

Chapter 4 Name Resolution

In this chapter, you will:

- ◆ Understand the domain name service (DNS)
- ◆ Identify the components of DNS
- ◆ Install and configure DNS in Windows and Linux
- ◆ Understand name resolution
- ◆ Configure zone files in Windows and Linux
- ◆ Troubleshoot DNS

To allow users to access a Web site, Web server administrators need to know how to resolve names. In general, name resolution involves taking a common name of a network resource—a Web server, for instance—and converting it into a corresponding IP address. This scheme is convenient for computer users, because they remember names more easily than complex numbers. The Web server administrator is usually the person who makes sure that all servers connected to the Internet have appropriate names and reference valid IP addresses so that users can access them.

You can use the domain name service (DNS) for name resolution. The primary purpose of DNS is to resolve an Internet name such as *www.redhat.com* to its corresponding IP address. Since 2000, Microsoft uses DNS to resolve computer names on a local area network (LAN). In the computer lab, you need to set up DNS servers for the operating systems you installed in Chapter 3 so you can simulate connecting to the servers using names such as www.technowidgets.com instead of IP addresses. Although browsers do not require that you use DNS names, e-mail servers do. That is, you can type `http://192.168.0.100` in a browser, but you cannot use ajones@192.168.0.100 in an e-mail client.

UNDERSTANDING THE DOMAIN NAME SERVICE (DNS)

You use DNS every time you surf the Internet, although you might not be aware of it. If you type *www.redhat.com* in your browser, for example, the DNS server translates that text into 66.187.232.56 because your request for the page must be sent to that IP address. In the following sections, you will learn about DNS components and their operation. First, examine the structure of the system.

DNS works like a telephone directory service. That is, just as a phone book or cell phone contact list correlates a person's name and phone number, DNS resolves common names for network resources to corresponding IP addresses. This process of converting a name to a numeric IP address, called **name resolution**, is convenient for Internet users—it is much easier to remember *www.linux.org* than

198.182.196.56. Name resolution also makes it easy for an administrator to move a server from one IP address to another. Without a naming service, the administrator would need to inform everyone using the server that the address had changed. With DNS, the administrator simply changes a record in a DNS configuration file and does not need to inform users.

While the best-known function of DNS is an Internet-wide service that converts host names into their corresponding IP addresses in browsers, DNS serves other important Internet functions. For example, it finds the IP addresses of e-mail servers for e-mail client software.

DNS is needed for more than the Internet. In a LAN, for example, computers must communicate with each other; therefore, the network requires a central directory of all its computers and their associated IP addresses. Windows Server supports Dynamic DNS (DDNS), which updates DNS automatically when the IP address of a workstation changes or a new workstation is added to the network.

Examining the Structure of the Internet Domains

DNS is organized into a hierarchical structure that defines domains. Likewise, the file system on your computer is arranged in a hierarchy. For example, the C: drive may have a folder called Program Files; within that folder could be a folder called Microsoft Office; and so on. Thus, each folder may contain one or more files. Linux uses a similar hierarchical structure. Just as you create folders to organize files, DNS arranges host names in a hierarchy to make them easier to manage and find. With tens of millions of hosts to organize, it faces a challenging task. The DNS hierarchical naming system consists of three levels:

- *Root level*—This level is the top of the hierarchy. The root is expressed by a period (“dot”). In common use, the trailing period is removed from domain names, but when you configure DNS services you must include it.
- *Top-level domain (TLD)*—This level identifies the most general portion of the domain name. It is the last part of the domain name—for example, com, edu, or org.
- *Second-level domain (SLD)*—This level identifies an entity within a top-level domain. The second-level domain name includes the top-level domain. Second-level domains can also be divided into further domain levels, called **subdomains**, as in the URL *www.arda.jones.name*. In this case, jones.name is the second-level domain controlled by the .name TLD, and arda.jones.name represents the subdomain that a person can register.

Identifying Top-Level Domains

Recall that a top-level domain identifies the most general part of the domain name, which is the highest category used to distinguish domain names. Table 4-1 lists the original top level domains approved by the Internet Corporation for Assigned Names and Numbers (ICANN).

Table 4-1 Top-level Internet domains

Top-level domain	Description
com	Commercial organizations
edu	Educational institutions
gov	Government institutions
mil	Military
net	Network support centers (ISPs)
org	Other organizations (originally nonprofit)
in-addr.arpa	Used for reverse lookups; that is, given an IP address, it finds the name

For a complete list of TLDs, see www.iana.org/cctld/cctld-whois.htm.

Identifying Second-Level Domains

Second-level domains include businesses and institutions that register their domain names with top-level domains through their respective registrars. They include registered names such as *iso.ch* and *amazon.com*. They can also be subcategories of top-level domains. For example, the United States domain (us) is categorized into second-level domains for each state, such as *ca.us* for California. Companies and academic institutions in the United Kingdom and most other countries are also categorized using second-level domains such as *co.uk* and *ac.uk*. Thus in the United Kingdom, companies and academic institutions can register names under their respective second-level domain. A subdomain is a further division of a second-level domain. In other words, a company that registers a domain can divide it into subdomains. For example, a subdomain of *technowidgets.com* may be *support.technowidgets.com*, and a host computer of this subdomain may be identified as *www.support.technowidgets.com*. However, on the Internet subdomains created by owners of domain names are not common. An organization might create subdomains when it has autonomous divisions that run their own Web servers and e-mail, yet want to be recognized as part of the domain. For example, the people at TechnoWidgets Inc. could create a host called *www.support* within *technowidgets.com* to use as a subdomain for TechnoWidgets Inc. Figure 4-1 shows a sample DNS namespace structure.

While many second-level domains, such as the ones under .com, need to be registered through a designated Internet authority such as *www.register.com*, you can create as many subdomains as you like within your own domain. The subdomain information is then propagated through the DNS system.

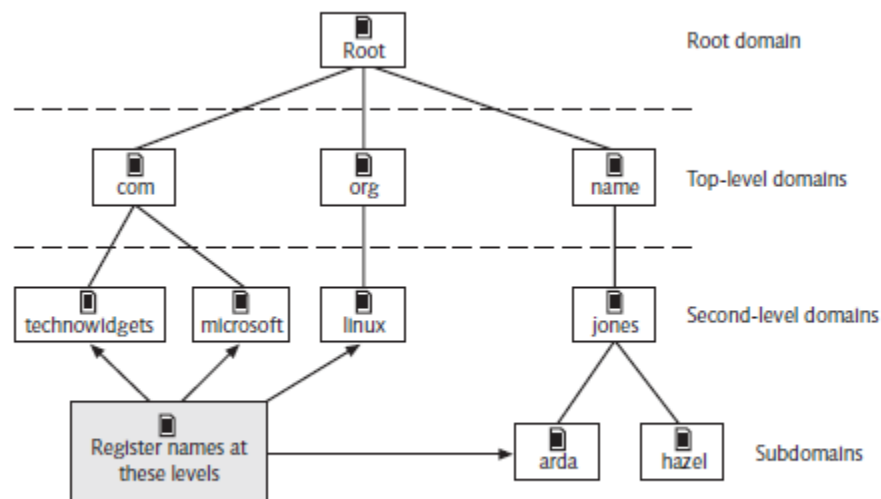


Figure 4-1 Structure of the DNS namespace

Understanding Host Names

The first portion of a URL is typically a host name. For example, in *www.technowidgets.com*, the *www* represents the Web server. Likewise, an FTP server could be called *ftp.technowidgets.com*. Although the Web server and the FTP server could reside on different computers, they could also be on the same computer with different IP addresses or even on the same computer with the same IP address, depending on how the DNS administrator configured the system.

When you access a host by using its DNS name, note that information moves from specific to general as you read from left to right. For example, to access a domain of *technowidgets.com* in a browser, you would enter *www.technowidgets.com*. The name “*www*” is most specific, because it refers to a particular host acting as a Web server. The name “*technowidgets*” refers to an entire domain, and “*com*” means it is one of many commercial entities. Figure 4-2 shows the components of www.technowidgets.com.

