Contents

Chapter 1

Troubleshooting Methodologies

1.1 General Considerations in Troubleshooting

Overview This study of troubleshooting begins by looking at a methodology that breaks down the process of troubleshooting into manageable pieces. This permits a systematic approach, minimizes confusion, and cuts down on time otherwise wasted with trial-and-error troubleshooting.

Troubleshooting networks is more important than ever. As time goes on, services continue to be added to networks and with each added service comes more variables. This adds to the complexity of the network troubleshooting as well as the network itself. Organizations increasingly depend on network administrators and network engineers having strong troubleshooting skills.

Note: Network engineers and network administrators have distinct job functions, but for the purposes of this curriculum, the term network engineer is the best term for describing the type of work described herein.

Troubleshooting is arguably the process that takes the greatest percentage of a network engineer's time (and also has the potential of reaping substantial financial rewards). Any procedural tools that can be used to simplify the process are welcome. Of course, with each procedural tool comes the time required to internalize it. Decisions come down to how much time must be spent 'up front' versus 'in the field'. These types of decisions are not easily made and finding the right balance comes with experience. One of the main goals here is to optimize your time up front to help shorten your time in the field.

Once all of the protocols and product lines are stripped away, troubleshooting is essentially an exercise in logic. Keeping in mind that logic comes in both the deductive and inductive flavors. Whenever you approach a network problem, you should use some sort of problem-solving model (a logical step-by-step method of converging toward a solution). The point should be made that network engineers do not stop and open a handbook on troubleshooting methodology when they get stuck. They work from their own personal skill set and with their own troubleshooting methodology that they have developed over time. The model proposed in 1.2.1 will be reinvented by each and every one of you to suit your particular style. The point is to minimize wasted time associated with erratic hit-and-miss troubleshooting.

1.1.1 Preventing network downtime

Keeping a network up is a primary consideration for IT management. Preventing network downtime is essential. Of course, minimizing downtime once the network is actually down is critical as well. In this case, two main factors come into play:

- A well-designed network with extensive documentation
- Skilled network engineers on staff or on contract ready to go into action

The first bullet refers to the network design piece of the puzzle, which is outside the scope of this curriculum — a good design reference is "CCIE Fundamentals, Network Design and Case Studies, Second Edition".

Note: All titles referenced in this volume refer to Cisco Press books unless otherwise indicated.

Network prevention and maintenance are intimately related to performance and fault monitoring. An excellent reference for this topic is "Performance and Fault Management", written by 5 CCIEs. This book is a comprehensive guide to designing and implementing effective strategies for monitoring performance levels and correcting problems in Cisco networks.

Note: Although half of the battle for a network engineer is prevention, this curriculum focuses only on what to do when preventative measures have failed.

Our focus is simply on using the basic tools available within the Cisco IOS to troubleshoot network problems. This primarily comes down to using the show and debug commands. Chapter 2 will detail some of the software tools used to accompany your troubleshooting efforts.

What makes the minimization of network downtime so critical is saving costs. A company, such as a large ISP, could literally go under if its network is down for a couple days. Imagine a financial services company that provides on-line stock trading having a network down for 24 hours, preventing a hundred thousand clients from conducting trades.

1.1.2 A systematic approach and documentation

A systematic approach to restore a network once it is down is required. A systematic troubleshooting methodology permits a network engineer to build a set of relational pointers, which organize a complex web of details into something workable. The gist of the matter in most troubleshooting scenarios is to move from the general to the specific. Eliminate variables to the point that one can focus on a subset of variables in which the solution is buried. This is a fundamental principle of science, not reserved to network engineering. Large complex problems are solved by breaking them down into smaller chunks and mapping out the interrelationships between the chunks. This makes it possible to extract a total solution once solutions to the smaller problems have been found.

Depending on the person, the hardest part of the problem comes after the problem is solved — documentation! A sample network diagram serves as a focal point for the compiled documentation. This figure is not of course a complete set of documentation for the network, but serves as a key reference for the actual documentation. Documentation should provide clear communication to those who need the information. This includes easy access to the information for these individuals. The documentation should be easy to update as well. Remember — documentation simplifies network management and greatly reduces the time required for problem resolution.

1.1.3 The role of network architectures in troubleshooting

Various network architectures have evolved to satisfy different organizations' traffic flow requirements. Technologies such as Multiprotocol Label Switching (MPLS — see MPLS and VPN Architectures) and Protocol Independent Multicast (PIM) are becoming more and more common and require greater attention to traffic flow patterns within an organization. Traffic engineering is a topic in networking that deals with these issues and has received an increasing amount of press - see for example the book "Developing IP Multicast Networks". With the complexity of today's networks, coupled with the proliferation of multiservice applications, systematic troubleshooting is more important than ever. Figure is an example of one type of network architecture: the Cisco E-commerce network architecture. Another type of network architecture is the IBM Systems Network Architecture.

Note: Many advanced network management software tools are available to aid in the mapping and analysis of network traffic flow. Cisco's NetFlow, FlowCollector, and FlowAnalyzer as well as CiscoWorks2000 Internetwork Performance Monitor are examples of such tools that may prove useful for your work. CCO (Cisco Connection Online at `http://www.cisco.com`) is the best place to look for information on such tools.

Network architectures are inextricably linked with network topologies. The term topology is used frequently in networking, so it's worthwhile to look at a definition of topology. It is the physical connectivity within a computer network which outlines the patterns between processors, memories, and peripherals. This physical connectivity pattern is fundamental to the traffic flow within a network. That in turn is a primary consideration in network troubleshooting. With network design, the importance of being familiar with various network architectures prescriptively conforming to a specific corporate or campus environment is paramount.

As you proceed to troubleshoot a network, one of the first steps is to have a logical picture of your network topology (either in your mind or in front of you). Again, this is where the documentation comes into use. A sequence of thoughts in a troubleshooting process might go something like Figure : "I'm on client K2 connected through a 2900 switch to Ethernet 0/0 on router Everest, which has ATM interface 1/0 connected to the LS1010 on Katmandu. Katmandu connects through the corporate LAN to FastEthernet interface 0/0 on router Tibet. Tibet connects to the private Frame cloud via subinterface serial 0/1.100 pointing via DLCI 100 to subinterface serial 0/0.101 on router Tibet2. Tibet2 is connected to several Ethernet segments, one of which Web server Nirvana resides on. FTP connectivity between K2 and Nirvana is not working, but pings and traceroutes from K2 to Nirvana are successful." A problem such as this probably has nothing to do with the intrinsic topology, but visualizing the topology provides a setting in which to cast the problem. In this case, because you were successful in both a ping and traceroute, the problem is clearly not a network (Layer 3) problem, but an upper layer problem. You may want to collect additional data, to determine if other hosts are having FTP problems with Nirvana or if you are able to access Nirvana via HTTP. The problem could reside in four different areas:

1. the client
2. the server
3. the transmission
4. security policies

Assuming that both the client and server are functional, you conclude that a network device is filtering the FTP application (typically ports 20/21) along the path, thus preventing FTP connectivity. FTPing from various locations throughout the network and inspecting the configurations of the routers along the path will assist in isolating the suspect filter.

Network architectures and topological considerations are ever-present in the work of a network engineer. When a network engineer goes in to solve a problem, one of the first things they'll likely do is get a picture in their mind of the network infrastructure. This provides a setting, a baseline, within which to cast the issue at hand. Of course, the more time that is put into a given network, the clearer the picture of the total network will become.

1.1.4 OSI reference model — a common language

The Open Systems Interconnection (OSI) model provides a common language for network engineers. Having looked at using a systematic approach, documentation, and network architectures, you can see that the OSI model is pervasive in troubleshooting networks. The model allows troubleshooting to be described in a structured fashion. Problems are typically described in terms of a given OSI model 'layer'. By this point in time, you have become intimately familiar with the model. Taking a quick look at the OSI model helps clarify its role in troubleshooting methodology.

The OSI reference model describes how information from a software application in one computer moves through a network medium to a software application in another computer. The OSI reference model is a conceptual model composed of seven layers, each specifying particular network functions. The model was developed by the International Organization for Standardization (ISO) in 1984, and it is now considered the primary architectural model for intercomputer communications. The OSI model divides the tasks involved with moving information between networked computers into seven smaller, more manageable task groups. A task or group of tasks is then assigned to each of the seven OSI layers. Each layer is reasonably self-contained, so that the tasks assigned to each layer can be implemented independently. This enables the solutions offered by one layer to be updated without adversely affecting the other layers. The figure details the seven layers of the Open System Interconnection (OSI) reference model.

The OSI model provides a logical framework and a common language used by network engineers to articulate network scenarios. The "Layer 1", "Layer 2", etc., terminology is so common that most engineers do not think twice about it any more. This is similar to learning a foreign language. Initially you have to think of a word when you are using it the first few times, but later it just rolls out of your mouth.

The upper layers (5-7) of the OSI model deal with application issues and generally are implemented only in software. The application layer is closest to the end user. Both users and application-layer processes interact with software applications that contain a communications component.

