# The Domain Name Service

*Last Edit*
*November 26, 1992*

Reg Quinton <reggers@julian.uwo.ca>
Computing and Communications Services
The University of Western Ontario
London, Ontario N6A 5B7
Canada

## Outline

(1) **Introduction —**
History, Goals, Where is Documentation?

(2) **DNS design —**
Components (daemon and resolver), Primary & Secondary servers, Boot Files and components, DNS data records (SOA, NS, etc).

(3) **Putting it all together —**
Registration, Responsiblity, Your DNS data, Configuring the resolver, implementing a server.

(4) **Help —**
the NIC and liaison.

# 1. History/Goals

The Domain Name Service (or DNS) is the *most important* service on the Internet. An essential component to any IP implementation.

Paul Mockapetris is the author of the DNS — the design of the protocol, the overall methodology, etc. This guy is God! But even god gets help (many people have worked on the DNS — cf. recent fix for `nslookup(8)`).

When you say, eg. `ftp uunet.uu.net`, you get connected to [192.48.96.2] (or perhaps [137.39.1.2]) because the DNS maps the name to the numbers.

```
[11:40am julian] nslookup uunet.uu.net
Name:    uunet.uu.net
Addresses:  192.48.96.2, 137.39.1.2
```

The IPnet works by number, people and programmers work using names (likewise for every other network).



When you say, eg. `mail rosner@vax.oxford.ac.uk`, your mail gets to Oxford in Britain because the DNS maps the name `vax.oxford.ac.uk` to a gateway on the Internet.

```
[11:52am julian] nslookup
> set type=mx
> vax.oxford.ac.uk
vax.oxford.ac.uk     preference = 13,
        mail exchanger = nsfnet-relay.ac.uk
Authoritative answers can be found from:
nsfnet-relay.ac.uk   inet address = 128.86.8.6
nsfnet-relay.ac.uk   inet address = 14.0.0.6
NS1.CS.UCL.AC.UK     inet address = 128.16.5.32
NS2.CS.UCL.AC.UK     inet address = 128.16.8.3
AOS.BRL.MIL          inet address = 192.5.25.82
AOS.BRL.MIL          inet address = 128.20.1.2
MCSUN.EU.NET         inet address = 192.16.202.1
```

## 1.1. Where to start

See RFC's 1032-1035 (for the brave). These are tedious readings — not for the faint of heart (I haven't read them!). These notes are found as `julian:~ftp/doc/courses/dns.ps`.

Read the On-line manuals and the chapters in your System Administrator's Guide.

```
[11:59am julian] man 8 named
[11:59am julian] man 5 resolv.conf
```

See also `julian:~ftp/nic` (lots of good readings there) and especially

(1) `~ftp/nic/named`: sample primary configuration (for `mcu.on.ca`).

(2) `~ftp/nic/named.secondary`: sample secondary configuration (for `*.uwo.ca`).

See also the newsgroup comp.protocols.tcp-ip.domains (for the bleeding edge).



## 1.2. DNS Evolution

The DNS is an *evolving* system although pretty stable these days. Evolving to provide more mail support, other networks (eg. AppleTalk), etc.

Recently we at CCS/UWO have started using `UNIFO` records to load the DNS with commentary text. The RFC spec says we ought to use `TXT` records but this isn't implemented by many servers.

You may have noticed recent change notices to fix a bug in `nslookup(8)` or the report of problems with `gethostbyname(3)`. Both are instances where we've loaded the DNS more information than the designers assumed — problems allocating space for answers.

On CCS Unix systems we've implemented some nice local tools that interface with nslookup — `byname(1)` and `bynum(1)`.

## 1.3. Goals

The goal is to provide a distributed data base that implements the traditional `/etc/host` (or `hosts.txt`) files of pre-DNS IP implementations.

```
[2:57pm julian] page /etc/hosts
# Wed Nov 11 17:08:18 EST 1992
# Data derived from official DNS data
#
129.100.0.0     uwo-net.uwo.ca
129.100.1.0     ccsexp.net.uwo.ca
129.100.1.1     romulus.ccs.uwo.ca
129.100.1.10    hadrian.ccs.uwo.ca
129.100.1.11    myst.ccs.uwo.ca
129.100.1.13    super.ccs.uwo.ca
129.100.1.14    debbie.ccs.uwo.ca
129.100.1.15    raj.ccs.uwo.ca
129.100.1.17    shadow.ccs.uwo.ca
129.100.1.18    crystal.ccs.uwo.ca
129.100.1.19    opers.ccs.uwo.ca
```
....etc.

There are some 1,500 (700 in 1990) IP numbers listed in the current `/etc/hosts` for Western (available as `julian:~ftp/nic/hosts`). If we were to list all hosts on the Internet we'd need a file of some 1,000,000 or more hosts!



## 1.4. History

Several events motivate the need for a distributed DNS service. Circa 1984, now, and in the forseeable future.

(1) ARPAnet addressing `user@host` didn't capture sites not on the ARPAnet.

(2) Addressing by network, eg. `user@host.ARPA` and `user@host.CSNET`, isn't necessarily the best route. For example, to get to the machine `CUNYVM.BINET` (aka `cunyvm.cuny.edu`) routing over the Internet is much faster.

(3) `Sendmail.cf` files were very difficult to administer — lots of rules for delivery to UUCP, BITNET, ARPAnet, MILnet, CSNet, Mailnet, etc.

(4) ARPAnet grew to a size where distributing `/etc/host` (or `hosts.txt`) would actually kill the net (cf. 1,500 hosts at Western and 1,000,000 or more on the Internet).

