Computer Networks
COL 334/672

Application Layer: Email and DNS

Tarun Mangla

Slides adapted from KR

Sem 1, 2024-25

Recap: Application Layer

- HTTP
- Email
- DNS
- P2P
- Video streaming

Email

- First “killer” application over the Internet

Journey of an Email

- Scenario: Arjun from IITD wants to send an email to his friend Aditi in IITB

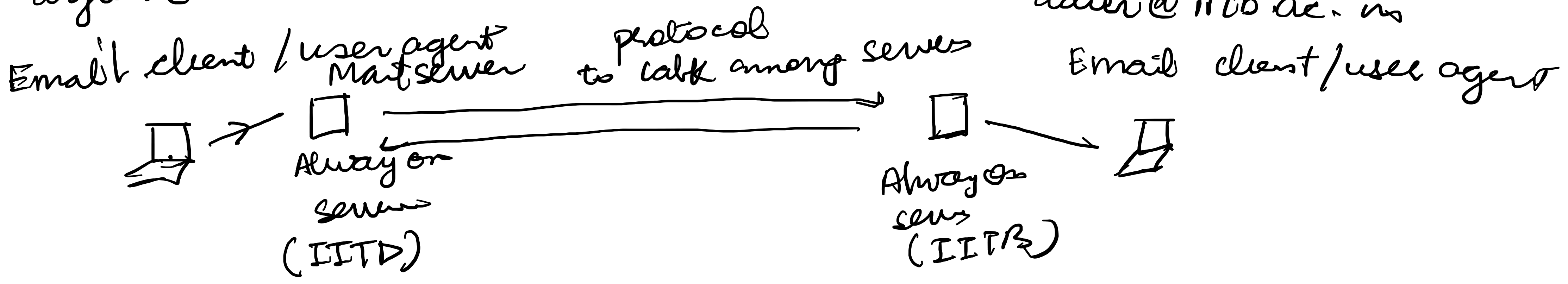
What are the major components involved in email application?

DNS : hostname to IP address

E Email address

Sender Email
arjun@iitd.ac.in

Receiver
aditi@iitb.ac.in



Three major components:

- user agents
- mail servers (SMTP / IMAP)
- simple mail transfer protocol: SMTP

Mail message format

SMTP: protocol for exchanging e-mail messages, defined in RFC 5321 (like RFC 7231 defines HTTP)

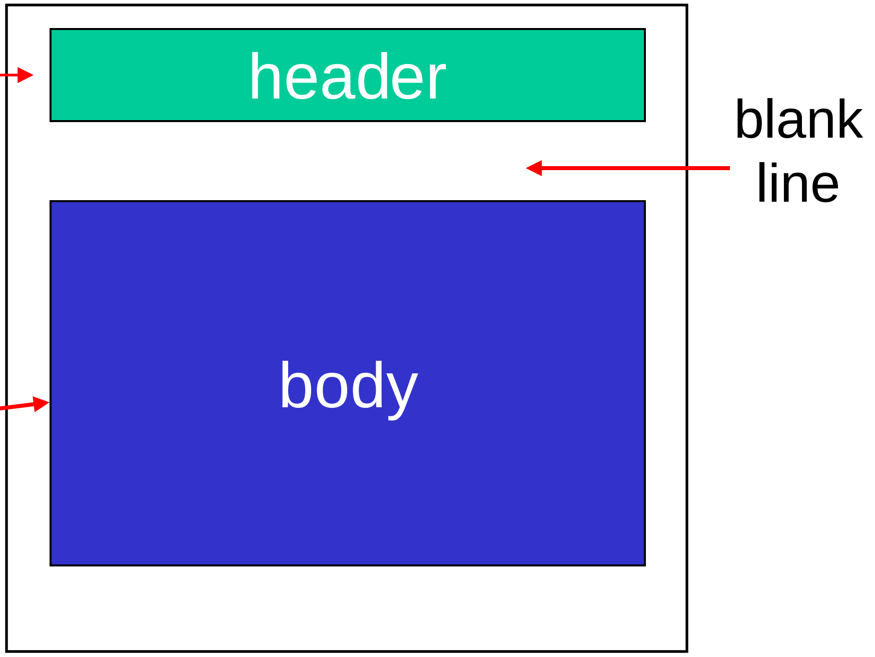
RFC 2822 defines *syntax* for e-mail message itself (like HTML defines syntax for web documents)