The lower layers (1-4) of the OSI model handle data-transport issues. The physical layer and data link layer are implemented in hardware and software. The other lower layers generally are implemented only in software. The physical layer is closest to the physical network medium (the network cabling, for example), and is responsible for actually placing information on the medium.

1.2 The General Problem Solving Model

1.2.1 Steps in the Problem Solving model

A Cisco Connection Online (CCO) document at `http://www.cisco.com/univercd/cc/td/doc/cisintwk/itg_v1/tr1901.htm` describes a general problem-solving model used with Cisco networking. This model is relied on in this curriculum. The remainder of this section is an excerpt from the CCO document.

Figure 1.1 on the following page illustrates the process flow for the general problem-solving model. This process flow is not a rigid outline for troubleshooting an internetwork. It is a foundation from which you can build a problem-solving process to suit your particular environment.

The following steps detail the problem-solving process outlined in the figure to the left:

Step 1 When analyzing a network failure, make a clear problem statement. You should define the problem in terms of a set of symptoms and potential causes.

To properly analyze the problem, identify the general symptoms and then ascertain what kinds of problems (causes) could result in these symptoms. For example, hosts might not be responding to service requests from clients (a symptom). Possible causes might include a misconfigured host, bad interface cards, or missing router configuration commands.

Step 2 Gather the facts you need to help isolate possible causes.

Ask questions to affected users, network administrators, managers, and other key people. Collect information from sources such as network management systems, protocol analyzer traces, output from router diagnostic commands, or software release notes.

Step 3 Consider possible problems based on the facts you gathered. Using the facts you gathered, you can eliminate some of the potential problems from your list.

Depending on the data, you might be able to eliminate hardware as a problem, so that you can focus on software problems. At every opportunity, try to narrow the number of potential problems so that you can create an efficient plan of action.

Step 4 Create an action plan based on the remaining potential problems. Begin with the most likely problem and devise a plan in which only one variable is manipulated.

Changing only one variable at a time allows you to reproduce a given solution to a specific problem. If you alter more than one variable simultaneously, you might solve the problem, but identifying the specific change that eliminated the symptom becomes far more difficult and will not help you solve the same problem if it occurs in the future.

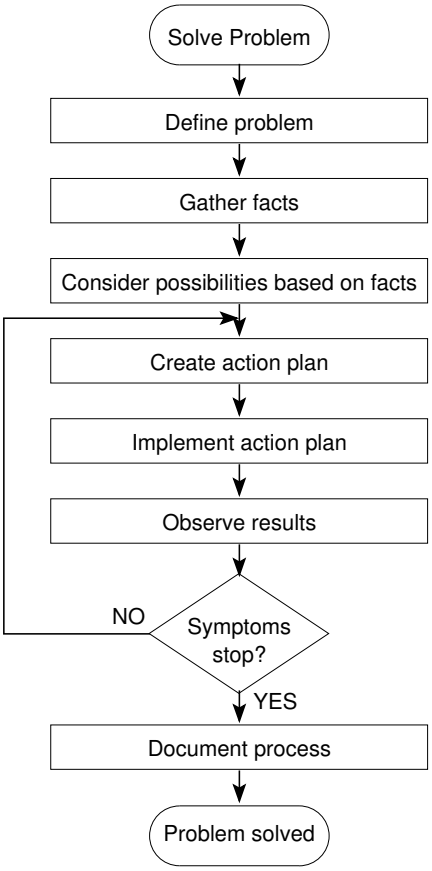

Figure 1.1: General Problem Solving Flowchart.

Step 5 Implement the action plan, performing each step carefully while testing to see whether the symptom disappears.

Step 6 Analyze the results to determine whether the problem has been resolved. If it has, then the process is complete.

Step 7 If the problem has not been resolved, you must create an action plan based on the next most likely problem in your list. Return to Step 4, change one variable at a time, and repeat the process until the problem is solved.

Step 8 Once you have identified the actual cause of the problem, you are now in a position to solve it. It's important at this point to document the problem and the solution for future reference. If all attempts to this point have failed, you have probably figured out that you need to ask for technical support from the manufacturer of the suspect equipment or to call in a professional expert or technical engineer to help you complete the troubleshooting process.

1.2.2 Analysis of step 1: define the problem

Now the particulars of each step in the problem-solving model are detailed. As with the OSI model, it is important to keep in mind that a model is exactly that: a framework that simplifies the description and analysis of otherwise complex ideas or processes. The real truth of the matter is that most do not really explicitly use models such as described here to solve networking problems. This is just a convenient categorization to help you organize your learning. Eventually, if not presently, you will solve problems in your own particular deductive, efficient manner.

In science, of which networking is but a part, a problem needs to be clearly defined before you can begin solving it. Sometimes it is your job to define the problem and sometimes it is defined for you, depending on the situation. One of the more difficult processes to master in science is that of formulating a problem statement. This often results from extensive research and sifting of information.

One major task that coincides with formulating or defining a networking problem is that of compiling an exhaustive set of symptoms from which to proceed. These establish a baseline that all subsequent analyses will depend upon. At this point, it is also wise to put together a list of possible causes.

Troubleshooting begins by identifying the symptoms and possible causes. Many possible causes might be given, but you will focus on what appear to be the most likely causes. This makes it possible to formulate a clear, concise definition of the problem and follow up by gathering all the relevant facts (Step 2).

It's probably a good idea to take the skeptic's approach when it comes to recording the diagnoses of the end users. Communication with coworkers involves humans so it is not something that lends itself to description in a book such as this. Suffice it to say that you all develop your own interpersonal skills through years of OJT (on-the-job-training). Judging the reliability of a given individual's analysis of the problem at hand is part of your job — this part of the job can very well save you a significant amount of time, so do not take it too lightly. Give people the benefit of the doubt and try to seek

out the core issues from your conversations, separating out all the extraneous issues in your mind (similar to solving word problems in math). If you haven't already, you'll likely meet some real cavalier network engineers in your work. They are the ones who treat end users like idiots. They are also the ones who have great difficulty communicating.

You will need to ask many questions and take a lot of notes in this process. Keeping an engineering journal is good practice — nowadays people use PDA's (Personal Digital Assistants) quite a bit in this context. Common sense dictates here — record times, frequency, and how the problem can be reproduced.

The baseline for the network that you are troubleshooting presumably has been established and is available to you. How you cast the problem is relative to this baseline. How the network behaves in reference to this baseline is absolutely critical.

In the subsequent chapters, you will work carefully through many examples to give you practice applying this model.

1.2.3 Analysis of step 2: gather facts

At this stage, you will compile information from affected users and key networking personnel. Collect information using network management software, `show`, and `debug` commands. [1]–[2] Print out information from CCO related to the issue, if appropriate. Refer to the documentation that came with the equipment and white papers related to the technology in question. Retrieve information from data captures if possible. Get copies of the configurations for the equipment in question. With this set of data to work from, you should be able to make a lot of progress. Remember to take notes, keeping track of dates and times as you proceed.

1.2.4 Analysis of step 3: list possible problems

Now that you have a collection of facts gathered, it is time to list the possible problems. At this stage, you can already start the process of elimination. Use logic at every turn as, contrary to popular opinion, networking has nothing to do with magic.

Record the set of possible problems, leaving room for additional items to be added later that you may have forgotten. You now have a narrowed list to work from to build an action plan.

1.2.5 Analysis of step 4: develop an action plan

Using the possible problems compiled in the previous step, you now devise an action plan. At this point, you break the problem down into smaller pieces, depending on the scope of the problem. Divide the problem into logically related sub-problems.

After breaking down the problem into manageable pieces, use the "substitution method". The substitution method manipulates only a single variable at time.

It helps to begin troubleshooting from the source and progress to the destination, moving step-by-step through the devices, thus inevitably isolating the physical location of the problem.

In this way, an action plan can be set down. This is a sequential, methodical plan that can be followed by other network engineers. In many cases, it is just a matter of obtaining the configurations of the suspect devices. The more clearly the plan is articulated, the easier it is to carry it out.

You may have noticed that some of the steps in the problem-solving model overlap with respect to the work involved. This is similar to the OSI model, where in some cases a technology or protocol does not clearly fit in a given layer.

1.2.6 Analysis of step 5: implementing the action plan

Here is where the substitution method really comes in. Based on the carefully developed action plan, you (or your team) are now ready to go forth and conquer. Remember to change one variable at a time and to record the results in each case so that you do not have to repeat them later.

A key consideration throughout this step is to minimize the impact on the end users and minimize security gaps. Of course, always be sure to keep the backup files required to return the network to its previous state should you make matters worse in the process of trying to implement your action plan.

One point that has not been made that is very important is to allow for secure remote access to networking devices in case of emergency. This may be done via dialup, SSH (secure shell), VPN (virtual private network), SNMP (simple network management protocol), or some other relatively secure technology (usually not telnet — in a private Frame Relay WAN you might use telnet). You may be many miles away from the location of the problem at hand, so you should have some mechanism, such as dialup, to access the equipment in extreme circumstances. However, some high-security installations have, as a rule, only console access for doing configurations.

1.2.7 Analysis of step 6: observe the results

Changing one variable at a time and recording your results, you have come to the end of the line. You need to analyze your results if the problem is not already fixed at this point.

