu, Apple Events, and so on
8	SNMP	Simple Network Management Protocol

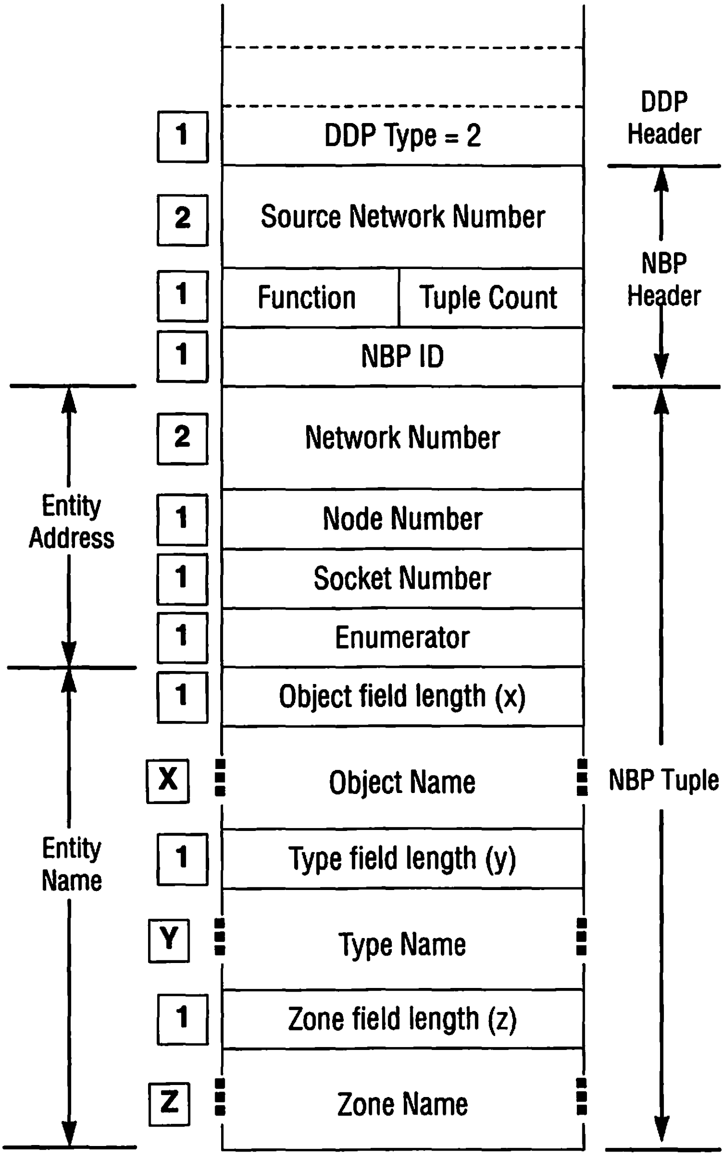
Name Binding Protocol (NBP)

NBP performs name-to-address mapping functions.

Troubleshooting Tips

1. NBP searches are routinely performed at startup when a process wants to check the uniqueness of a service name that it intends to register.

2. AppleTalk services such as LaserWriters, file servers, and mail servers typiclly register a Session Listening Socket (SLS) that allows other devices to contact them by name and log on. Subsequent communication usually takes place on other sockets that are identified during the login process.



Special Characters (Wildcards)

1. In the Name and Type fields "=" matches any name or type.
2. In the Zone field, "*" means "this zone," and matches any zone name.

3. In all NBP packets except the LkUp-Reply, the entity address is the address of the device that initiated the request.

4. For multi-purpose serves, the socket address can provide a good way of filtering the particular service or session that you're troubleshooting.

5. Many AppleTalk services register an entity name with the zone field "*" instead of a particular zone name.

6. The NBP ID is useful when a device is searching for multiple entity names. Each search will have a different NBP ID so the node can tell which reply resulted from which search.

Table D-4. NBP Functions

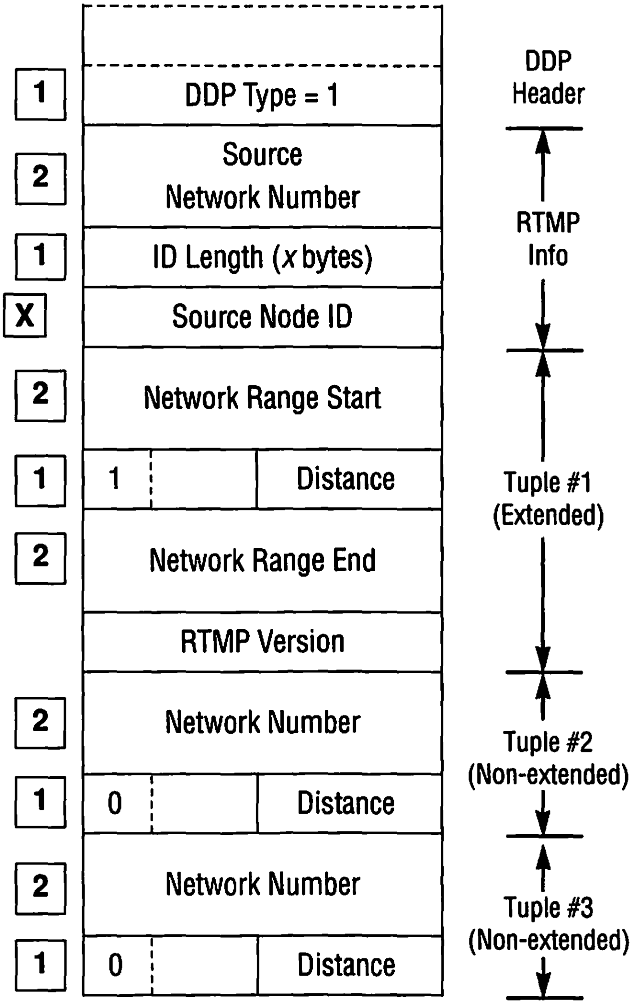
Code	Function	English Translation (for the entity name Dave:LaserWriter@Sales)
1	Broadcast-Request	Router, tell the LaserWriter named Dave in the Sales zone to contact me.
2	LkUp	If any device has a socket registered with the service name Dave, the LaserWriter in the Sales zone, report to the address listed in this packet.
3	LkUp-Reply	I'm Dave the LaserWriter in the Sales zone and my address is listed in this packet.
4	FwdReq	Next Router, please broadcast a LkUp for Dave the Laser-Writer in the Sales zone to the network specified using the address listed in this packet.

Routing Table Maintenance Protocol (RTMP)

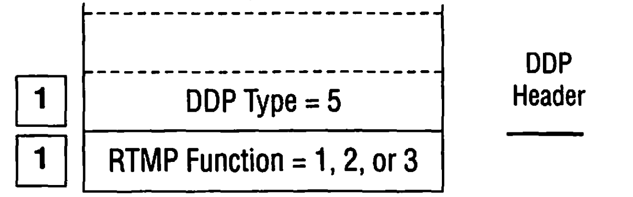
RTMP is the protocol for routers to learn and exchange routing information.

Troubleshooting Tips

1. Using a protocol analyzer, filter on RTMP Response packets. The number of nodes that you see sending these packets tells you how many routers are on your network.



RTMP Response Packet



RTMP Request or RTMP Route Data Request

2. Most Phase 1 devices, including devices on LocalTalk networks, use the RTMP Request packet (Function 1) to learn what network they're on. Most Phase 2 devices use the ZIP GetNetInfo packet for this purpose. During the startup sequence, this is also how a device learns whether there's a router on the network, which influences how it conducts its NBP searches prior to registering its sockets.

3. Nonrouting devices keep an RTMP Stub in the memory that contains the network and node address of the last rou;ter they heard from. On a network with many routers, the address in this stub constantly changes. In Phase 1, whenever a ndoe needs a router fro any purpose, the router used is the one currently identified in its RTMP Stub. In Phase 2, the router in the RTMP Stub is only used for nontransmission purposes, such as when a Mac's Chooser needs to obtain a zone list. For transmission, the node users a "best router" algorithm (explained elsewhere).

Table D-5. RTMP Request Functions

Code	Name	English Translation
1	Request	Q: What network am I on?
2	Route Data Request (Split)	Q: What networks do you have connected to the network ports other than the one I'm connected to?
3	Route Data Request (No Split)	Q: What are all the networks that you know?

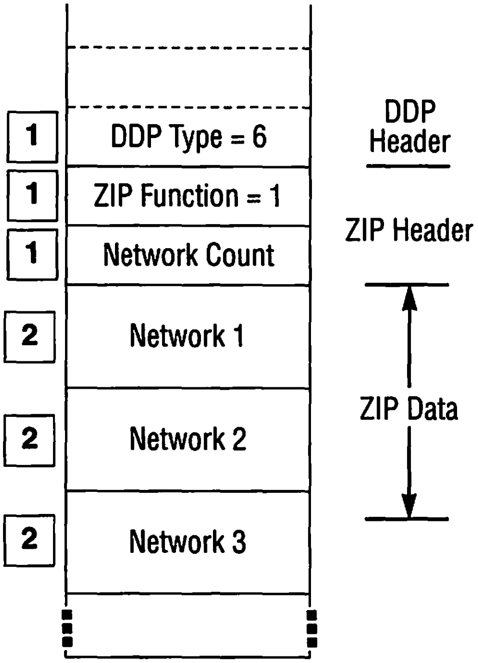
Zone Information Protocol (ZIP)

ZIP Query and Reply—Router-to-Router Communication

ZIP maps networks to zones. This format is used by a router to get zone information from another router.

