EECS 122, Lecture 4

Kevin Fall

kfall@cs.berkeley.edu

Domain Name System

- Internet (IP) only understands addresses
- Naming easier for humans (e.g. files)
- Need a way to map names to numbers
- DNS (Domain Name System):
  - hierarchical distributed database
  - Internet *application* layer
  - see RFC 1034 and 1035

Naming

- Important theme in systems engineering
  - files in a file system
  - processes in operating system
  - web pages
  - printers and other services
- Name and location decoupling

Decoupling

- DNS provides a *level of indirection* between name and its location (solves any CS problem!)
- How to do this?
  - Flat vs hierarchical name space
  - Distributed vs centralized approach

Original Name System

- Flat name space:
  - simple (string, address) pairs
  - manual coordination
  - examples: ucbvax, sdcsvax, sri-nic
- Centralized Management
  - HOSTS.TXT file
  - single point of failure/update

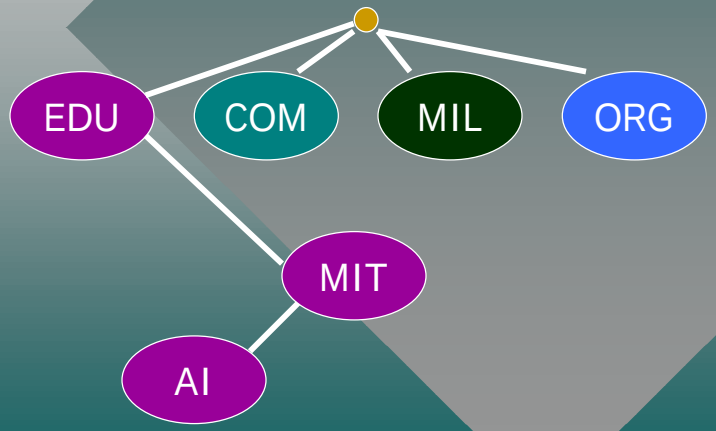
Scaling Problems

- Name space overlap
  - had to coordinate with other users of name space to avoid overlap
  - greater management time
- Inconsistencies and poor performance
  - centralized updates
  - single point of failure
  - congestion at central point

DNS Approach

- Hierarchical, nested domain naming
- Distributed, recursive servers
- Basic ideas:
  - distribute name/address database across network hierarchically
  - implement query/response protocol
  - use caching heavily

Naming Hierarchy



Naming Conventions

- "Top-Level Domains" (TLDs)
- Non-geographic
  - .COM, .NET, .ORG, .INT, .EDU, .MIL, .GOV
- Geographic
  - (baed on ISO3166 country codes)
  - .JP, .AU, .UK, .DE, .US, ...
- The special ".ARPA" (reverse) domain

Naming Example

- www.cs.Berkeley.EDU
  - host: www, subdomain cs, domain: Berkeley.EDU
  - case insensitive
  - a "fully qualified" domain name (FQDN)
- Hierarchy is right-to-left with "." delimiter
- Not necessarily tied to network topology/geography

DNS Components

- (Mockapetris & Dunlap, 1983, pub 1988)
- *Zones* contain *resource records* (RRs)
- *Name server(s)* manage each zone
- Client *resolvers* query name servers

Zones

- Complete description of a contiguous section of the total name space, plus some linkage info to other contig zones (separately administered DNS subtrees)
- Associated maintenance (>1 server)
- Zone transfers between redundant servers

Caching

- Servers and some clients *cache* data retrieved
- Resource records contain time-to-live (TTL), set by provider
- Higher TTL: less traffic, stale info
- Lower TTL: more traffic, current info

DNS Resource Records

- Components: owner (which domain), class (IN is only significant one), type, TTL (secs ok to cache), record data
- Types:
  - A, CNAME, HINFO, MX, NS, SOA, PTR
- Record Data
  - variable length, specific to type

Address-related Types

- **A** type (internet address(es))

```
www.AAD          A 128.32.51.214
```

- **CNAME** type (alias(es))

```
boalt CNAME boalt363-001-d6.Law.Berkeley.EDU
```

- **PTR** type (used for reverse queries)

```
214.51  PTR  www.AAD.Berkeley.EDU
```

Authority Record

- **SOA** type (start of authority)
  - current serial number of zone data
  - refresh, retry, and expire info
  - Berkeley.EDU SOA ns1.Berkeley.EDU dns-roadkill.NAK.Berkeley.EDU

```
90001481        ; serial (vers)
3600            ; refresh period
900             ; retry refresh this often
3600000         ; expiration period
86400           ; minimum TTL
```

Name Server Records

- **NS** type (name server)
  - indicates authoritative name servers
  - used to construct the hierarchy

```
Berkeley.EDU.   NS    vangogh.CS.Berkeley.EDU
Berkeley.EDU.   NS    cgl.UCSF.EDU
CS              NS    vangogh.CS.Berkeley.EDU
CS              NS    nexus.EECS.Berkeley.EDU
```

Other Records

- **MX** type (mail exchanger)
  - indicates e-mail relay host and its preference

```
Berkeley.EDU.   MX    5 mailhost.Berkeley.EDU
```

