Domain Name System

Outline

- What and why?
- Structure of DNS
- Management of Domain Names
- Name Service in Practice

Need

- Network addresses are numbers
- Addresses are topologically oriented
  - Used for routing purposes
  - Moving a host may require change of address
  - Are not easy to remember
- Names can be used for users *and* for applications
  - Easy for humans
  - Can be used as a low level service discovery mechanism
  - Changing the server machine requires just changing the name-> IP binding
  - Names can have a logical structure

Some history

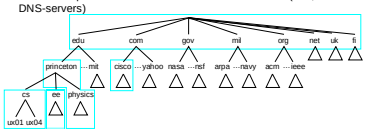
- In the beginning, there was the *hosts.txt*
  - A file containing the names and addresses of all hosts in the network
  - Problems: maintainability, size
  - Still used as a backup (local network host information)
- IEN-116 Name service
  - Non scaleable, topology-oriented
- DNS
  - Tree-structured
  - Delegation
  - Separated from network structure and topology
  - uses UDP, port number 53 for queries, TCP for zone transfers

DNS system

- Terms:
  - **namespace** = set of possible names, flat or hierarchical
  - naming system maintains a collection of **bindings** of names to values
  - given a name, a **resolution mechanism** returns the corresponding value
  - a **name server** is an implementation of the resolution mechanism
  - DNS (Domain Name System) = name service in Internet
  - Zone is an administrative unit, domain is a subtree

DNS domain hierarchy

- First level hierarchy
  - domains for each country + edu., com., gov., mil., org., net., int.
  - New domains: aero., biz., coop., info., museum., name., pro.
  - DNS first level managed by Internet Corporation for Assigned Names and Numbers (ICANN), also manages address allocations
- Hierarchy is partitioned into subtrees, zones
  - zone corresponds to administrative boundaries in DNS (and, often also of DNS-servers)



Name structure

1. Root "."
   - 13 root servers (one backup in Finland)
   - Know where to find the addresses of all the hosts in the world
   - Targets of DDoS attacts
2. Top level domains (common domain names + country codes)
3. Organisation type domain name
   - (such as co, edu)
   - In some countries, e.g. uk
4. Organisation
   - Abbreviation (hut, tkk)
   - Registered trademark
   - Full name
5. Organisation subdomain
   - Within organisation, e.g. tct
6. Host name

- Each part at most 63 characters
- The whole name at most 256 characters
- [A-Za-z0-9\-]
- Case insensitive

Fully Qualified Domain Name (FQDN)

- host.suborg*.org.type.tld.
- Hostname + domain name + "."
- Read from right to left
- A host can be addressed by
  - FQDN
  - Hostname + partial domain names
- E.g.
  - www.netlab.hut.fi. (FQDN)
  - www.netlab.hut.fi (host + partial domain name (subject to supplements))
  - www.netlab

Getting a domain name

- TLD: ICANN delegated name registrars
- Country level: local administrations.
- Finland:
  - Ficora (Viestintävirasto)
  - Companies, registered associations
  - For public institutions, their name or abbreviation
  - Must not violate registered trademarks

Elements

- RESOLVER
  - A library within the operating system, provides an API and handles queries
  - Contains a cache
- PRIMARY NAME SERVER
  - One per domain. Contains the binding information for all hosts
- SECONDARY NAME SERVER
  - Duplicates the information of primary servers, used for sharing load and for reliability
- CACHE NAME SERVER
  - Contains cache, but no binding info. Queries other DNS servers
- PROXY NAME SERVER
  - As cache NS but without cache :) For load balancing etc.

Bind (1)

- Zones defined in two or more name servers (redundancy)
  - clients send queries to name servers
  - servers response with final answer or pointer to another server
- Name binding database consists of resource records
  - format: <Name, Value, Type, Class, TTL>
  - Type: how Value is interpreted,
    - A: means that Value is an IPv4 address, name-address mapping
    - AAAA, A6: IPv6 address
    - NS: Value contains name for host that knows how to resolve the name
    - CNAME: Value is a canonical name for host, used to define aliases
    - DNAME: Subdomain redirecting
    - HINFO: Host information
    - MX: Value gives the domain name for a host running a mail server
    - PTR: Pointer to domain name (reverse DNS)

