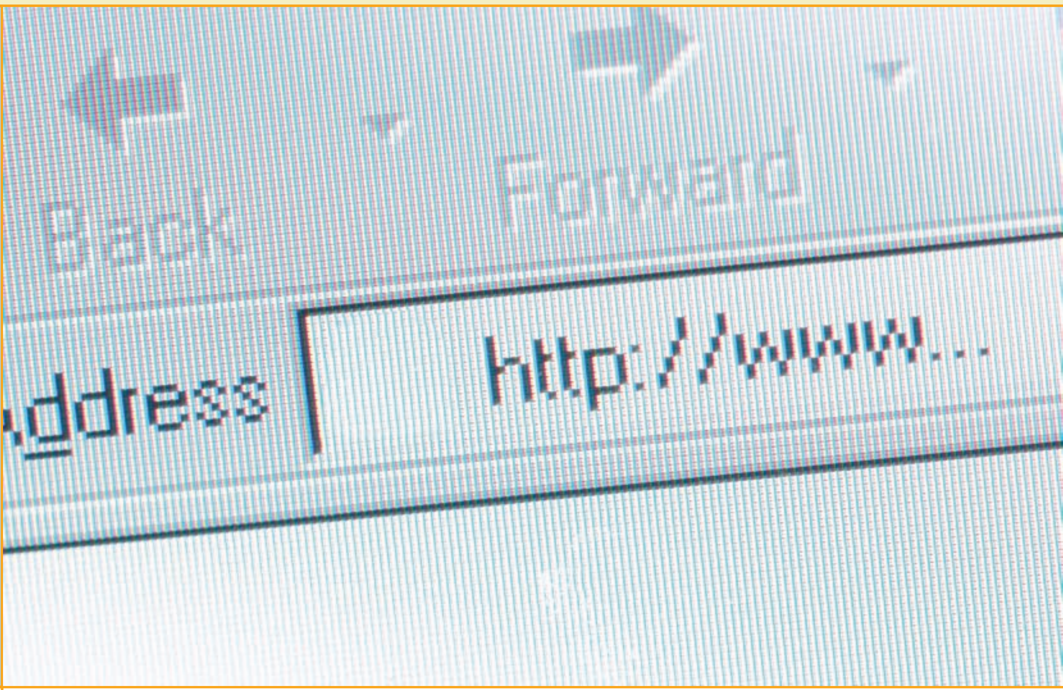
chapter 10

Network Naming

"What's in a name? That which
we call a rose
By any other name would smell
as sweet."

—William Shakespeare



In this chapter, you will learn how to

- **Describe the function and capabilities of DNS**
- **Configure and troubleshoot WINS**
- **Use common TCP/IP utilities to diagnose problems with DNS and WINS**

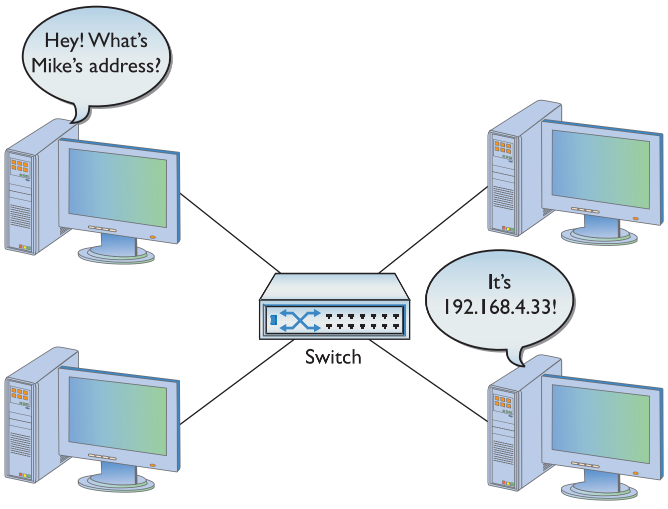
Did the last chapter seem a bit IP address heavy to you? When you open a Web page, for example, do you normally type something like http://192.168.4.1 or do you usually type something like www.totalsem.com? Odds are good you normally do the latter and only rarely the former. There's a good reason for this: people are terrible at memorizing numbers but are pretty good at memorizing words. This creates an interesting dilemma.

Although computers use IP addresses to communicate with each other over a TCP/IP network, people prefer easy-to-remember names over IP addresses. To solve this problem, TCP/IP developers created a process called **name resolution** to convert names to IP addresses (and *vice versa*) to make it easier for people to communicate with the computers (Figure 10.1).

Like any process that's been around for a long time, name resolution has gone through a number of evolutions over the years: some dramatic and some subtle. Entire TCP/IP applications have been written, only to be supplanted (but never totally abandoned) by newer name resolution protocols.

The end result is that today we have a single major name resolution protocol called **Domain Name System (DNS)**, but your brand new system running the latest version of whatever operating system you prefer still fully supports a number of much older name resolution protocols! Name resolution in today's networking world is like a well-run home that's also full of ghosts that can do very strange things if you don't understand how those ghosts think.

In this chapter, you'll take an in-depth tour of name resolution, starting with an in-depth discovery of DNS. After DNS, the chapter looks at one of the scariest ghosts running around inside your computer: an ancient and theoretically abandoned name resolution protocol invented by Microsoft called **Windows Internet Name Service (WINS)**. Despite what Microsoft claims, the ghost of WINS still lingers, not only on Windows computers, but even on most Linux and Macintosh OS X systems too, as these folks discovered that if you don't respect these ghosts, you won't be able to do name resolution when you connect to a Windows computer.



● Figure 10.1 Turning names into numbers

Odds are good you have a system that is connected—or at least can connect—to the Internet. If I were you, I'd fire that system up, because the vast majority of the programs you're going to learn about here come free with every operating system made. Finding them may be a challenge on some systems, but don't worry—I'll show you where they all hang out.

Historical/Conceptual

DNS

When the Internet was very young and populated with only a few hundred computers, name resolution was pretty simple. The original TCP/IP specification implemented name resolution using a special text file called HOSTS. A copy of this file was stored on every computer system on the Internet. The **HOSTS file** contained a list of IP addresses for every computer on the Internet, matched to the corresponding system names. Remember, not only was the Internet a lot smaller then, there weren't yet rules about how to compose Internet names, such as that they must end in .com or .org, or start with www or ftp. Anyone could name their computer pretty much anything they wanted (there were a few restrictions on length and allowable characters) as long as nobody else had snagged the name first. Part of an old HOSTS file might look something like this:

```
192.168.2.1    fred
201.32.16.4    school2
123.21.44.16   server
```

If your system wanted to access the system called fred, it looked up the name fred in its HOSTS file, and then used the corresponding IP address to contact fred. Every HOSTS file on every system on the Internet was updated every morning at 2 A.M. This worked fine when the Internet was still the province of a few university geeks and some military guys, but when the Internet grew to about 5000 systems, it became impractical to make every system use and update a HOSTS file. This created the motivation for a more scalable name resolution process, but the HOSTS file did not go away.

Believe it or not, the HOSTS file is still alive and well in every computer. You can find the HOSTS file in the \WINNT\SYSTEM32\DRIVERS\ETC folder in Windows 2000, and in \WINDOWS\SYSTEM32\DRIVERS\ETC in Windows XP/2003/Vista/Windows 7. On OS X and Linux systems, it's usually found in the /etc/ folder. It's just a text file that you can open with any text editor. Below are a few lines from the default HOSTS file that comes with Windows. See the # signs? Those are remark symbols that designate lines as comments (for humans to read) rather than code. Windows ignores any line that begins with #. Remove the # and Windows will read the line and try to act on it. While all operating systems continue to support the

Try This

Editing the HOSTS File

Every Windows computer has a HOSTS file that you can edit, so try this!

1. Go to a command prompt and type `ping www.totalsem.com`. You may or may not be successful with the PING, but you will get the IP address for my Web site. (You may get a different IP address from the one shown in this example.)

```
C:\>ping www.totalsem.com
Pinging www.totalsem.com [209.29.33.25] with 32 bytes of data:
Reply from 209.29.33.25: bytes=32 time=60ms TTL=51
Reply from 209.29.33.25: bytes=32 time=60ms TTL=51
Reply from 209.29.33.25: bytes=32 time=60ms TTL=51
Reply from 209.29.33.25: bytes=32 time=60ms TTL=51
Ping statistics for 209.29.33.25:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 60ms, Maximum = 60ms, Average = 60ms
```

2. Open your HOSTS file using any text editor and add this line (keep in mind you may have a different IP address from the one shown in this example). Just press the SPACEBAR a few times to separate the IP address from the word "timmy."

```
209.29.33.25      timmy
```

3. Save the HOSTS file and close the text editor.
4. Open your Web browser and type **timmy**. You can also type **http://timmy** if you'd like. What happens?

HOSTS file, it is rarely used in the day-to-day workings of most TCP/IP systems.

```
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#         102.54.94.97      rhino.acme.com              # source server
#          38.25.63.10      x.acme.com                  # x client host
127.0.0.1          localhost
```

Even though the HOSTS file is rarely used, *every* operating system will always first look in the HOSTS file before anything else when attempting to resolve a name. To see the power of the HOSTS file, do the first Try This! sidebar in this chapter.

The Try This! sidebar example uses a Web browser, but keep in mind that a name in a HOSTS file resolves names for *every* TCP/IP application on that system. Go to a command prompt and type `ping timmy`. It works for PING too.

HOSTS files still have their place in today's world. Lots of people place shortcut names in a HOSTS file to avoid typing long names in some TCP/IP applications. Yet even though HOSTS still has some use, for the most part we use the vastly more powerful DNS.

Test Specific

How DNS Works

The Internet folks, faced with the task of replacing HOSTS, first came up with the idea of creating one supercomputer that did nothing but resolve names for all the other computers on the Internet. Problem: even now, no computer is big enough or powerful enough to handle the job alone. So, they fell back on that time-tested bureaucratic solution: delegation! The top-dog DNS system would delegate parts of the job to subsidiary DNS systems, who, in turn, would delegate part of their work to other systems, and so on, potentially without end. These systems run a special DNS server program, and are called, amazingly enough, **DNS servers**.

This is all peachy, but it raises another issue: you need some way to decide how to divvy up the work. Toward this end, they created a naming system designed to facilitate delegation. The top-dog DNS server is actually a bunch of powerful computers dispersed around the world and working as a team, known collectively as the **DNS root servers** (or simply as the *DNS root*). The Internet name of this computer team is "."—that's right, just "dot." Sure, it's weird, but it's quick to type, and they had to start somewhere.

The DNS root for the entire Internet consists of 13 powerful DNS server clusters scattered all over the world. Go to http://www.root-servers.org to see exactly where all the root servers are located.

DNS root has the complete definitive name resolution table, but most name resolution work is delegated to other DNS servers. Just below the DNS root in the hierarchy is a set of DNS servers—called the **top-level domain servers**—that handle what are known as the top-level domain names. These are the famous COM, ORG, NET, EDU, GOV, MIL, and INT. The top-level DNS servers delegate to thousands of second-level DNS servers; they

DNS servers use TCP and UDP port 53.

The original top-level domain names were COM, ORG, NET, EDU, GOV, MIL, and INT.

Tech Tip

Going Beyond Three-Tier Names

The Internet DNS names are usually consistent with this three-tier system, but if you want to add your own DNS server(s), you can add more levels, allowing you to name a computer www.houston.totalsem.com if you wish. The only limit is that a DNS name can be a maximum of 255 characters.

handle the millions of names like totalsem.com and whitehouse.gov that have been created within each of the top-level domains. Second-level DNS servers support individual computers. For example, stored on the DNS server controlling the totalsem.com domain is a listing that looks like this:

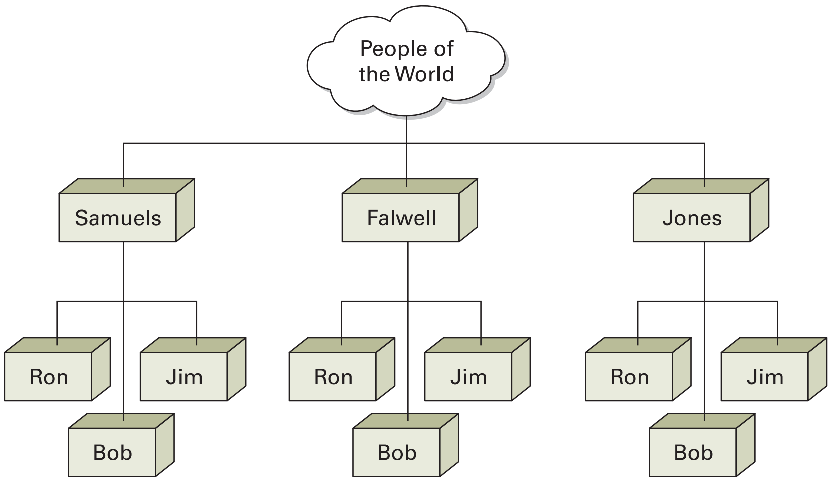
```
www     209.29.33.25
```

This means the totalsem.com domain has a computer called *www* with the IP address of 209.29.33.25. Only the DNS server controlling the totalsem.com domain stores the actual IP address for *www*.totalsem.com. The DNS servers above this one have a hierarchical system that enables any other computer to find the DNS server that controls the totalsem.com domain.

Name Spaces

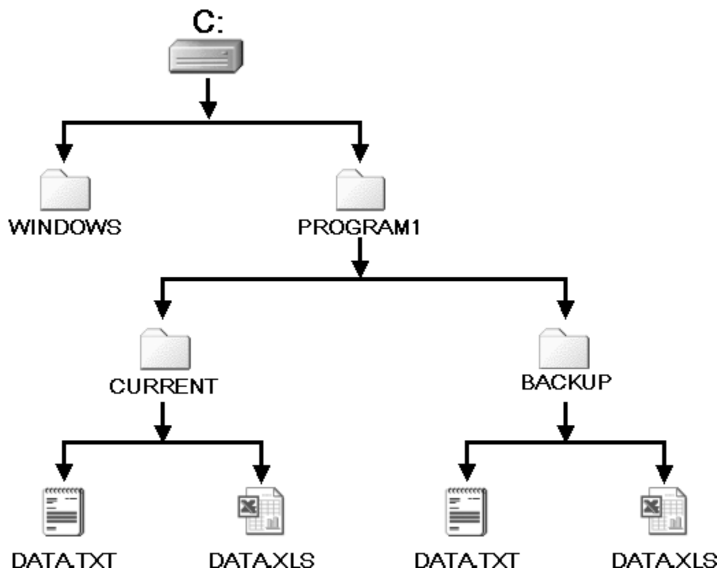
What does *hierarchical* mean in terms of DNS? Well, the DNS **hierarchical name space** is an imaginary tree structure of all possible names that could be used within a single system. By contrast, a HOSTS file uses a **flat name space**—basically just one big undivided list containing all names, with no grouping whatsoever. In a flat name space, all names must be absolutely unique—no two machines can ever share the same name under any circumstances. A flat name space works fine on a small, isolated network, but not so well for a large organization with many interconnected networks. To avoid naming conflicts, all its administrators would need to keep track of all the names used throughout the entire corporate network.

A hierarchical name space offers a better solution, permitting a great deal more flexibility by enabling administrators to give networked systems longer, more fully descriptive names. The personal names people use every day are an example of a hierarchical name space. Most people address our town postman, Ron Samuels, simply as Ron. When his name comes up in conversation, people usually refer to him as Ron. The town troublemaker, Ron Falwell, and Mayor Jones's son, Ron, who went off to Toledo, obviously share first names with the postman. In some conversations, people need to distinguish between the good Ron, the bad Ron, and the Ron in Toledo (who may or may not be the ugly Ron). They could use a medieval style of address, and refer to the Rons as Ron the Postman, Ron the Blackguard, and Ron of Toledo, or they could use the modern Western style of address and add their surnames: "That Ron Samuels—he is such a card!" "That Ron Falwell is one bad apple." "That Ron Jones was the homeliest child I ever saw." You might visualize this as the People name space, illustrated in Figure 10.2. Adding the surname creates what you might fancifully call a *Fully Qualified Person Name*—enough information to prevent confusion among the various people named Ron.



