

Troubleshooting Macintosh Networks

Kurt VanderSluis
The Network Group, Inc.



128 N 82nd Street Seattle WA 98103

Table of Contents

Introduction.....	1
A World of Abstraction.....	1
Using your Pre-Computer Expertise to Overcome the Difficulty.....	1
Read and Learn Both Practice and Theory.....	2
Troubleshooting Requires a Willingness To Be Both Impulsive and Methodical.....	3
Overcoming the Difficulty—Using this Book.....	3
Two Ways to Read the Book.....	4
What If This Book Doesn't Have the Specific Example I Need?.....	4
The Layout of the Chapters.....	5
Troubleshooting Preview.....	6
Troubleshooting Requirement 1: The Normal Network.....	6
Troubleshooting Requirement 2: Tools.....	7
Troubleshooting Requirement 3: Understanding.....	7
Troubleshooting Algorithms.....	8
Random Troubleshooting.....	8
Hunch-based Troubleshooting.....	8
Setup-Oriented Troubleshooting.....	9
Layer-Oriented Troubleshooting.....	10
Process-Oriented Troubleshooting.....	11
Choosing the Best Troubleshooting Mode.....	12
Summary.....	13
Network Concepts for Troubleshooters.....	14
Layered Protocols.....	15
Functions of the Layers.....	16
Network Addressing.....	17
Group and Individual Addressing.....	17
Hardware and Protocol Addresses.....	18
AppleTalk Addresses and Ethernet Addresses.....	19
Address Numbering Systems.....	19
Address Fields.....	20
Broadcast and Multicast Addresses.....	20
AppleTalk Addressing.....	21
Network Address.....	21
Extended Networks.....	22
Network Numbers.....	23
Node Address.....	23
Node Address Numbers.....	24
Duplicate Node Addresses.....	24
Node Addresses in Extended Networks.....	24
Socket Addresses.....	25
Socket Numbers.....	25
Network Relationships.....	25
Control of the Relationship—Client-Server and Peer-to-Peer.....	25
Configuration of Computing Elements.....	26
Flow Limitations—Finding the Bottleneck.....	28

TROUBLESHOOTING MACINTOSH NETWORKS

Speed Factors.....	28
Effect of Errors.....	29
Loss of Connection.....	29
An Example of a Client-Server Relationship.....	30
An Example of a Peer-to-Peer Relationship.....	31
Signaling and Transmission.....	31
Network Transmission Theory.....	32
Signals Decay over Distance.....	32
Loss of Power.....	32
Frequency Distortion.....	32
Reflections.....	33
Electrical Noise.....	33
Network Bandwidth and Utilization.....	34
Calculating Bandwidth.....	35
The Effect of Packet Size.....	36
LocalTalk Bandwidth-Calculation # 1.....	37
LocalTalk Bandwidth-Calculation # 2.....	37
Utilization—Using a Stopwatch.....	38
Utilization-Calculation #1.....	39
Utilization With PacketCrunch.....	39
Utilization With NetStats.....	41
Summary.....	42
Setup Troubleshooting.....	43
Troubleshooting LocalTalk Setups: General Information.....	44
The Relationship Between Design and Troubleshooting.....	45
Conceptualizing the LocalTalk Setup.....	46
The LocalTalk Connector.....	46
Problems with the Patch Cord.....	48
Problems with the RJ-11 Connector.....	49
Problems at the Wall Outlet.....	50
Problems with the Distribution Cabling.....	51
Low Quality Distribution Cabling.....	51
Too Many Wiring Changes and Connections.....	53
Problems with the Punchdown Block or Patch Panel.....	54
Problems with Punch Down Blocks.....	54
The LocalTalk hub.....	54
Special Considerations for Troubleshooting LocalTalk.....	55
Special Considerations for Troubleshooting Ethernet.....	56
Troubleshooting by the Layers— The Theory.....	57
Application Layer.....	57
Application Layer Problems.....	57
Presentation Layer.....	58
Presentation Layer Problems.....	59
Session Layer.....	59
Session Layer Problems.....	61
Transport Layer.....	62
Transport Layer Problems.....	63
Network Layer.....	64
Network Layer Problems.....	64
Data Link Layer.....	64
Data Link Layer Problems.....	65
Physical Layer.....	65

Physical Layer Problems.....66

Troubleshooting by the Layers– The Practice.....67

 Testing Electrical Continuity.....68

Data Link Layer Troubleshooting.....72

 Two kinds of test.....72

 Monitoring Data Link Errors.....73

 Data Link Layer Error Results in LocalTalk.....74

 Measuring Data Link Layer Errors in Ethernet.....77

 Ethernet Error Types and Their Meaning.....78

 Using the Echo Test.....80

 Interpreting Echo Test Results.....81

 Interpreting the Echo Round Trip Time.....81

 A Special Consideration for LocalTalk.....82

Network Layer Troubleshooting.....83

 Verifying the Existence of Zones.....83

 Verifying the Existence of Networks.....84

Transport Layer Troubleshooting.....85

 Using the Progressive Echo Test.....85

 Interpreting the PET Hop Count Results.....86

 Interpreting the PET Time and Reliability Results.....87

 Automating Reliability Checks.....89

Session Layer Troubleshooting.....89

 Special Considerations of the Session Layer.....90

 Checklist Item #1-The Service is Advertised.....90

 Checklist Item #2-The Service Can Communicate.....91

 Checklist Item #3-The Server’s Version is Correct.....93

 Checklist Item #4-The Server’s System Configuration is OK.....95

 Checklist Item #5-The Server Allows the Access That You Need.....95

 Checklist Item #6-Previous Users May Have Damaged the Server.....97

Presentation Layer Troubleshooting.....98

 Operations at the presentation layer.....98

 Troubleshooting System Configurations.....101

 Establishing a Common File System.....106

 Troubleshooting File Formats and Components.....106

Network Process Descriptions.....108

 Device Startup.....108

 Startup on LocalTalk with No Router.....109

 Registering an NBP Service Name.....110

 More About Name Registration.....111

 Startup on LocalTalk with a Router.....113

 What Happens When You Turn AppleTalk Off?.....114

 Router Startup on LocalTalk.....114

 Startup on EtherTalk—Non-Routing Nodes.....115

 Returning to a Familiar Network.....115

 Starting up on an Unfamiliar Network.....116

 Checking for Duplicate Service Names on EtherTalk.....117

 Router Startup on EtherTalk.....117

 Maintaining Routing Tables and Zone Tables.....117

 Protocol Problems.....118

 Router Configuration.....118

 Router Startup.....120

 The Routing Table.....120

TROUBLESHOOTING MACINTOSH NETWORKS

The Zone Information Table (ZIT).....	121
Route Information Broadcasts.....	121
The RTMP Process.....	122
The First RTMP Packet is Received.....	123
The Process Continues.....	124
Changing Zone Names.....	125
Using the Chooser.....	125
Basic Chooser Operation.....	128
The Role of Routers in Using the Chooser.....	129
NBP Search Mechanics.....	131
Named Services.....	132
Chooser Troubleshooting Example.....	133
The Chooser as a Troubleshooting Tool.....	135
Printing from an Application.....	135
Overview of the Network Process.....	136
Locating the LaserWriter by Name.....	136
Establishing the LaserWriter Connection.....	137
Initializing the LaserWriter Connection.....	138
Sending Data in a Read-Driven Connection.....	138
Overview of the Print Session.....	140
Querying the LaserWriter for Laser Prep Version.....	140
LaserWriter's Answer (<i>Three possible responses</i>):.....	141
Querying the LaserWriter for Font Information.....	141
Sending the Document Instructions.....	141
Troubleshooting Techniques.....	145
Performing a Network Health Scan.....	145
Keeping Records.....	146
Benchmark Test Definition.....	146
Check the Network Utilization.....	147
Characterizing Bursty Traffic.....	147
Traffic Monitoring Tools.....	148
Check the Service Activity.....	148
Run Benchmark Operations.....	149
Check Error Logs and Statistics.....	151
Finding and Correcting Router Configuration Problems.....	152
How Router Problems Start.....	152
AppleTalk's Susceptibility to Router Configuration Errors.....	155
Easy Ways to Identify Router Configuration Errors.....	156
Counting the Routers on Your Backbone Network.....	159
Finding an Unknown Router.....	161
Checking for ZIP Query Packets.....	163
Mis-typed Zone Names.....	165
Checking Your Backbone Network's Routers.....	168
Analyzing Network Traffic.....	172
What Analysis of Traffic is Needed?.....	172
The Inadequacy of the Current Generation of Tools.....	172
Traffic Analysis with Protocol Analyzers.....	173
Documenting a Network for Troubleshooting.....	174
Random Mode Documentation.....	174
Documentation for Hunch Troubleshooting.....	175
Documentation for Setup-Mode Troubleshooting.....	176
Documentation for Layer-Mode Troubleshooting.....	178
Documentation for Process Mode Troubleshooting.....	178

Building Your Tool Kit.....	179
The Basic Tool Kit.....	180
The Intermediate Tool Kit.....	182
The Advanced Tool Kit.....	185
Summary.....	186
Case Study.....	187
Troubleshooting a Printing Problem.....	187
What Happened.....	187
First Pass—Hunch Mode.....	189
Second Pass—Process Mode.....	189
Exploring the Trouble Process.....	190
Examining the Printing Packets.....	194
Conclusion: It Pays to Keep Going.....	196
Many Tools and References.....	196
LocalTalk Paramaters.....	198
Ethernet Parameters.....	198
Types of Ethernet Media:.....	198
Pinouts:.....	198
Ethernet Manufacturer Designators.....	199
Ethernet Protocol Designators.....	202
Packet Glossary	206
Introduction-How to Use this Section.....	206
Data Link Formats-LLAP, ELAP.....	206
LLAP Format.....	207
ELAP Format.....	208
Ethernet Format-Phase 1.....	209
DDP Limitation.....	210
AARP Packets on Ethernet.....	210
Differences Between Phase 1 and Phase 2 AARP.....	210
DDP Packets.....	211
Short DDP Format.....	212
Long DDP Format.....	213
DDP Type.....	214
Glossary.....	214
Bibliography.....	233
AppleTalk Books:.....	233
Other Useful Books:.....	233

Introduction

Network troubleshooting is hard for two reasons—networks are hard to understand and troubleshooting is hard to perform. I believe that when you face a hard task, the best course of action is to admit that it's hard, discover why it's hard and do what you can to overcome the difficulties. It's not effective to pretend that the difficulty doesn't exist, because then you'll do all the easy things that you know and give up, shaking your head and saying how you can't understand why your measures weren't effective.

Sometimes you must change your approach in order to overcome a difficulty. Mathematicians know many kinds of equations and have elegant solutions to many of them. Others, transcendental equations are an example, are only solved by "brute force". Still others require a combination of elegance to get near to the answer and brute computational force to home in on the final answer. It's important to have more than one approach to a situation.

A World of Abstraction

Networks are hard to understand because they require the evolving network expert to create mental linkages between several abstract concepts. This chain of abstractions is similar 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 cannot imagine anything else. The troubleshooter must conceive that:

A screen projects *images* that are controlled by an *application*, which performs a specific set of computing tasks under the guidance of an *operating system* that works in cooperation with a set of *network protocols* to manage the sending and receiving of data between systems using *network hardware* to provides 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 from different vendors interacting together. Small changes in one component in one part of the system can produce enormous changes in the working of another component somewhere far away in the network.

Using your Pre-Computer Expertise to Overcome the Difficulty

Almost everyone in the computer industry today was something else before. I was a metallurgical engineer for several years before deciding that computers interested me a lot more than metal did. A few weeks after our metallurgy department bought a VAX, I found I couldn't tear myself away from it. When I decided to make a complete switch a few years later, I was very concerned (as were my parents) that I was throwing away my hard-earned education (paid for by making an extraordinary number of pizzas and hamburgers) and that it would be very difficult to make the switch to a new kind of work. I found that it was indeed hard, but I didn't throw away my education.

I was able to apply a lot of my background to computers, using engineering and metallurgical concepts as stepping stones to understanding computers and then networks. For example, engineers understand very

TROUBLESHOOTING MACINTOSH NETWORKS

well that every decision involves a trade-off. For example, if you change a metal alloy to make it's surface more wear-resistant, it usually reduces the alloy's ability to absorb impact damage. Engineers understand the engineering design process as a series of tradeoffs to produce results that have the right mix of trade-offs, or as we say, "maximize the desired parameters." Network design involves these same kinds of tradeoffs and because of my engineering background, I know that my job is not to produce the perfect network, but to maximize the most important parameters.

Other backgrounds can also lend themselves. If you grew up on a farm, you can just as 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 and that the soil has to have the right ingredients and that it must be properly irrigated and weeded to produce healthy results. Networks need that same kind of preparation and care to produce good results.

If you are trained as a nurse, you know that a person has needs, both physical and emotional, in order to be healthy. You know how to perform actions that minister to these needs to promote a person's health. These can be related to the actions you take on a network—of inoculating the network against viruses, of scanning the network's vital signs to get a picture of its condition. You may have been an athlete, a car mechanic, a visual artist, a real estate agent, an accountant, or a community organizer or maybe you were just interested in ants; look for ways you can apply your previous knowledge to the task of becoming a computer or network expert.

Read and Learn Both Practice and Theory

In computers, it's impossible to make progress or even maintain a constant level of competence without constant and aggressive reading and learning. In computing, everything is constantly changing, so the computer experts must always learn something new. This is our blessing and our curse. It seems like a contradiction, but the more you know, the faster you have to learn just to stay in the same place of knowledge. That's why there is so much written material out there about every aspect of computing—AI, operating systems, connectivity and telecommunications, graphics and publishing, hypermedia and multimedia, etc., etc.

It's almost to a point now where the publishing industry isn't fast enough to keep up with the pace of change. One of the reasons I chose M&T Books as the publisher of this book is that they had the shortest cycle time between writing and publishing. When I write for *MacUser Magazine*, the article that you read was written four months before it appears on the newsstands. If the article involves research or product reviews, the research or product comparison often takes place 5-6 months before the article appears, and during that time many of the vendors may have upgraded their software or new vendors have appeared. This of course reduces the article's relevance and currency. Six months is just too long in an industry that moves this fast.

That's why I think it's so important to learn the theory behind your area of expertise. Although someone in metallurgy may invent a new method for the protective thermal spray of a metal this year, the principles governing the diffusion of one metal into another remains the same as it was last year and the year before and for many years before that. In computing, while it's very important to read weekly magazines like *MacWeek* to find out the latest network gadgets and inventions, it's equally important that you clearly understand the principles of networking so that the new gadgets and inventions can fit into a larger context than simply "what does it do?" By understanding fundamental networking principles such as media access, routing and bridging and switching, the protocols, the differences and similarities between networking and serial communications and telecom and ISDN, 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, as in the classes I teach, I've tried to give the right mix of both theory and practice. Hopefully, there's enough "practice" so that you'll be able to use the book to help you 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 are all different.

Some of the books on the market are more heavily slanted toward theory. Some of these are worthwhile to read. One of my favorite networking books, Andrew Tannenbaum's *Computer Networks* is virtually all theory, using examples only to illustrate how the theory takes form. I love this book because it explains in very precise terms the tradeoff process I mentioned earlier. It talks about the process of designing a protocol and how each design decision makes certain things more possible and other things more difficult. Other books are more slanted toward practice. Farallon's (LocalTalk) StarController manual is about the best book I've seen on the subject of building a network. The simple fact is that you need both theory and practical information to design, build and maintain an effective network.

Troubleshooting Requires a Willingness To Be Both Impulsive and Methodical

Some people are very impulsive and although they frequently get themselves in hot water, they also gain a lot of experience about a great number of things. Others are very methodical, and while the kinds of experience they gain covers less territory, they usually have a very deep understanding of the subjects which interest them. Both kinds of approaches can be useful in troubleshooting.

When some troubleshooters arrive on the scene of the trouble, I've frequently seen them plunge almost immediately into action, trying this and trying that until they exhaust all their ideas. At that point, they begin a more careful consideration. I've also seen people behave like this while looking for their car keys. When they notice the keys missing, they immediately begin looking in all of the places where their car keys usually are kept. After they exhaust all of these places, they begin the more methodical process, asking themselves "Where did I last see my car keys?" or "Who was the last person to use the car?" They begin sweeping their house or office in a methodical fashion, closing doors to rooms that have already been thoroughly searched.

Impulsive methods are in many cases the right thing to do. They will in some cases locate the car keys or fix the network. In some cases, impulsive fixes will cause more problems than what existed before troubleshooting began. In other cases, they will only be ineffective. Then the troubleshooter must use a more methodical method.

Methodical approaches are more likely to find subtle or hidden causes of trouble, but sometimes the narrowness of the method may lead the troubleshooter away from the cause of the trouble. In order for the methodical approach to work, the troubleshooter must already be on the right track. When 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 locate the problem.

Overcoming the Difficulty—Using this Book

I've tried to include both methodical approaches and impulsive approaches in this book because both kinds are sometimes appropriate. Please read the introductory chapter that outlines the 5 kinds of troubleshooting algorithms (or modes). You will begin to understand the situations in which each mode of troubleshooting is useful and also the situations in which a mode might be harmful.

Good luck in your troubleshooting—it's a unique learning environment. Problems almost always place you in a "new" situation and fixing a problem gives you a clear objective and a way of knowing whether you've attained your objective. I hope on one hand that you'll see troubleshooting as this kind of learning experience and learn to enjoy it, but on the other hand, that as you become better and better at it, that you'll have to do it less often.

This book contains detailed information about specific AppleTalk troubleshooting problems as well as conceptual information about general types of problems and how to figure them out. While some specific solutions are given in the book, the book is primarily concerned with the methodologies, knowledge and sensibilities required for troubleshooting. The reason for this is simple; the problems we face as

TROUBLESHOOTING MACINTOSH NETWORKS

troubleshooters change rapidly as new software and hardware continues to flow into the marketplace while the principles of troubleshooting change more slowly.

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. Others will quickly scan through it and place it on a bookshelf until actually involved in a troubleshooting problem, then try to find a section that deals with the situation they find themselves in.

If you are one of those reading the book in a comfortable chair, reading the book 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 I hope that you'll find some interesting information in there regardless of your current level of expertise. Later on, some of the investigative techniques you'll read about, like the Progressive Echo Test (PET) described in Section 5.5, "Transport Layer Troubleshooting" are fairly easy to "get". You'll probably be able to perform a PET after reading through the description just once. Other techniques, like the router configuration troubleshooting detailed in Section 7.2, are considerably more involved and will probably require frequent reference to the book while you're in the middle of that kind of a problem.

For those of you who are still in your first couple of years in networking, I've included information in the book that may be slightly tangential for a troubleshooting book, but may 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 or not to investigate a particular theory. For example, if you are troubleshooting a LocalTalk network that is only 25 feet long, transmission theory would tell you that improper termination is not a likely candidate for 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 3000 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 a graduate course in electrical engineering. They're about half text and half differential equations. I've struggled through a couple of them, and then begged my friends who have graduate degrees in electrical engineering to explain the concepts to me until I finally understood them. Since some of you may not have patient, articulate friends who have graduate degrees in electrical engineering, I've included the important concepts of transmission theory in this book and left the differential equations inside the other books.

If you are one of the people who is not reading this book in a comfy chair, and instead are using the book "live", perhaps with a perturbed user breathing down your neck, skip right ahead to Chapter 3. I hope that you'll get acquainted with the concept of the 5 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 may also try a few randomly selected remedies. Even the novice network manager has a few random methods that they try during this period, such as restarting the workstation. A list of random remedies can be found in "Random Troubleshooting" in Chapter 7. Once you have gained an understanding of the nature of the problem, then you begin to try your hunches. When your hunches are exhausted, you need to switch from Hunch mode to one of the 3 analytical modes. Chapter 3 has some information about how to choose the right mode. Then, you can jump to either the Setup, Hunch or Process 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 guidebook to the process of troubleshooting, not a catalogue of troubleshooting solutions. There are many specific topics that are missing from this book. For example, there is virtually no discussion of the problems of electronic mail. Despite this fact, you can still use the concepts in this book to help you troubleshoot electronic mail problems or problems involving other kinds of network applications that are also not addressed in this book. The concepts and diagnostic methods in Chapter 3

apply to any AppleTalk troubleshooting problem. Likewise, the principles in the Setup and Layer Troubleshooting chapters are just as applicable to mail setups and functions. For process troubleshooting, you will 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 has many parallels in e-mail operations. If you focus on the principles of the book as you use it, you will most likely find the information that you need to apply to your specific problem.

The Layout of the Chapters

Chapter 1 is an introductory chapter that tells how to orient yourself to network problem solving. It contains, among other things, an organization of troubleshooting methods into 5 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. It boils down some of the large concepts of networking. The primary reason for including this knowledge in a troubleshooting book is that I have always found it helpful to understand the big picture before I begin to work on the small one.

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.

Chapter 4 deals with Setup Troubleshooting, which normally occurs when the network either doesn't work at all or seems to have hardware or wiring problems.

Chapter 5 is about Layer Troubleshooting, which you typically do when you're network is behaving unusually or the problem is hard for you or the users to describe succinctly. Each of the Layers has a question that the troubleshooter tries to answer through the use of software and hardware tools.

Chapter 6 is a set of process descriptions for several basic network operations. I've chosen a number of processes that are representative and tried to detail what occurs in the devices and on the networks in each of them.

Chapter 7 describes some troubleshooting tips and techniques that I find useful.

Chapter 8 is a case study of solving a printing problem. It illustrates the flow of a troubleshooting session from beginning to end. Besides providing information about printing to a LaserWriter, it can serve as a model for many other kinds of troubleshooting situations.

Chapter 9 contains reference material. For those of you who are new to protocol analysis, I've included a packet glossary as well as standard text glossary.

Chapter 10 is a bibliography for people who want to continue their self-education. I've only listed books that I've found useful

Troubleshooting Preview

Troubleshooting is, by definition, a process performed when there is “trouble”—something is not behaving the way it should. The purpose of troubleshooting is to find the trouble and correct it, restoring the network to its normal condition—trouble-free. The process of troubleshooting proceeds as follows:

1. The troubleshooter compares the network in its troubled state to the network in its normal state
2. Hypothesizes about the cause of the difference
3. Applies corrective measures until the trouble is eliminated

Troubleshooting Requirement 1: The Normal Network

In order for the troubleshooting process to solve problems, certain requirements must be met. The first is that there must be a condition called “normal” where the network operates in a trouble-free manner. The first step in troubleshooting is a comparison to this “normal”. If there is no “normal”, no useful comparison can be made. Further, “normal” should be something that is well understood and documented by the troubleshooter, since the normal network is not available for comparison during the troubleshooting process.

The better you understand “normal”, the more easily you can describe the trouble. That’s why your doctor checks your blood pressure when you’re healthy. If you are feeling sick, she can take your blood pressure and compare it to the values she has in her files. If your “normal” blood pressure is not known in advance, your “sick” blood pressure has less meaning to the doctor and is less useful in her diagnosis. While your “sick” blood pressure can be compared to the national average for people of your age, weight, height and gender, it’s more meaningful when it’s compared to your own “normal” blood pressure.

Another reason why you need to understand “normal” is that you need to know when you’ve corrected the problem and you’re finished troubleshooting. Again using the medical analogy, the doctor needs to know when you’re cured—when your blood pressure and other measurable aspects of your body have returned to their normal state. If your normal blood pressure is different than the national average, your doctor needs to know that in order to know when to stop prescribing medicine.

Troubleshooting should be a rare event . If troubleshooting is a frequent activity, 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 will require 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. The Ethernet and its routers should also require the same low level of troubleshooting. “Once a month” should be thought of 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, the network design should be changed in such a way 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 minimum reliability.

Troubleshooting Requirement 2: Tools

The second requirement for the troubleshooting process is that the troubleshooter must have tools to quantify, characterize and adjust the network. Some of these tools are simple—like a wristwatch with a second hand. If you develop some benchmarks—you might have a word processing document that you know takes a 1:50 to print when the network is healthy—this tool provides a way to quantify the difference between normal and trouble. Other tools are more complex. You may use an oscilloscope, for example, because you need to see the shape of the network signal as it travels along the wire. Each tool tells its own story and is useful for different purposes and comparisons.

The current state of AppleTalk network management is that we must have a large collection of tools that measure, display and adjust various aspects of the network. The all-in-one network tool has not yet been invented. Networks are complex and involve many interdependent components—hardware, software, and wires. The network manager must have tools that deal with all of these components.

It cannot be stressed enough that all of the tools should be used on the normal network before trouble occurs to provide baseline data for comparison when the network is in trouble. Data gathered on the normal network should be saved or printed for reference when the network breaks down. Although it may appear obvious, when the network is in trouble, the normal network is not available and therefore pre-existing “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 in the tools that is currently not available. Even a very smart tool, like the Norton disk utilities, runs into problems that it cannot diagnose or fix. This is because the creator of a tool cannot possibly conceive of all the possible problems that may occur in a system. While the tool may be useful for a wide variety of problems, the tool cannot invent a new method for dealing with situations that it was not explicitly programmed for. 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 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 may encounter are more predictable than the possible structures and problems in a network. Networks, as mentioned above, 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 the network processes and interactions. Troubleshooting tools can gather and display the information needed to make the assessment, but a human troubleshooter can possess the understanding necessary to make the diagnosis and perform the repair, even 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 does not develop overnight. It must be built with experience and knowledge. We all start out as beginners and gradually develop understanding and expertise through our activities, our reading and our training. More advanced troubleshooters, besides knowing more specific information, have more methods of diagnosis available to them.

The act of troubleshooting itself helps you expand your understanding of networks because of the way that it focuses your attention. Reading a book such as this one in your armchair is one way of developing your knowledge, but when you’re reading it because it’s midnight and you need to get your data base back on line and the users are coming in 6 hours, you become a more focussed reader.

TROUBLESHOOTING MACINTOSH NETWORKS

This book was specifically designed to be used during the troubleshooting process. While you should read this introductory chapter and the next chapter about troubleshooting in advance, the rest of the book can be used “live”.

Troubleshooting Algorithms

There are 5 algorithms or modes of troubleshooting that are useful in different kinds of situations. All 5 modes proceed by the three steps outlined above—compare, hypothesize, experiment. There is a natural progression in troubleshooting ability as a person develops knowledge and experience. The more experienced a troubleshooter becomes, the more modes he has at his disposal.

Arranged from elementary to advanced, the five modes are:

1. *Random.* Try something and see if the trouble goes away.
2. *Hunch-based.* Try something that was previously useful in a similar situation.
3. *Setup-oriented.* Verify the continuity of the components and connections.
4. *Layer-oriented.* Verify the network functions according to the 7-layer model.
5. *Process-oriented.* Compare the process that is occurring with the normal process.

Random Troubleshooting

This is the most elementary method of troubleshooting. The troubleshooter simply tries a number of random methods to clear the trouble and proceeds until one of the methods succeeds or the methods are exhausted. Methods might include restarting a system, re-installing software, removing INIT's, or re-configuring the hardware or software. Although these activities may occur in one of the more advanced methods, in random troubleshooting they happen prior to a formal analysis.

The goal of random troubleshooting is to make the problem go away, not to understand the problem or prevent it from happening in the future. There are a dozen or so things that a network troubleshooter can do that will make many problems go away. These are detailed in the section following.

Some network managers may elect to give their users a subset of these “Things to Try”, and request that the user try all the items on the list before logging a trouble call. These are usually mild cures like re-selecting a device in the Chooser or wiggling connectors, but they can greatly reduce the volume of trouble calls if properly chosen.

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

For more information, see “Techniques-Random Troubleshooting”

Hunch-based Troubleshooting

Hunch-based troubleshooting involves applying previous experience to the current situation. The troubleshooter recognizes a similarity between different experiences. Sometimes this similarity is obvious, sometimes obscure, but the corrective methods applied are chosen because they have previously been effective.

If a beginning user complains of network trouble, experienced network managers usually begin with the hunch that the user is not performing a procedure correctly and observe the user's procedure to find whether the user is making a mistake. This hunch is made because new users frequently make mistakes or have misconceptions about network operations. If the troubleshooter determines that the procedure is being performed correctly but the operation does not complete successfully, he might check the Chooser configuration next because many users misunderstand the Chooser. If the Chooser configuration is

correct, the troubleshooter may have a hunch to check that the network connector is firmly in place or that the Network CDEV is properly configured or that a terminator is properly placed.

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

In the example above, if adjusting the Chooser configuration is frequently an effective cure, the troubleshooter may begin to suspect a system software configuration problem. This might lead to a more thorough examination of the version, structure and location of Chooser extensions, preference files and other system documents and possibly the discovery of the cause of the trouble.

Another possibility is that if the Chooser is frequently misunderstood, the naming scheme for network services might be inadequate or confusing. The troubleshooter might suggest either more user training or reference documents or a restructuring of the naming scheme. Administrative aspects of the network must always be analyzed with the same 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 the user’s system between the last time the network “worked” and when the problem was noticed. Often, this change was responsible for the failure of the network to function properly. Reverting back to the state of the network before the change was made will sometimes make the problem go away.

It is sometimes difficult to obtain information about these changes from users because they generally do not remember when they made which structural changes to their systems. For example, the troubleshooter may question the user and determine that a network operation had worked until a week ago, the last time the user had a requirement to use a particular network process. If the troubleshooter probes until he asks the right question, he may get the user to remember that a co-worker added a screen saver that he got from “somewhere” a few days ago. Also, more than one change may have happened between the time that a process worked and when it stopped working, making it more difficult to figure out which change caused the problem.

Do not to overuse this method of examining changes, however, as 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 and 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. It may be that the device was not installed properly, or that its presence dictates that other changes be made in the network to accommodate the new device. By simply removing the device and examining no farther, you may develop a negative attitude toward the device even though it was not actually the cause of the trouble.

Setup-Oriented Troubleshooting

This 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 cannot be established at all. For example, your screen might be displaying a dialog box with “Looking for the Microsoft Mail Server” and “Give Up”. The Setup-oriented method proceeds 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, the location of the break in the continuity can more easily found.

One of the prerequisites for setup-oriented troubleshooting is, of course, an understanding of the setup. A user or beginning network manager understands the setup for printing as:

TROUBLESHOOTING MACINTOSH NETWORKS

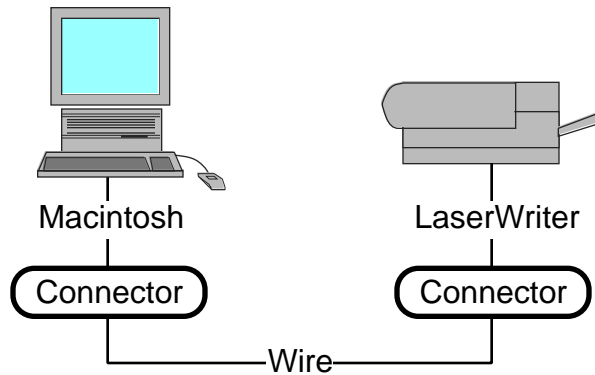


Figure 1-1. A Simple Way of Viewing the Mac-LaserWriter Setup while a more experienced troubleshooter may conceptualize the printing setup as:

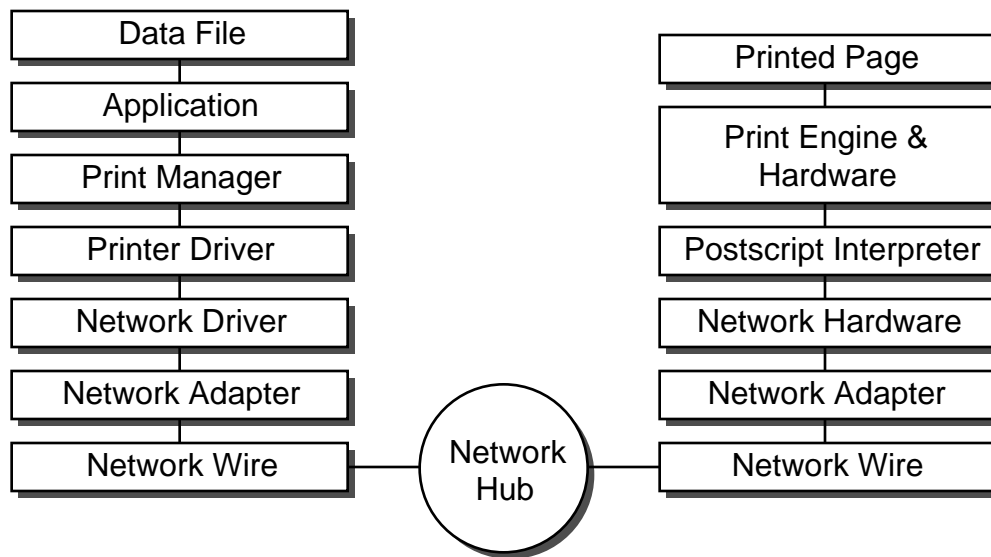


Figure 1-2. A More Detailed View of the Mac-LaserWriter Setup

After conceptualizing the connection between the Mac and the LaserWriter as a series of components, the troubleshooter begins to perform continuity tests to the components until he finds the place where the continuity is interrupted.

Continuity is verified by conducting tests on sub-groups 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, this link might be verified by printing to a different printer, perhaps an ImageWriter. The linkage from Postscript interpreter to printed page might be tested by having the printer print a startup page.

This kind of testing and analysis is performed until the continuity of the entire setup is established or the point of discontinuity is discovered and the defective component replaced or repaired.

Layer-Oriented Troubleshooting

In Setup-oriented troubleshooting, the network was viewed as a series of components. In Layer-oriented troubleshooting, the network is seen as a collection of interdependent functions. It's called Layer-oriented because it proceeds according to concepts established by the 7 layer model of the International Standards Organization (ISO), which served as a conceptual model for the AppleTalk Protocol family.

In the ISO 7-layer model, each of the 7 layers has a specific set of network functions to carry out. By methodically checking the presence of each of the functions individually, the troubleshooter can isolate which aspect of the network is broken. This method is best used in locating the source of a problem when the problem is not easily defined, occurs intermittently, or just seems “weird”.

When a problem seems weird, it means that your instincts are leading you in the wrong direction. An example would be when you see services in the Chooser dropping in and out and you suspect that your router may be malfunctioning. You check the router and everything seems to be OK, so the problem seems “weird”. You may have tried the hunch-based method and completely exhausted your hunches. Layer-oriented troubleshooting can help you identify the aspect of the network that is causing the problem.

For Layer-oriented troubleshooting, the troubleshooter will 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, then the layer below is checked. On the other hand, if the layer’s functions check out OK, then the analysis proceeds to the next higher layer.

Layer-oriented troubleshooting is similar to the child’s game “Chutes and Ladders”. At each protocol layer, there is a question and 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”, then you slide down the chute to the layer below.

For example, one of the functions 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. That particular function of the data link layer function can be verified by watching the traffic lights on the network hub blink when the device sends data. If the lights do not blink, then 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, then the analysis proceeds to the network layer.

The question at the data link layer is, “Are all the devices logically visible on the network?”. Besides 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 if all of the network devices you are accustomed to seeing are present. Inter•Poll also provides an echo test (detailed in the Technique Section), which can quantify the level of reliability for the connection.

Process-Oriented Troubleshooting

In Setup-oriented troubleshooting, the network was viewed as a series of components; in Layer-oriented, as a collection of functions. In Process-oriented troubleshooting, the network is viewed as a series of microevents. These microevents are captured by a protocol analyzer as a series of packets that contain the commands and information that a process such as printing needs in order to be successful.

While the Setup-oriented method was best used for troubleshooting when the connection couldn’t be made at all, Process-oriented troubleshooting is best used when a network process behaves abnormally. You may be able to communicate with the printer, but it seems very slow or quits part way through. Process-oriented 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-oriented troubleshooting is effective for many of the same situations as Layer-oriented troubleshooting. Process oriented troubleshooting is almost always more effective and usually faster than Layer-oriented troubleshooting, but it requires an order of magnitude more knowledge on the part of the troubleshooter.

By observing the process and comparing it with the normal process, the troubleshooter finds out where the process bogs down or terminates. The best tool to carry out Process-TS 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 amount of detailed information, it is difficult for the beginner to know exactly what information to look for. Protocol analyzers become easier to use as your experience and knowledge of the network develops.

TROUBLESHOOTING MACINTOSH NETWORKS

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

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

The LaserWriter printing process has, of course, many more events. These are detailed in the Process section and simplified here for the purpose of explaining the method. Process-oriented troubleshooting method checks the continuity between events. Each event is verified in sequence until the discontinuity is found.

Process-TS, as mentioned before, is a very advanced technique. There are some good training companies that offer advanced classes in AppleTalk protocols. My company, The Network Group, Inc., as well as Bear River Institute, offer classes that will help you become more familiar with reading packets.

If you cannot attend these classes, "Inside AppleTalk" is very useful when performing Process-TS, as well as a previously-captured protocol analyzer "trace" of a similar process that did complete satisfactorily. Like all of the troubleshooting tools, protocol analyzers should be used on the normal network. 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 will give you the clues you need to diagnose the trouble.

Choosing the Best Troubleshooting Mode

The first 2 modes, Random-TS and Hunch-TS, are useful when a quick solution is needed. While they do not usually yield useful information about the cause of the problem, they frequently provide at least a temporary cure. They are useful especially to beginning network managers or their alternates as well as to network users. Many experienced network managers will use one of these two methods when they first arrive on the scene or as an attempt at troubleshooting over the phone. When all of the hunches and random methods are exhausted, it is necessary to proceed to one of the more formal analyses found in the remaining 3 methods.

Process-TS, Setup-TS and Layer-TS are similar in that the network connection is conceptualized as a series of "things" and the continuity of those "things" are methodically examined in sequence. The difference is that the 3 troubleshooting modes use different sets of "things". Setup-TS sees the network as a series of connections between components, Layer-TS as a series of connections between functions and Process-TS as a series of connections between events.

The troubleshooter should choose the analysis method based on how the trouble appears. If the connection cannot be made at all, use Setup-TS. If the process begins, but does not go well, use Process-TS. If you cannot seem to figure out where the problem is occurring, use Layer-TS.

Summary

Each of the 5 troubleshooting modes is useful in different circumstances. All 5 modes proceed through the same 3 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 until the problem goes away.

The modes differ from each other mainly in the first step—the comparison process. In Random-TS, the difference is noted but not analyzed. At the other end of the spectrum, Process-TS, the comparison and analysis process is often highly detailed.

Each troubleshooting method depends on remembering the characteristics of the normal network for a mental comparison when the network is troubled. Since network troubleshooting and management tools are used for the comparison process, these should be used on the normal network and saved or printed for a baseline.

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 a network; in fact, the only peripheral that AppleTalk shared back then was an Apple LaserWriter.

Unbeknownst 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 was introduced and has been steadily improved. AppleTalk Data Stream Protocol has made more flexibility possible in AppleTalk Session Layer, and major enhancements have been made to Routing Table Maintenance Protocol (RTMP) and Zone Information Protocol (ZIP). At the writing of this book, we’re on the verge of having brand new protocols for routing, asynchronous network access, network management and electronic mail.

We’ll probably also see some architectural improvements that will allow Macs to be even more easily integrated into a Unix environment. The AppleTalk Network System designers continue to dream large. It’s very unusual for a company that only makes microcomputers to design and promote a network architecture at all, let alone one as ambitious as AppleTalk. Typically, network systems are only designed by manufacturers of larger, more powerful computers like Digital Equipment’s DECnet, and IBM’s Systems Network Architecture (SNA) or by international committees of network engineers. The success of AppleTalk is something of an anomaly and is a tribute to the vision of its founders and designers.

Although all of the emerging functions mentioned above are available today in proprietary form, 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, QuickMail and Microsoft Mail are currently the most popular mail packages, but they do not 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 which user interface appealed to them more.

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. 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 a peripherals on cable and connectors made by Apple), we now refer to as LocalTalk. LocalTalk is a physical implementation of the AppleTalk Network System.

LocalTalk also refers to alternate cabling schemes such as Farallon’s PhoneNET System, NuvoTech’s Hub and components and the Tribe LocalSwitch. LocalTalk is a physical network, and it is the physical

means for hosting the AppleTalk Network System. LocalTalk is specified by a set of properties that we associate with the bottom 2 layers of the OSI 7-Layer model, Physical and Data Link—twisted pair wiring, 230.4 KBits per second, Collision Avoidance, etc.

AppleTalk networks can be constructed on the shoulders of other physical networks such as Ethernet, Token Ring and Arcnet. Like LocalTalk, all of these networks are defined by their Physical Layer characteristics. AppleTalk operates on these networks because there is 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 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 that they think, “LocalTalk is a slow network”.

It’s not just AppleTalk terminology that has changed. The terminology of all of 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 only 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 that were made in AppleTalk’s early years have come back to haunt as AppleTalk networks become larger, faster and 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 seems very aware of the problems, even if they do not move as fast as we would like in correcting them. However, when you compare Apple’s speed to the speed of other organizations as they correct well-known and publicized problems, Apple compares quite favorably.

Computer networking is changing at an incredible pace, and overall, AppleTalk is keeping up and, in some ways, setting that pace. I sincerely 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 are referring to a family of protocols or a protocol suite. Protocol families such as AppleTalk govern the process that starts with the user clicking on the mouse or making a menu choice and ends with 0’s and 1’s being signalled 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 and has many sub-processes and functions. A traditional way to cope with the complexity is to create a structural framework that organizes and categorizes the sub-processes and functions. The sub-processes become more understandable and easier to grasp when placed in their categories.

TROUBLESHOOTING MACINTOSH NETWORKS

There are 2 conceptual frameworks for network protocols. One was created by a team of network engineers in the early 70's under the auspices of the US Department of Defense and is referred to as "the DoD Model". The protocol suite that is referred to as "TCP/IP" is organized according to the DoD Model.

The other conceptual model was developed later by the International Standards Organization and is steadily increasing in favor as the preferred model for network protocols. The new model is called the OSI 7-layer Model (OSI stands for Open Systems Interconnect), and forms the basis for organizing the AppleTalk protocol family, among others.

You can think of the 7-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 7-layer model, with some minor variances.

The 7-layer model, as its name implies, organizes the sub-processes into 7 distinct categories or layers. Each layer has a specific set of tasks to perform. Protocol designers, like 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.

Besides defining rules of construction and operation, organizing a protocol family into a framework like the 7-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 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, is Apple's official specification of the AppleTalk Protocols. It is 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

A way to understand the layers is to think of them as the set of functions that are set in motion when a user gives a command to the computer. The 7 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 7-Layer model are:

- | | |
|---------------------|---|
| 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 2 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 password and privilege information, ending the relationship and de-allocating 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	performs the delivery that Transport Layer manages.
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 is “readable”. In AppleTalk, the Data Link Layer also makes sure that each device has a unique address.
Physical	performs transmission under the guidance of Data Link Layer—creates outgoing signals and decodes incoming signals.

A useful way to remember the names and placement of the 7 layers from top to bottom is the sentence “**All People Seem To Need Data Processing**”. An alternative to this acronym goes from the bottom up, “**Pretty Darn Near The Silliest Possible Arrangement**”.

Network Addressing

Humans use addresses to communicate. In verbal communications, we typically use some combination of names to distinguish one person from another. If your office contains 3 people who are named Robert Stevens, Robert Hastings and Robert Wallace, the office will probably develop an addressing system that clearly distinguishes the 3 “Roberts”. For example, you could refer to them as Robert, Hastings and Bob, respectively. The point is that the addressing system should clearly and unmistakably distinguish any entity from any other.

Any entity can have numerous identifiers. Humans have, for instance, a social security number, driver’s license number, and telephone number, among others. Each identifier signals something about a different aspect of the entity’s existence and is useful for a special kind of communication.

In computer networks, we need addressing because networks are populated with many devices. We 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 is only intended for a single device. Because of this, 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 of the other devices can discover quickly that the packet is not directed to them. The devices can then ignore the rest of the data in the packet and spare their CPU the task of processing this useless data.

The device to which the packet is sent will read its own address in the front of the packet and continue reading and processing the packet. The packet will usually contain 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 simultaneously. It might have an address that refers to it as a hardware device, an address that refers to a particular software process it has operating inside itself, an address that refers to it as a member of a particular class of devices and there may be 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 tell something important to a number of individuals who are 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 10:00 Economics class!” and only members of that group will listen to your announcement.

TROUBLESHOOTING MACINTOSH NETWORKS

- “Hey Mark Fletcher: Economics class is cancelled today because Prof. Olmstead is sick.”
Of course, you’d have to repeat this for every class member.

These are three different kinds of addressing that correspond to addressing used in network systems. Addressing identifies the entities with which you wish to communicate. The first kind (“Hey everybody”) is called broadcasting. The second kind (“Hey members of a particular group”) is called multicasting and the third kind (“Hey Mark Fletcher”) is called directed transmission.

AppleTalk networking supports all of these kinds of address. AppleTalk devices interpret and respond to packets sent to their specific address, a group address that refers to “all AppleTalk devices” and 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, like Ethernet and Token Ring, it is 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, multi-protocol networks like Ethernet must make more distinctions in their addressing system than a network like LocalTalk. To make these distinctions, the network will use 2 addressing systems, one that refers to the network hardware and one the 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 at manufacturing time and will stay with a device regardless of which network it’s placed on or in which company the device is installed. Protocol addresses are established at boot-up time and have to be assigned either by the network administrator or (in the case of AppleTalk) assigned 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 will place the protocol address deeper in the packet. It is the hardware address that determines whether the device will process the packet or not. After the packet is properly received by the hardware, the protocol address determines how the receiver will process the packet.

As an example, any Macintosh placed on an Ethernet will be 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 00:00:89 was made by Cayman Systems. “Appendix C-Ethernet Addresses” contains a more complete list of Ethernet manufacturer designations.

To: 00:00:89:E1:12:17	From: 08:2C:60:39:22:94	
--------------------------	----------------------------	--

Figure 2-1. Ethernet and other kinds of multi-protocol 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 identifies a particular software address on a particular node on a particular network, and is established independently from the hardware address. When a device activates the AppleTalk Protocols, its AppleTalk address is established through a dynamic process that is described in “Device Processes-Booting a Mac”.

When a device uses AppleTalk to send information on a network like Ethernet that 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 on 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—LocalTalk, Arcnet, Ethernet, IBM Token Ring, over synchronous and asynchronous modems, through X.25 links and (soon!) over Fiber Data Distributed Interface (FDDI) links as well as the Integrated Services Digital Network (ISDN). In each case, AppleTalk will take the information from the sending process and place it in a data frame according to the rules of the 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 will be embedded inside the packet, but the packet must first be delivered to the Ethernet adapter. AppleTalk packets sent over Ethernet (EtherTalk) have four parts:

1. the Ethernet addresses of the source and destination devices
2. the AppleTalk addresses of the source and destination devices
3. protocol control information that tells the software process how to interpret the data
4. 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 and the rest of the packet is handed over to the AppleTalk process. In the remaining portion of the packet, the AppleTalk protocol address identifies the recipient software process which will use the control information to determine how to process the packet. Following the control information will be the actual data, the “payload” of the packet.

Ethernet Address Information	AppleTalk Address Information	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 that the data can be sent to the right application, AppleTalk control information so that 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 each addressing system. While the other AppleTalk protocols communicate in terms of the AppleTalk address, it is AARP’s responsibility to supply the necessary hardware addresses.

Because LocalTalk can only carry 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. Above, it was stated that the first 3 numbers (bytes) of an Ethernet address indicate the manufacturer of the card. AppleTalk addresses identify

TROUBLESHOOTING MACINTOSH NETWORKS

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.

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 a 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 range of 0 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 2^8 (256) 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. Since AppleTalk uses 32 bits to specify a software process, the total (theoretical) number of software processes that could simultaneously use AppleTalk to communicate in a single internet is 2^{32} or 4.32 billion addresses. Ethernet addresses use 48 bits, which allows 2^{48} addresses. This is a number (281 trillion) roughly equal to the number of pennies in the federal deficit. Since the first 24 bits are used to designate the manufacturer and the last 24 bits are used to designate the device, the Ethernet addressing system could support millions of manufacturers who could 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. An Ethernet address of 00:00:00:00:00:00 is an example. It means “this Ethernet adapter” and 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 only be used for AppleTalk Phase 2 devices. When an AppleTalk device sends this on Ethernet, it will set the Ethernet address to “09:00:07:FF:FF:FF”. Inside the packet, the AppleTalk network and node address will be “0.255”. The AppleTalk socket address will be set according to the nature of the communication. For example, socket 2 on every AppleTalk device is designated as the Names Information Socket. It is the address of a software process that manages name tables. When you search for AppleShare servers with the Chooser, you need to send a packet to every AppleTalk device’s Name Information Socket and ask if it has a name of “AFP Server”. 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. The address distinguishes that node from every node in the internet. The US Post Office has a similar need to uniquely identify every building in the United States. This can be done by using the combination of a building's ZIP Code and its 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 identifies a device uniquely and is referred to as the Internet Node Address.

Inside an AppleTalk device there may be several processes that need to communicate over the network, just as in a single building there may 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 name. Software applications inside a device, like 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 is sent 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, Farallon Liaison, etc.). Other nodes learn their network address from routers at boot time. Each network must have a network address different from every other network. The network address can be a single number or, on AppleTalk Phase 2 networks, can also be a range of numbers like 1-10. However, all of the numbers within the range must be assigned only to that network.

The figure below shows 2 routers creating 3 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.

TROUBLESHOOTING MACINTOSH NETWORKS

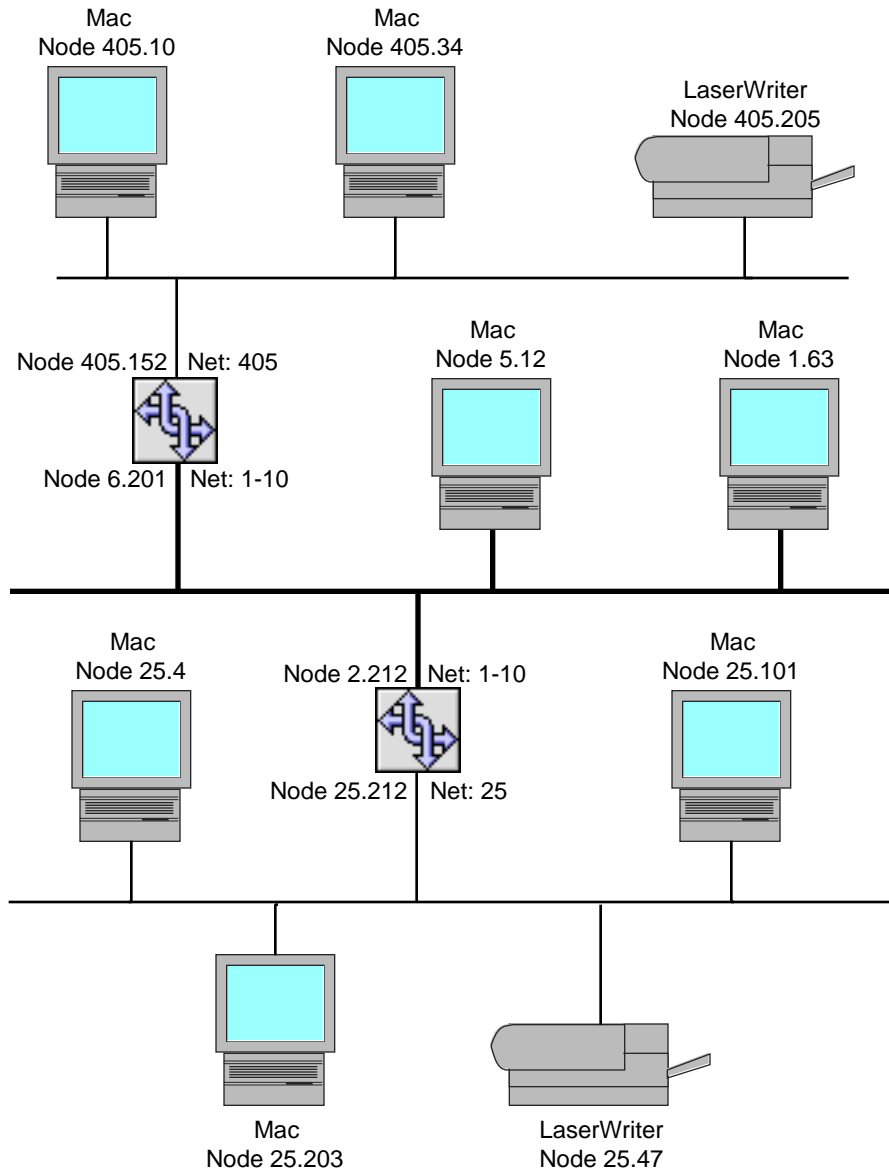


Figure 2-4. Every AppleTalk device must have a unique Internet Node Address. Routers define the valid network numbers for each network. Devices like Macintosh workstations discover their network address from the router. All nodes must establish their own node address during startup.

Extended Networks

AppleTalk Phase 2 introduced the concept of the “extended network”. In AppleTalk Phase 1, networks could only have a single network number. Because of the limitations of node addressing (described below), one network number would only allow 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 could 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 network is not valid. LocalTalk networks may only be assigned a single network number and are, by definition, non-extended networks.

Network Numbers

Network addresses use 16 bits. This allows network addresses of 0 to 65535. Network managers may only assign network numbers between 1 and 65279, however. Network “0” is reserved to describe a Phase 1 network with no routers. The network address of 65,535 is not used. Network numbers higher than 65,279 are reserved for a “startup range”. The startup range is used during the node address process, and is explained below.

Node Address

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

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

In AppleTalk Phase 1, and on LocalTalk networks, a device will use “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 it is attaching to, it temporarily uses a network address in the startup range of 65280-65534. It will also pick a randomly selected node number and uses this temporary address to ask a router to supply it with the range of valid network numbers. After receiving this information, the device will take a network address within the range supplied to it by the router along with a randomly selected node address. The device will then check 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 that do not have the Responder INIT and do not have any other INIT’s that register names or applications which check the network for duplicate serial numbers may wait to initialize AppleTalk until the user opens the Chooser or prints for the first time. In PC’s, node address acquisition occurs when the batch file that loads AppleTalk is run. This batch file may or may not be run as part of the startup process.

Sometimes a device may de-activate the 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 re-initialization 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 may re-initialize AppleTalk and repeat the node address acquisition process. Applications that do this are usually network management tools; examples include TrafficWatch and LocalPeek.

Instead of using a random number, Macs and many other network devices will recover the node address that they used during their most recent connection from non-volatile memory (PRAM or similar) 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 will use for the first address bid. If the first address is already taken, the device will use 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 Section 6.1 “Address Acquisition and Name Registration”.

Node Address Numbers

Node addresses use 8 bits. This allows node addresses from 0 to 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 of the numbers in the 1 to 254 range can be used for node numbers, but there is a distinction between personal computers and shared devices. The lower half of this range, 1-127, is typically used by personal computers and the upper range is typically used by shared devices such as printers, routers, file servers, shared modems, etc. The “rule” that divides the address range in half has never been very firm, and was intended to avoid a potential problem that the troubleshooter should be aware of.

Duplicate Node Addresses

The node address acquisition process is highly successful in avoiding duplicate addresses, but can be defeated if a user turns his computer on before plugging it into the network. This often happens when a user returns from a trip or after bringing his computer home for the weekend. If the node acquisition process occurs while the network is not connected, the first address attempted will be successful. Because there are no devices connected, no other device will be able to indicate that it already has the address being checked. When the user discovers that he forgot to plug his computer into the network and does so after the computer is running, there is a small chance that the address his computer is using will duplicate another device’s address.

If two devices have duplicate addresses, communication will be very difficult for both of the computers, although typically, some communication will sporadically get through. AppleTalk currently has no way of automatically detecting this situation and correcting it. The network troubleshooter can see the problem by seeing duplicate listings for a single address in Inter•Poll, but the problem can only be remedied by turning AppleTalk off in one of the devices and then re-activating the AppleTalk protocols.

Besides the user mistake mentioned above, another scenario that can result in duplicate addresses is when a network manager removes a router and joins two existing networks into a single network without turning the devices off. This may sound far-fetched, but it happens more often than you might expect.

By dividing the node address range in half—lower range for personal computers, upper half for “servers”—a user who boots his computer and then later connects would not cause problems with one of the shared devices, only another person’s workstation. On Ethernet and Token Ring networks, this precaution is not necessary because the AppleTalk protocols will not 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 to 253.

Node Addresses in Extended Networks

In extended networks, since there are a range of network numbers to choose from, it is the combination of network and node number that must be tested for uniqueness. In the figure above, the extended network in the middle of the figures has range of 1-10, so devices will choose a network address in that range and select a random node address. The device will then check to see if 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 a “mailbox number” for themselves. A node may have many sockets open simultaneously—each socket a “mailbox” for a particular software process. All data sent via AppleTalk is

sent to socket addresses. For example, when devices log on to a server, the device and the server will inform each other of which sockets to use for further communication.