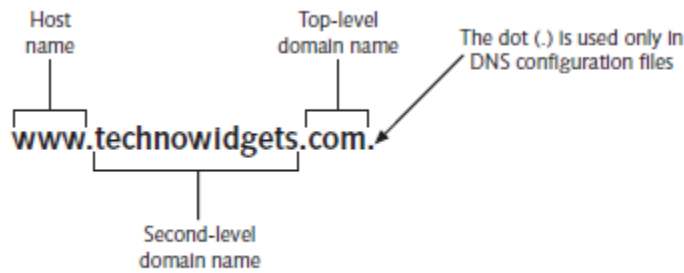


Figure 4-2 Components of a URL

Figure 4-3 shows what happens when a user types *www.technowidgets.com* into a Web browser. DNS is a distributed database of IP addresses; the root servers know only the addresses for all of the top-level domain servers. Each TLD server (such as for com) knows only the addresses for the second-level domain servers, such as the one for *technowidgets.com*. Finally, the DNS server for *technowidgets.com* returns the IP address for *www*. Every domain name such as *technowidgets.com* must have associated DNS servers. If you worked at TechnoWidgets, it could be your job to configure the *technowidgets.com* DNS server.

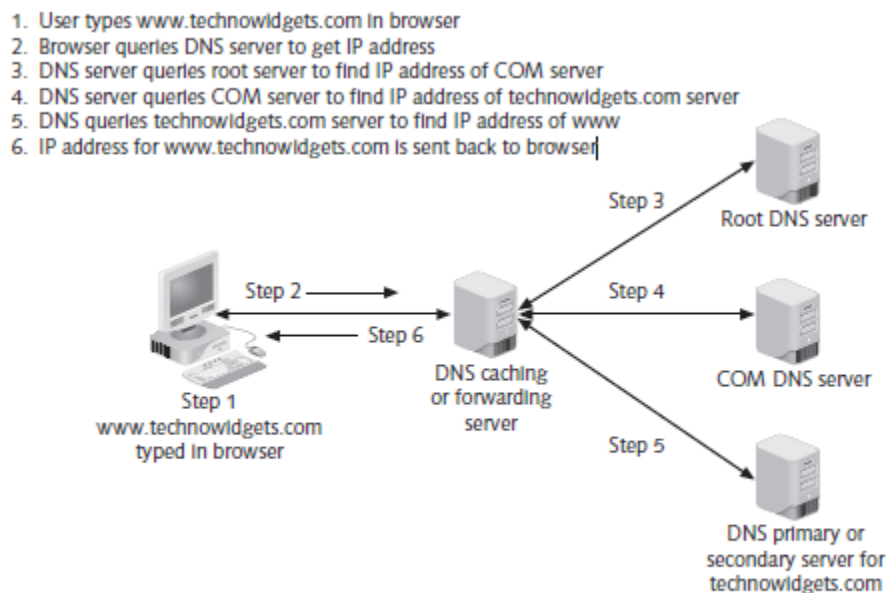


Figure 4-3 Finding the IP address for *www.technowidgets.com*

IDENTIFYING THE COMPONENTS OF DNS

Now that you have explored the structure of the domain system, you are ready to explore the parts of the DNS system. It consists of two key components:

- **Name server**—Also known as a DNS server, the name server is an application that supports name-to-address and address-to-name translation. You will learn about the various types of name servers later in this chapter. When you configure DNS files, the term “name server” will be used within the files.

- **Name resolver**—Commonly called a DNS client, a name resolver is technically the client software component that uses the services of one or more name servers. Each client must know how to contact at least one name server so that the name resolver software can exchange query packets with the DNS server. When the client software needs to send a DNS query to look up an IP address for a given name, the resolver sends the query to the name server. The address of the name server is part of the TCP/IP configuration. Windows clients use the term “DNS server address” to describe this information, while Linux clients use the term “nameserver address”. `nameserver` is a single word in a Linux configuration.

DNS follows the standard client/server model—the client makes a request and the server fulfills it. DNS servers can fill several different roles, depending on the needs of an organization. No matter which role the server undertakes, however, the client must specify the IP address of the DNS server. Two categories of DNS servers exist. The first category includes primary and secondary DNS servers, which are necessary for the Internet to function. These servers contain the host names for an individual domain on the Internet. The second category of DNS servers includes caching-only and forwarding servers, which search the Internet for the host names. A special type of server called the **root server** identifies all top-level domains on the Internet. If a client requests information about a host in another domain, a caching-only or forwarding DNS server can communicate the request to the root server. The InterNIC determines which systems are root servers. You can obtain the list of Internet servers at <ftp.rs.internic.net/domain/named.cache>.

Understanding DNS Servers That Define the Internet

Primary and secondary servers define the hosts for a particular domain such as *technowidgets.com*. The primary server defines the domain and contains the host names and associated IP addresses for each host. The secondary server backs up and retrieves domain data from the primary server at regular intervals.

Working with Primary Servers

The **primary server** stores files for a domain. Configuration files refer to the primary server as a master server because a primary server is the authority for the current domain, meaning that it controls host names and updates to the secondary server. The primary server maintains the DNS databases for its DNS **zone**, the set of records contained within a domain. For example, if the domain name for your company is *technowidgets.com*, then you need to create a forward lookup zone for the *technowidgets.com* domain on the primary server. In the zone files, you create records that have the host names and corresponding IP addresses for all hosts in the zone. If your organization decided to create subdomains, they would be maintained in separate zones.

Working with Secondary Servers

A **secondary server** receives its authority and database from the primary server. It provides fault tolerance, load distribution, and remote name resolution for the primary DNS server. When the secondary server first starts, it requests from the primary server all the data for a zone. The secondary server then periodically checks with the primary name server to determine whether it needs to update its data. Configuration files refer to the secondary server as a slave server. Each primary server can have multiple secondary servers.

Understanding DNS Servers That Resolve Names

Primary and secondary servers rely on other servers to search the data they store. The caching-only and forwarding servers search primary and secondary servers. When you install a DNS server, it is a caching-only server by default, so no extra configuration is needed. You can configure a caching-only server to use a forwarding server for resolving names instead of using the root servers on the Internet. In addition, you can combine the caching-only server that is configured by default with a primary server, secondary

server, or forwarding server. However, caching-only and forwarding DNS servers can be used in organizations that do not have domain names. Remember, they are used only to resolve names.

Working with Caching-Only Servers

A **caching-only server** is not authoritative for any zone. Instead, it handles queries by asking other servers for information. All servers cache the information they receive until the time specified in the Time to Live (TTL) field expires. That is, a caching-only server stores name resolution information until the data expires. Caching-only servers can be used in organizations in which many users connect to the Internet and access many common sites. Using a local caching-only server can significantly reduce response time for URL resolution. If you install a DNS server in Linux or Windows, it remains a caching-only server until you modify it to become a primary or secondary server.

Working with Forwarding Servers

Forwarding servers, or forwarders, process requests that DNS servers cannot resolve locally. A forwarding server is not a separate type of server, but rather a caching-only server used in a particular way. A forwarding server accesses the Internet, as shown in Figure 4-4. To make a DNS server become a forwarding server, you add a record to reference the DNS server on the Internet that will resolve names. You then add records in the caching-only servers that reference the forwarding server.

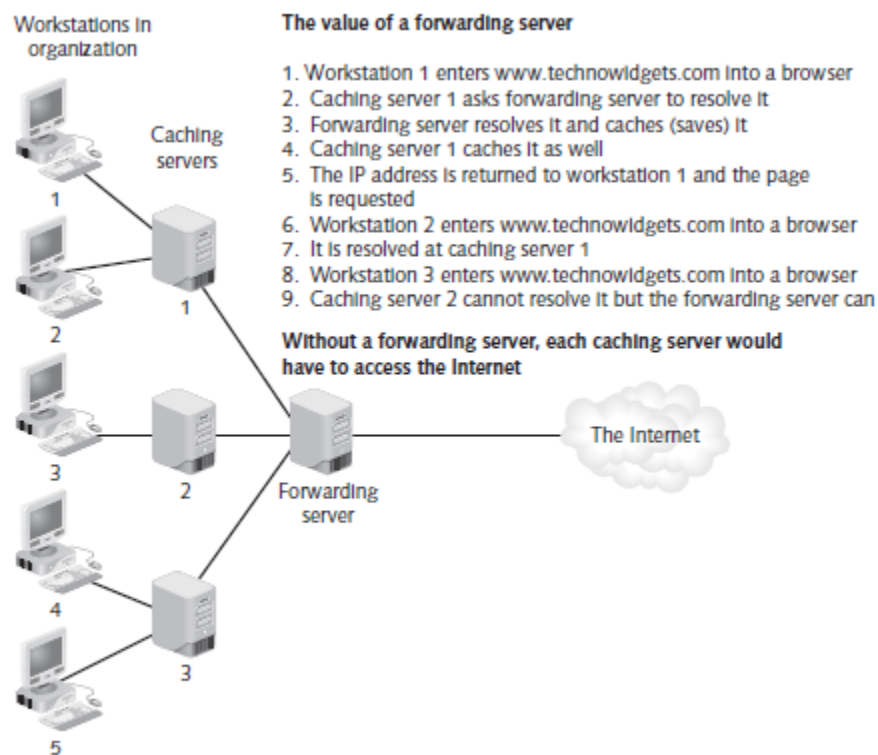


Figure 4-4 Caching servers using a forwarding server

Forwarding servers work in combination with caching-only servers. They are useful in organizations that have many caching-only servers but want to reduce traffic on their connection to the Internet. If all the caching-only servers cached names and IP addresses independently, then information would be duplicated because each caching-only server would access the Internet to resolve names. A forwarding server acts as the single point of contact for the Internet, so it caches often-used names requested by the other servers. For example, if the users of five caching-only servers wanted to access www.redhat.com, then each

