CN (IT-3001)

Application Layer: Client-Server Paradigm

Prof. Amit Jha

School of Electronics Engineering (SOEE)

KIIT Deemed to be University

Content

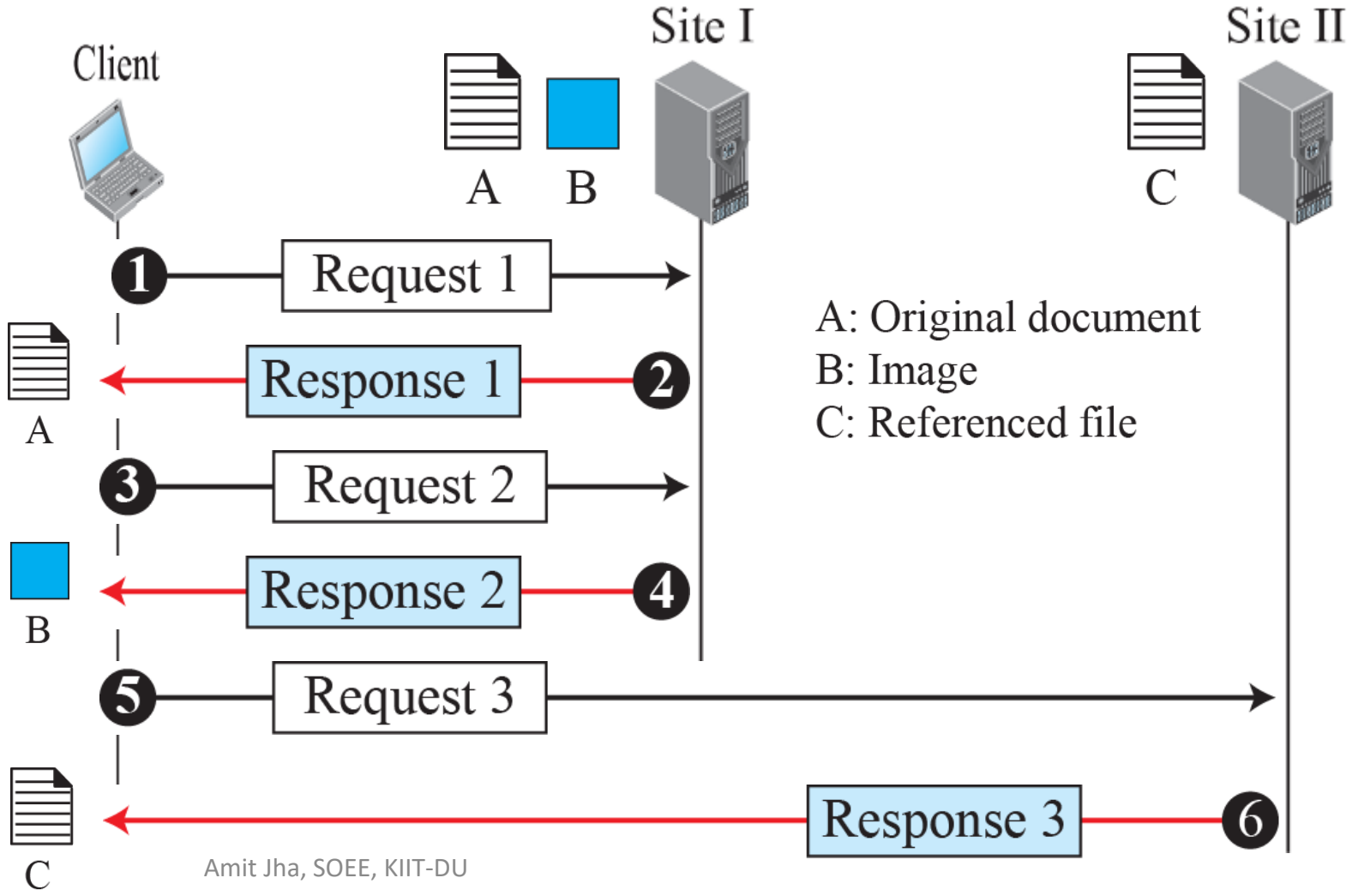
- Different application of standard **client-server paradigm** such as
  1. Web
  2. HTTP
  3. FTP
  4. E-mail
  5. DNS
- Different application of standard **Peer-to-Peer paradigm** such as
  1. P2P network
  2. P2P Application: BitTorrent

Client-Server Application: WWW

- It is a distributed client-server service, in which a client using a browser can access a service using a server.
- However, the service provided is distributed over many locations called *sites*.
- Each site holds one or more documents, referred to as web pages.
- A web page can be simple or composite:
- Simple web page→ No link to other web pages
- Composite web page→ link to other web pages