If the problem has not been resolved, then you must use your recorded results to further restrict the list of problems and adjust the action plan accordingly.

If you are new to doing advanced troubleshooting, remember that it is very common to get to this step and not have a solution in hand. This model has one more step just to make this absolutely clear.

1.2.8 Analysis of step 7: repeat the process as necessary

If you have not reached a solution at this stage, what should be done next? Well, naturally you go back to Step 4 and modify your action plan. Before going further, be sure to undo any changes you have made that did not improve the situation and document those changes that will be preserved.

At the very least, at this point you should be able to simplify your action plan a bit based on the variables you have eliminated so far. Use your modified action plan and repeat Steps 5-6.

For those of you with a computer science background, you are now doing recursion but the computer is replaced with yourself. You repeat this process until you identify and fix the problem.

1.2.9 Analysis of step 8 — resolve the problem

Once you have identified and corrected the problem(s), all that remains is the last part — document everything. [1]–[2]

If you just couldn't get the thing working, it is time to call in the special forces. This may mean starting a trouble ticket with the TAC (Technical Assistance Center at Cisco), or calling on other engineers on your team, or possibly arranging for contracting additional personnel.

It's worth reiterating that this process is ominously formal as you see it here. Over time you will essentially perform the steps shown without even thinking about it. Once you have reached that point, you have to keep telling yourself:

- Change one thing at a time and undo changes that do not work
- Document, document, document
- Maintain humility

1.2.10 Summary

This curriculum will guide you through many of the major LAN and WAN technologies of the day. You will see how to troubleshoot networks in each case. Part of that process is to break down the problem into smaller pieces and to apply a methodical troubleshooting methodology. This chapter gave a seven-step sequence to do just that. The role of logic, common sense, and documentation were emphasized throughout this process. You reviewed the OSI model and its use as a common language for network engineers. You saw some network architectures and the importance of being familiar with the type of network architecture involved in a given troubleshooting scenario.

Troubleshooting many times requires a culmination of all your skills. It is not a coincidence that this course comes last in the CCNP program. It is assumed that you have mastered the basics of the core technologies. Now it is time to pull it all together. The good news is that the fun is actually doing it, and the labs that accompany this curriculum will start you on your way.

Chapter 2

Troubleshooting TCP/IP

Overview The Defense Advanced Research Projects Agency (DARPA) originally developed the Transmission Control Protocol/Internet Protocol (TCP/IP) to interconnect various defense department computer networks. The Internet, an international Wide Area Network, uses TCP/IP to connect government and educational institutions across the world. TCP/IP is also in widespread use on commercial and private networks. TCP/IP is pervasive throughout the network world and with IPv6 working its way into the Internet, TCP/IP is destined to remain the protocol suite of choice for many years to come. For example, Japan's government has imposed a 2005 deadline for the country to upgrade its information infrastructure to support IPv6.

TCP/IP is the bread and butter of internetworking. Its normally the first subject that one learns in their study of networking. The troubleshooting accompanying TCP/IP should already be somewhat familiar to anyone who has configured dialup networking on a PC or the Ethernet card settings for a Windows machine on a LAN. This chapter will explore in detail the troubleshooting of TCP/IP in both LAN and WAN environments.

The focus for this curriculum is troubleshooting issues related to technologies you have already studied in previous courses. To get a quick perspective of the types of issues dealing with troubleshooting TCP/IP, a list of common TCP/IP issues is explored after a brief review of TCP/IP theory. The common TCP/IP issues will be followed by an analysis of specific TCP/IP applications and utilities useful in troubleshooting, including an analysis of the standard router show and debug commands used for TCP/IP troubleshooting.

2.1 TCP/IP Theory

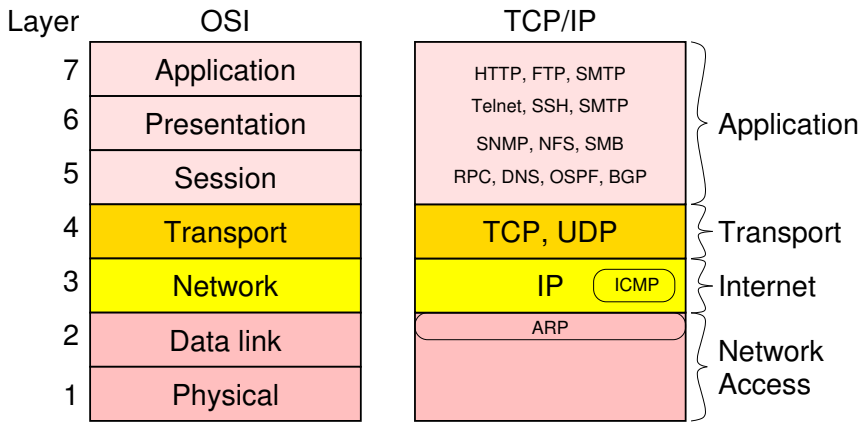

Figure 2.1: A comparison of the OSI and TCP/IP models.

2.1.1 TCP/IP evolution

Hey, man, it's the 70s! At about this time, the Defense Advanced Research Projects Agency (DARPA) became interested in establishing a packet-switched network to provide communications between research institutions in the United States. DARPA and other government organizations understood the potential of packet-switched technology and were just beginning to face the problem virtually all companies with networks now have — communication between dissimilar computer systems.

With the goal of heterogeneous connectivity in mind, DARPA funded research by Stanford University and the company of Bolt, Beranek, and Newman (BBN) to create a series of communication protocols. The result of this development effort, completed in the late 1970s, was the Internet Protocol suite, of which the Transmission Control Protocol (TCP) and the Internet Protocol (IP) are the two best-known protocols.

Internet protocols can be used to communicate across any set of interconnected networks. They are equally well suited for local-area network (LAN) and wide-area network (WAN) communications. The Internet Protocol suite includes not only lower-layer specifications (such as TCP and IP), but also specifications for such common applications such as email, terminal emulation, and file transfer. The figure shows some of the most important Internet protocols and their relationships to the Open System Interconnection (OSI) reference model.

Creation and documentation of the Internet Protocol suite closely resembles an academic research project. The protocols are specified in documents called Requests for Comments (RFCs). RFCs are published and then reviewed and analyzed by the Internet community. Protocol refinements are published in new RFCs. Taken together, the RFCs provide a colorful history of the people, companies, and trends that have shaped the development of what is today the world's most popular open-system protocol suite.

IPv6 is the next generation of IP. It uses 128 bits instead of 32 bits in its addressing scheme. For its implementation to be successful it will eventually need to be supported not only by the core Internet routers and switches, but also by devices on edge networks. The main difficulty in rolling out IPv6 is the fact that devices will need to be upgraded or replaced to support it. It is not clear at this point if the change will take place from the top down or the bottom up (driven by consumer network appliances).

2.1.2 TCP/IP protocol stack (Layer 3)

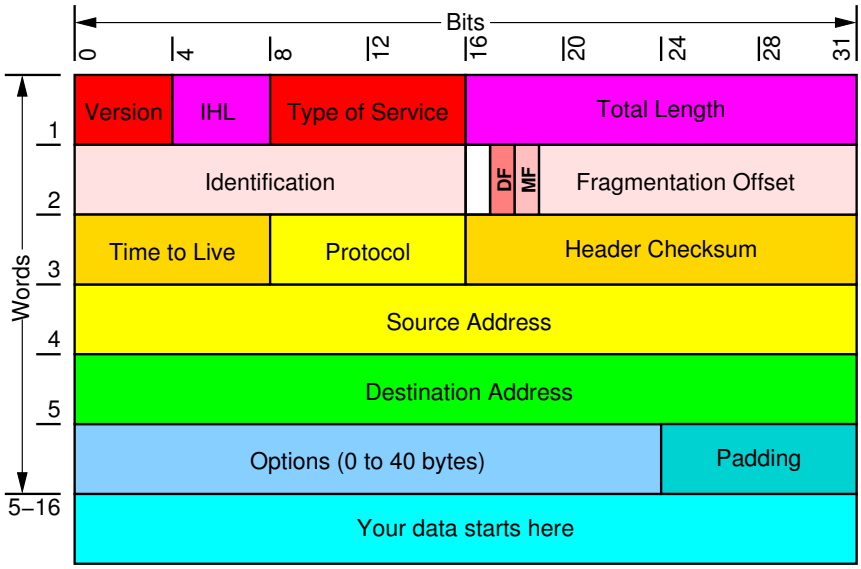

Figure 2.2: IP Header.

IP is the primary Layer 3 protocol in the Internet Protocol suite. In addition to internetwork routing, IP provides fragmentation and reassembly of datagrams and error reporting. Along with TCP, IP represents the heart of the Internet Protocol suite. The IP packet format is shown in the figure.