Bind (2)

  - RP: Responsible person
  - LOC: co-ordinates of the host
  - TXT: Free text
  - SIG, TSIG, KEY,CERT: security attributes
  - Class: only widely used class IN (Internet)
  - TTL: how long resource record is valid (used by servers that cache resource records from other servers)
- can use service specific aliases (www, smtp, nntp, print, etc.)
- MX allows administrators to redirect all mail of a host to a specified mail server

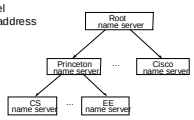
Example

```
tct.hut.fi IN SOA keskus.tct.hut.fi. puhuri.tct.hut.fi. (
        101008602 ; serial number
        10800 ; Refresh 3 hours
        3600 ; Retry 1 hour
        604800 ; Expire 1 week
        86400 ) ; TTL 1 day

        IN NS keskus.tct.hut.fi. ; primary name server
        IN NS ns1.hut.fi. ; first secondary
        IN NS ns2.hut.fi. ; second secondary

        IN MX 10 keskus ; primary mail server
        IN MX 20 smtp-1.hut.fi. ; backup
        IN MX 20 smtp-2.hut.fi. ; second backup

keskus  IN A 130.233.154.176
        IN MX 10 keskus

www     IN CNAME keskus
smtp    IN CNAME keskus

kytkin.ne IN A 10.0.0.1
```


DNS domain hierarchy (cont)

- Root name server: NS record for each 2nd level server + A record that translates name into IP address

  <princeton.edu, cit.princeton.edu, NS,IN>
  <cit,princeton.edu, 128.196.128.233, A, IN>



- At 2nd level, records contain either final answers or pointer to 3rd level name servers

  <cs.princeton.edu, gnat.cs.princeton.edu, NS, IN>
  <gnat.cs.princeton.edu, 192.12.69.5, A, IN> (pair like above)

  <jupiter.physics.princeton.edu, 128.196.4.1, A, IN> (final record)

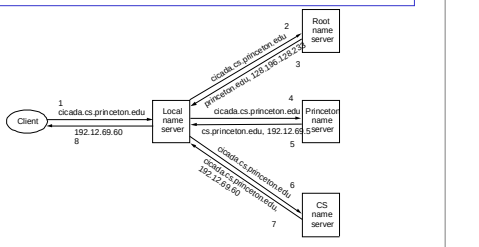
- Lowest level contains final records, aliases for hosts (CNAME) and MX records

Name resolution

- How did the client locate the root server in the first place?
  - name-to-address mapping for one or more name servers is well know (published outside the naming system itself)
  - in practice, resolver initialized with the address of a local name server
    - client makes a query to local server ⇒ local server makes queries further
    - advantages
      - only the servers need to know about root name servers
      - local server gets to see the responses (can cache these)
  - on a host running DNS (in Unix), try "dig", "nslookup", or "host <hostname>"

- Note: Internet has identifiers at several levels - domain names, IP addresses, and physical network addresses
  - users give domain names in applications ⇒ applications use DNS to translate these into IP addresses ⇒ IP does forwarding at each router, so it maps IP addresses into another (next hop router) ⇒ IP engages ARP to translate the next hop IP address into a physical address

Name resolution (cont)



Numbers (1-8) show the sequence of steps in the process

Reverse DNS

- Finding the name when knowing the address
- A different hierarchy: in-addr.arpa
- E.g. What is the hostname of 130.233.154.148 ?
  - Query 148.154.233.130.in-addr.arpa
- A separate hierarchy, organized as the address space
- Used for “security purposes”
  - A server might ask if the client name and address match

DNS as a Service

- Requires high reliability
- No single failure should affect -> servers located in different parts of the network
- E.g. .fi.
  - Hydra.helsinki.fi
  - ns-se.elisa.net
  - prifi.ficora.fi
  - ns1-fin.global.sonera.fi
  - t.ns.verio.net
  - ns.uu.net
- Difficult to organise -> Secondary DNS is an easy and important service to provide

Future

- Security still weak
- Using DNS as a directory structure (?)
  - Service Location
  - Generalisation of MX records
- Mapping Telephone numbers to IP addresses ?
  - Problems of policy (secret numbers, value)
- Character set