640-801 Lab

Basic Router Operation
IP Addressing
IP Routing
ISDN and DDR
LAN Switching
Network Management
Network Security
Other VPNs
VLANs
WAN Protocols

Exam Description

The CCNA exam is the qualifying exam available to candidates pursuing a single-exam option for the Cisco Certified Network Associate CCNA certification. The CCNA (640-801) exam will test materials from the new Interconnection Cisco Network Devices (ICND) course as well as the new Introduction to Cisco Networking Technologies (INTRO) course. The exam will certify that the successful candidate has important knowledge and skills necessary to select, connect, configure, and troubleshoot the various Cisco networking devices. The exam covers topics on Extending Switched Networks with VLANS, Determining IP Routes, Managing IP traffic with Access Lists, Establishing Point-to-Point connections, and Establishing Frame Relay Connections.

Exam Topics

The following information provides general guidelines for the content likely to be included on the Introducing Cisco Network Design Exam. However, other related topics may also appear on any specific delivery of the exam.

Planning & Designing

- Design a simple LAN using Cisco Technology
- Design an IP addressing scheme to meet design requirements
- Select an appropriate routing protocol based on user requirements
- Design a simple internetwork using Cisco technology
- Develop an access list to meet user specifications
- Choose WAN services to meet customer requirements

Implementation & Operation

- Configure routing protocols given user requirements
- Configure IP addresses, subnet masks, and gateway addresses on routers and hosts
- Configure a router for additional administrative functionality
- Configure a switch with VLANS and inter-switch communication
- Implement a LAN
- Customize a switch configuration to meet specified network requirements
- Manage system image and device configuration files
- Perform an initial configuration on a router
- Perform an initial configuration on a switch
- Implement access lists
- Implement simple WAN protocols

Troubleshooting

- Utilize the OSI model as a guide for systematic network troubleshooting
- Perform LAN and VLAN troubleshooting
- Troubleshoot routing protocols
- Troubleshoot IP addressing and host configuration
- Troubleshoot a device as part of a working network
- Troubleshoot an access list
- Perform simple WAN troubleshooting

Technology

- Describe network communications using layered models
- Describe the Spanning Tree process
- Compare and contrast key characteristics of LAN environments
- Evaluate the characteristics of routing protocols
- Evaluate TCP/IP communication process and its associated protocols
- Describe the components of network devices

- Evaluate rules for packet control
- Evaluate key characteristics of WANs

640-801 Labs

640-801 Labs

Basic Router Operation Lab

Tutorial
Lab Abstract
Lab Scenario

Basic Router Operation (IOS)

The purpose of this tutorial is to introduce you to Cisco router operation. This tutorial will also help you study for several topic areas that may be tested in Cisco certification exams including IOS basics, Router CLI, and troubleshooting.

This tutorial is divided into four sections: Overview of Cisco Router Hardware and Software, Basic Router IOS, Basic Router Configuration, and Basic Router Maintenance and Troubleshooting. There are also Appendices that include review questions (with answers) and additional material that you may find useful.

640-801 Labs

Basic Router Operation
Tutorial
Lab Abstract
Lab Scenario

Basic Router Operation Labs

Basic Router Operation Lab

Tutorial
Lab Abstract
Lab Scenario

Basic Router Operation

Introduction
Overview of Cisco Router Hardware and Software
  Hardware and Software Components of the Cisco 2501 Router
  RAM, ROM, NVRAM, and Flash in Cisco Routers
  Router Boot Sequence
  IOS Options
Basic Router IOS
  Router Access through the Console Port
  Router Access through Telnet
  Command Interpreter
    User Mode
    Privileged Mode
  CLI Help
    Inline Help -- Words
    Inline Help -- Command Syntax
    Command Line Completion
    Syntax Checking
    Hot Keys Used for Editing
    Advanced Editing
    Command History
Basic Router Configuration
  Setup Mode
  Manual Router Configuration
  Setting Router Passwords
    The Console Password
    The Enable Password and Enable Secret Password
    The VTY Password
    The Auxiliary Line Password
  Configuring an IP Address
  Configuring Banners
  Committing Configuration Changes to NVRAM
  Configuring Clock Rate
Basic Router Maintenance and Troubleshooting
  Backing up Router Configuration Files
  Backing Up IOS Software Images
  Show Commands
  Basic Show Commands
    Show Version
    Show Interfaces
    Show Protocols
    Show Flash
  Advanced Show Commands
    Show Memory
    Show Processes
    Show Stack
    Show Buffers
    Show Processes CPU
    Show CDP Neighbors
  Debug Commands
  Fallback
  The Configuration Register
    Changing the Configuration Register

Booting from ROM (Boot Field = 01)
Booting from Flash (Boot Field = 02)
Appendix A. Review Questions and Answers
Appendix B. Cisco Router Series
Series 700
Series 1600
Series 2500
Single LAN
Dual LAN
Router/Hub Combo
Access Servers
Series 2600
Series 3600
Series 4000
AGS+ and Series 7000
Series 7100 and 7200
Series 7500 and 10000
Series 12000

Introduction

Today Cisco Systems(r) has become the world's foremost developer and manufacturer of internetworking equipment and software. Cisco(r) develops and manufactures over 80% of the routing equipment that controls the flows of information traveling on the Internet and private internetworks.

With this market dominance comes a huge demand for engineers and administrators that understand Cisco's routers and other products. One way that you can demonstrate understanding of Cisco routers and internetworking fundamentals is to pass the Cisco Certified Network Associate (CCNA(tm)) exam (is not associated with Cisco). In July 2000 Cisco retired the CCNA 1.0 exam (640-407) and in August began offering the CCNA 2.0 exam (640-507). The new exam has a new list of objectives that are broader in scope than previous objectives and in fact are no longer really objectives but recommended topic areas for study. You can find these new topic areas here: http://www.cisco.com/warp/public/10/wwtraining/certprog/testing/pdf/ccna_507.pdf. (is not associated with Cisco.)

The purpose of this Tutorial is to introduce you to Cisco router operation. This paper will also help you study for several topic areas that may be tested in the CCNA 2.0 exam including IOS basics, Router CLI, and troubleshooting.

This paper is divided into four sections: Overview of Cisco Router Hardware and Software, Basic Router IOS, Basic Router Configuration, and Basic Router Maintenance and Troubleshooting. There are also Appendices that include review questions (with answers) and additional material that you may find useful.

Overview of Cisco Router Hardware and Software

Hardware and Software Components of the Cisco 2501 Router

You could say that a router is nothing more than a small PC with a smaller operating system and no direct user interface hardware, such as keyboards or video monitors. If you look at what a router does for networking, it is essentially the same as a personal computer. A router looks and acts like a PC in many ways. Like a PC, the router is built with input/output (I/O) ports, it has a processor and memory chips, it provides a set of instructions that tells the router what to do, and it has an operating system that runs the router.

This paper will focus on the components and functions of one of Cisco's entry-level router models, the Cisco 2501. Throughout this text when a router configuration or setup is described we will be speaking of a 2501 unless the statement specifically refers to another router model or number. The 2501 router is a member of the 2500 series family of Cisco routers. It has a single Ethernet interface and two serial interfaces and is powered by a Motorola processor running at 25MHz. The 2501 router is the router of choice for most people getting started with hands-on Cisco router experience; however the Cisco 1600 router, with only one serial port, is adequate and costs a lot less. You can purchase refurbished 2501s for around $900 from various Web auction sites or network hardware resellers that specialize in Cisco equipment. The Cisco 1600 router should cost about $700.

Let's take a closer look at the Cisco 2501 starting at the back of the router (see Figure 1). On the far left is one 10 megabit Ethernet AUI connector used for LAN connections. You will need an Ethernet transceiver/adapter attached to this port so that you can change this port from AUI to RJ45 and make it Category 5 compliant for a typical 10baseT or 100baseT network cable. To the right of the AUI connector are two high-density 60-pin serial connectors. These serial connectors are used for WAN connections. Next to the serial ports are interfaces marked CONSOLE and AUX. Both of these interfaces look like telephone wall jacks that you plug your phone into. We'll discuss the purpose of these ports soon. To the right of the AUX port is small green LED. If this light does not come on when the router is turned on, your router may have a bad memory card or processor. On the far right is the connector where the power cord plugs in.

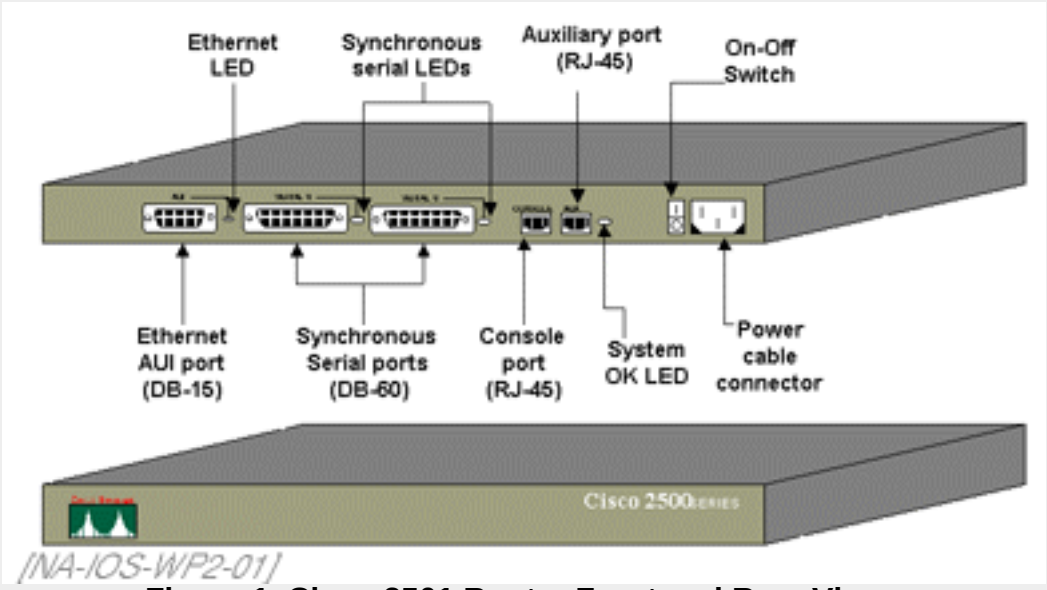


Figure 1. Cisco 2501 Router Front and Rear Views

As you can see, the 2501 router has three types of interfaces. Interfaces are where the router meets the outside world and are the means by which a router receives or sends information. Serial interfaces are mostly used to connect long-distance as in a WAN (Wide-Area Network). Later in this paper we will describe how to connect routers together using their serial ports via DTE/DCE cables to simulate various WAN connections. Besides Ethernet, you can have other LAN interfaces on a router, such as Token Ring or FDDI (Fiber Distributed Data Interface) interfaces. For example, the Cisco router 2502 has two serial interfaces and one Token Ring interface instead of an Ethernet interface. See Appendix B for descriptions of various Cisco routers and the type of interfaces they have.

There are two kinds of interfaces on a router, fixed interfaces and modular interfaces. The three interfaces we just described are fixed interfaces -- they are connected directly to the motherboard and can't be removed. Modular interfaces can be dynamically added or removed by plugging add-in modules or cards into the modular router bus interface. In Cisco routers, the type of bus depends on the model (or family) of router. In a 3600 router, the bus is a PCI bus; in a 7200, it is dual independent PCI buses; in the 7500 series, it is a CyBus. For more information about these types of bus architectures, you can consult the Cisco web site (is not associated with Cisco.)

A name and a number denote each of the interfaces on a Cisco router. The first of any of the interface types starts numbering at zero instead of one. For example, the 2501 router has two serial interfaces and one Ethernet, so we would have **interface serial 0**, **interface serial 1**, and **interface ethernet 0**.

The interfaces discussed show how on a Cisco 2500 series router each interface is designated by interface type and number, such as ethernet0. On newer, modular routers and switches you will run into other interface notations. If the device is modular, it will have slots, cards, and ports on those cards, and you will run into interface notations such as ethernet 1/0 or serial 2/1/1. Remember that all slots and interfaces start at 0 and count up from there. In the case of ethernet 1/0, this represents the first port on the second slot interface card. This two-level representation is seen on all 2600 and 3600 routers, as well as on Catalyst 5000 series switches.

Cisco 7200, 7500 and 12000 routers can have cards called Versatile Interface Processor that can accommodate multiple port adapters on a single slot. Where you have multiple port adapters with multiple ports in a single slot, you will see a three-level notation. Where you see a three-level notation, such as serial 2/1/0, this represents the first port on the second card in the third slot. Each level simply means a more granular description of which port you are looking at. As you work on more routers, you will encounter this more frequently, and it will become more apparent. As the 2500 series routers become more frequently replaced by the 2600 series routers, this will become commonplace notation.

RAM, ROM, NVRAM, and Flash in Cisco Routers

In order to understand what a router does, it helps to know where the basic components are found on the motherboard. In the following paragraphs we will describe the physical characteristics and the position of each component and give a brief explanation of each component's function.

There are three main parts of a router: central processing unit (CPU), memory, and interfaces. The CPU is basically the same as that of a PC; it controls the execution of commands and instructions and directs the flow of information inside and out of the router. The memory comprises four different memory elements: Random Access Memory (RAM), Read-Only Memory (ROM), Non-Volatile RAM (NVRAM), and Flash memory.

Notice the position of the several memory cards and chips in Figure 2. On the left, you can have up to two Flash memory cards. The Flash memory is where the IOS images are located. Cisco routers are almost completely useless without the Cisco IOS, also known as the system software image. Flash memory is a type of erasable, programmable, read-only memory and is available on most Cisco routers as a Flash memory chip, and on some models as a PCMCIA Flash card.

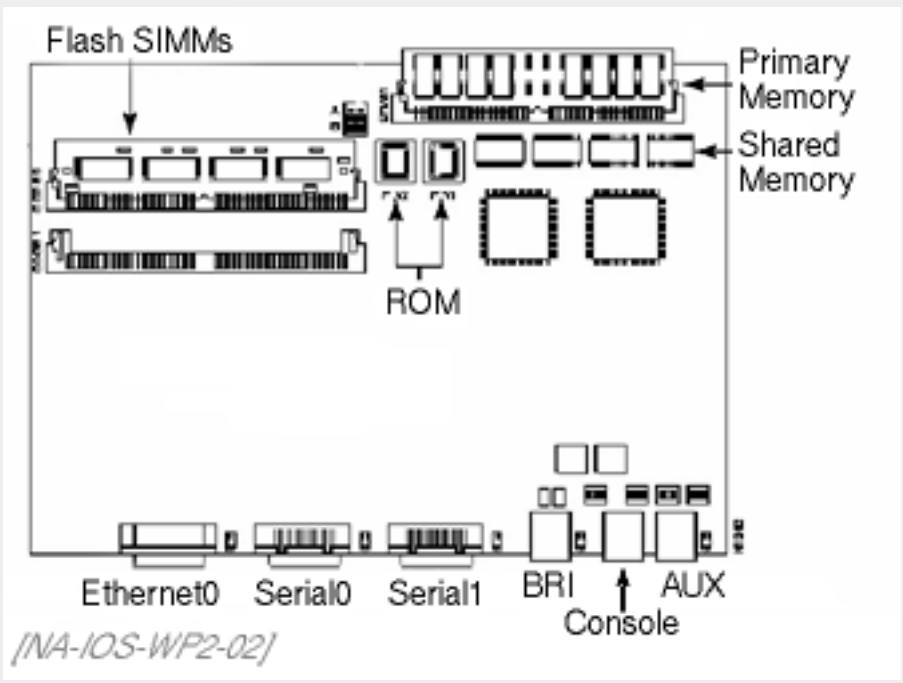


Figure 2. Cisco 2503 Motherboard

Situated at the top-right is the Primary memory, which is a DRAM SIMM memory card, and right below this card are soldered chips called shared memory. There are two types of DRAM memory in the Cisco 2500 series routers: primary and shared (packet). Primary memory is used to store the operating configuration, routing tables, caches, and queues. Shared memory is used to store incoming and outgoing packets. If you have an extra 16 or 8 MB SIMM card lying around from an old 486, it should work fine as an upgrade for Primary memory.

The RAM we just talked about is used strictly for running operations and will get erased with every power off or reload. Router RAM is used just like the RAM in your PC. It stores the running configuration and is where all working information is stored. The running configuration is a partner to the startup configuration. After a router boots, the startup configuration is delivered to the running configuration and it's with this configuration that you typically work to make changes. After you're finished with the running configuration, you then save your changes to the more permanent startup configuration. If you could view the RAM after the router starts up, what you would see is the following: a cached subset of operating system commands, a copy of the startup configuration for running access, all route tables and anything else that dictates how the router behaves.

NVRAM is different from RAM. The contents of NVRAM can be changed, modified, or erased at the user's command. NVRAM will not lose its contents when the router is turned off. The startup configuration file is stored in NVRAM or on the network. The startup configuration file is the instruction set the router uses to boot with. NVRAM is not in a socket. It is part of the motherboard, which makes it difficult to upgrade and hard to find.

Both Flash and Primary memory can be upgraded by simply snapping in new cards. The amounts of Flash memory and Primary memory are typical configuration elements that describe a particular 250*x* router when advertised. You may see or hear of a router advertised as having an 8 by 8 configuration. This means that the router has 8 MB of Flash RAM and 8 MB of Primary memory.

The Cisco router ROM is stored on memory chips that are located on the motherboard. Just underneath Primary memory are two ROM chips. Much like in a PC, the ROM stores the most elemental functions a router must perform to begin operation. ROM is a form of permanent memory used by the Router to store the "Power-On Self-Test" that checks the Router on boot up and the "Bootstrap Startup Program" that gets the Router going. In addition, ROM contains a very basic form of the Cisco IOS software, which is used during certain occasions. In order to upgrade ROM you have to remove and replace chips. The two ROM chips easily pop right out.

Router Boot Sequence

It is important to know the router's boot process. In the event of a boot problem, you may need to spot where the problem is occurring in order to correct it. When a router is powered up, there is a predefined sequence of events that must take place for the router to complete the booting process. First is the Power-On Self-Test (POST), where a test routine is run on the CPU, memory, and interface electronics to make sure there are no circuitry problems. The boot sequence steps are listed below:

1. The "Power-On Self-Test" checks the Router Hardware. This includes the CPU (Central Processing Unit), memory, and interfaces.
2. The "Bootstrap Program," which is stored in ROM, runs itself.
3. The "Bootfield" from the configuration register (discussed later) is read to find out the proper Operating System source.
4. The "IOS software image" is loaded into RAM. The IOS software image can be loaded from Flash, TFTP, or ROM.
5. The Startup Configuration File is read from NVRAM or a TFTP server and then loaded into the RAM. The Configuration File is then executed one line at a time and starts the processes to run the router according to that file.
6. If no "Startup Configuration File" is found in NVRAM, the Cisco IOS will offer you the chance to use the "System Configuration Dialog" or commonly called the "Setup Script." This is a set of questions for you to answer to create a basic configuration.

To see how the boot sequence relates to the four types of memory we discussed earlier, refer to Figure 3. Some of the descriptions in the figure may contain terms you're not familiar with yet, but we'll explain those later in the paper. It is important to understand the relationship between memory and boot sequence for troubleshooting purposes, which we'll also discuss later.

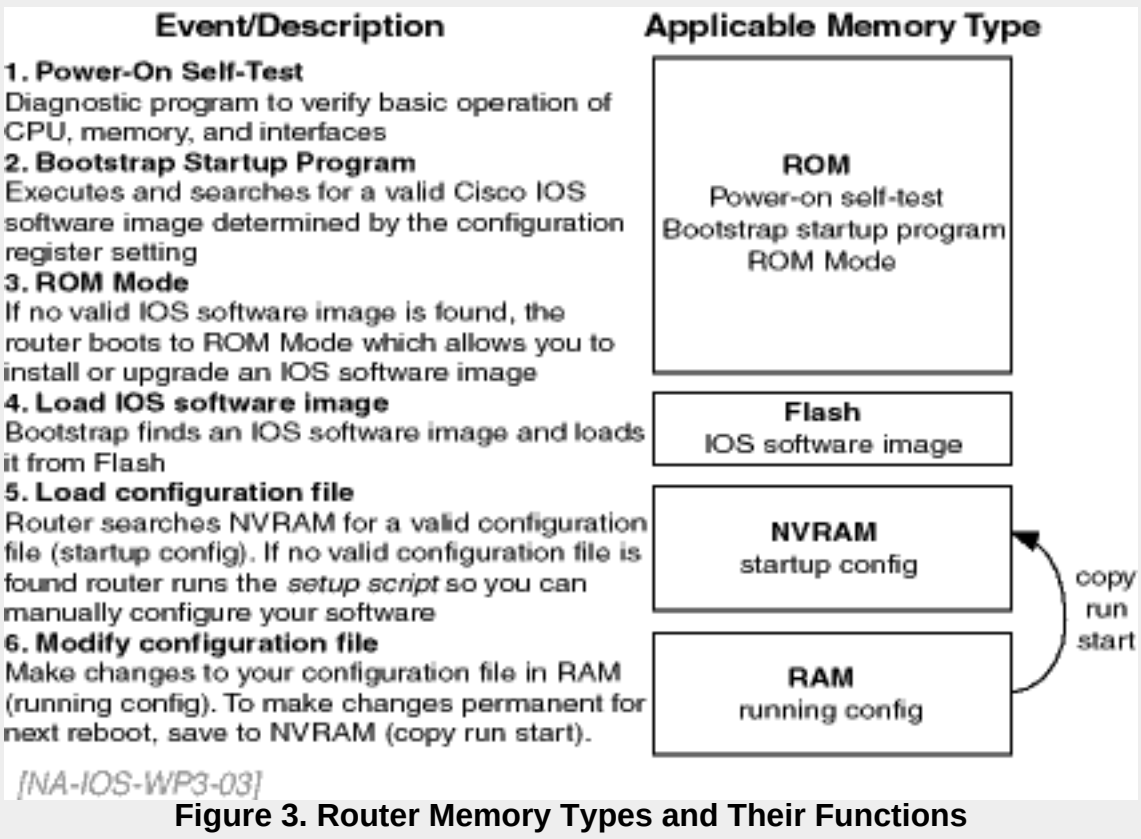


Figure 3. Router Memory Types and Their Functions

IOS Options

Cisco is a hardware vendor company, but if you ask anyone at Cisco, they will likely tell you they are a software company. Their hardware is nothing more than an expensive boat anchor without the IOS software. Knowing this, we can see why knowledge of IOS system software manipulation is crucial to success in working with Cisco routers.

Cisco routers have evolved much over the past 15 years. The IOS software has evolved along with them. In this evolution, Cisco has incorporated different features and functionality with every new release and version. IOS software images are bundled based on feature sets. Each feature set contains support for a certain protocol, group of protocols, or added feature. Some examples include the Desktop feature set that bundles most of the basic LAN networking protocols together, such as IP, IPX, AppleTalk, DECnet, bridging, WAN protocols, etc. Other feature set capabilities you may wish to implement include:

- IP means the router can manage protocols for the Internet.
- IPX means the Novell protocols can be handled.
- AT stands for the AppleTalk protocol for Macintoshes.
- DEC stands for the Digital Equipment Corp. protocols.
- APPN is for the Advanced Peer to Peer Networks (IBM).
- PLUS means NAT (Network Address Translation) can be performed.
- IPSEC is an Internet SECurity feature (encryption) usually not found or needed in typical WANs.
- RAS is an alternate security solution.
- FW means there are firewall capabilities built into the IOS.

Basic Router IOS

Let's now cover how we access the Cisco router and its operating system through the user interface (UI).

Router Access through the Console Port

Since a router doesn't come with a video monitor, you need to use your computer monitor as the router's screen. When you use your monitor like this you call it a terminal. The terminal has an interface called the user interface (UI), which is text command line based instead of a mouse-operated graphic user interface (GUI).

First, let's look at how we can access the UI on a Cisco router. If you receive a Cisco router in the box, there will be a couple of things that accompany the router itself. There will be a black cable called a console cable. If you don't have the black cable, you can use any category 5 cable. Both ends of the black console cable have an RJ45 pin, which looks like a phone plug. Also included is a 9-pin or 25-pin serial adapter to be attached to one end of the black cable that you then connect to the serial port on your computer. The RJ45 end of the cable goes into the Console port on your router.

Now you're ready to deal with the terminal program that will allow your computer to become the router's terminal window. You will need to use some sort of terminal emulation program, such as Hyper Terminal for Windows 95. If you have not used Hyper Terminal, click Start->Programs->Accessories->Hyper Terminal. Once the folder with Hyper Terminal comes up, double click on Hypertrm.exe. As shown in Figure 4, Hyper Terminal will come up and ask you to enter a name for your connection. Type in **direct to com1** and hit Enter.

Figure 4. Name Your Connection

On the next screen, there will be a drop-down list box titled "Connect using." Click on the arrow at the right and choose **Direct to com1** and hit Enter.

Figure 5. Enter Connection Method

The next screen will be the settings for how the terminal emulation program communicates with the console port on the router. The settings should be as shown in the following diagram, bits per second is set to 9600, data bits is 8, parity is none, stop bits is 1, and flow control is hardware. Click OK.

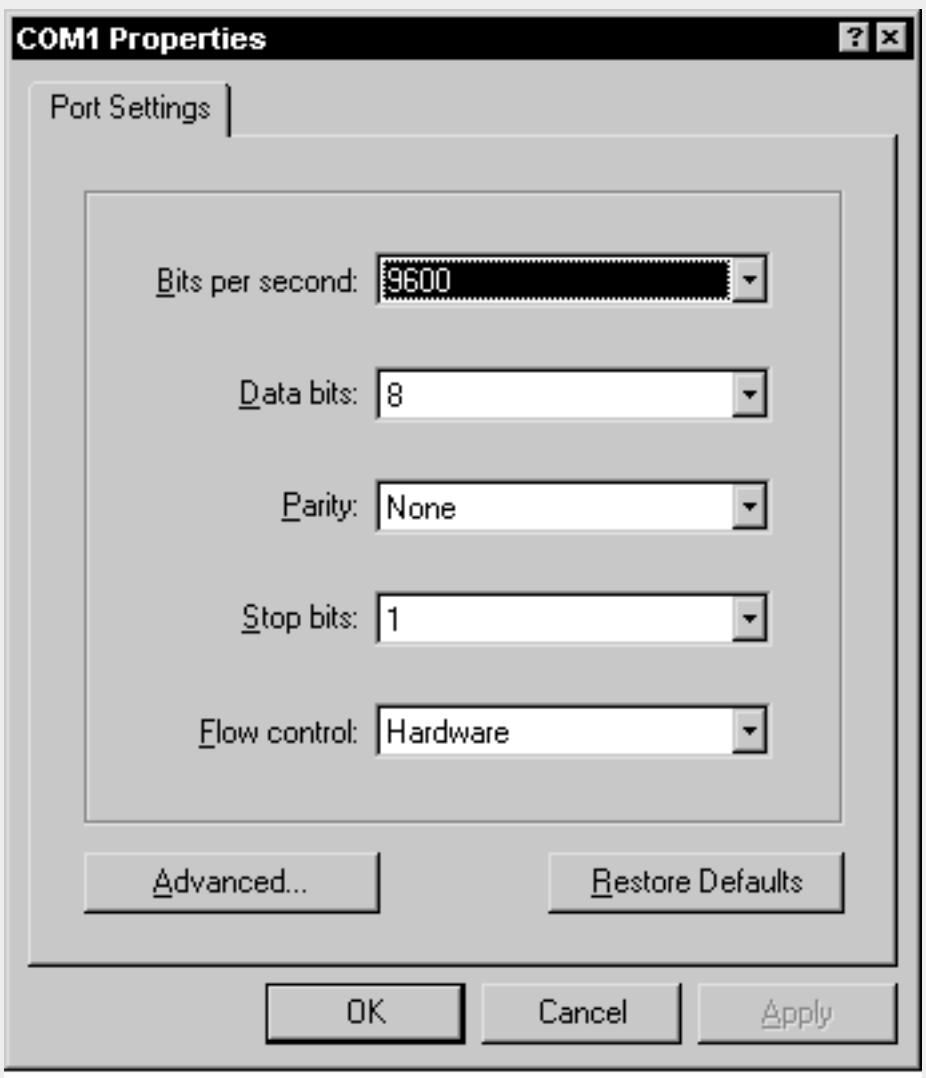


Figure 6. Enter Settings for the Serial Connection

Once you are done with your session, you can close Hyper Terminal. When you do, you will be asked if you want to disconnect. Answer "yes." After this, you will be asked if you would like to save the session. You can go ahead and save this if you'd like, or you can say "no." If you say "no," however, you will have to go through this configuration again every time you go into Hyper Terminal. If you do save the session, when you bring up Hyper Terminal from the start menu, there will be an icon in that folder that will act as a shortcut to get around having to do the configuration steps every time.

