C H A P T E R 5

Introduction to DNS

5

Domain Name System (DNS) enables you to use hierarchical, friendly names to easily locate computers and other resources on an IP network. The following sections describe the basic DNS concepts, including features explained in newer Requests for Comments (RFCs), such as dynamic update, from the Internet Engineering Task Force (IETF). The Microsoft® Windows® 2000–specific implementation of DNS is not covered within this chapter, except where indicated.

For information about the Windows 2000 implementation of DNS, see "Windows 2000 DNS" in this book.

DNS is a distributed database that contains mappings of DNS domain names to data. It is also a protocol for Transmission Control Protocol/Internet Protocol (TCP/IP) networks, defined by the Requests for Comments (RFCs) that pertain to DNS. DNS defines the following:

- Mechanism for querying and updating the database.
- Mechanism for replicating the information in the database among servers.
- Schema for the database.

In This Chapter

Related Information in the Resource Kit

- For more information about TCP/IP protocols, see "Introduction to TCP/IP" in this book.
- For information about the Windows 2000 implementation of DNS, see "Windows 2000 DNS" in this book.

Introduction to the Domain Name System

Although TCP/IP uses IP addresses to locate and connect to hosts (computers and other TCP/IP network devices), users typically prefer to use friendly names. For example, users prefer the friendly name ftp.reskit.com, instead of its IP address, 172.16.23.55. The Domain Name System (DNS), defined in RFCs 1034 and 1035, is used on the Internet to provide a standard naming convention for locating IP-based computers.

On the Internet, before the implementation of DNS, the use of names to locate resources on TCP/IP networks was supported by a file called Hosts. Network administrators entered names and IP addresses into Hosts, and computers used the file for name resolution.

Both the Hosts file and DNS use a namespace. A *namespace* is a grouping in which names can be used to symbolically represent another type of information, such as an IP address, and in which specific rules are established that determine how names can be created and used. Some namespaces, such as DNS, are hierarchically structured and provide rules that allow for the namespace to be divided into subsets of names for distributing and delegating parts of the namespace. Other namespaces, such as the Hosts namespace cannot be divided and must be distributed in their entirety. Because of this, using the Hosts file posed a problem for network administrators. As the number of computers and users on the Internet grew, the task of updating and distributing the Hosts file became unmanageable.

DNS replaces the Hosts file with a distributed database that implements a hierarchical naming system. This naming system allows for growth on the Internet and the creation of names that are unique throughout the Internet and private TCP/IP-based intranets.

Domain Namespace

The naming system on which DNS is based is a hierarchical and logical tree structure called the *domain namespace*. Organizations can also create private networks that are not visible on the Internet, using their own domain namespaces. Figure 5.1 shows part of the Internet domain namespace, from the root domain and top-level Internet DNS domains, to the fictional DNS domain named reskit.com that contains a host (computer) named Mfgserver.

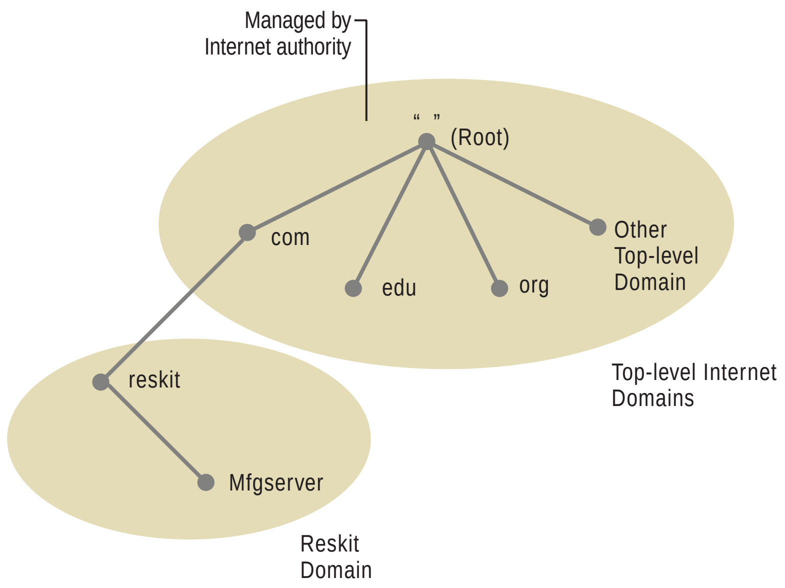

Figure 5.1 Domain Name System

Each node in the DNS tree represents a DNS name. Some examples of DNS names are DNS domains, computers, and services. A DNS domain is a branch under the node. For example, in Figure 5.1, reskit.com is a DNS domain. DNS domains can contain both hosts (computers or services) and other domains (referred to as *subdomains*). Each organization is assigned authority for a portion of the domain namespace and is responsible for administering, subdividing, and naming the DNS domains and computers within that portion of the namespace.

Subdividing is an important concept in DNS. Creating subdivisions of the domain namespace and private TCP/IP network DNS domains supports new growth on the Internet and the ability to continually expand name and administrative groupings. Subdivisions are generally based on departmental or geographic divisions.

For example, the reskit.com DNS domain might include sites in North America and Europe. A DNS administrator of the DNS domain reskit.com can subdivide the domain to create two subdomains that reflect these groupings: noam.reskit.com. and eu.reskit.com. Figure 5.2 shows an example of these subdomains.

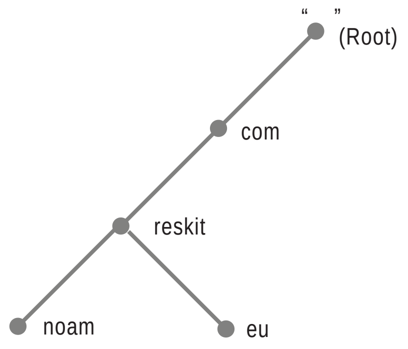

Figure 5.2 Subdomains

Domain Name

Computers and DNS domains are named based on their position in the domain tree. For example, because reskit is a subdomain of the .com domain, the domain name for reskit is reskit.com.

Every node in the DNS domain tree can be identified by a *fully qualified domain name* (FQDN). The FQDN is a DNS domain name that has been stated unambiguously so as to indicate with absolute certainty its location relative to the root of the DNS domain tree. This contrasts with a relative name, which is a name relative to some DNS domain other than the root.

For example, the FQDN for the server in the reskit.com DNS domain is constructed as Mfgserver.reskit.com., which is the concatenation of the host name (Mfgserver) with the primary DNS suffix (reskit.com), and the trailing dot (.). The trailing dot is a standard separator between the top-level domain label and the empty string label corresponding to the root.

Note In general, FQDNs have naming restrictions that allow only the use of characters a-z, A-Z, 0-9, and the dash or minus sign (-). The use of the period (.) is allowed only between domain name labels (for example, “reskit.com”) or at the end of a FQDN. Domain names are not case-sensitive.

You can configure the Windows 2000 DNS server to enforce some or all RFC character restrictions or to ignore all character restrictions. For more information, see “Windows 2000 DNS” in this book.

Internet Domain Namespace

The root (the top-most level) of the Internet domain namespace is managed by an Internet name registration authority, which delegates administrative responsibility for portions of the domain namespace to organizations that connect to the Internet.

