DNS Architecture

07/02/2012 • 17 minutes to read

In this article

Applies To: Windows Server 2008

DNS architecture is a hierarchical distributed database and an associated set of protocols that define:

- A mechanism for querying and updating the database.
- A mechanism for replicating the information in the database among servers.
- A schema of the database.

DNS originated in the early days of the Internet when the Internet was a small network established by the United States Department of Defense for research purposes. The host names of the computers in this network were managed through the use of a single HOSTS file located on a centrally administered server. Each site that needed to resolve host names on the network downloaded this file. As the number of hosts on the Internet grew, the traffic generated by the update process, as well as the size of the HOSTS file, increased. The need for a new system, which would offer features such as scalability, decentralized administration, support for various data types, became more and more obvious.

The Domain Name System introduced in 1984 became this new system. With DNS, the host names reside in a database that can be distributed among multiple servers, decreasing the load on any one server and providing the ability to administer this

naming system on a per-partition basis. DNS supports hierarchical names and allows registration of various data types in addition to host name-to-IP address mapping used in HOSTS files. Because the DNS database is distributed, its potential size is unlimited and performance is not degraded when more servers are added.

The original DNS was based on Request for Comment (RFC) 882 (Domain Names: Concepts and Facilities) and RFC 883 (Domain Names–Implementation and Specification), which were superseded by RFC 1034 (Domain Names–Concepts and Facilities), and RFC 1035 (Domain Names–Implementation and Specification). Additional RFCs that describe DNS security, implementation, and administrative issues later augmented the original design specifications.

The implementation of DNS — Berkeley Internet Name Domain (BIND) — was originally developed for the 4.3 BSD UNIX operating system. The Microsoft implementation of DNS became a part of the operating system in Microsoft Windows NT Server 4.0. The Windows NT 4.0 DNS server, like most DNS implementations, has its roots in RFCs 1034 and 1035.

The RFCs used in the Windows Server® 2008 operating system are 1034, 1035, 1886, 1996, 1995, 2136, 2308, and 2052.

DNS domain names

The Domain Name System is implemented as a hierarchical and distributed database containing various types of data, including host names and domain names. The names in a DNS database form a hierarchical tree structure called the domain namespace. Domain names consist of individual labels separated by dots, for example: mydomain.microsoft.com.

A fully qualified domain name (FQDN) uniquely identifies the host's position within the DNS hierarchical tree by specifying a list of names separated by dots in the path from the referenced host to the root. The following figure shows an example of a DNS tree with a host called mydomain within the microsoft.com. domain. The FQDN for the host would be mydomain.microsoft.com.

