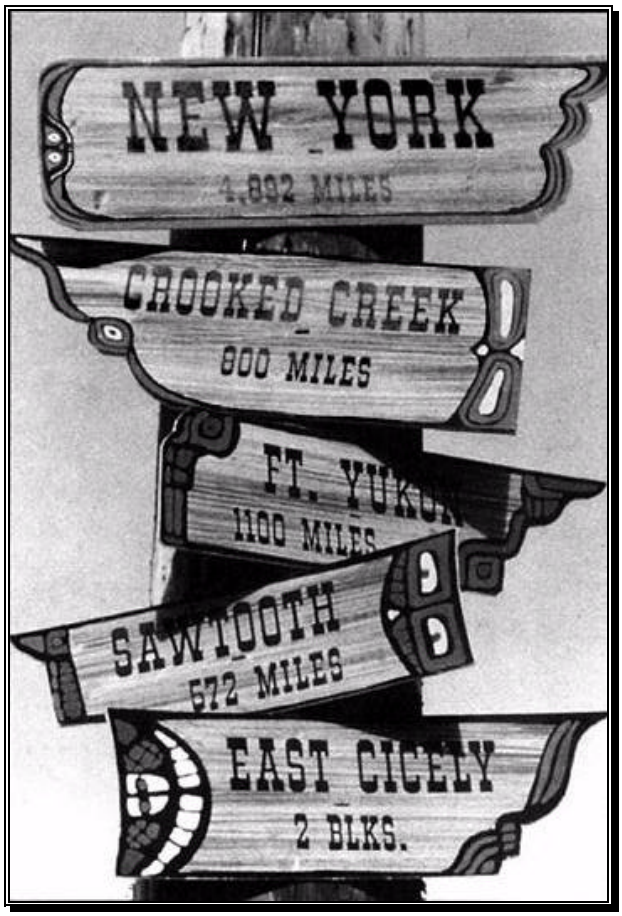
DNS

(Domain Name Services)



The road sign from "Northern Exposure"

Written February, 1996 by Gary A. Donahue

lordgad@planet.net
Comments welcome

What is DNS anyway?

DNS stands for Domain Name System. Simply put, Computers like to speak to each other using numbers, and humans prefer words. DNS is a way of letting us request complicated address information from systems, using simple names instead of the internal numbers.

For example,

netman.compucom.com is easier to remember than 198.177.254.196

DNS: Who needs it?

DNS is needed anytime you wish to access a device over a network using TCP/IP, and you know only the name of the device as opposed to its IP address. You can also use DNS to get the name of a device, using its IP address. This is called reverse name lookup, and we'll look at that a little later.

The Internet: A Brief History

Remember the ARPANET? No? Not to worry.

Before there was the Internet, there was the ARPANET. The ARPANET was originally an experiment started by the Department of Defense's Advanced Research Projects Agency, or ARPA (Later changed to DARPA). This super network was designed so that the government could share valuable computing resources. In the 80's, more and more networks connected to the ARPANET, making what was originally a network of only a handful of hosts, a network of tens of thousands.

In 1988, DARPA decided the ARPANET experiment was over, and dismantled it. The National Science Foundation built another network in its place called NSFNET. Today this network is the backbone of the Internet.

Today the Internet connects hundreds of thousands of hosts and is growing every day by a staggering amount. The current bandwidth of the NSFNET is 45 megabits per second. (As a comparison, twisted pair Ethernet is 10mbps, and fast Ethernet is 100mbps!) The original ARPANET sported a bandwidth of only about 50 kilobits per second. Almost 1000 times slower!

A Brief Note on TCP/IP Addresses

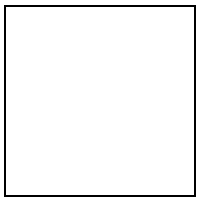
IP addresses are written in what is called *dotted octet* format. This the common method of displaying a 32 bit address as four 8 bit numbers separated by a dot. As each number between the dots is 8 bit, they can be any number between 0 and 255. Sometimes these addresses are written in hex format, however here we will stick to decimal.

DNS: How do we use it?

DNS works in a hierarchical structure, much the way the UNIX or DOS file structure works. The symbols used for the root and branches however, are different. At the very top of the tree, you have the root, which is symbolized by the null character. Sometimes this is symbolized by a single dot. This is in fact a separator followed by a null. A DNS address name is given with each node in the string separated by a dot, starting with the least significant node up to and including the top level domain under the root.

