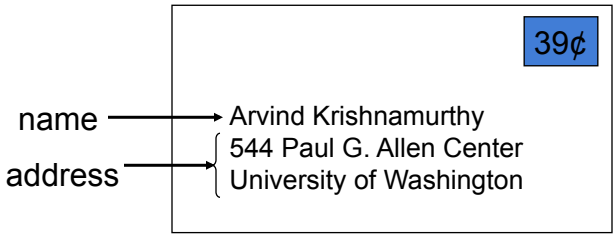
Naming and the DNS

Names and Addresses



- Names are identifiers for objects/services (high level)
- Addresses are locators for objects/services (low level)
- Binding is the process of associating a name with an address
- Resolution is the process of looking up an address given a name

Internet Hostnames

- Hostnames are human-readable identifiers for end-systems based on an administrative hierarchy
  - dogmatix.dyn.cs.washington.edu is my desktop machine
- IP addresses are a fixed-length binary encoding for end-systems based on their position in the network
  - 128.208.7.230 is uranium's IP address
- Original name resolution: HOSTS.TXT
- Current name resolution: Domain Name System
- Future name resolution: ?

Original Hostname System

- When the Internet was really young …
- Flat namespace
  - Simple (host, address) pairs
- Centralized management
  - Updates via a single master file called HOSTS.TXT
  - Manually coordinated by the Network Information Center
- Resolution process
  - Look up hostname in the HOSTS.TXT file

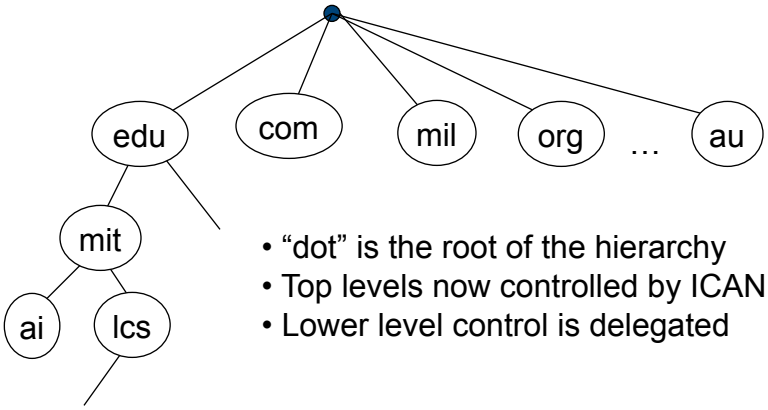
Scaling Problems

- Coordination
  - Between all users to avoid conflicts
- Inconsistencies
  - Between update and distribution of new version
- Reliability
  - Single point of failure
- Performance
  - Competition for centralized resources

Domain Name System (DNS)

- Designed by Mockapetris and Dunlap in the mid 80s
- Namespace is hierarchical
  - Allows much better scaling of data structures
  - e.g., dogmatix.dyn.cs.washington.edu
- Namespace is distributed
  - Decentralized administration and access
  - e.g., *.cs.washington.edu managed by CSE
- Resolution is by query/response
  - With replicated servers for redundancy
  - With heavy use of caching for performance

DNS Hierarchy



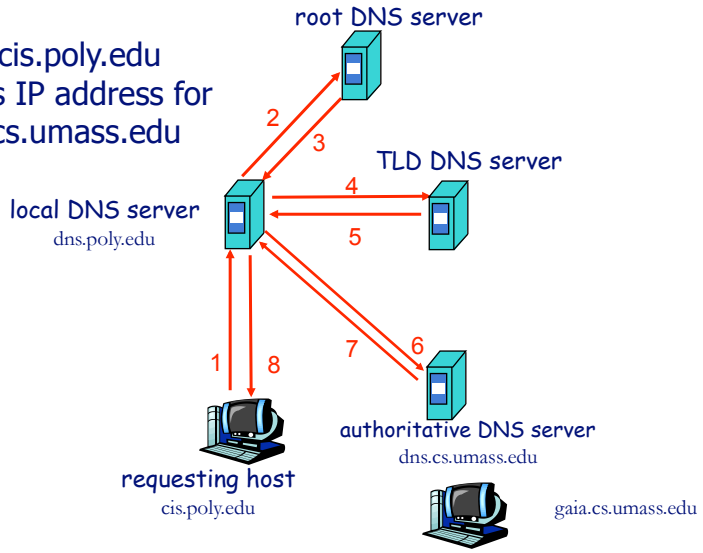
- "dot" is the root of the hierarchy
- Top levels now controlled by ICANN
- Lower level control is delegated

