Chapter 14 – Domain Name System

Evolution of DNS

Prior to DNS
hosts.txt was maintained by the NIC at SRI
hosts.txt −−> All hosts −−−> filter −−−> /etc/hosts

Fatal problems with hosts.txt

Traffic and load (downloaded once a week or so by all hosts)
Flat file structure made name collision mgmt difficult.
Consistency (Rapid changes to the file meant most copies were always out of date)

The solution: DNS

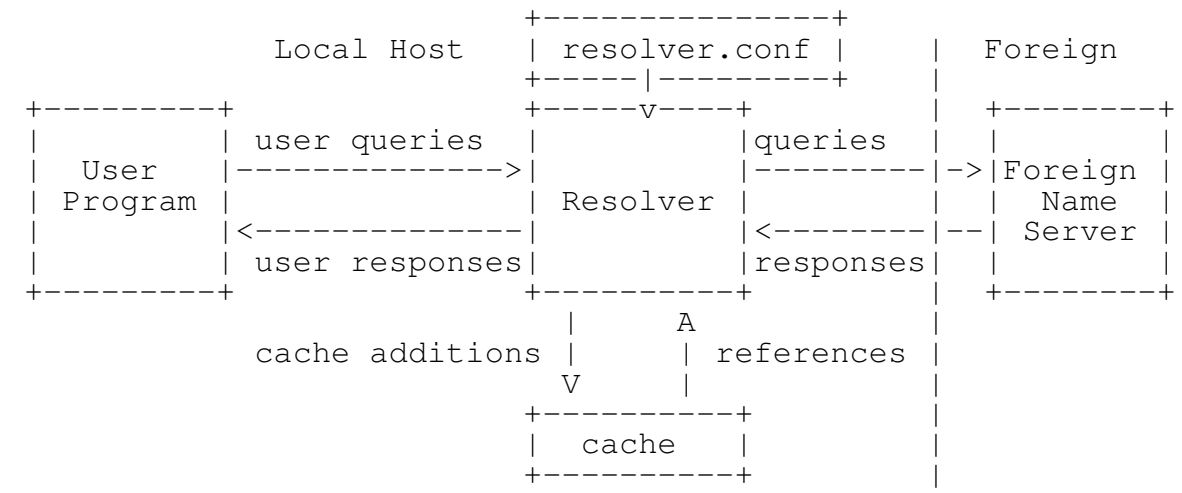
A distributed database used to map
    Host names to IP addresses
    IP addresses to host names
NO single site required to keep all the names
Distributed –> most name are kept near where they're needed
InterNIC assigns domain names –> no collisions
Local administrators system names –> low bureaucracy overhead
Load / consistency tradeoff via caching policies

DNS structure

The domain name system is a tree with an unnamed root
Top level domains include: com, edu, gov, int, mil, net, org, ae, au, ... , us, zw
Names that do end with period are viewed as fully qualified
    jmw.cs.clemson.edu.
Names that don't end with a period are often supplied with the rest of the current domain
    plato
Names that begin with a period often refer to a particular *zone* .cs.clemson.edu

Access to DNS via the *resolver* library

Applications access DSN through a *resolver*



(Caching at the resolver is non standard –– but caching at the *local name server* is standard)

Two functions provide the *high level* interface to the resolver.

```
struct hostent *gethostbyname(
char *name)

struct hostent *gethostbyaddr(
char *addr,
int  len,
int  type)
```

```
DESCRIPTION
     gethostent, gethostbyname, and gethostbyaddr() each return a
     pointer to an object with the following structure containing
     the broken-out fields of a line in  the  network  host  data
     base, /etc/hosts.  In the case of gethostbyaddr(), addr is a
     pointer to the binary format address of length  len  (not  a
     character string).

     struct    hostent {
        char *h_name;       /* official name of host      */
        char **h_aliases;   /* alias list                 */
        int  h_addrtype;    /* address type               */
        int  h_length;      /* length of address          */
        char **h_addr_list; /* addr list from name server */
     };
     #define  h_addr   h_addr_list[0] /* address, for backward compatiblity */
```

Most common implementation of the DNS is called BIND (Berkeley Internet Name Domain)
The name server is called `named`.

