AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

21CSE211
COMPUTER NETWORKS

APPLICATION LAYER

- **Learn about Application Layer protocols**

  DNS (Domain Name System)

Domain Name System (DNS)

One identifier for a host is its **hostname.** Hostnames—such as `cnn.com`, `www.yahoo.com`, `gaia.cs.umass.edu`, and `cis.poly.edu`. Because hostnames can consist of variable length alphanumeric characters, they would be difficult to process by routers. For these reasons, hosts are also identified by so-called **IP addresses.**

- An IP address consists of four bytes and has a rigid hierarchical structure.
- An IP address looks like 121.7.106.83, where each period separates one of the bytes expressed in decimal notation from 0 to 255.
- An IP address is hierarchical because as we scan the address from left to right, we obtain more and more specific information about where the host is located in the Internet

People prefer the more mnemonic hostname identifier, while routers prefer fixed-length, hierarchically structured IP addresses

The Domain Name System (DNS) is the phonebook of the Internet. When users type domain names such as 'google.com' or 'nytimes.com' into web browsers, DNS is responsible for finding the correct IP address for those sites.

What happens when a URL is requested ?

- Consider what happens when a browser (that is, an HTTP client), running on some user's host, requests the URL www.someschool.edu/index.html. In order for the user's host to be able to send an HTTP request message to the Web server www.someschool.edu, the user's host must first obtain the IP address of www.someschool.edu. This is done as follows.

1. The same user machine runs the client side of the DNS application.
2. The browser extracts the hostname, www.someschool.edu, from the URL and passes the hostname to the client side of the DNS application.
3. The DNS client sends a query containing the hostname to a DNS server.
4. The DNS client eventually receives a reply, which includes the IP address for the hostname.
5. Once the browser receives the IP address from DNS, it can initiate a TCP connection to the HTTP server process located at port 80 at that IP address.

The desired IP address is often cached in a "nearby" DNS server, which helps to reduce DNS network traffic as well as the average DNS delay

DNS: services, structure

DNS services

- hostname to IP address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance

A: *doesn't scale!*

DNS Services: -

- DNS provides following important services
    - **Hostname to IP address translation, Host Aliasing, Mail server aliasing, Load distribution**

Host aliasing. A host with a complicated hostname can have one or more alias names. For example, a hostname such as relay1.west-coast.enterprise.com could have, say, two aliases such as enterprise.com and www.enterprise.com. In this case, the hostname relay1.westcoast.enterprise.com is said to be a **canonical hostname**

DNS can be invoked by an application to obtain the canonical hostname for a supplied alias hostname as well as the IP address of the host

Mail Server Aliasing : It is highly desirable that e-mail addresses be mnemonic. However, the hostname of the Hotmail mail server is more complicated and much less mnemonic than simply hotmail.com (for example, the canonical hostname might be something like relay1.west-coast.hotmail.com). DNS can be invoked by a mail application to obtain the canonical hostname for a supplied alias hostname as well as the IP address of the host.

MX record permits a company's mail server and Web server to have identical (aliased) hostnames; for example, a company's Web server and mail server can both be called enterprise.com.

DNS Services:-

Load distribution:

- DNS is also used to perform load distribution among replicated servers, such as replicated Web servers
- For replicated Web servers, a *set* of IP addresses is thus associated with one canonical hostname. The DNS database contains this set of IP addresses.
- When clients make a DNS query for a name mapped to a set of addresses, the server responds with the entire set of IP addresses, but rotates the ordering of the addresses within each reply. Because a client typically sends its HTTP request message to the IP address that is listed first in the set, DNS rotation distributes the traffic among the replicated servers
- DNS rotation is also used for e-mail so that multiple mail servers can have the same alias name.

The DNS is specified in RFC 1034 and RFC 1035, and updated in several additional RFCs

Why not centralize DNS?

