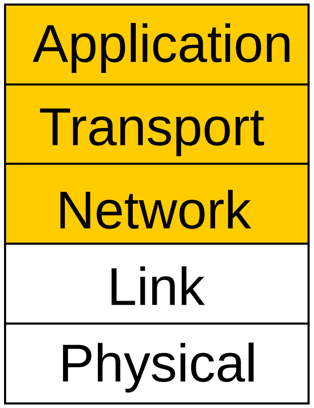
CSE561 – Naming and DNS

David Wetherall
djw@cs.washington.edu

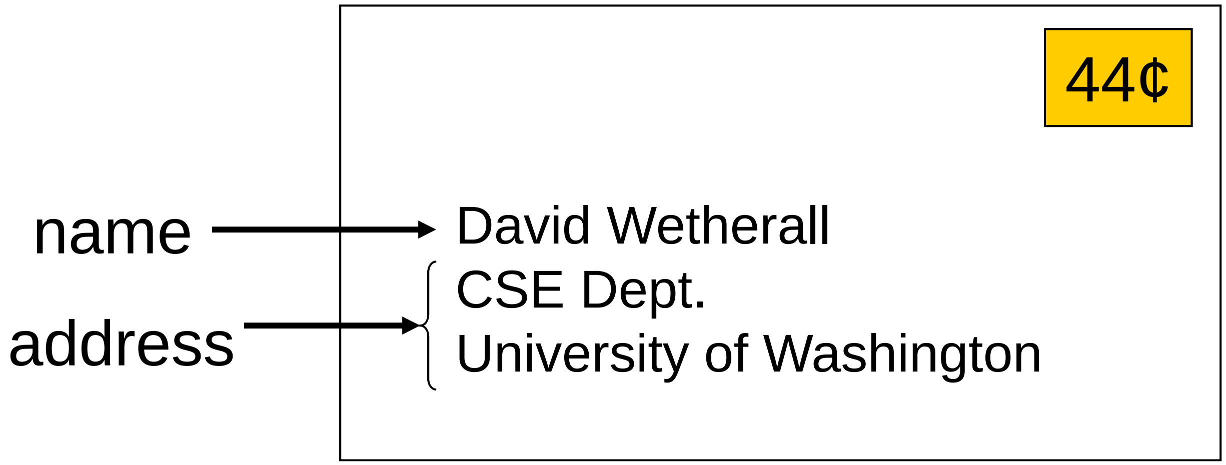
Naming and DNS

- 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

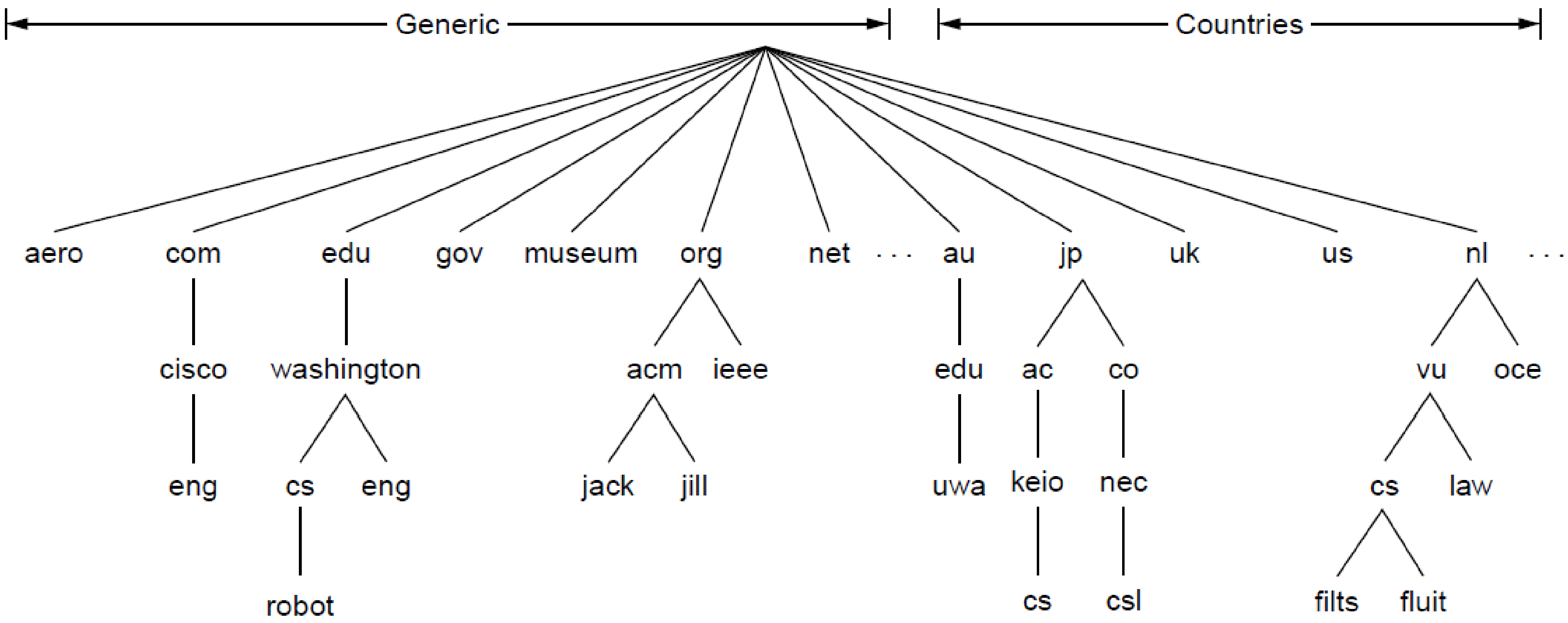