Beneath the root DNS domain lie the top-level domains, also managed by the Internet name registration authority. There are three types of top-level domains:

- Organizational domains. These are named by using a 3-character code that indicates the primary function or activity of the organizations contained within the DNS domain. Organizational domains are generally only for organizations within the United States, and most organizations located in the United States are contained within one of these organizational domains.

- Geographical domains. These are named by using the 2-character country/region codes established by the International Standards Organization (ISO) 3166.
- Reverse domains. This is a special domain, named in-addr.arpa, that is used for IP address-to-name mappings (referred to as *reverse lookup*). For more information, see "Name Resolution" later in this chapter. There is also a special domain, named IP6.INT, used for IP version 6 reverse lookups. For information, see RFC 1886.

The most commonly used top-level DNS name components for organizations in the United States are described in the Table 5.1.

Table 5.1 Top-Level Name Component of the DNS Hierarchy

Top-Level Name Component	Description	Example DNS Domain Name
.com	An Internet name authority delegates portions of the domain namespace under this level to commercial organizations, such as the Microsoft Corporation.	microsoft.com
.edu	An Internet name authority delegates portions of this domain namespace to educational organizations, such as the Massachusetts Institute of Technology (MIT).	mit.edu
.gov	An Internet name authority delegates portions of this domain namespace to governmental organizations, such as the White House in Washington, D.C.	whitehouse.gov
.int	An Internet name authority delegates portions of this domain namespace to international organizations, such as the North Atlantic Treaty Organization (NATO).	nato.int
.mil	An Internet name authority delegates portions of this domain namespace to military operations, such as the Defense Date Network (DDN).	ddn.mil
.net	An Internet name authority delegates portions of this domain namespace to networking organizations, such as the National Science Foundation (NSF).	nsf.net
.org	An Internet name authority delegates portions of this domain namespace to noncommercial organizations, such as the Center for Networked Information	cnidr.org

Discovery and Retrieval (CNIDR).

In addition to the top-level domains listed above, individual countries have their own top-level domains. For example, .ca is the top-level domain for Canada.

Beneath the top-level domains, an Internet name authority delegates domains to organizations that connect to the Internet. The organizations to which an Internet name authority delegates a portion of the domain namespace are then responsible for naming the computers and network devices within their assigned domain and its subdivisions. These organizations use DNS servers to manage the name-to-IP address and IP address-to-name mappings for host devices contained within their portion of the namespace.

Basic DNS Concepts

This section provides brief definitions of additional DNS concepts, which are described in more detail in the following sections of this chapter.

DNS servers. Computers that run DNS server programs containing DNS database information about the DNS domain tree structure. DNS servers also attempt to resolve client queries. When queried, DNS servers can provide the requested information, provide a pointer to another server that can help resolve the query, or respond that it does not have the information or that the information does not exist.

DNS resolvers. Programs that use DNS queries to query for information from servers. Resolvers can communicate with either remote DNS servers or the DNS server program running on the local computer. Resolvers are usually built into utility programs or are accessible through library functions. A resolver can run on any computer, including a DNS server.

Resource records. Sets of information in the DNS database that can be used to process client queries. Each DNS server contains the resource records it needs to answer queries for the portion of the DNS namespace for which it is authoritative. (A DNS server is authoritative for a contiguous portion of the DNS namespace if it contains information about that portion of the namespace.)

Zones. Contiguous portions of the DNS namespace for which the server is authoritative. A server can be authoritative for one or more zones.

Zone files. Files that contain resource records for the zones for which the server is authoritative. In most DNS implementations, zones are implemented as text files.

Zones

A zone is a contiguous portion of the DNS namespace. It contains a series of records stored on a DNS server. Each zone is anchored at a specific domain node. However, zones are not domains. A *DNS domain* is a branch of the namespace, whereas a zone is a portion of the DNS namespace generally stored in a file, and can contain multiple domains. A domain can be subdivided into several partitions, and each partition, or zone, can be controlled by a separate DNS server. Using the zone, the DNS server answers queries about hosts in its zone, and is authoritative for that zone. Zones can be primary or secondary. A *primary zone* is the copy of the zone to which the updates are made, whereas a *secondary zone* is a copy of the zone that is replicated from a master server.

Zones can be stored in different ways. For example, they can be stored as zone files. On Windows 2000 servers, they can also be stored in the Active Directory™ directory service. Some secondary servers store them in memory and perform a zone transfer whenever they are restarted.

Figure 5.3 shows an example of a DNS domain that contains two primary zones. In this example, the domain reskit.com contains two subdomains: noam.reskit.com. and eu.reskit.com. Authority for the noam.reskit.com. subdomain has been delegated to the server noamdc1.noam.reskit.com. Thus, as Figure 5.3 shows, one server, noamdc1.noam.reskit.com, hosts the noam.reskit.com zone, and a second server, reskitdc1.reskit.com, hosts the reskit.com zone that includes the eu.reskit.com subdomain.

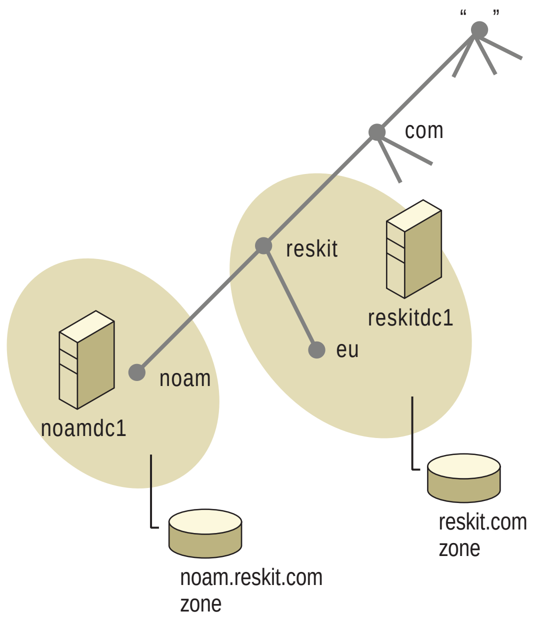

Figure 5.3 Domains and Zones

Rather than delegating the noam.reskit.com zone to noamdc1.noam.reskit.com, the administrator can also configure reskitdc1 to host the zone for noam.reskit.com. Also, you cannot configure two different servers to manage the same primary zones; only one server can manage the primary zone for each DNS domain. There is one exception: multiple computers can manage Windows 2000 Active Directory–integrated zones. For more information, see "Windows 2000 DNS" in this book.

You can configure a single DNS server to manage one zone or multiple zones, depending on your needs. You can create multiple zones to distribute administrative tasks to different groups and to provide efficient data distribution. You can also store the same zone on multiple servers to provide load balancing and fault tolerance.

For information about what zones contain, see “Resource Records and Zones” later in this chapter.

DNS Servers

DNS servers store information about no zones, one zone, or multiple zones. When a DNS server receives a DNS query, it attempts to locate the requested information by retrieving data from its local zones. If this fails because the server is not authoritative for the DNS domain requested and thus does not have the data for the requested domain, the server can check its cache, communicate with other DNS servers to resolve the request, or refer the client to another DNS server that might know the answer.

DNS servers can host primary and secondary zones. You can configure servers to host as many different primary or secondary zones as is practical, which means that a server might host the primary copy of one zone and the secondary copy of another zone, or it might host only the primary or only the secondary copy for a zone. For each zone, the server that hosts the primary zones is considered the *primary server* for that zone, and the server that hosts the secondary zones is considered the *secondary server* for that zone.

