Information Network

Lecture 5 : DNS

Holger Thies

KYOTO UNIVERSITY

京都大学

Today's lecture

2.2 Web and HTTP

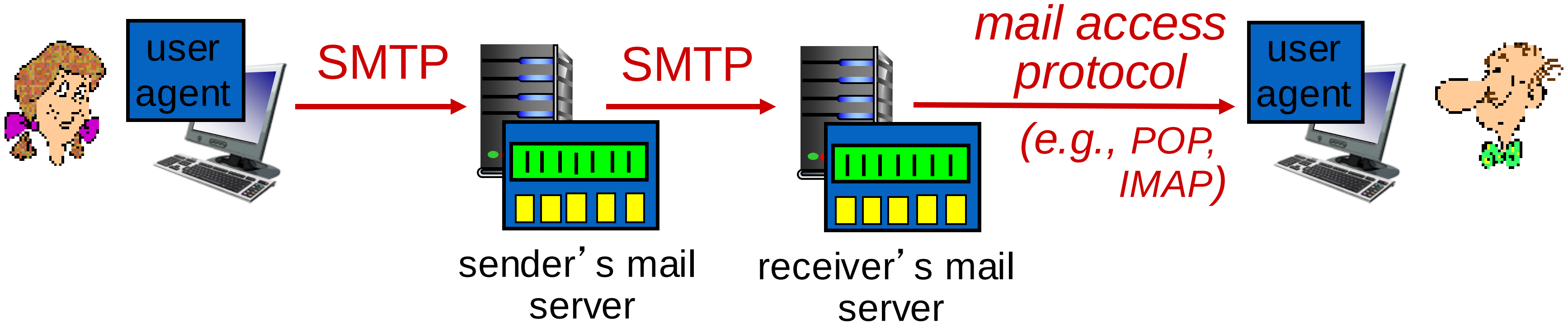
2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

京都大学

Mail access protocols



- SMTP: delivery/storage to receiver's server
- mail access protocol: retrieval from server
  - POP: Post Office Protocol [RFC 1939]: authorization, download
  - IMAP: Internet Mail Access Protocol [RFC 1730]: more features, including manipulation of stored messages on server
  - HTTP: gmail, Hotmail, Yahoo! Mail, etc.

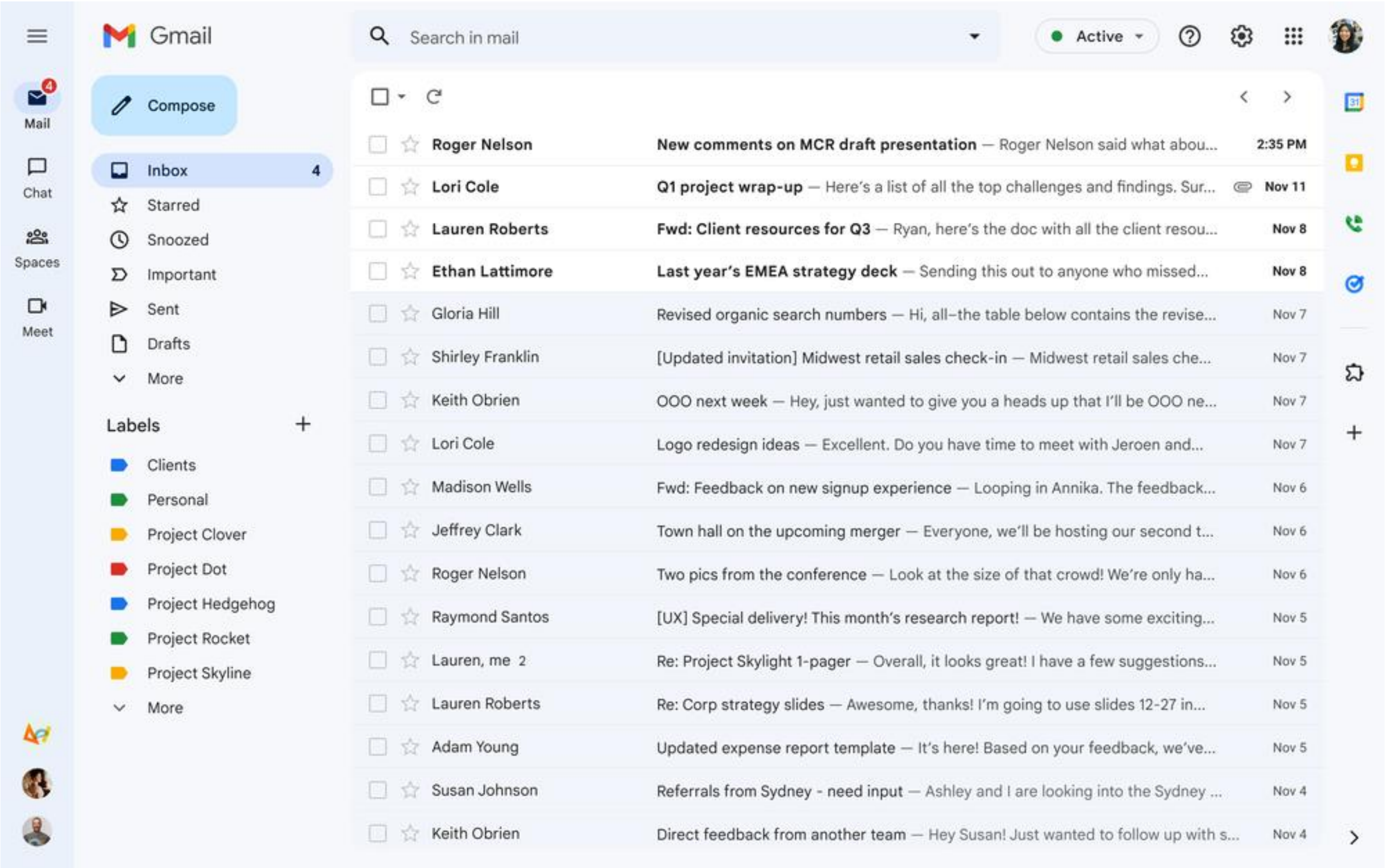
京都大学

IMAP vs POP3

Feature	IMAP (Internet Message Access Protocol)	POP3 (Post Office Protocol 3)
Message Storage	Server-based; emails are stored on the email server.	Client-based; emails are downloaded to the local device.
Multiple Devices	Supports synchronization across multiple devices.	Limited support for accessing emails from multiple devices.
Message Management	Allows folder creation, organization, and server-based operations (e.g., marking emails as read/unread).	Limited folder support; typically only Inbox.
Email Download	Downloads email headers first and provides options to download full messages.	Downloads entire messages but doesn't sync with the server.
Storage Space	Emails are stored on the server, so storage space is managed by the email provider.	Local device storage must be managed by the user.
Use Cases	Ideal for users who need access to email from multiple devices and want server-based email management.	Suitable for users who want to download emails to a single device and prefer local email management.