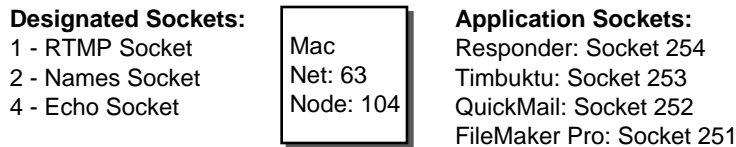


Figure 2-5. Each process in an AppleTalk device that needs to communicate over the network will establish a socket address itself.

Socket Numbers

AppleTalk sockets use 8 bits and have a range of 0 to 255. Socket number “0” is not used, nor is socket number “255”. Sockets in the range of 1-127 are referred to as *statically assigned*, which means that a software process would open a particular one of these sockets as its mailbox because that socket would have a specially designated purpose. Typically, commercial applications like Timbuktu and Microsoft Mail use the *dynamically assigned* sockets in the range of 128-254. In this case, the application asks 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 4 of these sockets for special purposes. The rest of the numbers are unavailable for use and continue to be reserved for a future use. The statically assigned sockets in the range of 64-128 may be used, but only experimentally—not for commercially released products. In spite of Apple’s request for programmers to avoid using these sockets, some programmers have chosen to use them anyway. An example is International Business Software, whose product, Data Club, sends broadcast packets (all nodes) to socket 130. Devices belonging to Data Club users will process this packet ; other devices will reject the packet.

The four sockets that Apple has designated are the mailboxes for four AppleTalk Protocol functions. Any information about routing is sent to socket 1, name information requests are sent to socket 2, echo packets to socket 4 and zone information to socket 6.

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 troubleshooters need to understand are:

1. How the course of the relationship is controlled
2. How the elements of computing are distributed: data storage, application code storage, processing, display
3. The factors that affect the flow of the relationship
4. The kinds of conditions that typically cause slowness, 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

There are several models that describe the relationship between two computers. They 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. As in all categorizations, however, the distinctions between the

TROUBLESHOOTING MACINTOSH NETWORKS

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 is finished. The server in the client-server model only reacts to the input that it receives from the server.

Examples of client-server relationships include:

- the Mac-LaserWriter connection
- the relationship between a mail or file server and a client workstation
- a FileMaker data base 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 2 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—mainframes, minis, workstations and micros—continues to break down.

Configuration of Computing Elements

In the client-server category, there are 5 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.

CHAPTER 2: NETWORK CONCEPTS FOR TROUBLESHOOTERS

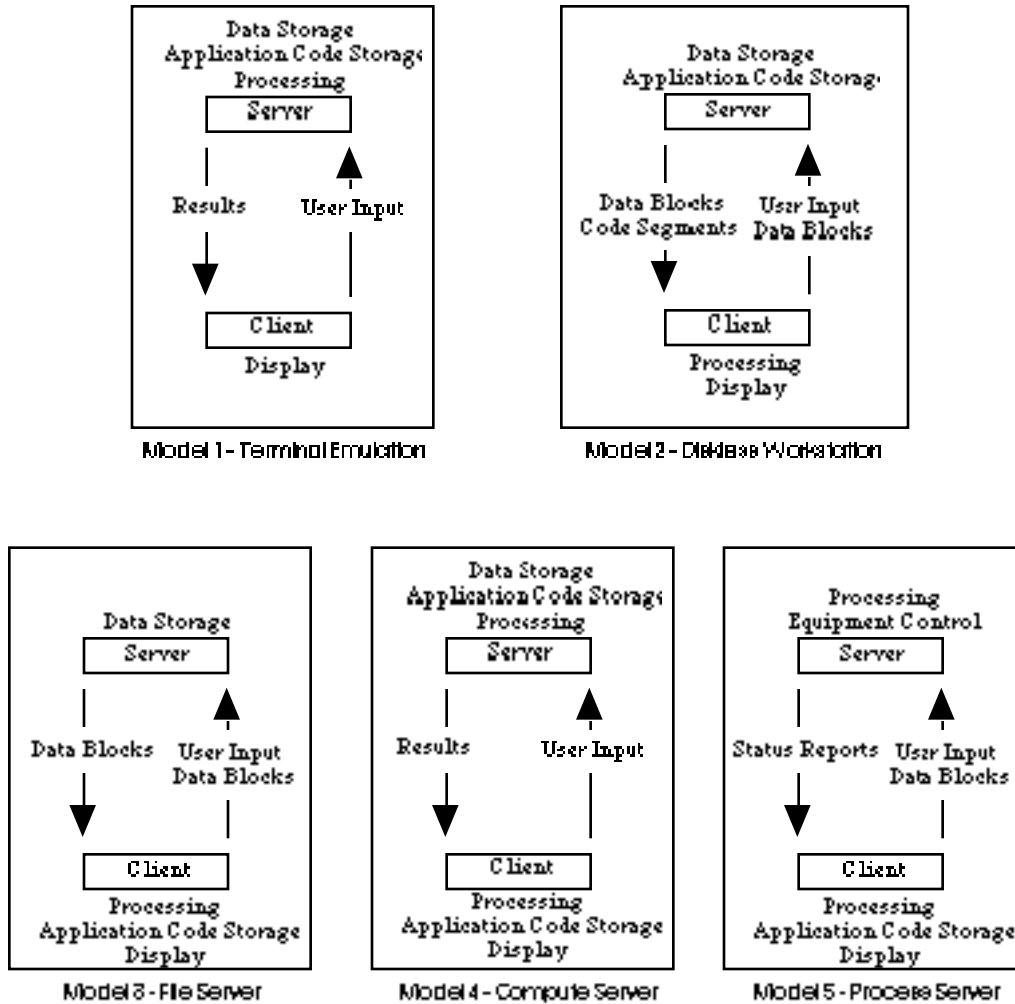


Figure 2-6. Models of client-server interaction

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.

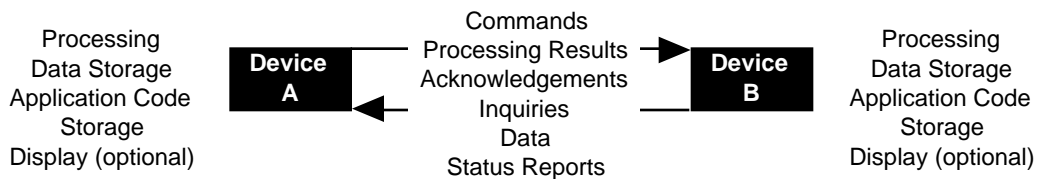


Figure 2-7. Peer-to-peer interaction

Of the two, client-server relationships are somewhat easier to troubleshoot because the troubleshooter can usually think of the devices one at a time. When the troubleshooter thinks in terms of the client, the server can usually be considered a “black box”, and vice versa. Also, commands and replies, questions and answers, etc., usually occur sequentially or close to it.

In peer-to-peer relationships, because actions can be initiated by any device at any time, the troubleshooter must keep both devices in mind simultaneously. It is also harder 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 8 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.

The "bottleneck" method is simplistic (flow relationships between multiple components are much more complex than this) but useful when considering, "What factors make this process go faster or slower?" If the user complains about the slowness of a process, you can assess whether (1) the process, given the configuration of equipment, is already proceeding as quickly as possible or (2) something is preventing the equipment from working optimally.

Situation (1) is a network management problem, and results in an analysis of whether the cost of upgrading the equipment will be offset by the projected productivity gains from a faster process. Identifying the bottleneck will help the network manager determine which component(s) need to be upgraded. Situation (2) is a troubleshooting problem that results in an investigation to find the cause of the slowness. Identifying the bottleneck will help the troubleshooter locate the source of slowness. For example, the LaserWriter bottleneck identified above can be analyzed by asking questions such 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? Only at the beginning?
- 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, the troubleshooter may be able to suggest ways to alleviate the user's complaint. For example, the troubleshooter may notice complicated graphic objects in the document that might be replaced with objects that the LaserWriter can draw more quickly.

For example, some graphics applications provide a wide range of fill patterns for objects. Some of these will usually take longer than others for the LaserWriter to draw. The troubleshooter might experiment a little to find a faster pattern fill type that produces a similar image but reduces printing time by 75%. The user could then make the choice to keep the gradient fill and tolerate the slowness or to change the object.

The bottleneck is not always easy to identify, and can be

- 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, etc.
- the speed of the network / network hardware
- RAM speed, size limitations

Speed Factors

Besides the "bottleneck", speed can further be affected by:

- 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 sequentiality of the interaction, and the efficiency of memory and cache management
- disk or memory fragmentation
- the number of routers/networks between devices

In multi-user situations, speed may also be affected by:

- the overhead of session management for simultaneous clients

- the sheer number of simultaneous users
- the workload from request from other users
- the way in which the server processes concurrent requests; for example, sequentially, complete request A, then complete B, etc., or “round-robin”, do a little of A, then a little of B, etc.

Effect of Errors

These errors may also slow things down:

- lost or garbled packets caused by bad wiring
- lost connections with other users forces the server to “look” for them
- one of a devices processes may be “stuck”, slowing other processes
- disk errors—bad blocks, corrupted directories, invalid pointers, etc.

The effect of a lost or garbled packet varies depending on what kind of packet it is. Almost all packets have an associated “retry timer”. After a device sends a packet, it usually expects a packet in return—an acknowledgement, data, a command, an answer to a query, etc. The duration of a packet’s retry timer is the amount of time the device will let pass before investigating whether its partner-device has received the packet or not. 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 senses that the other device is no longer available. There can be many reasons why a device would make this assessment, but in each case, the symptom that the device perceives is that it is no longer receiving packets from it partner.

Possible causes can be:

- 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 crashed (e.g. a router or hub), an intermediate network device was removed from service, or network reliability deteriorated to the point that the percentage of successful transmission was not high enough to support the connection

Many network relationships are “bursty” in nature—they are characterized by periods of relatively high usage and activity alternating with periods of relative idleness. During the idle periods, the devices will typically have some way of notifying each other that the connection is still healthy by sending a “hello”, or “tickle” packet.

Tickle packets usually have a “tickle timer” associated with them—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 long as 5 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 will usually initiate a search for its partner that will last until the connection timer expires. The connection timer is the maximum amount of time that can pass before the connections will be considered lost; the search for the partner will be terminated and the device will initiate the process of closing down the connection.

When closing down a connection, the device will make every effort to recover and return to a normal state. It will de-allocate any resources that were being consumed by the connection such as memory and sockets, close any files that remain open and terminate 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 may not recover at all, and will require restarting, or may result in the corruption of

TROUBLESHOOTING MACINTOSH NETWORKS

data. 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 multi-user data base. The worst case is when one device crashes and sets off a domino effect, freezing or crashing other devices around the network in its wake.

An extreme, yet possible domino scenario is:

- 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 and the searches terminated
- some devices crash because they cannot recover from the loss of connection
- the high level of traffic subsides and the network returns to normal
- devices are restarted and connections are re-initiated

The above scenario is admittedly an extreme case, yet it happens occasionally, particularly in networks with reliability problems. Poor reliability is 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 re-send 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

The relationship of a Mac printing to a LaserWriter is 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 LaserWriter 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 Laser 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 cannot print as fast as LocalTalk can deliver data; it 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 only accept one client at a time. However, it can report it’s current state to other prospective clients on request.

Printer Access Protocol manages the relationship and has a tickle timer of 1 minute and a connection timer of 2 minutes. The loss of 1 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 will have noticeably slower performance during the search.

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

An Example of a Peer-to-Peer Relationship

An example of a non-sequential peer-to-peer process is International Business Software'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, file names 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 offline. Offline 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 an illegal socket) 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 as multiple Macs may 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 may be multiple versions of the desktop information. The Data Club software will coordinate these changes until the information is identical on all Macs.

As you might imagine, the dynamics of this process are considerably harder to follow during troubleshooting than 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, of course, to be troubleshooting a commercial product at this level, because if Data Club didn't coordinate the changes properly, there's really nothing anyone could do to help it out. I've never needed to troubleshoot Data Club—it's always worked fine—I was just curious to see how it worked. I've included it here because it is a good example of this type of 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, illegal socket is not troublesome as long as no other applications try to use that same socket; it was probably the most efficient way to inter-communicate.

Signaling and Transmission

Because you have talked for many years, you have learned how to adjust how loudly you speak based on environmental parameters. You know that listeners at farther distances have a harder time understanding your words because the sound of your voice grows softer with distance. To speak to people far away, you have to speak more loudly to compensate for the distance. You might even have a maximum distance, beyond which you would not even try to speak with someone, no matter how loudly you were able to shout.

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

Because you have been speaking and listening to sounds for many years, you have a good feel for the relationship between distance and loudness. You are 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. From your experience, you have developed an understanding of the physics of sound transmission.

Network Transmission Theory

Transmission theory books 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 themselves are simple and share many similarities to the principles of sound transmission that you already know.

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

Signals Decay over Distance

After the 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. In order 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 1's that make up the communication. Square waves, pictured in the figure below, are useful to encode digital information because square waves themselves are digital—they oscillate between 2 voltage states.

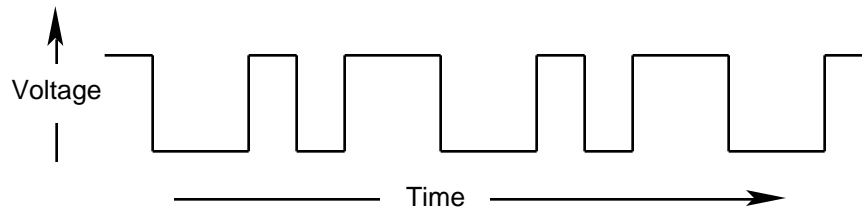


Figure 2-8. Square waves easily represent digital information because they oscillate between two voltage states.

On twisted pair wire, there are 4 ways that square waves decay—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 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 between the number of branches at the junction. When a device on one branch of a 4-branch passive star sends a signal, the signal will travel to the center of the star, where it will divide and distribute its signaling power among the other 3 branches. If the amplitude of the original signal was 6 volts, the split signals will 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), 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 1's and 0's, the signal becomes harder to interpret the farther it has to travel. Frequency distortion is sometimes referred to as "AC Loss". It 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 you stand 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 had different transmission characteristics than the air, some of the energy of your voice wave echoed. 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. 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 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 is 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, changing the wire's impedance locally. Many network specifications include limits on how many network adapters can share a single wire. In the 10Base2 Ethernet specification, for example, a limit of 30 devices per network segment keeps the level of reflections to a reasonable level. Other examples of wiring situations that cause reflections are:

- any junction—a punch down block, patch panel, wiring block
- changes in wire thickness, quality or insulating materials
- any place the wire changes from shielded to unshielded, even when a balun is used

Reflections are a part of any wiring system, and reflections caused by the conditions listed above 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. Because of this, it's very hard for a manufacturer to set limits that apply in all situations. Normally small reflections do not cause problems unless there are many of them and they add together in the right ways. More information about reflections can be found in "Troubleshooting by the Layers-Physical Layer".

By far the biggest reflection that can occur is at the end of the wire, but these reflections are easily prevented 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 .

In coaxial Ethernet wiring, lack of termination will prevent the network from working at all. In LocalTalk systems, lack of termination or improper termination can have a range of effects, from no effect at all to a very serious effect. In LocalTalk, however, even with the worst termination scheme, some of the nodes will be able to communicate with some other nodes sporadically. LocalTalk termination problems do not 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.

The tendency for a signal to be adversely affected by electrical noise is dependent on many factors. "Those factors" are what the troubleshooter must examine to see whether EMI is a problem. Like reflections, EMI

TROUBLESHOOTING MACINTOSH NETWORKS

in both forms is ubiquitous, and can be minimized but not eliminated. The likelihood that a signal will be negatively affected by electrical noise is influenced by the following factors:

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 may change. Higher quality wire has 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. For more information, refer to “Components-Wire”.

Noise Strength

Stronger noise signals have greater influence and are more likely to produce a negative effect. If there is a strong noise source, the troubleshooter may need to alter one of the other factors to compensate—move the network wires away from the noise source or use wire that has more twists, for instance.

Similarity Between Noise and Signal

Noise that has similar characteristics, particularly similar frequency, will be more likely to cause disturbance. That is why the wires and connectors carrying normal 60 Hz electrical power, a very strong signal, do not typically affect the quality of a network signal. Fluorescent lights give off a very wide spectrum of radiation and will always contain some frequencies that are similar to the network signal. Any device that uses an electric motor, such as copy machines and dot matrix printers can be suspects, as well.

A most extreme case of signal similarity is in 10BASET networks, where two pairs of wire in a single jacket will carry the receive signals and transmit signals, respectively. If the twist is not adequate, a signal on one of the pairs will cause the other pair to resonate (crosstalk). This will be interpreted as a collision by the transceiver and will render the station connected to this wire inoperable. That is why higher grades of wire are required for 10BASET; they have the greater amount of twist necessary to prevent crosstalk between the 2 pairs of wires.

Shielding

Shielded cable is very 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, say 10 times faster than Ethernet, shielding will probably be 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 will interact with each other. That is why 10BASET 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 “bandwidth” does not strictly apply

to digital networks like LocalTalk or Ethernet, it is 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 it is 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 per cent, that's in use at a given time.

NetStats and TrafficWatch, among others, are 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 the troubleshooter get a qualitative "feel" for a particular network process. Is the network busy right now? How is the network affected if a user backs up their hard disk 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 will a file transfer take to complete? How is a complicated process like printing coming along? By looking at a graph of utilization vs. time, a network manager can gain insight into these questions.

In troubleshooting, utilization graphs are best used for comparison to a normal situation. If you know that a process, say printing, has a certain utilization profile under normal circumstances and you look at the graph of utilization during the trouble, it may provide an important clue to locate the trouble. Utilization information can also be useful without comparison data—It can help a network manager estimate how long it might take for a long process to complete or why a certain data base query takes longer than another because it shows the volume of information transferred over the network in order 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, in order to make good judgements based on the data that is gathered. In order to better explain exactly what the utilization data means, I'm going to discuss the major issues surrounding these concepts. I'll use a file transfer over LocalTalk as the basis for the discussion.

Calculating Bandwidth

The speed of LocalTalk is 230.4 Kbits/sec. 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 KBytes/sec. 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 send data. The time that the network devotes to maintaining control and order is time that it cannot spend sending data. Bandwidth is always less than the speed of a network due to this overhead.

An useful analogy can be made to a highway. For safety reasons, drivers leave gaps in between their car and the next. The carrying capacity of the highway is diminished because of these unusable spaces.

LocalTalk's specifications require that there must be some dead time where no communication occurs. Just like the gaps between cars in the analogy, 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 is a by-product of the way that it avoids collisions. All networks have some dead time in their design. We can call this "hardware" overhead, since it arises from the needs of the network and not from the needs of the protocol using the network.

Besides 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 indicate 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.

TROUBLESHOOTING MACINTOSH NETWORKS

In our highway analogy, the purpose of the highway is to carry people and goods, but people and goods have to 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.

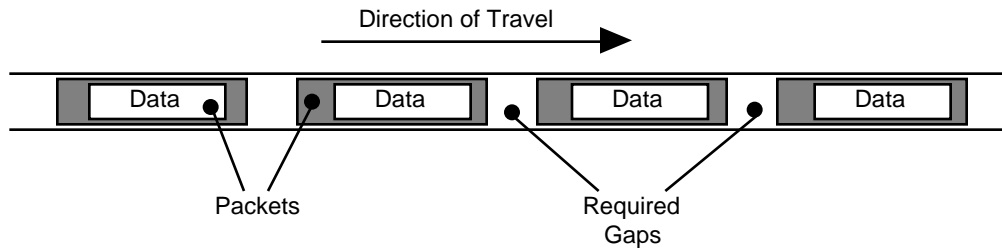


Figure 2-9: Just like passenger cars on a highway, there is network carrying capacity that is not useable for data due to the needs of the network and the protocol using the network.

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 or not you take into account the overhead. When you measure the utilization of a file transfer over LocalTalk to be 40%, does that mean the network is sending data at 40% of LocalTalk's speed, or 40% of LocalTalk's bandwidth or 40% of AppleTalk's ability to use LocalTalk as a conduit for data? How does the value of 40% relate to how many bytes are being sent per second and how long it will take the file transfer to complete?

Most often bandwidth calculations and utilization measurements 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, a HyperCard tool I've invented called PacketCrunch and from Farallon's NetStats.

The Effect of Packet Size

For any network, there is a limit to the amount of information that can be sent in a single packet. Since the size of the required gap is independent of the size of the packet, it's obvious that the network is more efficient and can carry more data if it uses larger packets. This is graphically shown in the figure below.

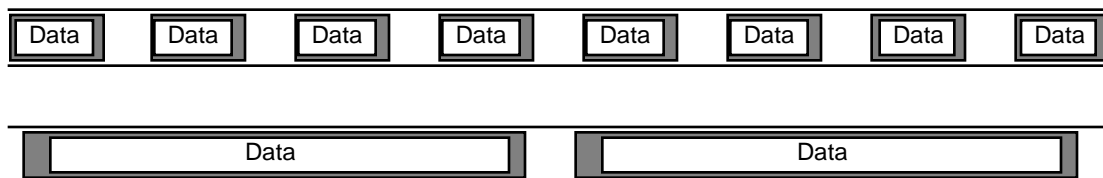


Figure 2-10. When the network uses larger packet sizes, it can send more data more efficiently because the effect of the gap in between the packets is minimized. In the figure, the bottom network is sending twice as much data as the top network in the same amount of time because it is using larger packets.

One way that you can 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 in 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%, the time spent sending data divided by the time needed to send the data. Using this efficiency rating, you could calculate that the bandwidth of LocalTalk is the speed of the network times the efficiency rating.

LocalTalk Bandwidth-Calculation # 1

(taking into account the hardware overhead for large packets)

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

Of course, this bandwidth calculation above 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 was calculated above.

LocalTalk Bandwidth-Calculation # 2

(taking 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

$$\begin{aligned}
 \text{Bandwidth} &= \frac{\text{Time needed to send data in packet}}{\text{total time of packet}} \\
 &= \frac{\text{number of bytes in packet} \times \text{time needed to send one byte}}{\text{time to send data} + 0.001 \text{ seconds}} \\
 &= \frac{N \text{ bytes} \times .0000347 \text{ seconds/byte}}{.0000347 N + .001}
 \end{aligned}$$

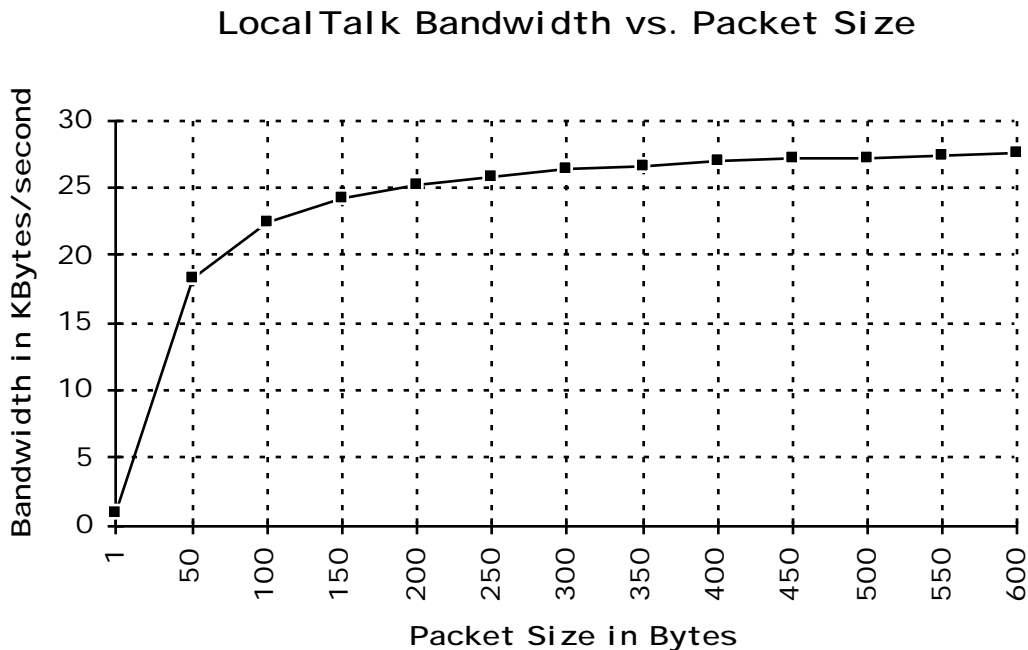


Figure 2-11. The bandwidth of LocalTalk varies according to the size of the packets used to carry the information. At packet sizes less than 150 bytes, the efficiency of the network drops quickly.

As expected, long packets can support a higher bandwidth. Because of this factor, the bandwidth cannot 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 1-byte packets for each keystroke. Other types of communication, like file transfer and printing, naturally take place in longer packets, but there will be some number of shorter packets used in the process for control purposes ¹.

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—² 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 1024 bytes, not 1000. The actual file size is 250,009 bytes—you can get that from the “Get Info” window. I’ll use figure this as the number of KBytes and for the rest of this section, 1 KByte means 1000 bytes.

Utilization—Using a Stopwatch

Using a stopwatch, you can calculate the utilization during a file transfer. Since utilization is the portion of bandwidth used, you could divide the transfer rate of the file by the bandwidth and call that 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.

¹Note: After refining our bandwidth calculation with the factor of packet size, we could keep refining the number based on more and more factors, but packet size affects the calculation the most. The next largest factor would be the level of activity on the network, which very slightly affects the size of the gaps between the packets.

In the test, the 250 KByte file took 24 seconds to transfer, so the transfer rate was 250 KBytes/24 seconds or 10.4 KBytes/second. This would show utilization to be 10.4/27.4, or about 38%.

Summarizing the calculation:

Utilization-Calculation #1

(using a stopwatch to time the transfer of a 250 KBytes)

$$\begin{aligned}
 \text{Utilization} &= \frac{\text{Transfer Rate}}{\text{Bandwidth}} \\
 &= \frac{250 \text{ Kbytes} / 24 \text{ seconds}}{27.4 \text{ Kbytes} / \text{second}} \\
 &= \frac{10.4 \text{ Kbytes} / \text{seconds}}{27.4 \text{ Kbytes} / \text{second}} \\
 &= 38\%
 \end{aligned}$$

Utilization With PacketCrunch

To refine the calculation, you could take into account that the file transfer did not 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 could refine the calculation above 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.

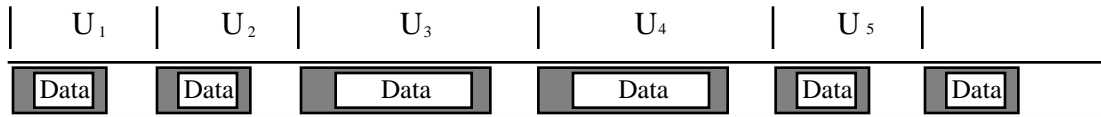


Figure 2-12. One way to calculate utilization is to use the summation of the “momentary” utilization of each packet.

PacketCrunch showed the size distribution of the 1127 packets and the network utilization during the file transfer as follows:

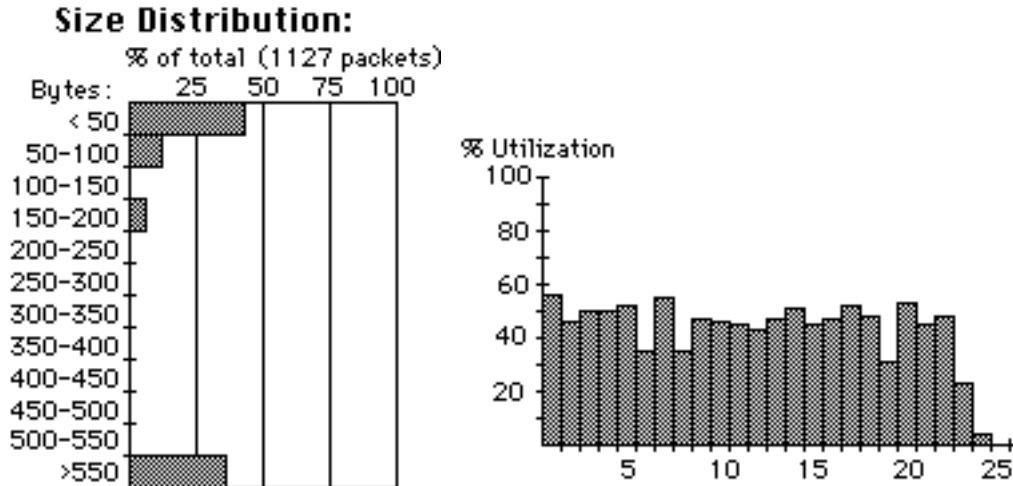


Figure 2-13. PacketCrunch shows a utilization higher than the stopwatch method because the network sends more data sent over the wire than just the contents of the file and because many short packets were used.

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

The second reason that the stopwatch calculation was low is that our 250 KBytes was sent along with a lot of AppleTalk protocol information. Timbuktu 4.0 uses ADSP for the file transfer mechanism and 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 250 KByte file was 265,680 bytes from sender to receiver, and 8580 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, then you get a transfer rate of 11.43 KBytes/second and about 42% 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 would not yield an accurate measurement because it could not 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 would contribute. The number of control packets with AppleShare would also be much higher. Those numbers of control bytes only apply to file transfers sent between two Macs on the same LocalTalk network. If the 2 Macs were on different networks, there would be an additional 8 bytes in each packet. Since the file transfer took place in 1127 packets, this would add an nearly 9 KBytes 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 is shown below.

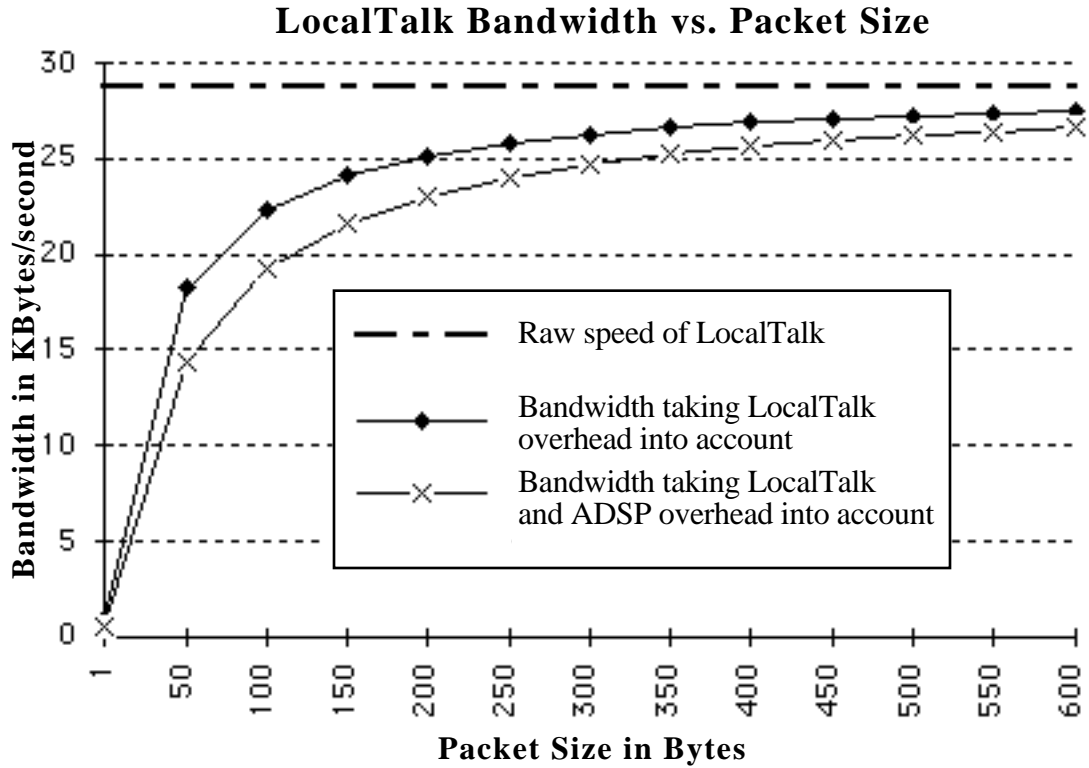


Figure 2-14. Timbuktu 4.0, which uses ADSP as the protocol control mechanism, adds 21 bytes to every packet sent on LocalTalk. This software overhead further reduces the bandwidth of LocalTalk as shown.

Utilization With NetStats

Let's compare the calculations we made before with 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 or not. 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 only reflects the percentage of time that there is a signal on the line. This is an entirely different method—but how does it's information compare to the numbers we've developed above?

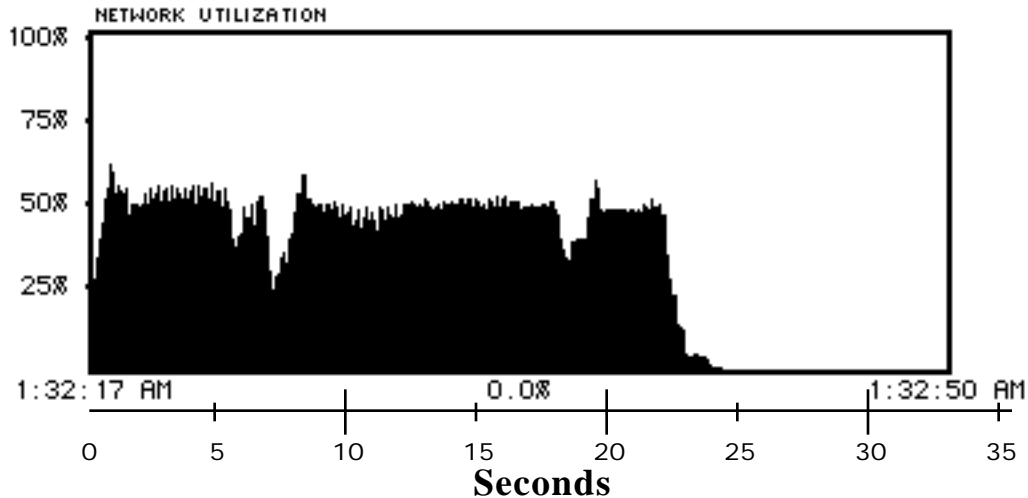


Figure 2-15. Farallon's NetStats uses a statistical means of measuring utilization. It tests whether the LocalTalk has a signal on it or not and reports the percentage of time the port showed activity.

I've added a time scale in seconds for convenience.

As you can see, NetStats shows a utilization slightly higher than either PacketCrunch or the stopwatch calculations. Utilization hovers in the 50% range, and you can 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 that NetStats makes its measurements. It samples the LocalTalk port to see if it is in synchronization with a signal or not. 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 is not very reliable for accurate measurements.

Despite this, NetStats remains a useful troubleshooting tool to gain a "feel" of the network and it's processes. For example, you can use it to gauge how fast the printer is accepting data by looking the time delay in between the spikes of activity. The longer the delay, the longer it is 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 that we used. The values ranged from a low of 38% using a stopwatch to a high of 50% using NetStats. Utilization measurements and graphs are useful troubleshooting tools, although the troubleshooter 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, as these factors significantly affect the values taken by the measuring tool.

Setup Troubleshooting

Setup troubleshooting is most appropriate when the user cannot establish contact with the device at all or when the Chooser is unable to find any devices or display any zones. If the Chooser can locate the device, but the user cannot establish the connection, Layer-oriented troubleshooting might be more appropriate. If the user can establish a connection, but cannot successfully complete the required transactions, Process-oriented troubleshooting might be more appropriate.

Setup-oriented troubleshooting is performed by testing the continuity of the network pathway that begins with the user's input devices and ends with network device that the user is trying to use. The network pathway is viewed as a collection of components and connections, each of which could potentially be missing, improperly installed or defective. Through an orderly process, each of the components and connections must be checked to confirm that it is present, properly installed and functional.

Testing each component in the pathway individually can very time-consuming, however, so the troubleshooter usually makes a first pass along the pathway by testing groups of components together. An example of this kind of check is when the troubleshooter tested the Chooser for zones to establish the continuity of the pathway between the user's workstation and the network router. Group testing helps the troubleshooter to "home in" on the general area of the problem. Once the general area is established, individual component testing then can be used to find the problem component.

Individual components can be tested by one of three methods — theoretical testing, practical testing and component swapping — each of which has its strengths and weaknesses. For an example that uses all three methods, lets imagine that the Chooser test described above was negative in an EtherTalk network. The troubleshooter suspects that one of the components in the pathway between the user's Macintosh and the router is defective.

A theoretical test could be performed on the network wire between the Macintosh and the network hub. Theoretical tests attempt to establish that the component has the required properties for operation. The strengths of this test is that it can be performed while the component is in place and that it is the most effective of the 3 methods in testing the component independently of other components. The greatest weakness of this test is that the tester won't always know or can't always test all of the required properties.

To perform a theoretical test of the network cabling, an troubleshooter may decide to test the electrical continuity of the cabling with an ohm-meter or tone test set. If the test is positive, an inexperienced troubleshooter might think that the cabling was satisfactory, but besides basic continuity, Ethernet cabling must also pass electrical quality tests. The troubleshooter could follow up the continuity check with a test of the cable's electrical qualities with a TDR (time-domain reflectometer). A good TDR would be able to test the cable for impedance, signal loss and crosstalk. These values could then be compared with the published values for Ethernet cabling. If the cabling passes both the continuity check and the electrical quality testing, the troubleshooter may feel satisfied that the network cabling is OK, but he has forgotten to check the polarity of the cable's conductors. Leaving out this important step could cause him to falsely identify the cabling as satisfactory.

The theoretical test is therefore only useful when the troubleshooter understands the qualities of the component necessary for successful operation. Components can also be tested for functionality with a practical test. For example, a theoretical test of an Ethernet card is impractical because of the advanced knowledge of electronics necessary for a thorough check. Even if the troubleshooter has such knowledge, because of the proprietary nature of the card's firmware, few but the manufacturer would be able to conduct a theoretical test. Some Ethernet card manufacturers include a software tool that checks a few of

TROUBLESHOOTING MACINTOSH NETWORKS

the the card's basic operations and lets you view the results. While these tools are often helpful, the troubleshooter may also decide to perform a practical test of the card's operations.

Practical tests check to see that the component functions in the capacity necessary for operation. A troubleshooter could check whether the suspect Ethernet card can send and receive packets under the control of a Macintosh-based protocol analyzer such as NetMinder Ethernet or EtherPeek. Protocol analyzers typically bypass much of the system software to directly control the Ethernet card. EtherPeek, made by the AG Group, could also be used to test the ability of the card to send packets by using its "Send" function. These packets could be collected using a second protocol analyzer on the same network.

A well-devised practical test will point in the direction of the problem. If the Ethernet card could successfully send and receive packets during the testing, but could not send and receive packets during normal system operation, the troubleshooter would next check the System software components that drive the card, possibly re-installing them or checking their settings. If the Ethernet card could not send or receive any packets during the practical test, the troubleshooter would check the jumper settings on the card to see that they were in their proper positions and check the connectors on the card. If the packets collected in the test were garbled, illegitimate packets, the troubleshooter would suspect that the polarity of the wiring was reversed.

In the example of testing the continuity of the network cabling above, the component was tested "in situ" or "in place". Another powerful method used in Setup-oriented troubleshooting is "component swapping". In swapping, the troubleshooter temporarily replaces a single component in the defective pathway with an identical component that is either new or borrowed from a working pathway. He then checks to see if the pathway was made whole and functional. Normally, if the replacement makes the pathway functional, the network manager concludes that the component was defective.

Each of these testing methods has its strengths and weaknesses. Swapping the network wire is not usually practical, so in situ testing would be appropriate.

Swapping is very time-consuming when performed randomly, so an experienced troubleshooter will try to "home in" on the problem by some logical method. For example, the troubleshooter may know that a particular component has caused trouble in the past and will check it first. One method of deducing a portion of the and the temporary replacement becomes a permanent replacement. is temporarily replaced by an identical component. The pathway is then checked to see if it has been made functional. , and the overall connection is checked Some of the testing performed during setup-oriented troubleshooting is purely practical Some of the components can be grouped together during a test. In the figure below, the network pathway consists of 10 components.

Troubleshooting LocalTalk Setups: General Information

LocalTalk is still a useful network in 1992. No one knows how much longer this will be true. Many people predict that LocalTalk is (or soon will be) obsolete because it does not provide the rate of data transfer needed to support the types of network that our computers and applications are capable of (or soon will be). Although all of this may be true, predictions like these are notoriously unreliable. After all, there are thousands of people who still make their living writing and maintaining programs in COBOL, a programming language whose imminent death was predicted before 1975.

LocalTalk is an extraordinarily cheap, efficient and reliable network. Yes, it's slow, but it's faster than today's LaserWriters, faster than a remote terminal application, faster than a networked modem — faster than most network transactions. LocalTalk's speed is the bottleneck for file transfer and for some kinds of data bases, but that's about it. For all other applications, operation on higher speed networks like Ethernet occurs at the same speed or only slightly faster than LocalTalk. In many cases, network managers cannot 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 and the speed of the networked components continues to increase (approximately

doubling every two years), as more and more services and resources are offered over the network, and as our concepts about how to design and build distributed computing systems continues to evolve.

Troubleshooting the setup of LocalTalk networks is in some ways easier than troubleshooting other networks because there is less equipment in the network path and the electronics are less complicated than in most other systems. On the other hand, LocalTalk can be very hard to troubleshoot because of two phenomena that are peculiar to LocalTalk networks: 1) network managers routinely under-design LocalTalk networks, exceeding the manufacturer's guidelines and the boundaries of common sense, and 2) users routinely modify LocalTalk networks without informing the network manager. Both of these factors are much less present in Ethernet and Token Ring networks. Network managers do not under-design because they know (or fear) that the technology is less forgiving than LocalTalk and users modify the network less because the knowledge and tools required to do it are not commonly in their possession.

Since the nature of troubleshooting is comparing “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 state that the network manager should be very well acquainted with — 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”, the troubleshooter 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” has been changed, and may not correspond to the network manager's expectations. Getting rid of these two impediments to network health — under-design and user modification — are key to solving many organization's networks woes, as well as key to making troubleshooting easier and faster.

The Relationship Between Design and Troubleshooting

The motivation for network managers to under-design is usually economic or due to 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 under-design their network without compromising reliability. For most networks, 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 under-design become entrenched in the minds of the network staff.

In any network, the frequency, severity and disruption of troubleshooting is good feedback to 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 that you create a design standard that set limits for what is acceptable network construction, you should also set limits for what is 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 may decide that troubleshooting any sub-system of a network more than once a month is excessive. You might also specify other limits, say, 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 to judge the validity of your network's design standard—you have a yardstick to measure whether your design is fulfilling your objectives.

For example, in a LocalTalk network using Active Star topology, you may decide to set a maximum of 4 branches per port, with a maximum of 4 users per branch. While this is not strictly beyond the manufacturers' guidelines, I do not 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 too many variables for easy troubleshooting. If, however, you choose this as your design standard, and you are troubleshooting only once every 2 months and typically have the problem corrected

TROUBLESHOOTING MACINTOSH NETWORKS

in less than 20 minutes, then, because this falls within the limits we've set above, your network design can be judged as "acceptable". If your design standard fails the test, however, you'll know that you need to cut back the limits of your network design— say to 3 branches with 3 users, or 2 branches with 2 maximum users — until you fall within the guidelines you set for yourself. By establishing the guidelines independently of the design, you automatically create the justification for change — it's not guesswork anymore.

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

Apple's LocalTalk connectors and cabling, using shielded twisted pair, are used by very few network managers because it is generally acknowledged to be less reliable, less flexible and less manageable than the connectors based on Farallon's circuitry. All of the comments in the remaining portion of this section are about the ungrounded (Farallon-type) connectors, which are referred to simply as LocalTalk connectors. Apple's connectors will be referred to only by "Apple's LocalTalk connectors", and will be treated in their own sub-section.

Conceptualizing the LocalTalk Setup

Setup troubleshooting is concerned with verifying the continuity and quality of the components of the connection. The LocalTalk components are shown for a typical star topology.

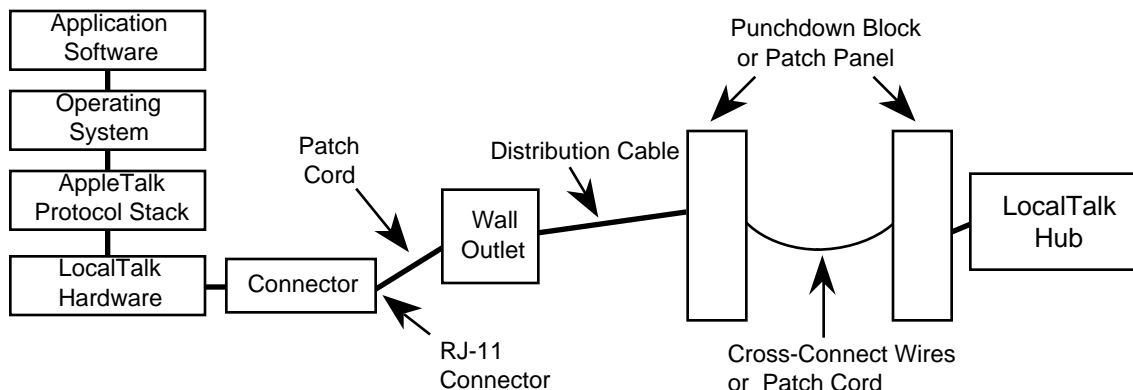


Figure 3-1. The setup for hub networks is shown in schematic form. The numbers in the figure refer to the sections concerned with the particular component.

Starting with the LocalTalk connector, the sections that follow will trace the part of the connection path that is external to the computing device, ending with the LocalTalk hub in Section 5.1.12. The material in these sections is an accompaniment to the material in Section 6.2, "Physical Layer Troubleshooting", which deals with this external path more as a single entity. Here the emphasis is on the way that the individual components work, and how they work together.

After tracing the external connection route, we'll return to our starting point of 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 2 most common problems with these connectors is that they are either faulty or have become damaged.

There are many suppliers of LocalTalk connectors, and some network managers (or their non-technical purchasing departments) often buy from the smaller manufacturers because they can save a few dollars per

connector. This is frequently a bad choice because most of these connectors are inferior. If you are one of these people, or your buyers are forcing you to use clone connectors, here are some thoughts about connector quality and what to look for to determine if the connector you are purchasing or preparing to purchase is of high enough quality.

One LocalTalk problem that is related to connector quality is when the connector itself is internally faulty. Faulty connectors result from manufacturing errors or from bad components. The rate of faulty connectors (faulty upon delivery) for all manufacturers 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 my own experience and in talking to many others, I estimate that it has never risen above more than 1 bad connector in 250. Almost 100% of faulty connectors are discovered during their initial installation; however, connectors rarely become faulty during service, except through damage. Because of the chance of faulty connectors, it's always good to have a few extra connectors on hand for spares.

The rejection rate of a connector is related to the overall quality of its manufacturing processes. A strong indicator of the quality of the manufacturing process is the quality of the soldering. Some manufacturers do not even bother to mask the circuit board or use an inferior soldering process instead of the highly reliable wave soldering process.

Besides being poorly made, some connectors are also poorly designed. The main component in a LocalTalk connectors is the transformer, which must faithfully reproduce the computing device's DC voltage impulses on the network wire in a crisp square wave. All LocalTalk connectors have transformers, but some manufacturer's transformers are better than others, the best ones being the transformers used in the Farallon PhoneNET connectors and the Focus (formerly NuvoTech) TurboNET connectors. You can tell if your connector has a good transformer or not by looking at 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 want to start out with the best signal possible.

Good connectors will also have a variable resistor that acts as a surge protector. Because the patch cord used in the LocalTalk connector looks very much like a telephone cord, users may 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, which is high enough to do serious damage to a computer's motherboard. A good surge protector will absorb the ringing voltage so that it does not cause damage to the motherboard.

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

If you are thinking of buying a connector other than the Farallon PhoneNET connector or the TurboNET, check the quality of the connectors by checking:

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

If you do not know how to perform these tests and are still interested in buying a connector other than the 2 mentioned, ask the manufacturer to confirm the quality of these aspects of their design and manufacturing. If you are considering the purchase of a large number of connectors, a prospective supplier should also be able to supply you with a photograph of an oscilloscope trace showing the quality of their connector's signal.

Also, I strongly recommend against the use of the TurboNET ST connector. This is a "self-terminating" connector with the LED's that blink when there is network traffic. These are the connectors that have a clear case to allow the viewing of LED's in the circuit that light up when there is traffic. The ST provides a

TROUBLESHOOTING MACINTOSH NETWORKS

kind of termination by *dampening* the signal, or reducing its amplitude, as it passes through the connector, instead of the more conventional termination. While the signal dampening provides an equivalent effect to conventional termination in many typical LocalTalk configurations, it does not work well in other configurations. Since the purpose of having a self-terminating connector is to free the network manager from the worry of termination, and since the ST cannot effectively eliminate that worry (because it does not work in all configurations), I do not 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 it's external electrical contact points, which are located both in the DIN-8 (or DB-9) plug that connects to the computing device as well as in the RJ-11 jack that receives the patch cord. In both the plug and the RJ-11 jack, there are contact pins that can become bent or broken through repeated rough use. I find that a large pair of tweezers is a good tool for straightening out the contacts in the LocalTalk connector. If the RJ-11 connector, it's usually best to just clip it off and install a new one.

Less commonly, the pins can become corroded. Because the contact pins are treated to resist corrosion in normal environments, corrosion normally happens only in unusually aggressive environments. Corrosion should be considered as a possibility in networks located very close to the ocean or in the vicinity of harsh industrial chemicals. A more common occurrence is that the contacts can become dirty. Usually, you can clean them with a good strong breath or a shot from one of those compressed gas canisters used by electronics hobbyists. If the pins have become corroded, there is nothing you can do except replace the connector.

Even when the connector has not 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 none of these kinds of damage, but still believe the problem is a faulty LocalTalk connector, then try swapping connectors with a node that is working. In rare cases, the connector will stop working for a reason that cannot be discovered by simple inspection. Most reputable manufacturers will replace connectors that you think are defective or have stopped working because they want to diagnose the connector's health as part of their quality control plan.

Problems with the Patch Cord

Besides making sure that the RJ-11 connectors on the patch cord fit snugly into their jacks, make sure that the components of the patch cord are of 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, with untwisted wire (flat cable) more commonly used. In the backbone configuration, the patch cord must be untwisted. Patch cord is called by many other names, some of the more common of which are: line cord, silver-satin, tinsel, and modular extension cable.

Because 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), pre-made cables, as well as the components and tools for making one's own patch cord from bulk supplies are readily available from a wide variety of sources. Some of these cables, components and tools are of much higher quality than others. Like the LocalTalk connectors, patch cords cause problems when they are faulty or damaged.

When checking for faulty cables, check to make sure the patch cord has four conductors. The four conductors are typically colored Black, Red, Green and Yellow. Some patch cord has only two conductors and 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, which are usually colored Green and Red. With an RJ-11 connection, LocalTalk always uses the outer two conductors, which are usually colored Black and Yellow.

Some network managers think that because patch cord is not expensive, they can be careless when making it or make it from unusual materials. I have seen patch cord made by cutting off the end of a 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. These irregular practices usually result in bad patch cord due to poorly made crimps.

LocalTalk uses a signal that is not polarity sensitive, which 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 4 conductors are present and that the 2 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, because 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 are 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 the ease with which you can 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 to custom lengths, often making the patch cord during the installation process to the length required for the location. It's often a temptation to buy the cheapest components and tools, but the poor connections that are made from these will very quickly make the tiny amount of money saved seem insignificant compared to the money lost to downtime and troubleshooting. To make sure that you are getting good components, buy them from a reputable telephone equipment dealer, not a neighborhood electronics hobby shop or a general purpose hardware store. Buy your parts and tools from a store where professional telephone technicians buy their gear, like Anixter Bros. or Graybar, not from your local Radio Shack store or Ace Hardware dealer.

Whether you crimp your own RJ-11 plugs or use pre-made cables, check the workmanship of the crimp to make sure the connection is solid. Here are the points to look for:

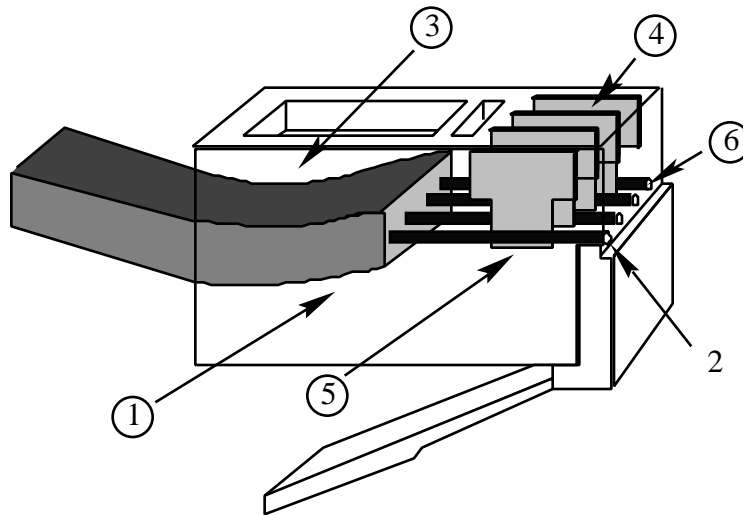


Figure 3-2. Check the workmanship of your RJ-11 crimp.

1. Check to see that the portion of cord that is not stripped is pushed all the way into the plug's chamber. You should not 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.

TROUBLESHOOTING MACINTOSH NETWORKS

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 check to 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 is wedged in place and will not slip out.
4. Make sure that the contact edges of the 4 metal plates are seated evenly and deeply in their slots. Before the crimp is made, these pins will 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, it means that you have a poor quality crimping tool or that 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 just slightly through the conductors.
6. Make sure that the ends of the conductors are neat, with no frayed filament ends sticking out. Frayed ends can cause short circuits between the conductors. If you see frayed ends, it usually means that your cutters are dull.

Problems at the Wall Outlet

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

When using RJ-45 jacks, there is divided opinion about how to make the connection. Some network managers claim that RJ-11 plugs should not be used 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 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 8 conductors, on the end of a 4-conductor patch cord can be a tricky operation, because you must make sure that the 4 stripped conductors go in their proper positions in the plug. If you decide to do this, you should 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 is no problem whatsoever with using an RJ-11 plug in an RJ-45 jack.

I have found that there are small variances in the dimensions of jacks and plugs from one manufacturer to another. The small size differences can sometimes make a difference in how reliably the mismatched connectors work. 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 Section 6.2, "Troubleshooting the Physical Layer". If your connection does not pass the ohm-meter test, then 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 a simple conversion of connector styles, baluns will sometimes have the responsibility of 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 is an impedance mismatch at the point where they join together. The balun is constructed in such a way that the effect of this mismatch is minimized. However, if you are 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, so it is wise to wire your network very conservatively. I recommend one run of wire per port of the wiring hub only, with no passive star wiring. Also, you should not mix LocalTalk connectors with Apple's LocalTalk connectors if you are using shielded twisted pair for your distribution cables. Use non-Apple LocalTalk connectors exclusively.

With 25-pair line taps, the greatest source of troubleshooting problems is simple installation mistakes, such as when the technician wires 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, each connection to 25-pair line taps should also be checked individually after installation or following wiring changes, either with a tone test or with an ohm-meter, both described in Section 6.2.

Like the pins in the LocalTalk connector jack, the pin contacts in the wall jack can become dirty or corroded over time, so they should be checked as described in the previous section.

Problems with the Distribution Cabling

There are two network problems associated with distribution cabling, although these are not very common in LocalTalk. The first is that the cabling may not be of high enough quality for the signal. The second is that the distribution cabling must pass through an overabundance of connections — intermediate punchdown blocks, patch panels, connecting blocks, changes in wire type and baluns — on its way from the LocalTalk hub to the wall outlet .

The reason that these problems are rare in LocalTalk is that the slow speed of the LocalTalk signal is not conducive to the types of problems that occur. As mentioned in Section 2.5, “Signaling and Transmission”, higher speed signals require higher quality transmission media and are 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 for is the use of voice grade cabling — cabling with a low number of twists per foot and poor electrical properties. In LocalTalk, this 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 5 twists per linear foot; 25-pair cabling of high enough quality for LocalTalk will have more than 3 twists per linear foot.



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

A more certain method to determine 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 is not usually mentioned, any unshielded twisted pair cable that is listed as being suitable for RS-232, 1BASE5, Digital Telephone or IBM 3270 (or faster signals such as Ethernet or Token Ring) is suitable for LocalTalk.

Wiring manufacturers each have their own classification system, but there are also standard classification systems that may be referenced in the manufacturers’ literature. These are slightly confusing because of the many aspects of cabling that they classify, but the classifications can be sorted out according to the term used to describe the cabling.

