CHAPTER 14

Going Large with TCP/IP

The Network+ exam expects you to know how to

- 2.9 Identify and differentiate between the following IP (Internet Protocol) addressing methods: static, dynamic, self-assigned (APIPA – Automatic Private Internet Protocol Addressing)
- 2.13 Identify the purpose of network services and protocols (for example: DNS – Domain Name Service, WINS – Windows Internet Naming Service)
- 4.1 Given a troubleshooting scenario, select the appropriate network utility from the following: ping, netstat, nbtstat, ipconfig/ifconfig, tracert/traceroute, winipcfg, nslookup/dig
- 4.2 Given output from a network diagnostic utility (for example: those utilities listed in objective 4.1), identify the utility and interpret the output
- 4.6 Given a scenario, determine the impact of modifying, adding, or removing network services (for example: DHCP – Dynamic Host Configuration Protocol, DNS – Domain Name Service, and WINS – Windows Internet Naming Service) for network resources and users

To achieve these goals, you must be able to

- Understand DNS in detail
- Learn about troubleshooting DNS
- Understand DHCP in detail
- Learn about troubleshooting DHCP
- Understand WINS in detail
- Learn about troubleshooting WINS
- Understand diagnosing TCP/IP networks

Well, no sooner do you think you're done with TCP/IP than I grab you by the collar and we're right back into it harder than ever! You didn't *really* think we were finished with TCP/IP yet, did you? I promise no more subnetting, but we do need to look more at some critical tools we use to make TCP/IP networks function, particularly large ones. In fact, we're going to work with the largest of all networks, the Internet itself.

In this chapter, you'll take an in-depth tour of DNS, WINS, and DHCP. Sure, I know you know how to set up your Windows clients for these fellas—if someone tells you what to enter in those TCP/IP Network Properties boxes, you know where to type them in. But now we're going to look at these in far more detail. In fact, we'll go a good bit beyond Network+ by looking at some real DNS, WINS, and DHCP servers! At the end of this chapter, we'll dive into the many software utilities (including some you've already seen, but in far more depth) and use them to diagnose and fix a number of the most common problems that occur in TCP/IP networks. You'll definitely want to know that part for the Network+ exam!

Odds are good you've got a system that is connected—or at least can connect—to the Internet. If I were you, I'd be firing 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

Chapter 11, "TCP/IP," gave you a brief, client-centric overview of DNS. Let's now go a bit deeper into DNS and take some time to appreciate how it works, see a DNS server in action, and then explore some of the neat tools you can use to diagnose DNS problems. We're going to revisit a few things, like HOSTS files and domain names, but we'll be looking at them in a more practical way—really getting into them, and even having a little fun.

DNS in Detail

In Chapter 11, you saw that DNS uses a hierarchical naming system and that it needs a magic box called a *DNS server* to resolve fully qualified domain names (FQDNs) to IP addresses. But how does this *really* work? More importantly, if you're having a problem, how can you determine if DNS is the culprit? If it is, what can you do about it without having to become an expert on DNS servers? Conveniently, it turns out that if you're having a DNS problem, it's probably not because of the DNS server crashing or some other problem over which you have no control. Instead, it's usually due to problems with the client systems, or another problem that you most certainly can control—*if* you understand DNS and know your DNS diagnostic tools.

DNS Organization

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, NetBIOS names use 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.

TIP NetBIOS names use a flat name space. Try making two computers in a Windows network with the same name and watch the errors appear.

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 14-1. 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.

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 FAT, 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

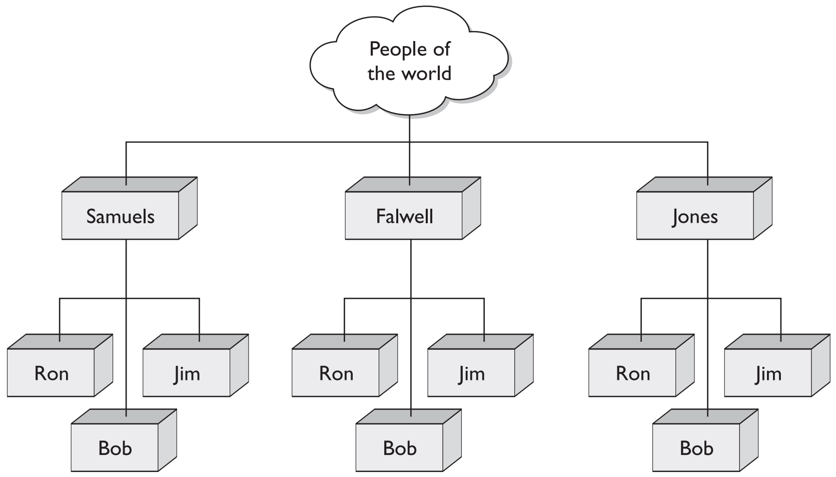


Figure 14-1 People name space

the file tree. In the example shown in Figure 14-2, 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 folders 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.

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 file names. 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.

NOTE As hard as this may be to believe, some early file systems used a flat naming 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 your computer's file system. The DNS name space is a hierarchy of *DNS domains* and individual computer

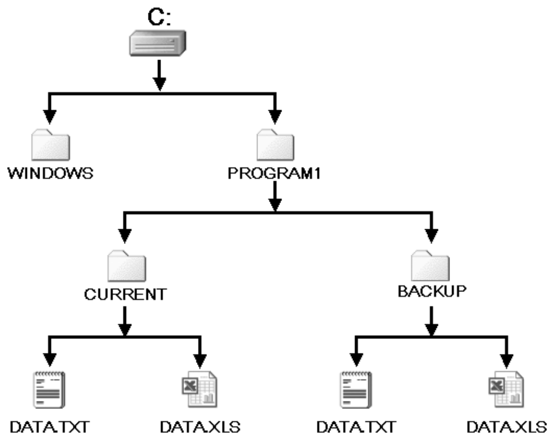


Figure 14-2 Two DATA.TXT files in different directories on the same system

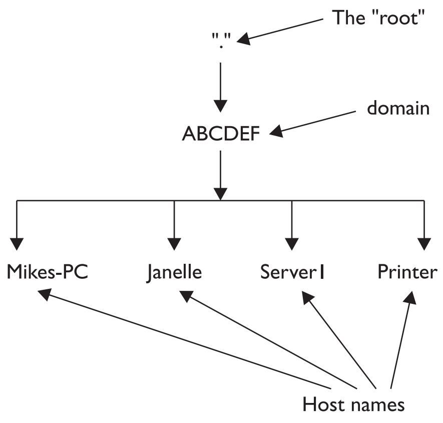
names organized into a tree-like structure that we call, rather appropriately, a *tree*. Each domain is like a folder—a domain is not a single computer, but rather a holding space into which you can add computer names. At the top of a *DNS tree* is the root. The *root* is the holding area to which all 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 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.

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

Figure 14-3 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.

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. 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

Figure 14-3
Small DNS tree



far right. The previous Figure 14-3 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

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 I make a point never to make these computers visible on the Internet, I can use any naming convention I want!

Let's look at another DNS namespace 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 14-4. Note that each domain has a computer called Server1.

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 14-5.

We 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 we get to the top of the DNS tree.

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

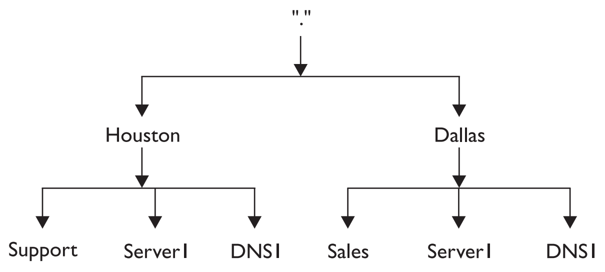


Figure 14-4 Two DNS domains

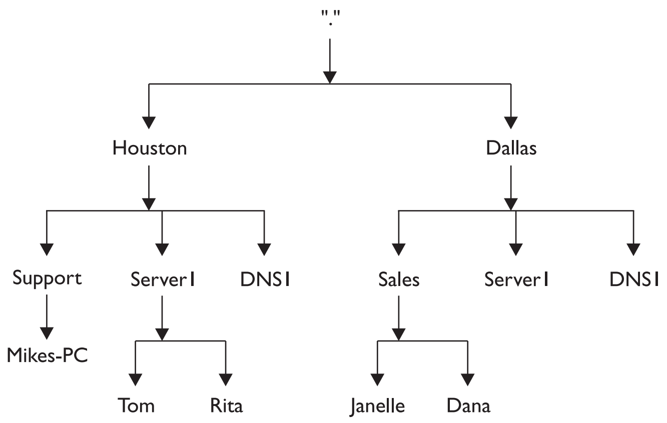


Figure 14-5 Subdomains added

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

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 to store a directory structure, like you have on a PC. Rather, we store the DNS information on systems running DNS server software. 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, like the one shown back in Figure 14-3, 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 address. It's known as the *authoritative DNS server* for the domain. 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 may act as the authoritative DNS sever for one domain or many domains—DNS is very flexible.