(5) Linear searches of `/etc/host` are tedious when the file is large.

(6) Updates to a central database distributed out to sites meant that changes to the net were not seen for months. (cf. BITNET routing tables, UUCP maps).

(7) `/etc/hosts` is managed by a person who may (or may not) make sure the data is in fact up to date. Likewise for `sendmail.cf` files.

(8) People on NetNorth, for example, wanted to have an addressing scheme that covered all systems on campus not just those on the particular network.

(9) The HOSTMASTER at `NIC.DDN.MIL` (the Network Information Centre for the Internet) was tired of constantly updating the central version of `hosts.txt`.

`NIC.DDN.MIL` stopped supporting `/etc/hosts` (actually `hosts.txt`) around 1988 — if you wanted to participate you had to use the DNS. This was at about the time Western connected.

If you want to participate, there's only one way. Use the DNS!



## 1.5. Goals/Needs

There was, and is, a clear need for a facility that

(1) Provides addressablity to sites not on the ARPAnet (eg. so you can address `user@do.ma.in` without worrying about **how** mail gets to the user).

(2) Didn't require a centralized data base (which would kill the net and the hostmaster).

(3) Can be managed in a distributed manner by those who have been given responsiblity for their network and/or domain.

(4) That would be backwards compatible with `/etc/hosts` and `hosts.txt` (note that `hosts.txt` listed name, number, machine type, and services).

(5) That would provide timely answers to questions like `gethostbyname(3)` (linear searches aren't fast!)

Without the Domain Name Service the Internet would not have survived! Certainly would not have survived at the current size.

## 2. DNS design

The problem managed by the DNS is the coordination and mapping of domain names to and from IP numbers (plus mail routing, machine type, services, owner, and more).

The coordination of IP numbers and Domain Names is managed by the `NIC.DDN.MIL` (actually by `hostmaster@nic.ddn.mil`). They delegate responsiblity and authority to campus NIC's (perhaps through regional contacts, eg. CAnet for Canada). The NIC for Western is CCS (by email as `NIC@UWO.CA`).

(1) `NIC.DDN.MIL` has assigned the class B IP-network [129.100.0.0] known as `UWONET` to the University of Western Ontario and responsibilty to CCS.

(2) `NIC.DDN.MIL` has assigned the domain name `CA` to Canada and delegated responsibility to `CAnet`; they have assigned the domain name `UWO.CA` to the University of Western Ontario and responsibilty to CCS.



This is a chain of authority with the top most responsiblity at the Internet NIC, delegated down through Country, University, on to departmental NIC's.

```
[11:13am suncon] nslookup
> set type=soa
> .
        origin = ns.nic.ddn.mil
        mail addr = hostmaster.nic.ddn.mil
        etc...
> ca.
        origin = relay.cdnnet.ca
        mail addr = hostmaster.relay.cdnnet.ca
        etc...
> uwo.ca.
        origin = julian.uwo.ca
        mail addr = nic.julian.uwo.ca
        etc...
> csd.uwo.ca
        origin = sybil.csd.uwo.ca
        mail addr = magi.csd.uwo.ca
```

The mail address in the Start of Authority (SOA) record is the person responsible for the domain. CCS delegates responsiblity to departments through a registration procedure. Networking comes with some responsiblities — at a minimum all hosts must be registered.

## 2.1. DNS components

There are two parts to the Domain Name Service:

(1) The **Name Daemon** — this is the program which runs on our campus name servers. Name Daemons answer questions about name to number translations. These are tcp/ip server processes. There are several running at UWO — `julian`, `hydra` and `rri` are the official name servers known to the rest of the internet.

(2) The **Resolver Routines** — these provide the interface between your application program (eg. `telnet(1)`, or `ftp(1)`) and the Name Daemon. The `gethostbyname(3)` call, for example, is implemented by a resolver routine which talks to the appropriate name daemon.

Cf. NNTP — the Network News Transport Protocol. People use programs like `rrn` (usually installed as `rn`) to read news. These news readers interact with an NNTP daemon (a server program running on some machine on the net) to get information that in earlier implementations required access to files.

Cf. SQL as a protocol for interacting with remote data bases.



### 2.1.1. Resolver routines

Programmers use library calls like `gethostbyname(3)`, `gehostbynumber(3)`, etc. These are now implemented using low level resolver calls. The DNS did not require people to rewrite their programs — the library interface is preserved.

Note that the manual entry for `gethostbyname(3)` has not changed. People program using tools that give answers — how the tool computes the answer is someone else's problem.

The Application Programming Interface (API) remains the same (cf. POSIX standards that define an API).

The resolver routines are hidden in the C library — not seen by the casual user. The reference point to connect resolver routines with name daemon on Unix systems is the file

```
/etc/resolv.conf
```

Resolver routines look for this file (if absent they look to `/etc/hosts`). The resolver file details **where** name servers are found. For example, most people have set up an `/etc/resolv.conf` that contains these lines (modulo your departmental domain name):

```
nameserver 129.100.2.12
nameserver 129.100.2.13
nameserver 129.100.7.100
domain     ccs.uwo.ca
```



The resolver file lists several name servers.

> The **nameserver** lines are used to find a name server. Three are given: `julian` [129.100.2.12], `hydra` [129.100.2.13] and `rri` [129.100.7.100]. These are the official name servers for `uwo.ca`.

There will be a lot of network traffic generated if the name servers you use aren't on the same network (ie. separated by a router). You'd also have real problems if these name servers weren't accessible (say they, or the net, were down). These are the *reasons* for running a local secondary server.