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

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 ft. length max.
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

Thus cable of the class “CMR” is data cable that has a fire retardant jacket that produces noxious fumes and which must be placed in conduit of non-flammable wireways when run through air plenums in vertical shafts such as are used for elevators.

TYPE. Refers to its physical characteristics and electrical properties. The IBM Cabling System originated this classification system in 1984 and many manufacturers specify their cabling to be compatible with a specific IBM Cable Type. Any cabling that is specified to be compatible with IBM Type 3 is suitable for LocalTalk. IBM Type 3 cabling refers to 2,3 or 4-pair unshielded twisted pair wire that has a DC resistance of less than 28.6 ohms/1000 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 1000 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. Type 1 cabling (2 shielded twisted pairs) can

also be used. 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 3 of these terms refer to nearly identical systems, and the term employed will depend on the affiliation of the speaker. For example, Anixter employees will always use the term “Level”. There is both a formal system and an informal system for specifying grades. The formal system specifies 7 grades, Grades 1 through 5 being for progressively higher quality (higher quality means that it will support 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 of the requirements for the 5 twisted pair grades are shown below:

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, 10BASET, ISDN, Arcnet
Grade 4 (Shielded)	7.5 dB/1000’ @ 1 Mhz 25 dB/1000’ @ 10 MHz	100 MB/sec. applications including 10BASET, 16 MB Token Ring
-&-		
Grade 4 (Unshielded)	4.9 dB/1000’ @ 1 Mhz 16 dB/1000’ @ 10 MHz	40 MB/sec. applications including 10BASET, 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

There is also an informal system that refers to cabling as being either of “voice grade” or “data grade”. In general, Grade 3 and higher are always referred to as “data grade” and Grade 1 is always considered to be “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 the quality by counting the twists per foot as noted at the top of this subsection.

Too Many Wiring Changes and Connections

Sometimes, many small factors can gang up to be one large problem. In the diagram below, I’ve shown a LocalTalk setup that is of marginal design. While no one would want to design their LocalTalk network this way, there are many LocalTalk networks around the country that are built similarly to this one because the network manager had no other choices.

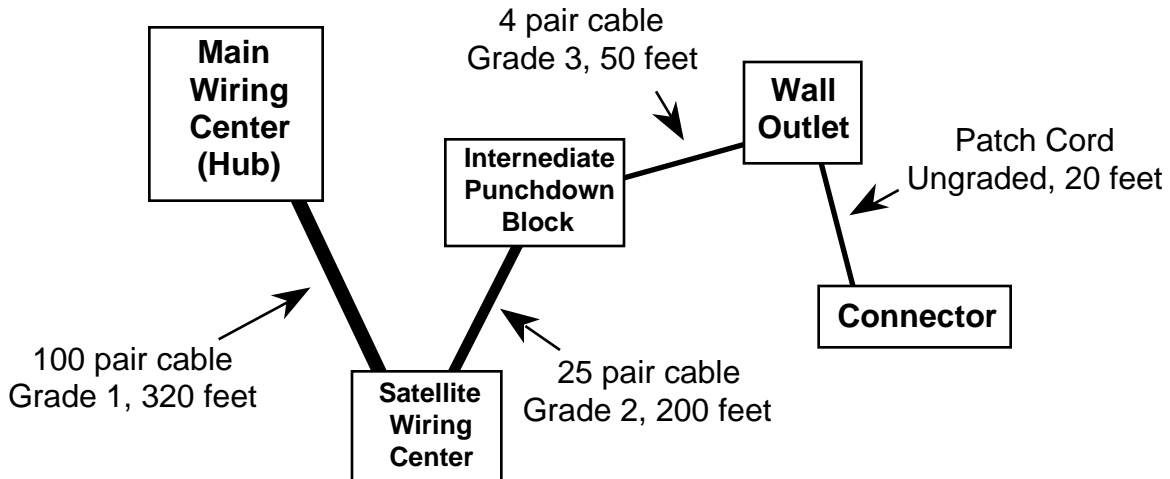


Figure 3-4. While this network is within all of the LocalTalk hub manufacturer's guidelines, it is of very marginal design.

There are several factors that could cause problems in this design. The first is the 100-pair cable that connects the hub with the satellite wiring center.

Problems with the Punchdown Block or Patch Panel

Punch-down blocks and patch panels are simple devices that allow the 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. Punchdown 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.

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 may find such irregularities as distribution cables that connected to a different wall outlet than their identification would indicate.

Problems with Punch Down Blocks

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

The LocalTalk hub

Most LocalTalk networks use a physical star configuration, with a wiring hub like Farallon's StarController, NuvoTech's TurboStar or Tribe's LocalSwitch at the center of the network. Some networks still use the traditional bus or use a passive star, where a small number of wires are physically joined in a central location. Some of the network managers who have put off buying a hub have done so at the expense of reliability, and while the troubleshooting tips in this section may be helpful to them, they really need to get themselves a hub. If you think you might fall into this category, here's a test:

A. You're not using a LocalTalk hub.

AND

B. You have more than 8 computing devices

OR

You have more than 3 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. Almost all of your reliability problems will vanish.

Some simplified shopping advice: The StarController has the most reliable signal processing, the TurboStar has slightly more useful management features and the LocalSwitch lets you attach a lot more users to a single LocalTalk network without routers.

Special Considerations for Troubleshooting LocalTalk

Troubleshooting the setup of LocalTalk networks is in some ways easier than troubleshooting other networks because there is less equipment in the network path and the electronics are less complicated than in most other systems. On the other hand, LocalTalk can be very hard to troubleshoot because of two phenomena that are peculiar to LocalTalk networks: 1) network managers routinely under-design LocalTalk networks, exceeding the manufacturer's guidelines and the boundaries of common sense, and 2) users routinely modify LocalTalk networks without informing the network manager. Both of these factors are much less present in Ethernet and Token Ring networks. Network managers do not under-design because they know (or fear) that the technology is less forgiving than LocalTalk and users modify the network less because the knowledge and tools required to do it are not commonly in their possession.

Since the nature of troubleshooting is comparing "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 state that the network manager should be very well acquainted with—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", the troubleshooter 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" has been changed, and may not correspond to the network manager's expectations. Getting rid of these two impediments to network health—under-design and user modification—are key to solving many organization's networks woes, as well as key to making troubleshooting easier and faster.

LocalTalk is still an important network technology for Macintoshes in 1992. No one knows how much longer this will be true. Many people predict that LocalTalk is (or soon will be) obsolete because it does not provide the rate of data transfer needed to support the types of network that our computers and applications are capable of (or soon will be). Although all of this may be true, predictions like these are notoriously unreliable. After all, there are thousands of people who still make their living writing and maintaining programs in COBOL, a programming language whose imminent death was predicted before 1975.

LocalTalk is an extraordinarily cheap, efficient and reliable network. Yes, it's slow, but it's faster than today's LaserWriters, faster than a remote terminal application, faster than a networked modem—faster than most network transactions. LocalTalk's speed is the bottleneck for file transfer and for some kinds of data bases, but that's about it. For all other applications, operation on higher speed networks like Ethernet occurs at the same speed or only slightly faster than LocalTalk. In many cases, network managers cannot justify the cost of making the transition to Ethernet based on the type of network computing their users

TROUBLESHOOTING MACINTOSH NETWORKS

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 and the speed of the networked components continues to increase (approximately doubling every two years), as more and more services and resources are offered over the network, and as our concepts about how to design and build distributed computing systems continues to evolve.

Special Considerations for Troubleshooting Ethernet

Ethernet technology, while not new, is still changing. The twisted pair standard, 10BASET, is less than 2 years old, for example, and Apple's adapter technology, the AAUI port, is even newer. Ethernet switches have appeared recently, and although they are rather expensive, they will probably gain in popularity as their price drops. The other big change in Ethernet networks recently is that large corporate Ethernets, which 5 years ago were almost exclusively interconnected by bridges, now use multi-protocol routers as the preferred interconnection device.

Ethernet for the Macintosh is a special animal, too, because of Apple's proliferation of bus architectures and their addition of the AAUI port. A look at the product sheet of any of the vendors who make the "full line" of Mac Ethernet cards will show how many combinations of Macintosh bus types (6) and connector types (4) that are necessary for a vendor to field (not to mention the RAM buffer size) in order to claim that they carry 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 10BASET standard on DATE???. The least expensive hub at that time was around \$250 per port and the the least expensive Ethernet card for the Macintosh was around \$400. A recent MacWeek (3/16/92) advertised cards for \$160 and an 8-port hub for \$299, less than \$38 per port. Ethernet has become a price-driven commodity.

The impact on troubleshooters of these trends is that Ethernet networks will often combine a multitude of different products from several vendors. While Ethernet standards such as 10BASET and 10BASE5 are fairly exact, there will still be the chance of incompatibility between 2 vendors implementation of the same standard. For example, Farallon's first iteration of its 10BASET hub had an AUI port that was incompatible with many other vendors' repeaters and at the time of this writing we still cannot mix Ethernet cards from different vendors in the same Macintosh (to make a router, for example) because the Ethernet drivers are vendor-specific.

Besides the real, technical incompatibilities between vendors and components, there is also the problem of imagined incompatibilities between vendors and components. It's very 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 may 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 (swap one component at a time, record all results, check all possible combinations) to avoid an erroneous conclusion.

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 to the user action and accomplish their tasks. The event is that a user double-clicks on a manilla folder icon from a server volume and receives a listing of the contents of that directory on her screen. I'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.

In order for the user to communicate with the computer, she needs to 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 the user can take (mouseclicks, keystrokes, menu choices, etc.) 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*, a book written by Apple Computer and published by Addison-Wesley, acts as the guidelines 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. Conversely, the application will also have a set of meanings for the various mouseclicks and keystrokes that the user employs to communicate to the application. In the Finder, the event of double-clicking on the manilla folder icon, the icon corresponds to a subdirectory on the server volume and the act of double-clicking on that icon communicates the 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 will create a new window on the user's monitor and place 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 is the relationship between desires and actions and between images and data, a problem in the Application Layer occurs when the user cannot understand or cannot consistently remember the relationships. Application Layer troubleshooting tries to minimize or remove 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 change the command procedure to something that user understands, educate the user or create reference information.

Although the Macintosh is the easiest computer of its kind to learn and use, it is still 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.

TROUBLESHOOTING MACINTOSH NETWORKS

The best Mac users are people who are “conceptually oriented”. These people create an understanding of their computers that gives them a way of conceptualizing the current state of their computer, the state they want their computer to attain and the alternate methods of making the adjustment. Conceptually oriented users appreciate the fact that you can open a file both 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 “procedurally oriented” want to have a precisely-defined series of steps to perform.

Networks invariably contain both kinds of users and their many sub-types, and both categories of users need to carry out their operations without confusion or annoyance. To some extent, the network manager can help a user customize his computing environment according to his individual preferences through the wise application of utilities such as CE Software’s QuicKeys, Now Software’s Now Utilities or Alladin’s ShortCut. These utilities can help either type of user extend 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 in such a way that you could give the user a reference guide (on paper) that says:

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	
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	
9. Press Command-S	Invokes the “Save As” command, since 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”)	
12. Press Return	Saves the file
13. Press Command-Option-Q	Quits application and dismounts server

You may 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 Presentation Layer, the user’s command of “Show me the contents of this subdirectory” is transformed into the proper computer commands to carry out the user’s wishes. Since the manilla folder icon in our example represented 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 will intercept the command and translate it from a Macintosh-specific file command into the equivalent

command 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 only needed for operations on storage volumes that are located elsewhere on the network.

The server will have 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 it happens that 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 systems 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 machines, VAX/VMS, many varieties of Unix systems as well as some proprietary operating systems used by Novell and 3Com Systems.

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 will transform the Mac's own imaging system, QuickDraw, into equivalent PostScript commands and send them to the printer. The printer will then translate the PostScript commands into its own native imaging language in order to print the image.

The application is almost completely isolated from the concerns of 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 a file from a server volume as easily as opening a file from a local volume.

Presentation Layer Problems

AFP masks the differences between file systems and computers, and this imposes certain restrictions on the usefulness of AFP. Like most “standards”, AFP is comprised of a minimum function set—a “lowest common denominator” of file systems—and may 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 will be 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.

For troubleshooting, the biggest concern is that the server and client versions of the AFP software are compatible with each other.

Session Layer

Function	manages the computing resources used by 2 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 password and privilege information, ending the relationship and de-allocating the resources.
-----------------	---

TROUBLESHOOTING MACINTOSH NETWORKS

The relationship between 2 computers consumes resources, and the job of the Session Layer is to manage those resources. In our example of the double-click on the manilla 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 (account name and password). When the user dismounts the server, all of those resources will be 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 may be periods of time during a session when no data needs to be sent, the Session Layer protocol in each device will have some kind of mechanism to discover whether its counterpart in the other device is still available over the network. A common way is to send and receive "tickle" packets to indicate that the session is still alive and well.

If the Session protocol does not hear these reassuring tickle packets from the other device, it may try to locate the other device. If it can't locate the other device, it will independently close down its end of the connection. This might happen if the other device crashes or was turned off suddenly. 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 is 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 2 devices may in fact be more like 2 peers. Also, ADSP allows other communication formats besides simple question/answer or command/reply. Because of its special features, programmers are beginning to use ADSP 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 is a Session Layer protocol that performs all of the tasks of both Session and Transport Layers, it breaks some of the rules of the 7-Layer model. It's normal for real-world protocols to break the rules once in a while.

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

As you can see, there may be more than one protocol available at any layer. An application programmer can choose to use any of the Apple-supplied protocols in any layer. The programmer makes the choice of protocol based on the suitability of the protocol for the kinds of communication that would normally occur during the use of his application. At the Session Layer, ASP and PAP are well suited to the jobs they perform, but they could also be used by a programmer for a different purposes than either using a file

server (ASP) or communicating with a printer (PAP). An alternative is that the programmer can invent a Session Layer protocol of his own. This is perfectly 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 International Business Software's Data Club, Sassafra Software's KeyServer and Farallon's Timbuktu when it controls a device over the network.

The invention of a proprietary protocol is normally performed when the programmer feels that the standard protocols are not efficient or fast enough to handle the communication needs of his program. Because the standard AppleTalk protocols have to handle a wide variety of tasks, they tend to be very generalized (good at a number of tasks), 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 7-Layer Model is Zone Information Protocol (ZIP). ZIP provides zone name 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 will be discussed in the Transport Layer's description below since 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 are usually not a problem on dedicated servers unless the network link is faulty or interrupted. If the network link between the 2 devices is inoperable long enough, it may trigger the automatic shutdown even though both devices are healthy. This creates more of a problem for the client than it does 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, the user may re-establish contact with the server by logging in for a new session. The user, however, may not be aware that the file he is working with is only a piece of the original file. If he decides to save his changes to the file on the server, he will replace the complete, older file on the server with the newer, incomplete file in his workstation's RAM. The "Save" operation should warn the user by asking if the user wants to "Replace existing file?", and a user might ignore this message.

AppleShare servers running on a dedicated Mac with no auxiliary processes mounted (like mail servers or routers) are very stable and can run for years without crashing. If a client workstation crashes during the session or is suddenly unavailable, 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 on the Macintosh, saving the changes to a file automatically replaces its previous version, a crash during this operation can cause significant damage or even result in 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, it's possible that irreversible file damage may occur. Another threat is that a user whose workstation is being used as a server may end the session abruptly by shutting the Macintosh off. Users should be trained to 1) avoid these kinds of problems in a file sharing environment and 2) know which operations have an adverse effect on file-sharing clients. These include turning their workstation off, turning off file sharing (which is necessary in System 7.0 to eject SCSI volumes on removeable media), switching their Data Link connection, etc. They should also know how to determine the identity of their clients with the File Sharing Monitor Control Panel.

TROUBLESHOOTING MACINTOSH NETWORKS

One of the newer features being implemented in AppleTalk-accessible servers these days is the ability of the server to set and adjust timers like the connection timer mentioned above based on local network conditions. This 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 been developed for optimum effect at this writing.

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 gets done correctly.

A 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 have to be able to handle these situations as well as be able to 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. 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.

The functions of reliability management and packet sequencing in our example of double-clicking on the manilla folder icon is 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 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 8) 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 requestor's memory space is full or the answer is complete, whichever happens first. In the case that 8 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 as well as providing 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 from service. 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.

The tasks of 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 the design of the protocols and in the implementation of the protocols in the current crop of internet routers. RTMP and ZIP processes, their common problems as well as the techniques to diagnose and fix their problems, are described in great detail in the Process and Technique chapters.

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 our example. 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 and the service name is viewable by the user in the Chooser. An example of this would be 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. The first is that it gives the user something more descriptive and meaningful to use than the numerical address. The second reason that named services are necessary comes from the dynamic addressing characteristics of AppleTalk (discussed in Data Link Layer below). Because AppleTalk addresses nodes dynamically, node (and socket) numbers can change arbitrarily from one day to the next. Names, however, only change 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 an address-based lookup insures that the the service can be contacted even if its address changes periodically.

It was mentioned earlier that ADSP performed 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 it does not use the concept of a transaction to segment and structure the communication during a session. In ADSP, the connection consists of 2 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 only identified 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 carrying 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 will include 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	Booting Up a Router
Device Processes	Maintaining Routing Tables Zone Tables
Techniques	Checking Router Configurations

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 it receives from the higher layer protocols and packages that information for transmission on the network. DDP has 2 different formats, a short format when the destination device is on the same network as the sender and a long format when the destination device is on a different network. It determines which format is needed by comparing the network number of the destination device to its own network number. If the network numbers are the same or the destination number is "0" (which means "this network"), the short format is used. When the network number of the destination device is different from its own, it uses the long format.

Once it has packaged the data in the right format, it can give the packet to the network hardware for delivery. Short format packets will be sent directly to the destination device and long format packets will be sent to the router for delivery.

Network Layer Problems

AppleTalk's Network Layer is very uncomplicated. As mentioned before, it does the work of delivery and Transport Layer manages its activities. If anything goes wrong, it is because Transport Layer is not functioning properly. Therefore there is no troubleshooting per se for the Network Layer unless the Transport Layer is also considered.

Data Link Layer

Function manages the tasks of packaging the information for transmission, gaining access to the network, 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 Transport Layer manages the activities of the Network Layer, Data Link Layer manages the activities of the Physical Layer. The Physical Layer is responsible for sending and receiving packets. Data Link Layer makes sure that this job is performed correctly.

AppleTalk has a Data Link protocol for every network system on which it can operate. The Data Link Layer protocols are named for the network system they use. LocalTalk Link Access Protocol (LLAP) runs on LocalTalk, EtherTalk LAP on Ethernet, TokenTalk LAP on Token Ring, etc. Apple has stated publicly that it intends to expand this list to include all the major networking systems 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 serve the purpose of synchronizing 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 may 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 when they want to send a packet, they 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 the signal passes by and then a little bit more. At this point the 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 is the condition when 2 devices are simultaneously transmitting, in which case neither signal will be 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 includes checking to see the packet is whole and that the packet remained in synchronization with the receiver's clock. The checking procedure may also use a mathematical check sequence to verify the integrity of the data inside the packet.

The kinds of error checking that Data Link Layer performs are different in nature than the kinds of 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 sent to the name of a real employee. If it fails any of these tests, he will reject the package. If it does pass all of the tests, he will forward the package to the employee who ordered the equipment. The employee will perform tests like the Transport Layer before he considers the goods "delivered". The employee will open the package and compare the contents of the package to the order that he placed. If there are missing items, he will call the mail order house and request 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 will have no way of knowing that he is missing anything. Only the employee has the expectation of arrival. The Data Link Layer will check the packets it receives and reject damaged packets. It does not request re-transmission. The Transport Layer has the sole responsibility of requesting packets to be resent if, for any reason, they are not received by the Transport Layer protocols.

Data Link Layer Problems

Data Link Layer problems are specific to the network type, and they are discussed in the Setup Section according to their specific type and below in the description of the Physical Layer. Data Link Layer problems stem from either physical hardware or network cabling problems.

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 signalling 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 signalling method used, the tolerance for signal errors and the maximum signal propagation delay between stations.

TROUBLESHOOTING MACINTOSH NETWORKS

A committee-produced standard like IEEE 802.3, the governing committee will have 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 specification 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 useable 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 in some cases 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 that is 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. Signalling problems are easily conceptualized—either the signal was not generated properly, it developed errors during transmission or it was not received properly.

Most of these problems are caused by improper installation of the network. Sometimes the network installer will deliberately install the network improperly in an attempt to make the network less expensive. 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, installs with larger and larger deviations from specification. The networks he installs 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 patchworked network. I refer to this kind of networking as "guerilla net".

In other cases, the mistakes are unintentional or result from a mistaken notion 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 may be surprised to find the wiring that worked well for LocalTalk signals performs only marginally for Ethernet or not at all. Another form of this kind of mistake is a misunderstanding about termination. This happens more frequently with LocalTalk than Ethernet. Although both networks need proper termination, a badly terminated Ethernet will simply not work. A badly terminated LocalTalk will work, but not well.

Another physical problem results from damage to the network cabling while in service. This 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. A transceiver from Vendor A may be incompatible with a card from Vendor B when used with software from Vendor C even though 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 of service. "Jabbering" is the condition where a device spontaneously transmits nonsensical signals on the wire.

In addition to electrical problems caused by improper installation, electrical problems that may 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.

Troubleshooting by the Layers— The Practice

Many Physical Layer problems are simple—the Physical network is broken because of a loose or unplugged cable, a plug in the wrong jack, a missing terminator—something that a visual inspection can find and a simple procedure can correct. Visual inspection should be the first check in troubleshooting the Physical Layer. Following the visual inspection and correction of simple problems, a more analytical approach can be used.

The quality and integrity of a network's Physical Layer depends on 3 conditions being simultaneously true:

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

These conditions are listed in the order that they are usually tested. The first condition is the criteria 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 through the use of Time Domain Reflectometers (TDRs).

While electrical continuity is a necessary condition for data networks, it is not a sufficient condition. Data networks require a higher quality cable than a telephone for 2 reasons. The first reason is that 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 experienced the effect of this frequency-based phenomena when they made the conversion from a LocalTalk (or PhoneNET, etc.) network to a twisted pair Ethernet network. Ethernet, of course, uses much higher frequency signals than LocalTalk. In many cases, the wire that carried the LocalTalk signal reliably was not suitable for the 10BASET signal and some additional wiring was necessary.

The second reason that data circuits require higher quality wiring than telephone circuits is that a computer's ability to make use of 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” noise, although you might find these conditions annoying. Computers have a very limited ability to accurately interpret a network signal in these kinds of 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. With the oscilloscope, you can see exactly how the signal has been distorted. With some of the better scopes, you could 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. These indirect tests are covered in the Data Link Layer section—Section 6.3.

Indirect tests are generally faster at helping you judge whether there is a problem with the path; direct examinations are better at determining the nature of a Physical Layer problem. That's why I suggest that

TROUBLESHOOTING MACINTOSH NETWORKS

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 there is a problem or not. 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 are well-matched and of high quality. This is less of a concern with Ethernet components than with LocalTalk components because they are manufactured to be compliant with one of several specifications, 10BASET, 10BASE 2, etc. If a manufacturer claims compliance with one of these specifications, barring a manufacturing problem, the component should work well and also work with any other components that comply with the same specification, even if they are from different manufacturers.

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

The most widely used wiring “standard” in LocalTalk networks is the construction rules published in Farallon’s PhoneNET System, which is not a published specification at all, but a system of guidelines. In the absence of a strict specification, there are some configurations of LocalTalk components from different manufacturers that do work reliably. These will be explained in this sub-chapter.

All of these criteria assume that the network hardware does not have any “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, however, the bug may be known and the manufacturer’s tech support department may tell you about a bug or known incompatibility. Of course, some vendors are more candid about known bugs in their products than others. Along this spectrum, network product manufacturers that began their history with an AppleTalk product are generally more candid than network manufacturers that began with another network system. For example, Shiva is one of the most candid of all manufacturers, while Novell is one of the least candid.

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. You’ll find some information about ANMA in the Resources Chapter.

Testing Electrical Continuity

Three tests are described in this sub-section—a tone test, an ohm-meter test, and a Time Domain Reflectometer (TDR) test. Each of these tests verify electrical continuity, which you should check when you suspect that something is physically wrong with your cabling. For example, you might check electrical continuity when nothing appears in the Chooser or Inter•Poll lists.

All of the tests require some equipment and 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 ohm-meter can help you determine the length of the cable as well as the presence of shorts and opens. If you are 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 receiver. Tone test sets are inexpensive, usually around \$100, and are a standard tool for telephone technicians. Some shopping advice: get a tone generator that has a recessed on/off switch. If the switch is not recessed, you may accidentally turn the tone generator on while rummaging through your toolkit. There is 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 wires (conductors) at one end of the wire and the tone is listened for at the other end of the wire. The receiver is usually 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 in an earpiece instead of in 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 for the receiver, rather than the more expensive butt set.

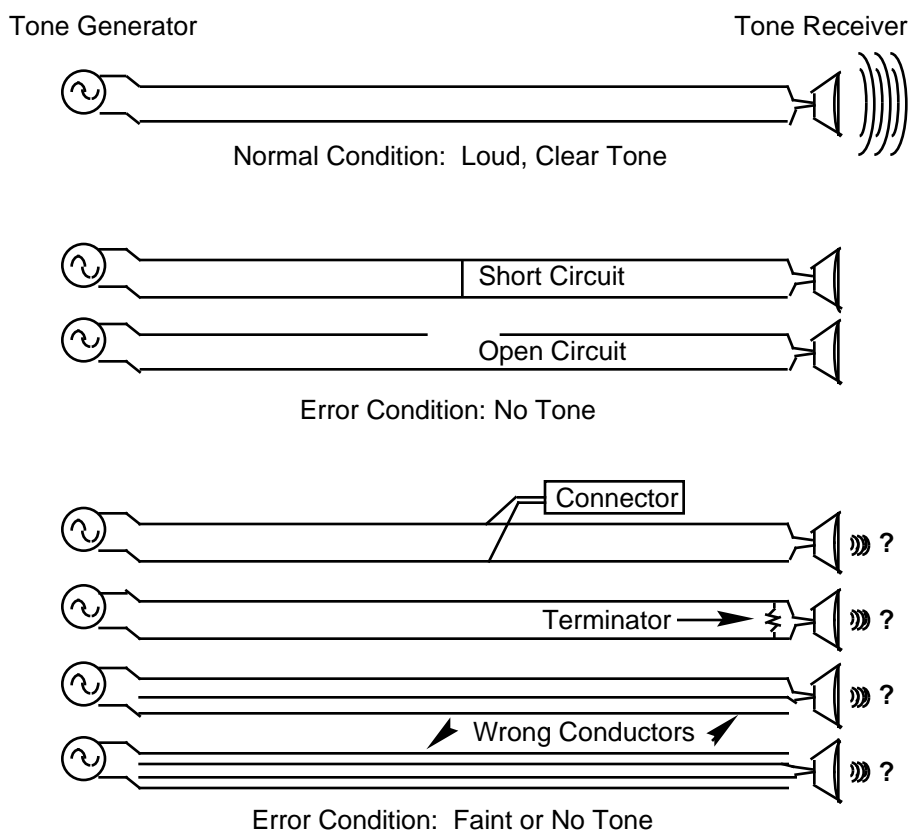


Figure 5-1. The tone test verifies electrical continuity.

Before you perform the 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 on the cable, attach the tone generator to the conductors. Usually the tone is activated at the user’s workstation and listened for in the wiring closet. I find it useful to use a modular adapter, which is a tool that plugs into an RJ-11 or RJ-45 jack and allows me to select any two of the jack’s conductors. In any case, be sure that you have a solid contact to a bare metal surface and not to any insulation. Turn on the tone generator and test and verify that it is generating a tone by holding the receiver near the contact.

Next, go to to the other end of the cable and hold the receiver so that the probe is touching or is very close to 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, some of which are catalogued in the chart above.

The Ohm-Meter Test for Continuity

Another test of continuity can be conducted using an ohm-meter. The ohm-meter test has advantages over the tone test because it can be used on either coax or twisted pair wire and can also detect the presence of a terminator or a network adapter. As in the tone test set, you should prepare for the test by removing all of the network equipment. In the user area, either short the conductors (connect them together) or place a terminator at the end of the cable. The ohm-meter test will then take place in the wire closet. A little shopping advice: you can purchase a digital ohm-meter that is adequate for the testing described in this section for about \$35. The higher priced meters offer a level of accuracy and precision that is higher than

TROUBLESHOOTING MACINTOSH NETWORKS

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. I like the probes that have a claw that firmly grabs onto the wire.

There is a tiny bit of math involved in using an ohm-meter, and I've included some example 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 its resistance into account. You must perform the math in order to know whether your ohm-meter's reading is normal or not.

The first thing to do is to know 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 1000 feet long should be expected to 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. The wire made by manufacturers in the United States is among the highest quality in the world, so if you're using wire from one of these manufacturers, your calculations will probably be within 5% of the actual value.

- For solid copper cabling at 68 degrees Fahrenheit, the resistance of 1000 feet of wire is 16.14 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 8-10 for thin coax cable (10BASE2) and around 1.5 per 1000 feet for thick coax (10BASE5).
- 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 anywhere that range anywhere from 100 to 1000 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 of the wire and measuring the resistance between those 2 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: Let's say that you short two conductors of a 24 AWG wire and measure 10.7 between the conductors at the other end of the wire. This is shown schematically below. Since you know that 1000 feet of 24 AWG wire would have 25.67 of resistance, you can calculate your wire's (round-trip) length by dividing your measurement by the standard value.

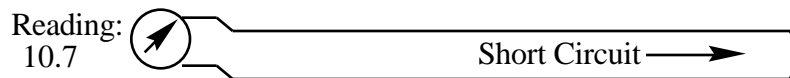


Figure 5-2. Measuring the resistance of shorted cable.

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

The equation for wire length when you short the conductors:

$$\text{Length} = (\text{Measured Resistance}/\text{Standard Resistance}) \times 1000$$

If you have a terminator at one end of the conductors, instead of a short circuit, 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.

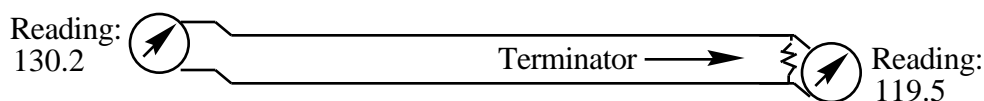


Figure 5-3. Measuring the resistance of terminated cable.

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

The equation for wire length on a terminated cable is:

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

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. Just like in the example above, the distance is calculated from the difference of the resistances. For our example circuit, the differences between the resistance measurements should be about the same, 10.7, but the values measured will be much lower.

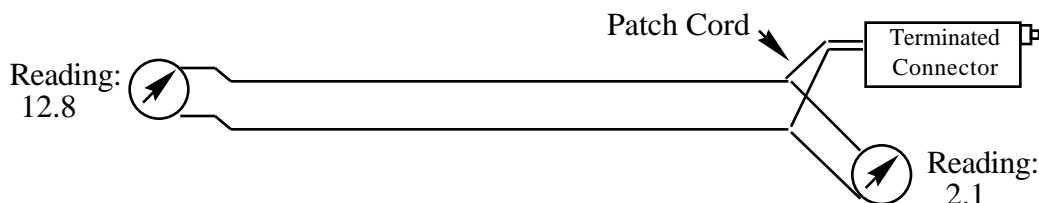


Figure 5-4. Measuring the resistance of a LocalTalk circuit with a connector.

When you look at the final example, you may wonder why the mounted terminating resistor in the connector didn'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 are 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 this test cannot discern whether a terminator is in the circuit or not; you'll have to rely on your eyes.

TDR Testing of Data Circuits

A good TDR is much more expensive than either an ohm-meter or a tone test set, but they also do a lot more than these tools. For example, Microtest makes a whole line of TDRs with all kinds of attachments for various testing situations. For general purpose cable testing, they make a product called the CableScanner, which is about \$1500. For more money, you can get a TDR that also has special features for 10BASET networks 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 TDRs that you'll probably notice is that they tell you, in English words on an LED screen, all of the numbers that we had to calculate above. The screen might say that your cable is "Open at

TROUBLESHOOTING MACINTOSH NETWORKS

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 will also report the presence of shorts and terminators, as well.

Although you won’t have to worry about calculation errors giving you false values for your length, TDRs are slightly less accurate at determining the length of solid conductor, twisted pair wire than the ohm-meter test. 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.

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

Because the capabilities and testing procedure are different on different TDR models, I’m going to leave out any procedural descriptions, and just let the preceding description of the capabilities of the instruments suffice.

Data Link Layer Troubleshooting

The Data Link Layer’s responsibility is to build a proper frame for the data which 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 must be built according to the rules of whichever data link—Ethernet, LocalTalk, 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.

Two kinds of test

There are two kinds of tests that can be performed to test the reliability of Data Link Layer functions. In one kind of test, the practical kind, we’ll get an empirical value of how well the packets are being transmitted based on how reliably we can shuttle packets back and forth between various nodes on the data link. Like most network protocols, AppleTalk has an echo (or “ping”) utility that lets us test a point-to-point connection by sending echo packets to a node. When a node receives one of these echo packets, it will simply return (echo) the packet back to the sender. The sender records whether the echo packet was returned or not and how long it took to receive the echo, then sends the next echo packet. I’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 looking at how effective the data link is at carrying packets under a simulated network condition, we will look 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 there are some anomalies that can be present on a link that are not always revealed by an echo test. For example, you may have a node that is intermittently transmitting nonsensical signals and wasting network bandwidth. Echo testing is a spot check, and you do not normally test all of the devices on a link. In the case of the jabbering device, unless you picked it as the echo recipient for one of your tests, you would probably not notice that this device was faulty.

Monitoring Data Link Errors

There are several network management tools that 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 tries to read every signal it can find and 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, it is necessary to qualify the results before they can be interpreted properly. There are many variables that 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 to watch the signals. The main weakness of the approach is that the Mac is barely capable of this rather intensive processing task and may show errors that do not really exist. Also, there are some types of Data Link errors that are just not sensible by a computer’s network adapter (NIC) or because they occur during a time when the NIC is not watching for signals.

The fact that it is difficult for a Macintosh to simultaneously sense, analyze and display errors is not all that surprising. . When you design instrumentation for any test, one rule of thumb is that you’d like to have instruments that are at least 10 times as sensitive (or precise) as the phenomena that you’re measuring. In this test, the Mac’s signaling hardware is exactly as precise as the signal that it’s measuring. In addition, the Mac will probably have many other tasks to perform while it’s reading the signals, such as updating display windows as the test progresses. These other tasks further reduce the Mac’s ability to accurately reflect what’s really happening on the link. As a result, Mac-based tools may report erroneous results—Data Link errors that are not really on the link but were instead manufactured because of the Mac’s inability to keep up with all of its normal responsibilities while continuously monitoring for Data Link errors.

Besides just a simple shortage of processing power and measurement precision, there are other factors that may cause erroneous results. One such factor is that because the different tools that perform this test work differently from each other, the tools may report different results when monitoring the same network. Some tools that report these Data Link values are, in my opinion, virtually worthless for this task. The most notable example of this is Farallon’s TrafficWatch II (v 1.0). I suspect that the reason TrafficWatch II reports such erroneous data is the application is trying to accomplish more than the Mac’s processing power allows—processing too many kinds of statistics and updating too many windows. Despite any other merits this tool might have, TrafficWatch II should not be relied upon for accurate reporting of Data Link Errors.

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 (I prefer version 3.1). Unfortunately, AppleTalk Peek is not for sale (and also not supported) and is available only on certain CD-ROM’s sent to developers and consultants, among others. 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, there is great disagreement among network experts and manufacturers over what is an acceptable level of errors. All networks, no matter how well built, will experience some errant signals and damaged packets occasionally. The difficult part is to determine what rate of errors (# of errors/# of total packets) indicates a network problem. What I’ll try to do in the next sections is to state 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. Data Link Layer error statistics should be viewed as a pointer for the direction of your investigation, not as proof of a problem.

TROUBLESHOOTING MACINTOSH NETWORKS

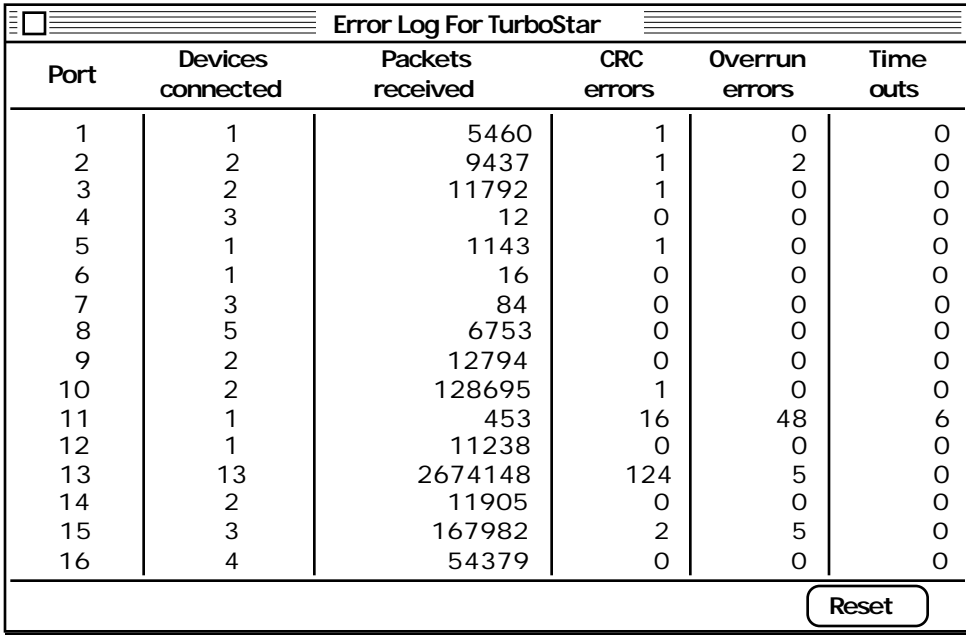
These qualifications and disclaimers being noted, let's look at some of the ways in which Data Link Layer error results can be used for diagnosis of network problems.

Data Link Layer Error Results in LocalTalk

In LocalTalk, there is only one kind of Data Link error that is meaningful for troubleshooting—the CRC error (Cyclic Redundancy Check). A high rate of CRC errors imply that the packet that was received 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 CRC that it calculates 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, there is a high degree of certainty that the data received is correct. In fact, there is probability of only 1 in 2^{16} (65,536) that two packets with different data could have the same CRC. If the CRC's don't exactly match, then a CRC error is recorded. Normally, the Data Link layer would just reject packets with CRC errors. Later, when one of the higher layers notice that some of the data that it is expecting is missing, it will ask its partner node to send the missing data again. The Data Link Layer in LocalTalk, as in most network systems, is not capable of spontaneously requesting re-transmissions.

Of the more popular LocalTalk hubs, the most useful measurements are made by the TurboStar (formerly made by NuvoTech, now made by Focus). In the TurboStar, the number of CRC's is tracked 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. LocalTalk errors other than CRC errors—overruns, underruns, length errors, etc.—may result from causes other than an unhealthy network, and are not very useful for diagnosis.



Port	Devices connected	Packets received	CRC errors	Overrun errors	Time outs
1	1	5460	1	0	0
2	2	9437	1	2	0
3	2	11792	1	0	0
4	3	12	0	0	0
5	1	1143	1	0	0
6	1	16	0	0	0
7	3	84	0	0	0
8	5	6753	0	0	0
9	2	12794	0	0	0
10	2	128695	1	0	0
11	1	453	16	48	6
12	1	11238	0	0	0
13	13	2674148	124	5	0
14	2	11905	0	0	0
15	3	167982	2	5	0
16	4	54379	0	0	0

Figure 5-5 TurboStar Data Link Statistics

The Tribe LocalSwitch and the Farallon StarController also track the number of errors per port, but they do not distinguish between the different types of LocalTalk errors. If a high number of errors is seen on a particular port, it 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 as described below or with the echo testing methods described at the end of this section.

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:

$$\frac{\text{The number of CRC errors}}{\text{The total number of packets}} = \frac{5}{10,000}$$

If you see a high rate of CRC errors on a particular port, you should 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 5 "Lost" packets / 1,000 packets received. On Farallon's StarController, there is no useful rule of thumb, but watch for abnormally high values on a particular port.

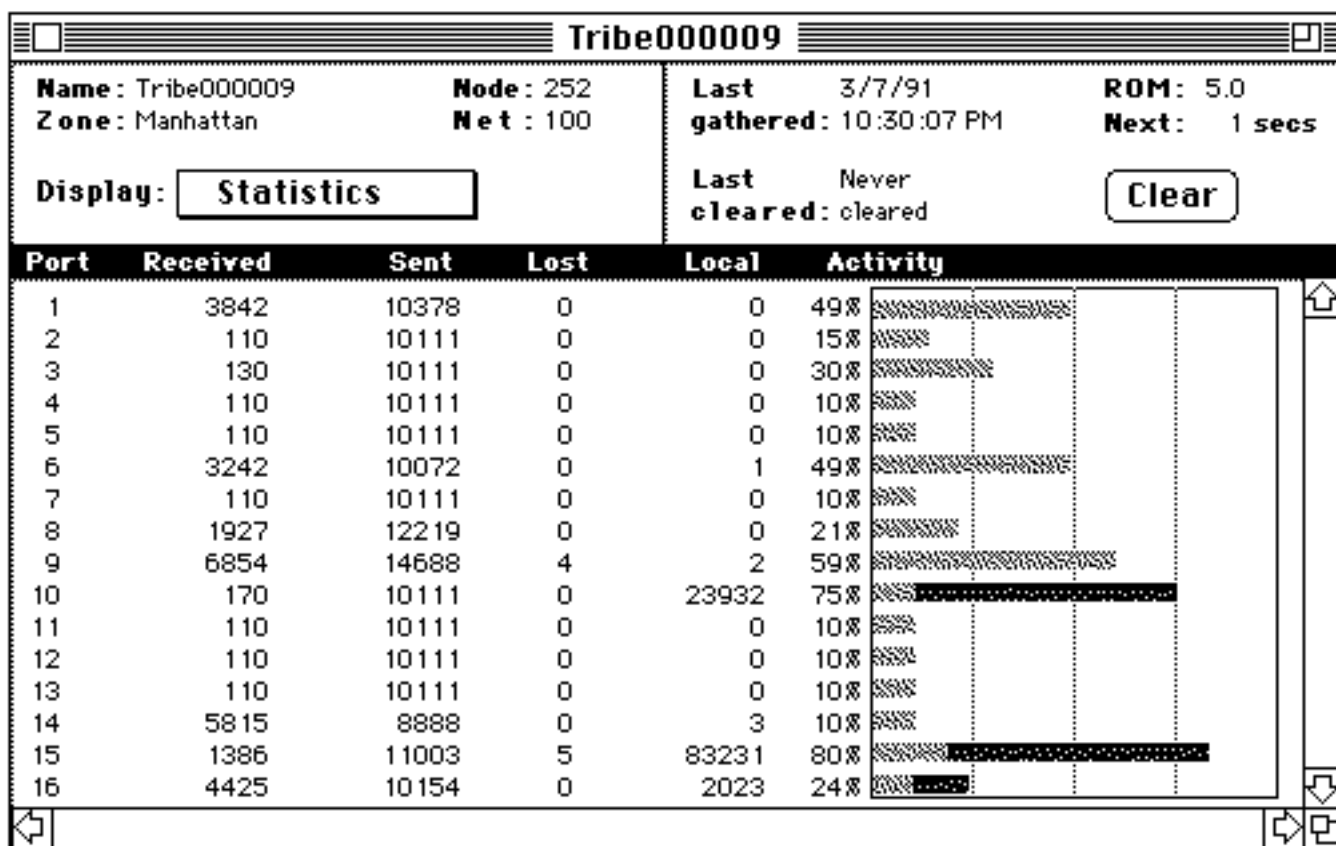


Figure 5-6. The Tribe LocalSwitch's port statistics tell you how many packets were lost due to errors on the line. If you see more than 5 "lost" packets per 1000 packets received, you should investigate the wiring on that port. Port 15 is running very close to this limit.

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 have captured several thousand packets. If you are in a hurry, perform an action that will create a lot of traffic, such as 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 of the Macs that you have running this test are set to capture these packets.

TROUBLESHOOTING MACINTOSH NETWORKS

Pkts in Q Sampling	Pkts Rcvd 9675	Overruns: 10 CRC errors: 1 Time Outs: 3	Rcv Status Receiving RTMP packets
------------------------------	--------------------------	---	---

Figure 5-7. 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 CRC's shown are sample data from 10,000 packets.

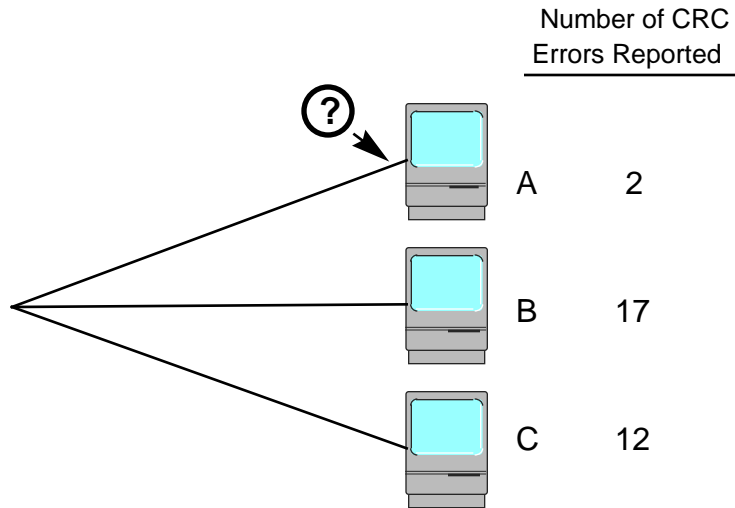


Figure 5-8. In this passive star topology, you would want to check whether Node A was properly terminated. Reflections from this node may be causing signal problems for Nodes B and C.

Improper termination or unusual wiring conditions can cause signal reflections. As in sonic echoes, the echo becomes more noticeable as you move away from the point of reflection. If you are next to the reflection point, the time delay between the incident signal and the reflected signal is so small that you may not even notice the reflection. In the passive star topology in Figure 5-9, a missing terminator at Node A may be causing the signaling problem experienced by Nodes B and C. The same reasoning would apply to the bus topology shown in Figure 5-10—reflections are more likely to have an adverse affect on nodes that are farther away from the reflection.

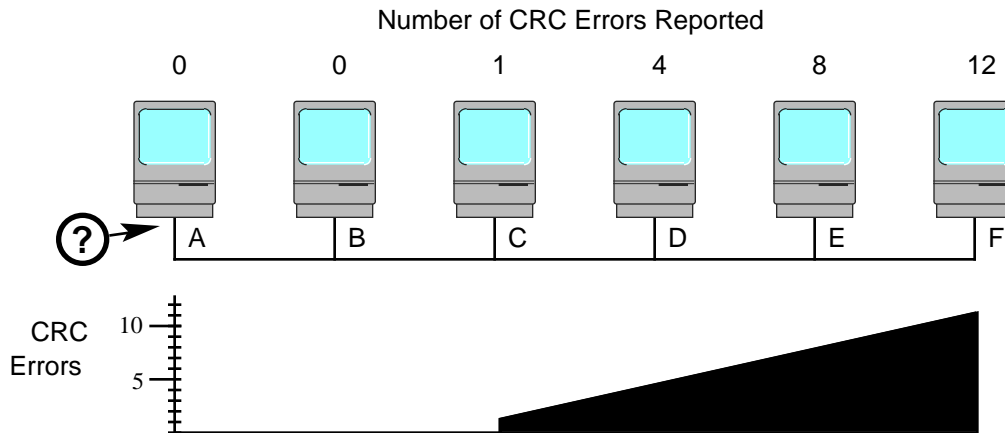


Figure 5-9. In this bus topology, you might suspect a missing terminator at the end of the bus near Node A, which could be causing the signal problems that grow worse as you approach the other end of the bus.

Sometimes the results are not so clear-cut. In Figure 5-10, also a bus topology, only one of the nodes, Node D, is showing 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* CRC's at every node.

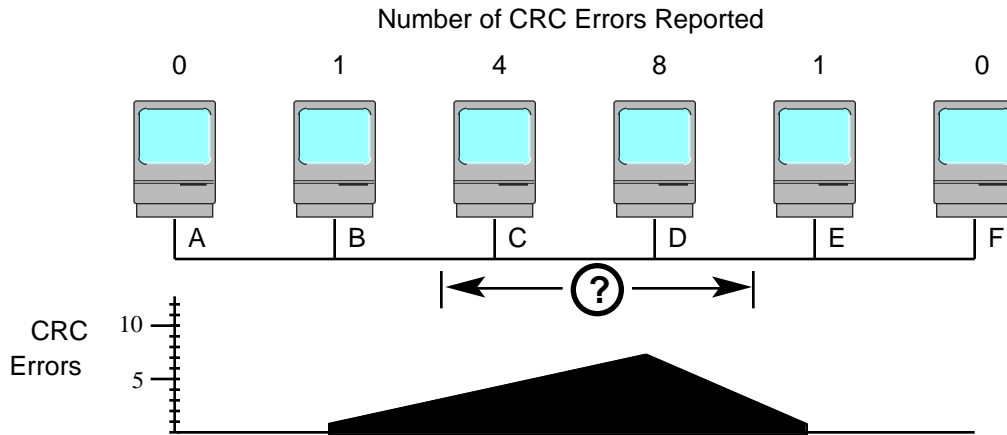


Figure 5-10. In this LocalTalk bus, you might suspect some bad wiring conditions near Node D, which reported a high level of CRC errors.

Measuring Data Link Layer Errors in Ethernet

When measuring the number of Data Link errors on Ethernet links, the meaning of the errors is more clear-cut and the number of errors reported is less subject to the kinds of variable we have discussed, but that does not necessarily make the interpretation of the test data any easier. Some of the errors clearly indicate a problem; others can be tolerated if they occur infrequently, like the level of CRC errors tolerated in LocalTalk.

On a coaxial Ethernet, such as a 10BASE5 or 10BASE2 Ethernet, an error that occurs can be seen by every node on the segment. Protocol analyzers monitoring the segment can monitor and record many of the errors listed below, and some bridges and repeaters also have the ability to track errors as well.

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

Some 10BASET hubs, such as the low-cost hubs 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 LED's on the body of the repeater. Others provide just enough management software so that they 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 10BASET), but they are 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 thing to try if you having trouble getting packets between the 10BASET 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 are the error types most commonly reported with an explanation of their meaning and their possible causes.

Link Integrity:

Most 10BASET hubs and external transceivers, as well as many 10BASET cards have Link Integrity indicators, typically an LED. Both 10BASET hubs and transceivers send a Link Integrity Pulse to each other every 5 seconds. The light on the transceiver indicates that the hub's Link Integrity Pulse is being received and vice versa. If basic electrical continuity exists (confirmed by some other test, such as an ohmmeter or tone test), but the light is not lit, something is preventing the transceiver and the hub from communicating. Check to make sure that the card's jumpers are properly set, that 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, then 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 above 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 cannot correct for it, and some hubs cannot even detect it. If your LED's 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 cannot correct for it. If the device is sending packets, but seems to be ignoring packets that it receives, use that device or the wall outlet it is using to run a Mac-based protocol analyzer. If the receive lines are reversed, all of the "packets" that are displayed should have nonsensical data in them.

Jabbers

Jabbers occur when a device's network hardware malfunctions, and the device sends a continuous stream of bits. Ethernet repeaters, including 10BASET hubs, are required to detect a continuous transmission and shut off the port that is jabbering.

Auto-Partition

10BASET hubs have the ability to automatically shut off a port if it senses a serious error condition on a port. One such condition is when a port experiences more than 30 consecutive transmit collisions or when a collision occurs over an extraordinary period of time. Some hubs have the ability to report these auto-partitions through the use of their management software or indicate the auto-partition with an LED. This may require a Setup Mode check of all of the components as indicated in Chapter 5. 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 a software check, the Network Control Panel settings, the EtherTalk drivers, the AppleTalk version number and the system software configuration.

Transmit Collisions

These happen when a 10BASET 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 only be expressed in terms of probability, and the probability would be related to the traffic level. When measured over an hour-long period, an Ethernet utilization of more than 15% is considered a high traffic density. When measured over a period of 1 second, utilization of more than 45% is considered high.

While there are no agreed upon limits for collision rates, you may want to investigate when the collision rate seems greater than normal. If you cannot correlate it to a high traffic level, you may 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, i.e. short events.

Receive Collisions

These mean that when the hub was not transmitting, it received a jam signal from a transceiver that was experiencing a collision. On a 10BASET port, since there may only one device per port, this should never happen; it takes two devices transmitting simultaneously on the same wire to have a collision. If you do see Receive Collisions on a 10BASET port, it could mean that that port is connected to more than one device. For example, it could be a connection to another 10BASET hub that is sending a jam signal or to a 10BaseT to thinnet repeater where the thinnet contains several devices. If the port showing receive collisions is a pure 10BASET port with only one device, it means that 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 so that late collisions should not happen. The Ethernet parameters that affect the likelihood of this occurrence 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. It could also suggest defective hardware.

Runts

(Also called Fragments) Runts are usually the result of collisions, and the concepts stated for Transmit Collisions above also apply to runts. If there are significantly more runts than collisions, then apply the concepts listed below for Short Events.

Short Events

(Also called Pygmy Packets) These are signals that are shorter than 74 bits in duration (7.4 microseconds). These might not be packets at all, but electrical noise on the wire that may be coming 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 1 Short Event per 1000 packets, you may want to investigate for electrical noise.

TROUBLESHOOTING MACINTOSH NETWORKS

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 is typically caused by defective hardware.

CRC Errors

CRC Errors are very similar in nature 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, more than that and the Physical Layer should be investigated. In coaxial Ethernets, check for violations of Ethernet's strict construction rules (number of nodes/segment, maximum segment distance, etc.) as well as kinked cable. In 10BASET, check for bad wiring or too many connection points between the hub and the transceiver.

Alignment Errors

An alignment error occurs 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 eight. Again, check for faulty wiring if there is more than 1 Alignment Error per 1000 packets.

Phase Lock Errors

This occurs when the receiving node cannot 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 is not really an Ethernet packet at all (line noise). Check for these conditions if you're getting more than 1 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 use the Echo Test later in other ways, but in Data Link Layer troubleshooting, we're just concerned 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, there will be one returned from that node. Echo testing will give you one of 3 answers to the question, "Does the network link work?" The possible answers are, "It works all of the time", "It works some of the time", or "It doesn't work at all".

If the answer that your echo test provides is, "It works all the time", you can move your investigation up to the Network Layer. The drawback to the Echo Test, however, is that when you get one of the other two answers, the Echo Test does not tell you much about the nature of what the problem might be. For that you'll have to rely on the methods outlined above concerning Data Link errors or the troubleshooting methods of the Physical Layer described in the previous section.

The echo test is built into several software tools, but I find that the most convenient tool to use for this particular kind of troubleshooting test is Apple's Inter•Poll (despite the fact that it has not 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 4 variables—how many packets are sent, the delay between the packets, how long to wait for the returning echo packet (timeout), and whether to send short or long echo packets. I use the same variables all the time for consistency: 100 long echo packets with no delay and a 1 second timeout. This places about a 30% utilization on a LocalTalk link, depending on how fast the other node can turn the packet around and send it back. A few kinds of devices do not 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 that I 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 I send out 100 packets, the number of packets that are 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 my 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 range between these 3 times. Generally, that indicates that the device is very busy.

On LocalTalk, you should get 100% of the echo packets returned every time. If you get 99/100, 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 truly is being overtaxed by its other processing demands. If it is, then use a protocol analyzer to verify that the echo packets sent were, in fact, OK. If you can verify that a good echo packet traveled from the test node to the busy recipient node, but saw no echo packet returned, you may be able to forgive a missing packet or two. If you see the opposite—an acceptable echo packet being returned but not being counted by your testing device, then you may want to 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, you should get 100% of your echo packets returned every time. If not, investigate your wiring. There is a good chance that there is something wrong. Over Ethernet or Remote Access links, use 98% as the good/investigate threshold.

Interpreting the Echo Round Trip Time

When an echo packet takes a long time to be returned, it is most often due to the fact that the other machine had other tasks to accomplish before it could “get around” to its obligation to return the echo packet. Some applications, including Claris’ FileMaker, may place such 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 that the average round trip time will 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” as described above. Then, the maximum time can stretch out to as long as 0.75 seconds. Even then, only 1 or 2 packets out of 100 should take such a long time.