• **Figure 10.2** Our People name space

A name space most of you are already familiar with is the hierarchical file name space used by hard drive volumes. Hard drives formatted using one of the popular file formats, like Window's NTFS or Linux's EXT3, use a hierarchical name space; you can create as many files named DATA.TXT as you want, as long as you store them in different parts of the file tree. In the example shown in Figure 10.3, two different files named DATA.TXT can exist simultaneously on the same system, but only if they are placed in different directories, such as C:\PROGRAM1\CURRENT\DATA.TXT and C:\PROGRAM1\BACKUP\DATA.TXT. Although both files have the same basic filename—DATA.TXT—their fully qualified names are different: C:\PROGRAM1\CURRENT\DATA.TXT and C:\PROGRAM1\BACKUP\DATA.TXT. Additionally, multiple subfolders can use the same name. There's no problem with having two subfolders using the name DATA, as long as they reside in different folders. Any Windows file system will happily let you create both C:\PROGRAM1\DATA and C:\PROGRAM2\DATA folders. We like this because we often want to give the same name to multiple folders doing the same job for different applications.



• **Figure 10.3** Two DATA.TXT files in different directories on the same system

In contrast, imagine what would happen if your computer's file system didn't support folders/directories. It would be as if Windows had to store all the files on your hard drive in the root directory! This is a classic example of a flat name space. Because all your files would be living together in one directory, each one would have to have a unique name. Naming files would be a nightmare! Software vendors would have to avoid sensible descriptive names like README.TXT, because they would almost certainly have been used already. You'd probably have to do what the Internet does for IP addresses: An organization of some sort would assign names out of the limited pool of possible filenames. With a hierarchical name space, on the other hand, which is what all file systems use (thank goodness!), naming is much simpler. Lots of programs can have files called README.TXT, because each program can have its own folder and subfolders.

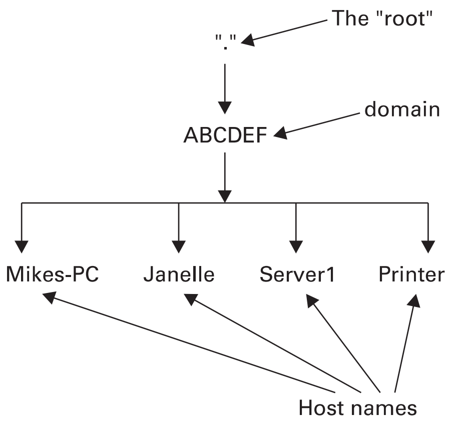
As hard as this may be to believe, some early file systems used a flat name space. Back in the late 1970s and early 1980s, operating systems such as CPM and the early versions of DOS did not have the capability to use directories, creating a flat name space where all files resided on a single drive.

The DNS name space works in a manner extremely similar to how your computer's file system works. The DNS name space is a hierarchy of *DNS domains* and individual computer names organized into a tree-like structure that we call, rather appropriately, a *tree*. Each domain is like a folder—a domain is not a single computer, but rather a holding space into which you can add computer names. At the top of a **DNS tree** is the root. The *root* is the holding area to which all domains connect, just as the root directory in your file system is the holding area for all your folders. Individual computer names—more commonly called **host names** in the DNS naming convention—fit into domains. In the PC, you can place files directly into the root directory. The DNS world also enables us to add computer names to the root, but with the exception of a few special computers (described in a moment), this is rarely done. Each domain can have subdomains, just as the

Even though you may use uppercase or lowercase, DNS does not differentiate between them.

folders on your PC's file system can have subfolders. You separate each domain from its subdomains with a period. Characters for DNS domain names and host names are limited to uppercase and lowercase letters (A–Z, a–z), numbers (0–9), and the hyphen (-). No other characters may be used.

Don't think DNS is only for computers on the Internet. If you want to make your own little TCP/IP network using DNS, that's fine, although you will have to set up at least one DNS server as the root server for your little private *intranet*. Every DNS server program can be configured as a root server; just don't connect that DNS server to the Internet, because it won't work outside your little network. Figure 10.4 shows a sample DNS tree for a small TCP/IP network that is not attached to the Internet. In this case, there is only one domain: ABCDEF. Each computer on the network has a host name, as shown in the figure.



• **Figure 10.4** Private DNS network

When you write out the complete path to a file stored on your PC, the naming convention starts with the root directory on the left, followed by the first folder, then any subfolders (in order), and finally the name of the file—for example, c:\sounds\thunder\mynewcobra.wav. The DNS naming convention is *exactly the opposite*. A complete DNS name, including the host name and all of its domains (in order), is called a **fully qualified domain name (FQDN)**, and it's written out with the root on the far right, followed by the names of the domains (in order) added to the left of the root, and the host name on the far left. Figure 10.4 shows the FQDNs for two systems in the ABCDEF domain. Note the period for the root is on the far *right* of each FQDN!

Mikes-PC.ABCDEF.
Janelle.ABCDEF.

Given that every FQDN will always have a period on the end to signify the root, it is commonplace to drop the final period when writing out FQDNs. To make the two example FQDNs fit into common parlance, therefore, you'd skip the last period:

Mikes-PC.ABCDEF
Janelle.ABCDEF

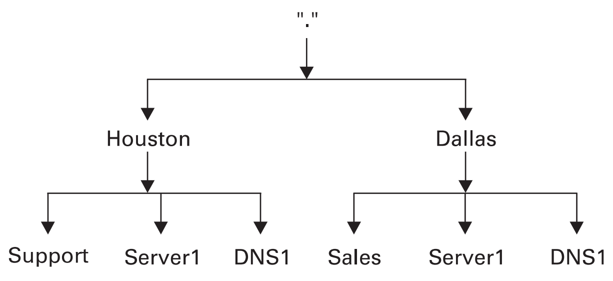
Tech Tip

.com Is Not Always

Don't get locked into thinking FQDNs always end with names like ".com" or ".net." True, DNS names on the Internet must always end with them, but private TCP/IP networks can (and often do) ignore this and use whatever naming scheme they want with their DNS names.

If you're used to seeing DNS names on the Internet, you're probably wondering about the lack of ".com," ".net," or other common DNS domain names. Those conventions are needed for computers that are visible on the Internet, such as Web servers, but they're not required on a private TCP/IP network. As long as you make a point never to make these computers visible on the Internet, you can use any naming convention you want!

Let's look at another DNS name space example, but make it a bit more complex. This network is not on the Internet, so I can use any domain I want. The network has two domains, Houston and Dallas, as shown in Figure 10.5. Note that each domain has a computer called Server1.

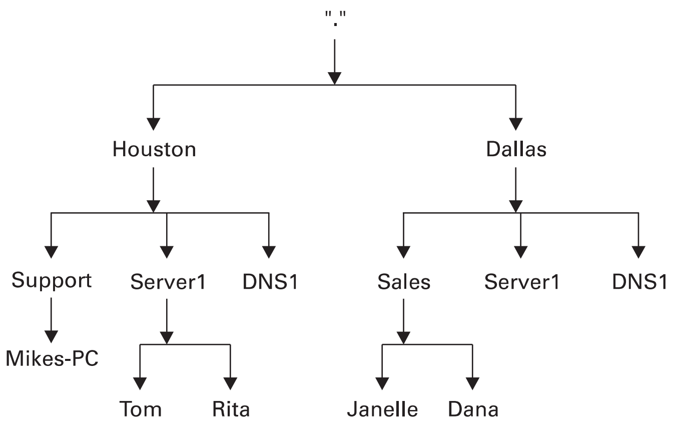


• **Figure 10.5** Two DNS domains

Because the network has two different domains, it can have two systems (one on each domain) with the same host name, just as you can have two files with the same name in different folders on your PC. Now, let's add some subdomains to the DNS tree, so that it looks like Figure 10.6.

You write out the FQDN from left to right, starting with the host name and moving up to the top of the DNS tree, adding all domains until you get to the top of the DNS tree:

Mikes-PC.Support.Houston
Tom.Server1.Houston
Janelle.Sales.Dallas
Server1.Dallas



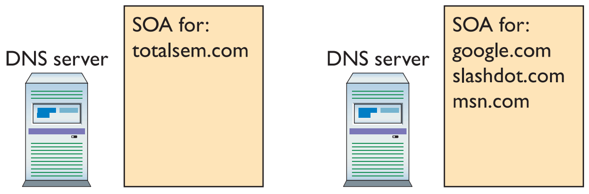
• Figure 10.6 Subdomains added

Name Servers

So where does this naming convention reside and how does it work? Here's where the analogy to the PC's file system breaks down completely. DNS does not have a single hard drive in which to store a directory structure, like you have on a PC. Rather, systems running DNS server software store the DNS information. When a system needs to know the IP address for a specific FQDN, it queries the DNS server listed in its TCP/IP configuration. On a simple network, there is usually one DNS server for the entire network. This single DNS server has a list of all the host names on the domain and their corresponding IP addresses. It's known as the **authoritative DNS server** for the domain (also called Start of Authority, or SOA).

The DNS naming convention allows for DNS names up to 255 characters, including the separating periods.

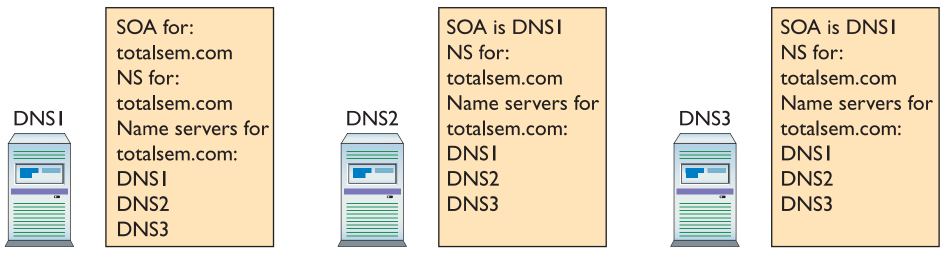
Folks who administer complex networks assign different domains to different DNS servers to keep a single server from being swamped by DNS requests. A single DNS server acts as the authoritative DNS server for one domain or many domains (Figure 10.7).



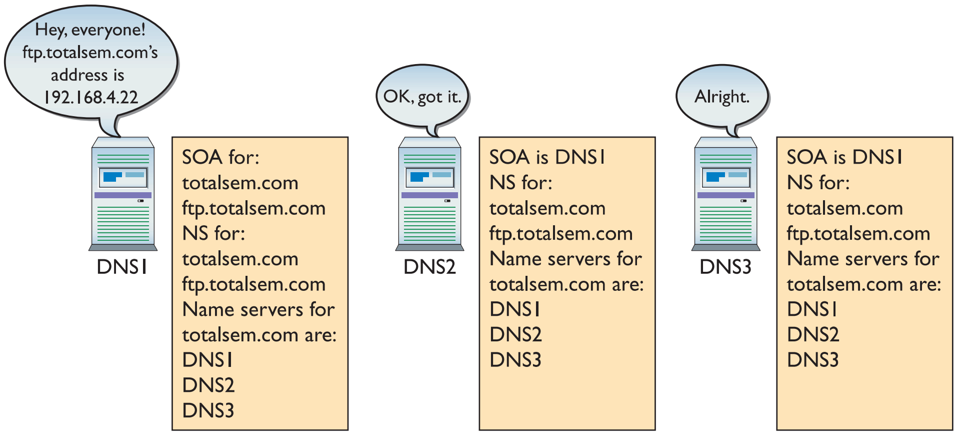
• Figure 10.7 A single SOA can support one or more domains.

Equally, a single DNS domain may have a single authoritative DNS server but a number of other DNS servers, known simply as **name servers** (we use the abbreviation "NS"), that are subordinate to the authoritative DNS server but all support the same domain, as shown in Figure 10.8. The SOA is a name server as well.

Note that every DNS server, whether it's the SOA or just an NS, knows the name and address of the SOA as well as every other NS server in the domain. It's the SOA's job to make sure that all the other name servers are



• Figure 10.8 DNS flexibility

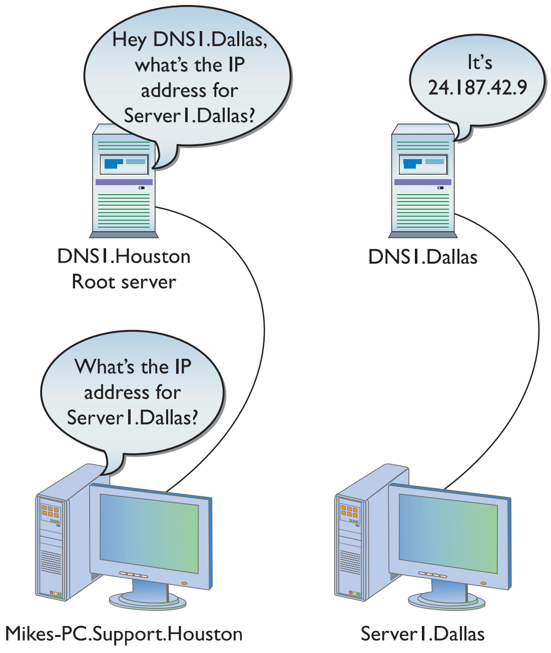

Hey, everyone! ftp.totalsem.com's address is 192.168.4.22
DNS1
SOA for: totalsem.com ftp.totalsem.com NS for: totalsem.com ftp.totalsem.com Name servers for totalsem.com are: DNS1 DNS2 DNS3
OK, got it.
DNS2
SOA is DNS1 NS for: totalsem.com ftp.totalsem.com Name servers for totalsem.com are: DNS1 DNS2 DNS3
Alright.
DNS3
SOA is DNS1 NS for: totalsem.com ftp.totalsem.com Name servers for totalsem.com are: DNS1 DNS2 DNS3

• **Figure 10.9** New information passed out

updated for changes. Let's say you add to the totalsem.com domain a new computer called ftp.totalsem.com with the IP address 192.168.4.22. As an administrator, you typically add this data to the SOA DNS server. The SOA will then automatically distribute this information to the other name servers in the domain (Figure 10.9). This is a critically important feature of DNS—you'll see more of this in detail later in this chapter. For now just appreciate that you can have multiple DNS servers for a single domain.

In the early days of DNS, you were required to enter manually into your DNS server the host name and IP address of every system on the network. While you can still do that if you want, today most DNS servers take advantage of Dynamic DNS (DDNS). DDNS enables systems to register their host names with their DNS server automatically, eliminating the need to enter them manually.

Now let's see how root servers work in DNS. What if Mikes-PC.Support.Houston needed the IP address of Server1.Dallas? Refer to Figure 10.10 for the answer. The network has two DNS servers: DNS1.Houston and DNS1.Dallas. DNS1.Dallas is the authoritative DNS server for all of the Dallas domains and DNS1.Houston is in charge of all the Houston domains. DNS1.Houston is also the root server for the entire network. (DNS servers may act as both a root server and a SOA at the same time—a very common practice in private networks.) As a root server, the Houston server has a listing for the SOA in the Dallas domain. This does *not* mean it knows the IP address for every system in the Dallas network. As a root server, it only knows


Hey DNS1.Dallas, what's the IP address for Server1.Dallas?
It's 24.187.42.9
DNS1.Houston Root server
DNS1.Dallas
What's the IP address for Server1.Dallas?
Mikes-PC.Support.Houston
Server1.Dallas

• **Figure 10.10** Root server in action that if any system asks for an IP address from the Dallas side, it will tell that system the IP address of the Dallas server. The requesting system will then ask the Dallas DNS server (DNS1.Dallas) for the IP address of the system it needs. That's the beauty of DNS root servers—they don't know the IP addresses for all of the computers, but they know where to send the requests!