Anyone can run a secondary name server (provided that they are able). A kit has been prepared in `julian:~ftp/nic/named.secondary`. PC's and MACs typically rely on some other machine and don't run a name server.

Note: a name server will require space for data — say 200 bytes/entry. For the 1,500 entries at UWO you're well over the free space available on a typical PC or MAC.

The resolver file lists a default domain.

> The **domain** mentioned in `/etc/resolv.conf` is used to qualify hostnames (a name not terminated by a '.' needs to be qualified).

When I say, for example `ftp hydra` the resolver routines try to find `hydra.ccs.uwo.ca`, if that fails they try `hydra.uwo.ca`, if that fails they try `hydra.ca`.

Some systems (eg. NeXT and A/IX) aren't very bright about the default domain search (ie. they'll try `hydra.ccs.uwo.ca` and that's all). These are early implementations — definitely not state of the art.



### 2.1.2. Resolver on Vax/VMS

The Multinet implementation of the resolver routines is similar (it's actually a port of the public domain BSD implementation).

Rather than a file `/etc/resolv.conf` Multinet uses Vax/VMS logicals

```
$ show logical multinet*
....
(LNM$SYSTEM_TABLE)
....
   "MULTINET_LOCALDOMAIN" = "UWO.CA"
   "MULTINET_NAMESERVERS" = "127.0.0.1"
         = "129.100.2.12"
...
```

This corresponds to the `/etc/resolv.conf` file:

```
nameserver 127.0.0.1
nameserver 129.100.2.13
domain     uwo.ca
```

So there's nothing strange or mysterious going on here.

### 2.1.3. Resolver on CUTCP

For CUTCP on MS/DOS your `config.tel` will have similar information as follows:

```
domaintime=4        # lookup timeout
domainretry=4       # max number of retries
domainslist="ccs.uwo.ca,uwo.ca"
                    # default domain assumptions
...
name=julian
host=julian.uwo.ca
hostip=129.100.2.12
nameserver=1        # first name server
...
name=hydra
host=hydra.uwo.ca
hostip=129.100.2.13
nameserver=2        # second name server
...
name=rri
host=rri.uwo.ca
hostip=129.100.7.100
nameserver=3        # third name server
```

This corresponds with `/etc/resolv.conf` on Unix so there's nothing strange or mysterious going on here either!



### 2.1.4. Minimum Configuration

At a *minimum* you can participate in the DNS by

(1) Constructing `/etc/resolv.conf` file (for Unix) with appropriate servers and default domain.

(2) Defining the appropriate logicals (for Vax/VMS).

(3) Configuration values in `config.tel` (for CUTCP on MS/DOS)

This will enable you to say, for example

```
[2:57pm julian] ftp uunet.uu.net
```

even if you don't have a tabled entry in `/etc/hosts` (or `hosts.txt`).

Single user work stations, eg. your desktop Sun 3/50, PC or MAC, are often configured this way. All DNS queries are handled by talking with a remote server (eg. the NFS file server for that system).

Larger systems, which do a lot more IP work, should have more than this minimal configuration — they should run a Name Daemon.



## 2.2. Name Daemons

The Name Daemon (or name server) is the process which answers queries made by resolver routines. This is an IP service listening on the `domain` port (see `/etc/services`, also check servers with `netstat(8)`).

Most Name Daemons are ports of BIND (the Berkeley Internet Name Daemon). This is a public domain implementation of the DNS (available on uunet and other archives).

```
[10:08am julian] what /etc/named
/etc/named:
        named 4.8.3 Tue Oct 20 15:32:08 EDT 1992
        db_dump.c       4.30 (Berkeley) 6/1/90
        etc...
```

We're running the most recent version of `named` on julian — not the vendor version (why are vendors so slow to keep up?).

### 2.2.1. Primary vs. Secondary Servers

There are two types of Name Daemon:

(1) A **primary** name server answers questions using authoritative data on file (and someone is supposed to be maintaining that data file!).

(2) A **secondary** name server answers questions using data retrieved periodically over the net from some primary name server and cached locally in a file.

(3) Most name servers are a mix of primary and secondary (eg. primary for the local domain, secondary for some others).

The Name Daemon is started at boot time and uses the configuration file `/etc/named.boot`. On Vax/VMS the file is:

```
multinet:domain-name-service.configuration
```



### 2.2.2. Named.boot

`Named.boot` is the configuration file read when the name daemon starts. For example, in julian's boot sequence

```
if [ -x /etc/named -a -f /etc/named.boot ];
then
        /etc/named -b /etc/named.boot;
fi
```

What is required in a configuration file to get a Name Daemon up, running, and communicating with other Name Daemons?

(1) Where are some other name daemons on the net? In particular, where are the **root** name servers?

(2) What domains do I serve? ie. what do I act as a primary/secondary server for?

In theory the **simplest** name server would only need to know where the root name servers are — everything else can be learned from them. In **practice** one would always have some authoritative data on hand (eg. my name and number).

The Named.boot file on julian looks like:

```
; ...
; primary master boot file (modelled after waterloo's)
;
domain     uwo.ca
directory  /usr/local/lib/named
; ...
primary    uwo.ca             data/uwo.domain
primary    apmaths.uwo.ca     data/apmaths.domain
primary    arts.uwo.ca        data/arts.domain
primary    astro.uwo.ca       data/astro.domain
primary    biostats.uwo.ca    data/biostats.domain
; ....
primary    100.129.in-addr.arpa    in-addr.arpa
primary    0.0.127.in-addr.arpa    named.local
; ...
; Secondary stuff for mcu.gov.on.ca
;
secondary mcu.gov.on.ca.           192.68.229.1 cache/mcu..
secondary 229.68.192.in-addr.arpa. 192.68.229.1 cache/229..
; ...
cache      .                  root.cache
```

The ellipsis indicates deleted data (for these slides only).



