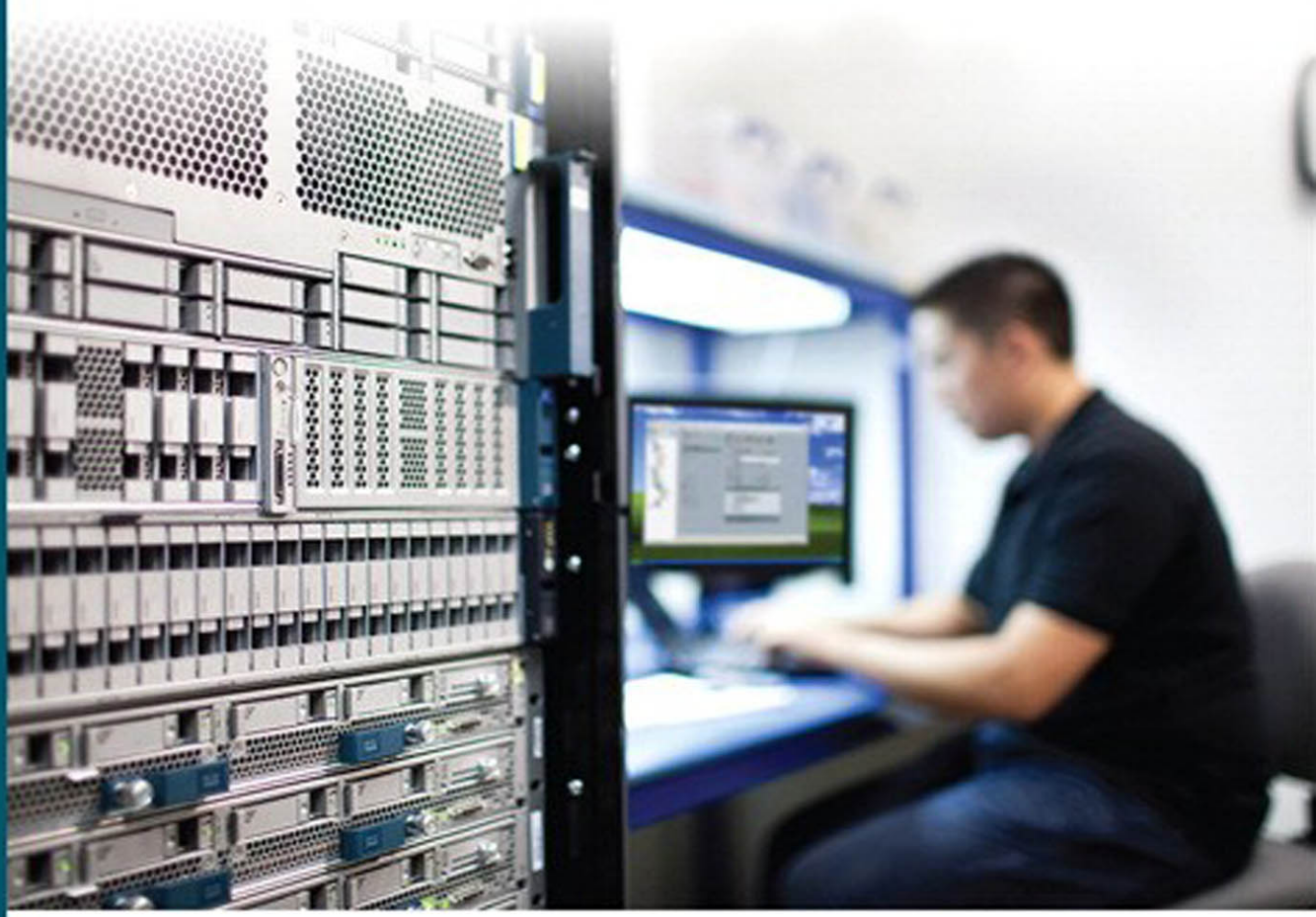
# Troubleshooting and Maintaining Cisco IP Networks (TSHOOT)

## Foundation Learning Guide



ciscopress.com

Amir Ranjbar

# Troubleshooting and Maintaining Cisco IP Networks (TSHOOT) Foundation Learning Guide

Amir Ranjbar,
CCIE No. 8669

**Cisco Press**

800 East 96th Street

Indianapolis, IN 46240

# Troubleshooting and Maintaining Cisco IP Networks (TSHOOT) Foundation Learning Guide

Amir Ranjbar



Published by:
Cisco Press
800 East 96th Street
Indianapolis, IN 46240 USA



Printed in the United States of America

Second Printing: January 2016

Library of Congress Control Number: 2014955936

ISBN-13: 978-1-58720-455-5

ISBN-10: 1-58720-455-X

## Warning and Disclaimer

This book is designed to provide information about the Troubleshooting and Maintaining Cisco IP Networks (TSHOOT) course, which is an element of the CCNP Routing and Switching certification curriculum. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as is" basis. The author, Cisco Press, and Cisco Systems, Inc. shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

## Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc., cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

## Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact international@pearsoned.com.

## Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

**Publisher:** Paul Boger

**Associate Publisher:** Dave Dusthimer

**Business Operation Manager, Cisco Press:** Jan Cornelssen

**Acquisitions Editor:** Mary Beth Ray

**Managing Editor:** Sandra Schroeder

**Development Editor:** Ellie Bru

**Senior Project Editor:** Tonya Simpson

**Copy Editor:** Keith Cline

**Technical Editor:** Ted Kim

**Team Coordinator:** Vanessa Evans

**Cover Designer:** Mark Shirar

**Composition:** Trina Wurst

**Indexer:** Lisa Stumpf

**Proofreader:** Debbie Williams

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.

## About the Author

**Amir Ranjbar, CCIE No. 8669**, is a Certified Cisco Systems Instructor and a senior network consultant. Operating under his own corporation, AMIRACAN Inc., Amir offers his training services to Global Knowledge Network, his consulting expertise to a variety of clients (mainly Internet service providers), and his technical writing skills to Cisco Press (Pearson Education, Inc.). Born in Tehran, Iran, Amir immigrated to Canada in 1983 at the age of 16 and completed his Master's degree in knowledge-based systems (a branch in artificial intelligence [AI]) in 1991. He has been involved in training, consulting, and technical writing for the greater part of his career. Amir Ranjbar can be contacted through his email address aranjbar@amiracan.com.

## About the Technical Reviewer

**Ted Kim, CCIE No. 22769** (Routing and Switching and Service Provider), has 10 years of experience in the IT industry, with a focus on data center technologies during the past several years. He has experience with designing, implementing, and troubleshooting large enterprise environments. Ted's networking career began at Johns Hopkins as a network engineer, and he has been with Cisco since 2013 as a network consulting engineer.

## Dedication

I dedicate this book to my father, Mr. Kavos Ranjbar, whom I lost on January 2, 2013. I wish we could all be so loving, helpful, and generous, yet humble, peaceful, and gentle, just like my dad.

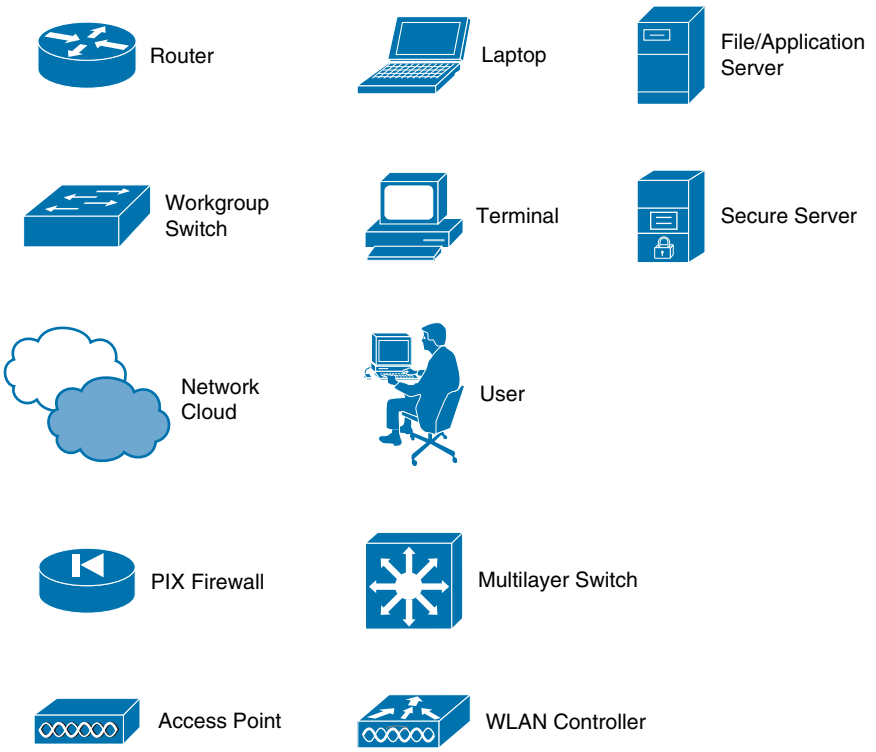
## Acknowledgments

This book is the result of work done by many individuals. I would like to offer my sincere gratitude to all of them, whether we worked together directly or otherwise. Mary Beth Ray, Ellie Bru, Tonya Simpson, Keith Cline, Vanessa Evans, Mark Shirar, Trina Wurst, and Lisa Stumpf, please accept my most sincere gratitude for the time and effort you put into this project. I wish I could attend the next Pearson Education social gathering and thank you all in person! Ted Kim, thank you for your technical review and feedback; I hope to meet you someday and thank you in person, too.

