15-441 Computer Networking

Lecture 13 – DNS

Outline

- DNS Design
- DNS Today

Naming

- How do we efficiently locate resources?
  - DNS: name → IP address
- Challenge
  - How do we scale this to the wide area?

Obvious Solutions (1)

Why not centralize DNS?

- Single point of failure
- Traffic volume
- Distant centralized database
- Single point of update
- Doesn't *scale!*

Obvious Solutions (2)

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

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

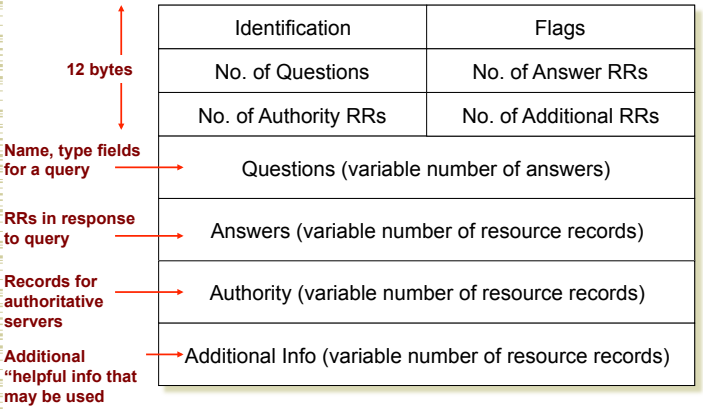
Programmer's View of DNS

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

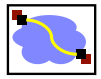
```
/* DNS host entry structure */
struct addrinfo {
        int     ai_family;      /* host address type (AF_INET) */
        size_t  ai_addrlen;     /* length of an address, in bytes */
        struct sockaddr *ai_addr;   /* address! */
        char    *ai_canonname;      /* official domain name of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

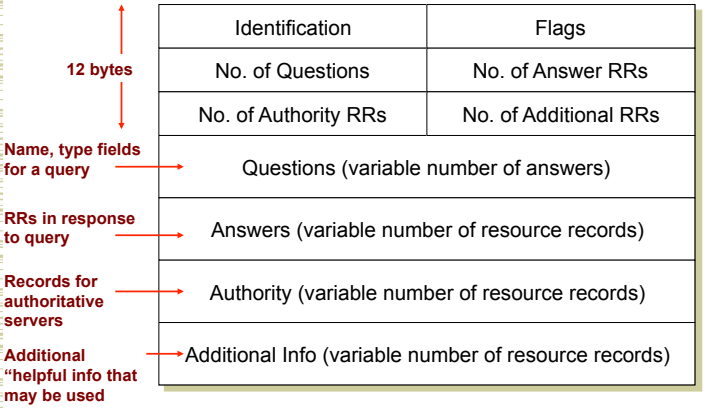
- Functions for retrieving host entries from DNS:
  - `getaddrinfo`: query key is a DNS host name.
  - `getnameinfo`: query key is an IP address.

DNS Message Format



DNS Message Format

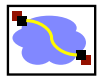




DNS Header Fields

- Identification
  - Used to match up request/response
- Flags
  - 1-bit to mark query or response
  - 1-bit to mark authoritative or not
  - 1-bit to request recursive resolution
  - 1-bit to indicate support for recursive resolution

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

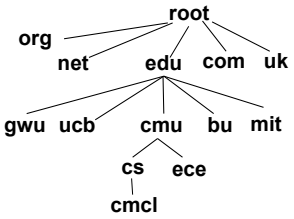
FOR IN class:

- Type=A
  - **name** is hostname
  - **value** is IP address
- Type=NS
  - **name** is domain (e.g. foo.com)
  - **value** is name of authoritative name server for this domain
- Type=CNAME
  - **name** is an alias name for some "canonical" (the real) name
  - **value** is canonical name
- Type=MX
  - **value** is hostname of mailserver associated with **name**

Properties of DNS Host Entries

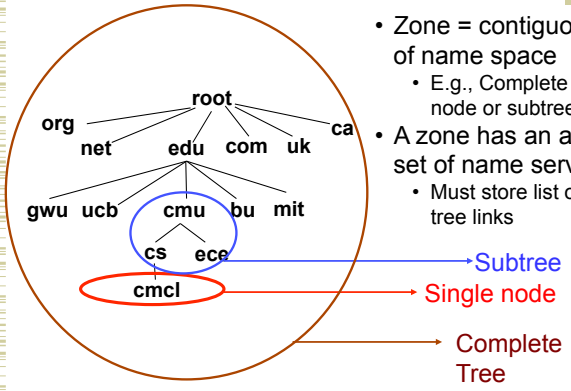
- Different kinds of mappings are possible:
  - Simple case: 1-1 mapping between domain name and IP addr:
    - `kittyhawk.cmcl.cs.cmu.edu` maps to `128.2.194.242`
  - Multiple domain names maps to the same IP address:
    - `eecs.mit.edu` and `cs.mit.edu` both map to `18.62.1.6`