Primary zones are locally updated. When a change is made to the zone data, such as delegating a portion of the zone to another DNS server or adding resource records in the zone, these changes must be made on the primary DNS server for that zone, so that the new information can be entered in the local zone.

In contrast, secondary zones are replicated from another server. When a zone is defined on a secondary server for that zone, the zone is configured with the IP address of the server from which the zone is to be replicated. The server from which the zone file replicates can either be a primary or secondary server for the zone, and is sometimes called a *master server* for the secondary zone.

When a secondary server for the zone starts up, it contacts the master server for the zone and initiates a zone transfer. The secondary server for the zone also periodically contacts the master server for the zone to see whether the zone data has changed. If so, it can initiate a transfer of the zones, referred to as a *zone transfer*. For more information about zone transfers, see “Zone Transfer” later in this chapter.

You must have a primary server for each zone. Additionally, you should have at least one secondary server for each zone. Otherwise, if the primary server for the zone goes down, no one will be able to resolve the names in that zone.

Secondary servers provide the following benefits:

Fault tolerance When a secondary server is configured for a zone, clients can still resolve names for that zone even if the primary server for the zone goes down. Generally, plan to install the primary and secondary servers for the zone on different subnets. Therefore, if connectivity to one subnet is lost, DNS clients can still direct queries to the name server on the other subnet.

Reduction of traffic on wide area links You can add a secondary server for the zone in a remote location that has a large number of clients, and then configure the client to try those servers first. This can prevent clients from communicating across slow links for DNS queries.

Reduction of load on the primary server for the zone The secondary server can answer queries for the zone, reducing the number of queries the primary server for the zone must answer.

The following sections describe servers that act as caching-only servers, forwarders, and slaves.

Caching-Only Servers

All DNS servers perform *caching*; whenever they receive information from other servers, they store the information for a certain amount of time. This speeds the performance of DNS resolution, reduces DNS-related query traffic, and improves reliability. For more information, see "Caching and Time to Live" later in this chapter.

Certain DNS servers, known as *caching-only servers,* simply perform queries, cache the answers, and return the results. They are not authoritative for any DNS domains and do not host any zones. They only store data that they have cached while resolving queries.

The benefit provided by caching-only servers is that they do not generate zone transfer network traffic because they do not contain any zones. However, there is one disadvantage: when the server is initially started, it has no cached information and must build up this information over time as it services requests.

Forwarders and Slaves

When a DNS server receives a query, it attempts to locate the requested information within its local zones and from the cache. If it cannot locate the requested information and is not authoritative for the requested information, it must communicate with other servers to resolve the request. However, in some cases network administrators might not want the server to communicate directly with other servers. For example, if your organization were connected to the Internet by means of a slow wide area link, you might not want every DNS server in your organization to connect directly to DNS servers on the Internet.

To solve this problem, DNS allows for the use of *forwarders*. Forwarders are DNS servers that are designated to provide forwarding of off-site queries for other DNS servers. For example, you could designate one DNS server as a forwarder for names of computers on the Internet, and then configure your other servers to use that forwarder to resolve names for which they are not authoritative.

You do not need to perform any special configuration on the computer designated as a forwarder. You must configure the DNS server that needs to forward queries by providing the IP address of the forwarders.

A server can use a forwarder in a nonexclusive or exclusive mode. In a nonexclusive mode, when a server receives a DNS query for which it is not authoritative and cannot resolve through its own zones or cache, it passes the query to one of the designated forwarders. The forwarder then carries out whatever communication is necessary to resolve the query and returns the results to the requesting server, which returns the results to the original requester. If the forwarder cannot resolve the query, the server that received the original query attempts to resolve the query on its own.

In an exclusive mode, servers rely completely on the name-resolving ability of the forwarders. Servers using forwarders in an exclusive mode are known as *slaves*.

When a slave receives a DNS query that it cannot resolve through its own zones, it passes the query to one of the designated forwarders. The forwarder then carries out whatever communication is necessary to resolve the query and returns the results to the slave, which returns the results to the original requester. If the forwarder cannot resolve the request, the slave returns a query failure to the original requestor. Slaves make no attempt to resolve the query on their own if the forwarder cannot satisfy the request.

Load Sharing

DNS servers use a mechanism called round-robin or *load sharing*, explained in RFC 1794, to share and distribute loads for network resources. Round-robin rotates the order of resource record data returned in a query answer in which multiple RRs exist of the same RR type for a queried DNS domain name.

For example, suppose you have three World Wide Web servers with the same domain name, WWWServer, that all display the Web page for www.reskit.com, and you want to share the load between them. On the name server, you would create the following resource records:

```
www.reskit.com.      IN  A       172.16.64.11
www.reskit.com.      IN  A       172.17.64.22
www.reskit.com.      IN  A       172.18.64.33
```

A name server configured to perform round-robin rotates the order of the A resource records when answering client requests. In this example, the name server would reply to the first client request by ordering the addresses as 172.16.64.11, 172.17.64.22, and 172.18.64.33. It would reply to the second client response by ordering the addresses as 172.17.64.22, 172.18.64.33, and 172.16.64.11. The rotation process continues until data from all of the same type of resource records associated with a name have been rotated to the top of the list returned in answering client queries. The client is required to try the first IP address listed.

By default, a Windows 2000 DNS server uses a different method to order the records returned to a client. It attempts to find the resource record containing the IP address closest to the client, then returns this resource record first in the list of records. However, you can modify the default so it performs traditional round-robin. For information, see “Windows 2000 DNS” in this book. Versions of BIND 4.9.3 and later perform this type of load sharing. Earlier versions of BIND performed a different type of load sharing; for more information, see RFC 1794.

Name Resolution

DNS clients use libraries called *resolvers* that perform DNS queries to servers on behalf of the client. Keep in mind throughout this discussion that a DNS server can also be a client to another server.

Note Computers running under Microsoft® Windows NT® Workstation or Microsoft® Windows NT®Server version 4.0 use DNS name resolution when a name query contains a name that contains a period or is greater than 15 bytes in length. Computers running Windows 2000 always try DNS name resolution. For more information about DNS and NetBIOS name resolution, see “TCP/IP Troubleshooting” and “Windows 2000 DNS” in this book.

DNS clients can make two types of queries: recursive and iterative.

Recursive and Iterative Queries

With a *recursive name query*, the DNS client requires that the DNS server respond to the client with either the requested resource record or an error message stating that the record or domain name does not exist. The DNS server cannot just refer the DNS client to a different DNS server.

Thus, if a DNS server does not have the requested information when it receives a recursive query, it queries other servers until it gets the information, or until the name query fails.

Recursive name queries are generally made by a DNS client to a DNS server, or by a DNS server that is configured to pass unresolved name queries to another DNS server, in the case of a DNS server configured to use a forwarder.

An *iterative name query* is one in which a DNS client allows the DNS server to return the best answer it can give based on its cache or zone data. If the queried DNS server does not have an exact match for the queried name, the best possible information it can return is a *referral* (that is, a pointer to a DNS server authoritative for a lower level of the domain namespace). The DNS client can then query the DNS server for which it obtained a referral. It continues this process until it locates a DNS server that is authoritative for the queried name, or until an error or time-out condition is met. This process is sometimes referred to as “walking the tree,” and this type of query is typically initiated by a DNS server that attempts to resolve a recursive name query for a DNS client.