Troubleshooting Tips

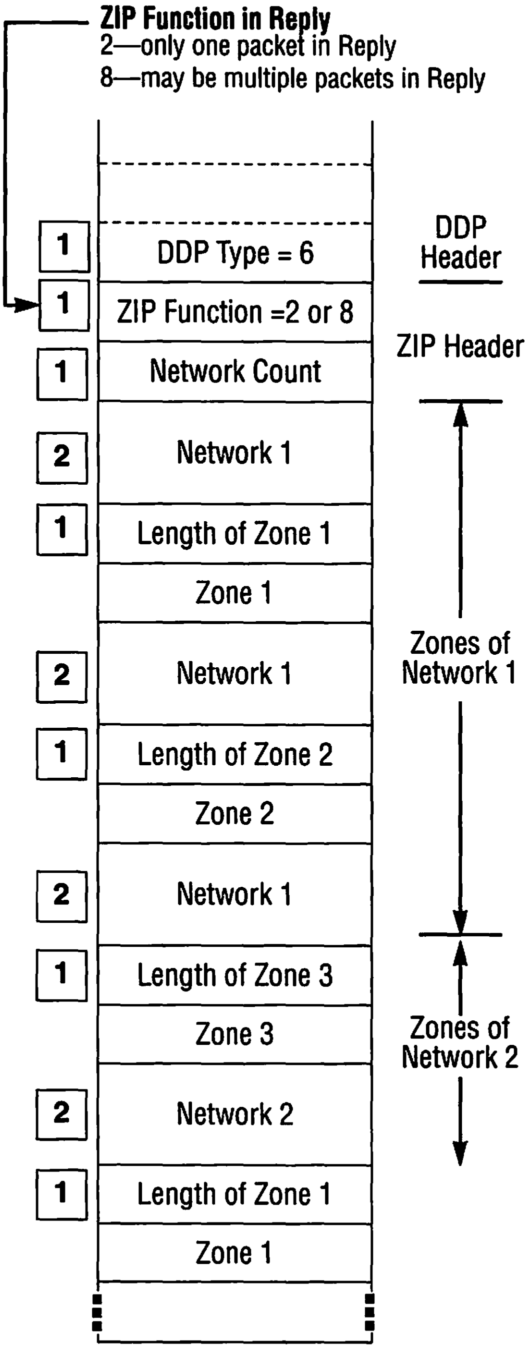
1. ZIP Queries and Replies should only be seen when routers start up; any other time means trouble. Constant flow of ZIP Queries *with no Replies* means that no zone was entered for a network. Check the configuration of the object of the Query.



ZIP Query

2. Constant flow of ZIP Queries *with Replies* means conflicting network information. Check configurations of the routers involved.

3. Sending a ZIP Query to all routers on a network (using PacketSend) should get identical Replies from all routers. Check the length of packets to see whether all routers have the same number and length of zones. Use GetNetInfo to check spelling of zone names.



ZIP Reply

ZIP ATP Functions—Node-to-Router Transactions

ZIP maps networks to zones. These formats are used by nonrouting nodes to get zone information from a router.

Table D-6.

Transaction	Query	Reply	Typical Use
GetZoneList GetZoneListReply	What zones are reachable in this internet?	The following zones are reachable: <Zone1>, <Zone2>, and so on.	The Chooser (or similar process) asks a router for the list of zones to display.
GetMyZone GetMyZoneReply	What zone am I in? (Phase 1 networks only)	You're in the <Zone Name> zone.	The Chooser of a LocalTalk device needs to know which zone in the list to highlight.
GetNetInfo GetNetInfoReply	Is Zone <Zone Name> a valid zone for the network I'm connected to? (Phase 2 networks only)	1. Zone <Zone Name> is a valid zone for your network, and corresponds to the network address range of <start> to <end>. *Or* 2. Zone <supplied zone name> is not a valid zone for this network. The default zone for this net is <default zone name>, which corresponds to the network address range of <start> to <end>.	When returning to a Phase 2 network, a device (routers and nonrouters alike) confirms its saved zone name using this transaction. Also used to confirm a home zone immediately after its selection and to become informed of its zone-specific multicast address.
GetLocalZones GetLocalZonesReply	What zones are available on my network? (Phase 2 networks only)	The following zones are associated with your network: <Zone1>, <Zone2>, and so on.	When a user chooses a home zone using the Network Control Panel.

Troubleshooting Tips

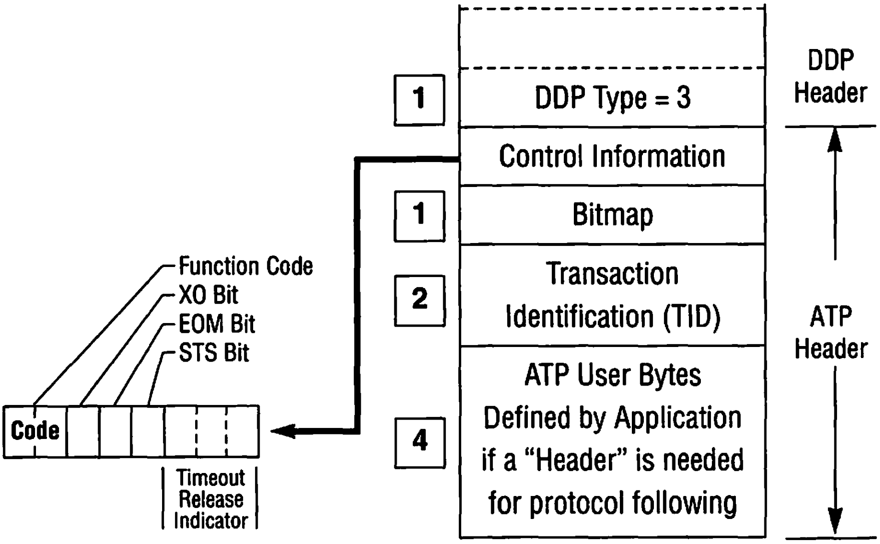
1. All of these transactions are sent to the router recorded in the device's RTMP stub—the last router that a device heard from. Exception: some routers broadcast a GetNetInfo on startup to check for conflicts with their configuration information.

2. A node choosing a home zone on a network with multiple routers sends GetLocalZones to the router currently in the node's RTMP Stub. The GetNetInfo is used to confirm the zone name and retrieve the zone multicast address. The GetNetInfo is typically broadcast and the node uses the information from the router that responds first. When the GetNetInfo indicates an invalid zone name, you have a router error. Users see a dialog box telling them they were placed in the router's default zone instead of the zone that they selected just a few moments before.

3. For the GetZoneList and GetLocalZones transactions, if the zone list does not fit into one packet, requests and replies are traded until the zone list has been completely transferred. The Start Index in the query tells the router where to begin the list in the reply. Zones are typically reported not alphabetically, but in the order that the router learned about them.

AppleTalk Transaction Protocol (ATP)

ATP manages DDP's delivery and associates commands with replies, questions with answers, and so on.


DDP Header
1
DDP Type = 3
Control Information
1
Bitmap
2
Transaction Identification (TID)
4
ATP User Bytes Defined by Application if a "Header" is needed for protocol following
ATP Header
Function Code
XO Bit
EOM Bit
STS Bit
Code
Timeout Release Indicator

Troubleshooting Tips

1. When the STS bit is set in a response packet, it implies that the memory is low in the node sending the response. After getting an acknowledgment of the successful delivery of the packets, it can clear them from its "send buffer," making room in memory for other packets.

2. In some protocols, such as ASP, the protocol analyzer may not know which decode (CmdReply, GetStatusReply, and so on) to apply to a Response Packet, and you must tell the analyzer which one is appropriate.

3. Many third-party applications use AppleTalk standard protocols through the Transport Layer, but use proprietary, application-specific protocols for the Session Layer and above. You cannot decode these protocols explicitly, but you can sometimes make reasonable guesses as to what functions they are performing (session maintenance, status retrieval, data transfer, and so on).

4. The Transaction ID provides a sequencing mechanism; each packet and each transaction is numbered. Typically, the Session and Presentation Layer protocols have their own sequencing mechanism in addition to ATP. Some exchanges between nodes may involve several transactions.

Table D-7. ATP Functions and Bitmap Meaning

Code	Name	Bitmap Purpose
1	Request	Tells how many buffers are being reserved for the answer (maximum 8). In response to an STS, the Request tells which packets have been delivered.
2	Response	Tells the sequence number of the packet being delivered.
3	Release	The Transaction release does not have a bitmap.

AppleTalk Data Stream Protocol (ADSP)

ADSP carries communication between two applications in continuous data streams.