DNS basics

A zone is a separately administered subtree of the DNS tree
    .clemson.edu is an example

Zones may subdivide themselves giving up authority to sub−zones
    .cs.clemson.edu
    .ces.clemson.edu

The owner of the zone

    must provide a primary name server for the zone and
    and provide one or more secondary name servers
    Primary name servers
        obtain their data from a disk resident database
        each host is *manually* added system admin to the database
        primary name servers can be "ordered" to refresh themselves when the DB is updated

    Example of a host entry in a DNS database

```
; @(#)cs.clemson.zone    1.271     9/8/95
;  Authoritative data for cs.clemson.edu (ORIGIN assumed
;       cs.clemson.EDU)
  :
```

    Many lines deleted

```
  :
jmw             IN      A       130.127.48.24
                IN      MX      0 jmw.cs.clemson.edu.
                IN      MX      10 cs.clemson.edu.
                IN      HINFO   SUN-4/75 SunOS4.1.1
                IN      WKS     130.127.48.24  TCP TELNET FTP SMTP
```

    Secondary name servers
        exist to eliminate single point failure problem
        obtain their data from primaries through *zone transfers*
        transfers typically occur every three hours

A name server may have authority in one *or more* zones
    The name server describe here has authority in both the *cs* and *csmath* zones.

Resolving names outside of the home zone owned by the primary nameserver

Primary name servers

must know the IP address of *each root nameserver*

may know the IP address of other (typically local) nameservers

Each *root nameserver* must know

the addresses of the primary name server for each 1st level domain (.edu, .gov, etc)

Each 1st level server *must* know

the IP addresses for its second level primaries etc.

An example DNS configuration (CS Dept. in Fall '95) (see ~/westall/class/881net/dns**)**

The tree is rooted at a file called `named.boot`

```
;
; boot file for authoritive master name server for cs.clemson.edu
;
;
domain     cs.clemson.edu.    <=== This is the name of the domain
directory  /usr/lib/domain    <=== This is the directory that contains the other stuff
;
;type      domain                    source host/file   backup file
;----      ------                     ----------------   -----------
cache      .                          root.cache
primary    cs.clemson.EDU             cs.clemson.zone
primary    0.127.IN-ADDR.ARPA         localhost.rev
primary    48.127.130.IN-ADDR.ARPA    48.127.130.rev
primary    56.127.130.IN-ADDR.ARPA    56.127.130.rev
primary    csmath.clemson.EDU         csmath.clemson.zone
primary    66.127.130.IN-ADDR.ARPA    66.127.130.rev
primary    70.127.130.IN-ADDR.ARPA    70.127.130.rev
primary    82.127.130.IN-ADDR.ARPA    82.127.130.rev
```

Notes:

The start of a comment is indicated by a ;

The table contains the location of the root cache file

and

For every domain in which this server is the primary authority

the domain name (also called the *origin)*

the origin is appended to each name in the name db's if the name does not end in .

the name of the file that describes the domain.

thus `48.127.130.rev` *is* actually a file name

we will get to its contents shortly

This is the contents of the file `root.cache`

```
NS.NIC.DDN.MIL.      99999999  IN  A  192.67.67.53
NS.NASA.GOV.         99999999  IN  A  128.102.16.10
NS.NASA.GOV.         99999999  IN  A  192.5.25.82
A.ISI.EDU.           99999999  IN  A  26.3.0.103
A.ISI.EDU.           99999999  IN  A  129.9.0.107
AOS.BRL.MIL.         99999999  IN  A  128.20.1.2
AOS.BRL.MIL.         99999999  IN  A  192.5.25.82
GUNTER-ADAM.AF.MIL.  99999999  IN  A  26.1.0.13
C.NYSER.NET.         99999999  IN  A  192.33.4.12
TERP.UMD.EDU.        99999999  IN  A  128.8.10.90
```

(The 99999999's are a non–longer used timeout value –– a vestige of an earlier design).
The root nameserver data also lives in `netinfo/root-servers.txt` at `nic.ddn.mil` (as of Fall '95)

```
A.ROOT-SERVERS.NET   198.41.0.4
B.ROOT-SERVERS.NET   128.9.0.107
C.ROOT-SERVERS.NET   192.33.4.12
D.ROOT-SERVERS.NET   128.8.10.90
E.ROOT-SERVERS.NET   192.203.230.10
F.ROOT-SERVERS.NET   39.13.229.241
G.ROOT-SERVERS.NET   192.112.36.4
H.ROOT-SERVERS.NET   128.63.2.53
I.ROOT-SERVERS.NET   192.36.148.17
```

Note the relatively small intersection with the file configured by our sysadmin at the time.

