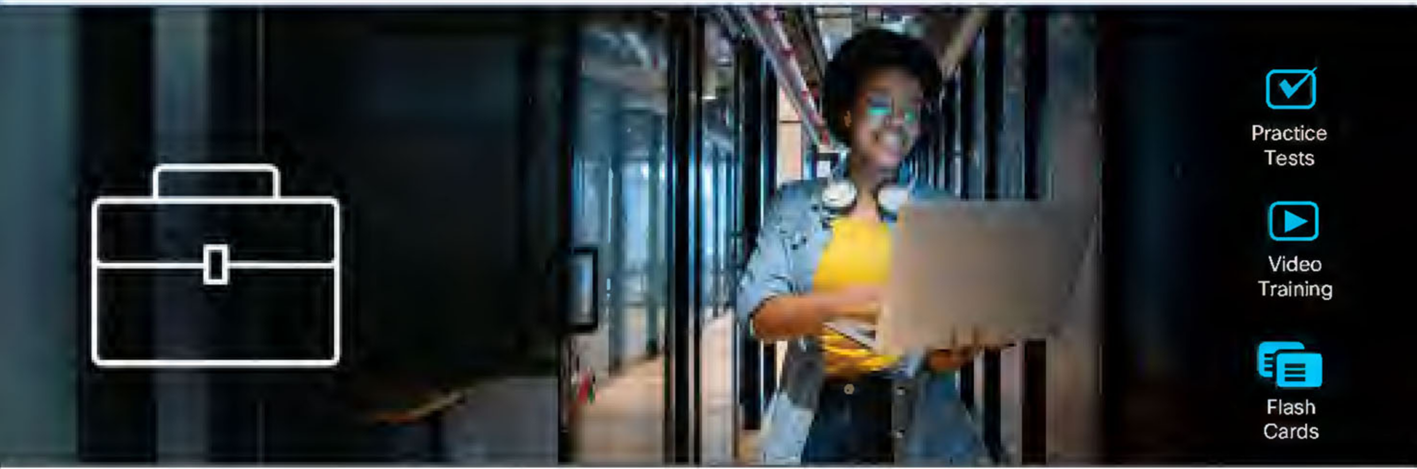
Official Cert Guide Library



Practice Tests

Video Training

Flash Cards

Study Planner

Review Exercises

Labs

CCNA

200-301

2nd Edition

WENDELL ODOM, CCIE® No. 1624
JASON GOOLEY, CCIE® No. 38759
DAVID HUCABY, CCIE® No. 4594

ciscopress.com

Video Training

Flash Cards

Practice tests

Hands-On Labs

Review Exercises

Config Checklists

Official Cert Guide

Advance your IT career with hands-on learning

CCNA 200-301

Volume 1

WENDELL ODOM,
CCIE® NO. 1624 EMERITUS

ciscopress.com

The failure of a ping, even with two devices on the same subnet, can point to a variety of problems, like those mentioned in this list. For instance, if the **ping 172.16.1.51** on R1 fails (refer to Figure 20-7), that result points to this list of potential root causes:

Key Topic

- **IP addressing problem:** Host A or the router could be configured with the wrong IP address.
- **IP mask problem:** Using an incorrect subnet mask on either the host or the router would change their calculation view of the range of addresses in the attached subnet, which would affect their forwarding logic. For example, the host, with address 172.16.1.51 but incorrect mask 255.255.255.240, would think that the router's address of 172.16.1.1 is in a different subnet.
- **DHCP problems:** If you are using Dynamic Host Configuration Protocol (DHCP), many problems could exist. Chapter 19, "IP Addressing on Hosts," discusses those possibilities in some depth.
- **VLAN trunking problems:** The router could be configured for 802.1Q trunking, when the switch is not (or vice versa).
- **LAN problems:** A wide variety of issues could exist with the Layer 2 switches, preventing any frames from flowing between host A and the router.

So, whether the ping works or fails, simply pinging a LAN host from a router can help further isolate the problem.

Testing LAN Neighbors with Extended Ping

A standard ping of a LAN host from a router does not test that host's default router setting. However, an extended ping can test the host's default router setting. Both tests can be useful, especially for problem isolation, because

Key Topic

- If a standard ping of a local LAN host works…
- But an extended ping of the same LAN host fails…
- The problem likely relates somehow to the host's default router setting.