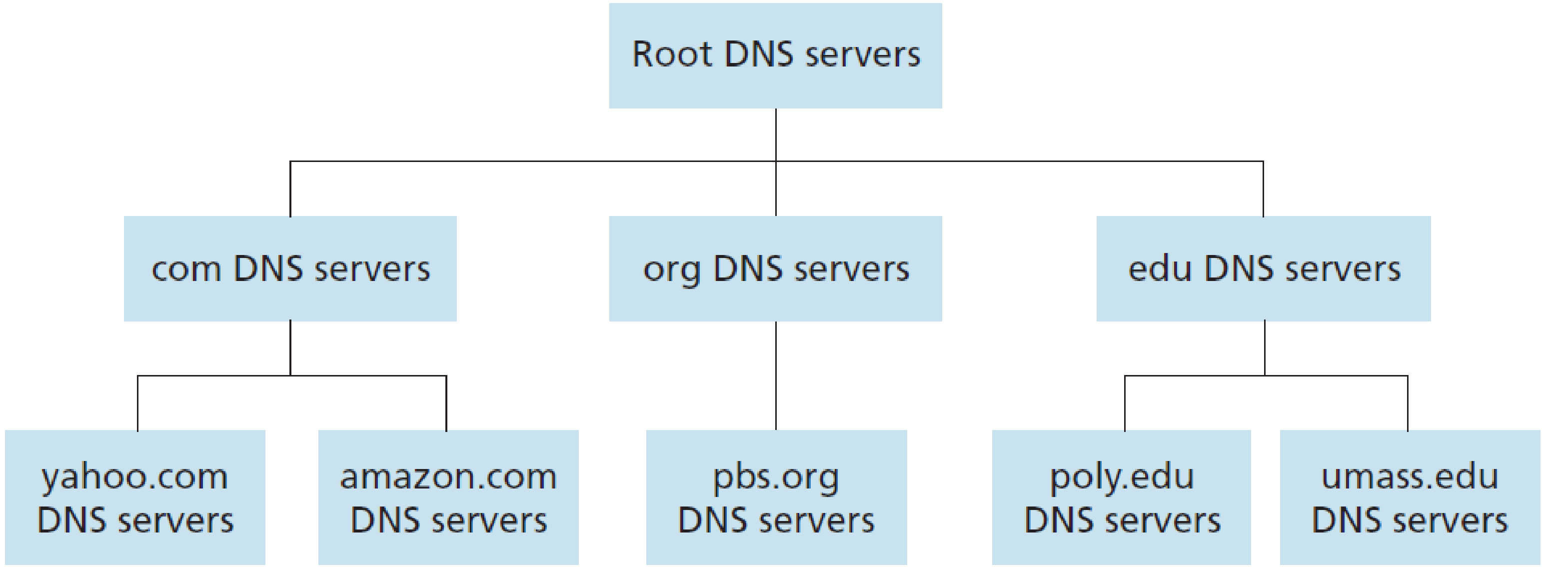
- **A single point of failure.** If the DNS server crashes, so does the entire Internet!
- **Traffic volume.** A single DNS server would have to handle all DNS queries (for all the HTTP requests and e-mail messages generated from hundreds of millions of hosts).
- **Distant centralized database.** A single DNS server cannot be “close to” all the querying clients. If we put the single DNS server in New York City, then all queries from Australia must travel to the other side of the globe, perhaps over slow and congested links. This can lead to significant delays.
- **Maintenance.** The single DNS server would have to keep records for all Internet hosts. Not only would this centralized database be huge, but it would have to be updated frequently to account for every new host.

A centralized database in a single DNS server simply *doesn’t scale*

DNS: a distributed, hierarchical database

client wants IP for www.amazon.com; 1st approximation:

- client queries root server to find com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com



