Microsoft®

Microsoft®
Windows NT® Server

Server Operating System

White Paper

DNS and Microsoft Windows NT 4.0

Special thanks to the following people for their help in this project:

Azfar Moazzam; James Gilroy; Dan Perry; Jim Harrison; Dave Macdonald; Steven Judd; Prakash Narasimhamurthy; Margaret Johnson; Rodger Seabourne

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

Microsoft Corporation • One Microsoft Way • Redmond, WA 98052-6399 • USA

Microsoft®
BackOffice™

Microsoft® Windows NT® Server

DNS and Microsoft Windows NT 4.0

This paper provides an overview of the Domain Name System (DNS) and how it can be implemented using Microsoft Windows NT 4.0.

With the onset of Enhanced Directory Services for the Windows NT operating system, which will be coming in a later release of Windows NT, the Domain Name System Server will be much more important than it was in any other Windows NT release. Because of this, installing and designing effective DNS implementations today will help with tomorrow’s migration to the next version of Windows NT.

CONTENTS

INTRODUCTION

The use of the Domain Name System (DNS) service in the Microsoft Windows NT 4.0 operating system is optional.

The DNS service that ships with Windows NT 4.0 is there if you need to use it, however there is nothing in Windows NT Directory Services today that requires it. We would like you to start using it for typical DNS-type activity.

It is important to note that DNS is NOT an alternative way to look at your Directory Services Domains (that is there is no Browser or Netlogon support and so forth).

If you choose to use DNS, this paper shows you how you may want to design your DNS infrastructure today in preparation for the future release of Windows NT Enhanced Directory Services (DS).

What we are about to experience is a paradigm shift. Up until today, the group using Windows NT within a company could play in their own world and design Windows NT domains, trusts, user accounts, shares, and so on. The Windows NT groups never had to worry about the Directory Services world—there was usually some other group within the company that worried about X.500, DNS, and similar services. With the onset of the Enhanced DS, which is slated to be in the next major version of Windows NT, this will change and these two groups will have to work together. This will be important because the Enhanced DS will be similar to X.500 and will use DNS to locate servers providing these Directory Services. That is why this paper is important. It will allow the groups using Windows NT to gain an understanding of the other Directory Services that are available. This paper is intended to help these two groups understand each others needs so that a smooth migration to enhanced Directory Services can occur.

This paper will first define the DNS technology (this is an important section for anyone that is new to DNS). It will then take a look at the Microsoft specific DNS (in this section, we talk about why the Microsoft DNS solution is the correct choice for a Microsoft or mixed environment), it will then go over some architectures (this section is important for anyone that is about to design a DNS solution) and last, will talk about the future of Windows NT (this is an important section for everyone to read).

We hope you enjoy this paper and find it of value. If you have any comments concerning the contents of this document, please e-mail **Scott Suhy (scottsu@microsoft.com)** or **Glenn Wood (glennwo@microsoft.com)** with Microsoft Corporation.

Before we examine the Microsoft implementation of DNS, it is important for the reader to know a little bit about the history of DNS. If you are already familiar with DNS and its functionality, you may want to skip past this section of the paper and go directly to "Implementing DNS using Windows NT 4.0."

History of DNS

The Domain Name System (DNS) is a set of protocols and services on a TCP/IP network which allows users of the network to utilize hierarchical user-friendly names when looking for other hosts (that is computers) instead of having to remember and use their IP addresses. This system is used extensively on the Internet and in many private enterprises today. If you've used a Web browser, Telnet application, FTP utility or other similar TCP/IP utilities on the Internet, then you have probably used a DNS server.

The DNS protocols best-known function is mapping user-friendly names to IP addresses. For example, suppose the FTP site at Microsoft had an IP address of **157.55.100.1**. Most people would reach this computer by specifying **FTP.microsoft.com** and not the less friendly IP address. Besides being easier to remember, the name is more reliable. The numeric address could change for any number of reasons, but the name can always be used.

Before the implementation of DNS, the use of user-friendly computer names was done through the use of HOSTS files which contained a list of names and associated IP addresses. On the Internet, this file was centrally administered and each location would periodically download a new copy. As the number of machines on the Internet grew, this became an unmanageable solution. The need for something better arose. This better solution became DNS.

According to Dr. Paul Mockapetris, principle designer of DNS, the original design goal for DNS was to replace this cumbersome singularly administered HOSTS file with a lightweight distributed database that would allow for a hierarchical name space, distribution of administration, extensible data types, virtually unlimited database size, and reasonable performance.

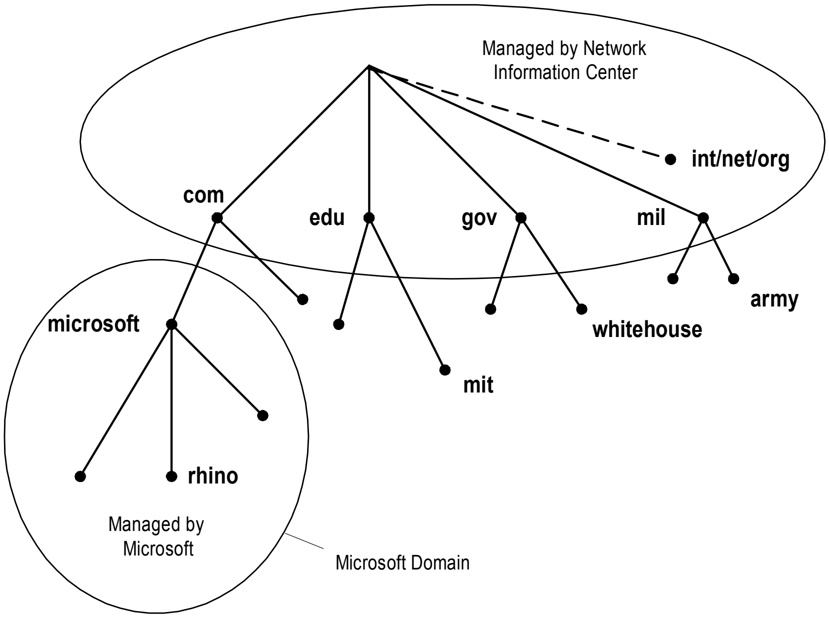
UNDERSTANDING THE TECHNOLOGY

DNS maps to level 7 in the OSI model and can use either UDP or TCP as the underlying protocol. Resolvers send UDP queries to servers first for increased performance and only resort to TCP if truncation of the returned data occurs.

The most popular implementation of the DNS protocol "*BIND*" was originally developed at Berkeley for the 4.3 BSD UNIX operating system. The name "BIND" stands for Berkeley Internet Name Domain. The primary specifications for DNS are defined in Requests for Comments (RFCs) 974, 1034, and 1035.

Overview

A Domain Name System is composed of a distributed database of names. The names in the DNS database establish a logical tree structure called the *domain name space*. Each node or domain in the domain name space is named and can contain subdomains. Domains and subdomains are grouped into zones to allow for distributed administration of the name space (zones will be discussed later in this section). The domain name identifies the domain's position in the logical DNS hierarchy in relation to its parent domain by separating each branch of the tree with a period ".". The following figure shows a few of the top level domains, where the Microsoft domain fits, and a host called "rhino" within the "microsoft.com" domain. If someone wanted to contact that host, they would use the *Fully Qualified Domain Name* (FQDN) **rhino.microsoft.com**.



DNS Servers and the Internet

The root of the DNS database on the Internet is managed by the Internet

Network Information Center (http://www.internic.com). The top-level domains were assigned organizationally and by country. These domain names follow the International Standard 3166. Two-letter and three-letter abbreviations are used for countries, and various abbreviations are reserved for use by organizations, as shown in the following examples.

com	Commercial (for example, microsoft.com for Microsoft Corporation)
edu	Educational (for example, mit.edu for Massachusetts Institute of Technology)
gov	Government (for example, whitehouse.gov for the Whitehouse in Washington D.C.)
int	International organizations (for example, nato.int for NATO)
mil	Military operations (for example, army.mil for the Army)
net	Networking organizations (for example, nsf.net for NSFNET)
org	Noncommercial organizations (for example, fidonet.org for FidoNet)

Domains

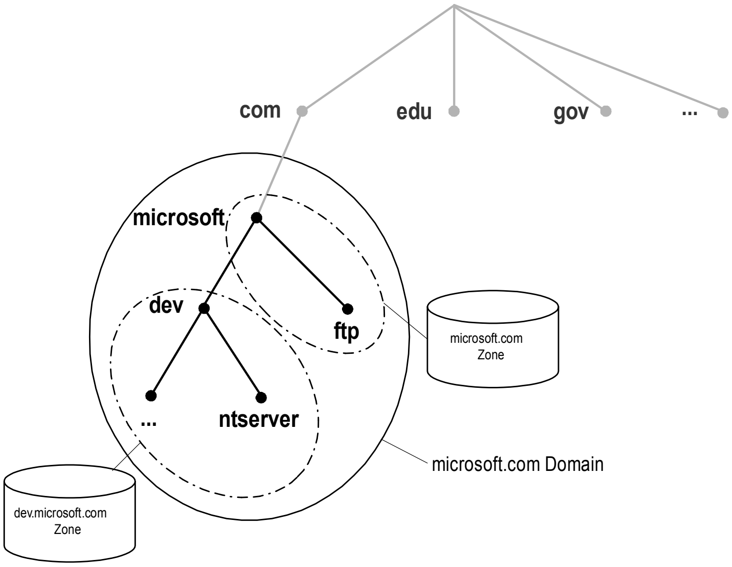
Each node in the tree of a DNS database, along with all the nodes below it, is called a *domain*. Domains can contain both hosts (computers) and other domains (subdomains). For example, the Microsoft domain **microsoft.com** could contain both computers such as **FTP.microsoft.com** and subdomains such as **dev.microsoft.com** which could contain hosts such as **ntserver.dev.microsoft.com**.

In general, Domain names and Host names have restrictions in their naming which only allow the use of characters “a-z”, “A-Z”, “0-9”, and “-” (dash or minus sign). The use of characters such as the “/”, “.”, and “_” (slash, period, and underscore) are not allowed.

Zones

A *zone* is some portion of the DNS namespace whose database records exist and are managed in a particular zone file. A single DNS server might be configured to manage one or multiple zone files. Each zone is anchored at a specific domain node—referred to as the zone’s “root domain.” Zone files do not necessarily contain the complete tree (that is all subdomains) under the zone’s root domain. For a comparison of domains and zones, look at the figure that follows. In this example, microsoft.com is a domain but the entire domain is not controlled by one zone file. Part of the domain is actually broken off into a separate zone file for dev.microsoft.com. Breaking up domains across multiple zone files might be needed for distributing management of the domain to different groups or possibly for efficiencies in data replication (that is zone transfers which will be discussed later).

It is very important that you understand the difference between a zone and a domain. A zone is a physical file composed of resource records that defines a group of domains. A domain is a node in the DNS namespace and all subdomains below it.



Name Servers

DNS servers store information about the domain name space and are referred to as name servers. Name servers generally have one or more zones for which they are responsible. The name server is then said to have *authority* for those zones.

When you configure a DNS name server (as we will soon see with the “NS” record), you tell it what are all of the other DNS name servers that are in the same domain.

Primary, Secondary, and Master Name Servers

A *primary name server* is a name server that gets the data for its zones from local files. Changes to a zone, such as adding domains or hosts, are done at the Primary Name Server. A *secondary name server* gets the data for its zones from another name server across the network which is authoritative for that zone. The processes of obtaining this zone information (that is the database file) across the network is referred to as a *zone transfer*.

There are three reasons to have *secondary* servers within an enterprise. Those reasons are:

- **Redundancy**

  You need at least two DNS name servers serving each zone, a primary and at least one secondary for redundancy. Like any fault tolerant system, the machines should be as independent as possible (that is different networks and so forth).
- **Remote locations**

  You should also have secondary servers (or other primary servers for subdomains) in remote locations that have a large number of clients. You would not want these clients to have to communicate across slow links for name resolution.
- **Reduce load on the primary**

  You also need secondary servers to reduce the load on the primary server.

Since information for each zone is stored in separate files, this primary or secondary designation is defined at a zone level. In other words, a particular name server may be a primary name server for certain zones and a secondary name server for other zones.

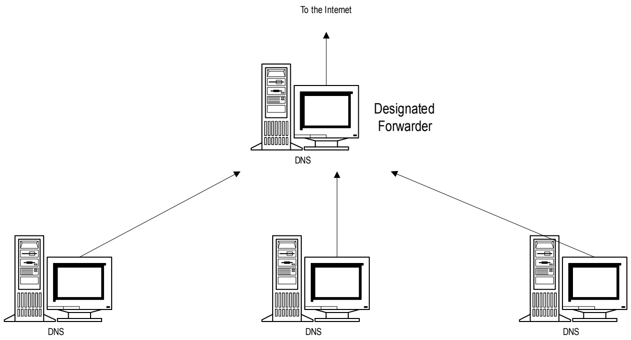
When defining a zone on a name server as a secondary, you must designate a name server from which to obtain the zone information. The source of zone information for a secondary name server in a DNS hierarchy is referred to as a *master name server*. A master name server can be either a primary or secondary name server for the requested zone. When a secondary name server starts up, it contacts its master name server and initiates a zone transfer with that server.

Use secondary servers as master servers when the primary is overloaded, or when there is a more efficient network path between "secondary to secondary" vs. "secondary to primary."

Forwarders and Slaves

When a DNS name server receives a DNS request, it attempts to locate the requested information within its own zone files. If this fails because the server is not authoritative for the domain requested, it must communicate with other DNS name servers to resolve the request. Since, on a globally connected network, a DNS resolution request outside a local zone typically requires interaction with DNS name servers outside of the company on the public Internet, you may want to selectively enable specific DNS name servers in your company to do this wide-area communication.

To address this issue, DNS allows for the concept of *forwarders*. Specific DNS name servers are selected to be forwarders, and only forwarders are allowed to carry out the wide-area communications across the Internet. All other DNS name servers within the company are configured to use forwarders and are configured with the IP addresses of the DNS name servers designated as forwarders. This configuration is done on a per server basis, not a per zone basis!



When a server which is configured to use forwarders receives a DNS request that it is unable to resolve (through its own zone files), it passes the request to one of the designated forwarders. The forwarder then carries out whatever communication is necessary to resolve the request and returns the results to the requesting server, which, in turn, returns the results to the original requester. If the forwarder is unable to resolve the query, the DNS server attempts to resolve the query on its own as normal.

Slaves are DNS servers that have been configured to use forwarders and have also been configured to return a failure message if the forwarder is unable to resolve the request. Slaves make no attempt to contact other name servers if the forwarder is unable to satisfy the request.

Caching-only Servers

Although all DNS name servers cache queries that they have resolved, *Caching-only servers* are DNS name servers whose only job is to perform queries, cache the answers, and return the results. In other words, they are not authoritative for any domains and only contain information which they have cached while resolving queries.

When trying to determine when to use such a server, keep in mind that when the server is initially started it has no cached information and must build up this information over time as it services requests. However, if you are dealing with a slow link between sites then there is much less traffic sent across the slow link because the server is not doing a zone transfer.

Name Resolution

There are three types of queries that a client can make to a DNS server, *recursive, iterative* and *inverse*. While discussing name resolution, keep in mind that a DNS server can be a client to another DNS server.

Recursive queries

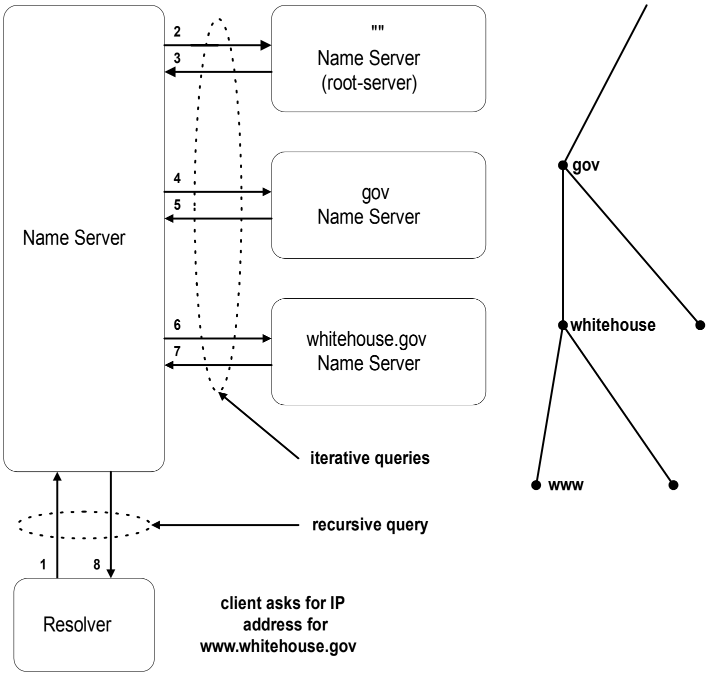
In a recursive query, the queried name server is petitioned to respond with the requested data, or with an error stating that data of the requested type doesn’t exist or that the domain name specified doesn’t exist. The name server ***can not*** just refer the querier to a different name server.

This type of query is typically done by a DNS client (a *resolver*) to a DNS server. Also, if a DNS server is configured to use a forwarder, the request from this DNS server to its forwarder will be a recursive query.

Iterative queries

In an iterative query, the queried name server gives the best answer it currently has back to the querier. This type of query is typically done by a DNS server to other DNS servers after it has received a recursive query from a resolver.

The following figure shows an example of both types of queries. Query 1/8 is a recursive query from a client resolver to its DNS server while 2/3, 4/5, and 6/7 are iterative queries from the DNS server to other DNS servers.



Getting the Host Name Given the IP Address

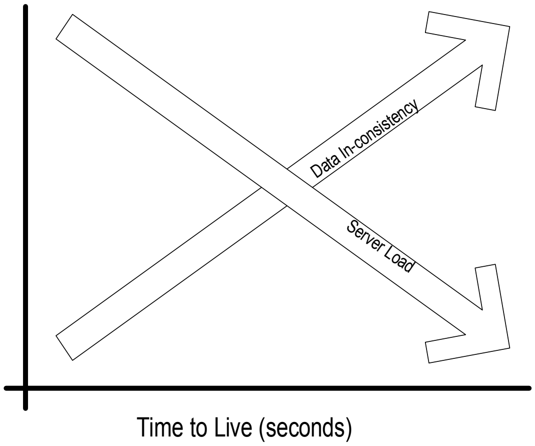
What if a resolver has the IP Address and would like to know the Host name for a particular machine? Instead of supplying a name and asking for an IP address, the client needs to provide the IP address and asks for the name. Since there is no direct correlation in the DNS name space between the domain names and the associated IP addresses they contain, only a thorough search of all domains could guarantee a correct answer.

To alleviate this problem, a special domain “in-addr.arpa.” in the DNS name space was created. Nodes in the in-addr.arpa domain are named after the numbers in the dotted-octet representation of IP addresses. But since IP addresses get more specific from left to right and domain names get less specific from left to right, the order of IP address octets must be reversed when building the in-addr.arpa tree. With this arrangement, administration of lower limbs of the DNS in-addr.arpa tree can be given to companies as they are assigned their class A, B, or C subnet address.

Once the domain tree is built into the DNS database, a special pointer record is added to associate the IP addresses to the corresponding host names. In other words, to find a host name for the IP address 157.55.200.2, the resolver would query the DNS server for a pointer record for 2.200.55.157.in-addr.arpa. If this IP address was outside the local domain, the DNS server would start at the root and sequentially resolve the domain nodes until he reached 200.55.157.in-addr.arpa which should contain the resource PTR record for 2 (that is 157.55.200.2).

Caching and Time to Live

When a name server is processing a recursive query, it may be required to send out several queries to find the definitive answer. The name server caches all of the information that it receives during this process for a time which is specified in the returned data. This amount of time is referred to as the **Time to Live** (TTL). The name server administrator of the zone that contains the data decides on the TTL for the data. Smaller TTL values will help ensure that data about your domain is more consistent across the network if this data changes often. However, this will also increase the load on your name server.



Once data is cached by a DNS server, it must start decreasing the TTL (that is counting down) from its original value so that it will know when to flush the data from its cache. If a query comes in that can be satisfied by this cached data, the TTL that is returned with the data is the current amount of time left before the data is flushed from the DNS server cache. Client resolvers also have data caches and honor the TTL value so that they know when to expire the data.

Overview

Most DNS systems must be configured by editing text files. With Microsoft DNS, as with all Microsoft Windows®-based products, there is a user friendly interface. The new administration interface makes it much easier to configure both local and remote Microsoft DNS servers. The DNS administrative tool configures the RFC-compatible text files for you.

Even though the graphical user interface give you the ability to modify the DNS files without an editor, you should know the makeup of the DNS system configuration files. In RFC compliant DNS systems, there are several files which define the DNS system configuration and database. These files include the database, cache, reverse lookup, and 127 reverse lookup files. These files also exist in the Windows NT 4.0 DNS and are also RFC compliant. These files are explained in detail in this section.

The Database File

A database file or "zone file" is the file which contains the *resource records* for that part of the domain for which the zone is responsible. Some of the common resource records are given below. For a more complete list, look in the Appendix of this document or refer to the appropriate RFCs. Windows NT 4.0 supplies a file as a template to work with called "place.dns.*"* This file should be edited and renamed before you use it on a production DNS server. It is generally a good idea to name this file the same as the zone it represents. This is the file that will be replicated between masters and secondaries.

The Start of Authority

The first record in any database file is the SOA Record.

IN SOA <source host> <contact e-mail> <ser. no.> <refresh time> <retry time> <expiration time> <TTL>

- **source host**—the host on which this file is maintained.
- **contact e-mail**—the Internet e-mail address for the person responsible for this domain's database file.
-

Instead of writing the "@" symbol in the e-mail name as would normally be done, the "@" must be replaced with a "." when placed in the zone files. In other words, the e-mail address glennwo@microsoft.com would be represented as glennwo.microsoft.com in the zone file.

-
- **serial number**—the "version number" of this database file. This number should increase each time the database file is changed.
- **refresh time**—the elapsed time (in seconds) that a secondary server will wait between checks to its master server to see if the database file has changed and a zone transfer should be requested.
- **retry time**—the elapsed time (in seconds) that a secondary server will wait before retrying a failed zone transfer.
- **expiration time**—the elapsed time (in seconds) that a secondary server will keep trying to download a zone. After this time limit expires, the old zone information will be discarded.
- **Time to Live**—the elapsed time (in seconds) that a DNS server is allowed to cache any resource records from this database file. This is the value that is sent out with all query responses from this zone file when the individual

resource record does not contain an overriding value.

In order for a resource record to span a line in a database file, parentheses must enclose the line breaks.

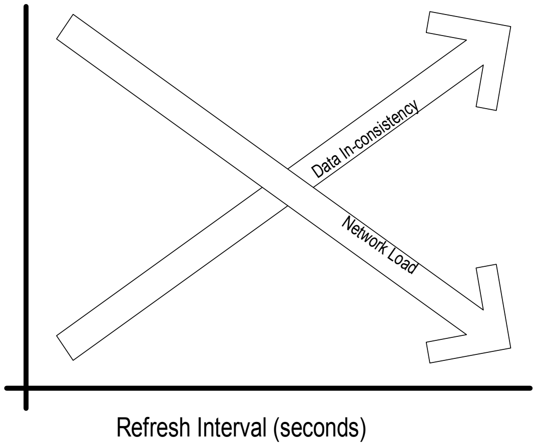
In a zone file, the “@” symbol represents the root domain of the zone (microsoft.com in the following examples). The “IN” in the following records is the class of data. It stands for Internet. Other classes exist, but none of them are currently in widespread use.

Any domain name in the database file which is not terminated with a period “.” will have the root domain appended to the end.

Example:

```
@  IN SOA  nameserver1.microsoft.com. glennwo.microsoft.com. (

           1         ; serial number

           10800     ; refresh [3 hours]

           3600      ; retry [1hour]

           604800  ; expire [7 days]

           86400 ) ; time to live [1 day]
```

Setting the servers refresh interval is a balance between data consistency (accuracy of your data) and your networks load.



The Name Server Record

Lists the name servers for this domain allowing other name servers to lookup names in your domain.

<domain> IN NS <nameserver host >

Example:

```
@ IN NS  nameserver2.microsoft.com.

@ IN NS  nameserver3.microsoft.com.
```

The Mail Exchange Record

This record tells us what host processes mail for this domain. If multiple mail exchange records exist, the resolver will attempt to contact the mail servers in order of preference from lowest value (highest priority) to highest value (lowest priority). By using the example records that follow, mail addressed to scottsu@microsoft.com is delivered to scottsu@mailserver0.microsoft.com first if possible and then to scottsu@mailserver1.microsoft.com if mailserver0 is unavailable.

<domain> IN MX <preference> <mailserver host >

Example:

```
@    IN MX  1  mailserver0

@    IN MX  2  mailserver1
```

The Host Record

A host record is used to statically associate hosts names to IP addresses within a zone. It should contain entries for all hosts which require static mappings including workstations, name servers, mail servers, etc. These are the records which make up most of the database file when static records are used.

<host name> IN A <ip address of host>

Example:

```
rhino           IN A  157.55.200.143

nameserver2     IN A  157.55.200.2

mailserver1     IN A  157.55.200.51
```

The Local Host Record

A local host record allows lookups for "localhost.microsoft.com." to return 127.0.0.1.

localhost IN A 127.0.0.1

The CNAME Record

These records are sometimes called "aliases" but are technically referred to as "Canonical Name” (CNAME) entries. These records allow you to use more than one name to point to a single host.

Using canonical names makes it easy to do such things as host both an FTP server and a Web server on the same machine.

<host alias name> IN CNAME <host name>

Example:

Assume that www.microsoft.com and FTP.microsoft.com are on the same machine. If this is the case then you might have the following entries in your zone file:

```
FileServer1  IN A                  157.55.200.41
FTP                 CNAME FileServer1
www                 CNAME FileServer1
```

If you ever intend on moving the FTP server service away from the Web service, then all you have to do is change the CNAME in the DNS server for FTP and add an address record for the new server. For example:

```
FTP                CNAME FileServer2
FileServer2  IN A                    157.55.200.42
```

The Cache File

The cache file contains host information that is needed to resolve names outside the authoritative domains. It contains names and addresses of root name servers. For users on the Internet, the default file provided with the Microsoft DNS Service should suffice. For installations NOT connected to the Internet, the file should be replaced to contain the name servers authoritative for the root of your private network.

For a current Internet cache file see **FTP://rs.internic.net/domain/named.cache**

Example:

```
; DNS CACHE FILE
;
; Initial cache data for root domain servers.
;
; YOU SHOULD CHANGE:
; —Nothing if connected to the Internet. Edit this file only when
;   update root name server list is released.
;   OR
; —If NOT connected to the Internet, remove these records and replace
;   with NS and A records for the DNS server authoritative for the
;   root domain at your site.
; Internet root name server records:
;  last update: Sep 1, 1995
;  related version of root zone: 1995090100
;
; formerly NS.INTERNIC.NET
.      3600000 IN NS A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.  3600000  A  198.41.0.4
; formerly NS1.ISI.EDU
.      3600000  NS B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.  3600000  A  128.9.0.107
; formerly C.PSI.NET
.      3600000  NS C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.  3600000  A  192.33.4.12
; formerly TERP.UMD.EDU
.      3600000  NS D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.  3600000  A  128.8.10.90
; formerly NS.NASA.GOV
.      3600000  NS E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.  3600000  A  192.203.230.10
; formerly NS.ISC.ORG
.      3600000  NS F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.  3600000  A  39.13.229.241
; formerly NS.NIC.DDN.MIL
.      3600000  NS G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.  3600000  A  192.112.36.4
; formerly AOS.ARL.ARMY.MIL
.      3600000  NS H.ROOT-SERVERS.NET.
; End of File
```

The Reverse Lookup File

This is a database file that is used for reverse-lookups in particular IP DNS zones of host names when supplied with the IP numbers. This allows a resolver to provide an IP address and request a matching host name. This file contains SOA and name server records similar to other DNS database zone files. It also contains pointer records.

This DNS reverse-lookup capability is important because some applications provide the capabilities to implement security based on the connecting host names. For instance, if a client tries to link to a Network File System (NFS) volume with this security arrangement, the NFS server would contact the DNS server and do a reverse-name lookup on the clients IP address. If the host name returned by the DNS server is not in the access list for the NFS volume or if the host name was not found by DNS, then the NFS mount request would be denied. This reverse-lookup capability is often used for troubleshooting reasons as well, which will be discussed later in this document.

Here are a couple example zones for different IP class networks.

Example class C zone:

100.200.192.in-addr.arpa

Example class B zone:

55.157.in-addr.arpa

The Pointer Record

Pointer records provide a static mapping of IP addresses to host names within a reverse-lookup zone. IP numbers are written in backward order and "in-addr.arpa." is appended to the end to create this pointer record. As an example, looking up the name for "157.55.200.51" requires a PTR query for the name "51.200.55.157.in-addr.arpa."