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

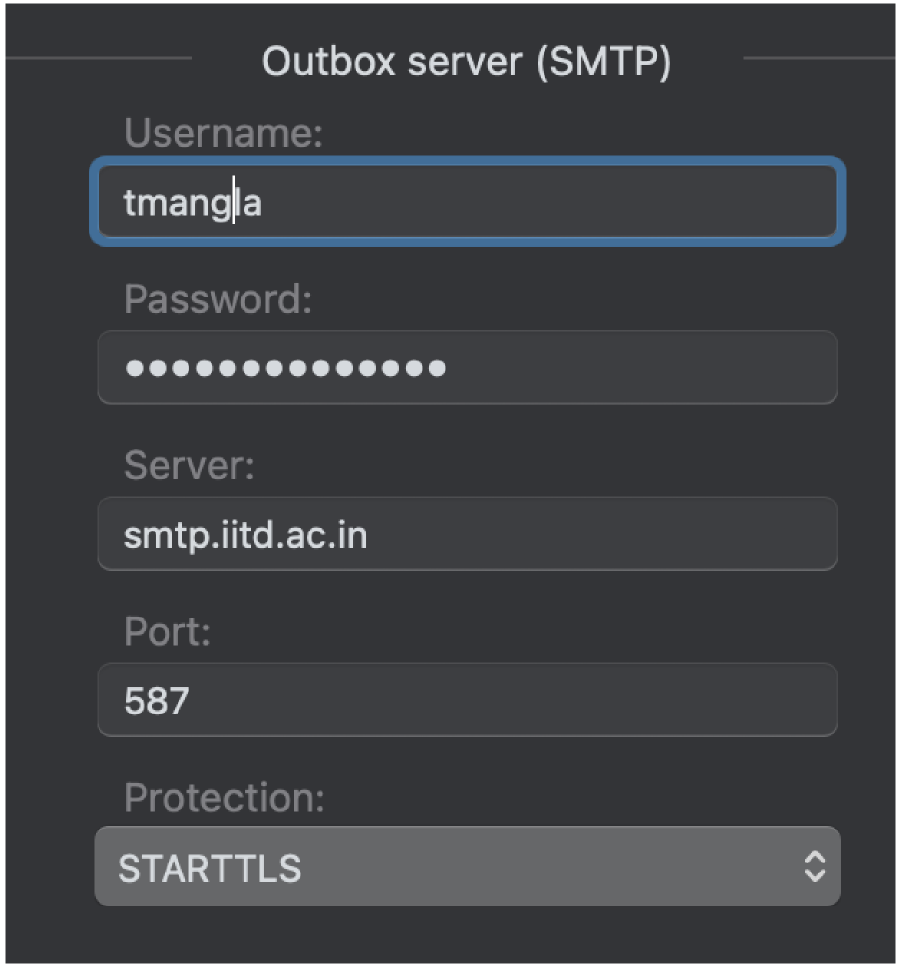
  these lines, within the body of the email message area different from SMTP MAIL FROM:, RCPT TO: commands!
- Body: the "message" , ASCII characters only

attaching an object

cc / bcc

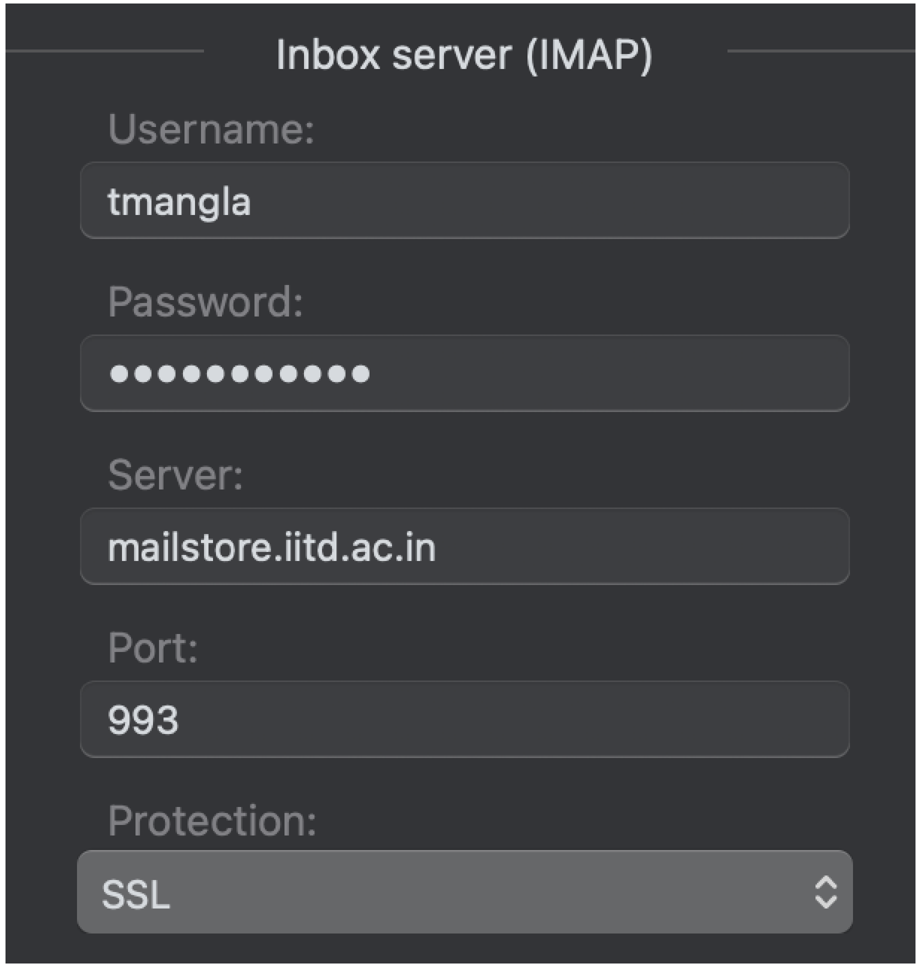


Configuring Mailbox



Outbox server (SMTP)

Username:	tmangla
Password:	●●●●●●●●●●●●●●
Server:	smtp.iitd.ac.in
Port:	587
Protection:	STARTTLS

