Human Involvement

• Just like your friend needs to tell you his phone number for you to call him
• Somehow, 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 the advertisement in the paper
  – Etc.
• But IP addresses are bad for humans to remember and tell each other
• So need names that makes some sense to humans
Internet Names & Addresses

• Names: e.g. www.northeastern.edu
  – human-useable labels for machines
  – conforms to “organizational” structure

• Addresses: e.g. 155.33.17.68
  – computer-useable labels for machines
  – conforms to “network” structure

• How do you map from one to another?
  – Domain Name System (DNS)

DNS: 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: alansdesktopatccsbuilding

• 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

• Domain Name System (DNS) was born
Basic DNS Features

- Hierarchical namespace
  - as opposed to original flat namespace

- Distributed storage architecture
  - as opposed to centralized storage (plus replication)

- Client--server interaction on UDP Port 53
  - but can use TCP if desired

Naming Hierarchy

- "Top Level Domains" are at the top
- Depth of tree is arbitrary (limit 128)
- Domains are subtrees
  - E.g.: .edu, neu.edu, ccs.neu.edu
- Name collisions avoided
  - E.g. neu.edu and neu.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. Alan controls names: x.ccs.neu.edu and y.ece.neu.edu

E.g. The President controls names: x.neu.edu and y.husky.neu.edu

Server Hierarchy

- Each server has authority over a portion of the hierarchy
  - A server maintains only a subset of all names

- Each server contains all the records for the hosts or domains in its zone
  - might be replicated for robustness

- Every server knows the root

- Root server knows about all top-level domains
DNS Name Servers

- Local name servers:
  - Each ISP (company) has local default name server
  - Host DNS query first goes to local name server
  - Local DNS server IP address usually learned from DHCP
  - Frequently cache query results

- Authoritative name servers:
  - For a host: stores that host's (name, IP address)
  - Can perform name/address translation for that host's name

DNS: Root Name Servers

- Contacted by local name server that can not resolve name
- Root name server:
  - Contacts authoritative name serv if name mapping not known
  - Gets mapping
  - Returns mapping to local name server
  - ~ Dozen root name servers worldwide
Basic Domain Name Resolution

- Every host knows a local DNS server
  - Through DHCP, for example
  - Sends all queries to a local DNS server
- 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
  - If we lookup www.neu.edu, and we have a cached entry for the .edu name server, then we can go directly to the .edu name server and bypass the root server

Example of Iterated DNS Query

Iterated query:
- Contacted server replies with name of server to contact
- “I don't know this name, but ask this server”

This is how today's DNS system behaves
DNS Resource Records

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

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

• Type = A:
  – name = hostname
  – value = IP address

DNS Resource Records (cont’d)

• 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