- **HINFO** type (host info)
  - indicates OS or type of host

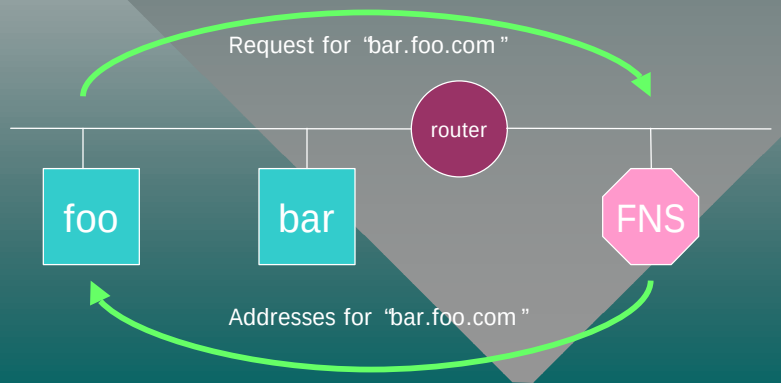
```
UCSD.EDU. HINFO      Sun        Unix
```

An Example (nslookup):

```
prompt> nslookup
> ls -t any cs.berkeley.edu
[cronus.CS.Berkeley.EDU]
 CS.Berkeley.EDU.               SOA   ns1.Berkeley.EDU hostmaster.NAK.Berkeley.EDU.
                                      (20001754 10800 3600 604800 3600)
 CS.Berkeley.EDU.               NS    vangogh.CS.Berkeley.EDU
 CS.Berkeley.EDU.               NS    nexus.EECS.Berkeley.EDU
 CS.Berkeley.EDU.               NS    huginn.CS.Berkeley.EDU
 CS.Berkeley.EDU.               NS    ns1.Berkeley.EDU
 CS.Berkeley.EDU.               NS    ns2.Berkeley.EDU
 CS.Berkeley.EDU.               NS    cgl.UCSF.EDU
 CS.Berkeley.EDU.               MX    10   alpaca.EECS.Berkeley.EDU
 CS.Berkeley.EDU.               A     128.32.35.123
 CS.Berkeley.EDU.               MX    20   mica.EECS.Berkeley.EDU
 CS.Berkeley.EDU.               MX    3    cs2.CS.Berkeley.EDU
 CS.Berkeley.EDU.               MX    5    hofmann.CS.Berkeley.EDU
 www-callug                     CNAME brain.CS.Berkeley.EDU
 sparc                          A     128.32.136.253
...
```

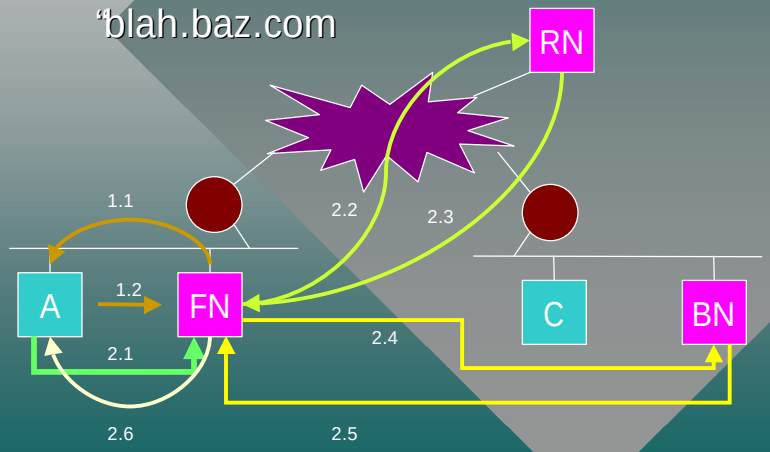
Locally-satisfied DNS query:

- User in domain "foo.com" asks for "bar"



Globally-satisfied DNS query:

- User in domain "foo.com" asks for "blah.baz.com



Reverse Queries

- Forward queries use domain name
- How to do reverse (addr-to-name) queries?
  - Addresses left-to-right, names right-to-left
  - Idea: REVERSE query
- Reverse network number, add ".IN-ADDR.ARPA" and perform PTR type query

Reverse Query Example

- Find the name of the host with address "208.212.172.33"
- This is a class "C" address, network 208.212.172.0
- So, look for the string "33.172.212.208.IN-ADDR.ARPA":

```
33.172.212.208.IN-ADDR.ARPA  PTR  www.nsa.gov.
```

Bootstrapping

- How does local host locate name server?
  - Set up during host configuration
- How do servers locate root servers?
  - Set up during DNS configuration
  - 13 root servers ([a-m].root-servers.net)
  - root servers do not provide recursion

Negative Caching

- Caching works well for correct queries
- With many wrong queries, scaling is hurt:
  - cache negative queries also!
  - Covers both nonexistent domain names and nonexistent resource records
- See RFC 2308
  - Set up during host configuration

DNS Protocol

- DNS is an Application
- Uses both TCP and UDP for transport
  - UDP: used for most queries
  - TCP: used for zone transfers, and when UDP results indicated message was too big
- Use of UDP requires clients to implement their own reliability

DNS Lessons

- Naming was first show-stopping scaling problem
- Scaling problem addressed with:
  - caching
  - decentralization
  - hierarchy