First, to understand why the standard and extended ping results have different effects, consider first the standard **ping 172.16.1.51** command on R1, as shown previously in Figure 20-7. As a standard **ping** command, R1 used its LAN interface IP address (172.16.1.1) as the source of the ICMP Echo. So, when the host (A) sent back its ICMP echo reply, host A considered the destination of 172.16.1.1 as being on the same subnet. Host A's ICMP echo reply message, sent back to 172.16.1.1, would work even if host A did not have a default router setting at all!

In comparison, Figure 20-8 shows the difference when using an extended ping on Router R1. An extended ping from local Router R1, using R1's WAN IP address of 172.16.4.1 as the source of the ICMP echo request, means that host A's ICMP echo reply will flow to an address in another subnet, which makes host A use its default router setting.

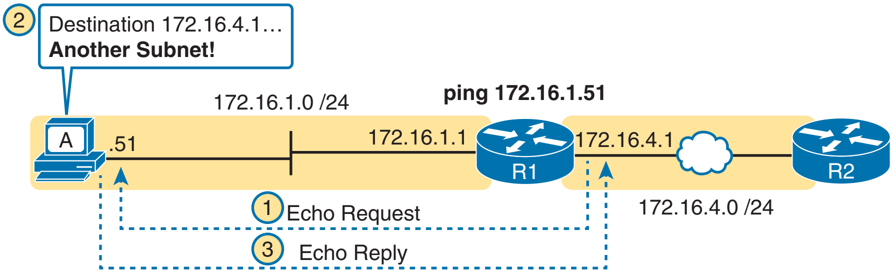


Figure 20-8 *Extended* **ping** *Command Does Test Host A's Default Router Setting*

The comparison between the previous two figures shows one of the most classic mistakes when troubleshooting networks. Sometimes, the temptation is to connect to a router and ping the host on the attached LAN, and it works. So, the engineer moves on, thinking that the network layer issues between the router and host work fine, when the problem still exists with the host's default router setting.

Testing WAN Neighbors with Standard Ping

As with a standard ping test across a LAN, a standard ping test between routers over a serial or Ethernet WAN link tests whether the link can pass IPv4 packets. With a properly designed IPv4 addressing plan, two routers on the same serial or Ethernet WAN link should have IP addresses in the same subnet. A ping from one router to the IP address of the other router confirms that an IP packet can be sent over the link and back, as shown in the **ping 172.16.4.2** command on R1 in Figure 20-9.

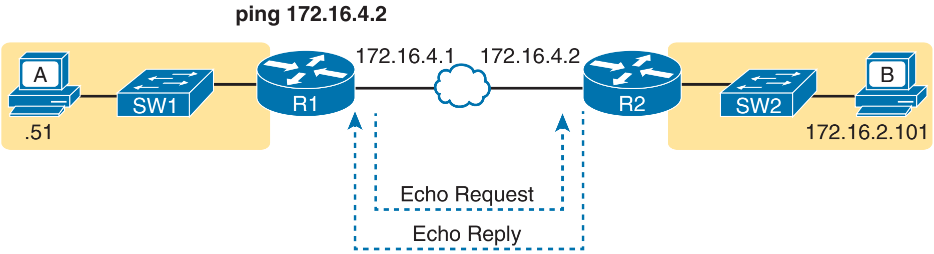


Figure 20-9 *Pinging Across a WAN Link*

A successful ping of the IP address on the other end of an Ethernet WAN link that sits between two routers confirms several specific facts, such as the following:

- Both routers' WAN interfaces are in an up/up state.
- The Layer 1 and 2 features of the link work.
- The routers believe that the neighboring router's IP address is in the same subnet.
- Inbound ACLs on both routers do not filter the incoming packets, respectively.
- The remote router is configured with the expected IP address (172.16.4.2 in this case).

Pinging the other neighboring router does not test many other features. However, although the test is limited in scope, it does let you rule out WAN links as having a Layer 1 or 2 problem, and it rules out some basic Layer 3 addressing problems.

Using Ping with Names and with IP Addresses

All the ping examples so far in this chapter show a ping of an IP address. However, the **ping** command can use **hostnames**, and pinging a hostname allows the network engineer to further test whether the Domain Name System (**DNS**) process works.

First, most every TCP/IP application uses hostnames rather than IP addresses to identify the other device. No one opens a web browser and types in 72.163.4.185. Instead, they type in a web address, like https://www.cisco.com, which includes the hostname www.cisco.com. Then, before a host can send data to a specific IP address, the host must first ask a DNS server to resolve that hostname into the matching IP address.