<ip reverse domain name> IN PTR <host name>

Example:

```
51.200.55.157.in-addr.arpa.  IN PTR  mailserver1.microsoft.com.
```

The Arpa-127.rev File

This is a database file for the 127.in-addr.arpa. domain. This domain is used for reverse-lookups of IP numbers in the 127 network, such as "localhost." The only thing in this file that changes is the SOA and NS records.

The BIND Boot File

Although the boot file is not actually defined in the RFCs and is not needed in order to be RFC compliant, it is described here for completeness. This file is actually a part of the BIND specific implementation of DNS. Microsoft DNS can be configured to use a boot file if you are going to administer it through changes to the text files instead of using the DNS Administrator UI.

This file controls the startup behavior of the DNS server. Commands must start at the beginning of a line and no spaces may precede commands. Recognized commands are: "directory, cache, primary, and secondary." The syntax for this file is as follows:

Directory Command

Specifies a directory where other files referred to in the boot file can be found.

directory <directory>

Example:

```
directory c:\winnts\system32\dns
```

Cache Command

Specifies a file used to help the DNS service contact name servers for the root domain. This command and the file it refers to MUST be present. A cache file suitable for use on the Internet is provided with Windows NT 4.0.

cache . <filename>

Example:

```
cache . cache
```

Primary Command

Specifies a domain for which this name server is authoritative and a database file which contains the resource records for that domain (that is zone file). Multiple primary command records could exist in the boot file.

primary <domain> <filename>

Example:

```
primary microsoft.com microsoft.dns
primary dev.microsoft.com dev.dns
```

Secondary Command

Specifies a domain for which this name server is authoritative, and a list of master server IP addresses from which to attempt downloading the zone information, rather than reading it from a file. It also defines the name of the local file for caching this zone. Multiple secondary command records could exist in the boot file.

secondary <domain> <hostlist> <local filename>

Example:

```
secondary test.microsoft.com 157.55.200.100 test.dns
```

Forwarders Command

Specifies another server that is willing to try resolving recursive queries on behalf of the system.

forwarders <hostlist>

Example:

```
forwarders 157.55.200.100 157.55.200.101
```

Slave Command

Specifies that the use of forwarders is the only possible way to resolve queries. Can only follow a forwarders command.

slave

Example:

```
forwarders 157.55.200.100 157.55.200.101
slave
```

You can use Microsoft DNS if you want to replace UNIX DNS servers with Windows NT-based systems or if you need to integrate WINS enabled Microsoft-based clients and servers into a UNIX-based environment.

INTRODUCTION TO MICROSOFT DNS

Some people might ask, what is the Microsoft DNS and why should I use it? Well let's start out by telling you what it is not. First, the Microsoft DNS server is not a port of the Berkley BIND code (which is currently at revision 10.4 as of the writing of this paper). We made a conscious decision to not port the BIND code, but rather write our own code that was fully RFC compliant and compatible with BIND. We made this decision because we wanted to be able to easily add performance enhancements to the product. The Microsoft DNS server is also not the code that was shipped in the Windows NT Resource Kit. If you used that utility, you probably noticed that it had problems with zone transfers. The DNS server service in Windows NT 4.0 is a complete rewrite and is not just a bug-fixed version. Rest assured that in the DNS included with Windows NT 4.0, RFC compliance has been thoroughly tested and it all works as defined, including zone transfers.

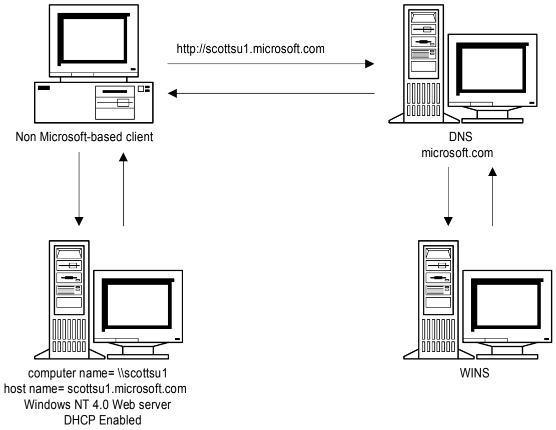
Now let's talk about what the Microsoft DNS server is. First and foremost, the DNS server in Windows NT 4.0 is an RFC compliant implementation of DNS. If there is a required feature in an RFC that is not found in the Microsoft DNS product, this would be considered a bug.

Because Microsoft DNS is an RFC compliant DNS server, it creates and uses standard DNS zone files and supports all standard resource record types. It is interoperable with other DNS servers and includes the defacto standard DNS diagnostic utility—NSLOOKUP. Microsoft DNS also has many features above and beyond those specified in the RFCs, such as dynamic update through tight integration with WINS and easy administration through the graphical administration utility called DNS Manager.

Microsoft DNS supports RFCs 1033, 1034, 1035, 1101, 1123, 1183 and 1536

With the Microsoft implementation of DNS, network administers can turn off the legacy DNS systems in favor of a Microsoft Windows NT implementation. They can remove any static entries for Microsoft-based clients in legacy DNS server zone files in favor of the dynamic WINS/DNS integration. For example, if a non Microsoft-based client wants to get to a Web page on an HTTP server that is DHCP/WINS enabled, the client can query the DNS server, the DNS server can query WINS, and the name can be resolved and returned to the client. Previous to the WINS integration, there was no way to reliably resolve the name because of the dynamic IP addressing.

The following figure shows how a non-Microsoft-based client might find a machine named "scottsu1.microsoft.com" which is running Microsoft Internet Information Server (IIS) which has a dynamically allocated address from DHCP and which has registered with WINS.


http://scottsu1.microsoft.com
Non Microsoft-based client
DNS
microsoft.com
computer name= \\scottsu1
host name= scottsu1.microsoft.com
Windows NT 4.0 Web server
DHCP Enabled
WINS

Also, as mentioned previously, the server running Microsoft DNS has a graphical user interface DNS Manager that allows for easy administration of any other Microsoft DNS server on the network via RPC (similar to the way other Windows NT administrative utilities such as Server Manager and Event Viewer work). The administrative UI also contains a zone wizard that enables someone less familiar to DNS to be successful in creating zones and zone database files. Keep in mind that you cannot administer non-Microsoft DNS servers with this administrative tool.

It's important to note that the Microsoft DNS server can easily use the database, boot, cache, rev and other files from any other DNS server implementation (that is UNIX or other Windows NT DNS implementation) as long as that DNS server is RFC compliant. All that needs to be done to port the files over to Microsoft DNS is to change the file names and locations in the boot file.

Although Microsoft DNS will support a boot file on initial installation, the Boot file is a BIND specific implementation and not a requirement of the RFCs. This feature is provided for easy migration from BIND-based DNS Servers. If the Microsoft DNS Manager UI tool is used to create and administer zone files, this "Boot From BootFile" option will be set to "Boot From Registry" and Microsoft DNS will store and use data in the Windows NT registry for locating and loading the zone file databases. A message will be written to the BOOT file that states that the information is now in the registry. To go back to booting from the boot file, the value of the "EnableRegistryBoot" key in the Windows NT registry will have to be modified manually.

The Microsoft DNS server can also be a primary or secondary to any other operating system (or other vendors Windows NT implementations). This makes it easy to begin to migrate away from a UNIX DNS-based solution to the Microsoft-based solution.

The next section will walk you through the setup of a Microsoft DNS server and explain the parameters required for a client resolver to work.

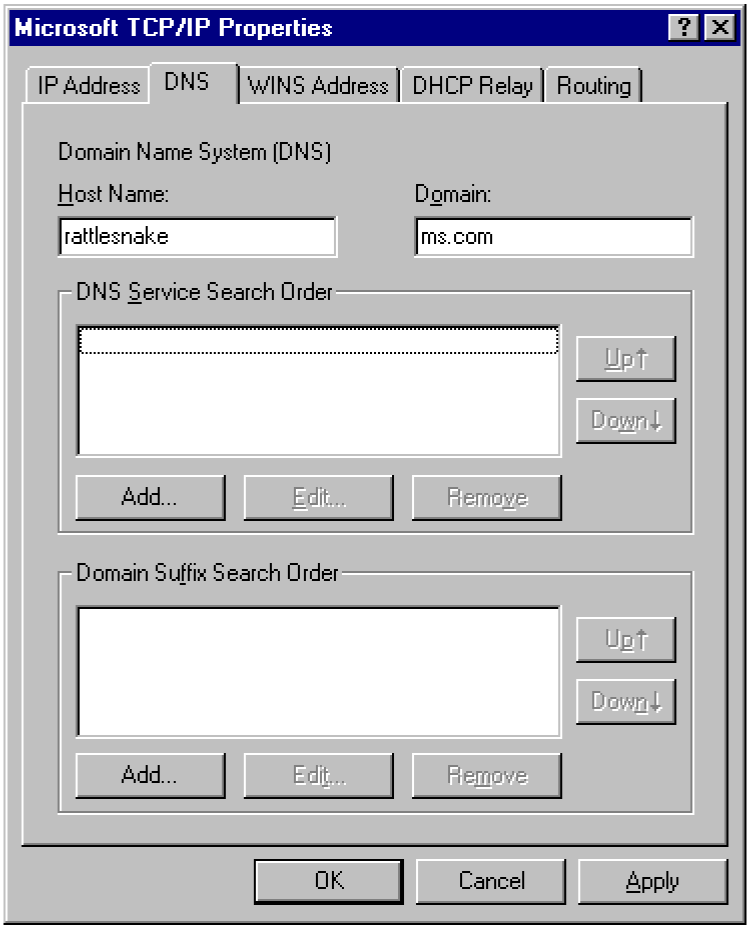
THE SERVER

Installing the DNS Service

Verify DNS Information in Windows NT

Before installing the actual Microsoft DNS service, it is important that the Windows NT Server 4.0 server's TCP/IP stack is configured correctly. In particular, the DNS section needs to be verified since the DNS service gets many of its default settings from this section during setup.

To verify this information, run Control Panel and go into Network. Click the Protocols tab. Pull up the properties dialog box for the TCP/IP Protocol and click the DNS tab. Verify that you have a correct Host Name and Domain filled in on this screen.



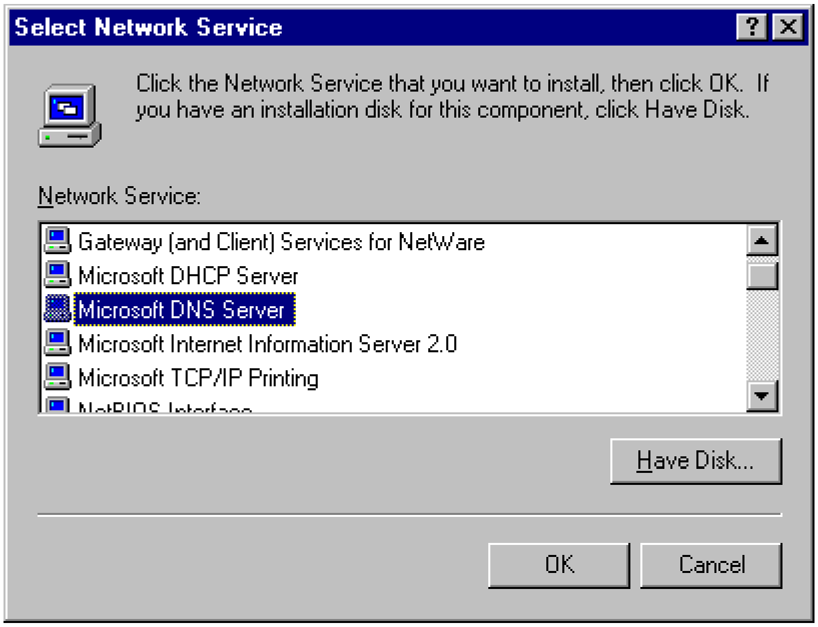
Host Name: rattlesnake

Domain: ms.com

If the domain name and host name are supplied in the network control panel setup when you create a zone, DNS will add an SOA, A, and NS record to the database. If the information is not configured then only the SOA record is created.

Installing the Microsoft DNS Service

To install the Microsoft DNS Service on a Windows NT 4.0-based system, run the Control Panel and go into Network. Click the Services tab and click the Add button. Highlight Microsoft DNS Server and click OK.



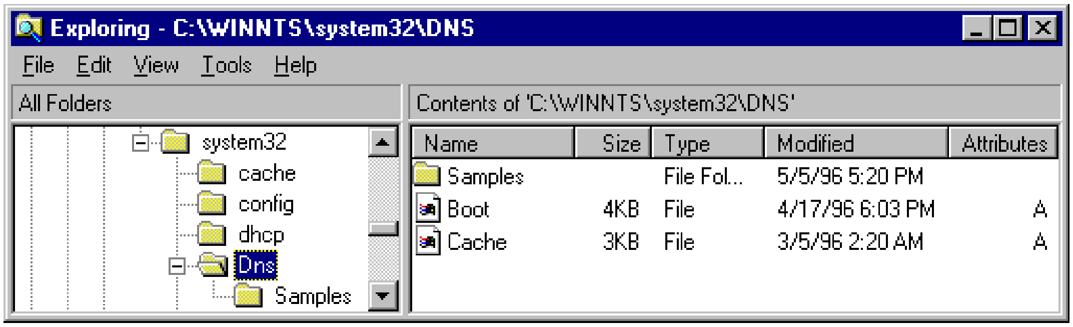
Select Network Service

Click the Network Service that you want to install, then click OK. If you have an installation disk for this component, click Have Disk.

Network Service:

- Gateway (and Client) Services for NetWare
- Microsoft DHCP Server
- Microsoft DNS Server
- Microsoft Internet Information Server 2.0
- Microsoft TCP/IP Printing
- NetBIOS Interface

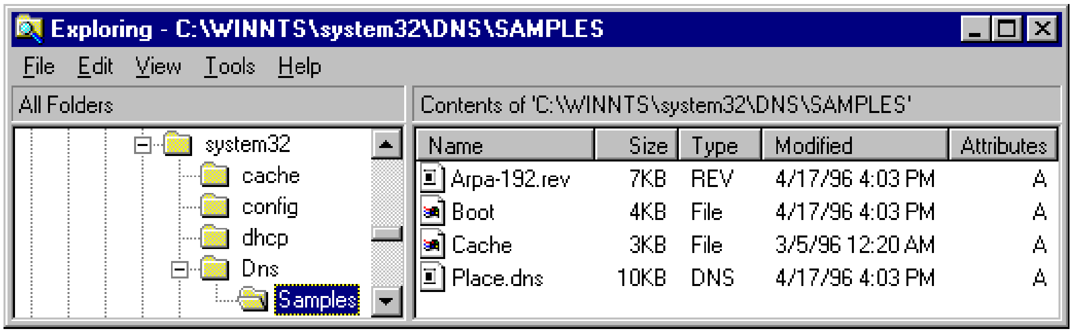
At this point, a default installation of Microsoft DNS server service will be installed on the Windows NT-based server. This installation will install files in your \<SystemRoot>\system32\Dns directory as shown. At this point you will be prompted to restart your server.



Exploring - C:\WINNTS\system32\DNS

Contents of 'C:\WINNTS\system32\DNS'

Name	Size	Type	Modified	Attributes
Samples		File Fol...	5/5/96 5:20 PM	
Boot	4KB	File	4/17/96 6:03 PM	A
Cache	3KB	File	3/5/96 2:20 AM	A



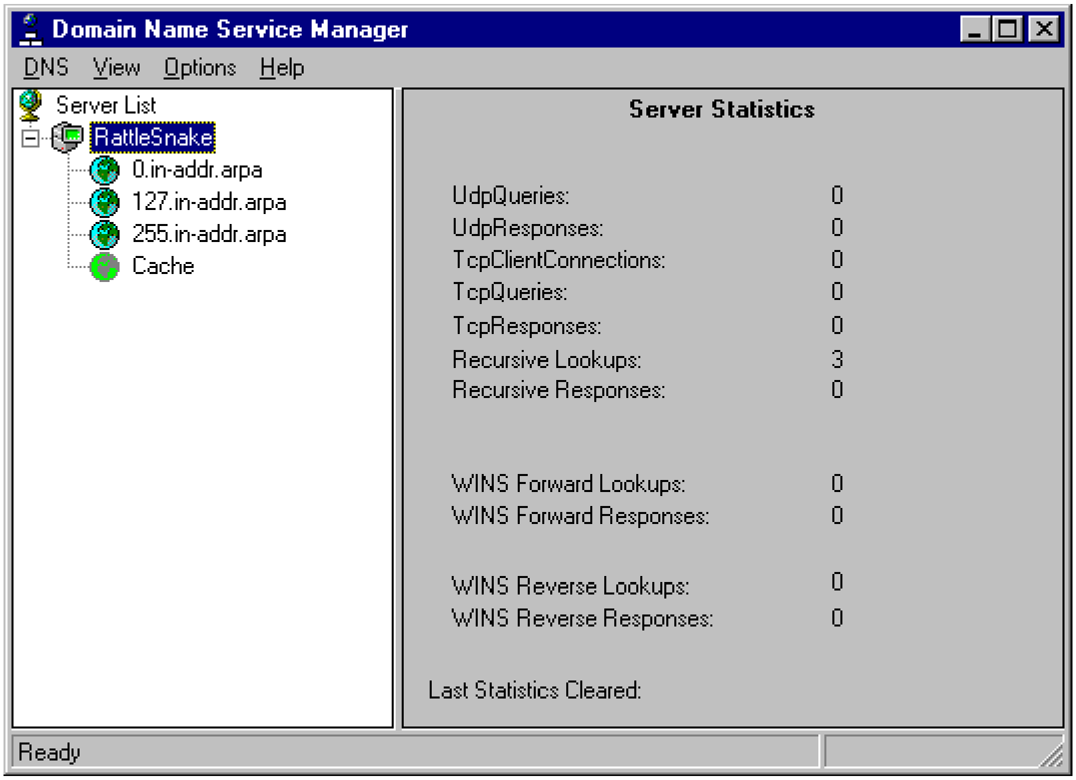
Exploring - C:\WINNTS\system32\DNS\SAMPLES

Contents of 'C:\WINNTS\system32\DNS\SAMPLES'

Name	Size	Type	Modified	Attributes
Arpa-192.rev	7KB	REV	4/17/96 4:03 PM	A
Boot	4KB	File	4/17/96 4:03 PM	A
Cache	3KB	File	3/5/96 12:20 AM	A
Place.dns	10KB	DNS	4/17/96 4:03 PM	A

Configuring DNS Domains and Zones

To configure the DNS server, run the DNS Manager under the Administrative Tools. Initially, the DNS Manager will not have any servers in its list. To add the local server, pull down the DNS menu and click New Server. Fill in the name of your local server and click OK. The server will then show up in the Server List. Double-click on the server to see the server statistics and zones that have been defined.



Server Statistics

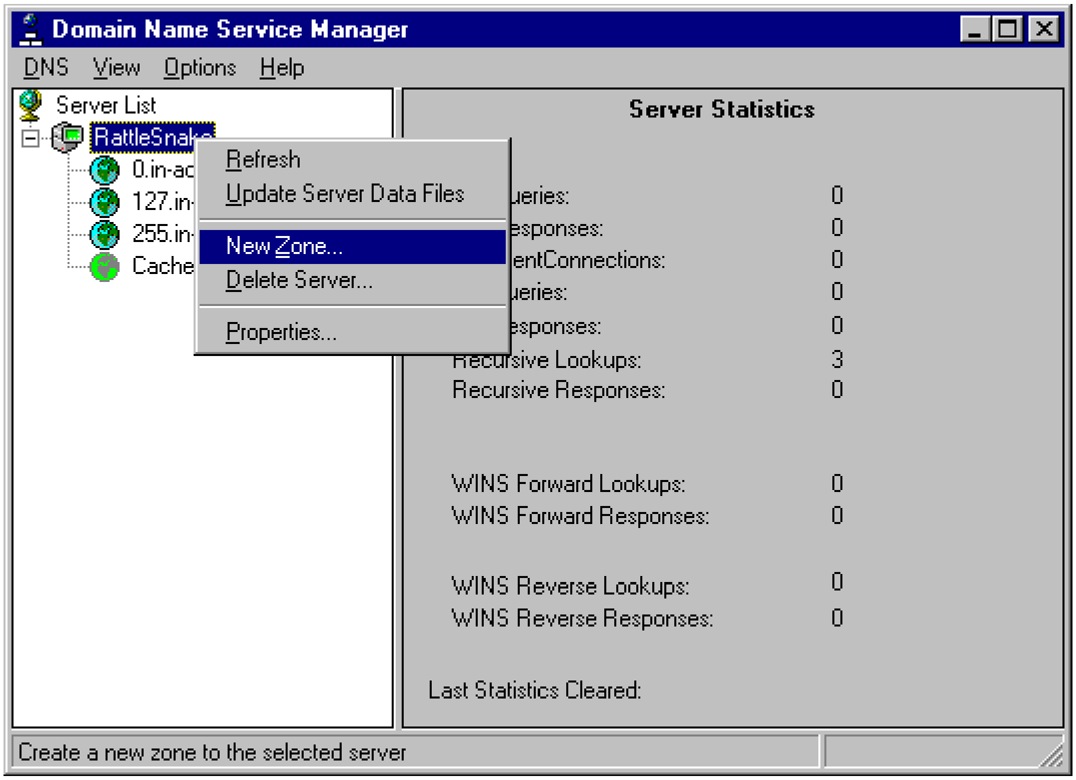
```
UdpQueries:                    0
UdpResponses:                  0
TcpClientConnections:          0
TcpQueries:                    0
TcpResponses:                  0
Recursive Lookups:             3
Recursive Responses:           0

WINS Forward Lookups:          0
WINS Forward Responses:        0

WINS Reverse Lookups:          0
WINS Reverse Responses:        0

Last Statistics Cleared:
```

Since initially, the DNS server has no information about your specific network, the DNS server service installs as a caching-only name server for the Internet. This means that DNS only contains information on the Internet root servers. For most DNS configurations, additional information must be supplied to obtain the desired operation.

The first step in configuring the DNS server is to determine the hierarchy for your DNS domains and zones. Design considerations for a DNS hierarchy are discussed in the next section of this paper. Once the domain and zone information has been determined, this information must be entered into the DNS configuration using the DNS Manager.

Since DNS information is grouped and controlled by zones, a zone must first be created. To do this, click the server name with the alternate mouse button and then click New Zone.

If you double-click on the CACHE zone, you can see all of the hosts that the DNS server has statically defined and dynamically cached due to a previous query. The CACHE will also show all of the WINS entries that have been resolved and have not elapsed their WINS specific Cache Time-out Value.



ueries: 0
esponses: 0
entConnections: 0
ueries: 0
esponses: 0
Recursive Lookups: 3
Recursive Responses: 0

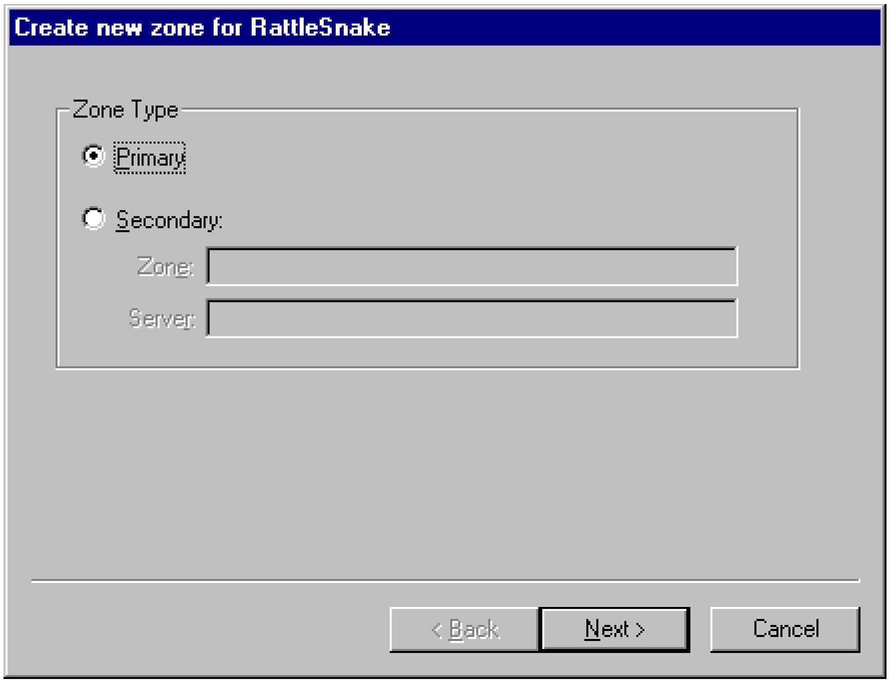
WINS Forward Lookups: 0
WINS Forward Responses: 0

WINS Reverse Lookups: 0
WINS Reverse Responses: 0

Last Statistics Cleared:

Create a new zone to the selected server

At this point, a dialog is presented which ask if the zone you are creating is a primary zone (information stored locally) or a secondary zone (information obtained from a master server via a zone transfer). If it is a primary zone, no additional information is needed at this step. If it is a secondary zone, you must also enter the zone and master server names in this screen.



Create new zone for RattleSnake

Zone Type

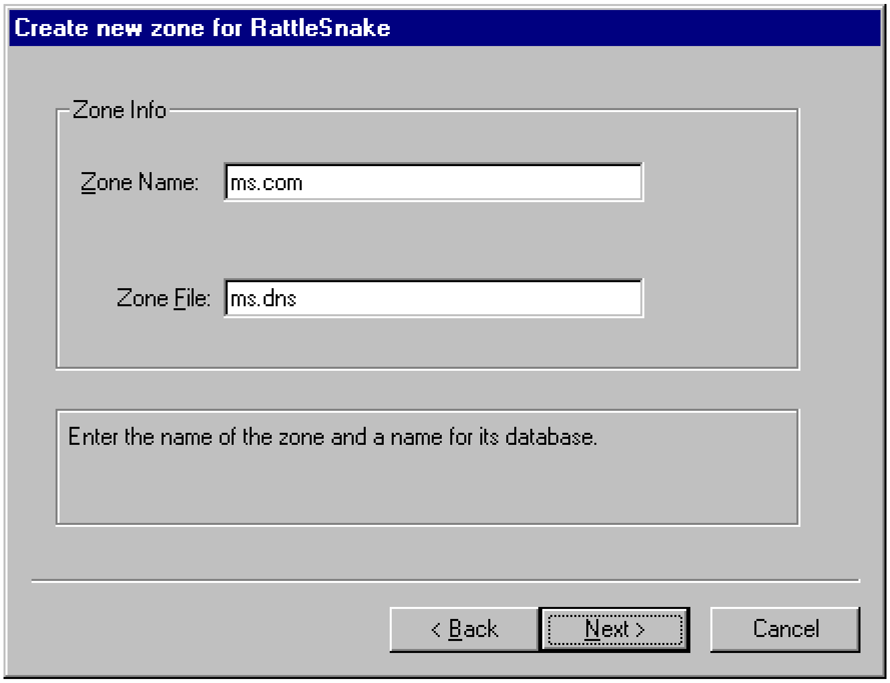
● Primary

○ Secondary:

Zone:

Server:

The next step is to fill in the zone name and zone file information for the local server. This will determine how the zone shows up in the DNS Manager and what file name it is stored under. If this is a secondary zone, this zone name should match the master server zone name. If this zone file already exists in the DNS directory, DNS will import these records automatically when this zone is created.



Create new zone for RattleSnake

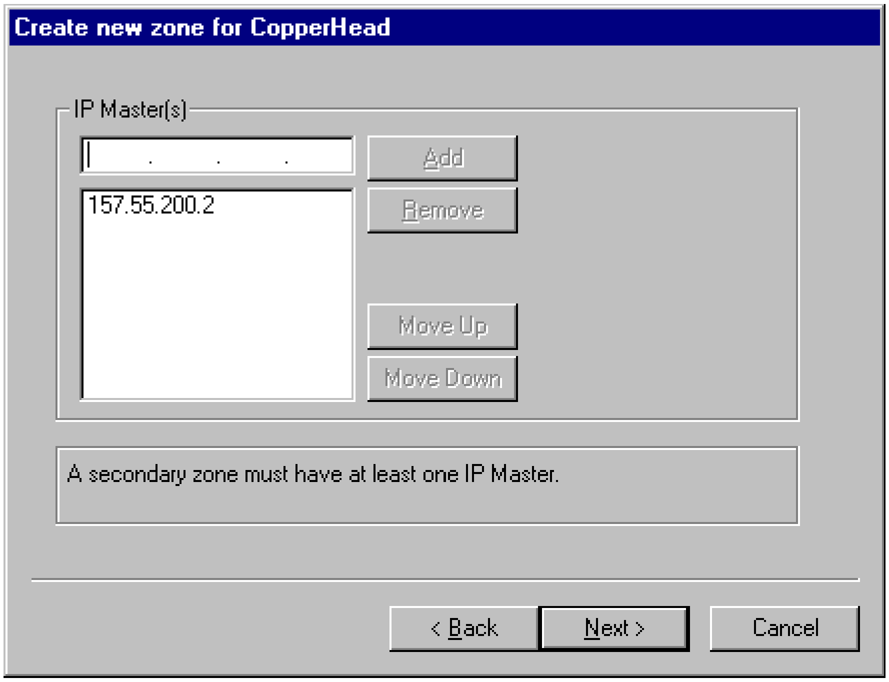
Zone Info

Zone Name: ms.com

Zone File: ms.dns

Enter the name of the zone and a name for its database.