If you are echo testing a link using a very busy node as your recipient, you may want to increase the timeout value if you think that less than perfect test results are being caused when returned packets fail the test because they exceed the 1 second timeout value I recommended earlier.

If you are using the Tribe LocalSwitch in your network, it will cause a slight delay as the packet is switched from one port to another. With no other traffic on the network besides your 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 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 needed switching was busy with a packet of its own. In this case, there might be an additional delay in the time because the LocalSwitch would stand by, holding the packet in memory until the port was clear. This could add approximately 0.02 seconds to the transmission, which could be in either direction—the outgoing or the returning echo packet—or in both directions. This time delay, which will probably be a rare occurrence, is within the measuring ability of Inter•Poll and will affect the test results.

TROUBLESHOOTING MACINTOSH NETWORKS

A very busy node can also affect the test in one other way—its Data Link layer may reject some of the packets because it “perceives” that a packet is damaged due to the fact that one of the two nodes involved in the test did not have enough processing time to adequately attend to the reading of the packet when it arrived.

You will certainly want to increase the timeout value if you are testing a Remote Access link or across a low-speed link (less than LocalTalk speed) between distant sites. A value of 3 seconds should be sufficient if there are no routers between your 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 and all of the electronic components involved 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 in comparison to most other Data Link protocols. LocalTalk uses a collision avoidance technique to arbitrate the use of the link. Part of LocalTalk’s collision avoidance mechanism is that each data packet on LocalTalk is preceded by two LLAP Control packets that establish whether it will be profitable for the sender to 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 to receive, in return, a Clear to Send (CTS) packet back from the receiver 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 is ready to receive the packet. Only then does the sender place the data packet on the wire.

This type of exchange has an analogy in 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 being able to talk to Mike.

The 200 microsecond limit is a guideline that is often flagrantly exceeded by some LocalTalk devices. Although there is some tolerance for variations from this maximum time, some LocalTalk devices take a very long time to return the CTS and may exceed the sender’s patience for waiting. The sender might “give up” and try again some time later by sending a new RTS, waiting again, having its patience exceeded, etc. 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 are 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. Notorious examples of devices that exhibit this phenomena are Dayna’s EtherPrint and EtherPrint Plus, the Hayes InterBridge and Apple’s LaserWriters II_f and II_g (when used on LocalTalk). 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 protocol analyzers can see that a CTS was in fact returned and no data packet followed it, no protocol analyzer can tell you whether there was a 200 μsecond or μ700 microsecond gap between the RTS and CTS. That type of information is only available by measuring the time between packets with an oscilloscope.

If you do not have a scope, then create a 3 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 if there was a CTS returned or not.

Network Layer Troubleshooting

At the Network Layer, the question is, “Are the routers working properly?” The tests that we’ll make at the Network Layer have two main issues, 1) “Are all of the routers that we expect to see in our internet functioning?” and 2) “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 4 or 5 routers, they are also fairly easy to answer. But in a large internet with a fifty or more routers, neither of these questions is particularly simple to answer directly or conclusively. There are tests methods, however, which can gather the circumstantial evidence surrounding these questions that can help us make a nearly sure determination.

The first question concerning the existence of the routers will be answered by testing to see whether all of the networks and zones that are expected in the internet actually do exist. This is done by looking for named services in those networks and zones. The second question concerning router configuration information will be answered 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 check for zones is to open your Chooser to see if you have a zone list at all. What this verifies is that there is at least one router on your network and that you can communicate with this router. In a small internet, it’s very easy to count the small number of zones in this list and check for misspelled or duplicate zone names. You should be aware, however, that this is only one router’s “opinion” of what zones are available in your internet. Other routers could have a different opinion, so you may want to open the Chooser in other networks to check the other routers.

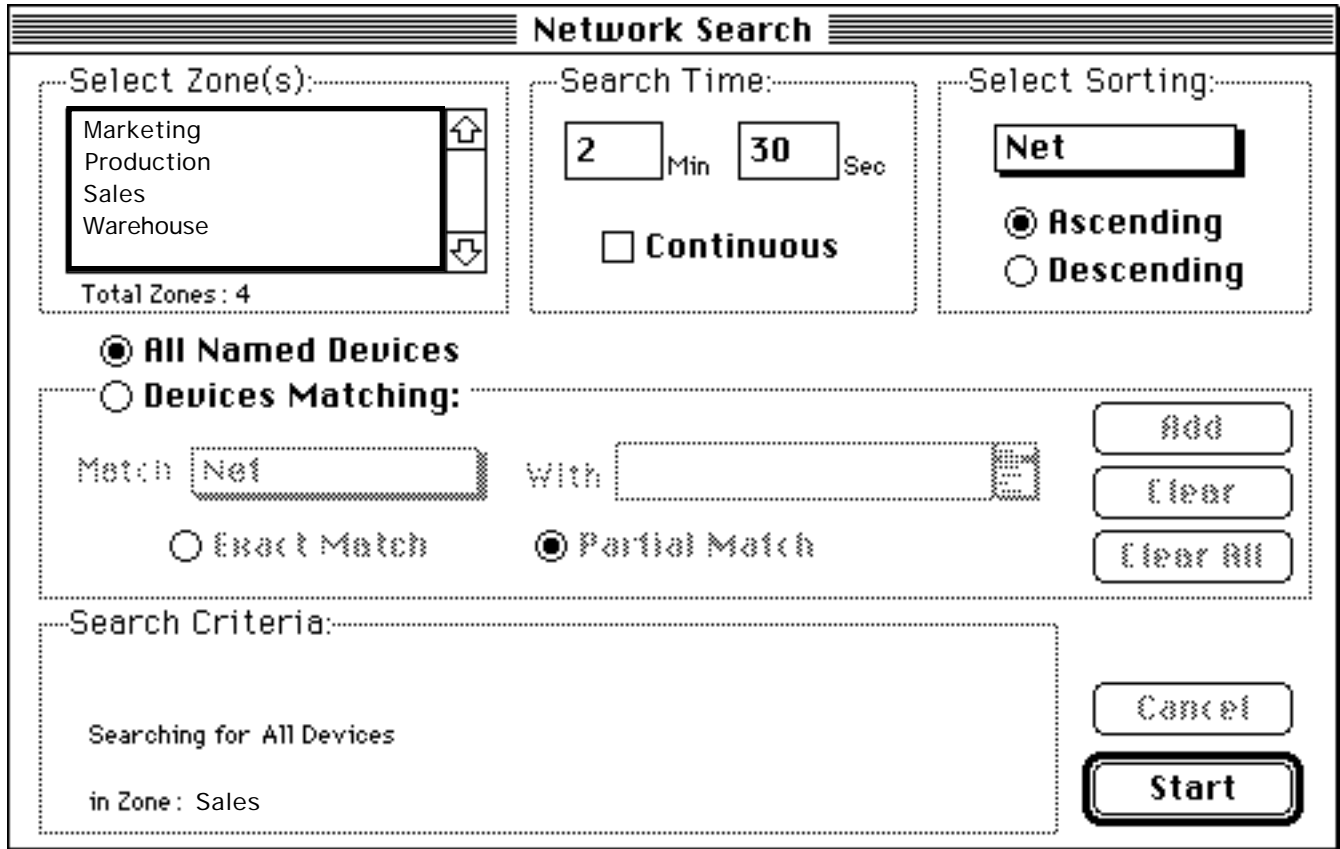


Figure 5-11. Inter•Poll's Network Search window tells you how many zones are in your internet and lists their names.

If you have long list of zones, the Chooser is not a very effective tool. It does not tell you how many zones exist, and it is difficult to count a long list of zones in the Chooser. A better method to verify your zones is to run Apple's Inter•Poll. When you first enter the program, you will see the Network Search window where you select which zones you want to monitor. In this window, as you can see in the Figure 5-11, Inter•Poll also tells you how many zones there are in your internet and lists their names as well. The zone list is printable as well to allow 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

This is a little harder than verifying the existence of the 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 do this. After getting your initial router list, double-click on one of the routers to bring up the Query window. In the Query window, you can ask the router for its list of routes (networks). RouterCheck will list all of the networks that the router has in its Routing Table and also tell you how many networks are in 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 information that may be helpful.

Once you have verified the existence of the zones and networks in your internet, you're ready to move on and investigate your routers' configuration information. This is handled in great detail in Section 8.3 "NAME".

Transport Layer Troubleshooting

The question to answer at the Transport Layer is, "Are the messages getting through the internet?" Just as the Echo Test was able to test the Data Link Layer's function of sending packets from one node to another on the same link, echo testing can also be used to 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), which is 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. A typical PET scenario is shown in Figure 5-12 below.

Then, 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.

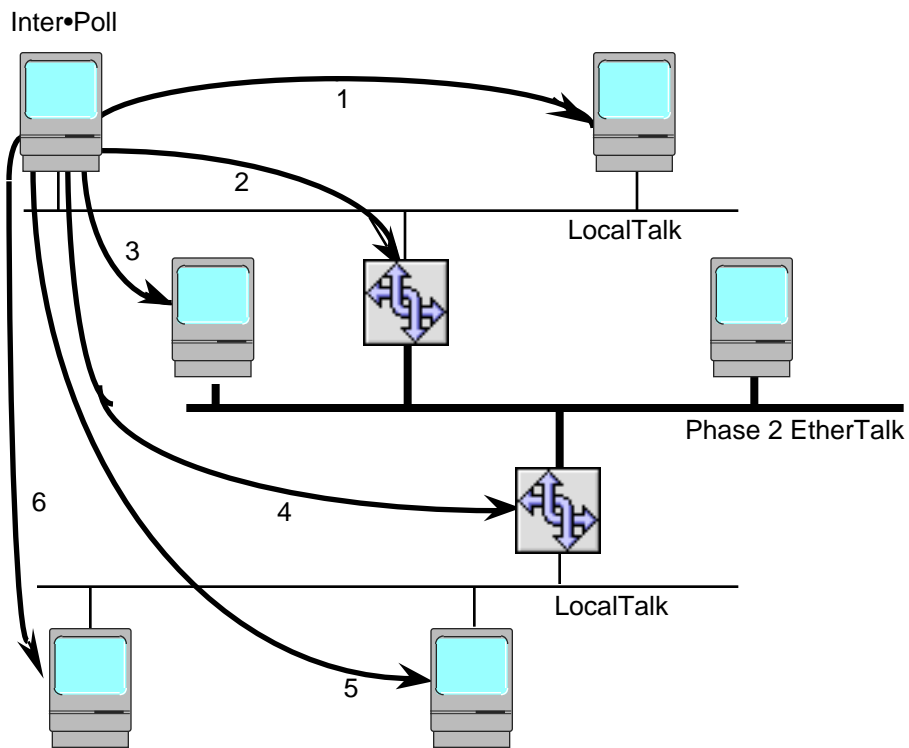


Figure 5-12. The Progressive Echo Test is a series of echo tests to test the reliability and speed of packets sent through the internet.

Using the Progressive Echo Test

Let's say that you had a user who was complaining about the poor reliability of his connection to an AppleShare server located in another network. He might report that the connection was dropping occasionally or seemed very slow at times. Dropped connections are usually caused by unreliable packet delivery. AppleShare servers and clients do not always have data to send to each other, and in the absence

TROUBLESHOOTING MACINTOSH NETWORKS

of data, each side will periodically check to see that the other is still online and reachable. They send packets to each other to verify that they are 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 were not getting through the network, at least for a time long enough to cause the connection to drop.

Slow performance indicates that the communication between the two devices is slow. 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. There are many possibilities, including:

- Perhaps the link that the user is on is unreliable (or very busy).
- Perhaps the link that the server is on is unreliable (or very busy).
- Perhaps one of the intermediate network links along the path is not reliable (or very busy).
- Perhaps one of the routers along the path is not reliable (or slow).
- Perhaps the internet routers have unreliable routing information.
- Perhaps 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, and your testing should include all of the pertinent intermediate points between the user and the server. By looking at how the reliability of delivery and time necessary for the echo packets to make their round trip varies among the tests, you will usually gain 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 and remember to clear the results of each test before beginning the next test. As I indicated elsewhere, I always send 100 long echo packets with no delay between packets and a timeout of 1 second. See Section 6.2 in Data Link Troubleshooting for more information about the echo test.

Sometimes, it may also be useful 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. The thing to notice here is whether there was any variation in the hop count, i.e., did some of the return packets show 3 hops and some of the return packets show 2 hops?

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 is not a very interesting aspect of the test results. In most cases, the hop count will be constant for every packet in the test. In the test in Figure 5-12, 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 were to get anything other than this, you would need to do some investigation, because the variation may show something wrong with the configuration of your internet routers. Even with router problems, however, this is a very rare event. Nonetheless, it needs to be covered here.

A rule can be made 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 using running a Phase 2 network driver.

If Condition 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. Explaining this condition in terms of Figure 5-12, this would be the case if the returning echo packets in Test 5 showed both 2 hops and 3 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), which forwards them to Router 1(third hop). In other words, Router 2 isn't very 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 make a thorough check of your routers as described in Section 8.3.

If Condition B is true and the hop count shows variation, this may not indicate a serious error, but it is still worth taking a look at. Test 3 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 makes the determination of 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 that they received any packet from.

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 know that the Best Router algorithm is not working correctly. At this point you need to figure out why. The best way to do this 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 are being sent to pick up the extra hop. The router that accounts for the extra hop should have its configuration checked carefully to see if it is in line with the rest of the internet’s routers.

Interpreting the PET Time and Reliability Results

When the test results have been compiled, I 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 5-13 below, the round trip time graph changes very dramatically between Test 5 and Test 6, the time of the round trip nearly doubling. The reliability data changes between Test 2 and Test 3, changing from perfect reliability to partial reliability.

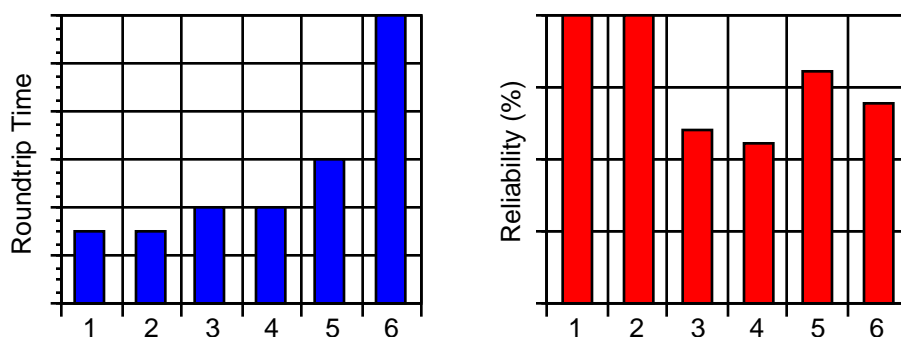


Figure 5-13. Graphing the data from a PET helps you see where the data changes suddenly.

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 involved 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. Well, before you jump to that conclusion, it would be a good idea to repeat Test 5 and Test 6 to see if you get the same results. There is always the chance with the Echo Test that there will be a temporary condition on the network greatly affect the test, so every time you think you have the “Aha!” data, it’s a good idea to repeat the test just to confirm your conclusions.

TROUBLESHOOTING MACINTOSH NETWORKS

If your second batch of tests confirms the data in the first, your conclusion could be 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, then 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 it 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 the figure above, the dramatic change in reliability occurred between Test 2 and Test 3. There are two possible causes for this; either the router is unreliable or the Ethernet is unreliable. There are many ways to check this. First, you may want to look at the statistics in the router's log to see if it dropped any packets due to either Data Link errors or because its buffers were full. Different routers have different abilities to gather and display statistics, of course, but this kind of information can sometimes tell you whether the router is the culprit. You will also want to check whatever Ethernet statistics you have access to, whether in a hub or by 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% to 10% errors) to indicate Ethernet reliability as the culprit, but remember that the time over which the Ethernet statistics are gathered and the time over which the Echo Test is done 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 are noticing may be transitory.

If you don't have access to statistics like these, you can stick with the PET method and perform a seventh test as shown below in Figure 5-14. Here, the Echo Test is performed between the nodes on the Ethernet to get a reading on the reliability of the Ethernet alone.

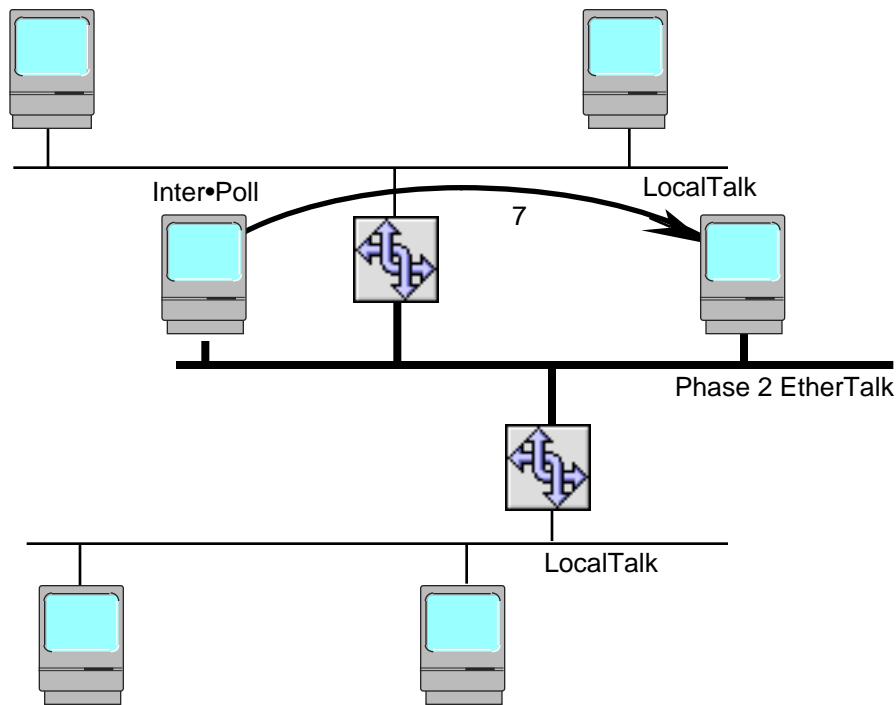


Figure 5-14. Test 7, when added to the data from previous testing, will show whether Router 1 was unreliable or the Ethernet is unreliable.

If Test 7 shows 99% 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 90's, then you can conclude that the router is OK 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 that you select for monitoring, you can specify how frequently you want to check its reachability and what actions you would like the program to take in the event that the service cannot be found. 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.

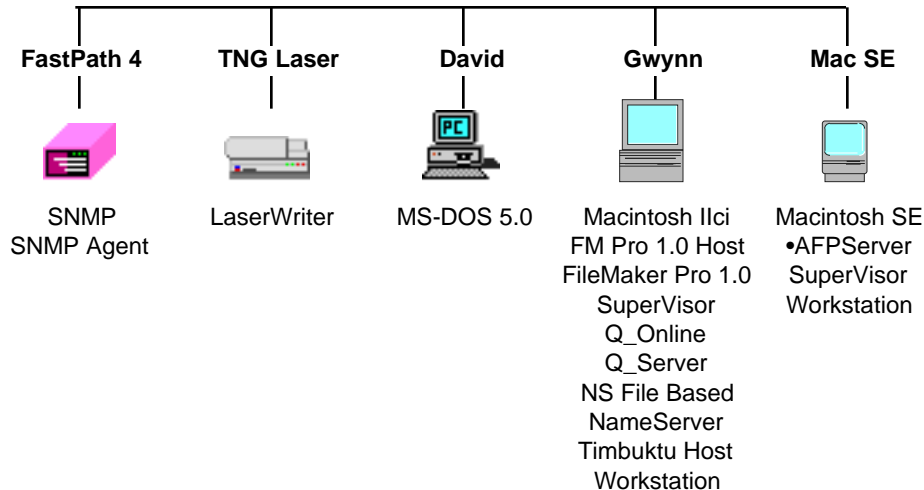


Figure 5-15. NetWatchMan, made by the AG Group, monitors the critical services in your internet and notifies you when they are reachable or not. The darkness of the Mac SE indicates that it is not currently reachable.

Both of these programs make their reachability determinations from a simple yes/no viewpoint and do not provide statistical data other than when the service was first noticed to be missing and when it was first noticed to be back on line. While this will usually not help you to diagnose the reasons that the node was not reachable, the usefulness of these programs is that they provide an “early warning system” for network problems. In the best cases, you may be able to notice problems before your users notice them. 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 for the relationship between two computing devices. This includes establishing or allocating those resources at login, to provide login and logout services, to manage the mechanics of passwords and privileges and to provide any session maintenance required such as the verification of the connection between the client and server.

The question that we’ll ask at the Session Layer is, “Does the server have the correct configuration?” Session Layer problems usually have their roots in an incompatibility between the server and client where one of them expects the other to have some resource or characteristic that is not actually present.

Symptoms of Session Layer problems include:

- some users can log into and use the service, but others cannot
- the server does not allow services that it should
- the server crashes or hangs unpredictably

TROUBLESHOOTING MACINTOSH NETWORKS

- the server is slow at times
- the server will drop 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 with each other
- the server software may be in conflict with the client's application software
- the particular user who is having trouble is not properly authorized to use the server

Session Layer troubleshooting proceeds by running through a checklist of questions that verify various aspects of Session Layer functions. Some of these questions may have been considered during previous attempts to discover the cause of the problem (in Hunch Mode, perhaps), but are asked now in a methodical way. The checklist proceeds as follows:

Check that:

1. The Service is Advertised.
2. The Service Can Communicate.
3. The Server's Version is Correct.
4. The Server's System Configuration is OK.
5. The Server Allows the Access That You Need.
6. Previous Users Have Damaged the Server.

Special Considerations of the Session Layer

The term "server", as it is used in this section, is used in the general sense to refer 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 Section 2.4, "Network Relationships". Examples of servers are LaserWriters, AppleShare servers, shared modems, SNA Gateways, a Timbuktu Host, multi-user data bases and Remote Access Servers, to name a few. "Client" refers to a device or software process that is attempting to use the server.

Each of the devices in this example list has a group of potential problems peculiar to the server function that it performs. In fact, each of the types of server has its own list of common problems, commonly made mistakes and common misconceptions. In addition, there are several manufacturers and models for each of the server functions in the list, each of which 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.

Mostly, Session Layer troubleshooting involves running through the items in the checklist, asking ourselves various questions and verifying that certain conditions were met and certain questions answered. Occasionally, we'll use techniques borrowed from Process-Oriented Troubleshooting; we might peek inside a packet or two to find out a software version or to find out what methods of user authentication the server offers.

Checklist Item #1-The Service is Advertised

If you have been following the procedures of Layer-Oriented troubleshooting, you should have already established by this time that the server is online, that it is 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 is not trapped by a router's security mechanism.

Still, just verify that it is advertising the correct service name using Inter•Poll or any similar tool. AppleShare servers should be advertising "AFPServer", LaserWriters should be advertising "LaserWriter", etc. It's possible that although the node is on and online and can send and receive echo packets, that it is not 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 5-16. Verify that your desired service is advertised. This CheckNet list is sorted by type and shows that “Mac SE”, the “AFPServer” is advertised properly.

If you do not see the service that you desire, but do see that the node itself is online, then find out why it is not advertising the service. In dedicated AppleShare servers, perhaps the server program is not 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 ExitServer command, such as a user might do if they were going to reset the LaserWriter or change the configuration of its permanent RAM. Perhaps the Timbuktu Host that you wish to connect with has Guest Access turned off.

If you have a non-dedicated server running on someone’s Mac, maybe they have done something to cause the server to stop. For example, if a user running a Microsoft or QuickMail server switches her network connection from LocalTalk to EtherTalk, the server is shut down and will not start up again after the switch is made, nor will it start up again if she switches back to LocalTalk. It will only start up again when she restarts her Mac.

Checklist Item #2-The Service Can Communicate

Although you have 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 the service or can use the service. You might try the service from another workstation or ask other users if they have 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 are not 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.

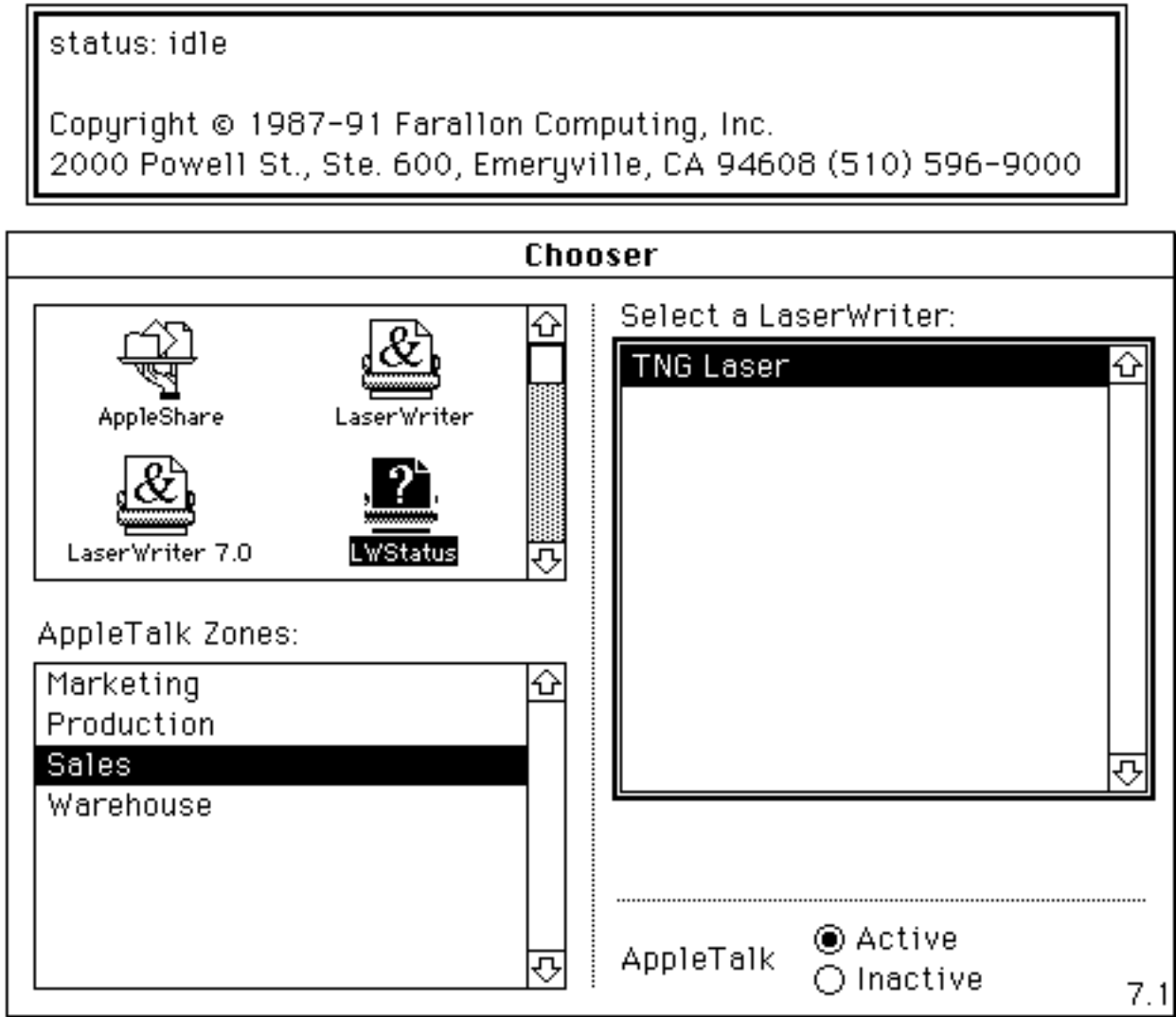


Figure 5-17. Farallon's LWStatus can display the current status of a LaserWriter. This proves that the service is not only available, but can communicate as well.

You can also use a protocol analyzer to verify that the service is communicating. Watch a user as he tries to log in and see whether the device responds at all.

If you are not comfortable with a protocol analyzer, you might try using a traffic analyzer like Farallon's TrafficWatch to see whether other devices are communicating with the node you are interested in. Although you can't be sure that the traffic that you're seeing is coming from the service you desire, 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.

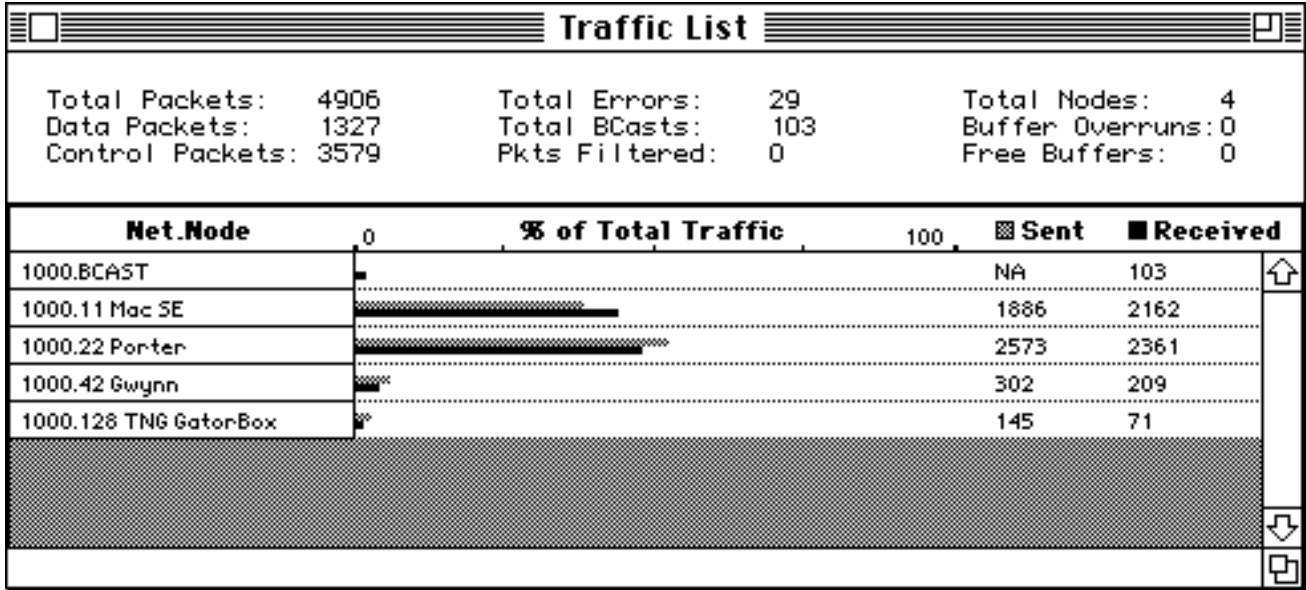


Figure 5-18. TrafficWatch shows that Mac SE is communicating heavily with Porter and somewhat less with Gwynn.

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 identity of the users that are currently enjoying the server’s attention.

Checklist Item #3-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 5-19 below 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 are using on their Start Page.

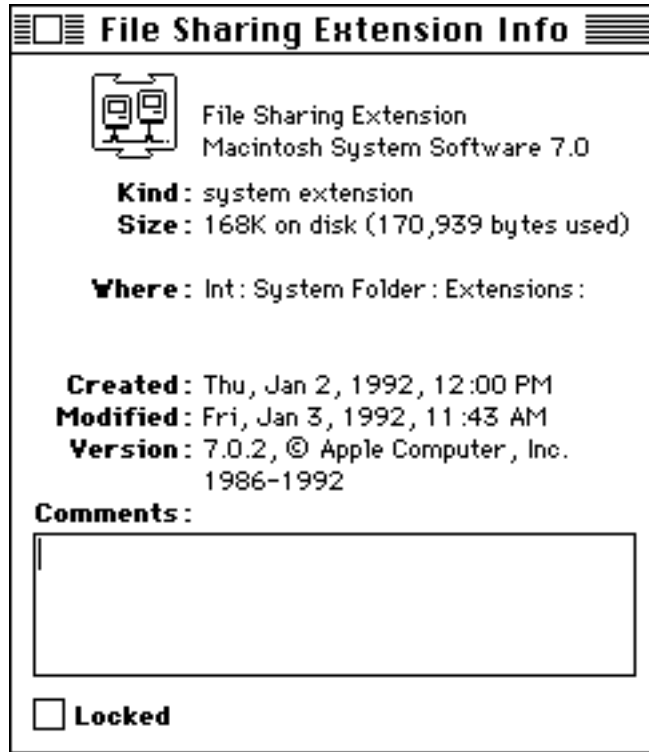


Figure 5-19. Services that run on a workstation may easily reveal which software version they are running.

It may not be apparent from Figure 5-19, however, which version of AppleShare client software is compatible with this server software. For many services, the match must be exact. For Microsoft Mail, for example, Version 3.0 Servers will only communicate with Version 3.0 clients. For other servers there may be a chart of which client software is compatible with which server software. Sometimes you can find this out by trial and error or by a call to the manufacturer of the server software.

Some clients and servers may have a limited capacity to work with dissimilar software versions. If you happen to be using one of these kinds of services, there will usually be a stage during the login process where either the client or the server will ask the other what software version it is using or is capable of interacting with. Often, this will be visible in one of the packets. In Figure 5-20 below, you can see many aspects of the server in a status packet that it sent to the client during login.

ASP Packet - AppleTalk Session Protocol

ASP GetStatusReply	← 0	This packet is the reply to a GetStatus request.
Unused:		
Status:		
-+-5-c-#---Mac S	←	The Name of the Server is "Mac SE".
E		
-Maci		
intosh--AFPVersion	←	This server allows clients with AppleShare versions 1.2, 2.0 and 2.1 to log in.
n 1.1-AFPVersion		
2.0-AFPVersion		
2.1--No User Aut		
hent-Cleartxt pa	←	"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.)
sswr-d-Randnum ex		
change-2-Way Ran		
dnum exchange---		

Figure 5-20. AppleShare servers send a status packet during the login to inform the client of its login options.

Checklist Item #4-The Server's System Configuration is OK

A check should be made of the server device to find out whether it has any incompatibilities on it that might cause problems. On Macintoshes, I find the application Help! to be 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 very 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):~Access PC
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 5-21. Help! lets you know what configuration errors may be present in a Macintosh.

Checklist Item #5-The Server Allows the Access That 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 their 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 of the previous tests, it may not be offering all of its services to all users. Check the server's rights and privileges configuration. Figure 5-22 shows the configuration for Timbuktu.

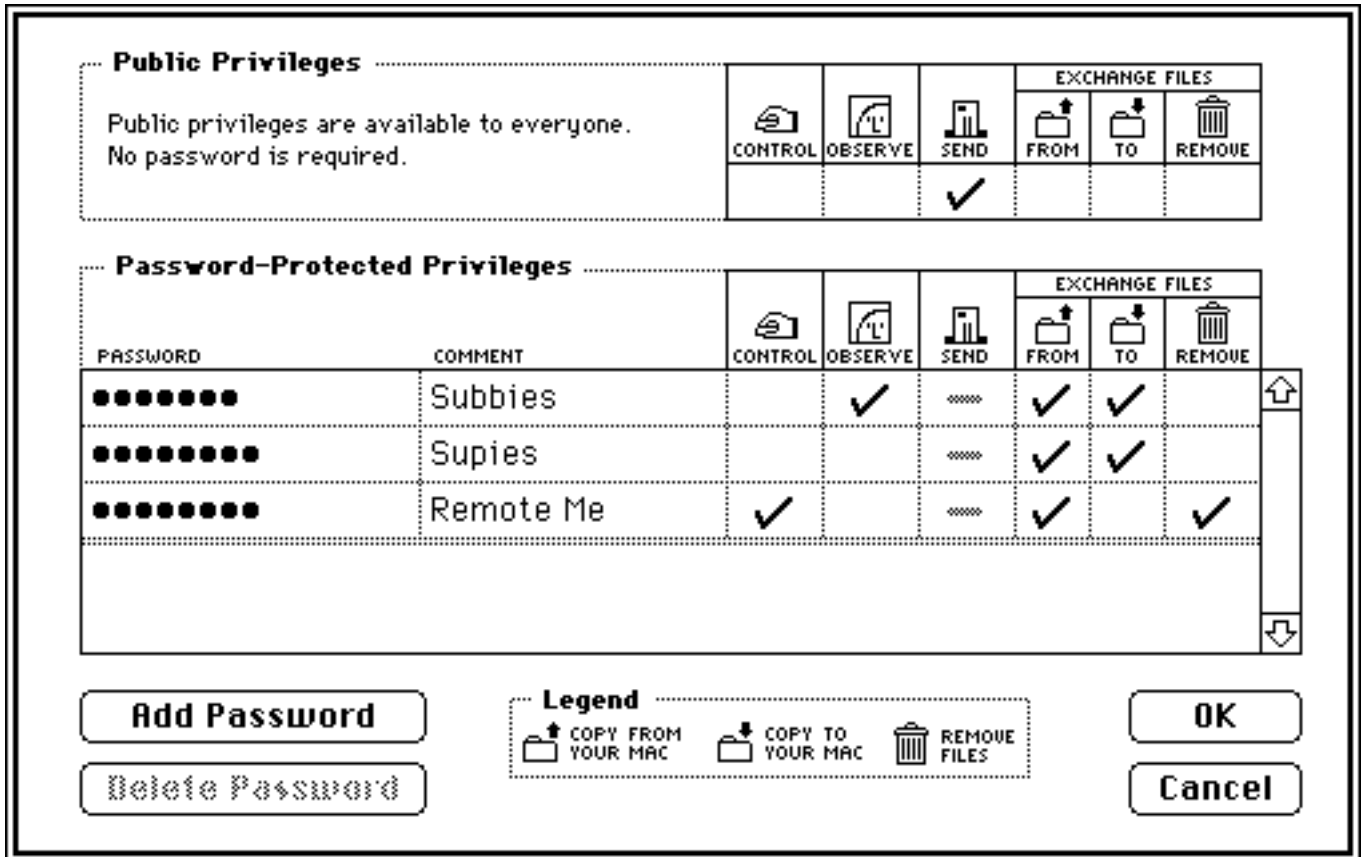


Figure 5-22. Farallon's Timbuktu allows its users to set different access privileges for different classes of guests.

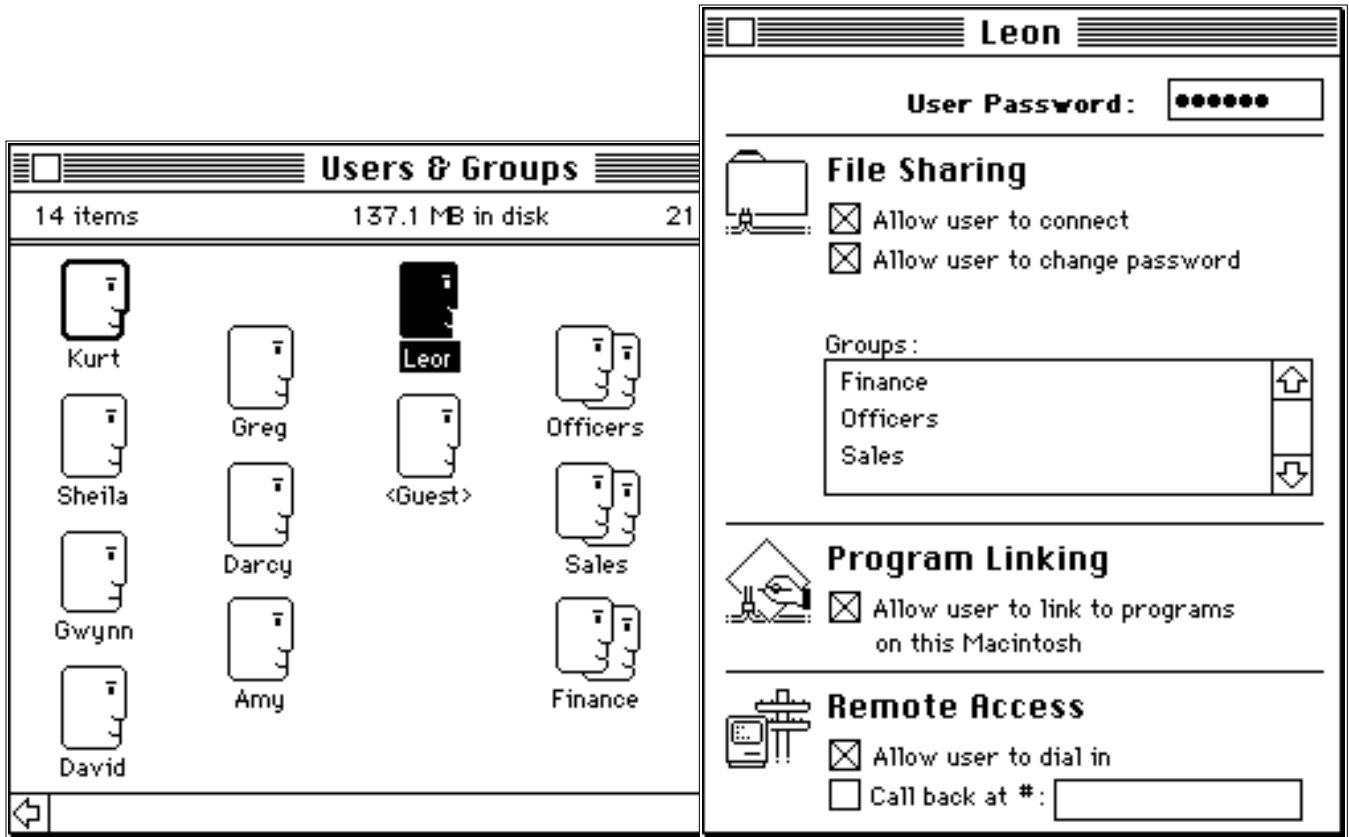


Figure 5-23. System 7 File Sharing lets users define who can use their Macs and for what purpose.

Checklist Item #6-Previous Users May Have Damaged the Server

Some servers may be rendered useless by a user action that the server was not prepared to handle. An illustrative example of this is 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 will typically have an AppleTalk session between itself and a client workstation and a session using a different network protocol between itself and a (non-AppleTalk) host.

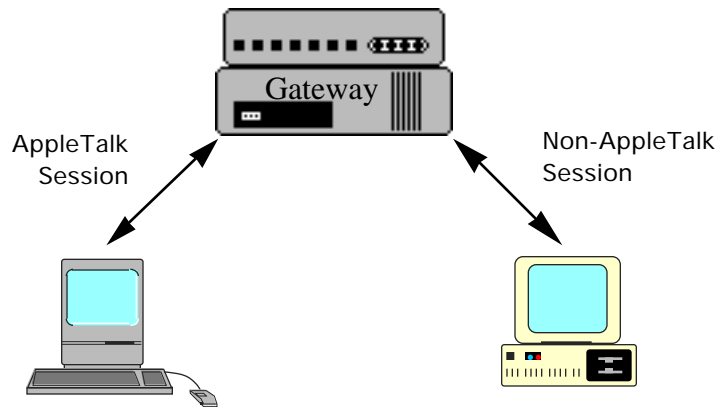


Figure 524. Gateways are particularly susceptible to unusual user actions that may impair their ability to function for subsequent users.

TROUBLESHOOTING MACINTOSH NETWORKS

An example of this type of arrangement might be a gateway to an SNA environment. Communication between a Mac workstation and the SNA gateway that it is using would be carried through the network by AppleTalk protocols. The SNA gateway would also have, on the Macintosh user's behalf, an SNA connection to a mainframe. The gateway would translate the Mac user's actions into mainframe commands and translate the mainframe data into Mac screen commands. A difficult circumstance arises if the Macintosh user suddenly terminates the AppleTalk half of that arrangement without gracefully 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 inappropriate action.

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 up again for a new user. The alternative is to restart the gateway, which also forcefully closes its connection to the mainframe.

Presentation Layer Troubleshooting

The purpose of the Presentation layer is to *present* 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. Since we are troubleshooting the Presentation Layer in this section, we'll assume that we have already established that the layers below have sufficient vitality and reliability; we 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 asked at the Presentation Layer is, "Do the two devices agree on the way in which the data is represented?"

The least complicated communication between two systems generally occurs when the similarity between systems is highest. 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, let's say that you create a document with Microsoft Word on your Mac in which describe your department's monthly activities. A friend in a different department sees it and calls you; he likes your report so much that he wants to use it as a template for his department's activity report. You tell him 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 him. You then give him the necessary access to the folder containing your document. If your friend also uses a Macintosh and Microsoft Word, the transfer is very straightforward. Using his AppleShare Extension, he can mount your folder as a remote volume and open the document, edit it and save the edited file to his own hard disk. As an alternative, he could copy the document from the remote volume to his own hard disk using the Finder and then do what he wants with it.

Operations at the presentation layer

Using the scenario above, let's look at the basic elements of the Presentation Layer necessary for this activity to take place. First of all, both machines must have the proper configuration of software and hardware to allow these actions to take place. 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. The first basic element of the Presentation Layer is the **Proper System Configuration**.

The second basic element of the Presentation Layer is that the two devices have a way of representing files in the language of a **Common File System**. With the devices are properly configured, your folder and its contents can be represented to your friend's Mac as a remote volume with AFP (AppleTalk Filing Protocol) as the file language. Your Mac's native file system, HFS, its data structures and command

language, are translated into their AFP equivalents and sent across across the network, and then are immediately translated back into HFS once the file arrives at your friend’s Mac.

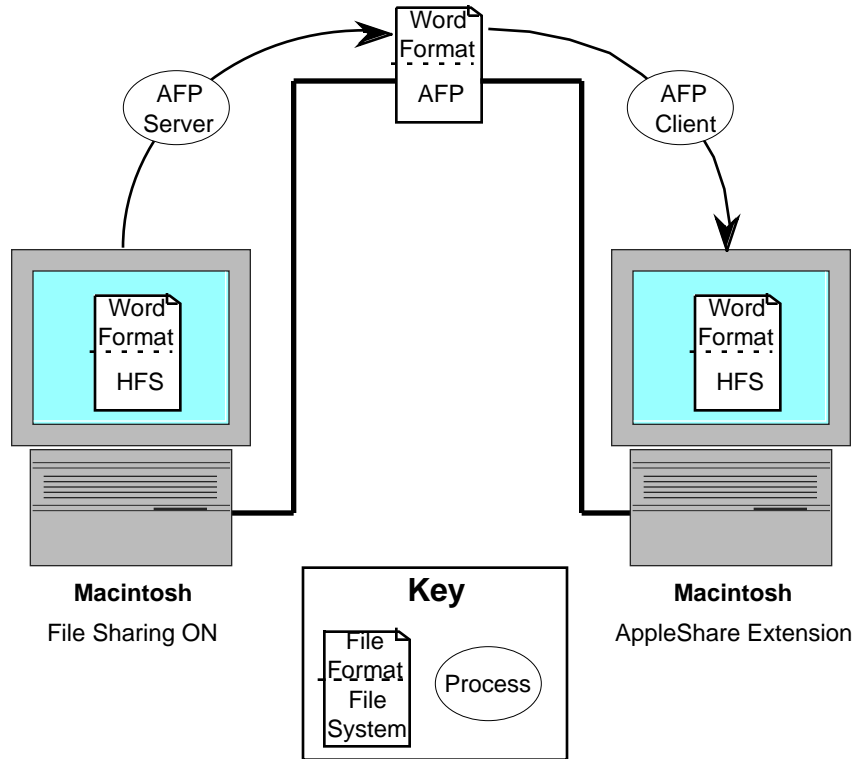


Figure 5-25. When sharing a file between 2 Macintoshes, the Mac’s use Apple’s network file system, AFP.

The third basic element of the Presentation Layer is that the application used on the two devices can use a **Common File Format**. In this case, since your friend will be editing the file with Microsoft Word, the same application that you used to create it, there’s no need to change the format of the file. If your friend was using WordPerfect on the Mac, you could use WordPerfect’s import capability to read the Word file. If your friend 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 the mainstream of computer types and applications. You may find it difficult if your friend is using a Tandy TRS-80, for example (there are still some in use).

TROUBLESHOOTING MACINTOSH NETWORKS

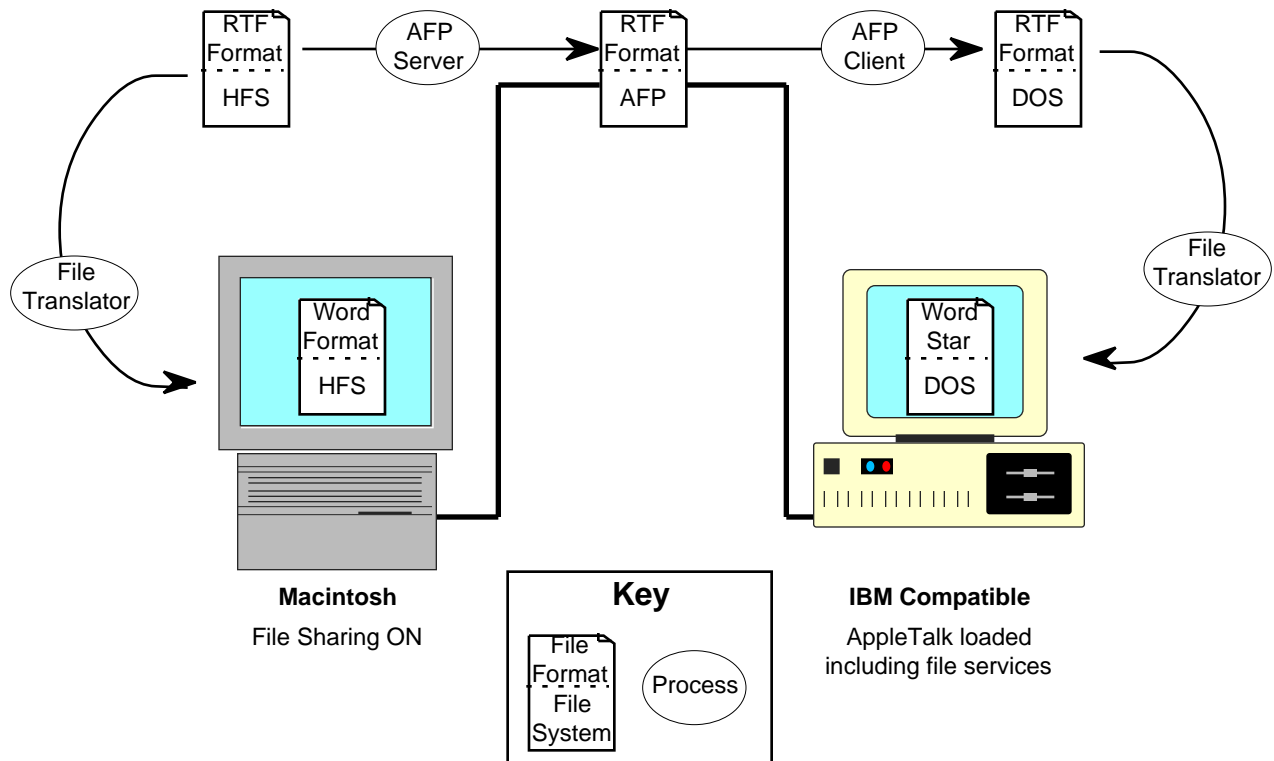


Figure 5-26. In this example, several levels of translation are required. Both the file systems and the file formats are dissimilar, and a common denominator for each must be found.

Although the file is now physically present on your friend's Mac's hard disk, there may yet be difficulties in the way the data is represented that may keep your friend from using the file successfully. This is the fourth basic element of the Presentation Layer—**Common File Components**. For example, let's say that because you enjoy working with graphics, you used Canvas to render your company's logo with your department name incorporated into the design. If your friend does not have Canvas, or another application that can edit this graphic, he'll have trouble modifying your design to indicate the name of his department. And since your rendition of the company logo uses the Futura font (with a point size of 45), unless your friend has an outline font of Futura in his system, the logo may look a little different when printed from his Mac.

Also, one of the things that your friend especially liked about your report was that you included an appendix that listed all of your company's current activity codes with a short description of each code. Since this information changes frequently, your appendix is actually a subscription to a report from an Omnis data base that gets updated automatically each time you open and print the document. Up until now, it hasn't mattered if your friend was using System 7, but now it does if he wants to make use of this live link to the report, which also means that he'll need access to the folder where the publisher of the subscription is located.

If you remember when I introduced this scenario, I said that this was a simple case because your friend has the same kind of system and will use the same application. As you can see, however, it can quickly become complicated depending on the nature of the data. If your friend was a DOS or Unix user, your problems would be even greater, because there would be more chance for disagreement among the various elements.

Even though some of the potential pitfalls that I have listed are not strictly network problems, they are all concerned with data representation and are within the concerns of troubleshooting at the Presentation Layer. In summary, then, in order for two devices to share or exchange data the critical issues of the Presentation Layer are:

1. Proper System Configurations
2. A Common File System
3. A Common File Format
4. Common File Elements

Troubleshooting System Configurations

Troubleshooting system configurations is as complex an activity as troubleshooting network problems. While it's not feasible to write a comprehensive guide to this subject in the context of this book, there are several aspects of this subject that deserve mention here because network managers are often called upon to troubleshoot computer systems in the course of troubleshooting a network problem

One of the aspects of the Mac that users like is its ability to be personalized and enhanced. This is accomplished through the use of files known as System Additions, which includes 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 because of this, they can potentially create problems for other applications and additions when a modification by one addition conflicts with the modification made by a different addition or with the needs of an application. If programmers were perfect (and Apple's documentation as well), 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 programs), and much of the following applies to these as well.

Some additions are made to extend the Mac's capability. For example, the System Extension MacTCP gives a Mac the ability to use TCP/IP protocols and DAL gives a Mac the ability to access Data Bases that speak this 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 amount of 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 his RAM cache and select whether he wants to use virtual memory or not. 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 SoundMaster, which among other features allows you to assign a "belch" sound to accompany the activity of emptying the trash on the desktop.

Other additions are designed to increase a user's productivity. My favorites in this category are QuicKeys, a macro utility, AutoDoubler, an automatic file compression manager, and 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 the After Dark and Pyro screen savers, for example, or between Super Boomerang and Shortcut. If they use one, they should not use the other. It's appropriate for a network manager to exert some control over the user's system configuration if he is expected to support the user properly and troubleshoot problems that occur or if license agreements are involved. There is currently a wide range of 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. Besides which additions are allowed, attention must also be paid to version control, since version *x* of an addition may cause a conflict with software *n*, but version *y* will 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

TROUBLESHOOTING MACINTOSH NETWORKS

NetOctopus, On Technology's Status*Mac and Sonic System's Radar. All of these applications were designed to be multi-purpose network management tools. Even though none of them were designed specifically for the needs of a troubleshooter, all of them have a few functions that can be used for troubleshooting and you'll definitely want to use them for that purpose.

When tracking system additions in a troubleshooting scenario, the 3 most important questions to consider are:

- Which system additions do the users have loaded on their computers?
- Which versions of those additions are they using?
- What have they added or subtracted recently?

The reason that it's good to establish what's been added recently is that you are often called on to troubleshoot a system that "used to work". In this type of situation, looking at what aspects of the system configuration have changed since the time when it used to work will often provide a good clue.

If you are currently trying to make a decision regarding which of these 5 products you wanted to invest your time and money in, you will want to consider use a broad range of purchasing criteria. How well each of these products perform with respect to this one particular troubleshooting task is not necessarily a good predictor of how well they will perform for the broad variety of tasks that you may want to use them for. Nevertheless, let me rate the effectiveness of the 5 products mentioned above in terms of how well they answer the 3 question posed above.

Status*Mac isn't particularly useful for this kind of troubleshooting because it can't scan systems on demand. It's orientation is 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 3 modules that are available separately. You can use the Network Manager module to gather the data from systems connected to the network and wither view this data on the 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 data base functionality. In troubleshooting, however, speed is often a consideration, and this two-step process may be slower than you'd like to have if you need to 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 that you exported from the Network Manager module with a spreadsheet or word processor program. Unfortunately, with regard to system additions, GraceLAN only tracks the file name 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 is the only one of these 5 with any DOS support, and there are many other things that GraceLAN does well.

Both Network SuperVisor and Radar can scan across the network for system additions and both can tell you the version numbers, but only Network SuperVisor can tell you how much memory is being used by an extension and what other processes it may have started when it was activated. Radar has an unattractive interface when viewed on a monochrome monitor, which usually doesn't bother me, but in Radar's case, detracts from it's usability. You can't tell, for example, if a button is inactive in monochrome. It also has no database functions, but it has some printer maintenance features found in the LaserWriter utility and has alarms similar to the AG Group's NetWatchMan.

The bottom line is that while none of these products has everything I would want in a troubleshooting tool, the best of the current crop for troubleshooting is Network SuperVisor, which in addition to the features I mentioned previously has a self-scan feature built into the Control Panel that you install on each users machine. Using the Network SuperVisor Responder alone (without the data base application), I can get much of the information I need for troubleshooting, because I can save the scan information to a tabular format and open it in a spreadsheet. That's a nice feature because when I'm out in a user area and away from my own Mac, and I don't have the Network SuperVisor application to work with or the data base files present, I still have a tool I can work with.