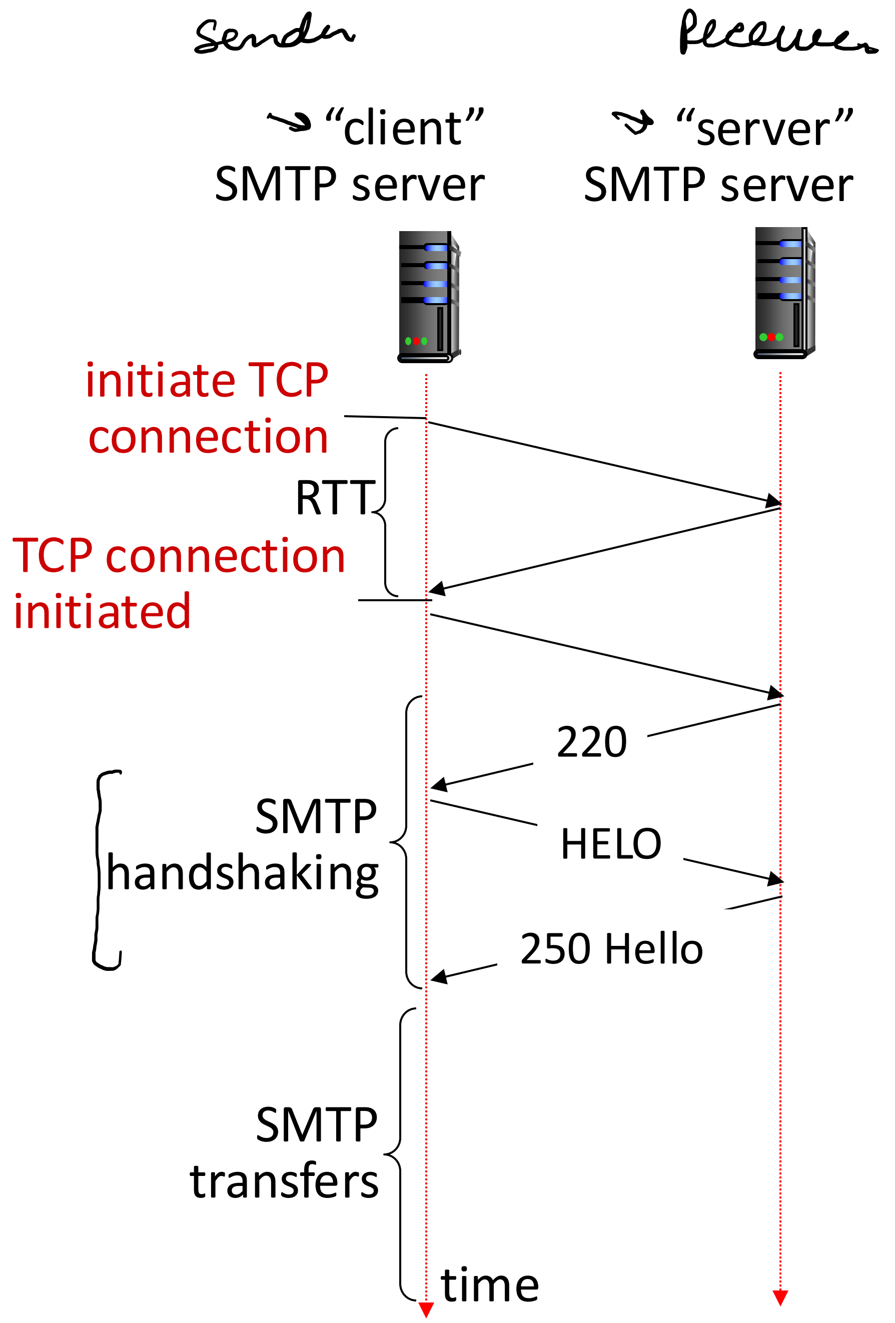


Inbox server (IMAP)

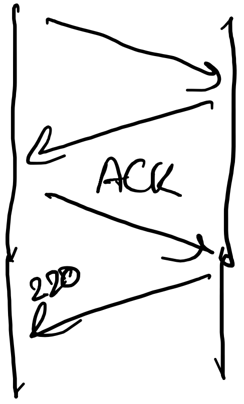
Username:	tmangla
Password:	●●●●●●●●●●●●
Server:	mailstore.iitd.ac.in
Port:	993
Protection:	SSL

SMTP RFC (5321)