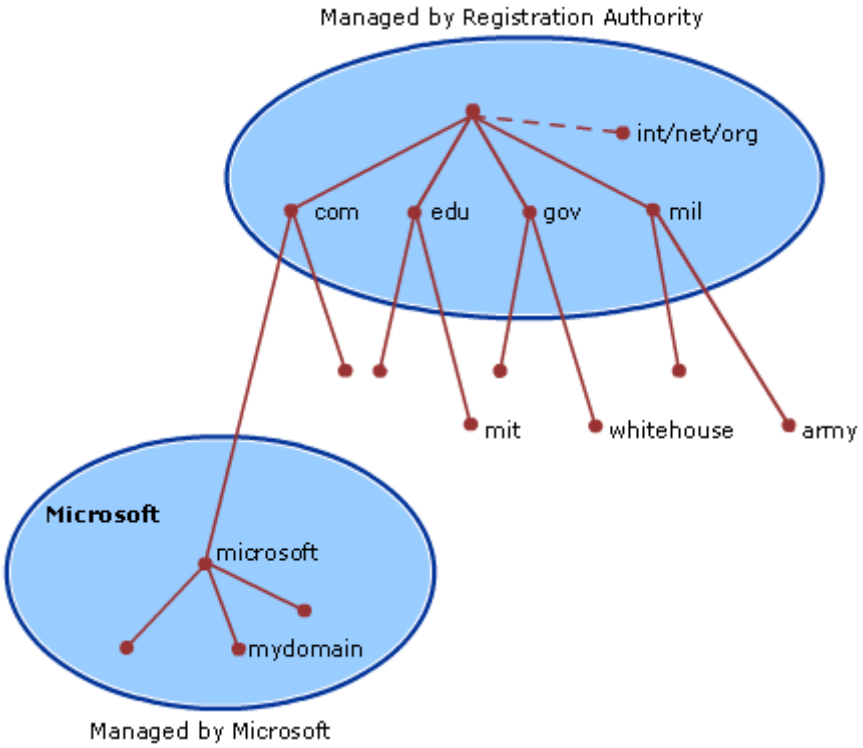
Understanding the DNS domain namespace

The DNS domain namespace, as shown in the following figure, is based on the concept of a tree of named domains. Each level of the tree can represent either a branch or a leaf of the tree. A branch is a level where more than one name is used to identify a

collection of named resources. A leaf represents a single name used once at that level

to indicate a specific resource.

DNS Domain Name Hierarchy



This figure shows how Microsoft is assigned authority by the Internet root servers for its own part of the DNS domain namespace tree on the Internet. DNS clients and servers use queries as the fundamental method of resolving names in the tree to specific types of resource information. This information is provided by DNS servers in query responses to DNS clients, which then extract the information and pass it to a requesting program for resolving the queried name. In the process of resolving a name, keep in mind that DNS servers often function as DNS clients, querying other servers in order to fully resolve a queried name.

How the DNS domain namespace is organized

Any DNS domain name used in the tree is technically a domain. Most DNS discussions, however, identify names in one of five ways, based on the level and the way a name is commonly used. For example, the DNS domain name registered to Microsoft (microsoft.com.) is known as a second-level domain. This is because the name has two parts (known as labels) that indicate it is located two levels below the root or top of the tree. Most DNS domain names have two or more labels, each of which indicates a new level in the tree. Periods are used in names to separate labels.

The five categories used to describe DNS domain names by their function in the namespace are described in the following table, along with an example of each name

type.

Types of DNS domain names

Name Type	Description	Example
Root domain	This is the top of the tree, representing an unnamed level; it is sometimes shown as two empty quotation marks (""), indicating a null value. When used in a DNS domain name, it is stated by a trailing period (.) to designate that the name is located at the root or highest level of the domain hierarchy. In this instance, the DNS domain name is considered to be complete and points to an exact location in the tree of names. Names stated this way are FQDNs.	A single period (.) or a period used at the end of a name, such as "example.microsoft.com."
Top-level domain	A name used to indicate a country/region or the type of organization using a name.	".com", which indicates a name registered to a business for commercial use on the Internet.
Second-level domain	Variable-length names registered to an individual or organization for use on the Internet. These names are always based on an appropriate top-level domain, depending on the type of organization or geographic location where a name is used.	""microsoft.com. ", which is the second-level domain name registered to Microsoft by the Internet DNS domain name registrar.
Subdomain	Additional names that an organization can create that are derived from the registered second-level domain name. These include names added to grow the DNS tree of names in an organization and divide it into departments or geographic locations.	""example.microsoft.com. ", which is a fictitious subdomain assigned by Microsoft for use in documentation example names.
Host or resource name	Names that represent a leaf in the DNS tree of names and identify a specific resource. Typically, the leftmost label of a DNS domain name identifies a specific computer on the network. For example, if a name at this level is used in a host (A) resource record, it is used to look up the IP address of computer based on its host name.	""host-a.example.microsoft.com.", where the first label ("host-a") is the DNS host name for a specific computer on the network.