Figure 5.4 shows an example of iterative and recursive queries. This example assumes that none of the servers have the requested information in their caches.

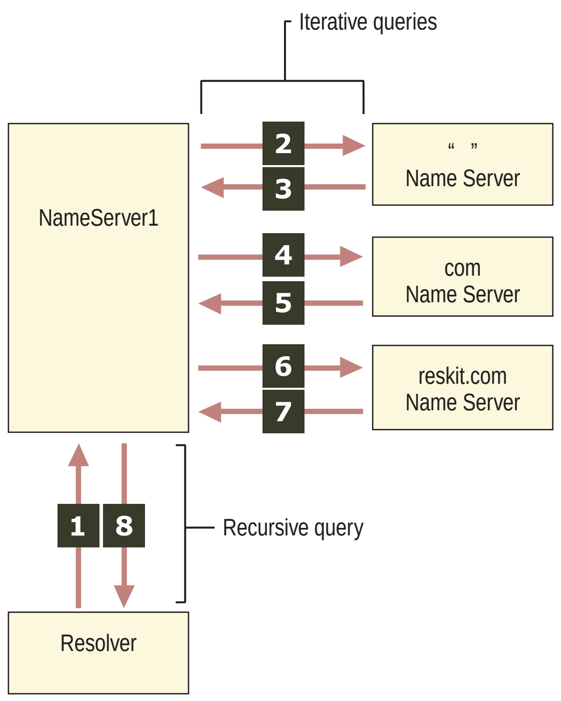

Figure 5.4 Iterative and Recursive Queries

In the example shown in Figure 5.4, a client somewhere on the Internet needs the IP address of noam.reskit.com. The following events take place:

1. The client contacts NameServer1 with a recursive query for noam.reskit.com. The server must now return either the answer or an error message.
2. NameServer1 checks its cache and zones for the answer, but does not find it, so it contacts a server authoritative for the Internet (that is, a *root server*) with an iterative query for noam.reskit.com.
3. The server at the root of the Internet does not know the answer, so it responds with a referral to a server authoritative for the .com domain.
4. NameServer1 contacts a server authoritative for the .com domain with an iterative query for noam.reskit.com.
5. The server authoritative for the .com domain does not know the exact answer, so it responds with a referral to a server authoritative for the reskit.com domain.
6. NameServer1 contacts the server authoritative for the reskit.com domain with an iterative query for noam.reskit.com.
7. The server authoritative for the reskit.com domain does know the answer. It responds with the requested IP address.

8. NameServer1 responds to the client query with the IP address for noam.reskit.com.

Caching and Time to Live

When a server is processing a recursive query, it might be required to send out several queries to find the definitive answer. The server caches all of the information that it receives during this process for a time that is specified in the returned data. This amount of time is referred to as the *time to live (TTL)* and is specified in seconds. The server administrator of the primary zone that contains the data decides on the TTL for the data. Smaller TTL values help ensure that information about the domain is more consistent across the network, in the event that this data changes often. However, this also increases the load on the name servers that contain the name, and it also increases Internet traffic. Because data is cached, changes made in resource records might not be immediately available to the entire Internet.

Once data is cached by a DNS server, the DNS server must start decreasing the TTL from its original value so that it will know when to flush the data from its cache. When the DNS server answers a query with its cached data, it includes the remaining TTL for the data. The resolver can then cache this data, using the TTL sent by the server.

Negative Caching

In addition to caching resolved queries, resolvers and servers can also cache negative responses, that is, the information that a specific resource record set (RRset) or DNS domain name does not exist. Negative caching can reduce the response time for negative answers. It can also reduce network traffic by reducing the number of messages that must be sent between resolvers and name servers or between name servers.

Negative caching is specified in RFCs 1034 and 2308. RFC 1034 describes how to cache negative responses and makes negative caching optional. RFC 2308 requires resolvers to cache negative responses if they cache any responses. It also describes a way for name servers to forward cached negative responses to resolvers. Just as with ordinary caching, they must also start decreasing the TTL.

For more information about negative caching with Windows 2000 DNS, see “Windows 2000 DNS” in this book.

Resource Records and Zones

To resolve names, servers consult their zones (also called DNS database files or simply db files). The zones contain resource records (RRs) that make up the resource information associated with the DNS domain. For example, some resource records map friendly names to IP addresses, and others map IP addresses to friendly names. Certain resource records not only include information about servers in the DNS domain, but also serve to define the domain by specifying which servers are authoritative for which zones. These resource records, the SOA and NS resource records, are described in more detail later in this section.

Resource Record Format

Resource records have the following syntax:

```
Owner   TTL     Class   Type    RDATA
```

Table 5.2 describes each of these fields.

Table 5.2 Typical Resource Record Fields

Name	Description
Owner	The name of the host or the DNS domain to which this resource record belongs.
Time to Live	A 32-bit integer that represents, in seconds, the length of time that a DNS server or resolver should cache this entry before it is discarded. This field is optional, and if it is not specified, the client uses the Minimum TTL in the SOA record.
Class	Defines the protocol family in use. It is almost always IN for the Internet system. The other value defined in RFC 1034 is CH for the Chaos system, which was used experimentally at the Massachusetts Institute of Technology.
Type	Identifies the type of resource record.
RDATA	The resource record data. It is a variable type that represents the information being described by the type. For example, in an A record, this is the 32-bit IP address that represents the host defined by the resource record.

Resource records are represented in binary form in packets when lookups and responses are made using DNS. In the database files, however, resource records are represented as text entries. Most resource records are represented as single-line text entries. If an entry is going to span more than one line, you can use parentheses to encapsulate the information. In many implementations of DNS, only the Start of Authority (SOA) record can be multiple lines. For readability, blank lines and comments are often inserted in the zone files, and are ignored by the DNS server. Comments always start with a semicolon (;) and end with a carriage return.

Resource Record Types

Different types of resource records can be used to provide DNS-based data about computers on a TCP/IP network. This section describes the following resource records:

- SOA
- NS
- A
- PTR
- CNAME

- MX
- SRV

Next, it lists some of the other resource records specified by RFC standards. Finally, it lists resource records that are specific to the Windows 2000 implementation and one resource record specified by the ATM Forum.

SOA Resource Records

Every zone contains a Start of Authority (SOA) resource record at the beginning of the zone. SOA resource records include the following fields:

- The **Owner**, **TTL**, **Class**, and **Type** fields, as described in “Resource Record Format” earlier in this chapter.
- The **authoritative server** field shows the primary DNS server authoritative for the zone.
- The **responsible person** field shows the e-mail address of the administrator responsible for the zone. It uses a period (.) instead of an at symbol (@).
- The **serial number** field shows how many times the zone has been updated. When a zone’s secondary server contacts the master server for that zone to determine whether it needs to initiate a zone transfer, the zone’s secondary server compares its own serial number with that of the master. If the serial number of the master is higher, the secondary server initiates a zone transfer.
- The **refresh** field shows how often the secondary server for the zone checks to see whether the zone has been changed.
- The **retry** field shows how long after sending a zone transfer request the secondary server for the zone waits for a response from the master server before retrying.
- The **expire** field shows how long after the previous zone transfer the secondary server for the zone continues to respond to queries for the zone before discarding its own zone as invalid.
- The **minimum TTL** field applies to all the resource records in the zone whenever a time to live value is not specified in a resource record. Whenever a resolver queries the server, the server sends back resource records along with the minimum time to live. Negative responses are cached for the minimum TTL of the SOA resource record of the authoritative zone.