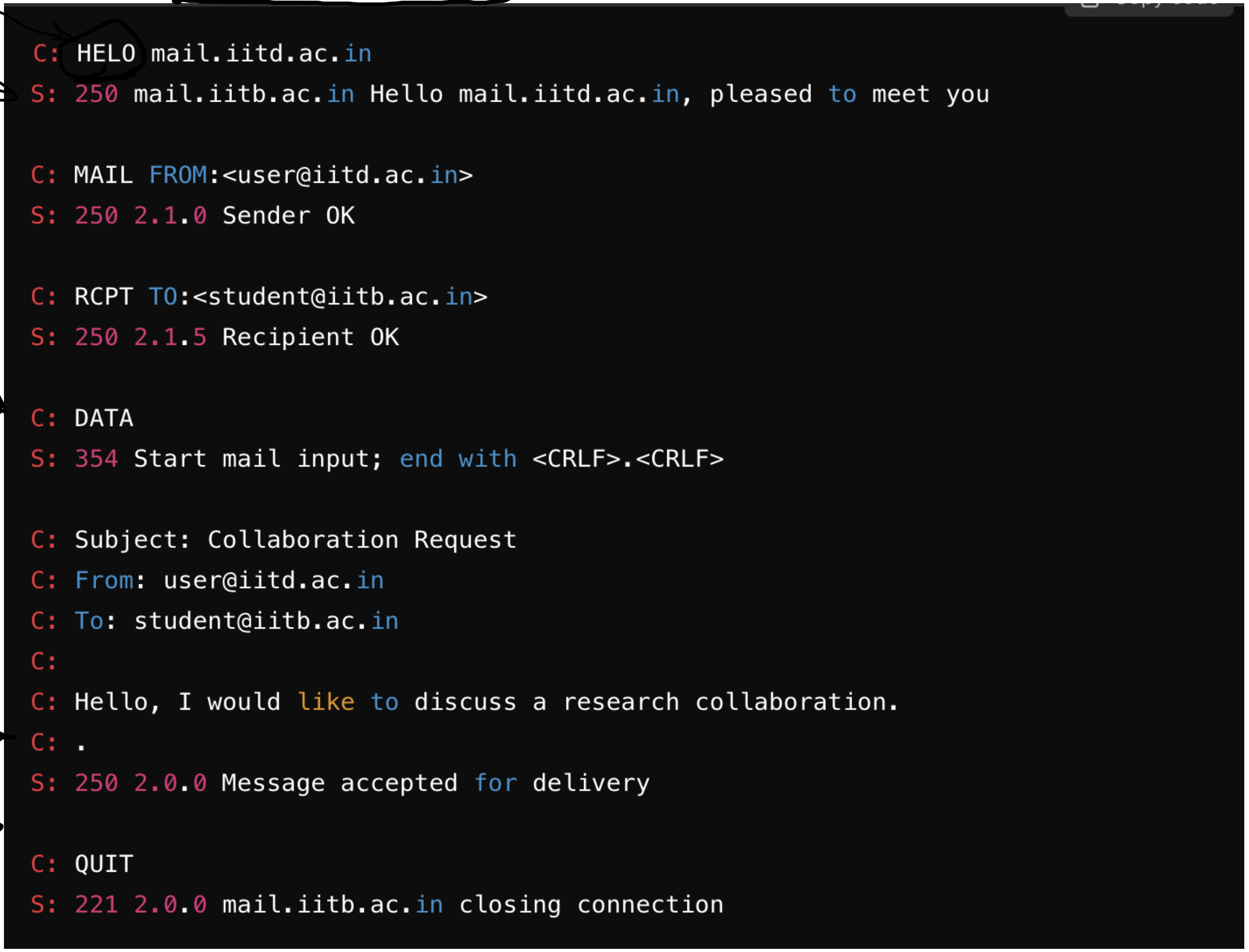
- uses TCP to reliably transfer email message from client (mail server initiating connection) to server, port 25
  - direct transfer: sending server (acting like client) to receiving server
- three phases of transfer
  - SMTP handshaking (greeting)
  - SMTP transfer of messages
  - SMTP closure
- command/response interaction (like HTTP)
  - commands: ASCII text
  - response: status code and phrase



Sample SMTP Interaction

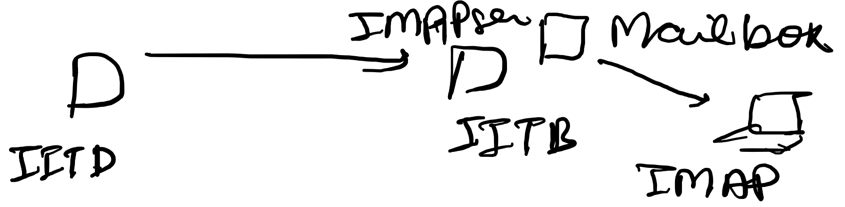


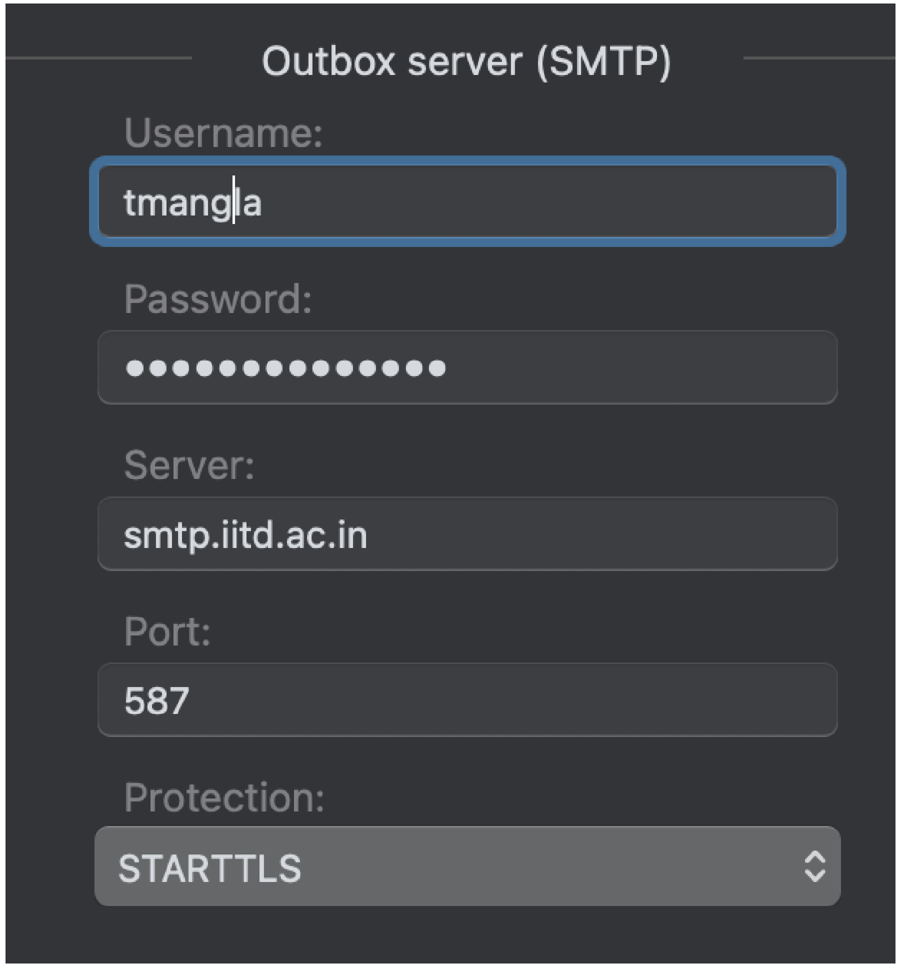
```
S: 220 mail.iitb.ac.in
```



```
C: HELO mail.iitd.ac.in
S: 250 mail.iitb.ac.in Hello mail.iitd.ac.in, pleased to meet you

C: MAIL FROM:<user@iitd.ac.in>
S: 250 2.1.0 Sender OK

C: RCPT TO:<student@iitb.ac.in>
S: 250 2.1.5 Recipient OK

C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>

C: Subject: Collaboration Request
C: From: user@iitd.ac.in
C: To: student@iitb.ac.in
C:
C: Hello, I would like to discuss a research collaboration.
C: .
S: 250 2.0.0 Message accepted for delivery

C: QUIT
S: 221 2.0.0 mail.iitb.ac.in closing connection
```