caching-only server would have to contact the Internet to resolve the address. If the caching-only servers used a forwarding server, then the forwarding server would get the IP address for *www.redhat.com* and cache it. When the other caching-only servers needed the IP address for *www.redhat.com*, they could get it from the cache at the forwarder.

The primary and secondary servers are accessed by users throughout the Internet to determine the IP addresses of your hosts, such as *www.technowidgets.com*. The caching-only and forwarding servers are accessed by users in your organization to determine the IP addresses of hosts on the Internet.

CONFIGURING ZONE FILES

Recall that a zone file contains records that specify the host names and corresponding IP addresses for all hosts in the zone. Once you configure the zone, you can use the information in these files for name resolution, regardless of whether you use DNS in Linux or Windows.

Often, your ISP configures the zone files; in some cases, the ISP may have a Web interface to make it easy for you to change these files. If your organization operates its own primary DNS server, then you will configure the zone files directly. When you register a domain name, you must provide the addresses of a primary DNS server and a secondary DNS server. Smaller organizations often use only the DNS servers at their ISP. This is an easy but inflexible approach; for example, you would need to contact your ISP every time you changed your domain. If you determine that you want more control over changes to your domain, you can set up a primary DNS server in your organization and have your ISP maintain the secondary server. Controlling your own DNS server also makes it easier to move to a new ISP because you control all the records that you need to change.

A primary server is the authoritative server for managing information on IP addresses and associated host names in your domain. All other systems on your network should be configured so that the DNS server entry in the TCP/IP configuration has the IP address of your DNS server. You can also enter an IP address for a second DNS server, which will be used if the first server cannot be contacted. Typically this entry applies to the DNS server at your ISP.

Understanding Zone Files and DNS Records

A zone file helps define a branch of the DNS namespace under the administrative control of a primary DNS name server. A **namespace** is a common grouping of related names—for example, hosts within a LAN. Although many IT professionals use the terms “domain” and “zone” interchangeably, they are not identical. A zone refers to a specific file that resides on a server. For instance, a company that registers the domain name *technowidgets.com* gains administrative control of the domain *technowidgets.com*. To exert control over that domain, the company must create zones. When you create a DNS server, you generally populate it with two types of zones:

- *Forward lookup*—These zones contain entries that map names to IP addresses.
- *Reverse lookup*—These zones contain entries that map IP addresses to names.

Every domain zone file consists of DNS records. A DNS record is an entry in a DNS database on a primary server that provides additional routing and resolution information. You can configure many different types of records, but only a few are essential for full name resolution routing. Common DNS records are listed in Table 4-3.

Table 4-3 Common DNS records

DNS record	Function
Address (A)	The most commonly used record; associates a host to an IP address. For example, you can establish an association between an IP address and a Web server by creating an address record.
Canonical name (CNAME)	Creates an alias for a specified host. As an example, suppose the name of a WWW server is <i>server1.company.com</i> . (Web servers are commonly named WWW.) A CNAME record creates an alias for the <i>server1.company.com</i> host.
Internet (IN)	Identifies Internet records; precedes most DNS record entries.
Mail Exchanger (MX)	Identifies a server used for processing and delivering e-mail for the domain.
Name server (NS)	Identifies DNS servers for the DNS domain. It can also refer to a DNS server that will resolve names for clients, making the DNS server act as a forwarding server.
Pointer (PTR)	Performs reverse DNS lookups. The PTR record allows DNS to resolve an IP address to a host name.
Start of Authority (SOA)	Identifies the DNS server with the most current information for the DNS domain. Because several secondary DNS servers may exist, this record identifies the primary server for the specified DNS domain. Although RFC 1035, the document that defines DNS, specifies SOA as the "start of the zone of authority," many authors define SOA as the "statement of authority."

Do not confuse forward and reverse zone files with primary or secondary servers. A primary or secondary server usually *contains* a forward or reverse zone. Zone files make it possible to populate a DNS server, either primary or secondary, with individual host information.

Configuring the Forward Lookup Zone

The forward lookup zone maps host names to IP addresses. Although Windows uses a GUI and Linux uses text files, both must supply the same information so that any other DNS server can read the forward lookup zone. Figure 4-5 shows a sample forward lookup zone file for *technowidgets.com*. For the purposes of this example, IP addresses that are invalid on the Internet are used. Spacing is not important in the files. However, each name, such as @, web1, or www, must appear in the first column of the line.

```
$TTL      86400
@ IN SOA  web1.technowidgets.com. admn.technowidgets.com. (
    2002072100 ; Serial
    28800      ; Refresh
    14400      ; Retry
    3600000    ; Expire
    86400 )    ; Minimum
IN      NS      web1
```

	IN	A	192.168.0.100
	IN	MX 10	mail.technowidgets.com.
web1	IN	A	192.168.0.100
www	IN	CNAME	web1
www.support	IN	CNAME	web1
research	IN	A	192.168.0.150
	IN	MX 10	mail
mail	IN	A	192.168.0.200

Figure 4-5. Forward lookup zone for technowidgets.com.

Each statement except for the first line contains IN, which stands for the Internet class. Historically there were other classes but the Internet class is the only one used today. The \$TTL 86400 tells caching software how long to cache the resource records such as www. The time is measured in seconds. Starting with the first character on the next line, the @ signifies the name of the zone, which is *technowidgets.com*. This line defines the **Start Of Authority (SOA)** record; it states that the primary DNS server resides on *web1.technowidgets.com*. The e-mail address of the contact person for this domain is admn@technowidgets.com.

The next five items define the relationship between the primary server and the secondary server. Other than the first item, the numbers are represented in seconds. Note that 28,800 seconds is 8 hours, 3,600,000 seconds is 1000 hours, and 86,400 seconds is 24 hours.

- The **serial number** can be any valid 32-bit number. When you change the DNS configuration, changing the serial number informs the secondary server that it should update its database. The standard format for the serial number is YYYYMMDDnn, where YYYYMMDD is the date of the configuration change and nn represents a sequential number allowing up to 100 changes per day.
- The **refresh** interval tells the secondary server how often to check for updates from the primary server.
- The **retry** interval describes how often the secondary server should try to contact the primary server if it fails to make contact after the refresh interval.
- The secondary server also needs to know how long to keep trying to contact the primary server before giving up and stopping all requests for name resolution. This information is provided by the value for **expire**.
- The last item, **minimum**, refers to the length of time that a negative response to a query should remain cached.

The SOA record is followed by the line IN NS web1. Because nothing precedes IN in the line, it takes the value of the previous IN statement, which was the @, meaning *technowidgets.com*. The NS stands for name server, which is shorthand for DNS server, and web1 is the name of the host that has the DNS server. In other words, the line means that web1 has the DNS server for *technowidgets.com*. Notice that web1 does not end in a period, yet *web1.technowidgets.com* does. When a name ends in a period, it is called a **fully qualified domain name (FQDN)**, which is the complete name of the host. When no period appears at the end, the host name is given relative to the zone name. In this case, using web1 is the same as using *web1.technowidgets.com* for the name. Again notice the period after “com.”; if the first line does not end with a period, the server appends the zone name. In this case, the server would interpret the host name as *web1.technowidgets.com.technowidgets.com*. The most common error in configuring DNS

servers is forgetting the period at the end of the FQDN.