### 2.2.3. Boot File Components

#### 2.2.3.1. Comments

Any line beginning with a ';' is a comment. If you don't comment your work then how can anyone help you?

```
; $Author: reggers $
; $Date: 1992/11/26 17:47:58 $
; $Id: named.boot,v 1.60 1992/11/12 15:57:52..
; $Source: /usr/local/lib/named/RCS/named.bo..
;
; primary master boot file (modelled after w..
```

#### 2.2.3.2. Domain

The `domain` keyword specifies the default domain. As the primary and secondary data is loaded any domain not terminated with a '.' gets this tacked on.

```
domain    uwo.ca
```

But note that the default domain can be changed in any file with an `ORIGIN` directive.

### 2.2.3.3. Directory

The `directory` line specifies the default directory where files are stored. Current working directory for execution.

```
directory /usr/local/lib/named
```

For example, on julian:

```
[10:09am julian] lc /usr/local/lib/named
Directories:
RCS            cache          data
registration

Files:
Distfile       aliases        in-addr.arpa
in-ether.arpa  in-novell.arpa named.boot
named.local    root.cache     root.zone
```

Note that we store everything in a local directory:

```
[11:35am julian] ls /etc/named.boot
/etc/named.boot -> /usr/local/lib/named/named.boot
```



### 2.2.3.4. Primary

Lines with keyword `primary` specify a domain and file where *authoritative* data for that domain is stored — Start of Authority (SOA) data.

```
primary   cogsci.uwo.ca.  data/cogsci.domain
```

The file `data/cogsci.domain` is found in the specified directory `/usr/local/lib/named` and contains SOA data for the mentioned domain.

The file *must* have one, and only one, SOA record for the domain.

This is the data file that the technical contact maintains and submits to the NIC.

### 2.2.3.5. Secondary

Lines with keyword `secondary` specify a domain, sites which have authoritative data, and file where *cached* data is stored.

```
secondary csd.uwo.ca. 129.100.11.5 cache/csd.uwo.ca
```

The file `cache/csd.uwo.ca` is retrieved from the specified server [129.100.11.5] and cached in the specified directory (if the server is down it can use cached data).

The file will contain SOA data for the mentioned domain (the server is the primary server for that domain and ought to have authoritative data on file).

The data *must* have one, and only one, SOA record for the specified domain.



### 2.2.3.6. Cache

Lines with keyword **cache** specify a file with hot-wired information. Usually the "root" domain servers are stored in the file `root.cache`.

```
cache           .               root.cache
```

You need to have some pointers to the root servers. On julian, the `root.cache` file is:

```
; ...lots of comments
;
.  99999999     IN NS NS.NIC.DDN.MIL.
                IN NS AOS.BRL.MIL.
                IN NS A.ISI.EDU.
; ...
;
NS.NIC.DDN.MIL. 99999999 IN A 192.67.67.53
AOS.BRL.MIL.    99999999 IN A 192.5.25.82
A.ISI.EDU.      99999999 IN A 26.3.0.103
; ...
```

These records specify the **names** of Root Name Servers (`IN NS` records) and their **address** (`IN A` records). These records are effectively permanent (with a 99999999 Time to Live TTL value).

### 2.2.3.7. Forwarders

Lines with keyword `forwarders` list IP addresses of local Name Servers who know more (eg. everyone on campus uses the three campus servers as forwarders — to minimize DNS traffic on ONet).

```
; If I can't answer it, these guys can:
;
forwarders      129.100.2.12  \
                129.100.7.100 \
                129.100.2.13
```

We don't want too much traffic on our ONet links and *insist* that sites use the official servers (`julian`, `hydra`, and `rri`) as forwarders. Those servers, of course, don't have any forwarders specified — they go out to the net.



## 2.3. DNS record format

The format of the records in root.cache is *exactly* the same as the format in **primary** data files.

Each record in the DNS looks like:

[*domain*] [*TTL*] `IN` *data*

Each record gives information about a domain (eg. address, machine type, services, mail routing, etc.).

### 2.3.1. Domain

DNS data records are all tagged by a **domain** name. All DNS queries (ie. low level resolver routines) ask for records by domain name.

If the domain name is missing on a record, the previous one is assumed. These are the equivalent:

```
hydra.uwo.ca. IN A   129.100.2.13
   IN WKS   129.100.2.13 TCP telnet ftp smtp
   IN MX    0       hydra.uwo.ca.
   IN MX    10      julian.uwo.ca.
   IN HINFO "VAX 4500" "VMS"
   IN UINFO "Machine room, CCS, 2nd flr., Nat.Sc."
   IN UINFO "Contact: Gerard Stafleu <gerard@uwovax.uwo.ca>"
   IN UINFO "Phone: 679-2111x(6043)"
   IN UINFO "also known as 'uwovax'"
```

and

```
hydra.uwo.ca. IN A   129.100.2.13
hydra.uwo.ca. IN WKS 129.100.2.13 TCP telnet ftp smtp
hydra.uwo.ca. IN MX  0       hydra.uwo.ca.
hydra.uwo.ca. IN MX  10      julian.uwo.ca.
```



Note the trailing dot convention to distinguish `uwo.ca.` (Canada) from `uwo.ca` (which might mean California in the US). A fully qualified domain name (ie. no assumptions about the current context) will terminate with a dot.