  - Single domain name 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:
    - for example: `cmcl.cs.cmu.edu`

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.cmu.edu
  - Fred.cmcl.cs.cmu.edu
  - Fred.cs.mit.edu

DNS Design: Zone Definitions



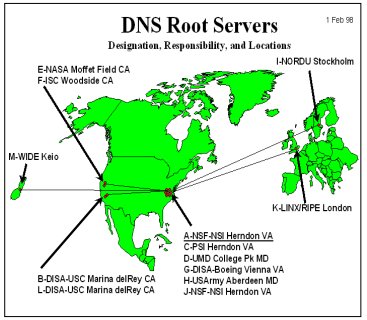
- Zone = contiguous section of name space
  - E.g., Complete tree, single node or subtree
- A zone has an associated set of name servers
  - Must store list of names and tree links

DNS Design: Cont.

- Zones are created by convincing owner node to create/delegate a subzone
  - Records within zone stored 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.CMU.EDU created by CMU.EDU administrators
  - Who creates CMU.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

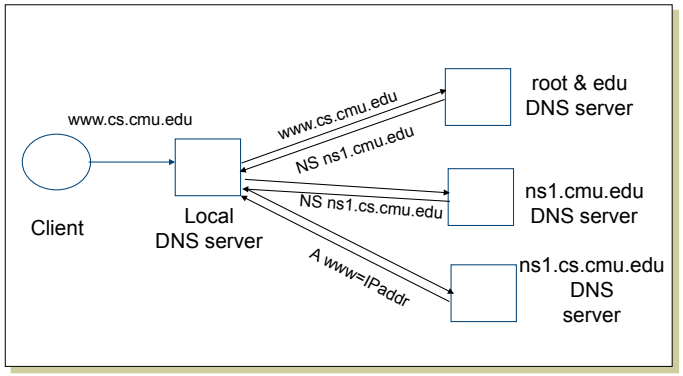


Servers/Resolvers

- 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

17

Typical Resolution



18

Typical Resolution

- Steps for resolving www.cmu.edu
  - Application calls gethostbyname() (RESOLVER)
  - Resolver contacts local name server (S_1)
  - S_1 queries root server (S_2) for (www.cmu.edu)
  - S_2 returns NS record for cmu.edu (S_3)
  - What about A record for S_3?
    - This is what the additional information section is for (PREFETCHING)
  - S_1 queries S_3 for www.cmu.edu
  - S_3 returns A record for www.cmu.edu

19

DNS Hack #1

- Can return multiple A records → what does this mean?
- Load Balance
  - Server sends out multiple A records
  - Order of these records changes per-client

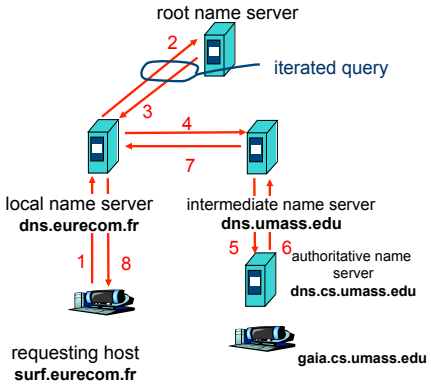
20

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"

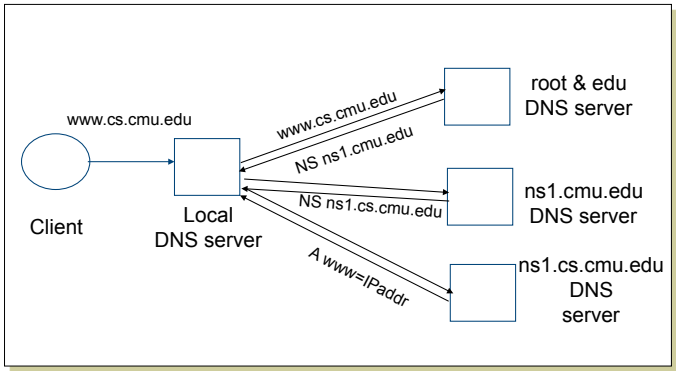
Workload impact on choice?
- Local server typically does recursive
- Root/distant server does iterative



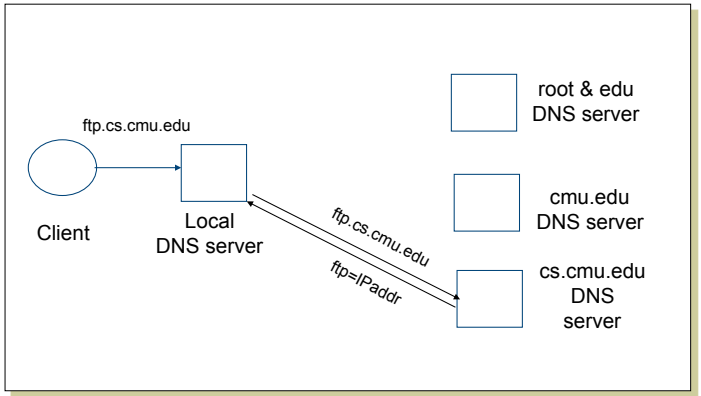
Workload and Caching

- 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



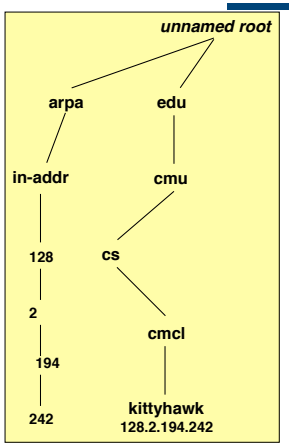
Subsequent Lookup Example



Reliability

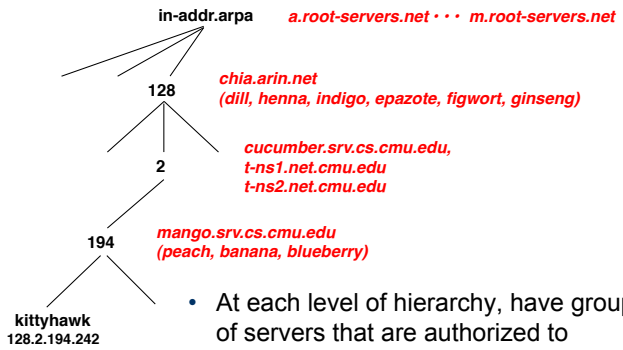
- DNS servers are replicated
  - Name service available if ≥ one replica is up
  - Queries can be load balanced between replicas
- UDP used for queries
  - Need reliability → must implement this on top of UDP!
  - Why not just use TCP?
- Try alternate servers on timeout