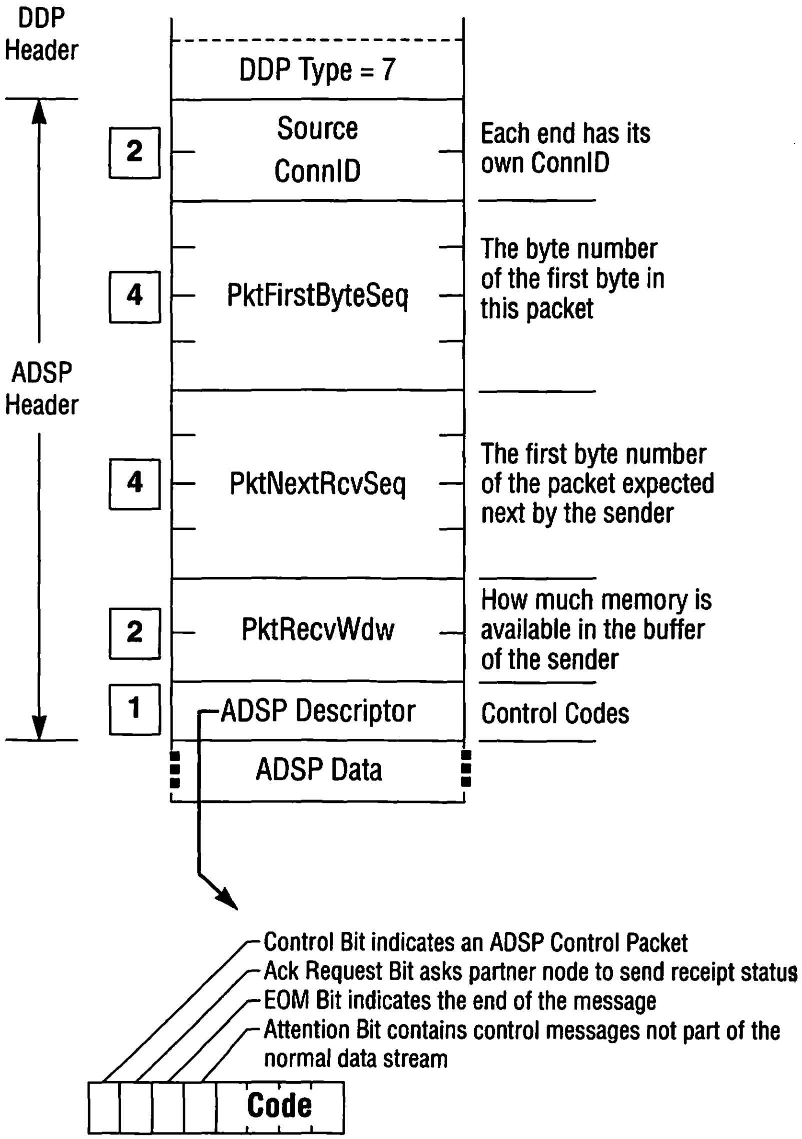
Table D-8. ADSP Control Codes

Code	Function	Meaning
1	Probe or Acknowledgment	Probes ask for confirmation of data delivery; acknowledgments contain data receipt information.
2	Open Connection Request	Establishes source connection ID and tells partner which socket to use for responses.
3	Open Connection Acknowledge	Agrees to open a connection and supplies the parameters listed in Code 2.
4	Open Request and Acknowledge	Combination of Codes 2 and 3—in ADSP, both ends must explicitly open the connection.
5	Open Connection Denial	When an application declines a connection request.
6	Close Connection Advice	Just as each node must open the connection, each must also close the connection.
7	Forward Reset	Asks partner node to disregard all bytes after FirstByteSeq.
8	Forward Reset Acknowledge	Acknowledges receipt and compliance with Forward Reset Command.
9	Retransmit Advice	Asks partner to retransmit beginning with the byte in FirstByteSeq.

Troubleshooting Tips

1. Watch for Control Codes 7–9; they often indicate a problem in delivery.

2. Within each data stream, FirstByteSeq should rise steadily as the connection progresses. When it backtracks, data is being retransmitted. Check this when the data flow is heavy in one direction. Look at the last packet in a block of packets. The sender usually sets the Ack Request bit in this packet to find out whether the receiver has been getting all the information. Compare the FirstByteSeq of this data packet witht the NextRcvSeq of the acknowledgment packet that the receiver must send to comply with the Ack Request. The receiver should indicate a NextRcvSeq equal to the sender's previous FirstByteSeq plus the data transferred in the last send packet. If the NextRcvSeq is a lower number, it means that some of the data was missed.


DDP Header
DDP Type = 7
ADSP Header
2
Source ConnID
Each end has its own ConnID
4
PktFirstByteSeq
The byte number of the first byte in this packet
4
PktNextRcvSeq
The first byte number of the packet expected next by the sender
2
PktRecvWdw
How much memory is available in the buffer of the sender
1
ADSP Descriptor
Control Codes
ADSP Data
Control Bit indicates an ADSP Control Packet
Ack Request Bit asks partner node to send receipt status
EOM Bit indicates the end of the message
Attention Bit contains control messages not part of the normal data stream
Code

As a group, applications that use ADSP do not handle packets delivered out of sequence as well as applications that use ATP. On internets that have simultaneous redundant pathways, you may experience slow performance when packets arrive at a node out of sequence. The reason is that there is no way for ADSP to ask for "all of the bytes between 1758 and 2024." ADSP can only ask for "all of the bytes beginning with 1758" using the Retransmit Advice Control packet. Some applications using ADSP can handle out-of-sequence packets better than others because they have a greater ability to "read ahead," meaning they will accept packets later in the stream (higher FirstByteSeq) and then wait for packets earlier in the stream to arrive later. This is not a required feature for applications using ADSP.

Glossary

AC. Alternating Current. An electrical transmission system in which the direction of current flow alternates on a periodic basis.

Accelerator. A hardware addition to an existing computing device that increases the computer's processing speed and capabilities.

Access. A computing device's ability to use data or resources beyond its native capabilities.

Access Method. The type of Media Access Control (MAC) method that a node uses to gain control of a network.

Accuracy. How closely a test instrument's measurements compare to a standard value, usually expressed as a percentage of the value measured.

ACK. Acknowledgment. An electronic signal or message that confirms the receipt of information.

Active Star. A LocalTalk implementation where a multiport repeater is used in a physical star topology.

Adapter. Hardware that allows a computing device physical access to a network.

Address. A numerical designation that uniquely refers to a specific communication entity. *See also* Hardware Address, Protocol Address.

Address space. The range of possible unique addresses allowed by an addressing scheme. Using a binary numbering system, the address space of an n-bit address is 2^n addresses.

Address resolution. When two addressing systems refer to the same entity, the process of translating or expressing the address of an entity in one system to the equivalent address of the same entity in the second system. Example: Translating an AppleTalk address to an Ethernet address.

AFP. AppleTalk Filing Protocol. The Apple proprietary specification for a set of commands and data structures that represent the commands and data structures of a computer's native file system.

Agent. 1. An active process in a computer that is responsible for a certain type of activity when demanded by an outside entity. 2. In SNMP, the active process in a computing device that is responsible for determining the parameters defined in the MIB (Management Information Base) and reporting them on demand to a Console.

Algorithm. A set of rules and decision structures for actions in a specifically defined set of circumstances.

Alias. In Apple's System 7, a file whose sole purpose is to represent another file.

Alphanumeric. A group of printable characters that includes the letters of the alphabet in both uppercase and lowercase, numerals, and a limited group of additional symbols and punctuation marks.

Ambient. A set of conditions that exist independently of the system of interest.

Amp. Ampere. A standard unit of measurement for electrical current flow.

Amplitude. In the terminology of wave motion, the height of the wave. Amplitude is usually measured from a reference point of 0. In electrical waves, amplitude is typically expressed in volts.

Analog. A system or component that uses a system of measurement, response or storage in which values are expressed as a magnitude using a continuous scale of measurement.

Anomaly. An unusual instance or circumstance.

Apple. Apple Computer, Inc.

AppleShare. An application published by Apple that allows a Macintosh to be a file server using the AFP protocol.

AppleShare PC. An obsolete Apple application that allowed a DOS computer with the appropriate hardware adapter to use AppleTalk protocols and provide file and print services. Superseded by Farallon's PhoneNET Talk.

AppleTalk. 1. Apple's proprietary network architecture. 2. The protocols, applications, networks, and services included in Apple's network architecture. 3. A common but improperly used synonym for LocalTalk (now obsolete by Apple's change in definitions).

Application. An independently executable set of algorithms and data structures that perform a specific set of functions.

API. Application Programming Interface. A set of tools and procedures provided by the programmer of an application so that other programmers can control, exchange data with, or extend the functionality of an application.

Architecture. The total of all the specifications, protocols, and implementations that define a particular networking system.

Archive. A storage of infrequently used or historical data.

ASCII. A standard 7-bit character system that includes the alphanumeric characters and printer control codes. Corresponds to the first 128 characters of the Macintosh character set.

Asynchronous. A system of communication in which each discrete delivery of information establishes its own timing pulse rather than conforming to the timing pulse of previous deliveries.

Asymmetry. In networking, a system in which the relationship between two entities is inherently unequal, with each entity restricted to a set of operations and prerogatives defined by its role in the relationship.

AT command set. In modems, a set of commands that control the modem or alter its characteristics. Originally developed by Hayes, the AT command set is now an industry standard.

Attenuation. A loss in the amplitude or strength of a signal due to an interaction with the signal's media.

Back end processor. A computer running an application that supplies data to other computers on demand, but has no user interface.

Backbone. 1. In an internet, a central network that provides the pathway for other networks to communicate. 2. A LocalTalk topology consisting of several wall outlets linked together with UTP (the trunk). From the wall outlets, short pieces of patch cord (stubs) connect the LocalTalk adapters.