DNS and Internet domains

The Internet Domain Name System is managed by a Name Registration Authority on the Internet, responsible for maintaining top-level domains that are assigned by organization and by country/region. These domain names follow the International Standard 3166. Some of the many existing abbreviations, reserved for use by organizations, as well as two-letter and three-letter abbreviations used for countries/regions are shown in the following table:

Some DNS top-level domain names (TLDs)

DNS Domain Name	Type of Organization
com	Commercial organizations
edu	Educational institutions
org	Non-profit organizations
net	Networks (the backbone of the Internet)
gov	Non-military government organizations
mil	Military government organizations
arpa	Reverse DNS
"*xx*"	Two-letter country code (for example, us, au, ca, fr)

Resource records

A DNS database consists of resource records (RRs). Each RR identifies a particular resource within the database. There are various types of RRs in DNS. This section provides information about the common structure of resource records.

The following table provides detailed information about the structure of common RRs.

Common DNS resource records

Description	Class	Time to Live (TTL)	Type	Data

| Description | Class | Time to Live (TTL) | Type | Data |

Start of Authority	Internet (IN)	Default TTL is 60 minutes	SOA	Owner Name Primary Name Server DNS Name, Serial Number Refresh Interval Retry Interval Expire Time Minimum TTL
Host	Internet (IN)	Record-specific TTL if present, or else zone (SOA) TTL	A	Owner Name (Host DNS Name) Host IP Address
Name Server	Internet (IN)	Record-specific TTL if present, or else zone (SOA) TTL	NS	Owner Name Name Server DNS Name
Mail Exchanger	Internet (IN)	Record-specific TTL if present, or else zone (SOA) TTL	MX	Owner Name Mail Exchange Server DNS Name, Preference Number
Canonical Name (an alias)	Internet (IN)	Record-specific TTL if present, or else zone (SOA) TTL	CNAME	Owner Name (Alias Name) Host DNS Name

Distributing the DNS database: zone files and delegation

A DNS database can be partitioned into multiple zones. A zone is a portion of the DNS database that contains the resource records with the owner names that belong to the contiguous portion of the DNS namespace. Zone files are maintained on DNS servers. A single DNS server can be configured to host zero, one, or multiple zones.

Each zone is anchored at a specific domain name referred to as the zone's root domain. A zone contains information about all names that end with the zone's root domain name. A DNS server is considered authoritative for a name if it loads the zone

containing that name. The first record in any zone file is a Start of Authority (SOA) RR. The SOA RR identifies a primary DNS name server for the zone as the best source of information for the data within that zone and as an entity processing the updates for the zone.

A name within a zone can also be delegated to a different zone that is hosted on a different DNS server. Delegation is a process of assigning responsibility for a portion of a DNS namespace to a DNS server owned by a separate entity. This separate entity can be another organization, department, or workgroup within your company. Such delegation is represented by the NS resource record that specifies the delegated zone and the DNS name of the server authoritative for that zone. Delegating across multiple zones was part of the original design goal of DNS.

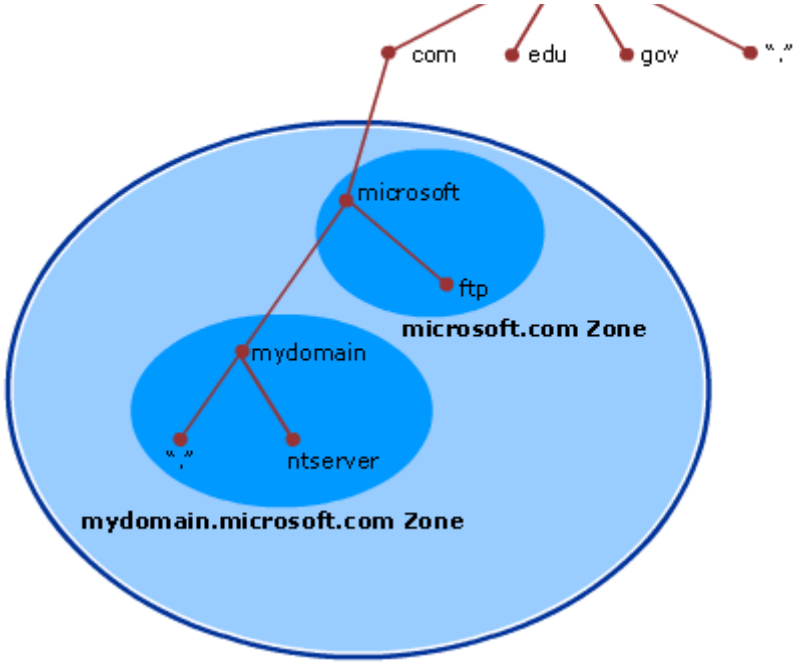
The primary reasons to delegate a DNS namespace include:

- A need to delegate management of a DNS domain to a number of organizations or departments within an organization.
- A need to distribute the load of maintaining one large DNS database among multiple DNS servers to improve the name resolution performance as well as create a DNS fault-tolerant environment.
- A need to allow for a host's organizational affiliation by including the host in appropriate domains.

The name server (NS) RRs facilitate delegation by identifying DNS servers for each zone and the NS RRs appear in all zones. Whenever a DNS server needs to cross a delegation in order to resolve a name, it will refer to the NS RRs for DNS servers in the target zone.

In the following figure, the management of the microsoft.com. domain is delegated across two zones, microsoft.com. and mydomain.microsoft.com.

DNS Delegation



> **Note**
>
> If multiple NS records exist for a delegated zone identifying multiple DNS servers available for querying, the Windows Server 2008 DNS Server service will be able to select the closest DNS server based on the roundtrip intervals measured over time for every DNS server.

Replicating the DNS database

There can be multiple zones representing the same portion of the namespace. Among these zones there are three types:

- Primary
- Secondary
- Stub

Primary is a zone to which all updates for the records that belong to that zone are made. A secondary zone is a read-only copy of the primary zone. A stub zone is a read-only copy of the primary zone that contains only the resource records that identify the DNS servers that are authoritative for a DNS domain name. Any changes made to the primary zone file are replicated to the secondary zone file. DNS servers hosting a primary, secondary, or stub zone are said to be authoritative for the DNS names in the zone.

Because a DNS server can host multiple zones, it can therefore host both a primary

zone (which has the writeable copy of a zone file) and a separate secondary zone (which obtains a read-only copy of a zone file). A DNS server hosting a primary zone is said to be the primary DNS server for that zone, and a DNS server hosting a secondary zone is said to be the secondary DNS server for that zone.

> **Note**
>
> A secondary or stub zone cannot be hosted on a DNS server that hosts a primary zone for the same domain name.

Zone transfer

The process of replicating a zone file to multiple DNS servers is called zone transfer. Zone transfer is achieved by copying the zone file from one DNS server to a second DNS server. Zone transfers can be made from both primary and secondary DNS servers.

A master DNS server is the source of the zone information during a transfer. The master DNS server can be a primary or secondary DNS server. If the master DNS server is a primary DNS server, then the zone transfer comes directly from the DNS server hosting the primary zone. If the master server is a secondary DNS server, then the zone file received from the master DNS server by means of a zone transfer is a copy of the read-only secondary zone file.

The zone transfer is initiated in one of the following ways:

- The master DNS server sends a notification (RFC 1996) to one or more secondary DNS servers of a change in the zone file.
- When the DNS Server service on the secondary DNS server starts, or the refresh interval of the zone has expired (by default it is set to 15 minutes in the SOA RR of the zone), the secondary DNS server will query the master DNS server for the changes.

Types of zone file replication

There are two types of zone file replication. The first, a full zone transfer (AXFR), replicates the entire zone file. The second, an incremental zone transfer (IXFR), replicates only records that have been modified.

