DNS

This lecture describes the Domain Name System, a distributed database used to store information used by many Internet services. The most important example is probably the use of the DNS to store IP addresses corresponding to domain names. After this lecture you should be able to: distinguish between authoritative, recursive and secondary name servers, write BIND-format A, NS, MX, and CNAME DNS records, write the nslookup command that would be used to look up a particular DNS record from the default DNS server, write the sequence of DNS queries that would be required to obtain an authoritative DNS query result, explain purpose of DNS record caching, and predict when updated DNS records will become effective based on TTL values of cached data, and write the domain name that would be used to look up the domain name corresponding to an IP address.
Modified 2015-3-31 to add diagram showing different name servers.

Domain Names

Although IP addresses are required to route IP frames between computers using IP, they are hard to remember. Computers are often given names because they are easier to remember than numbers.

A hierarchical naming scheme called the Domain Name System (DNS) was developed for the Internet in the early 1980's. Using a naming technique similar to that used in file systems, the naming scheme uses a sequence of names separated by periods.

Exercise 1: Compare the file (path) name `/usr/local/src/bind/hosts.txt` and the domain name `red.sf.isc.com`. In each case what is the most specific portion of the name? What is the least-specific? What characters are used to separate different parts of the names? What would be the (null) name of the top level of the hierarchy?

In domain names the host name comes first and is at lowest level of hierarchy while each position to the right is a higher level in the hierarchy. At the highest level of the hierarchy (Top Level Domain) we can have a country code (.us, .ca, .uk, called ccTLD), or generic domains (.edu, .com, .org called gTLD).

Hosts File

In the earliest days it was common for host names to be stored in a file called hosts (`/etc/hosts`)/. This hosts file still exists on most operating systems[1] and has one line per host, with the IP address followed by the hostname(s). For example:

```
127.0.0.1 localhost
142.232.77.1 bcit.ca www.bcit.ca
```

While this works well for small private networks it is impractical for large networks like the public Internet. Today hosts file are only used for names that may be required when networking is not available and that don't change (e.g. localhost at 127.0.0.1).

The Domain Name System

Static host files were replaced by a distributed network-based database that can, in principle, be used to store any type of data not just IP addresses.

The operation of the DNS is defined in RFC's RFC1034 and RFC1035. These define how the database is organized and the protocols used to query the database.

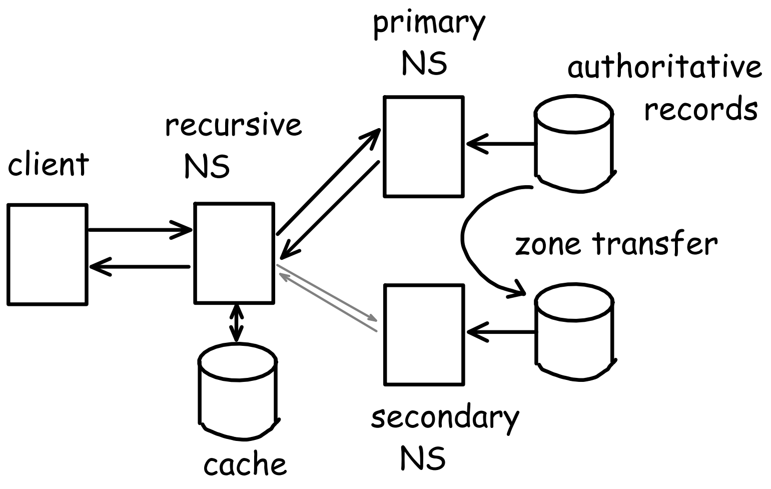
DNS Records

The data stored in the DNS is organized in records called Resource Records (RRs). Each RR has a domain name (the key into the database), a type and a value.

The most common types of records are:

- SOA: start of authority (various miscellaneous bits of information about a domain)
- A: IP address (the IPv4 address for the domain name)
- MX: Mail Exchangers (the host that is able to accept mail for that domain)

[1] For example, on Windows 7 at `/Windows/System32/drivers/etc/hosts`.



- NS: the IP address of a host that is an authoritative DNS server for the domain name
- PTR: a link to another domain name
- CNAME: canonical name (the "real" domain name for a domain)

Although DNS records are transmitted between clients and servers in binary format, there is a semi-standard way of representing RRs in text format. This is the format understood by the BIND software that is used by many DNS servers.

The DNS information is typically entered into a text file in BIND format and it is then read by the DNS server software.

Figure 1 shows a commented DNS record from Wikipedia.

Exercise 2: Based on these records, what is the IP address of ns.example.com? What is the IP address of example.com? What IP addresses would you connect to in order to send e-mail addressed to mail.example.com?

Recursive Lookup

The data stored in the DNS is distributed among many servers. Typically each organization operates it's own DNS servers which store and "serve" information related to the domain(s) used by that organization.

The leaves the problem of how to find the server responsible for a particular domain name. This is done by having each domain store NS records that point to the DNS servers for each subdomain of that domain.