(as of Spring '97)

```
A.ROOT-SERVERS.NET   198.41.0.4         BIND (UNIX)
B.ROOT-SERVERS.NET   128.9.0.107        BIND (UNIX)
C.ROOT-SERVERS.NET   192.33.4.12        BIND (UNIX)
D.ROOT-SERVERS.NET   128.8.10.90        BIND (UNIX)
E.ROOT-SERVERS.NET   192.203.230.10     BIND (UNIX)
F.ROOT-SERVERS.NET   192.5.5.241        BIND (UNIX)
G.ROOT-SERVERS.NET   192.112.36.4       BIND (UNIX)
H.ROOT-SERVERS.NET   128.63.2.53        BIND (UNIX)
I.ROOT-SERVERS.NET   192.36.148.17      BIND (UNIX)
J.ROOT-SERVERS.NET   198.41.0.10        BIND (UNIX)
K.ROOT-SERVERS.NET   198.41.0.11        BIND (UNIX)
L.ROOT-SERVERS.NET   198.32.64.12       BIND (UNIX)
M.ROOT-SERVERS.NET   198.32.65.12       BIND (UNIX)
```

Tables maintained by the "infrastructure"

Root name servers: root.zone

```
.    IN   SOA  A.ROOT-SERVERS.NET. hostmaster.INTERNIC.NET. (
                    1998021900     ;serial
                    1800  ;refresh every 30 min
                    900    ;retry every 15 minutes
                    604800 ;expire after a week
                    86400  ;minimum of a day
                    )
.                          518400 IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.        518400     A     198.41.0.4
.                          518400     NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.        518400     A     128.63.2.53
.                          518400     NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.        518400     A     128.9.0.107
.                          518400     NS    C.ROOT-SERVERS.NET.
:
:
EDU.                       172800     NS    A.ROOT-SERVERS.NET.
                           172800     NS    H.ROOT-SERVERS.NET.
                           172800     NS    B.ROOT-SERVERS.NET.
                           172800     NS    C.ROOT-SERVERS.NET.
                           172800     NS    D.ROOT-SERVERS.NET.
                           172800     NS    E.ROOT-SERVERS.NET.
                           172800     NS    I.ROOT-SERVERS.NET.
                           172800     NS    F.ROOT-SERVERS.NET.
                           172800     NS    G.ROOT-SERVERS.NET.
EE.                        172800     NS    NS.KBFI.EE.
NS.KBFI.EE.                172800     A     192.121.251.13
EE.                        172800     NS    SUNIC.SUNET.SE.
                           172800     NS    NS.EU.NET.
                           172800     NS    NS.UU.NET.
EG.                        172800     NS    FRCU.EUN.EG.
FRCU.EUN.EG.               172800     A     193.227.1.1
```

Here is the zone data for the EDU. domain

```
EDU. IN   SOA  A.ROOT-SERVERS.NET. hostmaster.INTERNIC.NET. (
                    1998021900     ;serial
                    1800  ;refresh every 30 min
                    900    ;retry every 15 minutes
                    604800 ;expire after a week
                    86400  ;minimum of a day
                    )
:
CLEARWATER.EDU.            172800     NS    NS3.MCI.NET.
CLEARY.EDU.                172800     NS    NS1.ISMI.NET.
NS1.ISMI.NET.              172800     A     206.31.56.1
CLEARY.EDU.                172800     NS    NS2.ISMI.NET.
NS2.ISMI.NET.              172800     A     207.51.208.2
CLEMSON.EDU.               172800     NS    HUBCAP.CLEMSON.EDU.
HUBCAP.CLEMSON.EDU.        172800     A     130.127.28.32
CLEMSON.EDU.               172800     NS    ENG.CLEMSON.EDU.
ENG.CLEMSON.EDU.           172800     A     130.127.200.5
CLEMSON.EDU.               172800     NS    RS0.INTERNIC.NET.
CLEVELAND.EDU.             172800     NS    NS.CLEVELAND.EDU.
NS.CLEVELAND.EDU.          172800     A     208.142.89.2
CLEVELAND.EDU.             172800     NS    NS2.MCI.NET.
CLEVELANDCHIROPRACTIC.EDU. 172800     NS    NS.LAGOON.COM.
NS.LAGOON.COM.             172800     A     208.130.188.1
CLEVELANDCHIROPRACTIC.EDU. 172800     NS    NS3.MCI.NET.
```

Organization of DNS Databases

Two basic types of DB exist

1 – Used to map names to addresses
2 – Used to map addresses to names (the .rev files)

Records of both types are called *Resource Records*

We start with type 1 – the file `cs.clemson.zone`

The domain name begins the file and must start in column 1
All records until another name is encountered apply to this nameserver

@	a substitute for the *origin* from the boot file
IN	stands for Internet
SOA	stands for start of authority.
cs.clemson.edu.	the nameserver name
root.cs.clemson.edu	the e–mail address for reporting problems.

```
;   @(#)cs.clemson.zone   1.271 9/8/95
; Authoritative data for cs.clemson.edu
;
@      IN SOA   cs.clemson.edu. root.cs.clemson.edu. (
                1.271        ; Serial
                43200        ; Refresh  12 hours
                1800         ; Retry   1/2 hour
                604800       ; Expire  168 hours (7 days)
                86400)       ; Minimum  24 hours (TTL)
```