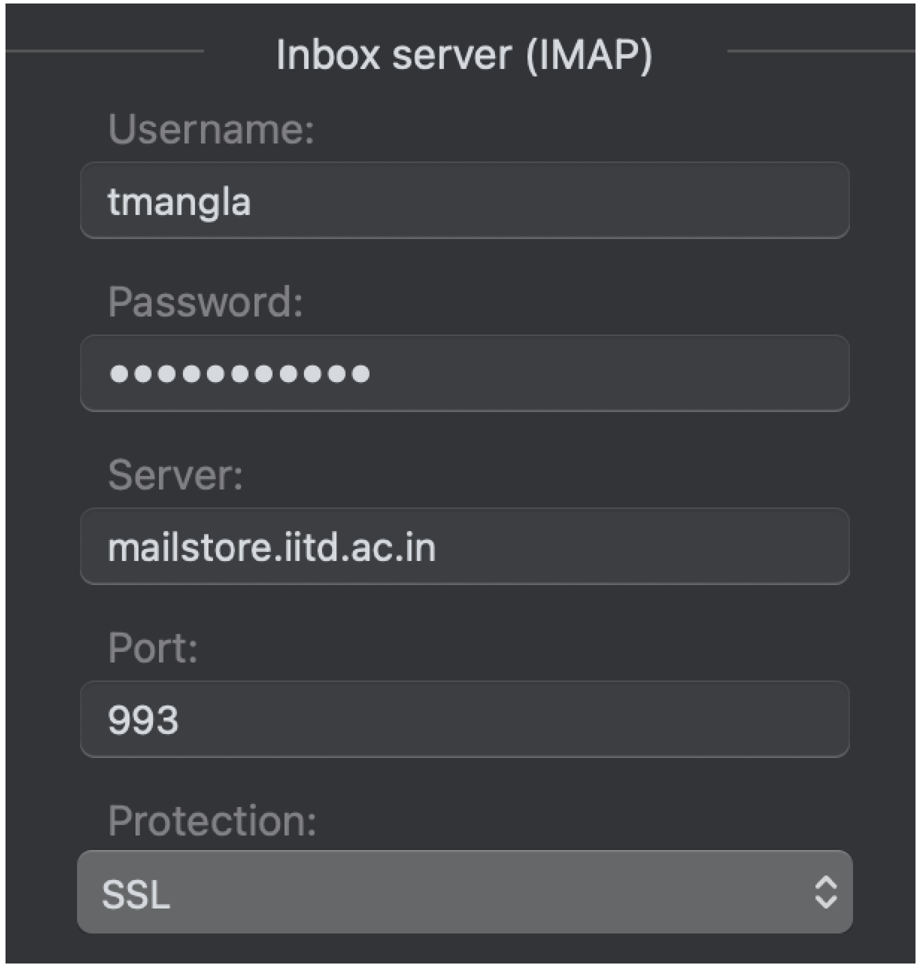
Configuring Mailbox





Outbox server (SMTP)

Username:	tmangla
Password:	●●●●●●●●●●●●●●
Server:	smtp.iitd.ac.in
Port:	587
Protection:	STARTTLS

