COMP/ELEC 429/556
Introduction to Computer Networks

Domain Name System

Some slides used with permissions from Edward W. Knightly, T. S. Eugene Ng, Ion Stoica, Hui Zhang
Data Packet

- Fundamental unit for communications in Internet
An IP address

2150268945
An IP address

128.42.128.17

10000000 00101010 10000000 00010001
A Domain Name

www.cs.rice.edu
Motivation

- Fact: A fundamental feature of the Internet is that every network interface is identified by a numerical IP address
- An application needs to know the IP address of the communication peer
- There is no magic, some out-of-band mechanism is needed
  - Word of mouth
  - Read it in magazine advertisements
  - Search engine
  - Etc.
- But IP addresses are bad for humans to remember and tell each other, need names that make sense to humans
Internet Names & Addresses

• Names: e.g. www.rice.edu
  – human-usable labels for machines
  – variable length, long

• Addresses: e.g. 128.42.204.11
  – 32-bit number, written this way for convenience
  – machine-usuable fixed length, short labels for machines
  – more efficient to process than names

• How do you lookup from one to another?

• Let’s try nslookup!

• What system is working behind the scene? What can you infer from the format of domain names?
Domain Name System is a case study of the importance of scalability

Who’s this guy?

Jon Postel

From Wikipedia, the free encyclopedia

Jonathan Bruce Postel (/pəˈstɛl/; August 6, 1943 – October 16, 1998) was an American computer scientist who made many significant contributions to the development of the Internet, particularly with respect to standards. He is known principally for
Domain names used to be arbitrary and stored in one shared file hosts.txt
History

• Initially all host-address mappings were in a file called hosts.txt (in /etc/hosts)
  – Changes were submitted to SRI by email
  – New versions of hosts.txt ftp’d periodically from SRI
  – An administrator could pick names at their discretion
  – Any name is allowed: eugenedesktopatrice

• As the Internet grew this system broke down because:
  – SRI couldn’t handled the load
  – Hard to enforce uniqueness of names
  – Many hosts had inaccurate copies of hosts.txt

• How to build a lookup system that scales!!!
  – billions of names and addresses to insert/delete/modify
  – billions of lookups per second

• Strategy: Divide and Conquer!
Basic DNS Features

• Hierarchical namespace
  – as opposed to original flat namespace

• Distributed storage architecture
  – as opposed to centralized storage
Naming Hierarchy

- "Top Level Domains" are at the top
- Depth of tree is almost arbitrary (limit 128)
- Domains are subtrees
  - E.g.: .edu, rice.edu, ece.rice.edu
- Name collisions avoided
  - E.g. rice.edu and rice.com can coexist, but uniqueness is job of domain
Host names are administered hierarchically

A **zone** corresponds to an administrative authority that is responsible for that portion of the hierarchy

E.g. Eugene controls names: x.cs.rice.edu and y.ece.rice.edu

E.g. The President controls names: x.rice.edu and y.natsci.rice.edu
DNS Server Hierarchy

- Each server has authority over a portion of the hierarchy called zone
  - That zone can be empty, i.e. server is non-authoritative
- Each server contains all the records for the hosts or domains in its zone
- “Root server” knows about all top-level domains
  - .com .edu .gov .org etc.
DNS Root Servers
(in the early days)

- About a dozen root server IP addresses
- Contacted by other servers that cannot resolve name
Basic Domain Name Resolution

• Every host knows a local DNS server
  – Through DHCP, for example
  – Sends all queries to a local DNS server
  – Local DNS server may be non-authoritative

• Every local DNS server knows the ROOT servers
  – When no locally cached information exists about the query, talk to a root server, and go down the name hierarchy from the root
Example of Iterated DNS Query

Root name server:
- May not know authoritative name server
- May know **intermediate name server**: who to contact to find authoritative name server

Iterated query:
- Contacted server replies with name/address of another server
- “I don’t know this name, but ask this server”

Diagram:
- Root name server
- Iterated query: Contacted server replies with name/address of another server
- Intermediate name server
- Authoritative name server
- Local name server
- Computer wants to lookup "www.google.com"
DNS Query & Resource Record

• DNS Query:
  – Two fields: (name, type)

• Resource record is the response to a query
  – Four fields: (name, value, type, TTL)
  – TTL (time to live) is the duration the response can be cached
  – There can be multiple valid responses to a query

• Type = A
  – name = hostname
  – value = IPv4 address
Other Common Record Types

• Type = AAAA
  – name = domain
  – value = IPv6 address
  – 4 A’s because IPv6 addresses have 4 times more bits than IPv4, get it? ;-)  
• Type = NS
  – name = domain
  – value = name of dns server for domain
• Type = CNAME
  – name = hostname
  – value = canonical name
• Type = MX
  – name = domain in email address
  – value = canonical name of mail server and priority
Is DNS good enough?
Caching

- Local name server caches answers (e.g. in steps 3, 5, 7) so popular queries can be resolved more quickly in the future.
- Greatly reduces latency.
- Masks failures in the server hierarchy temporarily.

Local name server caches answers (e.g. in steps 3, 5, 7) so popular queries can be resolved more quickly in the future.
- Greatly reduces latency.
- Masks failures in the server hierarchy temporarily.
Anycast

- A root server IP address is “shared” by many servers at very different locations in the world
  - Request to the IP address routed to “nearest” server
  - How that is possible is a subject we’ll visit later
- Reduces latency, increase throughput, and masks failures
Load Balancer

• One root server IP address points to a large cluster of servers
• Increases throughput
• e.g. Amazon AWS provides an auto-scaling service for web hosting
Discussions

• DNS caching
  – Each record has a TTL (time-to-live, max duration caching is allowed)
  – Crucial to scalability and robustness
  – Improve performance by saving results of previous lookups
  – E.g. results of address records and name server records (e.g. if rice.edu name server is cached, then can bypass root server the second time looking for a rice.edu host)

• DNS “hacks”
  – Return records based on requester’s IP address
  – Round-robin over a list of IP addresses mapped to the same name for load balancing
  – Return address of least loaded machine
  – Basis for many web content distribution networks such as Akamai and Limelight
How to Programmatically Use the DNS System?

Operating system comes to the rescue

- `getaddrinfo()
- `getnameinfo()

- On CLEAR, you can type “man getaddrinfo” to get a detailed description of the function
  - “man” stands for manual