But what if more than one DNS server is in control of different domains? How does an FQDN get resolved if your DNS server is not authoritative for a particular FQDN? Let's refer to Figure 14-5 for the answer. Let's say Mikes-PC.Support.Houston needs the IP address of Server1.Dallas. The network has two DNS servers: DNS1.Houston and DNS1.Dallas. DNS1.Dallas is the authoritative 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. This means that the Houston server has a listing for every domain in the Dallas domain structure. This does *not* mean it knows the IP address for every system in the Dallas network! It only knows 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 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!

NOTE In the early days of DNS, you were required to enter the host name and IP address of every system on the network into your DNS server manually. 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.

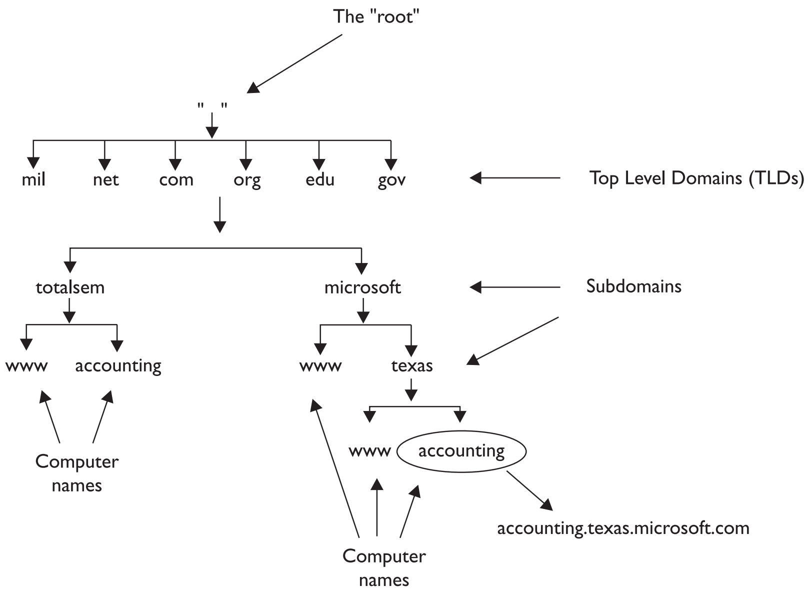
The Internet most certainly uses a DNS tree. Not only does it use a DNS tree, but the folks who designed the Internet created a specific naming convention using the now-famous first-level domain names we've come to know and love: .com, .net, and so forth. The Internet also has a root. The DNS root for the entire Internet consists of 13 powerful DNS servers scattered all over the world. The Internet root is 13 "logical" servers—each single logical server is many DNS servers acting as one monstrous server.

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 fully qualified name.

TIP 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 14-6 shows the host named *accounting* with a fully qualified domain name of *accounting.texas.microsoft.com*.

Figure 14-6 DNS domain *texas.microsoft.com* containing the host *accounting*



NOTE Technically, the texas.microsoft.com domain shown in Figure 14-6 is a subdomain of microsoft.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 14-7 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.

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 a locally stored text file we discussed earlier, called HOSTS, or by contacting a DNS server.

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 14-8.

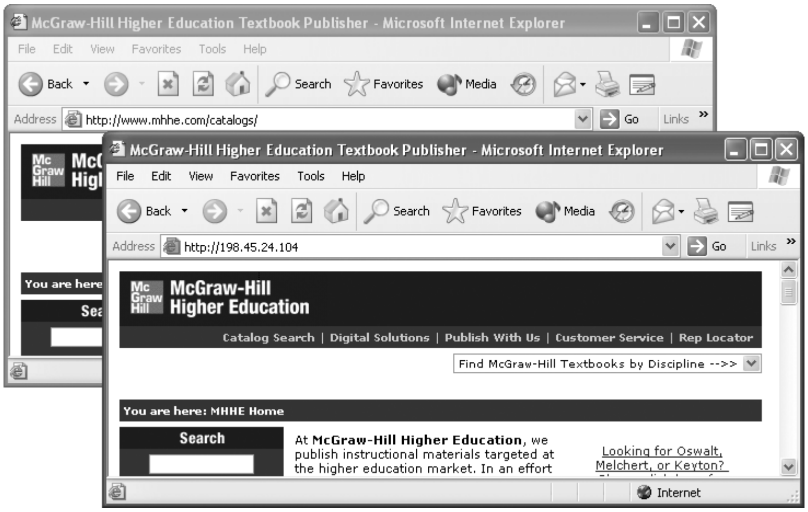
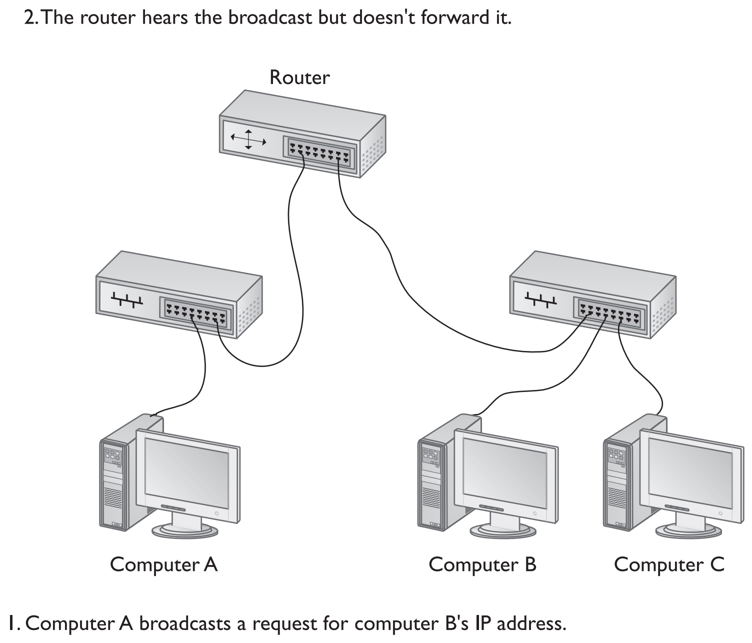

McGraw-Hill Higher Education Textbook Publisher - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://www.mhhe.com/catalogs/
Go Links
McGraw-Hill Higher Education Textbook Publisher - Microsoft Internet Explorer
Address http://198.45.24.104
McGraw-Hill Higher Education
Catalog Search | Digital Solutions | Publish With Us | Customer Service | Rep Locator
Find McGraw-Hill Textbooks by Discipline -->>
You are here: MHHE Home
Search
At McGraw-Hill Higher Education, we publish instructional materials targeted at the higher education market. In an effort
Looking for Oswalt, Melchert, or Keyton?
Internet

Figure 14-7 Sockets-based applications like Internet Explorer can accept IP addresses or DNS names.

Figure 14-8 Name resolution by broadcast does not work across routers because routers do not forward broadcasts.


2. The router hears the broadcast but doesn't forward it.
Router
Computer A
Computer B
Computer C
1. Computer A broadcasts a request for computer B's IP address.