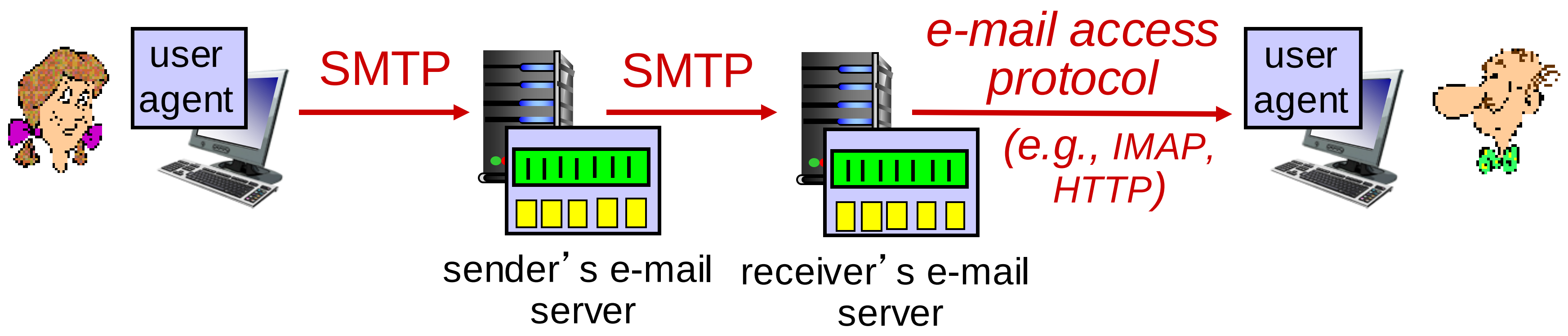


Inbox server (IMAP)

Username:	tmangla
Password:	●●●●●●●●●●●●
Server:	mailstore.iitd.ac.in
Port:	993
Protection:	SSL

$$\begin{tikzcd} \text{C}\,\square \arrow[r, "\text{HTTP}"'] & \square\ \boxed{\text{HTTP Server}}^{\text{user agent}} \arrow[r] & \square\ \text{SMTP Server}^{\text{gmail}} \end{tikzcd}$$

Retrieving email: mail access protocols



- SMTP: delivery/storage of e-mail messages to receiver's server
- mail access protocol: retrieval from server
  - IMAP: Internet Mail Access Protocol [RFC 3501]: messages stored on server, IMAP provides retrieval, deletion, folders of stored messages on server
- HTTP: gmail, Hotmail, Yahoo!Mail, etc. provides web-based interface on top of STMP (to send), IMAP (or POP) to retrieve e-mail messages

Pull-based protocol

$\xrightarrow{\text{GET}}$

$\xleftarrow[\text{Response}]{}$

Text

DATA

TEXT

SMTP: observations

comparison with HTTP:

- HTTP: client pull
- SMTP: client push
- both have ASCII command/response interaction, status codes
- HTTP: each object encapsulated in its own response message
- SMTP: multiple objects sent in multipart message

SMTP/Email:

IMAGES

- Open standards
- Interoperability among email clients

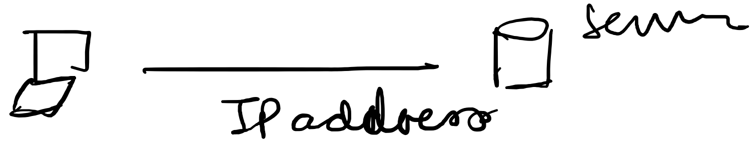
ATTACH

- Exemplify the design spirit of Internet

Recap: Application Layer

- HTTP
- Email
- **DNS** → Domain Name System
- P2P
- Video streaming

google.com



$\boxed{\text{DNS}}$ Host Name/Domain name

→ IP address

DNS: Domain Name System

DNS: Database
① Dynamic → consistency
② Store a large # of domain name

- Humans understand names (google.com),
- Internet hosts, routers understand IP address (12.123.12.12)
- Q: how to map between IP address and name, and vice versa ?

→

Domain Name System (DNS):

③ low latency
④ Reliable

- *phone book* of the Internet
- *application-layer protocol:* hosts, DNS servers communicate to *resolve* names (address/name translation)
  - *note:* core Internet function, implemented as application-layer protocol
  - complexity at network's "edge"

DNS: Design Goals

Large number of host names

- ~ billion records, each simple

handles many *trillions* of queries/day:

- *many* more reads than writes
- *performance matters:* almost every Internet transaction interacts with DNS - msecs count!

reliability

How do we go about designing such a system?

Can you keep the database on a single machine?

→ SPF

→ storage

→ slow server

→ Non-technical

Approach 1: Centralized DNS

- single point of failure
- traffic volume
- maintenance
- ..