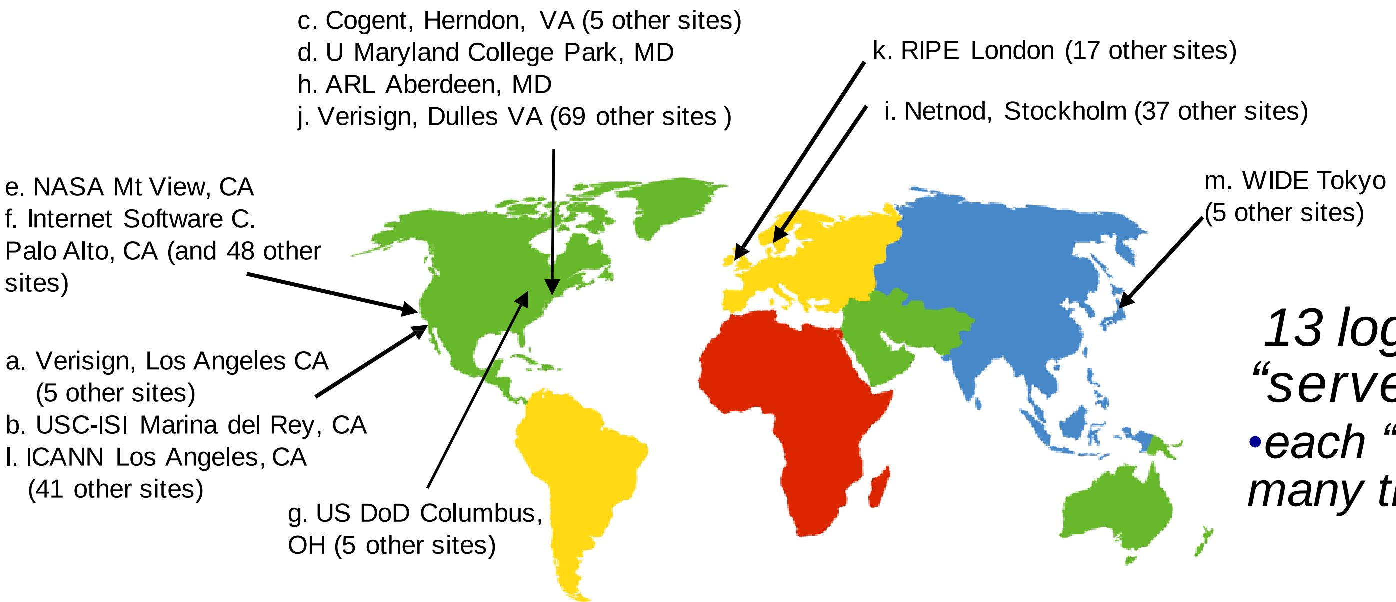
- In order to deal with the issue of scale, the DNS uses a large number of servers, organized in a hierarchical fashion and distributed around the world.
- No single DNS server has all of the mappings for all of the hosts in the Internet. Instead, the mappings are distributed across the DNS servers

there are three classes of DNS servers—Root DNS servers, top-level domain (TLD) DNS servers, and authoritative DNS server

DNS: root name servers

- contacted by local name server that can not resolve name
- root name server:
  - contacts authoritative name server if name mapping not known
  - gets mapping
  - returns mapping to local name server

Although we have referred to each of the 13 root DNS servers as if it were a single server, each "server" is actually a network of replicated servers, for both security and reliability purposes.
All together, there are 247 root servers as of fall 2011.



13 logical root name "servers" worldwide

- *each "server" replicated many times*

TLD, authoritative servers

top-level domain (TLD) servers:

- responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp
- Network Solutions maintains servers for .com TLD
- Educause for .edu TLD

authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

- Suppose a DNS client wants to determine the IP address for the hostname www.amazon.com.
- To a first approximation, the following events will take place.
  - The client first contacts one of the root servers, which returns IP addresses for TLD servers for the top-level domain com.
  - The client then contacts one of these TLD servers, which returns the IP address of an authoritative server for amazon.com.
  - Finally, the client contacts one of the authoritative servers for amazon.com, which returns the IP address for the hostname www.amazon.com

Local DNS name server

- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one
  - also called "default name server"
- When a host connects to an ISP, the ISP provides the host with the IP addresses of one or more of its local DNS servers
- when host makes DNS query, query is sent to its local DNS server
  - has local cache of recent name-to-address translation pairs (but may be out of date!)
  - acts as proxy, forwards query into hierarchy