Router Access through Telnet

Once a router has been initially configured and has basic network connectivity, the UI can be accessed remotely via Telnet instead of just through the console port. To Telnet from a Windows 95/98 or NT machine, click Start->Run. When the Run box comes up, type in the word "Telnet," and hit Enter. This will run the basic Telnet utility that comes with Microsoft Windows products. Once the Telnet utility program has come up, click **Connect** from the menu, then **Remote System**. Enter the IP address of the remote router to which you wish to Telnet and hit Enter. From there on, the interface will be mostly the same as a console connection. We will cover this again after we show you how to perform initial configuration on a Cisco router. Telnet access to any router can only be accomplished after initial configuration has been performed (from the Console port).

Command Interpreter

Once the router is booted, the Cisco IOS command interpreter, called the Exec, is ready to accept your commands. If you are familiar with DOS, the Exec is much like COMMAND.COM. In the Exec command interpreter, there are two modes of operation: user mode and privileged mode.

User Mode

User mode is denoted by a greater than (>) sign after the router prompt, like this:

```
Router>
```

It allows the user to enter and execute some limited and basic monitoring commands. For example, you can only do things like view basic router information, check status of the router and routing tables, and test simple connectivity. There are no configuration permissions and only limited troubleshooting commands available in user mode.

Privileged Mode

Privileged mode is denoted by a pound (#) sign after the router prompt, like this:

```
Router#
```

In privileged mode, the user has access to all commands available in user mode, all commands necessary to troubleshoot and debug, as well as access to router configuration mode, which is where all router configuration commands are entered. We discuss router configuration mode later in this chapter.

Commands in the Exec are entered via the Command Line Interface (CLI). This is not a new or different access level with its own prompt, rather it's the set of tools associated with the Exec modes.

CLI Help

The CLI has some functions that will give the user a little extra help in entering commands, troubleshooting, and configuring the router.

An editing feature of the CLI is that you can arrow back and forward on a line to edit misspellings. There is one caveat, however. The backspace and delete keys do the same thing in the Cisco CLI.

If you have access to a Cisco router and tried these commands, you probably noticed the **--more--** notation when you entered the **show version** command.

You have three options when you see **--more--**:

- If you press the [space] bar, the command interpreter will display another full screen of information.
- If you press the Enter key, you will get one more line of output.
- If you want to exit before seeing the rest, you can press any other key. The [q] key is most often used.

Inline Help -- Words

Another CLI feature that is a very helpful tool is inline help, also known as the question mark, which provides us with context-sensitive help. Context-sensitive help can be used in two ways, command syntax and word help. Let's try one of the above commands on a Cisco router and see what the ? will do for us:

```
Router# show v?
version  vines  vpdn
Router# show v
```

This is called word help. The output will be a line of data with several possible commands to complete the word that starts with the letter v. As you can see, the outcome was **version vines vpdn**, but what you might not notice is the second **show v** on the next command line. When using context sensitive help, the CLI will automatically repeat the command to where you left off to save you from having to re-enter that text. When using word help, make sure to fill in as many of the letters as you can, then immediately follow that with the question mark (?). Make sure not to leave a space. This inline help can be used in all levels and positions of a command.

Inline Help -- Command Syntax

Help with command syntax can be attained from the question mark as well. If you are configuring the IP address of an Ethernet interface, but are not sure of the syntax, you can use the (?) to help you along:

```
Router(config-if)# ip add ?
  A.B.C.D   IP address
Router(config-if)# ip add 192.168.1.1 ?
  A.B.C.D   IP subnet mask
Router(config-if)# ip add 192.168.1.1 255.255.255.0 ?
  secondary  Make this IP address a secondary address
```

```
  <cr>
```

Here we used the command syntax help to get us all the way through setting the IP address for the ethernet0 interface. You will notice that each step along the way gets us a little closer to where we want to be, and each step gives us a little different information. The last step shown with a question mark shows a couple of responses. The key one to notice here is that the <cr> symbol means that we have fulfilled the necessary command requirements and we can now simply hit the Enter key to execute it. (The <cr> stands for **c**arriage **r**eturn, i.e., the Enter key).

Command Line Completion

Another CLI feature is command line completion, the function of the [tab] key. Let's take the **show version** command and try it again, but this time let's put the [tab] key in the place of the question mark:

```
Router# show v[tab]
Router# show ve[tab]
Router# show version
```

The tab key will fill in with the command that matches the text you enter. If the amount you typed isn't enough, like the first line above, the CLI will make a sound and do nothing but duplicate what you have typed on the next line. This means that the command was too ambiguous, in other words there is more than one command that starts with the typed letter(s).

The Cisco router IOS is actually derived from a Unix operating system kernel. From this origin, the Cisco CLI gained the ability to accept truncated commands. As long as the truncated command you enter is enough of the command to distinguish it from any other command with similar text, the CLI will accept it and the Exec will process it. As an example, let's look at the **show version** command from earlier. To accomplish the same thing, you could also type in **sh ve**. This is an easy way to get what you want done while conserving the maximum number of keystrokes, and as every Unix-head knows, that is the key to slowing the rapid expansion of the universe. In the text of this Tutorial, you will see terse versions of each command, but if you would like to see the full command, get some time in front of a router and hit the [tab] key a lot.

If you truncate the command you are typing too much, and there is a command that has the same beginning, you will see the following error:

```
Router# con
% Ambiguous command: "con"
Router# con?
configure connect
Router# con
```

As you can see, there are two commands that begin with "con," and you must specify which one you wish to use. In this case you can type "con?" and get the two options available to you.

Syntax Checking

Automatic syntax checking is built into the CLI. If a command is improperly spelled, or is not a valid command, the router will respond by placing a caret symbol below the errant letter, word, or argument. If you were to type in **show versoin** like this example, here is what you would receive in response:

```
Router# show versoin
                  ^
% Invalid input detected at '^' marker.
```

Hot Keys Used for Editing

Hot keys are built into the CLI editor to help with simple editing functionality. If you are familiar with Unix, you will recognize quite a few of these:

Table 1. IOS CLI Hot Keys

Hot Key	Function
Delete	Removes one character from the right of the cursor.
Backspace	Removes one character from the left of the cursor.
Tab	Fills in remaining text of a partial command.
Ctrl-a	Moves to beginning of current line.
Ctrl-r	Redisplays a line.
Ctrl-u	Erases all characters on current line from cursor left.
Ctrl-w	Erases word (all characters left of cursor up to next space character).
Ctrl-z	Ends config mode (same as **end** command in config mode).
Up arrow	Scrolls through previously entered commands.

Down arrow	Scrolls forward through former commands (after using Up arrow).

Advanced Editing

If the end of a line goes too long, it will not automatically wrap to the next one. Instead the Cisco IOS command shell gives you a dollar sign ($) at the beginning or end of the line. This indicates that you are an over-achiever and have typed too much, at least too much to be shown on the screen. If you type in a very long command, your line would now look like this:

```
Router#$ this is a way too long line that is full
```

Note that the $ goes after the Router Prompt. If you keep typing the line will shift over as you type, hiding more of the beginning of the sentence.

```
Router#$ is full of sound and fury, signifying nothing!
```

You can get back to the beginning of your novel by pressing [CTRL-A], and this would be the effect:

```
Router# For Demo Only this is a long line that is full $
```

If you want to you can turn off these Advance Editing Tools by simply typing in **terminal no editing** at the prompt. A reason to turn off the Advanced Editing is that the tools are often incompatible with computer-executed scripts. Since this would be a silly thing to do if you are typing things in yourself, please turn them back on by typing in **terminal editing**.

Command History

The CLI keeps a history of the most recent commands entered, accessible by pressing the up arrow. For an example, let's say that a user entered the following three commands:

```
show version
show clock
show user
```

If that user decided that she wanted to show the version once again, she could press the up arrow key three times, and that command would show up on the CLI. Now just press the Enter key.

The Router keeps the last 10 commands you issued in its HISTORY, which is a special memory buffer that holds the "Command History." If you are using the VT-100 Emulator we talked about before, simply do the following.

- Press the UP Arrow key to go to the next most recent command.
- Press the DOWN Arrow key to go back down through the previous commands (after pressing UP arrow).

If you are a poor unfortunate without VT-100, you can use these instead:

- CTRL-P takes you to the "Previous" command.
- CTRL-N takes you to the "Next" commands.

Typing the command **show history** at the prompt gives you the list of the last 10 commands you have entered.

```
Router# show history

1. Command One
2. Command Two
3. Command Three
4. Command Four
5. Command Five
6. Command Sixx - (with a mistake!)
7. Command Six - (fixed now)
8. Command Eight - "There is No Command 7!"
9. Command Nine
10. Command Ten
```

You can increase the size of your History buffer by using the command **terminal history size**. The command below would give you 99 commands to play with.

```
Router# terminal history size 99
```

Basic Router Configuration

There are two ways to configure a router, manually and through the Setup script. If you have a router that has never been turned on or has recently

had the configuration file erased, the router will launch the Setup script, also known as the System Configuration Dialog (much easier to say "Setup script"). Manual configuration is quicker and more flexible, but the Setup script is easier for beginners because it steps you through the whole configuration process. Let's walk through a Setup script as if we have just booted a new router that has never been configured before.

Setup Mode

Setup mode is intended only for minimal configuration of an out-of-the-box newly arrived router. Do not fall into the trap of using it routinely. Almost any real-world configuration will require configuration features that are not available in setup.

The first part of the data capture here is from the system boot process:

```
System Bootstrap, Version 11.0(10c), SOFTWARE
Copyright (c) 1986-1996 by cisco Systems
2500 processor with 14336 Kbytes of main memory

Notice: NVRAM invalid, possibly due to write erase.

F3: 8022188+98780+316356 at 0x3000060

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706

Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L), Version 11.2(18),
 RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
Image text-base: 0x03040270, data-base: 0x00001000
cisco 2500 (68030) processor (revision N) with
 14336K/2048K bytes of memory.
Processor board ID 06160684, with hardware revision 00000000
Bridging software.
SuperLAT software copyright 1990 by Meridian Technology Corp).
X.25 software, Version 2.0, NET2, BFE and GOSIP compliant.
TN3270 Emulation software.
1 Ethernet/IEEE 802.3 interface(s)
2 Serial network interface(s)
32K bytes of non-volatile configuration memory.
16384K bytes of processor board System Flash (Read ONLY)
```

Note that we have been given a notice that NVRAM is invalid. This is not a problem since this router has never been configured. All it means is that NVRAM is empty, and there is no configuration to run from. That being the case, the Cisco router defaults to running the Setup script. As the system configuration dialog prompts you, there will always be a default answer to the question being asked in [brackets]. If this is the answer you want, you can just hit the Enter key.

Let's continue:

```
Notice: NVRAM invalid, possibly due to write erase.
         --- System Configuration Dialog ---

At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[]'.
Would you like to enter the initial
  configuration dialog? [yes]: y

First, would you like to see the current
  interface summary? [yes]: y

Any interface listed with OK? value "NO" does
  not have a valid configuration

Interface  IP-Address OK? Method Status Protocol
Ethernet0  unassigned NO  unset  up     up
Serial0    unassigned NO  unset  down   down
Serial1    unassigned NO  unset  down   down
```

```
Configuring global parameters:

  Enter host name [Router]: Router

The enable secret is a one-way cryptographic secret used
instead of the enable password when it exists.

  Enter enable secret: cisco

The enable password is used when there is no enable secret
and when using older software and some boot images.

  Enter enable password: cisco2
  Enter virtual terminal password: cisco
  Configure SNMP Network Management? [yes]: n
  Configure LAT? [no]: n
  Configure AppleTalk? [no]: n
  Configure DECnet? [no]: n
  Configure IP? [yes]: y
  Configure IGRP routing? [yes]: n
  Configure RIP routing? [no]: y
  Configure CLNS? [no]: n
  Configure IPX? [no]: n
  Configure Vines? [no]: n
  Configure XNS? [no]: n
  Configure Apollo? [no]: n
  Configure bridging? [no]: n
```

Let's summarize what the Setup script has asked and told us so far.

The first thing it asked is if we want to enter the Setup script.

Then the script asked us if we'd like to see an interface summary. This is not a necessary evil on a 2500, but could be useful with a 3600 or higher router that has a flexible configuration.

Now the Setup script asked us for the router name. This will be the name of the router that you see at the IOS prompt. Do not confuse this with a DNS name -- this router name will be locally significant only.

Next it asked us for the enable secret and the enable password. The reason the Setup script does this is stated in its block of text above, that some older software images cannot understand the encryption used on enable-secret passwords. This problem is prevalent in cases where people or businesses are using older routers that have older IOS images in the boot ROMs of their routers.

Next we will be prompted for the virtual terminal password, which is long hand for "Telnet password."

Configure SNMP, LAT, AppleTalk, DECnet, CLNS, IPX, XNS, Apollo, bridging are all other routable protocols that can be configured through the Setup script. For this example, we are only going to set up IP with RIP routing protocol.

Now we will move on to configuring the physical interfaces on the router:

Configuring interface parameters:

```
Configuring interface Ethernet0:
  Is this interface in use? [yes]: y
  Configure IP on this interface? [yes]: y
    IP address for this interface: 192.168.1.1
    Number of bits in subnet field [0]: 0
    Class C network is 192.168.1.0, 0 subnet bits;
      mask is /24
```

There is a statement here in the Setup script that sometimes causes confusion. The question is the number of bits in the subnet **field**. This is not the same as the subnet mask! As shown above, the IP address for the interface is an address in a Class C network (192.168.1.0). The Setup script asks us for the number of bits in the **subnet field**. This means anything beyond the normal Class C mask of 24 bits or 255.255.255.0. If we were to decide that we wanted to subnet this network with a subnet mask of 255.255.255.192 (equivalent to 26 bits), we would respond to that question with the number 2.

For another example of this, we could say that we entered an IP address of 10.1.1.1, and we wanted it subnetted with the same size network that a Class C network has. This mask would be 24 bits, or 255.255.255.0. Since 10.1.1.1 is an address in a Class A network, the Setup script would consider the network mask to be 8 bits or 255.0.0.0. If we wanted the 24-bit mask, we would respond to this subnet field question with the answer 16. 8+16=24, and that gives us the 255.255.255.0 subnet mask. Subnet masking is covered in more detail in the CCNA Tutorial on IP Addressing. Now, back to the configuration:

```
Configuring interface Serial0:
  Is this interface in use? [no]: y
  Configure IP on this interface? [no]: y
  Configure IP unnumbered on this interface? [no]: n
```

```
    IP address for this interface: 192.168.2.1
    Number of bits in subnet field [0]:
    Class C network is 192.168.2.0, 0 subnet bits;
      mask is /24

Configuring interface Serial1:
  Is this interface in use? [yes]: n

The following configuration command script
  was created:

hostname Router
enable secret 5 $1$gTpr$wimCV1ieyQAMEP/vfkEeF1
enable password cisco2
line vty 0 4
password cisco
no snmp-server
!
no appletalk routing
no decnet routing
ip routing
no clns routing
no ipx routing
no vines routing
no xns routing
no apollo routing
no bridge 1
!
interface Ethernet0
ip address 192.168.1.1 255.255.255.0
no mop enabled
!
interface Serial0
ip address 192.168.2.1 255.255.255.0
no mop enabled
!
interface Serial1
shutdown
no ip address
!
router rip
network 192.168.1.0
network 192.168.2.0
!
end

Use this configuration? [yes/no]: y
Building configuration...
Use the enabled mode 'configure' command
to modify this configuration.
```

Now we are done with the Setup script, and the router has entered the configuration commands to setup the router the way we specified. All that is left now is to reboot the router, and we are done. After you answer "yes" to the "Use this configuration?" question, a bunch of stuff will happen. This is okay. What is happening is that the router is doing a quick reset and implementing the configuration you just entered.

```
Press RETURN to get started!

%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
%LINK-3-UPDOWN: Interface Serial0, changed state to down
%LINK-3-UPDOWN: Interface Serial1, changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
                     changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0,
                     changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1,
                     changed state to down
%LINK-5-CHANGED: Interface Serial0, changed state to down
%LINK-5-CHANGED: Interface Serial1, changed state to
                 administratively down
%SYS-5-RESTART: System restarted --
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L), Version 11.2(18),
 RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
```

At this point, the router will appear to be hung up, but don't worry, it's just waiting for you to push it along. Press the Enter Key here and you will see the router prompt show up. If you followed the configuration above, notice the router prompt when you do hit the Enter key, it should look like this:

```
<cr>
Router>
```

Manual Router Configuration

Right after running the Setup script, or whenever you need to change your router's configuration, the best way to get the most out of a Cisco router configuration is to do it manually via the CLI and config mode.

Let's pretend that we didn't go through the Setup script as shown in the last section. Through our terminal emulation program, we will see the router boot up just like before. When we get the prompt asking if we'd like to enter the initial configuration dialog, we can simply type "n" as shown below and then press the Enter Key. The router will ask us if we want to terminate autoinstall. Our answer to this will be "yes." If we were to say "no," the router would spend a lot of time looking for configuration files from network servers. (We'll discuss this more in a later section.) As shown in the Setup script instructions, another way to get out of the Setup script is to just press [Ctrl-c] at any point in the script.

```
Notice: NVRAM invalid, possibly due to write erase.
         --- System Configuration Dialog ---

At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[ ]'.
Would you like to enter the initial
  configuration dialog? [yes]: n
Would you like to terminate autoinstall? [yes]: y
Press RETURN to get started!

%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
%LINK-3-UPDOWN: Interface Serial0, changed state to down
%LINK-3-UPDOWN: Interface Serial1, changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
                     changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0,
                     changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1,
                     changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
                     changed state to down
%LINK-5-CHANGED: Interface Ethernet0, changed state to
                 administratively down
%LINK-5-CHANGED: Interface Serial0, changed state to
                 administratively down
%LINK-5-CHANGED: Interface Serial1, changed state to
                 administratively down
%SYS-5-RESTART: System restarted --
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L), Version 11.2(18),
 RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
```

As you can see here, a couple things happen when you exit from the Setup script. Nothing that happens here is very complicated or problematic. This is a simple system reset that occurs as the router tries to boot. You'll notice above that the router tells you to Press RETURN to get started! This means that the router will wait for your input to present you with a prompt.

Once you exit the Setup script, you can manually configure the router. In order to configure the router via the CLI, you must do three things: first login to user mode, then login to privileged mode, and then enter configuration mode.

After exiting the Setup script (and pressing Enter), you will be presented the **Router>** prompt. This is the prompt of the user exec mode and it's like the DOS prompt on a PC. As described in an earlier section, user exec mode has limited functionality and no configuration access. From this prompt, you must enter the command **enable**, or in CLI shorthand, **en**. Once you have done this, you will be presented the **Router#** prompt (privileged exec mode prompt). Since we are looking at this as if there were no previous configuration on the router, we shouldn't be prompted for a password because currently there are no passwords associated with logging in or entering enable mode. We will cover that after we finish the initial configuration section.

Now that we have entered privileged mode, we can proceed to configuration mode, also known as config mode. From the Router# prompt, type in the command **configure terminal**, in shorthand, **conf t**, and press the Enter key. In config mode, you will see the Router(config)# prompt. From this config prompt, we can enter any configuration commands to setup the router to fit our needs. If you remember the Setup script that we went through, we will show the exact same configuration, but step by step as we go through the initial manual configuration.

In the previous configuration, we configured the router to do the following:

- Set router name
- Set passwords
- Configured interfaces and IP addressing (for Ethernet 0 and Serial 0 interfaces)

- Configured RIP routing for IP
- Committed the configuration to memory (NVRAM)

Now, we will do this via the CLI and router config mode. To begin, once again, we will login to the router, enter enable mode and then begin config mode. The following sequence will get us to this point:

```
<cr>
Router> en
Router# conf t
Router(config)#
```

Now that we are in config mode, we will run through the configuration items listed above. In each of these command sequences, you will see an abbreviated version of the command. If you'd like to see the full text version of the commands, remember that you can hit the [tab] key to fill in the rest of the command. In order to rename the host, you need to use the command host followed by the name you want for the router. In this case we picked "Router". To set the router name we follow this sequence:

```
Router(config)# host Router
Router(config)#
```

Setting Router Passwords

Security is important on a router. Remember that you can access a router from connections other than the console. There are five separate passwords you can set to protect your router:

Console: protects the Console Port

Enable Password: guards the use of the Enable mode super-user status

Enable Secret: an encrypted secret form of the above (better!)

VTY: protects against unauthorized Telnet port logons

Auxiliary: protects the AUX Port (for your modem)

The Console Password

As we continue with our manual reconfiguration tasks, your very next step should be to set the password for the Console Port. Starting from the Router(config)# prompt you need to put in the following series of commands to create the password.

```
Router(config)# line console 0
Router(config-line)# login
Router(config-line)# password cisco
Router(config-line)# Ctrl-Z
Router#
```

Notice that the Router prompt changes to Router(config-line) when you put in the **line console 0** command. It's important to know that **line** is a major command that puts you into "sub-command" mode. This is similar to another "sub-command" mode used for configuring interfaces Router(config-if). Only in the Router(config-line)# mode can you configure individual "lines." Also note that the Ctrl-Z (also written ^Z) ends your session, and brings you back up to the Router# prompt.

The Enable Password and Enable Secret Password

There are two different passwords that allow access to privileged mode, the enable password and the enable secret password.

The purpose of the enable password is to prevent unauthorized access to the privileged mode and configuration mode of the router. You can set this password from configuration mode like this:

```
Router(config)# enable password cisco2
```

The "enable secret" password is a "one-way cryptographic secret password." In other words, once you put in the plain text password, the Cisco IOS takes the text and encrypts it so that no one, not even you, can ever read it again. This is why it is good advice not to forget your enable secret password. Also, the router doesn't like the enable secret to be the same as the enable (as we'll see soon). You can set this password from configuration mode like this:

```
Router(config)# enable secret cisco
```

While you can configure both of these passwords, you can only use one of these passwords at a time on your router. If you choose the enable secret password (and you should because it uses an encryption algorithm to keep your password secure), the enable password will not be used. Even though you may have set your enable password and see it in your configuration file, if you also set the enable secret password your enable password will be ignored.

The only time you should rely only on the enable password is if you are working with an old version of the Cisco IOS (prior to version 10.3) or if your router has an older boot ROM that doesn't recognize the enable secret command. In other words, if your router is new, use the enable secret password.

If you entered the same password for enable secret as you did for enable password, you would receive the following message:

```
Router(config)#enable secret cisco
The enable secret you have chosen is the same as
your enable password. This is not recommended.
Re-enter the enable secret.
```

The IOS gives you this warning because your enable password is listed in clear text right in your configuration file, which anyone can see from user exec mode. If your enable secret password is the same as your enable password and your enable password is shown in your configuration file, you've essentially given away your enable secret password to anyone who guesses that they may be the same.

The VTY Password

VTY ports are not real ports. In other words, you won't find a port on the back of your router labeled VTY. They are also called "Virtual Ports" and they wait for a remote connection, usually using Telnet, to log in. So the virtual terminal password is essentially the same as a Telnet password. Configuring the VTY password is very similar to configuring the Console. The only difference is that there are five VTY virtual ports, which are named 0, 1, 2, 3, and 4. You can use the shortcut 0 4 (a zero, a space, and 4) to set all five passwords at the same time. In order to set the virtual terminal (vty) password, you must enter the following configuration commands:

```
Router(config)# line vty 0 4
Router(config-line)# pass cisco
Router(config-line)# exit
Router(config)#
```

Notice we see the config-line text after the router prompt. When you see this, the commands you are entering here are specific to a certain line interface. If you are in an interface or line configuration mode, you can get back out to the global config mode by typing in **exit** and pressing Enter instead of using CTRL-Z (^Z). By the way, it is not necessary to exit back to the global config mode every time you are finished configuring one interface. You can travel from one interface to another by giving the next interface command.

The Auxiliary Line Password

You can set the Auxiliary Line Password for external modem connections by entering the following commands:

```
Router# config t
Router(config)# line aux 0
Router(config-line)# login
Router(config-line)# password cisco3
Router(config-line)# Ctrl-Z
Router#
```

And now your Router has a password protecting the AUX port.

Now that you have successfully entered all the passwords your router needs, this is a good time to do a quick practice session. To leave the enable mode you need to type in the word **disable** (the opposite of **enable**, the command that got you into this mode). Remember again that enable mode is formally called "Privileged Exec Mode."

```
Router# disable
```

This will leave you at the User Exec Mode prompt, Router >. Now you are going to leave User Exec Mode by typing **quit** or **exit**:

```
Router> exit (or type quit)
```

You will now see the friendly message:

```
Press ENTER to get started.
```

At this point press the ENTER key. The next thing you will see on the screen will be:

```
User Access Verification Password
(please type in your User Password here)

Router>
```

You quickly recognize the "Router >" as the User Exec Mode prompt. Now type in your Enable Secret Password.

```
Router> cisco2
```

If you typed in your enable Secret Password correctly, you should now be in the Privileged Exec Mode.

```
Router#
```

Congratulations! You have now set up your router, created passwords, and successfully logged back into it. Now don't forget your passwords!

Configuring an IP Address

As we continue with the manual reconfiguration of our router, we will now configure two interfaces: Ethernet0 and Serial0. To configure IP routing and then configure Ethernet0 and Serial0 interfaces with IP addresses we enter the following:

```
Router(config)# ip routing
Router(config)# int e0
Router(config-if)# ip add 192.168.1.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# int s0
Router(config-if)# ip add 192.168.2.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)#
```

You can see that we entered commands for the Ethernet0 and Serial0 interfaces without having to exit back to the global config mode between interface commands.

Notice that we are in the global config mode when we turn on IP routing. However, turning on IP routing is not necessary on any Cisco router, because IP routing is enabled by default.

We enter the interface "subcommand" mode with int e0 (interface Ethernet0). We must specify the subnet mask when entering an IP address. Notice that we are using the full sense of the 24-bit mask with three 255s, where each 255 represents 8 bits. Remember we entered a zero when we ran the Setup script, because the Setup script uses zero to represent subnet mask defaults, and in this case, the default subnet mask for this Class C network is 24 bits.

Notice the use of the command **no shutdown**. Use this command to change the state of the interface to up. Remember that an interface is always "shut down" by default and it is a good idea to enter no shutdown just to be on the safe side. If your Ethernet 0 interface is plugged in correctly and you enter a no shutdown, you will see the following appear on the screen that tells you that your newly configured Ethernet0 interface is working:

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
                     changed state to up
%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
```

Since we enabled IP routing, we must turn on some type of routing protocol for IP route tables to propagate through the network. One such routing protocol is RIP, Routing Information Protocol. To enable RIP for IP, we enter:

```
Router(config)# router rip
Router(config-router)# network 192.168.1.0
Router(config-router)# network 192.168.2.0
Router(config-router)#
```

In the above example, RIP is advertising two networks, 192.168.2.0 and 192.168.1.0, so other connected routers can build routing tables.

Note: Now that we finished reconfiguring the most important parts of the router's configuration, let's learn about other configuration features that are either necessary for router maintenance or act as enhancements for day-to-day operations.

Configuring Banners

An IOS banner is used to give information to users or administrators when they log in to a router via a terminal line. We are going to cover three types of banners:

• MOTD (Message Of The Day)

• Login

• Exec

An MOTD banner is sent to a terminal as soon as the terminal's connection becomes active. A login banner is also sent to a terminal when a terminal becomes operative. The login banner is displayed after an MOTD banner if there is one. An exec banner is displayed to a terminal immediately after a person has successfully logged in. We use the global configuration mode command **banner** to create a banner.

```
Router# banner {exec | login | motd} dc message dc
```

The argument ***dc*** is a delimiting character. The delimiting character can be any character as long as it is not part of the message, and it must be the same at the end as it is at the beginning of the ***message***. The banner command should include one of the arguments exec, login, or motd. All three types of banners can be created by issuing the banner command three times, once for each type.

To create a banner, type the command including the first delimiting character, and press Enter. On the next blank line type the rest of the banner message. Banners can have multiple lines (that's the reason for using a dc). When you have finished with the banner message, just type the delimiting character and press Enter again. In the following three examples we will use the percent sign (%) as the delimiting character. Everything between the percent signs is the banner. When the configuration is displayed, IOS uses its own standard delimiter (^C).