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
```

NOTE 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. We've seen what a wonderful thing DNS is for the user; now let's look at how it does the job. To resolve the name www.microsoft.com, the host contacts its DNS server and requests the IP address, as shown in Figure 14-9. The local DNS server may not know the address for www.microsoft.com, but it does know the address of a DNS root server. 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 server in charge of all .com addresses. 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. 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 14-10 shows the process of resolving a fully qualified domain name into an IP address.

Figure 14-9 A host contacts its DNS server.

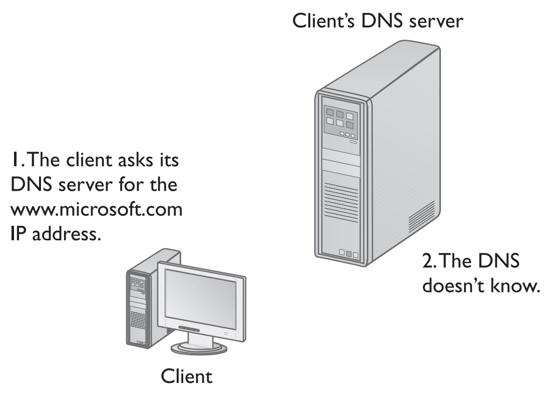
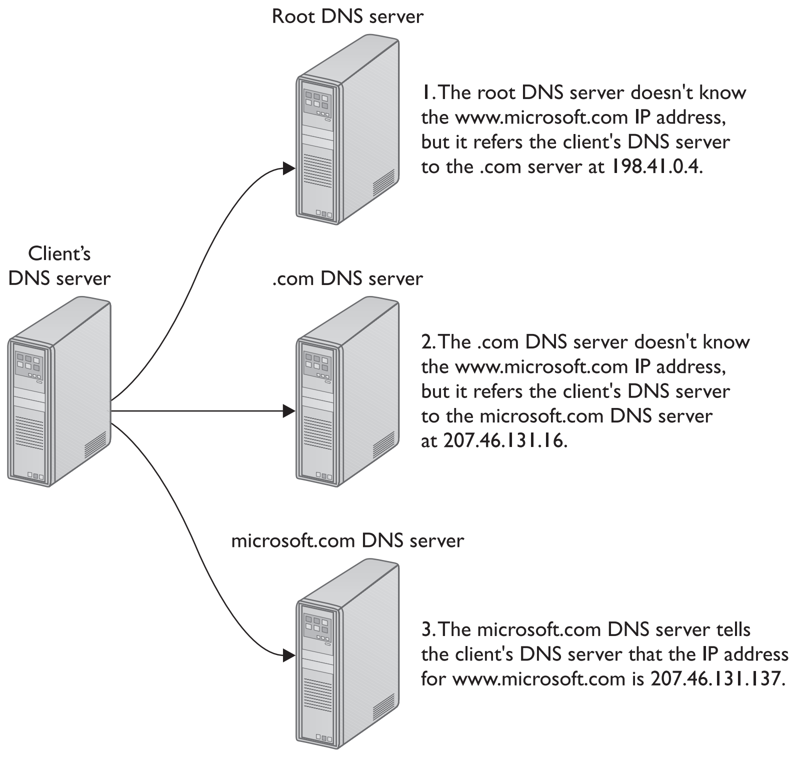

Client's DNS server
1. The client asks its DNS server for the www.microsoft.com IP address.
2. The DNS doesn't know.
Client

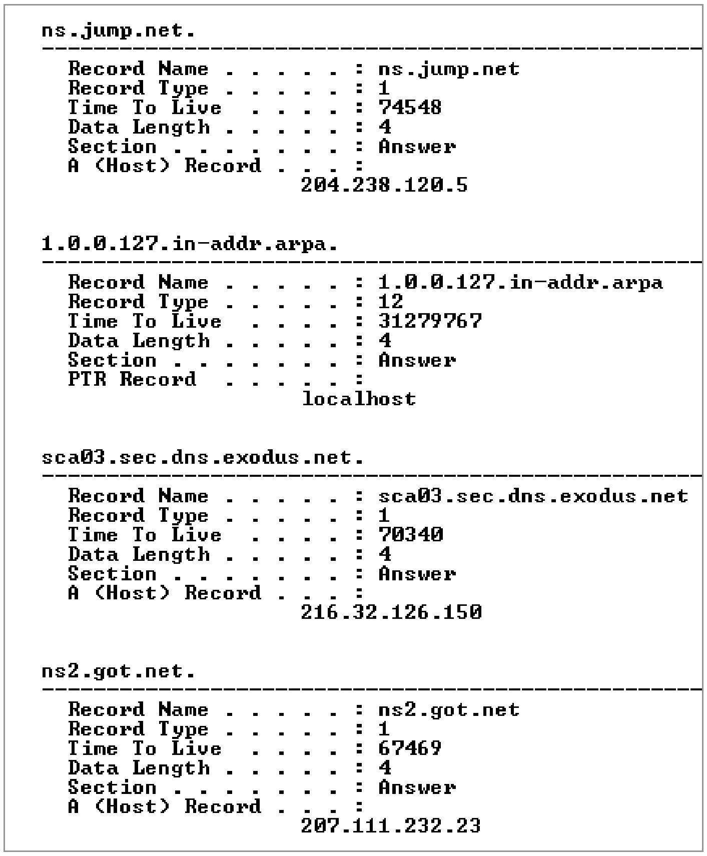
Figure 14-10 A DNS server contacts other DNS servers to resolve a FQDN into an IP address.


Root DNS server
1. The root DNS server doesn't know the www.microsoft.com IP address, but it refers the client's DNS server to the .com server at 198.41.0.4.
Client's DNS server
.com DNS server
2. The .com DNS server doesn't know the www.microsoft.com IP address, but it refers the client's DNS server to the microsoft.com DNS server at 207.46.131.16.
microsoft.com DNS server
3. The microsoft.com DNS server tells the client's DNS server that the IP address for www.microsoft.com is 207.46.131.137.

No single machine needs to know every DNS name, as long as every machine knows who to ask for more information. The distributed, decentralized nature of the DNS database provides a great deal of flexibility and freedom to network administrators using DNS. DNS still requires an administrator to type in each name and address, just as they do with a HOSTS file. There are two key advantages, however, to maintaining this information in a DNS database, compared to maintaining the same information in the form of HOSTS files. First, because the database is centralized on the DNS server, an administrator can add new entries just once, rather than walking around the network to add new entries to each machine. Second, the database is distributed, meaning that no single administrator must maintain a database that knows about every other machine in the world. A DNS server simply has to know about the other DNS servers where it can go for more information.

The DNS Cache

Most web browsers and Windows 2000/2003/XP systems keep track of DNS via a cache, logically called the *DNS resolver cache*. After a web browser or a Windows 2000/XP system has made a DNS request, it keeps that IP address in its own personal DNS cache. If you want to see the DNS cache on a Windows 2000, 2003, or XP system, use the *IPCONFIG* utility—run the command **IPCONFIG /displaydns** at a command prompt. Internet Explorer and Netscape Navigator also have DNS caches, but you need a third-party utility to see their contents. **IPCONFIG /displaydns** creates a rather messy, long output, so be ready to do some scrolling. Figure 14-11 shows just a small bit of the typical output from **IPCONFIG /displaydns**. You can then erase this cache using the **IPCONFIG /flushdns** command. Remember this command—you'll need it in just a moment! For now, let's take a look at a real DNS server running on a Windows 2000 Server system.



```
ns.jump.net.
----------------------------------------------------
  Record Name . . . . . : ns.jump.net
  Record Type . . . . . : 1
  Time To Live  . . . . : 74548
  Data Length . . . . . : 4
  Section . . . . . . . : Answer
  A (Host) Record . . . :
                    204.238.120.5


1.0.0.127.in-addr.arpa.
----------------------------------------------------
  Record Name . . . . . : 1.0.0.127.in-addr.arpa
  Record Type . . . . . : 12
  Time To Live  . . . . : 31279767
  Data Length . . . . . : 4
  Section . . . . . . . : Answer
  PTR Record  . . . . . :
                    localhost


sca03.sec.dns.exodus.net.
----------------------------------------------------
  Record Name . . . . . : sca03.sec.dns.exodus.net
  Record Type . . . . . : 1
  Time To Live  . . . . : 70340
  Data Length . . . . . : 4
  Section . . . . . . . : Answer
  A (Host) Record . . . :
                    216.32.126.150