DNS Distribution

- Data managed by zones that contain resource records
  - Zone is a complete description of a portion of the namespace
  - e.g., all hosts and addresses for machines in washington.edu with pointers to subdomains like cs.washington.edu
- One or more nameservers manage each zone
  - Zone transfers performed between nameservers for consistency
  - Multiple nameservers provide redundancy
- Client resolvers query nameservers for specified records
  - Multiple messages may be exchanged per DNS lookup to navigate the name hierarchy

Example

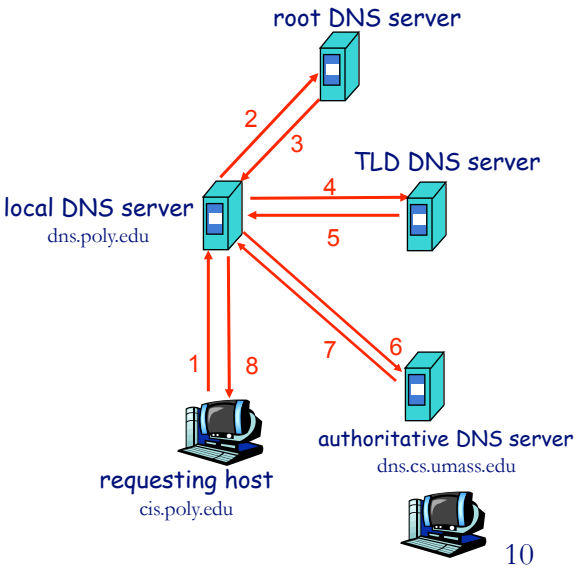
Host at cis.poly.edu wants IP address for gaia.cs.umass.edu



9

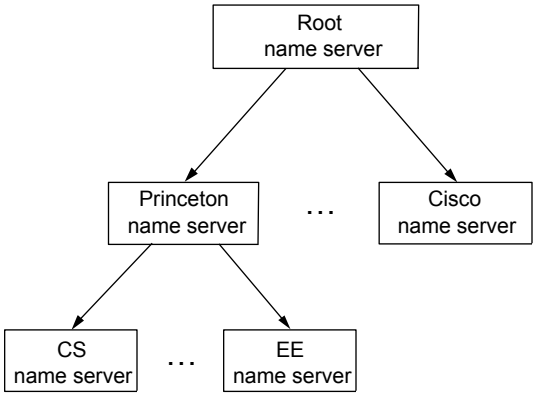
Recursive vs. Iterative Queries

- Recursive query
  - Ask server to get answer for you
  - E.g., request 1 and response 8
- Iterative query
  - Ask server who to ask next
  - E.g., all other request-response pairs



10

Hierarchy of Nameservers



DNS Bootstrapping

- Need to know IP addresses of root servers before we can make any queries
- Addresses for 13 root servers ([a-m].root-servers.net) handled via initial configuration (named.ca file)

12

DNS Caching

- Performing all these queries take time
  - And all this before the actual communication takes place
  - E.g., 1-second latency before starting Web download
- Caching can substantially reduce overhead
  - The top-level servers very rarely change
  - Popular sites (e.g., www.cnn.com) visited often
  - Local DNS server often has the information cached
- How DNS caching works
  - DNS servers cache responses to queries
  - Responses include a "time to live" (TTL) field
  - Server deletes the cached entry after TTL expires

13

Negative Caching

- Remember things that don't work
  - Misspellings like www.cnn.comm and www.cnnn.com
  - These can take a long time to fail the first time
  - Good to remember that they don't work
  - … so the failure takes less time the next time around