Although you have likely seen this numerous times in your studies to date, it is useful to look once again at the descriptions of the fields in the IP packet:

- Version — this is a 4-bit IP header length field that indicates the version of IP currently used. The current version of IP is 4 (IPv4) but IPv6 is already being implemented experimentally and will be supported on future versions of the IOS.
- IP Header Length (IHL) — this indicates the datagram header length in 32-bit words.
- Type of Service (ToS) — ToS specifies how a particular upper-layer protocol would like the current datagram to be handled. Datagrams can be assigned various levels of importance with this field.
- Total length — this specifies the length of the entire IP packet, including data and header, in bytes.
- Identification — this field contains an integer that identifies the current datagram. This field is used to help piece together datagram fragments.
- Flags — a flag is a 3-bit field of which the 2 low-order bits control fragmentation. One bit specifies whether the packet can be fragmented; the second bit specifies whether the packet is the last fragment in a series of fragmented packets.
- Time-to-Live — this field maintains a counter that gradually decrements down to zero, at which point the datagram is discarded. This prevents packets from looping endlessly.
- Protocol — protocol indicates which upper-layer protocol receives incoming packets after IP processing is complete.

- Header Checksum — this field helps ensure IP header integrity.
- Source Address — this field specifies the sending node.
- Destination Address — this field specifies the receiving node.
- Options — this field allows IP to support various options, such as security.
- Data — this field contains upper-layer information.

2.1.3 TCP/IP Layer 4

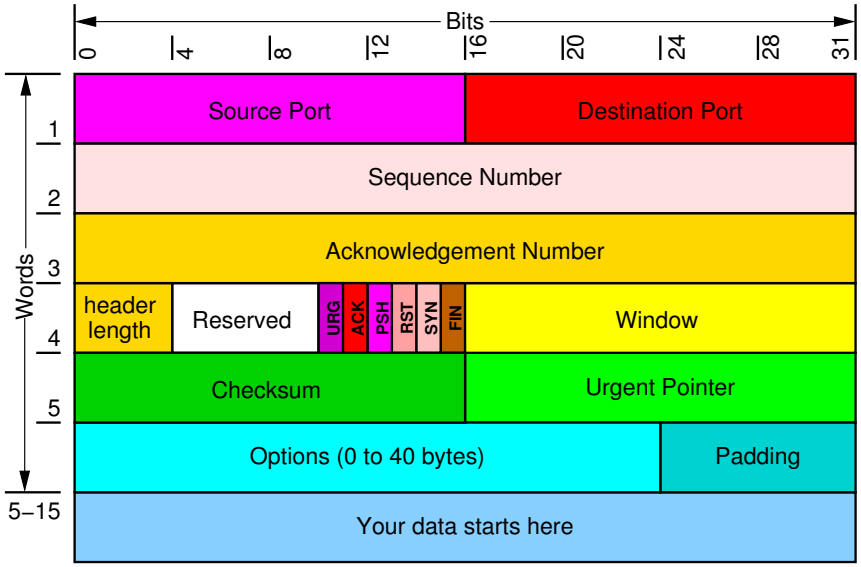

Figure 2.3: TCP Header.

The Internet transport layer is implemented by TCP and the User Datagram Protocol (UDP). TCP provides connection-oriented data transport, whereas UDP operation is connectionless.

TCP provides full-duplex, acknowledged, and flow-controlled service to upper-layer protocols. It moves data in a continuous, unstructured byte stream where bytes are identified by sequence numbers. TCP can also support numerous simultaneous upper-layer conversations. The TCP packet format is shown in Figure .

The fields of the TCP packet are defined as follows:

- Source port and destination port — these fields identify the points at which upper-layer source and destination processes receive TCP services.
- Sequence number — this field usually specifies the number assigned to the first byte of data in the current message. Under certain circumstances, it can also be used to identify an initial sequence number to be used in the upcoming transmission.
- Acknowledgment number — this field contains the sequence number of the next byte of data the sender of the packet expects to receive.
- Data offset — this field indicates the number of 32-bit words in the TCP header.
- Reserved — this field is reserved for future use.
- Flags — this field carries a variety of control information.
- Window — this field specifies the size of the sender's receive window (that is, buffer space available for incoming data).
- Checksum — this field indicates whether the header was damaged in transit.
- Urgent pointer — this field points to the first urgent data byte in the packet.
- Options — this field specifies various TCP options.
- Data — this field contains upper-layer information.

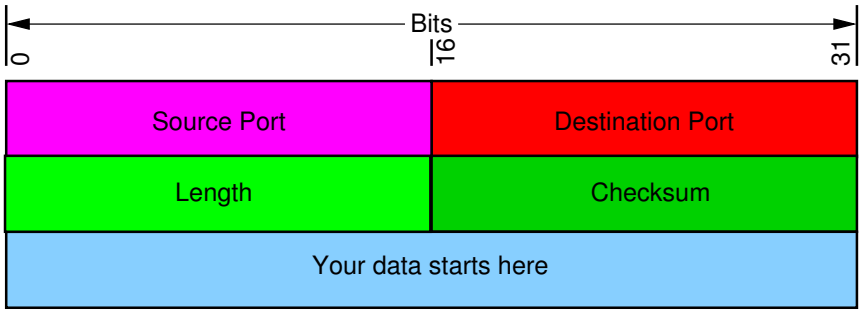

Figure 2.4: UDP Header.

A much simpler protocol than TCP, UDP is useful in situations where the reliability mechanisms of TCP are not necessary. The UDP header has only four fields: Source Port, Destination Port, Length, and UDP Checksum. These fields are illustrated in Figure 2.4.

The Source and Destination Port fields serve the same functions as they do in the TCP header. The Length field specifies the length of the UDP header and data, and the Checksum field allows packet integrity checking. The UDP checksum is optional.

2.1.4 TCP/IP upper-layer protocols

The Internet Protocol suite includes many upper-layer protocols, representing a wide variety of applications, including network management, file transfer, distributed file services, terminal emulation, and electronic mail. The figure maps well-known applications to the upper-layer protocols supporting them.

File Transfer Protocol (FTP) provides a way to move files between computer systems. Telnet allows virtual terminal emulation. The Simple Network Management Protocol (SNMP) is a network management protocol used for reporting network conditions and setting network threshold values. X Window is a popular protocol that permits intelligent terminals to communicate with remote computers as if they were directly attached. Network File System (NFS), external data representation (XDR), and remote-procedure call (RPC) combine to allow transparent access to remote network resources. The Simple Mail Transfer Protocol (SMTP) provides an electronic mail transport mechanism. These and other network applications use the services of TCP/IP and other lower-layer Internet protocols to provide users with basic network services.

2.2 Common Problems with TCP/IP

2.2.1 Symptoms of TCP/IP problems

The symptom modules following this section (see the figure) refer to TCP/IP internetwork problems. Each symptom will be explored by using the relevant TCP/IP commands on either Windows machines or Cisco routers. In this manner, you will develop a toolkit for troubleshooting TCP/IP networks.

The Windows commands used throughout the chapter include:

- ipconfig
- winipcfg
- ping
- telnet
- tracert
- nbtstat
- arp

The Cisco router commands in this chapter include:

- show running-config
- show ip interfaces
- show interfaces
- show interfaces brief
- show ip route
- show ip protocol
- show ip traffic
- show access-lists
- show ip access-list
- show ip arp
- debug ip packet
- debug ip icmp
- debug arp
- debug ip igrp events
- debug ip rip
- redistribute
- auto-summary
- distribute-list
- distance
- ip access-group
- (extended) ping
- (extended) trace
- telnet
- ipx type-20-propagation
- ip helper-address
- ip forward-protocol udp

2.2.2 Symptom: host cannot access host(s) on another network

There are several possible causes to the problem described in the figure. One problem could be that no default gateway has been configured on the workstation. If this is the suspected cause, perform the following steps:

Step 1 Determine whether a default gateway is actually configured on the host attempting to make a connection (Host-A in the figure). Use the winipcfg or ipconfig command from the DOS prompt (if the host is running Windows 95/98/NT/2000/XP).

Step 2 Inspect the output of this command for a default gateway specification.

Step 3 If the specified default gateway is incorrect, or if it is not present at all, you can change or add a default gateway using the network neighborhood properties. If a network uses DHCP the default gateway IP can be specified by the DHCP server.

Another possible problem here could be a misconfigured subnet mask on the host or router. Verify the subnet mask information on the workstation as in Step 1. You can verify the subnet mask information on the router by using the show running-config command or the show ip interface command.

Lastly, it is possible that a router or WAN link between the hosts is down. Use the ping command to determine whether the routers are reachable. If a router does not respond, isolate the problem and repair the broken interconnection.

2.2.3 Host cannot access certain networks

The first possible cause of this scenario could very well be that a default gateway has not been configured on the host. Verify that the host has a default gateway configured as specified in the last section.

A common cause of this type of problem is a misconfigured access list on the router. This access list could either be denying a specific network or host or it could be blocking routing updates about the network in question if it is being used with a distribution list. If you suspect an access list is filtering routing updates, perform the following actions:

Step 1 Check the routing table with the show ip route command and check protocol exchanges with the appropriate debug command (such as debug ip igrp events or debug ip rip).

Step 2 Look for information in the routing table concerning the specific network with which you are unable to communicate.

Step 3 Check the use of access lists on the routers in the path and make sure that a distribute-list or distance command does not filter out the route.