京都大学

Web-Based E-Mail



Web-Based E-Mail

- User agent is an ordinary Web browser
  - User communicates with server via HTTP
  - E.g., Gmail, Yahoo mail, Hotmail
- Reading e-mail
  - Web pages display the contents of folders
  - … and allow users to download and view messages
  - "GET" request to retrieve the various Web pages
- Sending e-mail
  - User types the text into a form and submits to the server
  - "POST" request to upload data to the server
  - Server uses SMTP to deliver message to other servers

京都大学

Email message format

- There is also a standard [RFC2822] on how the email message itself should be formatted.
  - The message does not only contain the body text but also additional header lines.
  - Similar to how HTML defines the language on how web documents are written.
- Header lines provide essential information
- Common header lines include:

  From: Sender's email address.

  To: Recipient's email address.

  Subject: The topic of the email.

  Date: The timestamp when the email was sent.

京都大学

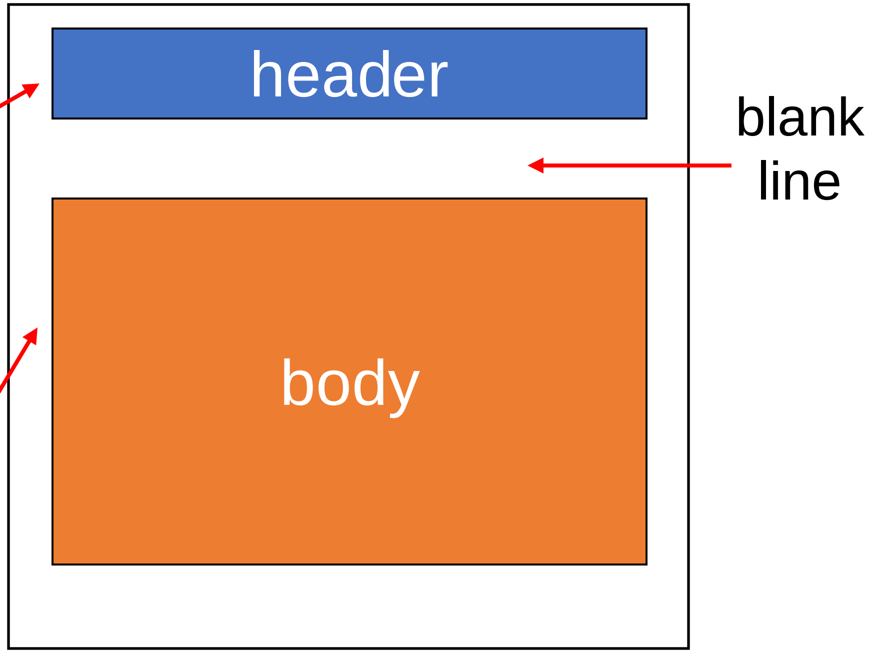
Email message format

SMTP: protocol for exchanging email messages

RFC 822: standard for text message format:

- header lines, e.g.,
  - To:
  - From:
  - Subject:

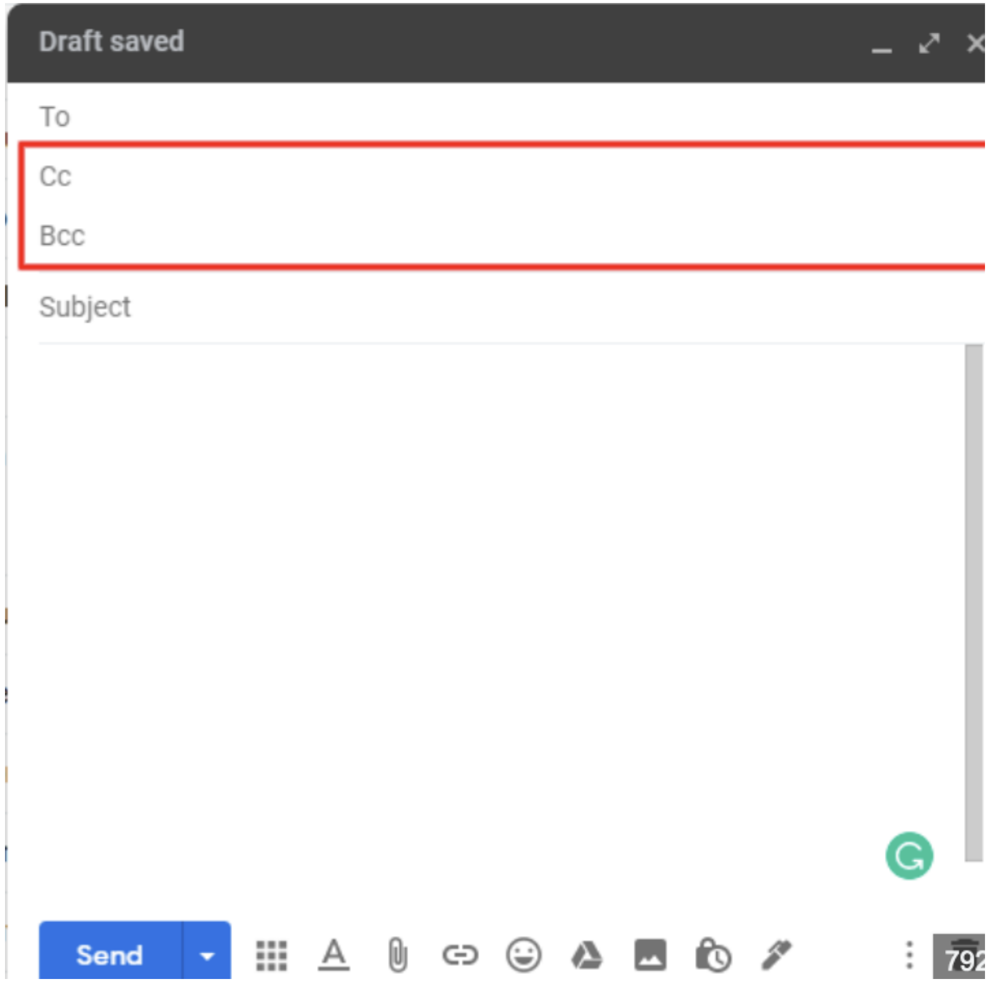
  different from SMTP MAIL FROM, RCPT TO: commands!
- Body: the "message"



京都大学

CC and BCC header lines

- E-mail clients usually have CC and BCC fields additional to the recipient address field.
- The CC (Carbon Copy) field allows to send copies of the email to other recipients.
- The CC header field is included in the sent message, so all recipients can view each other's addresses.
- The BCC (Blind Carbon Copy) field is used to include recipients without revealing their email addresses to others.
- The email client typically removes the BCC header field from the message, ensuring that recipients cannot see the addresses of others included in the BCC field.



京都大学

Today's lecture

2.2 Web and HTTP

2.3 electronic mail

- SMTP, POP3, IMAP

2.4 DNS

京都大学

IP Addresses

- To send a message from one device to another, each device needs a unique address.
- The internet uses IP addresses to identify devices and route data correctly.
- Each device has unique 32-bit (IPv4) or 128-bit (IPv6) IP address
  - 32-bit address, expressed as four decimal numbers separated by dots (e.g., 192.168.0.1).
  - 128-bit address, expressed in hexadecimal and separated by colons (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
- IP addresses are not permanently tied to a device, i.e., the IP address can change over time.
  - Some servers etc. have static IP addresses but most devices use dynamic IP addresses which are assigned by the network when they are connected and change over time.