Names of other name servers.

```
       IN NS    cs.clemson.edu.
       IN NS    eng.clemson.edu.
       IN NS    hubcap.clemson.edu.
```

Internet addresses of this name server

```
       IN A     130.127.44.18
       IN A     130.127.48.1
```

MX = Mail exchange, WKS = Well known services

```
       IN MX 0  cs.clemson.edu.
       IN HINFO SPARC10 SunOS4.1.3
       IN WKS   130.127.44.18 TCP TELNET FTP SMTP
       IN WKS   130.127.48.1  TCP TELNET FTP SMTP

localhost   IN A  127.0.0.1
```

Alias names are specified as follows:

CNAME means canonical name (the true name of the host).

Thus citron is actually an alias –– cs is the true name

```
citron      IN CNAME cs.clemson.edu.
loghost     IN CNAME cs.clemson.edu.
timehost    IN CNAME cs.clemson.edu.
mailhost    IN CNAME cs.clemson.edu.

www         IN CNAME plato.cs.clemson.edu.


; Dr. Peck asked that this CNAME be put in because of a typo
; in a journal.  Keep indefinately.
sc.clemson.edu. IN CNAME cs.clemson.edu.
```

This mechanism is used to associate a specific name with each of multiple interfaces.

```
citron-gw      IN A  130.127.44.18
citron-local   IN A  130.127.48.1
```

Regular workstation entries look as follows:

MX stands for Mail eXchange.

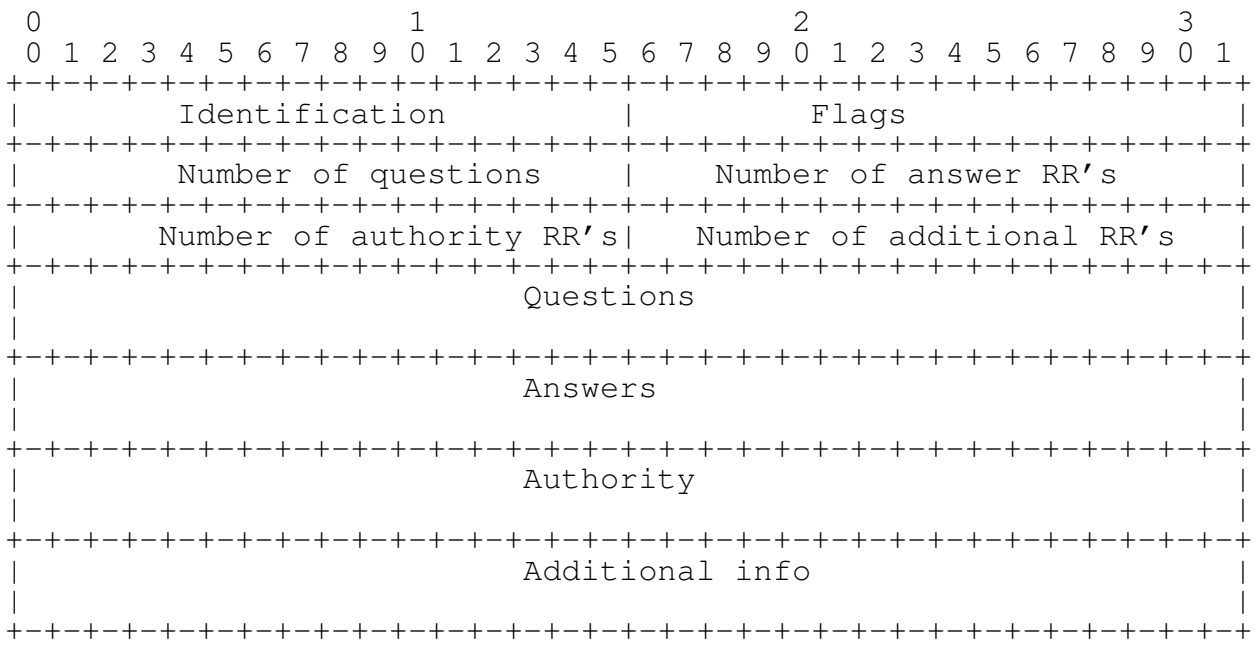
The lower the value the better the choice as an (intermediate) destination

```
jmw        IN A      130.127.48.24
           IN MX 0   jmw.cs.clemson.edu.
           IN MX 10  cs.clemson.edu.
           IN HINFO  SUN-4/75 SunOS4.1.1
           IN WKS    130.127.48.24  TCP TELNET FTP SMTP
```