If this is a secondary zone, you would then be asked to enter the IP address of the master name servers (the name servers with which you will do zone transfers for this zone).

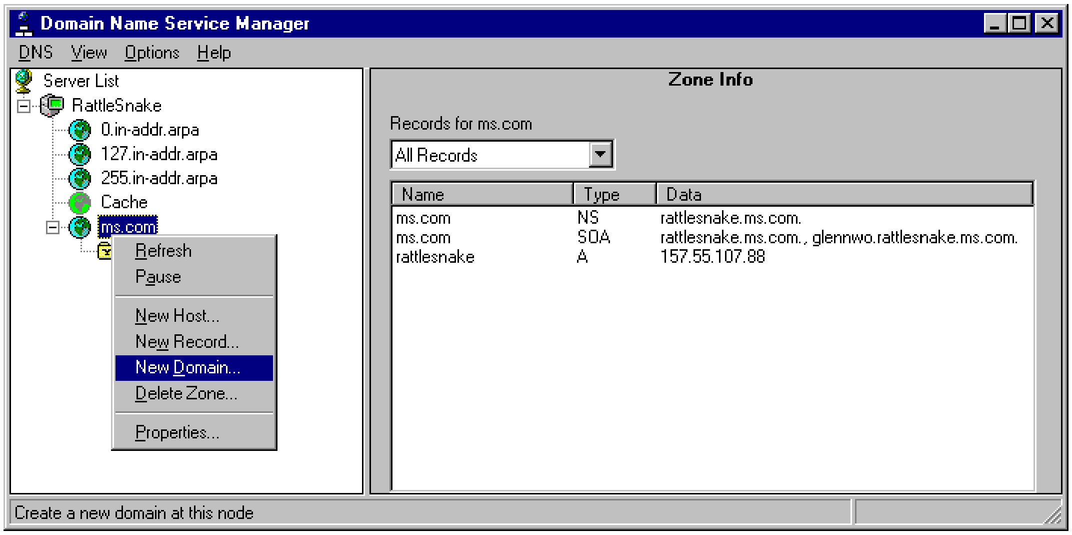

Create new zone for CopperHead

IP Master(s)

157.55.200.2

A secondary zone must have at least one IP Master.

Once all this information is entered, the zone will be added to the DNS hierarchy. If you have additional zones that need to be added, follow the same procedure for each of them. Once all zones have been added to the server, you should then add any DNS subdomains under the zones that your hierarchy might contain. To do this, click the appropriate zone with the alternate mouse button and click New Domain.


Domain Name Service Manager

Records for ms.com

All Records

Name	Type	Data
ms.com	NS	rattlesnake.ms.com.
ms.com	SOA	rattlesnake.ms.com., glennwo.rattlesnake.ms.com.
rattlesnake	A	157.55.107.88

Create a new domain at this node

Enter the name of the new subdomain in the dialog box and click OK. If multiple levels of subdomains are needed, create each successive sub-domain by clicking on the immediate parent with the alternate mouse button, clicking on New Domain, and entering the name of the new subdomain.

This process can also be used for adding new host or resource records at any domain location within the DNS hierarchy.

Integration with WINS Lookup

The WINS Record

Although DNS may seem similar to Windows Internet Naming Service (WINS), there are a couple of major differences. DNS is a static database of IP addresses for name-to-address mapping which must be manually updated by an administrator. Also, DNS has the concept of hierarchy which allows the administration and replication of the database to be broken up into "zones." WINS, on the other hand, allows machines to dynamically register their name-to-address mappings and therefore requires far less administration. WINS is also a flat name space, without the concept of hierarchy and requires each WINS server to maintain a complete database of entries through replication.

The Microsoft DNS server works hand in hand with the Microsoft WINS server and provides a great deal of interoperability. To provide this interoperability, a new record was defined as part of the zone database file. The WINS record is specific to Windows NT and may be attached **only** to the zone root domain. The presence of a WINS record instructs the name server to use WINS to lookup any requests for hosts in the zone root which do not have static addresses in the IP database. This functionality is particularly useful for UNIX-based clients that need to contact DHCP/WINS enabled clients via IP.

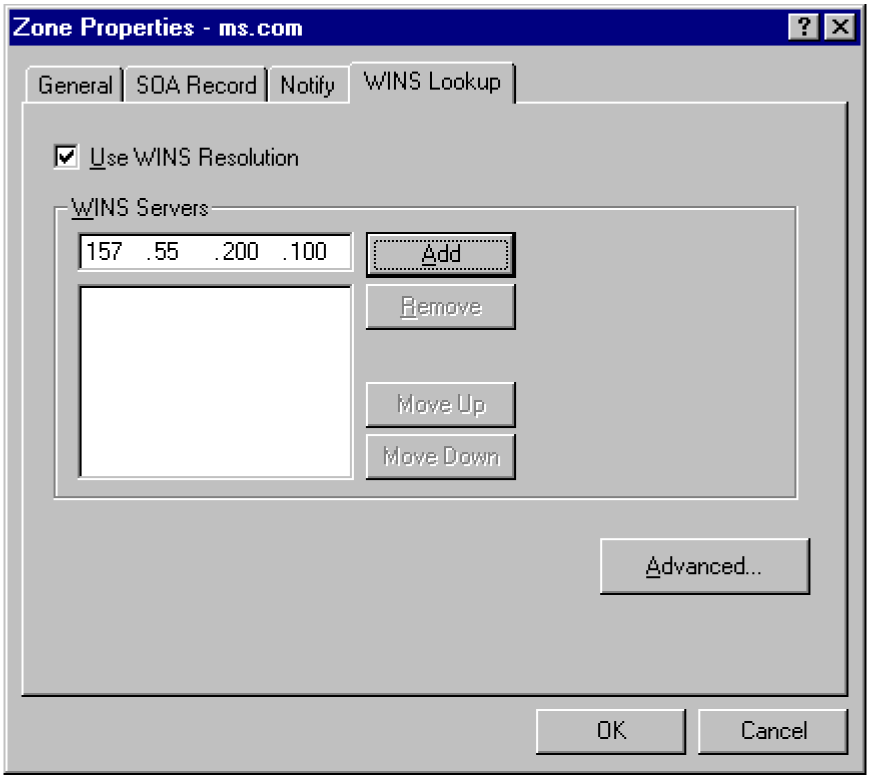
<domain> IN WINS <IP address of WINS server>

Example:

```
@ IN WINS 157.55.200.81
```

Enabling WINS Lookup

WINS Lookup can be enabled for a zone through the DNS Manager instead of requiring manual entry of the WINS record. This is accomplished by clicking the zone with the alternate mouse button and clicking properties. Then click the WINS Lookup tab. Check the Use WINS Resolution checkbox and fill in the IP address of the WINS Server that you wish to use and click Add. Multiple WINS server addresses can be entered.



☑ Use WINS Resolution

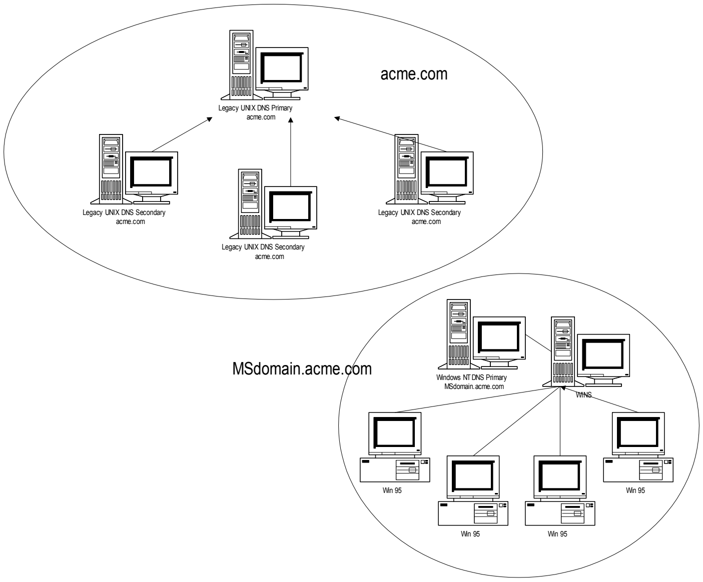
WINS Servers: 157 .55 .200 .100

You probably only need to use WINS lookup, if you have non-Microsoft-based TCP/IP clients that need to resolve Host Name to IP addresses. For example, if there is a need in your organization to be able to use FTP or HTTP on your servers running Windows NT from non-Microsoft-based (that is UNIX) clients.

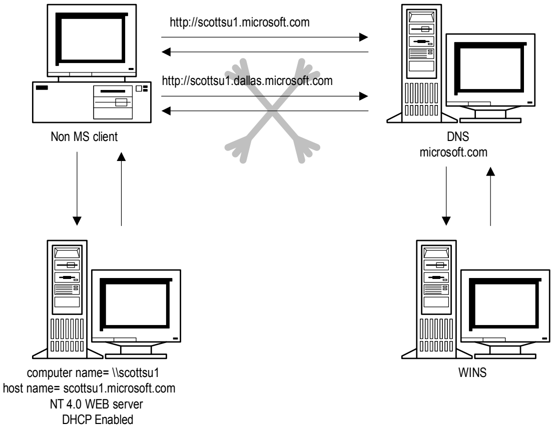
Warning

If you have a zone configured to do WINS lookup, then all DNS servers that are authoritative for that zone need to be able to do WINS lookup or you will have intermittent behavior.

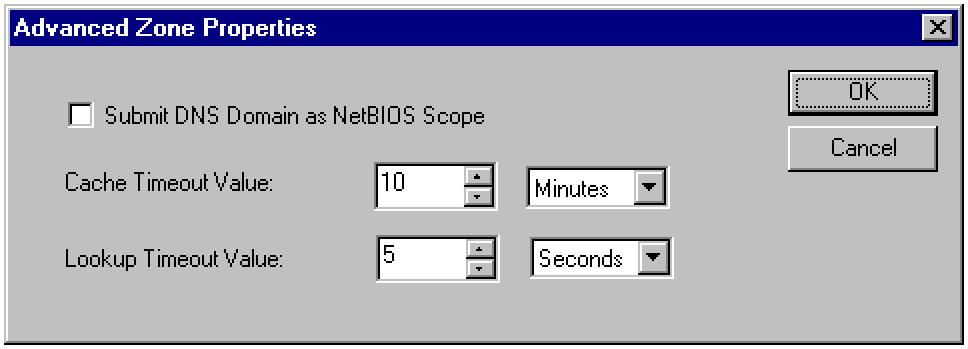
In order to easily add the Microsoft WINS / DNS lookup to a legacy DNS architecture, simply create a new DNS subdomain in your enterprise and have the Windows NT-based primary and secondary servers enabled to do WINS lookup in this domain. For example, in the following figure there is an **acme.com** domain and a **Msdomain.acme.com** domain. All of the Microsoft-based clients register with the WINS server in the **Msdomain.acme.com** domain.



WINS lookup is done on a DNS-zone basis. So a query to a DNS server for **scottsu1.microsoft.com** would go to the WINS server if the DNS that had the WINS lookup record was authoritative for zone **microsoft.com**, but a query for **scottsu1.dallas.microsoft.com** would not go to that same WINS server. This is shown in the following figure.


http://scottsu1.microsoft.com
http://scottsu1.dallas.microsoft.com
Non MS client
DNS
microsoft.com
computer name= \\scottsu1
host name= scottsu1.microsoft.com
NT 4.0 WEB server
DHCP Enabled
WINS

If you are using a WINS record the Time to Live (TTL) in the SOA record is not the default for WINS as well, the WINS TTL is configured via the "Advanced Zone Properties" dialog box (under the WINS lookup tab) when your configuring the zone. When an IP address / Host name gets resolved via WINS the address is cached for the WINS Cache Time-out Value. If this address is ever forwarded to another DNS the WINS Cache Time-out Value TTL is what is sent.



Advanced Zone Properties

□ Submit DNS Domain as NetBIOS Scope

Cache Timeout Value: 10 Minutes

Lookup Timeout Value: 5 Seconds

If your data doesn't change much then you will want to set your TTL high. Keep in mind that you can set the TTL on individual records as well.

If the TTL on an individual RR's address is lower or higher than the TTL in the SOA record the individuals TTL takes precedence.

WINS and Reverse Lookup

The WINS Reverse Lookup Record

Although WINS was not constructed to provide reverse lookup capabilities, this functionality can still be accomplished for DHCP/WINS enabled clients when using Microsoft's DNS server. The presence of a WINS-R record at the zone root instructs the name server to use a NetBIOS node adapter status lookup for any reverse lookup requests for IP addresses in the zone root which are not statically defined with PTR records.

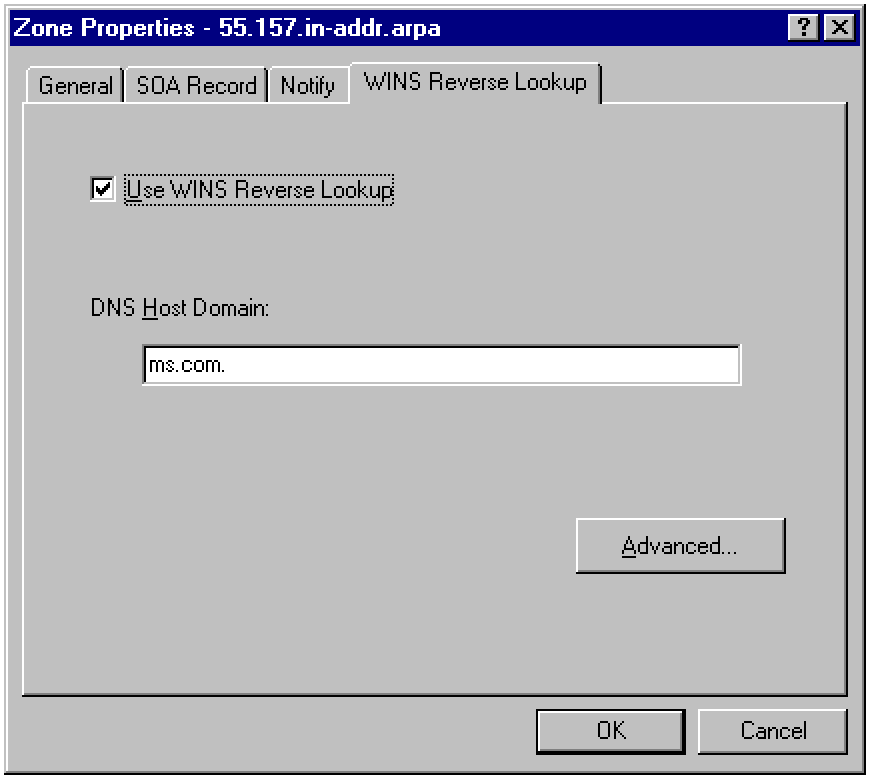
<domain> IN WINS-R <domain to append to returned NetBIOS names>

Example:

```
@  IN WINS-R  microsoft.com.
```

Enabling WINS Reverse Lookup

WINS Reverse Lookup can be enabled for a zone through the DNS Manager instead of requiring manual entry of the WINS-R record. This is accomplished by clicking on the appropriate in-addr.arpa zone with the alternate mouse button and clicking on properties. Then click on the WINS Reverse Lookup tab. Check the Use WINS Reverse Lookup checkbox and fill in the DNS Host Domain to be appended to the NetBIOS name before returning the response to the resolver.



☑ Use WINS Reverse Lookup

DNS Host Domain: ms.com.

The Microsoft DNS server can be configured to NOT send WINS records to secondary servers. This is necessary if you have a mixture of Microsoft- and UNIX-based DNS servers since UNIX-based DNS servers do not have the capability to do WINS lookup.

Other Things to Know

The Microsoft DNS server supports "Notify"

The DNS NOTIFY transaction allows master servers to inform secondary servers when the zone has changed (notify is set on the master server on a per zone basis)—an interrupt as opposed to poll model. This should reduce propagation delay while not unduly increasing the master server's load.

The use of this feature depends on how often the master server's data will change and how slow the link is between the secondary and the primary. If the master server's data changes a great deal, and it is important that the data on the secondaries is very accurate, and the link between the site that houses the primary and the secondary is not negatively impacted by the zone transfer, then it may be a good idea to use this feature.

It's also a good idea to use this feature if the master server's data does not change very often. If this is the case then you can make the refresh time on the master very long and a notification will be sent to the secondaries when they need to update their zone database.

Microsoft DNS supports "Round Robin"

Round Robin is a technique used as a form of load balancing between servers. You can read more about load balancing in RFC 1794. Here's an example that shows how this works:

On the DNS server you could have 2 address entries for the same host, such as the following:

copperhead.glennwo.scottsu.com A 157.55.106.193

copperhead.glennwo.scottsu.com A 157.55.107.88

If you make a query via some mechanism such as "**PING copperhead.glennwo.scottsu.com**," the DNS server will send both IP addresses back, but typically the client will always use the first one. The next time the DNS server receives a query for this host the order of the list is changed in a round robin fashion (the address that was first in the previous list will be last in the new list), hence when the client resolver chooses the first IP address in the list, it chooses a different server. This is typically used for load balancing.

Here is a trace of a query that shows how the feature works. In frame 1 a query is sent to get the IP address for host "**copperhead.glennwo.scottsu.com**". Since there were 2 entries in the database, both were returned. The client resolver, in most implementations (including the Microsoft resolver) uses the first entry and discards the other.

```
1  SCOTTSU-7  Xircom40417A DNS  0x1:Std Qry for copperhead.glennwo.scottsu.com

DNS: 0x1:Std Qry for copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.

2  Xircom40417A SCOTTSU-7  DNS  0x1:Std Qry Resp. for copperhead.glennwo.scottsu.com

DNS: 0x1:Std Qry Resp. for copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.
....
DNS: Answer section: copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.(2 records present)
   DNS: Resource Record: copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.
    DNS: Resource Name: copperhead.glennwo.scottsu.com
    DNS: Resource Type = Host Address
    DNS: Resource Class = Internet address class
    DNS: Time To Live = 0 (0x0)
    DNS: Resource Data Length = 4 (0x4)
    DNS: IP address = 157.55.107.88
   DNS: Resource Record: copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.
    DNS: Resource Name: copperhead.glennwo.scottsu.com
    DNS: Resource Type = Host Address
    DNS: Resource Class = Internet address class
    DNS: Time To Live = 3600 (0xE10)
    DNS: Resource Data Length = 4 (0x4)
    DNS: IP address = 157.55.106.193
```

The next time someone makes a query for the host **copperhead.glennwo.scottsu.com** the first IP address in the list will be different.

```
DNS: Answer section: copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.(2 records present)
   DNS: Resource Record: copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.
    DNS: Resource Name: copperhead.glennwo.scottsu.com
    DNS: Resource Type = Host Address
    DNS: Resource Class = Internet address class
    DNS: Time To Live = 3600 (0xE10)
    DNS: Resource Data Length = 4 (0x4)
    DNS: IP address = 157.55.106.193
   DNS: Resource Record: copperhead.glennwo.scottsu.com of type Host Addr on class INET addr.
    DNS: Resource Name: copperhead.glennwo.scottsu.com
    DNS: Resource Type = Host Address
    DNS: Resource Class = Internet address class
```

```
DNS: Time To Live = 0 (0x0)
DNS: Resource Data Length = 4 (0x4)
DNS: IP address = 157.55.107.88
```

NetBIOS Scope

The main thing to remember about DNS support for NetBIOS scope is ***DON'T USE IT*** unless your network already uses NetBIOS scope. In a scope configuration, all hosts are assigned a scope ID, and they register this scope ID—along with their NetBIOS name with WINS. If DNS is configured to use scope, when it queries WINS, in addition to passing the DNS host name as the single-part NetBIOS name, it also passes the DNS domain as the NetBIOS scope ID.

A very important point to remember about scope is that when scope is used, NetBIOS applications can not see hosts that are in another scope! One such NetBIOS application happens to be the Windows NT Netlogon service which is responsible for trust relationships between Windows NT domains. This means, for example, that if scope is used a user cannot logon in a domain whose domain controllers have a different scope than that of the users station. It also means that a user cannot access resources on a network server in a domain whose scope is different than that of the users station.

When does Microsoft DNS read and write to the zone files?

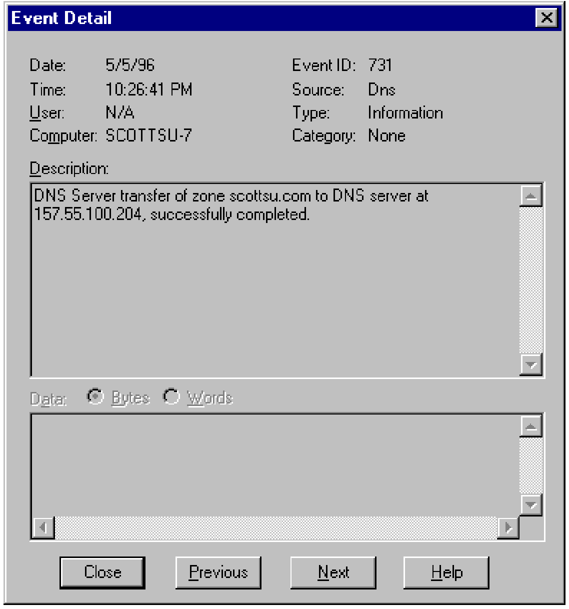
On startup of the DNS service, the zone files are read from disk. As changes are made through the DNS Manager, the service periodically flushes these changes to disk. In general, you should always use the DNS Manager utility to add, delete, or modify resource records in the database, however if you feel a need to modify the files with an editor, you should only make changes to these files if the DNS server service is stopped. By default, the files are located in \%SystemRoot%\system32\Dns.

If you use the DNS administrative utility, and choose the DNS | "Update Server Data Files" menu item, the files will also be flushed from memory to disk.

The Eventlog

The DNS server will report events into the Eventlog. This allows an administrator to determine when a zone transfer has been completed, or when the DNS service has started, stopped or an error has occurred.

THE RESOLVER (CLIENT)



Date: 5/5/96
Time: 10:26:41 PM
User: N/A
Computer: SCOTTSU-7
Event ID: 731
Source: Dns
Type: Information
Category: None

Description:
DNS Server transfer of zone scottsu.com to DNS server at 157.55.100.204, successfully completed.

Setup

The exact procedure needed to setup individual client machines will vary depending on the operating system and TCP/IP software being used. The specific procedures for the clients will not be covered in this paper, however, the general setup parameters required and their use will be discussed.

The Host Name

The host name for the client can be any combination of the letters A through Z, the numerals 0 through 9, and the hyphen (-). By default, in Microsoft networking environments, this value is the NetBIOS computer name, but you can assign a different host name without affecting the NetBIOS computer operation if desired. If WINS Lookup is used with Microsoft DNS, this value should match your NetBIOS computer name for consistency.

Some characters that can be used in NetBIOS names, especially the underscore and period, cannot be used in DNS host names.

The Domain Name

The domain name is used with the host name to create a fully qualified domain name (FQDN) for the computer. The FQDN is the host name followed by a period (.), followed by the domain name. For example, this could be wkstn1.microsoft.com, where wkstn1 is the host name and microsoft.com is the domain name. During DNS queries, the fully qualified domain name is used.

The DNS Servers

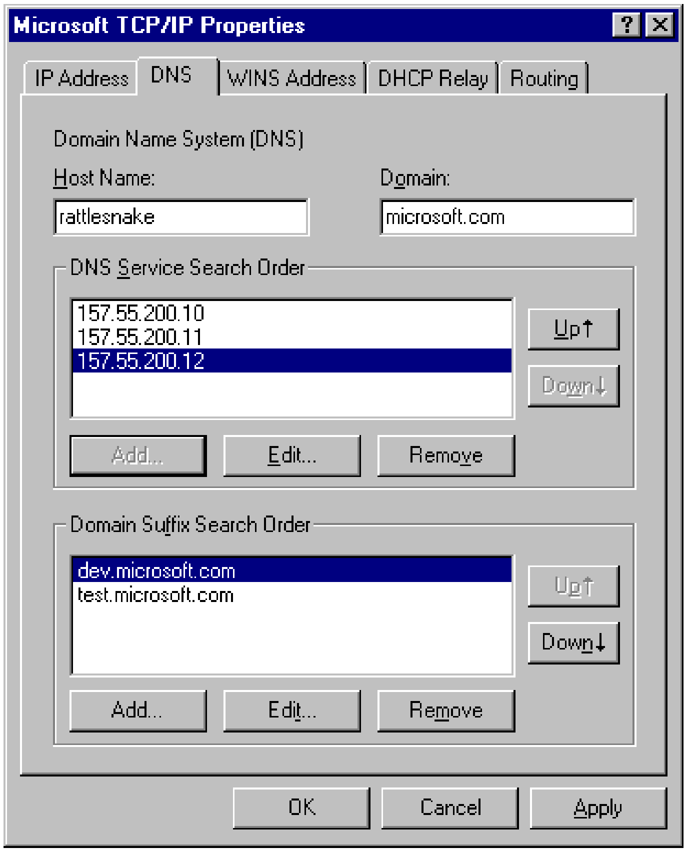
Many operating systems allow you to specify several DNS servers in their configurations. When this capability is provided, a priority order is also provided so that a preferred server can be identified. For a given DNS query, the system attempts to get DNS information from the first IP address in the list. If no response is received, it goes to the next server in the list, and so on.

In most systems, if you have two DNS servers listed, the system only checks the second server if no response is received from the first server. If the system attempts to check a host name with the first server and receives a message that the host name is not recognized, the system does not try the second DNS server.

The Domain Suffix Search Order

In some systems, a Domain Suffix Search Order capability is provided. The Domain Suffix Search Order specifies the DNS domain suffixes to be appended to the host names during name resolution. When attempting to resolve a fully qualified domain name (FQDN) from a host only name, the system will first append the local domain name. If this is not successful, the system will use the Domain Suffix list to create additional FQDNs in the order listed and query DNS servers for each.

An example client configuration from a machine running Windows NT 4.0 is illustrated below.



Host Name: rattlesnake

Domain: microsoft.com

DNS Service Search Order:
157.55.200.10
157.55.200.11
157.55.200.12

Domain Suffix Search Order:
dev.microsoft.com
test.microsoft.com

Name Resolution

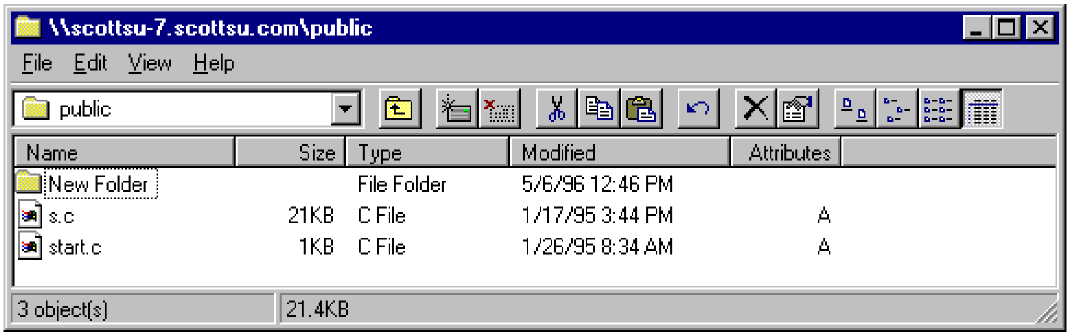
Windows NT TCP/IP 3.5*x* systems may use several methods for locating NetBIOS resources:

- NetBIOS name cache
- NetBIOS name server
- IP subnet broadcasts
- Static *LMHOSTS* files
- Static *HOSTS* files
- DNS servers

Earlier implementations used only cache, broadcasts, and LMHOSTS files; however, in version 3.5, a NetBIOS name server (WINS) was added, and modifications were made to allow NetBIOS applications to query the DNS name-space by appending configurable domain suffixes to a NetBIOS name. However, the DNS name resolution was always done last after the NetBIOS cache, LMHOSTS file (if enabled), WINS (if enabled), and broadcast were tried. With Windows NT 4.0 the DNS name resolution will be tried first if the name is greater than 15 characters (maximum length of a NetBIOS computer name on Windows NT). The Host name can now be used in any of the Windows NT utilities (such as Microsoft Explorer seen below) that previously supported NetBIOS computer names. This functionality will also be made available in the Windows® 95 operating system in the future.