Domain names

- A **domain** is a human-readable address used to identify a location on the internet.
- Domains consist of a series of characters (letters and numbers) and are structured in levels, separated by dots.
    - www.kyoto-u.ac.jp
    - www.google.com
    - etc.
- Humans use domain names to navigate the web, however computers use IP addresses to communicate.
- How to translate between domain names and IP addresses?

DNS: domain name system

people: many identifiers:

- Name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing by routers
- IP address looks like 127.7.106.83
- Hierarchical: Going from left to right gives more and more specific information about the location
- "name", e.g., www.yahoo.com - used by humans

Q: how to map between IP address and name, and vice versa ?

Domain Name System:

Service that translates hostnames to IP addresses

- *distributed database* implemented in hierarchy of many *name servers*
- *application-layer protocol:* hosts, name servers communicate to *resolve* names (address/name translation)

京都大学

DNS overview

1. **Domain Name Request:** When a user types a domain name into a browser, a DNS query is initiated.
2. **DNS Server Lookup**: The query is sent to a DNS resolver, which looks up the corresponding IP address from a DNS server.
3. **IP Address Retrieval**: The IP address is retrieved and sent back to the browser.
4. **Connection Established**: The browser uses the IP address to connect to the web server hosting the website.

京都大学

DNS example

- A browser requests the URL www.kyoto-u.ac.jp/index.html.
- In order to send the HTTP request to the Web server of Kyoto University, the host must first obtain its IP address.

1. The browser extracts the hostname www.kyoto-u.ac.jp from the URL and passes it to the DNS client application.
2. The DNS client sends a query containing the hostname to a DNS server.
3. The DNS client eventually receives a reply, including the IP address.
4. The browser receives the IP address and opens a TCP connection to the HTTP server process at that IP address.

DNS: services

- hostname to IP address translation
- host aliasing
  - A host with a complicated host name can have alias names.
  - The original hostname is called the **canonical hostname**.
  - DNS can be used to obtain the canonical hostname from an alias hostname.
- mail server aliasing
  - Usually host names for email servers are complicated.
  - DNS can be invoked by a mail application to obtain the canonical hostname.
  - This also allows to have both web server and mail server to have identical hostnames, e.g. kyoto-u.ac.jp.
- load distribution
  - Busy sites are replicated over multiple web servers. Each server runs on a different end-system and has a different IP address.
  - DNS allows a set of IP addresses to be associated with a single hostname.
  - DNS rotates the ordering of IP addresses in each reply, the traffic is distributed over multiple servers.

DNS: structure

DNS is decentralized

- The client side of DNS sends a query message into the network asking for the IP address of a specific hostname.
- DNS query and reply messages use UDP (port 53).
- There is a large number of DNS servers distributed around the world.
- No single DNS server has all mappings for all host names on the internet.

why not centralize DNS?

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

doesn't scale!

京都大学

Domain names and DNS

- Domain names consist of one or more parts that are separated by dots:

www.i.h.kyoto-u.ac.jp

- Domain names are hierarchical with the hierarchy descending from right to left.
- The right most part is called top-level domain.
  - Country specific (jp, de, uk, cn, …)
  - Generic (com, net, edu, gov, …)
- Each part to the left specifies a subdomain to the domain of the right.

Top Level Domains

Original top-level domains (generic)

- com – *Commercial*
  - The most popular and widely used TLD.
  - Originally intended for commercial businesses but now used broadly.
- .org – *Organization*
  - Commonly used by non-profit organizations, charities, and open-source projects.
- .net – *Network*
  - Initially designated for network-related organizations but now widely used for various purposes.
- .edu – *Education*
  - Restricted for educational institutions, especially universities and colleges.
- .gov – *United States Government*
  - Reserved for U.S. government entities and agencies.
- .mil – *United States Military*
  - Reserved for U.S. military organizations.

Top Level Domains

<u>Country code TLDs</u>

- .us – *United States*
- .jp – *Japan*
- .uk – *United Kingdom*
- .de – *Germany*
- .cn – *China*
- .fr – *France*
- .au – *Australia*
- .in – *India*
- .br – *Brazil*
- .ru – *Russia*
- .nz – *New Zealand*
- .sg – *Singapore*
- .no – *Norway*
- .tr – *Turkey*
- *… and many more*

Other popular TLDs

Generic:

- .info
- .xyz
- .online
- .shop
- .top
- etc.

Brand names

- .accenture
- .amazon
- .bmw
- .google
- .lego
- etc.

Cities and geographic regions:

- .kyoto
- .tokyo
- .london
- .berlin
- .nyc
- .eu
- .asia
- .africa
- etc.

京都大学

Other popular TLDs

Generic:

- .info
- .xyz
- .online
- .shop
- .top
- etc.

Brand names

- .accenture
- .amazon
- .bmw
- .google
- .lego
- etc.

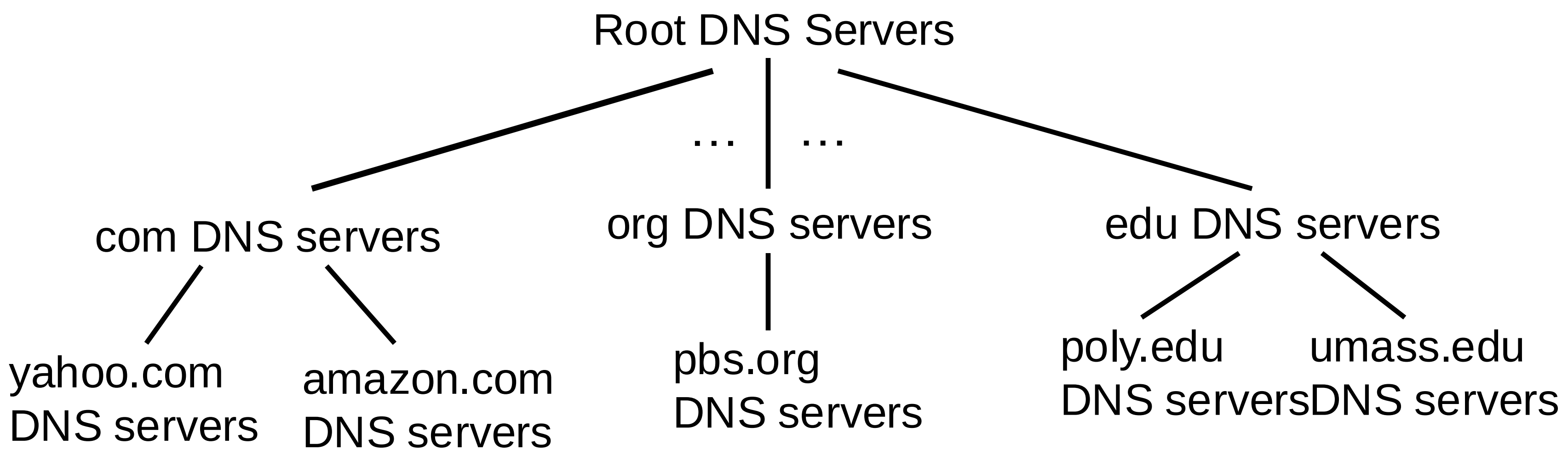
Cities and geographic regions:

- .kyoto
- .tokyo
- .london
- .berlin
- .nyc
- .eu
- .asia
- .africa
- etc.