If the domain name **isn't fully qualified** then the default domain is tacked on (but this is dangerous, be careful). These are the same:

```
hydra.uwo.ca. IN A 129.100.2.13
```

and

```
$ORIGIN uwo.ca.
hydra         IN A 129.100.2.13
```

The moral here is to always fully qualify all domain names.

Everyone who has ever maintained DNS data has, at some time, dropped a trailing '.' and suffered for it.

### 2.3.2. TTL

**TTL** — time to live. A permanent record would have a large value (eg. 99999999). You only see these in root.cache files on hot wired addresses. For example, from julian's `root.cache`:

```
NS.NIC.DDN.MIL. 99999999 IN A 192.67.67.53
```

All records in the DNS have a TTL value (specified in the `SOA` record).

Name Daemons *cache* answers they've received. This minimizes traffic on the net (the first query for a domain is cached, subsequent queries get the cached answer).



### 2.3.3. IN

`IN` is short for Internet. The DNS is designed to support a directory service for arbitrary networks — there's talk of defining Apple, Decnet, ISO, etc. records. But not much has come of that (can you say X.500?).

To date all records in production DNS servers are internet records.

### 2.3.4. IN SOA

A Start of Authority record lists important information about the primary DNS data for a domain. Each data file listed with a **primary** record in the named.boot will have an SOA record. That includes the in-addr.arpa. map.

For example, from the primary DNS data file for gp.uwo.ca.

```
$ORIGIN gp.uwo.ca.

; SOA means Start of Authority, who's in charge?

@ IN SOA julian.uwo.ca. john.seis.gp.uwo.ca. (
               1.2       ; Serial EDIT ME TOO!!
               3600      ; Refresh
               300       ; Retry
               3600000   ; Expire
               3600 )    ; Minimum
```



The `@` is just a way of say `gp.uwo.ca`. This is a record for the `gp.uwo.ca` domain.

`julian.uwo.ca`. That's the name of the primary server for this domain. Authoritative data is on file on that system.

`john.seis.gp.uwo.ca`. That's the mail address of the person who maintains the primary data. If the data is wrong, it's his problem. Note the addressing style — no '@' sign. Use the machine address not an alias.

The Serial Number — each change to the primary data had better change the serial number. Secondary servers regularly query the primary and check serial numbers (to see if their cache is out of date). Some of the other numbers tailor querying and expiration times.

Time to live (TTL). No promises, but I believe the Minimum number is the TTL applied to all data in this domain.

## 2.3.5. IN NS

A Name Server record, lists a Name Server for the domain. Every domain must have at least two name servers (for the case when one is down).

The root name servers (eg. at `NS.NIC.DDN.MIL`) have `NS` records for `CA`. that look like:

```
ca.      IN NS   relay.cdnnet.ca.
ca.      IN NS   nnsc.nsf.net.
ca.      IN NS   clouso.crim.ca.
ca.      IN NS   ugw.utcs.utoronto.ca.
```

These servers, in turn, have `NS` records for `UWO.CA`. that look like:

```
uwo.ca. IN NS   julian.uwo.ca.
uwo.ca. IN NS   relay.cdnnet.ca.
uwo.ca. IN NS   hydra.uwo.ca.
uwo.ca. IN NS   rri.uwo.ca.
```

With only a very few network queries (and redirects to other Name Servers) you get the information you need. Of course none of these records are worth very much if they don't also have Address records cached with a long TTL!



### 2.3.5.1. NS chain of responbility

`NS` records pointing from the root, through `CA` and on down to `UWO.CA` nameservers don't appear in the DNS by magic; *people* put them into primary DNS data files. However

> Only CCS is the authority for `UWO.CA`, and only CCS can ask for `UWO.CA NS` records. This we have done.
>
> Likewise only `CAnet` is the authority for `CA`, and only they can ask for `CA NS` records. This they have done.

If *you* want to run a primary name server for your domain (so that you don't have to communicate changes to the campus NIC and wait to have them install your data) you *must* ask for and obtain a `NS` record from the campus NIC.

A name server will never be queried unless it appears in the chain of responsibility.

You *can* run a primary name server. But, this comes with some responsiblities — if you don't know how, or can't, run your DNS responsibly then we won't delegate.

For example, let's follow the chain of authority from the root down to the server for `csd.uwo.ca`:

```
[11:53am suncon] nslookup
> set type=ns
> server ns.nic.ddn.mil
> ca.
ca nameserver = relay.cdnnet.ca
ca nameserver = nnsc.nsf.net
etc..
> server relay.cdnnet.ca.
> uwo.ca.
uwo.ca nameserver = julian.uwo.ca
uwo.ca nameserver = hydra.uwo.ca
etc..
> server julian.uwo.ca.
> csd.uwo.ca.
csd.uwo.ca nameserver = sybil.csd.uwo.ca
csd.uwo.ca nameserver = julian.uwo.ca
etc..
```



### 2.3.6. IN A

An Address record, list the IP number for the domain. For example

```
hydra.uwo.ca.   IN A       129.100.2.13
```

A machine might have several IP numbers (if it has several network interfaces).

```
romulus.ccs.uwo.ca.IN A 129.100.1.1
             IN A       129.100.2.1
             IN A       129.100.3.1
             IN A       129.100.8.1
             IN A       129.100.31.1
             IN A       129.100.32.1
             etc..
             IN HINFO   CISCO "Dec/IP Router"
```

`nslookup(8)` returns address records:

```
[12:13pm suncon] nslookup romulus
Name:    romulus.ccs.uwo.ca
Addresses:  129.100.1.1, 129.100.2.1,
          129.100.3.1, 129.100.8.1,
          129.100.65.1, 129.100.67.1, etc..
```

A machine (or IP address) might be known by several names. For example,

```
phonebook.uwo.ca.   IN A       129.100.2.13
hydra.uwo.ca.       IN A       129.100.2.13
uwovax.uwo.ca.      IN A       129.100.2.13
```

There's another way of pointing different names at the same IP address (using `CNAME` records). I've never found `CNAME` records to be too useful and have run into some problems. So I recommend this strategy for pointing several names at the same machine.