Scaling Problems

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

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 Hierarchy

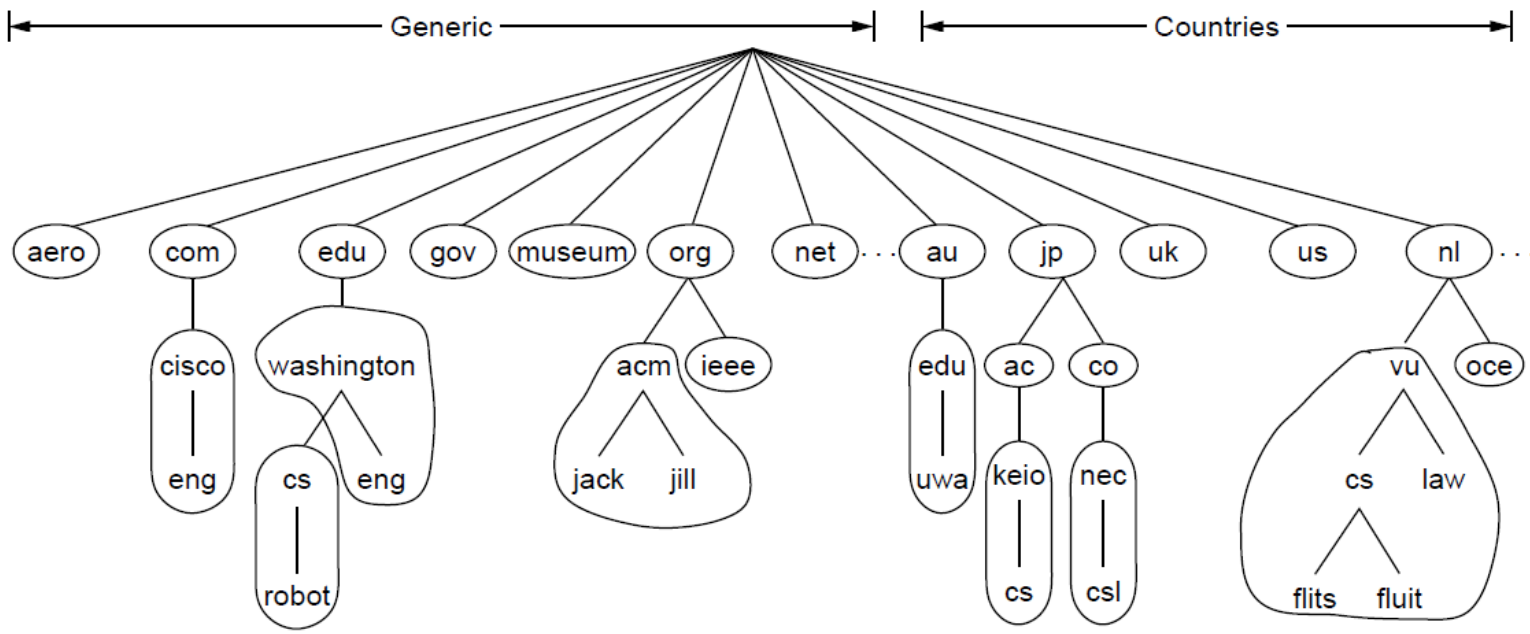


- “dot” is the root, top levels now controlled by ICANN
- Usage governed by conventions