For example, in the small internetwork used for several examples in this chapter, a **ping B** command on host A tests A's DNS settings, as shown in Figure 20-10. When host A sees the use of a hostname (B), it first looks in its local DNS name cache to find out whether it has already resolved the name B. If not, host A first asks the DNS to supply (resolve) the name into its matching IP address (Step 1 in the figure). Only then does host A send a packet to 172.16.2.101, host B's IP address (Step 2).

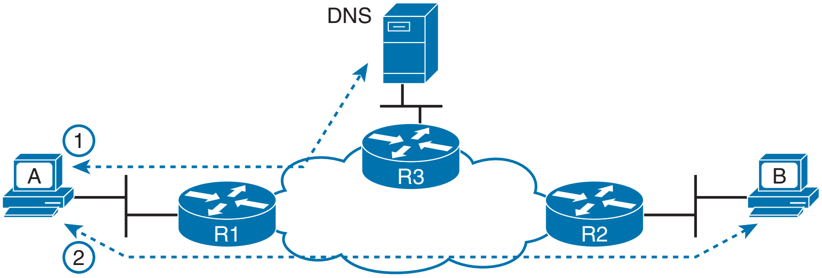


Figure 20-10 *DNS Name Resolution by Host A*

When troubleshooting, testing from the host by pinging using a hostname can be very helpful. The command, of course, tests the host's own DNS client settings. For instance, a classic comparison is to first ping the destination host using the hostname, which requires a DNS request. Then, repeat the same test, but use the destination host's IP address instead of its name, which does not require the DNS request. If the ping of the hostname fails but the ping of the IP address works, the problem usually has something to do with DNS.

Routers and switches can also use name resolution for commands that refer to hosts, such as the **ping** and **traceroute** commands. The networking device can use DNS, locally defined hostnames, or both. Example 20-4 shows an example DNS configuration on Router R1 from the most recent examples. In particular:

Key Topic

- The **ip domain lookup** command tells the router to attempt to use a DNS server.
- The **ip name-server** {*address* [*address address*]} command defines the list of DNS server IP addresses.
- The **ip domain name** *domain-name* command defines the domain used by the device.

In the example, note that once configured to use DNS, the **ping hostB** command works. The command output shows the IP address the DNS resolution process found for name hostB, 172.16.2.101.

Example 20-4 *Configuring to Use DNS (Router R1), DNS Has hostB Name*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# ip domain lookup
R1(config)# ip domain name example.com
R1(config)# ip name-server 8.8.8.8 8.8.8.4
R1(config)# ^Z
R1#
R1# ping hostB
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.2.101, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/4 ms
```

NOTE Older IOS versions used a syntax of **ip domain-name** *domain-name* rather than the newer **ip domain name** *domain-name* (with a space instead of a dash).

NOTE When practicing, you might want to disable DNS resolution, particularly in lab devices, using the **no ip domain lookup** command. Cisco routers and switches enable DNS resolution by default with a setting of **ip domain lookup**, but with no name servers identified with the **ip name-server** command. The result of these two default settings causes the router or switch to perform name resolution on a name by broadcasting for a DNS server on each connected subnet. Additionally, if you mistype the first word of a command, IOS thinks you mean that word to be a hostname, and it attempts to perform name resolution on the mistyped command. The result: for any typo of the first word in a command, the default name resolution settings cause a few minutes wait until you get control of the CLI again.

In a lab environment, when not expecting to use DNS, disable DNS resolution with the **no ip domain lookup** command.

You can also configure the router or switch to use locally configured hostnames (or to use both locally configured names and DNS). Use a configuration like that in Example 20-5, adding the **ip host** *name address* global configuration command for each hostname. The router or switch will look for local hostnames whether you use DNS or have the **ip domain lookup** command configured.

Example 20-5 *Configuring to Use Local Hostnames, R1 Config Has hostB Name*

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# ip host hostB 172.16.2.101
R1(config)# ^Z
R1#
```

```
R1# show hosts
Default domain is example.com
Name servers are 8.8.8.8, 8.8.8.4
NAME  TTL  CLASS   TYPE     DATA/ADDRESS
-----------------------------------------
 101.2.16.172.in-addr.arpa      10       IN       PTR      hostB
 hostB                          10       IN       A        172.16.2.101
```

Problem Isolation Using the traceroute Command

Like **ping**, the **traceroute** command helps network engineers isolate problems. Here is a comparison of the two:

- Both send messages in the network to test connectivity.
- Both rely on other devices to send back a reply.
- Both have wide support on many different operating systems.
- Both can use a hostname or an IP address to identify the destination.
- On routers, both have a standard and extended version, allowing better testing of the reverse route.