The hierarchical aspect of DNS has a number of benefits. For example, the vast majority of Web servers are called www. If DNS used a flat name space, only the first organization that created a server with the name www could use it. Because DNS naming appends domain names to the server names, however, the servers www.totalsem.com and www.microsoft.com can both exist simultaneously. DNS names like www.microsoft.com must fit within a worldwide hierarchical name space, meaning that no two machines should ever have the same FQDN.

Just because most Web servers are named www doesn't mean they must be named www! Naming a Web server www is etiquette, not a requirement.

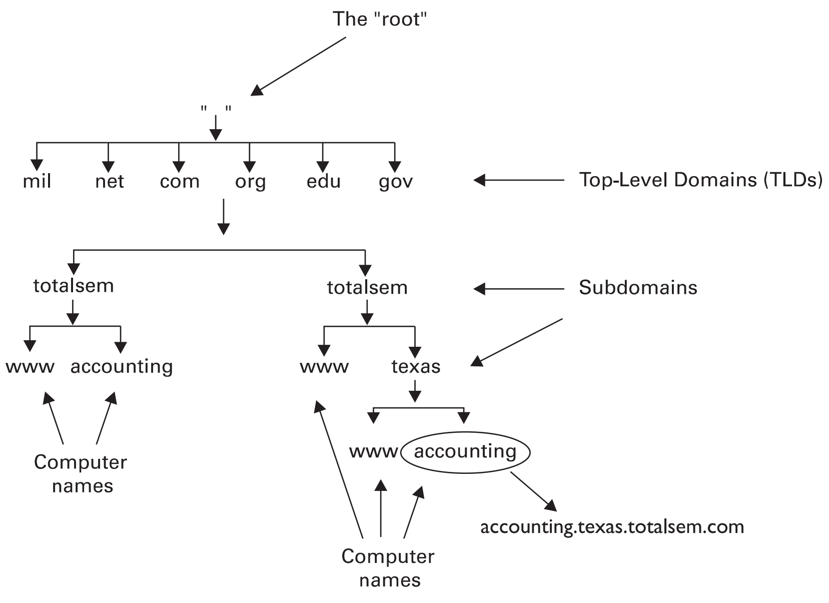
Figure 10.11 shows the host named accounting with an FQDN of accounting.texas.totalsem.com.

These domain names must be registered for Internet use with an organization called the Internet Corporation for Assigned Names and Numbers, or ICANN (www.icann.org). They are arranged in the familiar "second level.top level" domain name format, where the top level is COM, ORG, NET, and so on, and the second level is the name of the individual entity registering the domain name.

Technically, the texas .totalsem.com domain shown in Figure 10.11 is a subdomain of totalsem.com. Don't be surprised to see the terms "domain" and "subdomain" used interchangeably, as it's a common practice.

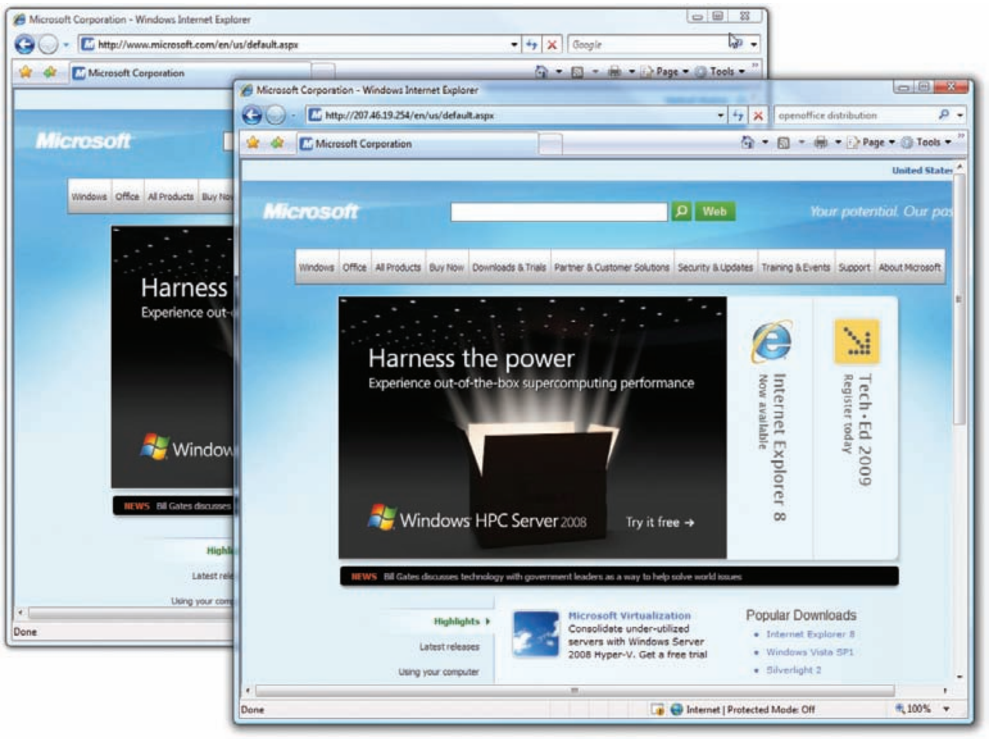
Name Resolution

You don't have to use DNS to access the Internet, but it sure makes life a lot easier! Programs like Internet Explorer accept names such as www.microsoft.com as a convenience to the end user, but utilize the IP address that corresponds to that name to create a connection. If you know the IP address of the system you want to talk to, you don't need DNS at all. Figure 10.12 shows Internet Explorer displaying the same Web page when given the straight IP address as it does when given the DNS name www .microsoft.com. In theory, if you knew the IP addresses of all the systems you wanted to access, you could turn off DNS completely. I guess you could also start a fire using a bow and drill too, but most people wouldn't make a habit of it if there were a more efficient alternative, which in this case, DNS definitely is! I have no trouble keeping hundreds of DNS names in my head, but IP addresses? Forget it! Without DNS, I might as well not even try to use the Internet, and I'd wager that's true of most people.



• Figure 10.11 DNS domain

When you type in a Web address, Internet Explorer must resolve that name to the Web server's IP address to make a connection to that Web server. It can resolve the name in three ways: by broadcasting, by consulting the locally stored HOSTS text file, or by contacting a DNS server.


Microsoft Corporation - Windows Internet Explorer
http://www.microsoft.com/en/us/default.aspx
http://207.46.19.254/en/us/default.aspx
Harness the power
Experience out-of-the-box supercomputing performance
Windows HPC Server 2008
Try it free
Internet Explorer 8
Tech·Ed 2009

• Figure 10.12 Any TCP/IP-savvy program accepts either an IP address or an FQDN.

To *broadcast* for name resolution, the host sends a message to all the machines on the network, saying something like, "Hey! If your name is JOESCOMPUTER, please respond with your IP address." All the networked hosts receive that packet, but only JOESCOMPUTER responds with an IP address. Broadcasting works fine for small networks, but it is limited because it cannot provide name resolution across routers. Routers do not forward broadcast messages to other networks, as illustrated in Figure 10.13.

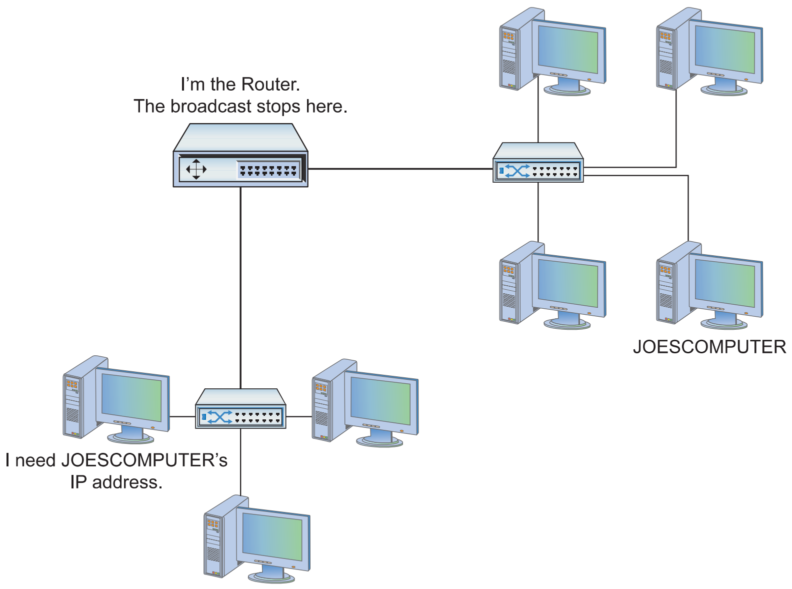
As discussed earlier, a HOSTS file functions like a little black book, listing the names and addresses of machines on a network, just like a little black book lists the names and phone numbers of people. A typical HOSTS file would look like this:

```
109.54.94.197      stephen.totalsem.com
138.125.163.17     roger.totalsem.com
127.0.0.1          localhost
```

Notice that the name "localhost" appears in the HOSTS file as an alias for the loopback address, 127.0.0.1.

The final way to resolve a name to an IP address is to use DNS. Let's say you type "www.microsoft.com" in your Web browser. To resolve the name www.microsoft.com, the host contacts its DNS server and requests the IP address, as shown in Figure 10.14.

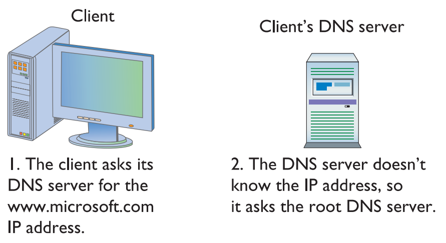
To request the IP address of www.microsoft.com, your PC needs the IP address of its DNS server. You must enter DNS information into your system. DNS server data is part of the critical basic IP information such as your IP address, subnet mask, and default gateway, so it's usually entered at the same time as the other IP information. You configure DNS in Windows Vista using the Internet Protocol Version 4 (TCP/IPv4) Properties dialog



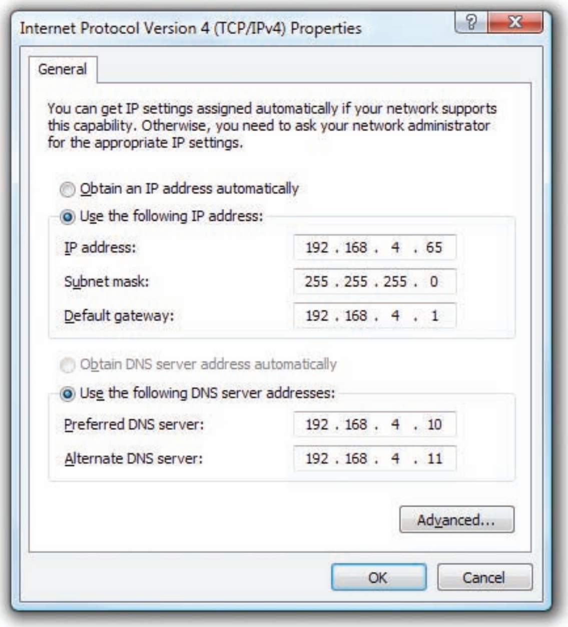
• **Figure 10.13** Routers don't forward broadcasts!

box. Figure 10.15 shows the DNS settings for my system. Note that I have more than one DNS server setting; the second one is a backup in case the first one isn't working. Two DNS settings is not a rule, however, so don't worry if your system shows only one DNS server setting, or perhaps more than two.

Every operating system has a way for you to enter DNS server information. In Linux you can directly edit the /etc/resolv.conf file


Client
Client's DNS server
1. The client asks its DNS server for the www.microsoft.com IP address.
2. The DNS server doesn't know the IP address, so it asks the root DNS server.

• **Figure 10.14** A host contacts its local DNS server.



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

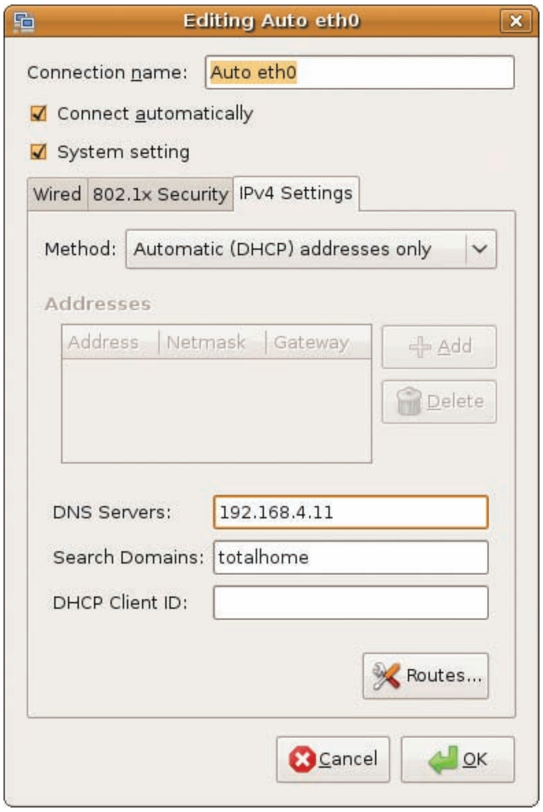
○ Obtain an IP address automatically
● Use the following IP address:

IP address:	192 . 168 . 4 . 65
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	192 . 168 . 4 . 1

○ Obtain DNS server address automatically
● Use the following DNS server addresses:

Preferred DNS server:	192 . 168 . 4 . 10
Alternate DNS server:	192 . 168 . 4 . 11

• **Figure 10.15** DNS information in Windows



Editing Auto eth0

Connection name: Auto eth0

☑ Connect automatically

☑ System setting

Wired | 802.1x Security | IPv4 Settings

Method: Automatic (DHCP) addresses only

Addresses

Address	Netmask	Gateway

DNS Servers: 192.168.4.11

Search Domains: totalhome

DHCP Client ID:

• **Figure 10.16** Entering DNS information in Ubuntu

using a text editor. Just about every version of Linux has some form of graphical editor as well to make this an easy process. Figure 10.16 shows Ubuntu's Network Configuration utility.

Every operating system also comes with a utility you can use to verify the DNS server settings. You can see your current DNS server settings in Windows by typing `ipconfig /all` at the command prompt (Figure 10.17). In UNIX/Linux type `cat /etc/ resolv.conf`.

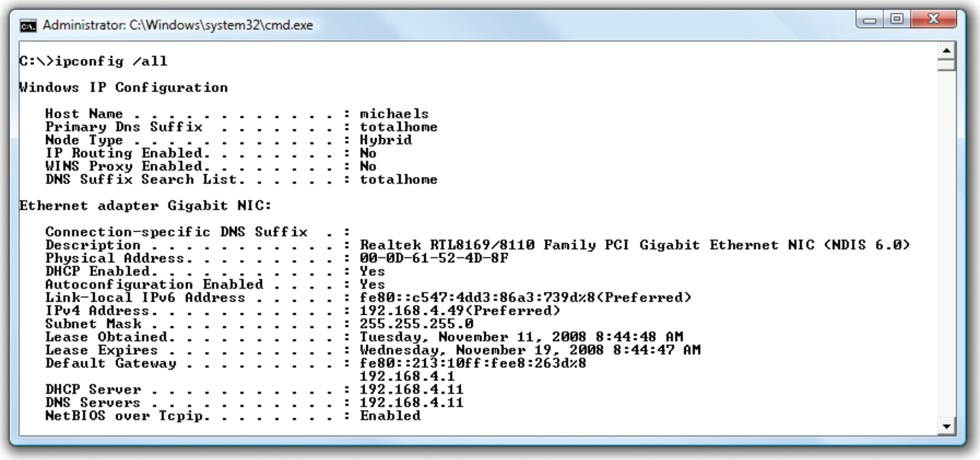
Now that you know how your system knows the DNS server's IP address, let's return to the DNS process.