Step 4 Temporarily disable access lists (by removing ip access-group commands) and use the ping command with the record route option set or the trace command to determine whether traffic can get through when the access list is removed.

A not-so-obvious cause of a problem like this could be discontiguous network addressing due to poor design or link failure (assuming that RIP or IGRP is in play). Use the show ip route command to examine which routes are known and how they are being learned. Again, you can also use the trace or ping commands to discover where traffic is stopping. If you have determined that this is the problem, fix the topology or reassign addresses to include all appropriate network segments into the same major network.

2.2.4 Connectivity available to some hosts, but not others

The two most common problems for this type of scenario are a misconfigured subnet mask or a missing default gateway on the host workstation. Using either the `winipcfg` or `ipconfig` command on the host quickly reveals this information.

A third common cause for this type of problem would be a misconfigured access list (host is denied by some router in the path). To determine if an access list is causing problems, follow these steps:

Step 1 Ping along the network path to determine where packets are being dropped.

Step 2 If you can identify the router that is stopping the traffic, use the show running-configuration command to see whether an access list is being used. You can also use the show access-lists and show ip interface commands to determine whether access lists are being used.

Step 3 Temporarily disable the access list.

Step 4 See whether traffic can get through the router (ping or telnet).

Step 5 If traffic can get through, carefully review the access list and its associated commands for proper authorization.

As you can probably tell by now, many problems in IP environments are caused by addressing and subnet mask problems. In the example shown in the figure, the symptom is that users can access some hosts but not others. Also, the router and hosts cannot reach certain parts of their own networks. This is a classic case of convoluted IP addressing - take a close look at the IP addresses on the hosts. IP addressing is the first thing that should be checked in the case of intermittent pings or pings succeeding between certain hosts and not others.

2.2.5 Some services are available, while others are not

A probable cause for this type of scenario is a misconfigured extended access list on one of the routers. To troubleshoot this possibility, follow these steps:

Step 1 Use the trace or extended ping command to determine the path taken to reach remote hosts.

Step 2 (Optional) On each router in the path, enable the debug ip icmp command. This will tell you if the router is actually receiving and replying to the ICMP echo request.

Step 3 If you can identify the router that is stopping the traffic, use the show running-configuration command to see if an access list is applied. You can also use the show access-lists and show ip interface commands to determine if access lists are being used and to which interface(s) they're applied.

Step 4 Temporarily disable the access list.

Step 5 Test to see if traffic can get past the router.

Step 6 If traffic can get through, carefully review the access list and its associated commands for proper authorization.

Step 7 In particular, look for any TCP parameters configured in the extended access list.

Step 8 If TCP ports are specified, be sure that all necessary ports are explicitly permitted by the access lists.

2.2.6 Users cannot make connections when one path is down

The figure illustrates one example of a situation in which this symptom can occur. Here, a major network (Net-B) has four paths into another major network (Net-C), with Serial-Z forming a link between the two separate subnets of Net-C. In this case, the campus spans a metropolitan area and has two major networks, Net-B and Net-C, and because it is not practical to connect the buildings with LAN media — serial connections are required. The problem is that when Serial-Z is down, traffic cannot traverse from Net-C1 to Net-C2 through Router-B1.

The reason for this problem is that major network Net-C becomes a discontiguous network because Router-B1 is separating the two Net-C subnets (Net-C1 and Net-C2). Traffic between Router-C1 and Router-C2 will not get through Router-B1 because Router-B1 assumes that Net-C1 and Net-C2 are directly connected to each other. As an alternative, use a secondary IP address configuration to ensure that all interfaces are included in the same major network.

Assuming the configuration above included secondary addresses, there are several other potential causes for this problem. For example, routing may not have converged. You can verify this by examining routing tables for routes that are listed as "possibly down." If this entry is found, the routing protocol has not converged.

Another possible problem is errors on either the serial or Ethernet link. If the link is a serial link, observe the counters on the interface (using the show interfaces serial command). If you think the problem is on the Ethernet segment, use a time domain reflectometer (TDR) to find any unterminated cables. Another option is to check host cables and transceiver cables to determine whether any are incorrectly terminated, too long, or damaged. Lastly, look for a jabbering transceiver attached to a host (this may require inspection of each host).

2.2.7 Router/host cannot reach certain parts of its own network

There are several possible causes for unreachable hosts connected to the same router. For example, refer to the figure and assume Host A cannot communicate with Host E.

One possible reason for this is a subnet mask configuration mismatch between the router and the host. If you can ping from the local host to the local router (but not to the remote host), and you can ping from the local router to the remote host, there is probably a subnet mask configuration problem on your local host or router.

Another possible reason for this is a misconfigured access list. As with other issues where an access list is suspect, use ping and trace commands as described at the beginning of this chapter to isolate the router with the misconfigured access list. Then take the appropriate measures to solve the problem.

The final possibility is that no default gateway is specified. Check the remote host for a proper default gateway and check configurations on the hosts and routers for static routes.

2.2.8 Problem isolation in TCP/IP

One thing to keep in mind when troubleshooting broken interconnections is that normally everything does not break at the same time. As a result, when trying to isolate a problem, you can typically work out from an operational node to the point of failure. To summarize what you have seen with TCP/IP troubleshooting up to now, the following basic steps should help when trying to isolate the source of connection disruption:

Step 1 First, determine whether your local host is properly configured (for instance, correct subnet mask and default gateway configuration).

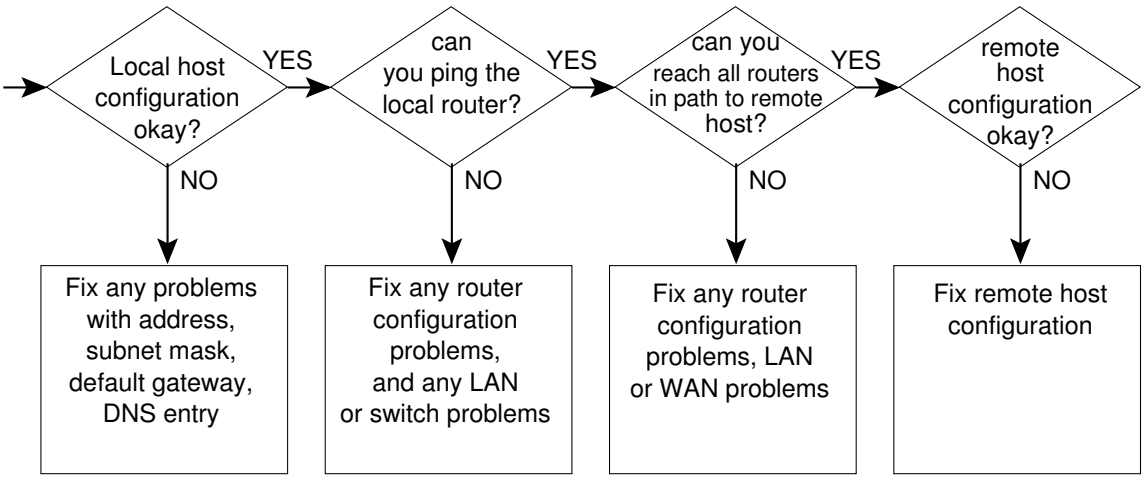

Figure 2.5: Troubleshooting TCP/IP

Step 2 Next, use the ping or tracert commands to determine whether the routers through which you must communicate can respond. Start with the most local router and progressively "ping out" through the Internet or simply do a trace.

Step 3 If you cannot get through a particular node, examine the node configuration and use the various show commands to determine the state of the router (these include show ip route, show ip arp, show running-configuration, and so on.)

Step 4 If you can get to all the routers in the path, check the host configuration at the remote host (or get someone's help to do so), and check its configuration.

This completes the symptomatic approach to generic TCP/IP troubleshooting issues. Next you will delve into troubleshooting in a Windows networking environment.

2.3 Troubleshooting in a Windows Environment

2.3.1 General Windows troubleshooting considerations

When you configure Cisco IOS Software, the commands you use depend on traffic requirements of both the Windows NT/2000/XP server and the Windows client. The table in Figure shows five variations.

Basic troubleshooting for TCP/IP on Windows machines combines facts gathered from both a router (or switch) perspective and facts gathered from a Windows client or server perspective.

Many of the same Cisco IOS Software tools that you use for diagnosing and correcting TCP/IP problems apply to a network containing Microsoft Windows systems.

The same system that you use to console to a router also can be used to provide telnet access, so as you know well by now, you can use the command line TCP/IP tools included with Windows operating systems while simultaneously consoling to a router.

To check the local host configuration, open a DOS window on the host and enter the `ipconfig /all` command (this only works on Windows NT/2000/XP systems), as shown in Figure . The results of this command show your TCP/IP address configuration, including the address of the Domain Name System (DNS) server. If any IP addresses are incorrect or if no IP address is displayed, determine the correct IP address and edit it or enter it for the local host.

Most incorrect IP address or subnet mask errors appear in Event Viewer. Examine the Event Viewer system log and look for any entry with TCP/IP or DHCP as the source. Read the appropriate entries by double-clicking them. Because DHCP configures TCP/IP remotely, DHCP errors cannot be corrected from the local computer.