The biggest differences relate to the more detailed results in the output of the **traceroute** command and the extra time and effort it takes **traceroute** to build that output. This next major section examines how traceroute works; plus it provides some suggestions on how to use this more detailed information to more quickly isolate IP routing problems.

traceroute Basics

Imagine some network engineer or CSR starts to troubleshoot some problem. The engineer pings from the user's host, pings from a nearby router, and after a few commands, convinces herself that the host can indeed send and receive IP packets. The problem might not be solved yet, but the problem does not appear to be a network problem.

Now imagine the next problem comes along, and this time the **ping** command fails. It appears that some problem does exist in the IP network. Where is the problem? Where should the engineer look more closely? Although the **ping** command can prove helpful in isolating the source of the problem, the **traceroute** command may be a better option. The **traceroute** command systematically helps pinpoint routing problems by showing how far a packet goes through an IP network before being discarded.

The **traceroute** command identifies the routers in the forward route from source host to destination host. Specifically, it lists the next-hop IP address of each router that would be in each of the individual routes. For instance, a **traceroute 172.16.2.101** command on host A in Figure 20-11 would identify an IP address on Router R1, another on Router R2, and then host B, as shown in the figure. Example 20-6, which follows, lists the output of the command, taken from host A.

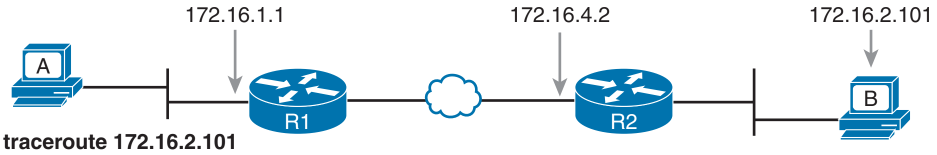


Figure 20-11 *IP Addresses Identified by a Successful* **traceroute 172.16.2.101** *Command*

Example 20-6 *Output from* **traceroute 172.16.2.101** *on Host A*

```
Mac_A$ traceroute 172.16.2.101
traceroute to 172.16.2.101, 64 hops max, 52 byte packets
  1 172.16.1.1  (172.16.1.1) 0.870 ms 0.520 ms 0.496 ms
  2 172.16.4.2  (172.16.4.2) 8.263 ms 7.518 ms 9.319 ms
  3 172.16.2.101  (172.16.2.101) 16.770 ms 9.819 ms 9.830 ms
```

How the traceroute Command Works

The **traceroute** command gathers information by generating packets that trigger error messages from routers; these messages identify the routers, letting the **traceroute** command list the routers' IP addresses in the output of the command. That error message is the ICMP Time-to-Live Exceeded (TTL Exceeded) message, originally meant to notify hosts when a packet had been looping around a network.

Ignoring traceroute for a moment and instead focusing on IP routing, IPv4 routers defeat routing loops in part by discarding looping IP packets. To do so, the IPv4 header holds a field called Time To Live (TTL). The original host that creates the packet sets an initial TTL value. Then each router that forwards the packet decrements the TTL value by 1. When a router decrements the TTL to 0, the router perceives the packet is looping, and the router discards the packet. The router also notifies the host that sent the discarded packet by sending an ICMP TTL Exceeded message.

Now back to traceroute. Traceroute sends messages with low TTL values to make the routers send back a TTL Exceeded message. Specifically, a **traceroute** command begins by sending several packets (usually three), each with the header TTL field equal to 1. When that packet arrives at the next router—host A's default Router R1 in the example of Figure 20-12—the router decrements TTL to 0 and discards the packet. The router then sends host A the TTL Exceeded message, which identifies the router's IP address to the **traceroute** command.

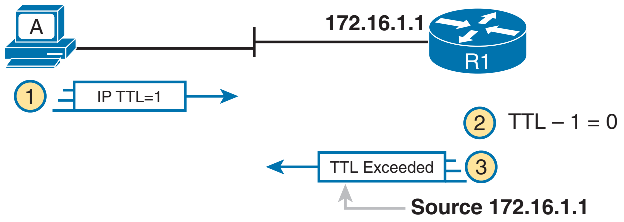


Figure 20-12 *How* **traceroute** *Identifies the First Router in the Route*