The following example shows the SOA resource record:

```
noam.reskit.com.  IN  SOA (
                noamdc1.noam.reskit.com.     ; authoritative  server
                                                 for the zone
                administrator.noam.reskit.com.   ; zone admin e-mail
                                             ; (responsible person)
                5099                         ; serial number
                3600                         ; refresh (1 hour)
                600                          ; retry (10 mins)
```

```
                    86400                              ; expire (1 day)
                    60       )                         ; minimum TTL (1 min)
```

NS Resource Records

The name server (NS) resource record indicates the servers authoritative for the zone. They indicate primary and secondary servers for the zone specified in the SOA resource record, and they indicate the servers for any delegated zones. Every zone must contain at least one NS record at the zone root.

For example, when the administrator on reskit.com delegated authority for the noam.reskit.com subdomain to noamdc1.noam.reskit.com., the following line was added to the zones reskit.com and noam.reskit.com:

```
noam.reskit.com.    IN    NS      noamdc1.noam.reskit.com.
```

A Resource Records

The address (A) resource record maps an FQDN to an IP address, so the resolvers can request the corresponding IP address for an FQDN. For example, the following A resource record, located in the zone noam.reskit.com, maps the FQDN of the server to its IP address:

```
noamdc1         IN        A      172.16.48.1
```

PTR Records

The *pointer (PTR) resource record*, in contrast to the A resource record, maps an IP address to an FQDN. For example, the following PTR resource record maps the IP address of noamdc1.noam.reskit.com to its FQDN:

```
1.48.16.172.in-addr.arpa.        IN      PTR
    noamdc1.noam.reskit.com.
```

CNAME Resource Records

The canonical name (CNAME) resource record creates an alias (synonymous name) for the specified FQDN. You can use CNAME records to hide the implementation details of your network from the clients that connect to it. For example, suppose you want to put an FTP server named ftp1.noam.reskit.com on your noam.reskit.com subdomain, but you know that in six months you will move it to a computer named ftp2.noam.reskit.com, and you do not want your users to have to know about the change. You can just create an alias called ftp.noam.reskit.com that points to ftp1.noam.reskit.com, and then when you move your computer, you need only change the CNAME record to point to ftp2.noam.reskit.com. For example, the following CNAME resource record creates an alias for ftp1.noam.reskit.com:

```
ftp.noam.reskit.com.    IN      CNAME       ftp1.noam.reskit.com.
```

Once a DNS client queries for the A resource record for ftp.noam.reskit.com, the DNS server finds the CNAME resource record, resolves the query for the A resource record for ftp1.noam.reskit.com, and returns both the A and CNAME resource records to the client.

Note According to RFC 2181, there must be only one canonical name per alias.

MX Resource Records

The mail exchange (MX) resource record specifies a mail exchange server for a DNS domain name. A mail exchange server is a host that will either process or forward mail for the DNS domain name. Processing the mail means either delivering it to the addressee or passing it to a different type of mail transport. Forwarding the mail means sending it to its final destination server, sending it using Simple Mail Transfer Protocol (SMTP) to another mail exchange server that is closer to the final destination, or queuing it for a specified amount of time.

Note Only mail exchange servers use MX records.

If you want to use multiple mail exchange servers in one DNS domain, you can have multiple MX resource records for that domain. The following example shows MX resource records for the mail servers for the domain noam.reskit.com.:

```
*.noam.reskit.com.   IN      MX      0   mailserver1.noam.reskit.com.
*.noam.reskit.com.   IN      MX      10  mailserver2.noam.reskit.com.
*.noam.reskit.com.   IN      MX      10  mailserver3.noam.reskit.com.
```

The first three fields in this resource record are the standard owner, class, and type fields. The fourth field is the mail server priority*,* or preference value. The preference value specifies the preference given to the MX record among MX records. Lower priority records are preferred. Thus, when a mailer needs to send mail to a certain DNS domain, it first contacts a DNS server for that domain and retrieves all the MX records. It then contacts the mailer with the lowest preference value.
For example, suppose Jane Doe sends an e-mail message to JohnDoe@noam.reskit.com on a day that mailserver1 is down, but mailserver2 is working. Her mailer tries to deliver the message to mailserver1, because it has the lowest preference value, but it fails because mailserver1 is down. This time, Jane’s mailer can choose either mailserver2 or mailserver3, because their preference values are equal. It successfully delivers the message to mailserver2.
To prevent mail loops, if the mailer is on a host that is listed as an MX for the destination host, the mailer can deliver only to an MX with a lower preference value than its own host.

Note The **sendmail** program requires special configuration if a CNAME is not referenced in the MX record.

SRV Records

With MX records, you can have multiple mail servers in a DNS domain, and when a mailer needs to send mail to a host in the domain, it can find the location of a mail exchange server. But what about other applications, such as the World Wide Web or telnet?

Service (SRV) resource records enable you to specify the location of the servers for a specific service, protocol, and DNS domain. Thus, if you have two Web servers in your domain, you can create SRV resource records specifying which hosts serve as Web servers, and resolvers can then retrieve all the SRV resource records for the Web servers.

The format of an SRV record is as follows:

_Service._Proto.NameTTL Class SRV Priority Weight Port Target

- The **_Service** field specifies the name of the service, such as http or telnet. Some services are defined in the standards, and others can be defined locally.
- The **_Proto** field specifies the protocol, such as TCP or UDP.
- The **Name** field specifies the domain name to which the resource record refers.
- The **TTL** and **Class** fields are the same as the fields defined earlier in this chapter.
- The **Priority** field specifies the priority of the host. Clients attempt to contact the host with the lowest priority.
- The **Weight** field is a load balancing mechanism. When the priority field is the same for two or more records in the same domain, clients should try records with higher weights more often, unless the clients support some other load balancing mechanism.
- The **Port** field shows the port of the service on this host.
- The **Target** field shows the fully qualified domain name for the host supporting the service.

The following example shows SRV records for Web servers:

```
_http._tcp.reskit.com. IN  SRV 0  0  80      webserver1.noam.reskit.com.
_http._tcp.reskit.com. IN  SRV 10 0  80      webserver2.noam.reskit.com.
```

Note This example does not specify a TTL. Therefore, the resolver uses the minimum TTL specified in the SOA resource record.

If a computer needs to locate a Web server in the reskit.com DNS domain, the resolver sends the following query:

```
_http._tcp.www.reskit.com.
```

The DNS server replies with the SRV records listed above. The resolver then chooses between WebServer1 and WebServer2 by looking at their priority values. Because WebServer1 has the *lowest* priority value, the DNS server chooses WebServer1.

Note If the priority values had been the same, but the weight values had been different, the client would have chosen a Web server randomly, except that the server with the *highest* weight value would have had a higher probability of being chosen.

Next, the resolver requests the A record for webserver1.reskit.com, and the DNS server sends the A record. Finally, the client attempts to contact the Web server. For more information about SRV records, see the link to the Internet Engineering Task Force (IETF) on the Web Resources page at http://windows.microsoft.com/windows2000/reskit/webresources. Windows 2000 supports the Internet Draft titled “A DNS RR for specifying the location of services (DNS SRV).”