```
Router(config)# banner motd %
Enter TEXT message. End with the character '%'.
This is the motd banner.
Remember to meet in cafeteria for Norman's party.
%
Router(config)# banner login %
Enter TEXT message. End with the character '%'.
This is the login banner.
You have accessed a private system.
Unauthorized access is prohibited.
%
Router(config)# banner exec %
Enter TEXT message. End with the character '%'.
This is the exec banner.
We just added a new IOS to all routers.
%
Router(config)#
```

Committing Configuration Changes to NVRAM

All configurations entered through the configuration mode are done dynamically, and stored in regular RAM (volatile RAM) as the current running configuration. Now that we are done with the configuration, we must exit config mode and save our changes. To exit configuration mode you can either enter the command **end**, or press the [Ctrl-z] key combination.

In order to commit these changes to non-volatile RAM (NVRAM) so that they will be enabled every time the router is restarted, we must "write" this running configuration to memory. Remember that the saved configuration in NVRAM is called the "Startup Configuration." Previous to IOS version 10.3, there was only one command to enter, **write memory**, or **wr mem**. In IOS versions 10.3 and later we can use either the old command **wr mem** or the newer command **copy running-config startup-config**, with the shorthand version being **copy run start**. This newer command is preferred by Cisco and is the one you will see most often in textbooks, courses, and exams.

Here are two ways you can exit configuration mode and commit your changes to NVRAM:

```
Router(config)# end
Router# wr mem
```

or

```
Router(config)# ^Z (just pressed [ctrl-z])
Router# copy run start
```

When you commit the running config to memory on a 2500 series router, you will have a little pause while the router saves the config into NVRAM. When it is done, the router will respond to you with the congenial message

```
Building configuration. . .[OK].
```

To summarize the whole configuration we have just gone through, here are all the commands to accomplish the above without interruptions:

```
<cr>
Router> en
Router# conf t
Router(config)# host Router
Router(config)# ena sec cisco
Router(config)# ena pass cisco2
Router(config)# line vty 0 4
Router(config-line)# pass cisco
Router(config-line)# exit
Router(config)#
Router(config)# ip routing
Router(config)# int e0
Router(config-if)# ip add 192.168.1.1 255.255.255.0
```

Notice the from/to syntax of the copy command. This is a critical concept to understand, and there can be serious repercussions if the command is written incorrectly.

With **copy run start** you are telling the IOS that you want to **copy** *from* the **running-config** *to* the **startup-config**. This will effectively save the configuration you've been changing to the more permanent startup-config in NVRAM.

However, if you were to type **copy start run**, you are telling the IOS that you want to **copy** *from* the **startup-config** *to* the **running-config**. This *restores* your running-config to the way it looked when your router booted (or to the most recently saved startup configuration) erasing any changes you've made to your running-config.

So remember, with the copy command the syntax is always from/to!

Later in the paper we'll discuss other examples of the copy command where the from/to syntax still applies.

```
Router(config-if)# no shut
Router(config-if)# int s0
Router(config-if)# ip add 192.168.2.1 255.255.255.0
Router(config-if)# no shut
Router(config-if)# router rip
Router(config-router)# network 192.168.1.0
Router(config-router)# network 192.168.2.0
Router(config)# ^Z  (just pressed [ctrl-z])
Router# copy run start
Building configuration...
[OK]
```

Now that we are done with creating the configuration and saved it, we can view it by entering the command **show running-config** or **sh run**. Here's what it will look like:

```
Router# sh run
Building configuration...

Current configuration:
!
version 11.2
no service password-encryption
no service udp-small-servers
no service tcp-small-servers
!
hostname Router
!
enable secret 5 $1$r.0I$4MbN8jBZLXq9siy9R1ELR1
enable password cisco2
!
!
interface Ethernet0
 ip address 192.168.1.1 255.255.255.0
!
interface Serial0
 ip address 192.168.2.1 255.255.255.0
 no fair-queue
!
interface Serial1
 no ip address
 shutdown
!
router rip
 network 192.168.1.0
 network 192.168.2.0
!
no ip classless
!
!
line con 0
line aux 0
line vty 0 4
 password cisco
 login
!
end
```

If your configuration looks like this, GOOD JOB!

Configuring Clock Rate

We'll end this chapter on Basic Router Configuration with a brief description of what clock rate is and how to configure it.

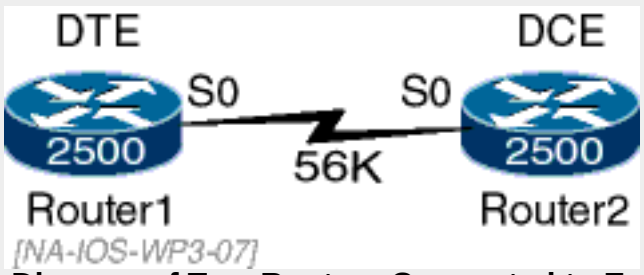


Figure 7. Diagram of Two Routers Connected to Each Other

Clock rate is the command you enter to supply the clock signal that paces the communications on a circuit. For instance, imagine that we wanted to connect two routers together as in Figure 7. In this case, we will connect the routers together with a serial cable (for more information on making connections with serial cables, see "Let's Connect: Your Serial Cable Guide". (is not associated with Cisco.)

If we wanted this link to simulate a 56K circuit, we could enter the following commands in Router2:

```
Router2# config t
Router2(config)# int s0
Router2(config-if)# clock rate 56000
Router2(config-if)# [ctrl-z]
```

Because Router2 is providing the clock for the circuit between itself and Router1, it is known as the Data Communications Equipment (or Data Circuit-terminating Equipment (DCE) depending on how you want to think about it). Router1 accepts this clock rate from Router2, which makes Router1 the Data Terminal Equipment (DTE).

Probably one of the most confusing things about building a network is trying to figure out how everything connects together. While this is primarily a physical cabling issue outside the scope of this paper, it is important to understand that the way you physically connect your equipment can affect the way you need to configure your routers.

Devices that communicate over a serial interface are either DCE or DTE. Some devices are always DCE like modems, CSU/DSUs, and multiplexers. Data terminals are always DTE (thus the name). However, routers, hubs, and switches can be either DCE or DTE.

Believe it or not, the end of the serial cable that's plugged into the router determines whether it is DCE or DTE. Generally, the female end of the serial cable makes the router the DCE. Likewise, the male end of the serial cable makes the router the DTE.

If the router is the DCE, it needs to set the clock rate for the circuit. This shows that physical connections can sometimes impact your router configuration.

For a more in-depth explanation of what serial cables to use in what circumstances, see "Serial Cables."(is not associated with Cisco.)

Basic Router Maintenance and Troubleshooting

Backing up Router Configuration Files

There are several ways to back up the configuration files you've worked so hard to create. The safest and most common way to do these backups is by using a Trivial File Transfer Protocol (TFTP) server. (This is different from the FTP protocol.)

The TFTP server serves as a central configuration repository. This one location can store the router configurations of all the routers in your network. This can be handy in the case of a router failure that causes it to lose its configuration or even if the router itself breaks and you have to get a replacement from Cisco. In these instances, it is possible to retrieve the configurations from a TFTP server and quickly have your router running with your most recently saved configuration.

A TFTP server can run on almost any computer with nearly every operating system. If you are interested in trying out a simple TFTP server for learning purposes, you can get one from the Cisco web site without having to log in. (is not associated with Cisco.) This TFTP server will run on Windows 95/98/NT and is very simple to install and setup. Of course there are other shareware TFTP server programs on the Web that you can try.

In order to save the configuration of your router to the TFTP server, you must first have some sort of network connectivity that will allow you to get to the TFTP server. The computer with the TFTP server running on it needs to be on the same Ethernet segment as your router. The easiest way to accomplish this is to have them both connected to a hub. This means that your router's AUI port will need an Ethernet transceiver with an RJ45 port. With this in place, you can save and retrieve configuration files to and from the TFTP server.

If the TFTP server IP address is 192.168.1.100, the procedure to save the configuration file to the TFTP server is as shown below. We first want to ping the IP address of the TFTP server to ensure that there is connectivity between the router and the server, and then we use the command **copy running-configuration tftp** or **copy run tftp**.

In addition to storing router configurations on a central server, you can also use a core router in your network as a TFTP server. For example, consider a typical hub-and-spoke network configuration that has a 7500 series router at the hub. On this central router you could outfit the router processor card with a decent size PCMCIA Flash card and use this as a storage compartment for all remote site router configuration files. Also, when each router boots each could be configured to retrieve its configuration from that 7500 series router, acting as a TFTP server.

```
Router# ping 192.168.1.100
Type escape sequence to abort.
Sending 5 100-byte ICMP echos to 192.168.1.100,
   timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5),
   round-trip min/avg/max = 4/4/4 ms
Router# copy run tftp
Host or network configuration file [host]? <cr>
Address of remote host [255.255.255.255]? 10.1.1.1
Name of configuration file{}? My-config <cr>
```

Now here's what the commands look like if you wanted to retrieve your configuration from the TFTP server to your router.

```
Router# copy tftp run
Host or network configuration file [host]? <cr>
Address of remote host [255.255.255.255]? 10.1.1.1
Name of configuration file{}? My-config <cr>
Configure using My-config from 10.1.1.1? [confirm] <cr>
Loading My-config....from 10.1.1.1 (via Ethernet0):
[OK - 717/32732]
Router#

Building configuration...
```

This just copies a saved configuration from the TFTP server to your currently running configuration, not your startup configuration. You can either do a **copy run start** to save this configuration to your startup configuration or you can load a saved configuration copy from the TFTP server directly to your startup configuration located in NVRAM. However, in order for this configuration to become your active configuration, you would then have to

copy the startup configuration to the running configuration by reloading your router.

```
Router# copy tftp start
Host or network configuration file [host]? <cr>
Address of remote host [255.255.255.255]? 10.1.1.1
Name of configuration file [My-config]? <cr>
Configure using My-config from 10.1.1.1? [confirm] <cr>
Loading My-config ...
```

> Instead of **reload**, you could use the command **copy start run** without requiring a reset. The problem with **copy start run** is that it actually merges the two configuration files and doesn't make a clean copy. This is not recommended unless you're sure you know what you're doing.
>
> Another alternative to the **reload** command is to use the command **conf mem** (short for **configure memory**), which executes the commands stored in NVRAM.

Now you need to put the saved configuration in startup back into the running configuration by resetting the router. The command to do this is reload, and it's this simple.

```
Router# reload
```

And now you know two methods of backing up your router's configuration files. Why would you want to do this? Well, it is good for resetting the router back to square one if you make a mistake. It is also good for doing a practice Lab a second time.

Which brings us to the ultimate of all configuration commands: **erase startup-config**. This command erases your NVRAM so that the next time you reload, you have a completely blank router. You can use this command to practice the Setup Script covered in an earlier section. Do **not** use this on a production router, as this will bring down your network and likely your job.

> Another way you can back up your router configuration files is to save them to your router's Flash memory. Configuration files are relatively small and there is usually plenty of Flash space on a new router. To see how much Flash space your router has and the names and number of files located there, type **show flash** in the global configuration mode.
>
> You can save your configurations to Flash by typing **copy run flash** or **copy start flash** depending on which configuration files you wish to save. The IOS will prompt you to name the file you are about to save.
>
> While saving your configuration files to Flash is an option if you don't have a TFTP server, a TFTP server is a safer backup method. Since it is located on a device other than your router it is still available in case something catastrophic happens to your router. Also, the files stored on a TFTP server can be included in regular network backups.

Backing Up IOS Software Images

You can also save your router's Flash memory, where the Cisco IOS is stored, to a TFTP server. The following shows the commands to copy Flash to the TFTP server. Notice that you are asked for the IP address of the TFTP server and the name of the file that contains the Cisco IOS on your router. You can find the name of this file by using the **show flash** command in the global configuration mode.

```
Router# copy flash tftp
File Length name/status
11233404  c2500-ajs40-1_113-5_T.bin
[11233468 bytes used, 5543748 available, 16777216 total]
Address or name of remote host [255.255.255.255]? 10.1.1.1
Source file name? c2500-ajs40-1_113-5_T.bin
Destination file name[c2500-ajs40-1_113-5_T.bin]? <cr>
```

You can do it the other way and copy Flash from the server to the router.

```
Router# copy tftp flash
         ****  NOTICE  ****
Flash load helper v1.0
This process will accept the copy
options and then terminate the current
system image to use the ROM based image
for the copy. Routing functionality will
not be available during that time. If
you are logged in via Telnet, this
connection will terminate. Users with
console access can see the results of
the copy operation.
         ---- ******** ----
Proceed? [confirm] <cr>

System Flash directory:
File  Length   Name/status
  1   8121000  c2500-js-l.112-18.bin
[8121064 bytes used, 8656152 available, 16777216 total]
Address or name of
  remote host [255.255.255.255]? 192.168.1.100
Source file name? c2500-js-l.112-18.bin
Destination file name [c2500-js-l.112-18.bin]? <cr>
Accessing file 'c2500-js-l.112-18.bin' on 192.168.1.100...
Loading c2500-js-l.112-18.bin
  from 192.168.1.100 (via Ethernet0): ! [OK]
```

```
Erase Flash device before writing? [confirm] <cr>
Flash contains files. Are you sure you want
  to erase? [confirm] <cr>

Copy 'c2500-js-l.112-18.bin' from server
  as 'c2500-js-l.112-18.bin' into Flash
    WITH erase? [yes/no] y

00:10:59: %SYS-5-RELOAD: Reload requested
%SYS-4-CONFIG_NEWER: Configurations from
  version 11.3 may not be correctly understood.
%FLH: c2500-js-l.112-18.bin
  from 192.168.1.100 to Flash ...

System Flash directory:
File  Length   Name/status
  1   8121000  c2500-js-l.112-18.bin
[8121064 bytes used, 8656152 available, 16777216 total]
Accessing file 'c2500-js-l.112-18.bin' on 192.168.1.100...
Loading c2500-js-l.112-18.bin
  from 192.168.1.100 (via Ethernet0): ! [OK]

Erasing device... eeeeeeeeeeeeeeeeee ...erased
Loading c2500-js-l.112-18.bin
  from 192.168.1.100 (via Ethernet0):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!
(There may be many more or fewer of
  these ! marks on your screen.)

[OK - 8121000/16777216 bytes]

Verifying checksum...  OK (0x5996)
Flash copy took 0:04:10 [hh:mm:ss]
%FLH: Re-booting system after download
(Here the router reboots like normal.)
```

You may have noticed that the router had to reboot in order to copy the image into Flash. This is okay, but we should investigate the reason why this happens.

Because this is a 2500 series router that runs the IOS straight from Flash, copying a file to Flash while the IOS is running from Flash will not work. In order to make software copies and updating as painless as possible, Cisco created the Flash Load Helper. The Flash Load Helper assists in setting the configuration register to run from ROM, reboots the router, copies the image into Flash, changes the configuration register back, then reboots the router again.

The Flash Load Helper was an optional feature in Cisco routers until software version 10.3. If you happen to be working with a router with earlier software, you must manually set the configuration register to boot with the default software in ROM, then perform the image copy, then change the configuration register back and reboot the router. We'll discuss the configuration register later in the paper. Most other router models run the IOS from RAM, so this may not be an issue if you are using another router model.

Show Commands

While making changes to your router's configuration, it is a good idea to frequently monitor your work before going on. The privileged mode is where we can examine our work by using certain verification (show) commands. After performing several manual configuration tasks either in the global configuration mode or other deeper "subcommand" configuration modes, you should slip back into the privileged mode to check things out.

If you're at the Router# prompt, just type **exit**, and that will send you back to the Router> prompt, which is where you need to be to use the verification commands.

We will examine the following verification commands and the information they provide. The **show** command is the portal that allows us to view anything we want on the router. If you'd like to see what show options are available, type **show ?** at the exec prompt.

We will discuss the more common show commands.

It's useful to know older commands from Cisco IOS version 10.3 or lower that are equivalent to those in recent software versions. Table 2 displays equivalent commands between newer and older software version.

Table 2. Equivalent Commands

Higher than Version 10.3	Version 10.3 or lower

Show startup-configuration	Show configuration
Show running-configuration	Write term
Erase startup-configuration	Write erase
Copy running-config startup-config	Write mem

Most show commands can be viewed from the regular User Exec mode. Some show commands can only be viewed from the Privileged Exec (Enable) mode. If you've been busily configuring interfaces and protocols in config mode and forget to change back to the Router# or Router> prompt, using a show command will not work. None of the show commands can be used from config mode. This will just give you an error, and you will feel very silly.

If you type in the command **show**, a space, and then a question mark at the Enable Mode "Router#" prompt, the Help function will give you a long list of show commands.

```
Router# show ?
show access-expression show access-list
show apple interface
show apple route
show appletalk
show atm
show bridge
show cam
show cam dynamic
show cdp neighbors
show config
...
```

...and so on going down through the entire alphabet. Luckily, you do not need to memorize all these right away for the tests. There are, however, two important show commands that allow you to see your router's full configurations and you'll want to remember: **show running-configuration** and **show startup-configuration**.

show startup-config shows you the configuration commands stored in the Router's NVRAM, the place where configurations live when the power is off.

The **show running-config** command shows you the configuration as you have changed it since turning on the router. We used this command earlier in the paper to show the changes we made to the router. This command shows the configuration that is actually running right now on your router, in RAM.

For security reasons, these commands are not available from the user prompt. If you do a **show run** or **show start** from the Router> prompt, you'll get an error message. The reason for this is that the enable password (but not the enable secret password) is shown in clear text by these commands.

Basic Show Commands

You are apt to use most or all of the information in these commands when doing routine troubleshooting.

Show Version

The **show version** command gives you information on the version of the Cisco Internetwork Operating System that your router is using. It also gives you lots of other basic information such as how long the router has been up, how the system was started, what processor your router uses, how much memory your router has, and from where the system image file was loaded. **show version** will also show you what interfaces the router has.

```
router# show version
Cisco Internetwork Operating System Software IOS (tm)
3000 software (IGS-I-L, Version 11.1(11)
  RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by Cisco Systems, Inc.
Compiled Tue 24-Jun-97 12:20 by jaturner
Image text-base: 0x0301E644, data-base 0x00001000

ROM: System Bootstrap, Version 11.0(10c), SOFTWARE ROM:
     3000 Bootstrap Software (IGS-BOOT-R),
     Version 11.0(10c) RELEASE SOFTWARE (fc1)

Router uptime is 12 minutes System restarted by power-on
System image file is "Flash:igs-i-l.110-16",
  booted via Flash

cisco 2500 (68030) processor (revision N) with
  2048K/2048K bytes of memory.
Processor board ID 06267777, with hardware revision 00000000
```

```
Bridging software X.25 software, Version 2.0, NET2,,
  BFE and GOSIP compliant.

1 Ethernet/IEEE 802.3 interface. 2 Serial network interfaces.

32K bytes of non-volatile configuration memory.
8192K bytes of processor board System Flash (Read ONLY)

Configuration register is 0x2102
```

Show Interfaces

The **show interfaces** command is like the Swiss Army knife of troubleshooting. It gives you information on all the interfaces in your router. Since the interfaces are where all the real work takes place, being able to see what they are doing is very helpful. In the example below, we are using a Cisco Router 2503, which includes two additional ISDN ports (BRI0:1 and BRI0:2).

One of the most important ways to check the health of a Cisco router interface is on the first line of this output. In this case, the serial interface is administratively down, and the line protocol is down. This means that the interface is in "shutdown" mode, and whoever is configuring this router wants that interface to remain shut down. If that line were to say that the interface was down instead of administratively down, that means that we have a problem with the physical connectivity on that line. Another basic status we could get on this interface is that the interface is up, but the line protocol is down. A lot of times this means that the OSI model Layer 1 is up but Layer 2 is down. Yet another status is that the line is up and the line protocol is up. This is how it should be, and it means everything is okay on that interface.

```
Router> show interfaces
BRI0 is administratively down, line protocol is down
Hardware is BRI
MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
   Conversations  0/0/256 (active/max active/max total)
   Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
     0 ignored, 0 abort
   0 packets output, 0 bytes, 0 underruns
   0 output errors, 0 collisions, 5 interface resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions

BRI0:1 is administratively down, line protocol is down
Hardware is BRI
MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
Encapsulation HDLC, loopback not set
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
   Conversations  0/0/256 (active/max active/max total)
   Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
      0 ignored, 0 abort
   0 packets output, 0 bytes, 0 underruns
   0 output errors, 0 collisions, 5 interface resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions

BRI0:2 is administratively down, line protocol is down
Hardware is BRI
MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
```

```
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
   Conversations  0/0/256 (active/max active/max total)
   Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
      0 ignored, 0 abort
   0 packets output, 0 bytes, 0 underruns
   0 output errors, 0 collisions, 5 interface resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions

Ethernet0 is administratively down, line protocol is down
Hardware is Lance,
address is 0010.7b3a.dea6 (bia 0010.7b3a.dea6)
MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
   reliability 252/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
ARP type: ARPA, ARP Timeout 04:00:00
Last input never, output 01:17:16, output hang never
Last clearing of "show interface" counters never
Queueing strategy: fifo
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
   0 input packets with dribble condition detected
   14 packets output, 840 bytes, 0 underruns
   14 output errors, 0 collisions, 1 interface resets
   0 babbles, 0 late collision, 0 deferred
   14 lost carrier, 0 no carrier
   0 output buffer failures, 0 output buffers swapped out

Serial0 is administratively down, line protocol is down
Hardware is HD64570
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output 01:17:18, output hang never
Last clearing of "show interface" counters 01:17:18
Queueing strategy: fifo
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
      0 ignored, 0 abort
   5 packets output, 853 bytes, 0 underruns
   0 output errors, 0 collisions, 2 interface resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions
   DCD=down  DSR=down  DTR=down  RTS=down  CTS=down

Serial1 is administratively down, line protocol is down
Hardware is HD64570
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output 01:17:50, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
   Conversations  0/2/256 (active/max active/max total)
   Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
```

```
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
      0 ignored, 0 abort
   6 packets output, 132 bytes, 0 underruns
   0 output errors, 0 collisions, 3 interface resets
   0 output buffer failures, 0 output buffers swapped out
   23 carrier transitions
   DCD=down  DSR=down  DTR=down  RTS=down  CTS=down

Serial2 is administratively down, line protocol is down
Hardware is CD2430 in sync mode
MTU 1500 bytes, BW 115 Kbit, DLY 20000 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
   Conversations  0/0/256 (active/max active/max total)
   Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
      0 ignored, 0 abort
   6 packets output, 1992 bytes, 0 underruns
   0 output errors, 0 collisions, 2 interface resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions
   DCD=down  DSR=down  DTR=down  RTS=down  CTS=down

Serial3 is administratively down, line protocol is down
Hardware is CD2430 in sync mode
MTU 1500 bytes, BW 115 Kbit, DLY 20000 usec,
   reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
   Conversations  0/0/256 (active/max active/max total)
   Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
   0 packets input, 0 bytes, 0 no buffer
   Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
   0 input errors, 0 CRC, 0 frame, 0 overrun,
      0 ignored, 0 abort
   6 packets output, 1992 bytes, 0 underruns
   0 output errors, 0 collisions, 2 interface resets
   0 output buffer failures, 0 output buffers swapped out
   0 carrier transitions
   DCD=down  DSR=down  DTR=down  RTS=down  CTS=down
```

Show Protocols

A protocol is an agreed-upon set of rules for speaking to others. It's sort of like having a conference call and everyone agreeing to speak German. The **show protocols** command lets you know if everyone is speaking properly. If they are not, then the router will tell you, "Line Protocol is down." Even if the interface is up, if the line protocol isn't working, nothing works.

All of our interfaces will be listed as administratively down since we have not yet turned any of them on. In fact, since we are only doing the basic setup of one router in this tutorial, we don't actually have anyone else with whom to speak.

```
Router> show protocols

Global values:
  Internet Protocol routing is enabled
BRI0 is administratively down, line protocol is down
BRI0:1 is administratively down, line protocol is down
BRI0:2 is administratively down, line protocol is down
```

```
Ethernet0 is administratively down, line protocol is down
Serial0 is administratively down, line protocol is down
Serial1 is administratively down, line protocol is down
Serial2 is administratively down, line protocol is down
Serial3 is administratively down, line protocol is down
```

Show Flash

The **show flash** command tells you how many bytes are used and available in Flash memory and what files are stored there.

```
Router> show flash
System Flash directory:
File  Length   Name/status
  1   11780820  12-04T.bin
[11780884 bytes used, 4996332 available, 16777216 total]
16384K bytes of processor board System Flash (Read ONLY)
```

Advanced Show Commands

These commands do give some information that may be useful on a day-to-day basis, but much of the information they show is meaningful only to Cisco technical support. Technical support has access to router code, detailed internal data structures, and other information not provided to customers. This sort of confidential information is needed to understand the full meaning of advanced displays.

Show Memory

The **show memory** command shows what memory is allocated by the management system for which purposes. There are two memory charts that get shown by the command: a Summary and a Detailed Block by Block memory chart.

```
Router> show memory
```

Summary:

```
Head Total(b) Used(b) Free(b) Lowest(b) Largest(b) Processor
EA90C 5326580 2056220 3270360 3270360 3231192
I/O 600000 2097152 465264 1631888 1579032 1631720
```

A Detailed Block-by-Block memory chart:

```
Allocator PC Summary for: Processor

pc=0x031FDE54, size=000963416, count=000056,
  name=List Elements pc=0x031D8060, size=000462508,
  count=000312, name=*Packet Data*
pc=0x03217BAE, size=000287992, count=000068, name=Interrupt Stack
pc=0x031D8028, size=000178496, count=000312, name=*Packet Header*
pc=0x031DCDEC, size=000115040, count=000008, name=Fair Queueing
pc=0x031C2BD2, size=000049196, count=000001, name=Exec
pc=0x031DDBA8, size=000044660, count=000011, name=*Hardware IDB*
pc=0x031957E4, size=000040840, count=000010, name=TTY data
pc=0x03214150, size=000033516, count=000063, name=Process
pc=0x0322E6F4, size=000032808, count=000001, name=Cfg EEPROM Copy
pc=0x031DDBBE, size=000025124, count=000011, name=*Software IDB*
pc=0x034A829A, size=000014468, count=000001, name=Init
pc=0x034A81F4, size=000014464, count=000001, name=Init
pc=0x03AA68C2, size=000013644, count=000001, name=Init
pc=0x03A772B6, size=000013644, count=000028, name=ATMSIG-SHOW
pc=0x031A2D10, size=000013512, count=000197, name=Parser
01:13:41: %SYS-3-CPUHOG: Task ran for 2008 msec (19/19),
  process = Exec, PC = 31 7A068.

-Traceback= 320F2A6 317A070 318F4A4 31904A2 318F54C 31C2EBE
 31C3028 31C3332 31A18F0 31B605C Linkage
pc=0x031368E0, size=000012044, count=000001, name=Init
pc=0x0320BCD8, size=000012032, count=000084, name=Watched Boolean
pc=0x032B17D0, size=000011420, count=000001,
  name=DHCPD Message Workspace
pc=0x0320BEE8, size=000011040, count=000064, name=Process Events
--More-
```

Right now, we are only concerned with a few of the columns in this output. The Total, Used and Free columns give us a sense of how much operating memory the router has, how much is being used right now by the processor and the I/O subsystem, and how much is free to each. The (b) on each of these columns means that this information is expressed in bytes.

If you are interested in seeing how much memory each process running in the router is taking, you may want to look at the rest of the output from the **sh mem** command. We must warn you, this is a lot of information, and gets somewhat boring to look at.

Show Processes

A process is part of a program, or if it is small, it can be the entire program. It's sort of like having a troupe of jugglers: each item they toss up in the

air is one process. As long as they keep them all going, everything is fine. If not, you can use **show processes** to do a little troubleshooting. The show processes command shows you all the active processes in the form of a chart containing the following information in columns:

PID - The ID number of each process.

Q - The Queue priority

TY - This is the status of the process

PC - Program Counter.

Runtime - The amount of CPU time in milliseconds used by the process

Invoked - This is the amount of time the process has been invoked.

uSecs - The CPU time in milliseconds for each process invocation.

Stacks - This shows both the "low water mark" / "total stack space" in bytes.

TTY - Shows you which terminal controls the process.

Process - Finally, this actually gives you the name of the process.

Pay particular attention to the first line, which shows CPU utilization. While there are no hard and fast rules, you generally don't want a router that does dynamic routing to average much more than 50-60% of utilization over 5-minutes.

```
Router> show processes

CPU utilization for five seconds: 7%/7%;
     one minute: 9%; five minutes: 12%

PID QTy      PC Runtime Invoked uSecs    Stacks TTY Process
                (ms)
  1 Csp 32134FE      8     872      9  736/1000   0 Load Meter
  2 M*        0   3632      82  44292 2960/4000   0 Exec
  3 Lst 3203DC6  14300     960  14895 3736/4000   0 Check heaps
  4 Cwe 3209FB6      0       1      0 3724/4000   0 Pool Manager
  5 Mst 318E706      0       2      0 3700/4000   0 Timers
  6 Mwe 311F992      8       2   4000 3696/4000   0 Serial Backgroun
  7 Lwe 323C858    340      78   4358 3684/4000   0 ARP Input
  8 Mwe 33877A6      0       3      0 3704/4000   0 DDR Timers
  9 Mwe 339B8CA      0       2      0 5712/6000   0 Dialer event
 10 Lwe 34BE0AC     36       2  18000 3684/4000   0 Entity MIB API
 11 Mwe 3125CA2      0       1      0 3732/4000   0 SERIAL A'detect
 12 Cwe 320D770      0       1      0 3740/4000   0 Critical Bkgnd
 13 Mwe 31E55AA    696     547   1272 4756/6000   0 Net Background
 14 Lwe 31857B2     16       7   2285 5604/6000   0 Logger
 15 Msp 319E1D4    172    4347     39 3568/4000   0 TTY Background
 16 Msp 31E4EB6   3084    4415    698 3736/4000   0 Per-Second Jobs
 17 Msi 3235488     40    4351      9 3724/4000   0 Partition Check
 18 Hwe 31E5014      0       1      0 3712/4000   0 Net Input
 19 Csp 31EC442     68     873     77 3728/4000   0 Compute load avg
 20 Msp 31E4EE4   4740      75  63200 3776/4000   0 Per-minute Jobs
 21 Mwe 309D71E      0       1      0 3824/4000   0 SYNCCD2430 Helpe
--More--
```

Show Stack

A stack is basically a portion of the memory that is used to monitor the internal operations of a program. These stacks are "Last In, First Out" (LIFO) data structures. The **show stacks** command looks at the manner in which the Cisco router's processes and interrupts utilize these stacks. If there was a reboot caused by a crash, then using **show stacks** may reveal the reason for that reboot.

