DNS Session 1: Fundamentals

Brian Candler
NSRC

Computers use IP addresses. Why do we need names?

➤ Easier for people to remember
➤ Computers may be moved between networks, in which case their IP address will change

Old solution: hosts.txt

➤ A centrally-maintained file, distributed to all hosts on the Internet

<table>
<thead>
<tr>
<th>SPARKEY</th>
<th>128.4.13.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCB-MAILGATE</td>
<td>4.96.133.7</td>
</tr>
<tr>
<td>FTPHOST</td>
<td>200.10.194.33</td>
</tr>
<tr>
<td>... etc</td>
<td></td>
</tr>
</tbody>
</table>

hosts.txt doesn’t scale

➤ Huge file
➤ Needs frequent copying to ALL hosts
➤ Consistency
➤ Always out-of-date
➤ Name uniqueness
➤ Single point of administration

The Domain Name System was born

➤ DNS is a Distributed Database for holding name to IP address (and other) information
➤ Distributed:
  ➤ Shares the administration
  ➤ Shares the load
➤ Robustness and performance through:
  ➤ Replication
  ➤ Caching
➤ A critical piece of Internet infrastructure

DNS is Hierarchical

Forms a tree structure
DNS is Hierarchical (2)

➤ Gives globally unique names
➤ Administered in zones (parts of the tree)
➤ You can give away ("delegate") control of part of the tree underneath you
➤ Example:
  → isoc.org on one set of nameservers
  → isocws.isoc.org on a different set
  → t1.isocws.isoc.org on another set

Domain Names are (almost) unlimited

➤ Max 255 characters total length
➤ Max 63 characters in each part
  → RFC 1034, RFC 1035
➤ If a domain name is being used as a host name, you should abide by some restrictions
  → RFC 952 (old)
  → a-z 0-9 and minus (-) only
  → No underscores (_)

Using the DNS

➤ A Domain Name (like www.tiscali.co.uk) is the KEY to look up information
➤ The result is one or more RESOURCE RECORDS (RRs)
➤ There are different RRs for different types of information
➤ You can ask for the specific type you want, or ask for "any" RRs associated with the domain name

Commonly seen RRs

➤ A (address): map hostname to IP address
➤ PTR (pointer): map IP address to name
➤ MX (mail exchanger): where to deliver mail for user@domain
➤ CNAME (canonical name): map alternative hostname to real hostname
➤ TXT (text): any descriptive text
➤ NS (name server), SOA (start of authority): used for delegation and management of the DNS itself

Simple example

➤ Query: www.tiscali.co.uk
➤ Query type: A
➤ Result:
  www.tiscali.co.uk. IN A 212.74.101.10

In this case just a single RR is found, but in general, multiple RRs may be returned
(N is the "class" for INTERNET use of the DNS)

Possible results

➤ Positive (one or more RRs found)
➤ Negative (definitely no RRs match the query)
➤ Server fail (cannot find the answer)
How do you use an IP address as the key for a DNS query?

- Convert the IP address to dotted-quad
- Reverse the four parts
- Add ".in-addr.arpa." to the end; special domain reserved for this purpose
  e.g. to find name for 212.74.101.10

  10.101.74.212.in-addr.arpa.
  ➔ PTR www.tiscali.co.uk.

Known as a "reverse DNS lookup" (because we are looking up the name for an IP address, rather than the IP address for a name)

Dress is a Client-Server application

- (Of course - it runs across a network)
- Requests and responses are normally sent in UDP packets, port 53
- Occasionally uses TCP, port 53
  ➔ for very large requests, e.g. zone transfer from master to slave

There are three roles involved in DNS

- Resolver
  ➔ Takes request from application, formats it into UDP packet, sends to cache
- Caching Nameserver
  ➔ Returns the answer if already known
  ➔ Otherwise searches for an authoritative server which has the information
  ➔ Caches the result for future queries
  ➔ Also known as RECURSIVE nameserver
- Authoritative Nameserver
  ➔ The SAME protocol is used for resolver-cache and cache-authoritative communication
  ➔ It is possible to configure a single name server as both caching and authoritative
  ➔ But it still performs only one role for each incoming query
  ➔ Common but NOT RECOMMENDED to configure in this way (see later)
ROLE 1: THE RESOLVER

➢ A piece of software which formats a DNS request into a UDP packet, sends it to a cache, and decodes the answer
➢ Usually a shared library (e.g. libresolv so under Unix) because so many applications need it
➢ EVERY host needs a resolver - e.g. every Windows workstation has one

How does the resolver find a caching nameserver?

➢ It has to be explicitly configured (statically, or via DHCP etc)
➢ Must be configured with the IP ADDRESS of a cache (why not name?)
➢ Good idea to configure more than one cache, in case the first one fails

How do you choose which cache(s) to configure?

➢ Must have PERMISSION to use it
  ➢ e.g. cache at your ISP, or your own
➢ Prefer a nearby cache
  ➢ Minimises round-trip time and packet loss
  ➢ Can reduce traffic on your external link, since often the cache can answer without contacting other servers
➢ Prefer a reliable cache
  ➢ Perhaps your own?

Resolver can be configured with default domain(s)

➢ If "foo.bar" fails, then retry query as "foo.bar.mydomain.com"
➢ Can save typing but adds confusion
➢ May generate extra unnecessary traffic
➢ Usually best avoided

Example: Unix resolver configuration

/etc/resolv.conf

```
search tiscali.co.uk
nameserver 212.74.112.66
nameserver 212.74.112.67
```

That's all you need to configure a resolver

Testing DNS

➢ Just put "www.yahoo.com" in a web browser?
➢ Why is this not a good test?
Testing DNS with "dig"

- "dig" is a program which just makes DNS queries and displays the results
- Better than "nslook_up", "host" because it shows the raw information in full

```bash
dig tiscali.co.uk.
-- defaults to query type "A"
dig tiscali.co.uk. mx
-- specified query type
dig @212.74.112.66 tiscali.co.uk. mx
-- send to particular cache (overrides /etc/resolv.conf)
```

Interpreting the results: header

- **STATUS**
  - NOERROR: 0 or more RRs returned
  - NXDOMAIN: non-existent domain
  - SERVFAIL: cache could not locate answer
- **FLAGS**
  - AA: Authoritative answer (not from cache)
  - QR: Query/Response (1 = Response)
  - RD: Recursion Desired
  - RA: Recursion Available

Interpreting the results

- Answer section (RRs requested)
  - Each record has a Time To Live (TTL)
  - Says how long the cache will keep it
- Authority section
  - Which nameservers are authoritative for this domain
- Additional section
  - More RRs (typically IP addresses for the authoritative nameservers)
- Total query time
- Check which server gave the response!
  - If you make a typing error, the query may go to a default server

Practical Exercise

- Configure Unix resolver
- Issue DNS queries using 'dig'
- Use tcpdump to show queries being sent to cache