Less Common Resource Records

Table 5.3 shows some other resource records and the RFCs that define them. Many of these resource records are considered experimental.

Table 5.3 Less Common Resource Record Types

Record Type	RFC	Description
AAAA	1886	Special address record that maps a host (computer or other network device) name to an IPv6 address.
AFSDB	1183	Gives the location of either an Andrew File System (AFS) cell database server, or a Distributed Computing Environment (DCE) cell's authenticated server. The AFS system uses DNS to map a DNS domain name to the name of an AFS cell database server. The Open Software Foundation's DCE Naming Service uses DNS for a similar function.
HINFO	1035	The host information resource record identifies a host's hardware type and operating system. The CPU Type and Operating System identifiers come from the computer names and system names listed in RFC 1700.
	1183	The Integrated Services Digital Network (ISDN) resource record is a variation of the A (address) resource record. Rather than mapping an FQDN to an IP address, the ISDN record maps the name to an ISDN address. An ISDN address is a phone number that consists of a country/region code, an area code or country/region code, a local phone number, and optionally, a subaddress. The ISDN resource record is designed to be used in conjunction with the route through (RT) resource record.
MB	1035	The mailbox (MB) resource record is an experimental record that specifies a DNS host with the specified mailbox. Other related experimental records are the mail group (MG) resource record, the mailbox rename (MR) resource record, and the mailbox information (MINFO) resource record.
MG	1035	The mail group (MG) resource record is an experimental record that specifies a mailbox that is a member of the mail group (mailing list) specified by the DNS domain name. Other related experimental records are the MB resource record, the MR resource record, and the MINFO resource record.
MINFO	1035	The MINFO resource record is an experimental record that

specifies a mailbox that is responsible for the specified mailing list or mailbox. Other related experimental records are the MB resource record, the MG resource record, and the MR resource record.

(continued)

Table 5.3 Less Common Resource Record Types *(continued)*

Record Type	RFC	Description
MR	1035	The MR resource record is an experimental record that specifies a mailbox that is the proper rename of another specified mailbox. Other related experimental records are the MB resource record, the MG resource record, and the MINFO resource record.
RP	1183	Identifies the responsible person (RP) for the specified DNS domain or host.
RT	1183	The route through (RT) resource record specifies an intermediate host that routes packets to a destination host. The RT record is used in conjunction with the ISDN and X25 resource records. It is syntactically and semantically similar to the MX record type and is used in much the same way.
TXT	1035	The text resource (TXT) record associates general textual information with an item in the DNS database. A typical use is for identifying a host's location (for example, Location: Building 26S, Room 2499). A single TXT record can contain multiple strings, up to 64 kilobytes (KB).
WKS	1035	The well-known service (WKS) resource record describes the services provided by a particular protocol on a particular interface. The protocol is usually UDP or TCP, but can be any of the entries listed in the Windows 2000 Protocols file located in %*SystemRoot*%\System32\Drivers\Etc\Protocol. The services are the services below port number 256 from the Windows 2000 Services file located in %*SystemRoot*%\System32\Drivers\Etc\Services.
X.25	1183	The X.25 resource record is a variation of the A (address) resource record. Rather than mapping a FQDN to an IP address, the X.25 record maps the name to an X.121 address. X.121 is the International Standards Organization (ISO) standard that specifies the format of addresses used in X.25 networks. The X.25 resource record is designed to be used in conjunction with the route through (RT) resource record.

Resource Records Not Defined in RFCs

In addition to the resource record types listed in the RFCs, Windows 2000 uses the following resource record types, shown in Table 5.4.

Table 5.4 Resource Record Types Not Defined in the RFCs

Name	Description
WINS	The Windows 2000 DNS server can use a WINS server for looking up the host portion of a DNS name that does not exist in the DNS zone authoritative for the name.
WINS reverse lookup (WINS-R)	This entry is used in a reverse lookup zone for finding the host portion of the DNS name if given its IP address. A DNS server issues a NetBIOS adapter status query if the zone authoritative for the queried IP address does not contain the record and does contain the WINS-R resource record.
ATMA	The ATMA resource record, defined by the ATM Forum, is used to map DNS domain names to ATM addresses. For more information, contact the ATM Forum for the ATM Name System Specification Version 1.0.

Delegation and Glue Records

Delegation and glue records are records that you add to a zone in order to delegate a subdomain into a separate zone. A *delegation* is an NS record in the parent zone that lists the name server authoritative for the delegated zone. A *glue record* is an A record for the name server authoritative for the delegated zone.

For example, suppose the name server for the DNS domain reskit.com, delegated authority for the noam.reskit.com zone to the name server noamNS.noam.reskit.com. You add the following records to the reskit.com zone:

```
noam.reskit.com.           IN      NS  noamNS.noam.reskit.com
noamNS.noam.reskit.com.         IN      A   172.16.54.1
```

Delegations are necessary for name resolution. Glue records are also necessary if the name server authoritative for the delegated zone is also a member of that domain. A glue record is necessary in the example above because noamNS.noam.reskit.com. is a member of the delegated domain noam.reskit.com. However, if it was a member of a different domain, the resolver can perform standard name resolution to resolve the name of the authoritative name server to an IP address.

When a resolver submits a query for a name in the child zone to the name server that is authoritative for the parent zone, the server authoritative for the parent zone checks its zone. The delegation tells it which name server is authoritative for the child zone. The server authoritative for the parent zone can then return a referral to the resolver.

Zones

The DNS standards do not specify the internal data structure that stores resource records, and various implementations differ. Generally, servers use zones stored on that server in plain text, but it is not required. With Windows 2000, you can integrate your DNS database with the Active Directory database, in which case the zones are stored in the Active Directory database.

One common implementation of DNS, the Berkeley Internet Name Domain (BIND) implementation, generally uses the file names shown in Table 5.5.

Table 5.5 Zone Names Used in BIND

Name	Description
db.*domain*	Forward lookup zone. For example, if your DNS domain is reskit.com, then this file is called db.reskit.com.
db.*addr*	Reverse lookup zone. For example, if your network is the class C network address 172.16.32 then this file is called db.172.16.32.
db.cache	Also known as the *root hints file*, this file contains the names and IP addresses for the name servers that maintain the root DNS domain. This file is essentially the same on all servers that use Internet root DNS servers, but must be modified for servers that use private root DNS servers. (A *root DNS server* is a DNS server that is authoritative for the root of the namespace.)
db.127.0.0.1	Used to resolve queries to the loopback address. It is essentially the same on all name servers.

The names of the database files are arbitrary and are specified in the configuration of the DNS server. By default, the Microsoft Windows 2000 DNS server does not use the same file names as a typical BIND DNS server but instead uses *zone_name*.dns. However, if you are porting DNS db files from another DNS server, you can configure the Microsoft Windows 2000 DNS server to use the BIND file names.

The following sections explain the contents of the zones and describe one additional file, the BOOT file, which is used by BIND servers, though not specified in the DNS standards.

Forward Lookup Zone

Forward lookup zones contain information needed to resolve names within the DNS domain. They must include SOA and NS records and can include any type of resource record except the PTR resource record.

Reverse Lookup Zone

Reverse lookup zones contain information needed to perform reverse lookups. They usually include SOA, NS, PTR, and CNAME records.