We've tried a number of alternatives for making uwovax and hydra name the same machine. Currently we do it as above, in the past we've tried:

```
uwovax.uwo.ca.  IN CNAME  hydra.uwo.ca.
```

and

```
uwovax.uwo.ca.  IN MX 0   hydra.uwo.ca.
```

Both have their place, but `CNAME` records sometimes cause problems.



### 2.3.7. IN WKS

Lists Well Known Services, ie. the TCP or UDP services <= 512 (see `/etc/services`), supported by the server. For example,

```
julian.uwo.ca. IN A       129.100.2.12.
    IN WKS     129.100.2.12 UDP (domain tftp)
    IN WKS     129.100.2.12 TCP (nntp smtp
                     telnet domain finger ftp)
```

The idea with `WKS` records is that client applications (eg. `ftp`) should check first if the server actually supports the service (eg. supports the `ftp WKS`). In *practice* this never happens.

Maintaining these records is just documentation.

Systems, which provide no services (ie. they're just consumers) should have list no WKS.

A typical confusion: a PC might implement POP (Post Office Protocol) and SMTP (Simple Mail Transfer) client applications but not the server applicaton. You only list the servers (ie. services offered).

### 2.3.8. IN MX

List a Mail exchanger. Mail for this domain should be sent where?

```
julian.uwo.ca. IN A       129.100.2.12
    IN WKS     129.100.2.12 UDP (domain tftp)
    IN WKS     129.100.2.12 TCP (nntp smtp
                 telnet domain finger ftp)
    IN MX      0       julian.uwo.ca.
    IN MX      10      hydra.uwo.ca.
```

Note that this makes it possible to list machines which don't have an IP address (eg. `uwocc1.uwo.ca` uses `hydra` as an `MX` gateway). The MX record lists a mail gateway. For example,

```
uwocc1.uwo.ca. IN MX 0  hydra.uwo.ca.
               IN MX 10 julian.uwo.ca.
```

We want all machines to have an `MX` record (perhaps pointing to `void.uwo.ca`) so mail doesn't sit in mail queues for days. If the machine can't accept mail it should bounce immediately!

The is no machine called `void.uwo.ca` so mail will bounce if the MX record points at this machine.



### 2.3.9. IN HINFO

Describes Host Information — machine type and operating system (documentation only).

```
julian.uwo.ca. IN A       129.100.2.12
    IN WKS     129.100.2.12 UDP (domain tftp)
    IN WKS     129.100.2.12 TCP (nntp smtp
                 telnet domain finger ftp)
    IN MX      0       julian.uwo.ca.
    IN MX      10      hydra.uwo.ca.
    IN HINFO   "CDC 4680" "EP/IX"
```

The `HINFO` record has two arguments — machine type and operating system. Use quotes if there's white space in either argument.

## 2.3.10. IN UINFO

Describes User Information (eg. who is the contact for this system). This is documentation only.

```
ccs.uwo.ca.  IN SOA ..etc.
       IN UINFO   "Computing & Communications Services"
       IN UINFO   "Natural Sciences Centre"
       IN UINFO   "Contact: Peter Marshall <peter@.....
```

Documenting information of this sort makes it possible to find the person in charge. This is particularly important when problems arise. The operators use this information when they notice a problem on a network — who should they contact?

Actually UINFO records are implemented by BIND but *not* defined in the RFC's — they define a TXT record.

In *practice* few implementations support TXT records; hence we use UINFO records to insert textual commentary.



## 2.3.11. IN PTR

Pointer records just point at a domain name. The most important (only?) application is the mapping of network numbers back to domain names. A domain with a PTR record has (should have) no other records.

The gethostbynumber(129.100.2.13) is implemented by resolver queries that ask for the PTR record for 13.2.100.129.in-addr.arpa. Note it returns one name — that's the official name of the machine.

The in-addr.arpa domain is a functional map, ie. only on PTR record for each domain. To list any other is a mistake. For example, although we have

```
phonebook.uwo.ca.  IN A      129.100.2.13
hydra.uwo.ca.      IN A      129.100.2.13
uwovax.uwo.ca.     IN A      129.100.2.13
```

There's only *one* record for the reverse map, it says:

```
13.2.100.129.in-addr.arpa. IN PTR hydra.uwo.ca.
```

The official name of this machine machine is **hydra.uwo.ca**.

Recall the named.boot file for julian had these records:

```
primary   100.129.in-addr.arpa  in-addr.arpa
primary   0.0.127.in-addr.arpa  named.local
```

There's a root server for in-addr.arpa. (ie. `NS.NIC.DDN.MIL`); it has `NS` records to subdomain servers.

```
> set type=ns
> 100.129.in-addr.arpa.
Server:  ns.nic.ddn.mil
Address:  192.67.67.53

100.129.in-addr.arpa nameserver = julian.uwo.ca
100.129.in-addr.arpa nameserver = hydra.uwo.ca
100.129.in-addr.arpa nameserver = rri.uwo.ca
```

The in-addr.arpa domain consists of IP numbers written in a reverse notation (domain heirarchies work right to left, IP heirarchies work the other way).