The commands to use for checking the router configuration with Cisco IOS Software are covered earlier in this chapter and elsewhere throughout this course.

You should check to see if you can connect using IP addresses. Use an IP address as a target for the standard TCP/IP commands such as ping, tracert, and telnet.

Also, check the configurations on the NT/2000/XP server. If you can connect using an IP address but are unable to connect by using "Microsoft networking" (for example, Network Neighborhood), try to isolate a problem on the Windows NT/2000/XP server configuration. Problem areas with Microsoft networking relate to NetBIOS support and associated mechanisms used to resolve non-IP entities with IP addresses. You can check for these non-IP problems using the nbtstat command.

As a last resort, try rebooting the Windows system — you should try to avoid this, but this step frequently does the trick.

2.3.2 Browsing problems

When you "browse" a Windows 9x/Me/NT/2000/XP network, you are searching for representations of the resources that are available on the network. For example, on a Windows 2000 desktop, when you click on the icon for My Network Places (followed by Entire Network and then Microsoft Windows Network), you see a list of available domains or workgroups as shown in the figure.

Users can click to view the subordinate lists of all the network resources that they can access. These resources include network-connected servers and specific resources on the servers such as disks, subdirectories, printers, and other shares.

Browsing problems are among the most frequently encountered problems in Windows networks. The browsing problems usually involve resources that have been added, removed, or changed; this alteration of the network resources is not accurately indicated when the user browses to see the list of network resources.

Nevertheless, if the user is unable to browse a Windows resource, the user may still be able to connect to it with some other process or network application. Although Microsoft's intent is that Windows networks allow for transparent browsing of network resources, network administrators tend to focus much of their troubleshooting attention on browsing issues.

Several possible causes of browsing problems follow:

- Incorrect configuration introduced as the network grows beyond a simple LAN — an organization may find that domain security or some other function of a campus network requires reconfiguration on the LAN.
- Inaccurate or incomplete resolution of non-IP entities into IP addresses — name resolution on Windows hosts require proper functioning of the LMHosts and Hosts files as well as the Windows Internet Name Service (WINS) and DNS servers.
- Inappropriate sources for browser update information on the network — a conflict can arise when several NT systems are set up as master browsers and they send inconsistent update information that hinders convergence.

2.3.3 The tracert tools

The `tracert` tool on an Windows NT/2000/XP host reports each node a TCP/IP packet crosses on its way to a destination. It does essentially the same thing as the `trace` command in the Cisco IOS Software. The syntax for the `tracert` command follows:

```
tracert [-d [-h maximum_hops] [-j host-list] [-w timeout] target_name
```

Parameters are as follows:

- d — specifies to not resolve addresses to host names
- h maximum_hops — specifies the maximum number of hops to search for target
- j host-list — specifies loose source route along the host list
- w timeout — waits the number of milliseconds specified by timeout for each reply
- target_name — name or IP address of the target host

Errors that may occur include the asterisk ('*') and a message about request timed out. These messages indicate a problem with the router or a problem elsewhere on the network. The error may relate to a forwarded packet or one that timed out.

Another common error is a report of destination network unreachable. This error may indicate that there is a proxy or a firewall between your computer and the computer you are targeting as your tracert destination.

2.3.4 Checking the routing table on a Windows NT/2000/XP system

To check the routing table, type the route print command at a command prompt. In this example, the local IP address is 192.1.1.3, and the default gateway is 192.1.1.254. Note that Windows NT/2000/XP boxes are not typically configured as routers, so most of the entries will be somewhat trivial as compared to the output of the show ip route command on a router.

2.3.5 Clearing the Windows NT/2000/XP system ARP cache

To view the arp cache, which is a list of recently resolved IP-to-MAC address mappings, at the command prompt type `arp -a`.

You can try to resolve an address problem by clearing the ARP cache. If an entry in the ARP cache is incorrect, the TCP/IP packet will be sent to the wrong computer. To clear the cache, type:

`arp -d` [IP] where [IP] is the IP address of the incorrect entry; another option is the command `arp -d *`, which clears the entire arp cache.

If you issue the `arp -a` command again, you will see the entry or entries will be cleared, as shown in Figure .

It is not unusual to have address resolution problems on a network, so you should be familiar with these commands, as well as the associated show ip arp commands on the router.

2.3.6 DNS configuration and host file issues

If you are using TCP/IP in your network and have verified that the IP settings are correct, the sequence of how to resolve non-IP entities will begin with NBT — NetBEUI over TCP/IP. This approach includes the following sequence of steps.

If you can only connect to the server using the IP address, you may be having trouble with your DNS service. If you do not have a DNS server address configured, you will not be able to communicate on the network via host names unless the required mapping is included in the Host file.

You need to contact your network administrator to obtain a valid DNS server address. Once you have a valid DNS address, you need to update your TCP/IP settings or Dial-Up Networking phone-book entry accordingly.

Check the Hosts file for an improper entry. The Hosts file is usually located in the `Winnt\System32\Drivers\Etc` folder on an NT system. The Hosts file is a text file that you can edit with any text editor (e.g., Notepad).

Search the file for the host name you are attempting to connect to, and verify that all entries are correct. Remove or correct any improper entries.

The LMHosts file is usually located in the `Winnt\System32\Drivers\Etc` folder on an NT system. The LMHosts file is a text file that you can edit with any text editor (e.g., Notepad). Check the LMHosts file for an improper entry. Search the file for the host name you are attempting to connect to, and verify that all entries are correct.

Microsoft recommends that if there are any #include entries, temporarily remove them; also remove any #BEGIN_ALTERNATE to #END_ALTERNATE blocks.

If removing lines in the LMHost file corrects the problem, add back one line at a time until the problem recurs, and then search the appropriate LMHosts files pointed to by the line most recently added. This process of elimination may provide you with a way to isolate the cause of the problem.

The NT/2000/XP system may be configured to use a proxy agent for connections to remote hosts. This use of a proxy can cause problems when the system tries to connect to hosts that should not have to go through the proxy.

To determine a Winsock proxy setting, check for a Winsock Proxy Client (WSP) icon in the Control Panel. If one exists, try disabling it by clearing the Enable Winsock Proxy check box. Check the Client check box. After rebooting the system (unnecessary for 2000 or XP), try connecting again.

If the check box was already cleared, click on it to select it. You may need to contact your system administrator for the name of the proxy server or you may want to examine a computer that can connect and copy its WSP information.

Many Web browsers have built-in support for proxies. If you are attempting to connect with Hypertext Transfer Protocol (HTTP) to a remote Web site, you may need to disable or enable proxy support on the system.

Again, although Microsoft intends that Windows browsing be somewhat transparent to the user and that resources add themselves in a "plug-and-play" manner, administrators tend to focus much of their troubleshooting attention on browsing.

Several of the possible causes of browsing problems follow:

- Incorrect configuration introduced as the network grows beyond a simple LAN — an organization may find that domain security or some other function on a campus network requires reconfiguration on the LAN.
- Inaccurate or incomplete resolution of non-IP entities into IP addresses — name resolution on Windows hosts requires proper functioning of the LMHosts and Hosts files as well as the WINS and DNS servers.
- Inappropriate sources for browser update information on the network — a conflict can arise when several NT systems are set up as master browsers and they send inconsistent update information that hinders convergence.
- Incorrect proxy settings may be configured (incorrect settings can prevent access to remote hosts) — a proxy may be enabled when it should not be or vice versa.

This completes the Windows troubleshooting portion of this chapter. Now it's time to change gears. You now have a fairly solid set of tools at your disposal for troubleshooting TCP/IP, based on the material in 4.1, 4.2, and 4.3. However, there's still a lot of ground to cover with TCP/IP troubleshooting. It's time to look at some diagnostic tools for TCP/IP. You will take a little side tour to look at the ICMP Router Discover Protocol (IRDP), as it is a TCP/IP protocol that may enter your world of troubleshooting. Following the survey of diagnostic tools you will learn in some detail how to use IOS show and debug commands to troubleshoot TCP/IP.

2.4 TCP/IP Diagnostic Tools

2.4.1 ICMP

nternet Control Message Protocol (ICMP) performs numerous tasks within an IP internetwork — the word to focus on in this acronym is message. The principal reason it was created was to report routing failures back to the source. In addition, ICMP provides helpful messages such as the following:

- Echo and echo reply messages to test node reachability across an internetwork
- Redirect messages to stimulate more efficient routing

- Time exceeded messages to inform sources that a datagram has exceeded its allocated time to exist within the internetwork
- Router advertisement and router solicitation messages to determine the addresses of routers on directly attached subnetworks

A more recent addition to ICMP provides a way for new nodes to discover the subnet mask currently used in an internetwork. Essentially, ICMP is an integral part of all IP implementations, particularly those that run in routers; this protocol is extremely important for troubleshooting.

One of the most common ICMP uses is as a diagnostic tool. As you can see in the graphic, a simple `ping` utilizes ICMP to determine whether or not a host is receiving packets. For more details on ICMP, refer to RFC 792.