ns2.got.net.
----------------------------------------------------
  Record Name . . . . . : ns2.got.net
  Record Type . . . . . : 1
  Time To Live  . . . . : 67469
  Data Length . . . . . : 4
  Section . . . . . . . : Answer
  A (Host) Record . . . :
                    207.111.232.23
```

Figure 14-11 Some output from the IPCONFIG / displaydns command

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 network operating systems come with built-in DNS server software, including Windows NT, 2000 and 2003 Server, NetWare 5.*x* and 6.*x*, 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 2000 Server primarily because (1) it takes the prettiest screen snapshots and (2) it's the one I use here at the office. You start the DNS server by selecting Administrative Tools | DNS from the Start menu. When you first open the DNS server, there's not much to see other than the name of the server itself; in this case, it has the boring name *TOTALHOMEDC1* (see Figure 14-12).

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

The first folder 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 a FQDN name 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 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. This sure makes it easy to see where folks have been web browsing! (See Figure 14-13.)

Now to the actual DNS serving work. Basically, there are two types of DNS servers. Authoritative DNS servers hold the IP addresses and names of systems for a particular domain or domains in special storage areas called *forward lookup zones*. In Figure 14-14, the record called SOA (Start of Authority) in the folder totalhome indicates that my server is the authoritative server for a domain called totalhome. The totalhome domain is an in-

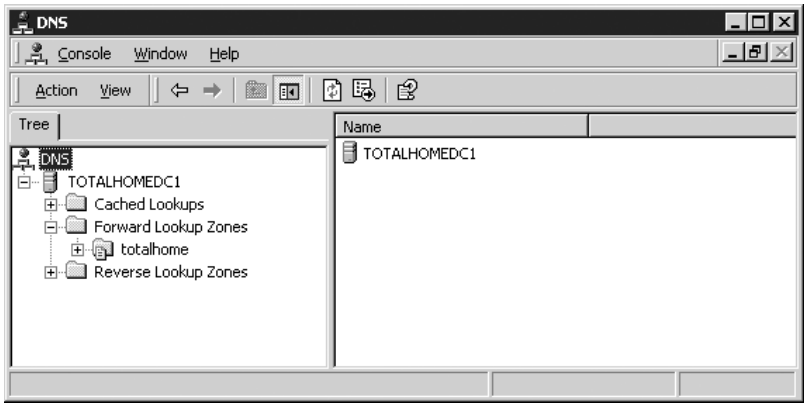
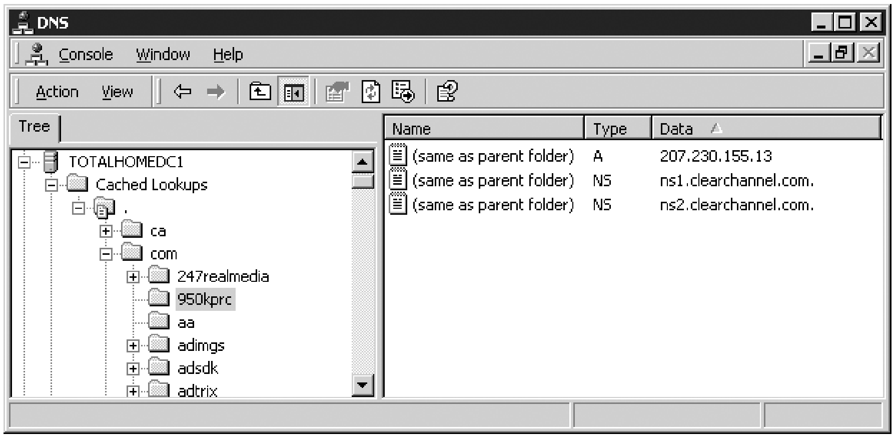


Figure 14-12 Mike's Windows 2000 DNS server when first opened

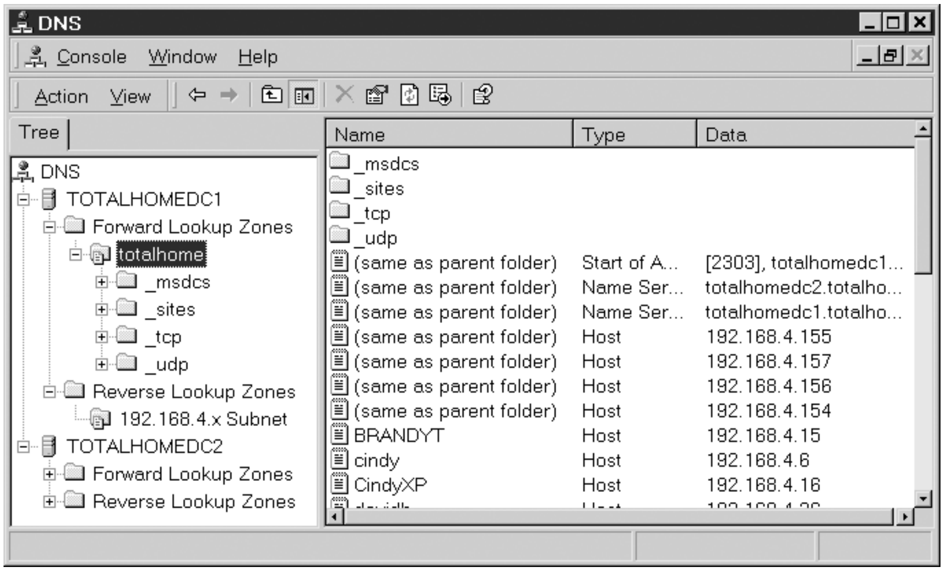


```
TOTALHOMEDC1
  Cached Lookups
    .
      ca
      com
        247realmedia
        950kprc
        aa
        adimgs
        adsdk
        adtrix
```

Name	Type	Data
(same as parent folder)	A	207.230.155.13
(same as parent folder)	NS	ns1.clearchannel.com.
(same as parent folder)	NS	ns2.clearchannel.com.

Figure 14-13 The Cached Lookups table

ternal DNS domain (none of the computers in the domain act as any type of Internet servers), so I don't have to keep with the official Internet naming structures like adding a ".com" to the end of 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 server for the totalhome domain. The NS records are all of the DNS servers for totalhome. Note that totalhome has two DNS servers: totalhomedc1 and totalhomedc2. Two DNS servers ensures that if one fails, the



```
DNS
  TOTALHOMEDC1
    Forward Lookup Zones
      totalhome
        _msdcs
        _sites
        _tcp
        _udp
    Reverse Lookup Zones
      192.168.4.x Subnet
  TOTALHOMEDC2
    Forward Lookup Zones
    Reverse Lookup Zones
```

Name	Type	Data
_msdcs		
_sites		
_tcp		
_udp		
(same as parent folder)	Start of A...	[2303], totalhomedc1...
(same as parent folder)	Name Ser...	totalhomedc2.totalho...
(same as parent folder)	Name Ser...	totalhomedc1.totalho...
(same as parent folder)	Host	192.168.4.155
(same as parent folder)	Host	192.168.4.157
(same as parent folder)	Host	192.168.4.156
(same as parent folder)	Host	192.168.4.154
BRANDYT	Host	192.168.4.15
cindy	Host	192.168.4.6
CindyXP	Host	192.168.4.16

Figure 14-14 The totalhome folder of the authoritative server

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.

The second type of DNS servers are *cache-only*. *Cache-only DNS servers* do not have any forward lookup zones. They will resolve names of systems on the Internet for the network, but are not responsible for telling other DNS servers the names of any clients. This is fairly common for DNS servers in smaller networks that still use NetBIOS. Internally, they use NetBIOS broadcasts to resolve each other's names, but then call on a cache-only DNS server when resolving names out on the Internet.

The other folder you can see in Figure 14-15 is called *Reverse Lookup Zones*. This rather strange setting enables a system to determine a FQDN by knowing the IP address; that is, it does the exact reverse of what DNS normally does! A few low-level functions and some security programs use reverse lookup zones, so DNS servers provide them.

Troubleshooting DNS

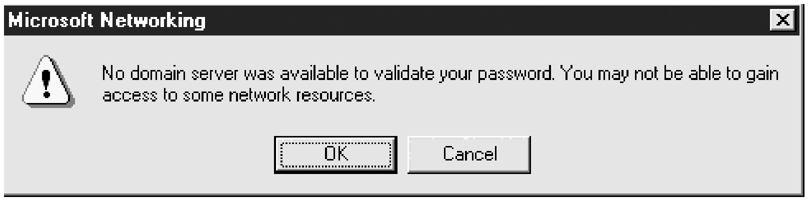
As I mentioned earlier, most DNS problems result from a problem with the client systems. This is because DNS servers rarely go down, and if they do, most clients have a secondary DNS server setting that lets them continue to work properly. DNS servers have been known to fail, however, so it's important 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 they're 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 "server not found." Figure 14-15 shows how this error appears in a web browser and in an e-mail client.

Before you start testing, you need to eliminate any DNS caches on the local system. If you're running Windows 2000 or Windows XP, 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!

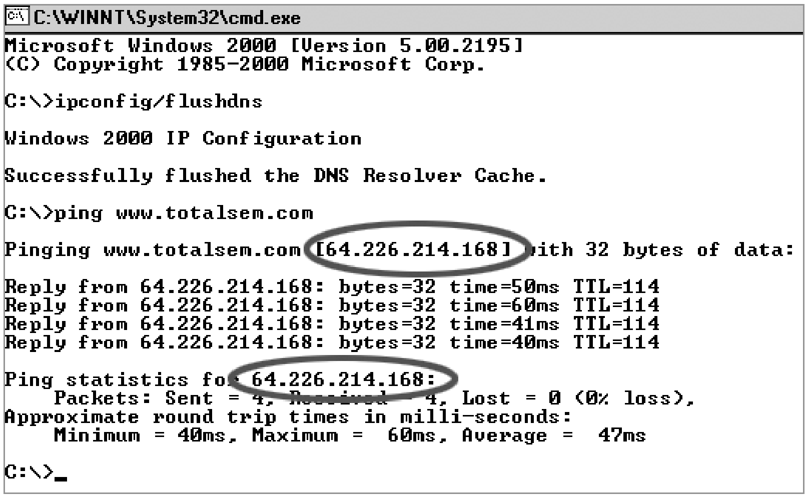
PING is your best friend when you're testing DNS. Run PING from a command prompt, first typing in the name of a well-known web site, such as www.microsoft.com. Watch the output carefully to see if you get an IP address. You may get a "request timed out" message, but that's fine; you just want to see if DNS is resolving FQDN names into IP addresses (see Figure 14-16).

Figure 14-15 "Server not found" errors in web browsers and e-mail clients


Microsoft Networking

No domain server was available to validate your password. You may not be able to gain access to some network resources.

Figure 14-16 Pinging www.totalsem.com



```
C:\WINNT\System32\cmd.exe
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>ipconfig/flushdns

Windows 2000 IP Configuration

Successfully flushed the DNS Resolver Cache.