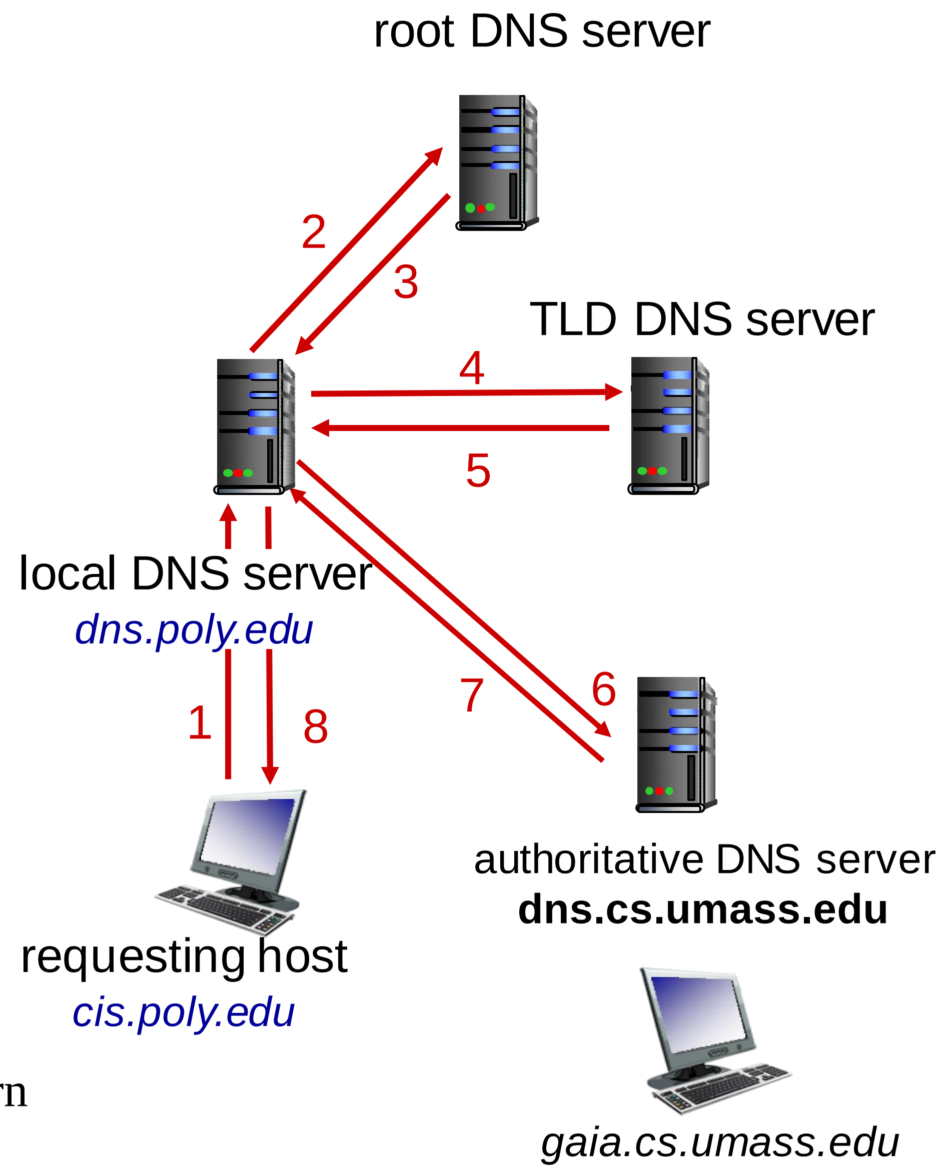
DNS name resolution example

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

iterated query:

- contacted server replies with name of server to contact
- "don't know this name, but ask this server"