The next line, IN A 192.168.0.100, means that the name *technowidgets.com* resolves to the IP address 192.168.0.100. To refer mail for *technowidgets.com* to a mail server, you need a mail exchange record of IN MX 10 mail.technowidgets.com. Because the system could include multiple mail servers, the value of 10 helps determine which mail server is the primary one. If there were multiple MX records, they would have different numeric values. The one with the lowest value would be the first e-mail server contacted; if it did not respond, the next e-mail server in numeric order would be contacted.

So far, the record refers to web1 twice without describing the IP address of the host. It is finally described in the line web1 IN A 192.168.0.100, which tells you that web1 is located at 192.168.0.100. However, because it is not standard practice to have a URL of *web1.technowidgets.com*, you must create a **canonical name**, also known as an alias, which states that www is equivalent to web1. This alias is shown in the next line, www IN CNAME web1, which means that *www.technowidgets.com* is the same as *web1.technowidgets.com*. Likewise, the next line creates a CNAME (alias) for www.support, which equates it with web1.

The next line includes a host for research at 192.168.0.150. Because the line following research does not have a name to the left of IN, it takes the value of the previous line, which is “research”. It is an MX record, which specifies that mail sent to *research.technowidgets.com* should go to the same mail server. Finally, the last line describes the IP address of the mail server with an A record.

The previous example incorporates all the elements of a typical forward lookup zone file. You use many of the same elements to configure the reverse lookup zone file, so the next section covers only the unique parts of this file.

Configuring the Reverse Lookup Zone

Along with converting domain names into IP addresses, the DNS system can do the reverse: it can convert IP addresses into names. For example, suppose that an e-mail system receives an e-mail from bgates@microsoft.com. The IP address of the sender is 38.246.165.21. The e-mail system may want to know whether 38.246.165.21 is actually associated with *microsoft.com*. A reverse lookup can tell the system the domain of the IP address. A reverse lookup is also useful when using DNS-based troubleshooting utilities such as nslookup and dig. Note that a DNS requires a forward lookup zone but does not require a reverse lookup zone.

DNS converts IP addresses to names by associating a domain name with a network address and placing the domain name in the top-level in-addr.arpa domain. To implement reverse lookups, you create reverse zone files and populate them with PTR records in the proper format. For example, suppose that your company has the class C network address 192.168.0.0. The associated in-addr.arpa zone name is 0.168.192.in-addr.arpa. You create this name by reversing the order of the bytes in the network address and adding in-addr.arpa at the end. By placing this information in a PTR record, you create the proper reverse DNS entry for the host. Figure 4-6 shows an example.

```
$TTL      86400
@ IN SOA  web1.technowidgets.com. admn.technowidgets.com. (
                                2002072100 ; Serial
                                28800      ; Refresh
```

		14400	; Retry	
		3600000	; Expire	
		86400) ; Minimum	
	IN	NS		web1.technowidgets.com.
100	IN	PTR		web1.technowidgets.com.
150	IN	PTR		research.technowidgets.com.
200	IN	PTR		mail.technowidgets.com.

Figure 4-6. Reverse lookup zone for technowidgets.com.

Conventionally, the bytes in IP addresses move from general to more specific networks as you proceed from left to right. However, URLs are opposite—the most specific is on the left. For example, in the URL *www.technowidgets.com*, “www” represents the specific host. With the corresponding IP address, 192.168.0.100, the host portion of the address is 100, and is on the right. To have the most specific portion on the left, you must reverse the address. You already know that the @ represents the name of the zone, which is 0.168.192.in-addr.arpa. You also know that a name without a period at the end means that the zone name is appended to it. Thus the text 100 IN PTR web1.technowidgets.com. is the same as 100.0.168.192.in-addr.arpa. IN PTR web1.technowidgets.com. These addresses now match; web1 is on the left in both cases. Matching is necessary because the search is similar; just as searches always start at the top-level domain of com for forward lookups, so searches start at in-addr.arpa for reverse lookups.

Now that you have learned about the forward and reverse lookup zones in a DNS server, you are ready to learn how to set them up in Linux and Windows.

Activity 4-1. Install and Configure DNS in Ubuntu

Note: be sure you’ve completed Chapter 3 lab work, including setting a static IP address, DNS server settings, computer name, and hosts file.

- 1) DNS in Ubuntu is called bind9. To install DNS, run as super-user and install bind9:
sudo apt-get install bind9

Make note of which packages were installed and make sure bind9 starts with an [OK] message.

```

andersml@U14-fml:~$ sudo apt-get install bind9
[sudo] password for andersml:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  dnsmasq-base iputils-arping libmbim-glib0 libmm-glib0 libmnl0
  libnetfilter-conntrack3 libnl-route-3-200 libqmi-glib0
  mobile-broadband-provider-info modemmanager usb-modeswitch
  usb-modeswitch-data
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  bind9utils
Suggested packages:
  bind9-doc
The following NEW packages will be installed:
  bind9 bind9utils
0 upgraded, 2 newly installed, 0 to remove and 9 not upgraded.
Need to get 432 kB of archives.
After this operation, 1,632 kB of additional disk space will be used.
Do you want to continue? [Y/n]

Get:1 http://us.archive.ubuntu.com/ubuntu/ trusty/main bind9utils amd64 1:9.9.5.dfsg-3 [145 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu/ trusty/main bind9 amd64 1:9.9.5.dfsg-3 [287 kB]
Fetched 432 kB in 6s (70.6 kB/s)
Preconfiguring packages ...

Selecting previously unselected package bind9utils.
(Reading database ... 163271 files and directories currently installed.)
Preparing to unpack .../bind9utils_1%3a9.9.5.dfsg-3_amd64.deb ...
Unpacking bind9utils (1:9.9.5.dfsg-3) ...
Selecting previously unselected package bind9.
Preparing to unpack .../bind9_1%3a9.9.5.dfsg-3_amd64.deb ...
Unpacking bind9 (1:9.9.5.dfsg-3) ...
Processing triggers for man-db (2.6.7.1-1) ...
Processing triggers for ureadahead (0.100.0-16) ...
ureadahead will be reprofiled on next reboot
Processing triggers for ufw (0.34~rc-0ubuntu2) ...
Setting up bind9utils (1:9.9.5.dfsg-3) ...
Setting up bind9 (1:9.9.5.dfsg-3) ...
Adding group `bind' (GID 125) ...
Done.
Adding system user `bind' (UID 116) ...
Adding new user `bind' (UID 116) with group `bind' ...
Not creating home directory `/var/cache/bind'.
wrote key file "/etc/bind/rndc.key"
#
* Starting domain name service... bind9 [ OK ]
Processing triggers for ureadahead (0.100.0-16) ...
Processing triggers for ufw (0.34~rc-0ubuntu2) ...
andersml@U14-fml:~$

```

The software used for DNS in Linux and other non-Windows servers is called **BIND** (Berkeley Internet Name Domain). With version 14 of Ubuntu, we're installing the version 9 of BIND called bind9.

- 2) The DNS service can be restarted (or stopped, or started) using the command:

sudo service bind9 restart

```

andersml@U14-fml:~$ sudo service bind9 restart
[sudo] password for andersml:
* Stopping domain name service... bind9
waiting for pid 3299 to die [ OK ]

* Starting domain name service... bind9 [ OK ]
andersml@U14-fml:~$

```

Submit a screenshot of the service restarting

When you installed Linux, it added a number of files and directories to your system. Table 4-4 describes the most important files. Look at the `/etc/bind/named.conf.local` before modification.