  - Exponential backoff when retrying 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 242.194.128.2.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



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

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"

Mail Addresses

- MX records point to mail exchanger for a name
  - E.g. mail.acm.org is MX for acm.org
- Addition of MX record type proved to be a challenge
  - How to get mail programs to lookup MX record for mail delivery?
  - Needed critical mass of such mailers

29

Outline

- DNS Design
- DNS Today

30

Root Zone

- Generic Top Level Domains (gTLD) = .com, .net, .org, etc…
- Country Code Top Level Domain (ccTLD) = .us, .ca, .fi, .uk, etc…
- Root server ({a-m}.root-servers.net) also used to cover gTLD domains
  - Load on root servers was growing quickly!
  - Moving .com, .net, .org off root servers was clearly necessary to reduce load → done Aug 2000

31

gTLDs

- Unsponsored
  - .com, .edu, .gov, .mil, .net, .org
  - .biz → businesses
  - .info → general info
  - .name → individuals
- Sponsored (controlled by a particular association)
  - .aero → air-transport industry
  - .cat → catalan related
  - .coop → business cooperatives
  - .jobs → job announcements
  - .museum → museums
  - .pro → accountants, lawyers, and physicians
  - .travel → travel industry
- Starting up
  - .mobi → mobile phone targeted domains
  - .post → postal
  - .tel → telephone related
- Proposed
  - .asia, .cym, .geo, .kid, .mail, .sco, .web, .xxx

32

New Registrars

- Network Solutions (NSI) used to handle all registrations, root servers, etc…
  - Clearly not the democratic (Internet) way
  - Large number of registrars that can create new domains → However NSI still handles A root server

33

Tracing Hierarchy (1)

- Dig Program
  - Allows querying of DNS system
  - 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 kittyhawk.cmcl.cs.cmu.edu

;; AUTHORITY SECTION:
edu.            172800  IN      NS      L3.NSTLD.COM.
edu.            172800  IN      NS      D3.NSTLD.COM.
edu.            172800  IN      NS      A3.NSTLD.COM.
edu.            172800  IN      NS      E3.NSTLD.COM.
edu.            172800  IN      NS      C3.NSTLD.COM.
edu.            172800  IN      NS      F3.NSTLD.COM.
edu.            172800  IN      NS      G3.NSTLD.COM.
edu.            172800  IN      NS      B3.NSTLD.COM.
edu.            172800  IN      NS      M3.NSTLD.COM.
```