Thus one might add

```
           IN MX 20  hubcap.clemson.edu.
```

DNS Message format

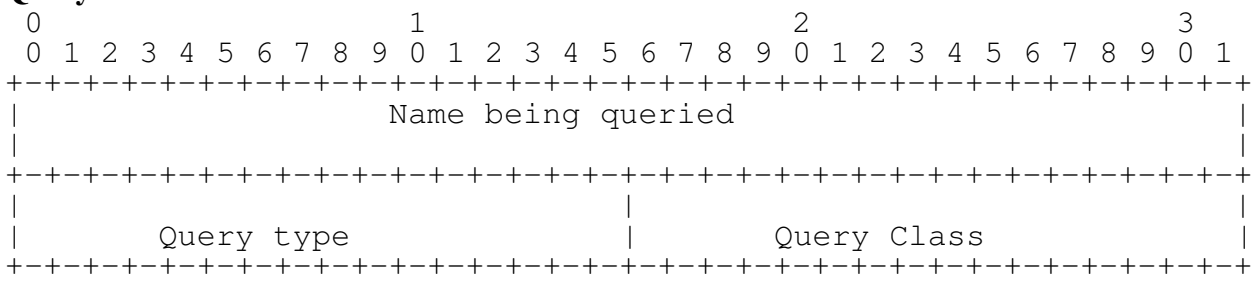


Identification – set by client and returned by server to match requests with replies

Flags

#bits	value	meaning
1	0	Query
	1	Response
4	0	Standard query
	1	Inverse query
	2	Server status
1	1	Authoritative answer
1	1	Truncated reply
1	1	Recursion desired
1	1	Recursion available
3	0's	Reserved
4	0	Successful status
	3	Domain doesn't exist

Query formats:



Name format:

0x2, ’c’,’s’,0x7,’c’,’l’,’e’,’m’,’s’,’o’,’n’,0x3,’e’,’d’,’u’,0

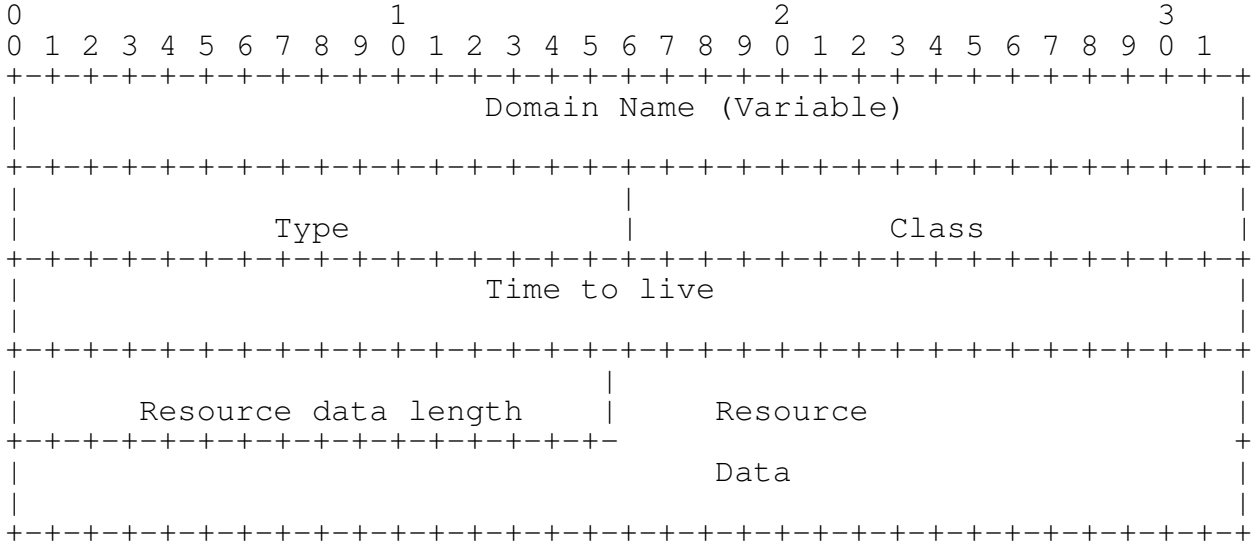
Query types

Value	Name	Description
1	A	IP Address
2	NS	Name server
5	CNAME	canonical name
12	PTR	pointer record
13	HINFO	host information
15	MX	mail exchange record
252	AXFER	request zone transfer

Query class

typically 1 meaning "Internet address" (or IN Resource Records)

Answers, authority, and additional information are reported in *resource records* (RRs).



Time to live is the number of seconds a RR can be cached (2 days is normal)
Resource data length and type depend on "type"
For IP address length = 4 and data = 4 byte address (not the dotted–decimal form).