Background task. A task that is executing on a computer while another task or application is displaying its user interface.

Backplane. The communication channels of a single computer's architecture.

Backup. A copy of a set of files made for replacement purposes in case the original set is damaged or lost.

Backward-compatible. An upgraded component of a computing system that can be used interchangeably with its previous version.

Balun. A device that links together dissimilar wire types and attempts to minimize any negative effects to the signal that would otherwise result from the dissimilarity.

Band. In analog communications, the range of frequencies over which a communication system operates.

Bandwidth. In analog communications, the difference between the highest and lowest frequencies available in the band. In digital communications, bandwidth is loosely used to refer to the information carrying capacity of a network or component of a network.

Base address. The lowest address available in an address range.

Baseband. A communication system in which only one signal is carried at any time.

Baud rate. The number of data symbols exchanged per second.

Benchmark. A test performed to compare a computer process among different sets of circumstances.

Binary. 1. A numerical system using 2 as its base. 2. Data that is encoded or presented in machine-readable form (1s and 0s).

Bit. The basic unit of data representation in digital computers. A memory location that can have one of two values.

Bit map. A data structure that uses bits to represent the attributes of an object that is not character-based.

Bit pattern. A sequence of bits that has a specific purpose or meaning.

Bit rate. The rate at which bits are transmitted or received during communication, expressed as the number bits in a given amount of time, usually 1 second.

Black box. A device whose external functions are known, but whose internal processes are not known or understood.

Block. The basic unit of storage on a computer disk.

BNC. 1. The locking connector type used in 10BASE2 (Thin Ethernet). 2. Any connector similar to the type used by 10BASE2 for CATV, and other electronic uses.

Board. A printed circuit and the substrate on which it lies.

Boot. The computer's startup operation.

Boot drive. The disk that contained the computer's startup instructions when it started operation during the current session.

BPS. Bits per second, a commonly used measure of the rate of data transmission that specifies the number of bits that are transmitted in one second. May be prefixed with multipliers such as K, M, and G which indicate rates of thousands, millions, and billions of bits per second, respectively.

Bridge. A Data Link Layer device that acts to limit traffic between two network segments by filtering the data between them based on hardware address. In many bridges, the filtering criteria may be expanded by the network manager.

Broadband. A transmission system capable of carrying many channels of communication simultaneously by modulating them on one of several carrier frequencies.

Broadcast. An information transmission that is intended to be interpreted by all entities capable of receiving it.

Brouter. A device that incorporates the functionality of a bridge and a router in a single unit.

Buffer. A temporary storage area for information.

Bug. A flaw in an algorithm.

Bundled. Additional applications or capabilities included in the sale or delivery of a separate computing component.

Bus. A type of network topology in which nodes are connected along a continuous path that is not a closed circuit.

Byte. A group of 8 bits.

Cable. The transmission media of a network.

Cache. A group of memory locations set aside for temporary storage of data, especially frequently used data or data needing high-speed retrieval by the CPU.

Card. A circuit board that plugs into a computer's bus to extend the computer's capability.

Case insensitive. A system or application that does not differentiate between upper- and lowercase letters.

Case sensitive . A system or application that differentiates between upper- and lowercase letters.

CDEV. A Control Panel Device in Macintosh System 6 and earlier (obsolete).

Character. 1. A symbol such as a letter, number or punctuation mark that can be arranged to represent higher units of meaning, such as words and sentences. 2. The group of bits that represents such a symbol.

Cheapernet. A synonym for 10BASE2 (archaic).

Checksum. The result of a mathematical operation that uses the binary representation of a group of data as its basis, usually to check the integrity of the data.

Circuit. 1. Any electrical pathway. 2. An arrangement of electrical and electronic devices and the conductive paths between them.

Client-Server. A type of relationship between two computers in which the client computer typically drives the relationship and uses a resource of the server computer.

Clock. 1. A component in a computer that provides a timing pulse for other components. 2. The timing pulse of a network transmission.

Coaxial cable. An electrical cable in which the conductors share a common axis.

Collision avoidance. A Media Access Control (MAC) method in which any node may take control of the network after taking certain steps to ensure that the cable is not in use or about to be used by another node.

Collision detection. A MAC method in which any node may take control of the network when it is not in use by another node.

Component. An indivisible unit of functionality, usually embodied in hardware.

Compression. An alteration performed on a unit of information intended to increase its density during storage or transmission.

Concentrator. A multiport repeater.

Conductor. The current-carrying component of a transmission cable, typically a copper wire.

Connection-oriented. In data networks, a type of computer relationship during which the network equipment constructs a circuit between the two devices. Once the circuit is established, the devices pass information back and forth through the circuit without regard to their physical addresses. The circuit may be physical or virtual.

Connectionless. The opposite of connection-oriented, a type of relationship between two devices where each information packet must contain the address of the partner device.

Connectivity. The ability of a device to trade data and share resources with other devices of a similar and dissimilar type through electronic means, including serial and parallel connections, networking, and telecommunication.

Connector. A device that establishes a physical connection between one conductor or circuit and another. Examples include an RJ-11 or RJ-45 telephone connector, which forms the connection between a patch cord and a distribution cable, or a PhoneNET connector, which establishes the connection between a Mac's LocalTalk circuitry and the network cabling.

Console. In SNMP (Simple Network Management Protocol), a software program that has the capability of interacting with an agent, including examining or changing the values of the data objects in the agent's Management Information Base (MIB).

Contention. A network access method where all the devices on a network have equal chances of gaining control of the network at any time. Includes both collision detection (CSMA/CD) and collision avoidance (CSMA/CA) access methods.

Control Panel. In the Macintosh, software that has the ability to control an aspect of system configuration, such as the pixel depth of the monitor, the choice of network type, the desktop pattern or the manner in which directories are displayed.

CPU. Central Processing Unit. The main processor in the computer's configuration that handles processing tasks or directs auxiliary processors (coprocessors) to perform them. In a Macintosh

IIci, for example, a Motorola MC-68030 acts as the CPU and directs the activities of other processors on the motherboard, such as the Zilog 8530 serial port processing unit and a MC68882 math coprocessor.

CPU-bound. A computing activity or network interaction whose speed is limited by the speed at which the CPU can perform the necessary computing tasks. *See also* I/O-bound, Network-bound.

Crash. An abrupt termination of computing activity caused by an error. In some instances, the computer becomes completely unusable and must be restarted before activity can resume. In other cases, a single application self-terminates, or "hangs," i.e., remains active but unusable. In some application crashes, computing can resume after the application self-terminates or is forcefully terminated ("killed").

CRC. Cyclic Redundancy Check. A method of insuring data integrity where a calculation is performed using the binary representation of the data itself as the basis of the calculation. The CRC is the numerical result of this calculation and is held separately from the data. The integrity of the data is checked by calculating a new CRC and comparing it to the previously calculated CRC. If the two CRCs match, then there is a high degree of confidence that the data has not changed.

Crosstalk. In electronic signaling, an error condition caused when the signal from one circuit causes a disturbance to the signal of a nearby circuit.

Daemon. In the UNIX operating system, a computing process that is not under user control, but runs in the background. Daemons usually perform a particular purpose on demand, such as supplying information to another process. An example in AppleTalk networking is the atalkad daemon, which supplies AppleTalk tunneling information to routers on request.

Daisy chain. In LocalTalk parlance, a daisy chain is made by linking LocalTalk connectors together with patch cord. In telephony, a daisy chain refers to the method of linking a series of wall outlets together with twisted pair cabling rather than the more typical practice of connecting the wall outlets to a central location (home run). In telephony, daisy chaining is equivalent to the "backbone" method of LocalTalk construction.

DAT. Digital Audio Tape. A storage media used for the backup of computing data.

Data. Information represented in a format readable by a computer.

Database. A collection of data that can be selectively retrieved by a type of application known as a Database Management System (DBMS).

Data Link. The physical connection between two devices such as Ethernet, LocalTalk or Token Ring that is capable of carrying information in the service of networking protocols such as AppleTalk, TCP/IP, or XNS.

Data Link Protocol. The protocol that has the responsibility of controlling the network signaling and receiving hardware, performing data integrity checks, and formatting (framing) information according to the rules of the data link.

Datagram. A data packet that contains the protocol address of the software process that is the intended receiver.

Decay. A loss in the clarity or readability of an electronic signal caused by the interaction of the signal with its carrier and electrical environment.

Decibel. A measure that refers to the ratio of the strength of one signal to another. Decibels are commonly used to express signal loss (the ratio of received strength to original strength) or the relationship of the signal strength to ambient noise (the signal-to-noise ratio).

Desktop. In the Macintosh user interface, the background image of the Finder on which the icons for applications, directories, and data files are displayed.

Device driver. Software that acts as an intermediary between a CPU and a peripheral device. The CPU sends a command to the device driver, which translates that command into a command meaningful to the peripheral device.

Diagnostic. A test or the data from a test which indicates the condition of a computer or network's health.

DOS. The operating system of most IBM-compatible personal computers.

Download. The transfer of a file from a remote computer to a local computer.

Downsizing. The transfer of computing tasks previously performed by mainframe or minicomputers to personal computers.

Downtime. 1. A temporary interruption in the usability of a computer system. 2. A work stoppage caused by the temporary lack of usability of a computer system.

Drive. A data storage device.