```
Router> show stacks

Minimum process stacks:
Free/Size   Name
2704/4000   Setup
3256/4000   Autoinstall
2776/4000   DNS Snoop
2680/4000   Init
1720/2000   LAPB Timer
5400/6000   BootP Resolver
3460/4000   RADIUS INITCONFIG
4632/5000   DHCP Client
3524/4000   Exec

Interrupt level stacks:
```

```
Level Called Unused/Size  Name
  1        0   3000/3000  CL-CD2430 transmit interrupts
  2        0   3000/3000  CL-CD2430 receive interrupts
  3       33   2772/3000  Serial interface state
                          change interrupt
  4       23   2872/3000  Network interfaces
  5    10771   2896/3000  Console Uart
```

Show Buffers

A buffer is a portion of memory in which data can rest while it waits to catch the next bus out. Buffers are sort of like bus stops, but some are bigger (like a bus station); and some of them are very large, like an airport! The **show buffers** command lets you see the size of the small, middle, big, very big, large, and huge buffers. It also gives statistics on their usage, kind of like baseball scores.

```
Router> show buffers

Buffer elements:

500 in free list (500 max allowed)
128 hits, 0 misses, 0 created

Public buffer pools: Small buffers, 104 bytes
  (total 56, permanent 50):
54 in free list (20 min, 150 max allowed)
87 hits, 2 misses, 0 trims, 6 created
0 failures (0 no memory)

Middle buffers, 600 bytes (total 28, permanent 25):
28 in free list (10 min, 150 max allowed)
76 hits, 1 misses, 0 trims, 3 created
0 failures (0 no memory)

Big buffers, 1524 bytes (total 50, permanent 50):
47 in free list (5 min, 150 max allowed)
19 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)

VeryBig buffers, 4520 bytes (total 10, permanent 10):
10 in free list (0 min, 100 max allowed)
0 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)

Large buffers, 5024 bytes (total 0, permanent 0):
0 in free list (0 min, 10 max allowed)
0 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)

Huge buffers, 18024 bytes (total 0, permanent 0):
0 in free list (0 min, 4 max allowed)
0 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)

Interface buffer pools: Ethernet0 buffers, 1524 bytes
   (total 32, permanent 32):
8 in free list (0 min, 32 max allowed)
24 hits, 0 fallbacks
8 max cache size, 8 in cache

BRI0 buffers, 1524 bytes (total 4, permanent 4):
3 in free list (0 min, 4 max allowed)
3 hits, 0 fallbacks
1 max cache size, 1 in cache

BRI0:1 buffers, 1524 bytes (total 16, permanent 16):
12 in free list (0 min, 16 max allowed)
12 hits, 0 fallback
4 max cache size, 4 in cache

BRI0:2 buffers, 1524 bytes (total 16, permanent 16):
12 in free list (0 min, 16 max allowed)
12 hits, 0 fallbacks
4 max cache size, 4 in cache

Serial0 buffers, 1524 bytes (total 32, permanent 32):
7 in free list (0 min, 32 max allowed)
25 hits, 0 fallbacks
8 max cache size, 8 in cache

Serial1 buffers, 1524 bytes (total 32, permanent 32):
```

```
7 in free list (0 min, 32 max allowed)
25 hits, 0 fallbacks
8 max cache size, 8 in cache

Serial2 buffers, 1524 bytes (total 8, permanent 8):
6 in free list (0 min, 8 max allowed)
6 hits, 0 fallbacks
0 max cache size, 0 in cache

Serial3 buffers, 1524 bytes (total 8, permanent 8):
6 in free list (0 min, 8 max allowed)
6 hits, 0 fallbacks
0 max cache size, 0 in cache

CD2430 I/O buffers, 1524 bytes (total 20, permanent 20):
10 in free list (0 min, 20 max allowed)
10 hits, 0 fallbacks
```

Show Processes CPU

Much like the memory output, the **show processes cpu** command contains a lot of good information but the most important part is the first line. This line shows some the running utilization of the router's CPU. This router has a 5- minute average utilization of 10%, which is not too bad. If you see a router with a 5-minute utilization of over 60%, you might want to do some serious investigating. If you see a router with a 5-minute utilization of over 95%, you may not be able to get processor time to do any investigating.

```
Router# sh proc cpu

CPU utilization for five seconds: 14%/11%; one minute: 10%;
   five minutes: 10%
PID Runtime Invoked uSecs 5Sec  1Min  5Min TTY Process
     (ms)
  1      92    2297     40 0.00% 0.00% 0.00%  0 Load Meter
  2    1128     163   6920 1.39% 0.31% 0.27%  0 Exec
  3   17676     384  46031 1.80% 0.22% 0.14%  0 Check heaps
  4       0       1      0 0.00% 0.00% 0.00%  0 Pool Manager
  5       4       2   2000 0.00% 0.00% 0.00%  0 Timers
  6       4     194     20 0.00% 0.00% 0.00%  0 ARP Input
  7       0       1      0 0.00% 0.00% 0.00%  0 SERIAL A'detect
  8       0     192      0 0.00% 0.00% 0.00%  0 IP Input
  9      20    1152     17 0.00% 0.00% 0.00%  0 CDP Protocol
 10       4      22    181 0.00% 0.00% 0.00%  0 MOP Protocols
 11     256   11693     21 0.00% 0.00% 0.00%  0 IP Background
 12      40    2301     17 0.00% 0.00% 0.00%  0 TCP Timer
 13       0       1      0 0.00% 0.00% 0.00%  0 TCP Protocols
 14       4       1   4000 0.00% 0.00% 0.00%  0 Probe Input
 15       0       1      0 0.00% 0.00% 0.00%  0 RARP Input
 16       4       1   4000 0.00% 0.00% 0.00%  0 BOOTP Server
 17      20     192    104 0.00% 0.00% 0.00%  0 IP Cache Ager
 18       0     192      0 0.00% 0.00% 0.00%  0 NBF Input
 19       0       2      0 0.00% 0.00% 0.00%  0 SPX Input
 20       0       2      0 0.00% 0.00% 0.00%  0 DDR Timers
 21       0       1      0 0.00% 0.00% 0.00%  0 SNMPConfCopyProc
 --More-
```

Show CDP Neighbors

The **show cdp neighbors** command shows you all the Cisco equipment to which your router has a direct physical connection. It is used once you have connected your router to other Cisco routers on your network. It uses the Cisco Discovery Protocol, which is proprietary to Cisco and is why it only finds Cisco devices. The command is very useful for troubleshooting networks. If **show cdp neighbors** doesn't show a connection, basically you aren't connected (to a Cisco device).

CDP is a layer 2 protocol. It is only used for informational updates on directly connected links. What this means to us is that when we look at CDP information on a Cisco router, it will tell us a lot of information about the other Cisco equipment connected directly to the same networks as the router at which we are looking. Since this is a layer 2 protocol, all communications for CDP are broadcast-based, and these broadcasts are sent out every 60 seconds by default. Let's take a look at what we can see from our test network environment.

The **sh cdp ?** command shows us what command arguments are available:

```
Router# sh cdp ?
  entry      Information for specific neighbor entry
  interface  CDP interface status and configuration
  neighbors  CDP neighbor entries
  traffic    CDP statistics
  <cr>
```

The real value of the **sh cdp** commands is being able to see the status of the network, the devices to which you are connected, and what their capabilities are. Each CDP status will show us three basic things: accessibility, capabilities, and device type. A simple example on the router from our example shows the following:

```
Router# sh cdp nei
```

```
Capability Codes: R - Router, T - Trans Bridge,
                  B - Source Route Bridge,
                  S - Switch, H - Host, I - IGMP,
                  r - Repeater
Device ID  Local  Holdtme Capability Platform Port ID
           Intrfce
Router2      Ser 0    144  R         2500     Ser 1
066538247    Eth 0    159  T B S     WS-C5505 4/8
Router#
```

What this shows us is basic connection information about to what a router is locally connected and with what it is communicating. At the moment of this capture, Router1 is connected to a 2500 router named Router2, as well as being connected to a WS-C5505, which is a Catalyst 5505 switch. As you can see, you can get connectivity information on all Cisco products that have CDP enabled. As of IOS version 10.3 and later, CDP is enabled by default on all interfaces.

To look at an individual entry in the CDP neighbor table, use the command where name is the router name or other Cisco device found in the neighbor table. The name is case sensitive. This command is useful if the neighbor table is large and you want to get detailed information about a single neighbor without having to view the details of all neighbors.

```
Router# show cdp entry Router1

Device ID: Router1
Entry address(es):
  IP address: 192.168.2.2
Platform: cisco 2500,  Capabilities: Router
Interface: Serial0,  Port ID (outgoing port): Serial1
Holdtime : 147 sec

Version :
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L),
Version 11.2(18), RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
```

The **show cdp neighbors detail** command gives the same output as **show cdp entry**, except it shows detail for the connected routers and other devices.

```
Router# sh cdp nei det

Device ID: Router1
Entry address(es):
  IP address: 192.168.2.2
Platform: cisco 2500,  Capabilities: Router
Interface: Serial0,  Port ID (outgoing port): Serial1
Holdtime : 147 sec

Version :
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L),
Version 11.2(18), RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner

Device ID: 066538247(GCD5505)
Entry address(es):
  IP address: 10.200.1.5
Platform: WS-C5505,
  Capabilities: Trans-Bridge Source-Route-Bridge Switch
Interface: Ethernet0,  Port ID (outgoing port): 4/8
Holdtime : 161 sec

Version :
WS-C5505 Software, Version McpSW: 5.1(1) NmpSW: 5.1(1)
Copyright (c) 1995-1999 by Cisco Systems
```

Showing the CDP neighbor detail statistics is a very handy troubleshooting tool if you know you are having problems with a certain IOS software version. You can use this command on core routers to check the software versions of each directly connected router.

If you want to see what the timer settings are for CDP broadcast updates, you can enter the following:

```
Router# sh cdp int
Ethernet0 is up, line protocol is up
  Encapsulation ARPA
  Sending CDP packets every 60 seconds
```

```
  Holdtime is 180 seconds
Serial0 is up, line protocol is up
  Encapsulation HDLC
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds
Serial1 is administratively down, line protocol is down
  Encapsulation HDLC
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds
```

The CDP timers (update interval and hold time) are global parameters. When they are changed, all interfaces running CDP are changed. The command to change the periodic update interval uses seconds. Without using the command the default value is 60. The command to change the hold time also uses seconds. The default value is 180. Remember that when timers are changed on one device, they should be changed on all the rest of the devices to help prevent neighbor-table synchronization problems.

```
Router# cdp timer seconds
Router# cdp holdtime seconds
```

CDP can be enabled and disabled either on individual interfaces or globally. If you want to turn off CDP on an individual interface, use the interface configuration mode command **no cdp enable**. To turn it back on, issue the **cdp enable** command. To turn off CDP on all interfaces simultaneously, issue the global configuration command **no cdp run**. Issue the **cdp run** command to turn CDP back on.

Debug Commands

Just as the show commands are useful for verifying router status and diagnosing configuration problems, the various debug commands can be helpful too. The **debug** command allows us to see what the IOS is doing as things happen, and it is normally used for troubleshooting and experimenting. We can turn on many different types of debug activities. Each one shows us something different about what is going on inside a router. To see the possible variations of the debug command, use the inline, context-sensitive help. Start by typing **debug ?**, and extend the command from there.

Debug output, by default, is logged to the console line and to terminal lines that have the monitor capability turned on. When we want to view debug output in a Telnet session, we can give our VTY the monitor capability by issuing the command **terminal monitor** in privileged mode.

There are a lot of debug commands that you can safely use in a classroom or test lab to show you what a router actually does while routing. Here's a tip: before doing any debug commands, type in the **no debug all** command. That way if something does go wrong while debugging and your router starts spewing out so much information that you can not type anything into the command line, all you need to do is to hit the UP arrow and press Enter. That'll turn the debug off.

To see all the debug commands just type **debug**, a space, and a question mark.

```
Router# debug ?

aaa               AAA Authentication,
                    Authorization and Accounting
access-expression Boolean access expression
all               Enable all debugging
alps              ALPS debug information
apollo            Apollo information
apple             Appletalk information
arap              Appletalk Remote Access
arp               IP ARP and HP Probe transactions
aspp              ASPP information
async             Async interface information
backup            Backup events
bri-interface     bri network interface events
bsc               BSC information
bstun             BSTUN information
callback          Callback activity
cdp               CDP information
chat              Chat scripts activity
clns              CLNS information
cls               CLS Information
cns               CNS Debugging
compress          COMPRESS traffic
condition         Condition
confmodem         Modem configuration database
cpp               Cpp information
custom-queue      Custom output queueing
decnet            DECnet information
dhcp              DHCP client activity
dialer            Dial on Demand
dlsw              Data Link Switching (DLSw) events
dnsix             Dnsix information
domain            Domain Name System
drip              DRiP debug information
```

```
dspu                DSPU Information
dxi                 atm-dxi information
eigrp               EIGRP Protocol information
entry               Incoming queue entries
ethernet-interface Ethernet network interface events
frame-relay         Frame Relay
fras                FRAS Debug
fras-host           FRAS Host Debug
funi                FUNI interface packets
gssapi              GSSAPI debugs
interface           interface
ip                  IP information
ipx                 Novell/IPX information
isdn                ISDN information
isis                IS-IS Information
kerberos            KERBEROS authentication and authorization
lapb                LAPB protocol transactions
lat                 LAT Information
ldap                LDAP debug commands
lex                 LAN Extender protocol
list                Set interface or/and access list
                      for the next debug command
llc2                LLC2 type II Information
lnm                 Lan Network Manager information
lnx                 generic qllc/llc2 conversion activity
local-ack           Local ACKnowledgement information
management          Management applications debugging
modem               Modem control/process activation
mop                 DECnet MOP server events
nbf                 NetBIOS information
ncia                Native Client Interface
                      Architecture (NCIA) events
netbios-name-cache NetBIOS name cache tracing
nhrp                NHRP protocol
ntp                 NTP information
nvram               Debug NVRAM behavior
packet              Log unknown packets
pad                 X25 PAD protocol
pcbus               PCbus interface information
ppp                 PPP (Point to Point Protocol) information
printer             LPD printer protocol
priority            Priority output queueing
probe               HP Probe Proxy Requests
qllc                qllc debug information
radius              RADIUS protocol
rif                 RIF cache transactions
rtr                 RTR Monitor Information
sdlc                SDLC information
sdllc               SDLLC media translation
serial              Serial interface information
sgbp                SGBP debugging
smf                 Software MAC filter
smrp                SMRP information
sna                 SNA Information
snapshot            Snapshot activity
snmp                SNMP information
source              Source bridging information
spantree            Spanning tree information
sscop               SSCOP
standby             Hot standby protocol
stun                STUN information
tacacs              TACACS authentication and authorization
tarp                TARP information
tbridge             Transparent Bridging
Telnet              Incoming Telnet connections
tftp                TFTP debugging
token               Token Ring information
translate           Protocol translation events
tunnel              Generic Tunnel Interface
v120                V120 information
vg-anylan           VG-AnyLAN interface information
vines               VINES information
vpdn                VPDN information
vprofile            Virtual Profile information
vtemplate           Virtual Template information
x25                 X.25, CMNS and XOT information
x28                 X28 mode
xns                 XNS information
```

```
xremote           XREMOTE
```

As you can see, there are lots of debug commands to choose from. The command **debug all** is too verbose -- it makes the information you're looking for hard to find in all the detailed data about literally everything in the router. We need to narrow down our focus a bit. Let's say we've been having a problem with our Ethernet interface. If we do a **debug ethernet-interface** command we can see what is going on.

```
Router# debug ethernet-interface
Ethernet network interface debugging is on
Router#
00:29:07: %LANCE-5-LOSTCARR: Unit 0,
  lost carrier. Transceiver problem?
00:29:17: %LANCE-5-LOSTCARR: Unit 0,
  lost carrier. Transceiver problem?
00:29:27: %LANCE-5-LOSTCARR: Unit 0,
  lost carrier. Transceiver problem?
```

As you can see, the Ethernet interface is definitely not working. Discovering what is not working is the whole key to troubleshooting. In this case, we know it's because the Ethernet is not connected to a network segment.

Fallback

You can define a fallback sequence of "boot system" commands that specify alternate ways for your router to boot if your router can't find the IOS on the first try. The boot system commands are placed in the startup configuration file and will only be used if the router is configured for normal boot sequence.

For instance, the boot system commands can specify that the router will load first from Flash memory, next from a TFTP server, and finally from ROM. In this case, the TFTP server is a fallback in case the Flash memory is corrupted and ROM is a fallback in case the TFTP server is unavailable.

After the Bootstrap Startup Program during the boot process (see Router Boot Sequence, above), the router scans the startup configuration in NVRAM for boot system commands to find out whether it should get its IOS image from a location other than the first file in Flash memory. If it finds boot system commands, it executes them in sequence until it finds a valid image that it can load. If there are no boot system commands, the router will attempt to load the first IOS image it finds in Flash.

Finally, if that fails, it will use the image in ROM as a fallback. This minimally featured image will allow IP addresses to be assigned to interfaces and will allow you to use ping and TFTP, but there's not much else in there. The prompt is

```
Router(boot)>
```

The ROM monitor is an unfriendly place to be, the commands are arcane, and the prompts give you no help.

We'll discuss how to write boot system commands in the next section.

The Configuration Register

During the boot process we can have the router look into the startup configuration for where to find the IOS image or we can have the router bypass this step and have it look in a specific place. The primary method for dictating how a router behaves on startup is the configuration register, also known as the config register. The config register on a Cisco router is a 16-bit register that tells the router what to do when booting.

With the config register, you can force the router to boot in ROM monitor mode, select boot source and filename, enable and disable the break function during boot, control broadcast address mapping, and control the load source of IOS. It is for this last reason that we will examine the last 4 bits of the configuration register. These last 4 bits are called the "boot field." Since the configuration register is shown in hexadecimal numbers, the last 4 bits will be represented by only one digit.

Another reason the config register is important is that when we forget our password, the config register plays a major part in password recovery. This is a topic for another Tutorial, but in the meantime the Cisco web site has more information about password recovery. (is not associated with Cisco.)

If the boot field is set to hex 00, then the router will boot to the ROM monitor mode on reload or power up. If this field is set to hex 01, the system will boot fully and load the first IOS image in Flash memory. A hex value of 02 is the most flexible, it allows default image booting from Flash as well as enabling "boot system" commands in the startup-config file. Table 3 shows the values and functions of the boot field.

Table 3. Boot Field Positions and Functions

Boot field value	Function
00	Boot to ROM monitor mode
01	Boot with default software contained in ROM
02-0F	Enables default booting from Flash, then enables "boot system" commands, then implicitly describes a default netboot filename

When the boot field is between 02 and 0F, the router boots from Flash. If it doesn't find an IOS image in Flash it will look to the boot system commands in the startup configuration. If it doesn't find one there, it can look outside of the router to the network. In this last case, the router will need a network filename or netboot filename. The default netboot filename is derived by taking the boot field value and pairing that up with the processor name (name=cisco<*n*>-*processor_name* where *n* is some number between 2 and 15).

If we put all these possibilities together, the config register will become 0x2102. Then the IOS will boot from Flash if a valid file exists or will enable "boot system" commands in the config file; if there is no valid software image in Flash, it will attempt to netboot the file name cisco2-2500.

By the way, to display the content of the configuration register, use the **show version** command. (See Show Version, above.)

Changing the Configuration Register

In order to modify the config register on your router, you **must** be connected to the console port. This cannot be performed via a telnet session to the router.

There are a couple of methods for changing the config register on a Cisco router. One of those methods can be done through regular config mode on the router. In order to change the config register, you must login to the router, enter privileged mode, and then enter config mode. Once there, you will simply enter a single command, **config-register**, that changes the config register (by the way, the shorthand for this is **conf**). In this command, the expected input is expressed in hex.

Since there is no difference between decimal and hex until digits get above 9, the way to denote hexadecimal notation is with the prefix 0x. For example, the notation for the configuration that would boot the router normally but ignore NVRAM is 0x2142, as shown below:

```
Router> en
Router# config t
Router(config)# config-register 0x2142
Router(config)# ctrl-z
Router#
```

This command sequence will change the router's config register from whatever it is to 0x2142. Now let's check to verify; type **sh ver**, Enter, and then hit the [space] bar. This will show the software version as well as the config register. At the bottom of the second screen of output, you will see the line

```
Configuration register is 0x2102
  (will be 0x2142 at next reload).
```

If you make a change to the config register, you will see the latter parenthesized statement. If you enter the config register command in config mode, but do not actually change the register, you will not see this output. The config register we have entered will cause the router to boot without reading NVRAM for the startup config, and we don't really want that, so let's change it back:

```
Router> en
Router# config t
Router(config)# config-register 0x2102
Router(config)# ctrl-z
Router#
```

Now when we perform a **sh ver**, we will see that the configuration register is 0x2102. There will no longer be the added statement telling us what the config register will be on the next reload.

NOTE: For heavens sake, if you do change the configuration register for any reason, be sure to set it back to what it was before so the router will use its proper configuration and find its IOS image properly the next time it boots up.

Booting from ROM (Boot Field = 01)

If the boot field of the config register is set to 01 (example: 0x2101) the router will boot from a default IOS image that is stored in ROM. In most cases, the software that is loaded in ROM is an earlier version and only has limited functionality. The reason for having this capability is in the case of a problem with the IOS software that resides in Flash.

If you have a problem with that image, you can boot to the default image and TFTP a new image to the router. If for some reason you are working on a router and happen to see a (boot) prompt (example: Router(boot)>), this means that the router is running in ROM monitor mode. This will give you a very quick check to see what the problem is with the router. Check the router and see what the config register value is. If it is 0x2101, then change the config register and reboot. If the config register is 0x2102, you know that something has happened to the software image loading from Flash.

Booting from Flash (Boot Field = 02)

If the config register has the boot field set to a value of 02, the router will boot from Flash. It is possible to have multiple images in Flash, as well as multiple partitions in Flash. If the boot field is set to 02, the router will look at the startup config to see if there is a boot system command telling it what file to load from Flash. If there are no boot system commands, the router will use the first valid system image in Flash. If there are no valid system images in Flash, then the router will attempt a netboot.

With a value of 02 in the boot field, a Cisco 2500 will send out broadcasts on the directly connected networks to retrieve an IP address and look for a

TFTP server with the file cisco2-2500 on it. If none of the above circumstances are met, then the bootstrap program will check the 13th bit of the config register. If the 13th bit has an "on" value, the router will load the default software out of ROM.

If there are multiple valid system images in Flash, boot system commands can be entered into the startup configuration to tell the router which one to use. The following is the syntax for configuring the router to boot from Flash and look for c2500-d-l.120-5.bin first, c2500-d-l.112-19.bin second, then boot from ROM:

```
Router# conf t
Router(config)# boot sys Flash c2500-d-l.120-5.bin
Router(config)# boot sys Flash c2500-d-l.112-19.bin
Router(config)# boot sys rom
Router(config)# [ctrl-z]
Router# copy run start
```

Boot system commands will be executed on startup in the order in which they are entered. In the above example, the system will try to boot with the 12.0 release software first, then the 11.2 software, and then revert to ROM.

There is another boot system command that tells the router to boot from the network (from a TFTP server). The syntax is similar, and the following will load the same images from a TFTP server with the IP address of 192.168.1.100.