BIND 4.9.3 and earlier DNS server software, as well as Windows NT 4.0 DNS, support full zone transfer (AXFR) only. There are two types of the AXFR: one requires a single record per packet, the other allows multiple records per packet. The DNS Server service in

Windows 2000 and Windows Server 2003 supports both types of zone transfer, but by default uses multiple records per packet. It can be configured differently for compatibility with servers that do not allow multiple records per packet, such as BIND servers versions 4.9.4 and earlier.

Querying the database

DNS queries can be sent from a DNS client (resolver) to a DNS server, or between two DNS servers.

A DNS query is merely a request for DNS resource records of a specified resource record type with a specified DNS name. For example, a DNS query can request all resource records of type A (host) with a specified DNS name.

There are two types of DNS queries that can be sent to a DNS server:

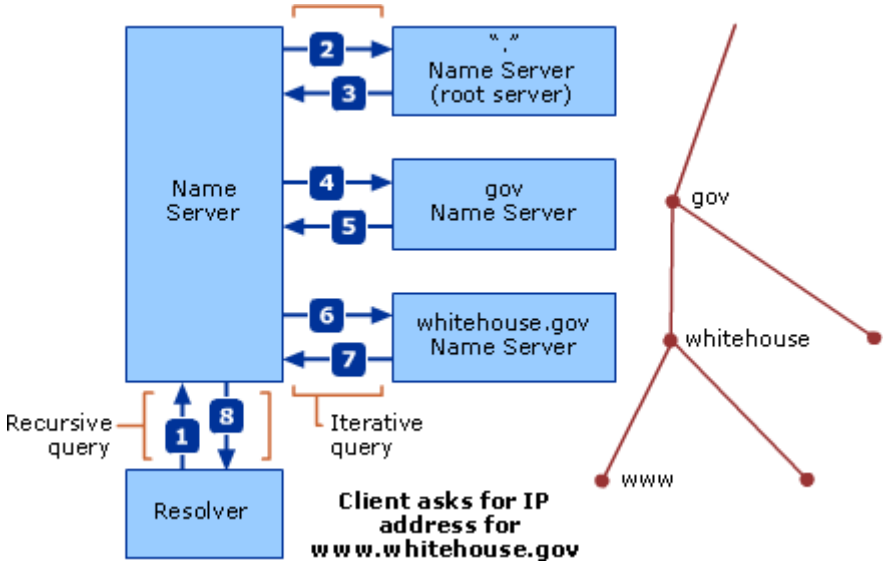
- Recursive
- Iterative

A recursive query forces a DNS server to respond to a request with either a failure or a success response. DNS clients (resolvers) typically make recursive queries. With a recursive query, the DNS server must contact any other DNS servers it needs to resolve the request. When it receives a successful response from the other DNS server (or servers), it then sends a response to the DNS client. The recursive query is the typical query type used by a resolver querying a DNS server and by a DNS server querying its forwarder, which is another DNS server configured to handle requests forwarded to it.

When a DNS server processes a recursive query and the query cannot be resolved from local data (local zone files or cache of previous queries), the recursive query must be escalated to a root DNS server. Each standards-based implementation of DNS includes a cache file (or root server hints) that contains entries for the root DNS servers of the Internet domains. (If the DNS server is configured with a forwarder, the forwarder is used before a root server is used.)

An iterative query is one in which the DNS server is expected to respond with the best local information it has, based on what the DNS server knows from local zone files or from caching. This response is also known as a referral if the DNS server is not authoritative for the name. If a DNS server does not have any local information that can answer the query, it simply sends a negative response. A DNS server makes this type of query as it tries to find names outside of its local domain (or domains) (when it is not configured with a forwarder). It might have to query a number of outside DNS servers in an attempt to resolve the name.
The following figure shows an example of both types of queries.