For example;

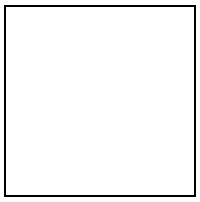
> The user djump
> in the domain ncc
> which is under the domain netman
> which is under the domain compucom
> which is under the top level domain com
> which is under the root domain
>
> would be written djump.ncc.netman.compucom.com.
>
> Note that each name is a node. Also note that all names except djump are domain names. They could also be called subdomain names as they are all contained within higher domains (.com is under the root domain!)
>
> Note also that the name string ends with a period. This is actually followed by a null character, which of course you cannot see. This last dot (and subsequent null) represents the root of the tree.



DNS: How does it work?

DNS is a wonderful thing. The problem is, who keeps track of all these names and how they relate to all these addresses? In the days of the ARPANET, the Stanford Research Institute (SRI) maintained a file containing all of the hosts on the net and their corresponding address. In the days of only a hundred hosts, this was not a big deal. As the net grew, however, this file (hosts.txt) became very large and cumbersome. Another issue is that since this file was created in one place, all other systems on the net would have to download it on a periodic basis, to keep their database current with SRI's. This not only lead to potential delays in information reaching the hosts, but a bottleneck in the net as 10,000 systems would try to download from the SRI machine in one day.

To alleviate all of these headaches, DNS was invented. DNS takes the responsibility of name/address resolution and places it on each individual domain. All the top level has to do is look at the name you give it and send it on the first leg of its journey. By the same token, no longer must we rely on other systems to supply us with table updates after a change. Since each domain updates its own tables, changes are reflected as soon as they are made.



Lets follow the steps of name resolution using the above example.

aeinstien.dev.nasa.gov pings djump.ncc.netman.compucom.com (It could happen!)
note: each query is to locate djump.ncc.netman.compucom.com
aeinsteins name server is dev.nasa.gov

QUERY	**RESPONSE**
aeinstein looks to dev.nasa.gov	Not Found: Look to .com
dev.nasa.gov looks to the root server	name server for .com is at 111.111.111.111
dev.nasa.gov looks to 111.111.111.111	name server for compucom.com is at 222.222.222.222
dev.nasa.gov looks to 222.222.222.222	name server for netman.compucom.com is at 333.333.333.333
dev.nasa.gov looks to 333.333.333.333	name server for ncc.netman.compucom.com is at 444.444.444.444
dev.nasa.gov looks to 444.444.444.444	djump is at address 444.444.444.555

dev.nasa.gov tells aeinstein the direct address for djump.ncc.netman.compucom.com.!

NOTE: *Some of these IP addresses are obviously fake. Real IP addresses numbers are 8bit and as such cannot exceed 255 in decimal, i.e. 198.177.254.196*

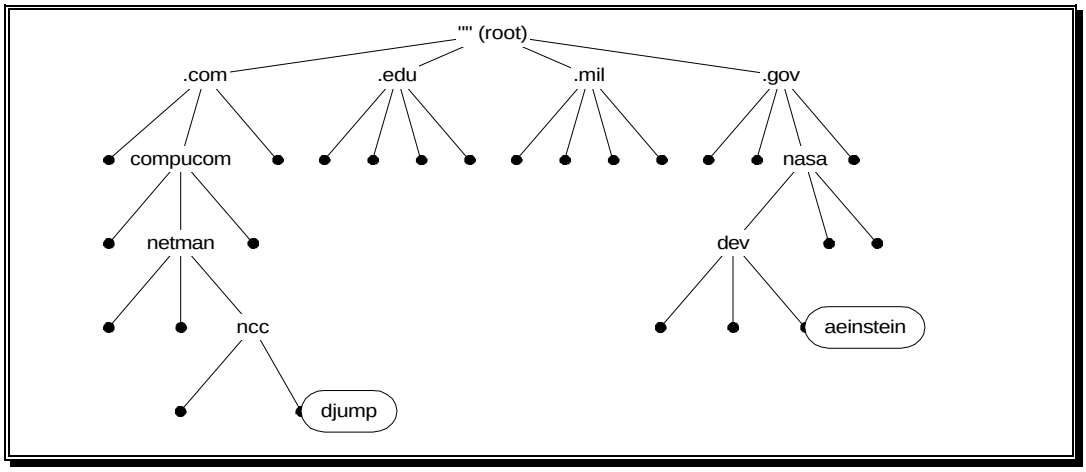
Caching

In the example we just gave, you can see that the name resolution process can be quite extensive. The name server dev.nasa.gov has to do a lot of work, repeatedly narrowing the search to find domains closer to the target host.