Remember that the \%SystemRoot%\system32\drivers\etc\HOSTS file can still be used for Host name resolution. However, DNS will be queried before the HOSTS file is parsed.

If the DNS Host name (larger than 16 characters) is passed to a utility and there are 2 transports loaded on a workstation (TCP and NetBEUI), only the TCP transport will try to setup the session. If the Host name is not used and the NetBIOS name is used then all transports will be tried.



Name	Size	Type	Modified	Attributes
New Folder		File Folder	5/6/96 12:46 PM	
s.c	21KB	C File	1/17/95 3:44 PM	A
start.c	1KB	C File	1/26/95 8:34 AM	A

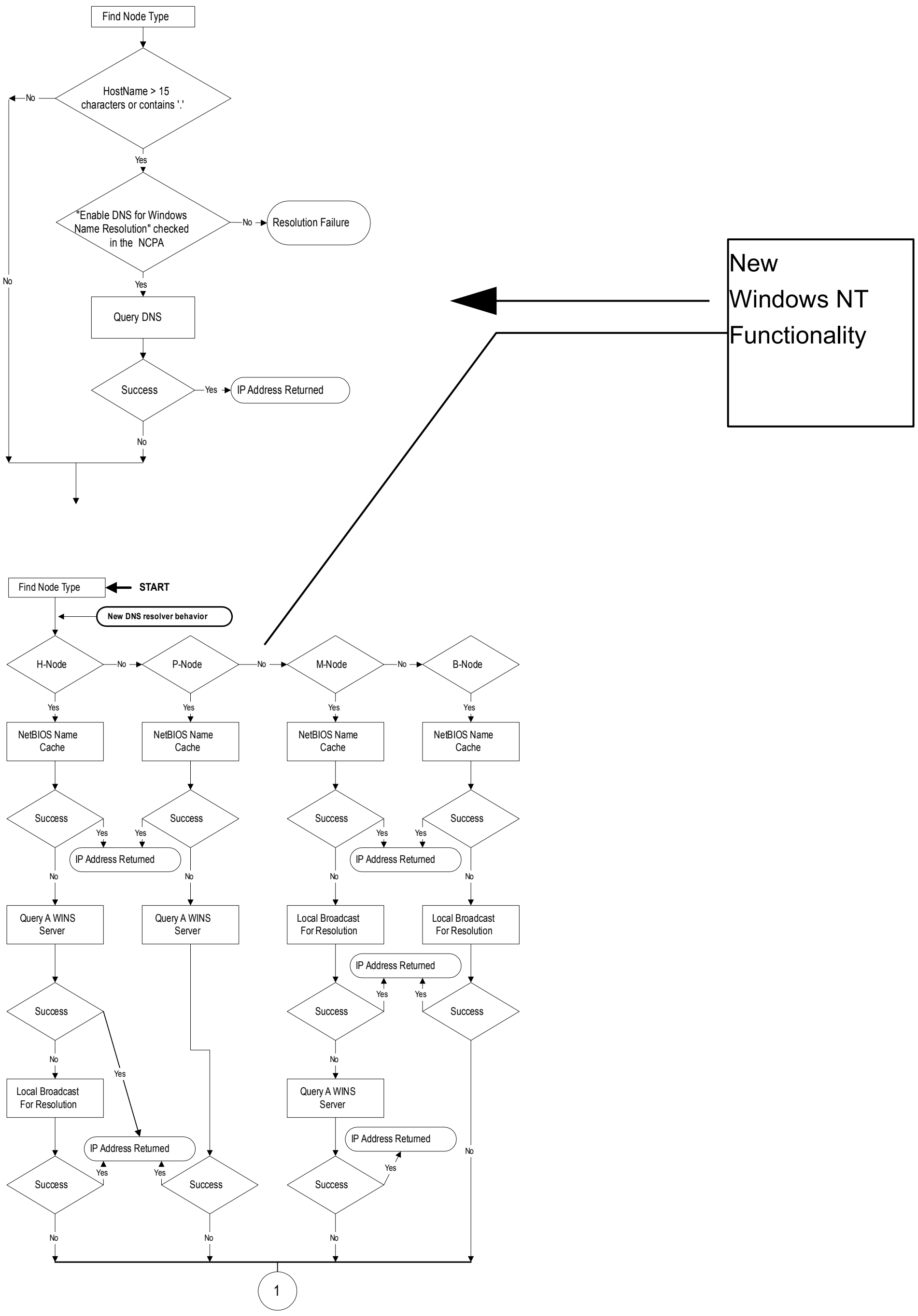
3 object(s) 21.4KB

In this example I could have also used:
\\157.55.100.204\public instead of **\\scottsu-7.scottsu.com\public**.

For more information on NetBIOS name resolution you can refer to the White Paper titled, “Microsoft Windows NT Server 3.5—Dynamic Host Configuration Protocol and Windows Internet Naming Service,” part number 098-56544.

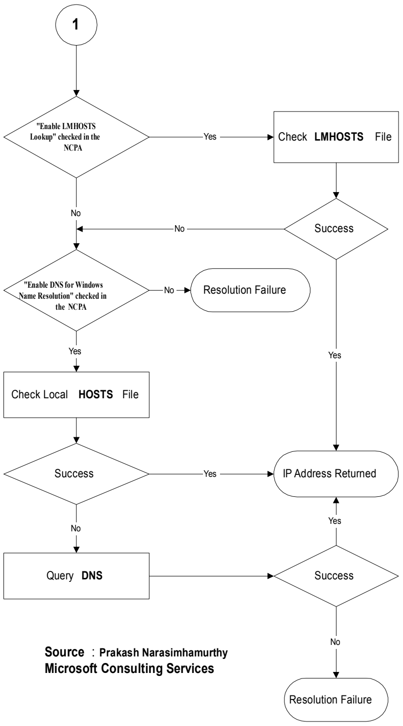
On a Windows NT-based workstation, if a Host name is not found via DNS resolution, the name will be passed to NetBT (NetBIOS over TCPIP) for resolution via WINS, LMHOSTS file or broadcast. This might be seen as a feature for intranets that have many Web or FTP servers and want to get resolution via WINS for host queries.

Prakash Narasimhamurthy of Microsoft Consulting Services provided the flow charts shown in the following figures to illustrate name resolution for the various node types.



New Windows NT Functionality

THE BIG PICTURE



Source : **Prakash Narasimhamurthy**
Microsoft Consulting Services

Nslookup

Nslookup will probably be one of your primary diagnostic tools when debugging a Domain Name System (DNS) architecture. It can be used to display any resource record on any DNS server including UNIX-based DNS implementations. NSLookup from most other DNS implementations should also work against the Microsoft-based DNS server. This is a command line utility and has the following syntax.

nslookup [[-option ...] [computer-to-find]] | - [server]

Modes

Nslookup has two modes: interactive and non-interactive.

If you only need to look up a single piece of data, use non-interactive mode. For the first argument, type the name or IP address of the computer to be looked up. For the second argument, type the name or IP address of a DNS name server. If you omit the second argument, the default DNS name server will be used.

If you need to look up more than one piece of data, you can use interactive mode. Type a hyphen for the first argument and the name or IP address of a DNS name server for the second argument, or omit both arguments and the default DNS name server will be used.

Parameters

-option . ..

You can specify one or more nslookup commands as a command-line option. For a list of commands, see “Nslookup Commands” in the WINNT.HLP file supplied with Windows NT 4.0. Each option consists of a hyphen (-) followed immediately by the command name and, in some cases, an equal sign (=), and then a value. For example, to change the default query type to host (computer) information and the initial timeout to 10 seconds, you would type:

```
nslookup -querytype=hinfo -timeout=10
```

The command line length must be less than 256 characters.

computer-to-find

This instruct Nslookup to Look up information for computer-to-find using the current default server or using server if specified. If computer-to-find is an IP address and the query type is A or PTR, the name of the computer is returned. If computer-to-find is a name and does not have a trailing period, the default DNS domain name is appended to the name. (This behavior depends on the state of the set options: domains, srchlist, defname, and search.) To look up a computer not in the current DNS domain, append a period to the name.

If you type a hyphen (-) instead of computer-to-find, the command prompt changes to Nslookup interactive mode.

server

Use this server as the DNS name server. If you omit server, the default DNS name server is used.

For more information on the Windows NT Nslookup utility, check the WIN-NT.HLP file supplied with Windows NT 4.0.

Weaknesses of Current Implementation

Inefficient Reverse Lookup (NbtStat)

DNS and WINS integration does not enable efficient DNS “reverse lookup.” Both the DNS server and the host being located are involved when a reverse lookup is requested for a WINS client since the DNS server does a NetBIOS node adapter status lookup to resolve this IP address to a name. If DNS dynamic update was a released RFC and was used instead of WINS integration, reverse lookup would be more efficient and involve only the DNS servers.

Non Microsoft-Based Hosts don’t Register with WINS

Non-Microsoft-based hosts do not register in WINS and therefore can not “dynamically register” in DNS. If true DNS dynamic update was supported by Microsoft’s DNS server, non-Microsoft-based hosts, that supported dynamic DNS, would be able to register in the Microsoft DNS and Microsoft-based hosts could dynamically register in non-Microsoft DNS that supported the DNS dynamic update standard.

WINS Registration not secure

WINS registration is not secure and it would be difficult to make it secure. The IETF is working to complete a standard for adding security to DNS dynamic update. The DNS security standard in not scheduled to enter IETF standards track until the fourth quarter of 1996.

DNS and WINS Integration is not an RFC

Though DNS dynamic update is not yet an IETF standard, some implementations already exist. Microsoft has chosen to wait until this is standardized in order to minimize impact on our customers if changes were made before the standard was released.

While WINS is actually based on IETF standards (RFC 1001, 1002), its NetBIOS “roots” have resulted in very limited acceptance by IETF and the UNIX community. The DNS and WINS integration is not standards based and is therefore not accepted.

Resolver APIs are not exposed

There is currently no way to programmatically query the DNS database other than gethostbyname(). It is currently not possible to do things such as programmatically query the DNS for an MX record.

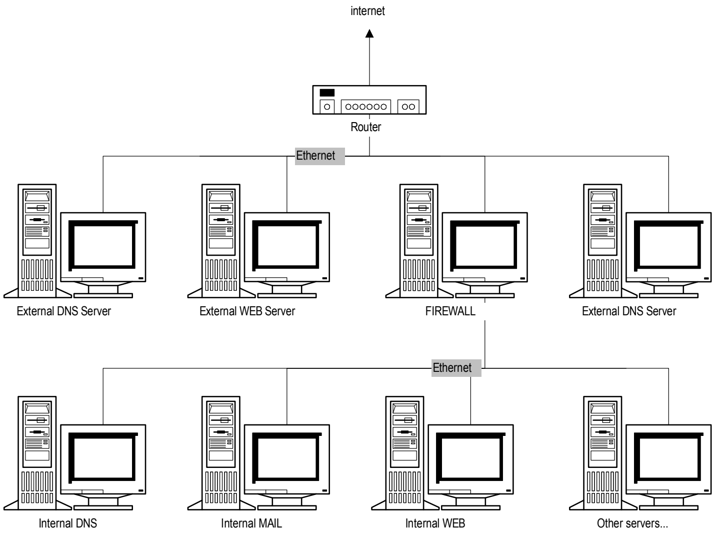
DNS is going to be an important part of the Enhanced Directory Services architecture in the next major revision of Windows NT Server, so it is important to design a DNS architecture that allows for a smooth transition/migration.

Security Considerations

Many corporations today are wanting to connect their internal networks to the Internet to provide access to these external resources to employees who need them to accomplish their assigned tasks. Although this is a very important capability, it is one that must be well planned to avoid possible data security risk from exposing the internal network to users outside the corporation. One common way to provide this protection is with the use of a firewall. In Internet terms, a firewall is a system or device which provides network security by allowing only certain authorized activities to be accomplished between internal networks and the Internet.

A firewall system can be very simple or extremely complex depending on the particular requirements of the individual company. This paper is not designed to provide an exhaustive description of firewall design but we will briefly discuss how the Microsoft DNS server can be used in a firewall environment.

Here is a typical Internet connectivity setup for a company using a dual-homed firewall (that is proxy).



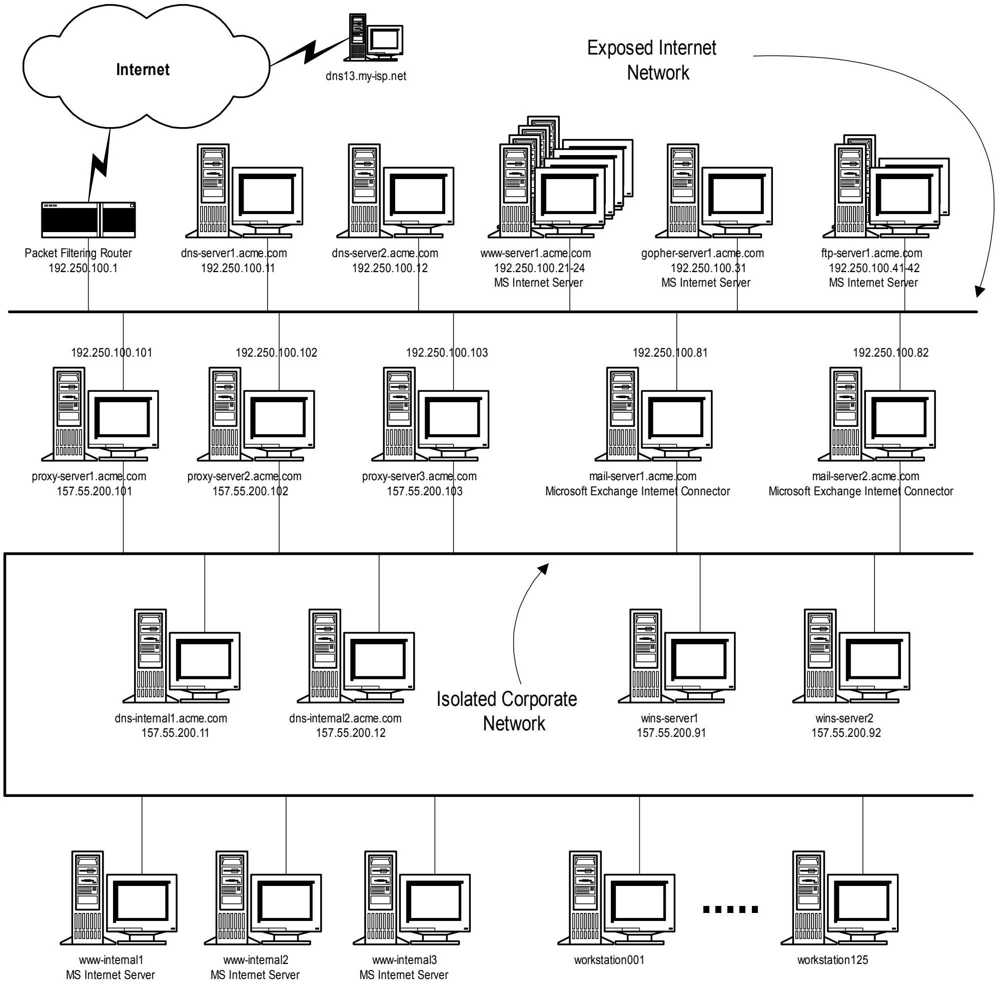
The firewall protects access from the outside while allowing clients on the internal network access to Internet resources. This design also allows for external WWW and FTP servers. These external servers must be closely monitored and secured as much as possible since they are directly on the Internet network with no “firewall” type access control. The router can provide some security by providing packet filtering, if desired, to limit the type of traffic allowed. Access to the internal network is controlled by the firewall.

The internal Web server can be setup exactly like the external server except that security concerns are limited to the access controls necessary for company employees. Typically, only those responsible for Web development will actually logon to the internal servers to change the files located there, although everyone would typically be given the right to view the Web pages using a browser.

The Domain Name System for the external and internal networks are usually entirely isolated from one another. This prevents external clients from obtaining the names and addresses for internal resources located inside the firewall. The only thing an outside user will see is the IP addresses of the servers which are providing external services (that is FTP, www, mail, and so on).

Typical Internet Connectivity Design

With these security concerns now established, let’s look at a typical Internet connectivity design and specifically at the way the DNS servers would be configured. Below is a diagram of a typical large company which has several internal and external resources which require DNS services.



There are several items which should be noted about this example network before we begin discussing the DNS configurations. These items are:

- Acme has a primary and secondary DNS server for their external services which they maintain (that is dns-server1.acme.com and dns-server2.acme.com). There is also a secondary DNS server for their **acme.com** zone which is maintained by their Internet Service Provider for redundancy (that is dns13.my-isp.net). These name servers are authoritative for domains **acme.com** and 100.250.192.in-addr.arpa.
- Some external services are being maintained on multiple mirrored servers.
- Multiple mail servers are used to supply SMTP mail connectivity to **acme.com**.
- This network design uses packet filtering at the router as well as using multi-homed proxy servers to implement security.
- The internal network is using WINS for registration/lookup of Microsoft-based clients and servers.
- Some intranet services (that is Web Services) are being provided on internal servers running Windows NT Server.
- DNS services on the internal network are isolated from the Internet DNS ser-

vices.

- The internal network is displayed as a simple non-routed environment for clarity.

With these details in mind, let's look at how this would affect the DNS architecture. We will go through the thought process of configuring the external DNS server first and then we will discuss the internal DNS server.

External DNS Server Configuration

After installing the DNS service on the Windows NT Server-based machines (that is dns-server1 and dns-server2), we use the Microsoft DNS Manager tool to create a primary zone for domain **acme.com** and a reverse lookup primary zone for 100.250.192.in-addr.arpa on dns-server1. This will create an SOA record which contains the primary name server "dns-server1.acme.com." and the e-mail address for the administrator of the DNS server "**web-master@acme.com"** as well as a separate NS record for dns-server1. Since the remaining default parameters for this zone are sufficient, we will not modify them and the defaults will be placed in the SOA record.

The following additional records must be created in the **acme.com** domain and should be done with the "Create Associated PTR Record" enabled where applicable.

Name server records (NS) should be created for dns-server2 and dns13.my-isp.com. Mail exchange records (MX) should be created for mail-server1 and mail-server2. We will set the preference for both of these to 10 for equal load balancing. Address records (A) should be created for each of the computers which connect directly to the "Exposed Internet Network."

The actual name of the gopher server in our example is "gopher-server1.acme.com". The "defacto" standard for supplying services on the Internet is to serve them on computers with host names that correspond with the service being supplied. To allow users to easily locate our gopher service by connecting to the standard "**gopher.acme.com"**, we will use a canonical name (that is alias) for this server. To do this, we create a canonical name record (CNAME) for "gopher" and associate it with "gopher-server1".

Since there are multiple mirrored web servers and FTP servers providing services to external users, there needs to be a way of grouping these so that load balancing across the servers can occur. There are several ways to do this with "round robin" techniques. The method which we will use here is to associate each of the mirrored servers with the same alias name. We will use the popular names of the services as the alias (that is www, FTP, etc.) which will allow easier location of the services by users. To do this, we create a canonical name record (CNAME) for each of the mirrored servers with the service name as the alias and the server name for the host. With this arrangement, each query to a particular DNS server for a particular server name using its alias (that is **www.acme.com**), will return the list of servers in a "round robin" fashion with a different server being listed first in the list each time. Since most resolvers try the first returned entry first, as described previously in this paper, this will provide load balancing between servers.

Once this zone has been setup on the primary DNS server, the **acme.com** and 100.250.192.in-addr.arpa zones should be created on the secondary DNS servers. These secondary zones should be pointing back to the primary DNS server as the master for the zone transfers.

Here is the resulting zone file for the external DNS servers:

```
;-----------------------------------------------------------------------------------
;
; Database file acme.dns for acme.com. zone. (External DNS Server)
;  Zone version: 10
;

@     IN       SOA    dns-server1.acme.com.  web-master.acme.com.   (
      10       ; serial number
      3600     ; refresh
      600      ; retry
      86400    ; expire
      3600     ) ; minimum TTL

;
; Zone NS records
;

@      IN      NS     dns-server1
@      IN      NS     dns-server2
@      IN      NS     dns13.my-isp.com.

;
; Zone records
;

@      IN      MX     10      mail-server1
@      IN      MX     10      mail-server2
dns-server1     IN    A       192.250.100.11
dns-server2     IN    A       192.250.100.12
FTP-server1     IN    A       192.250.100.41
FTP-server2     IN    A       192.250.100.42
gopher-server1  IN    A       192.250.100.31
mail-server1    IN    A       192.250.200.81
mail-server2    IN    A       192.250.200.82
proxy-server1   IN    A       192.250.100.101
proxy-server2   IN    A       192.250.100.102
proxy-server3   IN    A       192.250.100.103
;
www-server1     IN    A       192.250.100.21
www-server2     IN    A       192.250.100.22
www-server3     IN    A       192.250.100.23
www-server4     IN    A       192.250.100.24
;
FTP      IN    CNAME  FTP-server1
FTP      IN    CNAME  FTP-server2
;
gopher  IN     CNAME  gopher-server1
;
www     IN     CNAME  www-server1
www     IN     CNAME  www-server2
www     IN     CNAME  www-server3
www     IN     CNAME  www-server4
;
;-----------------------------------------------------------------------------------
```

Internal DNS Server Configuration

After installing the DNS service on the Windows NT Server-based machines (that is dns-internal1 and dns-internal2), we use the Microsoft DNS Manager tool to create a primary zone for the internal domain **acme.com** and a reverse lookup primary zone for 200.55.157.in-addr.arpa on dns-internal1.

This will create an SOA record which contains the primary name server “dns-internal1.acme.com.” and the e-mail address for the administrator of the DNS server “**web-master@acme.com”** as well as a separate NS record for dns-internal1. Since the remaining default parameters for this zone are sufficient, we will not modify them and the defaults will be placed in the SOA record.

After creating these zones, we will enable WINS Lookup on the **acme.com** zone and enter the IP addresses of the two internal WINS servers. This will create a WINS record (WINS) in this zone file. We will also enable WINS Reverse Lookup on the 200.55.157.in-addr.arpa zone and enter the DNS host domain as “**acme.com**.”. This will create a WINS reverse lookup record (WINS-R) in this zone file.

The following additional records must be created in the internal **acme.com** domain and should be done with the “Create Associated PTR Record” enabled where applicable.

A name server record (NS) should be created for dns-internal2. Address records (A) should be created for each of the computers which connect directly to the “Isolated Corporate Network” which are not WINS clients. It may be that all of the servers within the corporate network are WINS aware and therefore no static entries would be needed, but just to show how you can mix static entries with WINS, we will statically define the proxy servers and the DNS servers in the DNS zone file. We will also create a “localhost” address record for the local address 127.0.0.1.

Since there are multiple mirrored web servers providing services to internal users, we will use the “round robin” technique using canonical names as we did on the external network and associate these mirrored servers with “**corp-web.acme.com”**. The difference here is that there are no associated address records (A) for the internal Web servers (that is www-internal1, www-internal2, www-internal3) since these are WINS clients and the DNS server will query the WINS server for these addresses when needed.

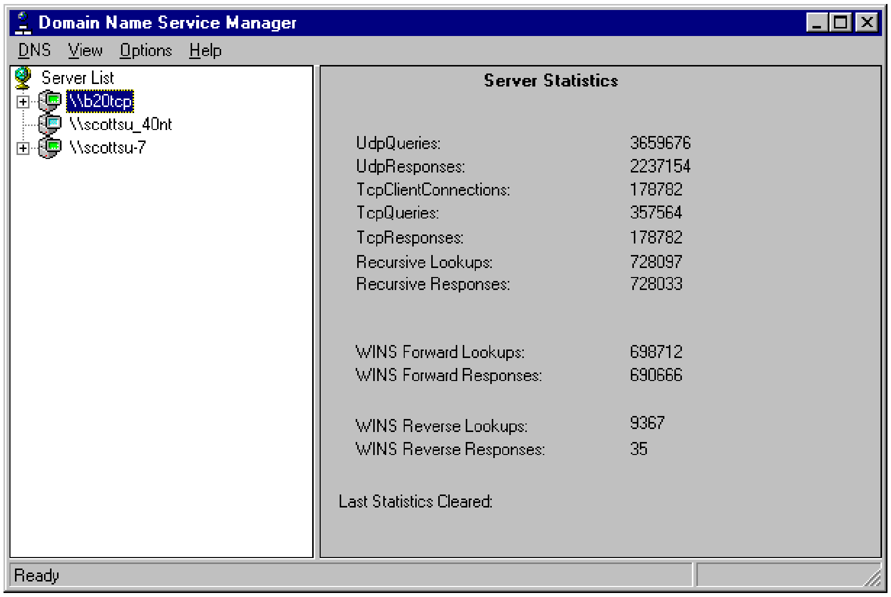
Once this zone has been setup on the primary DNS server, the **acme.com** and 200.55.157.in-addr.arpa zones should be created on the secondary DNS server. These secondary zones should be pointing back to the primary DNS server as the master for the zone transfers.

Here is the resulting zone file for the internal DNS servers:

```
;------------------------------------------------------------------------------------
;
; Database file acme.dns for acme.com. zone. (Internal DNS Server)
;  Zone version: 12
;

@      IN      SOA     dns-internal1.acme.com.       web-master.acme.com.(
      12       ; serial number
      3600     ; refresh
      600      ; retry
      86400    ; expire
      3600     ) ; minimum TTL

;
; Zone NS records
;

@      IN      NS      dns-internal1
@       IN      NS      dns-internal2

;
; WINS lookup record
;

@       IN      WINS    157.55.200.91 157.55.200.92

;
; Zone records
;

dns-server1     IN      A       157.55.200.11
dns-server2     IN      A       157.55.200.12
proxy-server1   IN      A       192.250.100.101
proxy-server2   IN      A       192.250.100.102
proxy-server3   IN      A       192.250.100.103
localhost                IN      A       127.0.0.1
;
corpweb      IN CNAME   www-internal1
corpweb      IN CNAME   www-internal2
corpweb      IN CNAME   www-internal3
;
;-------------------------------------------------------------------------------------
```

Statistics

Microsoft DNS keeps track of a number of query statistics. These statistics can be seen via the DNS Manager. These statistics are cumulative since the last time the DNS service was started. They are useful when comparing relative loading between two DNS servers to determine how many requests each is servicing. If one server is busier than the other, then you might want to consider changing the current DNS architecture or reconfiguring some of the resolvers. The following figure displays the statistics for server \\B20TCP:



Server Statistics

```
UdpQueries:                  3659676
UdpResponses:                2237154
TcpClientConnections:        178782
TcpQueries:                  357564
TcpResponses:                178782
Recursive Lookups:           728097
Recursive Responses:         728033


WINS Forward Lookups:        698712
WINS Forward Responses:      690666

WINS Reverse Lookups:        9367
WINS Reverse Responses:      35

Last Statistics Cleared:
```

For capacity planning purposes, on a dedicated DNS server, it is important to log the statistics periodically and watch the growth of the queries vs. the memory, disk, network, and CPU performance of the server. Since the statistics are kept since the DNS service was restarted, you will probably want to note the date and time the DNS service was started (you can get this information from the Eventlog).

Now that you know how many queries per second the DNS server is answering, how many is too many? The answer depends on:

- What hardware is the server running on?
- Is the machine dedicated to DNS or is it being used for other purposes?
- What is the current network bandwidth?

If you feel that your DNS server is not responding as fast as it should, then you should first look at the following performance monitor counters on that server. If any of these thresholds are being hit, then the system has a performance problem and you will need to do more investigation:

• processor	• % processor time	• > 80%	•
• memory	• available bytes	• < 1MB	•
• physical disk	• % disk time	• > 67%	• only available if you start the counters from the command line with diskperf -y and reboot
• network segment	• % network utilization	• > 40%	• only available if Network Monitor Agent is installed

If memory is an issue then you should check the “process” object, “DNS” instance, “Private Bytes” counter. If this counter is very high then the DNS service is what is causing the memory available bytes counter to be low. If this is the case then you should add more memory, or possibly investigate what records are currently active in the DNS cache and find out the TTL associated with each.

If processor is an issue then you should check the “process” object, “DNS” instance, “%processor time” counter. If this is > 80 % then you might consider running the DNS server on a more powerful server. If this is not high, and processor %processor time is still high, look at all the other processes and determine which process is requiring large amounts of CPU time.

On a DNS server, physical disk %disk time should not be a problem, unless the system has started paging for lack of memory. If this is the case then you need to treat this as a memory issue.

If network utilization is a problem, you need to find out if it is the DNS server that is causing all of the traffic or another server on the same segment. To do this, use the full featured Network Monitor utility that ships with Microsoft Systems Management Server (this allows you to capture all traffic to and from any specific hardware address) or the reduced feature set utility that ships with Windows NT 4.0 (this only allows you to track traffic to and from the local server).

Load Balancing DNS

If you have multiple DNS servers, for example a primary and 2 secondaries, then you will probably want to configure some of the clients to point to one DNS secondary and other clients to point to the other DNS secondary.