DNS Query Types



The figure shows a number of queries were used to determine the IP address for www.whitehouse.gov. The query sequence is described as follows:

1. Recursive query for www.whitehouse.gov (A resource record)
2. Iterative query for www.whitehouse.gov (A resource record)
3. Referral to the .gov name server (NS resource records, for .gov); for simplicity, iterative A queries by the DNS server (on the left) to resolve the IP addresses of the Host names of the name server's returned by other DNS servers have been omitted.
4. Iterative query for www.whitehouse.gov (A resource record)
5. Referral to the whitehouse.gov name server (NS resource record, for whitehouse.gov)
6. Iterative query for www.whitehouse.gov (A resource record)
7. Answer to the iterative query from whitehouse.gov server (www.whitehouse.gov's IP address)
8. Answer to the original recursive query from local DNS server to resolver (www.whitehouse.gov's IP address)

Time-to-Live for resource records

The Time-to-Live (TTL) value in a resource record indicates a length of time used by other DNS servers to determine how long to cache information for a record before expiring and discarding it. For example, most resource records created by the DNS

Server service inherit the minimum (default) TTL of one hour from the start of authority (SOA) resource record, which prevents extended caching by other DNS servers.

A DNS client resolver caches the responses it receives when it resolves DNS queries. These cached responses can then be used to answer later queries for the same information. The cached data, however, has a limited lifetime specified in the TTL parameter returned with the response data. TTL ensures that the DNS server does not keep information for so long that it becomes out of date. TTL for the cache can be set on the DNS database (for each individual resource record, by specifying the TTL field of the record and per zone through the minimum TTL field of the SOA record) as well as on the DNS client resolver side by specifying the maximum TTL the resolver allows to cache the resource records.

There are two competing factors to consider when setting the TTL. The first is the accuracy of the cached information, and the second is the utilization of the DNS servers and the amount of network traffic. If the TTL is short, then the likelihood of having old information is reduced considerably, but it increases utilization of DNS servers and network traffic, because the DNS client must query DNS servers for the expired data the next time it is requested. If the TTL is long, the cached responses could become outdated, meaning the resolver could give false answers to queries. At the same time, a long TTL decreases utilization of DNS servers and reduces network traffic because the DNS client answers queries using its cached data.

If a query is answered with an entry from cache, the TTL of the entry is also passed with the response. This way the resolvers that receive the response know how long the entry is valid. The resolvers honor the TTL from the responding server; they do not reset it based on their own TTL. Consequently, entries truly expire rather than live in perpetuity as they move from DNS server to DNS server with an updated TTL.

> **Note**
>
> In general, never configure the TTL to zero. The difference between a setting of 0 or 60 is minimal to the accuracy of the record, but when the TTL is set to 0, there is a significant impact on DNS server performance because the DNS server is constantly querying for the expired data.

Updating the DNS database

Because the resource records in the zone files are subject to change, they must be updated. The implementation of DNS in Windows Server 2008 supports both static and

dynamic updates of the DNS database. The details of the dynamic update are discussed