To help alleviate the load on the name servers, they do something called caching. When a name server successfully locates another name server, it records the name servers address, and also what domains or zones it has authority over. The next time a request is made to that domain, the local name server does not have to start at the root and work its way down. Instead, having a closer match already stored, it goes directly to the closest match and takes it from there.

The drawback with this method (and with all caches) is that the information in the cache may not match the actual data. If actual tables somewhere else on the net get updated, these changes are not updated in all the caches around the world.

To help minimize the use of out of date data in a cache, data in the cache is assigned a time to live (TTL). After the time to live has expired, that item is deleted from the cache. When a match is no longer found in the cache, data must be again retrieved the old fashioned way. If the TTL is high, then performance is very good, but accuracy over time suffers. Inversely if the TTL is low, then accuracy is high, but performance may suffer from having to do more lookups.



Using the same example, lets look at name resolution after the first request was made, and the resulting name server entries are cached on the dev.nasa.gov server.

aeinstien.dev.nasa.gov pings djump.ncc.netman.compucom.com

QUERY	**RESPONSE**
aeinstein looks to dev.nasa.gov	ncc.netman.compucom.com found in cache! IP address for this server is 444.444.444.444
dev.nasa.gov looks to 444.444.444.444	djump is at address 444.444.444.555

dev.nasa.gov tells aeinstein the direct address for djump.ncc.netman.compucom.com.!

NOTE: *These IP addresses are obviously still fake.*

As you can see, there are considerable less steps involved now that the ncc.netman.compucom.com server address is in dev.nasa.gov's cache! No longer does dev.nasa.gov have to start at the root and keep looking. (At least until TTL expires.)

Reverse name lookup

Say you have an IP address, and you wish to know the host name associated with it. Well in UNIX you would use nslookup. But how does nslookup get this information? Through an inverse query. Inverse queries are also used for some authentication processes across the net.

Think about how the DNS structure is designed. Starting at the root, a tree grows downward branching further and further until the desired domain and host names are found. Using a table in the local domain, that name is then mapped to a given address.

To find a host name given an address would require an exhaustive search of just about every domain in the tree! No small task considering the number of domains out there!

The solution is rather ingenious. What if every domain in the tree also had a table that mapped IP addresses to their given host names? If these addresses were also somehow in their own domain, shared by all systems on the Internet, then it would be a very simple thing to find a given IP address anywhere on the net!

In reality, there is a domain called “in-addr.arpa”. Each DNS name server then has a DNS db file that maps the addresses within its domain to it associated host name. This is basically the reverse of the host table. A sample reverse table may be viewed later in this document.

Each node in the table is named after the IP address of the node, in dotted octet notation. Note however that in this table, the IP address appears backwards! This is because it is not truly an address but a name. Remember that IP addresses are given in most to least significant order, and host names are given in least to most significant order. When we make the IP address a name, we reverse the order, thus allowing us to use existing name hierarchy methods.

For example (See example files later in this document for an explanation of these records):

In the host table for allerion.com we have the entry:

gdonahue	IN	A	198.177.251.50
	IN	HINFO	Gary donahue’s PC

In the reverse table for the 198.177.251 network we have the entry:

50.251.177.198.in-addr.arpa	IN	PTR	gdonahue.allerion.com.

DNS Related Files

How does a name server know that 198.177.251.50 = gdonahue.allerion.com? Database files of course! All data regarding DNS is stored in a set of files which are pointed to by the file named.boot. This is a BIND default file name and can be changed on the startup of the DNS daemon (named) with a command line. The examples shown here are taken from the actual database files of the name server for allerion.com. The host name for allerion.com's name server is linus.allerion.com.

/etc/named.boot

```
directory /etc

cache       .                          named.ca
primary     allerion.com                       named.hosts
primary     netman.compucom.com                named.netman.hosts
primary     0.0.127.in-addr.arpa       named.localhost
primary     251.177.198.in-addr.arpa   named.reverse.251
primary     252.177.198.in-addr.arpa   named.reverse.252
primary     253.177.198.in-addr.arpa   named.reverse.253
primary     254.177.198.in-addr.arpa   named.reverse.254
primary     242.104.192.in-addr.arpa   named.reverse.242
```

This file points to the location of all other database files for use by named (The daemon that controls DNS).