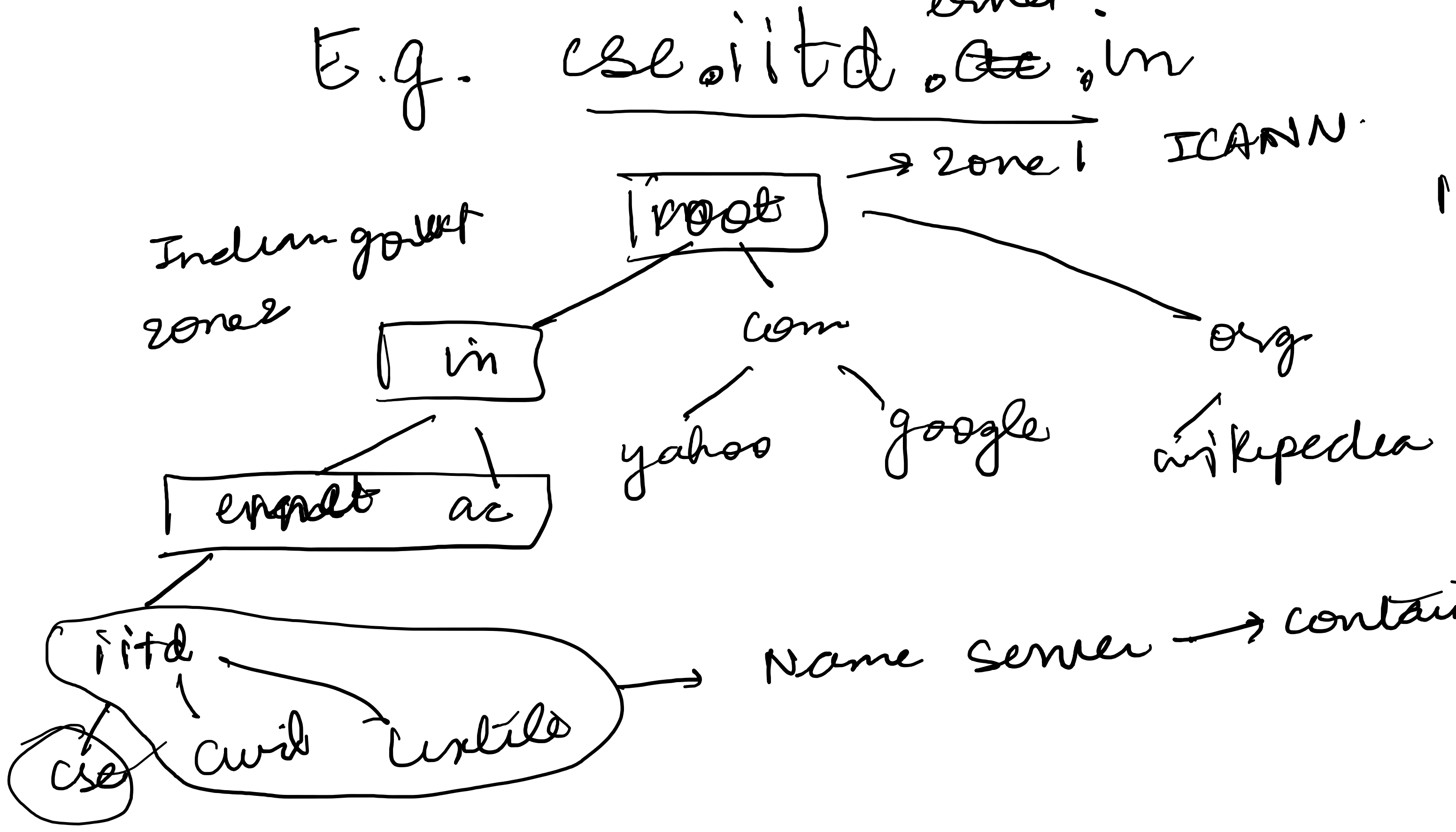
Decentralized and distributed system

Multiple entities man

↳ Not physically centrally

- Q: On what basis to decentralize?

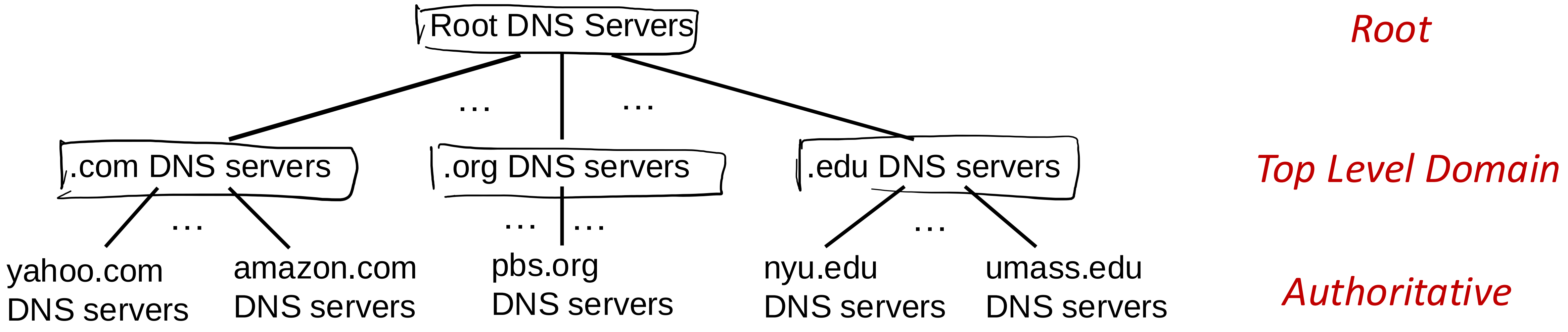
~~Hierarchical domain name space~~ Name | IP address



Decentralized and distributed system

- Q: On what basis to decentralize? Hierarchical domain name space
- Partition domain name hierarchy into zones managed by some authority
  - E.g., ICANN is responsible for storing information about top-level domains
- Each zone corresponds to a name server