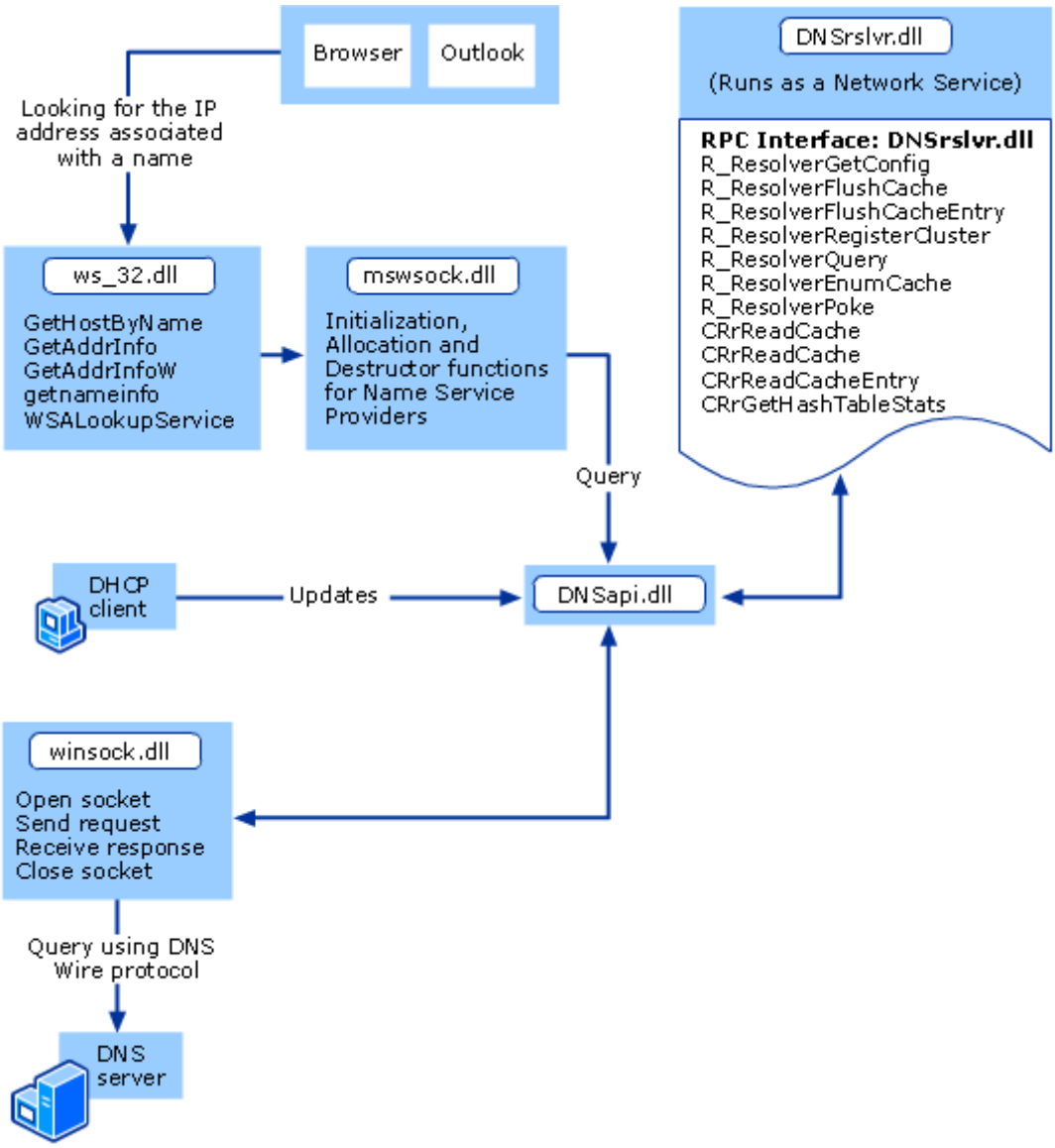
in DNS Processes and Interactions and DNS Physical Structure.

DNS architecture diagrams

The following diagrams illustrate how the DNS Client and Server services work and provide additional information about name resolution, update, and administration operations.