C:\>ping www.totalsem.com

Pinging www.totalsem.com [64.226.214.168] with 32 bytes of data:

Reply from 64.226.214.168: bytes=32 time=50ms TTL=114
Reply from 64.226.214.168: bytes=32 time=60ms TTL=114
Reply from 64.226.214.168: bytes=32 time=41ms TTL=114
Reply from 64.226.214.168: bytes=32 time=40ms TTL=114

Ping statistics for 64.226.214.168:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 40ms, Maximum =  60ms, Average =  47ms

C:\>_
```

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 216.30.120.1. If PING works with the IP address but not with the web site name, you know you've got a DNS problem.

Once you've determined 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 dialups 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 or WINIPCONFIG 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/WINIPCONFIG match those settings, you 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 Windows (except 9*x*), Linux, and NetWare. 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 Network+, there are a few places where NSLOOKUP makes for a great basic tool. For instance, just running NSLOOKUP alone shows you some output similar to the text shown here:

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

>
```

NOTE NSLOOKUP has its own prompt. To get back to a command prompt, type **exit** and press ENTER.

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 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 *dig*—short for *domain information groper*. Dig is very similar to NSLOOKUP, but it runs non-interactively. 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]$dig -x 13.65.14.4
 ; <<>> DiG 8.2 <<>> -x
 ;; res options: init recurs defnam dnsrch
 ;; got answer:
 ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4
 ;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2
 ;; QUERY SECTION:
 ;;      4.14.65.13.in-addr.arpa, type = ANY, class = IN
 ;; ANSWER SECTION:
 4.14.65.13.in-addr.arpa.  4H IN PTR  server3.houston.totalsem.com.
 ;; AUTHORITY SECTION:
 65.14.4.in-addr.arpa.  4H IN NS  kernel.risc.uni-linz.ac.at.
 65.14.4.in-addr.arpa.  4H IN NS  kludge.risc.uni-linz.ac.at.
 ;; ADDITIONAL SECTION:
 kernel.risc.uni-linz.ac.at.  4H IN A  193.170.37.225
 kludge.risc.uni-linz.ac.at.  4H IN A  193.170.37.224
 ;; Total query time: 1 msec
 ;; FROM: kernel to SERVER: default -- 127.0.0.1
 ;; WHEN: Thu Feb 10 18:03:41 2000
 ;; MSG SIZE  sent: 44  rcvd: 180
[mike@localhost]$
```

DHCP

Earlier, you saw that *Dynamic Host Configuration Protocol (DHCP)* could somehow magically take all your TCP/IP setup blues away by enabling any system to get all its necessary TCP/IP settings automatically from a DHCP server. In this section, we take a look at a DHCP server, consider some of the configuration issues, and then review some basic DHCP troubleshooting techniques.

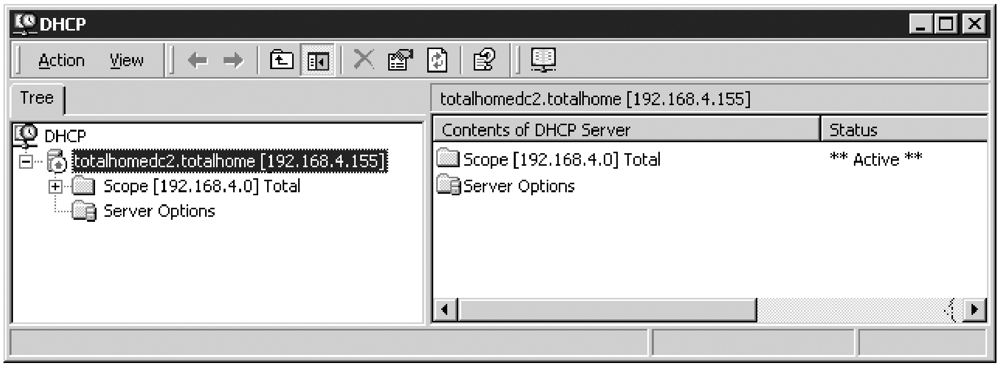
DHCP in Detail

DHCP is Microsoft's answer to TCP/IP client autoconfiguration. The DHCP server stores IP information and disperses it to the client systems on the network. At first glance, you might think that DHCP gives out only IP addresses—but in fact, it can give out all types of IP information: IP addresses, default gateways, DNS servers, and so on. While DHCP is useful, it's important to note that the network's client systems aren't required to use DHCP. It's no problem at all if some systems on the network get their IP information from the DHCP server, while others use static IP information. Finally, a system can use DHCP to obtain some IP information, and use static IP information for other purposes—there's no rule that a single system must use only DHCP or static IP information.

The beauty of DHCP is that it massively reduces the administration of a TCP/IP network. If you need to change the IP address of your default gateway, you could find yourself facing hours of work in a network where all the systems have static IP information. If you have DHCP set up to provide IP information dynamically, on the other hand, the DHCP would update all those PCs for you.

DHCP Servers

DHCP Servers are common. You'll find a DHCP server built into NetWare, Linux, and Windows server versions, as well as in most routers. For an example, I'll show you the DHCP server that comes with Windows 2000 Server. At first glance, you might think you're looking at the DNS server program, because you only see the name of the server running the DHCP server program (see Figure 14-17).

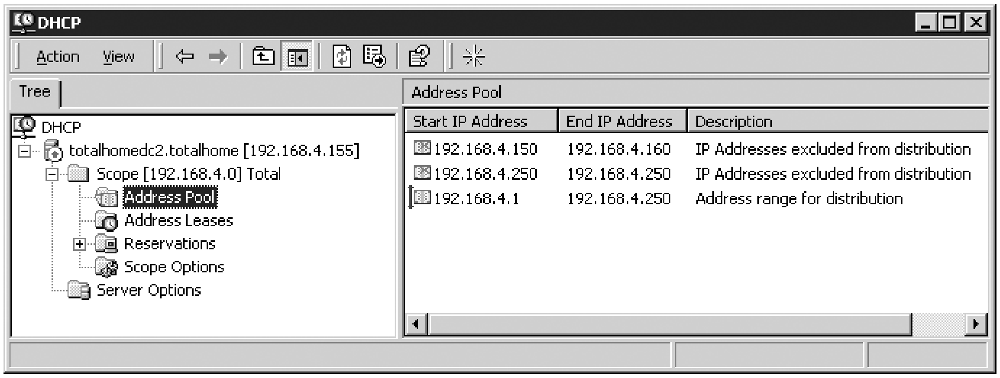

DHCP
Action
View
Tree
DHCP
totalhomedc2.totalhome [192.168.4.155]
Scope [192.168.4.0] Total
Server Options

totalhomedc2.totalhome [192.168.4.155]

Contents of DHCP Server	Status
Scope [192.168.4.0] Total	** Active **
Server Options	

Figure 14-17 The Windows 2000 DHCP server program

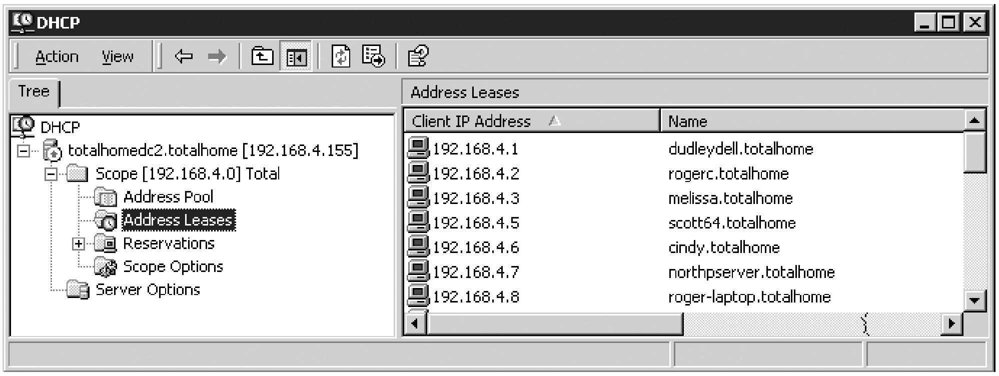