Pointer queries: Mapping addresses to names

The domain `in-arpa.arpa` is used as a root for this mapping.
Zones in this namespace are allocated using IP address *in reverse*

Recall `named.boot` file

```
primary    cs.clemson.EDU           cs.clemson.zone
primary    0.127.IN-ADDR.ARPA       localhost.rev
primary    48.127.130.IN-ADDR.ARPA  48.127.130.rev
primary    56.127.130.IN-ADDR.ARPA  56.127.130.rev
primary    csmath.clemson.EDU       csmath.clemson.zone
primary    66.127.130.IN-ADDR.ARPA  66.127.130.rev
primary    70.127.130.IN-ADDR.ARPA  70.127.130.rev
primary    82.127.130.IN-ADDR.ARPA  82.127.130.rev
```

Note the reversed addresses

48.127.130.rev is actually the name of a file that contains our inverse mappings:

```
@       IN      SOA     cs.clemson.edu. root.cs.clemson.edu.
                                1.170   ; Serial
                                43200   ; Refresh  12 hours
                                300     ; Retry     5 minute
                                604800  ; Expire  168 hours
                                86400 ) ; Minimum  24 hours
        IN      NS      cs.clemson.edu.
        IN      NS      eng.clemson.edu.
        IN      NS      hubcap.clemson.edu.
;
; Servers
1       IN      PTR     citron.cs.clemson.edu.
2       IN      PTR     plato.cs.clemson.edu.
4       IN      PTR     diogenes.cs.clemson.edu.
5       IN      PTR     atlantic.cs.clemson.edu.
:
24      IN      PTR     jmw.cs.clemson.edu
:
```

etc... one entry for every workstation or pc on the .48 subnet.
(The numeric value *is* the last part of the IP address).

Configuring the resolver

The resolver code is linked with each application and reads `/etc/resolv.conf`

In SunOS 4.x two types of statements are defined in the man pages
domain – gives the domain name to be added to unqualified queries
nameserver – gives the address of a nameserver that should be contacted

```
domain cs.clemson.edu
# citron.cs.clemson.edu
nameserver      130.127.48.1
# hubcap.clemson.edu
nameserver      130.127.8.1
```

3 nameservers can be specified (they are useful only as backups)
DNS traffic uses port 53 on UDP and TCP

Alternative form supported in Linux.

```
search atm.cs.clemson.edu cs.clemson.edu
```

Domain names are tried in order specified

Caching

Objectives:
Reduce DNS traffic on internet
Provide faster response to queries

Cache is normally implemented in the **nameserver *not* the resolver**

Resolver caching is bad for two reasons
(both are a function of the resolver being linked with the application)

1 – There are multiple instances of the resolver active at any time with no good way for them to share data
2 – Resolvers come and go with applications thus not permitting any long term caching.

TCP vs UDP

DNS normally uses UDP but..
A TCP/IP host is required to handle only a 576 byte datagram (512 byte payload)
So.. DNS only sends max of 512 bytes in UPD records

TCP is used when
A response returns with TC bit set and the query is reissued
Any time a zone transfer takes place.

A simple example:

```
/* query.c */

/* A simple example of how to use resolver services to */
/* make a name server query.                           */

#include <stdio.h>
#include <sys/types.h>
#include <arpa/nameser.h>
#include <netdb.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <resolv.h>
#include <sys/param.h>
#include <string.h>
#include <ctype.h>

typedef union {
   HEADER qb1;
   char qb2[PACKETSZ];
} querybuf;

int getinfo(
char *name,
char *domain,
int   type)
{
   HEADER   *hp;
   char     *eom, *cp;
   querybuf buf, answer;
   int      n;
   int      ancount, nscount, arcount, qdcount;
   char     host[2*MAXDNAME+2];

   if (domain == NULL)
      (void)sprintf(host, "%.*s", MAXDNAME, name);
   else
        (void)sprintf(host, "%.*s.%.*s", MAXDNAME, name, MAXDNAME,
                        domain);

   printf("Querying name: %s\n", host);

   n = res_mkquery(QUERY, host, C_IN, type, (char *)NULL, 0,
                        NULL, (char *)&buf, sizeof(buf));

   if (n < 0)
   {
      if (_res.options & RES_DEBUG)
         printf("res_mkquery failed\n");
      h_errno = NO_RECOVERY;
      return(0);
   }

   n = res_send((char *)&buf, n, (char *)&answer,
                              sizeof(answer));
```

```
    if (n < 0)
    {
       if (_res.options & RES_DEBUG)
          printf("res_send failed\n");
       h_errno = TRY_AGAIN;
       return (0);
    }

    eom = (char *)&answer + n;
    hp = (HEADER *)&answer;

    ancount = ntohs(hp->ancount);
    qdcount = ntohs(hp->qdcount);
    nscount = ntohs(hp->nscount);
    arcount = ntohs(hp->arcount);

    printf("Ans %d -  QDs %d - NSs %d - ARs %d \n",
            ancount, qdcount, nscount, arcount);
    return(n);
}

void main(
int argc,
char **argv)
{
    int status;

    res_init();

    _res.options |= RES_USEVC;
    _res.retry = 1;
    _res.retrans = 15;
    _res.options &= ~RES_RECURSE;
    _res.options |= RES_DEBUG;

    status = getinfo(argv[1], "cs.clemson.edu.", T_A);
    printf("%d bytes returned \n", status);


}
```