File	Description
<code>/etc/init.d/bind9</code>	Shell script to run the DNS service
<code>/etc/bind/named.conf</code>	DNS configuration file with an include for <code>named.conf.local</code> for local configs
<code>/etc/bind/named.conf.local</code>	Local configs including references to your forward and reverse lookup zones
<code>/var/lib/bind/</code>	The folder where you'll save your forward and reverse lookup zones
<code>/etc/</code>	<code>/etc/</code> has system files like the DNS config file & Apache config file
<code>/var/</code>	<code>/var/</code> has user files like the DNS zone files and your web pages

Table 4-4

- 3) Next write the forward and reverse lookup zones for www.technowidgets.com using your static IP address and computer name. You may copy the content from examples in Figure 4-5 for your forward lookup zone and Figure 4-6 for your reverse lookup zone. However, you'll have to:
 - a. Copy the file from the textbook and then paste it into Notepad and copy it again. This will remove any invisible characters that may be hiding in your document. Then paste it into your Linux editor. You can paste from VMware Workstation using Edit > Paste. From VMware Player, SHIFT+INS may work.
 - b. Replace the server name *web1* with your server name (U14-fml where f is your first-name initial, m is your middle-name initial and l is your last-name initial).
 - c. Change the IP address in the examples (**192.168.0.100**) to match your static IP address. Leave the final octet for research (.150) and mail (.200) as is. These IP addresses aren't live but you will be able to do DNS table lookups (nslookups) on these subdomains. We will activate them in a later chapter.
 - d. Save the forward lookup file as `/var/lib/bind/named.technowidgets.com` – you can check a zone file's syntax with:

`sudo named-checkzone technowidgets.com /var/lib/named.technowidgets.com`

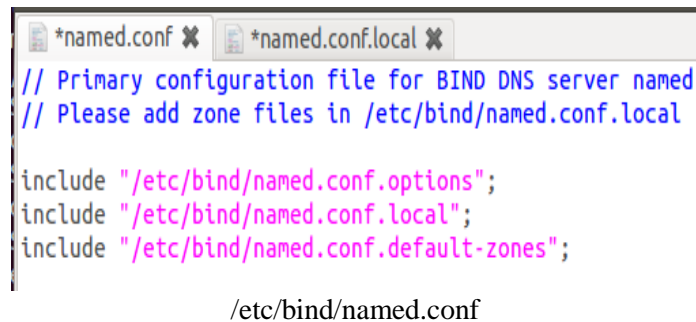
- e. Save the reverse lookup file in `/var/lib/bind/` but vary the name depending on your IP address. It will take the general form `/var/lib/bind/named.3.2.1` where 3 is the third octet of your IP address, 2 is the second octet of your IP# and 1 is the first octet of your IP address. You can use the **named-checkzone** command above to check your reverse lookup zone.

Submit screenshots of your forward and reverse lookup zones

After you write the zone files you identify their names and locations in the local DNS configuration file. We have used this standard naming convention for the zone files:

- The name of the forward lookup zone file started with “named” followed by a dot and then the name of the zone. Because the name of the forward lookup zone is *technowidgets.com*, you would name the file “named.technowidgets.com”.
- The name of the reverse zone started with “named” followed by a dot and then add the network address of the zone with the octets reversed; the name of the reverse lookup zone file in Figure 4-6 would then be “named.0.168.192”.

Notice the DNS configuration file named.conf includes a statement for /etc/bind/named.conf.local. This is where you enter the names and locations of your zone files.



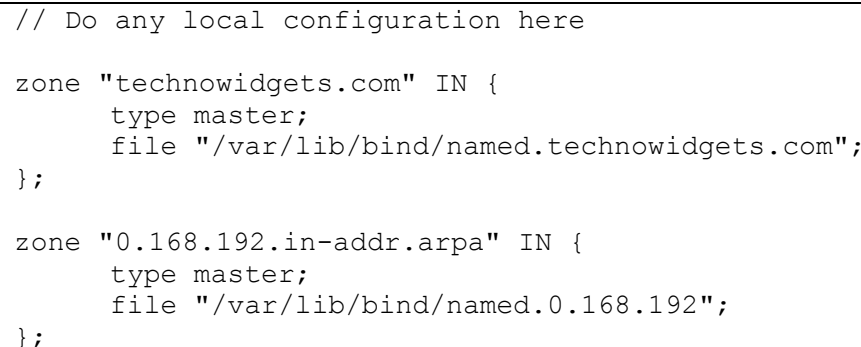
```
// Primary configuration file for BIND DNS server named
// Please add zone files in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

/etc/bind/named.conf

- 4) Rewrite your /etc/bind/named.conf.local as shown below but using the reverse lookup zone reference which matches your static IP address. Check the syntax of your named.conf* files with:

sudo named-checkconf



```
// Do any local configuration here

zone "technowidgets.com" IN {
    type master;
    file "/var/lib/bind/named.technowidgets.com";
};

zone "0.168.192.in-addr.arpa" IN {
    type master;
    file "/var/lib/bind/named.0.168.192";
};
```

/etc/bind/named.conf.local

Submit a screenshot of your modified /etc/bind/named.conf.local

- 5) After configuring the DNS server, restart the service
sudo service bind9 restart

Next you will configure the client portion of DNS. Recall that when the client software needs to resolve a name, it looks in the TCP/IP configuration for a DNS server. The server address is listed in dns-nameservers in the /etc/network/interfaces file. The table below shows the /etc/network/interfaces file for IP address is 192.168.0.100.

```
# This file describes the network interfaces

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
#iface eth0 inet dhcp
iface eth0 inet static
address                192.168.0.100
netmask                255.255.255.0
network                192.168.0.0
broadcast              192.168.0.255
gateway                192.168.0.2
dns-nameservers        192.168.0.100 8.8.8.8
```

/etc/network/interfaces

Modify the dns-nameservers line in your /etc/network/interfaces for **your static IP address**. After making changes to the network interfaces file, reboot your server.

sudo reboot

Submit a screenshot of your modified /etc/network/interfaces file

- 6) Troubleshoot your DNS setup. There is more than one way to make sure that DNS is working, and many of these techniques are similar in Windows and Linux. The most common technique is to use the **ping** utility, which you learned about in Chapter 3. Ping will verify that you have an active IP address and that DNS is resolving correctly. Another helpful utility is **nslookup**, which is available on both Linux and Windows. The nslookup utility can give you more information about a host name because it actually reads DNS information; for example, it can tell you that *www.technowidgets.com* is a canonical name for *web1.technowidgets.com*. However, nslookup only interrogates the DNS tables. It does not verify that the IP address is active.

```
andersml@u14-fml: ~  
andersml@u14-fml:~$ nslookup research.technowidgets.com  
Server:      192.168.136.129  
Address:     192.168.136.129#53  
  