  - All .edu names handled by set of servers

34

Tracing Hierarchy (2)

- 3 servers handle CMU names

```
unix> dig +norecurse @e3.nstld.com NS kittyhawk.cmcl.cs.cmu.edu

;; AUTHORITY SECTION:
cmu.edu.        172800  IN      NS      CUCUMBER.SRV.cs.cmu.edu.
cmu.edu.        172800  IN      NS      T-NS1.NET.cmu.edu.
cmu.edu.        172800  IN      NS      T-NS2.NET.cmu.edu.
```

35

Tracing Hierarchy (3 & 4)

- 4 servers handle CMU CS names

```
unix> dig +norecurse @t-ns1.net.cmu.edu NS kittyhawk.cmcl.cs.cmu.edu

;; AUTHORITY SECTION:
cs.cmu.edu.     86400   IN      NS      MANGO.SRV.cs.cmu.edu.
cs.cmu.edu.     86400   IN      NS      PEACH.SRV.cs.cmu.edu.
cs.cmu.edu.     86400   IN      NS      BANANA.SRV.cs.cmu.edu.
cs.cmu.edu.     86400   IN      NS      BLUEBERRY.SRV.cs.cmu.edu.
```

- Quasar is master NS for this zone

```
unix>dig +norecurse @blueberry.srv.cs.cmu.edu NS
        kittyhawk.cmcl.cs.cmu.edu

;; AUTHORITY SECTION:
cs.cmu.edu.     300     IN      SOA     QUASAR.FAC.cs.cmu.edu.
```

36

Do you trust the TLD operators?

- Wildcard DNS record for all .com and .net domain names not yet registered by others
  - September 15 – October 4, 2003
  - February 2004: Verisign sues ICANN
- Redirection for these domain names to Verisign web portal (SiteFinder)
- What services might this break?

Protecting the Root Nameservers

Attack On Internet Called Largest Ever

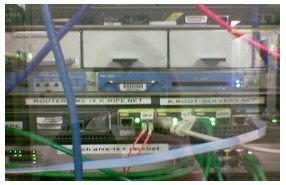
By David McGuire and Brian Krebs
washingtonpost.com Staff Writers
Tuesday, October 22, 2002; 5:40 PM

The heart of the Internet sustained its largest and most sophisticated attack ever, starting late Monday, according to officials at key online backbone organizations.