# Contents at a Glance

# Contents

## Icons Used in This Book




## Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the IOS Command Reference. The Command Reference describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italic* indicates arguments for which you supply actual values.
- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets ([ ]) indicate an optional element.
- Braces ({ }) indicate a required choice.
- Braces within brackets ([{ }]) indicate a required choice within an optional element.

# Introduction

This book is based on the Cisco Systems TSHOOT course, which was recently introduced as part of the CCNP curriculum. It provides troubleshooting and maintenance information and examples that relate to Cisco routing and switching. It is assumed that readers know and understand as much Cisco routing and switching background as covered in the Cisco ROUTE and SWITCH courses. The book is enough to prepare you for the TSHOOT exam, too.

Teaching troubleshooting is not an easy task. This book introduces you to many troubleshooting methodologies and identifies the benefits of different techniques. Technical routing and switching topics are briefly reviewed, but the emphasis is on troubleshooting commands, and most important, this book presents many troubleshooting examples. Chapter review questions will help you evaluate how well you absorbed material within each chapter. The questions are also an excellent supplement for exam preparation.

## Who Should Read This Book?

Those individuals who want to learn about modern troubleshooting methodologies and techniques and want to see several relevant examples will find this book very useful. This book is most suitable for those who have some prior routing and switching knowledge but would like to learn more or otherwise enhance their troubleshooting skill set. Readers who want to pass the Cisco TSHOOT exam can find all the content they need to successfully do so in this book. The Cisco Networking Academy CCNP TSHOOT course students will use this book as their official textbook.

## Cisco Certifications and Exams

Cisco offers four levels of routing and switching certification, each with an increasing level of proficiency: Entry, Associate, Professional, and Expert. These are commonly known by their acronyms CCENT (Cisco Certified Entry Networking Technician), CCNA (Cisco Certified Network Associate), CCNP (Cisco Certified Network Professional), and CCIE (Cisco Certified Internetworking Expert). There are others, too, but this book focuses on the certifications for enterprise networks.

For the CCNP certification, you must pass exams on a series of CCNP topics, including the SWITCH, ROUTE, and TSHOOT exams. For most exams, Cisco does not publish the scores needed for passing. You need to take the exam to find that out for yourself.

To see the most current requirements for the CCNP certification, go to Cisco.com and click **Training and Events**. There you can find out other exam details such as exam topics and how to register for an exam.

The strategy you use to prepare for the TSHOOT exam might differ slightly from strategies used by other readers, mainly based on the skills, knowledge, and experience you have already obtained. For instance, if you have attended the TSHOOT course, you might take a different approach than someone who learned troubleshooting through on-the-job training. Regardless of the strategy you use or the background you have, this book is designed to help you get to the point where you can pass the exam with the least amount of time required.

## How This Book Is Organized

Although this book can be read cover to cover, it is designed to be flexible and allow you to easily move between chapters to cover only the material for which you might need additional remediation. The chapters can be covered in any order, although some chapters are related to and build upon each other. If you do intend to read them all, the order in the book is an excellent sequence to follow.

Each core chapter covers a subset of the topics on the CCNP TSHOOT exam. The chapters cover the following topics:

- Chapter 1 introduces the troubleshooting principles and discusses the most common troubleshooting approaches.
- Chapter 2 defines structured troubleshooting and analyzes all the subprocesses of structured troubleshooting.
- Chapter 3 introduces structured network maintenance and discusses network maintenance processes and procedures. Network maintenance services and tools, along with how you can integrate troubleshooting into the network maintenance process, are also presented in this chapter.
- Chapter 4 reviews the Layer 2 switching and Layer 3 routing processes and shows how to do selective information gathering using the IOS **show** command, **debug** command, ping, and Telnet.
- Chapter 5 discusses troubleshooting tools: traffic-capturing features and tools, information gathering with SNMP, information gathering with NetFlow, and network event notification with EEM.
- Chapters 6 through 10 are all troubleshooting cases. Each chapter is about a different network with many different problems. Each problem is dealt with in the form of a real-life trouble ticket, and it is fixed following the structured troubleshooting methodology using the appropriate approach. All stages of troubleshooting, including fact gathering, are presented with output from Cisco IOS routers and switches. The network diagrams for Chapters 6 through 10 appear at the beginning and end of each chapter. For easier reference, a PDF of these network diagrams is available to download and print out or read on your e-device. Go to ciscopress.com/title/9781587204555 and click on the Downloads tab.

There is also an appendix that has answers to the review questions found at the end of each chapter.

Chapter 1

# Troubleshooting Methods

This chapter covers the following topics:

- Troubleshooting principles
- Common troubleshooting approaches
- Troubleshooting example using six different approaches

Most modern enterprises depend heavily on the smooth operation of their network infrastructure. Network downtime usually translates to loss of productivity, revenue, and reputation. Network troubleshooting is therefore one of the essential responsibilities of the network support group. The more efficiently and effectively the network support personnel diagnose and resolve problems, the lower impact and damages will be to business. In complex environments, troubleshooting can be a daunting task, and the recommended way to diagnose and resolve problems quickly and effectively is by following a structured approach. Structured network troubleshooting requires well-defined and documented troubleshooting procedures.

This chapter defines troubleshooting and troubleshooting principles. Next, six different troubleshooting approaches are described. The third section of this chapter presents a troubleshooting example based on each of the six troubleshooting approaches.

## Troubleshooting Principles

Troubleshooting is the process that leads to the diagnosis and, if possible, resolution of a problem. Troubleshooting is usually triggered when a person reports a problem. In modern and sophisticated environments that deploy proactive network monitoring tools and techniques, a failure/problem may be discovered and even fixed/resolved before end users notice or business applications get affected by it.

Some people say that a problem does not exist until it is noticed, perceived as a problem, and reported as a problem. This implies that you need to differentiate between a problem,

as experienced by the user, and the actual cause of that problem. The time a problem is reported is not necessarily the same time at which the event causing the problem happened. Also, the reporting user generally equates the problem to the symptoms, whereas the troubleshooter often equates the problem to the root cause. For example, if the Internet connection fails on Saturday in a small company, it is usually not a problem, but you can be sure that it will turn into a problem on Monday morning if it is not fixed before then. Although this distinction between symptoms and cause of a problem might seem philosophical, you need to be aware of the potential communication issues that might arise from it.

Generally, reporting of a problem triggers the troubleshooting process. Troubleshooting starts by defining the problem. The second step is diagnosing the problem, during which information is gathered, the problem definition is refined, and possible causes for the problem are proposed. Eventually, this process should lead to a hypothesis for the root cause of the problem. At this time, possible solutions need to be proposed and evaluated. Next, the best solution is selected and implemented. Figure 1-1 illustrates the main elements of a structured troubleshooting approach and the transition possibilities from one step to the next.

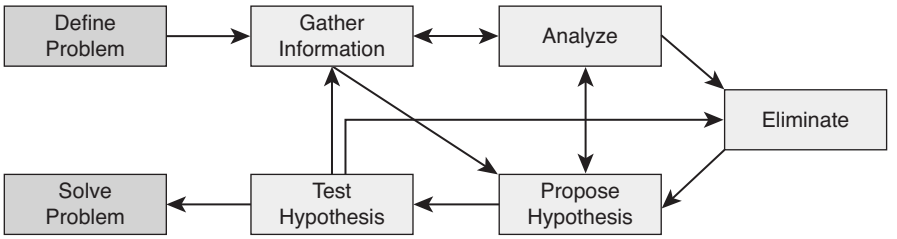



**Figure 1-1** *Flow Chart of a Structured Troubleshooting Approach*

**Note** It is noteworthy, however, that the solution to a network problem cannot always be readily implemented and an interim workaround might have to be proposed. The difference between a solution and a workaround is that a solution resolves the root cause of the problem, whereas a workaround only alleviates the symptoms of the problem.

Although problem reporting and resolution are definitely essential elements of the troubleshooting process, most of the time is spent in the diagnostic phase. One might even believe that diagnosis is all troubleshooting is about. Nevertheless, within the context of network maintenance, problem reporting and resolution are indeed essential parts of troubleshooting. Diagnosis is the process of identifying the nature and cause of a problem. The main elements of this process are as follows:

- **Gathering information:** Gathering information happens after the problem has been reported by the user (or anyone). This might include interviewing all parties (user) involved, plus any other means to gather relevant information. Usually, the problem report does not contain enough information to formulate a good hypothesis without first gathering more information. Information and symptoms can be gathered directly, by observing processes, or indirectly, by executing tests.

- **Analyzing information:** After the gathered information has been analyzed, the troubleshooter compares the symptoms against his knowledge of the system, processes, and baselines to separate normal behavior from abnormal behavior.
- **Eliminating possible causes:** By comparing the observed behavior against expected behavior, some of the possible problem causes are eliminated.
- **Formulating/proposing a hypothesis:** After gathering and analyzing information and eliminating the possible causes, one or more potential problem causes remain. The probability of each of these causes will have to be assessed and the most likely cause proposed as the hypothetical cause of the problem.
- **Testing the hypothesis:** The hypothesis must be tested to confirm or deny that it is the actual cause of the problem. The simplest way to do this is by proposing a solution based on this hypothesis, implementing that solution, and verifying whether this solved the problem. If this method is impossible or disruptive, the hypothesis can be strengthened or invalidated by gathering and analyzing more information.

All troubleshooting methods include the elements of gathering and analyzing information, eliminating possible causes, and formulating and testing hypotheses. Each of these steps has its merits and requires some time and effort; how and when one moves from one step to the next is a key factor in the success level of a troubleshooting exercise. In a scenario where you are troubleshooting a complex problem, you might go back and forth between different stages of troubleshooting: Gather some information, analyze the information, eliminate some of the possibilities, gather more information, analyze again, formulate a hypothesis, test it, reject it, eliminate some more possibilities, gather more information, and so on.