Second-level domains

- **Second-level domains** are the part of a domain name that comes directly before the top-level domain.
  - For example, in example.com, example is the second-level domain of the .com TLD.
- Many country code domain registries introduce a second-level hierarchy underneath their country code TLD that indicates the type of entity intended to register.
- For example, the following second-level domains are common under the .jp TLD
  - .co.jp – Commercial organizations
  - .or.jp – Non-profit organizations
  - .ne.jp – Network services
  - .ac.jp – Academic institutions
  - .ed.jp – Primary and secondary educational institutions
  - .go.jp – Government organizations
  - .lg.jp: local government authorities
  - .ad.jp – Administrative organizations
  - .gr.jp – Groups and associations

DNS: a distributed, hierarchical database



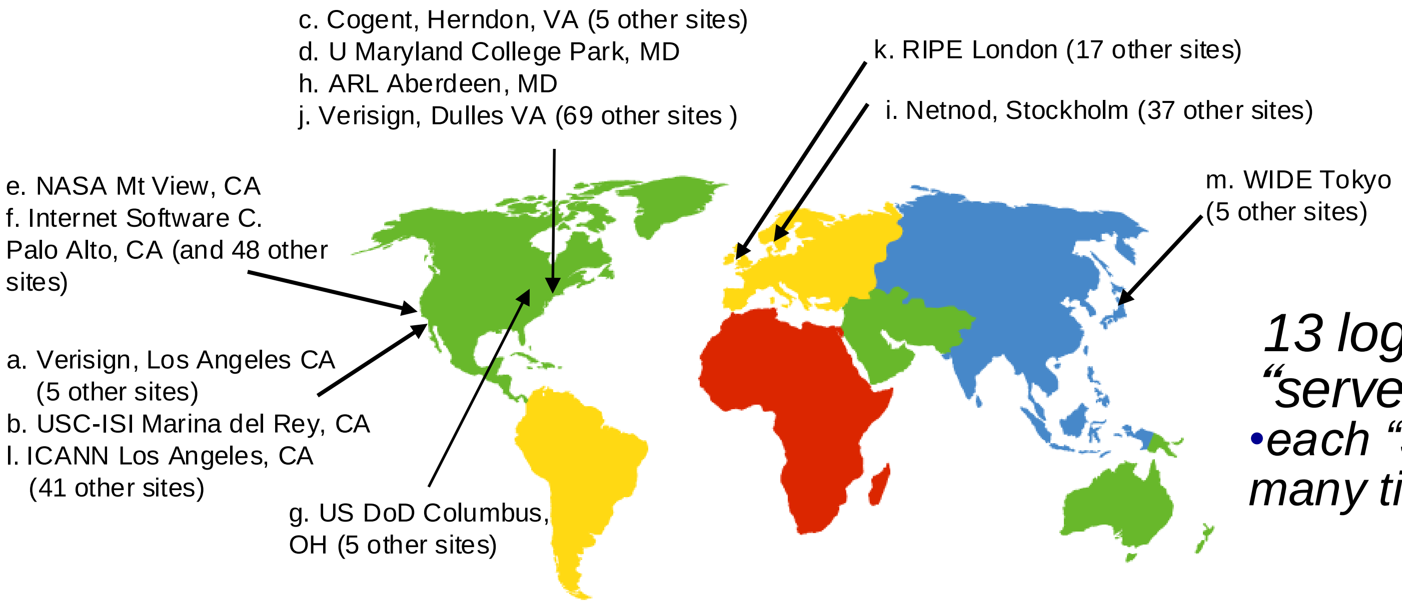
client wants IP for www.amazon.com; (simplified)

- 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

京都大学

DNS: root name servers

- contacted by local name server that can not resolve name
- root name server:
    - Provides IP address of top-level domain servers
    - gets mapping
    - returns mapping to local name server



13 logical root name "servers" worldwide

- *each "server" replicated many times*

京都大学

root-servers.org

FAQ RSSAC001 RSSAC002

Verisign USC-ISI Cogent UMD NASA OCIO ISC DISA DoD NIC ARL Netnod RIPE NCC ICANN WIDE

News and publications show all

2024-04-15	**Root Server System Operational Information**
2022-08-02	**Statement on adding ZONEMD to the root zone**
2021-03-30	**Statement on DNS Encryption**

Meeting agendas show all

2024-07-21	**IETF 120/Vancouver/Virtual (PDF)**
2024-03-17	**IETF 119/Brisbane/Virtual (PDF)**
2023-11-05	**IETF 118/Prague/Virtual (PDF)**

As of 2024-10-30T03:21:48Z, the root server system consists of 1916 instances operated by the 12 independent root server operators.

京都大学

M.ROOT-SERVERS.NET started its operational in Aug 1997 in Tokyo, JAPAN by WIDE Project. It was the only Root DNS Server operational in the west of the Pacific Ocean until Jan 2002 when global anycast is started to deploy in the Root DNS servers. Our service IP addresses are **202.12.27.33** and **2001:dc3::35** respectively, and the routing information covering these addresses is announced from **AS7500** regardless of the versions of IP. M-Root has been offering IPv6 service since February in 2008, and currently all M-Root anycast clusters offering IPv6 services.

Currently M-Root server is operated jointly by **WIDE Project** and **Japan Registry Services Co., Ltd.** (JPRS), a company running .JP ccTLD registry, under responsibility of WIDE Project.

Each cluster of M-Root server system consists of a few active servers for enhancing performance and redundancy.

- In Tokyo (JP), 3 clusters are connected to 3 independent major internet exchanges each -- DIX-IE (former NSPIXP-2), JPIX, and JPNAP.
- In Osaka (JP), 1 cluster touches down at all of NSPIXP-3, JPNAP-Osaka, JPIX-Osaka.
- In Seoul (KR), 1 cluster is connected to KINX.
- In San Francsico area, California (US) , 2 clusters are connected to SFMIX, AMS-IX Bay Area, and FCIX.
- In Paris (FR), 2 clusters are connected to SFINX, Equinix Paris, NL-ix, and France-IX.
- In Brisbane (AU), 1 cluster (as a local site) is connected to MegaIX Brisbane QLD-IX, and IX Australia (Brisbane).
- In Hanoi (VN), 1 cluster (as a local site) is connected to VNIX Hanoi.
- In Guam (GU), 1 cluster (as a local site) is connected to MARIIX.
- In Kuala Lumpur (MY), 1 cluster (as a local site) is connected to MyIX.
- In Bangkok (TH), 1 cluster (as a local site) is connected to BKNIX.
- In Singapore (SG), 1 cluster is connected to SingAREN Open Exchange and Equinix Singapore.
- In Kaohsiung (TW), 1 cluster (as a local site) is connected to TWIX.
- In Jakarta (ID), 1 cluster (as a local site) is connected to IIX-Jakarta and OpenIXP.
- In Ulan Bator (MN), 1 cluster (as a local site) is connected to MISPA-IXP.
- In Hong Kong (HK), 1 cluster is connected to HKIX.
- In Phnom Penh (KH), 1 cluster (as a local site) is connected to CNX.
- In Kathmandu (NP), 1 cluster (as a local site) is connected to npIX DH (NPIX PTS).
- In Sao Paulo (BR), 1 cluster is connected to IX.br/PTT Sao Paulo.
- In Dhaka (BD), 1 cluster (as a local site) is connected to BDIX.
- In Mumbai (IN), 1 cluster (as a local site) is connected to NIXI Mumbai.