Here's an example spreadsheet analysis I did on the saved output that I got from Network SuperVisor Responder on the same Mac on two different days. Besides the two columns of information, one for each day, I made a simple function in a third column to automatically compare them. I find this helpful because it's easy to miss small differences between all of this text data.

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:	OK	4096K	Change!
AppleTalk:	Version	Version 57	
QuickTime™	441K	QuickTime™	441K
SCSI Probe 3.4	20K	SCSIProbe 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
SuperVisor Responder™	245K	SuperVisor Responder™	245K
System	1918K	System	2022K
System 7 Tuner	21K	System 7 Tuner	21K
Timbaktu	127K	Timbaktu	127K
Voice Record	159K	Voice Record	159K

Figure 5-27. Using the Network SuperVisor Responder and a spreadsheet, you can compare system scans from different days. On the left is the system configuration information from the same Macintosh on two different days. On the right 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.

Finding information about potential conflicts between system additions can be difficult. In rare cases, they're listed in the user's guide for a product. The server installation guide for QuickMail 2.5, for example, warns that the server process was 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 were aware of the conflicts between Now Utilities 3.0 and After Dark 2.0u, and would tell you that you needed 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, among other public online systems, in addition to providing a disk-based upgrade for a nominal charge.

In other cases, you'll have to use a trial-and-error method 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 of the number of combinations that must be tried.

Typically, you start by clearing out every non-essential system addition, start the machine and try it for a while. If there are no problems, then you can add a system addition back in. The principles of Scientific Method would dictate that you must re-incorporate only one addition at a time, but very few network

TROUBLESHOOTING MACINTOSH NETWORKS

managers actually do this, and I am not among them. Many times you can make some guesses about what addition might be causing the problem by thinking about the condition under which the problem occurred. For example, if your Mac crashes only when it's unattended, then you 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 your problem happens when you have an Open or Save Dialog Box on the screen, then you should first suspect that an addition that changes the appearance of these kinds of dialog boxes is the cause; you may check the ShortCut, SuperBoomerang or Directory Assistance II system additions first. If your problem only occurs during a complex analysis or computation, then you may suspect a bug in the program or a lack of memory. This kind of analysis does not always provide a fruitful clue, but it is often of help. The key is to notice the common threads in how and when the problem occurs, consider the resources that are involved in the processes that are occurring at that time and look at the system additions that deal with those resources.

Online services can also provide you with good information both in their discussion sections and in files uploaded to the services by users and user groups who track such conflicts. In the figure below, you can see a screen shot from "INITInfo Stack 4.3.1" by Glenn Brown and Gary Ouellet that I downloaded from America Online. It's a shareware stack (\$15), that is frequently updated and contains conflict information for over 140 system additions and applications, as well as hardware problems that are known to exist in all of the Macintosh models.

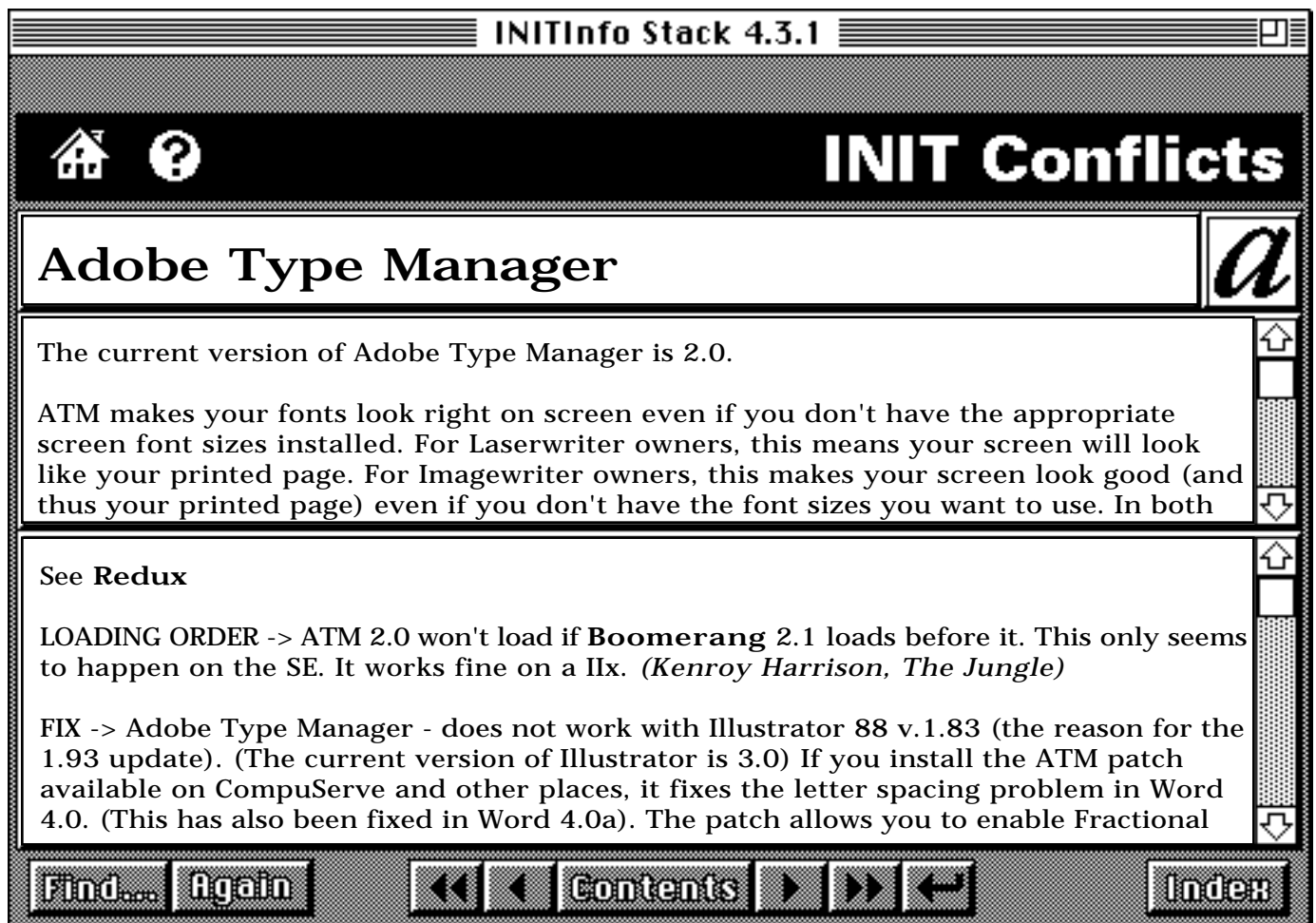


Figure 5-28. This shareware HyperCard stack, available on America Online, Compuserve and Genie, contains a wealth of system addition conflict information

A terrific utility worth mentioning here is Help!, which can perform a complete scan of your hardware and software configuration, then checks your 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. Below is the summary at the beginning of 22 pages of documentation and analysis and the listing of one of the 11 non-critical 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 5-29. Help! analyzes a Mac's system configuration and reports any anomalies. The summary of the analysis and the listing for one of the non-critical problems is shown.

Of course, you also should check for problems with the user's disk drive and also check for viruses. With disk analysis programs, even though you may want to recommend only one for your users to purchase, it's worthwhile for you to have them all in your own toolkit. My favorite in this class of utilities is the Norton Utilities for the Macintosh 2.0, which will work for 98% of the problem cases you throw at it. For that other 2% of your problems, have Disk FirstAid, MacTools Plus, the Fifth Generation utilities and SilverLining waiting in the wings. You never know what will work.

If your system is crashing, it may be giving you a numerical code that indicates the conditions that may have 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 a bus error resulted from a software problem, there is nothing more specific that can be determined.

A great source for interpreting Macintosh error codes can be found in the appendices 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 mine can be found on America Online, and is called System Errors Guide 7.0.1 by “Dr. Pete” Corless, an Apple employee.

You can sometimes make a configuration problem go away by re-installing the software. The installer utility will check for the existence of all of the necessary components and install them if they do not exist or are out of date. Help!, for all of its abilities only checks for the presence of configuration errors, does not look for the absence of components required to perform a desired activity, only the presence of components that may obstruct a desired activity.

When you re-install software, there are 2 levels of rigor. In the first, a “dirty” installation, you simply run the installer with the software that you are trying to install already in place. The installer may, depending on how its script is programmed 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.

TROUBLESHOOTING MACINTOSH NETWORKS

You usually won't know for sure which one of those happened. In a clean re-installation, you first remove all of the software, and install the software as if it were new.

Establishing a Common File System

If you are sharing files between Macintoshes, then 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. Even when performing complicated file operations, such as allowing multiple concurrent access to the same file in a data base, there are typically no problems because AFP was designed specifically for the needs of HFS and generally for the needs of other native file systems such as DOS.

When using Mac files from a PC, however, there can be problems because of the way that the PC application may 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 1-2-3 cannot reserve the file for its use in a way that it finds satisfactory. 1-2-3 will tell 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 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, of course, the more likely that a problem of this nature will occur. That's because the way the native file system accomplishes an operation such as reserving a file for use may not have a suitable equivalent in the Mac's native file system, HFS, or in AppleTalk's network file system, AFP.

Typically, the remedy of transferring the file from a network volume to a native volume as mentioned above will cure these problems, although it may reduce the users' ability to share the information. Another remedy to try in this instance is to try to share the file when it is on a dedicated AppleShare server (still doesn't work) or on another type of file server that is mountable by both DOS and Mac platforms, such as a Novell file server (works). Even though all of these file servers make use of AFP, they might do so in ways that are different enough to alleviate the source of the problem. Also, you should probably notify Lotus of the problem (I did).

When you use a gateway such as GatorShare the translation passes through one more step, and so the potential for problems is even greater. Here the Mac's HFS representation of the file system is translated to AFP by the AppleShare extension, the GatorShare transforms the AFP representation into an NFS representation of the file system, which is finally translated into (typically) a Unix-style file system. When you have this many translations going on, no matter how well they are performed, there is always the likelihood of error, particularly when performing complicated transactions. Usually, though, by experimentation such as was mentioned concerning the Lotus problem, a solution can be achieved.

Troubleshooting File Formats and Components

When dealing with file formats and components, you also may have to experiment a bit to get the right combination. There is a great multitude of formats for both text and graphic files and an equally great number of utilities to convert files between them both in the public domain and in the form of commercial products. As in the translation of file systems, there is always the risk that a bit of the meaning may get lost. If you are trying to use a word processing file that included footnotes in its original form in a word processor that does not have a feature for footnotes, 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 may be times when you can't translate your macros or charts.

Some programs are especially good at being able to work with a wide variety of files. An example is Denba's Canvas application, which can work with a wide variety of both Mac and PC graphics file formats. Some network managers buy Canvas specifically for the purpose of helping users perform file format conversion and have no other reason to use the program. Another popular utility is the Apple File Exchange program, which can be extended with translators such as those from DataViz.

One problem that you may run into when you try to create or export a Mac file while working on another type of computer a PC is that the program might not have a way of giving the Mac file the file attributes it needs. Let's say that you want to create an EPS (Encapsulated PostScript File) from a graphic created on a PC so that you could use the image in an Aldus Freehand drawing on the Mac. When you try to open the file in Freehand, the file name will not appear in the "Open" dialog box unless the file's attributes indicate that is an EPS file by having a file type of "EPSF". There are many ways that you can 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 the figure below.

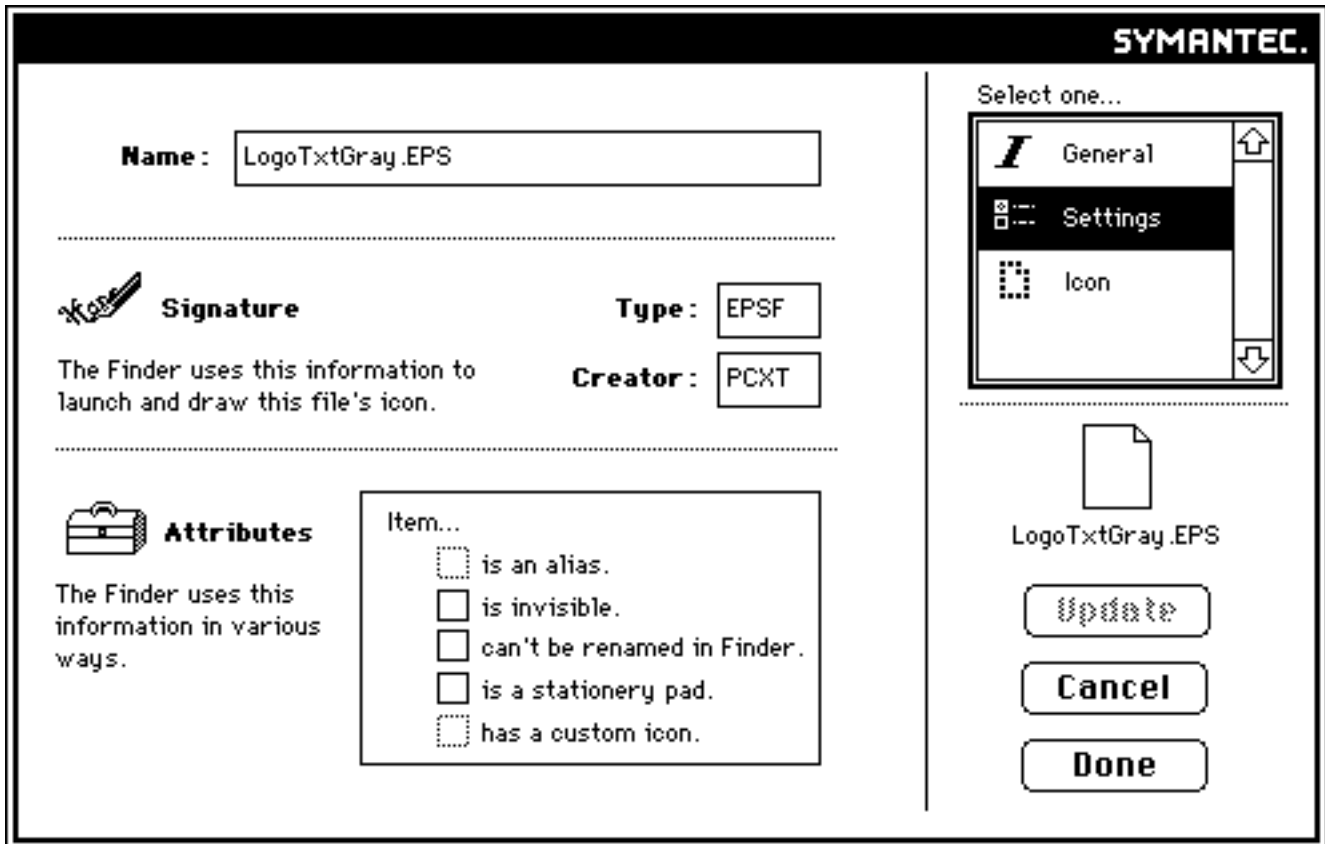


Figure 5-30. With the FastFind utility in Norton Utilities for the Macintosh 2.0, you can easily change the "type" of a Mac file.

Network Process Descriptions

Process mode is the most revealing and most challenging mode of network troubleshooting. It's most revealing because the troubleshooter looks at exactly what happened—the record of network events—in a protocol analyzer. It's most challenging because it requires the greatest knowledge. The troubleshooter gets 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, ask for and receive status reports, ask for, accept and refuse connections, as well as send and receive data. In most processes, there is a defineable 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 though, though, 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 the troubleshooter must understand the inter-dependencies of the events. The troubleshooter must conceptualize each device as an information agent, “The Gonkulater performed Event A in order to gain the information necessary to perform Event B, which sets the stage for Events C & D. If Event A does not get a valid answer, then the Gonkulater will perform this other chain of events in order to get the information it desired from Event A.”

When the troubleshooter views any event that a device initiates or responds to, his main concerns for are:

1. What information does the device need to know in order to perform this event to perform this events?
2. How did it find out that information?
3. What information does this event gain 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; the task of the troubleshooter is to sift through those mountains for the crystals that lead to an understanding of the problem.

Device Startup

When a networked device powers up, sooner or later it needs to use the network. Most often, 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, settings and preferences are retrieved—and 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 a troubleshooter. Other events can be seen on the network using a protocol analyzer.

Watching a workstation initialize its protocols will tell you what software processes it intends to run, how those software processes make their presence known on the network, and may give you insight into why the device is or is not making a connection .

In this section, I'm going to give 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 they are placed in. Because the burden of routing is placed almost exclusively on the internet routers, routers behave differently during startup than non-routers.

The two kinds of devices do have some similar tasks during startup because both kinds of 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 will 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 non-routers. It is 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 non-routers, LocalTalk and EtherTalk, and the presence or absence of a pre-existing router—are covered in this section.

I have found that “watching” the startup procedure often provides important clues that can lead toward 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) needs to register an NBP service name on the network. The first network-visible event is that the Mac will perform an “address bid” to secure an AppleTalk node address for itself.

It does this by choosing a “tentative address” and then checking that no other device already has that address. If the tentative address is unique, the Mac will adopt the address as its permanent node address. If the address is already in use, the Mac will randomly pick another address and check it for uniqueness. The process proceeds until a unique address is found or the Mac determines that a unique address cannot be found.

One of the elements that Macs store in Parameter Ram (PRAM) is their node address. When they are turned off and then later restarted, they will 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 is booting up. The Macs will send a large number (640) of LLAP (LocalTalk Link Access Protocol) NodeEnq packets to the Hint Address, node 63. The purpose of a NodeEnq 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 will receive this packet and very quickly respond by sending a NodeAck packet back to the booting Mac. In effect, saying “Yes, I’m using 63 for my node address.”

If the booting Mac receives a NodeAck, it will stop sending NodeEnq packets to Node 63 immediately. It will then quickly (<.001 sec) choose a random number between 1 & 127 for its next tentative node address and send NodeEnq 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 use “63” as the node address successfully checked and adopted by the device. This is the node address portion of the AppleTalk address only—at this time the Mac still does not know whether there is a router on the network or not, or whether the network has a network address assigned to it or not.

Because LocalTalk can only have a single network number, it is not 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

TROUBLESHOOTING MACINTOSH NETWORKS

non-routers on LocalTalk assume a network address of “0” when they enter service. “0” means “this network”.

Registering an NBP Service Name

Following a successful address bid, the name registration process will begin. First, the System extension (we’ll use Timbuktu as the example) will use NBP (Name Binding Protocol) to search for the name that 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 that the Mac is on is part of a larger zone, then the Mac will need to use the router to search for names in all of the networks that make up the zone.

There are two ways to discover whether there is a router 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 10 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 that it’s on. It uses the default network number, “0”, which means simply, “this network”. Without routers “this zone” is synonymous with “this network”. On a network with a router, the Mac always makes the assumption that the zone is larger than the network.

The Timbuktu software process will engage 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 3 parts—an object name, a service type and a zone. Timbuktu might ask NBP to search 3 times for the object name of “040567”, service type of “TB2 Serial” in the “*” zone. The form of this query is expressed as “040567:TB2 Serial@*”. NBP will respond by sending 3 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 data base query in a structured language. In fact, NBP conceptualizes the network as a distributed data base—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 wild card characters. An asterisk, “*”, in the zone field means, “this zone”, and matches any zone name. While there is no wild card 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 is not already in use, Timbuktu will then register the name that others users will see when they start up their copies of Timbuktu and look for available hosts. In our case, the 3 LookUps for “040567:TB2 Serial@*” will be followed by 3 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. If there were any devices on the network that have one of these two service names in use, they would send an NBP Reply packet to node 63, socket 254. Socket 254, however, is not necessarily the socket that Timbuktu will take when it registers its name; it is merely the socket used for searching.

The NBP LookUps are usually sent to insure that the name that a process wants to register is not 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 could not 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 wild card 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 will repeat 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 mail and file server that the user selected for automatic login during a previous session. Login processes are detailed later in this section.

Performing 3 searches per name is the most common practice, but some software processes use more than 3 LookUps and some use less. The PPCToolBox, for example, will send 18, while a QuickMail server will use only 2 searches for each of the 5 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 2 processes—an address bid using NodeEnq's followed by NBP name searches. Each device is programmed to send a large number of NodeEnq's to make sure that their address is unique. The Mac ROM tells the Mac that a test of 640 packets without a NodeAck is sufficient to establish uniqueness. Other devices' programs may use a different number of NodeEnqs as the test for uniqueness—some will send many more NodeEnqs than the Mac's 640. Due to the fact that I've never seen this take more than 3 packets, even 640 NodeEnqs seems like overkill, but the address bid on a Mac only lasts a second or so and consumes less than 30% of the bandwidth during that second.

A device that is 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 is not connected—the LocalTalk signaling hardware has no method for checking whether its connected to a "live" network or not. In this case, all of the searches for addresses and name will, of course, reveal no duplicates. Later, if the device is connected to the network, its address and service names could possibly duplicate other devices' names and address.

All of the extensions that require name service perform LookUps of this type. Most extensions will send 3 LookUps per name, but some, such as the PPCToolBox, will send more than 3 if a more rigorous search is desired.

More About Name Registration

Using NBP to search for names is not 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 previously mentioned, 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 AppleTalk internet addresses, which devices must use to address the packets they send. Just

TROUBLESHOOTING MACINTOSH NETWORKS

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 only take the trouble to register sockets if they want to provide a way for other software processes to be able to locate them later on. There are 3 reasons why a software process would 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 function 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 function 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 of these.

In serial number searches like the one described above, the finding of 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 in when the application was first run (fourth Dimension, e.g.). Sometimes the user is presented with a dialog box to enter a new serial number and may be notified of the name of the user with the duplicate serial number.

When I cover Name Binding Protocol in the classes that I teach, a frequently question is, “If NBP searches for serial numbers are only conducted in a zone, does that mean I 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 does not do this, but two users with the same serial number, even in different zones cannot connect with each other.

While serial number duplications usually result in a negative action on the part of the software process, for duplications of names that don’t involve serial numbers, a more normal response is that the extension will 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 will then register that name along with the type and zone.

Although this name modification process happens in Macintoshes, a more significant example of this type of name duplication occurs with shared devices like LaserWriters. A common scenario is that an organization will buy 2 LaserWriters of the same type (say 2 LaserWriter II NTX’s). 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 his System File and LaserWriter Driver. It’s conceivable that at some point in the future 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 is avoided if the installer uses the “Namer” software to give the LaserWriters unique names.

Some hardware devices include their serial number within their default NBP name, not for serial registration, but to avoid the occurrence of this kind of duplication. An example is the GatorBox CS, which has a default name like “CS023468:GatorBox”, where CS023468 is the serial number. In a case where more than one GatorBox is attached to the same network, the default NBP name of each GatorBox would already be unique.

Startup on LocalTalk with a Router

As mentioned earlier, that a Mac checks the network for routers by sending 2 RTMP Request packets before it performs any name searches. If there are routers on the network, they will 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 only sent 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”. Anytime 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 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 is a router, the Mac does not 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 will 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 does not know and does not need to know what zone it is 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 determines which of any of the other networks that it knows about are also part of that zone. With a list of networks gained from the zone table, 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 is directly attached, it will simply broadcast 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 is not directly connected, it will send 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”. The reason is because that is 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 will send 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 common situation that 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 during the entire process, they will still be locatable by other software processes because of the “*” zone name.

TROUBLESHOOTING MACINTOSH NETWORKS

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 AppleTalk Off?

There are times when 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 is put “to sleep”, when some network management tools are used or when the network is switched from one data link to another, say from LocalTalk to Ethernet. For whatever reason AppleTalk was deactivated, when AppleTalk is started up again without a system startup, say when AppleTalk is reactivated in the Chooser, only some of the network processes that initialize at system startup will re-initialize themselves and become active again. A few others can be “jump-started” into action again, but most are not recoverable without restarting 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 and is described in more depth in Section 7.7-“Using an AFP Server”.

Timbuktu is an example of a process that can be jump-started. In Timbuktu, this is done very simply by opening the Timbuktu desk accessory. In other applications, there may be a menu choice to start the process up.

Both QuickMail and Microsoft Mail Servers, as well as Liaison, are examples of processes that can only be started when the Mac boots up. The majority of network applications fall into this category.

Care should always be taken when disrupting AppleTalk for any reason, but especially when communication between applications is in progress. Work can be lost and people can become very angry when this happens. In light of the fact that System 7 provides the opportunity for every Mac to be a file server, users should be made aware of the consequences of deactivating AppleTalk and should be taught to first check the File Sharing Monitor to discover whether any users are currently logged into their Mac as a file server. This is becoming more important as the number of ways in which networked computers are inter-connected and inter-dependent is increasing.

Router Startup on LocalTalk

Routers, of course, are also AppleTalk nodes, and their startup processes share many characteristics with non-routing nodes. Like non-routers, they also must acquire a node address and register their NBP names. Typically, however, routers will perform more thorough checks for duplicate addresses than Macs. Also, there are some small differences in the startup sequences of routers depending on the manufacturer. The particular 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 (<\$1500 per port). The example assumes that no other router will already be present on the LocalTalk network (the vast majority of situations).

The GatorBox sends out 3840 NodeAck’s over a 20 second period and then sends an RTMP Response packet (Routing Table Maintenance Protocol, see Section 7.2-Maintaining Routing Tables and Zone Tables for more details) with only one tuple for the local network. Then it looks for the names of the services it wants to register on the network. It registers 2 services with the same object name, but 2 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 many 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 only performs 2 name lookups for the “GatorBox” type and 1 for the “SNMP” type.

After these 2 events, about 25 seconds after sending its first NodeAck, it achieves steady state as an AppleTalk node, broadcasting its RTMP information every 10 seconds. During this time it has also begun the processes described in Section 7.2.

Among the more expensive “multi-protocol” routers made by manufacturers such as cisco, Wellfleet and Ungermann-Bass, there may be significant differences in not only 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 they have already developed design philosophies slightly different than designers who began their design careers with AppleTalk. One example is how the Ungermann-Bass MaxTalk router handles the situation when its tentative node address is already in use. Unlike “traditional” AppleTalk routers (FastPath, GatorBox, Ether•Route, Liaison, etc.), MaxTalk routers do not 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 verify that its AppleTalk address is unique and valid and register its sockets. Phase 2 EtherTalk nodes must also be sure that they are assigned to a specific zone. On startup, an EtherTalk node must:

1. Make sure that it has a unique address that is valid for the network to which it is attached.
2. Make sure that 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 than in LocalTalk. Instead of of the NodeEnq packet used in LocalTalk, the Ethernet node uses the AARP Probe packet. The message of the AARP Probe packet is the same as the NodeEnq 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 only happen when Ethernet was chosen during a previous session), it will send 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 will send 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 below 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 on 0.2 second intervals. Some nodes, routers for example, may send more AARP Probes than this number if a higher degree of certainty is required for the unique address.

If the address bid is successful (no AARP Replies are received), the node will then verify its saved zone name. In our 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

TROUBLESHOOTING MACINTOSH NETWORKS

name for my network?”. If there is no reply, the Mac will send out 3 identical ZIP GNI’s for this zone name. If it receives a reply to the first ZIP GNI, it will not send any more. Only routers can respond to ZIP GNI’s. The ZIP GNI Reply packet will tell:

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 does not know which network numbers are valid, it will take send 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 a temporary address that will allow them to contact a router. The node will then ask the router for the range of valid network numbers and then bid 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

There are several situations in which a Mac may not know which network numbers are valid; the most common is when a user switches his 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. A typical example is shown below. Notice that the node address, 35, is the same as above for the same device (compare Ethernet addresses). That is because Macs will use the node address portion of their hint address and randomly choose a network address in the startup range.

After the Mac gains an address in the startup range, it will broadcast a ZIP GNI. If the Mac has a saved zone name, it will verify that zone name in the ZIP GNI. If the Mac does not have a saved zone name the ZIP GNI will be broadcast with a blank zone name.

On networks with more than one router, the Mac will use 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) will 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 experience a symptom of that misconfiguration in that they might be informed that their home zone is invalid at this point in the boot process. This can occur because when users select a home zone using a ZIP GetLocalZones packet, the Mac does not broadcast the packet, but send it to the address of the router in their RTMP Stub. This router is typically the last router they heard from—either the last router to send that Mac a packet or the last router to broadcast an RTMP packet.

There is a certain degree of randomness to the router that gets chosen to supply the zone list and a certain lack of randomness for the router that is first to respond to the ZIP GNI broadcast. Because of this, the randomly chosen router may supply a zone name that is considered invalid by the router that is asked to confirm the zone name. This will happen only periodically, and although the users are notified that they are being placed in the default zone, many users do not report this occurrence; they simply re-select their home

zone from the Network Control Panel and continue working. Only after it happens to them 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. Finding these problems is discussed in Section 8.3.

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 that it belongs to. As in LocalTalk, it will send an NBP Broadcast-Request to the last router it heard from for this purpose. An interesting sidelight to this 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 will do so without the normal check for duplicate names. This typically has little consequence, but depending on the importance of having a unique service name, could cause a problem.

Router Startup on EtherTalk

There is a great deal of variation in router startup procedures on EtherTalk. As in the case of the non-router, the device will use 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 going about this.

Beyond meeting the necessary requirements for being an EtherTalk node, a router must, of course, fulfill its routing mission: supplying network and zone information on request and forwarding packets through the internet at the request of non-routing nodes. The exact details of this are covered later in this chapter.

To the extent that the router needs to cooperate and work with other routers, each seed router (network and zone information is pre-configured) will make an attempt to discover 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, but 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 with regard to this process is what they will do when they encounter inconsistent configurations on the network. There are 3 basic choices that the router can make:

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.

Examples of routers that have 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 have response 3.

Maintaining Routing Tables and Zone Tables

Routers separate AppleTalk networks from each other and act as “door-keepers” for the information that passes from network to network. Besides passing information, routers give a network its definition—network address and zone names. When a network manager configures a router (offline usually), she assigns network addresses and zone names only for the networks that will be directly connected to the router. All other networks and zones must be learned from other routers that are already on the network. When the newly-configured router is attached to the network, it will exchange its network definitions with the definitions that other routers hold and eventually learn the entire internet structure—the (logical)

TROUBLESHOOTING MACINTOSH NETWORKS

location of every network and every zone as well as the method for sending packets to those networks and zones.

Some routers, called non-seed routers, not only learn about distant networks from other routers, they also must learn about the local network. A non-seed router is configured without any information about the network that it will be attached to.

The method for data exchange among router 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 Routing Table Maintenance Protocol and the ZIP Process by the algorithms 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 is 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 will be the difference between the use of “Ethernet” and “EtherTalk”. I'll use “Ethernet” to indicate the physical network to which the router is attached and “EtherTalk” to indicate the AppleTalk network that is created on that Ethernet.

The RTMP Process always goes first. It is constantly monitoring the network. When the RTMP Process learns about a new network or a change to a network that it already knows about, it will store this new knowledge in the Routing Table. The ZIP Process is not monitoring the network, but is monitoring the Routing Table. When it notices a change in the Routing Table, the ZIP Process will make 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 the protocols are optimized for smaller networks, and don't scale well to large internets. Besides scalability issues, there is also considerable disagreement among vendors in how they interpret some of the rules outlined in the specifications for the RTMP and ZIP protocols (and in some cases the rules are under-specified). The result is that in a few key situations, routers from different manufacturers react in different ways to identical information. The most significant of these situations is how the router reacts when another router on the network has an incompatible configuration and how a router “ages out” obsolete routes.

Because of the troublesome nature of these protocols and the products that implement them, it is doubly important that network troubleshooters understand the details of the interactions between routers as well as the interactions between Macs and routers. This section is a companion to a section in Techniques that tells how to diagnose and resolve router problems; this section provides the theoretical background that can help you avoid router conflicts and understand the scenarios that foster conflicts.

Router Configuration

Routers are configured by using software bundled with the router. Every router has its own configuration management software and cannot be managed by other manufacturer's software. To describe configuration, I'll use GatorKeeper 2.0 configuration software and the GatorBox CS as the example. In addition, the discussion will be confined to AppleTalk Phase 2 protocols to keep the flow of this section as linear as possible. Also, although a few networks still use Phase 1 protocols, their number is rapidly diminishing.

The GatorBox, like many routers, has the capability to route data that uses other protocols besides AppleTalk (TCP/IP and DECnet), but routes only AppleTalk data in its default configuration. Each protocol must be activated and configured in order to function.

AppleTalk information is entered 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 would, of course, have a separate entry for each port.

The manager enters network address and zone information for both ports. If it is desired for one of the ports to be non-seeded, the network address entry for that port will be "0" and the zone list will be left blank. Router ports should only be non-seed if there are other routers already on that network that can supply them with information.

Some routers are "self-configuring", which means that they will generate network addresses and zone names automatically if none are supplied. This process will typically include 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 must have another router supply it with information in order to work. In the absence of other routers, a non-seed router that is not self-configuring will not acquire network address and zone information.

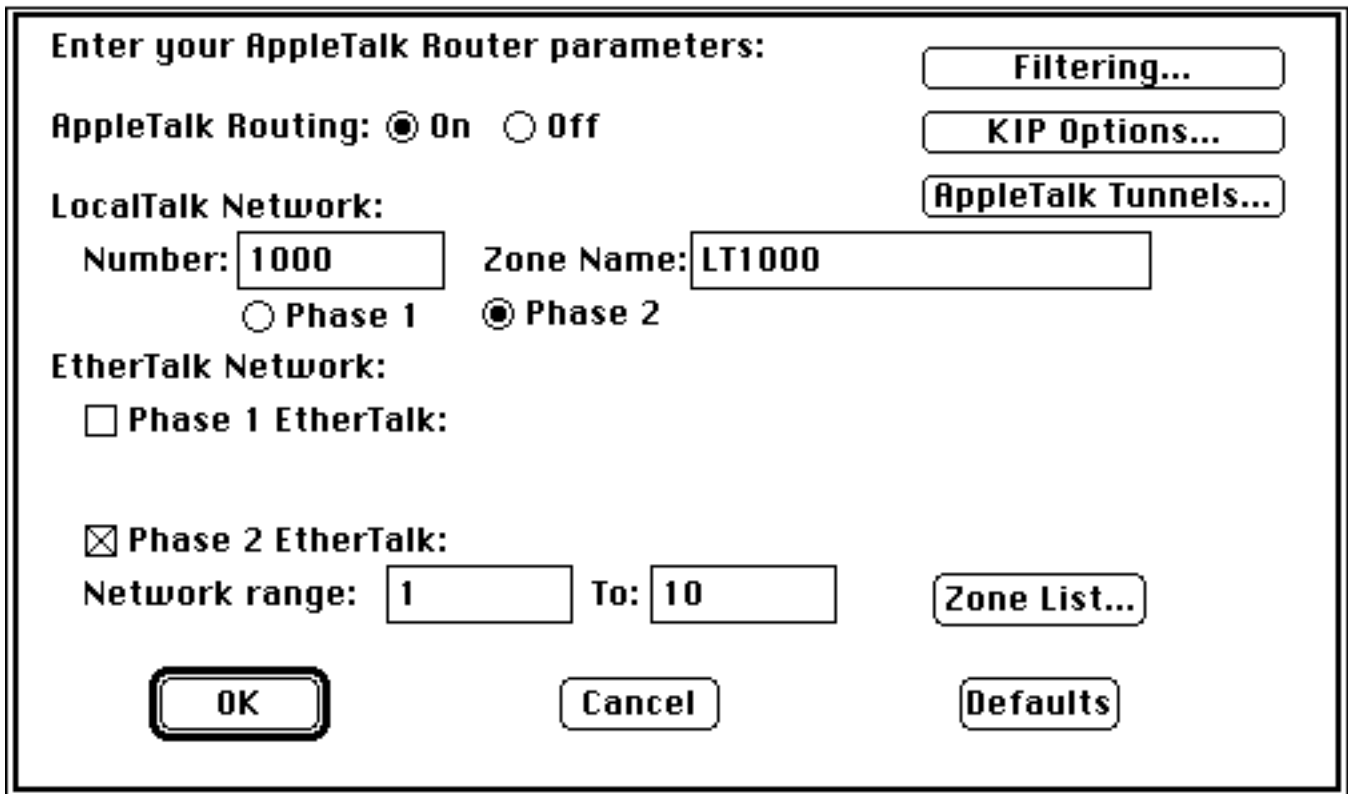


Figure 6-1. The GatorBox Configuration Window

In the GatorBox example, 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 4 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 below).

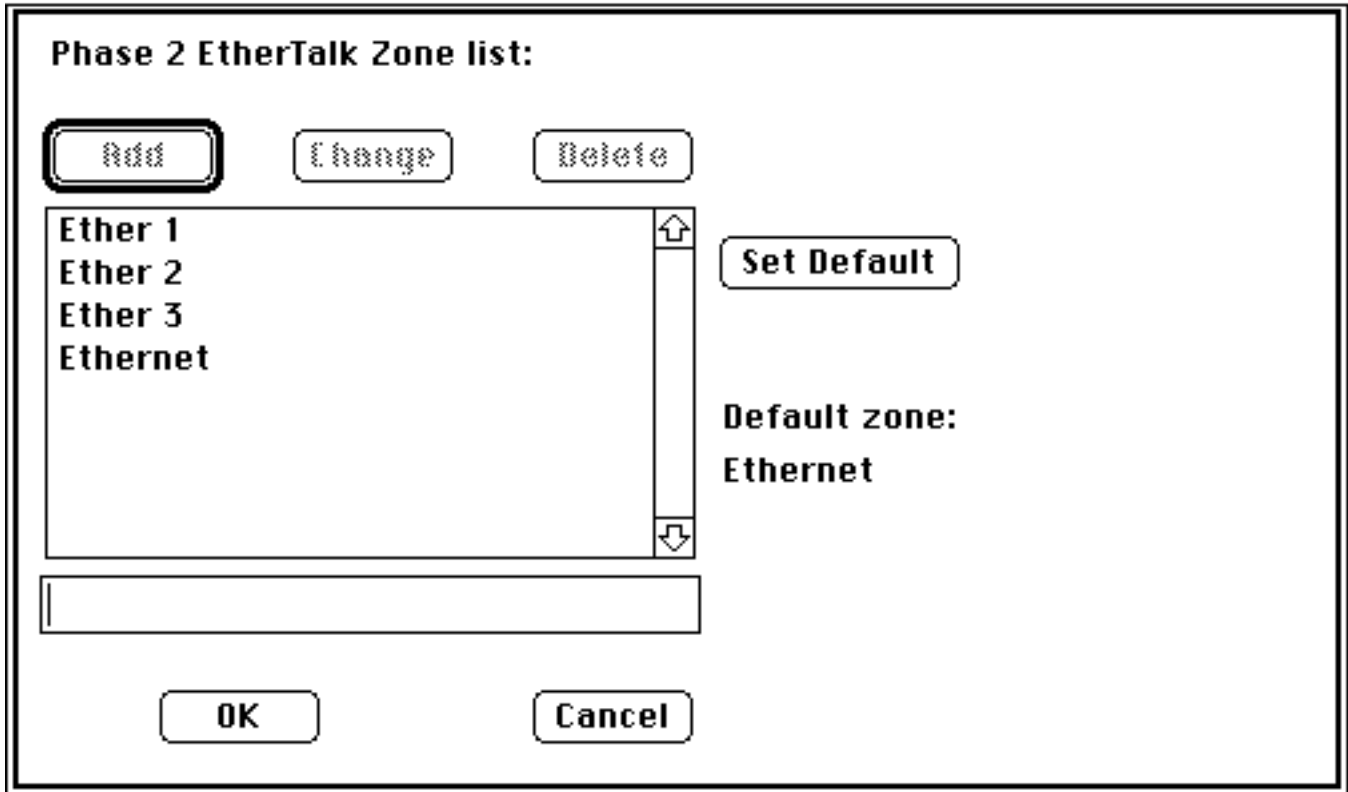


Figure 6-2. Entering the Zone List for EtherTalk

Routers are generally configured offline, powered off, connected to the live network and then powered back on.

Router Startup

Routers, like any network node, must take an AppleTalk address and register their service names. Both ports need an address, as each is a member of a different network. This process is described more thoroughly in the two sub-sections about routers in “7.1-Device Startup”.

The Routing Table

The RTMP Process will keep a routing table that lists all of the networks that the router knows about. The router will begin its service knowing only about the two networks that were described by the network manager when 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 described), the port that the router will use to send a packet to that network, the next router in the path to get to that network and a rating of the router’s belief in the reliability of the path to that network. The Routing Table for the GatorBox shown below 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 6-3. The GatorBox Routing Table at Startup

Because both of the networks that the GatorBox currently knows are local networks (directly attached), the hop distance to these networks is “0”, and the “Next Router” entry for both of them is not applicable. The “reliability” rating reflects how frequently the GatorBox has heard from the “Next Router” in the path that a network is reachable. With directly attached networks, however, the value will always be “Good”. For networks that are not directly attached, the reliability entry will start out “Good” and can be demoted to “Suspect” or “Bad” if the “Next Router” does not regularly report the network’s reachability.

The Zone Information Table (ZIT)

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 GatorBox at the moment it enters service the Table will only contain the entries for the EtherTalk and LocalTalk networks.

Network Number(s)	Zone Name(s)
1000	LT1000
1-10	Ethernet , Ether 1, Ether 2, Ether 3

Figure 6-4. 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 will make the appropriate change in the ZIT.

Route Information Broadcasts

Every AppleTalk router is required to broadcast an RTMP Response packet every 10 seconds that lists a subset of the networks that it knows about along the number of “hops” to those networks—one broadcast will be sent out each port every 10 seconds. The hop distance is the number of routers that are between a router and a network. For example, let’s say that our GatorBox, in order to send a packet to network “2305”, must send the packet to Router A, which in turns 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 0 hops away for Router C, 1 hop for Router B, 2 hops for Router A and 3 hops for the GatorBox.

In the beginning of a router’s service, however it only knows about the networks that it is directly connected to (networks with a hop distance of “0”), so these are the only networks that will listed in the first RTMP Response Packet is sends.

When the GatorBox first comes online it will broadcast an RTMP Response packet out each port that lists two tuples: one tuple that describes the LocalTalk network and one tuple that describes the Ethernet network. I’ll indicate the information in this packet by the shorthand of (1000,0 ^ 1-10,0) and 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.”

TROUBLESHOOTING MACINTOSH NETWORKS

In Phase 2, when routers send their RTMP Response packet, they perform what is referred to as “split horizon broadcasting”. Instead of listing every network they know about (as in Phase 1), the broadcast to each port simply lists the network on that port followed by a listing of all networks available on their other ports. Each network is described in a “tuple”, which is a pair of data values: the network address and the number of hops. Most routers list the network that they are sending to in the first tuple, but this is not required.

In the GatorBox’s case, 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 will not 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 “aging timer” associated with it 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 lapses after a tuple, the Reliability gets demoted to “Suspect”. At 60 seconds, the Reliability rating is demoted again to “Bad”. Some routers will eventually delete a “Bad” entry, but this is not required.

The RTMP Process

Rather than describe the RTMP Process occurring as other routers receive the GatorBox’s RTMP Response Packets, I’ll just focus on what the GatorBox is doing to build and maintain its tables as it receives packets from other routers. You should be aware, however, that just as the GatorBox is changing its tables as it learns about networks from other routers, the other routers are changing their tables based on information that the GatorBox is supplying. Network 1000, with the zone name of “LT1000”, is a new network as far as the other routers are concerned and will be entered into their tables accordingly.

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 Gator Box’s RTMP process will examine each tuple in the packet to see if a change in its routing table is warranted. There are 3 situations in which the router makes a change:

1. On hearing about a network for the first time, it will *add* a new entry in the Routing Table. It will list the network number advertised in the tuple, the identity of the port that received the packet, add 1 to the hop count shown in the packet and use the result for the “Hop Distance”, list the router that sent the packet as the “Next Router”, and set the reliability rating to “Good”.
2. On hearing about a network that is already known, the RTMP Process will compare the hop count in the tuple (first adding 1 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 will simply refresh the reliability rating. If the hop count shows a better path to the network (a lower number of hops), the RTMP Process will *change* the “Hop Distance” and “Next Router” data for that entry to reflect the better information. It will also set the reliability entry to “Good”, if it is currently at a different

setting. Note that RTMP uses hop distance alone as the basis for routing decisions. It does not consider link speed, reliability (based on experience), or 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) is not very well-defined in AppleTalk, and is one of the aspects of this process that varies as different manufacturers implement RTMP in their router products.

The First RTMP Packet is Received

After the GatorBox completes all of its startup activity, its view of the world currently looks like this (also reflected in the routing and zone tables shown above):

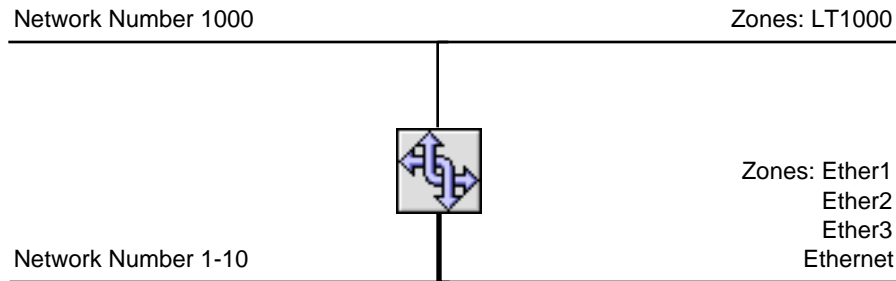


Figure 6-5. What the GatorBox Knows at Startup

Then it receives its first RTMP Response packet 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 does not have a 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 does not know about, so it creates a new entry in its Routing Table, which now looks like this:

Network Number	Port	Hop Distance	Next Router	Reliability
1000	LocalTalk	0	-	Good
1-10	Ethernet	0	-	Good
375	Ethernet	1	3.147	Good

Figure 6-6. 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 does not, 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 6-7. ZIP Makes a Change in the ZIT When the Routing Table Changes

TROUBLESHOOTING MACINTOSH NETWORKS

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 this, it will ask the “Next Router” in the Routing Table, node 3.147. The packet that it will send is a “ZIP Query” packet, whose English translation is, “What zone names are associated with network 375?” Node 3.147 will respond by saying, “Network 375 has the zone name of ‘Eastern Operations’”, in the ZIP Reply packet.

Note: ZIP Queries and Replies are only sent when new routers and networks come into existence and should not be seen in steady state networks.

After receiving the ZIP Reply packet, the GatorBox can update its Zone Table:

Network Number(s)	Zone Name(s)
1000	LT1000
1-10	Ethernet , Ether 1, Ether 2, Ether 3
375	Eastern Operations

Figure 6-8: The ZIP Reply Packet Tells the ZIP Process the Name of the Zone

And the GatorBox will now see the Internet as:

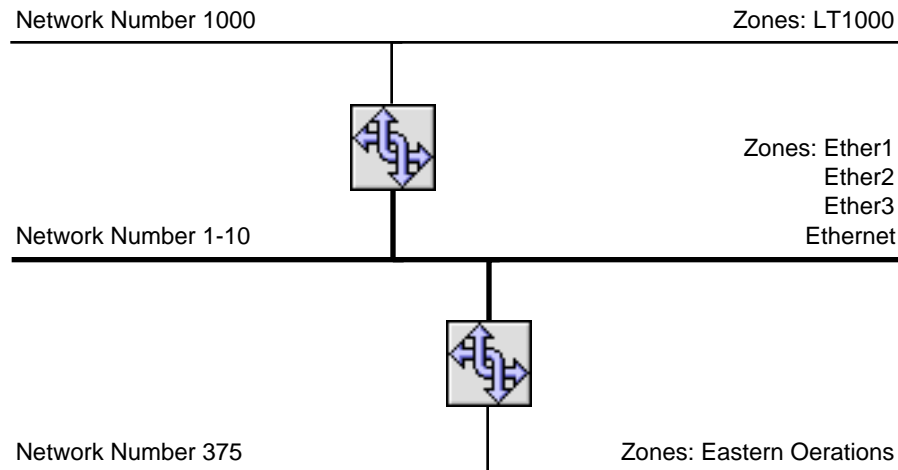


Figure 6-9. The GatorBox Has Learned Its First New Network

Notice, however, that the GatorBox never asked router 3.147 for the the zone names that went with the EtherTalk network, network 1-10. That’s because network 1-10 was already known. This makes some sense because it cuts down on the amount of communication necessary, but if the 2 routers had different zone lists, because of a configuration mistake, the routers would never discover their discrepancy. Finding this kind of error is covered in “Chapter 8-Checking Router Configurations”.

The Process Continues

The process of building the Routing Table and the ZIT will continue until the GatorBox has learned every network in the internet. It will send a ZIP Reply and receive a ZIP Query from every router on the Ethernet as it learns of their networks and zones. It will also receive a ZIP Query and respond 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 of "Operations" to the EtherTalk zone list.

To change the name of the GatorBox's LocalTalk network, we'll need to do so 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, "Eastern Operations" will never become part of their zone lists. But routers only ask zone names when the routing table shows a new network, so we must make the zone change in a way that the other routers will consider LocalTalk network 1000 to be a new network. There are two ways to accomplish this: 1) Change the network number along with the zone name, or 2) Wait long enough so that the other routers will have "forgotten" about network 1000.

The most sure 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 of network 1001. Besides sureness, advantage of this method is that the change happens as quickly as the GatorBox can re-boot.

The other method of changing 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 online. That way, when the GatorBox begins reporting network 1000 again, the other routers will treat network 1000 like a new network and send a ZIP Query to get the zone name. If you don't wait long enough, they will just refresh the reliability rating and use the old zone name. The question, "How long should you wait to make sure all the routers have aged out a network?" is a little like the question of, "How long after a meal should you wait before swimming?". There is a bit of 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 all "got it".

The second zone name change that we wanted to accomplish was to add a zone name to the list of zones on Ethernet. Unfortunately, this is much harder, because we'll have to re-configure and restart every single seed router and then restart every non-seed router to accomplish the change—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 above—either wait long enough for them to age out network 1-10, or change the network number. Like any internet configuration change, this change should be confirmed by checking router configurations using one of the router techniques outlined in Chapter 8.

Note: Apple has defined a way to accomplish this kind of zone name change automatically 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.

Using the Chooser

Although the Chooser is a "standard interface" for network resources, the Chooser's behavior varies widely among the different network resources that 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. All of the selections made in the Choose have no immediate action, but affect and print jobs that are begun later. 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 server. The Chooser works with a special kind of file, called a Chooser Extension, which is a software driver for a particular network resource. Chooser Extensions are special system files that are placed in the user's Extensions Folder (or in the System Folder in System 6). The Chooser is also a utility that selects and manages non-network resources that may be connected to either of your Mac's two serial ports.

TROUBLESHOOTING MACINTOSH NETWORKS

In Figure 6-10 below, the icons for the Chooser Extensions LaserWriter, AppleShare, MS Mail and Smart Labels Plus are visible. In the Chooser interface, the user selects which kind of service he would like to locate and the zone in which he would like to search. With the settings shown in the figure, the Chooser searches for all of the LaserWriters in the “Sales” zone. A broadcast is sent to each of the networks that make up the “Sales” zone that instructs any device that has an active process of the type “LaserWriter” to respond back to this Chooser. The Chooser then displays the names of the processes that respond.

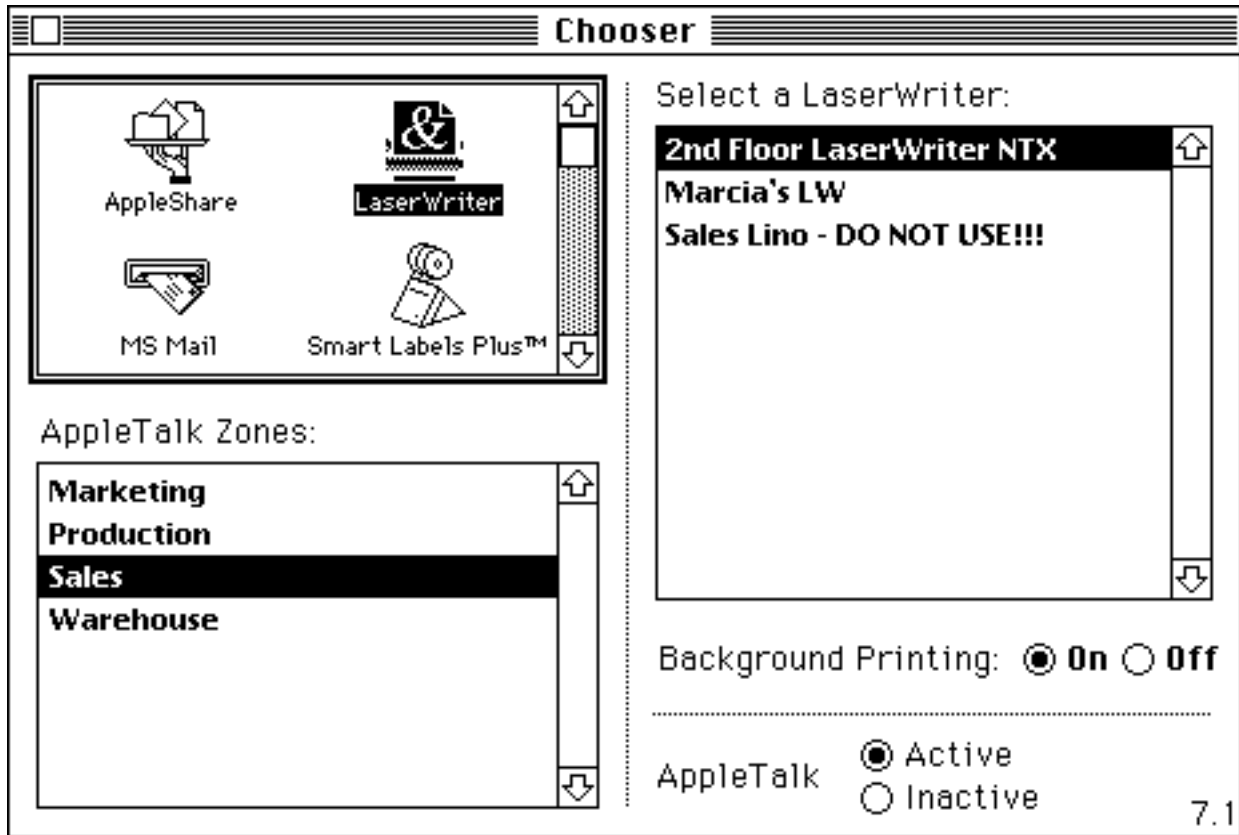


Figure 6-10. AppleTalk network resources are uniquely identified by a 3 component service name: Name, Type and Zone.

Some other network applications use proprietary utilities very similar to the Chooser to find and invoke resources. Examples are Farallon's Timbuktu (see Figure 6-11), Claris FileMaker Pro and Sitka's TOPS. These Chooser-like utilities share many traits with the Chooser and many of the processes described in this chapter will apply to these utilities as well.

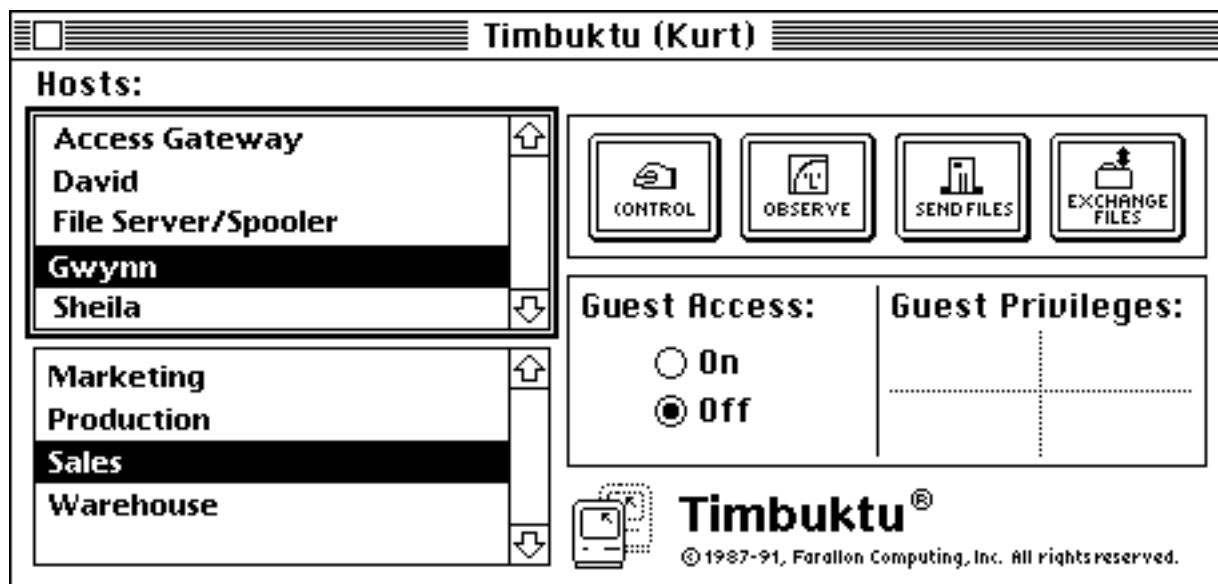


Figure 6-11. Farallon's Timbuktu has a Chooser-like utility to select network resources with the type of "Timbuktu Host". The zone and name components of the service names appear in the utility.