For example, gp.uwo.ca is a subdomain of uwo.ca while [129.100.10.0] is a subnet of [129.100.0.0]. So the trick is to write network numbers backwards.

Records in the file in-addr.arpa on julian look like:

```
11.34.100.129.in-addr.arpa. IN PTR \
                            wkstn11.slis.uwo.ca.
11.36.100.129.in-addr.arpa. IN PTR \
                            tyrfingr.astro.uwo.ca.
11.37.100.129.in-addr.arpa. IN PTR \
                            seispc1.gp.uwo.ca.
11.38.100.129.in-addr.arpa. IN PTR \
                            vaxi.sscl.uwo.ca.
```

...etc

At UWO people who administer DNS data for their site include the `in-addr.arpa.` map as a speciallly formatted **comment** in the primary data files for their domain. Those comments are pulled out and used to construct the primary data file for `100.129.in-addr.arpa`.

```
[3:12pm julian] page data/ccs.domain
; Authoritative data for Computing and Communi..
; the University of Western Ontario (ccs.uwo.ca)
; ...
;1.1.100.129.in-addr.arpa.  IN PTR romulus.ccs.uwo.ca.
;1.65.100.129.in-addr.arpa. IN PTR romulus.ccs.uwo.ca.
```

...etc

```
[3:12pm julian] page in-addr.arpa
; Authoritative data for the University of West..
; ....
; The following from official data provided to ..
; #### Wed Nov 11 17:08:09 EST 1992 ####
1.1.100.129.in-addr.arpa.  IN PTR romulus.ccs.uwo.ca.
1.65.100.129.in-addr.arpa. IN PTR romulus.ccs.uwo.ca.
```

...etc

The management of the many DNS data files and construction of derivative data (like `/etc/hosts`, `in-addr.arpa` maps, mailing lists, etc.) is largely automated by a `Makefile`.



We also use `PTR` records to document ether net addresses and to document novell server numbers:

```
[10:43am julian] page named.boot
; ...
; Miscellaneous and non-standard maps
;
primary in-ether.arpa.  in-ether.arpa
primary in-novell.arpa. in-novell.arpa
```

These maps, like the in-addr.arpa map, are maintained as commentary text in the SOA data managed by the technical contact:

```
[10:44am julian] page data/ccs.domain
; Authoritative data for Computing and Communi..
; the University of Western Ontario (ccs.uwo.ca)
; ...
;81640133.in-novell.arpa.    IN PTR dtp.ccs.uwo.ca.
;02608C977786.in-ether.arpa. IN PTR dtp.ccs.uwo.ca.
;33.67.100.129.in-addr.arpa. IN PTR dtp.ccs.uwo.ca.
```

These extra (and non standard) maps make it possible for Network Operations to chase back by Novell Network and/or Ethernet numbers.

### 2.3.12. IN Miscellaneous Records

There are many more internet records

(1) `MB` a mailbox domain name (domain)

(2) `MG` a mail group member (domain)

(3) `MR` a mail rename domain name (domain)

(4) `NULL` a null resource record (no format or data)

(5) `MINFO` mailbox or mail list information (request_domain error_domain)

I've not had an occasion to use any of these and I believe most implementations don't handle them. They seem to be for managing mailing lists but I'm not aware of any mailer (eg. `sendmail`) that uses them.



## 3. Putting it all together

The *first* step is to register a Domain Name and Network Number with the campus NIC (by mail to NIC@UWO.CA).

Important registration documents (available from the NIC or by anonymous ftp from `julian:~ftp/nic`):

REGISTRATION.README
Introduction to Registration Document (UWO)

REGISTRATION.DOMAIN
Domain Registration Document (UWO)

REGISTRATION.NETWORK
Network Registration Document (UWO)

These *contracts* delegate responsiblity for managing a domain or network to the Administrative and Technical Contacts listed.

## 3.1. Responsibility

In each case the registration documents **require** that the Technical Contact provide the NIC with DNS data. For example, from the Domain Registration contract:

> It is the responsiblity of the Technical Contact to provide the NIC with the data required of the Domain Name Service for this domain. Information must be accurate and up to date. Adding new machines to the network without informing the NIC is forbidden.

From the Network Registration contract:

> It is the responsiblity of the Technical Contact to provide the NIC with the data required of the Domain Name Service for this network. Information must be accurate and up to date. Adding new machines to the network without informing the NIC is forbidden.

These are important obligations that **must** be met.



## 3.2. Provide Information

From reggers Mon Nov 26 12:27:05 1990
To: psyhma@uwocc1.uwo.ca
Subject: Psych network/domain
Cc: les@vaxr.sscl.uwo.ca

Heather, thanks for filling out those registration documents. You’ve been assigned the IPnet 129.100.47.0. I’ll return copies of those documents under separate cover. Please keep copies on file at your site.

One of your responsiblities is to provide me (the NIC) with an accurate and up to date list of machines on your network. This is required information for the Domain Name Service and is an ongoing job -- you have to keep the NIC informed of changes to your network (eg. when machines come or go). You should start on this right away (ps. there’s a course on this very topic on Thurs 29th 1:30-3:30 in Kresege bldg room 7 you might want to attend).

Here’s what I need for each machine on your network.

(1) Name, eg. bert.psych.uwo.ca (you can pick any names you like so long as they end in psych.uwo.ca). You likely want to name your vax "psych.uwo.ca" and use it as your mail-drop for

your PCSA clients.