Dynamic addressing. A system of addressing where the computer selects its own address without the user's intervention.

E-mail. Electronic mail. A network application that can deliver messages from one computer user to another.

Echo. In electronic signaling, the reflection of a signal caused by a sudden change in the impedance of the carrier.

Echo Protocol. In the AppleTalk protocol family, a protocol that allows a computer to return test packets. The purpose of Echo Protocol is to test the delivery conditions to a remote node, including reachability, reliability, and round-trip time.

Echo test. A diagnostic test in which packets are sent by one node to another node, which immediately returns them to the first node. The data recorded by the echo test includes the success rate of the return as well as the time needed to complete the round trip.

Electromagnetic Interference (EMI). Interference in the integrity of a signal caused by radiation. An example is the radiation from a fluorescent light, which emits a broad spectrum of electromagnetic radiation, including radiation that may be harmful to a signal not protected by either shielding or adequate twisting.

Emulation. A network activity in which a computer acts as if it is another kind of computer or terminal. An example is when a Macintosh user opens a remote terminal session to a VAX and runs a program that *emulates* a DEC VT240 terminal.

Enterprise network. A networking system that allows communication and resource sharing among all of a company's business functions and workers. (In some circles, this would even include the company's business including its suppliers and distributors.)

Entity. A hardware (or firmware) device or software process capable of initiating or responding to communication. Entities typically possess a unique address.

Entropy. 1. A measure of the disorder of a system. 2. The thermodynamic tendency of a system to reduce its overall energy state by increasing its disorder. Theoretically, an equilibrium is reached where the energy reduction that can be gained by a further increase in entropy is offset by the energy necessary to contain that increase.

Error checking. In data transmission, an action where the integrity of data is checked by a system. A typical method is to have the sending node calculate a number using binary representation of the data as the basis of the calculation. This number is delivered along with the data. The receiving node duplicates the calculation and compares the result with the number shipped along with the data. Examples including parity checks, checksums and CRC, checking mechanisms.

Ethernet. A specification for a transmission system including Layers 1 and 2 of the OSI 7-layer model using the CSMA/CD access method. In common usage, *Ethernet* refers to either the DIX (DEC - Intel - Xerox) version of this specification or to the IEEE version, more formally known as 802.3. The DIX version is distinguished by the reference Ethernet V.2.

EtherTalk. 1. EtherTalk Link Access Protocol (ELAP), the protocol that places AppleTalk's DDP formatted packets in Ethernet frames. 2. The implementation of AppleTalk using Ethernet as a delivery system. In AppleTalk Phase 1, Ethernet V.2 is used; in Phase 2, 802.3 is used.

Extension. A software addition to the Macintosh operating system that extends its functionality.

FCC. Federal Communications Commission. The United States government agency that regulates electronic communication and the domestic manufacture and importation of communication equipment.

FDDI. Fiber Data Distributed Interface. A specification for 100-Mbps (Mbits/-second) token passing ring implemented on fiber-optic cabling.

Female connector. Also called a jack. A connector that joins with a male connector by providing recesses into which the male connector's contact points or pins are inserted.

Fiber optic. A transmission media that use a light wave for signaling.

Field. In an information packet, a group of one or more bytes that performs a specific function, such as designating the recipient of the packet, the length of the packet, or the type of protocol encoded in the packet.

File system. The collection of system software routines that manages and accesses files located on a computer's storage volumes.

Filter. A network manager-defined conditional test placed on incoming packets in a network bridge or protocol analyzer. Generally, if the packet meets the conditions defined in the filter criteria, it undergoes further processing. If the packet does not meet the filter criteria, it is rejected.

Finder. A software application included with Macintosh system software that allows users to perform basic file access and management functions using icons and pull-down menus.

Firmware. A collection of programmed routines and instructions that is implemented in a computer chip or similar hardware form instead of a software form.

Flag. In a packet, a bit (or a group of bits) that indicates a condition. For example, the ZIP GetNetInfo Reply includes a 1-bit flag that indicates whether the zone name specified is a valid home zone name for the node that requested the information.

Flag byte. In LocalTalk signaling, the bit pattern 01111110, which signals the beginning and ending of a LLAP frame. The Flag bytes preceding the packet establish the synchronization of the frame similar to the function of an Ethernet preamble.

Flash memory. A type of non-volatile RAM.

FM. Frequency Modulation. In data transmission, a system of signaling in which the data is encoded by varying the frequency of the signal. In LocalTalk, for example, a square wave is used where a frequency of 230.4 KHz indicates a 1 and a frequency of 460.8 KHz indicates a 0. Frequency modulated signals are generally polarity-insensitive.

FOIRL. Fiber optic inter repeater link. An asynchronous fiber optic connection that links two Ethernet repeaters (hubs) with a maximum transmission distance of 2 kilometers when used at 10 Mbits/second.

Format. A specification for the arrangement of information of a disk, file or packet.

Frame. In data networks, the information packet and all preceding and succeeding signals (flag bytes, preambles, frame checks, abort sequences, etc.) necessary to convey it along the data link.

Frequency. 1. A measure of the rate of change of a signal. 2. In a periodic signal, the reciprocal of the time necessary to complete one period.

Frequency Distortion. A loss of integrity in a signal caused by a differential rate of decay of the signal according to its frequency components. In a signal made up of many frequency components (such as a square wave), the higher frequency components of a signal typically decay faster than the lower frequency components.

Full-duplex. A communication system between two entities in which either entity can transmit at any time.

Gateway. 1. A device that performs a protocol translation at the Session Layer or higher. An example is the Avatar NetWay 2000, which translates the user interface, data representation, and session management functions between an AppleTalk session and an SNA session. 2. (archaic) A TCP/IP router that routes packets between different network numbers.

GatorBox. A routing device made by Cayman System that can be augmented with gateway software (GatorShare) to provide file and print services to foreign systems.

Giga. A prefix denoting a billion (10^9).

Grade. (Also *level.*) In the specification of wiring for data networks, a standard designation used to describe the electrical quality of the wiring with regard to its suitability to carry high-speed signals.

Ground. An electrical conductor that is neither negatively nor positively charged.

Hacker. (Slang.) 1. An expert computer programmer. 2. A knowledgeable but disruptive computer user.

Hardware address. An address, fixed at the time of manufacturing, that identifies a network adapter such as an Ethernet card.

Header. In a network packet or frame, a section of data that describes the data that immediately follows.

Heap. The RAM memory allocated by system software and system extensions to hold frequently used instructions and data not contained in ROM or firmware.

Hertz (Hz). A unit of measure of the frequency of cyclic wave forms, equal to one cycle or period per second.

Hexadecimal. A numerical system with a base of 16 that is useful for expressing digital data. One hexadecimal digit represents four bits.

Hierarchical File System (HFS). The Macintosh's native file system.

Hint. In dynamic addressing, an address that a node tests for uniqueness first. The hint is either the last successful address the node used previously (the Macintosh keeps such a hint in PRAM) or a particular address that is specific to a particular model of device (the GatorBox always tries 128 first on LocalTalk networks).

Hop. In routed networks, the passage of a packet through a router on the way to its destination.

Hop count. In AppleTalk packets, a 4-bit counter in the DDP header that is incremented each time a packet passes through a router on the way to its destination.

Hop distance. A unit of measure used to express the number of routers that a packet must pass through its way to its destination.

Host. In terminal emulation, the remote computer that is being controlled by the terminal emulation software.

Hub. A term used to describe multiport network repeaters, usually of twisted-pair networks such as 10BASE-T or LocalTalk.

I/O. Input/Output. A computer activity where data is either loaded into (input) or extracted from (output) RAM.

I/O-bound. A computer or network activity whose speed is limited by the time necessary to perform I/O functions.

IBM. International Business Machines. (Can also refer to acronyms such as "inferior but marketable".)

Icon. A symbol used to represent a computer concept or operation.

IEEE. The Institute of Electrical and Electronic Engineers, a non-profit organization that, among many other activities, endeavors to coordinate, synthesize and promote data networking standards.

IEEE 802 Committee. The committee of the IEEE responsible for coordinating standards at the Data Link Layer. The committee also oversees the work of many subcommittees that govern individual data link standards such as the 802.3 standard.

Impedance. A measure of the opposition to the flow of an alternating signal by its media.

Implementation. The physical manifestation of a network standard or design.

Info Window. In a Macintosh program, a window that displays the attributes of an object.

Infrared. 1. A portion of electromagnetic spectrum situated between visible light and microwaves. 2. A means of short distance wireless networking that depends on an unobstructed line of sight path.

INIT. A process that is initiated by a Control Panel or Extension during startup that allocates memory in the System Heap and remains operative during the entire computing session. Similar to TSR (terminate and stay resident) process in DOS or a daemon in Unix. In System 6, the term INIT also referred to the System Extension file itself.

Initialization. The entry of a set of process parameters (performed by a human or automatically loaded from a file) that are necessary to begin a software process.

Instance. The single occurrence of a phenomena or event.

Insulator. A material that is highly resistant to electrical and/or thermal conduction.

Integrity. In networking, a desirable condition where the information received is exactly equal to the information sent.

Intelligence. The ability of a system to use general information to respond appropriately to specific events.

Internet. 1. A worldwide system of linked networks that is capable of exchanging mail and data through a common addressing and naming system based on TCP/IP protocols. 2. Any group of linked networks capable of exchanging electronic mail and data using a common protocol.