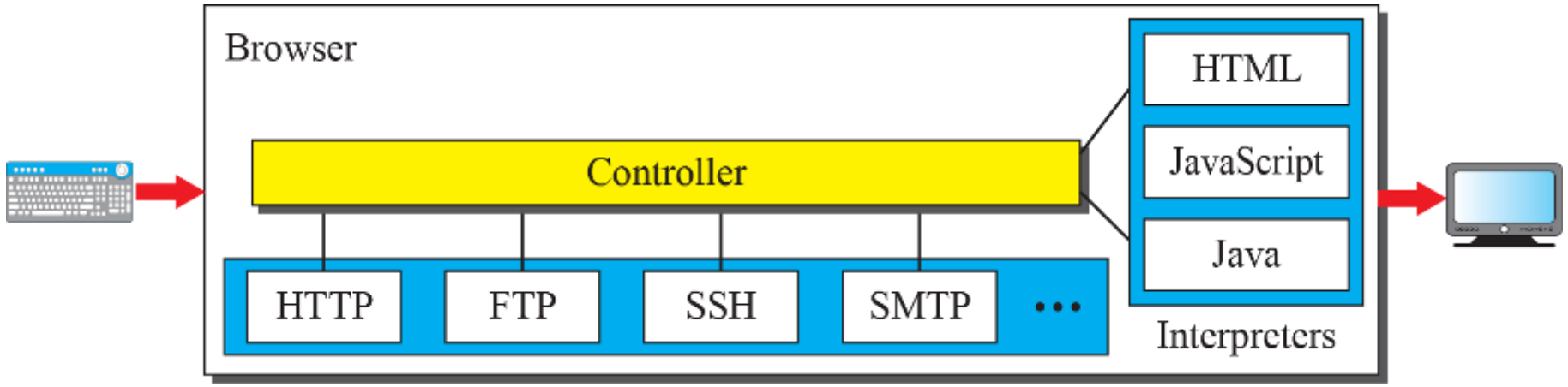
WWW: Example

- File **A, B** and **C** are **independent** Web pages.
- Links for web page B and C are included in A.
- However, B and C can also be reached independently.



WWW: key elements

- **Web Client (Browser):** Its responsibility is to interpret and display a web page.
- Each browser usually consist of three parts:
  1. **Controller:** It receives input from the keyboard and mouse and uses the client program to access the documents. After the documents are accessed, the controller uses one of the interpreters to display the document on the screen.
  2. **Client Protocol**: Client protocols are like HTTP, FTP, SSH, etc.
  3. **Interpreter:** It can be HTML, Java, or JavaScript depending upon the type of documents.



WWW: key elements

- **Web Server:** The web pages are stored at the servers which are expected to run for all the time. To improve the efficiency, a cache memory is used generally. Ex. Apache and Microsoft Information Server.
- **Uniform Resource Locator (URL)**: It is an identifier used to distinguish a web page from another.
  - It has three fields: *host, port* and *path*.
  - However, before defining the web page, we need to tell the browser what client-server application we want to use.
  - Thus, we now have **four identifiers** (instead of three) to uniquely identify a web page.

1. **Protocol:** Its like a **vehicle** to be used to access the web page. E.g., HTTP, FTP, etc.
2. **Host:** It is IP address or unique name of the server.
3. **Port:** It is 16-bit integer which is normally pre-defined for client-server based application.
   For e.g., if HTTP is used, then port no is 80 (well-known). However, if different port is used, it should be clearly written while accessing the web page.
4. **Path:** It is location of the file in underlying operating system.

Example:

- **protocols://host/path** → used for most of the time
- **protocols://host:port/path** → used when port number is needed

Practical Example

https://sites.google.com/view/amitkumarvjha/home

https://192.168.101.1:8090/httpclient.html

WWW: key elements

- **Web Documents:** It can be grouped into three broad categories: static, dynamic and active.

1. Static Documents:
   - It is fixed-content documents that are created and stored in a server.
   - The content of the file is decided when it is created.
   - Static documents are prepared using: HTML, XML, XSL, XHTML
2. **Dynamic Documents:**
   - It is created by a web server whenever a browser requests the documents.
   - Dynamic documents are prepared using: Java Server Page (JSP), Active Server Page (ASP), ColdFusion.
3. **Active Documents:**
   - For many applications, we need a program or a script to be run at the client site. These are called active documents.
   - *Java applets*, is used to create such documents. *JavaScript* can also be used, but it needs to download and then run.

Client-Server Application: HTTP

- The HTTP is a protocol that is used to define how the client-server programs can be written to retrieve web pages from the web.
- The server uses port no 80; whereas client uses a temporary port number.
- HTTP uses TCP service of the transport layer which is reliable, and thus;
  - Client and server do not need to worry for the error in message exchanged or loss of the message as it will be taken care by the TCP.
- I have a doubt…
  - **How a file on the web page will be accessed**?