If you do not take a structured approach to troubleshooting and do troubleshooting in an ad hoc fashion, you might eventually find the solution; however, the process in general will be very inefficient. Another drawback of ad hoc troubleshooting is that handing the job over to someone else is very hard to do; the progress results are mainly lost. This can happen even if the troubleshooter wants to resume his own task after he has stopped for a while, perhaps to take care of another matter. A structured approach to troubleshooting, regardless of the exact method adopted, yields more predictable results in the long run. It also makes it easier to pick up where you left off or hand the job over to someone else without losing any effort or results.

A troubleshooting approach that is commonly deployed both by inexperienced and experienced troubleshooters is called shoot-from-the-hip. After a very short period of gathering information, taking this approach, the troubleshooter quickly makes a change to see if it solves the problem. Even though it may seem like random troubleshooting on the surface, it is not. The reason is that the guiding principle for this method is prior and usually vast knowledge of common symptoms and their corresponding causes, or simply extensive relevant experience in a particular environment or application. This technique might be quite effective for the experienced troubleshooter most times, but it usually does not yield the same results for the inexperienced troubleshooter. Figure 1-2 shows how the “shoot-from-the-hip” approach goes about solving a problem, spending almost no effort in analyzing the gathered information and eliminating possibilities.

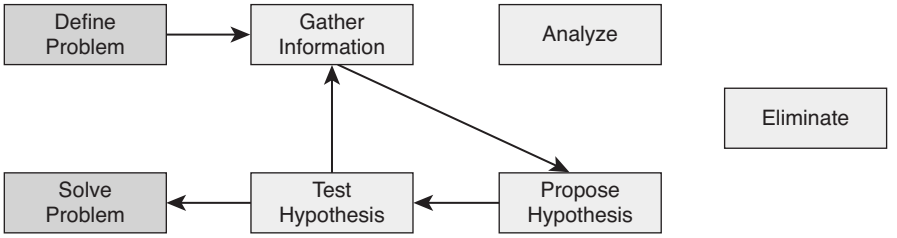



**Figure 1-2** *Shoot-from-the-Hip*

Assume that a user reports a LAN performance problem and in 90 percent of the past cases with similar symptoms, the problem has been caused by duplex mismatch between users' workstations (PC or laptop) and the corresponding access switch port. The solution has been to configure the switch port for 100-Mbps full duplex. Therefore, it sounds reasonable to quickly verify the duplex setting of the switch port to which the user connects and change it to 100-Mbps full duplex to see whether that fixes the problem. When it works, this method can be very effective because it takes very little time. Unfortunately, the downside of this method is that if it does not work, you have not come any closer to a possible solution, you have wasted some time (both yours and users'), and you might possibly have caused a bit of frustration. Experienced troubleshooters use this method to great effect. The key factor in using this method effectively is knowing when to stop and switch to a more methodical (structured) approach.

## Structured Troubleshooting Approaches

Troubleshooting is not an exact science, and a particular problem can be diagnosed and sometimes even solved in many different ways. However, when you perform structured troubleshooting, you make continuous progress, and usually solve the problem faster than it would take using an ad hoc approach. There are many different structured troubleshooting approaches. For some problems, one method might work better, whereas for others, another method might be more suitable. Therefore, it is beneficial for the troubleshooter to be familiar with a variety of structured approaches and select the best method or combination of methods to solve a particular problem.

A structured troubleshooting method is used as a guideline through a troubleshooting process. The key to all structured troubleshooting methods is systematic elimination of hypothetical causes and narrowing down on the possible causes. By systematically eliminating possible problem causes, you can reduce the scope of the problem until you manage to isolate and solve the problem. If at some point you decide to seek help or hand the task over to someone else, your findings can be of help to that person and your efforts are not wasted. Commonly used troubleshooting approaches include the following:

- **The top-down approach:** Using this approach, you work from the Open Systems Interconnection (OSI) model's application layer down to the physical layer. The OSI seven-layer networking model and TCP/IP four-layer model are shown side by side in Figure 1-3 for your reference.

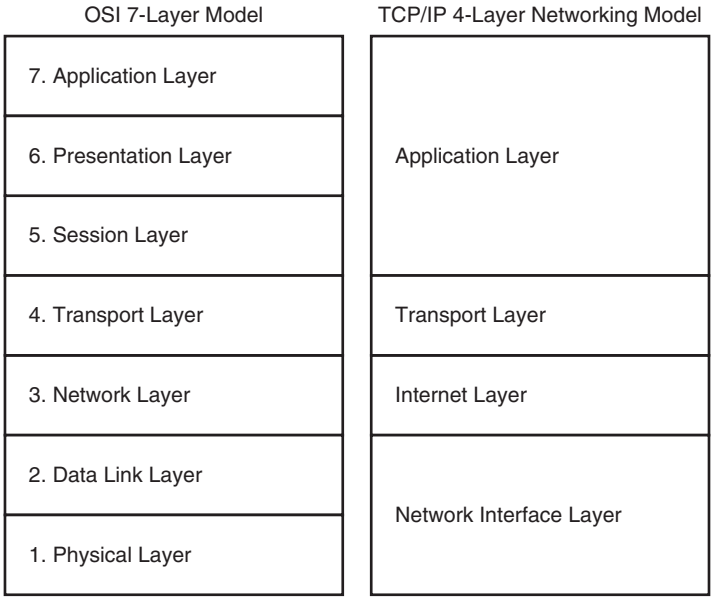



**Figure 1-3** *The OSI and TCP/IP Networking Models*

- **The bottom-up approach:** This approach starts from the OSI model's physical layer and moves up toward the application layer.
- **The divide-and-conquer approach:** Using this approach, you start in the middle of the OSI model's stack (usually the network layer), and then, based on your findings, you move up or down the OSI stack.
- **The follow-the-path approach:** This approach is based on the path that packets take through the network from source to destination.
- **The spot-the-differences approach:** As the name implies, this approach compares network devices or processes that are operating correctly to devices or processes that are not operating as expected and gathers clues by spotting significant differences. In case the problem occurred after a change on a single device was implemented, the spot-the differences approach can pinpoint the problem cause by focusing on the difference between the device configurations, before and after the problem was reported.
- **The move-the-problem approach:** The strategy of this troubleshooting approach is to physically move components and observe whether the problem moves with the moved components.

The sections that follow describe each of these methods in more detail.

## The Top-Down Troubleshooting Approach

The top-down troubleshooting method uses the OSI model as a guiding principle. One of the most important characteristics of the OSI model is that each layer depends on the underlying layers for its operation. This implies that if you find a layer to be operational, you can safely assume that all underlying layers are fully operational as well.

Let's assume that you are researching a problem of a user that cannot browse a particular website and you find that you can establish a TCP connection on port 80 from this host to the server and get a response from the server (see Figure 1-4). In this situation, it is reasonable to conclude that the transport layer and all layers below must be fully functional between the client and the server and that this is most likely a client or server problem (most likely at application, presentation, or session layer) and not a network problem. Be aware that in this example it is reasonable to conclude that Layers 1 through 4 must be fully operational, but it does not definitively prove this. For instance, nonfragmented packets might be routed correctly, whereas fragmented packets are dropped. The TCP connection to port 80 might not uncover such a problem.

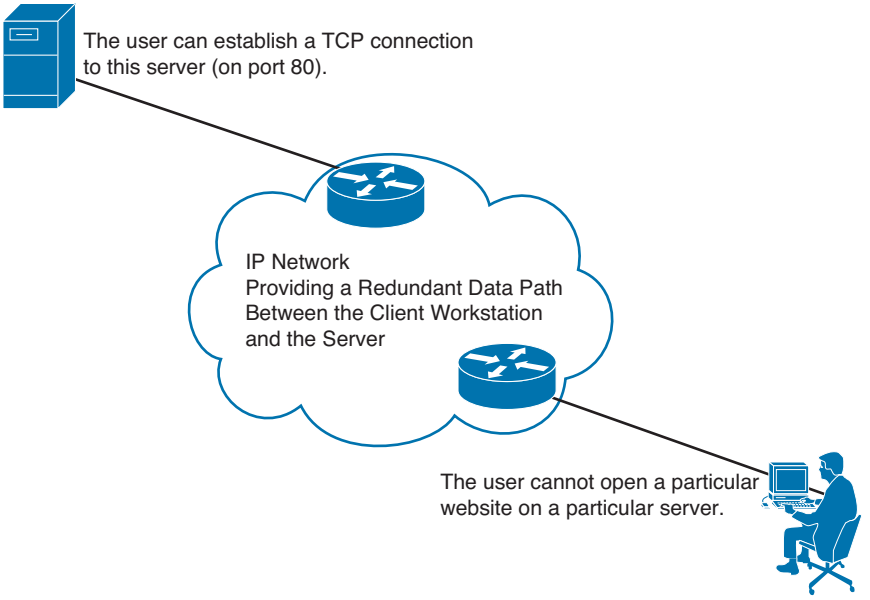



**Figure 1-4** *Application Layer Failure*

Essentially, the goal of the top-down approach is to find the highest OSI layer that is still working. All devices and processes that work on that layer or layers below are then eliminated from the scope of the troubleshooting. It might be clear that this approach is most effective if the problem is on one of the higher OSI layers. It is also one of the most straightforward troubleshooting approaches, because problems reported by users are typically defined as application layer problems, so starting the troubleshooting process at that layer is a natural thing to do. A drawback or impediment to this approach is

that you need to have access to the client's application layer software to initiate the troubleshooting process, and if the software is only installed on a small number of machines, your troubleshooting options might be limited.