Name servers

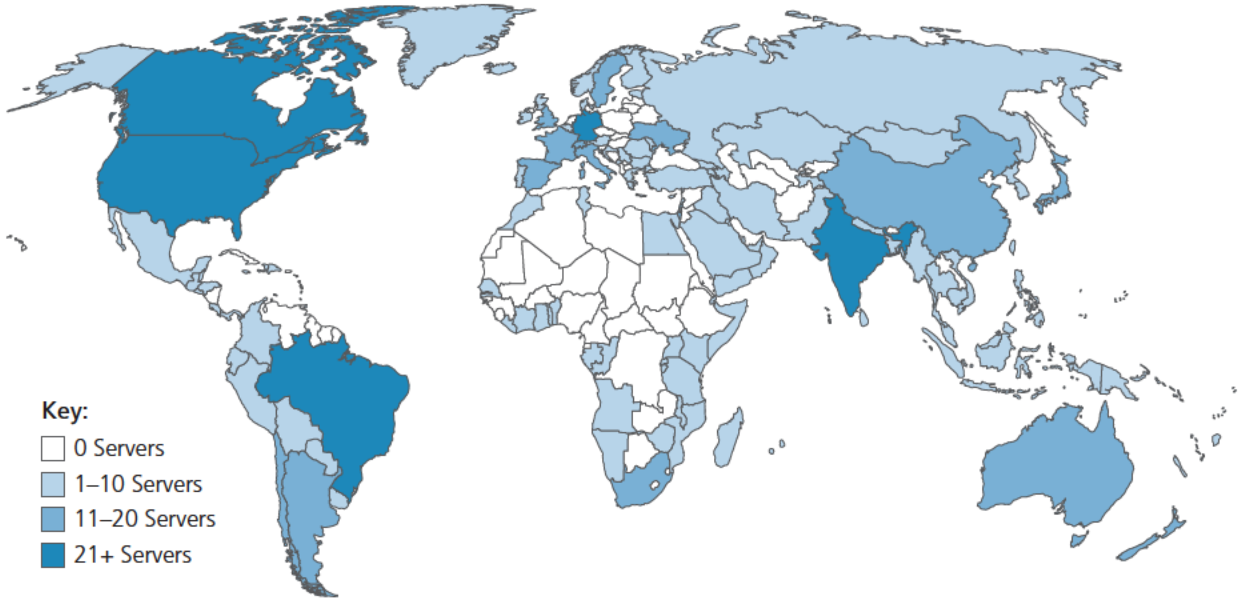


- Name servers are replicated and may be geographically distributed for reliability

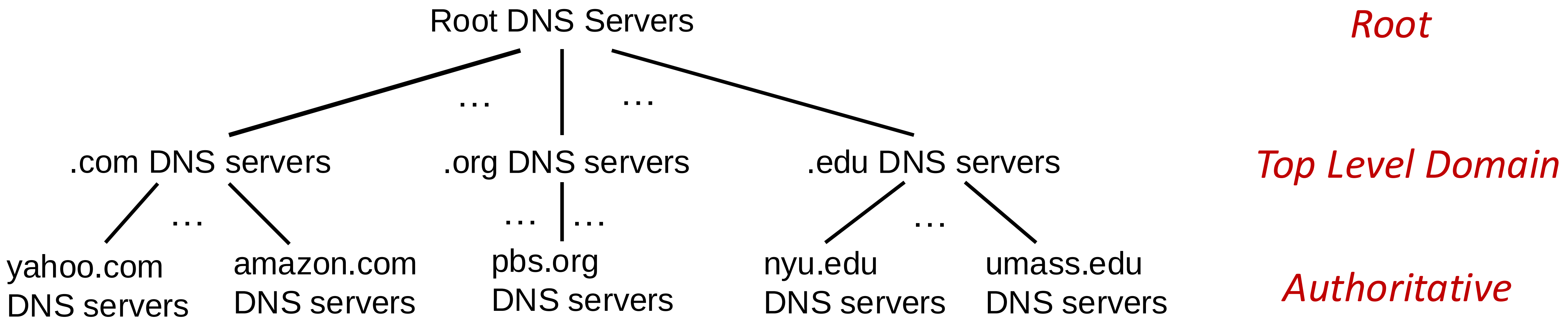
DNS: root name servers

- official, contact-of-last-resort by name servers that can not resolve name
- *incredibly important* Internet function
  - Internet couldn't function without it!
  - DNSSEC – provides security (authentication, message integrity)
- ICANN (Internet Corporation for Assigned Names and Numbers) manages root DNS domain

13 logical root name "servers" worldwide each "server" replicated many times (~200 servers in US)



Name servers

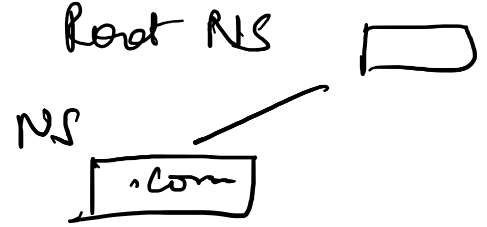


- Name servers are replicated and may be geographically distributed for reliability
- Name server implements zone information as collection of resource records

DNS records

Name , V

Domain Name , IP address



DNS: distributed database storing resource records (RR)

RR format: (name, value, type, ttl), class

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
mail server records
- value is name of SMTP mail server associated with name

Example

Root Name server

① .in , dns.xyz.in , NS, TTL

② dns.xyz.in , 112.112.112.112, A,

③ com , dns.xyz.com , NS

④ dns.xyz.com , 113.113.113.113 , A

IN name server

.ernet.in , dns.ernet.in
NS

dns.ernet.in , x.y.z.w, A

How to do name resolution?

Censorship : DNS

Public DNS resolvers

: 8.8.8.8

1.1.1.1 → cloudflare resolver

Local DNS name servers

- when host makes DNS query, it is sent to its *local* DNS server
  - Local DNS server returns reply, answering:
    - from its local cache of recent name-to-address translation pairs (possibly out of date!)
    - forwarding request into DNS hierarchy for resolution
  - each ISP has local DNS name server; to find yours:
    - MacOS: `% scutil --dns`
    - Windows: `>ipconfig /all`
- local DNS server doesn't strictly belong to hierarchy