With most queries, the client supplies a name and requests the IP address that corresponds to that name. This type of query is typically described as a *forward lookup*.

But what if a client already has a computer's IP address and wants to determine the DNS name for the computer? This is important for programs that implement security based on the connecting FQDN, and is also used for TCP/IP network troubleshooting. The DNS standard provides for this possibility through *reverse lookups*.

If the only means to answer a reverse lookup were to conduct a thorough search of all DNS domains in the DNS namespace, the reverse query search would be too exhaustive to perform in any practical way.

To solve this problem, a special DNS domain called in-addr.arpa was created. This domain uses a reverse ordering of the numbers in the dotted-decimal notation of IP addresses. With this arrangement, administration of lower limbs of the in-addr.arpa domain can be delegated to organizations as they are assigned their class A, B, or C IP network IDs. For more information about creating classless reverse lookup zones, see "Windows 2000 DNS" in this chapter. See also RFC 2317, "Classless IN-ADDR.ARPA delegation."
Figure 5.5 shows a branch of the in-addr.arpa namespace.

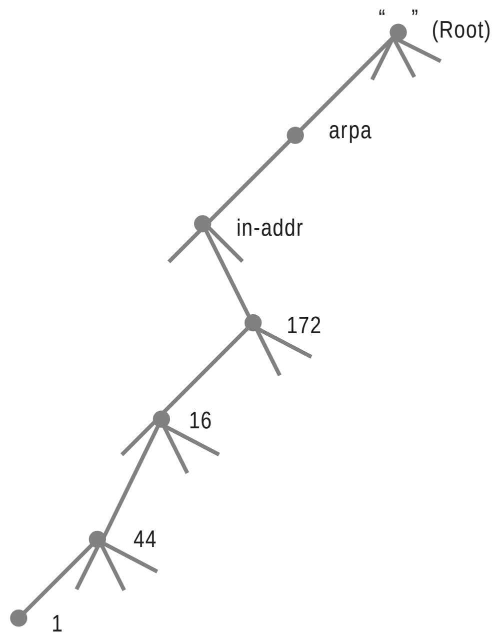

Figure 5.5 In-addr.arpa Namespace

The in-addr.arpa domain tree requires PTR resource records to store and provide reverse mappings for IP addresses of their corresponding FQDNs.
If a client needs to find the FQDN associated with the IP address 172.16.44.1, the client queries for the PTR record of the 1.44.16.172.in-addr.arpa domain name.

Inverse Queries

In addition to reverse lookups, some DNS servers support what is known as an inverse query. Just as with a reverse lookup, a client making an inverse query provides the IP address and requests the FQDN. However, the server does not use the in-addr.arpa domain to find the answer, and it does not query any other servers. Instead, it simply checks its own zones for the answer, and if it does not find the answer, it returns an error message. There is no way for either the server or the client to know whether the IP address is simply missing from the zones of that server, or whether the IP address does not exist.

Because support for inverse queries is optional and because servers often cannot provide a definitive answer, inverse queries are of limited use. Only certain applications use inverse queries, such as earlier versions of **nslookup**.

The Windows 2000 server responds to inverse query requests by replying with the IP address specified in the query enclosed in square brackets. For example, if it receives an inverse query for 172.16.72.1, it responds with [172.16.72.1].

For more information about inverse queries, see RFC 1035.

Root Hints Files

The *root hints file*, also called the *cache hints file*, contains host information that is needed to resolve names outside of the authoritative DNS domains. It contains the names and addresses of root DNS servers.

If your network is connected to the Internet, the root hints file should contain records for the root DNS servers on the Internet. In Windows 2000, a root hints file with current Internet root DNS mappings is installed as the file Cache.dns in the directory %*SystemRoot*%\System32\Dns.

If your network is not connected to the Internet, you must replace the NS and A records in the cache file with NS and A records for the DNS servers that are authoritative for the root of your private TCP/IP network.

For example, suppose you have created two internal root DNS servers (InternalRoot1.reskit.com. and InternalRoot2.reskit.com.). You then create a root hints file on other DNS servers in your network, pointing to the internal root DNS servers. That way, if other name servers receive a query they cannot resolve, they can simply query the internal root servers specified in the file.

The following reskit root hints file provides NS resource records and name-to-IP address mappings for name servers in the reskit.com domain.

```
; Internal root hints file for reskit.com, which is not connected to
; the Internet
```

```
.   86400       IN      NS      InternalRoot1.reskit.com.
.   86400       IN      NS      InternalRoot2.reskit.com.

InternalRoot1.reskit.com.    86400   IN  NS      172.16.64.1
InternalRoot2.reskit.com.    86400   IN  NS      172.16.64.2
```

Note The Windows 2000 Configure DNS Server wizard makes a best effort to determine whether the network is connected to the Internet and, if not, creates its own cache file.

Boot Files

Although the boot file is not actually defined in the RFCs and is not needed for a server to be RFC compliant, it is described here for completeness. This file is part of the BIND-specific implementation of DNS.

It is not used by default in the Microsoft implementation of DNS. If advantageous, you can port an existing BIND boot file to a Microsoft DNS server. The Windows 2000 DNS server supports only a subset of the BIND boot file directives, and only for the file type used by BIND 4.*x.x* servers.

The BIND Boot file controls the startup behavior of the DNS server. Commands must start at the beginning of a line and no spaces can precede commands. Table 5.6 shows descriptions of some of the boot file commands supported by Windows 2000.

Table 5.6 Descriptions of Boot File Commands

Command	Description
Directory command	Specifies a directory where other files referred to in the boot file can be found.
Cache command	Specifies a file used to help the DNS server contact name servers for the root domain. This command and the file it refers to must be present.
Primary command	Specifies a primary zone for which this name server is authoritative and a zone file that contains the resource records for that zone. Multiple primary command records can exist in the boot file.
Secondary command	Specifies a secondary zone for which this name server is authoritative and a list of IP addresses of master servers from which to attempt zone transfer. It also defines the name of the local file for saving this zone. Multiple secondary command records can exist in the boot file.

Table 5.7 shows the syntax of some of the boot file commands supported by Windows 2000.

Table 5.7 Syntax of Boot File Commands Supported by Windows 2000

Syntax	Example
directory <*directory*>	directory c:\winnts\system32\dns
cache <*file name*>	cache cache
primary <*domain*> <*file name*>	primary reskit.com reskit.com.dns primary dev.reskit.com dev.reskit.com.dns
secondary <*domain*> <*hostlist*> <*local file name*>	secondary test.reskit.com 157.55.200.100 test.reskit.com.dns
forwarders <*hostlist*>	forwarders 172.16.64.4 172.16.64.8
slave (follows the forwarders parameter)	forwarders 172.16.64.4 172.16.64.8 slave

For more information about the BIND boot file, see the link to the Microsoft TechNet Web site on the Web Resources page at http://windows.microsoft.com/windows2000/reskit/webresources. On the TechNet Web site, search for the keywords "structure" and "domain name system" and "boot file."

Zone Transfer

When changes are made to the zone on a master server, these changes must be replicated to all the secondary servers for that zone, using a mechanism called *zone transfer*. In the original DNS specifications, only one form of zone transfer was available, known as full zone transfer. New RFCs discuss an additional type of zone transfer: incremental zone transfer. This section describes both types. It also describes DNS Notify, a mechanism by which the master server for a zone can notify the secondary servers of zone changes.