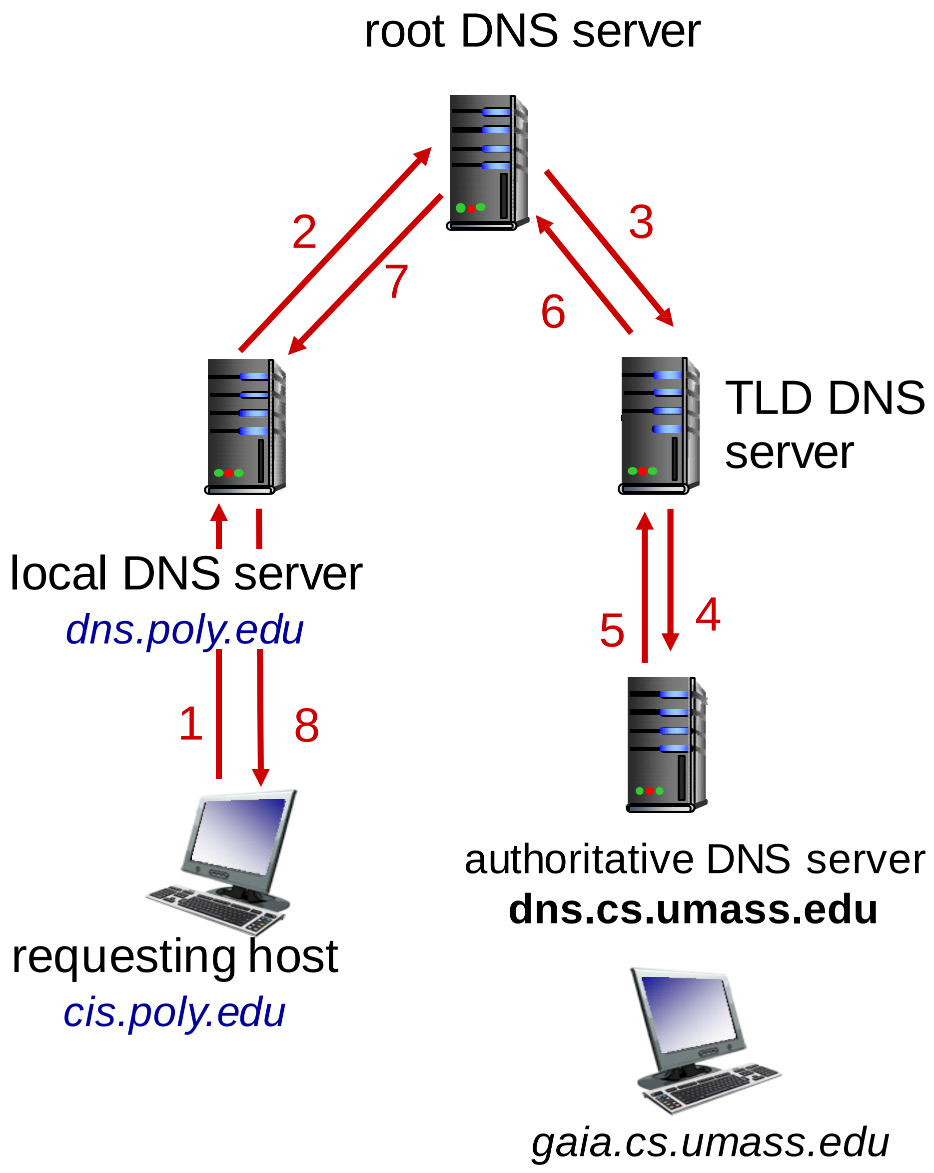
TLD server may not know the authoritative DNS server for the hostname. Instead, the TLD server may know only of an intermediate DNS server, which in turn knows the authoritative DNS server for the hostname.



DNS name resolution example

recursive query:

- puts burden of name resolution on contacted name server
- heavy load at upper levels of hierarchy?



DNS: caching, updating records

- once (any) name server learns mapping, it *caches* mapping
  - cache entries timeout (disappear) after some time (TTL- Time To Live)
  - TLD servers typically cached in local name servers
    - thus root name servers not often visited
- cached entries may be *out-of-date* (best effort name-to-address translation!)
  - if name host changes IP address, may not be known Internet-wide until all TTLs expire
- update/notify mechanisms proposed IETF standard
  - RFC 2136

A local DNS server can also cache the IP addresses of TLD servers, thereby allowing the local DNS server to bypass the root DNS servers in a query chain

In a query chain, when a DNS server receives a DNS reply (containing, for example, a mapping from a hostname to an IP address), it can cache the mapping in its local memory

Because hosts and mappings between hostnames and IP addresses are by no means permanent, DNS servers discard cached information after a period of time (often set to two days).