2.4.2 IRDP

The ICMP Router Discovery Protocol (IRDP) uses router advertisement and router solicitation messages to discover addresses of routers on directly attached subnets.

With IRDP, each router periodically multicasts router advertisement messages from each of its interfaces. Hosts discover the addresses of routers on the directly attached subnet by listening for these messages. Hosts can use router solicitation messages to request immediate advertisements, rather than wait for unsolicited messages.

IRDP offers several advantages over other methods of discovering addresses of neighboring routers. Primarily, it does not require hosts to recognize routing protocols, nor does it require manual configuration by an administrator.

Router advertisement messages allow hosts to discover the existence of neighboring routers, but not which router is best to reach a particular destination. If a host uses a poor first-hop router to reach a particular destination, it receives a redirect message identifying a better choice.

The only required task for configuring IRDP routing on a specified interface is to enable IRDP processing on an interface. The command to do this on the router is `router(config-if)#ip irdp` shown in the figure. As you can see in the example, this command is used only on the LAN interfaces. Lastly, hosts must also be running IRDP for this configuration to function correctly.

Note: IRDP is configured by default on TCP/IP for Windows 2000 hosts; a Windows 2000 server can be configured as an IRDP router as well.

Hot Standby Routing Protocol (HSRP) is another method of allowing redundant host-to-router connectivity. HSRP runs on participating subnet interfaces on Cisco routers, not on the hosts on the network. HSRP also relies on multicast messaging, but not on ICMP. HSRP is useful for hosts that do not support a router discovery protocol (such as IRDP) and cannot switch to a new router when their selected router reloads or loses power. Because existing TCP sessions can survive the fail over, this protocol also provides a more transparent recovery for hosts that dynamically choose a next hop for routing IP traffic. The fail over rate is much faster for HSRP than IRDP as well, so it is more commonly used in network installations with Cisco routers.

2.4.3 `ping` command

To check host reachability and network connectivity, use the `ping` command. After you login to the router or access server, you are automatically in user EXEC mode. The EXEC commands available at the user level are a subset of those available at the privileged level. In general, the user EXEC commands allow you to connect to remote devices, change terminal settings on a temporary basis, perform basic tests, and list system information. The `ping` command can be used to confirm basic network connectivity on AppleTalk, ATM, ISO Connectionless Network Service (CLNS), IP, Novell, Apollo, VINES, DECnet, or Xerox Network Systems (XNS) networks

For IP, the `ping` command sends ICMP echo messages. Recall that ICMP is the Internet protocol that reports errors and provides information relevant to IP packet addressing. If a station receives an ICMP echo message, it sends an ICMP echo reply message back to the source.

A loopback `ping` is one of the first `ping` tests you should perform when connectivity is in question. A loopback `ping` is addressed to 127.0.0.1 (the loopback address) to check the local TCP/IP stack integrity. An example of this is shown in Figure .

To perform a basic check of host reachability and network connectivity from a router, you can use the `ping` user EXEC command as follows:

```
ping {protocol} {host | address}
```

The protocol field identifies which ping protocol to use. This can include IP, Internetwork Packet Exchange (IPX), AppleTalk, and so on. The second field specifies either a host name or IP address of the system you are trying to ping.

An example of a successful ping is shown in Figure 2. . However, if the `ping` was not successful, you would see periods instead of exclamation points. The possible results of a ping command are listed in Figure .

To abort a `ping` session on a router, you can use the escape sequence Ctrl-Shift-6 by pressing these keys simultaneously.

2.4.4 Extended ping command

Cisco internetworking devices provide a mechanism to automate the sending of many ping packets in sequence. Figure illustrates the menu you can use to specify extended ping options. This example specifies 20 successive pings (in the Repeat Count field). However, when testing the components on your serial line, you should specify a much larger number, such as 1000 pings. Each field entry is explained in the table. (PDF, 14 Kb)

You can configure the extended ping command to send different data patterns and packet sizes. Figures and illustrate two useful ping tests, an all-zeros 1500-byte ping and an all-ones 1500-byte ping, respectively. Varying the data pattern in this field (to all ones or all zeros, for example) can be useful when debugging data sensitivity problems on CSU/DSUs or detecting cable-related problems such as crosstalk.

Another useful component of the extended ping command is the Set DF bit in IP Header option — DF stands for Don't Fragment. This option specifies that if the packet encounters a node in its path that is configured for a smaller MTU than the packet MTU, the packet is to be dropped and an error message is to be sent to the protocol translator at the packet source address. If performance problems are encountered on the network, a node configured for a small MTU could be a contributing factor. This feature can be used to determine the smallest MTU in the path.

Further, in the Loose, Strict, Record, Timestamp, Verbose [none] option, you can use the Record option to trace a path to a particular destination address. Be aware, however, that the trace EXEC command performs a similar function, but the latter does not have the nine-hop limitation. An example of this is shown in Figure .

Figure , five ping echo packets are sent to the destination address 128.171.1.1. The echo packet detail section includes specific information about each of these echo packets.

The lines of ping output that are unique when the Record Route option is specified are described below:

- The following line of output allows you to specify the number of hops that will be recorded in the route. (Range: 1 to 9. Default: 9).

```
Number of hops [ 9 ]:
```

- The following line of output indicates that IP header options have been enabled on the outgoing echo packets and shows the number of option bytes and padded bytes in the headers of these packets:

```
Packet has IP options: Total option bytes= 39; Padded length = 40
```

- The following lines of output indicate the fields that will contain the IP addresses of the nodes in the routes have been zeroed out in the outgoing packets.

```
Record route: <*>
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
        (0.0.0.0)
```

- The following lines of output display statistics for the first of the five echo packets sent. The 0 is the number assigned to this packet to indicate that it is the first in the series. The 16 ms indicates the round-trip travel time for the packet.

```
Reply to request 0 (16 ms). Received packet has options
        Total option bytes= 40, padded length=40
        Record route:
                (166.122.22.151)
                (166.122.24.3)
                (166.122.20.4)
                (166.122.5.230)
                (128.171.64.29)
                (128.171.1.201)
                (128.171.1.1)
                (128.171.64.217)
                (166.122.5.237)
                <*>
                 End of list
```

- The following line of output indicates that seven nodes were included in the packet route, including the protocol translator at source address 166.122.22.151, five intermediate nodes at addresses 166.122.24.3, 166.122.20.4, 166.122.5.230, 128.171.64.29, and 128.171.1.201, and the destination node at address 128.171.1.1.

```
Record route:
        (166.122.22.151)
        (166.122.24.3)
        (166.122.20.4)
        (166.122.5.230)
        (128.171.64.29)
        (128.171.1.201)
        (128.171.1.1)
```

In Figure , note that the return path of the packets differs from the original route.

2.4.5 trace command

The `traceroute` (`trace` for short) user EXEC command discovers the routes that packets follow when traveling to their destination. The `trace` privileged EXEC command permits the supported IP header options to be specified, allowing the router to perform a more extensive range of test options.

The `trace` command works by using the error message generated by routers when a datagram exceeds its Time-To-Live (TTL) value — data from ICMP messages sent from routers along the path are used to generate output. First, probe datagrams are sent with a TTL value of 1. This causes the first router to discard the probe datagrams and send back time exceeded error messages. The `trace` command then sends several probes and displays the round-trip time for each. After every third probe, the TTL is increased by one.

Each outgoing packet can result in one of two error messages. A time exceeded error message indicates that an intermediate router has seen and discarded the probe. A port unreachable error message indicates that the destination node has received the probe and discarded it because it could not deliver the packet to an application. If the timer goes off before a response comes in, `trace` prints an asterisk (*).

The trace terminates when the destination responds, when the maximum TTL is exceeded, or when the user interrupts the trace with the escape sequence.

As with `ping`, it is a good idea to use the `trace` command when the network is functioning properly to see how the command works under normal conditions so you have something to compare against when troubleshooting.

2.4.6 Privileged trace command

Due to bugs in the IP implementations of various hosts and routers, you may notice one or more of the following behaviors when using the trace command.

Sometimes not all destinations will correctly respond to a probe message by sending back an ICMP port unreachable message. A long sequence of TTL levels with only asterisks, terminating only when the maximum TTL has been reached, may indicate this problem.

There is a known problem with the way some hosts handle an ICMP TTL exceeded message. Some hosts generate an ICMP message but they reuse the TTL of the incoming packet. Because this is zero, the ICMP packets do not make it back. When you trace the path to such a host, you may see a set of TTL values with asterisks (*). Eventually the TTL gets high enough that the ICMP message can get back. For example, if the host is six hops away, trace will timeout on responses 6 through 11. Responses 12 and after should be fine.

An example of the extended trace command is shown in the figure. For trace output, an “!N” represents ICMP network unreachable, “!H” represents ICMP host unreachable, “!P” represents ICMP protocol unreachable, and “!Q” represents source quench received. The asterisk “*” represents timeout, and “?” represents unknown packet type.