Full Zone Transfer

In a *full zone transfer*, defined in the original DNS specifications, the master server for a zone transmits the entire zone database to the secondary server for that zone. Secondary servers initiate full zone transfers using the following process:

1. The secondary server for the zone waits a certain amount of time (specified in the **Refresh** field of the SOA resource record), and then polls the master server for its SOA.
2. The master server for the zone responds with the SOA resource record.
3. The secondary server for the zone compares the returned serial number to its own serial number. If the serial number sent by the master server for the zone is higher than its own serial number, that means its zone database is out of date, and it sends an AXFR request (a request for a full zone transfer).

4. The master server for the zone sends the full zone database to the secondary server.

If the master server for the zone does not respond to polling by the secondary server, the secondary server continues to retry after the interval specified in the **Retry** field of the SOA resource record. If there is still no answer after the interval specified in the **Expire** field since the last successful zone transfer, it discards its zone.

Note Name servers running versions of BIND earlier than 4.9.4 can send and receive only one resource record per message during a full zone transfer. Name servers running versions of BIND 4.9.4 and later, and Windows 2000, can send and receive multiple resource records per message. This improves the performance of full zone transfers.

However, to provide backward compatibility, name servers running Windows 2000 default to sending only one resource record per message if any secondary servers that are not running Windows are configured for the zone. If you have secondary name servers running versions of BIND 4.9.4 and later, configuring Windows 2000 to send multiple resource records per message improves performance. For more information, see the link to the Microsoft TechNet Web site on the Web Resources page at http://windows.microsoft.com/windows2000/reskit/webresources. On the TechNet Web site, search for "DNS Compatibility," "BIND," and "4.9.4."

Incremental Transfer

Full zone transfers can consume a great deal of network bandwidth, especially for complex DNS configurations. To solve this problem, RFC 1995 specifies an additional standard, *incremental zone transfer*. With incremental zone transfer, only the modified part of the zone must be transferred.

Incremental zone transfer works much the same as full zone transfer. The secondary server for the zone still uses the SOA resource record to determine when to poll the master server for the zone, when to retry, and so on. However, if it needs to perform a zone transfer, it sends an incremental zone transfer (IXFR) query instead of an AXFR query, requesting that the master server for the zone perform an incremental zone transfer.

The master server for the zone, meanwhile, maintains a recent version history of the zone, which observes any record changes that occurred in the most recent version updates of the zone. Then, if the master server for the zone has a newer version of the zone, it can forward only those record changes that have occurred between the two different versions of the zone (the current versions on the master and secondary servers) to the secondary server for the zone. The master server sends the oldest updates first and the newest updates last.

When the secondary server receives an incremental zone transfer, it creates a new version of the zone and begins replacing its resource records with the updated resource records, starting with the oldest updates and ending with the newest updates. When all of the updates have been made, the secondary server replaces its old version of the zone with the new version of the zone.

The master server for the zone is not required to perform an incremental transfer. It can perform a full zone transfer if it does not support incremental zone transfer, if it does not have all the necessary data for performing an incremental zone transfer, or if an incremental zone transfer takes more bandwidth than a full zone transfer.

Even though incremental zone transfer saves network bandwidth, it uses space on the server to record the version history. To conserve space, servers can purge the version history.

DNS Notify

DNS Notify is a revision to the DNS standard (RFC 1996) that proposes that the master server for a zone notify certain secondary servers in that zone of changes, and the secondary servers can then check to see whether they need to initiate a zone transfer. This process can help improve consistency of zone data among secondary servers.

To determine which secondary servers in a zone to send the changes to, the master server for the zone contains a *notify list*, which is a list of the IP addresses for those secondary servers. The master server for the zone notifies only listed servers when zone updates occur.

When the local zone on a master server for the zone is updated, the following events take place:

1. The **Serial Number** field in the SOA record is updated to indicate that a new version of the zone has been written to a disk.
2. The master server then sends a notify message to other servers that are part of its notify list.
3. All secondary servers for the zone that receive the notify message respond by initiating an SOA-type query back to the notifying master server to determine if the zone of the notifying server is a later version than the currently stored copy of the zone.
4. If a notified server determines that the serial number used in the SOA record of the zone of the notifying server is higher (more recent) than the serial number used in the SOA record for its current zone copy, the notified server requests an AXFR or IXFR zone transfer.

Note On servers running Windows 2000, you can configure the Notify set.

Dynamic Update

Dynamic update is a new standard, specified in RFC 2136, that provides a means of dynamically updating zone data on a zone's primary server.

Originally, DNS was designed to support only static changes to a zone database. Because of the design limitations of static DNS, the ability to add, remove, or modify resource records could only be performed manually by a DNS system administrator. For example, a DNS system administrator would edit records on a zone's primary server and the revised zone database is then propagated to secondary servers during zone transfer. This design is workable when the number of changes is small and updates occur infrequently, but can otherwise become unmanageable.

With dynamic update, on the other hand, the primary server for the zone can also be configured to support updates that are initiated by another computer or device that supports dynamic update. For example, it can receive updates from workstations registering A and PTR resource records, or from DHCP servers. Updates are sent using a standard UPDATE message format and can include the addition or deletion of individual resource records (RRs) or sets of resource records (RRsets). For information about the UPDATE message format, see RFC 2136.

In order for a request for a dynamic update to be performed, several prerequisite conditions can also be identified. Where prerequisites are set, all such conditions must be met before an update is allowed. Some examples of prerequisites that can be set are:

- A required RR or RRset already exists or is in use prior to an update.
- A required RR or RRset does not exist or is not in use prior to an update.
- A requester is permitted to initiate an update of a specified RR or RRset.

Each prerequisite must be satisfied in order for an update to occur. After all prerequisites are met, the zone's primary server can then proceed with an update of its local zones. Multiple updates can be processed concurrently only if one update does not depend on the final result of another update.

DNS Standards

Although the core DNS standards (as set forth in RFCs 1034 and 1035) have been well-established since their acceptance as standards in 1987, numerous revisions to DNS have been made in the intervening years. These revisions have been brought about by additional RFCs and Internet drafts, which are independently authored and submitted to the IETF for further circulation and review before being accepted as standards throughout the Internet.

Table 5.8 lists the accepted and proposed RFC standards that Windows 2000 supports.

Table 5.8 DNS-related RFC Standards Supported by Windows 2000

Number	Status	Title
1034	Standard	Domain Names - Concepts and Facilities
1035	Standard	Domain Names - Implementation and Specification
1123	Standard	Requirements for Internet Hosts - Application and Support
1886	Proposed	DNS Extensions to Support IP Version 6
1995	Proposed	Incremental Zone Transfer in DNS
1996	Proposed	A Mechanism for Prompt DNS Notification of Zone Changes
2136	Proposed	Dynamic Updates in the Domain Name System (DNS UPDATE)

2181	Standards Track	Clarifications to the DNS Specification
2308	Standards Track	Negative Caching of DNS Queries (DNS NCACHE)

Additional Resources

- For more information about conceptual information about DNS, see *DNS and BIND, 3rd Edition* by Paul Albitz and Cricket Liu, 1998, O'Reilley & Associates, Inc..
- For more information about Requests for Comments (RFCs) and Internet drafts, see the link to the IETF on the Web Resources page at http://windows.microsoft.com/windows2000/reskit/webresources.