14

DNS Resource Records

DNS: distributed db storing resource records (RR)

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

- Type=A
  - name is hostname
  - value is IP address
- Type=NS
  - **name** is domain (e.g. foo.com)
  - **value** is hostname of authoritative name server for this domain
- Type=CNAME
  - name is alias name for some "canonical" (the real) name
    www.ibm.com is really
    servereast.backup2.ibm.com
  - value is canonical name
- Type=MX
  - value is name of mailserver associated with name

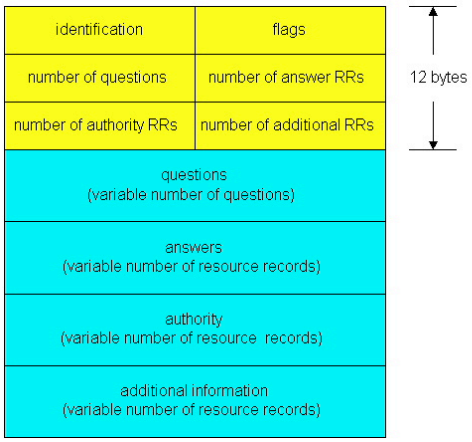
15

DNS Protocol

DNS protocol : *query* and *reply* messages, both with same *message format*

Message header

- Identification: 16 bit # for query, reply to query uses same #
- Flags:
  - Query or reply
  - Recursion desired
  - Recursion available
  - Reply is authoritative



16

Reliability

- DNS servers are replicated
  - Name service available if at least one replica is up
  - Queries can be load balanced between replicas
- UDP used for queries
  - Need reliability: must implement this on top of UDP
- Try alternate servers on timeout
  - Exponential backoff when retrying same server
- Same identifier for all queries
  - Don't care which server responds

17

Inserting Resource Records into DNS

- Example: just created startup "FooBar"
- Register foobar.com at Network Solutions
  - Provide registrar with names and IP addresses of your authoritative name server (primary and secondary)
  - Registrar inserts two RRs into the com TLD server:
    - (foobar.com, dns1.foobar.com, NS)
    - (dns1.foobar.com, 212.212.212.1, A)
- Put in authoritative server dns1.foobar.com
  - Type A record for www.foobar.com
  - Type MX record for foobar.com

18

Playing With Dig on UNIX

- Dig program
  - Allows querying of DNS system
  - Use flags to find name server (NS)
  - Disable recursion so that operates one step at a time

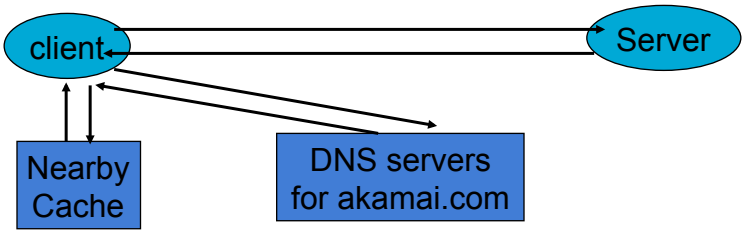
19

Future Evolution of the DNS

- Design constrains us in two major ways that are increasingly less appropriate
- Static host to IP mapping
  - What about mobility (Mobile IP)
- Location-insensitive queries
  - What if I don't care what server a Web page comes from, as long as it's the right page?
  - e.g., a yahoo page might be replicated

Akamai

- Use the DNS to effect selection of a nearby Web cache



- Leverage separation of static/dynamic content

DNS DoS Attacks

October 22, 2002

- The attack lasted for approximately one hour. Of the thirteen servers, nine were disabled
- The largest malfunction of the DNS servers before this event were seven machines in July 1997, due to a technical glitch

DNS DoS Attacks

February 6, 2007

- The attack lasted about five hours. none of the servers crashed, two of the root servers "suffered badly", while others saw "heavy traffic".
- The botnet responsible for the attack has reportedly been traced to South Korea.

"If the United States found itself under a major cyberattack aimed at undermining the nation's critical information infrastructure, the Department of Defense is prepared, based on the authority of the president, to launch a cyber counterattack or an actual bombing of an attack source."