## The Bottom-Up Troubleshooting Approach

The bottom-up troubleshooting approach also uses the OSI model as its guiding principle with the physical layer (bottom layer of the OSI seven-layer network model) as the starting point. In this approach, you work your way layer by layer up toward the application layer and verify that relevant network elements are operating correctly. You try to eliminate more and more potential problem causes so that you can narrow down the scope of the potential problems.

Let's assume that you are researching a problem of a user that cannot browse a particular website and while you are verifying the problem, you find that the user's workstation is not even able to obtain an IP address through the DHCP process (see Figure 1-5). In this situation it is reasonable to suspect lower layers of the OSI model and take a bottom-up troubleshooting approach.

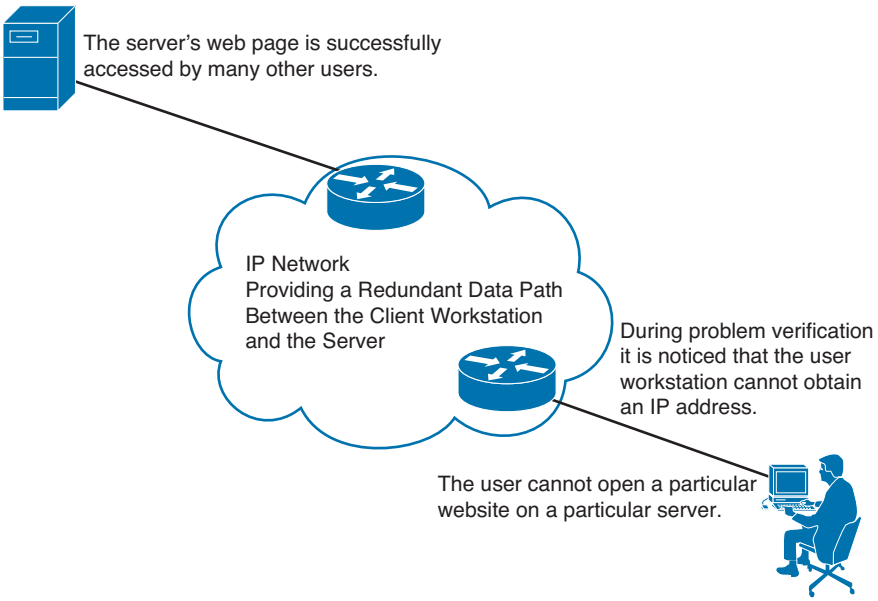



**Figure 1-5** *Failure at Lower OSI Layers*

A benefit of the bottom-up approach is that all the initial troubleshooting takes place on the network, so access to clients, servers, or applications is not necessary until a very late stage in the troubleshooting process. In certain environments, especially those where many old and outdated devices and technologies are still in use, many network problems

are hardware related. The bottom-up approach is very effective under those circumstances. A disadvantage of this method is that, in large networks, it can be a time-consuming process because a lot of effort will be spent on gathering and analyzing data and you always start from the bottom layer. The best bottom-up approach is to first reduce the scope of the problem using a different strategy and then switch to the bottom-up approach for clearly bounded parts of the network topology.

## The Divide-and-Conquer Troubleshooting Approach

The divide-and-conquer troubleshooting approach strikes a balance between the top-down and bottom-up troubleshooting approaches. If it is not clear which of the top-down or bottom-up approaches will be more effective for a particular problem, an alternative is to start in the middle (usually from the network layer) and perform some tests such as ping and trace. Ping is an excellent connectivity testing tool. If the test is successful, you can assume that all lower layers are functional, and so you can start a bottom-up troubleshooting starting from the network layer. However, if the test fails, you can start a top-down troubleshooting starting from the network layer.

Let's assume that you are researching a problem of a user who cannot browse a particular website and that while you are verifying the problem you find that the user's workstation can successfully ping the server's IP address (see Figure 1-6). In this situation, it is reasonable to assume that the physical, data link, and network layers of the OSI model are in good working condition, and so you examine the upper layers, starting from the transport layer in a bottom-up approach.

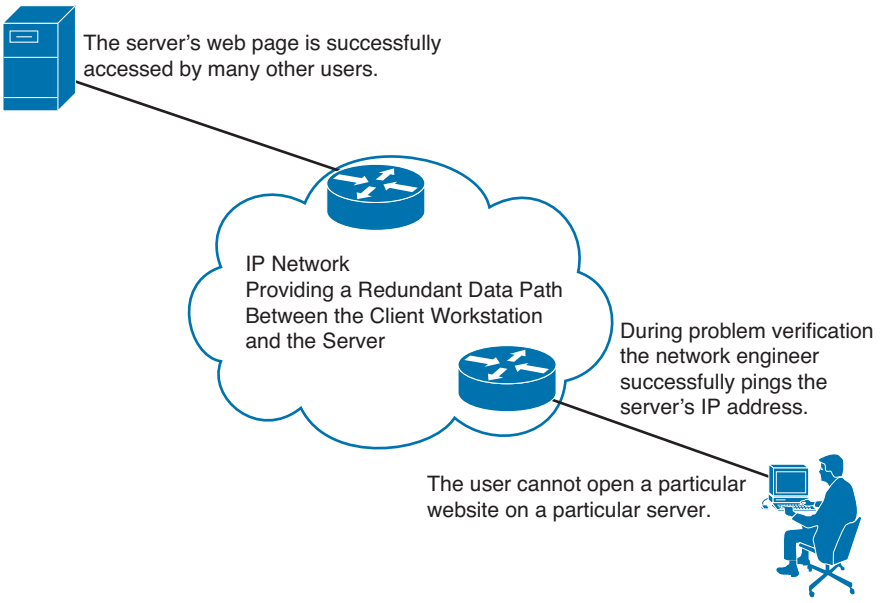



**Figure 1-6** *Successful Ping Shifts the Focus to Upper OSI Layers (Divide-and-Conquer Approach)*

Whether the result of the initial test is positive or negative, the divide-and-conquer approach usually results in a faster elimination of potential problems than what you would achieve by implementing a full top-down or bottom-up approach. Therefore, the divide-and-conquer method is considered highly effective and possibly the most popular troubleshooting approach.

## The Follow-the-Path Troubleshooting Approach

The follow-the-path approach is one of the most basic troubleshooting techniques, and it usually complements one of the other troubleshooting methods such as the top-down or the bottom-up approach. The follow-the-path approach first discovers the actual traffic path all the way from source to destination. Next, the scope of troubleshooting is reduced to just the links and devices that are actually in the forwarding path. The principle of this approach is to eliminate the links and devices that are irrelevant to the troubleshooting task at hand.

Let's assume that you are researching a problem of a user who cannot browse a particular website and that while you are verifying the problem you find that a trace (tracert) from the user's PC command prompt to the server's IP address succeeds only as far as the first hop, which is the L3 Switch v (Layer 3 or Multilayer Switch v) in Figure 1-7. Based on your understanding of the network link bandwidths and the routing protocol used on this network, you mark the links on the best path between the user workstation and the server on the diagram with numbers 1 through 7, as shown in Figure 1-7.

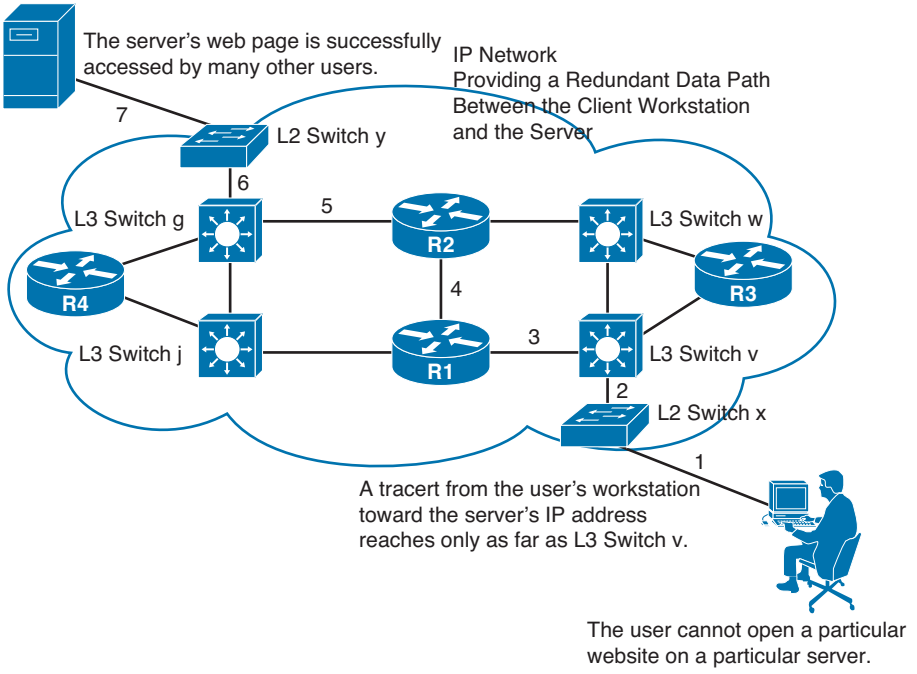



**Figure 1-7** *The Follow-the-Path Approach Shifts the Focus to Link 3 and Beyond Toward the Server*

In this situation it is reasonable to shift your troubleshooting approach to the L3 Switch v and the segments beyond, toward the server along the best path. The follow-the-path approach can quickly lead you to the problem area. You can then try and pinpoint the problem to a device, and ultimately to a particular physical or logical component that is either broken, misconfigured, or has a bug.