If you click the name of the DHCP server, you'll see the cornerstone of DHCP: the DHCP scope. A *DHCP scope* is the pool of IP addresses that a DHCP server may allocate to clients requesting IP addresses, or other IP information like DNS server addresses. This DHCP server has a pool of IP addresses running from 192.168.4.1 up to 192.168.4.250 (see Figure 14-18). It passes out these IP addresses in order as they are requested; so the first system that asks for an IP address gets 192.168.4.1, the second one gets 192.168.4.2, and so on.



Start IP Address	End IP Address	Description
192.168.4.150	192.168.4.160	IP Addresses excluded from distribution
192.168.4.250	192.168.4.250	IP Addresses excluded from distribution
192.168.4.1	192.168.4.250	Address range for distribution

Figure 14-18 DHCP scope—a pool of IP addresses

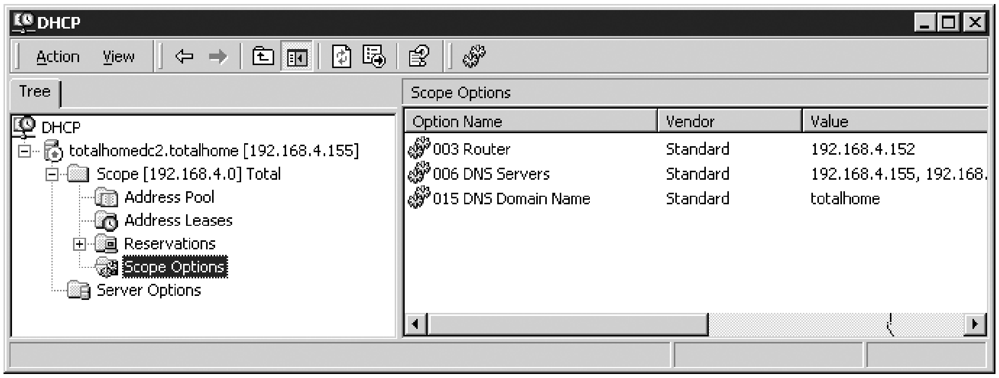
When a system requests DHCP IP information, the DHCP server creates a *DHCP lease* for the requested IP information; this means the client may use these settings for a certain amount of time. Windows 2000 and 2003 set the lease duration to eight days by default (NT set it to three days). To see the systems currently leasing DHCP IP addresses, look under Address Leases in DHCP (see Figure 14-19).



Client IP Address	Name
192.168.4.1	dudleydell.totalhome
192.168.4.2	rogerc.totalhome
192.168.4.3	melissa.totalhome
192.168.4.5	scott64.totalhome
192.168.4.6	cindy.totalhome
192.168.4.7	northpserver.totalhome
192.168.4.8	roger-laptop.totalhome

Figure 14-19 Address leases showing currently leased DHCP IP addresses

Remember that DHCP does more than just provide dynamic IP addresses. DHCP is great at dispensing all sorts of IP information your system might need. Figure 14-20 shows these DHCP scope options, and that I have the options for Default Gateway (Router), DNS server, and domain information activated, as well as my WINS server. All this information is passed to DHCP clients when they get their dynamic IP addresses.



Option Name	Vendor	Value
003 Router	Standard	192.168.4.152
006 DNS Servers	Standard	192.168.4.155, 192.168.
015 DNS Domain Name	Standard	totalhome

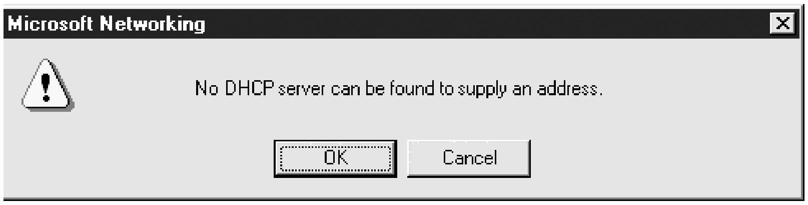
Figure 14-20 DHCP scope options

Troubleshooting DHCP

DHCP is a highly automated process that requires little configuration from the client side. This makes DHCP problems rare indeed. Once a system connects to a DHCP server and gets its dynamic IP information, it will run on those settings until the end of its lease period or until it reboots. If the DHCP client can't find a DHCP server to start or renew a lease, you'll get an error pointing to a DHCP problem. On reboot, you'll see something like the error shown in Figure 14-22.

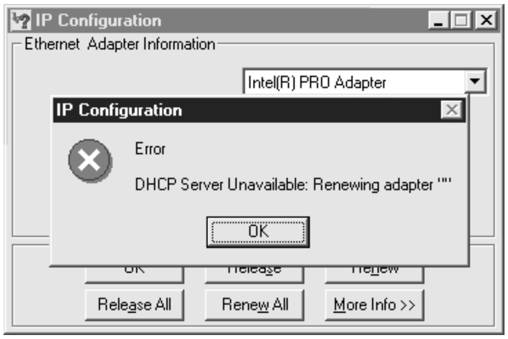
Any Windows 98 and later Windows client that is configured for DHCP, but unable to access a DHCP server, will always default to a special IP address starting with 169.254. This is the *Automatic Private IP Address (APIPA)*. APIPA enables DHCP clients that cannot find a DHCP server a chance at working by giving them an IP address in the 169.254/16

Figure 14-21 No DHCP server present



No DHCP server can be found to supply an address.

network. Without APIPA, a DHCP client simply will not get an IP address and will not function. If you think DHCP access is a problem, use IPCONFIG or WINIPCFG to check if your system is using an APIPA address. If you do this check and get a 169.254.x.x IP address, run the **IPCONFIG /RENEW** command, or click the Renew button in the WINIPCFG dialog box. If you get an error like the one shown in Figure 14-22, you're not getting to the DHCP server. Make sure you're connected to the network, and then contact the person in charge of the DHCP server.



Error

DHCP Server Unavailable: Renewing adapter ""

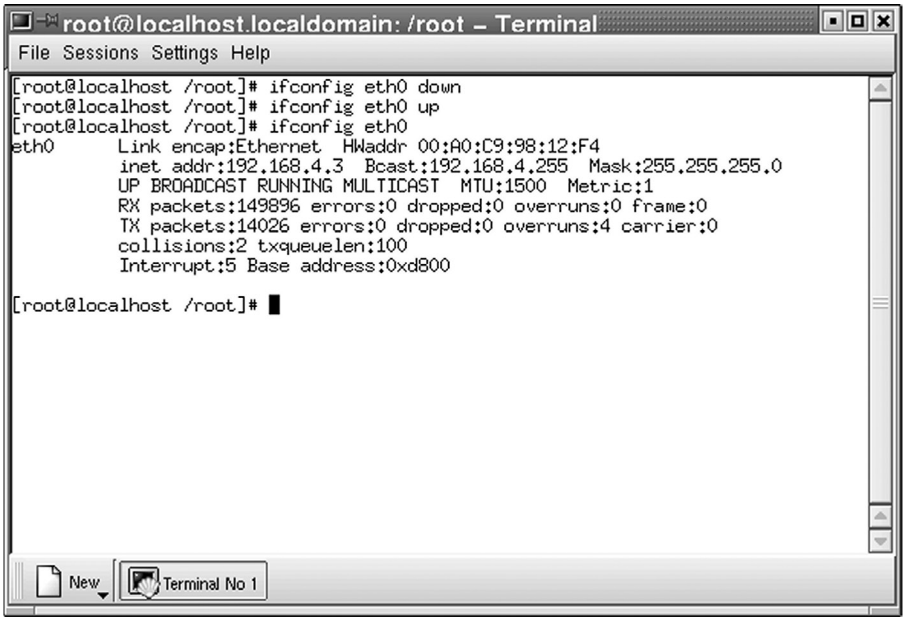
Figure 14-22
Renewing adapter

The one time DHCP will make some mistakes is during the initial setup. If you fail to provide the correct DNS server to your DHCP clients, they won't be able to resolve IP addresses. If you give them a pool of IP addresses that is not part of your network ID, they may not even be able to see other systems on the network. These aren't DHCP problems; they're regular IP configuration errors that happen to involve DHCP, which happily disperses them to all your networked systems.

Release or Renew?

WINIPCFG and **IPCONFIG** both come with the handy *release* and *renew* options. When do you release and when do you renew? This one's simple: if you know you're changing to a *new* DHCP server, first release, and then renew. If you know you're sticking with the same DHCP server, just renew. Linux users don't have the handy release and renew options, so they have to turn the NIC off and back on again to get the same result. To do this, run the **IFCONFIG eth0 down** command (*eth0* is what Linux names your NIC; if you have multiple NICs, the second would be *eth1*, and so on), and then the **IFCONFIG eth0 up** command. When the NIC comes back on, it will automatically try to renew the DHCP lease (see Figure 14-23).

PART III



root@localhost.localdomain: /root – Terminal
File Sessions Settings Help
New
Terminal No 1