```
Router# conf t
Router(config)# boot sys tftp c2500-d-l.120-5.bin
                192.168.1.100
Router(config)# boot sys tftp c2500-d-l.112-19.bin
                192.168.1.100
Router(config)# boot sys rom
Router(config)# [ctrl-z]
Router# copy run start
```

In this example, we explicitly told the router to look for the TFTP server at IP address 192.168.1.100. If the TFTP server was located on a LAN local to the router, we could have left this information off, and the router would have sent out a broadcast and located the server on its own. Another note regarding this type of configuration is that since there are options to booting, the router has a fallback. If the first option doesn't work, it can fall back on the second option, then the third, which is boot with the default software image in ROM. (See Fallback, above).

Appendix A. Review Questions and Answers

Review Questions

(Answers are provided at the end.)

Answer the following questions as either True or False.

1. The most common method for a router to find the configuration commands are the ones saved in NVRAM.

2. You can specify enabled config-mode boot system commands to enter fallback sources for the router to use in sequence.

3. In order to check any config-register setting you use either the **show running-config** or **show startup-config** commands.

4. The system image in ROM contains the full set of Cisco IOS software, which is equal to the one found in Flash.

5. The system image stored in Flash memory can be copied to ROM with the command **copy flash rom.**

6. By default, Cisco routers are DTE devices, but sometimes we need to turn them into DCE devices.

7. The configure console command is used to configure manually from the console terminal.

8. You can configure a message-of-the-day banner to be displayed on all connected terminals using the **banner config** command.

9. The first global parameter for which you are prompted by a router during Setup allows you to set the router's host name.

10. If Flash memory is corrupted, or its IOS is missing and the network server fails to load an IOS image, booting from ROM is the final bootstrap option in software.

11. During router Setup you are not prompted for parameters for each installed interface, you must enter these separately using the **configure interfaces** command.

12. During router setup you need not enter an enable secret password, but you must enter just the enable password.

13. Flash memory holds the operating system image and microcode, allowing updates to software without removing and replacing chips on the processor.

14. CDP runs over a data link layer, connecting lower physical media and upper-network-layer protocols.

15. Default values for CDP timers set the frequency between CDP updates and for aging CDP entries.

16. The **show version** command displays information about the Cisco IOS software version that is currently running on the router.

17. CDP can only discover information about directly connected Cisco devices if they are using the same protocol suite.

18. The TFTP server can be another router, or it can be a host computer system.

19. If no free Flash memory space is available, or if the Flash memory has never been written to, the erase routine is usually required before new files can be copied to it.

20. A router can only have one incoming Telnet session at a time.

21. The terminal editing command enables advanced editing.

22. Commands available in privileged mode are a subset of the commands available in the user mode.

23. From the user mode, you can also access global configuration mode and the other specific configuration modes.

24. You can press Control-C to terminate the setup process and start over at any time.

25. For many of the prompts in the system configuration dialog of the setup command facility, default answers appear in square brackets ([ ]) following the question.

Review Answers

1. True
2. False
3. False
4. False
5. False
6. True
7. False
8. False
9. True
10. True
11. False
12. False
13. True
14. True
15. True
16. True
17. False
18. True
19. True
20. False
21. True
22. False
23. False
24. True
25. True

Appendix B. Cisco Router Series

It can be useful to think of Cisco routers as having been introduced in several product generations. This isn't completely pure, because some devices called "switches" actually route. In general, however, it's useful to think of:

- First generation routers: totally obsolete proof-of-concept
- Second-generation: IGS, CGS, MGS, and AGS (which ran a recognizable IOS)
- Third-generation: 2500, 4000, and 7000, running at least IOS 9.0
- Fourth-generation: broadband access routers for ISDN, xDSL, CATV, etc., (all running IOS), 1600, 2600, and advanced 7100/7200, 7500. "Layer 3 switching" in 5000/5500 switches.
- Fifth-generation: very high capacity including 10000 and 12000; Layer 3 switching in 6000/6500/8500 switches.

Having looked at the most common Cisco router, we can now discuss the different models of Cisco routers and give some particulars about each one. There are several varieties of Cisco routers. The relevant router models are the 2500, 4000, 7000, and 7500 series. The 4000 is the next step up after the 2500 series in Cisco's product line. The following lists show some of the Cisco routers and give their primary uses. When the description of any Cisco router series includes the term "slot" you should think of an opening where a removable card or component can be inserted.

Series 700

This family of products is ISDN dial-on-demand routers for home offices and telecommuters (base $400 to $800).

- 761 - One Ethernet port plus ISDN BRI S/T
- 762 - One Ethernet port plus ISDN BRI NT1

- 765 - One Ethernet port plus ISDN BRI S/T plus two POTS
- 766 - One Ethernet port plus ISDN BRI NT1 plus two POTS
- 771 - Four Ethernet ports plus ISDN BRI S/T
- 772 - Four Ethernet ports plus ISDN BRI NT1
- 775 - Four Ethernet ports plus ISDN BRI S/T plus two POTS
- 776 - Four Ethernet ports plus ISDN BRI NT1 plus two POTS

Series 1600

This family of routers is a slightly scaled down version of the more expensive and popular 2500 Series family. Generally, it has one serial and one Ethernet port rather than two like the 2500 has. Unlike most of the 2500 family, the 1600's have a WAN module slot for flexibility.

- 1601 - One Ethernet port and one serial port
- 1602 - One Ethernet port and one 56K CSU/DSU
- 1603 - One Ethernet port and one ISDN BRI S/T
- 1604 - One Ethernet port and one ISDN BRI NT1
- 1605R - Two Ethernet ports

Series 2500

The 2500 Series is the world's most popular small-to mid-sized router. There are so many types of routers available in this series that it is helpful to organize the list into several groups as is done below:

Single LAN

- 2501 - One Ethernet port, two serial ports
- 2502 - One Token Ring port, two serial ports
- 2503 - One Ethernet port, two serial ports, one ISDN BRI
- 2504 - One Token Ring port, two serial ports, one ISDN BRI
- 2520 - One Ethernet port, two serial ports, one ISDN BRI, 2 low speed serial
- 2521 - One Token Ring port, two serial ports, one ISDN BRI, 2 low speed serial
- 2522 - One Ethernet port, two serial ports, one ISDN BRI, 8 low speed serial
- 2523 - One Token Ring port, two serial ports, one ISDN BRI, 8 low speed serial

Dual LAN

- 2513 - One Ethernet, one Token Ring, two serial ports
- 2514 - Two Ethernet ports, two serial ports
- 2515 - Two Token Ring ports, two serial ports

Router/Hub Combo

- 2505 - Eight-port Ethernet hub, two serial ports
- 2507 - 16 port Ethernet hub, two serial ports
- 2516 - 14 port Ethernet hub, one ISDN BRI, two serial ports

Access Servers

- 2509 - One Ethernet, eight asynch, two serial ports
- 2511 - One Ethernet, 16 asynch, two serial ports
- 2512 - One Token Ring, 16 asynch, two serial ports

Series 2600

This family of routers is very comparable to the popular 2500 series, except they have slots, including one Module slot for features like voice and fax, and two WAN slots for options like built-in CSU/DSU, ISDN, serial ports, asynch ports, and so forth.

- 2610 - one Ethernet, one Module slot, two WAN slots
- 2611 - two Ethernet, one Module slot, two WAN slots
- 2612 - one Ethernet, one Token Ring, one Module slot, two WAN slots
- 2613 - one Token Ring, one Module slot, two WAN slots
- 2620 - one Ethernet 10/100, one Module slot, two WAN slots
- 2621 - two Ethernet 10/100, one Module slot, two WAN slots

Series 3600

The 3600 Series is a multifunction platform that combines dial access, routing, LAN-to-LAN services, and multifunction integration of voice, video, and data in the same device. It is somewhat similar to the 2600 Series in regard to available slot options except it has a much richer list of those options. Also, the 3600 is a router meant to be installed in a larger network, like a WAN headend where the core of the WAN is located.

Series 4000

The third-generation Cisco 4000 series consists of the Cisco 4000-M, the Cisco 4500-M, and the Cisco 4700-M. The Cisco 4000 series is used in middle-size networks and at the distribution layer of large internetworks. All models provide a configurable modular router platform using network processor modules including FDDI, ISDN BRI, ISDN PRI, Ethernet 100baseT, Token Ring, HSSI, and ATM. The Cisco 4000 series routers support up to three network processor modules at a time. The Cisco 4700-M contains a 133-MHz RISC microprocessor, 16 to 64 MB main memory, and a 512-KB secondary cache. The faster speed of the Cisco 4700-M allows higher throughput for high-speed interfaces. The 512-KB secondary cache is useful for process switching applications such as compression and encryption.

In new installations, the 4000 series has been superceded by the 3600 series.

Be sure to pay attention to the model number. The routers are the 4000, 4500, and 4700. There is a new series of gigabit layer 2 switches called the 4000 series.

AGS+ and Series 7000

Some Cisco routers were the most powerful of their time, but have been superceded by newer models. The AGS+ is a "second generation" router, which you may find available cheaply. It generates substantial heat and noise, and will run IOS versions no later than early 11. Nevertheless, many certification candidates find it cost-effective in their labs. Be sure that you buy a version with the CSC/4 processor if you expect to run any recent IOS. The AGS+ was the first Cisco router that could use different processors for path determination and packet forwarding.

The third-generation replacement for the AGS+ was the Cisco 7000 series of multi-protocol routers, including the Cisco 7000 and the Cisco 7010. The 7000 and 7010 differ only in their number of card slots, Network interfaces reside on modular interface processors, which provide a direct connection between the high-speed Cisco Extended Bus (CyBus) and the external network. Distributed processing is accomplished by the Route Processor (RP), Switch Processor (SP), and Silicon Switch Processor (SSP).

Series 7100 and 7200

The fourth-generation Cisco 7200 series of multi-protocol routers delivers the performance, port density, and availability typically associated with high-end systems. This router is used as a high-speed backbone aggregation router for high-speed enterprise interconnectivity. It is used for a WAN edge concentrator at the backbone and IBM interconnectivity. The 7100 has additional hardware to assist it in encryption for virtual private networks.

Series 7500 and 10000

The Cisco 7500 was designed to meet the demands of emerging high-end application environments -- in terms of density, performance, and system availability. Like the 7200 series, this series is used for high-speed backbone aggregation needed for high-speed enterprise interconnectivity. It is used for a WAN edge concentrator at the backbone.

Series 12000

Cisco's new family of gigabit switch routers (GSR) provides high-performance solutions ranging from five to 60 Gbps for Internet and large-scale WAN intranet backbone applications.

Basic Router Operation Labs

Basic Router Operation Labs

Basic Router Operation Lab

Tutorial
Lab Abstract
Lab Scenario

In this lab you will complete the following tasks:

- Back up a router configuration file to a TFTP server
- Back up a router IOS software image to a TFTP server
- Restore a router configuration file from a TFTP server
- Restore a router IOS software image from a TFTP server

Basic Router Operation Labs

Basic Router Operation Labs

Basic Router Operation Lab

Tutorial
Lab Abstract
Lab Scenario

Backing Up and Restoring Router Configuration Files and IOS Software Images

Objectives
Setup
  TIP
Scenario
  Task 1: Back up a router configuration file to a TFTP server
    Step 1-1
    Step 1-2
    Step 1-3
    Step 1-4
  Task 2: Back up a router IOS software image to a TFTP server
    Step 2-1
    Step 2-2
    Step 2-3
  Task 3: Restore a router configuration file from a TFTP server
    Step 3-1
  Task 4: Restore router IOS software image from a TFTP server
    Step 4-1
Solutions
  Task 1, Step 2.
  Task 1, Step 3.
  Task 1, Step 4.
  Task 2, Step 1.
  Task 2, Step 2.
  Task 3, Step 1.
  Task 4, Step 1.

Complete this lab to practice what you learned in the Basic Router Operation Tutorial.

Objectives

In this lab you will complete the following tasks:

• Back up a router configuration file to a TFTP server

• Back up a router IOS software image to a TFTP server

• Restore a router configuration file from a TFTP server

• Restore a router IOS software image from a TFTP server

Setup

Your router should have at least a basic configuration from either completing the router's setup script or manually configuring your router similar to the sample configuration in the Basic Router Operation Tutorial.

You should also install and configure a TFTP server in your lab network. This is fairly easy to do, and any PC or laptop can function as your TFTP server for the purposes of this lab. For more information about finding TFTP software, see Backing Up Router Configuration Files in the Basic Router Maintenance and Troubleshooting section of the Basic Router Operation Tutorial.

TIP

Many TFTP server implementations require you to access files, for both upload and download, by their fully qualified name -- not their name relative to a directory.

Scenario

You've configured your router, and now you want to save your configuration somewhere other than NVRAM. You also want to back up your router's IOS software image in case your router suffers a critical failure. Finally, you want to test these backups by restoring them to your router.

Task 1: Back up a router configuration file to a TFTP server

Step 1-1

Enter privileged EXEC mode.

Step 1-2

Save your running configuration to NVRAM.

What command saves your running configuration to NVRAM?

_____

Step 1-3

Backup your running configuration to the TFTP server.

What command will back up your running configuration to the TFTP server?

_____

Refer to the Backing Up Router Configuration Files section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Give your configuration file a unique name.

Step 1-4

Find and view your configuration file on the TFTP server.

What is different about the configuration file on the TFTP compared to your original running configuration? (Do a **show run** on your router if necessary.)

_____

Task 2: Back up a router IOS software image to a TFTP server

Step 2-1

Look at the contents of Flash memory.

What command shows you the contents of Flash memory?

_____

How many files are currently in Flash memory?

_____

List the files currently in Flash memory:

_____

What file, ending in .bin, will always be found in the Flash memory of newly configured routers?

_____

How many bytes of Flash memory is the .bin file taking up?

_____

How many bytes of Flash memory are left?

_____

Step 2-2

Back up your IOS software image to your TFTP server.

What command backs up your IOS software image to your TFTP server?

_____

Refer to the Backing Up Software Images section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Use the name of the .bin file you wrote above as the source file name.

Step 2-3

Find your IOS software image on the TFTP server to verify that it transferred correctly.

Task 3: Restore a router configuration file from a TFTP server

Step 3-1

Restore the configuration you saved to the TFTP server in Task 1 to your router's current running configuration.

What command restores the configuration you saved to the TFTP server in Task 1 to your router's current running configuration?

_____

Refer to the Backing Up Router Configuration Files section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Use the name you gave your configuration file.

What command would you enter to restore the configuration from the TFTP server directly to your router's startup configuration?

_____

Task 4: Restore router IOS software image from a TFTP server

Step 4-1

Restore the IOS software image you saved to the TFTP server in Task 2 to your router's Flash memory.

What command restores the IOS software image you saved to the TFTP server in Task 2 to your router's Flash memory?

_____

Refer to the Backing Up Software Images section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Use the name of the .bin file you wrote in above as the source file name.

Why would the router need to erase Flash memory in order to restore an IOS software image from a TFTP server?

_____

How does the router indicate that Flash memory is being erased?

_____

How does the router indicate that the IOS is being copied?

Solutions

Task 1, Step 2.

- **copy run start**

Task 1, Step 3.

- **copy run tftp**

Task 1, Step 4.

- The configuration file on the TFTP server will have no comments [lines beginning with an exclamation point (!)] in it. The comments get stripped out when being transferred to the TFTP server. You can add them back in on your TFTP server with a regular text editor.

Task 2, Step 1.

- **show flash**

The number and name of files currently in Flash will vary from router to router. On newly configured routers the only file in Flash is the system image file (the file that ends in .bin). The number of bytes of Flash memory that the system image file takes up will vary from router to router. There should be at least several megabytes of Flash memory still available, however.

Task 2, Step 2.

- **copy flash tftp**

Task 3, Step 1.

- **copy tftp run**

- **copy tftp start**

Task 4, Step 1.

- **copy tftp flash**

Note the size of the IOS software image from Task 2, Step 1. If there is not enough room in Flash memory to save a second copy of the IOS software image, it will need to erase the Flash to make room.

The letter 'e' is output to the screen to indicate that Flash memory is being erased.

Exclamation points are output to the screen to indicate that the IOS is being copied.

Basic Router Operation Labs

640-801 Labs

IP Addressing Lab

Tutorial
Lab Abstract
Lab Scenario

IP Addressing

There are several related areas you must understand before you can make a router do useful things. You must understand: (1) how hosts and routers connect to physical media; (2) the potential relationships among physical media (and simulated physical media) to logical media; and (3) the conventions for IP and IPX addressing that involve setting up identifiers for points on logical media, and how to code these identifiers into routers.

In this tutorial, the author begins with a review of topologies and then reviews IP addressing in a more binary and classless way, finally mapping into the decimal and classful way.

640-801 Labs

IP Addressing
Tutorial
Lab Abstract
Lab Scenario

IP Addressing Labs

IP Addressing Lab

Tutorial
Lab Abstract
Lab Scenario

Topology and IP Addressing: the CCNA Perspective

Introduction
Current Addressing
Topologies
  MAC Addressing
  Logical and Physical Relationships: Topologies
  One-to-One Relationships
  Relationships beyond 1:1
    Partial Meshes
    Demand Circuits
  Logical and Physical: Mappings
  Basic Mappings
    Multiple Logical per Physical Medium: Secondary Addressing
    Multiple Physical Treated as Single Logical: Basic VLAN
    Multiple Physical with Multiple Logical: VLAN with secondaries
What do Routers do with Addresses?
  A Telephone Analogy
  Network Addresses and Routing
  Matching Routes
  A Special Case: The Default Route and ip classless
    ip classless
    Default Gateway
    Default Network
    Gateway of Last Resort
IPv4 Evolution
    Just because you can do something, it isn't necessarily a good idea.
  Routers and Prefixes
  Classes
  Getting Address Space
    Provider Assigned Address Space
    Private Addresses
IP Addresses: Computer Views, Human Views
  Dotted-Decimal Notation is for People, not Routers
    Principles of Dotted Decimal
    Weighted Binary
  Subnetting versus Subnet Masks
    Prefix Length Display Formats
    Extracting Prefixes from Addresses
    Reviewing the AND operation
  Reserved Host Field Values
  Prefix Practice
Setting Up Simple Address Plans
  What kind of network? What Mask?
  Addressing Simple Interconnected LANs
Configuring IP Addresses into Cisco Routers
  Basic interface statments
  Secondary Addressing
    Stupid hosts and Classful Addressing
  Subinterfaces
Conclusion
References

Introduction

Wernher von Braun is said to have described research as "that which you do when you don't know what you are doing, and know you don't know." When beginners to Internet Protocol (IP) addressing start dealing with it, all too often they assume some of its quirks are part of extremely subtle and thoughtful original design ideas. In reality, many of IP's quirks exist because researchers made a best guess based on the experience of the time.

Some things have remained true over time. A network address is always hierarchical. The higher-order part identifies a medium, and the lower-order part identifies a host on that medium. The routing system makes decisions about forwarding based on the high-order prefix part of network addresses.

Network addresses therefore not only identify devices, but also give hints on the path to take to get to them. End hosts have network addresses and transport layer identifiers that tell how to get to a specific software process once a packet reaches that host. Routers have collections of interfaces, each interface with its own logical address.

With experience, many of the guesses made for the first IP addresses turned out to be less than optimal. By the time the industry gained this experience, however, the software that implemented this code was too widely deployed to be easily changed.

Current Addressing

The current version of IP is Version 4 (IPv4). A new version, IPv6, has been developed, built on 15-plus years of experience with IPv4. Given the very large installed base of IPv4 systems, it is unclear how quickly, if at all, IPv6 will replace IPv4. The trend to v6 is accelerating in new networking industries. Third-generation wireless networks have selected IPv6, so commercial products in 2002 or 2003 seem likely. In other words, don't worry about IPv6 in the near term.