A host wanting to look up the value of RR associated with a domain name can then start at the top of the domain hierarchy and query it for NS records for the desired subdomains until it reaches the DNS server for the desired information.

The DNS servers for the top-level (root) of the DNS hierarchy have to be hard-coded into the software.

Exercise 3: Explain the sequence of servers queried and the records retrieved to find the MX record for bcit.ca.

DNS Servers

Things would bog down very quickly if each host had to go through the above sequence of steps each time it wanted to get DNS information.

In practice, most hosts delegate DNS lookups to servers that provide DNS services for the hosts used by that organization. These servers remember ("cache") the results of DNS queries which can then be supplied to other hosts that make requests for the same information. For example, an ISP's DNS servers cache the results of their client's DNS queries and so only have to do lookups for unseen domain names or those whose records have expired.

Each result returned by a DNS server has a time to live (TTL) value associated with it. When this time expires the information needs to be retrieved again. This makes sure that DNS information can be updated. The authoritative DNS server is responsible for setting the TTL value on any particular RR.

```
ORIGIN example.com.     ; designates the start of this zone file in the namespace
$TTL 1h              ; default expiration time of all resource records without their own TTL value
example.com.  IN  SOA  ns.example.com. username.example.com. (
              2007120710 ; serial number of this zone file
              1d         ; slave refresh (1 day)
              2h         ; slave retry time in case of a problem (2 hours)
              4w         ; slave expiration time (4 weeks)
              1h         ; maximum caching time in case of failed lookups (1 hour)
              )
example.com.  NS    ns                    ; ns.example.com is a nameserver for example.com
example.com.  NS   ns.somewhere.example. ; ns.somewhere.example is a backup nameserver for example.com
example.com.  MX   10 mail.example.com.  ; mail.example.com is the mailserver for example.com
@           MX   20 mail2.example.com. ; equivalent to above line, "@" represents zone origin
@           MX   50 mail3           ; equivalent to above line, but using a relative host name
example.com.  A     192.0.2.1             ; IPv4 address for example.com
ns            A     192.0.2.2             ; IPv4 address for ns.example.com
www           CNAME example.com.          ; www.example.com is an alias for example.com
wwwtest     CNAME www              ; wwwtest.example.com is another alias for www.example.com
mail          A     192.0.2.3             ; IPv4 address for mail.example.com,
                                          ;  any MX record host must be an address record
                                          ; as explained in RFC 2181 (section 10.3)
mail2         A     192.0.2.4             ; IPv4 address for mail2.example.com
```

Figure 1: Sample DNS “zone” record.

Types of Servers

DNS servers can be of two types.

Authoritative name servers only return information they are configured with and that they are responsible for. They do not return information retrieved from other servers. They do not need to cache information.

Recursive name servers query other name servers in a recursive fashion until they reach a server that has the desired information (either from its cache or if it is the authoritative name server for the required information) and then returns these results.

In practice, many DNS servers combine these two functions.

It is often desirable to have redundant DNS servers. *Secondary* DNS servers act as backups for authoritative *Primary* DNS servers. They are listed the same way as authoritative DNS servers in NS RRs but periodically reload their database from the primary DNS server.

The figure above summarizes the different types of servers.

Exercise 4: If you were updating the RR for a domain, what type(s) of DNS server(s) would you have to update?

DNS Queries

DNS queries typically use UDP (although TCP is used for “zone transfers” from primary to secondary DNS servers). Servers listen on port 53 (called the “domain” service). Each DNS query specifies the type of record being requested and the key (typically a domain name). The response will be either the requested information or an error code.

DNS Query Utilities

Networking programs such as web browsers call library functions to do DNS queries. These functions then exchange messages with recursive DNS servers to get the necessary information.

There are command-line programs that can be used run DNS queries and display the results. These are useful for testing DNS servers.

The most widely-available of these programs is `nslookup`. The -type option can be used to select the type of record desired, the first argument is the domain name (the database look up “key”) and the optional second argument is the server to query. For example:

```
nslookup -type=mx bcit.ca 8.8.8.8
```

would query the DNS server at IP address 8.8.8.8 (Google's public DNS server) for the mail exchanger records for the bcit.ca domain. The program also has an interactive mode entered by running the program without arguments.

Exercise 5: How would you look up the NS RR for the `.ca` domain? What type of DNS server is the server at 8.8.8.8?

Reverse Lookup

It is often useful to be able to look up the domain name corresponding to an IP address. Most owners of IP addresses have entered their addresses into a separate DNS hierarchy that allows efficient retrieval of the host name from an IP address. The top-level domain of this hierarchy is in-addr.arpa and subdomains correspond to bytes of the IP address in order from least to most significant byte. For example, to look up the host name corresponding to the IP address 216.32.180.22 we would use the command:

```
nslookup -type=ptr 22.180.32.216.in-addr.arpa
```

Exercise 6: Look up the IP address for google.com. What is returned by the reverse DNS lookup?