Program output:

```
     Generated by the program itself
     Automagically generated by the resolver (debug flag)

Querying name: jmw.cs.clemson.edu.

res_mkquery(0, jmw.cs.clemson.edu., 1, 1)
res_send()
HEADER:
     opcode = QUERY, id = 1, rcode = NOERROR
     header flags:
     qdcount = 1, ancount = 0, nscount = 0, arcount = 0

QUESTIONS:
     jmw.cs.clemson.edu, type = A, class = IN

Querying server (# 1) address = 130.127.48.6
got answer:
HEADER:
     opcode = QUERY, id = 1, rcode = NOERROR
     header flags:  qr aa ra
     qdcount = 1, ancount = 1, nscount = 0, arcount = 0

QUESTIONS:
     jmw.cs.clemson.edu, type = A, class = IN

ANSWERS:
     jmw.cs.clemson.edu
     type = A, class = IN, ttl = 1 day, dlen = 4
     internet address = 130.127.48.24

Ans 1 -  QDs 1 - NSs 0 - ARs 0
52 bytes returned
```

Note that other tools such as `nslookup` and `host` similarly allow generation of resolver debug messages.

Example of the use of nslookup

```
/local/jmw ==> nslookup
Default Server:  citron.cs.clemson.edu
Address:  130.127.48.1

> set all
Default Server:  citron.cs.clemson.edu
Address:  130.127.48.1

Set options:
  nodebug         defname         search          recurse
  nod2            novc            noignoretc      port=53
  querytype=A     class=IN        timeout=5       retry=4
  root=ns.nic.ddn.mil.
  domain=cs.clemson.edu
  srchlist=cs.clemson.edu/clemson.edu

==================

> set querytype=HINFO
> jmw
Server:  citron.cs.clemson.edu
Address:  130.127.48.1

jmw.cs.clemson.edu      CPU = SUN-4/75  OS = SunOS4.1.1

==========================
```

```
> set q=NS
> cs.unc.edu
Server:  citron.cs.clemson.edu
Address:  130.127.48.1

Non-authoritative answer:
cs.unc.edu      nameserver = mcenroe.cs.unc.edu
cs.unc.edu      nameserver = borg.cs.unc.edu
cs.unc.edu      nameserver = ncnoc.ncren.net
cs.unc.edu      nameserver = ns2.unc.edu
cs.unc.edu      nameserver = ns.unc.edu

Authoritative answers can be found from:
cs.UNC.EDU      nameserver = mcenroe.cs.unc.edu
cs.UNC.EDU      nameserver = borg.cs.unc.edu
cs.UNC.EDU      nameserver = ncnoc.ncren.net
cs.UNC.EDU      nameserver = ns2.unc.edu
cs.UNC.EDU      nameserver = ns.unc.edu
mcenroe.cs.unc.edu      internet address = 152.2.128.184
borg.cs.unc.edu internet address = 152.2.128.183
ncnoc.ncren.net internet address = 192.101.21.1
ncnoc.ncren.net internet address = 128.109.193.1
ns2.unc.edu     internet address = 152.2.100.1
ns2.unc.edu     internet address = 152.2.120.99
ns.unc.edu      internet address = 152.2.21.1
```

```
> server borg.cs.unc.edu
Default Server:  borg.cs.unc.edu
Address:  152.2.128.183

> ls cs.unc.edu > unc.log
[borg.cs.unc.edu]
###############################################################################
####################
Received 4415 records.

>
(Here is some of the output)
> ls cs.unc.edu
[borg.cs.unc.edu]
 cs.unc.edu.                    server = mcenroe.cs.unc.edu
 mcenroe                        152.2.128.184
 cs.unc.edu.                    server = borg.cs.unc.edu
 borg                           152.2.128.183
 cs.unc.edu.                    server = ncnoc.ncren.net
 ncnoc.ncren.net.               192.101.21.1
 ncnoc.ncren.net.               128.109.193.1
 cs.unc.edu.                    server = ns2.unc.edu
 ns2.unc.edu.                   152.2.100.1
 ns2.unc.edu.                   152.2.120.99
 cs.unc.edu.                    server = ns.unc.edu
 ns.unc.edu.                    152.2.21.1
 cs.unc.edu.                    152.2.128.159
 luebke-slip                    152.2.132.239
 mac-lt6                        152.2.132.7

> set q=HINFO