The DNS server receives the request for the IP address of www.microsoft.com from your client computer. At this point, your DNS server checks a cache of previously resolved FQDNs to see if www.microsoft.com is there (Figure 10.18). In this case, www.microsoft.com is not in the cache.

It's now time for your DNS server to get to work. The local DNS server may not know the address for www.microsoft .com, but it does know the addresses of the DNS root servers. The root servers, maintained by InterNIC, know all the addresses of the top-level domain DNS servers. The root servers don't know the address of www.microsoft.com, but they do know the address of the DNS servers in charge of all .com addresses. The root servers send your DNS server an IP address for a .com server (Figure 10.19).

The .com DNS server also doesn't know the address of www.microsoft.com, but it knows the IP address of the microsoft.com DNS server. It sends that IP address to your root server (Figure 10.20).

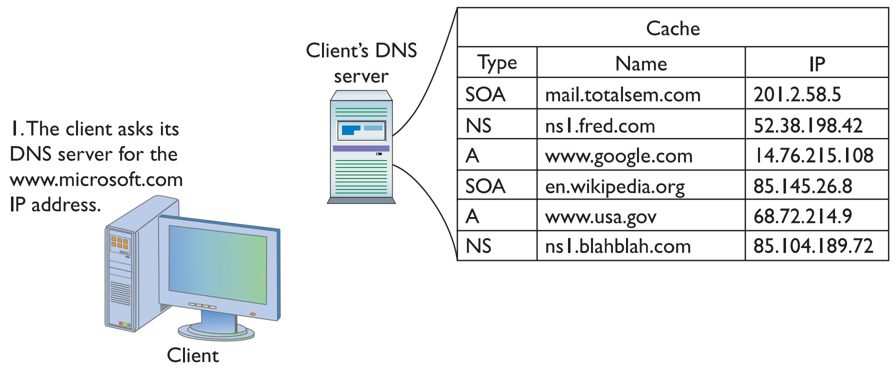
The microsoft.com server does know the IP address of www .microsoft.com, and can send that information back to the local DNS server. Figure 10.21 shows the process of resolving an FQDN into an IP address.



Administrator: C:\Windows\system32\cmd.exe

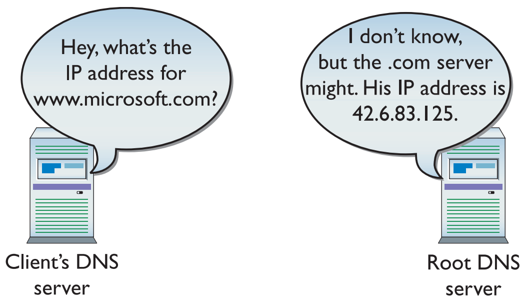
```
C:\>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : michaels
   Primary Dns Suffix  . . . . . . . : totalhome
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : totalhome

Ethernet adapter Gigabit NIC:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek RTL8169/8110 Family PCI Gigabit Ethernet NIC (NDIS 6.0)
   Physical Address. . . . . . . . . : 00-0D-61-52-4D-8F
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::c547:4dd3:86a3:739d%8(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.4.49(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Tuesday, November 11, 2008 8:44:48 AM
   Lease Expires . . . . . . . . . . : Wednesday, November 19, 2008 8:44:47 AM
   Default Gateway . . . . . . . . . : fe80::213:10ff:fee8:263d%8
                                       192.168.4.1
   DHCP Server . . . . . . . . . . . : 192.168.4.11
   DNS Servers . . . . . . . . . . . : 192.168.4.11
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

• **Figure 10.17** IPCONFIG /ALL showing DNS information in Windows

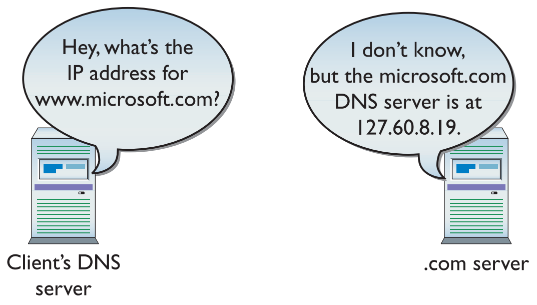

Client's DNS server
1. The client asks its DNS server for the www.microsoft.com IP address.
Client

Cache		
Type	Name	IP
SOA	mail.totalsem.com	201.2.58.5
NS	ns1.fred.com	52.38.198.42
A	www.google.com	14.76.215.108
SOA	en.wikipedia.org	85.145.26.8
A	www.usa.gov	68.72.214.9
NS	ns1.blahblah.com	85.104.189.72

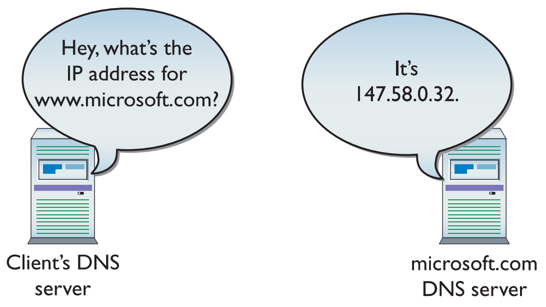
• **Figure 10.18** Checking the DNS cache


Hey, what's the IP address for www.microsoft.com?
I don't know, but the .com server might. His IP address is 42.6.83.125.
Client's DNS server
Root DNS server

• **Figure 10.19** Talking to a root server


Hey, what's the IP address for www.microsoft.com?
I don't know, but the microsoft.com DNS server is at 127.60.8.19.
Client's DNS server
.com server

• **Figure 10.20** Talking to .com server



• **Figure 10.21** Talking to microsoft.com DNS server

Cross Check

HTTP Process

You learned the specifics of HTTP in Chapter 9, "TCP/IP Applications," so check your memory now. Is the HTTP process connectionless or connection-oriented? At what OSI layers does the process happen?

Now that your DNS server has the IP address for www.microsoft.com, it stores a copy in its cache and sends the IP information to your PC. Your Web browser then begins the HTTP request to get the Web page.

Your computer also keeps a cache of recently resolved FQDNs. In Windows, for example, open a command prompt and type `ipconfig /displaydns` to see them. Here's a small part of the results of IPCONFIG /DISPLAYDNS:

```
    gizmodo.com
    ----------------------------------------
    Record Name . . . . . : gizmodo.com
    Record Type . . . . . : 1
    Time To Live  . . . . : 70639
    Data Length . . . . . : 4
    Section . . . . . . . : Answer
    A (Host) Record . . . : 69.60.7.199
    ftp.totalsem.com
    ----------------------------------------
    Record Name . . . . . : ftp.totalsem.com
    Record Type . . . . . : 1
    Time To Live  . . . . : 83733
    Data Length . . . . . : 4
    Section . . . . . . . : Answer
    A (Host) Record . . . : 209.29.33.25
C:\>
```

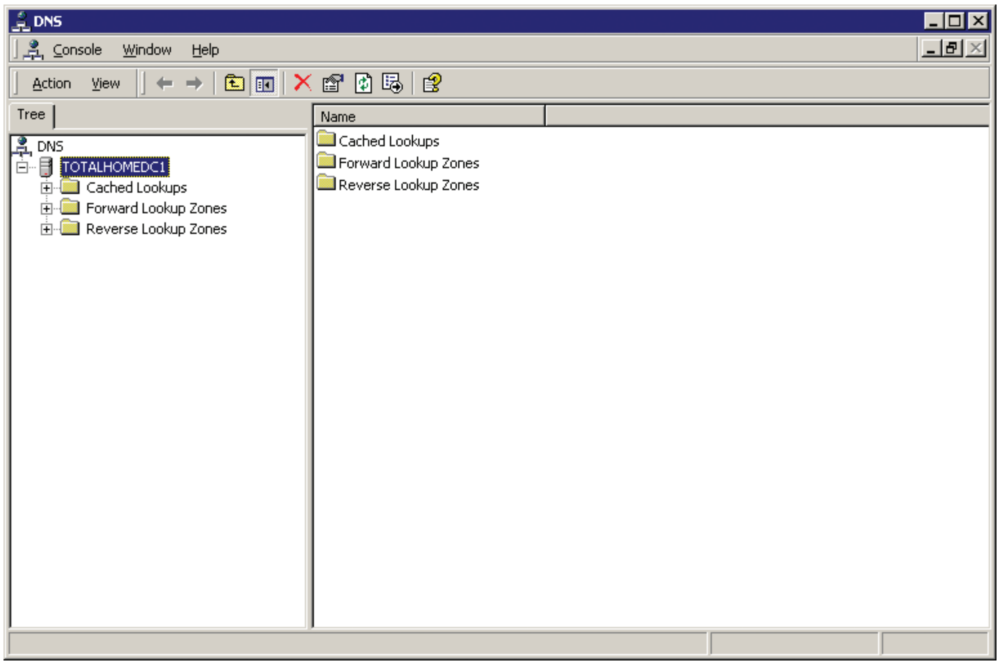
DNS Servers

We've been talking about DNS servers for so long, I feel I'd be untrue to my vision of an All-in-One book unless we took at least a quick peek at a DNS server in action. Lots of operating systems come with built-in DNS server software, including Windows Server and just about every version of UNIX/Linux. There are also a number of third-party DNS server programs for virtually any operating system. I'm going to use the DNS server program that comes with Microsoft Windows Server 2003 primarily because (1) it takes the prettiest screen snapshots and (2) it's the one I use here at the office. You access the Windows DNS server by selecting **Start | Administrative Tools | DNS**. When you first open the DNS server, there's not much to see other than the name of the server itself. In this case, Figure 10.22 shows a server, imaginatively named TOTALHOMEDC1.

The most popular DNS server tool used in UNIX/Linux systems is called BIND.

The DNS server has three folder icons visible: Cached Lookups, Forward Lookup Zones, and Reverse Lookup Zones. Let's look at all three.

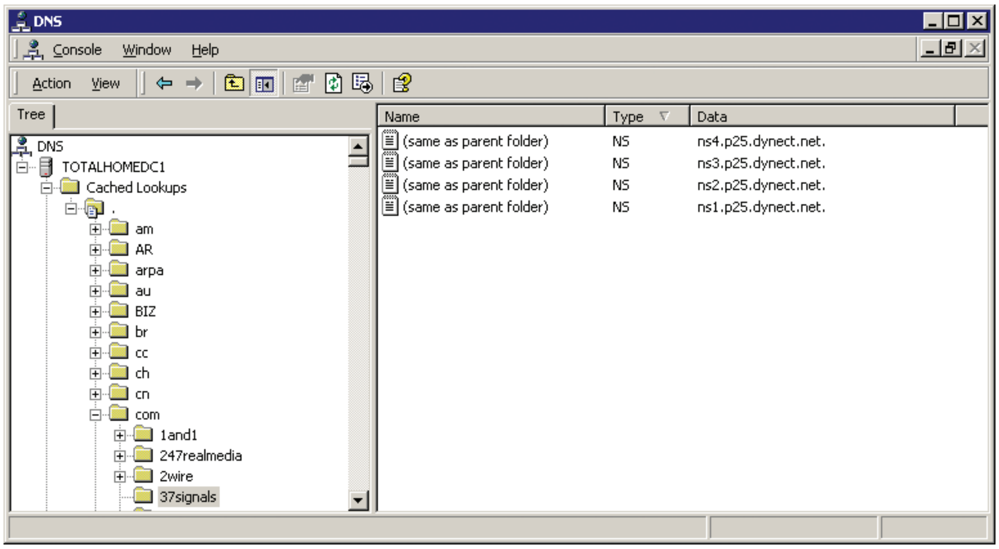
When you open the tree on a Windows DNS server, the first folder you see is called Cached Lookups. Every DNS server keeps a list of **cached lookups**—that is, all the IP addresses it has already resolved—so it won't have to re-resolve an FQDN it has already checked. There is a limit to the size of the cache, of course, and you can also set a limit on how long the DNS server holds cache entries. Windows does a nice job of separating these



• **Figure 10.22** DNS server main screen

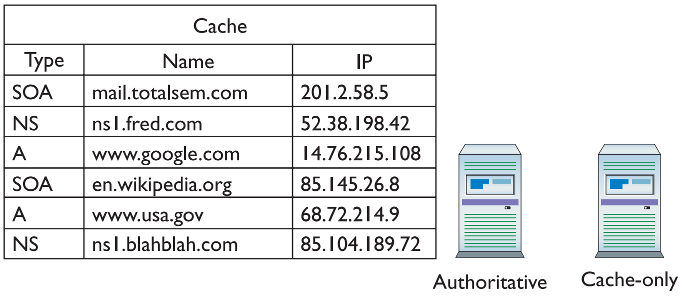
cached addresses by placing all cached lookups in little folders that share the first name of the top-level domain with subfolders that use the second-level domain (Figure 10.23). This sure makes it easy to see where folks have been Web browsing!

Now let's watch an actual DNS server at work. Basically, you choose to configure a DNS server to work in one of two ways: as an authoritative DNS server or as a cache-only DNS server. Authoritative DNS servers store



Name	Type	Data
(same as parent folder)	NS	ns4.p25.dynect.net.
(same as parent folder)	NS	ns3.p25.dynect.net.
(same as parent folder)	NS	ns2.p25.dynect.net.
(same as parent folder)	NS	ns1.p25.dynect.net.

• **Figure 10.23** Inspecting the DNS cache



Cache		
Type	Name	IP
SOA	mail.totalsem.com	201.2.58.5
NS	ns1.fred.com	52.38.198.42
A	www.google.com	14.76.215.108
SOA	en.wikipedia.org	85.145.26.8
A	www.usa.gov	68.72.214.9
NS	ns1.blahblah.com	85.104.189.72

• **Figure 10.24** Authoritative vs. cache-only DNS server

Microsoft DNS servers use a folder analogy to show lookup zones even though they are not true folders.

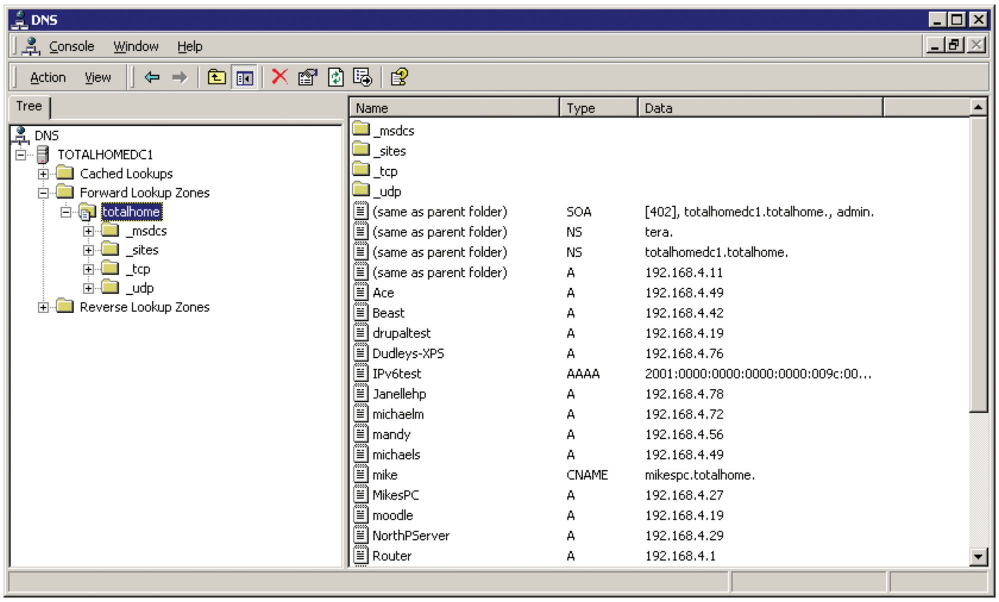
IP addresses and FQDNs of systems for a particular domain or domains. **Cache-only DNS servers** do not store any FQDNs and are only used to talk to other DNS servers to resolve IP addresses for DNS clients (Figure 10.24).

The IP addresses and FQDNs for the computers in a domain are stored in special storage areas called **forward lookup zones**. Forward lookup zones are the most important part of any DNS server. Figure 10.25 shows the DNS server for my small corporate network. My domain is called "totalhome." I can get away with a domain name that's not Internet legal because none of these computers are visible on the Internet. The totalhome domain only works on my local network for local computers to find each other. I have created a forward lookup zone called totalhome.

Let's look at the contents of the totalhome domain. First, notice a number of folders: _msdcs, _sites, _tcp, and _udp. These folders are unique to Microsoft DNS servers and you'll see what they do in a moment. For now ignore them and concentrate on the individual computer listings. Every forward lookup zone requires a Start of Authority, the single DNS server in charge. The record called SOA (Start of Authority) in the folder totalhome indicates that my server is the authoritative DNS server for a domain called totalhome. You can even see a few of the systems in that domain (note to hackers: these are fake, so don't bother). A tech looking at this would know that totalhomedc1.totalhome is the authoritative DNS server for the totalhome domain. The **NS records** are all of the DNS servers for totalhome. Note that totalhome has two DNS servers: totalhomedc1.totalhome and tera. Tera is not a member of the totalhome domain. In fact, tera isn't a member of *any* domain. A DNS server does not have to be a member of a domain to be a name server for that domain.

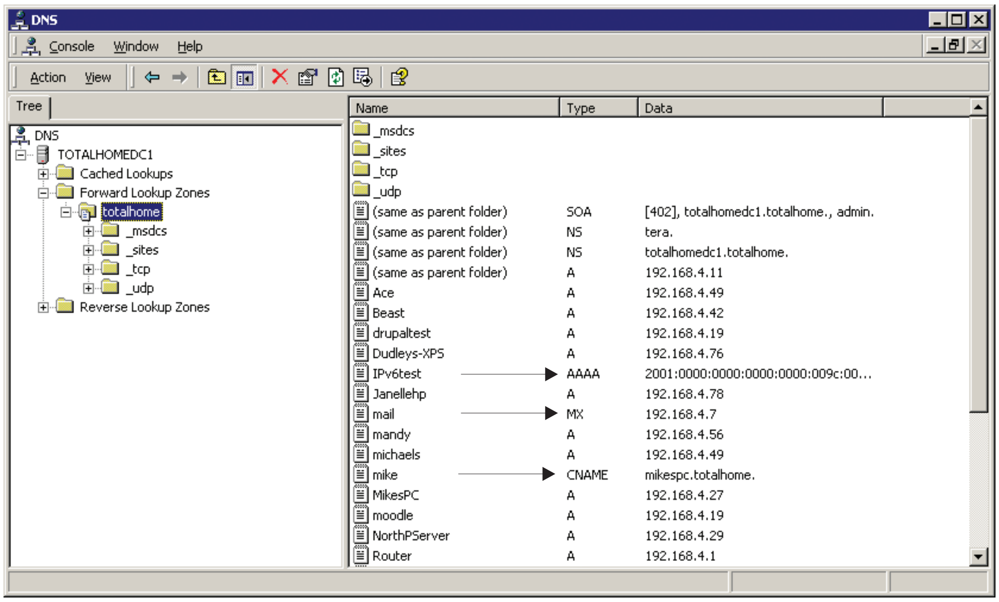
Having two DNS servers insures that if one fails, the totalhome domain will continue to have a DNS server. The **A records** in the folder are the IP addresses and names of all the systems on the totalhome domain.

Every DNS forward lookup zone will have one SOA and at least one NS record. In the vast majority of cases, a forward lookup zone will have some



Name	Type	Data
_msdcs		
_sites		
_tcp		
_udp		
(same as parent folder)	SOA	[402], totalhomedc1.totalhome., admin.
(same as parent folder)	NS	tera.
(same as parent folder)	NS	totalhomedc1.totalhome.
(same as parent folder)	A	192.168.4.11
Ace	A	192.168.4.49
Beast	A	192.168.4.42
drupaltest	A	192.168.4.19
Dudleys-XPS	A	192.168.4.76
IPv6test	AAAA	2001:0000:0000:0000:0000:009c:00...
Janellehp	A	192.168.4.78
michaelm	A	192.168.4.72
mandy	A	192.168.4.56
michaels	A	192.168.4.49
mike	CNAME	mikespc.totalhome.
MikesPC	A	192.168.4.27
moodle	A	192.168.4.19
NorthPServer	A	192.168.4.29
Router	A	192.168.4.1

• **Figure 10.25** Forward lookup zone totalhome



Name	Type	Data
_msdcs		
_sites		
_tcp		
_udp		
(same as parent folder)	SOA	[402], totalhomedc1.totalhome., admin.
(same as parent folder)	NS	tera.
(same as parent folder)	NS	totalhomedc1.totalhome.
(same as parent folder)	A	192.168.4.11
Ace	A	192.168.4.49
Beast	A	192.168.4.42
drupaltest	A	192.168.4.19
Dudleys-XPS	A	192.168.4.76
IPv6test	AAAA	2001:0000:0000:0000:0000:009c:00...
Janellehp	A	192.168.4.78
mail	MX	192.168.4.7
mandy	A	192.168.4.56
michaels	A	192.168.4.49
mike	CNAME	mikespc.totalhome.
MikesPC	A	192.168.4.27
moodle	A	192.168.4.19
NorthPServer	A	192.168.4.29
Router	A	192.168.4.1

• **Figure 10.26** Less common DNS record types

number of A records. But there are a number of other records you may or may not see in your standard DNS server. Look at Figure 10.26 for these less common types of DNS records: CNAME, MX, and AAAA.

A **canonical name (CNAME)** record acts like an alias. My computer's name is mikespc.totalhome, but you can also now use mike.totalhome to reference that computer. A PING of mike.totalhome returns the following:

```
C:\>ping mike.totalhome
Pinging mikespc.totalhome [192.168.4.27] with 32 bytes of data:
Reply from 192.168.4.27: bytes=32 time=2ms TTL=128
Reply from 192.168.4.27: bytes=32 time<1ms TTL=128
```
(rest of PING results deleted)

If your computer is a member of a domain and you are trying to access another computer in that domain, you can even skip the domain name because your PC will simply add it back:

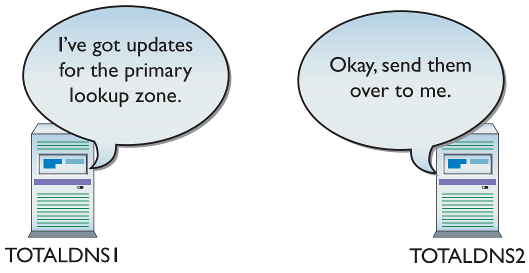
```
C:\>ping mike
Pinging mikespc.totalhome [192.168.4.27] with 32 bytes of data:
Reply from 192.168.4.27: bytes=32 time<1ms TTL=128
Reply from 192.168.4.27: bytes=32 time<1ms TTL=128
```
(rest of PING results deleted)

MX records are used exclusively by SMTP servers to determine where to send mail. I have an in-house SMTP server on a computer I cleverly called mail. If other SMTP servers wanted to send mail to mail.totalhome (they can't because the SMTP server isn't connected to the Internet and lacks a legal FQDN), they would use DNS to locate the mail server.

MX stands for Mail eXchanger.

AAAA records are for a newer type of IP addressing called IPv6. You'll learn a lot more about IPv6 in Chapter 13.

There are two common types of forward lookup zones: a primary zone and a secondary zone. **Primary zones** are created on the DNS server that will act as the SOA for that zone. **Secondary zones** are created on other DNS


I've got updates for the primary lookup zone.
Okay, send them over to me.
TOTALDNS1
TOTALDNS2

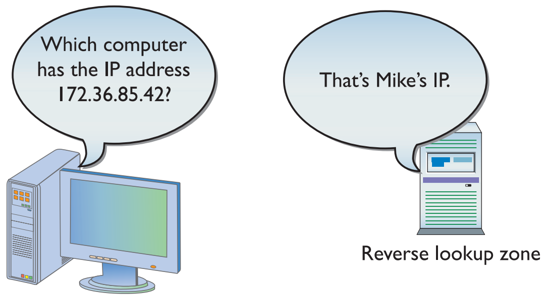
• **Figure 10.27** Two DNS servers with updating taking place

servers to act as backups to the primary zone. It's standard practice to have at least two DNS servers for any forward lookup zone: one primary and one secondary. Even in my small network I have two DNS servers: TOTALDNS1, which runs the primary zone, and TOTALDNS2, which runs a secondary zone (Figure 10.27). Any time a change is placed on TOTALDNS1, TOTALDNS2 is quickly updated.

A **reverse lookup zone** (Figure 10.28) enables a system to determine an FQDN by knowing the IP address; that is, it does the exact reverse of what DNS normally does! Reverse lookup zones take a network ID, reverse it, and add the term "in-addr-arpa" to create the zone. A few low-level functions (like mail) and some security programs use reverse lookup zones, so DNS servers provide them. In most cases the DNS server will ask you if you want to make a reverse lookup zone when you make a new forward lookup zone. When in doubt, make one. If you don't need it, it won't cause any trouble.

If you're looking at a Windows server and adding a new forward lookup zone, you'll see a third type called an Active Directory–integrated forward lookup zone. I'll cover those in just a moment.

Microsoft added some wrinkles to DNS servers with the introduction of Windows 2000 Server, and each subsequent version of Windows Server retains the wrinkles. Windows Server can do cached lookups, primary and secondary forward lookup zones, and reverse lookup zones, just like UNIX/Linux DNS servers. But it also has a Windows-only type of forward lookup zone called an Active Directory–integrated zone.



• **Figure 10.28** Reverse lookup zone

Enter Windows

DNS works beautifully for any TCP/IP application that needs an IP address of another computer, but it has one glaring weakness: you need to add A records to the DNS server manually. This can be a problem, especially in a world where you have many DHCP clients whose IP addresses may change from time to time. Interestingly, it was a throwback to an old Microsoft Windows protocol that fixed this and a few other problems all at the same time.

Even though TCP/IP was available, back in the 1980s Microsoft popularized another networking protocol called **NetBIOS/NetBEUI**. NetBIOS/NetBEUI was pretty simplistic compared to TCP/IP. It had a very simple naming convention (the NetBIOS part) that used broadcasts. When a computer booted up, it just told the world its name (Figure 10.29).

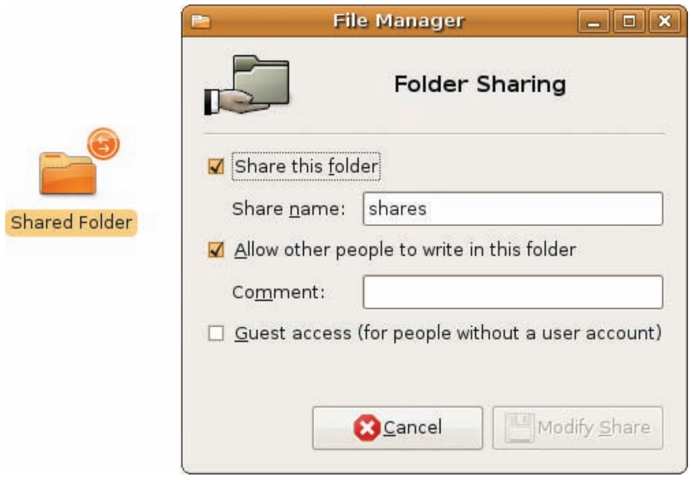
• **Figure 10.29** NetBIOS broadcast

NetBIOS/NetBEUI was suitable only for small networks. It provided no logical addressing like IP addresses; you just had to remember the NetBIOS name and the MAC address. NetBIOS/NetBEUI was almost exclusively used to share folders and printers. There was no such thing as Telnet or the Web with NetBIOS/NetBEUI, but it worked well for what it did at the time.

By the mid-1990s Microsoft realized that the world was going to TCP/IP and it needed to switch too. Instead of dumping NetBIOS/NetBEUI entirely, Microsoft designed a new TCP/IP protocol that enabled it to keep using the NetBIOS names but dump the ancient NetBEUI protocol and instead run NetBIOS on top of IP. In essence Microsoft created its own name resolution protocol that had nothing to do with DNS!

Microsoft managed to crowbar the NetBIOS naming system into DNS basically by making the NetBIOS name the DNS name. Technically, NetBIOS no longer exists, but the overlying protocol that used it to share folders and printers is still very much alive. This protocol was originally called Server Message Block (SMB); the current version is called **Common Internet File System (CIFS)**.

Microsoft has used DNS names with the SMB/CIFS protocol to provide folder and printer sharing in small TCP/IP networks. SMB/CIFS is so popular that other operating systems have adopted support for SMB/CIFS. UNIX/Linux systems (including Mac OS X) come with the very popular Samba, the most popular tool for making non-Windows systems act like Windows computers (Figure 10.30).



[✓] Share this folder

Share name: shares

[✓] Allow other people to write in this folder

Comment:

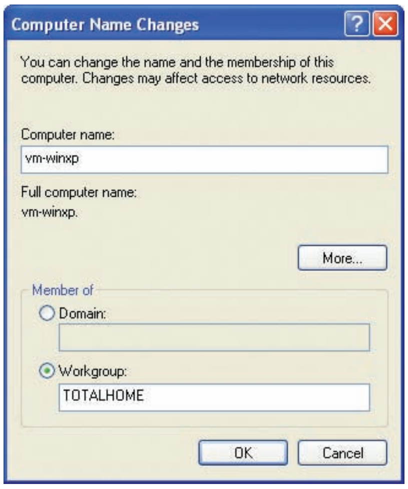
[ ] Guest access (for people without a user account)

• **Figure 10.30** Samba on Ubuntu (it's so common that the OS doesn't even use the term in the dialog box)

Living with the Legacy of CIFS CIFS makes most small networks live in a two-world name resolution system. When your computer wants to access another computer's folders or files, it uses a simple CIFS broadcast to get the name. If that same computer wants to do anything "Internety," it uses its DNS server. Both CIFS and DNS live together perfectly well and, while there are many alternatives to this dual name resolution world, the vast majority of us are happy with this relationship.

Well, almost happy, except for one little item: CIFS organizes your computers into groups. There are three types of groups: workgroup, Windows domain, and Active Directory. A **workgroup** is just a name that organizes a group of computers. A computer running Windows (or another operating system running Samba) joins a workgroup, as shown in Figure 10.31. Once a computer joins a workgroup, this does little more than organize all the computers in the Network/My Network Places folder, as shown in Figure 10.32.

A **Windows domain** is a group of computers controlled by a computer running Windows Server. This Windows Server computer is configured as a domain controller. You would then have your computers join the domain.



You can change the name and the membership of this computer. Changes may affect access to network resources.

Computer name: vm-winxp

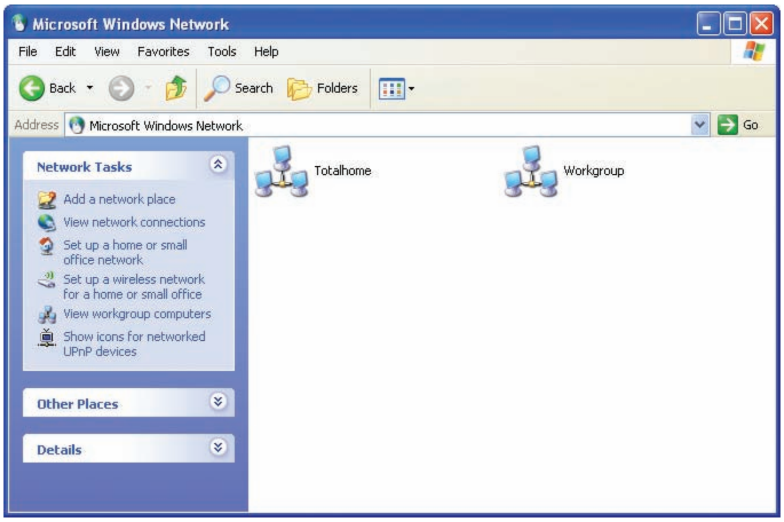
Full computer name: vm-winxp.

Member of

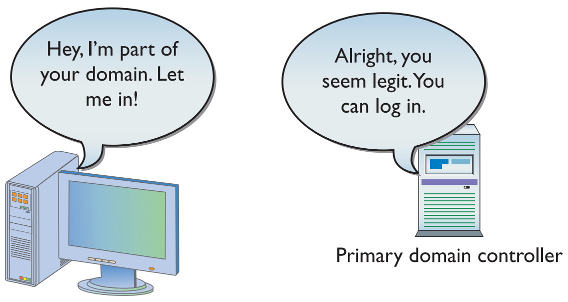
( ) Domain:

(•) Workgroup: TOTALHOME

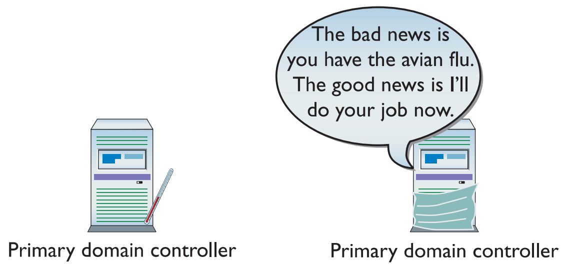
• **Figure 10.31** Joining a workgroup


Microsoft Windows Network
File Edit View Favorites Tools Help
Back
Search
Folders
Address Microsoft Windows Network
Go
Network Tasks
Add a network place
View network connections
Set up a home or small office network
Set up a wireless network for a home or small office
View workgroup computers
Show icons for networked UPnP devices
Other Places
Details
Totalhome
Workgroup

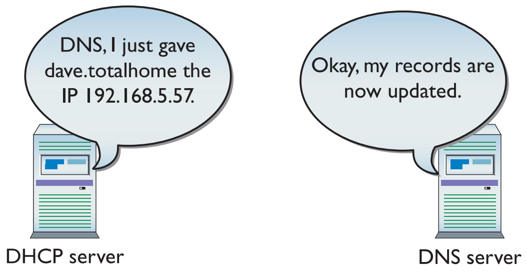
• **Figure 10.32** Two workgroups in Network folder


Hey, I'm part of your domain. Let me in!
Alright, you seem legit. You can log in.
Primary domain controller

• Figure 10.33 Logging into the domain


The bad news is you have the avian flu. The good news is I'll do your job now.
Primary domain controller
Primary domain controller

• Figure 10.34 If one domain controller goes down, another automatically takes over.


DNS, I just gave dave.totalhome the IP 192.168.5.57.
Okay, my records are now updated.
DHCP server
DNS server

• Figure 10.35 Updating DNS information in Windows

All the computers within a domain authenticate to the domain controller when they log in. This is a very powerful part of the control Windows gives you over who can access what on your network (Figure 10.33).

Note that a Windows domain is not the same as a DNS domain. In the early days, a Windows domain didn't even have a naming structure that resembled the DNS hierarchically organized structure.

Microsoft eventually revamped its domain controllers to work as part of DNS, and Windows domains now use DNS for their names. A Windows domain must have a true DNS name. DNS domains that are not on the Internet should use the top-level name .local (although you can cheat, as I do on my totalhome network, and not use it).

On a bigger scale, a Windows network can get complicated, with multiple domains connecting over long distances. To help organize this, Windows uses a type of super domain called Active Directory. An **Active Directory** is an organization of related computers that shares one or more Windows domains. Windows domain controllers are also DNS servers. The beauty of Active Directory is that there's no single domain controller: all of the domain controllers are equal partners and any domain controller can take over if one domain controller fails (Figure 10.34).

Active Directory–Integrated Zones Now that you have an understanding of Windows domains and Active Directory, let's return to forward lookup zones and DNS. You have enough information to understand an Active Directory–integrated zone.

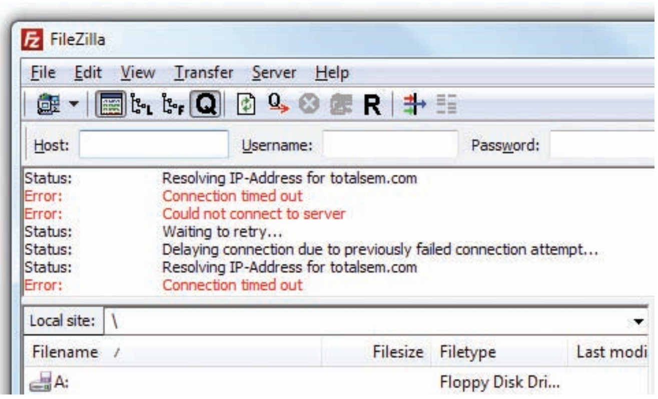
A standard primary zone stores the DNS information in text files. You then need secondary zones on other DNS servers to back up that server. If the primary DNS server goes down, the secondary servers can resolve FQDNs, but you can't add any new records. Nothing can be updated until the primary DNS server comes back up. In an Active Directory–integrated zone, all of the domain controllers (which are all also DNS servers) are equal and the whole DNS system is not reliant on a single DNS server.

Last, Windows DHCP servers will automatically update all the client information in the Active Directory–integrated zone. If a computer gets a DHCP lease, the DHCP server (or, in the latest versions of Windows, the client itself) tells the DNS server that it has new DNS information (Figure 10.35). Non-Windows systems can only do this using Windows clients (like Samba).

Troubleshooting DNS

As I mentioned earlier, most DNS problems result from a problem with the client systems. This is because DNS servers rarely go down, and if they do, most clients have a secondary DNS server setting that enables them to continue to resolve DNS names. DNS servers have been known to fail, however, so it's important to know when the problem is the client system, and when you can complain to the person in charge of your DNS server. All of the tools you're about to see come with every operating system that supports TCP/IP, with the exception of the **IPCONFIG** commands, which I'll mention when we get to them.

So how do you know when to suspect DNS as a problem on your network? Well, just about everything you do on an IP network depends on DNS to find the right system to talk to for whatever job the application does. E-mail clients use DNS to find their e-mail servers, FTP clients use DNS for their servers, Web browsers use DNS to find Web servers, and so on. The first clue is usually a user calling you and saying he's getting a "server not found" error. Server not found errors look different on different applications, but you can count on something in there that says in effect "server not found." Figure 10.36 shows how this error appears in an FTP client.