DNS records

If a server is not authoritative for a hostname, then the server will contain a Type NS record for the domain that includes the hostname; it will also contain a Type A record that provides the IP address of the DNS server in the `Value` field of the NS record.

DNS:

distributed database 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

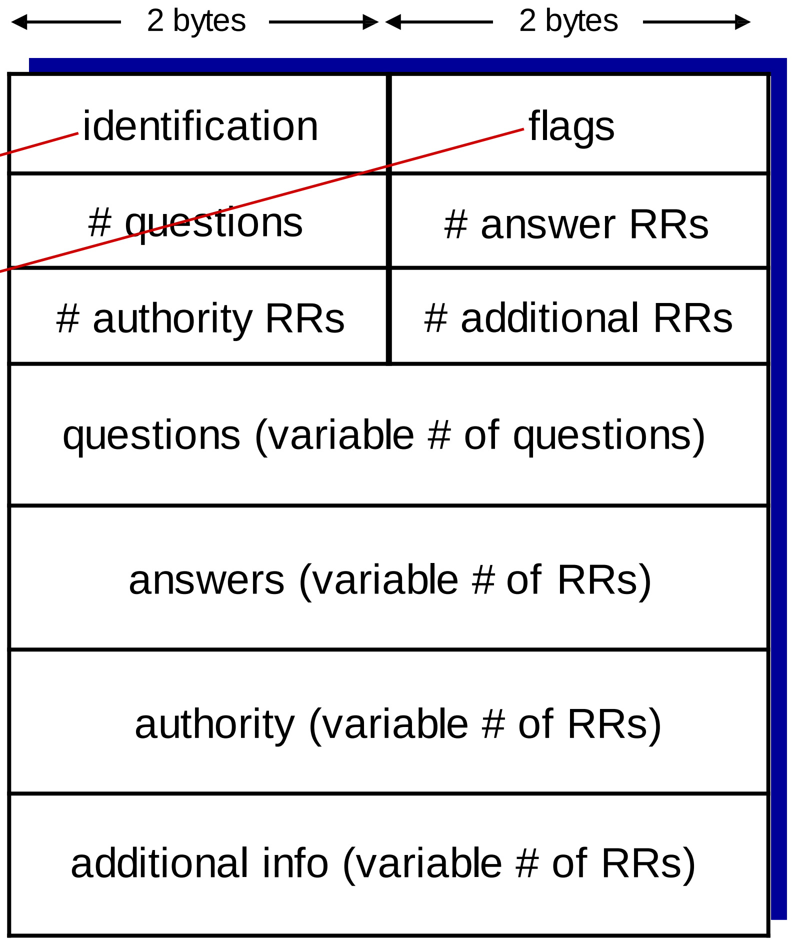
- **`value`** is name of mailserver associated with **`name`**

To obtain the canonical name for the mail server, a DNS client would query for an MX record; to obtain the canonical name for the other server, the DNS client would query for the CNAME record

If a DNS server is authoritative for a particular hostname, then the DNS server will contain a Type A record for the hostname