Internet address. An address that identifies a communication entity on an internet.

Internet node address. In AppleTalk, the combination of network and node address that uniquely defines an AppleTalk protocol running in a currently active device.

Internet socket address. In AppleTalk, the combination of network, node and socket address that uniquely identifies a software process that is using AppleTalk protocols to communicate.

Internet router. A router that uses the rules of one or more Network Layer protocols to forward packets between networks.

IP. Internet Protocol. The Network Layer protocol used in TCP/IP.

IPX. A protocol family that is proprietary to the Novell NetWare system.

ISDN. Integrated Services Digital Network.

ISO. The International Standards Organization.

Jack. A female connector.

Jacket. The protective outer covering of a computer or network cable.

Kbps. 1000 bits per second.

Kbyte. A unit of measure used to describe an amount of information equal to 1024 (2^{10}) bytes.

Kludge. A word used to describe a solution that lacks elegance or that contains components for a purpose significantly different than its original design purpose.

LAN. Local area network. A communication infrastructure that supports data and resource sharing within a small area (less than 2 km diameter) that is completely contained on the premises of a single owner.

LAN Server. A device that manages and allows the use of more than one kind of resource, such as storage or file services, print services, communication services, data base services, etc.

LAP. Link Access Protocol. Any protocol of the Data Link Layer, such as EtherTalk or LocalTalk.

LaserWriter. Any of a group of laser printers that uses PostScript as an imaging language and can communicate using AppleTalk protocols.

Latency. In data transmission, the delay in transmission time that occurs while information remains in a device's (such as a bridge or router) buffered memory before it can be sent along its path.

Layer. A group of communication functions and the protocols implemented to perform them as defined by a network standards organization, Usually refers to a group of functions as described by the OSI 7-Layer Model designated by the ISO.

LocalTalk. A Data Link Layer protocol defined in *Inside AppleTalk* by Apple Computer that covers the transmission of data on twisted-pair wire using CSMA/CA at 230.4 Kilobits per second.

Login. A formal procedure where a computing device initiates a sustained connection to another computing device for the purpose of using a resource managed by the second device.

Logout. A formal procedure where a computing device severs its connection to another computing device.

Loopback packet. A test packet sent by a network adapter with a destination address equal to the adapter's own hardware address. The purpose of this test is typically to establish that the adapter is connected to a network that is functional enough to support a data transmission.

Loss. The aggregate attenuation of a signal due to interaction with its environment.

Mbps. One millions bits per second.

Mbyte. 1,048,576 (2^{20}) bytes.

Macintosh (Mac). Any of a group of computers manufactured by Apple Computer in the Macintosh family such as the Macintosh IIsi, the Macintosh Classic II, or the Macintosh IIx.

MAC. Media Access Control. The method whereby a computing device takes control of the transmission media for the purpose of sending an information packet.

Magnetic field. The area surrounding an electrically charged body in which an electromagnetic force can be detected.

Mainframe. An expensive, general purpose computer that can be used by many users simultaneously.

Male connector. A connector whose points of electrical contact are exposed.

MAU. Media Access Unit. The component of a network adapter that directly attaches to the transmission media.

Media. The environment in which the transmission signal is carried.

Memory. In computing, a system where data is stored for direct, high-speed access by a microprocessor.

Memory allocation. The amount of memory, usually RAM, that an application or process reserves for its use.

Metalanguage. A language that represents another language.

Micro. 1. A prefix that denotes one millionth of a unit of measure, such as a microsecond or microampere. 2. A prefix that denotes something small. 3. A slang term for any personal computer.

Microwave. 1. Any electromagnetic radiation with a wavelength between 1 millimeter and 1 meter. 2. A point-to-point data transmission system employing electromagnetic radiation using a carrier frequency in the microwave region.

Milli. A prefix denoting a one thousandth of a unit of measure, such as a millisecond or millimeter.

Minicomputer. (archaic) A multi-user computer capable of supporting 4 to 16 simultaneous users.

MIS. Management Information System. Used to describe the set of computing resources that hold and allow access to the information owned by an organization.

Mode. 1. One particular method or way of accomplishing a goal. 2. In fiber-optic transmission, a particular path between a light source and a receiver. 3. In statistics, the result with the highest frequency within the sample group.

Modem. A device that can covert data signals between analog and digital signaling systems.

MOV. Metal Oxide Varister. A device that acts as a surge suppresser by forcing transient high voltages to ground.

Multitasking. A computing device whose operating system can handle several tasks concurrently. In mono-processors such as the Mac, each active task is given short periods of time to use the CPU in a rotational fashion. There are many varieties and levels of sophistication of multitasking.

Multi-user. A computing process that can handle the requirements of several users simultaneously.

Named service. In AppleTalk, a computing process that has used Name Binding Protocol to register a process so that it may be located using a network resource manager like the Chooser.

Nano. A prefix that denotes 1 billionth of a unit of measure, as in nanosecond or nanometer.

Native. A standard component of a computer system, such as a native file system or a native protocol.

NB card. A hardware addition to a computer such as a Macintosh that communicates with the CPU via a NuBus.

NBP. Name Binding Protocol. The AppleTalk protocol that associates the name, type, and zone of a process with its Internet Socket Address.

NetWare. A trademark of Novell that includes network operating systems and LAN server processes that run on and/or serve many computing platforms, operating systems, and protocols.

Network. 1. The infrastructure that supports electronic data exchange. 2. In AppleTalk, a group of computers using the AppleTalk protocols that share a common cabling system and a common AppleTalk network number or range. Computing devices in a network may be separated by a repeater, hub, or bridge, but not by an AppleTalk router.

Network adapter. A hardware device that translates electronic signals between a computing device's native network hardware and the transmission media. A network adapter may also include memory or additional hardware or firmware to aid or perform the computing device's network operations.

Network administrator. Also Network manager. A person who is charged with the responsibility of caring for a network and the communication abilities of its users.

Network architecture. A set of specifications that defines every aspect of a data network's communication system, including but not limited to the types of user interfaces employed, the networking protocols used, and the structure and types of network cabling used.

Network management. A set of activities and duties whose goal is to provide high-quality, reliable communication among a group of networked computer users. Typical activities may include resource planning, network design, providing user assistance and training, reconfiguration of the network due to a change in user requirements, assessing user needs and designing appropriate solutions, and troubleshooting and remedying network problems as they arise.

Network Operating System. A term mostly used in DOS networking systems to refer collectively to the proprietary protocols and network file systems that computers use to exchange data with their LAN Servers. Examples include Banyan Vines, NetWare, and Microsoft LAN Manager.

NFS. Network File System. A file metalanguage and set of procedure calls to access and manage files. NFS is standard issue on nearly every computer that uses TCP/IP protocols as its standard network protocols.

Nibble. One half of a byte, represented by a single hexadecimal digit.

Node. A networked computing device that takes a protocol address and can initiate and respond to communication from other networked devices that employ similar protocols.

Non-volatile RAM. Any RAM that remains intact despite loss of power or shutdown.

Noise. Undesirable electrical or electromagnetic signals.

NuBus. One of a large number of computer bus architectures used in Macintosh computers.

Object. In NBP, the proper term for describing the individual name given to a service. In the Chooser, it is the name that is displayed in the list of devices.

OEM. Original Equipment Manufacture. A system of distribution where a company markets equipment purchased from another company under its own label.

Ohm (Ω). A measure of the opposition to the flow of electric current.

Operating system. A collection of system software routines that manages all of the peripherals, hardware components, and other computing resources and processes in a computing device.

Oscillate. A condition where a system changes between two or more configurations on a recurring basis.

Oscilloscope. A test device that can capture and display an oscillating electrical signal.

OSI. Open Systems Interconnection. The subset of the ISO that promotes and defines standards for open networking systems.

OSI Seven-Layer Model. A method of describing the relationships between network protocols by grouping them according to the communication functions the protocols provide. The OSI model defines seven distinct categories (layers) that act successively on data as it makes its way between the user and the transmission media.

Out of band. A transmission system that is separate and auxiliary to the network's transmission system and whose operability is independent of the operability of the network.

Output. A representation of a system's data that is externally visible.

Overwrite. An error condition that occurs when a process stores data in a location that it has not properly allocated for that purpose or that has been allocated by another process.

Packet. A discrete chunk of communication in a predefined format.

Padding. Meaningless data added to a packet to increase its size.

Parity checking. A method of error checking where an extra bit is used to signify the condition of a small group of data bits.

Passive star. A LocalTalk construction method where three or more pairs of network wires are joined in a single location.

PDS. Processor Direct Slot. One of a large number of computer bus architectures used in Macintosh computers.

Peek. 1. A term used to describe the viewing of network data not ordinarily visible to a user. 2. The name of a widely distributed LocalTalk protocol analyzer developed by Apple Computer, that is not currently sold or supported.

Peer. In networking, a device to which a computer has a network connection that is relatively symmetrical network connection, i.e., where both devices can initiate or respond to a similar set of requests.

Peripheral. 1. Any hardware resource used by a computer that is external to a computer's enclosure case and is accessed not by a network system, but by serial or parallel connections, bus architectures such as SCSI, MIDI, or VME. 2. In a user-centered network system, any hardware computing resource that can be accessed by a user.

PhoneNET System. A system of physical layer networking components manufactured by Farallon Computing for AppleTalk networks.