```
Status:    Resolving IP-Address for totalsem.com
Error:     Connection timed out
Error:     Could not connect to server
Status:    Waiting to retry...
Status:    Delaying connection due to previously failed connection attempt...
Status:    Resolving IP-Address for totalsem.com
Error:     Connection timed out
```

• **Figure 10.36** DNS error

Before you start testing, you need to eliminate any DNS caches on the local system. If you're running Windows, run the `ipconfig /flushdns` command now. In addition, most Web browsers also have caches, so you can't use a Web browser for any testing. In such cases, it's time to turn to the **PING** command.

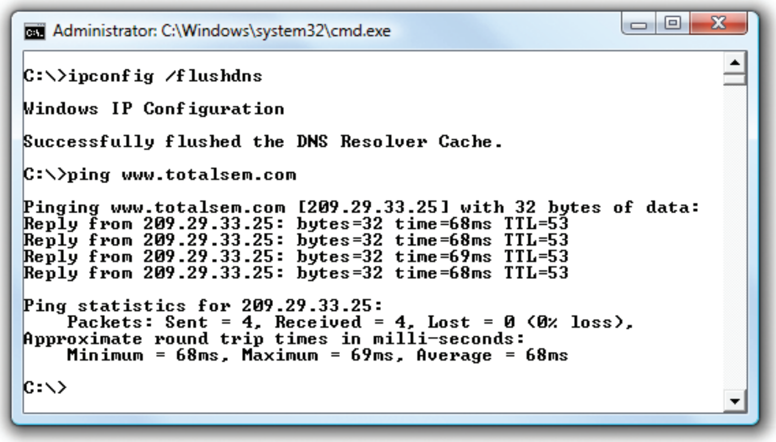
PING is your best friend when you're testing DNS. Run `ping` from a command prompt, followed by the name of a well-known Web site, such as www.microsoft.com. Watch the output carefully to see if you get an IP address. You may get a "request timed out" message, but that's fine; you just want to see if DNS is resolving FQDNs into IP addresses (Figure 10.37).

PING is your friend. If you can PING an IP address but not the name associated with that address, check DNS.

If you get a "server not found" error, you need to ping again using just an IP address. Most network techs keep the IP address of a known server in their heads. If you don't have one memorized, try 74.125.95.99. If PING works with the IP address but not with the Web site name, you know you have a DNS problem.

Once you've determined that there's a DNS problem, check to make sure your system has the correct DNS server entry. Again, this information is something you should keep around. I can tell you the DNS server IP address for every Internet link I own—two in the office, one at the house, plus two dial-ups I use on the road. You don't have to memorize the IP addresses, but you should have all the critical IP information written down. If that isn't the problem, run `ipconfig /all` to see if those DNS settings are the same as the ones in the server; if they aren't, you may need to refresh your DHCP settings. I'll show you how to do that next.

If you have the correct DNS settings for your DNS server and the DNS settings in IPCONFIG /ALL match those settings, you



```
C:\>ipconfig /flushdns

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.

C:\>ping www.totalsem.com

Pinging www.totalsem.com [209.29.33.25] with 32 bytes of data:
Reply from 209.29.33.25: bytes=32 time=68ms TTL=53
Reply from 209.29.33.25: bytes=32 time=68ms TTL=53
Reply from 209.29.33.25: bytes=32 time=69ms TTL=53
Reply from 209.29.33.25: bytes=32 time=68ms TTL=53

Ping statistics for 209.29.33.25:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 68ms, Maximum = 69ms, Average = 68ms

C:\>
```

• **Figure 10.37** Using PING to check DNS can assume the problem is with the DNS server itself. There's a popular command for working with DNS servers called **NSLOOKUP** (name server lookup). NSLOOKUP comes with all operating systems. It's a handy tool that advanced techs use to query the functions of DNS servers.

NSLOOKUP is an amazingly complex program that you run from a command prompt. With NSLOOKUP, you can (assuming you have the permission) query all types of information from a DNS server and change how your system uses DNS. While most of these commands are far outside the scope of the CompTIA Network+ exam, there are a few places where NSLOOKUP makes for a great basic tool. For instance, just running `nslookup` alone from a command prompt shows you some output similar to the following:

```
C:\>nslookup
Default Server:  totalhomedc2.totalhome
Address:  192.168.4.155

>
```

Make sure you know how to use NSLOOKUP to determine if a DNS server is active!

Running NSLOOKUP gives me the IP address and the name of my default DNS server. If I got an error at this point, perhaps a "server not found" error, I would know that either my primary DNS server is down or I might not have the correct DNS server information in my DNS settings. I can attach to any DNS server by typing `server`, followed by the IP address or the domain name of the DNS server:

```
> server totalhomedc1
Default Server:  totalhomedc1.totalhome
Addresses:  192.168.4.157, 192.168.4.156
```

This new server has two IP addresses; it probably has two multihomed NICs to ensure that there's a backup in case one NIC fails. If I get an error on one DNS server, I use NSLOOKUP to check for another DNS server. I can then switch to that server in my TCP/IP settings as a temporary fix until my DNS server comes back up.

Those using UNIX/Linux have an extra DNS tool called **domain information grouper (DIG)**. DIG is very similar to NSLOOKUP, but it runs noninteractively. In NSLOOKUP, you're in the command until you type `exit`; NSLOOKUP even has its own prompt. The DIG tool, on the other hand, is not interactive—you ask it a question, it answers the question, and it puts you back at a command prompt, with nothing to exit. When you run DIG, you tend to get a large amount of information. The following is a sample of a `dig` command run from a Linux prompt:

```
 [mike@localhost]$dig -x 13.65.14.4
 ; <<>> DiG 8.2 <<>> -x
 ;; res options: init recurs defnam dnsrch
 ;; got answer:
 ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4
 ;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2,
ADDITIONAL: 2
 ;; QUERY SECTION:
 ;;      4.14.65.13.in-addr.arpa, type = ANY, class = IN
 ;; ANSWER SECTION:
```