The Zone Transfer

A zone transfer is nothing more than a file copy. The entire contents of the database get copied from the primary (or master) to the secondary each time the secondary is “notified” by the primary (or master) that there has been a change.

When the secondary’s DNS service starts or the secondary’s “refresh interval” in its SOA record has expired (by default this is set to 3 hours), it will query its primary to determine the serial number in the SOA record. If it is different than that of its local database, it will ask for a zone transfer.

The first frame is the request for the primary’s SOA record.

```
secondary  primary  DNS  0x4000:Std Qry for scottsu.com of type SOA on class INET

+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
+ IP: ID = 0x7701; Proto = UDP; Len: 57
+ UDP: Src Port: DNS, (53); Dst Port: DNS (53); Length = 37 (0x25)
 DNS: 0x4000:Std Qry for scottsu.com of type SOA on class INET addr.
  DNS: Query Identifier = 16384 (0x4000)
 + DNS: DNS Flags = Query, OpCode—Std Qry, RCode—No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 0 (0x0)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: scottsu.com of type SOA on class INET addr.
   DNS: Question Name: scottsu.com
   DNS: Question Type = Start of zone of authority
   DNS: Question Class = Internet address class
```

The next frame is the SOA record being returned.

```
primary  secondary  DNS  0x4000:Std Qry Resp. for scottsu.com of type SOA on class INET

+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
+ IP: ID = 0x3617; Proto = UDP; Len: 149
+ UDP: Src Port: DNS, (53); Dst Port: DNS (53); Length = 129 (0x81)
 DNS: 0x4000:Std Qry Resp. for scottsu.com of type SOA on class INET addr.
  DNS: Query Identifier = 16384 (0x4000)
 + DNS: DNS Flags = Response, OpCode—Std Qry, AA RA Bits Set, RCode—No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 1 (0x1)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: scottsu.com of type SOA on class INET addr.
   DNS: Question Name: scottsu.com
   DNS: Question Type = Start of zone of authority
   DNS: Question Class = Internet address class
  DNS: Answer section: scottsu.com of type SOA on class INET addr.
   DNS: Resource Name: scottsu.com
   DNS: Resource Type = Start of zone of authority
   DNS: Resource Class = Internet address class
   DNS: Time To Live = 0 (0x0)
   DNS: Resource Data Length = 80 (0x50)
   DNS: Primary Name Server: scottsu-7.scottsu.com
   DNS: Responsible Authorative Mailbox: Administrator.scottsu-7.scottsu.com
   DNS: Version number = 19 (0x13)
   DNS: Refresh Interval = 3600 (0xE10)
   DNS: Retry interval = 600 (0x258)
   DNS: Expiration Limit = 86400 (0x15180)
   DNS: Minimum TTL = 3600 (0xE10)
```

The next frame is the request for the zone transfer.

```
secondary  primary  DNS  0x0:Std Qry for scottsu.com of type Req. for zn Xfer

+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
+ IP: ID = 0x7A01; Proto = TCP; Len: 71
+ TCP: .AP..., len: 31, seq: 365696, ack: 35871892, win: 8760, src: 1072 dst: 53
 DNS: 0x0:Std Qry for scottsu.com of type Req. for zn Xfer on class INET addr.
  DNS: TCP Length = 29 (0x1D)
  DNS: Query Identifier = 0 (0x0)
 + DNS: DNS Flags = Query, OpCode—Std Qry, RCode—No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 0 (0x0)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: scottsu.com of type Req. for zn Xfer on class INET addr.
   DNS: Question Name: scottsu.com
   DNS: Question Type = Request for zone transfer
   DNS: Question Class = Internet address class
```

The next frame contains the database being returned from the master.

```
primary  secondary  DNS  0x0:Std Qry Resp. for scottsu.com of type SOA

+ FRAME: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP: DOD Internet Protocol
+ IP: ID = 0x3817; Proto = TCP; Len: 163
+ TCP: .AP..., len: 123, seq: 35871892, ack: 365727, win: 8729, src: 53 dst: 1072
 DNS: 0x0:Std Qry Resp. for scottsu.com of type SOA on class INET addr.
  DNS: TCP Length = 121 (0x79)
  DNS: Query Identifier = 0 (0x0)
 + DNS: DNS Flags = Response, OpCode—Std Qry, RA Bits Set, RCode—No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 1 (0x1)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: scottsu.com of type Req. for zn Xfer on class INET addr.
   DNS: Question Name: scottsu.com
   DNS: Question Type = Request for zone transfer
   DNS: Question Class = Internet address class
  DNS: Answer section: of type SOA on class INET addr.
   DNS: Resource Name: scottsu.com
   DNS: Resource Type = Start of zone of authority
   DNS: Resource Class = Internet address class
   DNS: Time To Live = 0 (0x0)
   DNS: Resource Data Length = 80 (0x50)
   DNS: Primary Name Server: scottsu-7.scottsu.com
   DNS: Responsible Authorative Mailbox: Administrator.scottsu-7.scottsu.com
   DNS: Version number = 19 (0x13)
   DNS: Refresh Interval = 3600 (0xE10)
   DNS: Retry interval = 600 (0x258)
   DNS: Expiration Limit = 86400 (0x15180)
   DNS: Minimum TTL = 3600 (0xE10)

00000000 00 A0 24 63 AB 22 00 80 C7 A9 EC 0F 08 00 45 00 ..$c."........E.
00000010 02 AB 39 17 40 00 80 06 B9 C6 9D 37 66 34 9D 37 ..9.@......7f4.7
00000020 64 CC 00 35 04 30 02 23 5D 0F 00 05 94 9F 50 18 d..5.0.#].....P.
00000030 22 19 58 A8 00 00 00 3D 00 00 80 80 00 01 00 01 ".X....=........
00000040 00 00 00 00 07 73 63 6F 74 74 73 75 03 63 6F 6D .....scottsu.com
00000050 00 00 FC 00 01 C0 0C FF 01 00 01 00 00 0E 10 00 ................
00000060 14 00 00 00 00 05 00 00 00 58 02 00 00 01 00 00 .........X......
00000070 00 9D 37 66 34 00 40 00 00 80 80 00 01 00 01 00 ..7f4.@.........
00000080 00 00 00 07 73 63 6F 74 74 73 75 03 63 6F 6D 00 ....scottsu.com.
00000090 00 FC 00 01 C0 0C 00 02 00 01 00 00 0E 10 00 17 ................
000000A0 09 73 63 6F 74 74 73 75 2D 37 07 73 63 6F 74 74 .scottsu-7.scott
000000B0 73 75 03 63 6F 6D 00 00 43 00 00 80 80 00 01 00 su.com..C.......
000000C0 01 00 00 00 00 07 73 63 6F 74 74 73 75 03 63 6F ......scottsu.co
000000D0 6D 00 00 FC 00 01 C0 0C 00 02 00 01 00 00 0E 10 m...............
000000E0 00 1A 0C 73 63 6F 74 74 73 75 5F 6E 74 34 30 07 ...scottsu_nt40.
000000F0 73 63 6F 74 74 73 75 03 63 6F 6D 00 00 51 00 00 scottsu.com..Q..
00000100 80 80 00 01 00 01 00 00 00 00 07 73 63 6F 74 74 ...........scott
00000110 73 75 03 63 6F 6D 00 00 FC 00 01 07 67 6C 65 6E su.com......glen
00000120 6E 77 6F C0 0C 00 02 00 01 00 00 0E 10 00 20 0A nwo.............
00000130 63 6F 70 70 65 72 68 65 61 64 07 67 6C 65 6E 6E copperhead.glenn
00000140 77 6F 07 73 63 6F 74 74 73 75 03 63 6F 6D 00 00 wo.scottsu.com..
00000150 40 00 00 80 80 00 01 00 01 00 00 00 00 07 73 63 @.............sc
00000160 6F 74 74 73 75 03 63 6F 6D 00 00 FC 00 01 0A 63 ottsu.com......c
00000170 6F 70 70 65 72 68 65 61 64 07 67 6C 65 6E 6E 77 opperhead.glennw
00000180 6F C0 0C 00 01 00 01 00 00 0E 10 00 04 9D 37 6A o.............7j
00000190 C1 00 36 00 00 80 80 00 01 00 01 00 00 00 00 07 ..6.............
000001A0 73 63 6F 74 74 73 75 03 63 6F 6D 00 00 FC 00 01 scottsu.com.....
000001B0 08 67 6F 6F 6F 6F 6F 6F 64 C0 0C 00 01 00 01 00 .goooood.......
000001C0 00 00 00 00 04 9D 37 64 64 00 37 00 00 80 80 00 ......7dd.7.....
000001D0 01 00 01 00 00 00 00 07 73 63 6F 74 74 73 75 03 ........scottsu.
000001E0 63 6F 6D 00 00 FC 00 01 09 73 63 6F 74 74 73 75 com......scottsu
000001F0 2D 37 C0 0C 00 01 00 01 00 00 0E 10 00 04 9D 37 -7.............7
00000200 66 34 00 3A 00 00 80 80 00 01 00 01 00 00 00 00 f4.:............
00000210 07 73 63 6F 74 74 73 75 03 63 6F 6D 00 00 FC 00 .scottsu.com....
00000220 01 0C 73 63 6F 74 74 73 75 5F 6E 74 34 30 C0 0C ..scottsu_nt40..
```

```
00000230 00 01 00 01 00 00 0E 10 00 04 9D 37 64 CC 00 79 ...........7d..y
00000240 00 00 80 80 00 01 00 01 00 00 00 00 07 73 63 6F .............sco
00000250 74 74 73 75 03 63 6F 6D 00 00 FC 00 01 C0 0C 00 ttsu.com........
00000260 06 00 01 00 00 00 00 00 50 09 73 63 6F 74 74 73 ........P.scotts
00000270 75 2D 37 07 73 63 6F 74 74 73 75 03 63 6F 6D 00 u-7.scottsu.com.
00000280 0D 41 64 6D 69 6E 69 73 74 72 61 74 6F 72 09 73 .Administrator.s
00000290 63 6F 74 74 73 75 2D 37 07 73 63 6F 74 74 73 75 cottsu-7.scottsu
000002A0 03 63 6F 6D 00 00 00 00 13 00 00 0E 10 00 00 02 .com............
000002B0 58 00 01 51 80 00 00 0E 10      X..Q.....
```

Memory consumed by DNS records

DNS RR records are very small, somewhere in the neighborhood of 16 bytes. Domain records are also very small, they are around 56 bytes. If you load a great deal of records into the server you might not see the DNS process Private Bytes counter jump because the DNS server allocates the memory in blocks.

The way DNS uses the cache is a different story. Virtual (pageable) memory is allocated for cached entries as needed (remember, as a DNS server does a resolution, it caches a copy of the data for that data's TTL). If the server is starved for memory and more is needed, then this memory will come from the page file (which is usually very slow).

When designing a DNS architecture it is important to ask a number of questions about the current and future infrastructures. Hopefully the following questions will help you develop a reasonable DNS backbone for your corporation.

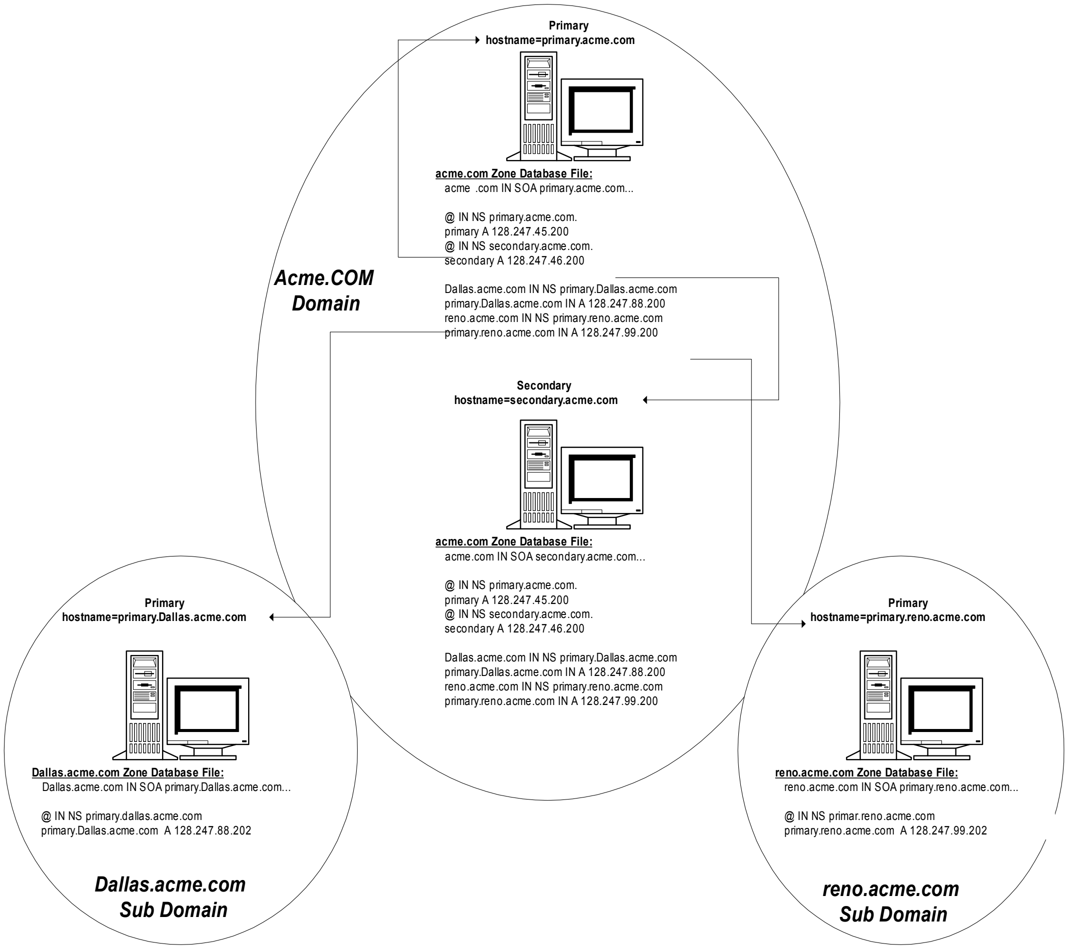
How Many Domains do I need?

This is by far the most important question you will have to answer!

There are two different routes to take when designing a DNS architecture for a company's network. The first option is the easiest—create one DNS domain for the whole company (for example **Acme.com**). This means that you would have one primary DNS server and one or more secondary DNS servers. There are some downsides to choosing this option. The first being that the Primary DNS may have a problem keeping up with all of the secondaries polling. There are solutions to this problem such as, increase the secondary refresh interval, make some of the secondaries load from other secondaries (that is multiple DNS masters) and/or create some caching servers (allows you to avoid the overhead of a zone transfer—Warning : they are only valuable once they build up their cache). If you choose this approach and you decide in the future that it should be re-designed it can always be broken up.

The second option is more for the larger network installations which span multiple sites (this is the option we will explore in great detail in the rest of the paper). In this design you would have more than one DNS domain. The architecture would consist of one root domain (with a primary DNS server and one or more secondary DNS servers) and with one or more sub-domains (each with a primary DNS server and one or more secondary DNS servers). An example of the subdomains might be **Dallas.Acme.Com** and **Reno.Acme.Com**. The reason for such a design is quite obvious. A network architect usually breaks up a corporate DNS domain into multiple DNS domains to distribute the management of parts of the domain to various entities within the organization or to enable multiple zone files since zone files can only be created at the domain level. Another less obvious reason may be to simply allow for organizational affiliation of your systems. If you choose this route then the structure of your domain should mirror the structure of your organization, especially your companies support structure. Either way, you will need to determine how many domains your company will need.

Here is an example of such a configuration showing the DNS servers and the zone database files on each:


Primary
hostname=primary.acme.com
Acme.COM
Domain
Secondary
hostname=secondary.acme.com
Primary
hostname=primary.Dallas.acme.com
Dallas.acme.com
Sub Domain
Primary
hostname=primary.reno.acme.com
reno.acme.com
Sub Domain

acme.com Zone Database File:

```
acme  .com IN SOA primary.acme.com...

@ IN NS primary.acme.com.
primary A 128.247.45.200
@ IN NS secondary.acme.com.
secondary A 128.247.46.200

Dallas.acme.com IN NS primary.Dallas.acme.com
primary.Dallas.acme.com IN A 128.247.88.200
reno.acme.com IN NS primary.reno.acme.com
primary.reno.acme.com IN A 128.247.99.200
```

acme.com Zone Database File:

```
acme.com IN SOA secondary.acme.com...

@ IN NS primary.acme.com.
primary A 128.247.45.200
@ IN NS secondary.acme.com.
secondary A 128.247.46.200

Dallas.acme.com IN NS primary.Dallas.acme.com
primary.Dallas.acme.com IN A 128.247.88.200
reno.acme.com IN NS primary.reno.acme.com
primary.reno.acme.com IN A 128.247.99.200
```

Dallas.acme.com Zone Database File:

```
Dallas.acme.com IN SOA primary.Dallas.acme.com...

@ IN NS primary.dallas.acme.com
primary.Dallas.acme.com  A 128.247.88.202
```

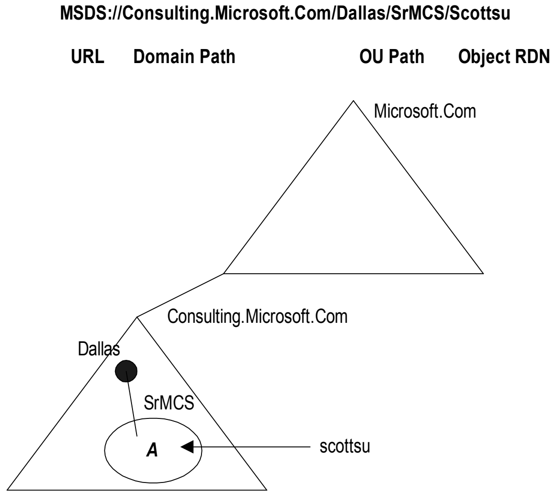
reno.acme.com Zone Database File:

```
reno.acme.com IN SOA primary.reno.acme.com...

@ IN NS primar.reno.acme.com
primary.reno.acme.com  A 128.247.99.202
```

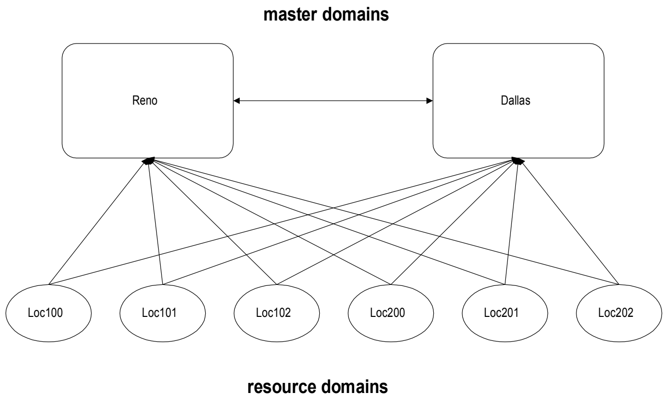
From a futures migration standpoint it's recommended that you create a DNS domain for every Windows NT domain that you currently have in your enterprise.

It may not be that apparent today, but for a smooth migration to the next major revision of Windows NT this will be important. The next major revision of Windows NT is tightly coupled with the DNS backbone. See the Futures section of this paper for more information.

The following figure tries to show how a name might look in the future, keep in mind that you will still be able to use friendly names such as **Scottsu@microsoft.com**.



For a general overview of how a Windows NT domain might map to a DNS domain see the following figure. It has a Multiple Master domain with 2 Masters Reno and Dallas, and 6 Resources L100 to L202.



For the previous domain design you might have 9 DNS domains, Acme.-com, Reno.Acme.com, Dallas.Acme.com, L100.Reno.com, L101.Reno.com, L102.Reno.com, L201.Dallas.com, L202.Dallas.com and L201.Dallas.com.

Example Configurations

Most corporate domains designs fall into one of the following models, the Single domain model, the Master domain model, the Multiple Master domain model or the Complete Trust domain model. For more information on each of these models refer to the Windows NT Adminstration guide.

In this section we are going to discuss two of these models, the Complete

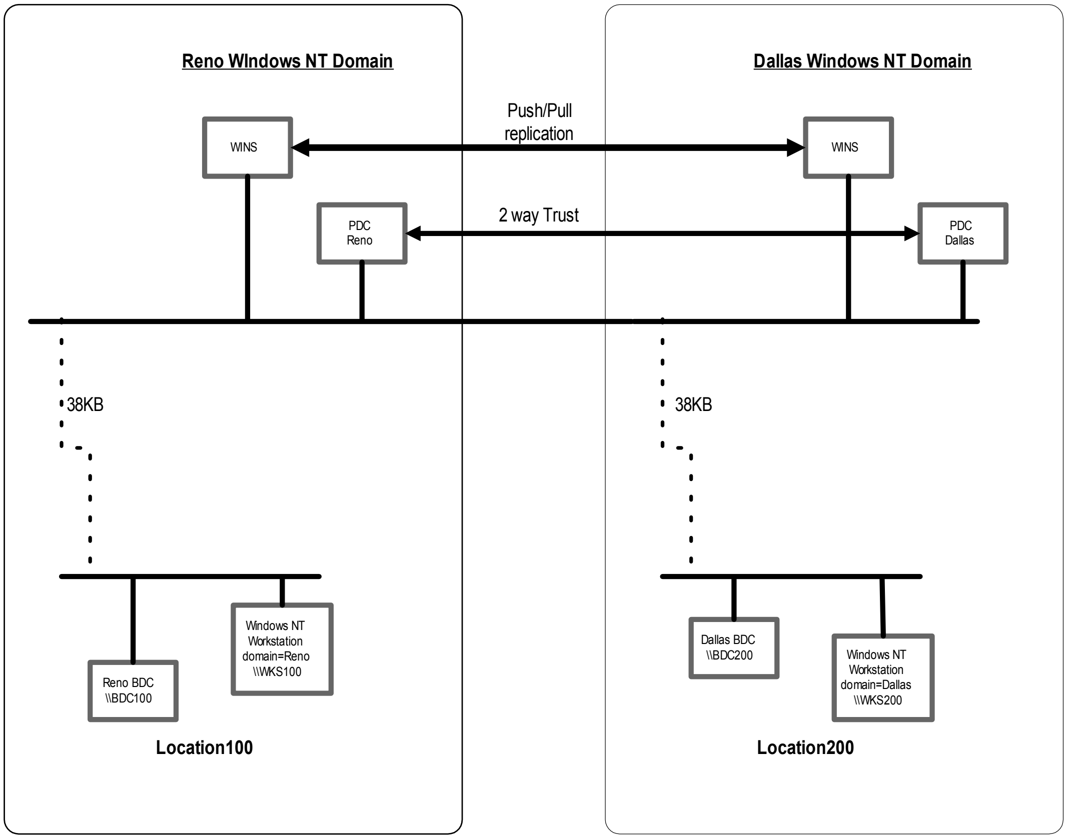
Trust model and the Multiple Master model. We will look at the models before DNS and the models after DNS. Hopefully, this will give you an understanding of how to implement DNS within your own environment.

Complete Trust Examples

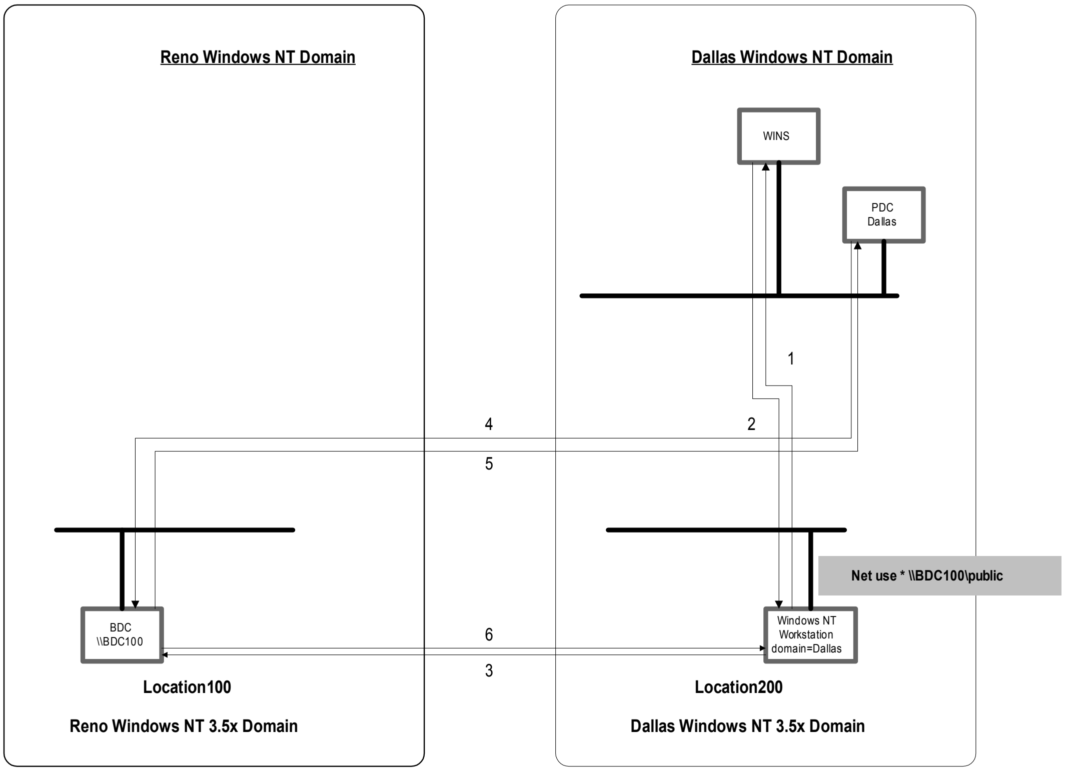
Before DNS

Let's start with the following domain design (without DNS) and determine the best approach to a DNS design architecture. The following is an example of a Complete Trust domain model. There are 2 domains "Reno" and "Dallas." Let's assume the following:

- There is a data-center in Dallas and Reno with a high speed link between both data-centers
- Each data-center has a large number of remote sites (500 each with 20 users each) that it supports via 38K links direct connected to the appropriate data-center (in our diagram we only show two).
- Each data-center has the primary domain controller (PDC) and WINS server that supports the remote locations.
- Each remote site has one or more Windows NT backup domain controllers (BDCs) that get their security account management (SAM) database from their respective PDC in the data-center. This means that user account replication is going to occur over the slow (38K) link, hence there are few user accounts in the database and the password expiration is turned off. A Complete Trust model was used in this design because there is only one server per remote location and there were 20 users per location. If a master or multiple master domain design was in place there would be a requirement for two servers. If one domain was used then the user account database would have been too large. A domain in each location was not considered because the entire corporate network for this company consists of 1000 remote sites.
- The remote BDCs contain a #PRE (preloads permanent entries into the NetBIOS cache—to view the cache type "nbtstat -c") entry for all Primary Domain Controllers in their LMHOSTS file (ex. 128.247.145.200 DALLAS_PDC #PRE #DOM:DALLAS). This is important because each remote BDC needs a secure session setup between its respective PDC and a PDC or BDC in all trusted domains. Because of the slow link to each of the remote locations and the fast link between data-centers it makes more sense to have the secure session setup between the local BDCs and the PDC of the other trusted domains.
- Each remote site has a number of Mnode (broadcast first and then check the WINS server) TCP/IP client workstations (in our diagram we only show two) on the same LAN as the local Hnode (check the WINS server for name resolution before broadcasting) Windows NT-based server(s).



When designing a solution it is important to consider the traffic flow of “session setup” and “logon validation.” You never want to design a solution that would keep a client from contacting its local server if the link between the remote location and its data-center was lost.

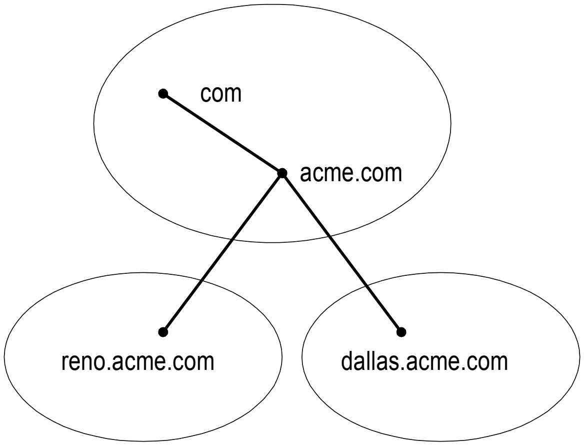


In our example logon validation will occur locally between the clients and the local servers. Session setup and name resolution to the “local” servers will also occur locally because the clients are set up as Mnode TCPIP clients. Session setup and name resolution between a client in one location and a server in another remote location will take a much different path and include both the WINS server in the data-center and the clients domain PDC. In our example (note that in our example some of the devices were removed to make the diagram easier to view), the Windows NT Workstation-based machine in the Dallas domain location 200 is connecting to the server \\BDC100 in the Reno domain location 100. The client first contacts the WINS server in its data-center to get the IP address of the \\BDC100 server (1 and 2). It then tries to establish a session to the \\BDC100 server (3). The \\BDC100 server in turn uses its secure session to the clients PDC for “pass through authentication” purposes (4 and 5). If the security is verified, the client is allowed to establish the session and proceed (6). In this example, there are three physical points of failure, the link between Location 200 and the Dallas data-center, the link between the Dallas data-center and the Reno data-center, and the link between Location 100 and the Reno data-center.