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 and can be completely unaware of network, node or socket addresses. This feature 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 was mentioned in Section 7.1, where I described how software processes established their service names and addresses on startup. In this section, the focus is on how NBP is used to locate and invoke processes after they are in service.

While the focus of this section is on understanding the underlying mechanics of the Chooser and its associated functions, this is also a good place to mention a few things I have 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 usually do not know which aspects of a resource's operation they can control, and when they do know, they frequently cannot remember which aspects of a network resource are controlled by using the Chooser, which aspects are controlled by using a Control Panel, and which aspects are controlled by menu choices within an application. The list-oriented nature of the Chooser also contributes to the confusion, particularly when network resources do not have names that have an immediate meaning for the user. Sometimes this can be solved by creating some simple reference documents for the user that tell them what options they can control and how to set those options. Another small adjustment that can make a big difference is programming in a few macros for the user to select services by keystrokes. For example, you might give the user the ability to switch between a LaserWriter and a Linotronics printer or the ability to mount a file server the Function Keys using a macro utility like CE Software's QuicKeys or UserLand's Frontier and placing a label on their keyboard identifying which Function Key performs which function.

Basic Chooser Operation

When the user opens the Chooser on a network with routers, a zone list is displayed 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.

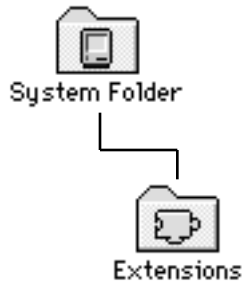


Figure 6-12. In System 7, Chooser Extensions are activated by placing them in the Extensions Folder, which is inside the System Folder.

When the user selects the icon for the type of service she wants to work with, the Chooser displays all of the available services of that type within the highlighted zone. An exception to this is when the user selects the LaserWriter Chooser Extension; then the Chooser displays all of the “LaserWriter” services in the zone of the previously chosen LaserWriter.

From the list of names that is displayed, the user selects the service that 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 will store the user’s choice in a resource in the extension file. In System 7, for example, the Chooser will store the name, type and zone name of the user’s preferred LaserWriter in the “PAPA” resource in the LaserWriter. 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 (LaserWriter, ImageWriter, LabelWriter, etc.) is stored in the System file.

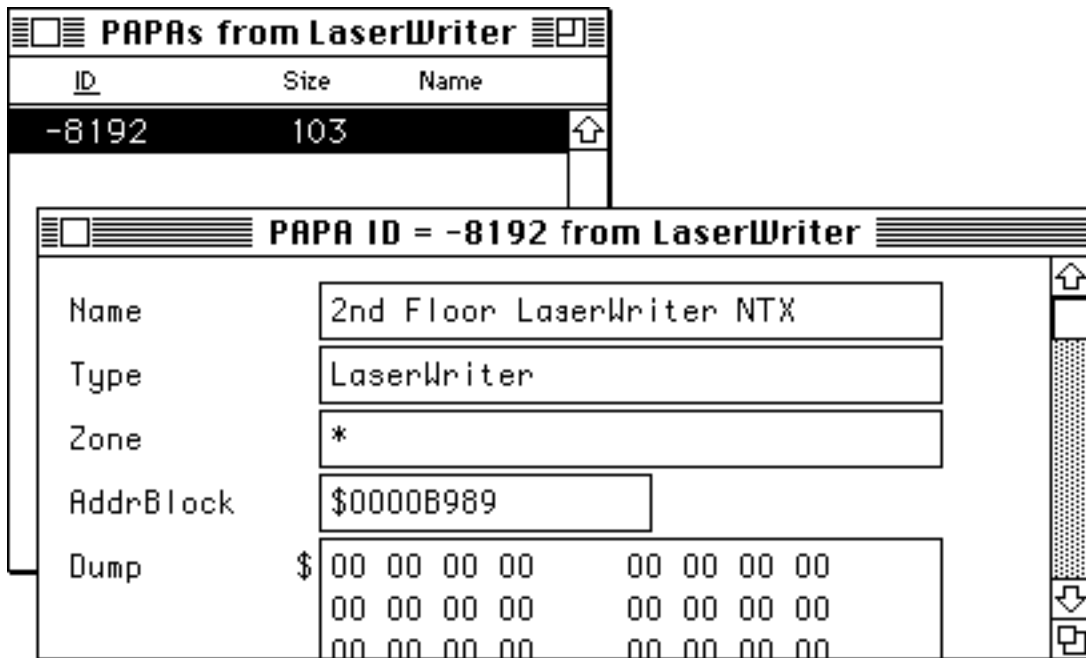


Figure 6-13. The PAPA resource in the LaserWriter Chooser Extension contains the name of the LaserWriter chosen by the user.

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 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 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 are available.

In System 7, the system will create an AppleShare Prep file if the user does not already have one, which is a great way to cure the problem created when the AppleShare Prep file is corrupted or contains undesirable values. You can just delete the old file and let the system create a new one. This does not 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

The Chooser must ask a router for a list of zones when the Chooser is opened in order to be able to display a zone list. On a network without routers, the Chooser window does not have a zone list. The Chooser knows whether or not there is a router on the network by examining a special memory location called the "RTMP Stub" (Routing Table Maintenance Protocol). 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 on 10 second intervals. Although non-routing devices ignore most of the information in these packets, they store the network and node address of the router that sent the RTMP packet in their RTMP Stub. On a network with more than one router, the identity of the router stored in the 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.

TROUBLESHOOTING MACINTOSH NETWORKS

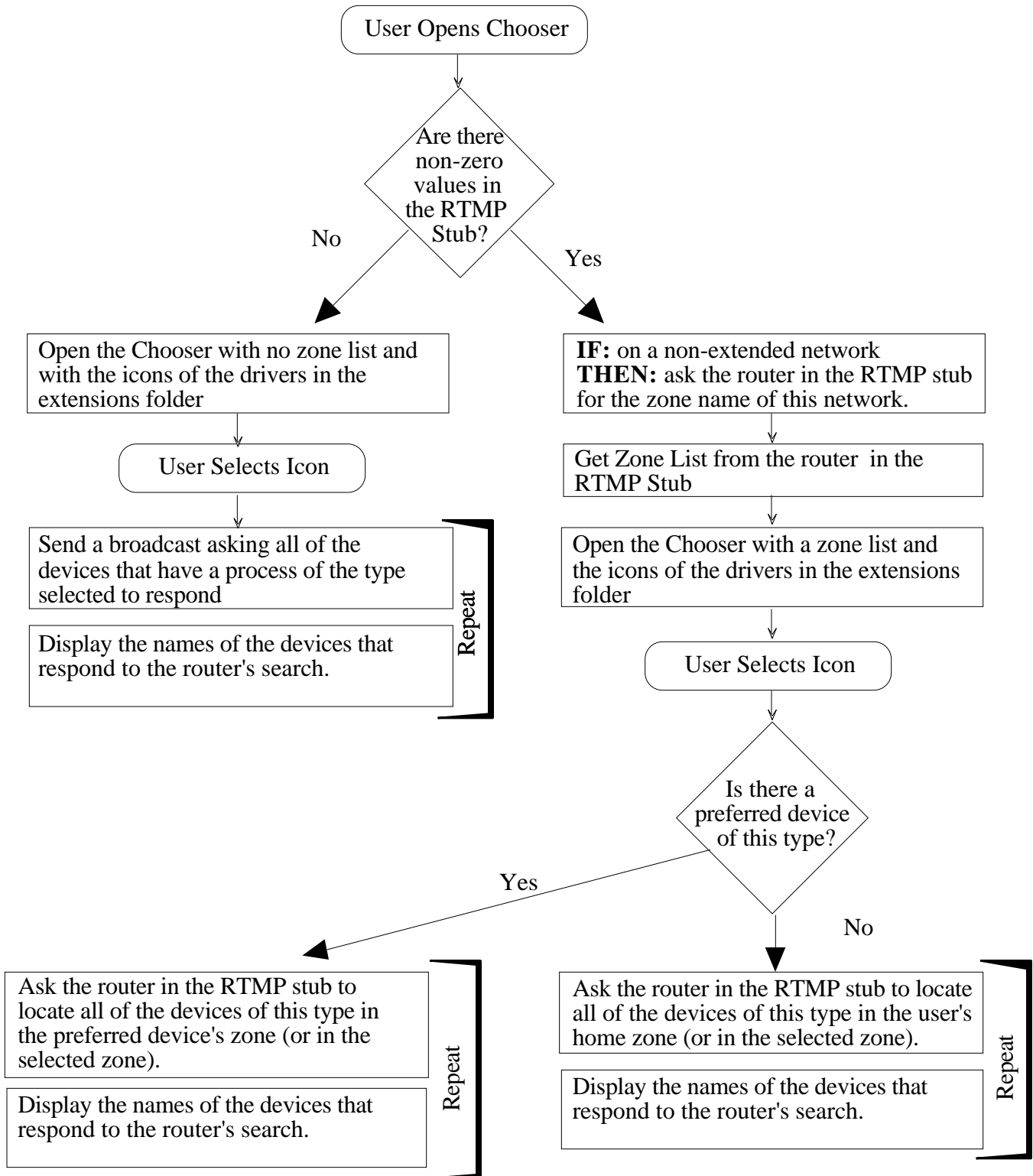


Figure 6-14. Chooser Process Flow

If there is no router 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 that matches the query will respond with an NBP LkUp Reply packet, which supplies the Internet Socket Address (ISA) of the process as well as its complete service name. The “Object Name” portion from 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 as described below.

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). This is shown in the figure below.

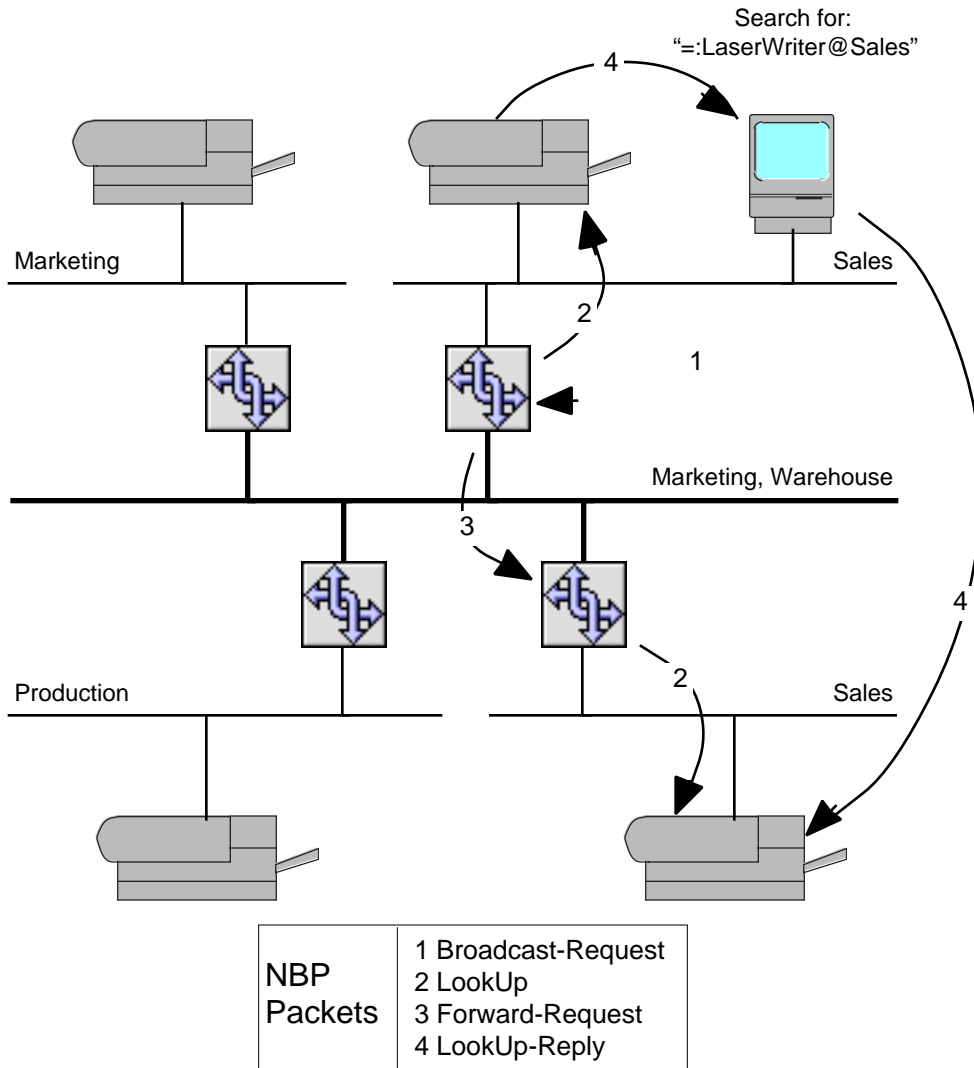


Figure 6-15. All four kinds of NBP packets may be required for a zone-wide search for LaserWriters.

The Chooser will send an NBP Broadcast-Request (Br-Req) packet to the router that lists the service type it is looking for and the zone it wants to search. The router will then send an NBP LookUp Packet to its

TROUBLESHOOTING MACINTOSH NETWORKS

network, asking all devices that have a service registered with that “type” to respond. The Internet Socket Address (ISA) of the Chooser that is 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 data base 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 is stated 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” and “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.

If the RTMP stub does contain a router’s address, then that router is asked to supply a zone list. When the user selects an icon, a router must also be used to search for services because the Chooser wants to find all of the services within a zone. Also, since the Chooser searches for and displays a list of all devices of a certain type within a zone, a network router must necessarily be involved in the search because Macs do not know the relationship between zones and networks. 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.

Named Services

An important concept necessary to understanding the Chooser is the concept of a “named service”. 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 be contacted by another process that does not know its numeric address. When the contact is made, the two processes exchange their ISA’s and the names are no longer needed.

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 we would say that the service Fred’s LaserWriter:LaserWriter@Sales is bound to ISA 12.185.128. Some services also make use of an enumerator, which is necessary when there is more than one service name bound to a particular socket. For example, in the excerpt from CheckNet below, Timbuktu shows 2 service names, each registered to the same ISA of 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 is the first 6 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 only registered if Gwynn (the user) is currently allowing guest access to her Mac. The Timbuktu Chooser-like utility (shown in Figure 6-16) looks for a service type of “Timbuktu Host”, and displays the object names of the processes that reply.

Name	Type	Zone	Net	Node	Skt	Enum
000321	Timbuktu Serial	Sales	4000	22	246	1
Gwynn	Timbuktu Host	Sales	4000	22	246	2

Figure 6-16. Software processes can register more than one name on a socket. This is an excerpt from a display using Farallon’s CheckNET.

There is something misleading in the figure above. Timbuktu is not actually registered in the “Sales” zone. The reason that CheckNET displays the zone name “Sales” in this list, despite the fact that Timbuktu is registered with a zone name of “*” is that CheckNET knew that it was searching the “Sales” zone when the reply was received.

Although you cannot tell from the figure or from CheckNET, network 4000 is a LocalTalk network. As a rule, processes on a LocalTalk network, Timbuktu included, do not register themselves with a specific zone name, but rather register themselves with the zone name of “*”, which means “this zone”. LocalTalk devices generally do not know what zone they are in. At the time the Chooser is opened, the Mac asks the router which zone it is in so that 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.

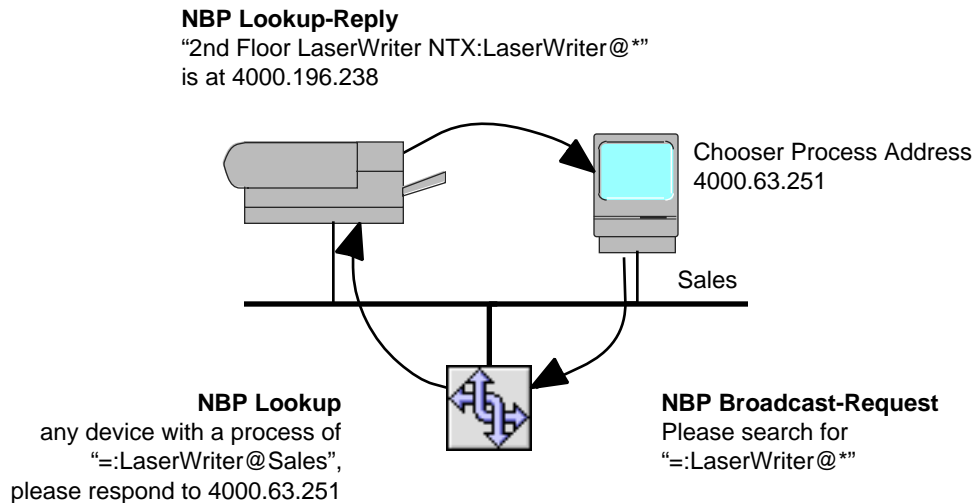


Figure 6-17. A Mac's Chooser searches its own zone for LaserWriters. Although the LaserWriter is registered in the “*” zone, the Chooser shows it as being in the “Sales” zone.

When a LocalTalk Mac’s Chooser searches its own zone for LaserWriters, 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

This small detail can have rather large implications, depending on the situation, and is often important to keep in mind when troubleshooting, especially when considering NBP searches like the ones the Chooser performs. Let me give one example that will stand for many.

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 offline for 10 minutes times the number of hops away to the farthest router. This gives all of the routers plenty of time to “forget” about the network, so that when the router comes back online, all of the routers will think there is a new network and inquire what zone name it has. If you don’t leave the router offline long enough, some of the routers may not notice that it went offline and will continue to use the old zone name.

In the internet below, the router in the upper right network is the one whose zone name was changed. The farthest router is only one hop away so the rule of thumb says that you should wait 10 minutes before activating the router again. In the example internet below, 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 of the routers are shown in the figure.

TROUBLESHOOTING MACINTOSH NETWORKS

If you were troubleshooting printing problems on this internet, which networks do you think that users 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 that users would be able to print to this LaserWriter from the other LocalTalk networks as well. The key is that the LaserWriter registers its name with the "*" zone.

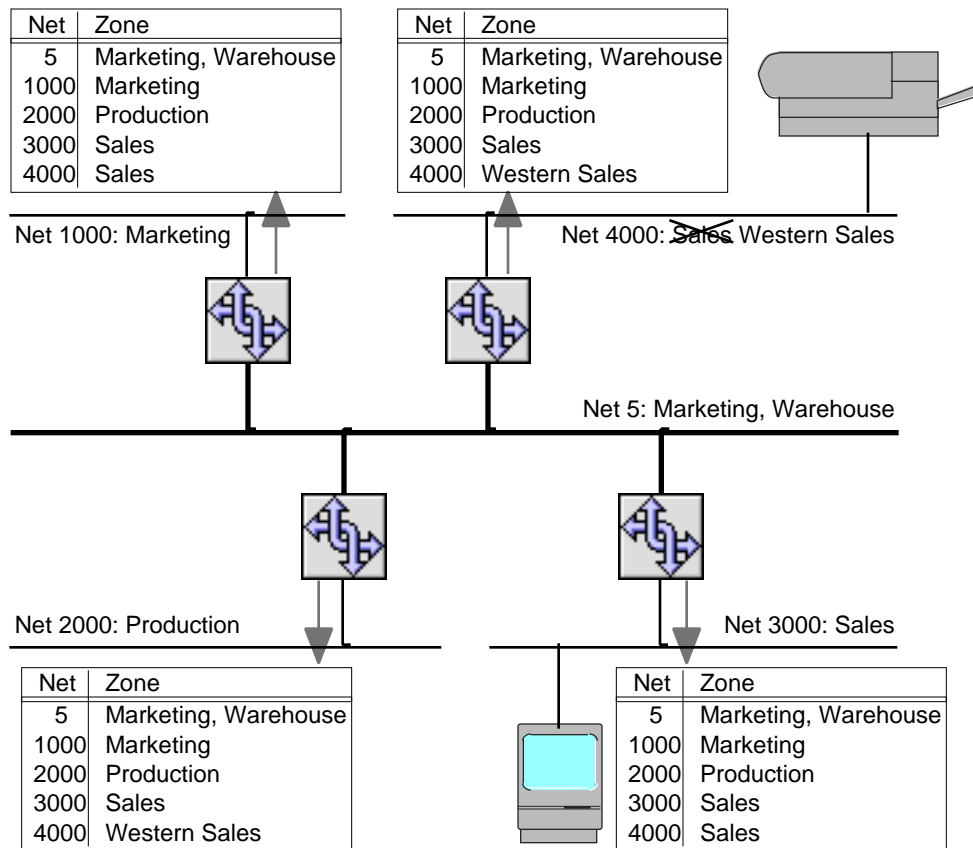


Figure 6-18. A change in the zone name of one of the routers was not noticed by one of the other routers on the internet.

If your Mac were attached to network 5, however, you would have only sporadic success printing to this LaserWriter. When you opened your Chooser, you would have a 50-50 chance of seeing Western Sales in the zone list because 2 of the routers know of that zone and 2 routers have not heard of it. When you select the LaserWriter icon, you will again use one of the routers to perform the NBP search, and will again have a 50-50 chance of getting a router that knows about Western Sales. Whether or not your Chooser can locate the LaserWriter will depend on whether the router that you used for the zone list and the router that you used for the search agreed on whether Western Sales existed. If they agreed, regardless of whether they both thought it did exist or they both thought they did not exist, your Chooser would locate that LaserWriter and write its full service name into the PAPA resource in the LaserWriter Chooser Extension. The zone name however, could be either Sales or Western Sales depending on which routers you used. Later, when you printed, you would again have the same 50-50 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 steadily show the same names as each search should yield the same results. Sometimes, however, this is not the case, and the list will change while you continue to 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 only happens in System 6 and it means that you have too many services of this type in the zone. The System 6 Chooser can hold just slightly less than 500 characters in the name list and a maximum of 16 names. When your Chooser receives replies in excess of these limits, the result is this rather entertaining symptom. There are 2 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 typically happens when you have wiring problems. This symptom implies that the NBP search is not 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, the most common symptom is bad wiring. To check this out, use the Echo Test or the Progressive Echo Test. A less common reason for this symptom is one of the many relatives of the problem mentioned above—the routers are in disagreement. To check this out, use Neon Software’s RouterCheck or one of the methods mentioned in Section 8.3, “Router Troubleshooting Technique”.

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’s 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. This is illustrated in the figure below. 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 foreground or background printing was chosen.

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 printer driver remain engaged in the process until printing is complete as all of these events occur in a locked series somewhat similar to a “bucket brigade”.

If the user has 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 this figure, and in the rest of this process description, we’ll assume that the user selected the LaserWriter printer driver. The choice of printer driver, 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.

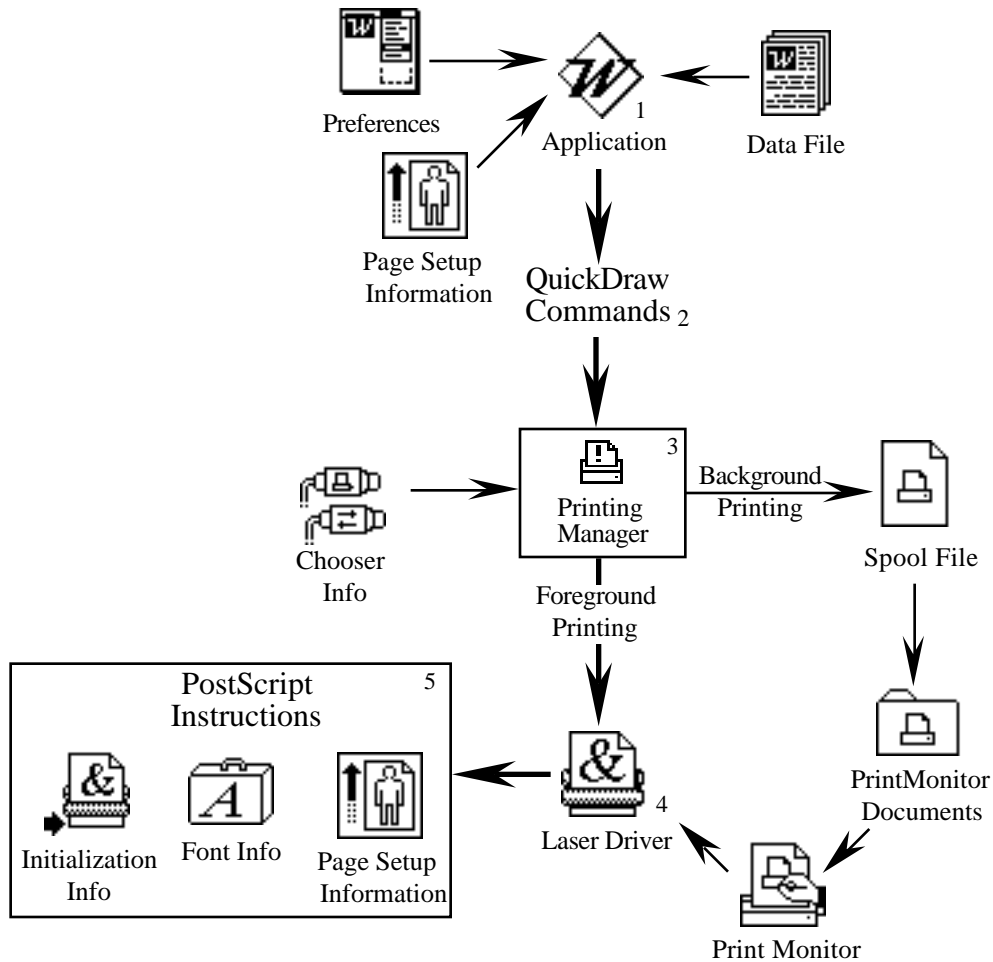


Figure 6-19. The Printing Manager accepts QuickDraw commands from the application and forwards them according to the user's Chooser selections.

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, etc.)
4. Describe the printing job (page characteristics, number of copies, paper tray. etc.)
5. Send the PostScript instructions for the printed output
6. Close the connection with the LaserWriter

Locating the LaserWriter by Name

Because the LaserWriter Driver stores the service name of the LaserWriter (name, type and zone) of the user's chosen LaserWriter, 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 will broadcast a Name Binding Protocol (NBP) LookUp packet, querying for the LaserWriter's service name, an example of which might be "Charlie:LaserWriter@Sales"¹. In a network that does

contain a router, the Mac will ask the router to locate the printer. The Mac will send an NBP Broadcast-Request packet to the router and the router will convert that Broadcast-Request into a series of LookUps—one LookUp will be broadcast to every network that is part of the zone. In either case, “Charlie” will send an NBP Reply packet directly back to the Mac that is initiating the search. The Reply packet contains the Internet Socket Address of the LaserWriter’s Session Listening Socket, the socket that is used to make initial contact with the LaserWriter. At the time that the LaserWriter accepts the user’s request to print, the LaserWriter will request 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” does not answer after a length of time, the LaserWriter Driver will display the alert shown below. In background printing, 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.

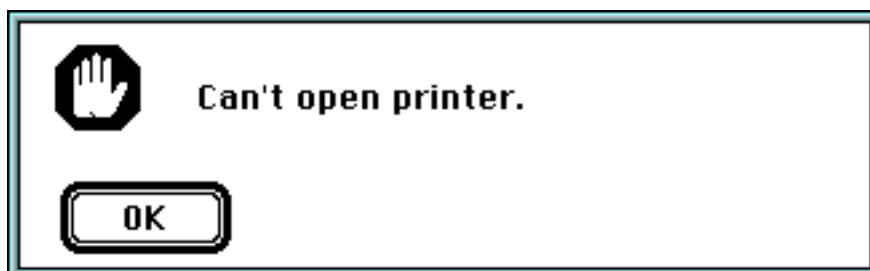


Figure 6-20. If Name Binding Protocol cannot locate the user’s chosen LaserWriter, the LaserWriter Driver will display this Alert.

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 is not subject to random changes. The name of the LaserWriter will only change if someone deliberately changes it or when there is more than one device trying to use exactly the same name, as mentioned in Section 7.1.4.

Establishing the LaserWriter Connection

Once the Mac has learned the identity of LaserWriter’s Session Listening Socket, it will attempt to establish a Printer Access Protocol (PAP) session with the LaserWriter to begin the printing process. To accomplish this, it will send 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 will respond to the Open Connection Request with an Open Connection Reply. If the LaserWriter is idle, the OpenConn Reply will contain an error code of 0, which indicates that the Mac should begin the connection. If The LaserWriter is servicing another user, it will return an error code of 65,535 and a status string that gives the user some information about why the LaserWriter is currently unavailable. This is the message that will be shown in the alert box on the user’s screen. The status string will include information such as the name of the current user, the name of the document, application, etc. The exact format and the contents of the string are determined by the manufacturer. Other LaserWriter activities and states that might be reported are the presence of a paper jam, a missing paper tray, or a report that the LaserWriter is currently re-initializing.

The Open Connection Request will be re-issued by the LD every 2 seconds. When the LaserWriter finishes printing a job, it will listen for approximately 4 seconds (twice the request interval) and accept the user who reports the longest waiting time as the next customer.

²For more on Name Binding Protocol, please refer to "Functions-Name Services"

Initializing the LaserWriter Connection

At the time the session is initiated, the two devices will 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, is not 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, will 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 that "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 the LaserWriter may fill up its memory with instruction data and then 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 that the network still works.

In addition to the tickle timer, there is also a connection timer. The connection timer is 2 minutes long, and is reset to "0" any time data or tickles are received from the other end. When either the Mac's or the LaserWriter's connection timer reaches its 2 minute limit, the device will shut 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 or shutdown or the loss of the network connection. This will allow the device to stay in service and recover the resources (memory, CPU time, etc.) being consumed by the connection.

The tickle timer is 1 minute long, half the duration of the connection timer. Tickle packets can verify the integrity of the connection in the absence of any printing communication. When a device's tickle timer expires, the device will send (and expect to receive) a tickle packet to verify the connection. If the tickle is not received, the device will begin searching the network for its connection partner and continue the search until the connection timer expires.

PAP is a session layer protocol, which means that its job is to establish and maintain the relationship (or "session") between 2 devices and manage the resources (memory, CPU time, etc.) used by the devices during the connection. LaserWriters have a special characteristic that PAP is especially designed for: 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 in it that prevents the workstation from sending data to the LaserWriter when the LaserWriter cannot receive it.

Sending Data in a Read-Driven Connection

PAP handles this special need by having a communication structure that is "read-driven". In a PAP connection, each device must signal the other when it is ready to receive by sending the other 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 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 requestor to the responder that the transaction is complete.

In a PAP session, however, the Transaction Request is used to carry the Send Data invitation. Questions and commands, as well as answers and command replies, are carried in the Response packets. Each device will have 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 an answer to a question.

After both ends have given the PAP Send Data invitation, the Mac “makes the next move” and asks the LaserWriter a question (detailed below). That uses up the Mac’s invitation to send. The LaserWriter will send the Release for the transaction. The LaserWriter will use 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.

Below are two figures illustrating the difference between PAP sessions and more typical ATP sessions. The first figure shows a more typical ATP transaction model, where one end 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 in the Transaction Response(s).

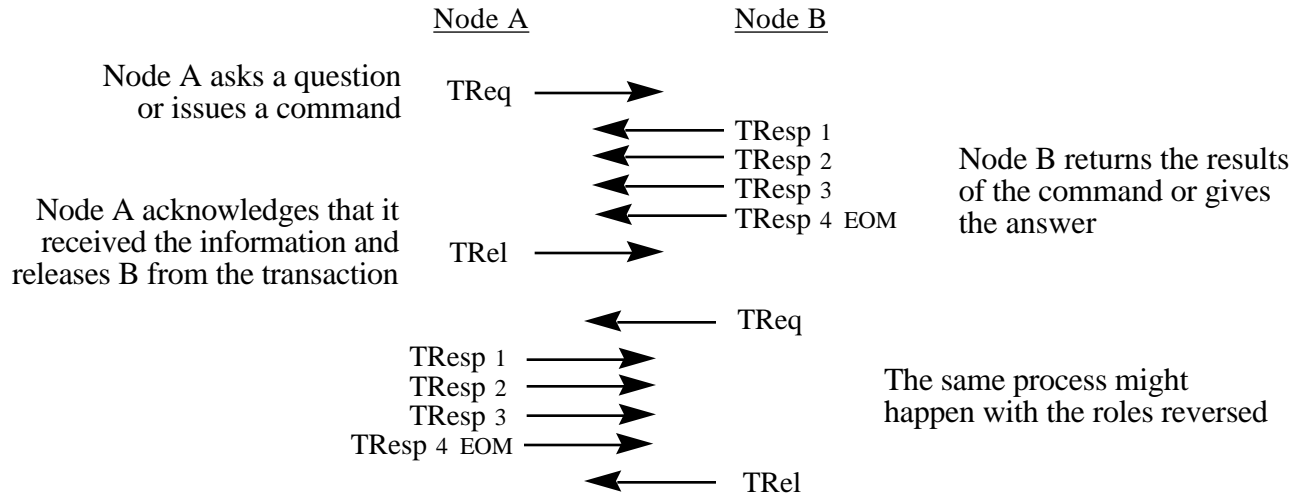


Figure 6-21. A typical client-server transaction model for AppleTalk Transaction Protocol (ATP). The 2 transactions shown above contain 2 question-answer pairs.

Because PAP is read-driven and each side must invite the other to send data, ATP works somewhat differently.

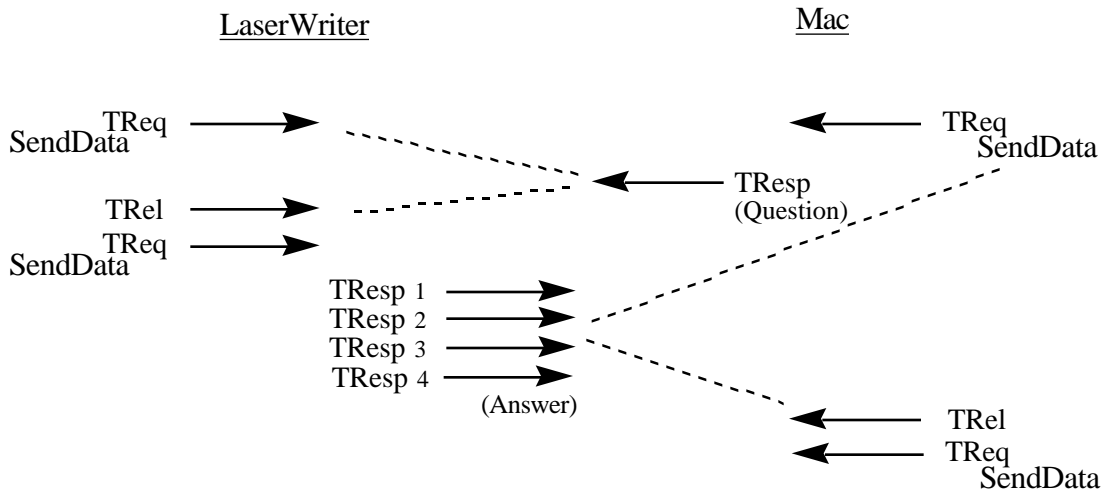


Figure 6-22. PAP sessions have 2 interleaved series of transactions, 1 set initiated by the Mac and the other by the LaserWriter. Each transaction Request contains a PAP SendData command. Transaction Responses may contain either a question or an answer. The dotted lines link the 3 components of a transaction—Request, Response and Release.

TROUBLESHOOTING MACINTOSH NETWORKS

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 is using the same version of PostScript definitions that the Mac is using. In most networks, the answer to this question is almost always “Yes” because network managers usually prefer to have one standard version of printer driver loaded on all stations. The events that occur when the answer is something other than “Yes” are described below.

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 that are not in the LaserWriter’s memory, then 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 includes data such as the paper size and margin information and any PostScript definitions that the LaserWriter may need in addition to the definitions that it already knows. 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 is 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*):

<u>Response</u>	<u>Meaning</u>	<u>LaserWriter Driver:</u>
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 wants to re-initialize the printer.

If the response is “2” the Mac closes the connection and the user is asked whether he wants to re-initialize the LaserWriter. If the user indicates that he wants to re-initialize the LaserWriter, the LaserWriter Driver will establish a new connection and perform this function. When the re-initialization is complete, the Mac will again break the connection, re-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 will compare the LaserWriter’s font list to the fonts required for the document, so that 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 will respond by sending the Mac a list of fonts that it has in memory. On LaserWriters with an attached hard disk, the fonts on the hard disk are also reported.

As described in the overview, this is the second stage of the normal print job. If you are trying to locate the beginning and end of these stages in the packet trace of a printing session, there are certain text patterns that you can search for. At the beginning of each stage, you will find the text string “%!PS-Adobe-2.0” and at the end of each stage you will 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 the transactions become very regular, or at least they should be if all is going well. For troubleshooting purposes, there are several things to watch for. In the figure below, I’ll point some of these out.

TROUBLESHOOTING MACINTOSH NETWORKS

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
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
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 6-23. In a print session, the transactions should flow smoothly and regularly in the third stage of sending the PostScript instructions.

This is a small portion of 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). If you capture a session on Ethernet or a print session on LocalTalk that is 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 trace, I'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), notice how the transaction response from the Mac to the LaserWriter includes 8 packets. In this trace, which has no lost packets, the transactions are moving along in regular batches of 8 responses. If packets are getting lost, you'll see retransmissions—a packet of 530 bytes sitting all by itself surrounded by smaller packets. If you see less than the usual number of packets in the transaction, it may be that the packet was sent, but that the protocol analyzer, for some reason, did not capture it. The other possibility is that the sending station did not elect to use all of the packets in the transaction. All of the packets in the transaction response are numbered, and if the sender elects not to use, say, packet #8, it will send an End of Message notifier in packet #7. This is not the end of the job, it just means that the sender did not want to use all of the allotted packets. You can quickly check the sequence to find out why there are missing packets.

The LaserWriter decides how many packets to include in a single transaction (maximum of 8) based on how much memory it has free when the session begins, and 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 4 packets for their Flow Quantum instead of 8, but the pattern will be the same.

Within one transaction, the Mac is supplying the responses at approximately the rate of 1 packet per 10 milliseconds. This is a reasonable rate for a Macintosh portable. A large LocalTalk packet like this takes a little over 2 milliseconds to transmit.

At (2), the LaserWriter releases the Mac from the previous transaction and then issues the SendData command that allows the Mac to send the next batch of 8 packets. One thing to look for here is the time gap between the Release and the next Request. If there is a gap, it is because the LaserWriter is busy processing the information it already has and its buffer is full. The other gap to look at is the time between the SendData command (in the transaction Request) and the first packet in the following transaction Response. In the trace above, 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 gaps can help you determine which end—the Mac or the LaserWriter—is causing the slowness. If it is the LaserWriter, perhaps the graphics that you’re sending it are particularly complex. The same image can be rendered in different ways, and some of those ways might be more or less efficient for the LaserWriter to draw. You may also try sending the same image from a similar application made by a different manufacturer or perhaps you could try to restructure the graphic to be less complex. You might also try reducing the number of fonts used or using PostScript fonts instead of TrueType fonts. Other things to check are the Page Setup and Print Options settings. Checking the option for Unlimited Downloadable fonts can slow down your printing considerably because each time you change fonts, a new font definition is sent to the LaserWriter.

If the Mac is the slow end, what else is going on in the Mac to cause the slow response? The response time depends on system variables like 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, of course, the Mac is doing something else in the foreground, so there is less processing power available to pay attention to the printing process. The gap in the trace above, which is only 3 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 6-24. A portion of the trace later in the print session.

In the figure above, 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 are the messages you see on your screen—preparing data, 9 pages to print, etc.

Notice that this status report was not specifically requested by the Mac, it was sent spontaneously by the LaserWriter. If the Mac had 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 will simultaneously have two outstanding transactions with the LaserWriter, a SendData invitation and a SendStatus Request. Earlier I had mentioned 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 will not be able to distinguish between status messages sent spontaneously and status messages sent due to a SendStatus Request.

TROUBLESHOOTING MACINTOSH NETWORKS

In this case, the LaserWriter sent a spontaneous status message using an outstanding SendData Request that had been issued many packets earlier. A Mac will ask for a status report if it is unsure of what's happening at the LaserWriter. A typical example is when the LaserWriter is busy and it has not 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 will contain a small packet (less than 530 bytes) to conclude the instructions. The Mac will immediately close the connection even though the print job has not finished processing in the LaserWriter. If there is an unrecoverable error at the end of the print job, and the LaserWriter discovers it after the connection is closed, the Mac will not be notified of the error. Many LaserWriters have the ability to terminate a job on their own in such a case and the only way that you'll know that there was an error is that some of your output will not be in the tray.

Troubleshooting Techniques

When you go to the doctor's office for a check-up, there are usually a few quick tests that her assistant will give to you while you're waiting for the doctor herself. The assistant might check your weight, your blood pressure, heartbeat rate, and reflexes, then draw blood and urine samples for routine analysis and ask you a few general questions like, "How have you been feeling lately? Have you had any headaches or felt tired?" All of the results of these checks and questions get written down on your chart and given to the doctor when she arrives. She'll look over the chart, checking for results that are unusual in general or just unusual for you. It's a way for her to begin her investigation of your health. These preliminary checks were chosen 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 that she's found before. It's not only the values of the numbers she's looking at, but how they compare with previously collected data.

Performing a Network Health Scan

The purpose of quick network health scan is to give the network manager a general answer to the 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 indeed there is one.

Instead of checking blood pressure as a doctor does, you might check the network utilization level. For a blood sample, you might take a look at the nature of the traffic and find out what services people are using. Instead of measuring the heartbeat rate, you might check the page count on the LaserWriters, or the message count on the mail servers. For the network equivalent of reflexes, you can launch a couple of benchmark jobs to see what kind of response times people are getting. Instead of analyzing a urine sample, you can 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. "Any server crashes lately?" "Have you experienced any unusually slow service?"

Sometimes, these checks do 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 2 or 3 times a year, anyway, because the information that you gather can help you with other aspects of network administration besides troubleshooting, such as 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 rise 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 is not going up or it's dropping, you might try to figure out why. It might be just a normal statistical fluctuation, but it might also be a trend that you may want to 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 re-designed 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 of their personal data files on the server. After examining the traffic in more detail, you may also decide to re-examine the placement of your routers and bridges to make sure that your network's configuration is at an optimum.

TROUBLESHOOTING MACINTOSH NETWORKS

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 I suggest in the next few sections can be automated. My 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 using MediaTracks unattended, because the file size 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 is heavily dependent on keeping records of previous testing. My best advice here is to keep your records simple so that you will 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 of your doctor if she did not keep records of your previous visits.

I find that a loose-leaf notebook works best for keeping numerical data, with each page devoted to a single measurement. For graphical data, like a NetStats screen shot, a collection of printed records in a notebook works very well. Later, I may transfer the numerical data to a computer for analysis, but the master data is always the paper records.

Each test that you develop for your own health scan should have its own test 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 there be noticeable difference during subsequent health scans. These errant values may be caused by subtle configuration changes later either in the network or the systems involved. This is impossible to do perfectly because there are so many factors that can have an impact on performance. I'm only suggesting that you specify as many of the major factors as is reasonable. 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 be like the one shown below.

Benchmark Test Definition

- **Task Definition**
Upload the file Aldus SuperPaint 3.0 from Dawn Vliem's Macintosh to the Safety Server
File Size is 1,032 KBytes. 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 45 MB 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 is on the Admin Server in "Managers:Benchmarks:Test Configs:File Transfer:Dawn".
The Safety Server is a Mac IIfx with 20MB RAM, a 210 MB 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
Background Network Utilization: Ethernet <2%, LocalTalk <10%

Check the Network Utilization

The Media Access Control method 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 long DEC or UNIX history are usually Ethernet and 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. A typical user of this kind may download the data file of a drawing he’s been working on from the file server and then not need the network for 10 or 20 minutes. Then he’ll retrieve his mail messages and stay off the network for a while, working on his drawing or chatting on the phone. Later, he’ll finish his changes to the drawing, save the file back on the file server and send the file to the printer on his way to lunch. This is the kind of traffic for which both Ethernet and LocalTalk are well suited. 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, and is likely to be very 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%, during the busiest minute of the day, 26%, and during the busiest second of the day, network utilization averaged 49%. This kind of information can be compared to previously captured data and can be compared against an ideal maximum that you have developed by experience. If you have not yet developed an ideal maximum for Ethernet, reasonable numbers to use are 15%, 40% and 75% respectively for the busiest hour, minute and second. If your network utilization is above this level, you should probably examine the traffic and re-examine your network design.

The reason why utilization of Ethernet should be kept 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. LocalTalk does not have this relationship between error rate and utilization. In LocalTalk, 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 it is more difficult for stations to gain access to the network. In LocalTalk, you might use 50% utilization (averaged over an hour) as the threshold for further investigation for re-examining your design.

Before you rush off to the drawing board, however, you should first try to ascertain that the level of traffic that you’re seeing is normal, and is caused by legitimate users and devices going about their normal daily tasks. This is not always the case, however, and you should also consider the possibility that there is a device on your network (or a specific group of devices) that is malfunctioning and creating an artificially high traffic load. One of the more well-documented (and feared) possibilities 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 have experienced AARP storms (most of these were caused by a particular router problem in 1987 and 1988) and ZIP storms (routers again, in 1991). Other 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 have to 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). In general, it either points to a bug in the software of the device or its inability to handle a particular kind of situation that it finds itself in. ZIP storms are described in more detail in Section 8.3, “Troubleshooting Router Configurations” and generally result from corrupt

TROUBLESHOOTING MACINTOSH NETWORKS

routing tables. The AARP storms of 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 just watch a moving graph of utilization vs. time. You can't save or print NetStats data except by doing a screen capture. NetStats' only setting is how sensitive you'd like the time scale to be and your choices impact the duration of the scale. On the most sensitive setting, 10 samples per second, this is around 42 seconds. On the least sensitive setting, 1 sample per second, you can see about 7 minutes of activity at one time.

One interesting way to use NetStats is to create a recurring QuickKey sequence (CE Software) 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 QuickKey sequence to take a screen capture every 7 minutes. For the screen capture, I use Flash-It, a shareware program (\$15) that I like because I can specify all of the options ahead of time and Flash-It can just do it's job without needing a user to respond to a dialog box. For example, when the Caps Lock key is down, it takes a screen shot of the active window only (NetStats, in this case). It can then save it to any one of several destinations that I specify beforehand and select by which QuickKey sequence I call. I typically specify a ScrapBook file as the destination for this procedure. One little tip—set the monitor for 2 colors. There's no advantage to having more colors for NetStats, and the screen shot sequence runs faster.

There's a bug in NetStats that you should know about, though, and it's one that may occasionally ruin your testing. Occasionally, NetStats will “peg”; it will indicate a network utilization of 100% that is clearly not an accurate measure of the traffic of the network. It does not usually recover from this error situation and if this happens during your automated testing, you may come back to a useless set of “results”.

For Ethernet, there are some very powerful tools in some of the more expensive Ethernet hubs on the market, 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 at the time of this writing are shipping, also appear as though they may be useful for traffic characterization, but only for Ethernet networks in their first release.

If you do spot irregularities in your data, or would like to read more about traffic analysis, there are more detailed suggestions for traffic analysis in “Analyzing Network Traffic”, Section 8.4.

Check the Service Activity

Several kinds of servers have ways of telling you how much they've been used, and these activity monitors may be a measure of your network's health, as well. 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 being 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 data your users' address books are and whether they need to update them. If you use a different package, such as Microsoft Mail or Word Perfect Office, there will probably be 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 of automating the reporting procedure.

For LaserWriters, an easy measure of usage is the page count, which is printed on the startup page, and can also be obtained through Apple's LaserWriter Utility 7.1. If you use AppleShare Print Server, or any other spooler that keeps a log of activity, you'll be able to see exactly how many pages were printed by which users at which times.

While AppleShare 3.0 still doesn't have a good set of statistics to show you how much the server has been used, one measure of the server's activity that you can determine is 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 using the "Choose..." criteria in the Selection Window. In the figure below are 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 will 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 Choose criteria to select only those files that have changed. Using this method on a server volume of 164 MB volume and 2704 files, Retrospect told me that there were 82 files that had changed in the last 2 weeks with a collective size of 5.6 MB. This represents about a 3.5% change in a 2 week period, a relatively light amount of activity.

This method of measurement has its weaknesses; an obvious one is that a file that changed 1000 times and a file that changed 1 time will appear the same to this measure. Still, this method has its merit for certain situations in the absence of a more direct method.



Figure 7-1. Using Dantz' Retrospect, you can find the number of files on your file server that have been modified in the last 2 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 to see how long they take. If you understand the range of values that a benchmark can have, an unusual value will flag your attention to the service that is ailing. In order for a benchmark to be useful, you'll need to be fairly familiar with the range of values it can have in normal operation. Start out by running the benchmark when no one else 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 large the difference between your scan value and your benchmark standard can be before you take notice and try to figure out what's happening to cause the difference.

If you've never developed a benchmark operation, now is a good time to develop your first one. Good benchmarks are real-world operations that are performed just 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 one, the variable you're interested in understanding. For our Network Health Scan, one variable that we're interested in understanding is 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, and 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.

TROUBLESHOOTING MACINTOSH NETWORKS

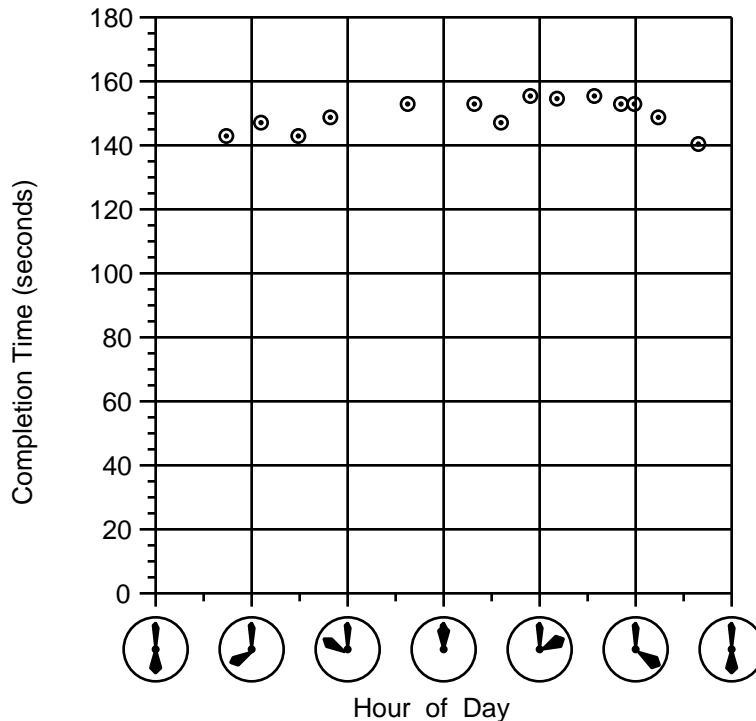


Figure 7-2. Run a benchmark operation several times during the day to get an idea of the range of values.

The figure above shows our benchmark file transfer results. The values ranged from 140 when the file server and network were unloaded to a maximum of 158 seconds in the mid-afternoon. One way of characterizing the data that I find useful is to express the range of variation as a percentage of the average value, which for this data is slightly over 10%. 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.

There are two factors that control the value of this measure—a) the variation in the number and kind of tasks that a server may be performing at the time of the test and b) 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 which 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 will see much more dramatic variations in the data. While a good portion of the data may fall within a well-behaved range, there may be individual values that vary greatly, as much as 100% 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 will 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 will 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 know what caused the delay.

In summary, a benchmark is most useful when the factors that affect its value are well understood. You should work with your benchmark a while to see how it varies with normal conditions—the number of users logged in, the number of users performing the same task concurrently, the network utilization, etc.,

so that when you take your benchmark measurement during a Network Health Scan, you will know what values are normal, what values indicate concurrent operations, what values are unusual, but still within the range of normal possibilities, and what values definitely indicate trouble.

Besides file transfer, here are some suggestions for benchmarks:

- The time to print a particular word processing file with graphics.
- The time for a particular user to be notified that he has mail after it is sent with sender and receiver on the same mail server.
- The time for a particular user to be notified that he has mail after it is sent with sender and receiver on different mail servers.
- The time to type the contents of a particular text file on a host computer.
- The time to receive the results of a particular search on a network data base.
- The time 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 encountered unusual conditions. These are also kept by some hubs and routers. A quick check of these on a regular basis can sometimes spot trouble conditions. 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 be helpful to spot potential problems, even if they are not specifically error logs. The Activity Log that Remote Access maintains, for example, may provide some clues to which users might need a little more training or who may be having equipment problems.

Activity Log		
11 Log Entries		
Date	User	Log Entry
Thu, Mar 5, 1992 10:05 PM	June	Dialing 3632020
Thu, Mar 5, 1992 9:58 PM	June	Dialing 3632020
* 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 0:00:22.
* 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 0:00:44.
* Thu, Mar 5, 1992 12:11 AM	June	Connection established at 9,600 bps.
* Thu, Mar 5, 1992 12:10 AM	--	Incoming call.

Figure 7-3. The Remote Access Activity Log.

Looking at the Remote Access log file above, 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. More information on the errors that hubs report is found in “Data Link Troubleshooting”.

Finding and Correcting Router Configuration Problems

With the advent of Phase 2, AppleTalk internetworking became more flexible than ever before. Network managers had many more possibilities in assigning network numbers and zone names. Some managers of large Phase 2 networks 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 discovered a very 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: I have been teaching people to diagnose router problems in my Advanced AppleTalk classes, and I have found it takes people a while to “get in the groove” of this kind of investigation. There are two conceptual hurdles that stand in their way. The first reason 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 are not used to thinking of routing and zone tables as components of a dynamic, ongoing process—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, I discussed how troubleshooting begins by comparing “what is” to “what should be”. There I said that 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 is another side to that coin, however. When a person has highly developed expectations, there is a very natural human tendency 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, they didn’t even know that it was a ship for 2 days! 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 are troubleshooting, you will see eccentricity; you will even be on the lookout for eccentricity—unusual behaviors, unfamiliar images, etc. When you see it, you cannot 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 the unusual image or strange behavior is the cause of the problem. The eccentricity that you notice is simply an effect of the error condition you’re trying to locate, and sometimes it is not 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 erroneous network and zone information gets introduced into an AppleTalk internet is by data entry errors. During the router configuration process, the network manager might misspell a zone name, forget to perform one of the configuration procedures (such as entering the zone list), forget to set one of the optional parameters, etc. He 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.

Another avenue for error introduction is when users other than network managers configure routers. Sometimes, a user who falls outside the jurisdiction of the network management staff, but is nonetheless connected to a common network, may purchase and install a router, perhaps in complete ignorance 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) is 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 desire 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 in which erroneous routing information can enter the network is 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 only happens 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. The ZIP Storm just illustrates the mechanism well.

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 did not foresee this condition and did not create a procedure to prevent catastrophe.

A network manager sets the ZIP Storm mechanism in motion when he adds a new router or a couple of extra networks or zones to an (already very large) internet. Some of the 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 that is 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 it is and also 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 of the 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 very quickly, the internet is completely flooded with ZIP and RTMP packets, all of the AppleTalk routers roll over and die, and your AppleTalk internet experiences catastrophic failure. (For you Monty Python fans, this is 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 your routers (each one of them) must be greater than the routing information requirements of the internet. The network manager may either a) only use routers that have a large enough memory space to hold the required network and zone information or b) keep the required network and zone information small

TROUBLESHOOTING MACINTOSH NETWORKS

enough so that it fits into the memory space of the router. This is typically accomplished by keeping the zone names very short so that less memory is used per zone name or by restricting the number of networks and zones. Also, some (not all) routers allocate memory for extended networks in an amount proportional to the size of the address range. In internets that use these routers, the use of large network address ranges is also avoided.

If you're on a large internet, and you're now worried that you may be on the brink of this disaster, you can verify your concern or allieviate 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 do not currently offer this ability, but the ability to check a router's free memory will be a standard feature of the AppleTalk SNMP router MIB.

Below, you can see 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 3 conditions, the GatorBox was configured for Phase 2 AppleTalk only—no additional routing or gateway services that would consume memory.

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).