Name:   research.technowidgets.com  
Address: 192.168.136.150  
  
andersml@u14-fml:~$ ping www.technowidgets.com  
PING U14-fml.technowidgets.com (192.168.136.129) 56(84) bytes of data.  
64 bytes from U14-fml.technowidgets.com (192.168.136.129): icmp_seq=1 ttl=64 time=0.041 ms  
64 bytes from U14-fml.technowidgets.com (192.168.136.129): icmp_seq=2 ttl=64 time=0.045 ms  
64 bytes from U14-fml.technowidgets.com (192.168.136.129): icmp_seq=3 ttl=64 time=0.046 ms  
64 bytes from U14-fml.technowidgets.com (192.168.136.129): icmp_seq=4 ttl=64 time=0.046 ms  
64 bytes from U14-fml.technowidgets.com (192.168.136.129): icmp_seq=5 ttl=64 time=0.043 ms  
^C  
--- U14-fml.technowidgets.com ping statistics ---  
5 packets transmitted, 5 received, 0% packet loss, time 4003ms  
rtt min/avg/max/mdev = 0.041/0.044/0.046/0.004 ms  
andersml@u14-fml:~$
```

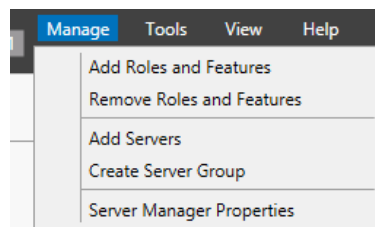
nslookup & ping

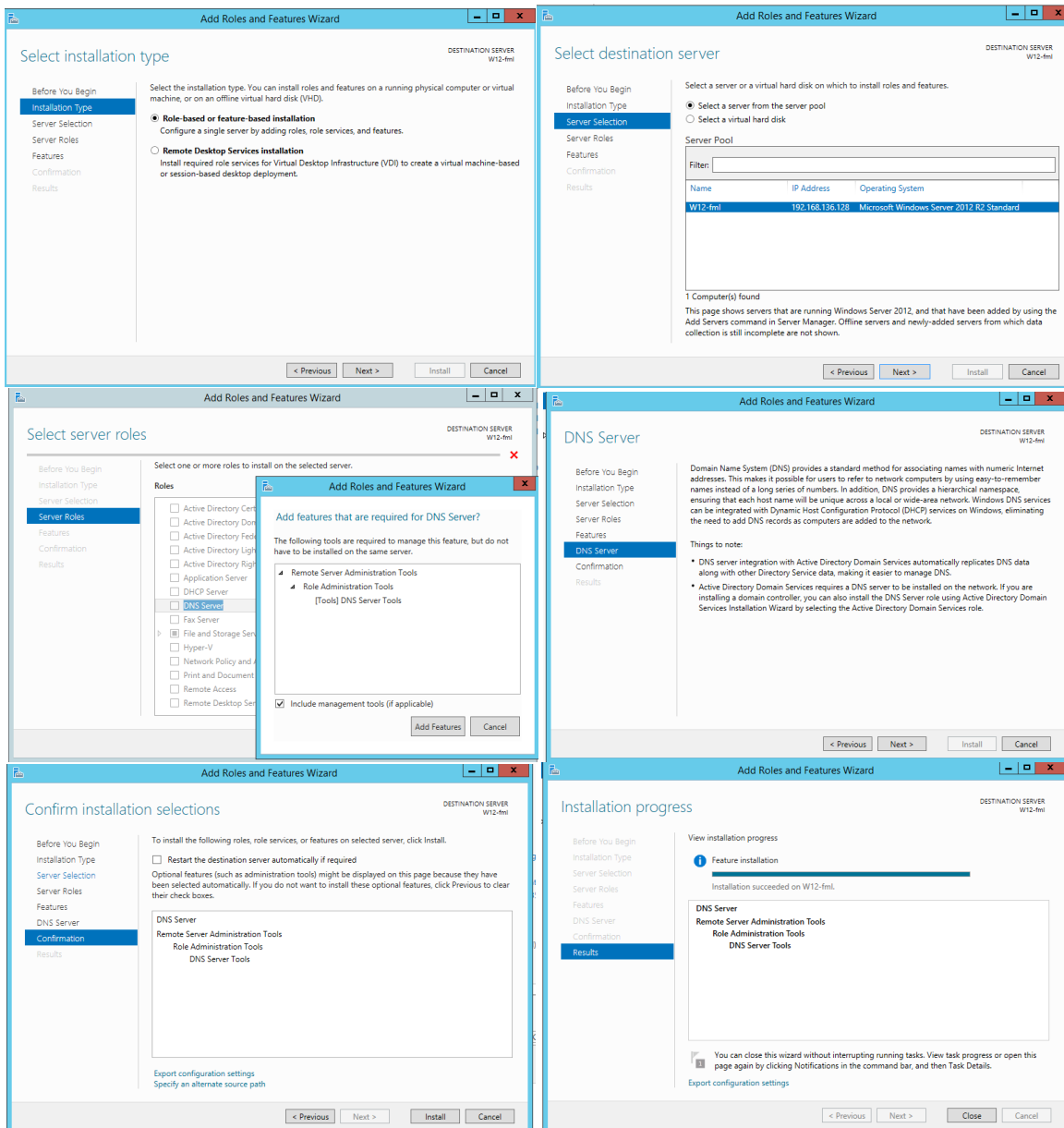
Submit a screenshot of a working nslookup for research.technowidgets.com, and nslookup for your static IP address (reverse lookup), and a working ping for www.technowidgets.com.

Activity 4-2. Install and Configure DNS in Windows

Make sure you have setup a static IP address and a computer name W12-fml (first, middle, last). Then go to Server Manager and add the DNS Role.

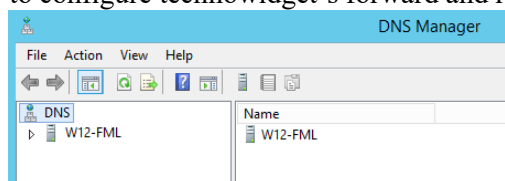
- 1) Start Server Manager and choose: Manage > Add Roles and Features



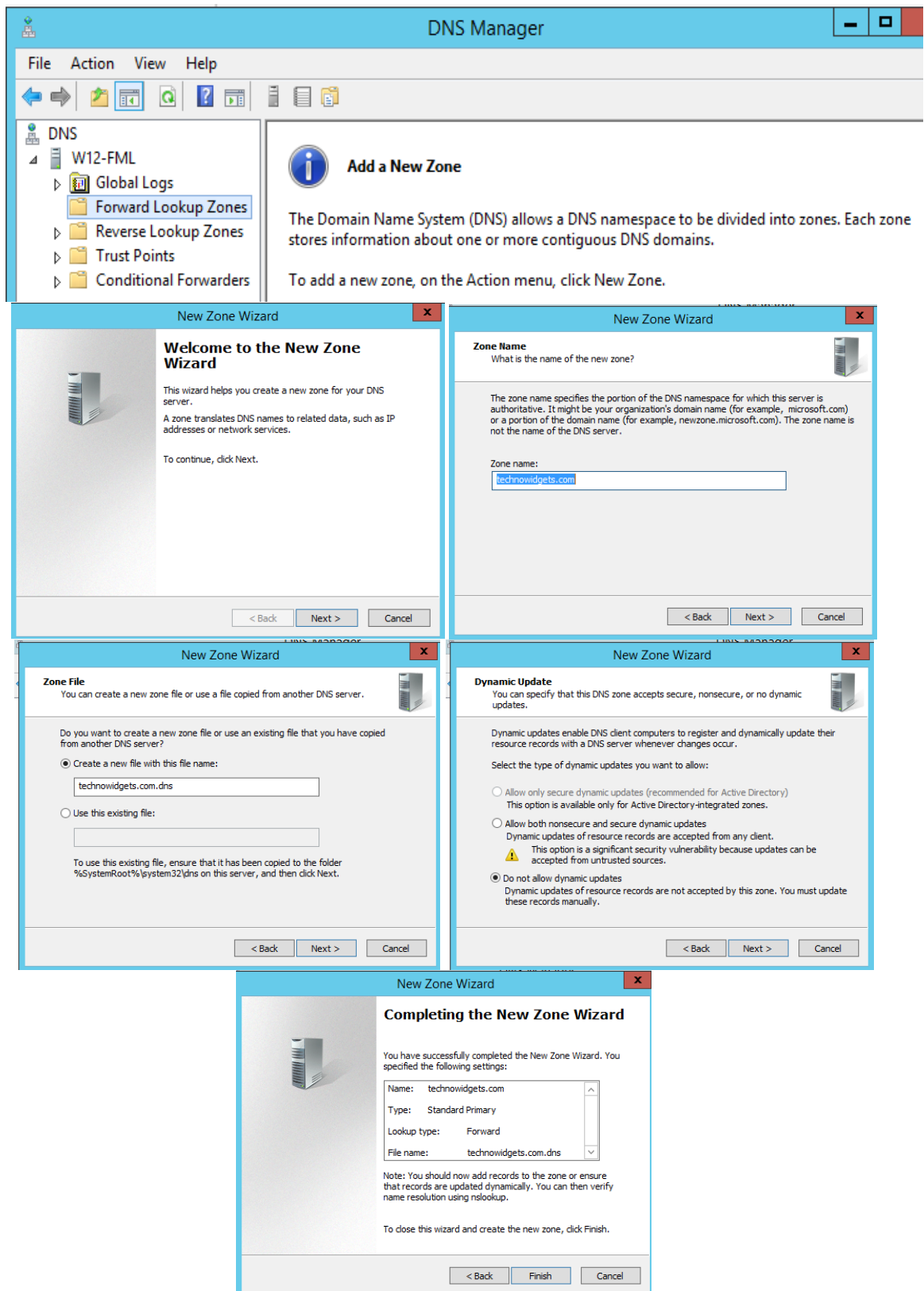


DNS Installation

2) Next go to **Tools > DNS** to configure technowidget's forward and reverse lookup zones

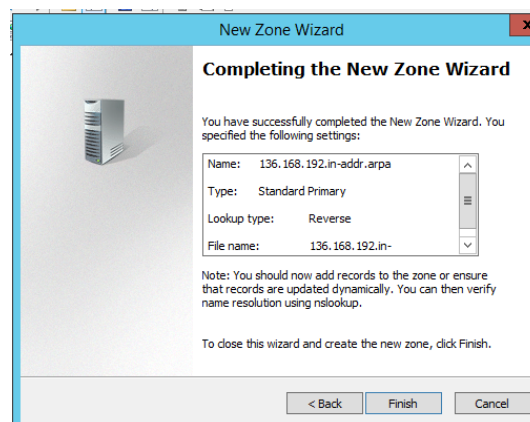
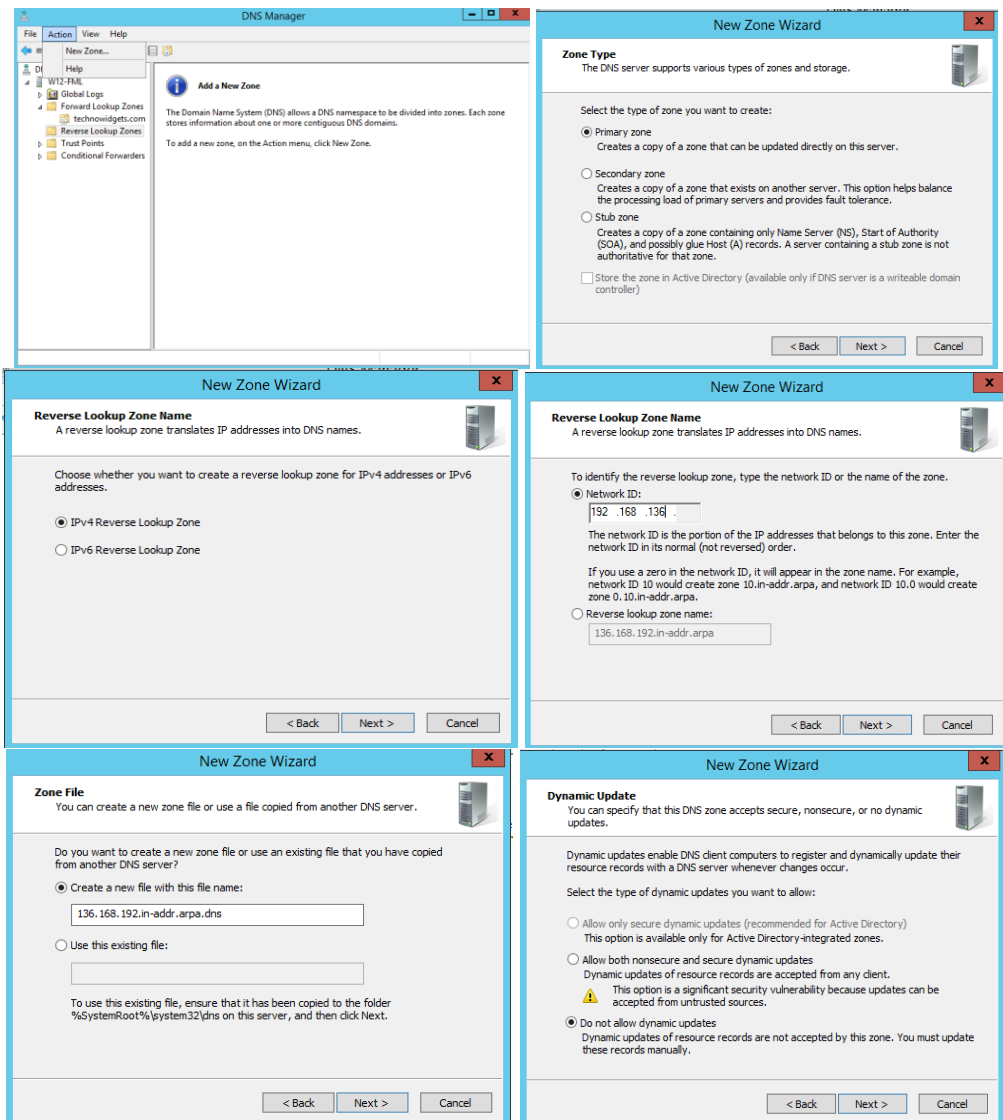


- a. Expand your server, right-click **Forward Lookup Zones** (or use the Action menu) and select **New Zone**.



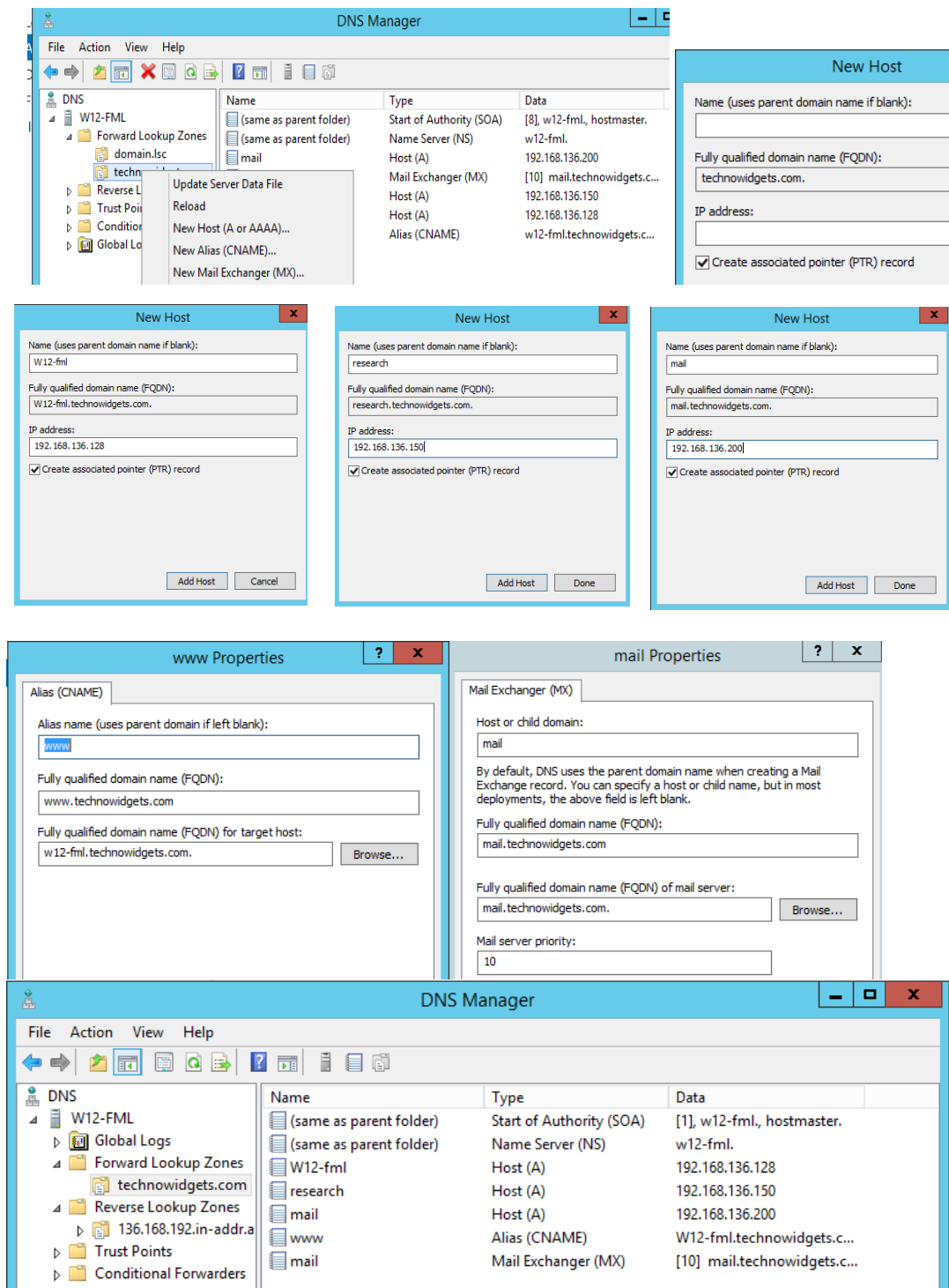
Forward Lookup Zone

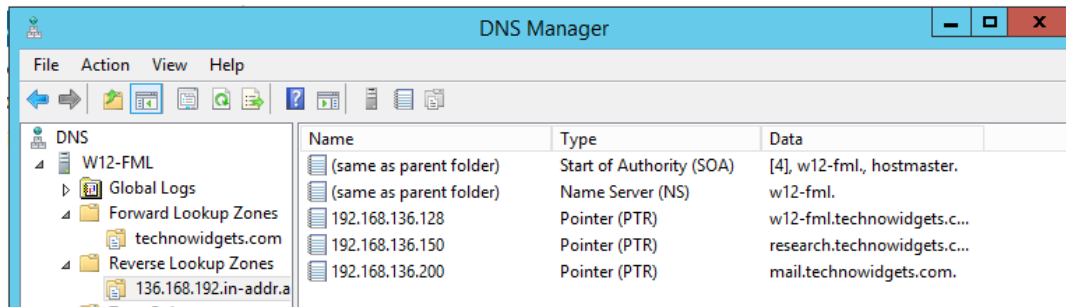
- b. Right-click **Reverse Lookup Zones** (or use the Action menu) and select **New Zone**.



Reverse Lookup Zone