The first line indicates that the files are to be located in the /etc directory.

The remaining lines are comprised of three fields, the work "primary", the domain that the server is authoritative for, and the file to reference for that domain.

The cache line indicates that the file for root name services information is named.ca.

You'll notice that the next two lines seem to indicate that this name server controls TWO domains. The plan is to make allerion.com become netman.compucom.com. While the transition takes place, we though it would be a good idea to still let people access using the old domain name. What makes this interesting is that while this domain is allerion.com, it is ALSO netman.compucom.com!

All the lines containing numbers relate to reverse name lookup. For example, if you request the name for address 198.177.251.50, then this table tells named to look to the file named.reverse.251 (see reverse name lookup section)

The word "primary" indicates that this is the primary DNS server for the listed domain.

The second record in each primary line also provides the origin for that lines db file. The origin is the domain name that is automatically appended to host names not ending with a period for that file.

/etc/named.localhost

```
0.0.127.in-addr.arpa.   IN   SOA  linus.allerion.com.  jbriar.allerion.com. (
            32          ; Serial
            3600        ; Refresh
            300         ; Retry
            3600000     ; Expire
            14400 )     ; Minimum

            IN     NS     linus.allerion.com.
1           IN     PTR    localhost.allerion.com.

1.0.0.127.in-addr.arpa.          IN     PTR     localhost.
```

This file is needed so that lookups from this name server to itself functions properly. Since no one has responsibility for the 127 domain, it is up to each host to maintain it. 127.0.0.1 is the address of the local host. Every system has this file or something similar. For a description of the SOA record, see the named.hosts file information.

/etc/named.ca

```
.                       99999999    IN    NS    ns.psi.net.
ns.psi.net.             99999999    IN    A     192.33.4.10
.                       99999999    IN    NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.     99999999    IN    A     198.41.0.4
.                       99999999    IN    NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.     99999999    IN    A     128.9.0.107
.                       99999999    IN    NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.     99999999    IN    A     192.33.4.12
.                       99999999    IN    NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.     99999999    IN    A     128.8.10.90
.                       99999999    IN    NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.     99999999    IN    A     192.203.230.10
.                       99999999    IN    NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.     99999999    IN    A     39.13.229.241
.                       99999999    IN    NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.     99999999    IN    A     192.112.36.4
.                       99999999    IN    NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.     99999999    IN    A     128.63.2.53
.                       99999999    IN    NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.     99999999    IN    A     192.36.148.17
```

This file is not truly a cache as the file would imply. Originally this information was loaded into the named cache, hence the name. Now the file contains hints as to the location of the root name servers. The “.” refers to the root domain.

This table must be manually updated. The information can be gotten from the inter-NIC, and is update locally by the administrator.

The 99999999’s are a TTL (time to live) indicator. 99999999 means a very long time. This is in fact not used anymore. Looks cool in the tables though…

/etc/named.hosts
(some data removed in the interest of space)

```
@    IN   SOA  linus.allerion.com.     jbriar.allerion.com.
               (
                 32             ; Serial
                 3600           ; Refresh
                 300            ; Retry
                 3600000        ; Expire
                 14400 )        ; Minimum

               IN     NS     allerion.com.
;              IN     MX     10     netmanwall.allerion.com.
               IN     MX     10     linus.allerion.com.
;
ncc.allerion.com.   IN        NS        ncc6k.ncc.allerion.com.
;
lab.allerion.com.   IN        NS        labdns.lab.allerion.com.
;
localhost           IN        A         127.0.0.1
news                IN        CNAME     linus.allerion.com.
;
;--------------------[ Production Ethernet ]----------------------
cisco_eth1          IN        A         198.177.251.1
                    IN        HINFO     cisco_eth1 interface
gdonahue            IN        A         198.177.251.50
                    IN        HINFO     Gary Donahue
wcampbel            IN        A         198.177.251.69
                    IN        HINFO     Warren Campbell
;
;-----------------------[ Production Token Ring ]----------------
cisco_tok0          IN        A         198.177.254.33
                    IN        HINFO     cisco token ring 0
jbriar              IN        A         198.177.254.34
                    IN        HINFO     John Briar x3340
;
;-----------------------[ Stuff in the NCC ]--------------------
ncc_router                    IN        A       192.104.242.1
ncc6k.ncc.allerion.com.       IN        A       192.104.242.2
;
;-----------------------[ LAB DNS Machine ]---------------------
labdns.lab.allerion.com.      IN        A       198.177.252.69
```

This file has been drastically reduced in size. The actual file is about three pages long.