TLD and 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 (in total more than 1000).
- Network Solutions maintains servers for .com TLD
- Japan Registry Services (JPRS) for .jp 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

nonauthoritative DNS servers:

- Other DNS servers can also provide hostname to IP mappings (e.g. cached entries). In that case they are called nonauthoritative DNS servers for the domain.

京都大学

Domain Servers

Root Servers:

- Located at the top of the DNS hierarchy.
- A distributed set of servers around the world (e.g., A through M root servers).
- Serve as the initial point of contact for DNS queries.

Top-Level Domain (TLD) Servers:

- Manage the highest level of the domain name system hierarchy (e.g., .com, .org, .net).
- Direct queries to authoritative servers for specific domains within their TLD.
- Play a crucial role in global DNS infrastructure.

Authoritative DNS Servers:

- Responsible for providing information about specific domains (e.g., example.com).
- Hold the most up-to-date and accurate data for their assigned domains.
- Answer queries from TLD servers and resolvers regarding specific domains.

Local DNS server

- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one
  - also called "default name server"
- 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

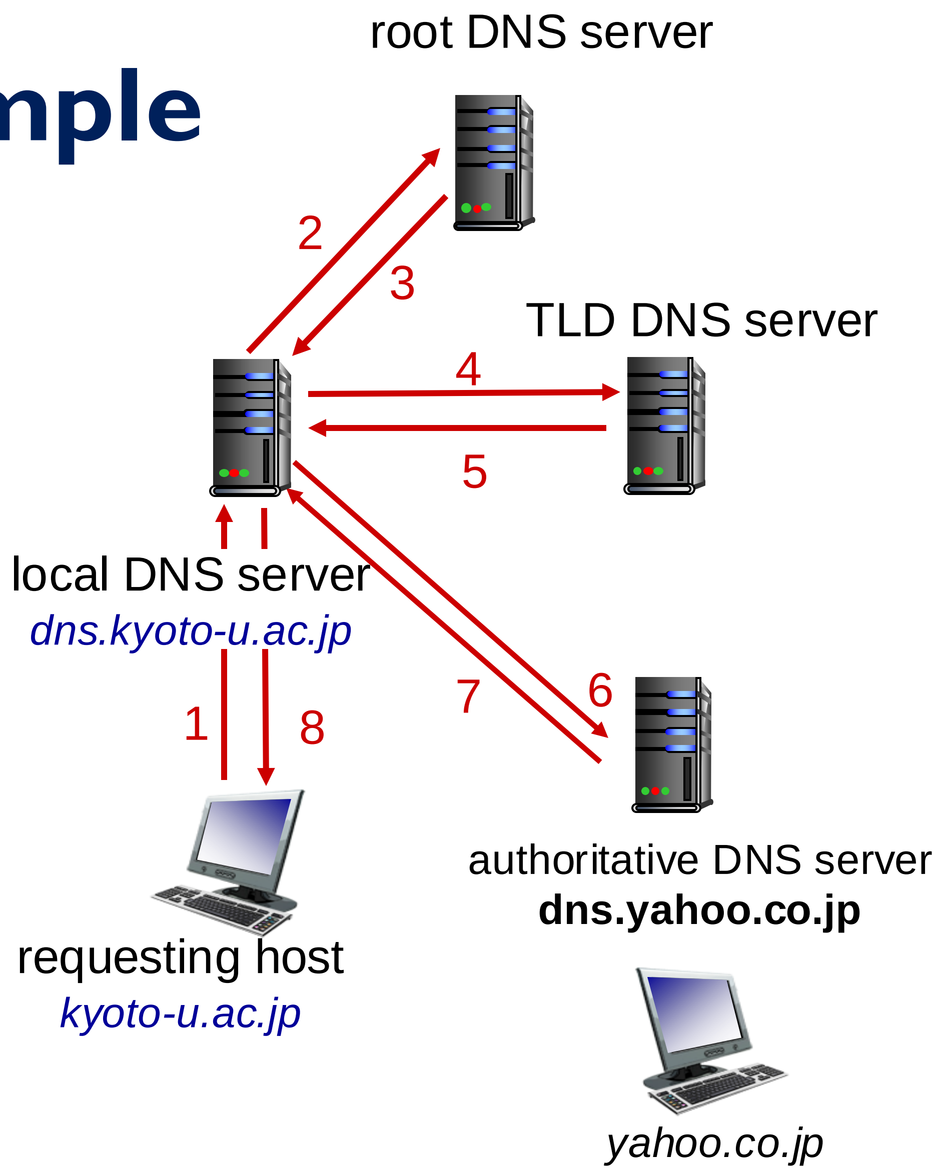
京都大学

DNS name resolution example

- host at kyoto-u.ac.jp wants IP address for yahoo.co.jp
- DNS has two types of queries – iterative and recursive

iterated query:

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

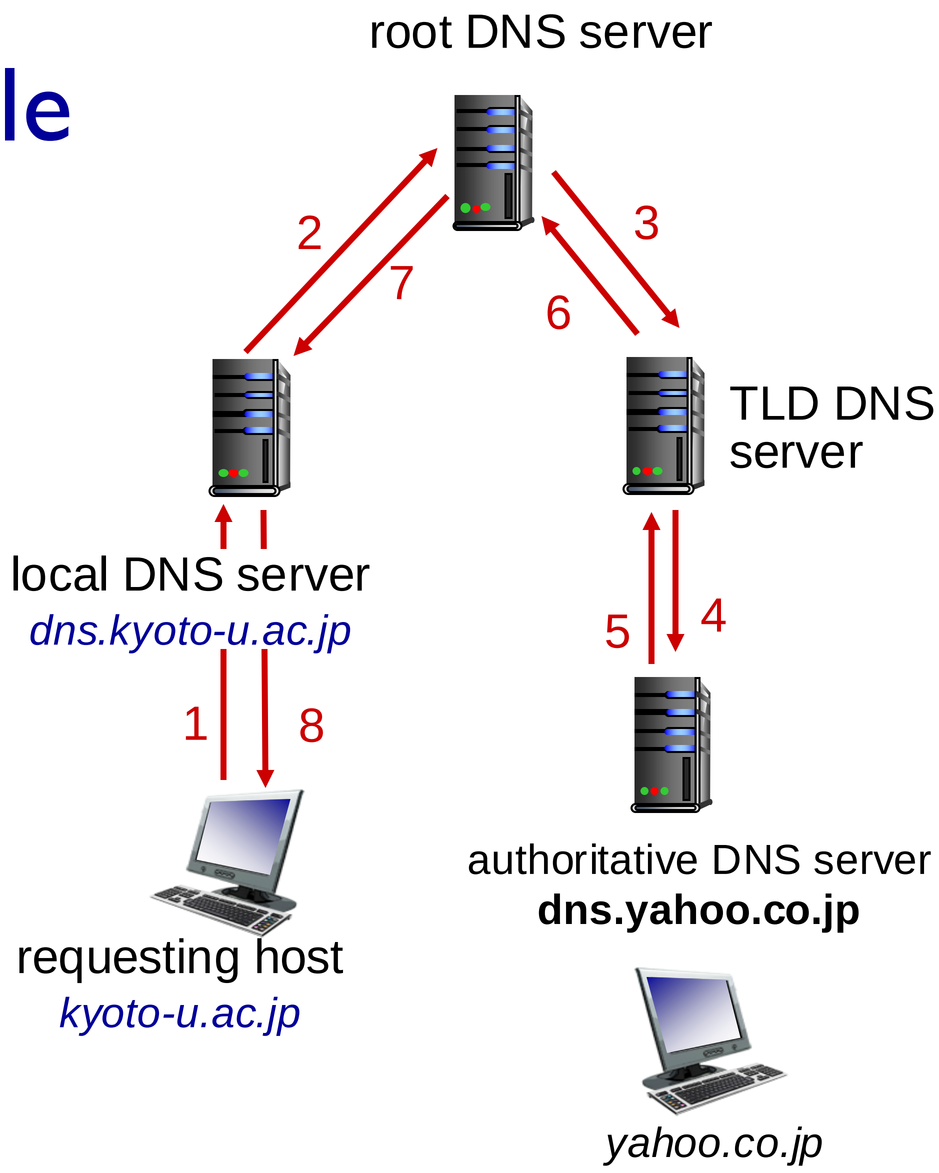


京都大学

DNS name resolution example

recursive query:

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



京都大学

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

京都大学

DNS records

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`**

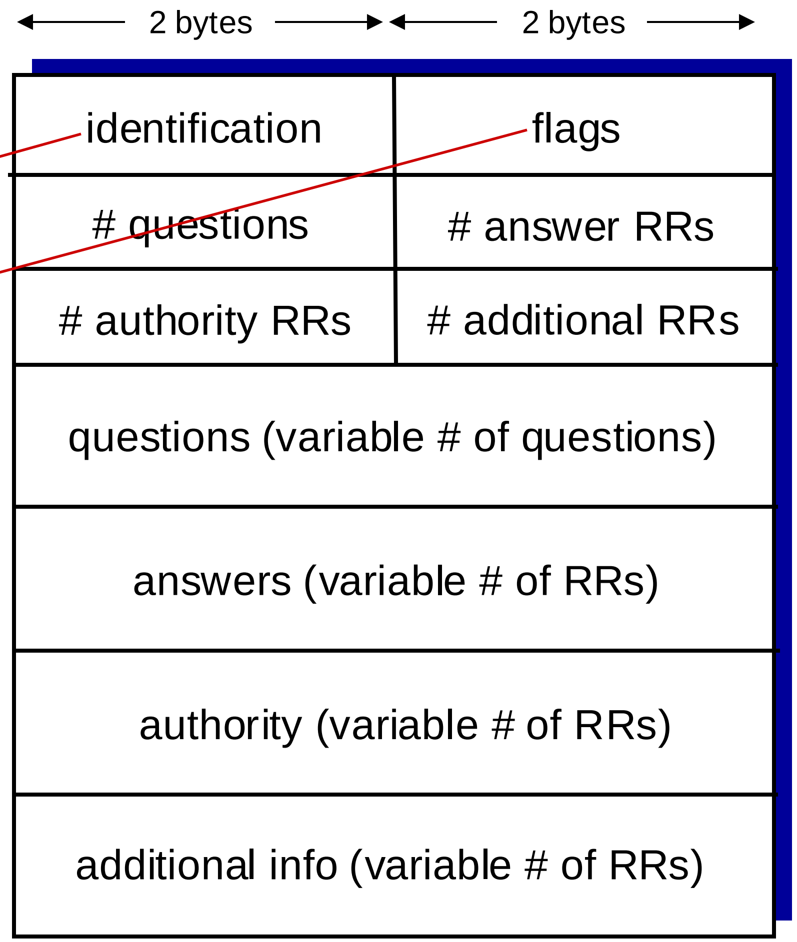
京都大学

DNS protocol, messages

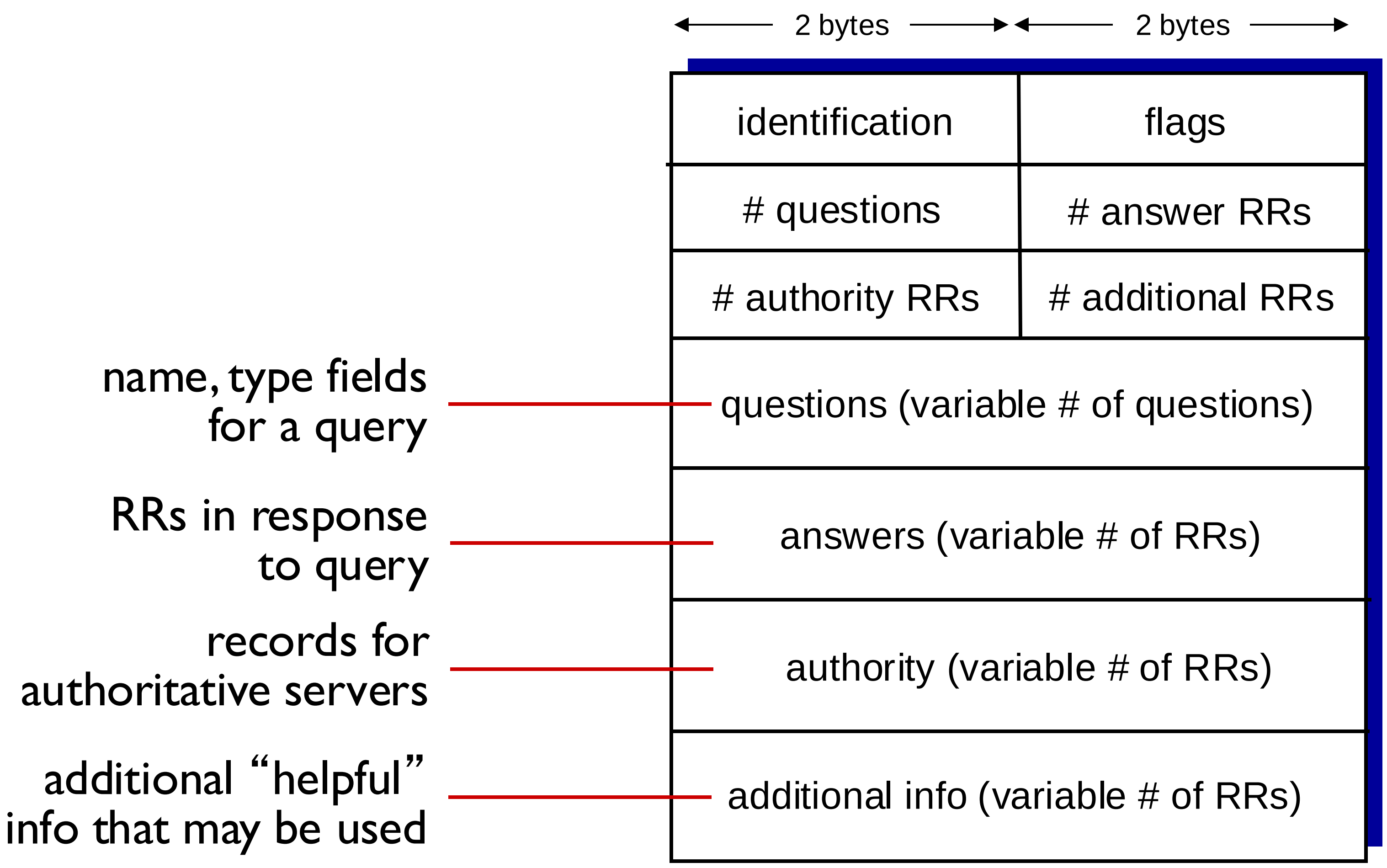
- *query* and *reply* messages, both with same *message format*