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 (coming soon)

DNS Zones



- Namespace divided into zones, each of which is maintained by a nameserver

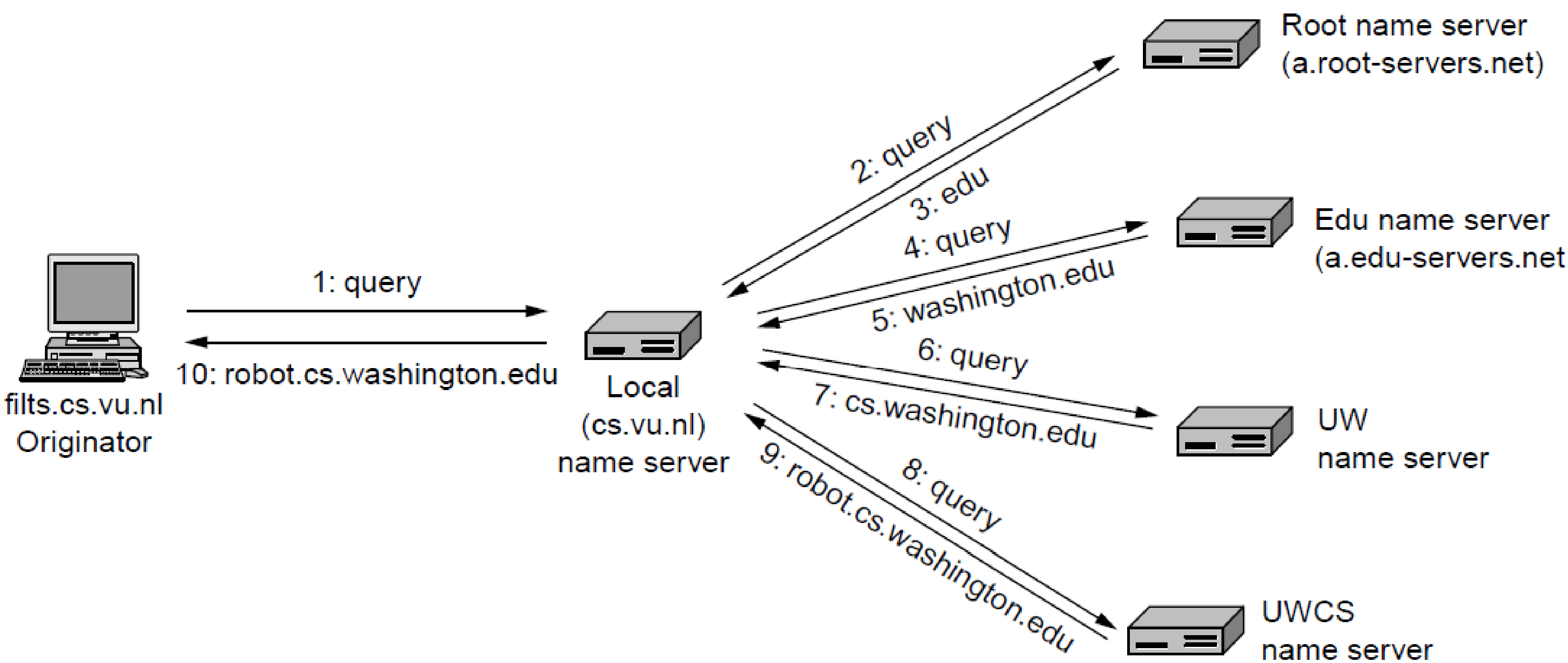
DNS Resource Records

- Human readable description of a zone database
- DNS queries return selected resource records