## The Compare-Configurations Troubleshooting Approach

Another common troubleshooting approach is called the compare-configurations approach, also referred to as the spotting-the-differences approach. By comparing configurations, software versions, hardware, or other device properties between working and nonworking situations and spotting significant differences between them, this approach attempts to resolve the problem by changing the nonoperational elements to be consistent with the working ones. The weakness of this method is that it might lead to a working situation, without clearly revealing the root cause of the problem. In some cases, you are not sure whether you have implemented a solution or a workaround.

Example 1-1 shows two routing tables; one belongs to Branch2's edge router, experiencing problems, and the other belongs to Branch1's edge router, with no problems. If you compare the content of these routing tables, as per the compare-configurations (spotting-the-differences) approach, a natural deduction is that the branch with problems is missing a static entry. The static entry can be added to see whether it solves the problem.

**Example 1-1** *Spot-the-Differences: One Malfunctioning and One Working Router*

```
------------- Branch1 is in good working order ----------
Branch1# show ip route
<...output omitted...>
10.0.0.0/24 is subnetted, 1 subnets
C   10.132.125.0 is directly connected, FastEthernet4
C   192.168.36.0/24 is directly connected, BVI1
S*  0.0.0.0/0 [254/0] via 10.132.125.1
------------- Branch2 has connectivity problems ----------
Branch2# show ip route
<...output omitted...>
10.0.0.0/24 is subnetted, 1 subnets
C 10.132.126.0 is directly connected, FastEthernet4
C 192.168.37.0/24 is directly connected, BVI1
```

The compare-configurations approach (spotting-the-differences) is not a complete approach; it is, however, a good technique to use undertaking other approaches. One benefit of this approach is that it can easily be used by less-experienced troubleshooting staff to at least shed more light on the case. When you have an up-to-date and accessible set of baseline configurations, diagrams, and so on, spotting the difference between the current configuration and the baseline might help you solve the problem faster than any other approach.

## The Swap-Components Troubleshooting Approach

Also called move-the-problem, the swap-components approach is a very elementary troubleshooting technique that you can use for problem isolation: You physically swap components and observe whether the problem stays in place, moves with the component, or disappears entirely. Figure 1-8 shows two PCs and three laptops connected to a LAN switch, among which laptop B has connectivity problems. Assuming that hardware failure is suspected, you must discover whether the problem is on the switch, the cable, or the laptop. One approach is to start gathering data by checking the settings on the laptop with problems, examining the settings on the switch, comparing the settings of all the laptops, and the switch ports, and so on. However, you might not have the required administrative passwords for the PCs, laptops, and the switch. The only data that you can gather is the status of the link LEDs on the switch and the laptops and PCs. What you can do is obviously limited. A common way to at least isolate the problem (if it is not solved outright) is cable or port swapping. Swap the cable between a working device and laptop B (the one that is having problems). Move the laptop from one port to another using a cable that you know for sure is good. Based on these simple moves, you can isolate whether the problem is cable, switch, or laptop related.

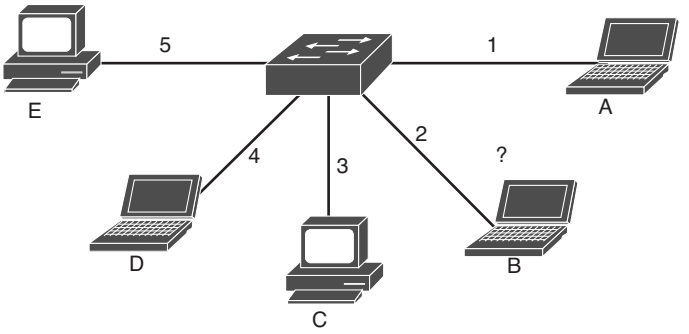



**Figure 1-8** *Swap-the-Component: Laptop B Is Having Network Problems*

Just by executing simple tests in a methodical way, the swap-components approach enables you to isolate the problem even if the information that you can gather is minimal. Even if you do not solve the problem, you have scoped it to a single element, and you can now focus further troubleshooting on that element. Note that in the previous example if you determine that the problem is cable related, it is unnecessary to obtain the administrative password for the switch, PCs, and laptops. The drawbacks of this method are that you are isolating the problem to only a limited set of physical elements and not gaining any real insight into what is happening, because you are gathering only very limited indirect information. This method assumes that the problem is with a single component. If the problem lies within multiple devices, you might not be able to isolate the problem correctly.

## Troubleshooting Example Using Six Different Approaches

An external financial consultant has come in to help your company's controller with an accounting problem. He needs access to the finance server. An account has been created for him on the server, and the client software has been installed on the consultant's laptop. You happen to walk past the controller's office and are called in and told that the consultant can't connect to the finance server. You are a network support engineer and have access to all network devices, but not to the servers. Think about how you would handle this problem, what your troubleshooting plan would be, and which method or combination of methods you would use.

What possible approaches can you take for this troubleshooting task? This case lends itself to many different approaches, but some specific characteristics can help you decide an appropriate approach:

- You have access to the network devices, but not to the server. This implies that you will likely be able to handle Layer 1–4 problems by yourself; however, for Layer 5–7, you will probably have to escalate to a different person.
- You have access to the client device, so it is possible to start your troubleshooting from it.
- The controller has the same software and access rights on his machine, so it is possible to compare between the two devices.

What are the benefits and drawbacks of each possible troubleshooting approach for this case?

- **Top-down:** You have the opportunity to start testing at the application layer. It is good troubleshooting practice to confirm the reported problem, so starting from the application layer is an obvious choice. The only possible drawback is that you will not discover simple problems, such as the cable being plugged in to a wrong outlet, until later in the process.
- **Bottom-up:** A full bottom-up check of the whole network is not a very useful approach because it will take too much time and at this point, there is no reason to assume that the network beyond the first access switch would be causing the issue. You could consider starting with a bottom-up approach for the first stretch of the network, from the consultant's laptop to the access switch, to uncover potential cabling problems.
- **Divide-and-conquer:** This is a viable approach. You can ping from the consultant's laptop to the finance server. If that succeeds, the problem is most likely at upper layers. For example, a firewall or access control list could be the culprit. If the ping fails, assuming that ping is not blocked in the network, it is safe to assume that the problem is at network or lower layers and you are responsible for fixing it. The advantage of this method is that you can quickly decide on the scope of the problem and whether escalation is necessary.

- **Follow-the-path:** Similar to the bottom-up approach, a full follow-the-path approach is not efficient under the circumstances, but tracing the cabling to the first switch can be a good start if it turns out that the link LED is off on the consultant's PC. This method might come into play after other techniques have been used to narrow the scope of the problem.
- **Compare-configurations:** You have access to both the controller's PC and the consultant's laptop; therefore, compare-configurations is a possible strategy. However, because these machines are not under the control of a single IT department, you might find many differences, and it might therefore be hard to spot the significant and relevant differences. The compare-configurations approach might prove useful later, after it has been determined that the problem is likely to be on the client.
- **Swap-components:** Using this approach alone is not likely to be enough to solve the problem, but if following any of the other methods indicates a potential hardware issue between the consultant's PC and the access switch, this method might come into play. However, merely as a first step, you could consider swapping the cable and the jack connected to the consultant's laptop and the controller's PC, in turn, to see whether the problem is cable, PC, or switch related.

Many combinations of these different methods could be considered here. The most promising methods are top-down or divide-and-conquer. You will possibly switch to follow-the-path or compare-configurations approach after the scope of the problem has been properly reduced. As an initial step in any approach, the swap-components method could be used to quickly separate client-related issues from network-related issues. The bottom-up approach could be used as the first step to verify the first stretch of cabling.

## Summary

The fundamental elements of a troubleshooting process are as follows:

- Defining the problem
- Gathering information
- Analyzing information
- Eliminating possible causes
- Formulating a hypothesis
- Testing the hypothesis
- Solving the problem

Some commonly used troubleshooting approaches are as follows:

- Top-down
- Bottom-up

- Divide-and-conquer
- Follow-the-path
- Compare-configurations
- Swap-components

## Review Questions

1. Which *three* of the following processes are subprocesses or phases of a troubleshooting process?

   **a.** Solve the problem
   **b.** Eliminate
   **c.** Compile
   **d.** Report the problem
   **e.** Define the problem

2. Which *three* of the following approaches are valid troubleshooting methods?

   **a.** Swap-components
   **b.** Ad Hoc
   **c.** Compare-configurations
   **d.** Follow-the-path
   **e.** Hierarchical

3. Which *three* of the following troubleshooting approaches use the OSI reference model as a guiding principle?

   **a.** Top-down
   **b.** Bottom-up
   **c.** Divide-and-conquer
   **d.** Compare-configurations
   **e.** Swap-components

4. Which of the following troubleshooting methods would be most effective when the problem is with the Ethernet cable connecting a workstation to the wall RJ-45 jack?

   **a.** Top-down
   **b.** Divide-and-conquer
   **c.** Compare-configurations
   **d.** Swap-components
   **e.** Follow-the-path

# Chapter 2

# Structured Troubleshooting

This chapter covers the following topics:

- Meaning of structured troubleshooting method and procedure
- The subprocesses of structured troubleshooting, the actions taken within each subprocess, and how and when you move from one to another progressively
- Troubleshooting example utilizing the structured troubleshooting method and procedures

Network troubleshooting is not an exact science, and no strict set of procedures, tasks, and steps available guarantees successful diagnosis and resolution of all networking problems in all situations. The troubleshooting process can be guided by structured methods, but it is not static, and its steps are not always the same and might not be executed in the exact same order every time. Each network has its own characteristics; there are an almost unlimited number of possible problems, and the skill set/experience of each troubleshooting engineer is unique. However, to guarantee a certain level of consistency in the way that problems are diagnosed and solved in an organization, it is quite important to identify the main subprocesses of structured troubleshooting and how one should move from one to another as the troubleshooting task progresses.