A

B

C Memory Statistics...
Total Memory 941236 bytes
Allocated Memory 521756 bytes
Free Memory 419480 bytes

Figure 7-4. The memory statistics from a Cayman GatorBox CS for 3 different internet configurations.

Via these 3 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, etc.—depends not only on what the nature of the error is, but also 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. There are also some router conflicts that will 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. In my travels, I often ask the local network manager if I can look for router configuration problems, even if the network manager does not suspect a router problem. In better than half of these cases, I 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 role 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. One 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 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 multi-router 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. Also, most network services register the zone component of their service name as simply "*"—"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 will allow users to continue using services despite the fact that their router configurations are in conflict, but 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, there will be a certain length of time before the SNMP definitions are finalized and vendors have uniformly implemented them and worked out bugs. I expect that SNMP management of AppleTalk routers will be universally implemented and useful before mid-1993.

3. A latitude in how vendors interpret key points of the RTMP and ZIP specifications.

Two of these key points are a) how a router should age and "forget" network and zone information when the routers that are serving those routes are removed from service or fail, and b) how a router responds when its configuration information contradicts or conflicts with the configuration information of other routers on the network. This lack of uniformity makes troubleshooting more difficult because different manufacturers' routers often respond in very different ways to the same situation.

4. The way in which Apple has differentiated between seed routers, which have network and zone information explicitly configured by the network manager, and 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 and some seem to take the information from the last router they hear. Non-seed routers, once they have learned network and zone information, are indistinguishable from seed routers. You cannot 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 can provide seed information.

5. AppleTalk has a simplistic way of determining the best route to a given network.

In LocalTalk, and other Phase 1 networks, a non-routing node that has a packet to send to a distant network simply sends the packet to the last router it received a packet from. That router then consults its routing tables to determine the path. The router's own routing algorithm to determine the best route is

TROUBLESHOOTING MACINTOSH NETWORKS

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 there have been some attempts 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 does not 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 and
- Be difficult to correct when the error is detected.

When the routers are in conflict with respect to their routing and zone information, there are characteristic patterns and symptoms that deviate from normal. As previously mentioned, 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 can be 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.

Short of packet analysis, there are other network management tools that can help you spot router problems. These tools, however, are (so far) not always effective. The reason is that these tools are limited to the situations that their programmers conceived when they wrote the program. Error conditions in routers are, as described above, unpredictable in both cause and effect. These tools can only spot the certain types of problems that their programmers designed into the software. These tools, however, can be effective at noticing that there is 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). I have used RouterCheck many times in many situations. When there is a problem, RouterCheck is generally reliable in its ability to notice that there is some problem, but generally ineffective at determining exactly what the problem is. In addition, RouterCheck, even when set for very large memory sizes, will crash when analyzing large internets.

When you first open RouterCheck, it searches for the routers in your internet and displays a list of the routers that it finds. After these have been located, 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 2 routines asks each of the routers that the RouterCheck initially found for a zone list and automatically compares the zone lists, noting any anomalies. “Zone Locator” makes a report that includes the zone list for each router so that the network manager can compare the lists himself.

RouterCheck also does not check for network number conflicts. For example, if there are routers on an extended network that disagree on the identity of the network range. RouterCheck’s abilities will grow, of course, and Neon has already shown an SNMP-equipped version of the product. Despite the shortcomings of the current version, it is still useful and will probably evolve into a terrific tool.

Here’s an example of using RouterCheck. The error situation is an internet with 5 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 4 names, “Ethernet, Ether 1, Ether 2, Ether 3”. 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.

Below is 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 5 routers, and 10 entries in the list, but there are not 2 entries per router. Three of the routers (GatorBox, FastPath 4 and FastPath 5) have 2 ports each, one LocalTalk port and one Ethernet port. You can see each of these 2-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 is not noted in the list.

Routers 1.181, 25000.65, 26000.75 are all ports on the same router, and you can tell because they all have the same name, the serial number for the APT ComTalk. The Webster router, however, is only listed once for its Ethernet port. It also has 4 LocalTalk ports, connected to networks 32000,32001, 32002 and 32003.

The reason that this list is incomplete is that RouterCheck uses an NBP LookUp for router ports on the LocalTalk side. If the router registers an NBP name on the LocalTalk side, RouterCheck will list it. If not, the port does not appear in the list.

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 7-5. RouterCheck’s Router List

When you run the RouterCheck’s Missing Zones and Conflicting Zones routines, the Session Log lists the following anomalies:

- 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 220: Ether 2

Figure 7-6. RouterCheck’s Session log has determined that there is a zone name problem.

You can tell from the Session Log that there is 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 says to reconfigure or restart every router that shows an anomaly, but that seems like a

TROUBLESHOOTING MACINTOSH NETWORKS

lot of work, especially since I 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.

The best thing to do at this point is to 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 ability. For this particular error, however, RouterCheck's Zone Locator will only be useful if you first export the Zone Locator report to a text file and view it with a word processor. The reason 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 is not 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 be possible to find it from one of RouterCheck's windows.

In the figure below is the portion of the Zone Locator Report (in Microsoft Word) that shows the error. I've changed the text font to Courier, which is a fixed-space font. Fixed space fonts are more useful when looking for spelling errors like this one. Another thing that you might notice is that the paragraph marks are left on—another visual aid.

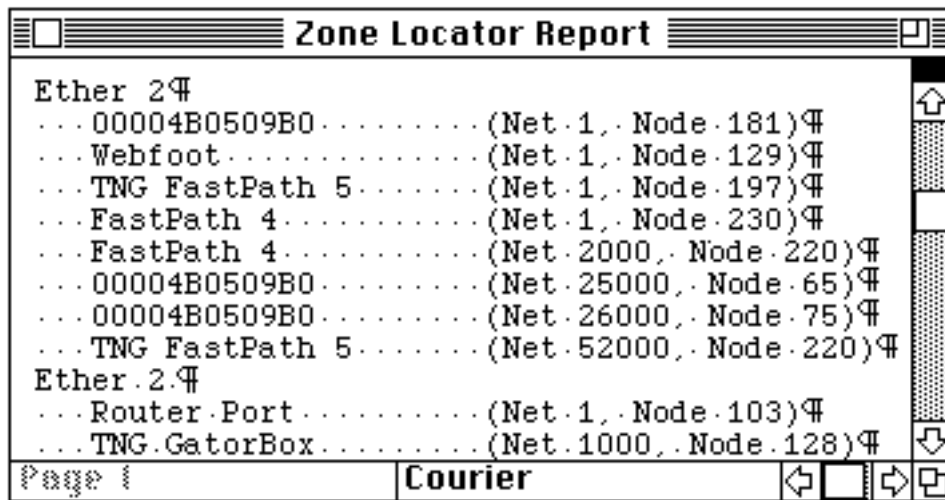


Figure 7-7. Using the Courier font and showing the spaces and paragraph marks, you can see that 8 router ports report "Ether 2" and 2 of the routers report the zone name "Ether 2".

As mentioned previously, the two routers listed as having the extra space are actually just different ports on the same router.

One special aspect of the above problem is that the symptoms are very subtle. The Chooser or Network Control Panel would not show a problem, because neither of those devices would show a problem because they do not indicate spaces. Also, none of the routers have both "Ether 2" and "Ether 2 ", so the zone name will not appear twice in anyone's Chooser or Control Panel. There will, however be symptoms, but only for users or services that want to have "Ether 2" for their home zone. These users would occasionally get an error message during bootup saying that their home zone was unavailable and they were being placed in the default zone. If the user next used the Control Panel to re-select Ether 2, they would be able to successfully do that most of the time. Their success would depend on which router was chosen to supply zone information, so their results would be inconsistent. All other things being equal, they would be successful 80% of the time, since 80% of the routers (4 out of 5) have the right information.

A class of network management tools that will sometimes notice router configuration problems are tools that build maps of the internet such as TechWorks' GraceLAN or Farallon's NetAtlas. Although this is 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—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 often indicates an unintended loop in the network structure.

It should be noted that even if you see an anomaly in NetAtlas, it does not necessarily mean that you have a router problem. There are many legitimate router configurations that will be represented strangely by NetAtlas. In some instances, NetAtlas's inability to accurately portray the network is related to weaknesses in the design of NetAtlas. Other harmless anomalies in the NetAtlas map can be attributed to the lack of a standard, robust network management protocol mentioned earlier. In lieu of this kind of a standard management protocol, NetAtlas must make inferences from the information that is available. Because different devices behave differently, particularly routers, NetAtlas will sometimes make an incorrect inference, which results in an anomaly on the map it draws.

As mentioned previously, packet analysis will be 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 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 there 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 only captures RTMP Response Packets. RTMP Response Packets are only sent by routers and every router should be sending one transmission to every network that it's connected to every 10 seconds. By examining the number of nodes sending packets that pass through this filter, you can determine how many routers are on your network.

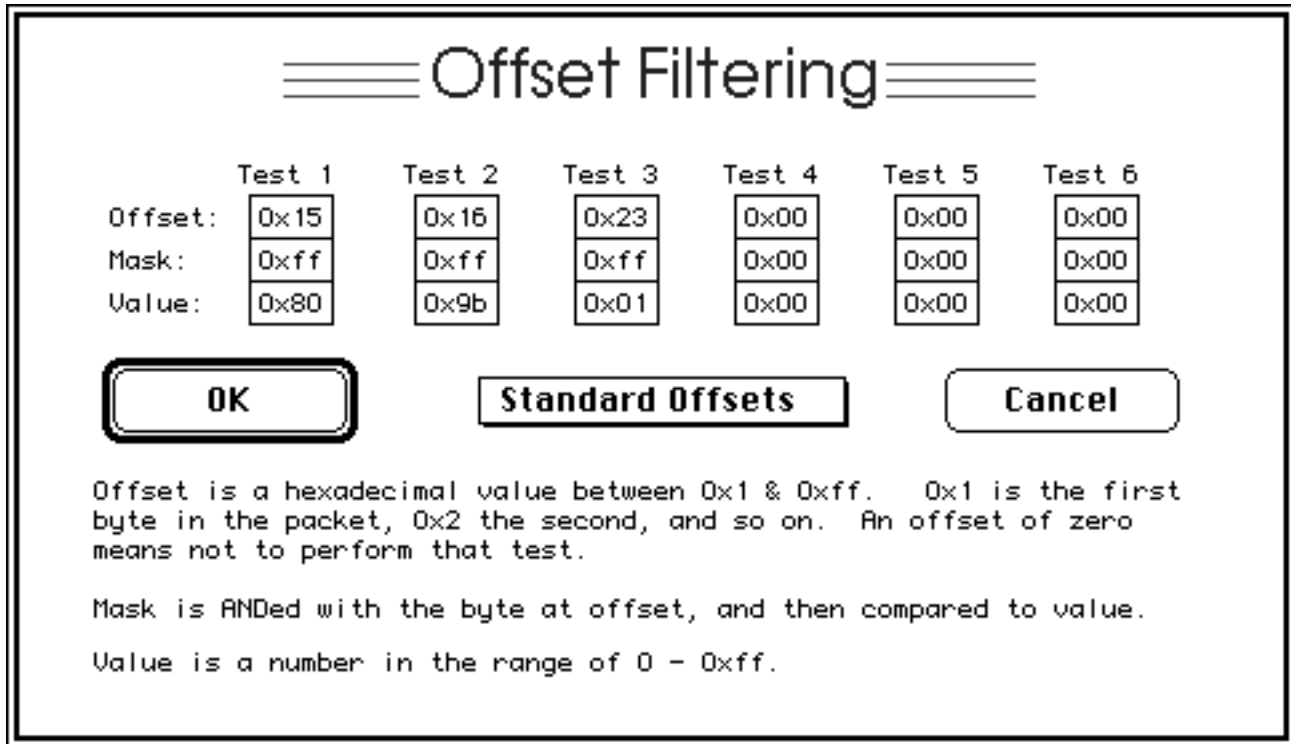


Figure 7-8. EtherPeek’s capture filter is set to accept AppleTalk Phase 2 RTMP packets. The first 2 tests establish that it is an AppleTalk Phase 2 packet. The third test establishes that the packet has “1” for its DDP type, which uniquely indicates RTMP Response packets. Filtering for RTMP packets is 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 there are in your packet trace. In the figure below, it is apparent that there are 5 routers on the network.

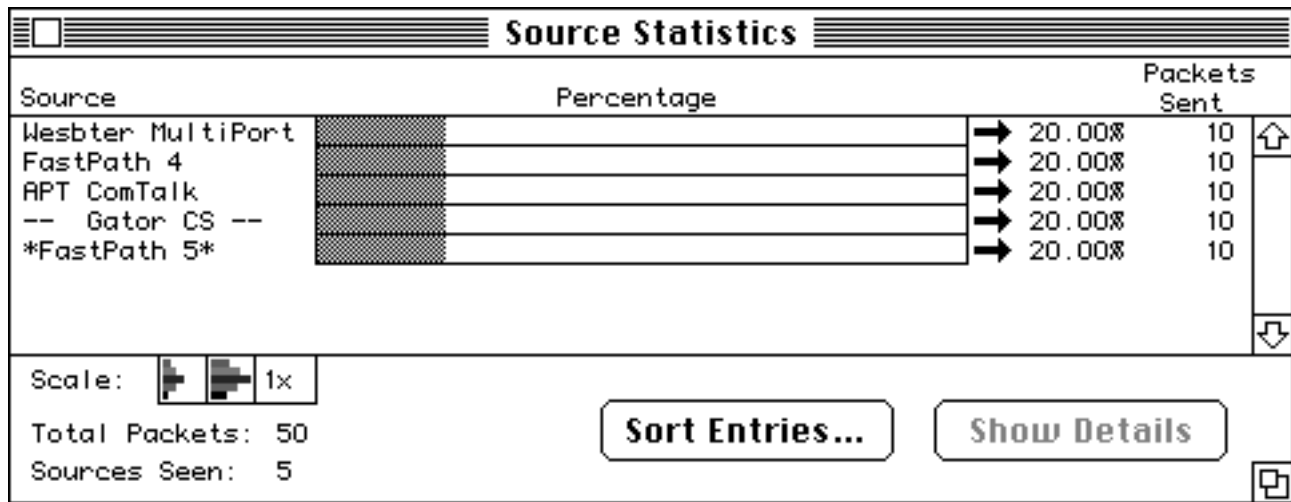


Figure 7-9. When we restrict the packets that we capture to RTMP Packets only, the Source Statistics show us how many routers there are on the network.

In the figure above, you may notice that the name table of the protocol analyzer had a pre-assigned name for every router found on the network. If you take the time to build your name table, new routers are very easily spotted—they will be 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 have been removed from service).

If there are more routers on your backbone network than you expect, then the task is to find out where they are and who has placed them on the network. The next subsection will deal with that case, so you may want to skip it for now, if the number of routers you found is correct and they all have names that were in your protocol analyzer's name table.

Finding an Unknown Router

This task is undertaken when there is a new router that appears on your network that you were not expecting. 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 and the identity of the person who is responsible for placing the router on the network so that 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 do this by comparing the first 3 bytes of the Ethernet address to the table of Ethernet addresses in Section 11.2. If the manufacturer designation is 3Com, Asanté, TechWorks or another manufacturer that makes Ethernet adapter cards but does not make a router, then you know that the router is actually a computer that is running router software. If the manufacturer is Cayman, Cisco, Wellfleet or another manufacturer that makes routers but does not make Ethernet cards that, too, is a clue. The third possibility is that the manufacturer may be Shiva or Compatible Systems or another manufacturer that makes both cards and routers. Knowing the manufacturer helps you determine where to look on the network and also where to look physically within the building. Also, if you have the router configuration software for that manufacturer, you may be able to re-configure the router so that it has correct information.

If the router is also a computer workstation, then a network listing tool such as Inter•Poll may be powerful enough to learn enough about the workstation to locate it and the person responsible.

First, find out the AppleTalk address of the router. Looking in the RTMP packet, you can find out the network and node address of the router that sent the packet. This information will be at the top of the RTMP header. In the example packet (portion) below, 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 will 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 that 1.181 has a Timbuktu socket with the name of "Richard Sherburne" or a QuickMail server with the name of "Room 204 Server". If you are not so lucky, then continue on with the method described below.

If the Inter•Poll search of the backbone network did not result in a good clue, 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 is 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".

TROUBLESHOOTING MACINTOSH NETWORKS

In the packet below, the router is indicating that it is directly connected to 2 networks, network 1-10 and network 52000. You can tell which of these was the network where I 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 that is directly connected to network 52000. In that network, there is another router that is connected to network 52250. You know this because network 52250 is 1 hop away, and since this is Phase 2 (because tuple 1 describes an extended network), this router is only going to describe the network it's on and all of 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 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 only have 1 zone name.

Packet to Send

Zip Query

Required Parameters

Net to check:	52000
Zone to check:	
Start Index:	

Packet Destination

Send Now	Network	1
	Node	181
	Socket	6

Figure 7-10. 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 have determined the zone name, then you can use Inter•Poll to search for clues in the names of the services registered in network 52000.

In some cases, the person that installed the router has such a naïve understanding of networking that they 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 and 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 will typically not be able to reach the unknown router networks and zones.

A good technique to cope 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 will be able to 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 let you discern the identity of 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 does not turn up a clue, and you feel confident that you have the proper authority, you may try more invasive techniques of discovering the identity of the router or the people responsible for its presence. Some of these techniques border on being malicious, so I suggest restraint. These techniques should only be used as a last resort.

In some servers, like QuickMail, you may be able to see the names of the people who are registered users even though you cannot access the server. You might look up some of these people in your company phonebook and ask them who's running their network. You may consider logging into one of their file servers that allow guest access and sending a message to the administrator to give you a call. You might try logging in to their AppleShare Server as a guest and creating a bunch of files named "Call John @ x2465" at the root directory to get their attention.

Once you've identified this new network administrator, you can begin to coordinate your future activities (provided your search activities haven't offended him too badly).

Checking for ZIP Query Packets

After you have restarted any routers missing from the list and tracked down the unknown routers in the list of routers generated in Section 8.3.3, a good next step in diagnosis is to look for ZIP Query Packets. The presence of ZIP Query packets is very often an indicator of a router configuration error. ZIP Query Packets should only be seen at router bootup; if they are seen at any other time, there is usually a problem, particularly if the ZIP Queries are sent repeatedly. (The absence of ZIP Queries, however, does not 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 below.

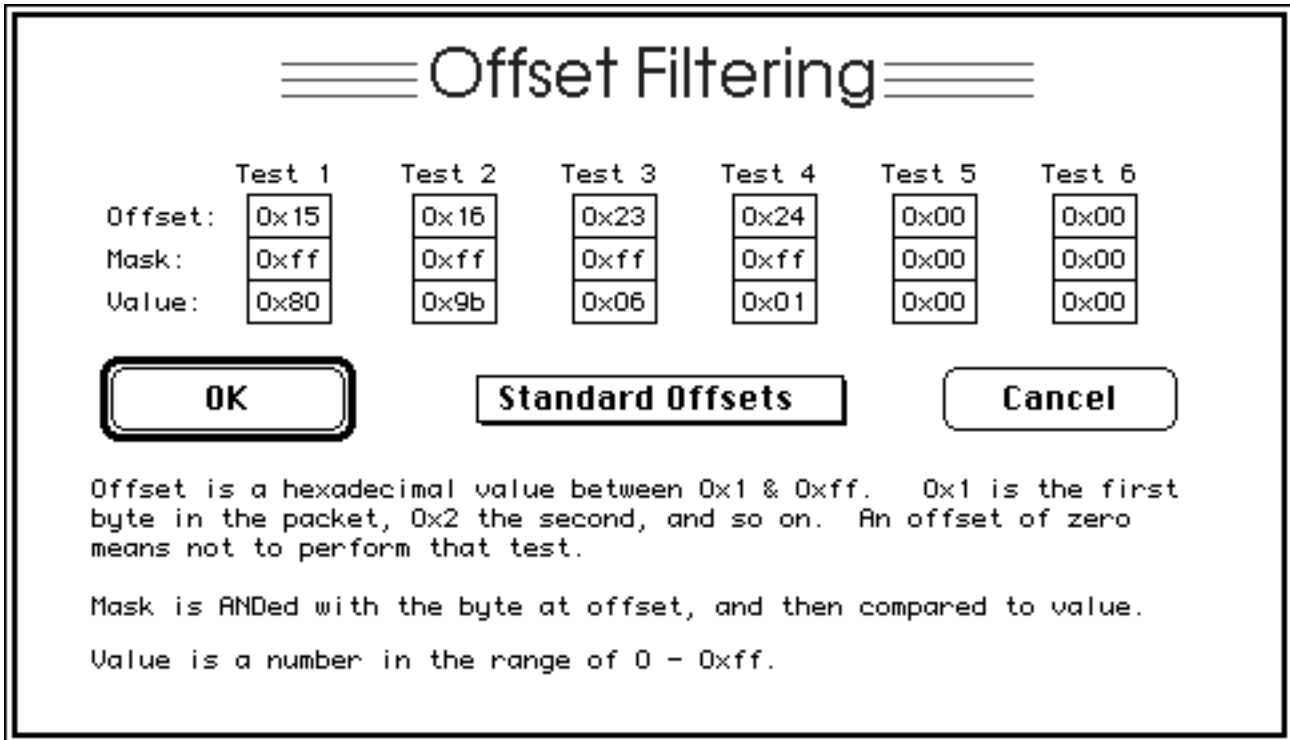


Figure 7-11. Offset filters in EtherPeek for ZIP Query packets. Test 3 checks for DDP type 6 (ZIP), Test 4 checks for ZIP function 1 (Query).

If you see ZIP Query Packets and you can not attribute them to a router startup, one or more of the routers probably has a problem. For example, if you forgot to enter the zone name for a LocalTalk network when you configured a router, the other routers would not be able to 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 will not respond because it has no answer to give (because you forgot to enter the zone name).

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 contain valid network and zone information—the router sending the Queries is probably the router that has the problem. If the ZIP Replies do not contain valid information or there are no Replies, chances are good that the router that is receiving the ZIP Queries is the router with the problem. In the case above, for example, the Query packets were not being answered, and it was the router receiving the Queries that 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 sub-group of them.

If all of the routers are sending Queries, the problem is most likely in the router receiving the Queries. In the case above, all of the other routers on the network would be sending ZIP Queries to the one misconfigured router. This points to the conclusion that the one router receiving the Queries is the one having the problem.

If a group of routers (but not all of the routers on that network) is sending the Queries, try to determine whether there are any identifying characteristics to the sub-group. For example, the AppleTalk Router built into Netware 3.11 has a very short aging timer for Network Information. If only Netware routers are sending the ZIP Queries, it may indicate that the router receiving the Queries was 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 you have 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 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, remove the router's battery for 10 minutes, restart the router and reload the router's software and configuration files.

There is one other interpretation for the situation above, and that is that 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 the network address within that range, "8" for example, to a LocalTalk network elsewhere in the internet. That situation would also cause some routers to send this periodic ZIP Query. If the reset advised above does not cure the problem, you may have to thoroughly look at all of the network numbers.

Mis-typed Zone Names

While ZIP Queries accompany some kinds of router problems, other kinds of router problems are not accompanied by such an obvious error signature. This is true of the misconfigured network shown below. The symptom in this network will not be obvious and will not appear consistently.

The network in this example has an Ethernet backbone with 9 LocalTalk-EtherTalk routers; 2 of them configured as seed routers and the other 7 as non-seed routers. Six zone names are assigned to the Ethernet, and some of the 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.

TROUBLESHOOTING MACINTOSH NETWORKS

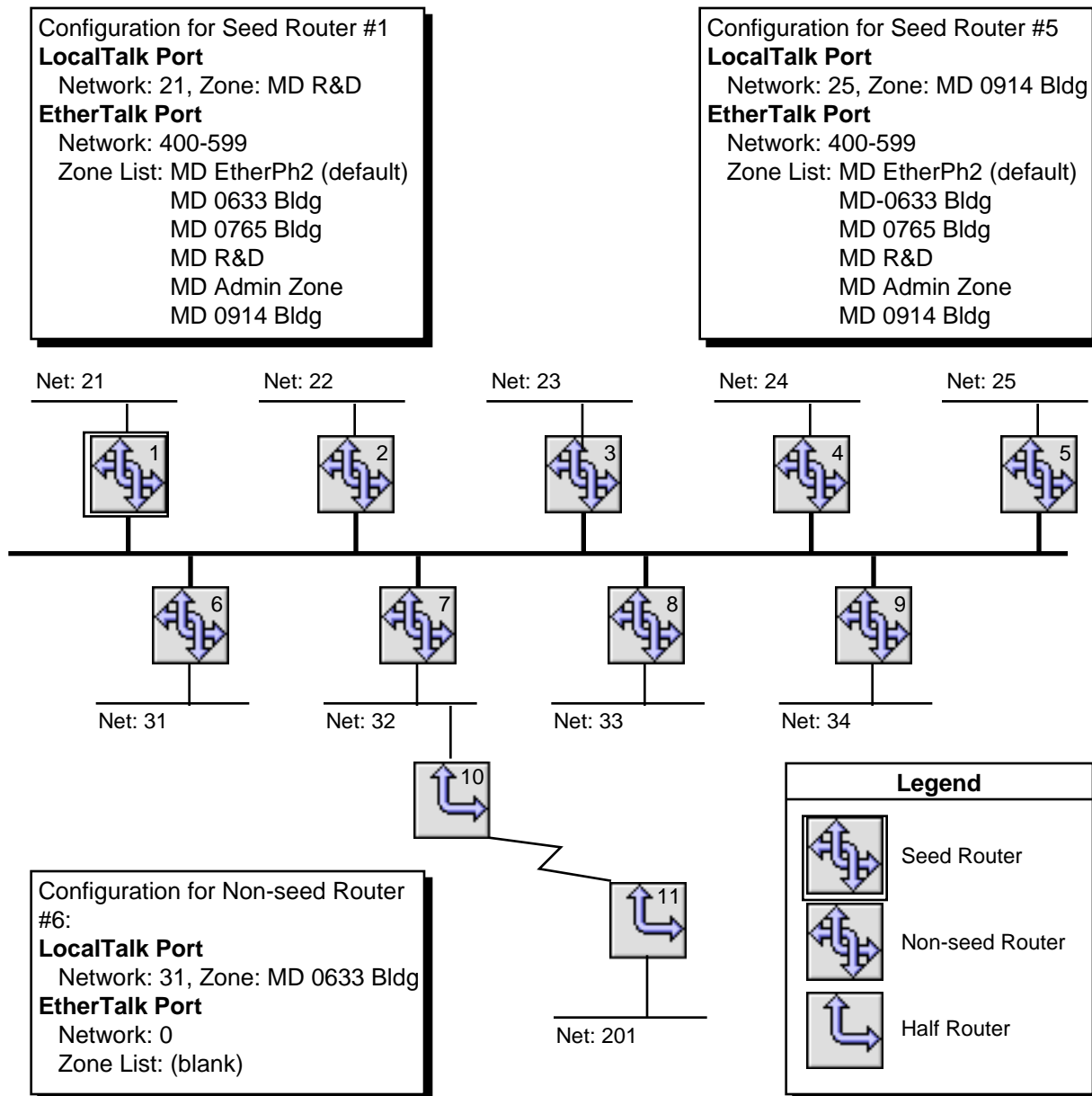


Figure 7-12. This internet has a problem with its zone names. One of its seed routers was configured incorrectly when a technician inadvertently typed a hyphen into one of the zone names.

Let's examine how this problem spreads. Let's say 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 8 other Ethernet routers, all of which supply seed information, and the new, non-seed router randomly picks which one of the 8 to "believe". (Non-seed routers already in service appear the same as actual seed routers.) It has a 1-in-8 chance of selecting SR5 to supply it with information, and if this is the case, 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 is a problem that will be noticed intermittently, if at all. I'll summarize 2 of the irregularities that would occur in this network. In describing these symptoms, I'll assume that the problem has not 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 chose the zone name without the hyphen. This problem is even more confusing to resolve when the difference between the zone names is that one of the zone names 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 the devices in network 31. They would be able to use devices in any LocalTalk network without problems and would be able to use any device on the Ethernet in any zone, including "MD 0633 Bldg".

On Ethernet, users' workstations will usually display a zone list with the correct name, but once in a while a user will see both names in the Chooser. When an 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 built-in checking mechanism in the process. When we perform our diagnostic tests, we'll use this checking method (in a modified form) to find the bad routers.

When selecting a home zone in the Network Control Panel, the user will see a list of the Ethernet zones available to him. This is obtained 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 home zone, the Macintosh will confirm that the zone name is valid 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.

Let's imagine a scenario where:

1. The Mac picked a "good router" for the local zone list.
2. The user selected "MD 0633 Bldg" for his home zone.
3. The Mac picked the "bad router", R5, for confirmation of the zone name "MD 0633 Bldg".

Result: The user will see a dialog box saying that zone "MD 0633 Bldg" is not a valid zone name for this Ethernet and that he will be placed in the default zone name of "MD EtherPh2".

R5 has to report an invalid zone because it believes that "MD 0633 Bldg" only refers to LocalTalk Network #31 and is not a zone on Ethernet. It will supply 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, 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 re-configure the erroneous seed routers first, then reset all of the non-seed routers to restore consensus.

Checking Your BackBone Network's Routers

The scenario described above illustrated 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 is only one scenario. Even if you don't suspect a problem with zone names, and even if you're not experiencing any symptoms, it is still a good idea to check your backbone routers periodically. Earlier in the Chapter, I used RouterCheck to automatically check for anomalies; now I'll use a method of checking the routers with PacketSend. The testing is far more laborious than using RouterCheck, but the labor required for 100 routers is no more than the labor required for two. Also, these methods can be performed 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 (reminder: 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 from there to the other networks. In the example tests, I'll be using only 2 routers, a FastPath 5 and a Gator CS, and in each case the FastPath 5 will have the wrong information. (This is no reflection on my attitude toward the FastPath 5; I flipped a coin to determine which one would be the "wrong" one.)

To verify that all of the routers is holding the same information for the backbone network, we need to verify that each router on the backbone network has: 1) the same network range assigned to the backbone network, 2) the same number of zone names for the backbone, and 3) the identical zone names assigned to the backbone.

I'll perform the three verifications in order. To make sure that each router has the same network address range, I'll broadcast a ZIP GetNetInfo (GNI) packet to all the routers. 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 to us as a verification of that network address range.

When using the ZIP GNI for the purpose of confirming the network address range, I leave the zone name blank. That way, each router's reply will tell me that the zone name that I asked about is invalid, and the network address range will be in the same offset position in every packet.



Figure 7-13. 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 you broadcast this packet, you should get a response from every router on the backbone. Since you counted them previously, 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 that is sending the GNI.

Once you have captured the GNI Replies, you can use your filters again to highlight just the Replies that contain the correct network range, leaving the incorrect Replies not highlighted. It's easiest if your protocol analyzer lets you filter on a text string—you can just enter 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 4 bytes of the network range occupy the 4 consecutive byte locations starting with offset \$26 (counting from the beginning of the Ethernet destination address). For the network range of 100-120, your offset filters would look like this:

	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 7-14. Offset filters to select GNI Replies with the network range 100-120.

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 be highlighted. If your protocol analyzer simply clears the packets instead of highlighting them, you'll need to reverse the criteria, of course; have the protocol analyzer clear all of the Replies that have the correct range and retain the Replies that have an incorrect range.

TROUBLESHOOTING MACINTOSH NETWORKS

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 7-15. After filtering, the selected packets will show which routers that 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. If you just have 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 is 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

Send Now	Network	0
	Node	255
	Socket	6

Figure 7-16. 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 criteria for this is shown below. The filter is set for a Reply with 4 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 7-17. The filter checks byte \$2b to see that there are 4 zones listed in the GLZ Reply.

After filtering, the routers that have the correct number of zones will not be highlighted, and the routers without the correct number will not be highlighted. This is shown below.

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 7-18. 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 in it as well as a different number of zones. I do not find it fruitful to use the packet length as the criteria for correctness, even though on this network, any correct Reply would be 81 bytes long. The packet length is not useful for the correctness criteria for 2 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 are using a start index higher than “1”, the routers will not 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 5 zones (also listing their names), but each router will probably list a different group of 5 zone names. Thus, all of the Reply packets might have the correct number of zones yet have different lengths. Because of these 2 reasons, the only reliable test is to check for the number of zones contained in the packet using the offset filters, as shown above.

Now that we have verified that each router has the same network range and the same number of zone names (and reconfigured the routers that we found to be in error), it is time to check the zone names one by one. We’ll use the ZIP GNI again, but now we’ll type in the name of each of the valid zone names 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 GNI’s 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, that can be discerned by looking at the packet length. In this test, packet length is the best criteria because it is fast and also reliable.

Let’s look at block of Replies to a GNI that asks whether “MD 0633 Bldg” is a valid zone name:

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.....[551.511]	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 7-19. Routers that do not have the valid zone name will have a slightly longer packet

The GNI Replies in the figure above are 65 bytes long, with the exception of 2 routers that sent 71-byte Replies. These 2 routers both reported that “MD 0633 Bldg” was not a valid zone name. These routers need to be re-configured if they’re seed routers, and restarted if they’re non-seed routers, after all of the other configuration problems have been eliminated.

Analyzing Network Traffic

I 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

TROUBLESHOOTING MACINTOSH NETWORKS

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. The purpose of understanding traffic flow is to provide data for the tasks of network design and resource planning. Some of the aspects of your network that you may want to quantify are:

1. 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 a power network user does with the network because power users are sometimes doing what many users will come to do in the near future. Once in a while, you may want to restrain a power user or suggest alternate ways for him to accomplish his computing goals, but in general, you just need to understand the nature of this kind of usage when planning for the future.
2. Who uses the network the least? These users should also be consulted. Why are they using the network so little? Do they lack training or lack of need to use the services? If these users lack a need for the network services, are the services properly configured to provide the most useful service possible? Perhaps these users have a better way of accomplishing the same business functions.
3. What is the level of traffic and how does it change over time? You may also 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?
4. Are there any times of the day (or week or month or quarter) when the usage of any particular network resource is predictably high? If so, you may want to think about your resources in terms of these heavy usage times or think of ways to reduce the usage. For example, if everyone needs to print enormous documents the last 2 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 that there is a tremendous amount of network activity on Thursday in the accounting department because of payroll activities, then perhaps Thursday evening is a good choice to schedule a full backup of the accounting server.

5. Are there any users who consistently use resources in other networks? Is there traffic going to just one other network or is it going to several different networks? Would the user be better off if he were placed in one of those other networks? Or would the service be better off if it were placed elsewhere? Remember, routers should be used to segregate traffic, not to funnel it.

The Inadequacy of the Current Generation of Tools

The group of questions above, particularly the last question, exposes 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. Farallon's TrafficWatch II (v1.0) is cumbersome, doesn't run well with anything less than a 68030 Mac II and has major bugs both in its design and in its code. For example, it's not able to save statistics for networks that have an address of 100 or greater (whoops!). It's also prone to giving false error statistics, as is reported in more detail in the section "Data Link Troubleshooting". Also, since it does not record the number of bytes sent between devices, only the number of packets, (which is meaningless unless you know the size of the packets) you don't have any useful information with which you can characterize your traffic patterns. Even if all of these things worked, it still couldn't tell you about traffic to other networks, since it only looks at the LAP portion of the packet. It can tell you that there were a lot of packets sent to the router, but not where the packets went beyond that. TrafficWatch II is an ambitious program and all of the windows are fun to watch, but it's just not very useful except when you want to pad reports to make them seem more

impressive and well-researched. I used it for this purpose once, and I have to admit that TrafficWatch really shines in this capacity. The reason that I did this was to out-maneuver some die-hard MacHaters, but this story is probably better saved for a book on network administration. The short story is that they were very impressed by my use of TrafficWatch data to back up my conclusions.

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 only tells you what the LocalTalk utilization is right *now*.

On some servers, the log file will provide 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 not designed to analyze traffic patterns, but to analyze the *content* of the traffic. Although the method is tedious, you can make use of some of features in EtherPeek to figure out what percentage of your traffic is going to and from which networks and which devices on those networks 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 (no packets except those that originate from or are destined for this user). I'll work through an example where I let the protocol analyzer run for 7 minutes and collected packets to and from a user named Mark. After stopping the capture and saving the file, I can open the Source Statistics window to get a breakdown of how many packets Mark sent to each node that he communicated with. While it's helpful to know the identity of each of the nodes that Mark communicated with, the numerical information in this window is not particularly useful for my purpose since it has the limitation of only giving the number of packets (not bytes) just like TrafficWatch data. The interesting numbers in this window are in the summary section, where you see the total number of packets that the user sent and the average number of bytes in those packets. Multiply those two numbers together and you get the total number of bytes sent out by this user.

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 7-20. With this summary information, you can calculate out how many bytes that a particular user sent during the time the data was captured.

With the summary information above, 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 7 minutes. This works out to 1608 bytes per second (12865 bits per second) or 0.13% 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 and make this calculation.

Documenting a Network for Troubleshooting

I've said many times in this book that 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". In addition to this primary criteria, the usefulness of the documentation will depend on how clear, detailed and up-to-date it is.

Each of the five modes of troubleshooting outlined in the Introduction has a different orientation to the concept of "what should be" and each mode has different documentation needs. This sub-section will be organized according to those 5 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 user's system is working. You try various measures to cure the problem and when the problem is gone, you are finished. You may not ever find out whether a network problem, hardware or software problem caused the symptom.

In order to notice that there is something peculiar in the way the 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: 8 zones listed in the Chooser, 4 LaserWriters in the "Southern Ops" zone, about 70 seconds to complete its startup cycle, about 3 minutes to print a particular file on its hard disk, about 1 minute to download a particular file from the file server. In small internets, some of these expectations will not need documentation per se, because they can simply be remembered. The network manager of this small internet would know exactly how many LaserWriters and NetModems he had installed in each zone.

In a larger internet, however, it is wise to make a list of the expected 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, not only have the ability to build a list of the devices in each zone, they have the ability to periodically check that the device is still reachable through the network and report the status of the devices on an on-going basis, visually on a graphical representation of the zone, by audible noises, and by a log entry. These programs simulate a user's desire to reach these services and monitor the success rate that a user would have if he wanted to make the connection. 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.

In terms of your network performance expectations, an easy method is to create and keep files on your disk that serve as benchmarks. These are files you have 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 6 pages long and contain a 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 a large application that doesn't get upgraded very often like Excel—the size of an application 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. Let me give an example of using estimation to predict "normal": Let's say that you were transferring a 500K file, and it took 19 seconds to transfer. Is that normal or not? Very few network managers can actually answer this question off the top of their heads. If you have previously developed data for a 500K file transfer, you could compare your previous data to the 19 second time and know immediately whether this was normal. If you don't have the data, you can use estimation to calculate whether it falls within the range of 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. We said that 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% to 80% 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% utilization (5K/sec), the file transfer would take 100 seconds, at 80% it would take half that time, 50 seconds. “Normal”, therefore, for a 500 K file will be between 50 and 100 seconds. 19 seconds is definitely outside the normal range.

For printing, I usually take the printer manufacturer’s claims (8 pages per minute, for example) and multiply it by a factor of 75% for straight text with few or no font changes in the document, by 50% for text and graphics, and by 25% for a document that is mostly graphics, then add 30 seconds. For example, I 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, I would estimate the normal time to be 4 and a half minutes and wouldn’t be suspicious of a printing time between 3 minutes and 7 minutes.

Documentation for Hunch 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 look at the “qualities” that the problem has, explore the boundaries of the problem—where and when it happens with which software and systems—and make intelligent guesses about which system variables to manipulate. Like Random-mode, you’re finished when the problem is gone, and you may not know what the problem was, but you may have some guesses because you used your previous experience as the basis for the process.

There are 2 documents required for Hunch-based troubleshooting. 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 in the log should also be useful in helping the troubleshooter see the big picture. The larger picture may help the network manager to fashion solutions that 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 what device or network component had the problem, the time and date that it occurred, what the user was trying to accomplish when the trouble happened, a “snapshot” of the device’s hardware and software configuration when the trouble occurred, the user’s experience level (beginner, intermediate, advanced) and a record of the steps taken to provide the remedy. You may notice trends in the log. For example, you may notice that a particular kind of problem only happens to users of a certain level or that a certain problem only happens on Monday mornings. The log should also record all system modifications such as software upgrades, hardware modifications and repairs, etc. Keeping a log is always beneficial, but 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 get used the majority of the time by the majority of the troubleshooters. Watch out that you don’t inadvertently put any impediments in the path to achieving that criteria. 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 data base 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 a simple notebook would be more appropriate. The point is that the logging system should serve the need of compiling experience in such a way that everyone finds it convenient to add to 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 benefit from this is that experience gained by one person can be used by

TROUBLESHOOTING MACINTOSH NETWORKS

another. The second benefit is that log books are often useful for building claims and justifications. If your wiring system is bad, but your management thinks it is too expensive to fix the problem by re-wiring, your log book can play a crucial role in building the justification for the critical task of resource allocation. If you can tell 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.”, that is a lot more convincing than, “Geez, boss, the other guys and me 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 re-wiring is called for, but the log book may show that the time and energy put into fixing wiring problems would not justify the cost.

Although you can’t count on this happening, if you keep a detailed log, you might also be able to notice patterns in the problems that you encounter that would lead to changes in network or system management policy. You might find, for example, that every single problem with Application A was accompanied by the presence of Screen Saver B. From your experience, you might be able to discern “rules” of system configuration from the continuity of experience embodied in the log.

The second crucial document of Hunch-based troubleshooting can be called by many names, “Escalation Chart”, “Tip Sheet”, “Help Card”, etc. This is the piece of documentation that should be given to every user to explain how to solve problems. How much latitude you should give to your users to solve problems is something to which you should give serious thought. If you give them too little latitude, 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 the troubleshooter arrives on the scene. If you give them 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 typical network problems.

Somewhere between the ability to change the cross-connects in the wiring closet and the ability to change fonts in their own documents, the network manager must draw the line between what the user can and cannot do.

For example, the document might say:

Printing Problems:

If you are having problems with 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, that it has a PhoneNET connector plugged into it and that 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 the troubleshooter where the computers are situated and which users use the computers, the troubleshooter also needs 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 3 points of view that can be taken when generating the map that specifies the connections between the components of the network. The first approach is to take the user's point of view. In this case, all we 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 diagram, a line might be drawn in to indicate the Sarah is also connected to a LaserWriter II NT nearby.

The second point of view is that of the wiring technician's. The "map" in this case will consist 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, then the extension number is listed. If the pair is used for a network adapter, then the type of adapter is listed and/or the port to which it is connected.

From the wiring technician's point of view, the 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 port's are listed, with the circuit and devices on each port listed:

```
LocalTalk Hub 5, Port 6
22L4/8 Sarah Connor Mac IIsi/80/5-LaserWriter II NT
```

The ideal mapping system would be able to show you the network map in any of these 3 views, but few of us are willing or able to make a system that fancy. 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.

Some of 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 information mentioned above. If you can't change your system or augment it to include information useful for troubleshooting, then make a second, simple system. A very simple system that records 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 SuperVisor, 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: 1) it requires that the network manager gain familiarity with the features, idiosyncrasies and limitations of each of his management applications, and 2) it produces the “normal” data needed for comparison.

I think the logic in the above paragraph is obvious to all, and I hope this will translate into action.

Documentation for Process Mode Troubleshooting

The primary tool for process analysis is the protocol analyzer. Having taught protocol analysis to hundreds of people, I know that learning to analyze network processes takes time.

The approach that I use in class 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 to the point where 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 what 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 trying to 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. By understanding the flow of these small network tasks, we eventually reach the point where we can dissect a complex process like printing. 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 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 within your reach and definitely worth the effort. Protocol analysis is absolutely the single most effective technique in network troubleshooting.

The point of the above-mentioned learning process is to build an expectation 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 all sorts of 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, the documentation should be easily accessible, up-to-date, 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

The kind of tools that I'm going to focus on in this section are the software tools that you need for troubleshooting. Most of you have all of the hardware tools that you need—all the crimpers and strippers and screwdrivers that are required for your daily tasks. I'm also making the assumption that you have all of the configuration management software required for your routers and servers. If you use a GatorBox, you obviously need (and have) GatorKeeper software. The information in this section is supplemental to other information in the book, particularly to the information in 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're just intending 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, etc. If you intend to do any technical climbing, you should add other items to the basic kit—a rope, some carabineers, a seat harness, to name a few. On glaciers, there are further requirements—crampons, an ice axe and dark goggles, for example.

None of the lists that you see in the guidebooks are complete lists; you can always take anything else that you like—a flask, your favorite bandana or a copy of "The Dubliners"—but only in addition to the basic necessities.

In contrast to mountaineering, it's harder to prepare for network troubleshooting 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 you got 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 and asking you, "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 different advantages in different situations. Also, something that is not appealing to me may very well be appealing to you. I have participated in group discussions where the merits of GraceLAN, Network SuperVisor, Status*Mac and NetOctopus have been discussed, and I 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 axe 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 axe and teach you how to arrest your fall before they take you to a place where your ignorance of this procedure would result in your death (and possibly the death of others). 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 will scan the environment, gather information and display it. You have 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 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.

OK, with that sermon out of the way, here are some thoughts about the contents of a good troubleshooting tool kit.

The Basic Tool Kit

The basic tool kit includes:

1. your network hub's management software
2. Apple's ancient but still useful Inter•Poll
3. Timbuktu (or Carbon Copy)

Your hub's wiring management software is crucial, provided it can give you the kind of information that you need about the health of the wire on a port. Most LocalTalk hubs should now be able to tell you the names or at least the node addresses of the devices that are connected to each port. 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.

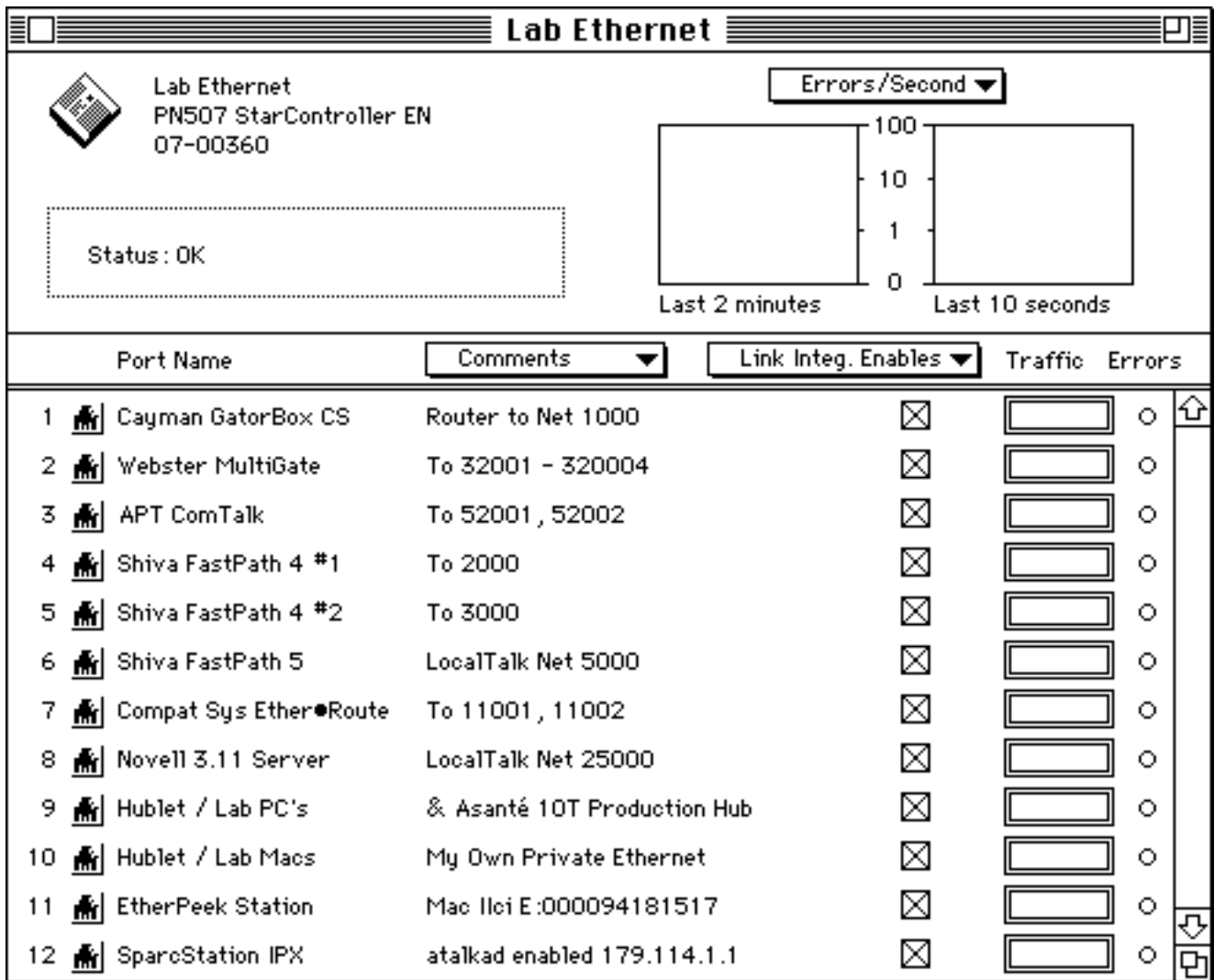


Figure 7-21a. Farallon's Hub Management Software-StarCommand 3.0


StarController Info	
	Lab Ethernet PN507 StarController EN 07-00360
Status: OK	
Statistics:	
Total Traffic (Bytes):	16,035,268
Total Errors:	11
Total Traffic (Packets):	242,379
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 12/2/91 KVS	
Map - NetWare::Int:NetDocs:Labnet:E Config	
Production Net on P9 (Asanté Hub)	
Current Status: Not Connected	

Figure 7-21b. Farallon's Hub Management Software-StarCommand 3.0

Inter•Poll can quickly check the health of any network or zone in your internet. I know that it's old, and that it beeps at you when you start it up, but it's still does what it does better than anything else. The best 3 features are that it tells you how many zones are in your internet, lists the devices in a zone and performs the echo test, the uses of which are described elsewhere in this book.

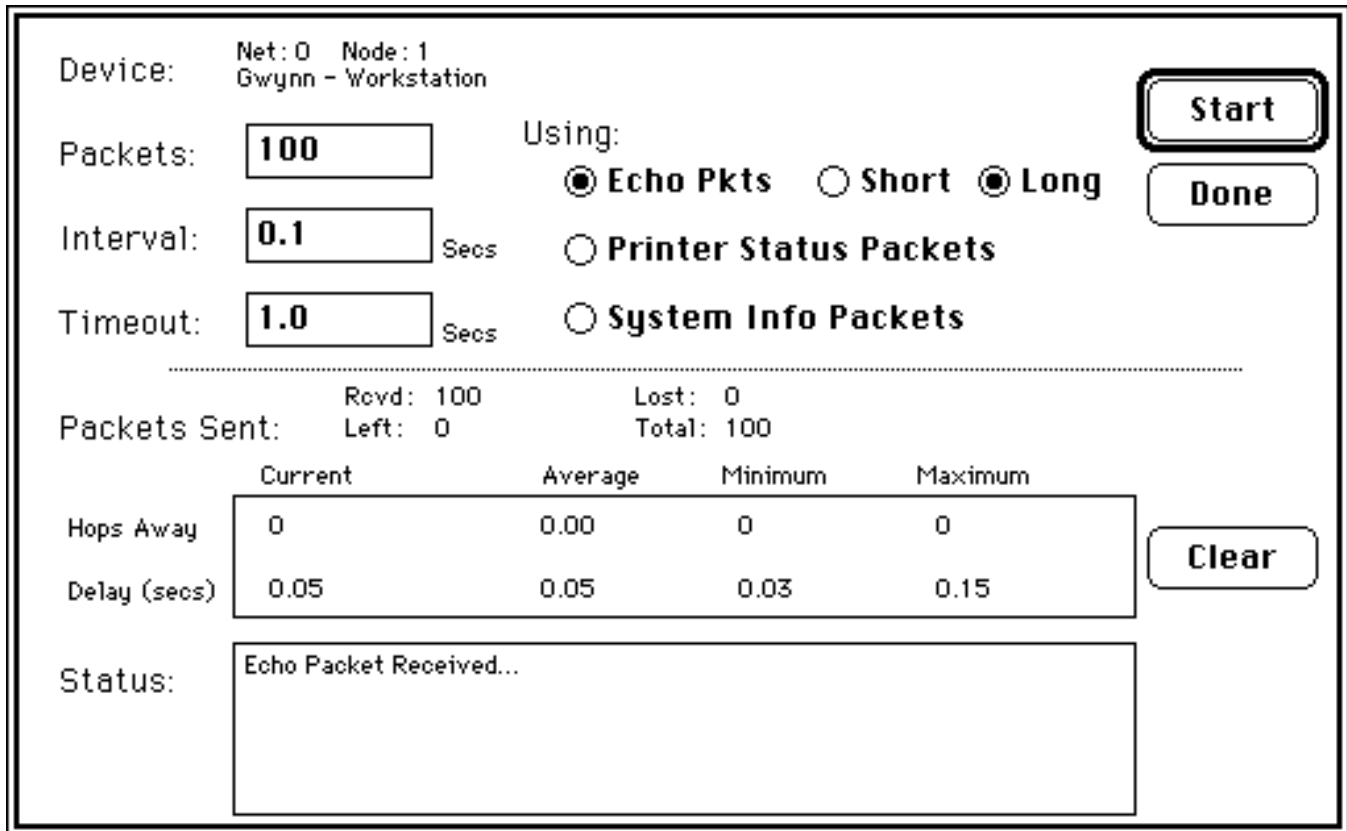


Figure 7-22. Apple Computer's Inter•Poll-Echo Test

Timbuktu (or Carbon Copy) is great for troubleshooting because it makes you more effective from more physical locations. I 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 at least 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 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, I think we'll see a Windows-based SNMP console before we see a Mac-based console.

I 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 3 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. system management package such as GraceLAN or Network SuperVisor

The 2 main examples of traffic monitors are both made by Farallon, TrafficWatch II and NetStats. Of the two, NetStats is more useful for troubleshooting purposes. TrafficWatch II, used with great thought, is of limited use for other network management functions like resource allocation and planning. TrafficWatch II (version 1.0) has many more drawbacks than utilities and I do not recommend its use for any purpose in its present condition, however

NetStats, a component of Farallon's Network Manager Pack (which also includes TrafficWatch II), works only on LocalTalk networks. It samples 1000 times per second and can show utilization values every 1/10th 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 only on the wire for a very short time.

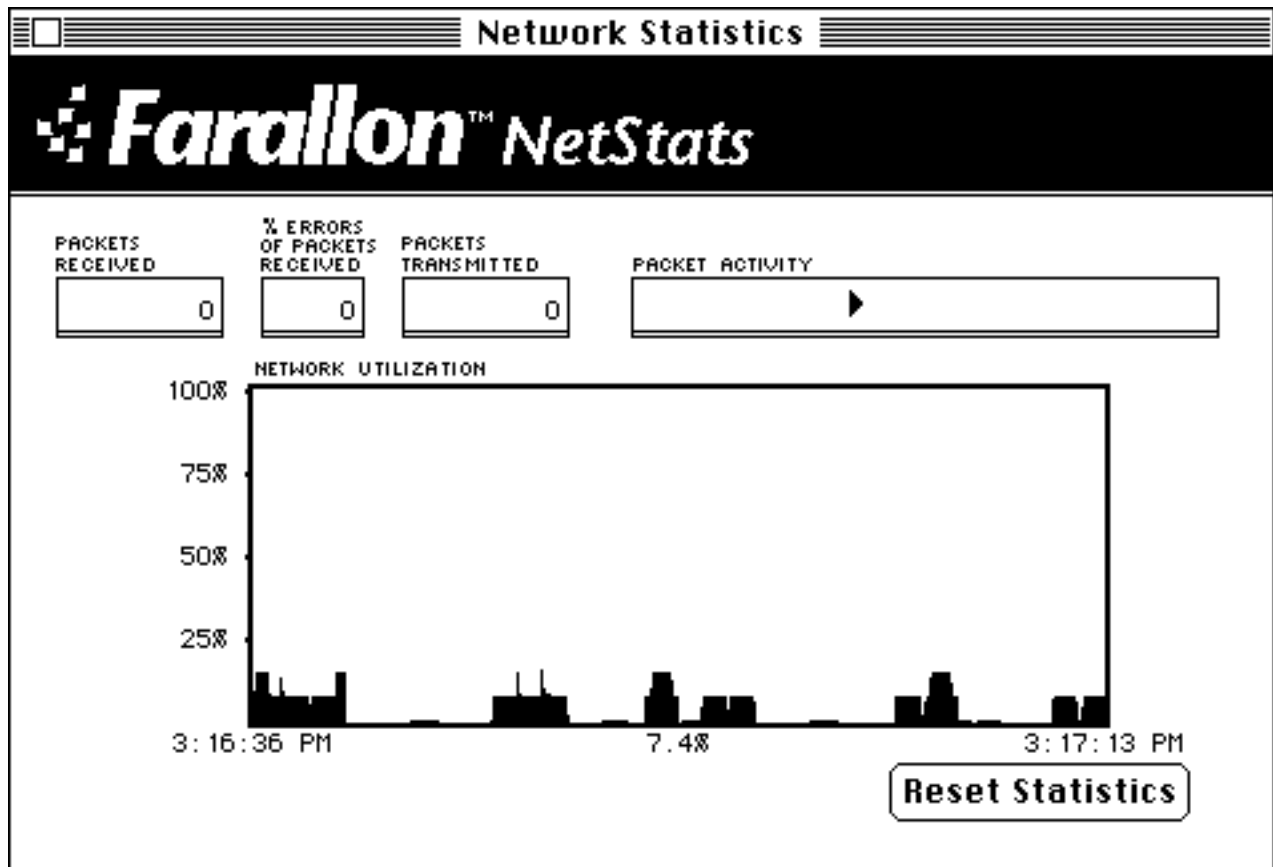


Figure 7-23. Farallon's NetStats

The traffic level that you see in NetStats can tell you if the (LocalTalk only) network's busy and 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 each slow the other down by jamming up the network with traffic? How about when 4 users are printing to 4 LaserWriters? The answer to both of these questions is "No", and this will be quite apparent to you after you have developed a familiarity with NetStats. This kind of familiarity is extraordinarily useful when making guesses about the nature of a problem.