Around 5:00 p.m. EDT on Monday, a "distributed denial of service" (DDOS) attack struck the 13 "root servers" that provide the primary roadmap for almost all Internet communications. Despite the scale of the attack, which lasted about an hour, Internet users worldwide were largely unaffected, experts said.

Sophisticated?
Why did nobody notice?

seshan.org. 13759 NS www.seshan.org.



Defense Mechanisms

- Redundancy: 13 root nameservers
- IP Anycast for root DNS servers {c,f,i,j,k}.root-servers.net
  - RFC 3258
  - Most *physical* nameservers lie outside of the US

Defense: Replication and Caching

Letter	Old name	Operator	Location
A	ns.internic.net	VeriSign	Dulles, Virginia, USA
B	ns1.isi.edu	ISI	Marina Del Rey, California, USA
C	c.psi.net	Cogent Communications	distributed using anycast
D	terp.umd.edu	University of Maryland	College Park, Maryland, USA
E	ns.nasa.gov	NASA	Mountain View, California, USA
F	ns.isc.org	ISC	distributed using anycast
G	ns.nic.ddn.mil	U.S. DoD NIC	Columbus, Ohio, USA
H	aos.arl.army.mil	U.S. Army Research Lab	Aberdeen Proving Ground, Maryland, USA
I	nic.nordu.net	Autonomica	distributed using anycast
J		VeriSign	distributed using anycast
K		RIPE NCC	distributed using anycast
L		ICANN	Los Angeles, California, USA
M		WIDE Project	distributed using anycast

source: wikipedia

DNS Hack #2: Blackhole Lists

- First: Mail Abuse Prevention System (MAPS)
  - Paul Vixie, 1997
- Today: Spamhaus, spamcop, dnsrbl.org, etc.

Different addresses refer to different reasons for blocking

% dig 91.53.195.211.bl.spamcop.net

;; ANSWER SECTION:
91.53.195.211.bl.spamcop.net. 2100 IN A 127.0.0.2

;; ANSWER SECTION:
91.53.195.211.bl.spamcop.net. 1799 IN TXT "Blocked - see http://www.spamcop.net/bl.shtml?211.195.53.91"

DNS (Summary)

- Motivations → large distributed database
  - Scalability
  - Independent update
  - Robustness
- Hierarchical database structure
  - Zones
  - How is a lookup done
- Caching/prefetching and TTLs
- Reverse name lookup
- What are the steps to creating your own domain?

41

Measurements of DNS

- No centralized caching per site
  - Each machine runs own caching local server
  - Why is this a problem?
  - How many hosts do we need to share cache? → recent studies suggest 10-20 hosts
- "Hit rate for DNS = 80% → 1 - (#DNS/#connections)
  - Is this good or bad?
  - Most Internet traffic was Web with HTTP 1.0
    - What does a typical page look like? → average of 4-5 imbedded objects → needs 4-5 transfers
    - This alone accounts for 80% hit rate!
- Lower TTLs for A records does not affect performance
- DNS performance really relies more on NS-record caching

42

DNS Experience

- 23% of lookups with no answer
  - Retransmit aggressively → most packets in trace for unanswered lookups!
  - Correct answers tend to come back quickly/with few retries
- 10 - 42% negative answers → most = no name exists
  - Inverse lookups and bogus NS records
- Worst 10% lookup latency got much worse
  - Median 85→97, 90th percentile 447→1176
- Increasing share of low TTL records → what is happening to caching?

43

DNS Experience

- Hit rate for DNS = 80% → 1-(#DNS/#connections)
  - Most Internet traffic is Web
  - What does a typical page look like? → average of 4-5 imbedded objects → needs 4-5 transfers → accounts for 80% hit rate!
- 70% hit rate for NS records → i.e. don't go to root/ gTLD servers
  - NS TTLs are much longer than A TTLs
  - NS record caching is much more important to scalability
- Name distribution = Zipf-like = $1/x^a$
- A records → TTLs = 10 minutes similar to TTLs = infinite
- 10 client hit rate = 1000+ client hit rate

44