; Authoritative data for cs.vu.nl

cs.vu.nl.	86400	IN	SOA	star boss (9527,7200,7200,241920,86400)
cs.vu.nl.	86400	IN	MX	1 zephyr
cs.vu.nl.	86400	IN	MX	2 top
cs.vu.nl.	86400	IN	NS	star
star	86400	IN	A	130.37.56.205
zephyr	86400	IN	A	130.37.20.10
top	86400	IN	A	130.37.20.11
www	86400	IN	CNAME	star.cs.vu.nl
ftp	86400	IN	CNAME	zephyr.cs.vu.nl
flits	86400	IN	A	130.37.16.112
flits	86400	IN	A	192.31.231.165
flits	86400	IN	MX	1 flits
flits	86400	IN	MX	2 zephyr
flits	86400	IN	MX	3 top
rowboat		IN	A	130.37.56.201
		IN	MX	1 rowboat
		IN	MX	2 zephyr
little-sister		IN	A	130.37.62.23
laserjet		IN	A	192.31.231.216

DNS Lookup/Resolution Example



Recursive vs. Iterative Queries

- Recursive query
  - Ask server to get answer for you
  - E.g., request 1 and response 10
- Iterative query
  - Ask server who to ask next
  - E.g., all other request-response pairs

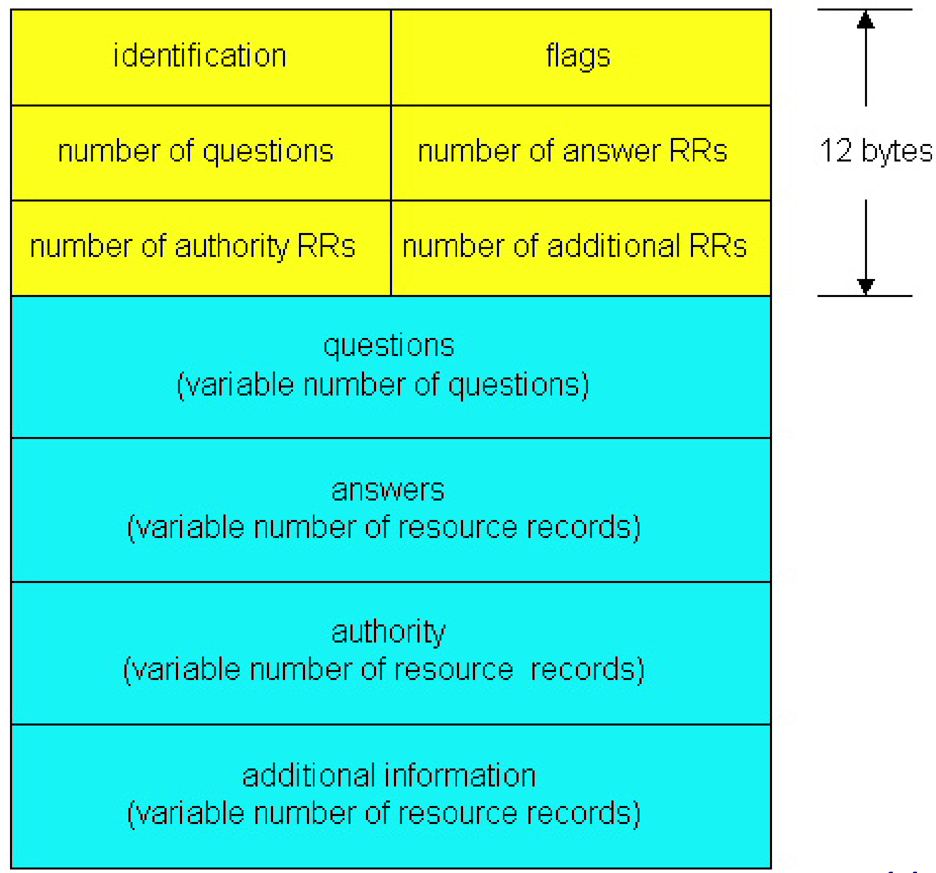
- When would you want recursive vs. iterative?

DNS Messages

Query /*Reply* messages have the 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



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)

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

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

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

Building on the DNS

- Other naming designs leverage the DNS

- Email:
  - e.g., djw@cs.washington.edu is djw in the domain cs.washington.edu

- Uniform Resource Locators (URLs) name for Web pages
  - e.g., www.cs.washington.edu/homes/djw
  - Use domain name to identify a Web server
  - Use "/" separated string to name path to page (like files)

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 …