(2) IP number, eg. [129.100.47.10] (your network registration document details the numbers you can use and briefly describes configuring your systems. You can use numbers 10-254 on net 47).

(3) Machine type and operating System, eg. IBM PS/2 running MS/DOS

(4) Machine location, eg. Room 7002 SSC

(5) Contact point (ie. who is running this system and how do I find them?) eg. Heather Acton <psyhma@uwoccl>; xt. 4729

If you can provide me with this information for all the machines on your network then I'll construct the data file required for the Domain Name Service and I'll then turn the file over to you to maintain.

For more information about the net (eg. what are good names to use?) you should look at the anonymous ftp area on our Unix system `julian:~ftp/nic`. There's a ton of good reading material there and I'd encourage you to look at them (eg. there's good Intro to IP, Intro to Sysadmin, and other documents).

Finally, I gather Les has volunteered to help you install multinet on your vax/vms system. If you need any help from us, please let us know. For information about NCSA/Telnet and other IP tools for your PCs you should



talk with Eva Placko <eva@uwovax>; xt. 6027

At any time, if you need network help send a note to <nic@uwo.ca>

## 3.3. Your DNS data

From the information provided the NIC will construct the DNS data file for your site:

```
; Authoritative data for NAME DAEMON Mappings
;
; administered by:
;  John Brunet <a03@seis.gp.uwo.ca>; xt 3144
; and communicated to:
;  Network Information Centre <nic@uwo.ca>; xt 2151
;

$ORIGIN gp.uwo.ca.

; SOA means Start of Authority, who's in charge?

@ IN SOA julian.uwo.ca. john.seis.gp.uwo.ca. (
                1.008     ; Serial EDIT ME TOO!!
                3600      ; Refresh
                300       ; Retry
                3600000   ; Expire
                3600 )    ; Minimum
        IN      NS      julian.uwo.ca.  ; primary
        IN      NS      hydra.uwo.ca.   ; secondary
        IN      NS      rri.uwo.ca.     ; secondary
;       IN      NS      some-one.gp.uwo.ca.
```



```
; This *may* become authoritative data managed
; locally within Geophysics for the moment it is
; editted locally and communicated to the NIC when
; and as the environment changes.
;
; README.NOW:
;
; Everything on the IPnet should be registered in a
; domain server.  This even includes dumb clunky
; things like routers, terminal servers, and client
; only PC's.  it's important that you have your
; machines registered because when we see problems
; with the net we need to chase it back to a
; building, room, machine, phone no.  and
; responsible person. Adding new systems to the net,
; even if they don't send/receive mail, without
; having them properly registered is consider VERY
; BAD FORM.  Every machine should have an MX
; forwarder even if it isn't a valid mail address
; (the MX forwarder in that case will simply refuse
; the mail).
; ...etc..

;10.37.100.129.in-addr.arpa. IN PTR seis.gp.uwo.ca.
seis.gp.uwo.ca. IN  A       129.100.37.10
    IN WKS   129.100.37.10 TCP telnet ftp
    IN MX    0       seis.gp.uwo.ca.
    IN MX    10      julian.uwo.ca.
    IN HINFO Sun4/80 "UNIX (SunOS 4.0)"
```

## 3.4. Configure Resolver

Configure your site to *use* the resolver routines.

Recall that for Unix machines you need to construct the file `/etc/resolv.conf`; for Vax/VMS you set the logicals when you install Multinet.

For Sun machines integrating YP (or NIS these days) with the DNS will work. Two strategies:

(1) Don't use the hosts map in YP, use the DNS. Instructions for doing this are found as `julian:~ftp/pub/resolver`.

(2) Use the DNS as a back end to YP, follow the Sun instructions in the YP Makefile.

```
[1:49pm suncon]  page /etc/resolv.conf
nameserver 129.100.2.12
nameserver 129.100.7.100
nameserver 129.100.2.13
domain     ccs.uwo.ca
[1:51pm suncon] ftp nic.ddn.mil
Connected to nic.ddn.mil. etc...
```



## 3.5. Secondary Server

For those who wish to run a secondary name server (and I recommend that large sites do) there is a kit provided.

```
[11:48am shadow] ftp julian
Connected to julian.uwo.ca.
220 julian.uwo.ca FTP server (from 4.3BSD Tahoe)
Name (julian:reggers): anonymous
331 Guest login ok, send ident as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> cd nic/name.secondary
250 CWD command successful.
ftp> mget *
....
ftp> quit
[11:48am shadow] page README
```

Following the instructions should be a painless exercise, others have done it:

```
[1:53pm charlie] page /etc/resolv.conf
nameserver 129.100.11.5
nameserver 129.100.11.251
nameserver 129.100.2.12
nameserver 129.100.7.100
domain     csd.uwo.ca
```

## 3.6. Primary Server

For those who wish to run a primary name server (and I recommend this for some of the larger sites).

(1) Demonstrate that you can keep your DNS data up to date.

(2) Demonstrate that you can run a Secondary Server.

(3) Coordinate with the NIC to set up a Primary Server.

An example for running a Primary Name Server is found as `julian:~ftp/nic/named` (configuration for `mcu.on.ca`).



# 4. HELP!

The campus NIC is there to help and will help. CCS will provide a liaison to those who need one.

The NIC at CCS/UWO has the experience. Trust their judgement.

The campus NIC has accumulated lots of good reading material — but you have to read it. Start by getting the README file from `julian:~ftp/nic`

You have to take on some responbility. Departmental Networks are not managed by the NIC at CCS, they're managed by Administrative and Technical contacts within the department.

The Internet will work if everyone does their job, please make sure you do yours.