After DNS

Now that you understand the current architecture, how would you design a Windows NT 4.0 DNS implementation?

Here is a possible solution. There are four zones, four domains (**com, acme.com, reno.acme.com and dallas.acme.com**) and a DNS server in **each** remote location in the following design.



The following are facts about the design:

- The DNS servers that contain the **reno.acme.com** and **dallas.acme.com** zone files are located in their respective data-centers and the DNS server that contains the **acme.com** zone file is located in one or the other data-center depending on which data-center has the most client-side remote location session setup traffic. Each DNS zone should also have a WINS record that points to its respective WINS server. There is also a DNS cache only server in each of the remote locations for the same zone.
- Since there was a server located in the remote location, a DNS server was put there as well. Since the link is so slow it makes more sense to make this a cache only server. If there were no servers in the remote location then a DNS probably would not really be necessary, however each case is different and fast name to IP address resolution is a business decision that has to deal with the cost of adding another server versus the cost of slow name resolution.
- In our case **reno.acme.com** and **dallas.acme.com** are going to be domains on the root server in the data-centers. One of the root servers, probably the one in Dallas, will also house domains **com** and **acme**. Each of the caching servers in the remote locations will have these servers listed in their cache file.
- DNS WINS lookup should be configured on the DNS servers in each data-center.
- We may want to consider a secondary to each primary located in the data-center for fault tolerance purposes.
- Since there are no secondary servers in the remote sites there will be no reason to add these to the notify list. However, the secondary servers in the data-centers should be in the notify list.

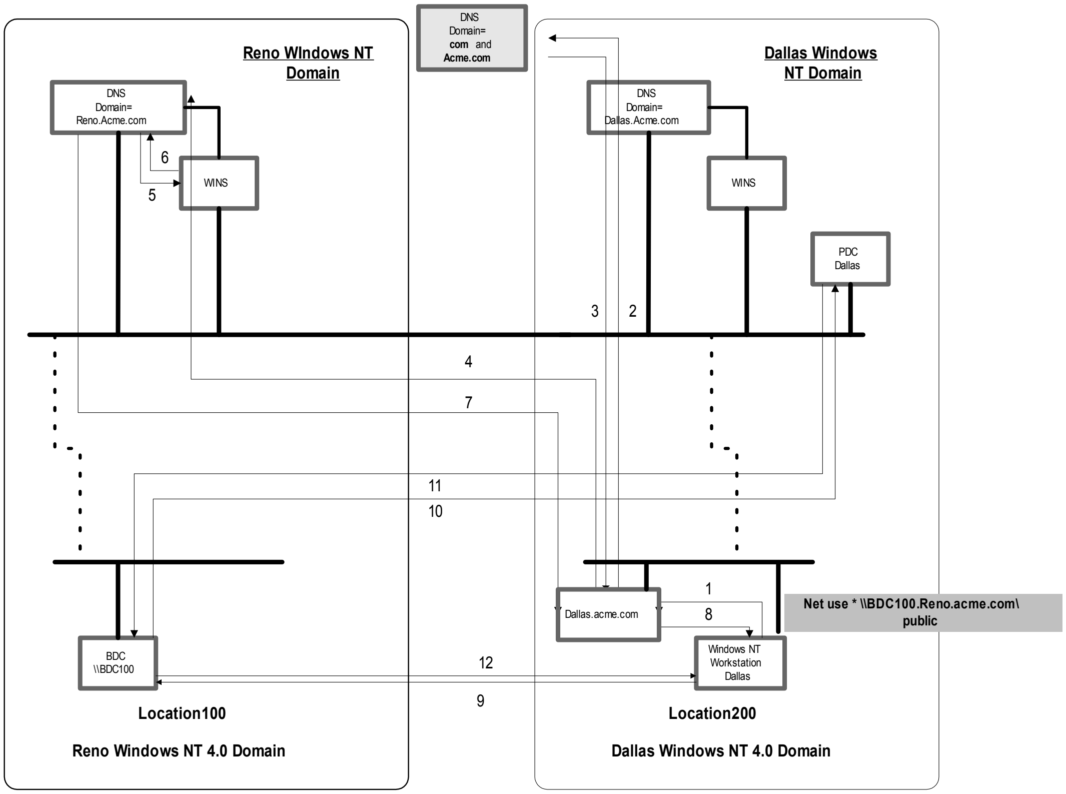
With this design in place a client from a remote location can now establish a session via HOST name rather than NetBIOS name. For example, instead of typing

Net use * \\BDC100\public

they can type

Net use * \\BDC100.reno.acme.com\public

Let’s follow the path of traffic if the client chooses to connect via the HOST name mechanism,



The client contacts its local DNS server with a *recursive* query (1). The **dallas.acme.com** cache-only DNS server checks its local memory cache, if the IP address for **BDC100.reno.acme.com** is not cached, the **dallas.acme.com** DNS server sends an *iterative* query to the ROOT server, then to “COM,” then to “DOMAIN.COM” (that is **Acme.com**) (2). The **acme.com** server will respond back (3) and tell **dallas.acme.com** to contact **reno.acme.com** (4). When **dallas.acme.com** contacts **reno.acme.com**, the DNS server **reno.acme.com** will check the WINS server (5) defined in its WINS record (6) and the **reno.acme.com** responds with the IP address (7) and the directed name query and session setup can continue (8,9,10,11 and 12). If the IP addresses were not cached, it would take eight frames for IP address to name resolution (compared to two frames in the WINS design).

*If the **dallas.acme.com** DNS server was configured with a “Forwarder” the request to **acme.com** would have still been iterative. However, if the **dallas.acme.com** DNS server was configured as a “Slave” server the request to **acme.com** would have been recursive.*

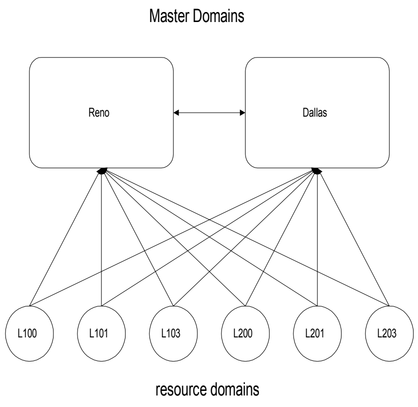
Once Dynamic DNS and Enhanced Directory Services become a reality, the WINS servers in this enterprise can be removed and NetBIOS can be turned off. From this point on DNS will be used as the index into the directory to find machine objects such as computers and domain controllers.

Multiple Master Example

In the following section we are going to go through a very simple “well architected” Windows NT 4.0 Multiple Master domain model and show how to implement DNS within that environment for a safe migration to future versions of Windows NT.

Before DNS

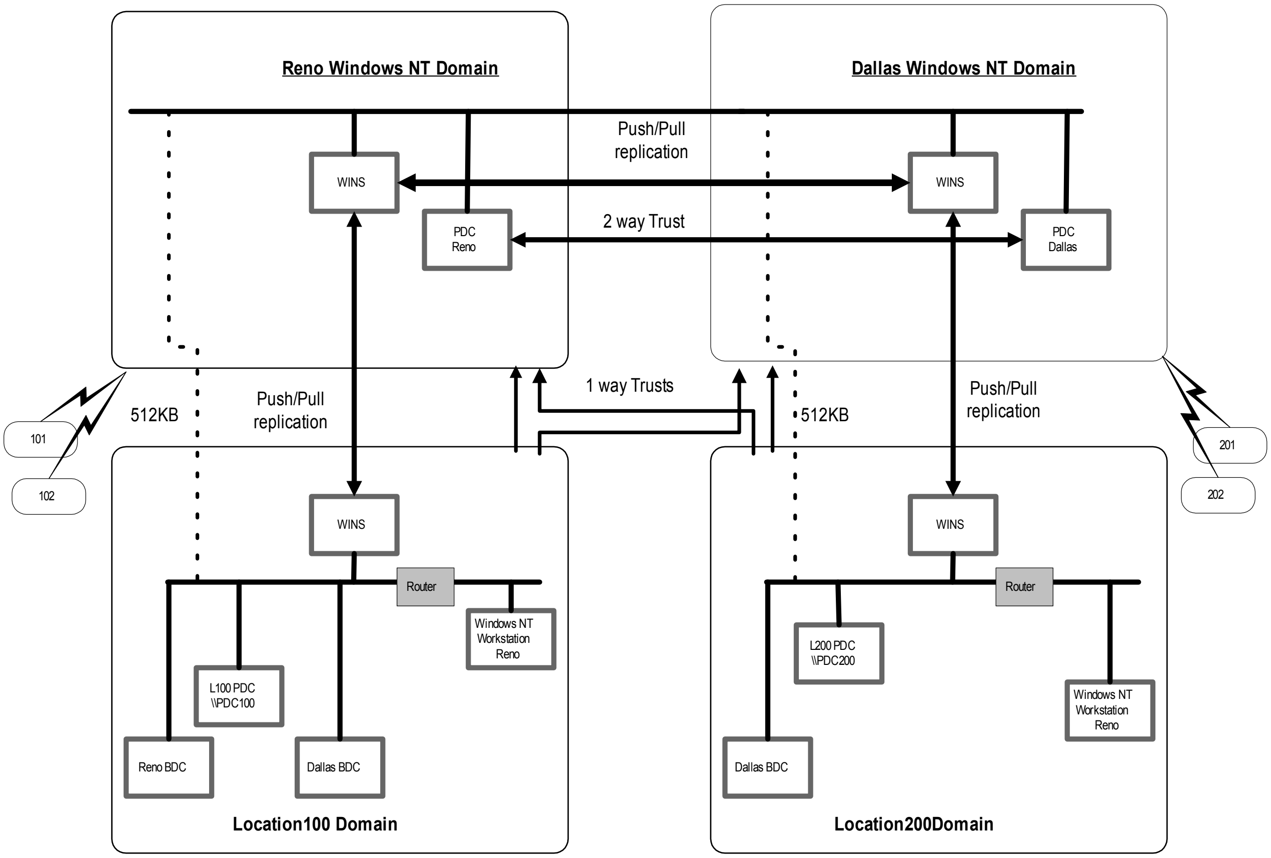
Here is the typical Multiple Master layout with two master domains and six resource domains:



Let’s use the “Reno” and “Dallas” domain structure and assume the following:

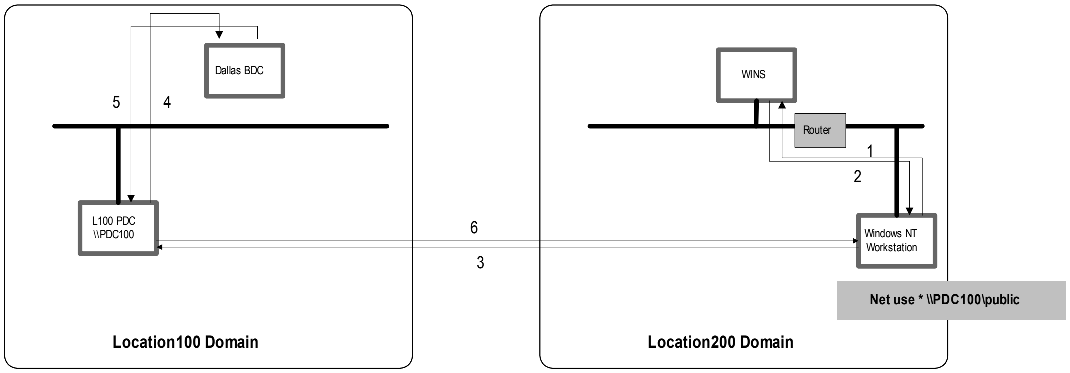
- There is a data-center in Dallas and Reno with a high-speed link between both data-centers (something like a T1).
- Each data-center has a number of remote sites that it supports via 512KB links (in our diagram we only show two, one for location 100 and another for location 200)
- Each data-center has the primary domain controller (PDC) for the data-centers Master domain, a WINS server and a backup domain controller (BDC) for the other Master domain.
- Each remote site has 1 Windows NT backup domain controller (BDC) for each Master domain (account domain) and a PDC (and an undetermined number of BDCs) for the resource domain (no user accounts defined).
- There is a WINS server, that is a push-pull partner to the WINS server in the respective data-center, in each remote location. The WINS servers in the data-centers are push-pull partners with each other as well.
- All clients and servers are Hnode because of the local WINS server.
- All Windows NT-based workstations in the Resource domains have their machine accounts in their respective resource domains, however the users log on to the Master domains.
- A router separates all clients from the servers in the remote locations.

The following figure shows the current layout:



Let's look at the traffic flow for logon validation and session setup name resolution. In most Multiple Master domain scenarios, the client Windows NT-based machine accounts exist in the resource domain and the user logs onto the Master domain. All logon validation will be done locally within the site. A Windows NT Workstation-based client upon bootup queries the local WINS server for a list of BDCs (via name query for DOMAIN <1C>). The WINS server will respond with a list of up to 25 BDCs (the list contains the PDC and all of the BDCs for the master domain that registered with it, as well as others that were replicated to it). The client then sends a request to each server on the list and uses the first server to respond (which normally will be the local master domain BDC) as its server that it sets up a secure channel with and uses to do logon validation.

Remote session setup from a client running Windows NT client in a resource domain to a server in another resource domain occurs as follows as is shown in the next diagram.



The client running Windows NT Workstation will query the WINS server for the address for the server in the domain it wants to contact (1). The WINS server will reply with the address (2) because it was replicated to it previously. The Windows NT-based workstation will then contact the server directly (3). That server will see that the session setup request is from a user in a trusted master domain and use its secure channel to a trusted domain BDC or PDC to verify the credentials (5) and then the server will either allow the client to connect or reject the request (6).

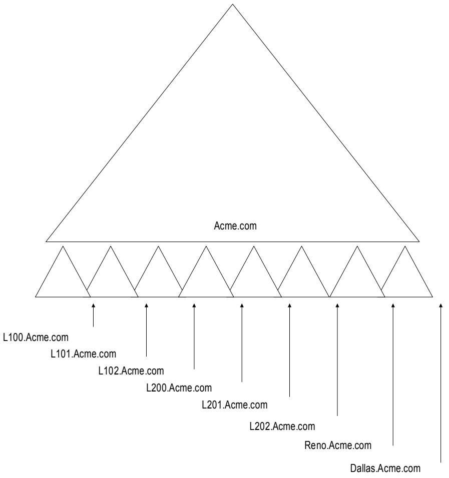
After DNS

Now that you understand the current architecture, how would you design a Windows NT 4.0 DNS implementation? The first thing we have to determine is how to structure the DNS domains. We know that for future migration purposes DNS domains should equal Windows NT 4.0 domains. The first and most important question is now answered—that is, how many domains do I need? The answer is, one for each Windows NT domain—ten.

The second question is, how should the domain names map? For example, should we have **L100.reno.acme.com** or **L100.acme.com**? In any design it's important to keep it simple! That means in our case, keep the name as short as possible. There is another part to this second question and that is, what DNS domain should the clients and servers be part of? In the Multiple Master scenario, the logical solution is to put all workstations within a DNS domain called **L100.Acme.Com** and **L200.Acme.Com** and so on, then put all servers within a domain called **Reno.Acme.Com** and **Dallas.Acme.Com** (this may require changes to each clients TCPIP configuration)**.** This will allow for easy migration to both Dynamic DNS and Enhanced Directory Services.

The third question is, how many servers will we need to support this? There will be 10 DNS domains (the eight expected + **acme.com** and **com**) and eight DNS servers, just like there are eight Windows NT domains and domain controllers (PDCs and BDCs). This will require that the DNS server service be running in each of the locations. The service can run on the existing PDC or BDC within the remote location assuming there is adequate capacity. The DNS servers in the locations that house the resource domains will be primaries for their Windows NT resource domain names such as **L100. Acme.Com** and secondaries to their respective Windows NT Master domains such as **Reno.Acme.Com**.

With all of this said, the DNS domain hierarchy might look something like the following:



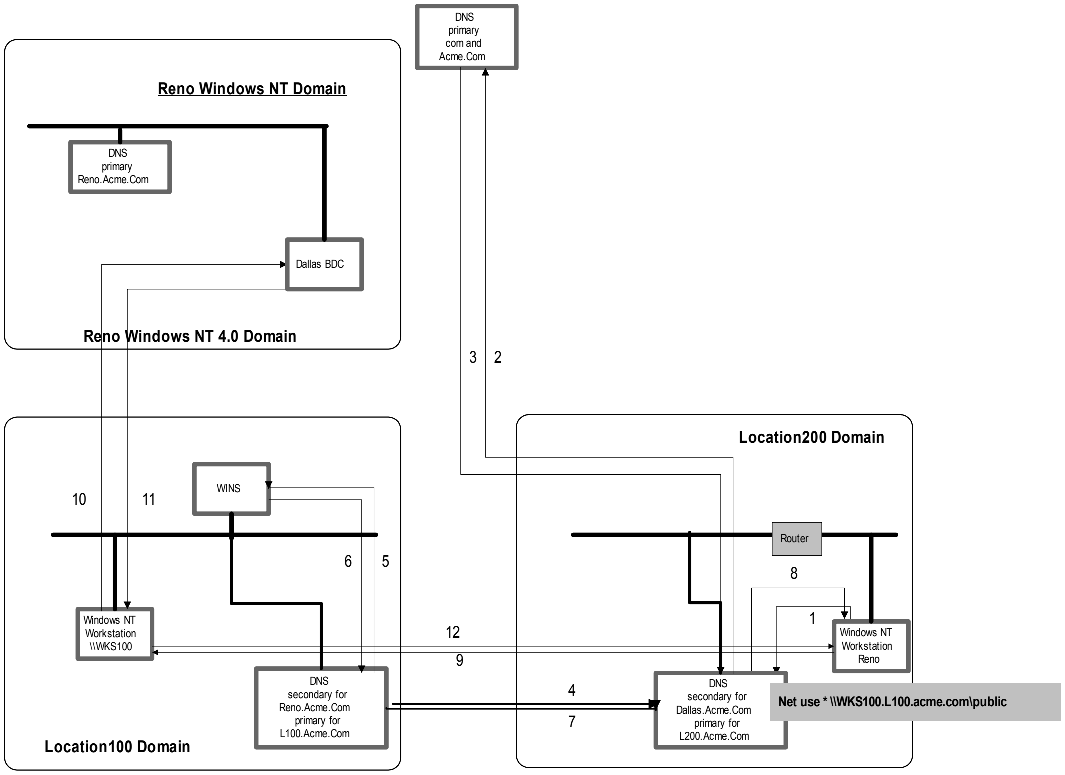
With this design in place, a client from a remote location can now establish a session via HOST name rather than NetBIOS name. For example, instead of typing

Net use * \\WKS100\public

they can type

Net use * \\WKS100.L100.acme.com\public

If the client chooses to connect via the HOST name mechanism, let's follow the path of traffic and compare it to the path via WINS (note that in our example some of the devices were removed to make the diagram easier to view).



The client contacts its local DNS server with a *recursive* query (1) for the "**NT workstation.domain.**" The local DNS server checks its local memory cache, and if the IP address for **WKS100.L100.acme.com** is not cached, the local DNS server sends an ***iterative*** query to the ROOT server, then to "COM," then to "DOMAIN.COM" (that is **acme.com**) (2). The **acme.com** server will respond (3) back and tell the local DNS server to contact **L100.acme.com** (4). **L100.acme.com** will check the WINS server (5 and 6) (the DNS server asks itself, "I should have the IP address for host **WKS100.L100.acme.com** because I am authoritative for domain **L100.acme.-com**, but I don't, so check WINS") defined in its WINS record and responds with the IP address (8) and the directed name query and session setup can continue (9,10,11,12). If the IP addresses were not cached, it would take eight frames for IP address to name resolution (compared to two frames in the WINS design).

•

Both zones on the DNS server within the remote location should point to the same local WINS server.

*The local DNS server authoritative for **reno.acme.com** did not contact its local WINS server because the DNS name being asked for was not a direct child of its DNS domain **reno.acme.com.** It was only a direct child of a DNS server that was authoritative for **L100.acme.com**.*

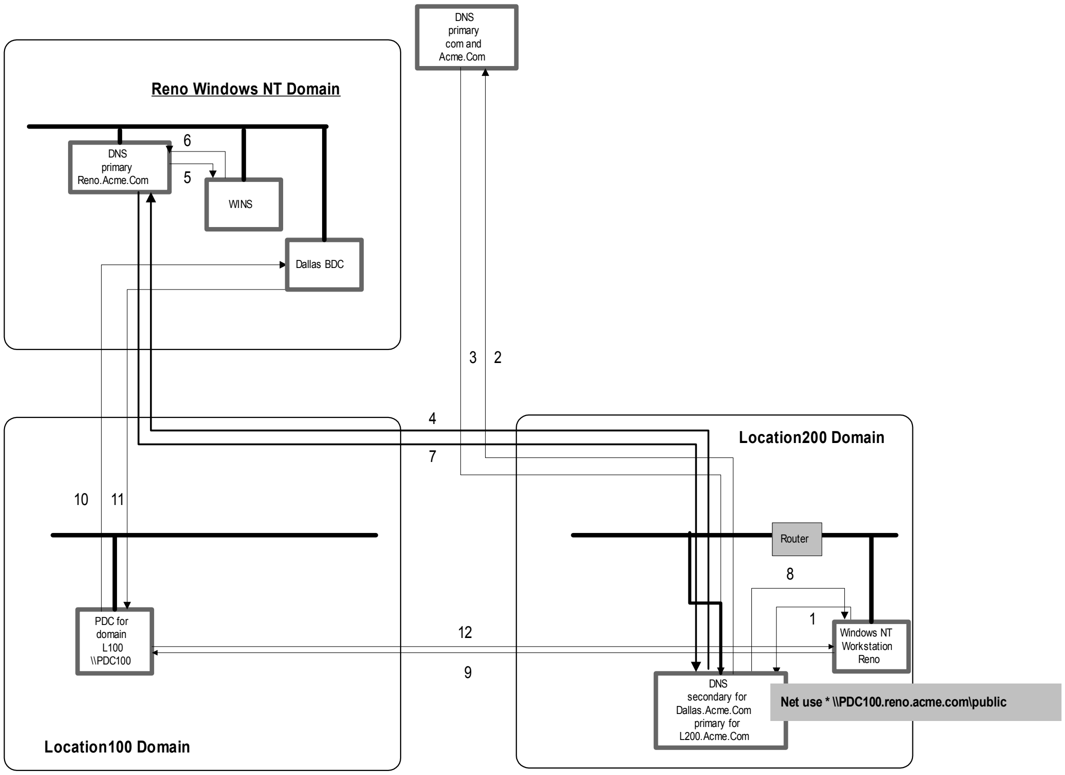
A client from a remote location can also establish a session with a remote server via the HOST name rather than NetBIOS name. For example, instead of typing

Net use * \\PDC100\public

they can type

Net use * \\PDC100.reno.acme.com\public

If the client chooses to connect via the HOST name mechanism, let's follow the path of traffic and compare it to the path via WINS (note that in our example some of the devices were removed to make the diagram easier to view).


DNS primary com and Acme.Com
Reno Windows NT Domain
DNS primary Reno.Acme.Com
6
5
WINS
Dallas BDC
3
2
4
7
Location200 Domain
10
11
Router
8
1
12
9
PDC for domain L100 \\PDC100
Windows NT Workstation Reno
DNS secondary for Dallas.Acme.Com primary for L200.Acme.Com
Net use * \\PDC100.reno.acme.com\public
Location100 Domain

The client contacts its local DNS server with a *recursive* query (1) for the "**NT workstation.domain."** The local DNS server checks its local memory cache, and if the IP address for **PDC100.reno.acme.com** is not cached, the local DNS server sends an *iterative* query to the ROOT server, then to "COM," then to "DOMAIN.COM" (that is **acme.com**) (2). The **acme.com** server will respond (3) back and tell the local DNS server to contact **reno.acme.com** (4). When the local DNS server contacts **reno.acme.com**, the DNS **reno.acme.com** will check the WINS server (5 and 6) (the DNS server asks itself, "I should have the IP address for host **PDC100.reno.acme.com** because I am authoritative for domain **Reno.acme.com**, but I don't, so check WINS") defined in its WINS record and respond with the IP address (7). The local DNS then returns that address to the client (8) and the directed name query and session setup can continue (9,10,11,12). If the IP addresses were not cached it would take eight frames for IP address to name resolution (compared to two frames in the WINS design).

*The local DNS server did not contact its local WINS server because the DNS name being asked for was not a direct child of a DNS domain **L200.acme.com** or **dallas.acme.com** it was only a direct child of a DNS server that was authoritative for **reno.acme.com**.*

Once Dynamic DNS and Enhanced Directory Services become a reality the WINS servers in this enterprise can be removed and NetBIOS can be turned off. From this point on, DNS will be used as the index into the directory to find machine objects such as computers and domain controllers.

It may seem like there are more points of failure for name resolution (eight frames via DNS versus. two frames via WINS), however with this architecture there will be less Dynamic DNS replication than there is WINS replication today and the flat 16 character NetBIOS name space will be history!

Dynamic DNS , IPv6, Incremental Transfer, Secure DNS, Limited use (or no use, it's your choice) of NetBIOS.

The next generation of Windows NT will contain an Enhanced Directory Services implementation. There will be a new way to think about domains, users, groups, and trusts. Enhanced DS domains will map directly to DNS domains, and administration for users and groups can be delegated to Organizational Units (OU) in the directory. The new Enhanced DS domains will be much more functional than Windows NT Server domains today.

The key thing to understand from a DNS implementation perspective, is that DNS will become the primary locator service in the Enhanced Directory Services network. The Enhanced DS clients will use DNS, similar to the way that clients running Windows NT 4.0 use WINS today, to find other servers running the DS service and other hosts on the network.

This section takes a look at some of the new Enhanced Directory Services concepts as well as some of the new standards that will effect DNS in the future. It then goes into what a good DNS/DS design might look like in the future and hopefully will give you an indication of how you might design your network today to be prepared for this future migration.

Dynamic DNS

Windows NT 4.0 includes two name-to-IP-address mapping services—WINS and DNS. One key difference between the two is that whereas WINS accommodates *dynamic* registration of NetBIOS names (and associated IP addresses), DNS names (and associated IP addresses) must be *statically* entered into the DNS service database.

Static registration of name-to-IP-address information is undesirable in the Windows NT/Windows 95 operating systems. This is because in all but very small Microsoft network installations, machines do not have static IP address assignments. The machines use the Dynamic Host Configuration Protocol (DHCP) to obtain an IP address assignment each time they initialize ("boot up").

A proposal for dynamic registration of DNS information is under consideration by the IETF (Internet Engineering Task Force). You can download the specification from "**http://ds.internic.net/internet-drafts/draft-ietf-dnsind-dynDNS-09.txt**". With DNS "dynamic update," a client machine, having obtained its assigned IP address from DHCP, can use a standard protocol to dynamically register its DNS name and IP address in the DNS database. In the Windows NT 4.0 time-frame, it was determined that "dynamic update" should not be implemented due to the fluid status of the specification, and scalability concerns. The current dynamic DNS proposal is based on a "pull from single master" replication model and consequently, if the single master is down or unreachable, dynamic updates cannot occur. Microsoft favors a multiple master arrangement similar to WINS that would allow registrations to continue without a single point of failure.

In the long run, using DNS dynamic update is more desirable than the Windows NT 4.0 DNS-WINS integration for the following reasons:

- DNS-WINS integration does not enable efficient DNS "reverse lookup" (resolution of IP address to DNS name). Reverse lookup is used for security purposes by Internet WWW (World Wide Web) and Firewall services. Due to the recent proliferation of such services, the need for efficient DNS reverse lookup is significantly increasing. If DNS dynamic update was used instead of WINS integration, this problem would no longer exist.
- DNS-WINS integration does not enable a proper "primary-backup" relationship between Microsoft and non-Microsoft DNS name servers. This is due to the fact that non-Microsoft DNS name servers are not capable of WINS lookup. For migration purposes, customers want to install Microsoft DNS servers as backups of non-Microsoft DNS primary servers. If DNS dynamic update was used instead of WINS integration, this problem would no longer exist.
- Non-Microsoft hosts do not register in WINS, and therefore can not "dynamically register" in DNS. DNS dynamic update is an IETF standard. If implemented (by Microsoft), non-Microsoft hosts [that support the DNS dynamic update standard] could dynamically register in the Microsoft DNS, and Microsoft hosts could dynamically register in a non-Microsoft DNS [that supports the DNS dynamic update standard].
- WINS registration is not secure, and has no reasonable means of becoming secure. The IETF is working to complete a standard for adding security to DNS dynamic update. The DNS security standard is not scheduled to enter IETF standards track until the fourth quarter of 1996. Some companies have already decided to forego a standard and "rolled their own" security implementation.