Looking at the file from the top down we see:

The SOA record Start Of Authority. You will find this record at the top of most of the DNS files. Note that in this file, the first character is a “@”, but in localhost, it is “0.0.127.in-addr.arpa.” The “@” is an abbreviation which means the domain name is the same as the origin. The parentheses allow the data to span more than one line. IN stands for Internet. The first name after the SOA, is the host where this file was created on. The next name is the host or mail address of the person responsible for the data.

Within the SOA record we see some additional line entries.

32	**; Serial**	This is version # 32 of the file
3600	**; Refresh**	Refresh after 3600 seconds
300	**; Retry**	Retry after 300 seconds
3600000	**; Expire**	Expire after a long time
14400)	**; Minimum**	Time to Live

NS Records Name Server. There will be one NS entry for every name server in this domain. These records point to the next level down in the DNS tree.

A Records Address Records. For every host name we wish to associate with an IP address, we enter an A record containing the host name and its corresponding IP address. Note that some host names have a period after them and some do not. If a host name is given WITH a period, it is considered an absolute address. If there is no period, the local domain is appended to the host name automatically.

HINFO Records Host INFOrmation. According to the specs, this is supposed to be two strings which supply information regarding the hosts hardware type and operating system. As you can see, that is not the case here. Each string is also supposed to be in quotes if it contains a space. Alas, another rule broken…

CNAME Records Canonical NAME. Note the host name “news”. The CNAME records makes an alias to another name. Whenever the host name news is referenced, it will be forwarded to linus.allerion.com.

Other acceptable types of records not seen here include;

TXT Text information

WKS Well known services

PTR Pointer. Used for address to name mapping (see reverse tables)

/etc/named.reverse.251
(some data removed in the interest of space)

```
@   IN  SOA  linus.allerion.com.  jbriar.allerion.com. (
             32              ; Serial
             3600            ; Refresh
             300             ; Retry
             3600000         ; Expire
             14400 )         ; Minimum

             IN     NS      linus.allerion.com.

1            IN     PTR     cisco_eth1.allerion.com.
2            IN     PTR     natprobe.allerion.com.
5            IN     PTR     dgopal.allerion.com.
6            IN     PTR     mvolikas.allerion.com.
7            IN     PTR     igurler.allerion.com.
10           IN     PTR     bkalish.allerion.com.
16           IN     PTR     allerion1.allerion.com.
18           IN     PTR     jdimuzio.allerion.com.
44           IN     PTR     apollo.allerion.com.
45           IN     PTR     dkeilty.allerion.com.
47           IN     PTR     jcort.allerion.com.
48           IN     PTR     jbauer.allerion.com.
49           IN     PTR     panger.allerion.com.
50           IN     PTR     gdonahue.allerion.com.
51           IN     PTR     gad.allerion.com.
55           IN     PTR     rperez.allerion.com.
60           IN     PTR     isbuild.allerion.com.
69           IN     PTR     wcampbel.allerion.com.
```

The reason that this file is the reverse lookup file for the 198.177.251 network is that the named.boot file says it is! (Take a look, you'll see!)

Note that the leftmost entries DO NOT contain the whole in-addr.arpa name! This is a shortcut. Remember that sometimes the period at the end of the name is important? Well if that period is left off, then the origin name is appended to the name given. In this case the origin (as given in named.boot) is 251.177.198.in-addr.arpa! Note also that the host names on the right DO have the period. The origin of this file is again, 251.177.198.in-addr.arpa. We wouldn't want that to be appended to gdonahue, as gdonahue's domain is allerion.com.

Note that the first line of this file is an SOA (Start Of Authority) record. The @ indicates that the origin should be inserted here in its place. Another shortcut. We administrators hate to type…

The next files show the reverse lookup tables for other networks belonging to allerion.com.