This chapter defines structured troubleshooting method and procedure, identifies the subprocesses of structured troubleshooting, suggests the order of executing these subprocesses, and specifies what tasks each subprocess consists of. This chapter concludes with an example that demonstrates a successful structured troubleshooting effort using the troubleshooting subprocesses covered, and executing them in the order as suggested in this chapter.

## Troubleshooting Method and Procedure

The generic troubleshooting process consists of the following tasks (subprocesses):

1. Defining the problem
2. Gathering information
3. Analyzing the information
4. Eliminating potential causes
5. Proposing a hypothesis (likely cause of the problem)
6. Testing and verifying validity of the proposed hypothesis
7. Solving the problem and documenting the work

A network troubleshooting process can be reduced to a number of elementary subprocesses, as outlined in the preceding list. These subprocesses are not strictly sequential in nature, and many times you will go back and forth through many of these subprocesses repeatedly until you eventually reach the solve the problem stage. Figure 2-1 illustrates the order of deploying the tasks/subprocesses within a structured troubleshooting process using a flowchart. A troubleshooting method provides a guiding principle that helps you move through these processes in a structured way. There is no exact recipe for troubleshooting. Every problem is different, and it is impossible to create a script for all possible problem scenarios.

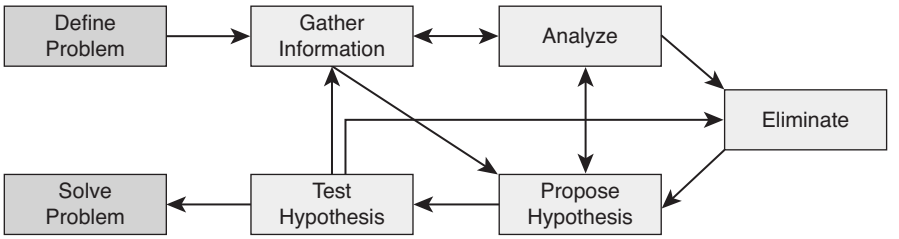



**Figure 2-1** *Flow Chart of a Structured Troubleshooting Approach*

Troubleshooting is a skill that requires relevant knowledge and experience. After using different methods several times, you will become more effective at selecting the right method for a particular problem, gathering the most relevant information, and analyzing problems quickly and efficiently. As you gain more experience, you will find that you can skip some steps and adopt more of a shoot-from-the-hip approach, resolving problems more quickly. Regardless, to execute a successful troubleshooting exercise, you must be able to answer the following questions:

- What is the action plan for each of the elementary subprocesses or phases?
- What is it that you actually do during each of those subprocesses?
- What decisions do you need to make?
- What kind of support or resources do you need?
- What kind of communication needs to take place?
- How do you delegate responsibilities properly?

Although the answers to these questions will differ for each individual organization, by planning, documenting, and implementing troubleshooting procedures, the consistency and effectiveness of the troubleshooting processes in your organization will improve.

## Defining the Problem

All troubleshooting tasks begin with defining the problem. However, what triggers a troubleshooting exercise is a failure experienced by someone who reports it to the support group. Figure 2-2 illustrates reporting of the problem (done by the user) as the trigger action, followed by verification and defining the problem (done by support group). Unless an organization has a strict policy on how problems are reported, the reported problem can unfortunately be vague or even misleading. Problem reports can look like the following: "When I try to go to this location on the intranet, I get a page that says I don't have permission," "The mail server isn't working," or "I can't file my expense report." As you might have noticed, the second statement is merely a conclusion a user has drawn perhaps merely because he cannot send or receive e-mail. To prevent wasting a lot of time during the troubleshooting process based on false assumptions and claims, the first step of troubleshooting is always verifying and defining the problem. The problem must first be verified, and then defined by you (the support engineer, not the user), and it has to be defined clearly. A good problem definition must also include information about when the problem started, if there have been any recent changes or upgrades, and how widespread the problem is. Knowing that the problem is experienced by a single user only (and not others) or that the problem has affected a group of users is quite valuable; it affects your analysis (eliminating causes and formulating hypotheses about the root cause of the problem) and your choice of the troubleshooting approach.

A good problem description consists of an accurate statement of symptoms and not of interpretations or conclusions. Consequences for the user are, strictly speaking, not part of the problem description itself, but can prove helpful to assess the urgency of the issue. When a problem is reported as "The mail server isn't working," you must contact the user and find out exactly what he has experienced. You will probably define the problem as "When user X starts his e-mail client, he gets an error message saying that the client cannot connect to the server. The user can still access his network drives and browse the Internet."

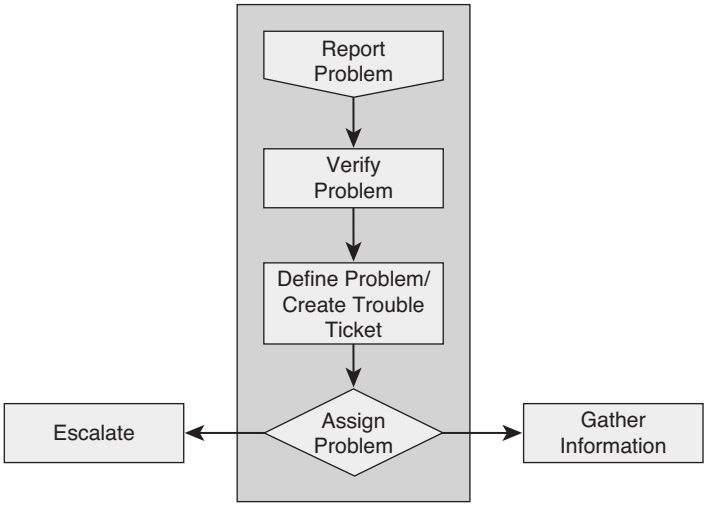



**Figure 2-2** *Defining the Problem: The Problem Must First Be Verified and Then Defined by Support Staff*

After you have clearly defined the problem and created the trouble ticket, you have one more step to take before starting the actual troubleshooting process. You must determine whether this problem is your responsibility or if it needs to be escalated to another department or person. For example, assume that the reported problem is this: "When user Y tries to access the corporate directory on the company intranet, she gets a message that says permission is denied. She can access all other intranet pages." You are a network engineer, and you do not have access to the servers. A separate department in your company manages the intranet servers. Therefore, you must know what to do when this type of problem is reported to you as a network problem. You must know whether to start troubleshooting or to escalate it to the server department. It is important that you know which types of problems are your responsibility to act on, what minimal actions you need to take before you escalate a problem, and how you escalate a problem. As Figure 2-2 illustrates, after defining the problem, you assign the problem: The problem is either escalated to another group or department, or it is network support's responsibility to solve it. In the latter case, the next step is gathering information.

## Gathering Information

Before gathering information, you should select your initial troubleshooting method and develop an information-gathering plan. As part of this plan, you need to identify what the targets are for the information-gathering process. In other words, you must decide which devices, clients, or servers you want to collect information from or about, and what tools you intend to use to gather that information (assemble a toolkit). Next, you have to acquire access to the identified targets. In many cases, you might have access to these systems as a normal part of your job role; in some cases, however, you might need to get information from systems that you cannot normally access. In this case, you might

have to escalate the issue to a different department or person, either to obtain access or to get someone else to gather the information for you. If the escalation process would slow the procedure down and the problem is urgent, you might want to reconsider the troubleshooting method that you selected and first try a method that uses different targets and would not require you to escalate. As you can see in Figure 2-3, whether you can access and examine the devices you identified will either lead to the problem's escalation to another group or department or to the analyzing the information step.

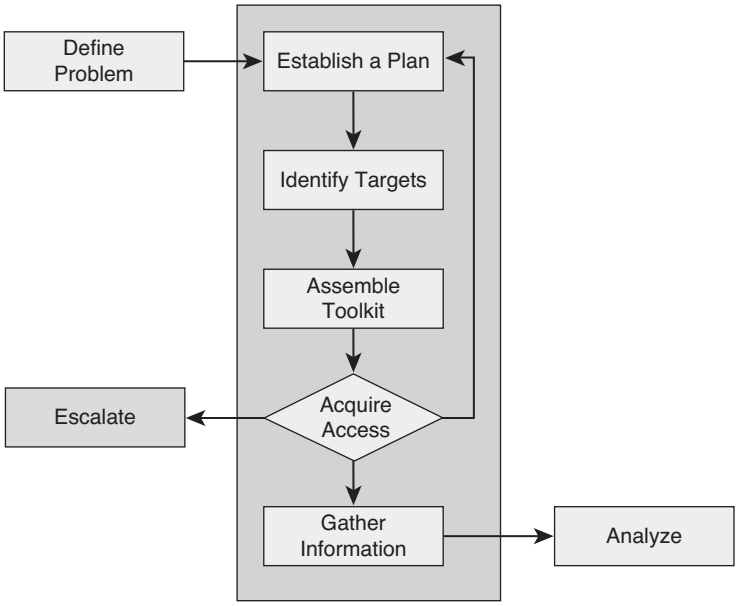



**Figure 2-3** *Gathering Information: Lack of Access to Devices Might Lead to Problem Escalation to Another Group.*

The example that follows demonstrates how information gathering can be influenced by factors out of your control, and consequently, it can force you to alter your troubleshooting approach. Imagine that it is 1:00 p.m. now and your company's sales manager has reported that he cannot send or receive e-mail from the branch office where he is working. The matter is quite urgent because he has to send out a response to an important request for proposal (RFP) later this afternoon. Your first reaction might be to start a top-down troubleshooting method by calling him up and running through a series of tests. However, the sales manager is not available because he is in a meeting until 4:30 p.m. One of your colleagues from that same branch office confirms that the sales manager is in a meeting, but left his laptop on his desk. The RFP response needs to be received by the customer before 5:00 p.m. Even though a top-down troubleshooting approach might seem like the best choice, because you will not be able to access the sales manager's laptop, you will have to wait until 4:30 before you can start troubleshooting. Having to perform an entire troubleshooting exercise successfully in about 30 minutes is risky, and it will put you under a lot of pressure. In this case, it is best if you use a combination

of the bottom-up and follow-the-path approaches. You can verify whether there are any Layer 1–3 problems between the manager's laptop and the company's mail server. Even if you do not find an issue, you can eliminate many potential problem causes, and when you start a top-down approach at 4:30, you will be able to work more efficiently.