Aside from the insights gained, sometimes you just want to answer the question, "Is the network busy right now?" On LocalTalk, NetStats will answer that question very quickly.

For EtherNet and Token Ring links, the traffic monitors are a bit more expensive. Network General's Watchdog is around \$2000 and provides utilization and error statistics. It allows you to set alarms when these values rise above a pre-defined threshold. Mac-based products that will have appeared in this product category by the time you read this are SkyLine/E from the AG Group and NetScope from Dayna.

These kinds of measurements can also be made using DOS-based protocol analyzers like the Sniffer or LANalyzer and in the hub management software of the more expensive Ethernet hubs. In these devices,

TROUBLESHOOTING MACINTOSH NETWORKS

there is special hardware and firmware that is used to capture and simultaneously analyze the traffic at a fast enough rate to provide the time resolution necessary.

All traffic monitors have one basic limitation, however. They can only tell you 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.

The next type of management program necessary at the intermediate level is the type that 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 get them set properly.

Some "network management" packages are not really for managing a network, but are instead for the purpose of 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 on the kinds of 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 the information in real time. ON Technology's Status*Mac may possibly 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 an ability to perform automatic comparisons between two configurations and simply give you result of the comparison—comparisons, for example between two similar devices, a comparison of all of the devices within a class of device or between a device and its standard configuration or between a device as it's configured "today" and the way it was configured at a previous time. These comparison would be very useful for troubleshooting because they could help you answer such questions as "Why are these 3 devices behaving normally, but this fourth one of the same type is having trouble?" or "This device working perfectly last Friday. Why is it not working today?" Unfortunately, while all of these comparisons are useful for this type of analysis, they must be performed by the "software" in your brain. With today's software, you can only use the system management packages to gather the information, not to analyze it.

The only exception to this rule is Teknosys' Help!, an analysis package that will analyze a single Macintosh's configuration for thousands of known configuration problems. Help! currently has no abilities to work over the network or with the 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 7-24. 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 2 basic types—software that runs on a Mac for less than \$1,000 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'll need. While the DOS-based protocol analyzers are very powerful, the Mac-based protocol 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 decide to get a DOS-based analyzer, you're going to 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. I have used most of the major contenders in this area, but for 95% of the AppleTalk troubleshooting I do, I 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 still cost enough money that you probably don't want to buy both. I'm sure that you'll have equal success with either one. I happen to prefer the AG Group's products because overall I find them slightly easier to use, but I have colleagues (who's opinions I greatly respect) that 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 will actually endow the devices with animate characteristics and needs. They say things like, "OK, 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. This viewpoint 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 the

TROUBLESHOOTING MACINTOSH NETWORKS

individual devices, placing yourself in their 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, how do I ask for this information?”

More than any other tool in your kit, protocol analyzers will take the longest to learn to use properly. 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 all of the information on the network is made visible.

Summary

An expert troubleshooter requires the following tools:

1. The network hub's management software
2. Inter•Poll
3. A screen-sharing program like Timbuktu or Carbon Copy
4. A traffic monitor
5. A network alert system like NetWatchMan or NetWORKS
6. A system management package like GraceLAN or Network SuperVisor
7. A protocol analyzer like EtherPeek or NetMinder Ethernet

Case Study

Since in the process of printing, the Mac and the LaserWriter exchange Postscript commands, the troubleshooter can see and understand the details of the printing process by reading the packets. Troubleshooting a printing process can be intimidating at first because of all the Postscript code you see inside the packets. Most network managers are unfamiliar with the PostScript language, but as you will see, PostScript uses commands that are English words or 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 and in the case study presented below, Hunch mode helps the troubleshooter gather clues and develop theories when the user says, “I can’t...”.

Troubleshooting a Printing Problem

In this case study, Hunch mode even discovered 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 terminology: The printer in this example is a GCC BLP IIS, which is very similar to a LaserWriter NTX in performance and use. I will refer to it as the “BLP” from now on.¹ “LaserWriter” will refer to LaserWriters as a class of devices. “LaserWriter Driver” refers to the Chooser extension file (which Apple formerly called an “RDEV”) that drives LaserWriters, including the BLP.

What Happened

A user had a disk that contained, among many other files, a 9-page Microsoft Word 4.0 document consisting of text and graphics. They 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 OK, but that it couldn’t be printed because of a PostScript error.”

The first thing I did was to repeat the user’s procedure, with one exception. I printed the document with PrintMonitor in the foreground so that I could see all the status messages coming from the printer. The job proceeded normally at first¹:

1. Pages to Print: 9

Looking for LaserWriter “TNG Laser”.

³The BLP is completely compatible with Apple’s LaserWriter Driver, although with a GCC-modified driver it can print nearly edge-to-edge. It has a few nice features, such as the ability to change many of its characteristics from a control panel with LCD display on the front of the unit .

TROUBLESHOOTING MACINTOSH NETWORKS

2. Pages to Print: 9

status: starting job

3. Pages to print: 9

user: Kurt; document: Instructions2; status: processing job

4. Pages to print: 9

user: Kurt; document: Instructions2; status: preparing data

5. Pages to print: 8

user: Kurt; document: Instructions2; status: preparing data

Then the following screen message appeared:

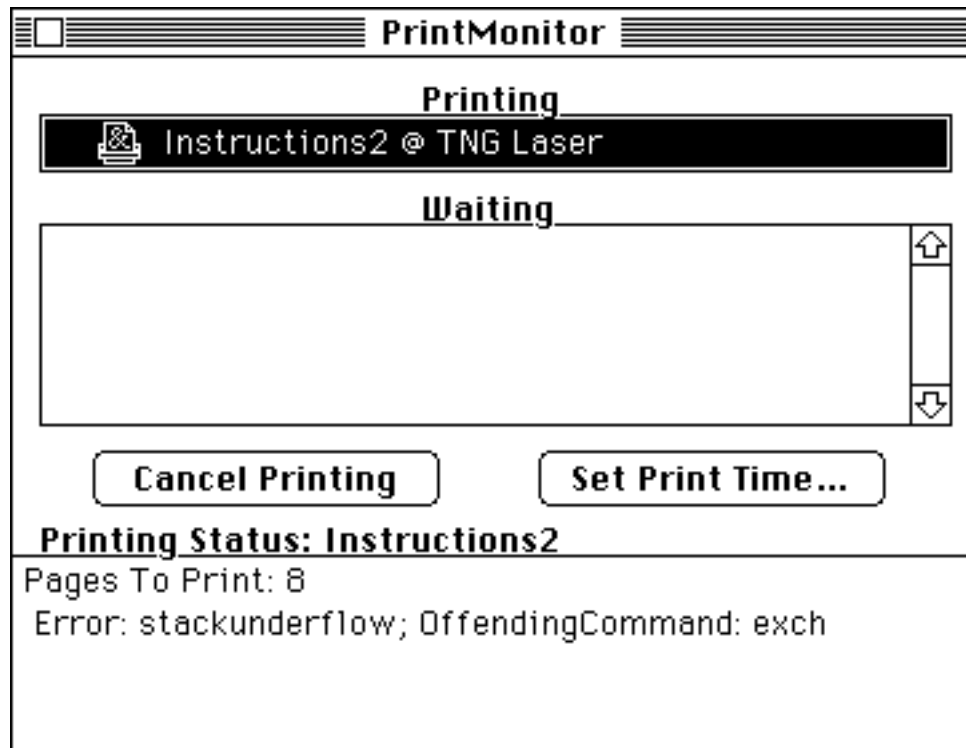


Figure 8-1. During a printing session this message appeared on the screen. The print session on this 9-page document terminated without a single page printed. Messages like this one that occur in the middle of the job, are only seen when you have PrintMonitor in the foreground.

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, I checked the BLP printer manual, which said that messages in this format are generated by the standard error handler, and that I should check the *PostScript Language Reference Manual*. By checking through a few PostScript and LaserWriter reference manuals, I found out 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 very common PostScript

⁴A little tip: Farallon's ScreenRecorder, part of their MediaTracks product, lets you record screen activities. This is a great way to create a reference document for later, if you want to review all of the screen messages. It's especially useful for those messages that are only on the screen for a second and disappear before you have time to read them.

command, there's no telling where this error occurs in the printing process without a little digging. We begin by stating the problem.

Problem Statement: Something in the document causes PostScript errors on the BLP.

Now that we have stated the problem, Hunch mode is a natural choice. We know that this is some kind of software error because the problem occurs in PostScript. We have to look at what software is involved. There is the application (Microsoft Word), the LaserWriter Driver, 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, I thought that there might be an incompatibility problem between the driver and the system. Nothing I've ever read about using the System 7 driver on a System 6 machine would indicate this possibility and the user had used 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 I had no knowledge of the document's contents other than what I had seen on the screen, I made software incompatibility my 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.

Testing the incompatibility theory, I printed the same file from a Mac running System 7 and got the same error response. So, I removed the System 7 LaserWriter Driver and installed System 6.0.7 LaserWriter Driver and Laser Prep system documents on the Portable. When I printed the file from the Portable once again, the LaserWriter, after re-initializing itself, printed the document without error.

Despite the fact that Hunch mode "cured" the problem (I was able to print the document), the cure was not acceptable to me. I couldn't keep System 6 on the Portable allowing the user to edit and print the remaining documents at any time, because the rest of the office used System 7 drivers and I didn't want to have the printer engaging in time consuming re-initializing over and over. I needed to find a way to be able to print the documents from System 7 LaserWriter Drivers.

At least I had the document to look at. I looked it over for any clues I could gather. It was nine pages of text and graphics, with 2 or 3 graphics per page. It looked like it had been made by a fairly advanced computer user since it used a lot of features that beginners don't use, like vertical lines, tables, and footnotes. Most of the graphics were screen dumps that had been enhanced with text annotation and arrows. Although there didn't seem to be any color graphics, judging either by the printed page or by what I saw on the screen of the System 7 print test, color graphics on LaserWriters or BLP's are usually not 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 showed me that there was something in the file that was compatible with System 6 LW drivers, but was incompatible with the System 7 drivers. Since Microsoft Word 4.0 definitely is compatible with System 7, the most likely conclusion is that there was an object in the file created by an application that was not System 7 compatible, probably one of the graphics. Since Microsoft Word doesn't have any native graphics ability, any graphic in the document would have come from another application. The next question, then, was, "Which graphic, and from which application?" To answer this question, I needed to switch to process mode.

Second Pass—Process Mode

First, I thought about the clues that were available. 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 LW driver.

Clues:

TROUBLESHOOTING MACINTOSH NETWORKS

- Since there were 9 pages in the document and the error message showed 8 pages 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 my Mac. I 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, here’s a 50% view of page 2 of the document I was trying to print:

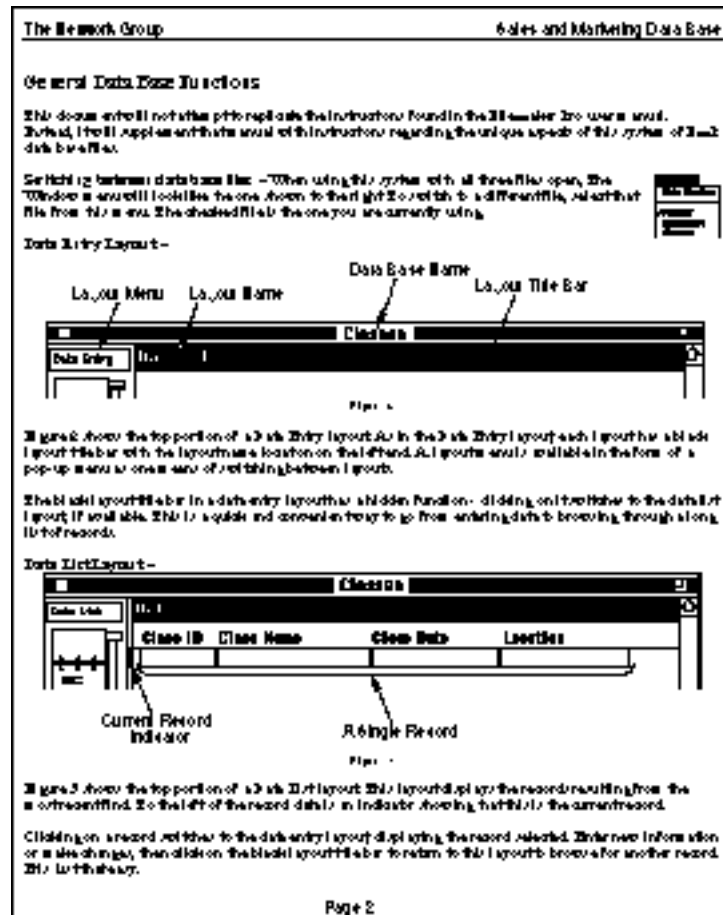


Figure 8-2. The current theory is that one of the graphic objects on this page is incompatible with the System 7 LaserWriter drivers.

Exploring the Trouble Process

The next step is to find the place in the file where the LaserWriter encountered the problem. I opened Word again and printed the file from the System 7 Mac and captured the packets with LocalPeek. I also created a PostScript file for reference purposes. You can do this by using the destination checkbox in the “Print” dialog. This PostScript file is created by the LaserWriter driver and contains all the information that would be sent to the printer—and a bit more.

The PostScript file, which is 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, all the user dictionaries, etc. Some of this information, however, is already stored in the BLP, and the LaserWriter Driver will try 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 will ask which fonts are in the printer’s memory, and download only the fonts in the document that the printer doesn’t already have. 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 just easier to read because the PostScript instructions are all in one place instead of being spread out over hundreds of packets.

To get a “feel” for the print job, I loaded the PSF into Word and numbered the lines (“Renumber...”, found in the Utility Menu). There were 7840 lines of PostScript code. Then I replaced all instances of “exch” with “exch”. This changes nothing, of course, but it let me know that there were 292 instances of “exch”, the command that caused the stack underflow error. Then I got an idea of where the document pages fell in the PSF by searching for “Page 1”, “Page 2”, etc. This will show where the information for a specific page begins. The pages came out like this:

<u>Instructions For:</u>	<u>Started On:</u>	<u>Instructions For:</u>	<u>Started On:</u>
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 I suspected that the problem was on Page 2, I selected the lines 1210-1764 in the PSF, and counted the number of instances of “exch”. This showed me that there were only 7 instances. When I looked to see where “exch” was located in the description for Page 2, I 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

```

TROUBLESHOOTING MACINTOSH NETWORKS

```
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. 00000000000000000000000000000000
1269. 00000000000000000000000000000000
1270. FFFFFFFFFFFFFFFFFFC0000000000
1271. FFFFFFFFFFFFFFFFFFC0000000000
1272. FFFFFFFFFFFFFFFFFFC0000000000
```

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 a 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*, I saw that Silicon Beach’s SuperPaint 2.0 was listed as “Mostly Compatible”, while Version 2.0a was listed as “Fully Compatible”¹. Time to update the current theory.

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 7 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 I had identified this graphic on Page as being a problem, I wondered whether there were other graphics that might have a printing problem. To investigate, I looked through the PSF to find out if there were any other instances of “Silicon Beach”. To my surprise, I 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 that is 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
```

⁵ A SuperPaint 2.0a header begins: % P2-v15-Copyright 1988 Silicon Beach Software, Inc.

```

667.  {lineto}b/bx{closepath}b/bp{gsave eofill grestore}b/bf{scale 1
      setlinewidth stroke}b/be
668.  {grestore}b/p{/gf false def}b/g{/gf true def}b g pk{/_pat/pat load
      def/_gr/gr load def}{/_gr
669.  div setgray}b}ifelse sk{/_sTM/setTxMode load def}if/gx{/tg exch
      def}b 0 gx end P2_b pk
670.  end{/pat{P2_b gf{end pop sg}{/_pat load end exec}ifelse}bind
      def}{/pat{P2_b pop _gr end}bind
671.  def}ifelse P2_b sk end{/setTxMode{P2_b/_sTM load end exec P2_b
      tg/_gr load end exec}bind def}
672.  {/setTxMode{pop P2_b tg/_gr load end exec}bind def}ifelse}if
673.  0 pen
674.  74 gm
675.  73 lin
676.  29 1 index neg 1 index neg matrix translate 3 1 roll
677.  currentpoint 2 copy matrix translate 6 1 roll
678.  104 gm
679.  105 lin
680.  71 currentpoint 1 index 6 index sub 4 index 9 index sub div
681.  index 6 index sub 4 index 9 index sub div
682.  matrix scale 11 1 roll
683.  [ 9 1 roll cleartomark
684.  2 roll matrix concatmatrix
685.  exch matrix concatmatrix
686.  /P2_tp exch def P2_b mk end{bn}if
687.  1 pen
688.  -257 gm
689.  -258 lin
690.  T 32 43.90908 73 150 6 33 46 T 1 db
691.  000000000000
692.  000000000000
693.  000000000000
694.  07FFFF000000

```

To my eye, this graphic definition from Page 1 is very similar to the graphic definition on Page 2 that was my suspect. Looking further, I found there were 3 more instances of a SuperPaint definition on Page 1. Now I wasn't sure about my SuperPaint theory. I had suspected the error to be on page 2, because PrintMonitor's error message said that there were 8 pages (of 9) left to print. It seems I should have suspected that the error was on page 1 since no pages were printed. I made a mental note to base future hunches on the number of pages printed rather than on the number in the status window. Then I remembered that since the printer has a buffer, it may have been that the printer was discovering the error in Page 1 while the Mac was loading the instructions for Page 2.

If I stopped here, I still would not have been able to accomplish my goal—the user would still need a System 6 LaserWriter Driver to print the documents, and I didn't want to have a mixed network. I knew that if I looked into the packets, I could discover more information that might help me create a better solution. In order to decide whether to continue, I made a list of all the things I didn't know.

The first unclear item was that my whole troubleshooting analysis was based on reading the PostScript file, which is not 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 me more precisely where the offending “exch” occurred. I would find out whether all of the Silicon Beach definitions were sent, or whether only portions of all of them were sent, or whether just one was sent or whether none were sent.

The second unclear item is that I had no idea how much data had been delivered before the BLP discovered and reported the error. Looking in the packets, I would be able to see the exact moment that the

TROUBLESHOOTING MACINTOSH NETWORKS

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, I decided to continue. Perhaps in finding out what I didn’t know, I would also discover a better solution.

Examining the Printing Packets

When you first approach the printing packets, it’s good 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 phases are covered in more detail in “Processes-Printing Process Description”, I’ll repeat some of that information here and show you how to gain a feel for the print job.

The entire printing session is driven by the LD, which:

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, # of copies)
7. Sends the document characteristics
8. Downloads any additional fonts needed (may occur in body of print info)
9. Sends the actual page descriptions
10. Closes the printing session

When I captured the packets, I used address filtering to capture only those packets that were sent by the BLP or the Portable. If I hadn’t done that during capture, I would filter all the other packets out at this time to make the print session easier to follow.

The unsuccessful print session used 221 packets and lasted 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 you can use to find them in your own print captures. The initialization info is shown as N/A since it was not 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
Re-start 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	Cancelled	N/A	Page 3

Session Ends 221 17.2 \$070000

Now that I knew the page breakdown, I 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 I was trying to print, I saw that there were four graphic objects in the instructions for Page 1 and three graphic objects in the instructions for Page 2. I 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 cancelled before it could be sent.

Question: Why did the BLP allow 6 SuperPaint graphics to be sent before reporting the error?

It occurred to me that since the 2MB BLP has a buffer of some size, it may have just 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 5 graphic objects were sitting in the buffer waiting to be processed. On the other hand, maybe each of the graphic objects have different characteristics and some characteristics are compatible with the System 7 LaserWriter Driver and some are not.

I tested the 7 graphics on the first 2 pages individually by copying them into a document by themselves and printing them one by one. Of the 7 graphics, 4 could be printed and 3 could not. I looked at their characteristics to see if I could 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), I took snapshots of all of the Draw objects, and then replaced them with their snapshots. Then, the two pages 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, Figure 1.4, but allowed 2 more graphics to be sent before reporting the error.

Conclusion: It Pays to Keep Going

This is an example of continuing the troubleshooting process past the point of the first “solution”. There were 2 places in the investigation where we could have called the job complete, but in each instance 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 gave us the solution of “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 has the unfortunate side effect of creating a network with mixed LaserWriter Drivers which means that the LaserWriter will frequently need re-initializing. While there are some Laser printers that re-initialize very quickly, most printers take at least a couple of minutes to do this. Because the user’s workstation is engaged in the process of re-initializing, this solution is not very attractive; most users don’t want to wait that long. What some network managers will do at this point is to designate one Laser printer a System 6 device and another a System 7 device.

Continuing the investigation, we discovered the solution of, “Get an upgrade to SuperPaint 2.0a and everything will be fine.” This of course is unattractive because of the time delay between ordering the upgrade and receiving the upgrade. If the files needed to be printed right away, this could pose a problem.

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 upgrade arrives. We could continue to explore the parameters of 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 one. We used PostScript Reference manuals, Disk Editing tools, packet analyzers, Apple manuals, etc. This is typical of a problem that occurs in the higher layers of the protocol stack. PostScript is a Presentation Layer (Layer 6) protocol that is dependent on many lower layers below it in order to function properly. In many printing problems, it will be one of those lower layers that is causing the problem, but in this case, it was a PostScript language incompatibility problem.

As I indicated in the Preface, networks involve many components in complex interaction with one another. No person can remember everything they need to troubleshoot problems like this one, so it is important to have a range of books available to you like the ones mentioned in the Bibliography. You don’t need to read them all when they first arrive, but you should get familiar with what each book provides in terms of knowledge and approach.

LocalTalk Parameters

Speed	230.4 Kbits /sec
Media Access Control	Collision Avoidance
Signalling	FM-0
Bit Time	4.34 microseconds
Maximum Transfer Unit	600 bytes
Minimum Transfer Unit	3 bytes

Ethernet Parameters

Types of Ethernet Media:

<u>Specification</u>	<u>Type</u>	<u>Cabling</u>	<u>Connector</u>
10BASE5	Thick Ethernet	Thick Coax	AUI
10BASE2	Thin Ethernet	Thin Coax	BNC
10BASET	Twisted Pair Ethernet	Unshielded T.P.	RJ-45

Pinouts:

<u>Pin</u>	<u>AUI</u>	<u>10BASET (RJ45)</u>	<u>AAUI(Apple Attachment Unit Interface)</u>
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 only has 14 pins)	
Shell	Ground	Ground	

Ethernet Manufacturer Designators

00-00-0F	NeXt
00-00-10	Sytek
00-00-18	Webster

APPENDIX B: ETHERNET PARAMETERS

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
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

TROUBLESHOOTING MACINTOSH NETWORKS

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 registrated)
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)
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

APPENDIX B: ETHERNET PARAMETERS

08-00-26	Norsk Data
08-00-27	PCS Computer Systems GmbH
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 Comm. Assoc.
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
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)(see 0A00)
0201	Xerox PUP Address Translation (conflicts...)(see 0A01)
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) Compatability
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
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 Grahics 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
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

TROUBLESHOOTING MACINTOSH NETWORKS

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 Comm. Assoc. Inc. (DCA)
80C6	Pacer Software
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

APPENDIX B: ETHERNET PARAMETERS

8108	Symbolics Private
8109	Symbolics Private
8130	Waterloo Microsystems Inc.
8131	VG Laboratory systems
8137	Novell, Inc.
8139-813D	KTI
814C	SNP over Ethernet (see RFC1089)
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

Packet Glossary

Introduction-How to Use this Section

The purpose of this Appendix is to provide a reference guide to AppleTalk packets so that when you are using a protocol analyzer, you can easily refer to this Appendix to find out what the packet is, what it does, when it is sent, and what the different components of the packet mean.

This section is only meant to refresh your memory and provide the particulars of a protocol process, it is not intended to provide a tutorial to how the protocol works. That job is accomplished elsewhere. As a result, some of the terms used in this section are not defined in this section, but are defined elsewhere in the book. This material supplements the process descriptions in the “Processes” and in “Troubleshooting by the Layers”.

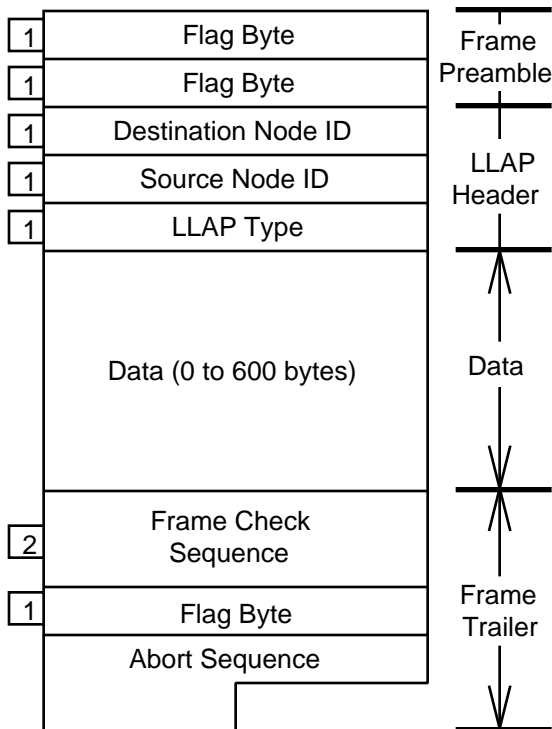
Data Link Formats-LLAP, ELAP

Most of the time, Data Link packets provide the capsule in which the rest of the protocol information is carried. While the format of the packet is specific to the type of network that carries it, the format of all “LAP packets”, regardless of network type consists of the following elements:

1. A preamble section or “frame header” that acts a delimiter for the packet. The preamble, besides signalling the beginning of the packet, synchronizes the receiver’s clock to the sender’s clock by sending a special bit pattern. The receiver’s hardware synchronizes to this pulse. The preamble (flag bytes in LocalTalk) is not viewable by protocol analyzers or any other computer-based tool because the hardware strips this part of the data frame prior to any software analysis. If you want to see the preamble, you’ll need to use an oscilloscope and analyze the raw waveform.
2. A destination address field that indicates the intended receiver. This address may be to particular node (directed transmission), to all nodes (broadcast) or to a specific group of nodes (multicast). Also, note that the destination address in the Data Link portion of the packet is only the destination device for that particular network. The ultimate destination of the packet may be in a different network.
3. A source address field that indicates the identity of the sending device. Like the destination address, the source address is only the source address for that network; the actual source may be from a different network.
4. A Data Link type that indicates to the receiving device how to process the packet. For LocalTalk, Apple has defined meanings for 6 type designations (LLAP type). For example, if the type field is “1”, then the packet is a “short DDP packet” and the receiving device will process it accordingly. In addition to Apple-defined types, some third party developers have designated particular types developers have designated certain types for their use. An example is Shiva’s use of LLAP Type “75” for FastPath diagnostic packets. These non-Apple LAP types are not standardized or coordinated among the developers. third party developers avoid LAP type conflicts only through the informal process of trial and error. In Ethernet, in lieu of a LAP type, the protocol type indicator is the equivalent of the LAP type.
5. A data field. While the rest of the packet is a capsule, this field holds the contents of the capsule. In the capsule will contain information for protocols above the Data Link Layer

- (Network Layer and higher) as well as any application data being sent. An empty data field indicates that the packet is providing a Data Link Layer control function.
6. A frame check sequence is provided on network links by a process in the signaling hardware. The signaling hardware in the sending device calculates and attaches a mathematically generated number based on the contents of the packet. That number is also calculated, using the same mathematical method, by the receiving device. The receiving device will then compare the number it calculated to the number it received. If the calculated number matches the received number, the packets “pass” and are turned over to the higher layer protocols for further processing. Both Ethernet and LocalTalk both use the CRC method for the calculation. CRC errors in those networks indicate that the numbers did not match and that the contents of the packet were corrupted in transmission.
 7. An abort sequence on the end of the packet signals the end of the data frame. The pulse of the abort sequence also provides a check that the synchronization received in the preamble is still “on the beat”.

LLAP Format



A minimum of two flag bytes (bit sequence “01111110”) begin every LLAP Packet. Destinations and source addresses are 1 byte long and contain the AppleTalk node address of the recipient. The AppleTalk node address is sufficient to identify the recipient because a LocalTalk network only contains one network number and because hardware addresses do not exist in LocalTalk.

In packets carrying data, the LAP type indicates the nature of the information following, however four Apple-defined LAP types exist for two pairs of LLAP control functions. They are:

- | | | |
|------|-----------------|---|
| \$81 | NodeEnq | Used to check tentative node address |
| \$82 | Node Ack | Used to indicate that tentative address is already in use |
| \$84 | Request to Send | Signal to receiving device that a data packet will follow |
| \$84 | Clear to Send | Intended receiver signals that it is ready to receive |

TROUBLESHOOTING MACINTOSH NETWORKS

LAP Types “\$01” and “\$02” indicate short and long DDP packets, respectively, and are explained below in the DDP section. Other LAP types are defined by third party developers and are specific to their software.

LLAP uses a 16-bit CRC method for the Frame Check Sequence. The 16-bit CRC method has a probability of only 1 in 65,535 that a packet will pass the CRC check despite having corrupted data in the packet. CRC errors are almost always caused by poor network cabling conditions. A ratio of 5 CRC errors per 10,000 data packets indicates that an examination of cable quality is in order.

The flag byte on the end of the packet provides a synchronization check. Overruns and underruns are synchronization errors that indicate that the packet did not remain synchronized over its length. There are two possible meanings for these errors—either the receiving or the sending hardware’s functioning was interrupted momentarily by another computer process during packet transmission (non-network error) or the integrity of packet was compromised due to cabling conditions.

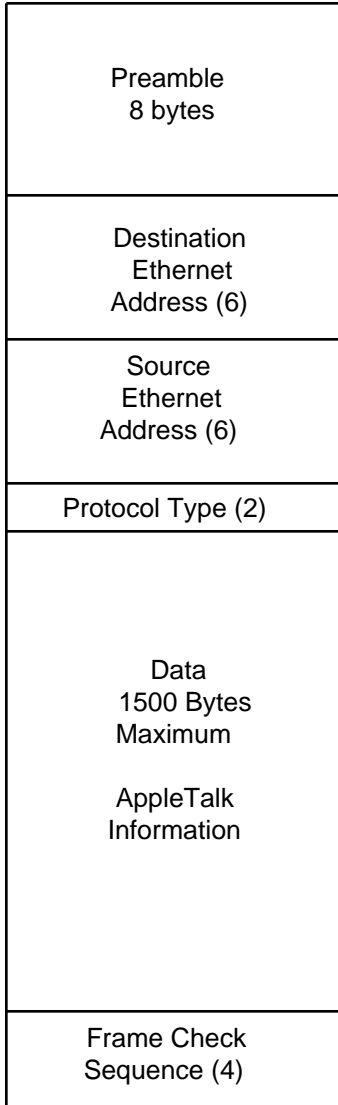
The abort sequence ends the LLAP frame.

ELAP Format

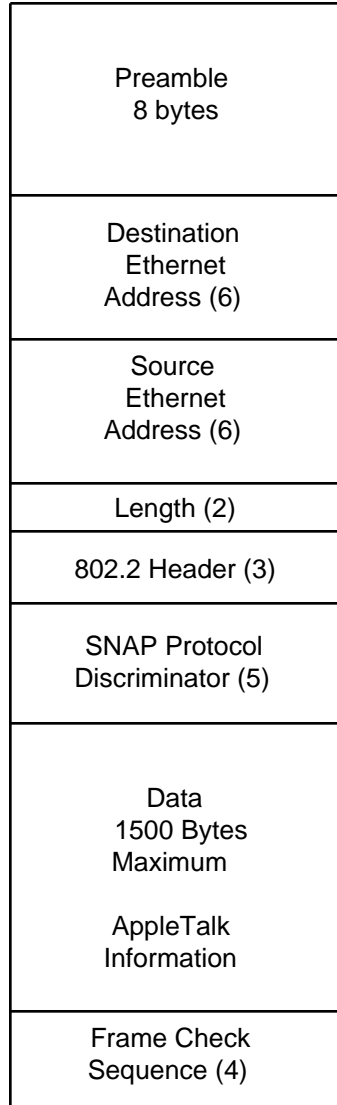
There are currently two versions of “Ethernet” frame formats in use—Ethernet and IEEE 802.3. They have many features in common—enough to co-exist on the same network—yet most devices can only accept one format and not the other. In AppleTalk, Ethernet packets are used by Phase 1 devices, who will reject 802.3 packets. Phase 2 devices use 802.3 packets and reject Ethernet frames. The sole exception is a “transition router”, whose job is to route packets between Phase 1 networks and Phase 2 networks.

Unlike LocalTalk, which can only carry AppleTalk protocol data, Ethernet (in either format) is a general purpose network that can carry many types of protocols, including AppleTalk. Because this, Ethernet frames also include a field that indicates which protocol (such as AppleTalk, DECnet, TCP/IP) is carried inside the frame. The 802.3 format includes an additional header section, called the 802.2 header, that is standard format for all IEEE 802.X packets.

Both formats use an 8-byte preamble to mark the beginning of the frame followed by the destination and source Ethernet hardware addresses. After the source address, the two formats diverge.



Ethernet Frame
Phase 1



802.3 Frame
Phase 2

Ethernet Format-Phase 1

The Ethernet format follows the source Ethernet address with a 2-byte protocol field that indicates the AppleTalk (\$809B) protocol type or the ARP (\$80F3) protocol type.

The data field can contain up to 1500 bytes, however AppleTalk packets will use considerably less than this due to the 600-byte limit of DDP (explained below). Ethernet packets must contain a minimum of 64 bytes (not counting the preamble, but including the Frame Check Sequence). If AppleTalk does not have enough information to place enough data in the data field to meet the minimum size requirement, the data field will be padded with meaningless data to make the packet 64 bytes long.

The Frame Check Sequence uses a 32-bit CRC method; the probability of corrupted data passing the CRC test is less than 1 in 4.32 billion.

802.3 Frame Format

Following the Ethernet source address is the 2-byte length field. The length encoded here refers to the length of meaningful data in the data field (number of bytes), not counting any padding.

Next is the 3-byte 802.2 Header, which doesn't contain an information useful to troubleshooters—it is a required field for the 802.3 format that indicates that the protocol inside is not an OSI protocol and that the frame uses SNAP Type 1 format, which is an 802.3 sub-type.

The SNAP protocol discriminator is 5 bytes long and ends in \$809B (indicating AppleTalk) or \$80F3 (indicating AARP), similar to the function of the protocol type field in Ethernet.

The data field, like in Ethernet, carries the AppleTalk protocol information and data. The 802.3 has the same restrictions on minimum and maximum packet lengths as Ethernet (above) and will be padded as necessary to meet the 64 byte limit.

DDP Limitation

As of this writing, all AppleTalk data is carried through the internet by DDP, which has maximum size of 600 byte. Because of DDP limitations, EtherTalk packets (Phase 1 or Phase 2) never use the full 1500 data bytes allowed by Ethernet.

AARP Packets on Ethernet

There are 3 types of AARP packets—probe, request and response. The probe is only used for address acquisition purposes; it is used by a device as it initializes its AppleTalk protocol software to test the uniqueness of its tentative address. The probe packet asks, “Is anyone using AppleTalk address X?” The request packet is very similar; it asks, “Who has AppleTalk address X?” The request packet is used when AppleTalk has a need to send to a device with an AppleTalk address of X, but the sender does not know the Ethernet address of the AppleTalk device.

The response packet answers both probes and responses with “I am using AppleTalk address X.” If the AARP response answers a probe, it discourages the probing device from using address X, and the prober will pick a new address and probe again. AARP responses that answer a request allow the requester to add a new entry to its AARP AMT table because the response packet carries the AppleTalk address-Ethernet address pairing.

Because AARP requests are only used when the AppleTalk protocols need to send information, a successful request-response pair of AARP packets should be immediately followed by an AppleTalk packet of another sort from requester to responder.

Differences Between Phase 1 and Phase 2 AARP

Aside from differences in the Ethernet frame type, the main difference between Phase 1 and Phase 2 comes from the use of extended networks in Phase 2. In Phase 1, because there was always only one network on an Ethernet, expressing AppleTalk addresses in terms of node address alone was sufficient. In Phase 2, however, the AppleTalk address must include both network and node components to specify the device. Luckily, Phase 1 left a few bytes unused and reserved for that purpose, so the packet format was left unchanged.

Ethernet Frame Header
Hardware Type (2) Ethernet = 1
Protocol Type (2) AppleTalk = \$809B
HW Address Length (1) = 6
Protocol Length (2) = 4
AARP Function (2) Req = 1, Rsp = 2, Prb = 3
Source Ethernet Address (6)
Source AppleTalk Address (4)
Destination Ethernet Address (6)
Destination AppleTalk Address (4)

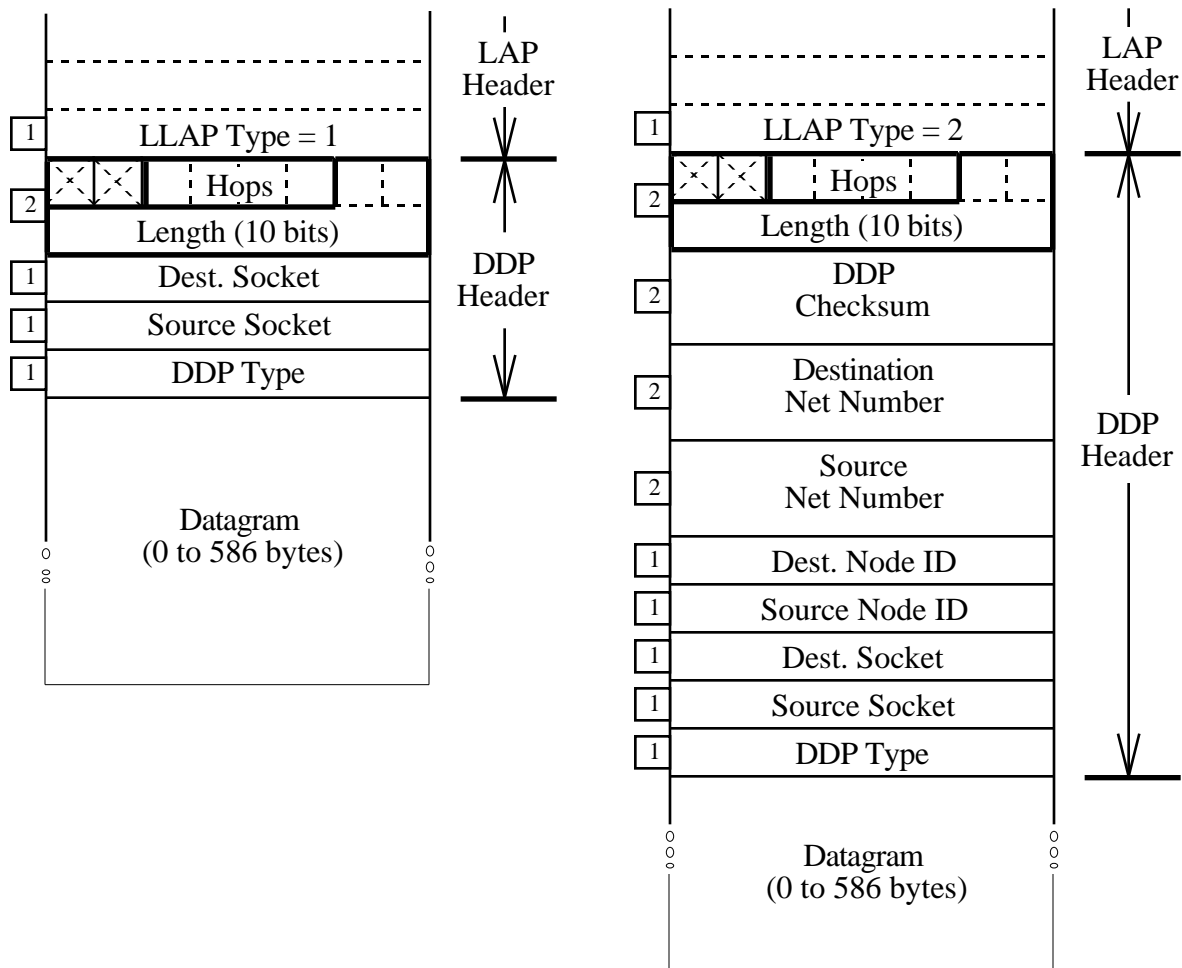
Both formats reserve 4 bytes for AppleTalk address fields. Phase 1 uses only the last byte of these four bytes for the node address, while Phase 2 uses the last 3 bytes to the network (2 bytes) and node address (1 byte).

The destination Ethernet address is left blank (all bits set to “0”) in both the probe and request packets.

DDP Packets

LAP headers like LLAP and ELAP only deliver node to node on a single network. By itself, the LAP header does not have enough information to 1) deliver data to a particular software process within a device or 2) to guide a packet to another network. These are the 2 capabilities that DDP provides by encoding the full Internet Socket Address of both source and destination nodes.

There are two DDP formats-long and short. The short format can be only used on non-extended networks, such as LocalTalk, when source and destination devices share the same network. In this case, DDP only needs to add socket addresses to the LAP information. In all other cases, the long DDP format must be used and the full ISA is encoded for source and destination nodes. The long format is always permissible, even in situations where a short DDP format could be used.



In LocalTalk packets, the last byte in the LLAP header is the LLAP Type, which indicates whether the long or short DDP formats will be used. There is no equivalent ELAP Type. Phase 2 Ethernet packets will always use the long form, and although I can't find any documentation that says this must be true, I've never seen a Phase 1 Ethernet packet use the short format, either.

The 2-byte length field carries both the length of the DDP portion of the packet (header and data) and the hop count of the packet. The hop count indicates how many routers have already forwarded the packet. With only 4 bits reserved, the hop count is limited to 15. This is a safety feature that allows the sixth router in the chain to throw away the packet in the case of a packet traveling in a closed loop that can result from some router configuration errors. For example, a closed loop will result if router A think the best way to get to network X is through router B and router B thinks the best way is through Router A.

Ten (10) bits are reserved for the length descriptor, which would allow DDP to specify a DDP length of 1024 bytes, however, DDP is currently limited to 586 bytes.

Short DDP Format

In the short format, DDP adds the information of source socket, destination socket and DDP Type to the LAP information. Souce and destination socket address are necessary to identify the particular software processes that are sending and receiving the data. The DDP type is explaine at the end of the description of the long format.

Long DDP Format

The long DDP header carries a 16-bit checksum, but this is normally not used on networks that include a CRC method in their LAP frame. Checksum-style error checking is less rigorous than the CRC method and so does not add any integrity to the frame.

The long format carries the full Internet Socket Addresses of both the source and destination devices. Note that these are potentially different than the destination and source addresses found in the LAP header.

DDP Type

The DDP type field is found in both long and short formats. Like all type fields, it indicates to the software “how to process the information that follows”. DDP types 1, 2, 4 and 6 are anlg0

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. Referring to the ability of a computing device to use data or resources beyond its native capabilities.

Access Method. The type of Media Access Control method that a node uses to gain control of a network.

Accuracy. Referring to 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 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. Referring to a group of printable characters that includes the letters of the alphabet in both upper and lower case, the numerals plus a limited group of additional symbols and punctuation marks.

Ambient. Referring to 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.** Referring to 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. c.f. Digital
- Anomaly.** An unusual instance or circumstance.
- Apple.** Referring to the company, products and policies of Apple Computer, Inc.
- AppleShare.** An application published by Apple that allows a Macintosh to be a file server using the AFP protocol.
- AppleShare PC.** Obsolete: An Apple application that allowed a DOS computer with the appropriate hardware adapter to use AppleTalk protocols and provided 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 (obsolete).
- Application.** An independently executable set of algorithms and data structures that perform a specific set of functions.
- API.** Application Program 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 sum total of all of the specifications, protocols and implementations that define a particular networking system.
- Archive.** A storage of infrequently used or historical data.
- ASCII.** General: Referring to 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 having to conform 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.** In a computer, a task that is executing 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.

TROUBLESHOOTING MACINTOSH NETWORKS

- Balun.** A device that links together dissimilar wire types and attempts to minimize any negative effects to the signal that would normally 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 in one set of circumstances to another.
- Binary.** 1) A numerical system using “2” as its base. 2) Data that is encoded or presented in machine-readable form (1’s and 0’s).
- 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 that performs a function using mechanisms that are unimportant or impossible to understand.
- 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 which 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.
- Router.** A device which 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.** Refers to the practice of automatically including an additional application or capability in the sale or delivery of a computing component that is not ordinarily associated with that 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.** Referring to a system in which upper case letters are not differentiated from their lower case form.
- Case Sensitive.** Referring to a system in which upper case letters are differentiated from their lower case form.
- CDEV.** The designation of 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 where the two have different roles in the relationship. Typically, the client computer 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 insure 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 synonym for a multi-port repeater.
- Conductor.** The current-carrying component of a transmission cable, typically a copper wire.
- Connection-oriented.** In data networks, a type of computer relationship in which the network equipment constructs a circuit between the two devices for the duration of their relationship. 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.

TROUBLESHOOTING MACINTOSH NETWORKS

- Connectionless.** The opposite of connectionless, a type of relationship between two devices where each information packet must contain the address of the partner device.
- Connectivity.** A term referring to 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, a software application that has the ability to control an aspect of system configuration, such as the pixel depth of the monitor (Monitors), the choice of network type (Network), the desktop pattern (General Controls) or the manner in which directories are displayed (Views).
- 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 IIfx, for example, a Motorola MC68030 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 will self-terminate 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 CRC's 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, once started, is not under user control, but continues to run 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 normal 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.
- Data Base.** A collection of data that can be selectively retrieved by a type of application known as a Data Base 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 its 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 the state of a computer or network's health.
- DOS.** The operating system of 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.** 1) 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 original 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.

TROUBLESHOOTING MACINTOSH NETWORKS

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, it may run 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 the 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. 1) 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 both 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 OS 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 MBits/sec. token passing ring implemented on fiber optic cabling.

Female connector. Also called a "jack". A connector which joins with a "male connector" by providing recesses into which the male connector inserts its contact points or pins.

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. Refers to 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 sometimes 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 either is or is not 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/sec.
- Format.** A specification for the arrangement of information. Examples include the format of disk, file or packet.
- Frame.** In data networks, the information packet and all of the preceding and succeeding signals necessary (flag bytes, preambles, frame checks, abort sequences, etc.) 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 type of 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 system.
- 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 or 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.

TROUBLESHOOTING MACINTOSH NETWORKS

Hertz (Hz). A unit of measure of the frequency of cyclic wave form, 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 will test 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 multi-port network repeaters, usually of twisted pair networks such as 10BaseT 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. Usually International Business Machines. Can also refer to acronyms such as "inferior but marketable".

Icon. A pictographic 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 charged with the responsibility of coordinating standards at the Data Link Layer. The committee also oversees the work of many sub-committees 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. Archaic. A Macintosh System Extension.

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. In statistical analysis, 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 device that is currently active.
- 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.** A unit of measure used to describe the rate of data transmission equal to 1000 bits per second.
- KByte.** A unit of measure used to describe an amount of information equal to 1024 (2^{10}) bytes.
- Kludge.** A descriptive word used to describe a solution to a problem that lacks elegance or that contains components for a purpose significantly different than their original design purpose.
- LAN.** A communication infrastructure that supports data and resource sharing within a small area (<2 km diameter) that is completely contained on the premises of a single owner.
- LAN Server.** A general term used to describe 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 buffered memory (such as a bridge or router) before it can be sent along its path.
- Layer.** A term used to describe a group of communication functions and the protocols implemented to perform them as defined by a network standards organization, most often referring 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 that computer.
- Logout.** A formal procedure where a computing device severs its connection to another computing device.

TROUBLESHOOTING MACINTOSH NETWORKS

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. A unit of measure used to describe the rate of data transmission equal to one millions bits per second.

MByte. A unit of measure used to describe an amount of information equal to 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 IIx, the Macintosh Classic II or the Macintosh IIfx.

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 with the ability to 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 a one millionth part 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 part of a unit of measure, such as a millisecond or millimeter.

Minicomputer. Archaic. A multi-user computer capable of supporting 4-16 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 Varistor. A device that acts as a surge suppresser by forcing transient high voltages to ground.

Multitasking. A descriptive term for 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 term used to describe 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 a 1 billionth portion of a unit of measure, as in nanosecond or nanometer.
- Native.** Something is 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) Referring to 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 may not be separated 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 that may be 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 that is standard issue on nearly every computer that uses TCP/IP protocols as its standard network protocols.
- Nibble.** One-half of a byte, which can be 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 architecture's 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.

TROUBLESHOOTING MACINTOSH NETWORKS

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. Referring to the subset of the ISO that promotes and defines standards for open networking systems.

OSI 7-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 7 distinct categories (Layers) that act successively on data as it makes its way between the user and the transmission media.

Out of Band. Referring to 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 1) has not properly allocated for that purpose or 2) has been allocated by another process.

Packet. A discrete chunk of communication in a pre-defined format.

Padding. Additional, 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 3 or more pairs of network wires are joined in a single location.

PDS. Processor Direct Slot. One of a large number of computer bus architecture's 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, 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 that is accessed not by a network system, but by serial or parallel connections, bus architecture's 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. A synonym for Hardware Address.

Pico. A prefix denoting a 1 trillionth portion of a unit of measure, as in picosecond or picofarad.

Pin. A type of electrical contact point for a single conductor.

Plug. A synonym for male connector.

- Poke.** 1) A term used to refer to the transmission of a packet on a network for test purposes only. 2) 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.** A term used to describe the orientation of a differential voltage.
- Polling.** A means of Media Access Control where a device may only transmit information 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 such port is connected to the devices other ports through the device's internal electronic structures.
- PostScript.** An extensible programming language that is capable of describing and drawing complex images that was developed by Adobe Systems.
- Precision.** Referring to the smallest difference in measurement that a test instrument can distinguish.
- Print Spooler.** A software process that accepts a print job from a workstation as if it were a printer and then sends the print 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, as in a computer program or application that is currently active, i.e. consuming CPU time and memory resources. While an "application" is a process, that term usually refers to a process that a user can launch from the Finder and directly control, where the use of 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 in" to its internal circuitry. The designation "EPROM" indicates that the program is Erasable.
- Proprietary.** Meaning that information concerning the methods or implementation of a technology are owned by an individual or a company. "Proprietary" may mean that the information is secret or that the information 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.** (or Protocol Suite or Protocol Family) A group of protocols, usually specified by a single vendor or organization, that are implemented at more than one Layer of the OSI 7-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, where four conductors (two pairs) are all twisted together with no independent twisting between the pairs.
- Query.** A general term to describe the formation, usually in a prescribed format, that a computer process would use 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.

TROUBLESHOOTING MACINTOSH NETWORKS

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 only be accessed after accessing memory locations 1-5.

Reboot. (or Restart or Reset) A user activity where the user boots a computing device without interrupting its source of electrical power. The goal of this activity 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. AppleTalk Remote Access.

Repeater. A Physical Layer device which restores, amplifies, re-clocks or otherwise improves a network signal that it receives 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, when pushed, will restart the Macintosh.

Resistor. A passive electrical device that adds resistance to a circuit.

Resource Fork. In a Macintosh file, the portion of the file that contains such auxiliary information as the menus, the 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 information it has 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 mouseclicks and key strokes and takes over the system during prolonged periods of idle activity to cover the screen with a display that will prevent damage to its 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 max.) 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 suitable for an 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 on-going relationship between two computing devices involving the allocation of resources and sustained data flow.
- Session Layer.** The layer in the OSI 7-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 with each other.
- 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 the ambient radiation and other signal disturbances that are present, 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 that is constructed by connecting computing devices to a common device.
- Switch.** A switch is 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 only transmit 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 provides basic functionality like file management, visual display and keyboard input and is used by application software to accomplish these functions.
- T1.** A telecommunications technology useful in wide area networks (WAN) 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.

TROUBLESHOOTING MACINTOSH NETWORKS

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 that was installed to support a telephone system.

Telephony. The science and practice of telecommunications.

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) Any of a number of T-series programmable androids optimized for security-related assignments, usually with external features similar to that of a human. 2) A resistor placed at the end of a bus to prevent the reflection of signals.

Thick Ethernet. Also 10Base5. A system for constructing an Ethernet using coaxial cabling approximately 1/2 inches in diameter.

Tightly Coupled. A relationship between two processes where the rate of accomplishment and success of the two processes are highly inter-dependent.

Token Passing. A MAC method where stations may only transmit 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 for the purpose of aiding a human in accomplishing a goal.

Topology. 1) The arrangement of computing devices in a network. 2) A term or short phrase describing such an arrangement.

Trailer. The group of bytes that marks the end of a frame and usually contains an error checking mechanism such as a CRC.

Transceiver. 1) In Ethernet, an electronic device that transforms signals between a node's internal circuitry and the Ethernet signals and also detects collisions. 2) Any device that can simultaneously transmit and receive.

Transfer Rate. The rate at which data is transferred from one device to another, usually expressed in bit per second 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 7-Layer Model that is concerned with management of the data flow between source and destination.

Transport Services. Any of 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 the user launches the Trojan Horse on his own system. 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 is to follow.
- UNIX.** A multi-tasking, multi-user operating system invented by Bell Labs that is used on many type of computer systems.
- Upload.** The activity of transferring a file from a user’s computer system to a remote system.
- UPS.** Uninterruptable Power Supply. An electrical supply system that conditions electrical power for a computer system and will allow continued operating in the event of a power failure.
- User.** A person who uses a computer system to accomplish a non-computing goal, 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 collection of display symbols and other sensory stimuli made by a computer that present information to a human and the collection of physical action that a human can take to present data to a computer.
- UTP.** Unshielded twisting pair wiring.
- UTP Ethernet.** 1) A synonym for 10BaseT. 2) Any of a number of Ethernet technologies in use before the adoption of the 10BaseT standard that 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 quality used to describe a situation where 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 indicating that the wiring is unsuitable for network data transfer.
- Volt.** The measure of a difference in electric potential.
- WAN.** Wide Area Network. A network that is created between and 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 in intensity, frequency or velocity with time.
- Well Behaved.** A condition that exhibits expected properties.

TROUBLESHOOTING MACINTOSH NETWORKS

Wireless. Any system that provides communication without the use of wires.

Wiring closet. In wiring for telephone or data systems, any location from which the communication wires emanate, usually enclosed.

Workgroup. A group of networked computer users who frequently communicate with each other and share common devices.

10BaseT. A specification of the IEEE 802.3 committee (802.3i) for the implementation of 10 MBit Ethernet on unshielded twisted pair wiring.

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
- 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*, second 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*. Addison-Wesley, 1987
- Belden Wire and Cable Guide*. Product Catalog, Yearly
- Freedman, Alan. *Electronic Computer Glossary (HyperCard Stack)*. The Computer Language Company, 1992
- Holzgang, David W.. *Programming the LaserWriter*. Addison-Wesley, 1991
- 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
- Rose, Marshall T.. *The Simple Book*. Prentice Hall, 1991
- Stallings, William. *Handbook of Computer Communications Standards*, Vol. 2, second edition. Howard W. Sams, 1990
- Tannenbaum, Andrew S.. *Computer Networks*, second edition. Prentice Hall, 1988