```
 4.14.65.13.in-addr.arpa.  4H IN PTR
server3.houston.totalsem.com.
 ;; AUTHORITY SECTION:
 65.14.4.in-addr.arpa.  4H IN NS  kernel.risc.uni-linz.ac.at.
 65.14.4.in-addr.arpa.  4H IN NS  kludge.risc.uni-linz.ac.at.
 ;; ADDITIONAL SECTION:
 kernel.risc.uni-linz.ac.at.  4H IN A  193.170.37.225
 kludge.risc.uni-linz.ac.at.  4H IN A  193.170.37.224
 ;; Total query time: 1 msec
 ;; FROM: kernel to SERVER: default -- 127.0.0.1
 ;; WHEN: Thu Feb 10 18:03:41 2000
 ;; MSG SIZE  sent: 44  rcvd: 180
[mike@localhost]$
```

WINS

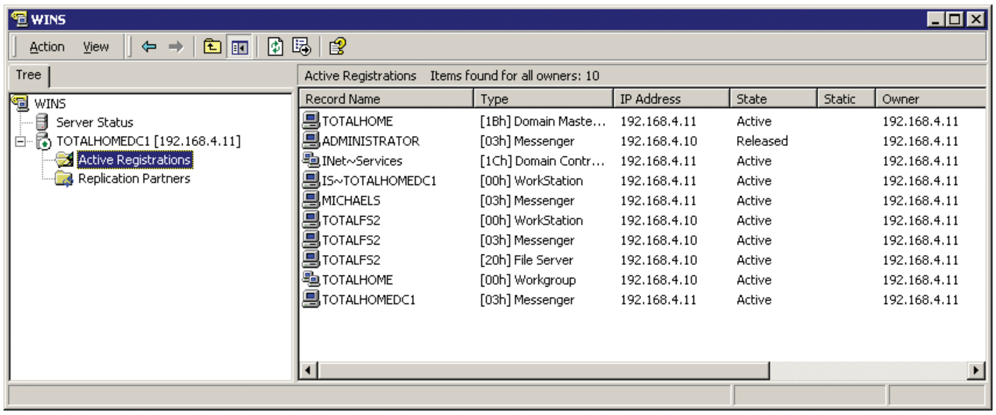
Even though current versions of Windows use either DNS or CIFS names, NetBIOS names can still appear in older versions of Windows like Windows 9*x* or some versions of Windows 2000. A Windows NetBIOS system claims a NetBIOS name for itself simply by broadcasting out to the rest of the network. As long as no other system is already using that name, it works just fine. Of course, broadcasting can be a bit of a problem for routers and such, but this example presumes a single network on the same wire, so it's okay in this context.

NetBIOS was invented way back in the early 1980s. Microsoft had a big investment in NetBIOS, and had to support a large installed base of systems, so even after NetBEUI began to lose market share to TCP/IP, Microsoft had to continue to support NetBIOS or incur the wrath of millions of customers. What happened next seems in retrospect more a comedy than the machinations of the most powerful software company in the world. Microsoft did something that should not have been possible: it redesigned NetBIOS to work with TCP/IP. Let's look at some of the strategies and techniques Microsoft used to make NetBIOS and TCP/IP coexist on the same network.

Eventually, Microsoft came up with CIFS, as you know from earlier in the chapter, and made NetBIOS DNS-compatible. But it tried a couple of things first.

One early strategy Microsoft came up with to reduce the overhead from NetBIOS broadcasts was to use a special text file called LMHOSTS. **LMHOSTS** contains a list of the NetBIOS names and corresponding IP addresses of the host systems on the network. Sound familiar? Well, it should—the LMHOSTS file works exactly the same way as the DNS HOSTS file. Although Microsoft still supports LMHOSTS file usage, and every Windows system has an LMHOSTS file for backward compatibility, networks that still need NetBIOS support will usually run **Windows Internet Name Service (WINS)** servers for name resolution. WINS servers let NetBIOS hosts register their names with just the one server, eliminating the need for broadcasting and thereby reducing NetBIOS overhead substantially. Figure 10.38 shows the copy of the WINS server that comes with Windows 2000 Server. Note that some of the PCs on this network have registered their names with the WINS server.

You can find an LMHOSTS .SAM file on your Windows system. Use Notepad to open the file and inspect its contents.



WINS

Action View

Tree

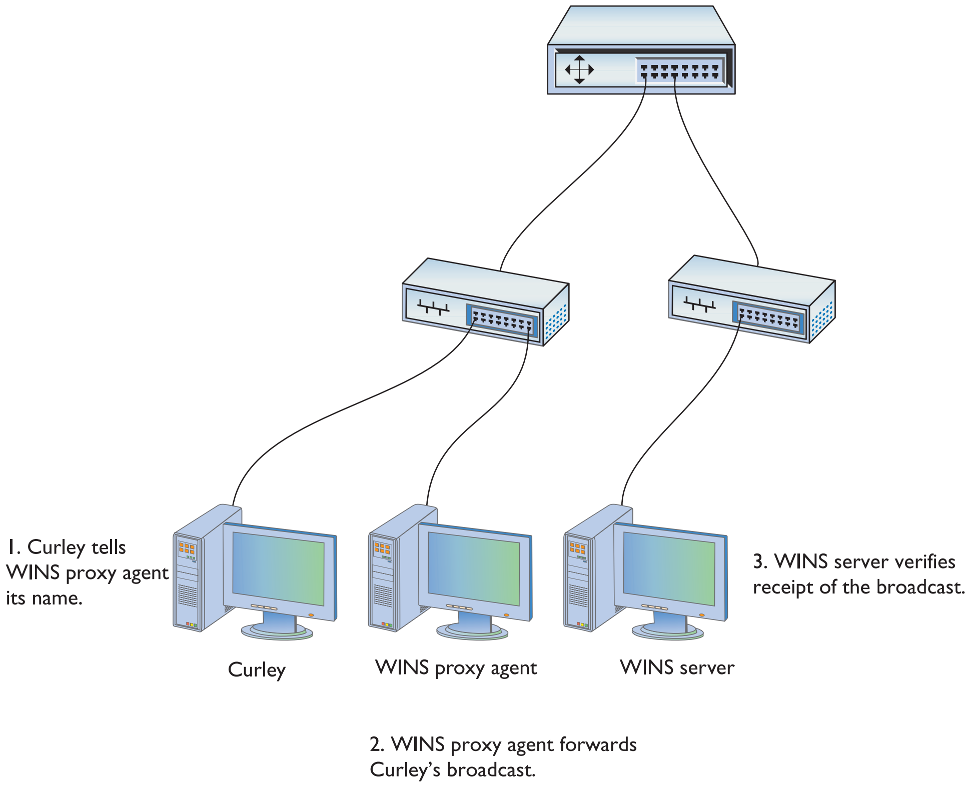
- WINS
  - Server Status
  - TOTALHOMEDC1 [192.168.4.11]
    - Active Registrations
    - Replication Partners

Active Registrations Items found for all owners: 10

Record Name	Type	IP Address	State	Static	Owner
TOTALHOME	[1Bh] Domain Maste…	192.168.4.11	Active		192.168.4.11
ADMINISTRATOR	[03h] Messenger	192.168.4.10	Released		192.168.4.11
INet~Services	[1Ch] Domain Contr…	192.168.4.11	Active		192.168.4.11
IS~TOTALHOMEDC1	[00h] WorkStation	192.168.4.11	Active		192.168.4.11
MICHAELS	[03h] Messenger	192.168.4.11	Active		192.168.4.11
TOTALFS2	[00h] WorkStation	192.168.4.10	Active		192.168.4.11
TOTALFS2	[03h] Messenger	192.168.4.10	Active		192.168.4.11
TOTALFS2	[20h] File Server	192.168.4.10	Active		192.168.4.11
TOTALHOME	[00h] Workgroup	192.168.4.10	Active		192.168.4.11
TOTALHOMEDC1	[03h] Messenger	192.168.4.11	Active		192.168.4.11

• **Figure 10.38** WINS server

There are only two good reasons to use a WINS server: (1) to reduce overhead from broadcasts; and (2) to enable NetBIOS name resolution across routers. What does a WINS server have to do with routers, you ask? Just this: the WINS server enables NetBIOS to function in a routed network. IP routers are programmed to *kill* all broadcasts, remember? While newer Windows clients will just register directly with the WINS server, older (pre-Win95) Windows systems will still try to broadcast. To get around this problem, you can configure a system to act as a **WINS proxy agent**, forwarding WINS broadcasts to a WINS server on the other side of the router (Figure 10.39).


1. Curley tells WINS proxy agent its name.
3. WINS server verifies receipt of the broadcast.
Curley
WINS proxy agent
WINS server
2. WINS proxy agent forwards Curley's broadcast.

• **Figure 10.39** WINS server

The bottom line with WINS servers is this: larger or routed networks that run NetBIOS still need them. As long as Windows NT and Windows 9*x* systems are out there running NetBIOS, don't be surprised to find that some system somewhere is running a WINS server.

Configuring WINS Clients

You don't need to do much to get a Windows client to use WINS. In fact, you only need to configure the IP address of a WINS server in its WINS settings under Network Properties. From now on, the Windows system will look for a WINS server to register its NetBIOS name. If it finds a WINS server, it will register its NetBIOS name to the WINS server; if it doesn't, it will automatically start broadcasting its NetBIOS name. You can add WINS information to DHCP if necessary, so unless you're running static IP addresses, you may never have to enter anything into your Windows clients to get WINS to work.

Think WINS is dead? Open **Manage network connections** in your Network and Sharing Center. There's still a WINS setting for backward compatibility with older computers on the network.

Troubleshooting WINS

Most WINS problems are not WINS problems at all. They are NetBIOS problems. By far, the most common problem is having two systems share the same name. In that case, you get a pretty clear error. It looks different in different versions of Windows, but it usually says about the same thing: another system has this name. How do you fix it? Change the name of the system!

You can use the **NBTSTAT** program to help deal with NetBIOS problems. NBTSTAT will do a number of jobs, depending on the switches you add to the end of the command. The –c switch, for example, tells NBTSTAT to check the current NetBIOS name cache (yup, NetBIOS caches names just like some systems cache DNS names). The NetBIOS name cache contains the NetBIOS names and corresponding IP addresses that have been resolved by a particular host. You can use NBTSTAT to see if the WINS server has supplied inaccurate addresses to a WINS client. Here's an example of the `nbtstat -c` command and its results:

```
C:\ >nbtstat -c

Node IpAddress: [192.168.43.5] Scope Id: []
                NetBIOS Remote Cache Name Table

    Name              Type       Host Address    Life [sec]
------------------------------------------------------------
WRITERS        <1B>  UNIQUE      192.168.43.13       420
SCOTT          <20>  UNIQUE      192.168.43.3        420
VENUSPDC       <00>  UNIQUE      192.168.43.13       120
MIKE           <20>  UNIQUE      192.168.43.2        420
NOTES01        <20>  UNIQUE      192.168.43.4        420
```

Diagnosing TCP/IP Networks

I've dedicated the entire Chapter 15 to network diagnostic procedures, but TCP/IP has a few little extras that I want to talk about here. TCP/IP is a pretty tough protocol, and in good networks, it runs like a top for years without problems. Most of the TCP/IP problems you'll see come from improper configuration, so I'm going to assume you've run into problems with a new TCP/IP install, and we'll look at some classic screw-ups common in this situation. I want to concentrate on making sure you can PING anyone you want to PING.

I've done thousands of IP installations over the years, and I'm proud to say that, in most cases, they worked right the first time. My users jumped on the newly configured systems, fired up their My Network Places/Network, e-mail software, and Web browsers, and were last seen typing away, smiling from ear to ear. But I'd be a liar if I didn't also admit that plenty of setups didn't work so well. Let's start with the hypothetical case of a user who can't see something on the network. You get a call: "Help!" he cries. The first troubleshooting point to remember here: it doesn't matter *what* he can't see. It doesn't matter if he can't see other systems in his network, or can't see the home page on his browser, because you go through the same steps in any event.

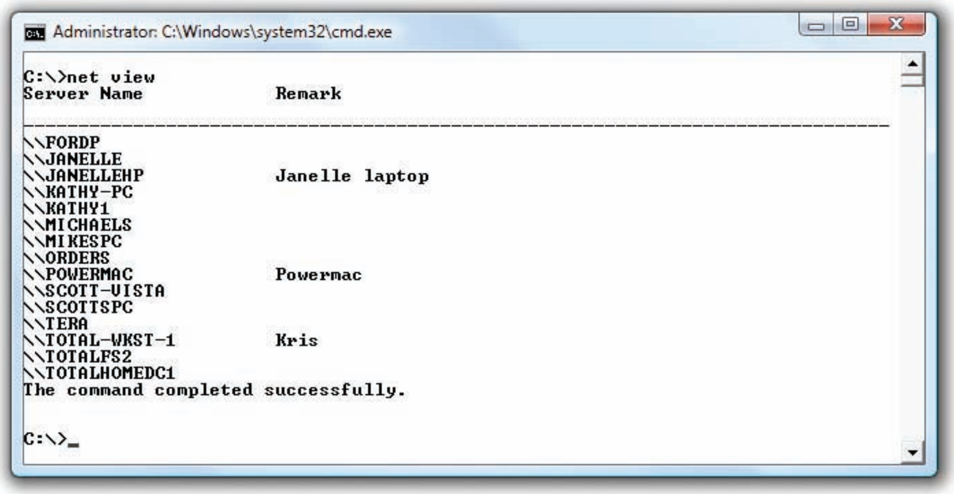
Remember to use common sense wherever possible. If the problem system can't ping by DNS name, but all the other systems can, is the DNS server down? Of course not! If something—*anything*—doesn't work on one system, *always* try it on another one to determine whether the problem is specific to one system or affects the entire network.

One thing I always do is check the network connections and protocols. We're going to cover those topics in greater detail later in the book, so, for now, we'll assume our problem systems are properly connected and have good protocols installed. Here are some steps to take:

1. *Diagnose the NIC*. First, use PING with the loopback address to determine if the system can send and receive packets. Specifically, type `ping 127.0.0.1` or `ping localhost` (remember the HOSTS file?). If you're not getting a good response, your NIC has a problem! Check your NIC's driver and replace it if necessary.
2. *Diagnose locally*. If the NIC's okay, diagnose locally by pinging a few neighboring systems, both by IP address and DNS name. If you're using NetBIOS, use the `net view` command to see if the other local systems are visible (Figure 10.40). If you can't ping by DNS, check your DNS settings. If you can't see the network using NET VIEW, you may have a problem with your NetBIOS settings.

A good testing trick is to use the NET SEND command to try sending messages to other systems. Not all versions of Windows support NET SEND, however.

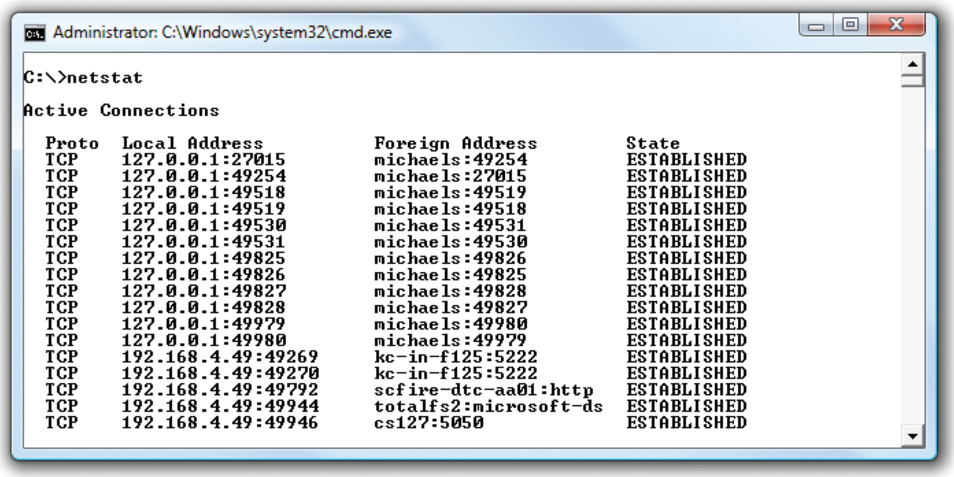
3. *Check IP address and subnet mask*. If you're having a problem pinging locally, make sure you have the right IP address and subnet mask. Oh, if I had a nickel for every time I entered those incorrectly! If you're on DHCP, try renewing the lease—sometimes that will do the trick. If DHCP fails, call the person in charge of the server.
4. *Run NETSTAT*. At this point, another little handy program comes into play called **NETSTAT**. NETSTAT offers a number of options. The two handiest ways to run NETSTAT are with no options at all, and with the **–s** option. Running `netstat` with no options shows you all the current connections to your system. Look for a connection here that isn't working with an application—that's often a clue to an application problem, such as a broken application or a sneaky application running in the background. Figure 10.41 shows a NETSTAT program running.



```
C:\>net view
Server Name            Remark

-------------------------------------------------------------------------------
\\FORDP
\\JANELLE
\\JANELLEHP            Janelle laptop
\\KATHY-PC
\\KATHY1
\\MICHAELS
\\MIKESPC
\\ORDERS
\\POWERMAC             Powermac
\\SCOTT-VISTA
\\SCOTTSPC
\\TERA
\\TOTAL-WKST-1         Kris
\\TOTALFS2
\\TOTALHOMEDC1
The command completed successfully.

C:\>_
```

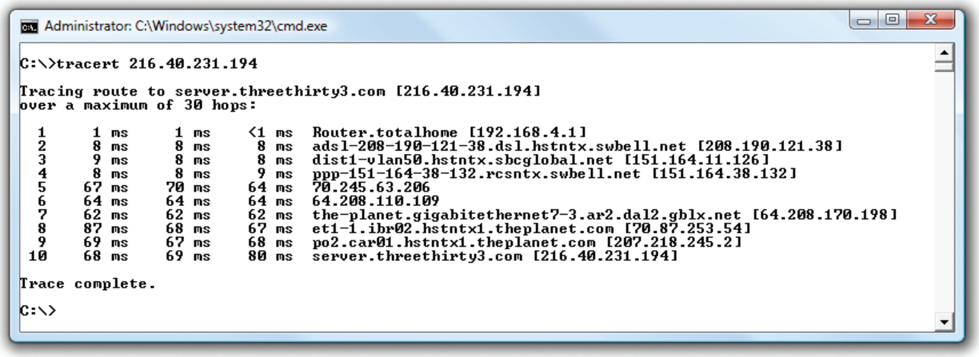
• **Figure 10.40** NET VIEW in action

5. *Run* `netstat -s`. Running NETSTAT with the **–s** option displays several statistics that can help you diagnose problems. For example, if the display shows you are sending but not receiving, you almost certainly have a bad cable with a broken receive wire.
6. *Diagnose to the gateway*. If you can't get out to the Internet, check to see if you can PING the router. Remember, the router has two interfaces, so try both: first the local interface (the one on your subnet), and then the one to the Internet. You *do* have both of those IP addresses memorized, don't you? You should! If you can't PING the router, either it's down or you're not connected to it. If you can only PING the near side, something in the router itself is messed up.