DNS protocol, messages

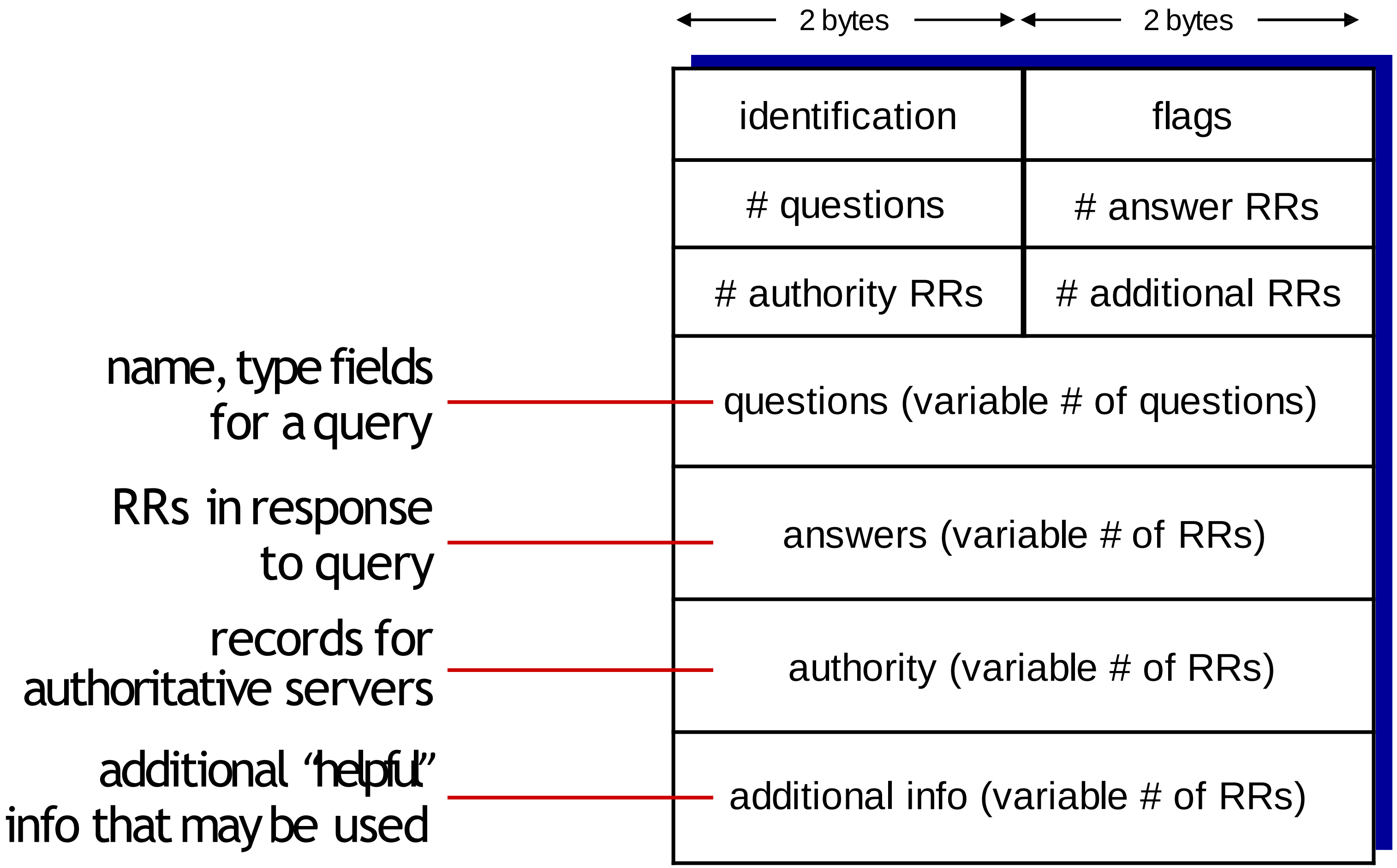
- *query* and *reply* messages, both with same *message format*



message header

- identification: 16 bit # for query, reply to query uses same #
- flags:
  - query or reply
  - recursion desired
  - recursion available
  - reply is authoritative

DNS protocol, messages



Inserting records into DNS

- example: new startup "Network Utopia"
- register name networkuptopia.com at *DNS registrar* (e.g., Network Solutions)
  - provide names, IP addresses of authoritative name server (primary and secondary)
  - registrar inserts two RRs into .com TLD server:

    `(networkutopia.com, dns1.networkutopia.com, NS)`

    `(dns1.networkutopia.com, 212.212.212.1, A)`
- create authoritative server type A record for www.networkuptopia.com; type MX record for networkutopia.com

Visit //www.internic.net.

Namah Shivaya