Physical address. Hardware address.

Pico. A prefix denoting 1 trillionth of a unit of measure, as in picosecond or picofarad.

Pin. A type of electrical contact point for a single conductor.

Plug. A male connector.

Poke. Transmission of a packet on a network for test purposes only. The name of a widely distributed application, developed by Apple Computer, that can create and transmit an AppleTalk test packet. Like Peek, Poke is not currently sold or supported.

Polarity. The orientation of a differential voltage.

Polling. A means of Media Access Control where a device may transmit information only when it is given permission to transmit by a controller device.

Port. On a network hub, bridge, or router, a physically distinct and individually controllable set of transmission hardware. Each port is connected to other ports through the device's internal electronic structures.

PostScript. An extensible programming language, developed by Adobe Systems, that is capable of describing and drawing complex images.

Precision. The smallest difference in measurement that a test instrument can distinguish.

Print spooler. A software process that accepts a print job from a workstation and then sends the job to an actual printer at a later time. There are two styles; a background spooler, where the print spooling process is resident in the same node as the process seeking the print service, and a hardware spooler, where the print spooling process is in a separate node.

Printer driver. In the Macintosh, a System Extension that is intermediate between the CPU and the printer. It accepts the Macintosh's internal representation of an image and translates it into the control codes and image descriptions necessary for the printer to manufacture an image.

Process. A set of instructions, such as a computer program or application that is currently active, i.e. consuming CPU time and memory resources. The term *application* usually refers to a process that a user can launch from the Finder and directly control, while the term *process* often implies that the process has been launched by the system and is not under direct user control.

Program. A set of instructions that has been coded into a computer language and compiled into a machine language.

PROM. Programmable Read-Only Memory. Firmware in which a chip has had a program "burned into" its internal circuitry. The designation "EPROM" indicates that the program is erasable.

Proprietary. Information concerning the methods or implementation of a technology that is owned by an individual or a company. Proprietary may also mean that the information is secret or must be licensed from the owner before it can be used by others.

Protocol. In networking, a specification of the data structures and algorithms necessary to accomplish a particular network function.

Protocol stack. (also *Protocol Suite* or *Protocol Family*) A group of protocols, usually specified by a single vendor or organization, implemented at more than one layer of the OSI Seven-layer Model that also have Service Access Points (inter-layer interfaces) defined.

Publish and Subscribe. An Apple proprietary method of defining and maintaining a live link between a file and an external piece of data.

Quad. A low-grade implementation of twisted-pair wire, in which four conductors (two pairs) are twisted together with no independent twisting between the pairs.

Query. The formation, usually in a prescribed format, that a computer process uses to retrieve specific information from another process, as opposed to a wholesale transfer of data.

Radio frequency. (or *RF*) 1. Electromagnetic radiation with a frequency in the range of 10 KHz to 300 GHz. 2. Wireless data communication using such radiation.

Radio Frequency Interference (RFI). The loss in the integrity of a signal due to RF.

RAM. Random Access Memory. A group of memory locations that are numerically identified to allow high-speed access by a CPU. In random access, any memory location can be accessed at any time by referring to its numerical identifier as compared to sequential access, where memory location 6 can be accessed only after accessing memory locations 1 through 5.

Reboot. (also *Restart* or *Reset*) A user activity where the user boots a computing device without interrupting its source of electrical power. The goal of rebooting is typically to return the device to a known configuration state or to remove unwanted data from volatile RAM. In a Macintosh, this can be done either from the Special menu in the Finder (preferred) or by using the Reset button.

Receiver. The node or process for which a packet or other information is intended.

Remote Access. A term used to describe the process of using a network's services via a modem connection. Also a common synonym for AppleTalk Remote Access Protocol (ARAP), which is one of the protocols commonly used for accessing AppleTalk networks.

Repeater. A Physical Layer device that restores, amplifies, reclocks, or otherwise improves a network signal received on one of its ports and transmits the improved signal out its ports without buffering or interpreting it.

ResEdit. A Macintosh utility that allows a user to modify the resource fork of an HFS file.

Reset button. On a "Programmer's Switch," a small apparatus shipped with but not installed on Macintosh systems, a small button that restarts the Macintosh.

Resistor. A passive electrical device that adds resistance to a circuit.

Resource fork. The portion of a Macintosh file that contains such auxiliary information as the menus, dialog boxes, and sounds that a file may require in addition to its data.

Response time. The gap between the time when a user initiates an action and the time that the action displays its results.

Ring. A type of network topology where the devices are connected to a continuous conductor.

ROM. Read Only Memory. A chip or other electronic device that contains memory that cannot be altered. In the Macintosh, the ROM contains the Macintosh OS and instructions for basic system operations.

Router. A device that forwards packets between networks according to the rules of a network layer protocol such as DDP and according to information gathered during its service concerning the structure of the internet.

SAP. Service Access Point. The interface between one protocol and another.

Scalability. The suitability of a system (particularly a network system) to operate properly and efficiently when configured on a large scale.

Screen saver. A background process that monitors system activity such as mouse clicks and keystrokes and takes over the system during prolonged periods of idleness. Screen savers prevent damage to a screen's phosphors either by reducing the intensity of electron flow or by displaying an image that changes often enough to avoid screen "burn-in." Some screen savers also offer a security option by requiring a password before a user can regain control of the system.

Script. A set of instructions that a computer can operate. Unlike a program, script instructions are not compiled into machine language but are interpreted by the system at the time of execution.

SCSI. Small Computer Systems Interface. A specification (ANSI X3T9.2. for a short distance (6 meters maximum) local area network using bus topology for up to eight devices. The Macintosh uses this network for attaching devices such as external disk drives, scanners, and other peripherals that are not for a internal bus structure (such as NuBus) slot due to their size nor suitable for a serial port due to data speed requirements.

Serial port. A port on a computing device that is capable of either transmitting or receiving one bit at a time. Examples include the Mac's printer and modem ports.

Server. A device that is shared by several users of a network.

Session. An ongoing relationship between two computing devices involving the allocation of resources and sustained data flow.

Session Layer. The layer in the OSI Seven-layer Model that is concerned with managing the resources required for the session between two computers.

Short. The direct connection of two or more conductors of a circuit.

Signal. The means of conveyance for a communication, typically an electromagnetic wave that is modulated to encode the information communicated.

Signal to noise ratio. In an electromagnetic signal, the ratio of the amplitude (strength) of a signal to the amplitude of ambient radiation and other signal disturbances, usually expressed in decibels (dB).

Slew rate. The maximum rate, usually expressed in volts/second, at which an active device such as a transistor or transformer can change its voltage state.

Source. The node or process transmitting information.

Specification. A document that defines a concept and its implementations.

Spike. A sudden and transient increase in the voltage from a power supply.

Square wave. An electromagnetic wave that oscillates between two voltage states, theoretically requiring no time to accomplish a state transition.

Standard. 1. A synonym for specification. 2. A component or way of accomplishing a task that is so frequently and widely used that it seems to be part of a specification.

Star. A network topology constructed by connecting computing devices to a common device.

Switch. A device that forwards packets between nodes based on the packet's destination node address (either hardware or protocol), typically with a buffer time longer than a repeater but shorter than the transmission time of the packet.

Synchronous. A communication system where stations may transmit only at prescribed intervals and must provide a timing pulse with their packet.

System. Any computer system that can be controlled by a user consisting of a CPU and optional equipment such as display monitors, disk drives, and other peripherals.

System file. The Macintosh file that contains system software for the Macintosh OS, including patches or replacements to obsolete code contained in ROM.

System folder. The directory in which the system file resides.

System management. Activities that focus on the care and management of a computer systems.

System software. Software in a computing system that is used by applications to provide basic functionality like file management, visual display, and keyboard input.

T1. A telecommunications technology useful in wide area networks (WANs) that uses a transmission speed of 1.554 MBPS.

Tap. An intrusion into a network cable by a connector.

Task. 1. Synonym for process. 2. An activity or group of activities necessary to accomplish a goal.

TCP/IP. Transmission Control Protocol/Internet Protocol. A Transport and Network Layer protocol, respectively, used by a large number of computers.

Telecommunications. The system of technologies used in telephone communication.

Telephone wiring. Wiring installed to support a telephone system.

Telephony. The science and practice of telecommunications.

10BASE-T. A specification of the IEEE 802.3 committee (802.3i) for the implementation of 10-MBit Ethernet on unshielded twisted-pair wiring.

Terminal emulation. A computing activity in which a computer runs an application and communicates with a host as if it were a terminal such as a DEC VT220.

Terminator. 1. A resistor placed at the end of a bus to prevent the reflection of signals. 2. Any of a number of T-series programmable androids optimized for security-related assignments, usually with external features similar to that of a human.

Thick Ethernet. (Also *10BASE5*.) A system for constructing an Ethernet using coaxial cabling approximately ½ inch in diameter.

Tightly coupled. A relationship between two processes where the rate of accomplishment and success of the two processes are highly interdependent.

Token passing. A MAC method on which stations may transmit only when they are in possession of a special bit sequence (token) passed from station to station.

Token Ring. A ring topology network that uses token passing for MAC.

TokenTalk. An implementation of the AppleTalk network system designed to work on an IBM Token Ring network.

Toner. (Also *tone generator.*) In a tone test set, the tone generating device.

