416 Distributed Systems

Feb 24, 2016 – DNS and CDNs
Outline

- DNS Design
- Content Distribution Networks
Naming

- How do we efficiently locate resources?
  - DNS: name $\rightarrow$ IP address

- Challenge
  - How do we scale this to the wide area?
Obvious Solutions (1)

Why not use /etc/hosts?

• Original Name to Address Mapping
  • Flat namespace
  • /etc/hosts
  • SRI kept main copy
  • Downloaded regularly

• Count of hosts was increasing: machine per domain → machine per user
  • Many more downloads
  • Many more updates
Obvious Solutions (2)

Why not centralize DNS?

- Single point of failure
- Traffic volume
- Distant centralized database
- Single point of update

- Doesn’t *scale*!
Domain Name System Goals

- Basically a wide-area distributed database
- Scalability
- Decentralized maintenance
- Robustness
- Global scope
  - Names mean the same thing everywhere
- Don’t need
  - Atomicity
  - Strong consistency
Programmer’s View of DNS

• Conceptually, programmers can view the DNS database as a collection of millions of host entry structures:

```c
/* DNS host entry structure */
struct addrinfo {
    int ai_family; /* host address type (AF_INET) */
    size_t ai_addrlen; /* length of an address, in bytes */
    struct sockaddr *ai_addr; /* address! */
    char *ai_canonname; /* official domain name of host */
    struct addrinfo *ai_next; /* other entries for host */
};
```

• Functions for retrieving host entries from DNS:
  • `getaddrinfo`: query key is a DNS host name.
  • `getnameinfo`: query key is an IP address.
DNS Records

RR format: (class, name, value, type, ttl)

- DB contains tuples called resource records (RRs)
  - Classes = Internet (IN), Chaosnet (CH), etc.
  - Each class defines value associated with type

**FOR IN class:**

- Type=A
  - name is hostname
  - value is IP address
- Type=NS
  - name is domain (e.g. foo.com)
  - value is canonical name
- Type=CNAME
  - name is an alias name for some “canonical” (the real) name
- Type=MX
  - value is hostname of mailserver associated with name
Properties of DNS Host Entries

- Different kinds of mappings are possible:
  - Simple case: 1-1 mapping between domain name and IP addr:
    - kittyhawk.cmcl.cs.cmu.edu maps to 128.2.194.242
  - Multiple domain names maps to the same IP address:
    - eecs.mit.edu and cs.mit.edu both map to 18.62.1.6
  - Single domain name maps to multiple IP addresses:
    - aol.com and www.aol.com map to multiple IP addrs.
  - Some valid domain names don’t map to any IP address:
DNS Design: Hierarchy Definitions

- Each node in hierarchy stores a list of names that end with same suffix
  - Suffix = path up tree
- E.g., given this tree, where would following be stored:
  - Fred.com
  - Fred.edu
  - Fred.cs.ubc.edu
  - Fred.ugrad.cs.ubc.edu
  - Fred.cs.mit.edu
DNS Design: Zone Definitions

- Zone = contiguous section of name space
  - E.g., Complete tree, single node or subtree
- A zone has an associated set of name servers
  - Must store list of names and tree links

Diagram:
- Complete Tree
- Single node
- Subtree
DNS Design: Cont.

• Zones are created by convincing owner node to create/delegate a subzone
  • Records within zone stored at multiple redundant name servers
  • Primary/master name server updated manually
  • Secondary/redundant servers updated by zone transfer of name space
    • Zone transfer is a bulk transfer of the “configuration” of a DNS server – uses TCP to ensure reliability

• Example:
  • CS.UBC.EDU created by UBC.EDU administrators
  • Who creates UBC.EDU or .EDU?
DNS: Root Name Servers

- Responsible for “root” zone
- Approx. 13 root name servers worldwide
  - Currently {a-m}.root-servers.net
- Local name servers contact root servers when they cannot resolve a name
  - Configured with well-known root servers
  - Newer picture → www.root-servers.org
Physical Root Name Servers

- Several root servers have multiple physical servers
- Packets routed to “nearest” server by “Anycast” protocol
- 346 servers total
Servers/Resolvers

- Each host has a resolver
  - Typically a library that applications can link to
  - Local name servers hand-configured (e.g. /etc/resolv.conf)

- Name servers
  - Either responsible for some zone or…
  - Local servers
    - Do lookup of distant host names for local hosts
    - Typically answer queries about local zone
Typical Resolution