- 3) Right-click your technowidgets.com forward lookup zone (or use the Action menu) add A, CNAME and MX records. Choose 'Create associated pointer (PTR) record' when creating each A record.



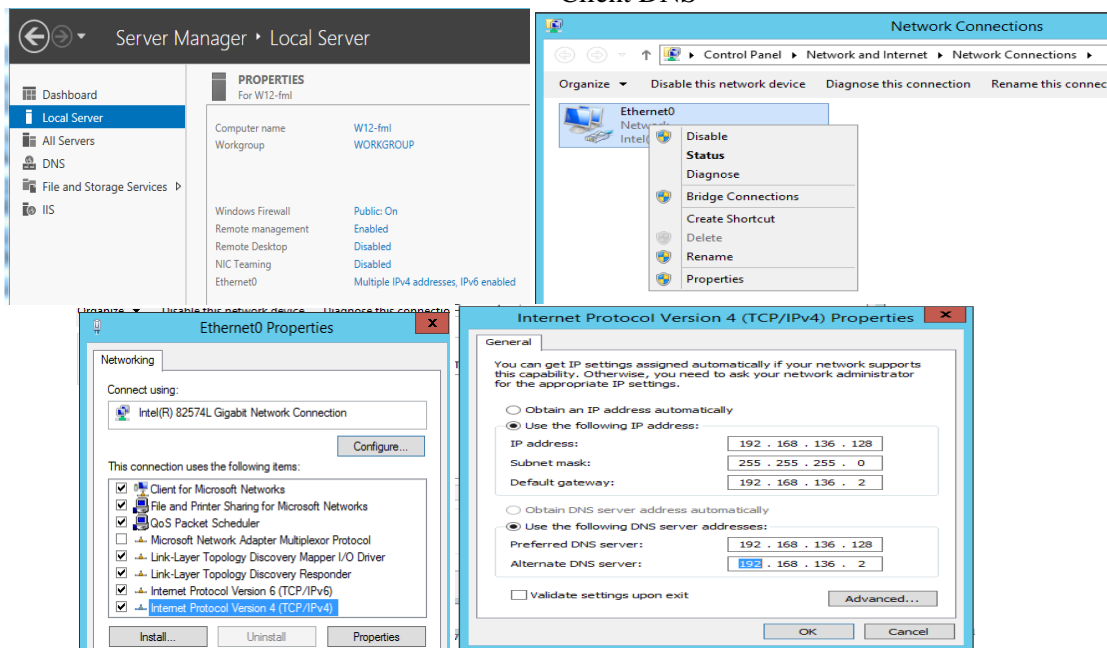


Technowidgets Forward & Reverse Lookup Zones in Windows

Submit a screenshot of a your technowidgets.com forward and reverse lookup zones in Windows.

- 4) Configure your client to use your DNS server. In Windows, this is done from the IPv4 properties of your Static IP address. From Server Manager choose your Ethernet connection, right-click Properties, right-click TCP/IPv4 Properties. Set one of your DNS server addresses to your Static IP address. The alternate DNS server can be Google's public DNS 8.8.8.8 or your gateway address x.x.x.2.

Client DNS



Submit a screenshot of your TCP/IPv4 properties

- 5) Verify that client DNS is configured (your IPv4 Static IP has your server's IP address setup for DNS resolution) and test your technowidgets setup with pings and nslookups.

Submit a screenshot of a working nslookup for research.technowidgets.com, and nslookup for your static IP address (reverse lookup), and a working ping for www.technowidgets.com.

Troubleshooting Techniques for DNS

If ping and nslookup do not give you the correct results, you need to track down the errors and fix them yourself. For example, make sure that you started the DNS server (In Ubuntu, `sudo service bind9 restart` or in Windows go to the DNS service and request a restart).

- If restart fails, you probably have a syntax error in the configuration file or zone files: (In Linux the `named.conf.local`, `named.technowidgets.com`, or `named.192.168.0`).
- If restart doesn't fail but ping or nslookup do not work
 - the DNS client configuration may be wrong (In Linux this information is in `/etc/network/interfaces`, in Windows check the IPv4 properties).
 - In Linux the filenames in `named.conf.local` may not match the actual files in `/var/lib/bind/`
 - `named-checkconf` is a useful DNS syntax checker in Linux

named-checkconf -z