```
C:\>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:27015        michaels:49254         ESTABLISHED
  TCP    127.0.0.1:49254        michaels:27015         ESTABLISHED
  TCP    127.0.0.1:49518        michaels:49519         ESTABLISHED
  TCP    127.0.0.1:49519        michaels:49518         ESTABLISHED
  TCP    127.0.0.1:49530        michaels:49531         ESTABLISHED
  TCP    127.0.0.1:49531        michaels:49530         ESTABLISHED
  TCP    127.0.0.1:49825        michaels:49826         ESTABLISHED
  TCP    127.0.0.1:49826        michaels:49825         ESTABLISHED
  TCP    127.0.0.1:49827        michaels:49828         ESTABLISHED
  TCP    127.0.0.1:49828        michaels:49827         ESTABLISHED
  TCP    127.0.0.1:49979        michaels:49980         ESTABLISHED
  TCP    127.0.0.1:49980        michaels:49979         ESTABLISHED
  TCP    192.168.4.49:49269     kc-in-f125:5222        ESTABLISHED
  TCP    192.168.4.49:49270     kc-in-f125:5222        ESTABLISHED
  TCP    192.168.4.49:49792     scfire-dtc-aa01:http   ESTABLISHED
  TCP    192.168.4.49:49944     totalfs2:microsoft-ds  ESTABLISHED
  TCP    192.168.4.49:49946     cs127:5050             ESTABLISHED
```

• **Figure 10.41** NETSTAT in action



```
C:\>tracert 216.40.231.194

Tracing route to server.threethirty3.com [216.40.231.194]
over a maximum of 30 hops:

  1     1 ms     1 ms    <1 ms  Router.totalhome [192.168.4.1]
  2     8 ms     8 ms     8 ms  adsl-208-190-121-38.dsl.hstntx.swbell.net [208.190.121.38]
  3     9 ms     8 ms     8 ms  dist1-vlan50.hstntx.sbcglobal.net [151.164.11.126]
  4     8 ms     8 ms     9 ms  ppp-151-164-38-132.rcsntx.swbell.net [151.164.38.132]
  5    67 ms    70 ms    64 ms  70.245.63.206
  6    64 ms    64 ms    64 ms  64.208.110.109
  7    62 ms    62 ms    62 ms  the-planet.gigabitethernet7-3.ar2.dal2.gblx.net [64.208.170.198]
  8    87 ms    68 ms    67 ms  et1-1.ibr02.hstntx1.theplanet.com [70.87.253.54]
  9    69 ms    67 ms    68 ms  po2.car01.hstntx1.theplanet.com [207.218.245.2]
 10    68 ms    69 ms    80 ms  server.threethirty3.com [216.40.231.194]

Trace complete.

C:\>
```

● **Figure 10.42** Using TRACERT

7. *Diagnose to the Internet*. If you can PING the router, it's time to try to PING something on the Internet. If you can't PING one address, try another—it's always possible that the first place you try to PING is down. If you still can't get through, you can try to locate the problem using the **TRACERT** (trace route) command. TRACERT will mark out the entire route the PING packet traveled between you and whatever you were trying to PING and, even better, will tell you where the problem lies (see Figure 10.42).

Chapter 10 Review

Chapter Summary

After reading this chapter and completing the exercises, you should able to do the following.

Describe the function and capabilities of DNS

- A HOSTS file maps a computer name to an IP address. When the Internet was in its infancy, every Internet-connected computer had a copy of the same HOSTS file. Today, computers have their own unique HOSTS file, which is always checked before a computer tries to resolve a name using another method.
- DNS is vital to IP networking, whether on the Internet or within the smallest of networks. DNS stands for Domain Name System, which functions as a hierarchical naming system for computers on a network. A DNS server resolves FQDNs (fully qualified domain names) to IP addresses.
- The 13 DNS root servers for the Internet are logical servers composed of many DNS servers acting as a single monstrous server.
- If one DNS domain name space cannot find out (resolve) the IP address of a computer, the request gets passed along further to another DNS server. The process continues until the request reaches the destination computer.
- Note that because not all computers are connected to the Internet, computer networks are not required to belong to a DNS domain. However, administrators can set up their own DNS domain name spaces without ever connecting to the Internet. These isolated internal intranets can be given elaborate naming structures of their own as well.
- DNS is a convenience, not a requirement. You can connect to a Web site by typing the correct IP address, bypassing the need to resolve an FQDN.
- Name resolution can be accomplished through broadcasting, by consulting the local HOSTS file, or by contacting a DNS server.
- IPCONFIG /ALL can be used to view your DNS server settings. IPCONFIG /DISPLAYDNS displays a cache of recently resolved FQDNs.
- DNS servers store a list of cached lookups—all IP addresses the server has already resolved.
- An authoritative DNS server stores IP addresses and FQDNs of all systems for a particular domain while a cache-only DNS server is used to communicate with other DNS servers.
- Forward lookup zones are the most important part of any DNS server because they contain the IP addresses and FQDNs.
- Of the two types of forward lookup zones, primary zones are created on authoritative DNS servers while secondary zones are created on other DNS servers to act as a backup to the primary zone.
- A records, CNAME records, and MX records must be properly configured on any DNS server.
- Reverse lookup zones resolve an IP address to an FQDN.
- Microsoft's Common Internet File System (CIFS), which began as Server Message Block (SMB), originated when NetBIOS/NetBEUI dropped NetBEUI in favor of IP and used the NetBIOS name as the DNS name. It was used primarily to share files and printers in small TCP/IP networks.
- CIFS organizes computers into one of three types of groups: workgroup, Windows domain, or Active Directory.
- A Windows domain provides centralized management and user authentication via a computer acting as a domain controller.
- An Active Directory is an organization of related computers that shares one or more Windows domains. There is no single domain controller in Active Directory because all domain controllers operate equally.
- Under Active Directory, all domain controllers are also DNS servers. Because Active Directory domain controllers operate equally, there is no single point of failure throughout Active Directory's DNS system. All domain controllers hold primary zones.
- The command IPCONFIG is useful for troubleshooting TCP/IP settings. Running `ipconfig /flushdns` will clear the local cache of DNS entries.

- The PING command is essential in establishing connectivity to a destination PC. If you can PING a host computer by IP address (for example, `ping 192.168.4.55`), but not by name (`ping acctngpc2`), then you have a DNS resolution issue. Check cables, check the DNS servers listed under each network adapter card's settings, and finally, check to see that the DNS server is truly up and operational.
- The NSLOOKUP command enables you to research what name servers are being used by a particular computer. Advanced variations of the NSLOOKUP command can query information from a DNS server and even change how your system uses DNS.
- UNIX/Linux users have an additional DNS tool called DIG, which is different from NSLOOKUP in that DIG runs noninteractively.

Configure and troubleshoot WINS

- An LMHOSTS file works almost the same as a HOSTS file, except it correlates NetBIOS names to IP addresses.
- WINS stands for Windows Internet Name Service, which is an older name resolution method. WINS servers help Windows systems (in place of the even older LMHOSTS files) with resolving NetBIOS computer names (like SALESPC7) to IP addresses (like 192.168.10.7) on a Windows network.
- WINS clients virtually configure themselves by using broadcasts to find WINS servers. A WINS proxy agent forwards WINS broadcasts across routers that would normally block such broadcasts.
- WINS problems relate directly to NetBIOS problems. The most common problem by far is having two systems share the same name. The resulting error message clearly indicates that another system is trying to use the same name. Simply change the computer's system name to fix this common problem.
- Using the NBTSTAT –c command will check the current NetBIOS name cache. This NetBIOS name cache contains the NetBIOS names (along with their corresponding IP addresses) that have been resolved already by a particular host.
- Use the NBTSTAT command alone to see whether the WINS server has supplied inaccurate addresses to a particular WINS client.

Use common TCP/IP utilities to diagnose problems with DNS and WINS

- Always try to connect from another system to determine the extent of the problem. You can then begin the steps in diagnosing TCP/IP errors on a single system.
- Remember to work "from the inside out"—that is, check for connectivity problems on the local system before moving on to check the larger network structure. First, type `ping 127.0.0.1` (or `ping localhost`) to ensure that the local NIC is seated properly and TCP/IP is installed.
- On Windows systems, the NET VIEW command is worth trying. If you can't see the network using NET VIEW, you may have a problem with your NetBIOS settings.
- Running `netstat` shows all the current connections on your system. Running `netstat -s` displays useful statistical information.
- The TRACERT command allows you to mark the entire route a ping packet travels, telling you exactly where a problem lies.

Key Terms

A record *(260)*
Active Directory *(264)*
authoritative DNS server *(251)*
cached lookup *(258)*
cache-only DNS server *(260)*
canonical name (CNAME) *(261)*
Common Internet File System (CIFS) *(263)*
DNS root server *(247)*
DNS server *(247)*
DNS tree *(249)*
domain information grouper (DIG) *(266)*
Domain Name System (DNS) *(245)*
flat name space *(248)*
forward lookup zone *(260)*
fully qualified domain name (FQDN) *(250)*
hierarchical name space *(248)*
host name *(249)*
HOSTS file *(245)*

IPCONFIG *(265)*
LMHOSTS *(267)*
MX record *(261)*
name resolution *(244)*
name server *(251)*
NetBIOS/NetBEUI *(262)*
NETSTAT *(270)*
NBTSTAT *(269)*
NSLOOKUP *(266)*
NS record *(260)*
PING *(265)*
primary zone *(261)*
reverse lookup zone *(262)*
secondary zone *(261)*
TRACERT *(272)*
top-level domain server *(247)*
Windows domain *(263)*
Windows Internet Name Service (WINS) *(245, 267)*
WINS proxy agent *(268)*
workgroup *(263)*

Key Term Quiz

Use the Key Terms list to complete the sentences that follow. Not all the terms will be used.

1. The _______________ command is used to establish connectivity.
2. Using _______________ alone can help determine whether a WINS server has supplied inaccurate addresses to a particular WINS client.
3. The term _______________ refers to networks that use DNS belonging to the same DNS system.
4. A helpful command that displays TCP/IP naming information is _______________.
5. _______________ is responsible for resolving NetBIOS names to IP addresses on predominately Windows networks.
6. To connect to systems on the Internet, your network needs the name of at least one _____________.
7. To forward WINS broadcasts to a WINS server on the other side of the router, you need to set up a(n) _______________.
8. You can use the diagnostic utility called _________ to trace the progress of a ping packet between your system and a remote computer.
9. To avoid having to re-resolve an FQDN that it has already checked, a Windows DNS server keeps a list of IP addresses it has already resolved, called _______________.
10. This single DNS server that has a list of all the host names on the domain and their corresponding IP addresses is the _______________.

Multiple-Choice Quiz

1. Which of the following are needed for e-mail clients to find their e-mail servers, FTP clients to find their file servers, and Web browsers to find Web servers?
   - **A.** DHCP servers
   - **B.** DNS servers
   - **C.** E-mail servers
   - **D.** WINS servers
2. What do DNS servers use to help resolve IP addresses to DNS names?
   - **A.** Authentication
   - **B.** Authorization
   - **C.** Backward lookup zones
   - **D.** Reverse lookup zones
3. What do DNS servers use to help resolve DNS names to IP addresses?
   - **A.** Accounting
   - **B.** Administration
   - **C.** Backward lookup zones
   - **D.** Forward lookup zones
4. What type of DNS servers do not have any forward lookup zones, and will resolve names of systems on the Internet for a network, but are not responsible for telling other DNS servers the names of any clients?
   - **A.** Cache-only servers
   - **B.** Primary servers
   - **C.** Secondary servers
   - **D.** WINS servers

5. What single command gives you the IP address and the name of your system's default DNS server?
   - **A.** `nbtstat`
   - **B.** `nslookup`
   - **C.** `ping`
   - **D.** `winword`
6. What file can be replaced when a network has a WINS server?
   - **A.** HOSTS
   - **B.** LMHOSTS
   - **C.** SAM
   - **D.** WINS
7. What file can be replaced when a network has a DNS server?
   - **A.** HOSTS
   - **B.** LMHOSTS
   - **C.** SAM
   - **D.** WINS
8. What will adding a WINS proxy agent enable you to accomplish on your network?
   - **A.** Cross a hub
   - **B.** Cross a server
   - **C.** Cross a switch
   - **D.** Cross a router
9. Folders with subfolders on a system, like domain names with subdomains, are said to have a structure resembling what?
   - **A.** Branch
   - **B.** Forest
   - **C.** Root
   - **D.** Tree
10. Which of the following commands clears the local cache of DNS entries?
    - **A.** `ipconfig /clear`
    - **B.** `ipconfig /cls`
    - **C.** `ipconfig /flushdns`
    - **D.** `ipconfig /renew`
11. Which variation of the NBTSTAT command will check the current NetBIOS name cache?
    - **A.** `nbtstat`
    - **B.** `nbtstat -c`
    - **C.** `nbtstat /checkupgradeonly`
    - **D.** `nbtstat /status`
12. Which of these terms are frequently used interchangeably? (Select two.)
    - **A.** Domain
    - **B.** Folder
    - **C.** Subdomain
    - **D.** Zone
13. Which of the following are valid DNS record entry types? (Select three.)
    - **A.** A
    - **B.** M
    - **C.** NS
    - **D.** SOA
14. Which of the following is an example of a top-level domain?
    - **A.** .com
    - **B.** totalsem.com
    - **C.** support.totalsem.com
    - **D.** houston.support.totalsem.com
15. How do authoritative DNS servers and cache-only DNS servers differ?
    - **A.** Authoritative DNS servers contain forward lookup zones while cache-only DNS servers contain only reverse lookup zones.
    - **B.** Authoritative DNS servers store IP addresses and FQDNs of systems for a particular domain or domains while cache-only DNS servers do not store any FQDNs because they are only used to talk to other DNS servers to resolve IP addresses.
    - **C.** Authoritative DNS servers service requests for top-level domains while cache-only DNS servers service requests for down-level domains.
    - **D.** Authoritative DNS servers are found only in Windows Active Directory networks while cache-only DNS servers are found universally throughout the Internet.

Essay Quiz

1. Some classmates at school have been playing with (and giggling over) the NET SEND command during class time. The instructor notices what's going on, and hoping to turn the experience into something useful, asks each student to write down a valid use of the NET SEND command. Write down your answer.
2. Your boss comes into your office in a panic. He can't reach the company's internal Web server from his office. It worked yesterday. Write an essay describing what you'd do to troubleshoot the situation. Which tool or tools would you use? Why?
3. After discussing flat versus hierarchical naming schemes in class, a feisty classmate proclaims that flat names should be used on individual systems as well as on the Internet for simplification. Write a brief reason or two why he is wrong in his oversimplification.
4. Jot down some brief notes about how you would troubleshoot and diagnose a TCP/IP issue on one of the systems on your network. You can list the actual commands if you like, too. Choose an interesting Web site that you would ping out on the Internet as your final step.

Lab Projects

• Lab Project 10.1

This chapter has presented a lot of variations of common network troubleshooting commands. You have decided it would be beneficial to create an alphabetized chart of these commands, including their variations and what they do. Using either a word processing program or spreadsheet program, create a chart like the following—you fill in the rightmost column:

Command	Variation	What It Does . . .
IPCONFIG	(blank)	
IPCONFIG	/ALL	
IPCONFIG	/RELEASE	
IPCONFIG	/RENEW	
IPCONFIG	/FLUSHDNS	
NBTSTAT	(blank)	
NBTSTAT	–c	
NET	SEND	
NET	VIEW	
PING	127.0.0.1	
PING	disney.com	
PING	localhost	

• Lab Project 10.2

There are potentially many trips a request must make when trying to resolve a fully qualified domain name to an IP address. Aside from the HOSTS file, there are primary DNS servers, secondary DNS servers, authoritative DNS servers, cache-only DNS servers, DNS root servers, top-level DNS servers, and second-level domain servers. On a piece of paper, sketch a diagram/flowchart showing how a request for www.example.com gets resolved to an IP address.