Tool. Any device, software program or instrument constructed to help aid a human accomplish a goal.

Topology. The arrangement of computing devices in a network, such as a star, ring, or bus. The logical topology of a network describes how the signal travels from one node to the next. The physical topology of a network, which may be different from its logical topology, describes how the network's wires are arranged.

Trailer. The group of bytes that marks the end of a frame and usually contains an error checking mechanism such as a CRC.

Transceiver. In Ethernet, the device that performs the signaling functions of transmission and reception. A collision occurs when the transceiver sends and receives simultaneously.

Transfer rate. The rate at which data is transferred from one device to another, usually expressed in bits or in bytes per second.

Transient. A short-lived electrical event.

Transmission. The activity of sending or conveying information.

Transmit. To send information.

Transport Layer. The protocol layer of the OSI Seven-Layer Model that is concerned with management of the data flow between source and destination.

Transport services. The functions carried out by protocols in the Network or Transport Layers.

Trojan horse. A maliciously created computer program or virus that causes significant damage to a system. Trojan horses are typically given attractive file names and placed on bulletin board systems. Damage usually occurs when users launch the Trojan horse on their own systems. Named after a successful offensive ploy made by the ancient Greek army at the siege of Troy.

TrueType. An outline font technology developed by Apple Computer.

Truncate. A method of formatting data by removing characters at the end of the data that do not conform to the format desired.

Trunk. 1. A synonym for bus. 2. In the LocalTalk backbone topology, the main carrier of the LocalTalk signal from which stubs emanate to connect the workstations.

Tuple. A set of two related data items.

Tweak. A minor adjustment.

Type field. A byte or group of bytes that indicates the protocol used by the data that follows.

UNIX. A multitasking, multiuser operating system invented by Bell Labs that is used on many computer systems.

Upload. The activity of transferring a file from a user's computer system to a remote system.

UPS. Uninterruptible Power Supply. An electrical supply system that conditions electrical power for a computer system and that allows continued operation in the event of a power failure.

User. A person who uses a computer system to accomplish their work, as compared to a programmer or network manager.

User area. The area where users work and where the computer systems are located.

User interface. The computer's collection of display symbols and other sensory stimuli that present information to a user and the physical actions that the user can take to present data to a computer.

UTP. Unshielded twisting-pair wiring.

UTP Ethernet. 1. A synonym for 10BASE-T. 2. Any of a number of Ethernet technologies used before the adoption of the 10BASE-T standard, which provided a 10-MBPS network system using CSMA/CD.

Vaporware. A computer program that has been publicly announced but is not shipping, especially if the announcement precedes the shipping date by more than 30 days.

Variable. 1. A situation or aspect that cannot be expressed explicitly because it may have one of several values. 2. In troubleshooting, an aspect of a problem that makes it differ from a normal situation. 3. In a mathematical expression, a symbol that represents a number.

VAX. A family of computers made by the Digital Equipment Corporation.

VINES. A network operating system developed by Banyan.

Virtual. A situation in which a computer simulates aspects of an activity or device but the activity or device does not have a physical form.

Virus. A piece of computer code that attaches itself to applications and data files without their consent or knowledge.

Voice grade. A classification of communication wiring that is unsuitable for network data transfer.

Volt. The measure of a difference in electric potential.

WAN. Wide Area Network. A network that is created among devices separated by large distances (typically in excess of 50 miles).

Watt. A measure of electrical power.

Wave. The phenomena that occurs when a physical medium is host to a measurable condition that varies with time in intensity, frequency, or velocity.

Well-behaved. A condition that exhibits expected properties.

Wireless. Any system that provides communication capability without the use of wires.

Wiring closet. In wiring for telephone or data systems, any location, usually enclosed, from which the communication wires emanate.

Workgroup. A group of networked computer users who frequently communicate and share common devices with each other.

Bibliography

AppleTalk Books

Apple Computer, Inc. *AppleTalk Network System Overview.* Addison-Wesley, 1989.

Apple Computer, Inc. *Planning and Managing AppleTalk Networks.* Addison-Wesley, 1991.

Kosiur, David and Nancy E. H. Jones. *Macworld Networking Handbook.* IDG Books, 1992.

Rizzo, John. *MacUser Guide to Connectivity.* Ziff-Davis Press, 1993.

Sidhu, Gurshuran S., Richard F. Andrews, and Alan B. Oppenheimer. *Inside AppleTalk*, second edition. Addison-Wesley, 1990.

Other Useful Books

Adobe Systems, Inc. *PostScript Language Reference Manual,* 2nd edition. Addison-Wesley, 1990.

Apple Computer, Inc. *Guide to the Macintosh Family Hardware.* Addison-Wesley, 1990.

Apple Computer, Inc. *Technical Introduction to the Macintosh Family,* second edition. Addison-Wesley, 1992.

Belden Wire and Cable Guide. Product Catalog, Yearly.

Freedman, Alan. *Electronic Computer Glossary (HyperCard Stack).* The Computer Language Company, 1992.

Human Interface Guidelines, second edition. Addison-Wesley, 1993.

Local Talk StarController Manual. Farallon Computing, 1988.

Malamud, Carl. *Stacks: Interoperability in Today's Computer Networks.* Prentice Hall, 1992.

Miller, Mark A. *LAN Troubleshooting Handbook.* M&T Books, 1989.

Pina, Larry. *Macintosh II Repair and Upgrade Secrets.* Brady Publishing, 1991.

Programming the LaserWriter. Addison-Wesley, 1991.

Rose, Marshall T. *The Simple Book.* Prentice Hall, 1991.

Stallings, William. *Handbook of Computer Communications Standards,* Vol. 2, 2nd edition. Howard W. Sams, 1990.

Tannenbaum, Andrew S. *Computer Networks,* 2nd edition. Prentice Hall, 1988.

Index

Tell us what you think and we'll send you a free M&T Books catalog

It is our goal at M&T Books to produce the best technical books available. But you can help us make our books even better by letting us know what you think about this particular title.Please take a moment to fill out this card and mail it to us. Your opinion is appreciated.

Tell us about yourself

Name____________________
Company____________________
Address____________________
City____________________
State/Zip____________________

Title of this book?

____

Where did you purchase this book?

□ Bookstore
□ Catalog
□ Direct Mail
□ Magazine Ad
□ Postcard Pack
□ Other

Why did you choose this book?

□ Recommended
□ Read book review
□ Read ad/catalog copy
□ Responded to a special offer
□ M&T Books' reputation
□ Price
□ Nice Cover

How would you rate the overall content of this book?

□ Excellent
□ Good
□ Fair
□ Poor

Why?

____
____

What chapters did you find valuable?

____
____
____

What did you find least useful?

____
____
____

What topic(s) would you add to future editions of this book?

____
____
____

What other titles would you like to see M&T Books publish?

____
____
____

Which format do you prefer for the optional disk?

□ 5.25" □ 3.5"

Any other comments?

____
____
____

□ Check here for
M&T Books Catalog

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS MAIL PERMIT 1184 NEW YORK, NY

POSTAGE WILL BE PAID BY ADDRESSEE

M&T BOOKS
Att: Special Markets
115 West 18th Street
New York, NY 10011

PLEASE FOLD ALONG LINE AND STAPLE OR TAPE CLOSED

Troubleshooting Macintosh Networks

Here is *the* book to have at your side when troubleshooting a Macintosh network. *Troubleshooting Macintosh Networks* provides expert advice on troubleshooting AppleTalk and System 7. It includes detailed instructions for troubleshooting specific problems plus conceptual information that teaches you how to diagnose and solve general networking problems. You'll learn how to troubleshoot the AppleTalk layers, perform a network health scan, find and correct router configuration problems, and more.

Troubleshooting Macintosh Networks is easy to understand and is accessible to all network troubleshooters. If you're responsible for maintaining a productive Macintosh network, this book belongs in your technical library.

Topics include:

- Troubleshooting printer problems
- Finding and correcting router configuration problems—before your network shows any symptoms
- Diagnosing and troubleshooting network problems using the five troubleshooting modes: Random, Hunch, Set, Layer, and Process
- Using AppleShare and AFP services
- Extensive glossary of networking terms

Packed with illustrations and helpful examples, *Troubleshooting Macintosh Networks* provides the tools, techniques, and know-how you need to maintain a healthy Macintosh network.

Why this book is for you—See page 1.

Disk contains two hypercard stacks, AppleTalk Reference and PacketSend, filled with useful information:

- Library of network tools
- Vendor listings
- Troubleshooting articles from *MacUser*
- Network design guidelines
- and more!

DISK INCLUDED

Kurt VanderSluis is President of The Network Group, Inc. He has provided networking seminars for national and international corporations, including Lawrence Livermore National Laboratory, McDonnell Douglas, Grumman, and TRW. His specialty is troubleshooting unique networking problems. Kurt is also a contributing editor to *MacUser* magazine.

Amr Eissa is a veteran writer, software developer, and consultant. President of International Consulting Group, he has written technical documentation and designed network troubleshooting courses for Fortune 500 companies, including award-winning documentation for Apple Computer, Inc. His articles have appeared in *Apple Direct*, a newsletter for Macintosh developers.

M&T BOOKS

LEVEL	Intermediate–Advanced
TOPIC	Network Troubleshooting
HARDWARE	Macintosh

53495
9 781558 513150
ISBN 1-55851-315-9
>$34.95