Microsoft-based clients can not register with any of the current versions of Dynamic DNS, and dynamic DNS servers can not replicate their dynamic data to other NON-Dynamic DNS servers.

Warning

In current implementations of Dynamic DNS, the primary is a single point of failure. All of the clients must register their names and IP addresses with this machine. If this machine is down then no updates to the DNS database can occur.

IPv6 (IPng)

IPv6 is defined in RFC 1883 (**http://ds2.internic.net/rfc/rfc1883.txt**). Sometimes this protocol is referred to as "IP Next Generation or IPng." IP version 6 (IPv6) is a new version of the Internet Protocol designed as a successor to IP version 4 (IPv4) [RFC-791]. The changes from IPv4 to IPv6 fall into the following categories:

- Expanded Addressing Capabilities
- Header Format Simplification
- Improved Support for Extensions and Options
- Flow Labeling Capability
- Authentication and Privacy Capabilities

With the onset of this new standard, changes will need to be made to the DNS protocol. RFC 1886 (**http://ds2.internic.net/rfc/rfc1886.txt**) defines these changes. The changes include a new resource record type to store an IPv6 address, a new domain to support lookups based on an IPv6 address, and updated definitions of existing query types that return Internet addresses as part of additional section processing. The extensions are designed to be compatible with existing applications and, in particular, DNS implementations themselves.

Current support for the storage of Internet addresses in the Domain Name System (DNS) cannot easily be extended to support IPv6 addresses since applications assume that address queries return 32-bit IPv4 addresses.

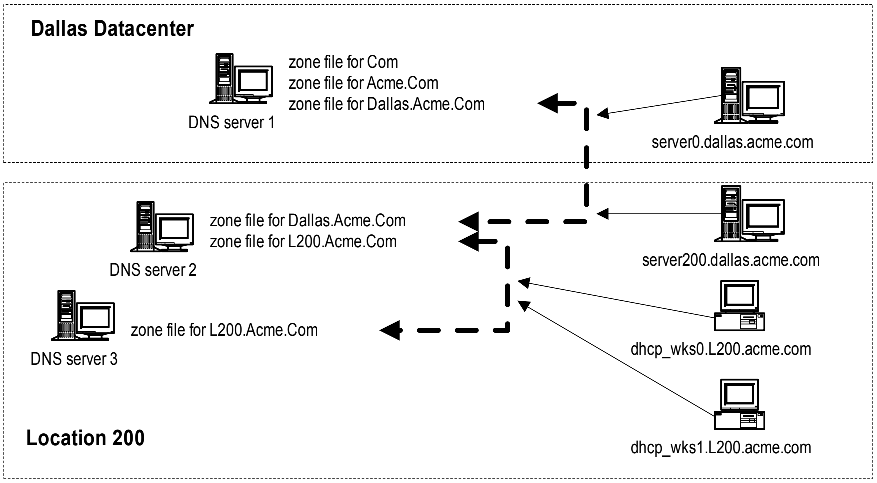
Incremental Transfers—Multimaster replication

Incremental Transfer is for rapid propagation of changes to a DNS database. Windows NT 4.0 does not support the Incremental Transfer protocol, however it will be supported in the future. The Incremental Transfer protocol is designed to reduce latency and decrease the amount of data send during a zone transfer. It does this through two mechanisms.

Notification is used to actively notifying servers of a change to a zone file. The notification is accomplished by the NOTIFY extension of DNS (Microsoft DNS support Notify in Windows NT 4.0). For more information on the notify specification refer to the Internet draft on **http://ds.internic.net/internet-drafts/draft-ietf-dnsind-notify-07.txt**.

Zone propagation is improved by sending just the information which has changed. This is a much better method than the current full zone transfer mechanism, since the current method always transfers the entire zone file. For more information on the Incremental Transfer protocol see the Internet draft on **http://ds.internic.net/internet-drafts/draft-ietf-dnsind-ixfr-06.txt.**

The replication of dynamic data will work very similar to the way that WINS works today, however the data stays within the zone instead of replicating unnecessary data to each and every server (like WINS). The following figure displays the resolver registration and how replication will work:



The dynamic database for L200.acme.com only replicates between DNS server 2 and DNS server 3. The dynamic database for **Dallas.acme.com** only replicates between DNS server 1 and DNS server 2.

Secure DNS

As DNS becomes a critical operational part of the Internet infrastructure, a defined method of security will need to be put in place to assure data integrity and authentication. Extensions to the DNS are described in the IETF-DRAFT “DNS Protocol Security Extensions--30 January 1996” that provide these services to security-aware resolvers or applications through the use of cryptographic digital signatures. These digital signatures are included in secured zones as resource records. Security can still be provided even through non-security-aware DNS servers in many cases.

The extensions also provide for the storage of authenticated public keys in the DNS. This storage of keys can support general public key distribution service as well as DNS security. The stored keys enable security-aware resolvers to learn the authenticating key of zones in addition to those for which they are initially configured. Keys associated with DNS names can be retrieved to support other protocols. Provision is made for a variety of key types and algorithms. In addition, the security extensions provide for the optional authentication of DNS protocol transactions. For more information see “**http://ds.internic.net/internet-drafts/draft-ietf-dnssec-secext-09.txt**”.

MIGRATION

Warning

In current implementations of Dynamic DNS, the vendors had to "roll their own" security protocol because of the lack of a defined standard. Be careful if you implement one of these products as it may become incompatible with future products when a specification is defined.

Where Do We Start?

Let's take a look at the migration process to Enhanced Directory Services. The new Enhanced DS domains will be fully interoperable with Windows NT Server domains. (that is Existing Windows NT Server domains will be able to trust Enhanced DS domains, just as they trust other Windows NT Server domains today.) Enhanced DS servers will also be able to function as backup domain controllers for Windows NT Server domains. This interoperability will allow the upgrade to Enhanced DS server to occur in an orderly fashion, while allowing existing Windows NT Server-based servers to work without modification.

Administrators will not be forced to use the Enhanced DS administration model until they're ready to do so. Even when Enhanced DS servers are on the network, for example, administrators will be able to maintain all account information in the Windows NT Server domain, using the current Windows NT Server administration tools. Thus, administrators will be able to deploy Enhanced DS servers without disrupting the administration of the network.

As Enhanced DS servers are deployed, administrators can begin to store user account information on Enhanced DS servers, while continuing to store and administer those accounts in and from the Windows NT Server domain. This will allow organizations to migrate account information to a Enhanced DS server in an incremental fashion as they gain confidence in the stability and capability of Enhanced DS. Once Enhanced DS servers are well-established, administrators will be able to maintain all account information on Enhanced DS servers, using the Enhanced DS tools. Further, for non-Enhanced DS clients, however, Enhanced DS servers will continue to look and act just like Windows NT Server 4.*x*-based servers.

Once all client and server transitions have been completed, the Enhanced DS environment will be the everyday environment for both end users and system administrators. And, the transition enabled by the interoperability between and integration of Windows NT Server and Enhanced DS will allow organizations to make an easy transition to the unified and global name space provided by Enhanced DS, without disrupting the day-to-day operations of the network.

Since DNS is based on TCPIP standards, and the next generation of Windows NT relies on DNS as its backbone, what happens to IPX and NetBEUI?

There will continue to be support for NetBEUI and IPX, however it is important to understand that the Microsoft stated direction is TCP/IP, as is true for most other networking vendors.

If you choose to use IPX and NetEUI in the next revision of Windows NT, you will be required to use NetBIOS. However, if you choose TCP/IP as your choice protocol, NetBIOS is not required as part of the architecture.

What about Legacy applications that rely on NetBIOS names such as Microsoft Systems Management Server.

You will have a need to use NetBIOS name resolution on your network, as long as you have applications that require NetBIOS names. When you choose to begin the migration to the Enhanced DS you need to find out all of the applications (services and so forth) that require NetBIOS and have a host name migration plan for each.

Finding DCs in an Enhanced Directory Services environment

On the path to a true Windows NT Enhanced Directory Services environment (without NetBIOS) there will be a required migration path that includes the use of all three standards to support backward compatibility with legacy NetBIOS systems.

As machines running Windows NT Enhanced Directory Services start up, they will use existing WINS protocol to register NetBIOS names with their configured WINS server, and register their “A” record(s) with their DNS server. Hence, the Windows NT NetBIOS and DNS name-to-IP-address mappings are made available to all machines using Windows NT Enhanced DS and down level machines (Windows 95, Windows for Workgroups, and so on).

As servers running Windows NT Enhanced DS, that contain the Directory Services Database, start up they will not only register a NetBIOS name with the WINS server, and an “A” record with the DNS server, but they will also register another record with the DNS server that defines the location, DS access protocols supported, transport protocols and so forth. There may be multiple “A” records registered for a machine. For example, an Enhanced Directory Services DC may register:

phoenix.nt.microsoft.com	A	123.123.123.123
domain-controllers.nt.microsoft.com	A	123.123.123.123

This will allow other Enhanced Directory Services workstations to find DCs to validate their security credentials.

Things You Can Count on for the Future

- Microsoft will adopt a “secure” Dynamic DNS solution and our clients will automatically register with the DNS. There will also be a process for using DNS to locate the closest Directory Services DC.
- The next revision of Directory Services will assume that DNS domains map to DS domains.

Rules to Architect a Good DNS Solution for the Future

- If there are servers within a site then there needs to be a DNS server within that site.

APPENDICES
APPENDIX A:
THE WIRE

- Create a DNS domain for each Windows NT 4.0 domain.

Every site DNS server should be a primary for the site specific DNS domain and a secondary for the parent DNS domain.

Windows-based clients should be registered in a site specific DNS domain.

Servers running Windows NT Server should be registered in a master DNS domain.

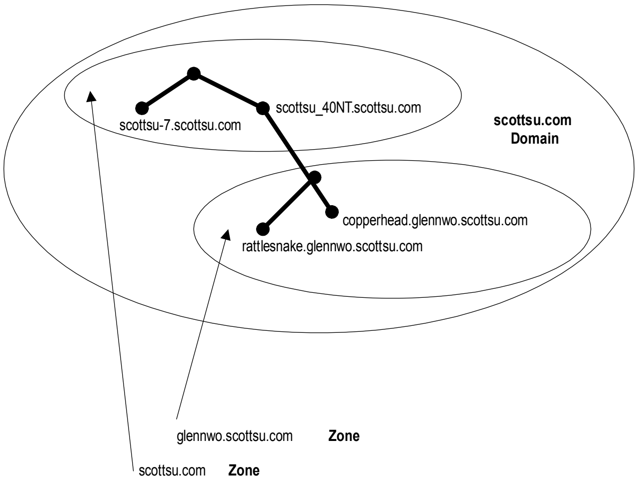
For more information

For the latest information on Windows NT Server, check out our World Wide Web site at http://www.microsoft.com/backoffice or the Windows NT Server Forum on the Microsoft Network (GO WORD: MSNTS). To access information via the World Wide Web, go to http://www.microsoft.com and select Microsoft BackOffice.

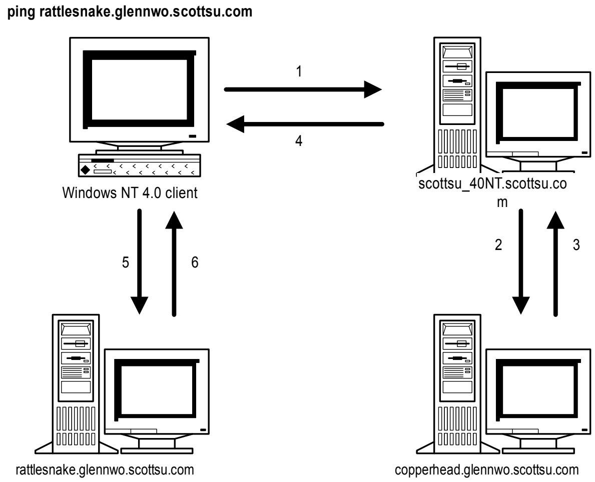
Tracing DNS Queries

Asking the DNS for an IP address of a HOST

The example consists of the following setup with four hosts, two of which are Primary DNS servers.



The Windows NT 4.0-based client does the following PING. The arrows represent a packet and the numbers associate the frame number with the trace below.


ping rattlesnake.glennwo.scottsu.com
1
4
Windows NT 4.0 client
scottsu_40NT.scottsu.com
5
6
2
3
rattlesnake.glennwo.scottsu.com
copperhead.glennwo.scottsu.com

The trace is of the following query:

Ping rattlesnake.glennwo.scottsu.com

That is the Host is rattlesnake and the domain is glennwo.scottsu.com

The client first has to query its DNS server for the names.

Note that the query is recursive.

```
1  SCOTTSU-7  SCOTTSU_NT40 DNS  0x1:Std Qry for rattlesnake.glennwo.scottsu.com

IP: ID = 0x9608; Proto = UDP; Len: 77
  IP: Version = 4 (0x4)
  IP: Header Length = 20 (0x14)
  IP: Service Type = 0 (0x0)
IP: Total Length = 77 (0x4D)
  IP: Identification = 38408 (0x9608)
  IP: Flags Summary = 0 (0x0)
IP: Fragment Offset = 0 (0x0) bytes
  IP: Time to Live = 128 (0x80)
  IP: Protocol = UDP—User Datagram
  IP: CheckSum = 0x9F28
  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
  IP: Data: Number of data bytes remaining = 57 (0x0039)
 UDP: Src Port: Unknown, (1066); Dst Port: DNS (53); Length = 57 (0x39)
  UDP: Source Port = 0x042A
  UDP: Destination Port = DNS
  UDP: Total length = 57 (0x39) bytes
  UDP: CheckSum = 0xB2D7
  UDP: Data: Number of data bytes remaining = 49 (0x0031)
 DNS: 0x1:Std Qry for rattlesnake.glennwo.scottsu.com of type Host Addr on class INET addr.
  DNS: Query Identifier = 1 (0x1)
  DNS: DNS Flags = Query, OpCode—Std Qry, RD Bits Set, RCode—No error
   DNS: 0............... = Query
   DNS: .0000........... = Standard Query
   DNS: .....0.......... = Server not authority for domain
   DNS: ......0......... = Message complete
   DNS: .......1........ = Recursive query desired
   DNS: ........0....... = Recursive queries supported by server
   DNS: .........000.... = Reserved
   DNS: ............0000 = No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 0 (0x0)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
   DNS: Question Name: rattlesnake.glennwo.scottsu.com
   DNS: Question Type = Host Address
   DNS: Question Class = Internet address class
```

The scottsu_40NT.scottsu.com is not authoritative for the domain glennwo.scottsu.com, so the scottsu_40NT.scottsu.com Host forwards the request to the subdomain DNS server copperhead.glennwo.scottsu.com.

Note that the query is iterative.

```
2  SCOTTSU_NT40 COPPERHEAD  DNS  0x5:Std Qry for rattlesnake.glennwo.scottsu.com

IP: ID = 0x9C1C; Proto = UDP; Len: 77
  IP: Version = 4 (0x4)
  IP: Header Length = 20 (0x14)
  IP: Service Type = 0 (0x0)
IP: Total Length = 77 (0x4D)
  IP: Identification = 39964 (0x9C1C)
  IP: Flags Summary = 0 (0x0)
IP: Fragment Offset = 0 (0x0) bytes
  IP: Time to Live = 128 (0x80)
  IP: Protocol = UDP—User Datagram
  IP: CheckSum = 0x9487
  IP: Source Address = 157.55.100.204
  IP: Destination Address = 157.55.106.193
  IP: Data: Number of data bytes remaining = 57 (0x0039)
 UDP: Src Port: DNS, (53); Dst Port: DNS (53); Length = 57 (0x39)
  UDP: Source Port = DNS
  UDP: Destination Port = DNS
  UDP: Total length = 57 (0x39) bytes
  UDP: CheckSum = 0xB33B
  UDP: Data: Number of data bytes remaining = 49 (0x0031)
 DNS: 0x5:Std Qry for rattlesnake.glennwo.scottsu.com of type Host Addr on class INET addr.
  DNS: Query Identifier = 5 (0x5)
  DNS: DNS Flags = Query, OpCode—Std Qry, RCode—No error
   DNS: 0............... = Query
   DNS: .0000........... = Standard Query
   DNS: .....0.......... = Server not authority for domain
   DNS: ......0......... = Message complete
   DNS: .......0........ = Iterative query desired
   DNS: ........0....... = Recursive queries supported by server
   DNS: .........000.... = Reserved
   DNS: ............0000 = No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 0 (0x0)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
   DNS: Question Name: rattlesnake.glennwo.scottsu.com
   DNS: Question Type = Host Address
   DNS: Question Class = Internet address class
```

The Copperhead.glennwo.scottsu.com Host replies back to the Scottsu_40NT DNS server with the data.

```
3  COPPERHEAD  SCOTTSU_NT40 DNS  0x5:Std Qry Resp. for rattlesnake.glennwo.scottsu.com

IP: ID = 0x5F04; Proto = UDP; Len: 93
  IP: Version = 4 (0x4)
  IP: Header Length = 20 (0x14)
  IP: Service Type = 0 (0x0)
```

```
IP: Total Length = 93 (0x5D)
  IP: Identification = 24324 (0x5F04)
  IP: Flags Summary = 0 (0x0)
IP: Fragment Offset = 0 (0x0) bytes
  IP: Time to Live = 128 (0x80)
  IP: Protocol = UDP—User Datagram
  IP: CheckSum = 0xD18F
  IP: Source Address = 157.55.106.193
  IP: Destination Address = 157.55.100.204
  IP: Data: Number of data bytes remaining = 73 (0x0049)
 UDP: Src Port: DNS, (53); Dst Port: DNS (53); Length = 73 (0x49)
  UDP: Source Port = DNS
  UDP: Destination Port = DNS
  UDP: Total length = 73 (0x49) bytes
  UDP: CheckSum = 0x8BD1
  UDP: Data: Number of data bytes remaining = 65 (0x0041)
 DNS: 0x5:Std Qry Resp. for rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
  DNS: Query Identifier = 5 (0x5)
  DNS: DNS Flags = Response, OpCode—Std Qry, AA RA Bits Set, RCode—No error
   DNS: 1............... = Response
   DNS: .0000........... = Standard Query
   DNS: .....1.......... = Server authority for domain
   DNS: ......0......... = Message complete
   DNS: .......0........ = Iterative query desired
   DNS: ........1....... = No recursive queries
   DNS: .........000.... = Reserved
   DNS: ............0000 = No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 1 (0x1)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
   DNS: Question Name: rattlesnake.glennwo.scottsu.com
   DNS: Question Type = Host Address
   DNS: Question Class = Internet address class
  DNS: Answer section: rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
   DNS: Resource Name: rattlesnake.glennwo.scottsu.com
   DNS: Resource Type = Host Address
   DNS: Resource Class = Internet address class
   DNS: Time To Live = 0 (0x0)
   DNS: Resource Data Length = 4 (0x4)
   DNS: IP address = 157.55.107.88
```

The data is returned back to the client.

```
4  SCOTTSU_NT40 SCOTTSU-7  DNS  0x1:Std Qry Resp. for rattlesnake.glennwo.scottsu.com


IP: ID = 0x9D1C; Proto = UDP; Len: 93
  IP: Version = 4 (0x4)
  IP: Header Length = 20 (0x14)
  IP: Service Type = 0 (0x0)
IP: Total Length = 93 (0x5D)
  IP: Identification = 40220 (0x9D1C)
  IP: Flags Summary = 0 (0x0)
IP: Fragment Offset = 0 (0x0) bytes
  IP: Time to Live = 128 (0x80)
  IP: Protocol = UDP—User Datagram
  IP: CheckSum = 0x9804
  IP: Source Address = 157.55.100.204
  IP: Destination Address = 157.55.102.52
  IP: Data: Number of data bytes remaining = 73 (0x0049)
 UDP: Src Port: DNS, (53); Dst Port: Unknown (1066); Length = 73 (0x49)
  UDP: Source Port = DNS
  UDP: Destination Port = 0x042A
  UDP: Total length = 73 (0x49) bytes
  UDP: CheckSum = 0x8F6D
  UDP: Data: Number of data bytes remaining = 65 (0x0041)
 DNS: 0x1:Std Qry Resp. for rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
  DNS: Query Identifier = 1 (0x1)
  DNS: DNS Flags = Response, OpCode—Std Qry, RD RA Bits Set, RCode—No error
   DNS: 1............... = Response
   DNS: .0000........... = Standard Query
   DNS: .....0.......... = Server not authority for domain
   DNS: ......0......... = Message complete
   DNS: .......1........ = Recursive query desired
   DNS: ........1....... = No recursive queries
   DNS: .........000.... = Reserved
   DNS: ............0000 = No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 1 (0x1)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
   DNS: Question Name: rattlesnake.glennwo.scottsu.com
   DNS: Question Type = Host Address
   DNS: Question Class = Internet address class
  DNS: Answer section: rattlesnake.glennwo.scottsu.com of type Host Addr on class INET
addr.
   DNS: Resource Name: rattlesnake.glennwo.scottsu.com
   DNS: Resource Type = Host Address
   DNS: Resource Class = Internet address class
   DNS: Time To Live = 0 (0x0)
   DNS: Resource Data Length = 4 (0x4)
   DNS: IP address = 157.55.107.88
```

The client can then PING the server.

```
5  SCOTTSU-7  RATTLESNAKE ICMP  Echo,  From 157.55.102.52 To 157.55.107.88

6  RATTLESNAKE SCOTTSU-7  ICMP  Echo Reply, To 157.55.102.52 From 157.55.107.88
```

The Net Use

The following is a trace of the new behavior of Windows NT 4.0 that allows a client to connect up to a Windows NT-based server via the HOST name rather than the NetBIOS computer name. This is a trace of the following command:

Net Use * \\scottsuPDC.scottsu.com\c$

Query the DNS for the Host name scottsuPDC.scotts.com

```
1  4.248 SCOTTSU-7  SCOTTSU_NT40 DNS  0x6:Std Qry for SCOTTSUPDC.SCOTTSU.COM

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
UDP: Src Port: Unknown, (1057); Dst Port: DNS (53); Length = 48 (0x30)
  UDP: Source Port = 0x0421
  UDP: Destination Port = DNS
  UDP: Total length = 48 (0x30) bytes
  UDP: CheckSum = 0x8FEA
  UDP: Data: Number of data bytes remaining = 40 (0x0028)
 DNS: 0x6:Std Qry for SCOTTSUPDC.SCOTTSU.COM of type Host Addr on class INET addr.
  DNS: Query Identifier = 6 (0x6)
  DNS: DNS Flags = Query, OpCode—Std Qry, RD Bits Set, RCode—No error
   DNS: 0............... = Query
   DNS: .0000........... = Standard Query
   DNS: .....0.......... = Server not authority for domain
   DNS: ......0......... = Message complete
   DNS: .......1........ = Recursive query desired
   DNS: ........0....... = Recursive queries supported by server
   DNS: .........000.... = Reserved
   DNS: ............0000 = No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 0 (0x0)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: SCOTTSUPDC.SCOTTSU.COM of type Host Addr on class INET addr.
   DNS: Question Name: SCOTTSUPDC.SCOTTSU.COM
   DNS: Question Type = Host Address
   DNS: Question Class = Internet address class
```

This is the response from the DNS server. The IP address for scottsuPDC.scottsu.com is at the end of the frame

```
2  4.255 SCOTTSU_NT40 SCOTTSU-7  DNS  0x6:Std Qry Resp. for SCOTTSUPDC.SCOTTSU.COM

  IP: Source Address = 157.55.100.204
  IP: Destination Address = 157.55.102.52
UDP: Src Port: DNS, (53); Dst Port: Unknown (1057); Length = 64 (0x40)
  UDP: Source Port = DNS
  UDP: Destination Port = 0x0421
  UDP: Total length = 64 (0x40) bytes
  UDP: CheckSum = 0x4932
  UDP: Data: Number of data bytes remaining = 56 (0x0038)
 DNS: 0x6:Std Qry Resp. for SCOTTSUPDC.SCOTTSU.COM of type Host Addr on class INET addr.
  DNS: Query Identifier = 6 (0x6)
  DNS: DNS Flags = Response, OpCode—Std Qry, AA RD RA Bits Set, RCode—No error
   DNS: 1............... = Response
   DNS: .0000........... = Standard Query
   DNS: .....1.......... = Server authority for domain
   DNS: ......0......... = Message complete
   DNS: .......1........ = Recursive query desired
   DNS: ........1....... = No recursive queries
   DNS: .........000.... = Reserved
   DNS: ............0000 = No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 1 (0x1)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: SCOTTSUPDC.SCOTTSU.COM of type Host Addr on class INET addr.
   DNS: Question Name: SCOTTSUPDC.SCOTTSU.COM
   DNS: Question Type = Host Address
   DNS: Question Class = Internet address class
  DNS: Answer section: SCOTTSUPDC.SCOTTSU.COM of type Host Addr on class INET addr.
   DNS: Resource Name: SCOTTSUPDC.SCOTTSU.COM
   DNS: Resource Type = Host Address
   DNS: Resource Class = Internet address class
   DNS: Time To Live = 0 (0x0)
   DNS: Resource Data Length = 4 (0x4)
   DNS: IP address = 157.55.100.204
```

Then the client does an Adapter Status on the IP address to find what names that machine has registered

```
3  4.254 SCOTTSU-7  SCOTTSU_NT40 NBT  NS: Query req. for *<00...(15)>

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
UDP: Src Port: NETBIOS Name Service, (137); Dst Port: NETBIOS Name Service (137); Length = 58 (0x3A)
  UDP: Source Port = NETBIOS Name Service
  UDP: Destination Port = NETBIOS Name Service
  UDP: Total length = 58 (0x3A) bytes
  UDP: CheckSum = 0x3A63
  UDP: Data: Number of data bytes remaining = 50 (0x0032)
 NBT: NS: Query req. for *<00...(15)>
  NBT: Transaction ID = 32860 (0x805C)
  NBT: Flags Summary = 0x0000—Req.; Query; Success
   NBT: 0............... = Request
   NBT: .0000........... = Query
   NBT: .....0.......... = Non-authoritative Answer
   NBT: ......0......... = Datagram not truncated
   NBT: .......0........ = Recursion not desired
   NBT: ........0....... = Recursion not available
   NBT: .........0...... = Reserved
   NBT: ..........0..... = Reserved
   NBT: ...........0.... = Not a broadcast packet
   NBT: ............0000 = Success
  NBT: Question Count = 1 (0x1)
  NBT: Answer Count = 0 (0x0)
  NBT: Name Service Count = 0 (0x0)
  NBT: Additional Record Count = 0 (0x0)
  NBT: Question Name = *<00...(15)>
  NBT: Question Type = Node Status Request
  NBT: Question Class = Internet Class
```