When tracing IP routes, the following trace command parameters may be set:

- Target IP address — you must enter a host name or an IP address. There is no default.
- Source address — this is one of the interface addresses of the terminal server or router to use as a source address for the probes. The terminal server or router will normally pick what it feels is the best source address to use.
- Numeric display — the default is to have both a symbolic and numeric display; however, you may suppress the symbolic display.
- Timeout — this is the number of seconds to wait for a response to a probe packet. The default is three seconds.
- Probe count — this is the number of probes to be sent at each TTL level. The default count is 3.
- Minimum Time to Live [1] — this is the TTL value for the first probes. The default is 1, but may be set to a higher value to suppress the display of known hops.

- Maximum Time to Live [30] — this is the largest TTL value that may be used. The default is 30. The
- trace command terminates when the destination is reached or when this value is reached. Port number — this is the destination port used by the UDP probe messages. The default is 33,434.
- Loose, strict, record, timestamp, verbose — these are IP header options. You may specify any combination. The
- trace command issues prompts for the required fields. Note that trace will place the requested options in each probe; however, there is no guarantee that all routers (or end nodes) will process the options.
- Loose source routing — you may specify a list of nodes that must be traversed when going to the destination.
- Strict source routing — you may specify a list of nodes that must be the only nodes traversed when going to the destination.
- Record — you may specify the number of hops to leave room for.
- Timestamp — you may specify the number of time stamps to leave room for.
- Verbose — if you select any option, the verbose mode is automatically selected and trace prints the contents of the Option field for any incoming packets. You can prevent verbose mode by selecting it again and toggling its current setting.

2.4.7 show ip access-list command

The `show ip access-list` command displays the contents of all current IP access lists. This can help you debug problems that occur because of access or security settings. The command shows the access-list number or name, the permit and deny source and destination addresses, and wildcard masks. The output of this command is identical to the show access-list command except for the fact that this command is limited to IP output only.

Figures and illustrate sample output from the `show ip access-list` command. In Figure , the network administrator has found that UDP packets are not getting past the router. The intent of the access list was to prevent Network Timing Protocol (NTP) traffic from crossing the router. By issuing the `show ip access-list` command, the network administrator was able to see that the first statement was configured incorrectly and actually denying all UDP traffic. The bottom configuration in Figure illustrates the modified configuration.

You can also examine a specific access list, as shown in Figure , by simply specifying the access-list name or number.

2.4.8 show ip arp command

The `show ip arp` command displays the contents of the IP ARP cache. ARP establishes correspondences between network addresses (an IP address, for example) and LAN hardware addresses (Ethernet addresses, for example). A record of each correspondence is kept in a cache for a predetermined amount of time and then discarded. The figure illustrates sample output of the command.

The following describes each field of the `show ip arp` command:

Protocol: Protocol for network address in Address field

Address: The network address that corresponds to hardware address

Age (min): Age, in minutes, of the last update of the cache entry

Hardware Addr: LAN hardware address that corresponds to network address

Type: Type of ARP:

- ARPA = Ethernet-type ARP
- SNAP = RFC 1042 ARP
- Probe = HP Probe Protocol

Interface: Interface to which this address mapping has been assigned

2.4.9 show ip interface command

The `show ip interface` command displays the usability status of interfaces. Among other things, this can help you make sure the router interface or subinterface is up and configured with the correct address and subnet mask; it can also be used to check if routes may have been learned from the wrong interface or protocol because of disabled split horizon on a LAN. Figure shows sample output form the show ip interface command.

As you can see, this command will yield much of the same output as the show interfaces command. However, the show ip interface command is limited to IP information.

To obtain a quick summary of IP interfaces, you can use the command `show ip interface brief` as in Figure .

2.4.10 show ip protocols command

The `show ip protocols` command displays information that is useful in debugging routing operations. Information in the Routing Information Sources field of the show ip protocols output can help you identify a router suspected of delivering incorrect routing information. Figure illustrates a sample output of the `show ip protocols` command, and Figure outlines the significant fields from the command.

2.4.11 show ip route command

The `show ip route` EXEC command is used to display the current state of the routing table. You can use this command to determine whether routes appear in the routing table. This can help you determine whether IP routing is running and whether the routing protocol is misconfigured on one or several of the routers in the network. This could very well be the cause of host access problems. An example of this command is shown in Figure , with explanations in Figure . This command can be used in both user EXEC and privileged EXEC command modes.

You can include the optional address keyword to limit the display to only information about that address, as shown in Figure , with explanations in Figure . You can also include a protocol keyword (e.g., OSPF, BGP, EIGRP) to limit the display to information about a specific routing protocol. An example of the command `show ip route bgp` is shown in Figure .

2.4.12 show ip traffic command

The `show ip traffic` command displays the statistics that the router has gathered about its IP protocol processes. An example of this command is shown in Figure which displays information regarding packets received and sent, and in some cases, the broadcasts and error counts. Figure describes the fields for the show ip traffic command. Error statistics (i.e., format errors, bad hop count, failed encapsulations, and packets discarded because there was no route) are included to aid your debugging.

2.5 TCP/IP debug Commands

2.5.1 debug ip icmp command

The debug privileged EXEC commands can provide a wealth of information about the traffic being seen (or not seen) on an interface, error messages generated by nodes on the network, protocol-specific diagnostic packets, and other useful troubleshooting data. Exercise care when using debug commands. Many debug commands are processor intensive and can cause serious network problems (e.g., degraded performance, loss of connectivity) if they are enabled on an already heavily burdened router. When you finish using a debug command, remember to disable it with the appropriate no debug command (or use the no debug all or undebug all command to turn off all debugging). In this section, you will see how the debug ip icmp, debug ip packet [detail], and debug arp commands are used in troubleshooting.

The `debug ip icmp` command is very useful when you are trying to determine whether or not a router is sending or receiving ICMP messages. An example of this is shown in Figure and . Here R1 is trying to ping R2. However, the ping keeps timing out, as you can see by the router output as shown in Figure . The `debug ip icmp` command will help you determine whether or not the packets are getting to R2 as shown in Figure . As you can see from the debug output in Figure , R2 is receiving echo requests resulting from R1's extended ping to the 10.104.104.114 host off R2. However, you may have noticed in Figure , that R2 does not have a static route back to 10.103.103.0, therefore the replies do not get sent to R1. The solution is to add the `ip route 10.103.103.0 255.255.255.0 10.102.102.1` command on R2.

The possible field descriptions for this command are shown in this table. (PDF, 13Kb)

2.5.2 debug ip packet command

The debug ip packet [detail] command displays general IP debugging information and IP security option transactions. The detail keyword gives you more output, including TCP/UDP port number information.

You can use this command to analyze messages traveling between local and remote hosts when troubleshooting an end-to-end connection problem. IP debugging information includes packets received, generated, and forwarded. Fast-switched packets do not generate messages.

It is not uncommon to see "encapsulation failed" in the output of the `debug ip packet` command, but that alone does not normally do much to help troubleshoot the problem at hand. This output normally means that the router knows which interface it has to send the packet through, but does not know how to do it. In this case, you have to understand how ARP works; use the `debug arp` command to better understand the problem (next section).

An optional access-list-number argument allows you to limit the scope (and load on the router) caused by the `debug ip packet` command.

Figure shows sample output from the `debug ip packet` command, and the table in Figure explains each field.

2.5.3 debug arp command

The `debug arp` command can be used to check the flow of information on ARP transactions. You use it for problems in which some nodes on a TCP/IP network respond but other nodes do not. The `debug arp` command checks whether the router is sending and receiving ARP requests and replies.

The figure illustrates sample output from the `debug arp` command.

The first line of output indicates that the router at IP address 131.108.22.7 and MAC address 0000.0c01.e117 sent an ARP request for the MAC address of the host at 131.108.22.96. The series of zeros at the end of this line indicate that the router is unaware of the host MAC address.

The second line of output indicates that the router has received an ARP reply containing the unknown MAC address.

The third line indicates that the router has received an ARP request from another host looking for the MAC address of a separate host.

The fourth line of the output indicates that another host on the network attempted to send the router an ARP reply for the router address. The router filters meaningless replies. Usually, meaningless replies are caused by a misconfiguration. For example, another station might be incorrectly configured with the router IP address.

From this output, you can tell that the router filters improper replies but displays them when the debug arp command is issued, which may prove useful in troubleshooting.

2.6 Summary

In this chapter, you learned a thorough set of troubleshooting tools and methods for TCP/IP internetworks. Cisco router show, debug, ping, and trace commands and related Windows commands were the major tools presented in the TCP/IP troubleshooting scenarios. Various Windows networking scenarios involving lost connectivity between hosts were analyzed. You learned about the protocols supporting IP within internetworks (e.g., ICMP, TCP, and UDP) and the applications they support. The function of each field in the headers of the protocols supporting TCP/IP was also explored.

The TCP/IP protocol suite and TCP/IP applications form the basis for the Internet as it exists today. What you have learned up to now serves as a solid foundation for the upcoming analyses of the respective technologies and protocols: LAN switching, Frame Relay, ISDN, IPX, AppleTalk, EIGRP, OSPF, and BGP. In particular, troubleshooting IPX and AppleTalk will be much easier now that you have somewhat mastered TCP/IP troubleshooting.

Troubleshooting in a LAN switch environment is the next topic to be explored in the curriculum.