```
andersml@u14-fml:/var/www/cgi-bin$ named-checkconf -z
zone technowidgets.com/IN: loaded serial 2002072100
zone 136.168.192.in-addr.arpa/IN: loaded serial 2002072100
zone domain.lsc/IN: loaded serial 2002072100
zone localhost/IN: loaded serial 2
zone 127.in-addr.arpa/IN: loaded serial 1
zone 0.in-addr.arpa/IN: loaded serial 1
zone 255.in-addr.arpa/IN: loaded serial 1
```

A common problem is mistakenly entered combinations of IP addresses. If you used different IP addresses, make sure that you entered them consistently throughout the configurations. The most common error is an FQDN without a period at the end of the name. Conversely, if you just have a host name, make sure no period appears at the end.

CHAPTER SUMMARY

□ DNS is an application that translates names to IP addresses and IP addresses to names. These names exist in a hierarchical structure. At the top of the structure is the root-level domain, which is where searches begin. The next level is the top level domain, which has names such as `.com`, `.org`, `.uk`, and `.name` that are controlled by ICANN. Second-level domain names include *microsoft.com* and *redhat.com*. Often these domain names can be registered. Thus, if one is not already being used, you can pay a fee and use it yourself.

□ Servers come in many forms. Root servers know the location of the DNS servers that take care of the top-level domains. Servers in the top-level domains know the location of the servers in the second level, which are typically DNS servers controlled by an organization. At the level of these DNS servers are primary servers, secondary servers, caching-only servers, and forwarding servers. The primary server defines the hosts for a domain. The secondary server provides backup for the primary server. The caching-only server caches IP addresses of hosts requested by users. The forwarding server passes requests to another DNS server when it cannot answer the request itself.

- To configure DNS, you must configure a forward lookup zone and a reverse lookup zone. The forward lookup zone translates host names to IP addresses. The reverse lookup zone translates IP addresses to host names.
- To configure DNS files in Linux, you modify three text files. The `/etc/bind/named.conf.local` file contains the names of the local zones and the names of the zone files. The other two files are the zone files for forward and reverse lookups.
- You configure DNS files in Windows using a GUI. Windows provides more guidance during DNS configuration than Linux does, but the concepts underlying both processes are exactly the same.
- Troubleshooting DNS problems involves using utilities such as `ping` and `nslookup`. `ping` tests basic connectivity. `nslookup` interrogates the DNS Tables and provides more details about DNS records.
- The most common error is an FQDN that does not end with a period.
- The Internet uses DNS for name resolution. Windows Server uses Dynamic DNS for name resolution.

Hands On Projects

- 4.1 Decide on the domain name you want to use for the web site you're building in this course. We'll use the fictitious TLD `.lsc`. If you choose the name `MyDomain`, your domain name would be `MyDomain.lsc`. Set up a forward lookup zone for `www.MyDomain.lsc` (use the name of your choice) in Ubuntu.

Submit a screenshot of your forward lookup zone for your new domain in Ubuntu.

Submit a screenshot of a working ping and nslookups for your new domain in Ubuntu.

- 4.2 Set up a forward lookup zone for `www.MyDomain.lsc` (use the name of your choice) in Windows.

Submit a screenshot of your forward lookup zone for your new domain in Windows.

Submit a screenshot of a working ping and nslookups for your new domain in Windows.

- Go to **D2L Discussions** to respond to Ch 4 Linux vs Windows, and **Quizzes** to complete the Ch 4 Review Questions. You have three attempts at the review questions and your score is the average of all three. Submit your completed worksheet to the Chapter 4 **Assignments** folder.

- DNS Made Easy

<https://www.youtube.com/watch?v=72snZctFFtA>