```
[root@localhost /root]# ifconfig eth0 down
[root@localhost /root]# ifconfig eth0 up
[root@localhost /root]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:A0:C9:98:12:F4
          inet addr:192.168.4.3  Bcast:192.168.4.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:149896 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14026 errors:0 dropped:0 overruns:4 carrier:0
          collisions:2 txqueuelen:100
          Interrupt:5 Base address:0xd800

[root@localhost /root]#
```

Figure 14-23 Renewing a DHCP lease in Linux using the IFCONFIG up command

WINS

Remember *Windows Internet Naming Service (WINS)*? It resolves NetBIOS names to IP addresses. WINS drives many network types absolutely crazy, because it's one of those services that most networks simply don't need. Because only Windows networks run NetBIOS, WINS only operates in pure or nearly pure Windows networks. A Windows network must be running both IP and NetBIOS to need WINS, and even in those cases, you may not need WINS, because NetBIOS clients broadcast over a single segment. The latest versions of Windows—2000 and XP—have dumped native support for WINS, instead relying completely on DNS, except when running in networks with Windows 9*x* or NT systems. Bottom line? If you've never seen a WINS server before, this may be your only chance.

TIP Even though WINS is fading away, the Network+ exam still expects you to know it!

WINS in Detail

Let's review what we know about NetBIOS. In a simple NetBIOS network—no matter what protocol you're running—a NetBIOS system claims a NetBIOS name for itself simply by broadcasting out to the rest of the network (Figure 14-24). 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 remember that this example presumes a single network on the same wire, so it's okay in this context.

Remember that 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 (the network protocol NetBIOS was designed for) 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: they redesigned NetBIOS to work with TCP/IP. Let's look at some of the strategies and techniques they used to make NetBIOS and TCP/IP coexist on the same network.

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 TCP/IP names 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 WINS servers. 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 14-25 shows the copy of the WINS server that comes with Windows 2000 Server. Note that the PCs on this network have registered their names with the WINS server.

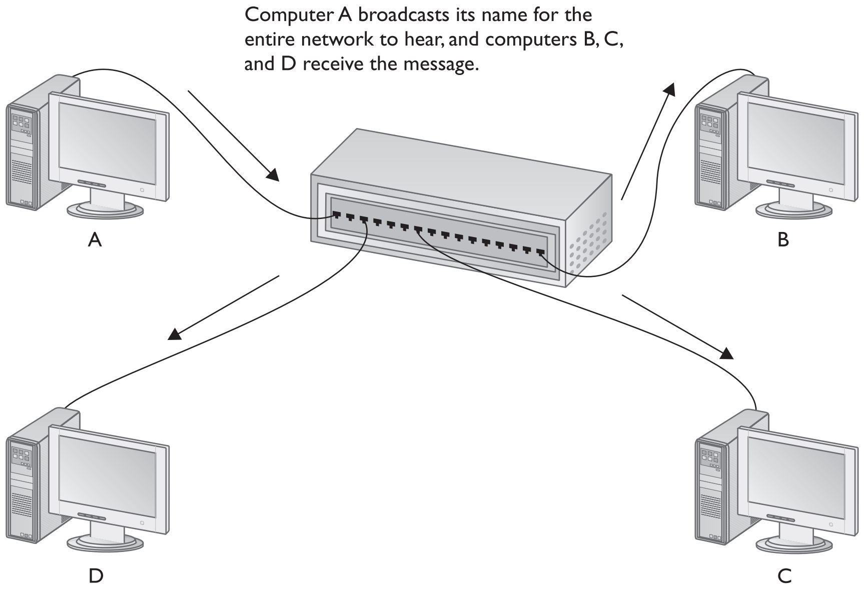
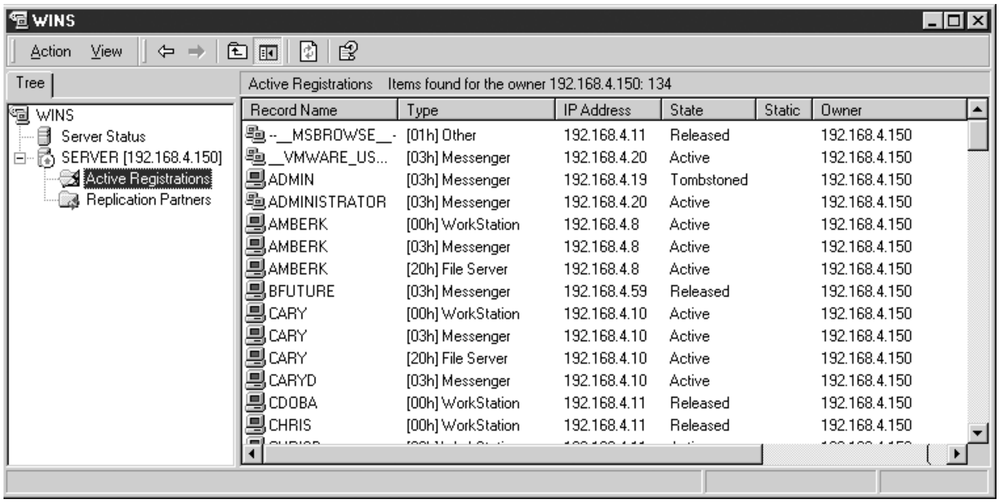

Computer A broadcasts its name for the entire network to hear, and computers B, C, and D receive the message.
A
B
D
C

Figure 14-24 A NetBIOS system broadcasting its name



Active Registrations Items found for the owner 192.168.4.150: 134

Record Name	Type	IP Address	State	Static	Owner
--__MSBROWSE__-	[01h] Other	192.168.4.11	Released		192.168.4.150
__VMWARE_US...	[03h] Messenger	192.168.4.20	Active		192.168.4.150
ADMIN	[03h] Messenger	192.168.4.19	Tombstoned		192.168.4.150
ADMINISTRATOR	[03h] Messenger	192.168.4.20	Active		192.168.4.150
AMBERK	[00h] WorkStation	192.168.4.8	Active		192.168.4.150
AMBERK	[03h] Messenger	192.168.4.8	Active		192.168.4.150
AMBERK	[20h] File Server	192.168.4.8	Active		192.168.4.150
BFUTURE	[03h] Messenger	192.168.4.59	Released		192.168.4.150
CARY	[00h] WorkStation	192.168.4.10	Active		192.168.4.150
CARY	[03h] Messenger	192.168.4.10	Active		192.168.4.150
CARY	[20h] File Server	192.168.4.10	Active		192.168.4.150
CARYD	[03h] Messenger	192.168.4.10	Active		192.168.4.150
CDOBA	[00h] WorkStation	192.168.4.11	Released		192.168.4.150
CHRIS	[00h] WorkStation	192.168.4.11	Released		192.168.4.150

Figure 14-25 The WINS server that comes with Windows 2000 Server

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

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 relay agent*, forwarding WINS broadcasts to a WINS server on the other side of the router (see Figure 14-26).

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

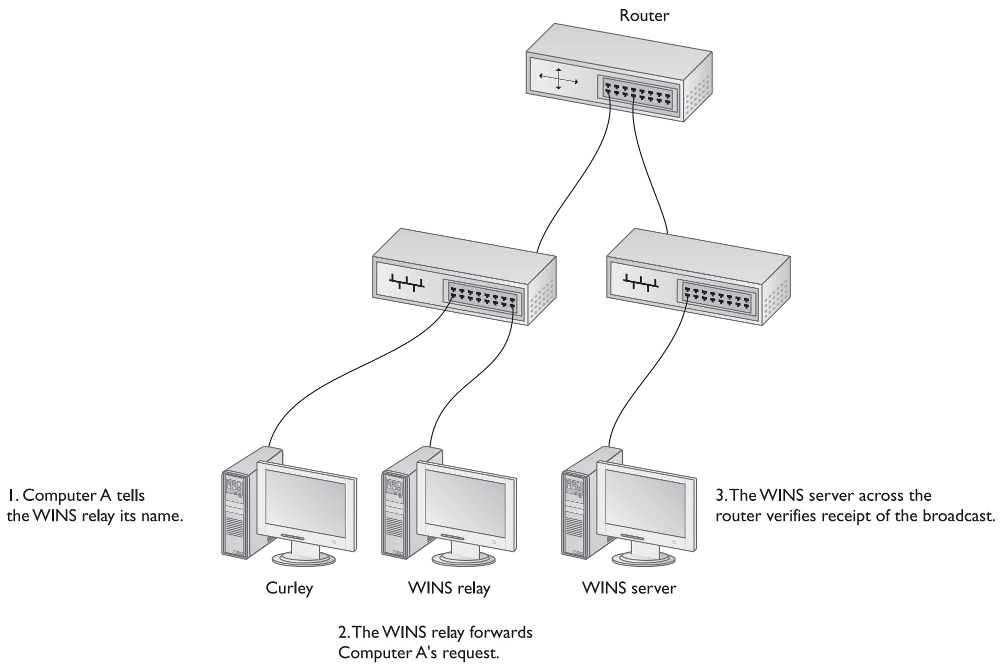

Router
1. Computer A tells the WINS relay its name.
3. The WINS server across the router verifies receipt of the broadcast.
Curley
WINS relay
WINS server
2. The WINS relay forwards Computer A's request.

Figure 14-26 A WINS relay agent forwarding broadcasts across a router

add WINS information to DHCP if necessary, so unless you're running static IPs, you may never have to enter anything into your Windows clients to get WINS to work.

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!

The program we turn to for help with NetBIOS problems is called *NBTSTAT*. 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

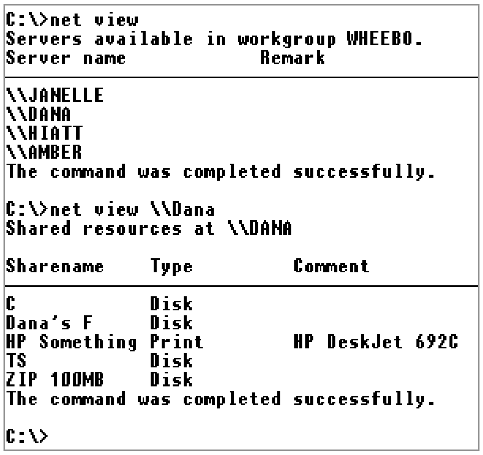
I've dedicated an entire chapter to network diagnostic procedures, but TCP/IP has a few little extras that I want to talk about here. TCP/IP is a pretty tough little protocol, and in good networks, it runs like a top for years without problems. Most of the TCP/IP problems you'll see come from improper 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 Network Neighborhoods, 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!" they cry. The first troubleshooting point to remember here: it doesn't matter *what* they can't see. It doesn't matter if they can't see other systems in Network Neighborhood, or they can't see the home page on their browser, because you go through the same steps in any event.

Remember to use common sense and more than one brain cell at a time 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 if 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 card'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 (see Figure 14-27).



