CSE561 – Naming and DNS

David Wetherall
djw@cs.washington.edu
SST Discussion

• What are the contributions of the paper?
• How does SST separate reliability & congestion control?
  - What is numbered, acknowledged; compare to TCP
• What does a developer do to implement HTTP/1.1?
• Could we use SST for DNS?
• What’s the best way to evaluate SST?
Naming and DNS

- Focus:
  - How do we resolve names to addresses

- Names and addresses
- DNS as a system design
Names and Addresses

- Names are identifiers for objects/services (high level)
- Addresses are locators for objects/services (low level)
- Resolution is the process of mapping name to address
- But, addresses are really lower-level names; many levels used
Naming in Systems

- Ubiquitous
  - Files in filesystem, processes in OS, pages on the web, …

- Decouple identifier for object/service from location
  - Hostnames provide a level of indirection for IP addresses

- Key issue is the resolution system
  - Likely to constrain names or addresses to function
  - DNS names are hierarchical, IP addresses constrained by location
Example: 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 (NIC)

• 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
Today: 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., galah.cs.washington.edu

- Namespace is distributed
  - Decentralized administration and access
  - e.g., galah managed by CSE

- Resolution is by query/response
  - With replicated servers for redundancy
  - With heavy use of caching for performance
DNS Lookups / Resolution

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

1. Requesting host cis.poly.edu
2. Local DNS server dns.poly.edu
3. Root DNS server
4. Edu DNS server
5. Local DNS server dns.poly.edu
6. Authoritative DNS server dns.cs.umass.edu
7. Requesting host cis.poly.edu
8. Requesting host cis.poly.edu
Design requirements

• Work well at large scale

• Provide highly available service

• Rapid name resolution

• Serve many organizations
Design Issues

- **Scaling up**
  - Use hierarchy and replication to spread work over servers
  - Use caching (TTL on replies) to cut down on work

- **Reliability**
  - Replicated servers

- **Performance**
  - Caching resolutions
  - Request/reply over UDP, not TCP
  - Replicated servers

- **Administration**
  - Use hierarchy to carve up namespace; but TLDs are contentious
  - Use local nameservers to relieve client of responsibility
DNS futures

- DNS works great to map hostname to IP!

- What has changed:
  - A static mapping is no longer what many applications want
  - e.g., return “an IP with the content I want”
  - e.g., return “the nearest IP with the content I want”

- This is tied up with CDNs …