This is the response to the Adapter Status. The machine has a number of NetBIOS names registered.

```
4  4.255 SCOTTSU_NT40 SCOTTSU-7  NBT  NS: Query (Node Status) resp. for *<00...(15)>, Success

  IP: Source Address = 157.55.100.204
  IP: Destination Address = 157.55.102.52
UDP: Src Port: NETBIOS Name Service, (137); Dst Port: NETBIOS Name Service (137); Length = 345 (0x159)
  UDP: Source Port = NETBIOS Name Service
  UDP: Destination Port = NETBIOS Name Service
  UDP: Total length = 345 (0x159) bytes
  UDP: CheckSum = 0xC551
  UDP: Data: Number of data bytes remaining = 337 (0x0151)
 NBT: NS: Query (Node Status) resp. for *<00...(15)>, Success
  NBT: Transaction ID = 32860 (0x805C)
  NBT: Flags Summary = 0x8400—Resp.; Query; Success
   NBT: 1............... = Response
   NBT: .0000........... = Query
   NBT: .....1.......... = Authoritative Answer
   NBT: ......0......... = Datagram not truncated
   NBT: .......0........ = Recursion not desired
   NBT: ........0....... = Recursion not available
   NBT: .........0...... = Reserved
   NBT: ..........0..... = Reserved
   NBT: ...........0.... = Not a broadcast packet
   NBT: ............0000 = Success
  NBT: Question Count = 0 (0x0)
  NBT: Answer Count = 1 (0x1)
  NBT: Name Service Count = 0 (0x0)
  NBT: Additional Record Count = 0 (0x0)
  NBT: Resource Record Name = *<00...(15)>
  NBT: Resource Record Type = Node Status Request
  NBT: Resource Record Class = Internet Class
  NBT: Time To Live = 0 (0x0)
  NBT: RDATA Length = 263 (0x107)
  NBT: Number of Names = 12 (0xC)
  NBT: ASCII Name = SCOTTSU_NT40
  NBT: Resource Record Flags = 17408 (0x4400)
   NBT: ......0......... = Non-Permanent
   NBT: .....1.......... = Active Name
   NBT: ....0........... = Name is not in Conflict
   NBT: ...0............ = Not Deregistering
   NBT: .10............. = M Node
   NBT: 0............... = Unique NetBIOS Name
  NBT: ASCII Name = SCOTTSU_NT40 00
  NBT: Resource Record Flags = 17408 (0x4400)
   NBT: ......0......... = Non-Permanent
   NBT: .....1.......... = Active Name
   NBT: ....0........... = Name is not in Conflict
   NBT: ...0............ = Not Deregistering
   NBT: .10............. = M Node
   NBT: 0............... = Unique NetBIOS Name
```

```
NBT: ASCII Name = SCOTTSU_NT40D 00
NBT: Resource Record Flags = 50176 (0xC400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 1............... = Group NetBIOS Name
NBT: ASCII Name = SCOTTSU_NT40D <1C>
NBT: Resource Record Flags = 50176 (0xC400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 1............... = Group NetBIOS Name
NBT: ASCII Name = SCOTTSU_NT40D <1B>
NBT: Resource Record Flags = 17408 (0x4400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 0............... = Unique NetBIOS Name
NBT: ASCII Name = SCOTTSU_NT40D <1E>
NBT: Resource Record Flags = 50176 (0xC400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 1............... = Group NetBIOS Name
NBT: ASCII Name = SCOTTSU_NT40 <03>
NBT: Resource Record Flags = 17408 (0x4400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 0............... = Unique NetBIOS Name
NBT: ASCII Name = SCOTTSU_NT40D <1D>
NBT: Resource Record Flags = 17408 (0x4400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 0............... = Unique NetBIOS Name
NBT: ASCII Name = <01><02>__MSBROWSE__<02><01>
NBT: Resource Record Flags = 50176 (0xC400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 1............... = Group NetBIOS Name
NBT: ASCII Name = INet~Services <1C>
NBT: Resource Record Flags = 50176 (0xC400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 1............... = Group NetBIOS Name
NBT: ASCII Name = IS~SCOTTSU_NT4000
NBT: Resource Record Flags = 17408 (0x4400)
 NBT: ......0......... = Non-Permanent
 NBT: .....1.......... = Active Name
 NBT: ....0........... = Name is not in Conflict
 NBT: ...0............ = Not Deregistering
 NBT: .10............. = M Node
 NBT: 0............... = Unique NetBIOS Name
```

```
    NBT: ASCII Name = SCOTTSU_NT40¿¿¿¿
    NBT: Resource Record Flags = 50176 (0xC400)
     NBT: ......0......... = Non-Permanent
     NBT: .....1.......... = Active Name
     NBT: ....0........... = Name is not in Conflict
     NBT: ...0............ = Not Deregistering
     NBT: .10............. = M Node
     NBT: 1............... = Group NetBIOS Name
    NBT: Adapter Address = 00A02463AB22
    NBT: Version Major = 0 (0x0)
    NBT: Version Minor = 0 (0x0)
    NBT: Duration = 0 (0x0)
    NBT: FRMRs Received = 0 (0x0)
    NBT: FRMRs Transmitted = 0 (0x0)
    NBT: IFrame Receive Errors = 0 (0x0)
    NBT: Transmit Aborts = 0 (0x0)
    NBT: Tranmitted = 0 (0x0)
    NBT: Received = 0 (0x0)
    NBT: IFrame Transmit Errors = 0 (0x0)
    NBT: No Receive Buffers = 0 (0x0)
    NBT: T1 Timeouts = 0 (0x0)
    NBT: Ti Timeouts = 0 (0x0)
    NBT: Free NCBS = 0 (0x0)
    NBT: NCBS = 0 (0x0)
    NBT: Max NCBS = 0 (0x0)
    NBT: No Transmit Buffers = 255 (0xFF)
    NBT: Max Datagram = 799 (0x31F)
    NBT: Pending Sessions = 32 (0x20)
    NBT: Max Sessions = 33008 (0x80F0)
    NBT: Packet Size = 20976 (0x51F0)
```

Now the client is going to do the TCP 3 way handshake.

```
5  4.257 SCOTTSU-7  SCOTTSU_NT40 TCP  ....S., len: 4, seq: 2817881, ack:   0, win: 8192

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
TCP: ....S., len: 4, seq: 2817881, ack:   0, win: 8192, src: 1056 dst: 139 (NBT Session)


6  4.257 SCOTTSU_NT40 SCOTTSU-7  TCP  .A..S., len: 4, seq: 5416017, ack: 2817882, win: 8760, src SCOTTSU_NT40 SCOTTSU-7  IP

  IP: Source Address = 157.55.100.204
  IP: Destination Address = 157.55.102.52
TCP: .A..S., len: 4, seq: 5416017, ack: 2817882, win: 8760, src: 139 (NBT Session) dst: 1056


7  4.258 SCOTTSU-7  SCOTTSU_NT40 TCP  .A...., len: 0, seq: 2817882, ack: 5416018, win: 8760

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
TCP: .A...., len: 0, seq: 2817882, ack: 5416018, win: 8760, src: 1056 dst: 139 (NBT Session)
```

Now the client will send a NetBT Session Request to the NetBIOS server name (not the HOST name).

```
8  4.259 SCOTTSU-7  SCOTTSU_NT40 NBT  SS: Session Request, Dest: SCOTTSU_NT40

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
TCP: .AP..., len: 72, seq: 2817882, ack: 5416018, win: 8760, src: 1056 dst: 139 (NBT Session)
  TCP: Source Port = 0x0420
  TCP: Destination Port = NETBIOS Session Service
  TCP: Sequence Number = 2817882 (0x2AFF5A)
  TCP: Acknowledgement Number = 5416018 (0x52A452)
  TCP: Data Offset = 20 (0x14)
  TCP: Reserved = 0 (0x0000)
  TCP: Flags = 0x18 : .AP...
   TCP: ..0..... = No urgent data
   TCP: ...1.... = Acknowledgement field significant
   TCP: ....1... = Push function
   TCP: .....0.. = No Reset
   TCP: ......0. = No Synchronize
   TCP: .......0 = No Fin
  TCP: Window = 8760 (0x2238)
  TCP: CheckSum = 0x722C
  TCP: Urgent Pointer = 0 (0x0)
  TCP: Data: Number of data bytes remaining = 72 (0x0048)
 NBT: SS: Session Request, Dest: SCOTTSU_NT40 , Source: SCOTTSU-7  <00>, Len: 68
  NBT: Packet Type = Session Request
  NBT: Packet Flags = 0 (0x0)
   NBT: .......0 = Add 0 to Length
  NBT: Packet Length = 68 (0x44)
  NBT: Called Name = SCOTTSU_NT40
  NBT: Calling Name = SCOTTSU-7  <00>
```

This is a positive session response from the server.

```
9  4.259 SCOTTSU_NT40 SCOTTSU-7  NBT  SS: Positive Session Response, Len: 0

  IP: Source Address = 157.55.100.204
  IP: Destination Address = 157.55.102.52
TCP: .AP..., len: 4, seq: 5416018, ack: 2817954, win: 8688, src: 139 (NBT Session) dst: 1056
  TCP: Source Port = NETBIOS Session Service
  TCP: Destination Port = 0x0420
  TCP: Sequence Number = 5416018 (0x52A452)
  TCP: Acknowledgement Number = 2817954 (0x2AFFA2)
  TCP: Data Offset = 20 (0x14)
  TCP: Reserved = 0 (0x0000)
  TCP: Flags = 0x18 : .AP...
   TCP: ..0..... = No urgent data
   TCP: ...1.... = Acknowledgement field significant
   TCP: ....1... = Push function
   TCP: .....0.. = No Reset
   TCP: ......0. = No Synchronize
   TCP: .......0 = No Fin
  TCP: Window = 8688 (0x21F0)
  TCP: CheckSum = 0x5D4C
  TCP: Urgent Pointer = 0 (0x0)
  TCP: Data: Number of data bytes remaining = 4 (0x0004)
 NBT: SS: Positive Session Response, Len: 0
  NBT: Packet Type = Positive Session Response
  NBT: Packet Flags = 0 (0x0)
   NBT: .......0 = Add 0 to Length
  NBT: Packet Length = 0 (0x0)
```

Now we will begin to set up a session between the redirector and the server (application layer). This starts out with a negotiate frame.

```
                 10  4.262 SCOTTSU-7  SCOTTSU_NT40 SMB  C negotiate, Dialect = NT LM 0.12


                 11  4.276 SCOTTSU_NT40 SCOTTSU-7  SMB  R negotiate, Dialect # = 7
```

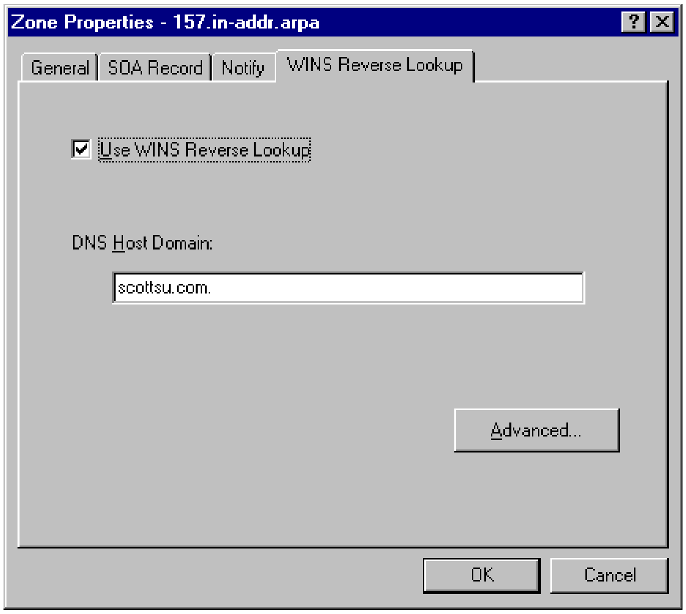
Now we are going to set up the session between the redirector and the server. This frame also has a Tree Connect attached to it. Note that the server name in the Tree Connect is the HOST name. The redirector doesn't actually use this name. It only knows about the NetBIOS computer name.

```
12  4.275 SCOTTSU-7  SCOTTSU_NT40 SMB  C session setup & X, and C tree connect

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
TCP: .AP..., len: 314, seq: 2818128, ack: 5416131, win: 8647, src: 1056 dst: 139 (NBT Session)
```

```
 TCP: Source Port = 0x0420
 TCP: Destination Port = NETBIOS Session Service
 TCP: Sequence Number = 2818128 (0x2B0050)
 TCP: Acknowledgement Number = 5416131 (0x52A4C3)
 TCP: Data Offset = 20 (0x14)
 TCP: Reserved = 0 (0x0000)
 TCP: Flags = 0x18 : .AP...
  TCP: ..0..... = No urgent data
  TCP: ...1.... = Acknowledgement field significant
  TCP: ....1... = Push function
  TCP: .....0.. = No Reset
  TCP: ......0. = No Synchronize
  TCP: .......0 = No Fin
 TCP: Window = 8647 (0x21C7)
 TCP: CheckSum = 0x5638
 TCP: Urgent Pointer = 0 (0x0)
 TCP: Data: Number of data bytes remaining = 314 (0x013A)
NBT: SS: Session Message, Len: 310
 NBT: Packet Type = Session Message
 NBT: Packet Flags = 0 (0x0)
  NBT: .......0 = Add 0 to Length
 NBT: Packet Length = 310 (0x136)
 NBT: SS Data: Number of data bytes remaining = 310 (0x0136)
SMB: C session setup & X, Username = Administrator, and C tree connect & X, Share = \\SCOTTSUPDC.SCOTTSU.COM\IPC$
 SMB: SMB Status = Error Success
  SMB: Error class = No Error
  SMB: Error code = No Error
 SMB: Header: PID = 0xCAFE TID = 0x0000 MID = 0x0000 UID = 0x0000
  SMB: Tree ID  (TID) = 0 (0x0)
  SMB: Process ID (PID) = 51966 (0xCAFE)
  SMB: User ID  (UID) = 0 (0x0)
  SMB: Multiplex ID (MID) = 0 (0x0)
  SMB: Flags Summary = 24 (0x18)
   SMB: .......0 = Lock & Read and Write & Unlock not supported
   SMB: ......0. = Send No Ack not supported
   SMB: ....1... = Using caseless pathnames
   SMB: ...1.... = Canonicalized pathnames
   SMB: ..0..... = No Opportunistic lock
   SMB: .0...... = No Change Notify
   SMB: 0....... = Client command
  SMB: flags2 Summary = 32771 (0x8003)
   SMB: ...............1 = Understands long filenames
   SMB: ..............1. = Understands extended attributes
   SMB: ..0............. = No paging of IO
   SMB: .0.............. = Using SMB status codes
   SMB: 1............... = Using UNICODE strings
 SMB: Command = C session setup & X
  SMB: Word count = 13
  SMB: Word parameters
  SMB: Next offset = 0x00E8
  SMB: Max Buffer Size = 4356
  SMB: Max MPX requests = 50
  SMB: VC number = 0
  SMB: Session Key = 0
  SMB: Password length = 24 (0x18)
  SMB: Unicode Password length = 24 (0x18)
  SMB: Capabilities = 212 (0xD4)
   SMB: ...............................0 = No Raw Reads and Writes.
   SMB: ..............................0. = No support for multiplexed commands.
   SMB: .............................1.. = Supports UNICODE strings.
   SMB: ............................0... = Does not support large files.
   SMB: ...........................1.... = Supports the NT SMB extensions.
   SMB: ..........................0..... = RPC remote API’s not supported.
   SMB: .........................1...... = Recognizes NT Status codes.
   SMB: ........................1....... = Supports level II oplocks.
   SMB: .......................0........ = Does not support Lock and Read.
  SMB: Byte count = 171
  SMB: Byte parameters
  SMB: Account name = Administrator
  SMB: Domain name = SCOTTSU_NT40D
  SMB: Native OS = Windows NT 1307
  SMB: Native Lanman = Windows NT 4.0
 SMB: Command = C tree connect & X
  SMB: Word count = 4
  SMB: Word parameters
  SMB: Next offset = 0x0000
  SMB: Disconnect flag = 0x0000
  SMB: Password length = 1 (0x1)
  SMB: Byte count = 67
  SMB: Byte parameters
  SMB: Password =
   SMB: Path name = \\SCOTTSUPDC.SCOTTSU.COM\IPC$

13  4.289 SCOTTSU_NT40 SCOTTSU-7  TCP  ...R.., len: 0, seq: 4933305, ack: 2818128, win: 0

14  4.289 SCOTTSU_NT40 SCOTTSU-7  TCP  ...R.., len: 0, seq: 3578897, ack: 2818128, win: 0

15  4.297 SCOTTSU_NT40 SCOTTSU-7  SMB  R session setup & X, and R tree connect & X, Type = IPC

16  4.479 SCOTTSU-7  SCOTTSU_NT40 TCP  .A...., len: 0, seq: 2818442, ack: 5416289, win: 8489

17  4.637 SCOTTSU-7  SCOTTSU_NT40 SMB  C tree connect & X, Share = \\SCOTTSUPDC.SCOTTSU.COM\C$

  IP: Source Address = 157.55.102.52
  IP: Destination Address = 157.55.100.204
TCP: .AP..., len: 107, seq: 2818442, ack: 5416289, win: 8489, src: 1056 dst: 139 (NBT Session)
  TCP: Source Port = 0x0420
  TCP: Destination Port = NETBIOS Session Service
  TCP: Sequence Number = 2818442 (0x2B018A)
  TCP: Acknowledgement Number = 5416289 (0x52A561)
  TCP: Data Offset = 20 (0x14)
  TCP: Reserved = 0 (0x0000)
  TCP: Flags = 0x18 : .AP...
   TCP: ..0..... = No urgent data
   TCP: ...1.... = Acknowledgement field significant
   TCP: ....1... = Push function
   TCP: .....0.. = No Reset
   TCP: ......0. = No Synchronize
   TCP: .......0 = No Fin
  TCP: Window = 8489 (0x2129)
  TCP: CheckSum = 0x99CE
  TCP: Urgent Pointer = 0 (0x0)
  TCP: Data: Number of data bytes remaining = 107 (0x006B)
 NBT: SS: Session Message, Len: 103
  NBT: Packet Type = Session Message
  NBT: Packet Flags = 0 (0x0)
   NBT: .......0 = Add 0 to Length
  NBT: Packet Length = 103 (0x67)
  NBT: SS Data: Number of data bytes remaining = 103 (0x0067)
 SMB: C tree connect & X, Share = \\SCOTTSUPDC.SCOTTSU.COM\C$
  SMB: SMB Status = Error Success
   SMB: Error class = No Error
   SMB: Error code = No Error
  SMB: Header: PID = 0xCAFE TID = 0x0000 MID = 0x0040 UID = 0x0801
   SMB: Tree ID  (TID) = 0 (0x0)
   SMB: Process ID (PID) = 51966 (0xCAFE)
   SMB: User ID  (UID) = 2049 (0x801)
   SMB: Multiplex ID (MID) = 64 (0x40)
   SMB: Flags Summary = 24 (0x18)
    SMB: .......0 = Lock & Read and Write & Unlock not supported
    SMB: ......0. = Send No Ack not supported
    SMB: ....1... = Using caseless pathnames
    SMB: ...1.... = Canonicalized pathnames
    SMB: ..0..... = No Opportunistic lock
    SMB: .0...... = No Change Notify
    SMB: 0....... = Client command
   SMB: flags2 Summary = 32771 (0x8003)
    SMB: ...............1 = Understands long filenames
    SMB: ..............1. = Understands extended attributes
    SMB: ..0............. = No paging of IO
    SMB: .0.............. = Using SMB status codes
    SMB: 1............... = Using UNICODE strings
  SMB: Command = C tree connect & X
   SMB: Word count = 4
   SMB: Word parameters
   SMB: Next offset = 0x0000
   SMB: Disconnect flag = 0x0000
   SMB: Password length = 1 (0x1)
   SMB: Byte count = 60
   SMB: Byte parameters
   SMB: Password =
   SMB: Path name = \\SCOTTSUPDC.SCOTTSU.COM\C$

18  4.654 SCOTTSU_NT40 SCOTTSU-7  SMB  R tree connect & X, Type = A:
```

Reverse Name Lookup

The following is a trace of a Reverse Name Lookup using the “PING -a IP address” command. This sends the IP address to the DNS server and asks for the HOST name associated with the NetBIOS name. Remember that if the DNS server didn't have this name in its table, the DNS server will actually do an Adapter Status on the IP Address rather than going to WINS. However, that's only if you have the following check box checked in the properties of the *in-addr.arpa zone:


Zone Properties - 157.in-addr.arpa

☑ Use WINS Reverse Lookup

DNS Host Domain: scottsu.com.

Ping -a 128.247.145.207

```
DNS: 0x1:Std Qry for 207.145.247.128.in-addr.arpa of type Dom. name ptr on class INET addr.
  DNS: Query Identifier = 1 (0x1)
  DNS: DNS Flags = Query, OpCode—Std Qry, RD Bits Set, RCode—No error
  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 0 (0x0)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: 207.145.247.128.in-addr.arpa of type Dom. name ptr on class INET
addr.
   DNS: Question Name: 207.145.247.128.in-addr.arpa
   DNS: Question Type = Domain name pointer
   DNS: Question Class = Internet address class
```

Note that the Question name is the IP address backwards 207.145.247.128.in-addr.arpa of type PTR.

```
*******************************************************************************************************************************
DNS: 0x1:Std Qry Resp. for 207.145.247.128.in-addr.arpa of type Dom. name ptr on class INET addr.
  DNS: Query Identifier = 1 (0x1)
  DNS: DNS Flags = Response, OpCode—Std Qry, AA RD RA Bits Set, RCode—No error

  DNS: Question Entry Count = 1 (0x1)
  DNS: Answer Entry Count = 1 (0x1)
  DNS: Name Server Count = 0 (0x0)
  DNS: Additional Records Count = 0 (0x0)
  DNS: Question Section: 207.145.247.128.in-addr.arpa of type Dom. name ptr on class INET addr.
   DNS: Question Name: 207.145.247.128.in-addr.arpa
   DNS: Question Type = Domain name pointer
   DNS: Question Class = Internet address class
  DNS: Answer section: 207.145.247.128.in-addr.arpa of type Dom. name ptr on class INET addr.
   DNS: Resource Name: 207.145.247.128.in-addr.arpa
   DNS: Resource Type = Domain name pointer
   DNS: Resource Class = Internet address class
   DNS: Time To Live = 86400 (0x15180)
```

APPENDIX B: MORE INFORMATION

```
DNS: Resource Data Length = 23 (0x17)
DNS: Pointer: scottsu-7.scottsu.com
```

Internet

- All IETF drafts: **http://ds.internic.net/Internet-drafts/**
- All RFCs: **http://ds2.internic.net/rfc**
- UseNet: **comp.protocols.tcp-ip.domains**
- Good DNS sites on the net: **http://www.is.co.za/dnsrd/**
- Mailing Lists:
  - **namedroppers** is the discussion forum of the DNS working group of the Internet Engineering Task Force, and is also used by the dnsind working group. The mission of dnsind is to work on extending the protocols relating to DNS to incorporate incremental zone transfers, change notification, and dynamic updates. Subscription requests should be sent to majordomo@internic.net. General DNS questions are not appropriate for this forum. Up-to-date archives are available.
  - **dns-security** is the discussion forum of the dnssec working group of the IETF. Subscription requests should be sent to dns-security-request@tis.com.

Books

- DNS and BIND

  All-in-one tutorial style DNS reference for those using BIND. Excellent for beginning to intermediate DNS administrators; lacks coverage of advanced topics.

  –1Published: October 1992, reprinted March 1993 with minor corrections.
  –2Authors: Paul Albitz and Cricket Liu
  –3Publisher: O'Reilly & Associates
  ISBN: 1-56592-010-4
- TCP/IP Network Administration

  Contains a self-contained overview of DNS as just one of the standard network services in a TCP/IP environment.

  –4Published: August 1992, reprinted January 1994 with minor corrections.
  –5Author: Craig Hunt
  –6Publisher: O'Reilly & Associates
  –7ISBN: 0-937175-82-X
- UNIX System Administration Handbook, Second Edition

  An update to Nemeth and Snyder's classic, now features Chapter 16 dedicated to DNS. Especially good for the feature comparison of currently shipping vendor variants.

  –8Published: 1995
  –9Authors: Evi Nemeth, Garth Snyder, Scott Seebass, Trent R. Hein

APPENDIX C: GETTING REGISTERED WITH THE NIC

–10Publisher: Prentice Hall

–11ISBN: 0-13-151051-7

- TCP/IP Illustrated, Volume 1

  –12For network and protocol issues, see Chapter 14.

  –13Published: 1994

  –14Author: W. Richard Stevens

  –15Publisher: Addison-Wesley Publishing Company

  –16ISBN: 0-201-63346-9

- Firewalls and Internet Security: Repelling the Wily Hacker

  –17Includes some advice on DNS in a firewall environment.

  –18Published: 1994

  –19Authors: William R. Cheswick and Steven M. Bellovin

  –20Publisher: Addison-Wesley Publishing Company

  –21ISBN: 0-201-63357-4

- Building Internet Firewalls

  Good all-round security book with a practical approach. Chapter 8 includes a detailed section on DNS in a firewall environment.

  –22Published: September 1995

  –23Authors: D. Brent Chapman and Elizabeth D. Zwicky

  –24Publisher: O'Reilly & Associates

  –25ISBN: 1-56592-124-0

Whitepapers

Microsoft Windows NT 3.5/3.51: TCP/IP Implementation Details (Part No. 098-62170)

DHCPWINS.DOC (Part No. 098-56544)

Courses

The Domain Name System (DNS) & Internet Naming and Directory Services—by Dr. Paul Mockapetris

Miscellaneous

DNSAdmin.HLP

Domain-request@uunet.uu.net (703) 876-5050
CIC@sh.cs.net (617) 873-2777
domains%bitnic.bitnet@cunyvm.cuny.edu

Send e-mail to listserv%bitnic.bitnet@cunyvm.cuny.edu
with text "SEND DOMAIN GUIDE"
This file contains examples and a template along with explanations.

Domain registration form

APPENDIX D: DNS RECORDS

FTP://rs.internic.net/templates/domain-template.txt

- A—The A (address) resource record maps a host (computer or other network device) name to an IP address in a DNS zone. Its counterpart, the PTR resource record, is used to map an IP address to a host (computer or other network device) name in a DNS reverse zone (those in the In-addr.arpa DNS domain).
- AFSDB—The AFSDB resource record gives the location of either an AFS (Andrew File System) cell database server, or a DCE (Distributed Computing Environment) cell's authenticated name server. Transarc's AFS is a network file system, similar to NFS, but designed more for wide-area networks (WANs). 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: mapping the DNS domain name of a DCE cell to authenticated name servers for that cell.
- CNAME—The CNAME (canonical name) resource record creates an alias (synonymous name) for the specified host (computer or other network device) name. You can use CNAME records to hide the implementation details of your network from the clients that connect to it. For example, Ftp.microsoft.com is an alias (CNAME) for the real name of the computer that runs the FTP server for Microsoft. Clients connect to Ftp.microsoft.com without regard for the real name of the computer. This also allows the FTP server to be moved to a different computer; only the CNAME record needs to change.
- HINFO—The HINFO (host information) resource record identifies a host's (computer or other network device) hardware type and operating system. The CPU Type and Operating System identifiers should come from the MACHINE NAMES and SYSTEM NAMES listed in RFC 1700 (Assigned Numbers).
- ISDN—The ISDN (Integrated Services Digital Network) resource record is a variation of the A (address) resource record. Rather than mapping a host (computer or other network device) name 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 code, an area code or country code, a local phone number, and, optionally, a subaddress. The ISDN resource record is designed to be used in conjunction with the RT (route through) resource record.
- MB—The MB (mailbox) resource record is an experimental record that specifies a DNS host (computer or other network device) with the specified mailbox. Other related experimental records are the MG (mail group) resource record, the MR (mailbox rename) resource record, and the MINFO (mailbox information) resource record.
- MG—The MG (mail group) 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 (mailbox) resource record, the MR (mailbox rename) resource record, and the MINFO (mailbox information) resource record.

- MINFO—The MINFO (mailbox information) 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 (mailbox) resource record, the MG (mail group) resource record, and the MR (mailbox rename) resource record.
- MR—The MR (mailbox rename) resource record is an experimental record that specifies a mailbox that is the proper rename of the other specified mailbox. Other related experimental records are the MB (mailbox) resource record, the MG (mail group) resource record, and the MINFO (mailbox information) resource record.
- MX—The MX (mail exchanger) resource record specifies a mail exchange server for a DNS domain name. A mail exchange server is a host (computer or other network device) 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 Message Transfer Protocol (SMTP) to another mail exchange server that is closer to the final destination, or queuing it for a specified amount of time.
- NS—The NS (name server) resource record identifies the DNS name server(s) for the DNS domain. NS resource records appear in all DNS zones and reverse zones (those in the In-addr.arpa DNS domain).
- PTR—The PTR (pointer) resource record maps an IP address to a host (computer or other network device) name in a DNS reverse zone (those in the In-addr.arpa DNS domain). Its counterpart, the A (address) resource record, is used to map a host (computer or other network device) name to an IP address in a DNS zone.
- RP—The RP (responsible person) resource record indicates who is responsible for the specified DNS domain or host (computer or other network device). You can specify multiple RP records for a given DNS domain or host. The record has two parts: an electronic mail address (in the same DNS format as the one in the SOA resource record), and a DNS domain name that points to additional information about the contact.
- RT—The RT (route through) resource record specifies an intermediate host (computer or other network device) 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.
- SOA—The SOA (start of authority) resource record indicates that this DNS name server is the best source of information for the data within this DNS domain. It is the first record in each of the DNS database files. The SOA

resource record is created automatically by DNS Manager when you create a new DNS zone.

- TXT—The TXT (text) resource record associates general textual information with an item in the DNS database. A typical use is for identifying a host's (computer or other network device) location (for example, Location: Building 26S, Room 2499). The text string must be less than 256 characters, but multiple TXT resource records are allowed.
- WKS—The WKS (well-known service) 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 PROTOCOLS file (\%SystemRoot%\system32\drivers\etc\protocol). The services are the services below port number 256 from the SERVICES file (\%SystemRoot%\system32\drivers\etc\services).
- X25—The X25 (X.25) resource record is a variation of the A (address) resource record. Rather than mapping a host (computer or other network device) name to an IP address, the X25 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 X25 resource record is designed to be used in conjunction with the RT (route through) resource record.
- Generic Record—The generic resource record is used to add a non-standard resource record to the DNS database. The generic resource record is a feature of Microsoft DNS Manager.