message header

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



DNS protocol, messages



京都大学

Inserting records into DNS

- example: new startup "Network Utopia"
- register name networkutopia.com at *DNS registrar* (e.g., Network Solutions for com domain)
  - 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

京都大学

DNS records

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`**

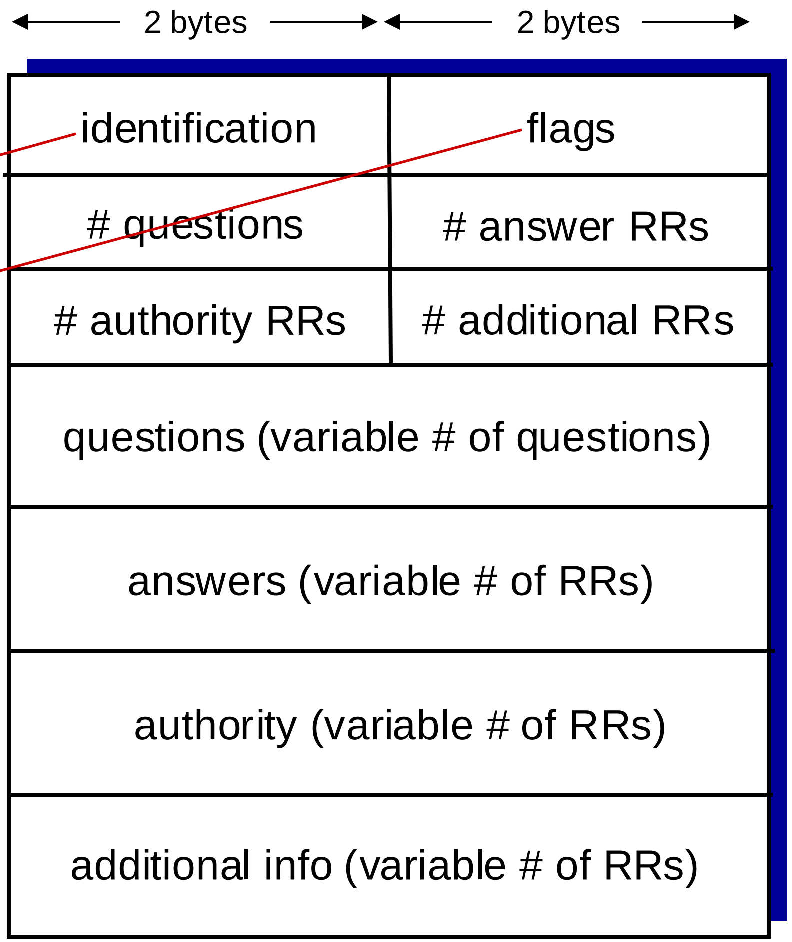
京都大学

DNS protocol, messages

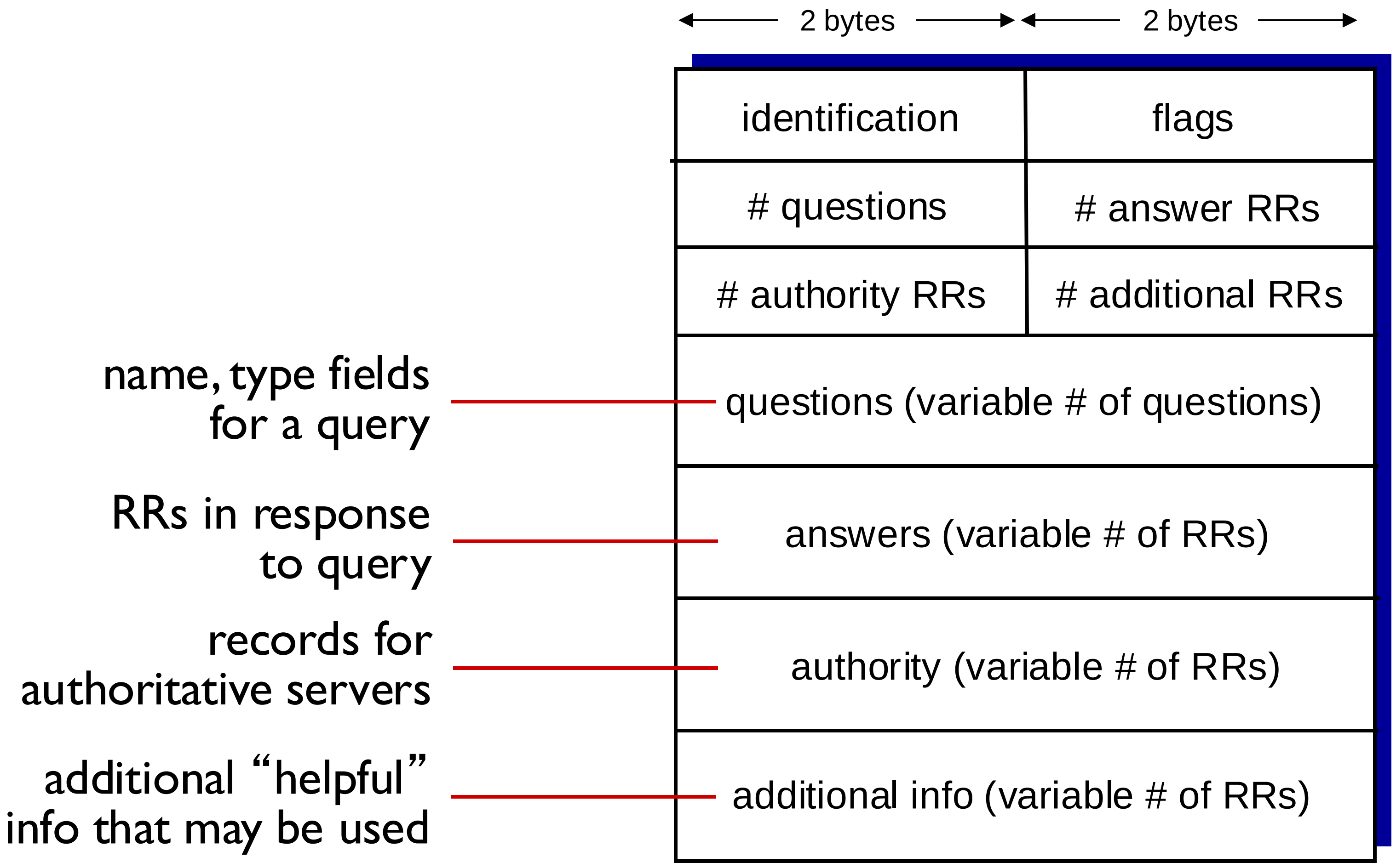
- *query* and *reply* messages, both with same *message format*

message header

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



DNS protocol, messages



Today's lecture

2.1 DNS

2.2 video streaming and content distribution networks (CDNs)