> mac-lt6
Server:  borg.cs.unc.edu
Address:  152.2.128.183

*** borg.cs.unc.edu can't find mac-lt6: Non-existent domain
> mac-lt6.cs.unc.edu
Server:  borg.cs.unc.edu
Address:  152.2.128.183

***  No  host  information  (HINFO)  records  available  for  mac-
lt6.cs.unc.edu
```

```
================================
> polaris.cs.unc.edu
Server:  borg.cs.unc.edu
Address:  152.2.128.183

***   No   host   information   (HINFO)   records   available   for
polaris.cs.unc.edu
> set q=A
> polaris.cs.unc.edu
Server:  borg.cs.unc.edu
Address:  152.2.128.183

Name:    polaris.cs.unc.edu
Address:  152.2.133.160

============================

> set nosearch
> set norec
> domino.cs.orst.edu
Server:  cs.clemson.edu
Addresses:  130.127.44.18, 130.127.48.1
Aliases:  citron.cs.clemson.edu

Name:    domino.cs.orst.edu
Served by:
- beasley.UCS.ORST.EDU
          128.193.128.3
          CS.ORST.EDU
- chert.cs.orst.edu
          128.193.38.7
          CS.ORST.EDU
- gibson.cs.orst.edu
          128.193.36.30
          CS.ORST.EDU
- NS.cs.orst.edu
          128.193.36.31
          CS.ORST.EDU
```

```
====================================

> set d2
> set norecurse
> set nosearch
> warp.cs.ucla.edu.

Default Server:  citron.cs.clemson.edu
Address:  130.127.48.1

> > > > Server:  citron.cs.clemson.edu
Address:  130.127.48.1

res_mkquery(0, warp.cs.ucla.edu, 1, 1)
------------
SendRequest(), len 34
    HEADER:
        opcode = QUERY, id = 2, rcode = NOERROR
        header flags:  query
           questions = 1,   answers = 0,   authority records = 0,
additional = 0

    QUESTIONS:
        warp.cs.ucla.edu, type = A, class = IN

------------
```

```
Got answer (247 bytes):
    HEADER:
        opcode = QUERY, id = 2, rcode = NOERROR
        header flags:  response, recursion avail.
          questions  =  1,   answers  =  0,   authority  records  =  5,
additional = 6

    QUESTIONS:
        warp.cs.ucla.edu, type = A, class = IN
    AUTHORITY RECORDS:
    ->  UCLA.EDU
        type = NS, class = IN, dlen = 10
        nameserver = Maui.CS.UCLA.EDU
        ttl = 50022 (13 hours 53 mins 42 secs)
    ->  UCLA.EDU
        type = NS, class = IN, dlen = 16
        nameserver = SPARKY.ARL.MIL
        ttl = 50022 (13 hours 53 mins 42 secs)
    ->  UCLA.EDU
        type = NS, class = IN, dlen = 6
        nameserver = NCC.UCLA.EDU
        ttl = 50022 (13 hours 53 mins 42 secs)
    ->  UCLA.EDU
        type = NS, class = IN, dlen = 9
        nameserver = KODIAK.UCLA.EDU
   ttl = 50022 (13 hours 53 mins 42 secs)
    ->  UCLA.EDU
        type = NS, class = IN, dlen = 8
        nameserver = POLAR.UCLA.EDU
        ttl = 50022 (13 hours 53 mins 42 secs)
    ADDITIONAL RECORDS:
    ->  Maui.CS.UCLA.EDU
        type = A, class = IN, dlen = 4
        internet address = 131.179.128.11
        ttl = 82131 (22 hours 48 mins 51 secs)
    ->  SPARKY.ARL.MIL
        type = A, class = IN, dlen = 4
        internet address = 128.63.48.85
        ttl = 56249 (15 hours 37 mins 29 secs)
    ->  SPARKY.ARL.MIL
        type = A, class = IN, dlen = 4
        internet address = 192.5.23.200
        ttl = 483795 (5 days 14 hours 23 mins 15 secs)
    ->  NCC.UCLA.EDU
        type = A, class = IN, dlen = 4
        internet address = 128.97.128.1
        ttl = 82131 (22 hours 48 mins 51 secs)
    ->  KODIAK.UCLA.EDU
        type = A, class = IN, dlen = 4
        internet address = 164.67.128.11
        ttl = 82131 (22 hours 48 mins 51 secs)
    ->  POLAR.UCLA.EDU
        type = A, class = IN, dlen = 4
        internet address = 164.67.128.12
        ttl = 82131 (22 hours 48 mins 51 secs)
```