## Analyzing the Information

After gathering information from various devices, you must interpret and analyze the information. To interpret the raw information that you have gathered (for example, the output of **show** and **debug** commands, or packet captures and device logs), you might need to research commands, protocols, and technologies. You might also need to consult network documentation to be able to interpret the information in the context of the actual network's implementation.

During the analysis of the gathered information, you are typically trying to determine two things: What is happening on the network and what should be happening. If you discover differences between these two, you can collect clues for what is wrong or at least a direction to take for further information gathering. Figure 2-4 shows that the gathered information, network documentation, baseline information, plus your research results and past experience are all used as input while you interpret and analyze the gathered information to eliminate possibilities and identify the source of the problem.

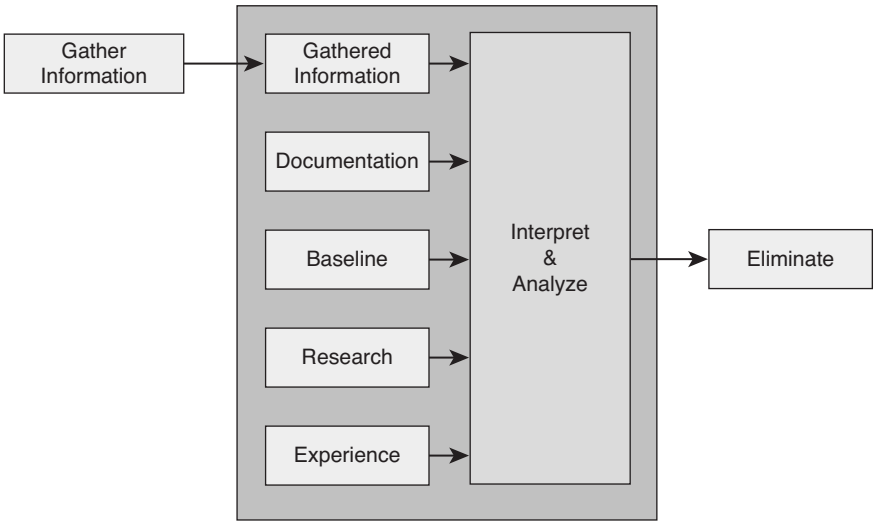



**Figure 2-4** *Analyze the Information: Gathered and Existing Information, Knowledge, and Experiences All Considered and Incorporated*

Your perception of what is actually happening is usually formed based on interpretation of the raw data, supported by research and documentation; however, your understanding of the underlying protocols and technologies also plays a role in your success level. If you are troubleshooting protocols and technologies that you are not very familiar with, you will have to invest some time in researching how they operate. Furthermore,

a good baseline of the behavior of your network can prove quite useful at the analysis stage. If you know how your network performs and how things work under normal conditions, you can spot anomalies in the behavior of the network and derive clues from those deviations. The benefit of vast relevant past experience cannot be overstated. An experienced network engineer will spend significantly less time (than an inexperienced engineer would) on researching processes, interpreting raw data, and distilling the relevant information from the raw data.

## Eliminating Potential Causes

Analyzing the gathered information while considering and incorporating existing information, such as network baseline and documentation, helps you eliminate many potential causes. For example, if a user can successfully ping a certain web server but cannot retrieve its main web page, you will comfortably eliminate many potential problem causes such as physical, data link, and network layer failures or misconfigurations. As Figure 2-5 illustrates, based on the gathered information and any assumptions made, from among all potential causes some are eliminated, leaving other potential causes to be evaluated and proposed in the next step.

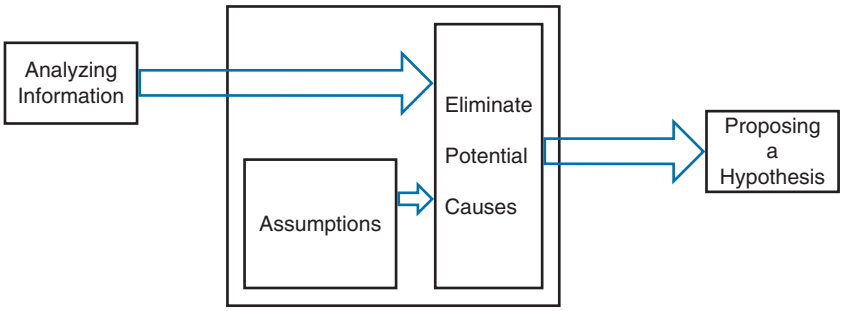



**Figure 2-5** *Eliminating Potential Causes: Gathered Information and Assumptions Help You Eliminate Some of the Potential Problem Causes*

You must make note of the important influence that your assumptions have in eliminating the potential causes. Assumptions may or may not be true. If you end up with conflicting conclusions or scenarios that make no sense, you might have to reevaluate your assumptions by gathering more information and analyzing the new information accordingly. For example, if you start troubleshooting a user's inability to access or use a particular service based on the false assumption that he or she could use or access the same service in the past, you might waste a long time in the analysis stage and draw no rational conclusion.

## Proposing a Hypothesis (Likely Cause of the Problem)

After eliminating potential problem causes, you are usually left with other potential causes. These potential causes must be ordered based on their likelihood, so that the most likely cause can form the proposed hypothesis. Ordering the remaining potential

causes based on their likelihood is once again dependent on your knowledge, past experiences, and assumptions. Figure 2-6 shows that the most likely cause that you propose may or may not lie within your territory or area of responsibility, and it may have to be escalated to another group or department. Figure 2-6 also shows that once the most likely cause has been determined, further fact gathering may be necessary, effectively triggering a new round of analysis, elimination, and proposing a hypothesis.

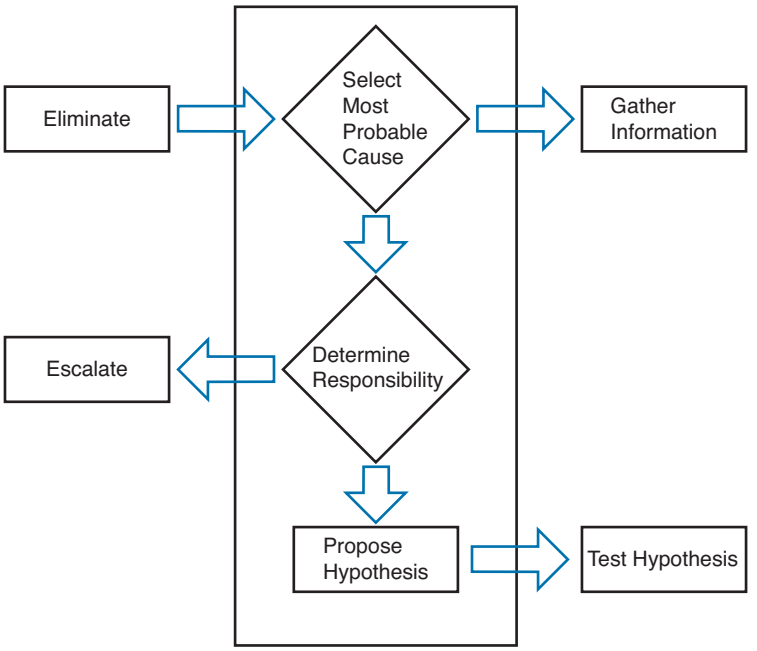



**Figure 2-6** *Proposing a Hypothesis: Selecting the Most Probable Cause of the Problem*

The proposed hypothesis leads us to the next stage of the structured troubleshooting process: testing the hypothesis. If at that stage the most probable hypothesis is determined not to be the potential cause, we normally propose the next most likely potential problem and once again go to the test hypothesis stage. This cycle may continue until the problem is solved or we exhaust all possible potential causes without solving the problem. In the latter case, we need to gather more facts and effectively restart the troubleshooting cycle, or we merely have to escalate the problem to more experienced staff or contact external sources such as consulting companies or the Cisco Technical Assistance Center (TAC).

If you decide to escalate the problem, ask yourself whether this ends your involvement in the process. Note that escalating the problem is not the same as solving the problem. You have to think about how long it will take the other party to solve the problem and how urgent the problem is to them. Users affected by the problem might not be able to wait long for the other group to fix the problem. If you cannot solve the problem, but it is too urgent to wait for the problem to be solved through an escalation, you might need to come up with a workaround. A temporary fix alleviates the symptoms experienced by the user, even if it does not address the root cause of the problem.