2.3 Principles of the Transport Layer

京都大学

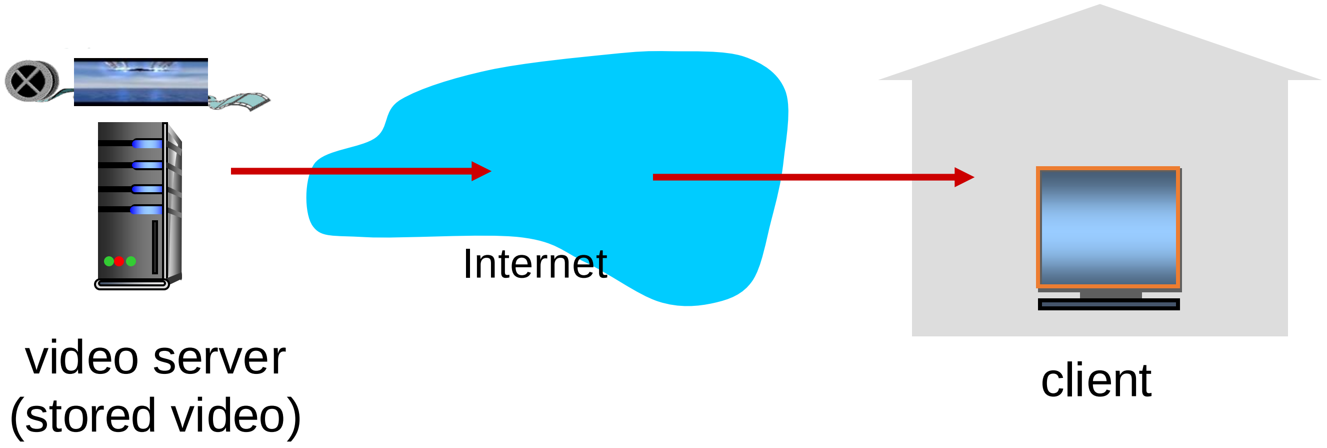
Video Streaming and CDNs: context

- video traffic: major consumer of Internet bandwidth
  - Netflix, YouTube, etc.
- challenge: scale
  - single mega-video server won't work
- challenge: heterogeneity
  - different users have different capabilities (e.g., wired versus mobile; bandwidth rich versus bandwidth poor)
- *solution:* distributed, application-level infrastructure

京都大学

Streaming stored video:

simple scenario:



京都大学

Streaming multimedia: DASH

- *DASH: D*ynamic, *A*daptive *S*treaming over *H*TTP
- *server:*
  - divides video file into multiple chunks
  - each chunk stored, encoded at different rates
  - *manifest file:* provides URLs for different chunks
- *client:*
  - periodically measures server-to-client bandwidth
  - consulting manifest, requests one chunk at a time
    - chooses maximum coding rate sustainable given current bandwidth
    - can choose different coding rates at different points in time (depending on available bandwidth at time)

京都大学

Streaming multimedia: DASH

- *DASH: Dynamic, Adaptive Streaming over HTTP*
- *"intelligence"* at client: client determines
  - *when* to request chunk (so that buffer starvation, or overflow does not occur)
  - *what encoding rate* to request (higher quality when more bandwidth available)
  - *where* to request chunk (can request from server that is "close" to client or has high available bandwidth)

京都大学

Content distribution networks

- *challenge:* how to stream content (selected from millions of videos) to hundreds of thousands of *simultaneous* users?

- *option 1:* single, large "mega-server"
  - single point of failure
  - point of network congestion
  - long path to distant clients
  - multiple copies of video sent over outgoing link

This solution *doesn't scale*

京都大学

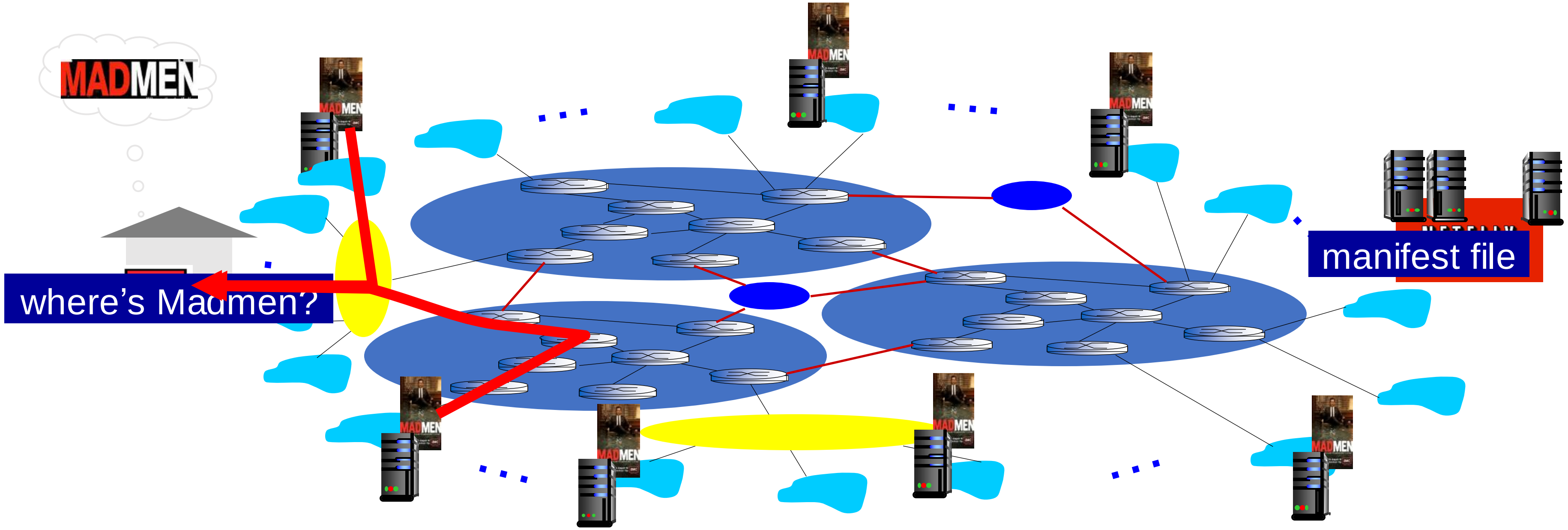
Content distribution networks

- *challenge:* how to stream content (selected from millions of videos) to hundreds of thousands of simultaneous users?

- *option 2:* store/serve multiple copies of videos at multiple geographically distributed sites *(CDN)*

Content Distribution Networks (CDNs)

- CDN: stores copies of content at CDN nodes
  - e.g. Netflix stores copies of MadMen
- subscriber requests content from CDN
  - directed to nearby copy, retrieves content
  - may choose different copy if network path congested



京都大学