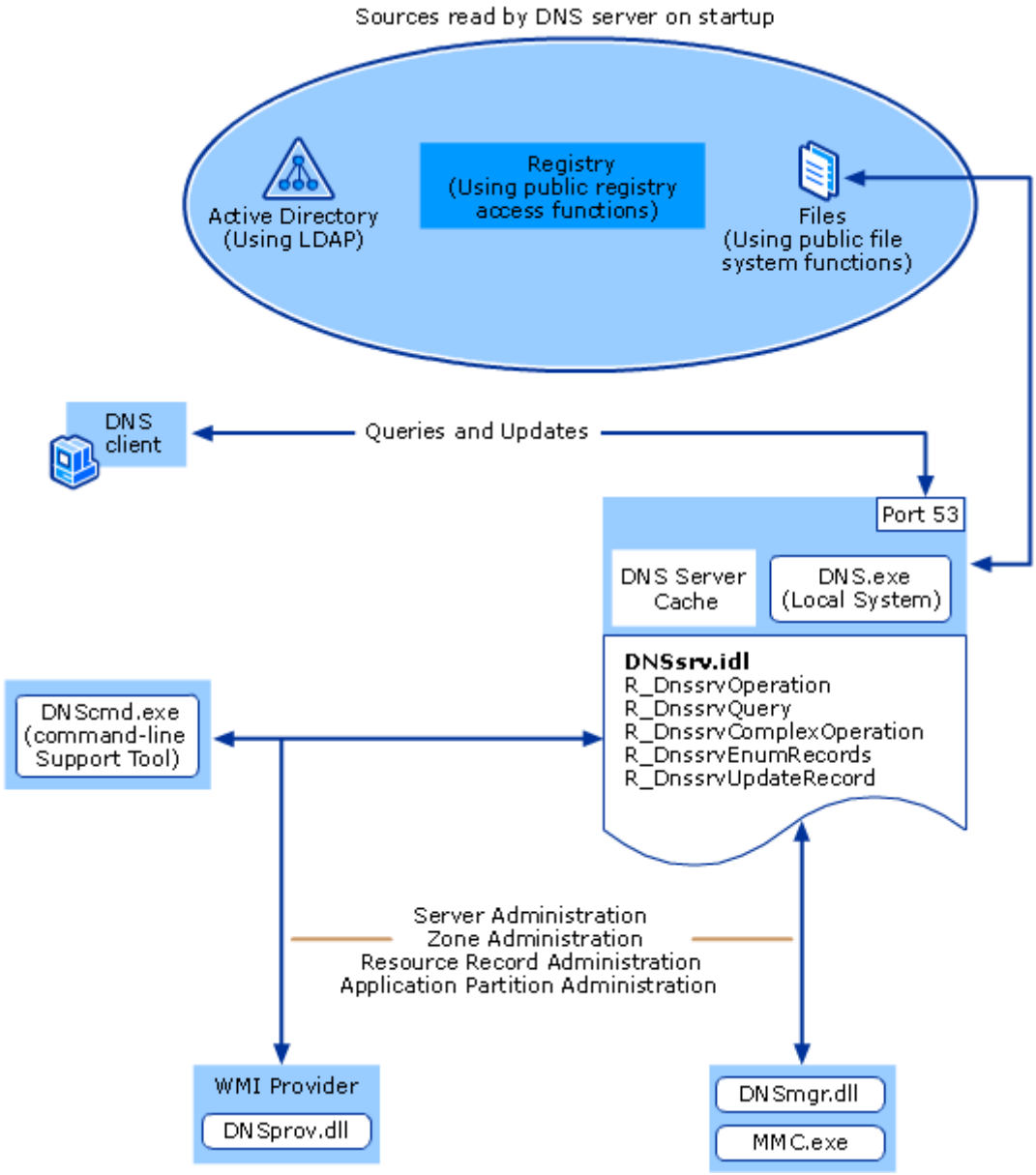
The first diagram illustrates the DNS Client service architecture in its name resolution and update operations. In this diagram, name resolution architecture is demonstrated using a Web browser and Microsoft Outlook and updates are represented by the DHCP client.

DNS Client Service Architecture



The following diagram illustrates the DNS Server service architecture with its administration tools and the Windows Management Instrumentation (WMI) interface.

DNS Server Service Architecture


Sources read by DNS server on startup
Active Directory
(Using LDAP)
Registry
(Using public registry
access functions)
Files
(Using public file
system functions)
DNS
client
Queries and Updates
Port 53
DNS Server
Cache
DNS.exe
(Local System)
DNSsrv.idl
R_DnssrvOperation
R_DnssrvQuery
R_DnssrvComplexOperation
R_DnssrvEnumRecords
R_DnssrvUpdateRecord
DNScmd.exe
(command-line
Support Tool)
Server Administration
Zone Administration
Resource Record Administration
Application Partition Administration
WMI Provider
DNSprov.dll
DNSmgr.dll
MMC.exe