/etc/named.reverse.252

```
allerion.com.   IN   SOA  linus.allerion.com.  jbriar.allerion.com. (
              32            ; Serial
              3600          ; Refresh
              300           ; Retry
              3600000       ; Expire
              14400 )       ; Minimum

252.177.198.in-addr.arpa.            IN     NS     labdns.lab.allerion.com.
```

/etc/named.reverse.253

```
allerion.com.   IN   SOA  linus.allerion.com.  jbriar.allerion.com. (
              32            ; Serial
              3600          ; Refresh
              300           ; Retry
              3600000       ; Expire
              14400 )       ; Minimum

253.177.198.in-addr.arpa.            IN     NS     labdns.lab.allerion.com.
```

Note the subtle differences in these files compared with the previous example. The first character in the file is not a "@", but rather the full domain name. Also the IP addresses on the left side are given in their canonical file names (The period at the end indicates this)

The biggest change in these files is the fact that there are NS records! The reason for this is that this Name server does not keep the database files for these domains! In order to get address to name resolution for say, 12.253.177.198.in-addr.arpa, this record tells us to look to the name server labdns.lab.allerion.com! Both of these examples are for subdomains under allerion.com.

/etc/named.reverse.254

```
@       IN   SOA  linus.allerion.com.  jbriar.allerion.com. (
                  32            ; Serial
                  3600          ; Refresh
                  300           ; Retry
                  3600000       ; Expire
                  14400 )       ; Minimum

                  IN     NS     linus.allerion.com.

;development token ring
33                IN     PTR    cisco_tok0.allerion.com.
34                IN     PTR    jbriar.allerion.com.
35                IN     PTR    slma.allerion.com.
36                IN     PTR    tjpe.allerion.com.
37                IN     PTR    development.allerion.com.

;development Ethernet
98                IN     PTR    carson.allerion.com.
99                IN     PTR    dave.allerion.com.
100               IN     PTR    mary.allerion.com.
101               IN     PTR    cms.allerion.com.

; [----NCC: Moved to new segment ----]
;
; secure internet segment
;
161               IN     PTR    cisco.allerion.com.
162               IN     PTR    ccgate.allerion.com.
167               IN     PTR    netmanwallin.allerion.com.
;
; UNsecure internet segment
193               IN     PTR    mstar.allerion.com.
196               IN     PTR    linus.allerion.com.
197               IN     PTR    knowitall.allerion.com.
198               IN     PTR    pubsrvr.allerion.com.
222               IN     PTR    netmanwall.allerion.com.
```

As you may have guessed, this name server IS responsible for this network. An interesting note here is that this network is divided into subnets. DNS doesn't really care about this, but we do, so there have been many comments added to help us understand what is going on.

/ etc/named.reverse.242

```
allerion.com.   IN   SOA  linus.allerion.com.  jbriar.allerion.com. (
                32          ; Serial
                3600        ; Refresh
                300         ; Retry
                3600000     ; Expire
                14400 )     ; Minimum

242.104.192.in-addr.arpa.           IN     NS      ncc6k.ncc.allerion.com.
```

This is yet another example of a network that allerion.com owns, but does not do DNS for. All requests are passed on to ncc6k.ncc.allerion.com (as per the NS record)

Terminology

domain name space	The whole tree of domain names, from the least significant node all the way up to the root
domain	A domain is simply a branch of the domain name space. The name of the domain is the same as the name of the top most node in the branch. A domain contains information regarding all the hosts within the domain.
subdomain	A subdomain is a branch under a domain. In reality, the terms subdomain and domain can be used almost interchangeably. For example, compucom is a domain unto itself. It is also a subdomain of the domain .com! The domain “root” is the only domain that is NOT a subdomain!
host	A device which is pointed to in a domain. A host may be the beginning of a new subdomain.
root	The absolute top of the domain name space tree. This is analogous to the root directory in UNIX or DOS.
node	Each unit of data in the domain name space
sibling node	any node that resides under another node. For example compucom is a sibling node of the node com (compucom.com).
top level domain	The highest level of the domain name space under the root. (.com, .edu, .mil etc.)
absolute domain name	The complete domain name from the host all the way to (and including) the root
fully qualified domain name	(FQDN) Same as absolute domain name
canonical name	A Fully qualified domain name.
name server	This is the device that contains the tables and programs regarding a zone’s host addresses. The name server has authority for the zone.
zone	the area of the DNS name space controlled by a name server. This is not necessarily the whole domain, as some subdomains can have their own name servers.
root name server	The root name server contains information pointing to the name servers responsible for the top level domains (In the US, the root name servers ARE responsible for this information). These servers are very important to the operation of the Internet. Because of this, there are several spread across the net. (These servers are queried as much as 20,000 times an hour!)
resolver	The client that requests name resolution from a name server

stub resolver	a resolver that off-loads most of the burned of finding an answer to a query to the name server. Most resolvers are stub resolvers.
name resolution	The act of resolving the associated IP address from a given host name