There are several related areas you must understand before you can make a router do useful things. You must understand how hosts and routers connect to physical media. You must understand the potential relationships among physical media (and simulated physical media) to logical media. You must learn the conventions for IP and IPX addressing that involve setting up identifiers for points on logical media, and how to code these identifiers into routers.

Cisco no longer publishes specific objectives for exams. Probably the most reliable reference on the CCNA, however, is the curriculum for the first four semesters of the Networking Academy. Intended to qualify students for the CCNA, this curriculum includes the following:

- Describe data link and network addresses and identify key differences between them. (a3)
- Define and describe the function of a MAC address. (g15)
- Describe the two parts of network addressing, then identify the parts in specific protocol address examples. (d2)
- Describe the different classes of IP addresses [and subnetting]. (d3)
- Configure IP addresses. (d4)
- Verify IP addresses (d5)

You will find additional material on IP-related objectives in the CCNA Tutorials on IP Routing and Network Security.

Unfortunately, IP addressing is one of those areas where there is a right way, a wrong way, and a Cisco way. The Cisco way for CCNA IP addressing objectives is based on obsolete classful principles, which will be explained below. Many, if not most, introductions to IP persist in using the decimal and classful approach, which is neither the way Internet addresses actually are assigned, nor a straightforward way to learn.

In this paper, I review IP addressing in a more binary and classless way, and then map the methods into the decimal and classful way. While some consider this an unconventional means of teaching, I have used it with hundreds of students and find they learn addressing faster and more accurately. The addressing presentations in Cisco's CID and ACRC courses are drawn from an internal briefing I did using this approach.

So, if you have learned the classful way and are expecting a review in the same manner in which you learned things, I ask for a bit of patience while I show you how to understand even better. If you are a beginner, you don't have bad habits!

Topologies

Routing has two main components, drawing the map (i.e., path determination) and moving packets, step by step, onto media one hop closer to the destination (i.e., packet forwarding). The logical destination addresses in routed packets give information about the appropriate next-hop medium to which the router will forward.

In routing, you relay packets from a sending host to its local medium, from the local medium to a local router, from the local router via another medium to intermediate router(s), and eventually to the egress router. The egress router sends the packet onto the medium where the destination host lives.

If you didn't notice it, let me make it explicit. Routing is based on the relationships among a set of media. There are several different kinds of medium topology. In data communications usage, topology refers to the number of devices that can connect to a physical or logical medium, and the ways in which they interconnect on that medium.

From basic data communications, you should be familiar with simple point-to-point lines -- no more complex than a cable -- and with LANs. All hosts connected to the same LAN segment can reach one another using layer 2 protocols.

Table 1. Possible Topology Types for Various Media

Technology	Physical Topology	Logical Topology	Endpoint address
Ethernet and 802.3 10Base5, 10Base2	bus, star	bus	48-bit MAC address
Ethernet 10BaseT	Star	bus	48-bit MAC address
Token Ring	star	ring	48-bit MAC address
FDDI dual	ring	ring	48-bit MAC address
FDDI single	point-to-point	ring	48-bit MAC address
FDDI slave to concentrator	star	ring or ring of stars	48-bit MAC address
Dedicated line	point-to-point	point-to-point	no
Frame relay	point-to-point	point-to-point or NBMA	10-bit data link connection identifier (DLCI)
ATM	point-to-point	point-to-point or NBMA	Primarily 20-byte NSAP. Some additional forms.
Dialup, ISDN	point-to-point	point-to-point or NBMA	National telephone number conforming to ITU-T E.163 or E.164 (for ISDN)
X.25	point-to-point	point-to-point or NBMA	Up to 14 byte X.121

You may also be familiar with partial mesh, virtual circuit media such as Frame Relay and ATM. In this section, we will formalize your knowledge, and give you the background to understand how you apply addressing to different medium types.

A slight bit of formalism will tighten up any discussion of topology. Adjacency and connectivity are terms from mathematical graph theory, which are extremely useful in discussing network topology. A graph is a set of nodes with paths among them. Terms discussed in this section, such as adjacency and connectivity, are as relevant to switches at layer 2 as to routers at layer 3. In Figure 1. Basic Topology, router **garlic** has two directly connected neighbors, **ginger** and **spearmint**. **ginger** has a neighbor of its own, **cinnamon**. Assume there are magical routing mechanisms in place that let each router learn about destinations on the other routers, so a host on **garlic** can reach one attached to an interface on **cinnamon**.

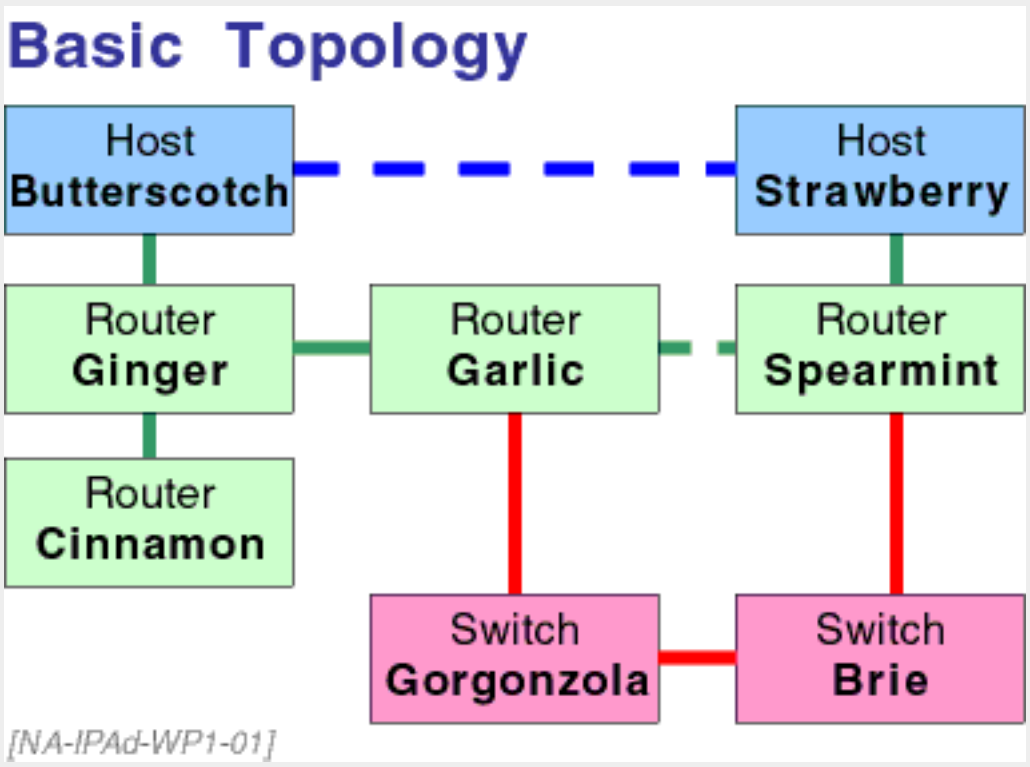


Figure 1. Basic Topology

Two nodes that are adjacent are also connected, but not all connected nodes are adjacent. In routing, the number of hops between two points is the number of routers between the two endpoints. In Figure 1. Basic Topology, there are two hops on the path between **cinnamon** and **spearmint**.

Application hosts **butterscotch** and **strawberry** believe they are directly connected at the application layer, but, as indicated by the dashed line, this is a virtual relationship mapped onto a lower layer. In like manner, routers **garlic** and **spearmint** think they are directly connected, but actually have a virtual relationship through the layer 2 switches **gorgonzola** and **brie**.

A more technical term for routers that are neighbors -- that are connected by a common medium -- is that they are adjacent or that they have adjacency. Routers that must go through intermediate routers to reach other destinations have connectivity, but not adjacency, with the destination. Connectable is a synonym for reachable -- it may be achieved either with connection-oriented or connectionless protocols.

More common terms are that adjacent things are neighbors, while connectable things are reachable through a network(s). There are multiple meanings of "connection" in networking, and we are only using the topological meaning here!

You can have logical connectivity between endpoints even if you don't have physical connectivity. For this to happen, the underlying transmission system has to be able to reach each successive link on the path between source and destination. Technologies such as dynamic routing may make the path change over time, but there always must be a set of reachable links for logical connectivity to exist.

Two points have connectivity when they can exchange information, but not necessarily directly. Intermediary things (e.g., the telephone network) may be needed to carry out the information exchange. In Figure 1, the application hosts are unaware of the routers. **Butterscotch** thinks it is talking directly to **Strawberry**.

The routers are unaware of the switches. **Garlic** thinks it is talking directly to **Spearmint.** There are different topologies at each protocol layer.

MAC Addressing

48-bit Medium Access Control (MAC) addresses are used for all current LANs. The ordering of bits inside a frame differs between Ethernet-style and Token Ring/FDDI style.

The most significant bit can appear as the leftmost (the "canonical" Ethernet style) or rightmost (non-canonical, Token Ring) bit of the first byte of the MAC address. This bit is set to **1** when the address describes a multicast or broadcast group address and to **0** if the address is for a unicast individual address. Bit 2 is set to **1** when the address conforms to the global IEEE convention and to **0** if it is locally administered.

Under global addressing, the first six hexadecimal digits of the address (i.e., its first 24 bits, of which 22 represent a vendor code and 2 bits are used in protocol functions)

- System administration practices associate a certain MAC address with a specific user or location. In such cases, the addressing system would be useless if a MAC address changed whenever a board was changed for maintenance. Locally administered MAC addresses can be set with an IOS command "mac-address address". This is most often used on Token Ring interfaces.
- Some network layer protocols, including those of DECnet, XNS, Novell, and Banyan, modify some or all MAC addresses to reflect layer 3 information.

Several non-IP protocols (DECnet, Novell IPX, XNS, and Banyan VINES) may change the MAC addresses of router interfaces, as part of their particular way of avoiding the need to ARP. These strategies can cause compatibility problems in multi-protocol environments.

Logical and Physical Relationships: Topologies

Throughout this paper, there are references to logical and physical topology. You will find more detail about these relationships as you go along. Figure 2 shows the various relationships.

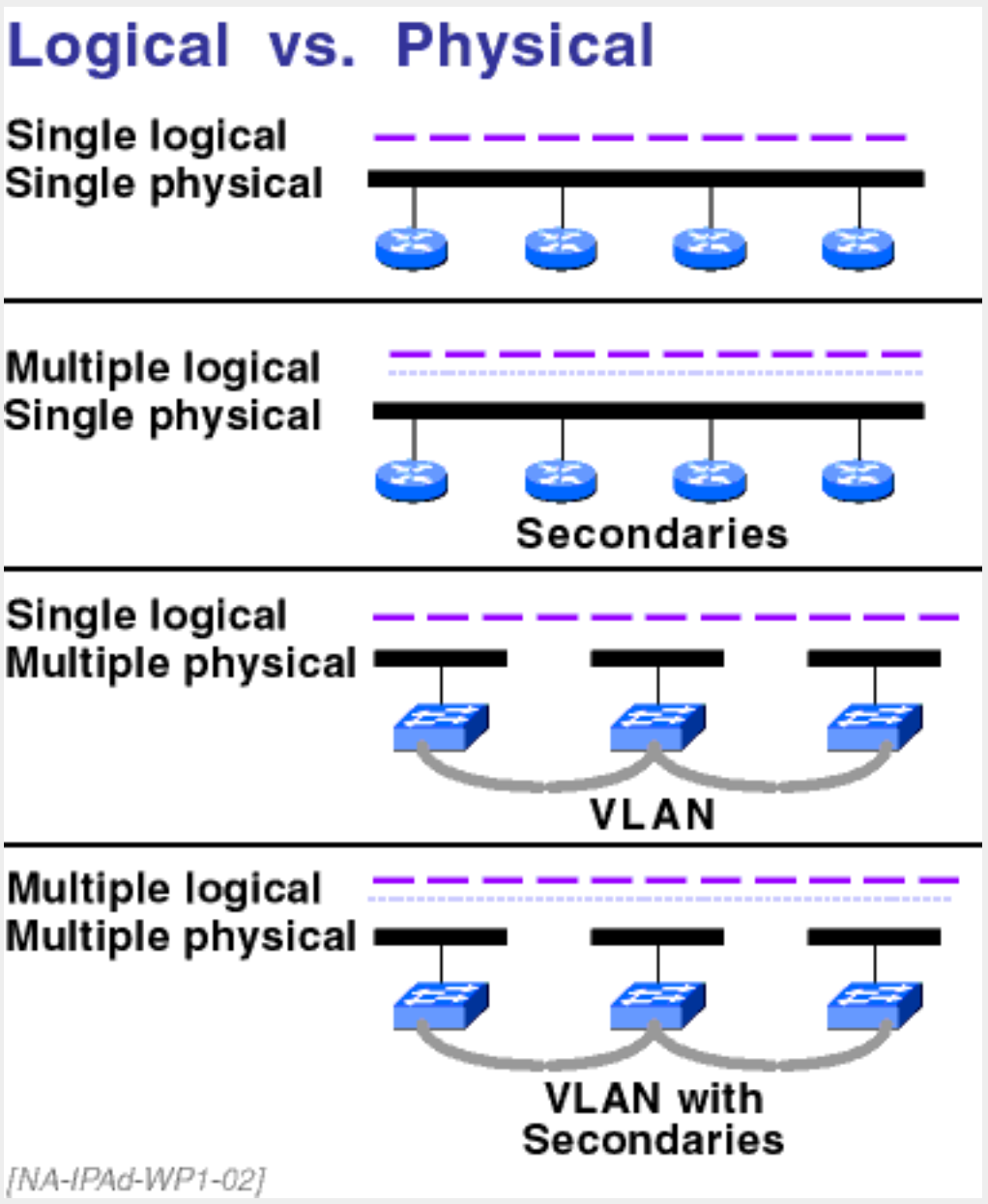


Figure 2

As mentioned above, *adjacent* nodes share a common medium. Nodes that are *connected* can have a path traced between them, but that path may go through multiple routers and media.

One-to-One Relationships

The simplest one-to-one network topology is a direct connection between two entities. Both users in a pure one-to-one relationship are zero hops from each other.

Several simple topologies can be created with direct lines between nodes. When speaking in terms of graphs, do not confuse the abstraction of a line with a physical, one-to-one transmission facility such as a "telephone line." In practical terms, however, a persistent one-to-one relationship is nicely illustrated by a cable, while a telephone call is a good example of a transient one-to-one relationship.

To consider more complex relationships, you will want to generalize the idea of a one-to-one topology to a one-to-many topology.

Relationships beyond 1:1

If a set of nodes either shares a common broadcast medium, or point-to-point connections exist among all of them, they are in a *full mesh* relationship. Figure 3 shows a full mesh.

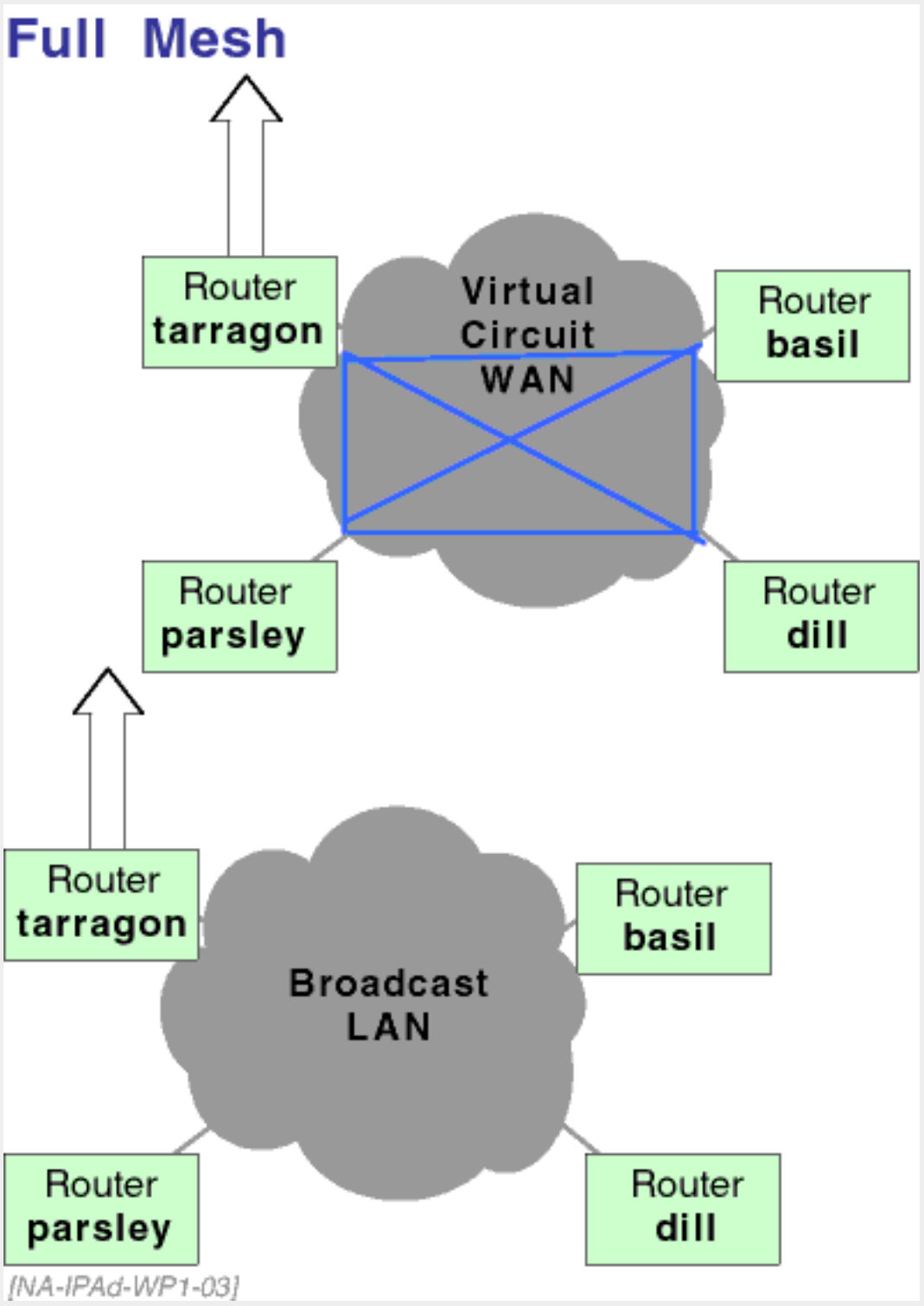


Figure 3. Full Mesh

The most general topology is full mesh. All stations have many-to-many connectivity to one another. It is the basic model of a LAN.

This may be a valid model for application connectivity, but simply does not scale well at the levels actually concerned with moving bytes. The need for all workstations to know the name and location of all other workstations would present, as the number of users grew, an overwhelming maintenance and performance penalty.

It is worth noting that a given station may belong to several different topological relationships. Tightly controlled hierarchies often are most reliable for the infrastructure task of maintaining the network itself, while more meshed structures are a better fit for the user view for interapplication communications. Even at the application level, there will often be hierarchy not visible to the end user. A local workstation, for example, may interact with a local server. The local server interacts with other servers only when it needs to, on behalf of a number of local workstations. The network is more scalable when every workstation does not need to communicate directly with every server.

When all nodes in a graph are directly connected, a mesh is formed. A mesh is a many-to-many topology. In actuality, a mesh is made up of multiple one-to-many relationships. Mathematically, each of the N nodes needs (N-1) links to all other nodes. These links can be point-to-point, or operate over a shared medium. In the top part of Figure 3, **tarragon**, **parsley**, **basil**, and **dill** are connected by a logical full mesh made up of point-to-point media. If any medium fails, the relationship among the routers is no longer point to point. In the bottom part of Figure 3, the same nodes are joined to a common broadcast-capable medium.

LANs exhibit full mesh behavior because they operate over a common medium. While it is possible to build full meshes out of WAN links, the reality is that WAN links occasionally fail, turning the full mesh into a partial mesh.

Partial Meshes

Partial mesh topologies are extremely common in modern networks, but they were not anticipated by the original IP addressing model. That original model assumes that if another address is in the same subnet, you will have layer 2 connectivity to it. If it is in a different subnet, you will need a router to reach it.

A wide range of issues occurs on nonbroadcast multiaccess (NBMA) media such as Frame Relay, ATM, and X.25. Many of these issues come from the way in which these media violate an early assumption in IP architecture called the local versus remote assumption. Figure 1. Basic Topology shows how the router assumes that hosts on the same subnet share layer 2 connectivity, but an intervening router is necessary to reach a host on a different subnet. This is a perfectly reasonable assumption on fully meshed broadcast media such as LANs. The assumption is a non-issue on point-to-point lines.

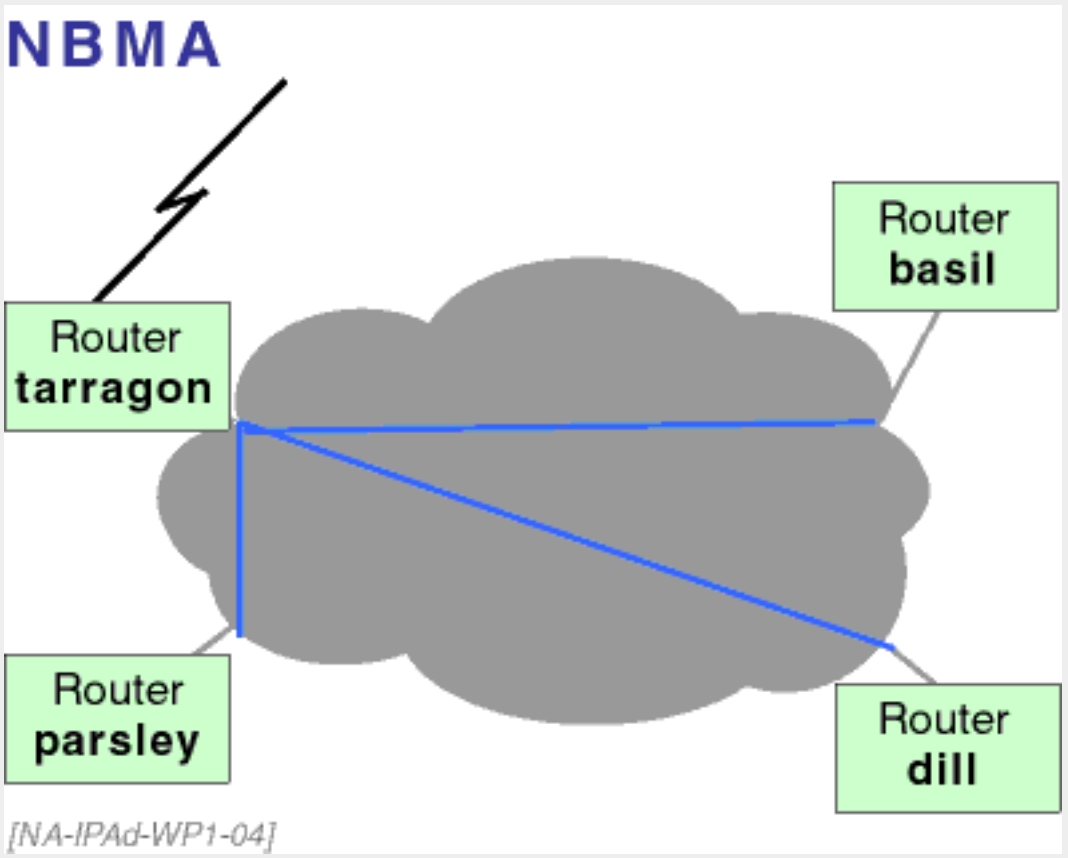


Figure 4. NBMA

As shown in Figure 4. NBMA, there is a problem in routers interconnected on NBMA partial mesh media. **Parsley** does not know it needs to forward to **tarragon** to reach **basil**.

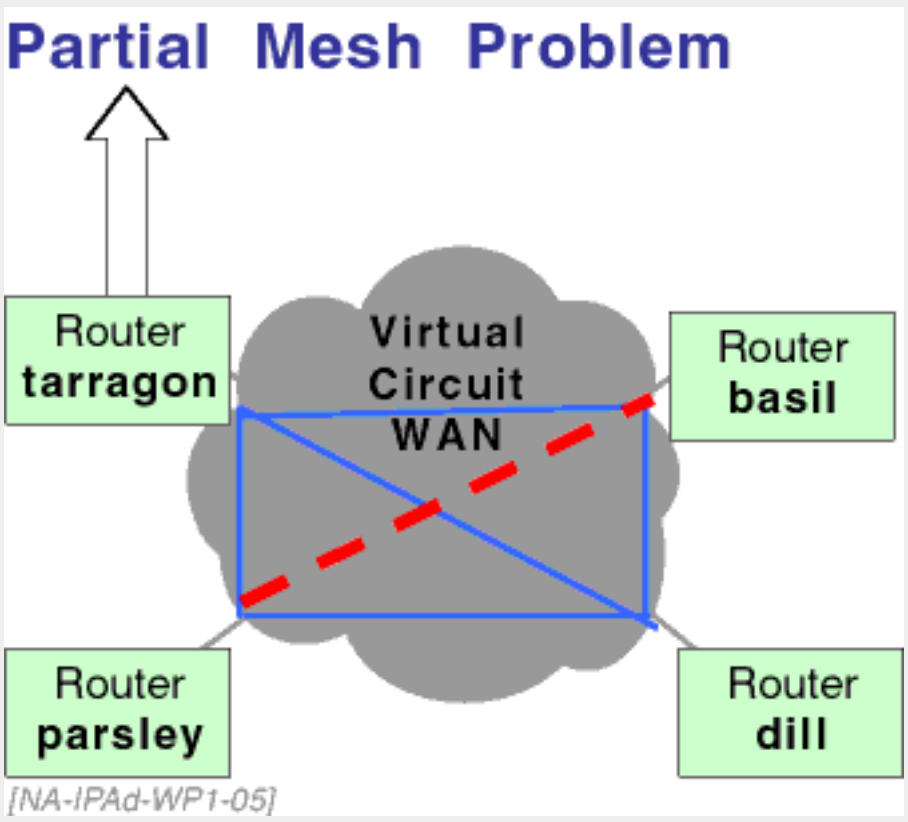


Figure 5. Partial Mesh Problem

In general, partial meshes should be avoided among routers. A danger is that you may have a partial mesh occur not because you have designed for it, but due to a failure as shown in Figure 5. The most common workaround is to create subinterfaces for each virtual circuit, and treat the virtual circuits as logical point-to-point subnets with a /30 prefix.

Each subinterface needs its own set of buffers, so large numbers of virtual circuits can require excessive amounts of memory. You can also declare point-to-multipoint subinterfaces, as shown in Figure 6. Point-to-Multipoint. Point-to-multipoint is feasible in a hub-and-spoke topology. In addition to subinterfaces, OSPF has an alternative way of defining point-to-multipoint networks.

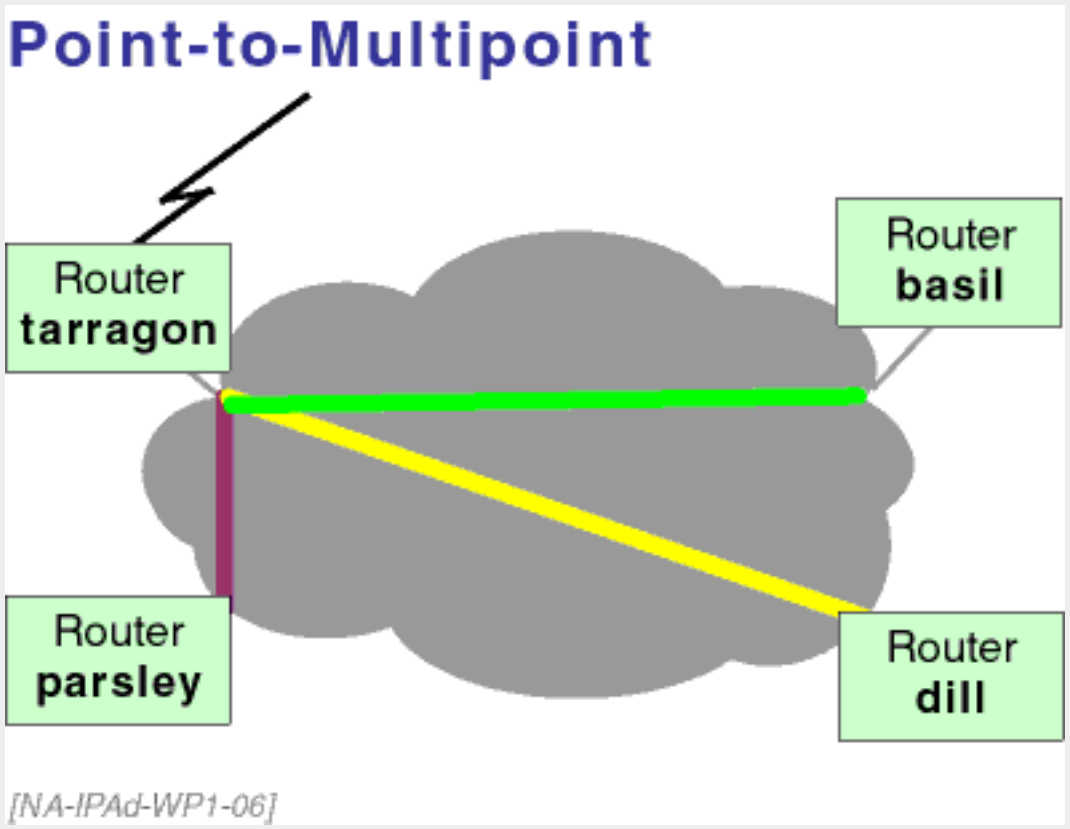


Figure 6. Point-to-Multipoint

Demand Circuits

Originally, IP addressing assumed there was a long-term relationship between the physical and logical address. Think, however, of a dial connection to an ISP. You dial a main number, but the specific dial-in port to which you will connect varies from session to session.

A physical relationship that only exists when commanded to do so is a *demand circuit*. At higher levels of certification, when you work on Cisco remote access, you will see many references to *dial on demand routing (DDR)*.

Logical and Physical: Mappings

For your network to work, both hosts and routers need to associate physical/media addresses with logical addresses.

Table 2. Relationships among Logical Addresses, Transmission System Addresses, and Mapping between Them

Logical	**IP**							
Mapping	ARP	IPCP, static	IPCP, static	Static	Inverse ARP, static	Inverse ARP, static	ARP, static	ARP, static
Tech-nology	LAN	Dial	ISDN	X.25	Frame	ATM AAL	SMDS	LANE
Protocol type ID	LLC, SNAP or Ether-type	PPP IPCP	PPP IPCP	RFC 1355	RFC 2427	RFC 2684, RFC 2225	LLC, SNAP or Ether-type	LLC, SNAP or Ether-type
Persistent endpoint identifier (i.e., medium)	MAC	E.163	E.164	X.121	DLCI	NSAP	MAC	MAC
Transient connection identifier	[1]	[2]	TEI	LCN		VPI & VCI	N/A	[3]
Next Lower Layer	LAN PHY	Analog	ISDN PHY	serial	serial	SONET, etc.	DS1, DS3, ATM	ATM

[1] Connectionless

[2] There is no specific identifier because analog lines do not carry any complex signaling. There is effectively a connection identifier, but it tends to be physical. Think of a multi-button key telephone, on which a button blinks for incoming calls, and stays on when a line is in use. That button is the connection identifier.

[3] The ATM VC identifier does not specifically point to the MAC address. Instead, it points to the LAN Emulation Client to which the MAC address is connected.

Basic Mappings

Briefly, the most basic relationship between a logical and physical medium is one to one. Historically, IP made the local versus remote assumption: if one IP host was on the same logical medium (i.e., subnet or prefix), it was assumed to have layer 2 connectivity with all other hosts on the same logical medium. If it were on a different logical medium, the host would have to use a router to get to the other host.

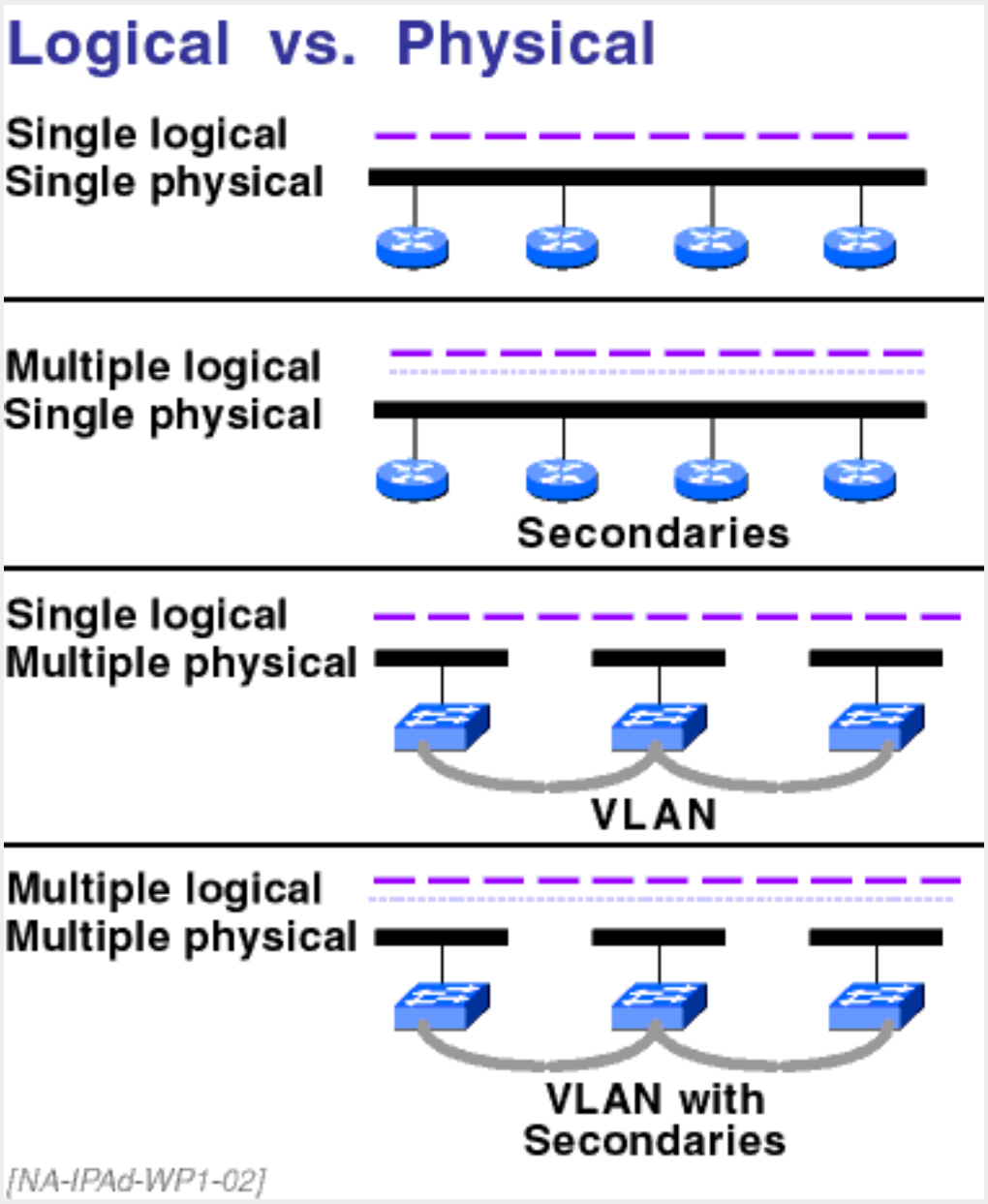
This situation can be addressed by using secondary addresses, which map multiple logical addresses to a single physical address, giving an interface the ability to be on more than one logical medium at a time.

The next refinement is typified by virtual LANs (VLANs), although its characteristics are shared by WAN media using virtual circuits, such as ATM and Frame Relay. In this refinement, several physical media are linked by a trunking mechanism that gives the impression of a single seamless physical medium.

Finally, secondary addresses can be combined with virtual networking. Doing so will allow you to have multiple logical networks mapped onto multiple physical networks.

Multiple Logical per Physical Medium: Secondary Addressing

In secondary addressing, detailed later, you assign more than one logical medium to a single physical medium. Nortel/Bay refers to this process as **multinet addressing.** Secondary addressing is often not necessary in more modern environments, but it can solve many addressing and routing problems found in the classful style of addressing that CCNA candidates will encounter.



Multiple Physical Treated as Single Logical: Basic VLAN

A VLAN, discussed in the CCNA LAN Switching Tutorial, creates a layer 2 relationship among multiple physical media, so users connected to different physical segments can appear as part of the same logical network. VLANs do more than simple bridging, because they can carry traffic belonging to several logical networks on a shared high-speed trunk. Trunks are generally 100 Mbps or faster, and are used to link wiring closets in different floors or buildings.

Multiple Physical with Multiple Logical: VLAN with secondaries

Combining secondary addressing with VLANs allows you to have users at an arbitrary location belong to one of several logical networks, to which they are not physically connected.

What do Routers do with Addresses?

Routers make forwarding decisions based on destination addresses in packets. These decisions are made on some number of high-order (i.e., leftmost) bits in the address. Routers rarely use all bits in an address to make a routing decision. They look at high-order bits to find the medium on which the destination is located. The low-order bits identify the specific host destination on a medium.

A Telephone Analogy

Think of a telephone number, considering all fields that may be used with a telephone number. The highest-order digits of a telephone number form a country code. In North America, the next three digits define the area code, the next three digits specify an exchange, and the final four digits define

the actual line within an exchange, to which a telephone is connected. Figure 7. Telephone Hierarchy shows the topological decisions made as a result of the structure of a telephone number.

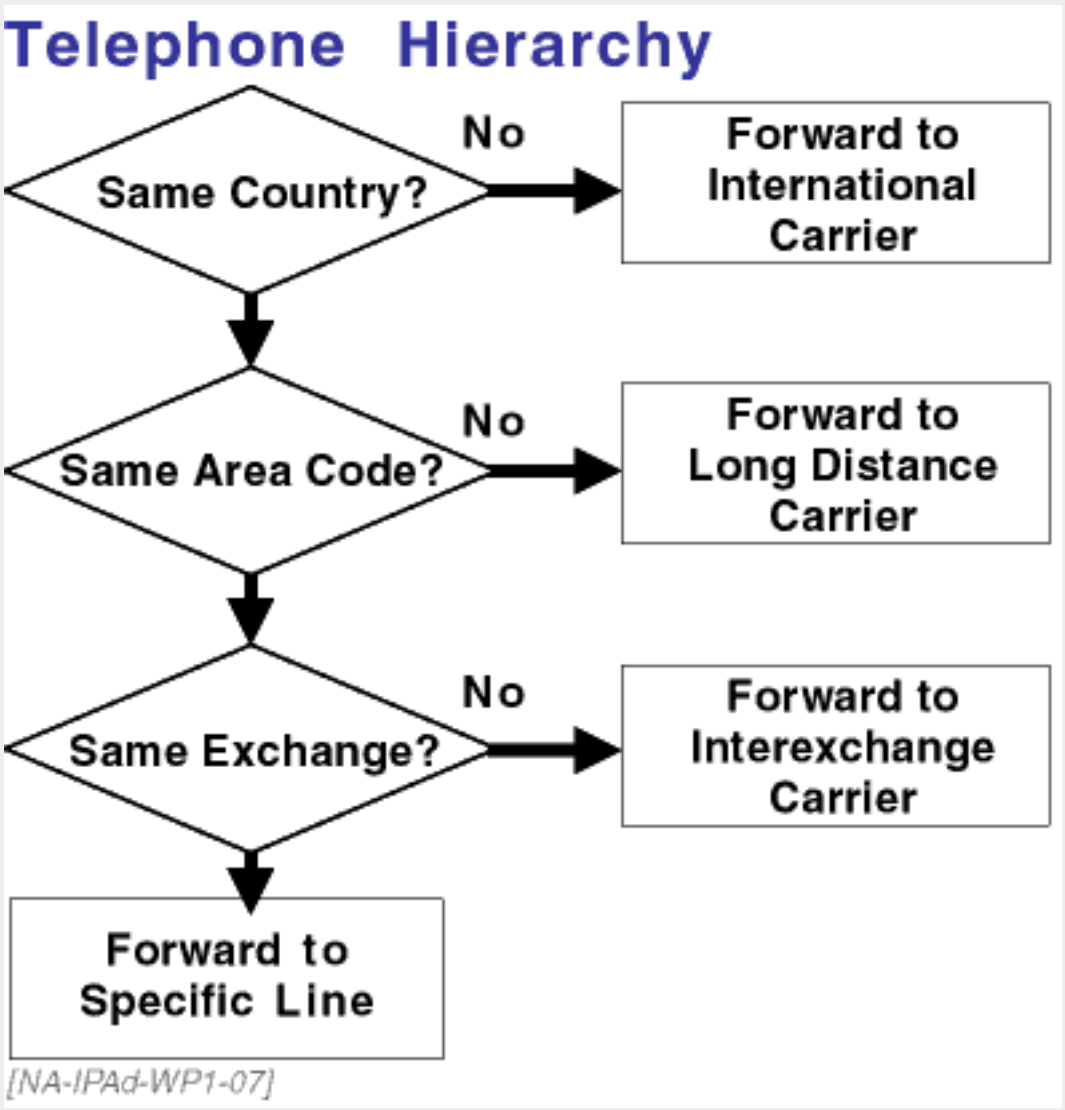


Figure 7. Telephone Hierarchy

Your local telephone switch first considers the country code prefix if a country code is present. It compares the country code to its own country, and, if the two codes do not match, it sends the call to an international switch. The local switch does not evaluate fields below the country code when processing international calls.

If the call is in the same country, the switch then evaluates the area code prefix digits. When the area code does not match the area code of the switch, the call is transferred to a long-distance switch. The local switch, on finding that the destination of a call is not in the same area code, does not evaluate fields below the area code level. It merely considers the three-digit area code prefix.

When the call is in the same area code, the switch then compares the exchange prefix digits with its own exchange. If these digits do not match, the switch passes the call to a switch that services that exchange. To find that exchange, it considers six digits, the exchange within an area code.

Only if the exchange codes match does the switch look at the line field and complete the call to the actual destination. In completing that call, it maps the software-defined line number, at the logical level, to a physical pair of wires that leads to a telephone.

Network Addresses and Routing

You can think of network addresses as having two basic parts shown in Figure 8. Prefix and Host. The prefix tells the router how to get closer to the ultimate destination medium. The host part tells the final router interface how to reach the specific destination on the final medium. In modern IP practice, prefixes can be of different lengths.

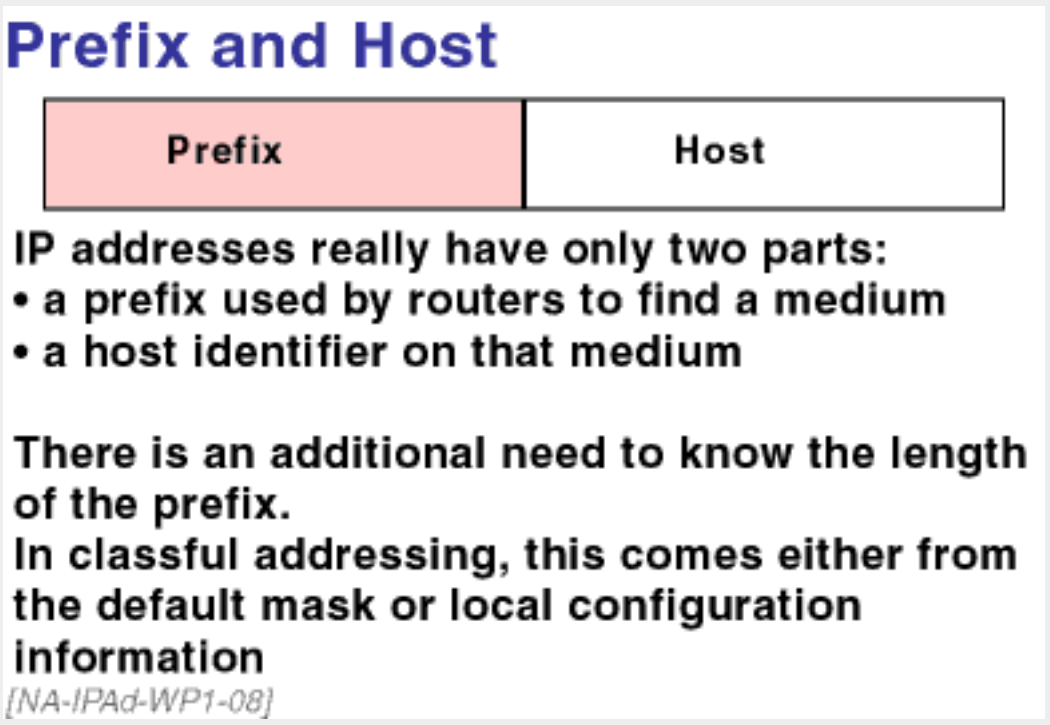


Figure 8. Prefix and Host

The prefix may have multiple internal levels, much as a telephone number can. In the approach to addressing that you will use at the CCNA level, the prefix will be divided into two major parts: the network part and the subnet part. We will discuss these parts in detail later, but, at this point, assume that the network part is administratively defined by a central authority, while the subnet part is an extension to the network part, and is defined by enterprise-level network administrators. Figure 9. Classful Routing parallels the telephone number analogy of Figure 7. Telephone Hierarchy.

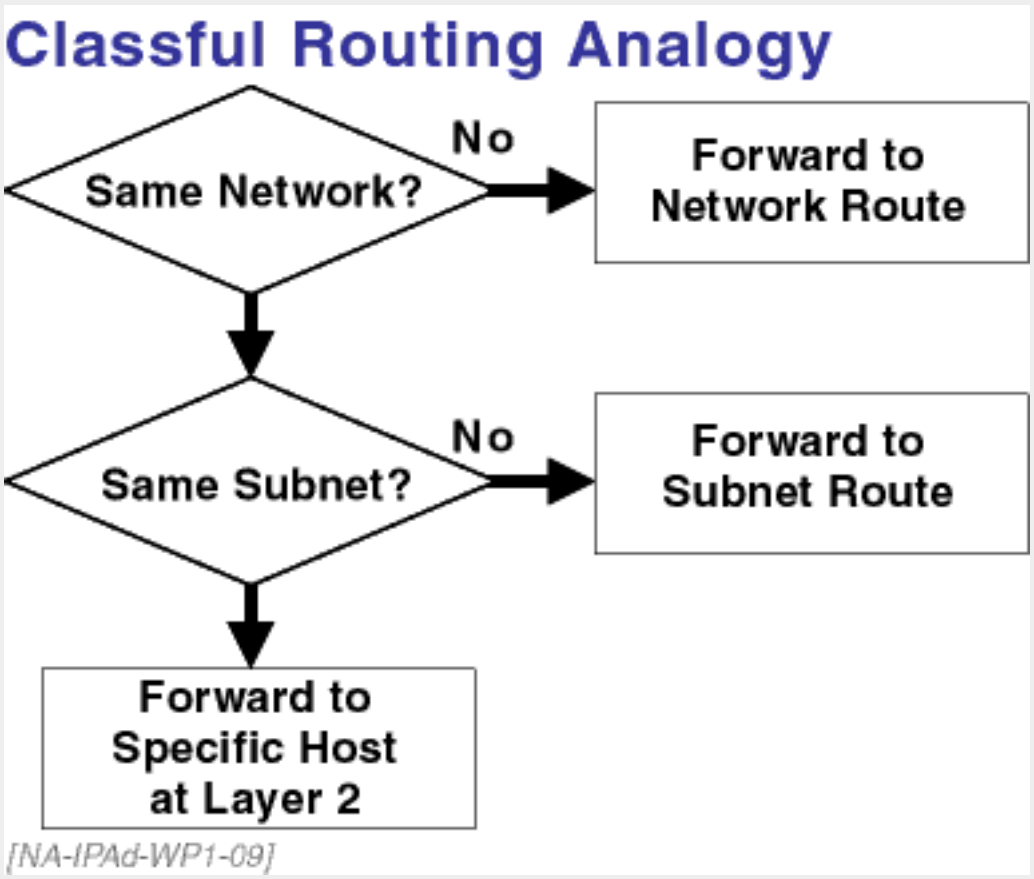


Figure 9. Classful Routing

Matching Routes

If an existing entry in a router's routing table matches a new route, but is less specific, the just-received route is added. "Less specific" means that the route in the Routing Information Base (RIB) matches the destination with a lesser number of prefix bits than does the new route. Another way of putting this is that a more specific route has a subnet mask with more one bits: 255.255.0.0 is more specific than 255.0.0.0.

For example, assume your routing table contains:

```
10.0.0.0/8 (mask 255.0.0.0), outgoing interface S0
```

and the router receives

```
10.1.0.0/16 (mask 255.255.0.0), outgoing interface e0
```

The new routing table will contain:

```
10.0.0.0/8     s0
10.1.0.0/16    e0
```

When routing a packet, routers use the *longest match* in their routing table to select the outgoing interface. In the routing table example above, 10.1.0.0 is more specific than 10.0.0.0, so traffic to 10.1.0.0 will exit on Ethernet 0.

One important special case is the *default route.*

A Special Case: The Default Route and ip classless

By convention, the address 0.0.0.0/0 is the default route, the least specific possible route. Cisco sometimes uses the term *pseudonetwork* to refer to 0.0.0.0/0. It is the route that you go to when you don't have anyplace else to go. When it came time to pick softball teams in my high school physical education classes, I was the default route.

As opposed to being something to put in right field and forget, default routes are quite useful in networking. Default routes can be declared with static routes, or they can be learned from dynamic routing protocols. While static routes are more a technique for the CCNP than the CCNA level, here's a quick example. To create a static route to define the local default, code:

```
ip route 0.0.0.0 0.0.0.0 {next_hop_IP | outgoing_interface}
```

Created as a static route with an administrative distance less than dynamic routing, a default route in the next-hop-ip format will be used for the local router box, but not advertised unless it is explicitly redistributed (or you use the outgoing interface form of the static route command). Statically declared default routes of the interface-name format will be advertised as if they were directly connected.

Local configuration is not the only way your router can learn the 0.0.0.0/0 default route. It can be learned from dynamic routing protocols such as OSPF and RIP. See a more detailed discussion in the CCIE Routing Principles Tutorial.

ip classless

While default routes are not a major issue at the CCNA level, you should be aware of a change in behavior of more recent releases that have the **ip classless** option.

In a routing table under the **no ip classless** option, where the router has learned about 10.1.0.0/16 from another router, it assumes that if a subnet of 10.0.0.0 is reachable through e0, so is any other part of 10.0.0.0.

```
10.1.0.0/16          e0
0.0.0.0/0            s1
```

Where will a packet destined for 10.2.0.0 go? With **no ip classless,** it should leave on Ethernet port 0.

When the **ip classless** option is coded, the router does not assume that if it knows how to reach a subnet, it can use that subnet's output interface to reach any other subnet of the major network. With this option, a packet destined to 10.2.0.0 will not match the more specific subnet entry, and will leave on Serial 1.

```
10.1.0.0/16       e0
0.0.0.0/0         s1
```

You may run across several terms that are often (and incorrectly), considered synonymous: default routes, default gateways (default routers), default networks, and gateways of last resort. These terms refer to slightly different mechanisms, all of which are useful. Knowledge of them is generally required at the CCNP level.

Default Gateway

The default gateway is specifically intended for the situation when no IP routing is enabled. It has the specific next hop address of the gateway router.

You would use this on a switch, or a router box that is only doing bridging, so the box can reach network management servers not on the same subnet. Another application for the default gateway comes during booting from ROM, to find the TFTP server.

In the IOS, you configure an IP default gateway with the command

```
ip default-gateway gateway-address
```

where *gateway-address* is the address of a router interface on a subnet to which your router is physically connected.

Default Network

The default network, used by IGRP and EIGRP, has only a prefix -- a network or subnet -- so unless internal assumptions are made, there's no way to know the specific next hop address. To specify a default network for IGRP, for EIGRP, or that will be known locally on your router, code:

```
ip default-network ip-prefix
```

The *ip-prefix* is **not** a host address as used in the next hop field of an **ip route** statement, or as the argument of **ip default-gateway**. It is a network or subnet address (i.e., with zeroes in all the host bit positions).

Gateway of Last Resort

The *gateway of last resort (GOLR)* is selected by the process that actually installs routes in the routing table. The GOLR represents the default destination that comes from the source of default that has the lowest administrative distance (AD).

So if you had a default static route, it would become the GOLR regardless of anything you received from any routing protocol. If you received a default network from EIGRP or IGRP, that network would become GOLR in preference to anything from RIP or OSPF, unless you changed the administrative distance for RIP or OSPF. An OSPF default would be preferred to anything from RIP. An OSPF Type 1 default would be preferred over an OSPF Type 2 default.

IPv4 Evolution

The original IP specification, RFC760, did not use classes. The network number was defined to be the first octet. That early "network number" was a prefix. Prefixes are the key to understanding how addressing and routing really work. Routers really don't care about hosts, but care about prefixes only. Prefixes identify blocks of potential host numbers, and it is to prefixes that routers route.

Any IPv4 address is 32 bits long. Routers make decisions based on some number of contiguous bits of this address, starting with the most significant bit on the left. This part of the IP address is called the prefix.

The three customary IP address classes each define a prefix of a certain length. Ignoring subnetting for the moment, a Class A address has a prefix length of 8, a Class B has a prefix length of 16, and a Class C has a prefix length of 24.

Early IP implementations, such as that in BSD 4.2 UNIX, stored no prefix information. Instead, they inferred a prefix length from the class of the address. Later implementations did store a specific subnet mask and thus supported subnetting but still associated one mask value with every address in a specific classful network number.

A useful convention in interpreting addresses comes from my book, *Designing Addressing Architectures for Routing and Switching*:

- **P** is a prefix bit, used in making routing decisions by the router we are talking about. In classful addressing, the type you will deal with as a CCNA candidate, the prefix bits are composed of the combination of network and subnet bits.
- **S** is a *subnet* or sub-prefix bit that identifies a specific medium or group of media within a larger prefix. In the North American telephone hierarchy, area codes are a subdivision of the country code, and in turn have subdivisions called exchange codes.
- **X** is a "don't care" bit from the perspective of routing at the point of topology we are examining.
- **H** is a host bit used to locate a specific host on a medium, or to indicate the medium itself (rather than prefix for it) or the broadcast address for that medium.

Think of a postal address, which consists superficially of a street with a building number on the street. It is the job of routers to deliver packets to the final street (medium) on which a destination host -- the building -- will recognize packets. Assume we route based on a single-bit prefix. The notation convention for this length is a /1 prefix. This notation convention was introduced with the current practice for global Internet addressing, Classless Inter-Domain Routing (CIDR). See Chapter 6 of [Berkowitz 1999a] for a discussion of the motivations for CIDR in scaling the Internet. At this point, simply accept that the prefix length notation introduced with CIDR is very useful. I find that the older notation, subnet masks, is harder for beginners to grasp. We will cover subnet masks once the underlying principles are clear.

Just because you can do something, it isn't necessarily a good idea.

Formally, according to RFCs 1517 and 1518, a prefix is "an IP address and some indication of the leftmost contiguous significant bits within this address." That indication of the leftmost contiguous part - the prefix - has conventionally been done with subnet masks (e.g., 10.1.0.0 with mask 255.255.0.0, the one bits of the mask corresponding to the prefix positions in the address), or more recently with a length indication (e.g., 10.1.0.0/16, the 16 indicating the prefix length). All RFC760-style prefixes, therefore, could have been written as N.0.0.0/8.

One bits in the mask must be contiguous from the left. In RFC1812, the current "Requirements for IPv4 Routers" document, patterns such as 255.0.255.0 are now specifically illegal, although earlier specifications were vague on this point [RFC1812].

Some hosts and older router implementations will let you enter noncontiguous masks. Don't be tempted, because addresses based on noncontiguous masks are both hard to maintain and will break addressing features (e.g., VLSM summarization) beyond the scope of this Tutorial.

Routers and Prefixes

In a router, if the value of the prefix is 0, the packet will leave the router via interface 0. See Figure 10. One Bit Address. If the value of the prefix is 1, the packet will leave the router via interface 1. In other words, only the first bit of these IP addresses is considered in making routing decisions.

The path taken in Figure 10 will be:

```
0    R1.int0, R1.1.int0
1    R1.int0, R1.1.int1
```

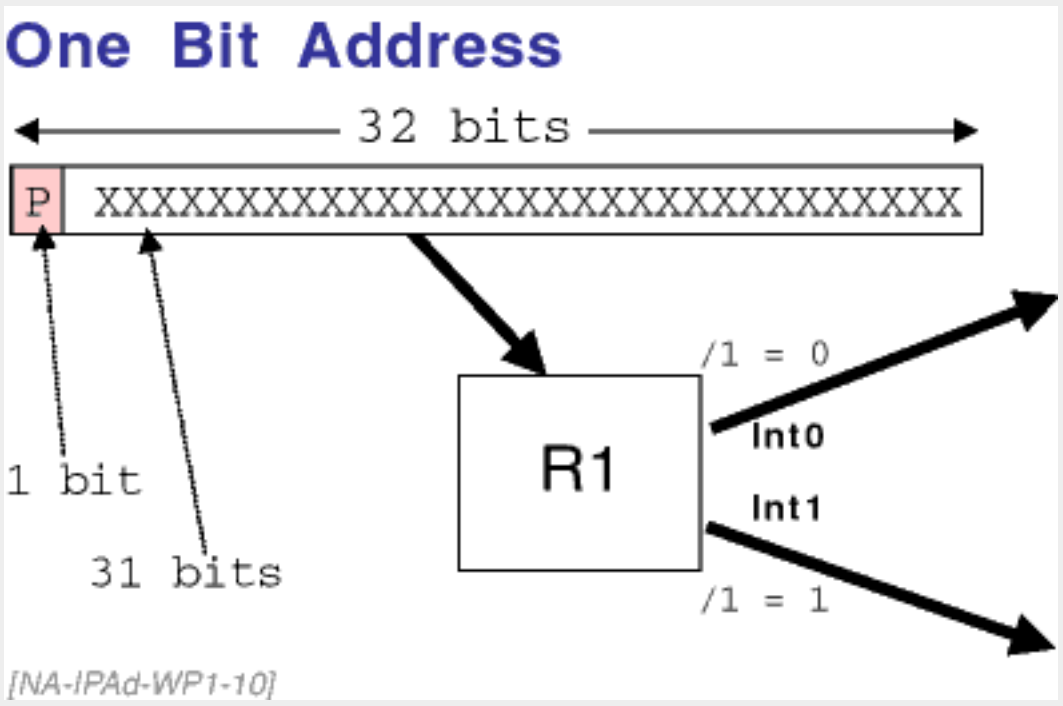


Figure 10. One Bit Address

A single-bit prefix gives us only two possible values. Staying with an essentially trivial example, assume that we have a two-bit prefix and four possible destinations. Each of these destinations is identified by a value of the /2 prefix. These destinations could be reached with four interfaces on one router, or with a tree of three routers.

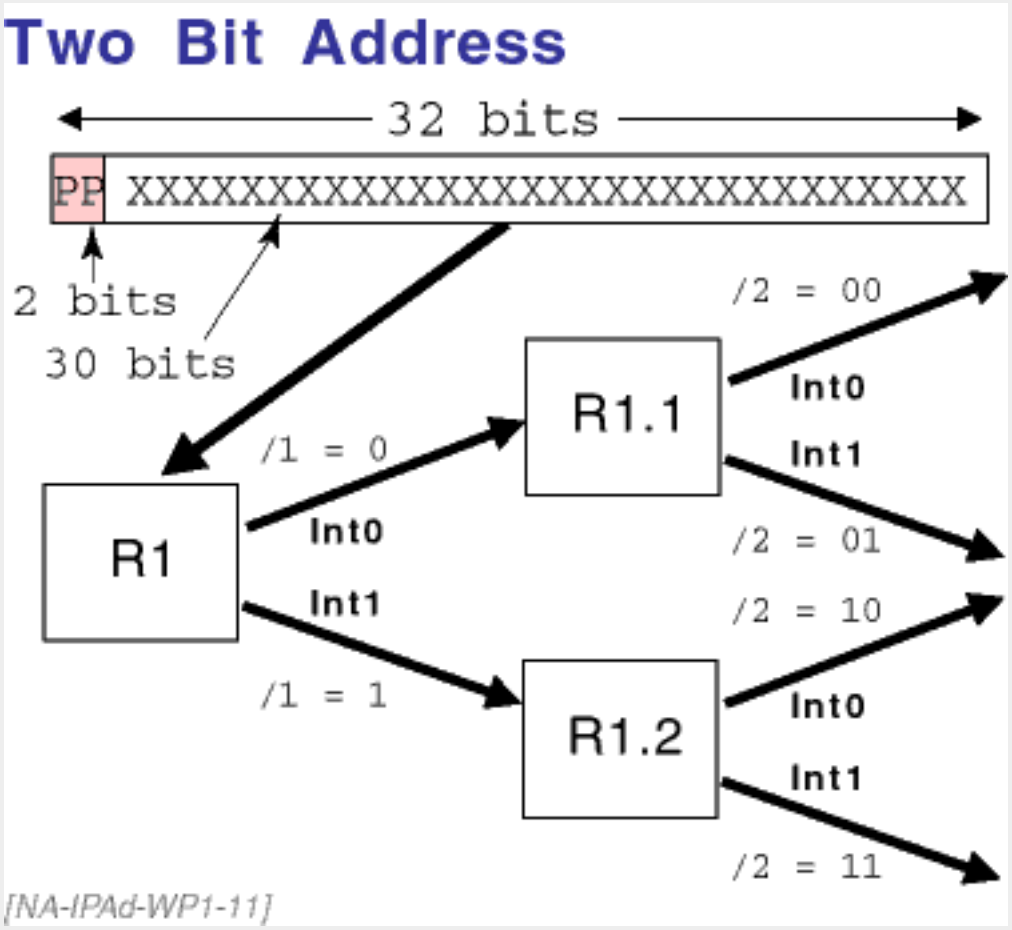


Figure 11. Two Bit Address

Real-world IP addresses have at least 8 bits in their prefixes, and usually many more. Practical routers associate many prefixes with each outgoing interface. In Figure 11, an X in one of the first 8 bits represents a don't-care bit position in making forwarding decisions for this particular routing scenario.

Another way to look at the don't-care mechanism is to remember that an IP address is always 32 bits long. If the prefix length (e.g., /8) is subtracted from 32, the result is the number of don't-care bits for router decision making. In other words, decisions are made only on the prefix length number of bits.

The information shown in Figure 11 is stored internally in a router as a routing table, also called *a Forwarding Information Base (FIB)* or *Routing Information Base (RIB).*

A minimal routing table contains a list of destinations and the output interface that should be used to reach them

The path taken in Figure 11 will be:

```
00    R1.int0, R1.1.int0
01    R1.int0, R1.1.int1
10    R1.int1, R1.2.int0
11    R1.int1, R1.2.int1
```

In these examples, the part of the IP address not included in the prefix bits used for decision making is composed of bits that identify the host on the destination medium, or of bits that will be used for path determination hierarchically lower in the routing fabric.

The very first method of assigning prefixes, which was obsolete almost as soon as it was defined, was to define all prefixes as a fixed 8-bit length, as shown in Figure 12. Fixed Prefix

Figure 12. Fixed Prefix

This fixed prefix length meant that the remaining 24 bits could be used for host addresses. It was assumed that all host addresses in a given prefix were managed by a central computer such as a mainframe, or were on the same LANs. In 1981, LANs were still primarily a research curiosity, as was the newly introduced personal computer.

Unfortunately, interconnection requirements grew quickly, and there were soon more than 200 networks. The first enhancement to the method of assigning prefixes was classful addressing. In 1981, a new convention, RFC 791, was developed to have three standard prefix lengths of 8, 16, and 24 bits, shown in Figure 13.

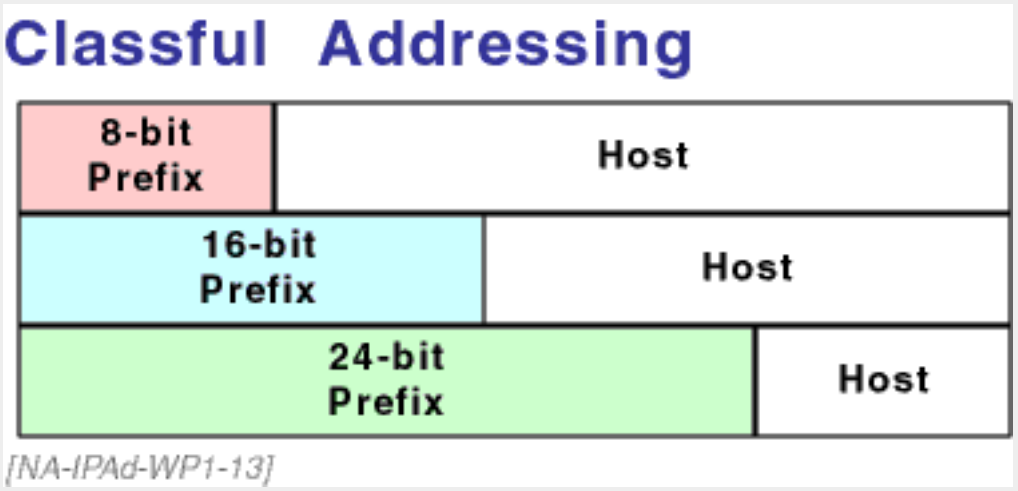


Figure 13. Classful Addressing

Classes

RFC 791 controlled the values of the most significant bits (i.e., leftmost) in the prefix to determine the prefix length. These bits were overloaded in that they were part of the address but also encoded how long the prefix was. The encodings are:

(Memorize this table!)

Pay careful attention to this, because it will affect terminology later on: The original IP address prefix was *fixed*. With the introduction of different prefix lengths, it was no longer fixed, but *variable*. Variable-length prefixes have been with us since almost the beginning of IP.

Table 3. Class Prefixes

Address	Class	First Octet Range in Dotted Decimal	CIDR/VLSM */bitcount* notation	Purpose
0xxx	A	1-126	/8	Unicast
10xx	B	128-191	/16	Unicast
110x	C	192-223	/24	Unicast
1110	D	224-249	Not applicable	Multicast
1111	E	240-255	Not applicable	Experimental

IPv4 addresses were (and are) 32 bits long. The three standard unicast prefix lengths meant, respectively, there could be host fields 24, 16, and 8 bits long.

Getting Address Space

In the original model for assigning IP addresses, the high-order part of the prefix -- the /8, /16, or /24 -- was assigned by a single administrative body. This high-order part was called a network number.

Organizations assigned network numbers then would define subnets under them. When we speak of "bits of subnetting," we mean the number of bits to the right that the prefix is extended.

Without getting into details of the administrative processes involved, IP addresses are either globally unique (registered), or they belong to the private address space defined in RFC 1918.

Three regional registries allocate large blocks of address space:

- American Registry for Internet Numbers (ARIN), serving the Americas and some other locations that do not yet have a regional registry: www.arin.net
- RIPE Network Coordination Centre (NCC) for Europe: www.ripe.net
- Asia-Pacific Network Information Center (APNIC): www.apnic.net

Provider Assigned Address Space

In practice, most enterprises will receive a part of their upstream provider's registered address space to be used for as long as they are a subscriber of that provider. This is called *provider assigned (PA)* address space, as opposed to *provider independent (PI)* address space allocated directly by one of the registries. In general, an organization needs to demonstrate it will have 8000 or more Internet-connected hosts before it becomes eligible for PI space.

The reality that most organizations will use PA space means that when you design networks, you should assume that they will be periodically renumbered, for example, if you change providers.

Private Addresses

Three blocks of addresses are reserved for "private use." Private use means that these addresses should never be seen on the public Internet [RFC

1918]. These blocks are normally described in dotted decimal:

- **10.0.0.0/8**, the "8-bit block" that contains the range of addresses 10.0.0.0 to 10.255.255.255
- **172.16.0.0/12**, the "12-bit block" that contains the range of addresses 172.16.0.0 to 172.31.255.255
- **192.168.0.0/16**, the "16-bit block" that contains the range of addresses 192.168.0.0 to 192.168.255.255.

IP Addresses: Computer Views, Human Views

The original IPv4 specification (RFC 760) was issued, with the intention of both being compatible with existing ARPANET addresses and providing growth for the future. Growth, in this context, meant the ability to interconnect over 200 networks.

Before discussing the structure of addresses, it is worthwhile to discuss the ways we talk about addresses. You, as a person, are unique. You, however, are addressed differently at different times by different people, in a manner appropriate to the context. Someone might be addressed as "William," "Big Bill," or "Stinky" in different addressing contexts.

In like manner, there are different ways to "say" the meaning of an address. The abstract semantics of an address deal with the meaning of address (e.g., its membership in a specific hierarchy). The semantics of an IP address indicate the way it is reached in an IP routing system.

Abstract syntaxes deal with human-readable notation for addresses. For IP addresses, this is the "dotted decimal" format.

Encodings are machine-readable forms of the address used in protocol data units. IP is encoded as a 32-bit string.

Dotted-Decimal Notation is for People, not Routers

Different architectures use these different ways according to their individual rules. All use some form of hierarchy to make addresses. Hierarchical organization may be intended simply to organize network administration, or additionally to maintain worldwide address uniqueness.

IPv4 proper is encoded as a 32-bit binary string. Its abstract syntax, however, is called dotted decimal. Dotted decimal, unfortunately, is one of those things that seemed a good idea at the time, but definitely tends to confuse the situation. Its use is so widespread that it's impractical to phase it out directly. IP version 6 uses hexadecimal, a much more rational notation. Admittedly, dotted decimal **is** easier to remember than binary.

Principles of Dotted Decimal

Even though the actual IP address is a 32-bit string, with prefix and host field sizes of arbitrary lengths, the dotted decimal convention splits the address into four 8-bit octets such as 160.65.2.66. These octets only have meaning in terms of human convenience and the administrative process of address assignment. The octets really are not seen separately by the routing process.

If we begin with the binary string

```
10100000010000010000001001000010
```

we can split it into four octets

```
10100000 01000001 00000010 01000010
```

• Binary **10100000** has a decimal value of 160

• binary **01000001** has a decimal value of 65

• binary **00000010** has a decimal value of 2

• binary **01000010** has a decimal value of 66

These four eight-bit values are written out as their decimal equivalents, separated by dots:

```
160.65.2.66
```

Historically, the class-based assignments of network numbers were done on an octet-aligned basis. This is obsolete. Most confusion about IP addressing comes from (incorrectly) implying meaning to the octets.

Here's another example, using the private address space. In binary, we can expand the 8-bit 10.0.0.0 block:

```
00001010 XXXXXXXX XXXXXXXX XXXXXXXX
```

or the range of values

```
00001010.00000000.00000000.00000000
```

```
   to
00001010.11111111.11111111.11111111
```

These binaries translate to the dotted decimal range 10.0.0.0 through 10.255.255.255.

Weighted Binary

There are some non-obvious conventions in converting between dotted decimal and binary. Look at the last octet in this example, 01000010, the decimal equivalent of which is 66. Let's assume that, for some reason, we need to split it into two 4-bit fields.

If you are a reasonably rational human being, proficient in binary and decimal arithmetic, you will come to the apparently reasonable conclusions that the two fields would be 0100 and 0010, which, respectively, would have the decimal values 6 and 2.

This would be perfectly rational, and it also would be wrong for the first field. Dotted decimal notation uses what is called the weighted binary convention. When a field is extracted from an octet, its bits must be evaluated in the same position, relative to the most significant bit on the left, in which the field started.

So in this case, the leftmost 4 bits must be evaluated as if they were:

```
01000000
```

and the right 4-bit field must be evaluated as if it were:

```
00000010
```

The proper value for these fields, if they were to be expressed as part of a dotted decimal expression, would be, respectively, 64 and 2.

Subnetting versus Subnet Masks

The limited range of three prefix lengths still proved inadequate, and subnetting was introduced as a means of providing more prefix lengths. Subnetting is the general process of extending a prefix to the right. With classful addressing, subnetting means you are borrowing host bits to extend the prefix. This creates more prefix space. In practice, the original prefix is known at a high level of the hierarchy, and the extended prefixes are usually known only in the lower parts of the hierarchy.

"Traditional" subnetting (See Figure 14. Traditional Subnetting) introduced several terms that tend to be quite confusing. If one looks at the process introduced in RFC 950 not as a subdividing -- subnetting -- of existing networks, but more correctly as a general mechanism of prefix extension, the terminology becomes much simpler.

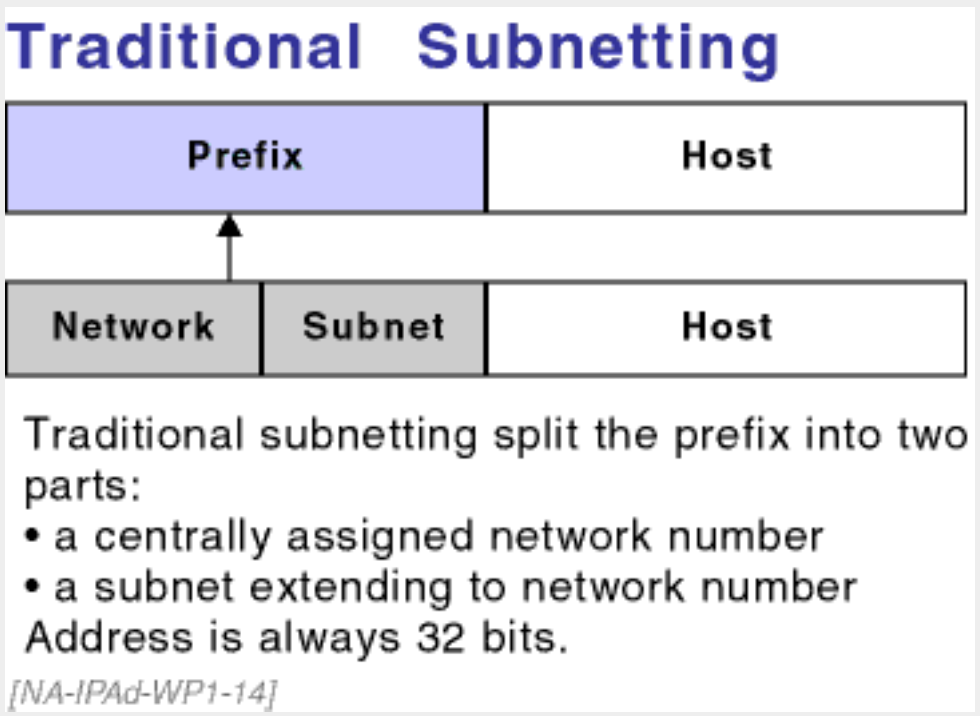


Figure 14. Traditional Subnetting

The additional, yet confusing, terms are:

- The process of subnetting: the general process of extending the prefix to the right
- Bits of subnetting: the number of bits to the right that a classful network prefix is extended.
- Subnet masks: one means of conveying the total length of the prefix.

In the classic RFC 950 method, subnetting is the process of further subdividing an assigned network number into a set of user "streets." It is a specific form of prefix addressing, based on "classful" addressing, where addressing authorities assign network numbers to organizations, and the user organization extends the routing-relevant part by adding bits from the user field. In a classful system, the original allocation will be /8, /16, or /24.

The number of bits of subnetting (m) is the number of bits the prefix is extended by the network administrator, from the prefix assigned by higher authority. When n is the number of basic prefix bits, m =< (30 - n). The basic prefix (n) is assigned by a higher-level administrative authority and given

to a network administrator.

Subnet masks are really used in two ways:

- Whether you use CIDR or not, the most efficient way to extract prefixes is to build a 32-bit string that is binary ANDed to the address, the result being the network prefix.
- Subnet masks are also a way of telling people or routers what prefix length should be used at a given interface. Either the /slash notation or dotted decimal masks can serve this purpose.

Each traditional class has a "natural" or "default" mask that can be inferred from the value of the first few bits. See Table 4.

Table 4. Natural/default Masks

High-order bits	Class	First Octet Range in Dotted Decimal	Natural or default mask
0xxx	A	1-126	255.0.0.0
10xx	B	128-191	255.255.0.0
110x	C	192-223	255.255.255.0

Confusingly, it is said that IP addresses are always associated with a subnet mask, even if the network associated with that address is not *subnetted* -- that is, extended from a preassigned, hierarchically higher prefix. I far prefer to use the term *prefix length*, because every IP address has a prefix length, whether subnetted or not.

Prefix Length Display Formats

Subnet masks are one technique of prefix length notation. The slash or CIDR "/*bitcount*" format is superior as a notation, but is not as widely deployed in software configuration tools.

As of IOS 11.0, most show commands default to using the bitcount, not the subnet mask, convention for showing prefix lengths. You can change back and forth between the subnet mask and "slash" notations by entering the command:

```
terminal ip netmask decimal
```

and change back with

```
terminal ip netmask bitcount
```

Again, remember these are not configuration editor commands, but entered while in the general exec.

Extracting Prefixes from Addresses

The binary value of a subnet mask, as opposed to the use of subnet masks for a prefix length notation, is the basis of extracting prefixes from the destination address fields of packets to be forwarded.

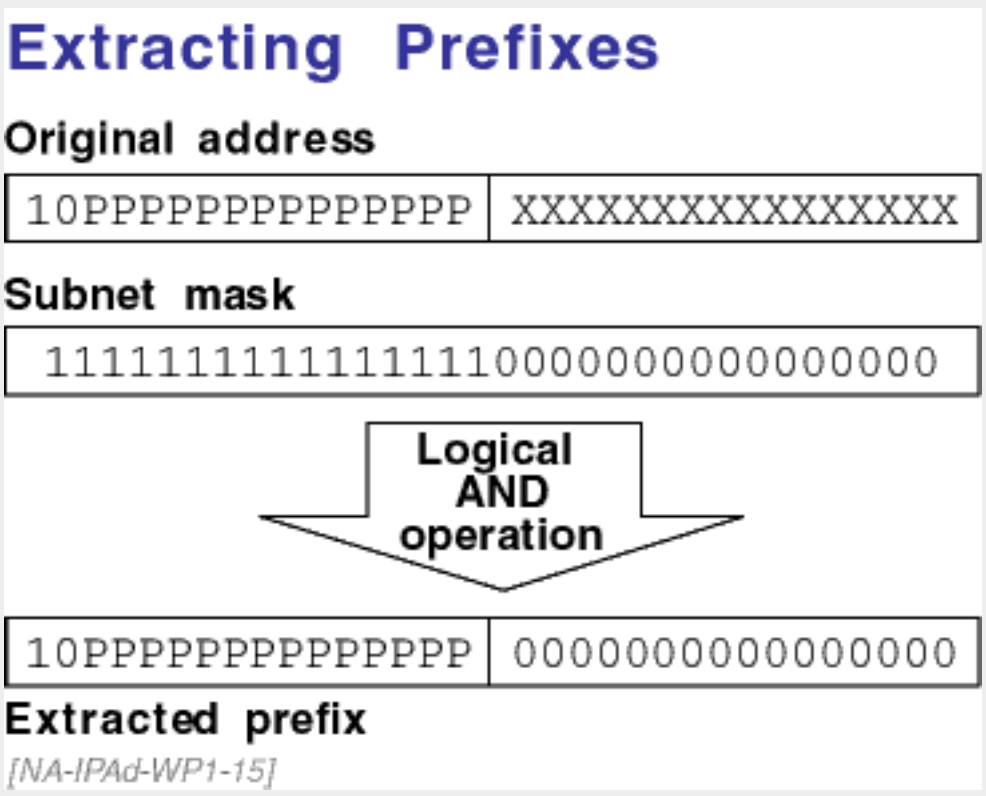


Figure 15. Extracting Prefixes

You extract the prefix from an IP address by a bit-by-bit logical AND operation between the 32 bits of the IP address and the 32 bits of the subnet mask. Essentially, the subnet mask is a bit pattern that will zero out the host field of an IP address. The number of one bits in the mask is the length of the prefix.

Logically ANDing the two binary strings has the effect of zeroing out all bits in the host field of the address, producing the prefix. When a router looks at a specific host destination address in an incoming packet, it uses this logical operation to extract the prefix. The router finds destinations in the

routing table not with specific host addresses, but with prefix values.

Using the prefix value, rather than the specific host value, is one of the fundamental strengths of routing. If specific host values were used, the router would have to track each address in the network. By using prefixes, the router only needs to track the much smaller number of prefixes, which are associated with destination media.

Relationships between subnet masks and prefix lengths are shown, in a classless way, in Table 6, derived from RFC 1878.

Table 6. Masks and Prefixes

Expanded Mask Value	Prefix Length	Traditional Subnet Mask Length	Host or Don't Care Bits	Hosts (-2 reserved)	Classful Equiv.
10000000000000000000000000000000	/1	128.0.0.0	31	2048M	128A
11000000000000000000000000000000	/2	192.0.0.0	30	1024M	64A
11100000000000000000000000000000	/3	224.0.0.0	29	512M	32A
11110000000000000000000000000000	/4	240.0.0.0	28	256M	16A
11111000000000000000000000000000	/5	248.0.0.0	27	128M	8A
11111100000000000000000000000000	/6	252.0.0.0	26	64M	4A
11111110000000000000000000000000	/7	254.0.0.0	25	32M	2A
11111111000000000000000000000000	/8	255.0.0.0	24	16M	1A
11111111100000000000000000000000	/9	255.128.0.0	23	8M	128B
11111111110000000000000000000000	/10	255.192.0.0	22	4M	64B
11111111111000000000000000000000	/12	255.240.0.0	20	1024K	16B
11111111111100000000000000000000	/13	255.248.0.0	19	512K	8B
11111111111110000000000000000000	/13	255.248.0.0	19	512K	8B
11111111111111000000000000000000	/14	255.252.0.0	18	256K	4B
11111111111111100000000000000000	/15	255.254.0.0	17	128K	2B
11111111111111110000000000000000	/16	255.255.0.0	16	64K	1B
11111111111111111000000000000000	/17	255.255.128.0	15	32K	128C
11111111111111111100000000000000	/18	255.255.192.0	14	16K	64C
11111111111111111110000000000000	/19	255.255.224.0	13	8K	32C
11111111111111111111000000000000	/20	255.255.240.0	12	4K	16C
11111111111111111111100000000000	/21	255.255.248.0	11	2K	8C
11111111111111111111110000000000	/22	255.255.252.0	10	1K	4C
11111111111111111111111000000000	/23	255.255.254.0	9	512	2C
11111111111111111111111100000000	/24	255.255.255.0	8	256	1C
11111111111111111111111110000000	/25	255.255.255.128	7	128	1/2C
11111111111111111111111111000000	/26	255.255.255.192	6	64	1/4C
11111111111111111111111111100000	/27	255.255.255.224	5	32	1/8C
11111111111111111111111111110000	/28	255.255.255.240	4	16	1/16C
11111111111111111111111111111000	/29	255.255.255.248	3	8	1/32C
11111111111111111111111111111100	/30	255.255.255.252	2	4	1/64C
11111111111111111111111111111110	/31	255.255.255.254	1	2	1/128C
11111111111111111111111111111111	/32	255.255.255.255	0	1	1/256C

Table 6 above is fully general for classless addressing. Tables 7 and 8 are more specific for Class B and Class C addresses, using classful assumptions about the all-zeroes and all-ones subnets.

Reviewing the AND operation

When you AND two binary bits together, the result will be zero unless the value of both bits is 1. The truth table for binary AND is:

Table 5. Logical AND Operation

First term	Second term	
	0	**1**
0	0	0
1	0	1

Table 7. Class B (/16) Subnetting

Hosts	Subnets	Subnet Mask
2	32766	255.255.255.252
6	16382	255.255.255.248
14	8190	255.255.255.240
30	4094	255.255.255.224
62	2046	255.255.255.192
126	1022	255.255.255.128
254	510	255.255.255.0
510	254	255.255.254.0
1022	126	255.255.252.0
2046	62	255.255.248.0
4094	30	255.255.240.0
8190	14	255.255.224.0
16382	6	255.255.192.0
32766	2	255.255.128.0
64534	1 (not subnetted)	255.255.0.0

Note that a subset of the subnet masks forms the Class C table.

Table 8. Class C (/24) Subnetting

Hosts	Subnets	Subnet Mask
2	126	255.255.255.252
6	62	255.255.255.248
14	30	255.255.255.240
30	14	255.255.255.224
62	6	255.255.255.192
126	2	255.255.255.128
254	1 (not subnetted)	255.255.255.0

Reserved Host Field Values

When a host field of m bits is defined, 2^m-2 values are available for actual host addresses. In this context, a host can either be an ordinary end host or a router interface.

Two values are reserved and have special meaning. An all-zeroes value in the host field, sometimes called "this subnet," effectively identifies the medium itself as opposed to any host on it.

It is the all-zeroes value -- the "name of the wire" -- that is stored in routing tables as the destination address. Routers will forward packets to that medium to reach any host on it.

Packets sent to the all-ones host field address under a given prefix are broadcast onto the associated medium, assuming the medium is broadcast capable. This value is called the local broadcast address of 32 one bits, usually written as the dotted decimal value 255.255.255.255. All hosts receiving such a packet can "hear" and respond to it as if it were sent to them specifically.

When a packet has the local broadcast address as its destination address, it will be broadcast onto the medium on which it originates, but will not be propagated to other media by routers. Cisco provides a feature called **ip helper** that can forward local broadcasts when there is a good reason to do so, in a well-controlled manner.

A broadcast sent to a specific prefix is called a directed broadcast. Directed, as opposed to local, broadcasts are routable. Packets with a directed broadcast address flow through the network based on their prefix but are converted to a local broadcast when they reach the final destination medium.

While the CCNA exam will almost certainly ask about directed broadcast addresses, which, on any subnet, are the subnet prefix with all ones in the

host field, be very careful about using them in real networks.

Applications for distributed broadcasts, in modern networks, seem to be limited to specialized internal functions such as host initialization (e.g., with DHCP and DNS), network management, and possibly database mirroring. Mirroring can be done better with multicasting rather than broadcasting.

A very common and nasty, malicious, hacking attack called smurfing depends on directed broadcast to do its damage. As of IOS 12.0, consistent with IETF recommendations, Cisco changed the default behavior on all its interfaces to **no ip directed broadcast**.

You must explicitly enable directed broadcasts if you need them. There is never a good reason to receive a directed broadcast from a packet arriving from the general Internet. So one wise policy is to enable directed broadcasts only on interfaces where they are needed and couple their use to filters that deny any packet with a source outside your internal network. Such filtering should be backed up by filters on all your interfaces coming in from the Internet that also deny packets with source addresses associated with your internal networks.

Prefix Practice

Try extracting a prefix: given the Class A address, 10.169.100.20/13, with 5 bits of subnetting. You are also given the subnet mask of 255.248.0.0, which is equivalent to 5 bits of subnetting extended from the natural Class A mask. What is the prefix associated with this address?

To extract the prefix, write out the binary equivalent of 10.169.100.20 with the binary equivalent of the subnet mask immediately below it:

```
00001010.10101001.01100100.00010100
11111111.11111000.00000000.00000000
```

applying a logical AND results in

```
00001010.10101000.00000000.00000000
10       .   168   .    0   .    0
```

Let's put the 10.168.0.0 prefix we have just extracted in context:

Prefix 10.168.0.0

Host value 0.1.0.0 (address 10.169.0.0) -- Identifies the medium

Host values 0.1.0.1 (10.169.0.1) through 0.1.255.254 (10.169.255.254) -- Available for hosts

Host value 0.1.255.255 (10.169.255.255) -- Directed broadcast to this specific prefix

Setting Up Simple Address Plans

In this section we will begin by examining a single-site addressing system and then look at addressing for basic inter-site connectivity. This simple example does not deal with methods for reducing routing table size, which are discussed in [Berkowitz 1998a].

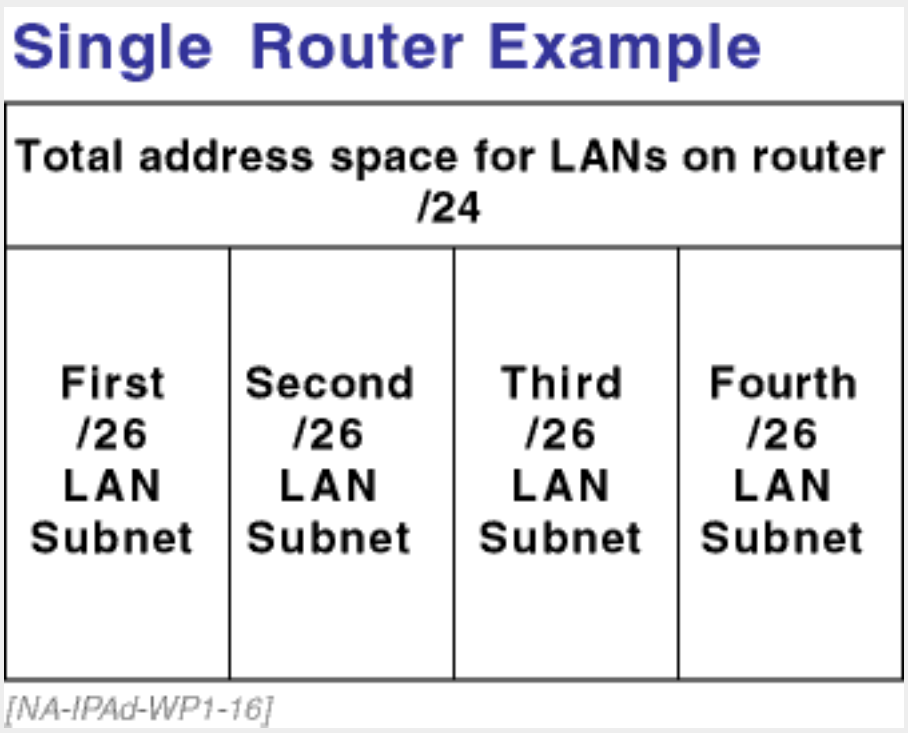


Figure 16. Single Router Example

Let's begin with the configuration shown in Figure 16. Single Router Example, a single router that interconnects four LANs. This router might interconnect four floors or workgroups inside a single building. Assume that there are 40 user workstations and two print servers in each LAN. Workstations and servers are both hosts in IP terminology, so at least 42 host addresses are needed.

Remember that a router usually does not have a single network address of its own, but is treated as a collection of interfaces, one on each directly connected medium. This is true of IP, and most protocol families, with the exceptions of DECnet and Banyan VINES. In our IP case, there will need to be a router interface on each medium, bringing the total host requirement to 43.

Consider reasonable growth. If we assumed that there might be 25% growth of user hosts, we would need space for an additional 10 hosts, for a total of 53.

What kind of network? What Mask?

You may be told, as part of a problem, whether to use a Class B or Class C network as your starting point. In either case, or if the decision is yours, you will need to decide first on the size of the host field. There are several things you should consider in this decision:

- Two addresses are not available for hosts: the all-zeroes and all-ones values. Think of the all-zeroes as the identifier for the medium itself, and the all-ones as the broadcast to everyone on that medium but nowhere else -- the "local fire alarm."
- If any traffic will leave the net, you will need at least one address for a router, unless any application hosts can route.
- While you may or may not have growth as part of a CCNA question, it's usually prudent to allow for growth in any addressing plan. While every enterprise will be different, a rough guideline is to allow 20-25% growth. If your organization is applying for its own allocation of address space, you will need to document your assumptions about growth.

In many real-world cases, you would receive several Class C blocks before you would receive a Class B, but effective use of multiple Class C's treated as a unit is beyond the scope of CCNA.

Look at Table 8 to find the number of host field bits that will contain this number of hosts. Table 6 contains the same information in a more general, classless way.

Either table will tell you that a six-bit host field will suffice. You have justified a prefix length of /26 to locate this specific medium. In traditional subnet mask notation, this prefix length has a subnet mask of 255.255.255.192. This equates to two bits of subnetting on a Class C, which has a natural mask of 255.255.255.0 or a /24 prefix.

When using traditional classful addressing, which assigns addresses only on class A, B, or C boundaries, you would have to round your requirement for a /26 up to a class C /24 block. The more modern CIDR convention assigns not just on "classful" boundaries, but on the boundary justified by your addressing requirements.

Addressing Simple Interconnected LANs

You have four LANs connected to the router, so you will need four media prefixes, each a /26. You will need two bits to identify four prefixes. Together, the host field and the medium identification bits take up eight bits. The 32-bit IP address, less these eight bits, justifies a /24 prefix.

In traditional terms, this /24 is a Class C block with two bits of subnetting. A better way to think of it, however, is that it is a /24 block containing four contiguous /26 blocks (i.e., subnets). We can say these four subnets summarize into the /24.

For the prefix 192.168.64.0/24, address assignments are shown in Table 9. Another way to describe this assignment is two bits of subnetting on a Class C.

Table 9. Address Assignments for /26 inside a /24

Binary Value of Address	Dotted decimal	Usage
00000000	192.168.64.0	Identifies the first subnet
00000001 through 00111110	192.168.64.1 through 192.168.64.62	First host on first subnet Last host on first subnet
01111111	192.168.64.63	Broadcast for first subnet
01000000	192.168.64.64	Identifies second subnet
01000001 through 01111110	192.168.64.65 through 192.168.64.126	First host on second subnet Last host on second subnet
01111111	192.168.64.127	Broadcast for second subnet
10000000	192.168.64.128	Identifies third subnet
10000001 through 10111110	192.168.64.129 through 192.168.64.190	First host on third subnet Last host on third subnet
10111111	192.168.64.191	Broadcast for third subnet
11000000	192.168.64.192	Identifies fourth subnet
11000001 through	192.168.64.193 through	First host on fourth subnet

11111110	192.168.64.254	Last host on fourth subnet
11111111	192.168.64.255	Broadcast for fourth subnet

Configuring IP Addresses into Cisco Routers

Actually configuring IP addresses into routers is not very complicated. The difficult part is deciding what addresses to use!

Configuration commands will become more complex when you are setting up filtering or dynamic routing, but basic interface addressing is not complicated.

Basic interface statments

The basic method to configure an IP address on a router is as a subcommand of an interface command:

```
no ip subnet-zero
interface e0
ip address 192.168.1.1 255.255.255.0
interface s0
ip address 192.168.0.1 255.255.255.252
```

You are configuring a host address on an interface, not a subnet address even though the interface belongs to a router. Note that you must configure both the address and a subnet mask, and write them in dotted decimal notation.

At the CCNA level, all interfaces belonging to the same classful network must have the same mask.

If you attempt to enter an address that, when masked, would have an all-zeroes or all-ones value in the subnet field, the router will return an error message, usually "bad mask." You will also get an error message if you try to code an all-zeroes or all-ones host field value.

In your testing, you will probably want to use loopback addresses so you have more interfaces on which you can practice. Cisco loopback interfaces exist only as a result of software definition, but you can configure and use them much like any physical interface. You can delete them, but not shut them down.

An example of adding interfaces to show connection to two subnets each on two major networks:

```
no ip subnet-zero
interface loop0
ip address 10.1.0.1 255.255.0.0
interface loop1
ip address 10.2.0.1 255.255.0.0
interface loop2
ip address 171.16.5.5 255.255.255.0
interface loop3
ip address 171.16.42.42 255.255.255.0
```

Secondary Addressing

Classful addressing is inefficient, because it tends to force users into receiving allocations that are too large (Class B), wasting space, or that are too small (Class C). Having multiple Class C blocks actually works fairly well, but even that doesn't always work because large switched networks often need 500 or more hosts in a single broadcast domain.

Until the much more flexible methods of classless addressing are much more widely accepted, workarounds are necessary, One technique widely used is to map more than one logical address (*secondary addresses*) to the same physical interface, as shown in Figure 17.

Secondary addresses are coded much like regular addresses, with the additional keyword **secondary**:

```
no ip subnet-zero
interface ethernet0
ip address 10.1.0.1 255.255.0.0
ip address 10.3.0.1 255.255.0.0 secondary
```

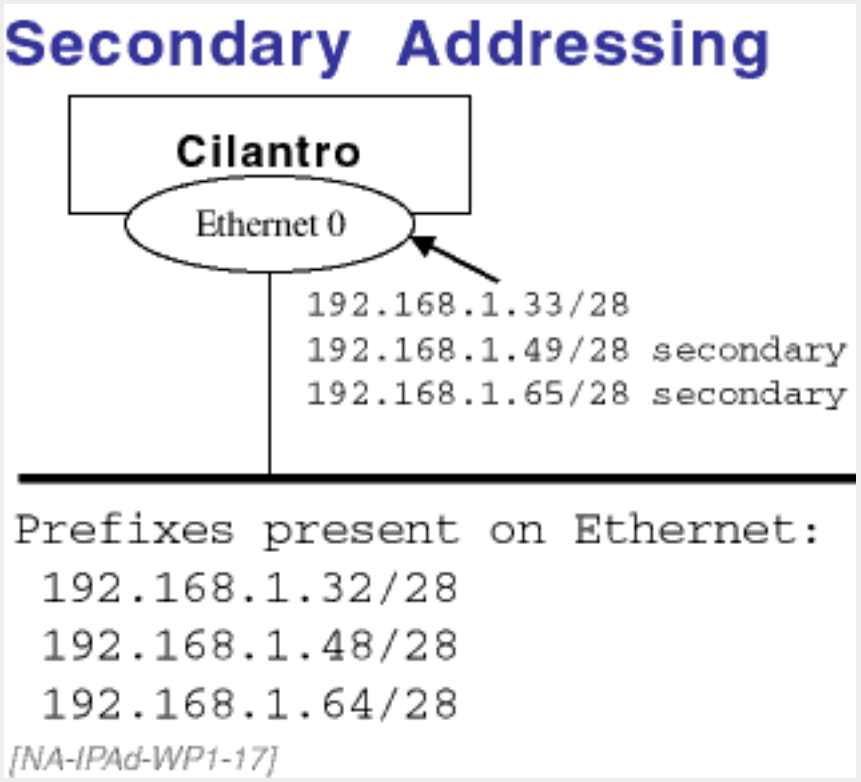
In Figure 17. Secondary Addressing, secondary addresses are being used to make more than 500 interfaces available on the switched subnet.

Stupid hosts and Classful Addressing

When you assign more than one subnet to the same medium, as in Figure 17. Secondary Addressing, you may have performance problems. The router knows perfectly well that multiple subnets map to the same medium, but the hosts may not.

As a consequence, the hosts may not send directly to other hosts on the switched medium. Instead, they may insist on sending to the router and having the router forward the packet to the destination. This behavior is especially common on older UNIX hosts, and on Apple hosts that use MacTCP rather than Open Transport.

Doing this means the packet must traverse the wire twice, and be handled by the router twice. There are "hacks" you can apply to work around the problem [Berkowitz 1998a]. You can reduce the performance hit on the router



by coding **ip route-cache same-interface** on interfaces where the in-and-out behavior is expected.

In the long run, this and other problems go away only when you accept completely classless structure for your addressing, and remove all hardware and software that does not understand classless addressing.

Figure 17. Secondary Addressing

A few other caveats apply to using secondary addressing. All routers connected to the same medium should have the same set of secondary addresses. Do not put one secondary address on one router and two secondaries on the others.

The primary address should be in the same subnet on every router and the secondaries should be in the same order: if 192.168.2.0/24 is the first secondary address on one router and 192.168.3.0/24 is the second, do not put 192.168.3.0 as the first secondary address on a different router connected to the same medium.

Subinterfaces

Subinterfaces might at first seem similar to secondary addresses, but they are significantly more flexible. You most commonly use subinterfaces with NBMA services such as Frame Relay.

Subinterfaces solve quite a number of problems with the interactions of NBMA media and routers. See the CCNA WAN Protocols Tutorial for more detail on their use. As a quick example, if you had a single serial interface over which three Frame Relay virtual circuits were established, you would code:

When you use point-to-point subinterfaces, the most efficient use of address space is to give them /30 prefixes (i.e., the mask 255.255.255.252). Variable-length subnet masks are most common for providing this address space.

```
no ip subnet-zero
interface s0
encapsulation frame-relay
interface s0.1 point-to-point
bandwidth 128 ! the CIR in Kbps
ip address 192.168.1.5 255.255.255.252
interface s0.1 point-to-point
ip address 192.168.1.9 255.255.255.252
bandwidth 64
interface s0.1 point-to-point
ip address 192.168.1.13 255.255.255.252
bandwidth 512
```

Conclusion

IP addressing is one of the most fundamental skills in networking. Addressing techniques have continued to evolve, but the CCNA focuses on the obsolete classful methods.

Whenever you take a Cisco examination, be sure, on every question involving addressing, whether the assumption is that the problem is defined for a classful or classless environment. One good tip is that the environment is classless if the **ip subnet-zero** option is coded. There are a few other commands that become involved in classless routing, such as **no auto-summary** and **ip classless**, but they are beyond the scope of this discussion.

References

[Berkowitz, 1998a] Berkowitz, H. Designing Addressing Architectures for Routing and Switching. Indianapolis, IN: Macmillan Technical Publishing, 1998.

[Huitema] Huitema, C. Routing in the Internet. Englewood Cliffs: Prentice-Hall, 1995.

[RFC0760] J. Postel. "DoD standard Internet Protocol." 1980.

[RFC0791] J. Postel. "Internet Protocol." 1981.

[RFC0950] J. Mogul, J. Postel. "Internet Standard Subnetting Procedure." J1985.

[RFC1517] R. Hinden. "Applicability Statement for the Implementation of Classless Inter-Domain Routing (CIDR)." September 1993.

[RFC1518] Y. Rekhter, T. Li. "An Architecture for IP Address Allocation with CIDR" 1993.

IP Addressing Labs