## Testing and Verifying Validity of the Proposed Hypothesis

After a hypothesis is proposed identifying the cause of a problem, the next step is to come up with a possible solution (or workaround) to that problem, and plan an implementation scheme. Usually, implementing a possible solution involves making changes to the network. Therefore, if your organization has defined procedures for regular network maintenance, you must follow your organization's regular change procedures. The next step is to assess the impact of the change on the network and balance that against the urgency of the problem. If the urgency outweighs the impact and you decide to go ahead with the change, it is important to make sure that you have a way to revert to the original situation after you make the change. Even though you have determined that your hypothesis is the most likely cause of the problem and your solution is intended to fix it, you can never be entirely sure that your proposed solution will actually solve the problem. If the problem is not solved, you need to have a way to undo your changes and revert to the original situation. Upon creation of a rollback plan, you can implement your proposed solution according to your organization's change procedures. Verify that the problem is solved and that the change you made did what you expected it to do. In other words, make sure the root cause of the problem and its symptoms are eliminated, and that your solution has not introduced any new problems. If all results are positive and desirable, you move on to the final stage of troubleshooting, which is integrating the solution and documenting your work. Figure 2-7 shows the flow of tasks while you implement and test your proposed hypothesis and either solve the problem or end up rolling back your changes.

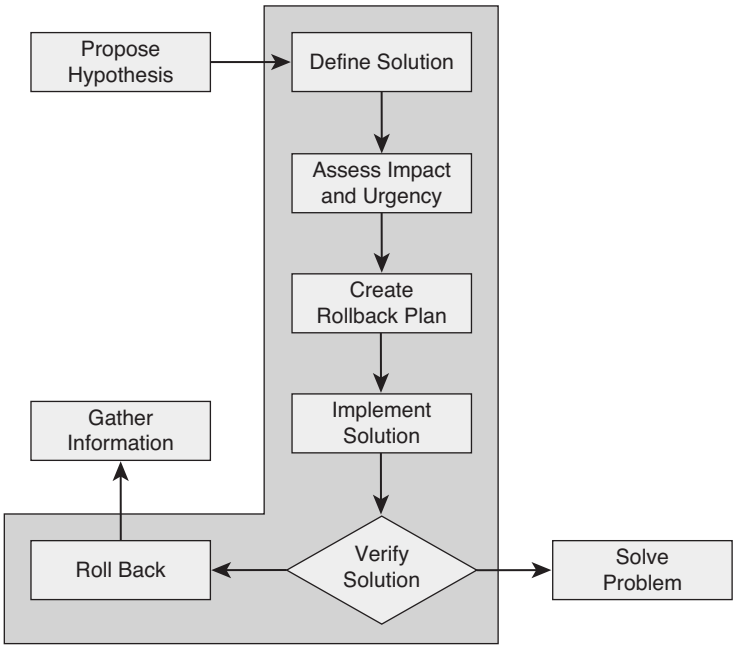



**Figure 2-7** *Test the Proposed Hypothesis*

You must have a plan for the situation if it turns out that the problem was not fixed, the symptoms have not disappeared, or new problems have been introduced by the changes that you have made. In this case, you should execute your rollback plan, revert to the original situation, and resume the troubleshooting process. It is important to determine whether the root cause hypothesis was invalid or whether it was simply the proposed solution that did not work.

## Solving the Problem and Documenting the Work

After you have confirmed your hypothesis and verified that the symptoms have disappeared, you have essentially solved the problem. All you need to do then is to make sure that the changes you made are integrated into the regular implementation of the network and that any maintenance procedures associated with those changes are executed. You will have to create backups of any changed configurations or upgraded software. You will have to document all changes to make sure that the network documentation still accurately describes the current state of the network. In addition, you must perform any other actions that are prescribed by your organization's change control procedures. Figure 2-8 shows that upon receiving successful results from testing your hypothesis, you incorporate your solution and perform the final tasks such as backup, documentation, and communication, before you report the problem as solved. Note that modern troubleshooting practices require that once the cause of a problem is determined and a solution has been implemented, a recommendation is to be made on how to eliminate or reduce occurrence of similar problems in the future.

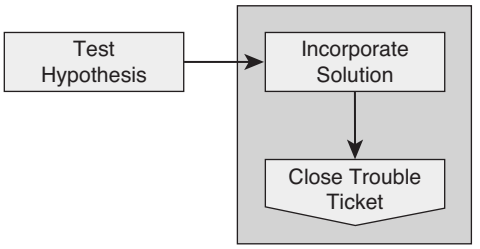



**Figure 2-8** *Solve the Problem and Document the Work*

Your troubleshooting job is not complete until you communicate that the problem has been solved. At a minimum, you will have to communicate back to the original user that reported the problem; if you have involved others as part of an escalation process, however, you should communicate with them, too. For any of the processes and procedures described here, each organization must make its own choices in how much of these procedures should be described, formalized, and followed. However, anyone involved in troubleshooting will benefit from reviewing these processes and comparing them to their own troubleshooting habits.

## Troubleshooting Example Based on the Structured Method and Procedures

Armando, a member of the network support team at AMIRACAN, Inc., receives a call from Ariana, who works for the accounting department. Ariana complains that the Internet is not accessible from her office workstation; she was trying to access www.cisco.com. At this stage, the problem is reported but the troubleshooting process has not begun yet. Armando followed the structured troubleshooting process, step by step, until he solved the problem and documented his work.

- **Define the problem:** Armando decided to verify the problem by going to Ariana's office. While there, he also found out that Ariana was able to access www.cisco.com yesterday. Armando creates a trouble ticket in the system and defines the problem by accurately stating Ariana's problem along with its date of occurrence, specifying that the problem was verified, and that the problem did not exist 24 hours ago.
- **Gathering information:** Armando decides to access www.cisco.com from his workstation in his office and is successful in doing so. Based on this gathered fact, Armando decides to take a bottom-up approach and start his work from Ariana's office using her workstation. Using Ariana's workstation, Armando notices that the workstation has an IP address, subnet mask, default gateway, and a DNS server address. Armando pings the configured DNS server's address and the ping succeeds 100 percent. However, because no web page can be retrieved from Ariana's workstation, Armando decides to use nslookup to see whether the DNS server returns proper IP addresses for some known URLs. Name resolution fails through the configured DNS address. Armando compares the DNS address configured on Ariana's workstation with the DNS address configured on other workstations in the accounting department, and notices that the DNS address on Ariana's workstation is different from all others.
- **Analyzing the information:** Knowing that Ariana's workstation is the only workstation in the accounting department that cannot access web pages using names, and that her workstation's DNS server address is different from everyone else's, and that the DNS server does not respond to nslookup, Armando associates Ariana's problem with the configured DNS server. Armando refers to the documentation related to user workstations and finds that the DNS server address must be assigned by the DHCP server.
- **Eliminate potential causes:** Armando eliminates physical and data link layer problems.
- **Propose a hypothesis:** Armando suspects that either Ariana's IP information has all been configured manually, or that basic IP information is gathered through Dynamic Host Configuration Protocol (DHCP), but other information, such as DNS server address, has been entered manually (and incorrectly). Armando decides that the invalid DNS address entered manually (and erroneously) is the most likely problem cause.

- **Test the hypothesis:** Armando modifies Ariana's workstation configuration to obtain its DNS server address through DHCP. Then he checks the result by attempting to access www.cisco.com. He is successful.
- **Solve the problem and document the work:** Armando enters the solution into the system for this trouble ticket and closes the case. Armando then explains to Ariana that the DNS server IP address must not be entered manually. He explains that using unknown third-party DNS servers can impose serious security threats, too. In the documentation related to the trouble tickets, Armando recommends that users' accounts should not have the privilege to change system settings.

## Summary

The generic troubleshooting process consists of the following tasks (subprocesses):

1. Defining the problem
2. Gathering information
3. Analyzing the information
4. Eliminating potential causes
5. Proposing a hypothesis (likely cause of the problem)
6. Testing and verifying validity of the proposed hypothesis
7. Solving the problem and documenting the work

A structured approach to troubleshooting (no matter what the exact method is) will yield more predictable results in the long run and will make it easier to pick up the process where you left off in a later stage or to hand it over to someone else.

The structured troubleshooting begins with problem definition followed by fact gathering. The gathered information, network documentation, baseline information, plus your research results and past experience are all used as input while you interpret and analyze the gathered information to eliminate possibilities and identify the source of the problem. Based on your continuous information analysis and the assumptions you make, you eliminate possible problem causes from the pool of proposed causes until you have a final proposal that takes you to the next step of the troubleshooting process: formulating and proposing a hypothesis. Based on your hypothesis, the problem might or might not fall within your area of responsibility, so proposing a hypothesis is either followed by escalating it to another group or by testing your hypothesis. If your test results are positive, you have to plan and implement a solution. The solution entails changes that must follow the change-control procedures within your organization. The results and all the changes you make must be clearly documented and communicated with all the relevant parties.

## Review Questions

1. Which *three* of the following processes are subprocesses of a structured troubleshooting process?
   - **a.** Eliminate potential causes
   - **b.** Test the hypothesis
   - **c.** Termination
   - **d.** Define the problem
   - **e.** Calculation
   - **f.** Compilation
2. Which *two* of the following resources will help in interpreting and analyzing information gathered during troubleshooting?
   - **a.** Documentation
   - **b.** Network baseline
   - **c.** Packet sniffers
   - **d.** Assumptions
3. Which of the following steps are parts of testing a hypothesis? (Choose four.)
   - **a.** Defining a solution
   - **b.** Creating a rollback plan
   - **c.** Implementing the solution
   - **d.** Defining the problem
   - **e.** Assessing impact and urgency
4. During which *three* of the troubleshooting phases could it be necessary to escalate a problem to a different department?
   - **a.** Defining the problem
   - **b.** Gathering information
   - **c.** Analyzing the facts
   - **d.** Eliminating possible causes
   - **e.** Proposing a hypothesis
   - **f.** Solving the problem