```
C:\>net view
Servers available in workgroup WHEEBO.
Server name            Remark
-------------------------------------------------------
\\JANELLE
\\DANA
\\HIATT
\\AMBER
The command was completed successfully.

C:\>net view \\Dana
Shared resources at \\DANA

Sharename    Type         Comment
-------------------------------------------------------
C            Disk
Dana's F     Disk
HP Something Print        HP DeskJet 692C
TS           Disk
ZIP 100MB    Disk
The command was completed successfully.

C:\>
```

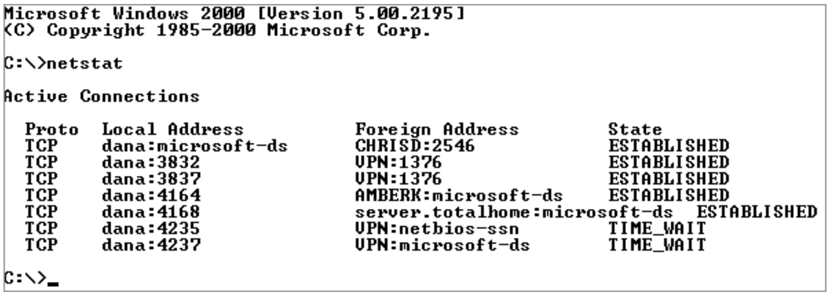
Figure 14-27 NET VIEW shows other local systems.

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.

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.

TIP 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.

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 will show 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 14-28 shows a NETSTAT program running.

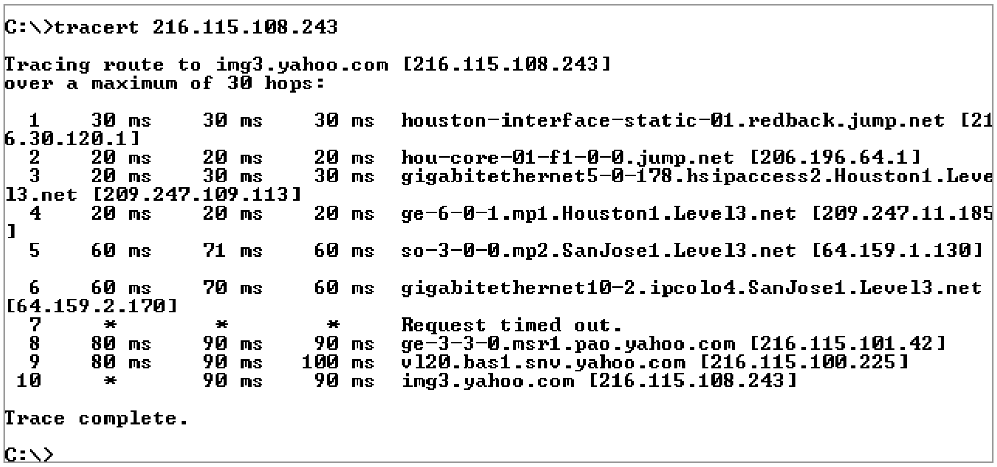


```
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    dana:microsoft-ds      CHRISD:2546            ESTABLISHED
  TCP    dana:3832              VPN:1376               ESTABLISHED
  TCP    dana:3837              VPN:1376               ESTABLISHED
  TCP    dana:4164              AMBERK:microsoft-ds    ESTABLISHED
  TCP    dana:4168              server.totalhome:microsoft-ds  ESTABLISHED
  TCP    dana:4235              VPN:netbios-ssn        TIME_WAIT
  TCP    dana:4237              VPN:microsoft-ds       TIME_WAIT

C:\>_
```

Figure 14-28 NETSTAT

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.

3. *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.
4. *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, it will tell you where the problem lies (see Figure 14-29).



```
C:\>tracert 216.115.108.243

Tracing route to img3.yahoo.com [216.115.108.243]
over a maximum of 30 hops:

  1    30 ms    30 ms    30 ms  houston-interface-static-01.redback.jump.net [21
6.30.120.1]
  2    20 ms    20 ms    20 ms  hou-core-01-f1-0-0.jump.net [206.196.64.1]
  3    20 ms    30 ms    30 ms  gigabitethernet5-0-178.hsipaccess2.Houston1.Leve
l3.net [209.247.109.113]
  4    20 ms    20 ms    20 ms  ge-6-0-1.mp1.Houston1.Level3.net [209.247.11.185
]
  5    60 ms    71 ms    60 ms  so-3-0-0.mp2.SanJose1.Level3.net [64.159.1.130]

  6    60 ms    70 ms    60 ms  gigabitethernet10-2.ipcolo4.SanJose1.Level3.net
[64.159.2.170]
  7     *        *        *     Request timed out.
  8    80 ms    90 ms    90 ms  ge-3-3-0.msr1.pao.yahoo.com [216.115.101.42]
  9    80 ms    90 ms   100 ms  v120.bas1.snv.yahoo.com [216.115.100.225]
 10     *       90 ms    90 ms  img3.yahoo.com [216.115.108.243]

Trace complete.

C:\>
```

Figure 14-29 Simple TRACERT

Chapter Review

Questions

1. NetBIOS uses what type of name space?
   A. Hierarchical name space
   B. People name space

C. DNS name space

D. Flat name space

2. The DNS root directory is represented by what symbol?

   A. . (dot)

   B. / (forward slash)

   C. \ (back slash)

   D. $ (dollar sign)

3. What command do you use to see the DNS cache on a Windows 2000 or XP system?

   A. winipcfg /showdns

   B. ipconfig /showdns

   C. ipconfig /displaydns

   D. winipcfg /displaydns

4. What do you call the pool of IP addresses that a DHCP server may allocate to client systems?

   A. DHCP pool

   B. DHCP scope

   C. DHCP array

   D. DHCP lease

5. What folder in the DHCS program lists the systems currently leasing DHCP IP addresses?

   A. Reservations

   B. Address Pool

   C. Address Leases

   D. Current Addresses

6. The users on your network haven't been able to connect to the server for 30 minutes. You check and reboot the server, but it's unable to ping either its own loopback address or any of your client systems. What should you do?

   A. Restart the DHCP server.

   B. Restart the DNS server.

   C. Replace the NIC on the server, because it has failed.

   D. Have your users ping the server.

7. What are the two reasons to use a WINS server?

   A. To reduce overhead from broadcasts

   B. To facilitate broadcast of NetBIOS names
   C. To support Windows XP systems on IP networks
   D. To enable NetBIOS name resolution across routers

8. What command do you use to check the current NetBIOS name cache?
   A. netstat –n
   B. netstat –c
   C. nbtstat –n
   D. nbtstat –c

9. A user calls to say she can't see the other systems on the network when she looks in Network Neighborhood. You are not using NetBIOS. What are your first two troubleshooting steps?
   A. Ping the address of a known web site.
   B. Ping the loopback address to test her NIC.
   C. Ping several neighboring systems using both DNS names and IP addresses.
   D. Ping the IP addresses of the router.

10. When troubleshooting a network using NetBIOS, what command do you use to see if the other local systems are visible?
   A. nbtstat
   B. net view
   C. nbt view
   D. view local

Answers

1. **D.** NetBIOS uses a flat name space, while DNS servers use a hierarchical name space.
2. **A.** The DNS root directory is represented by a dot (.).
3. **C.** To see the DNS cache on a Windows 2000 or XP system, run the command **ipconfig /displaydns** at a command prompt.
4. **B.** A DHCP scope is the pool of IP addresses that a DHCP server may allocate to client systems.
5. **C.** The Address Leases folder in the DHCS program lists the systems currently leasing DHCP IP addresses. The Address Pool folder lists the range of available IP addresses.

6. **C.** You should replace the server's NIC, because it's bad. It doesn't need either DNS or DHCP to ping its loopback address. Having the users ping the server is also pointless, as you already know they can't connect to it.
7. **A, D.** Two reasons to use a WINS server are to reduce overhead from broadcasts and to enable NetBIOS name resolution across routers. WINS servers eliminate the need for broadcasting NetBIOS names. They are needed by Windows 9*x* and NT systems running on IP networks; WINS is not even native to Windows 2000 and XP.
8. **D.** You use the **nbtstat –c** command to check the current NetBIOS name cache.
9. **B, C.** Your first two troubleshooting steps are to ping the loopback address—to check the client's NIC—and to ping neighboring systems. If the NIC and the local network check out, then you might try pinging the router and a web site, but those are later steps.
10. **B.** When troubleshooting a network using NetBIOS, use the **net view** command to see if the other local systems are visible.