Client

Local DNS server

NS ns1.cmu.edu

www.cs.cmu.edu

NS ns1.cs.cmu.edu

A www=IPaddr

ns1.cs.cmu.edu DNS server

ns1.cmu.edu DNS server

root & edu DNS server
Typical Resolution

- **Steps for resolving www.cmu.edu**
  - Application calls gethostbyname() (RESOLVER)
  - Resolver contacts local name server (S₁)
  - S₁ queries root server (S₂) for (www.cmu.edu)
  - S₂ returns NS record for cmu.edu (S₃)
  - S₁ queries S₃ for www.cmu.edu
  - S₃ returns A record for www.cmu.edu
Lookup Methods

Recursive query:
- Server goes out and searches for more info (recursive)
- Only returns final answer or “not found”

Iterative query:
- Server responds with as much as it knows (iterative)
- “I don’t know this name, but ask this server”

Workload impact on choice?
- Local server typically does recursive
- Root/distant server does iterative

Diagram:
- Requesting host: surf.eurecom.fr
- Local name server: dns.eurecom.fr
- Intermediate name server: dns.umass.edu
- Root name server: dns.cs.umass.edu
- Authoritative name server: gaia.cs.umass.edu
Workload and Caching

- Are all servers/names likely to be equally popular?
  - Why might this be a problem? How can we solve this problem?

- DNS responses are cached
  - Quick response for repeated translations
  - Other queries may reuse some parts of lookup
    - NS records for domains

- DNS negative queries are cached
  - Don’t have to repeat past mistakes
  - E.g. misspellings, search strings in resolv.conf

- Cached data periodically times out
  - Lifetime (TTL) of data controlled by owner of data
  - TTL passed with every record
Typical Resolution

Client → Local DNS server

- www.cs.cmu.edu
- ns1.cs.cmu.edu
- ns1.cmu.edu

Local DNS server → root & edu DNS server

- www.cs.cmu.edu
- www.cs.cmu.edu
- www.cs.cmu.edu

ns1.cmu.edu DNS server

- A www=IPaddr
Subsequent Lookup Example

Client
ftp.cs.cmu.edu

Local DNS server

root & edu DNS server
cmu.edu DNS server
cs.cmu.edu DNS server

ftp.cs.cmu.edu
ftp=IPaddr
Reliability

- DNS servers are replicated
  - Name service available if $\geq$ one replica is up
  - Queries can be load balanced between replicas
- UDP used for queries
  - Need reliability $\rightarrow$ must implement this on top of UDP!
  - Why not just use TCP?
- Try alternate servers on timeout
  - Exponential backoff when retrying same server
- Same identifier for all queries
  - Don’t care which server responds
Reverse DNS

- **Task**
  - Given IP address, find its name

- **Method**
  - Maintain separate hierarchy based on IP names
  - Write 128.2.194.242 as 242.194.2.128.in-addr.arpa
    - Why is the address reversed?

- **Managing**
  - Authority manages IP addresses assigned to it
  - E.g., CMU manages name space 128.2.in-addr.arpa
.arpa Name Server hierarchy/replication

- At each level of hierarchy, have group of servers that are authorized to handle that region of hierarchy.

Diagram:

```
in-addr.arpa  a.root-servers.net  ...  m.root-servers.net
  |           |                     |
  128        |                     |
  |           |                     |
  2          |                     |
  |           |                     |
  194        |                     |
  |           |                     |
kittyhawk  128.2.194.242
```

- chia.arin.net
  (dill, henna, indigo, epazote, figwort, ginseng)

- cucumber.srv.cs.cmu.edu,
  t-ns1.net.cmu.edu
  t-ns2.net.cmu.edu

- mango.srv.cs.cmu.edu
  (peach, banana, blueberry)
Prefetching

- Name servers can add additional data to response
- Typically used for prefetching
  - CNAME/MX/NS typically point to another host name
  - Responses include address of host referred to in “additional section”
Tracing Hierarchy (1)

- Dig Program
  - Use flags to find name server (NS)
  - Disable recursion so that operates one step at a time

```
unix> dig +norecurse @a.root-servers.net NS
greatwhite.ics.cs.cmu.edu

;; ADDITIONAL SECTION:
a.edu-servers.net  172800 IN  A   192.5.6.30
c.edu-servers.net. 172800 IN  A   192.26.92.30
d.edu-servers.net. 172800 IN  A   192.31.80.30
f.edu-servers.net. 172800 IN  A   192.35.51.30
g.edu-servers.net. 172800 IN  A   192.42.93.30
g.edu-servers.net. 172800 IN  AAAA 2001:503:cc2c::2:36
l.edu-servers.net. 172800 IN  A   192.41.162.30
```

- All .edu names handled by set of servers