- **How a file on the web page will be accessed**?
- **Answer:**
  - It is accessed by establishing a connection (TCP Connection) between client and server.
  - The data is retrieved based on request and response method. Where,
    - Client sends request and
    - Server responds to the client's request
  - As it is TCP, so once data transmission is over, the connection has to be released.

Can you answer these doubts!

1. If object is located on the different servers, do we need to establish a TCP connection to each server ?

2. If objects are located on the same server, do we need to establish a TCP connection to the server
    1. Each time to access the object?
    2. Only once to access all the objects on the server?

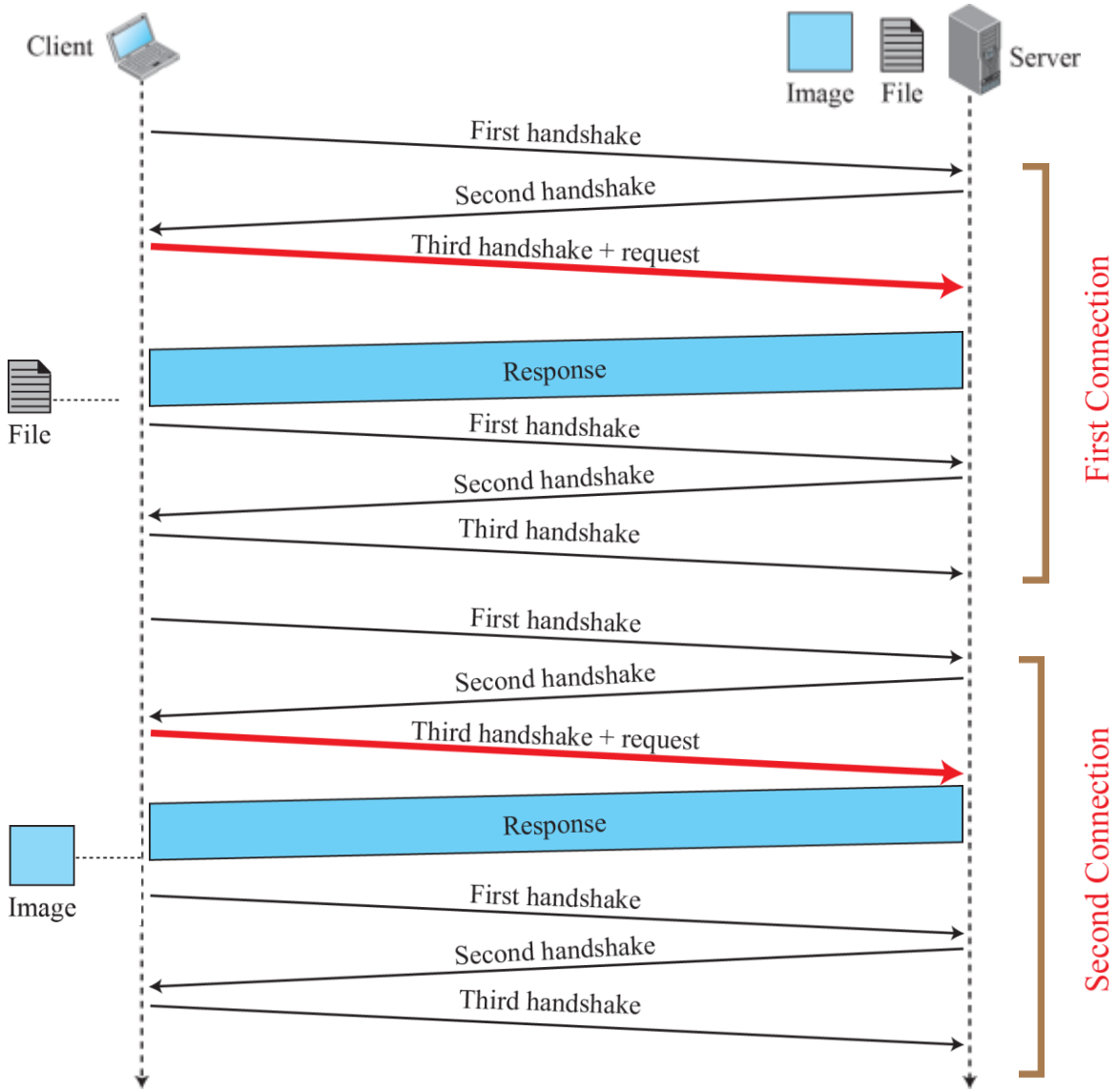
Can you answer these doubts!

1. If object is located on the different servers, do we need to establish a TCP connection to each server ?→ Yes, we have to!

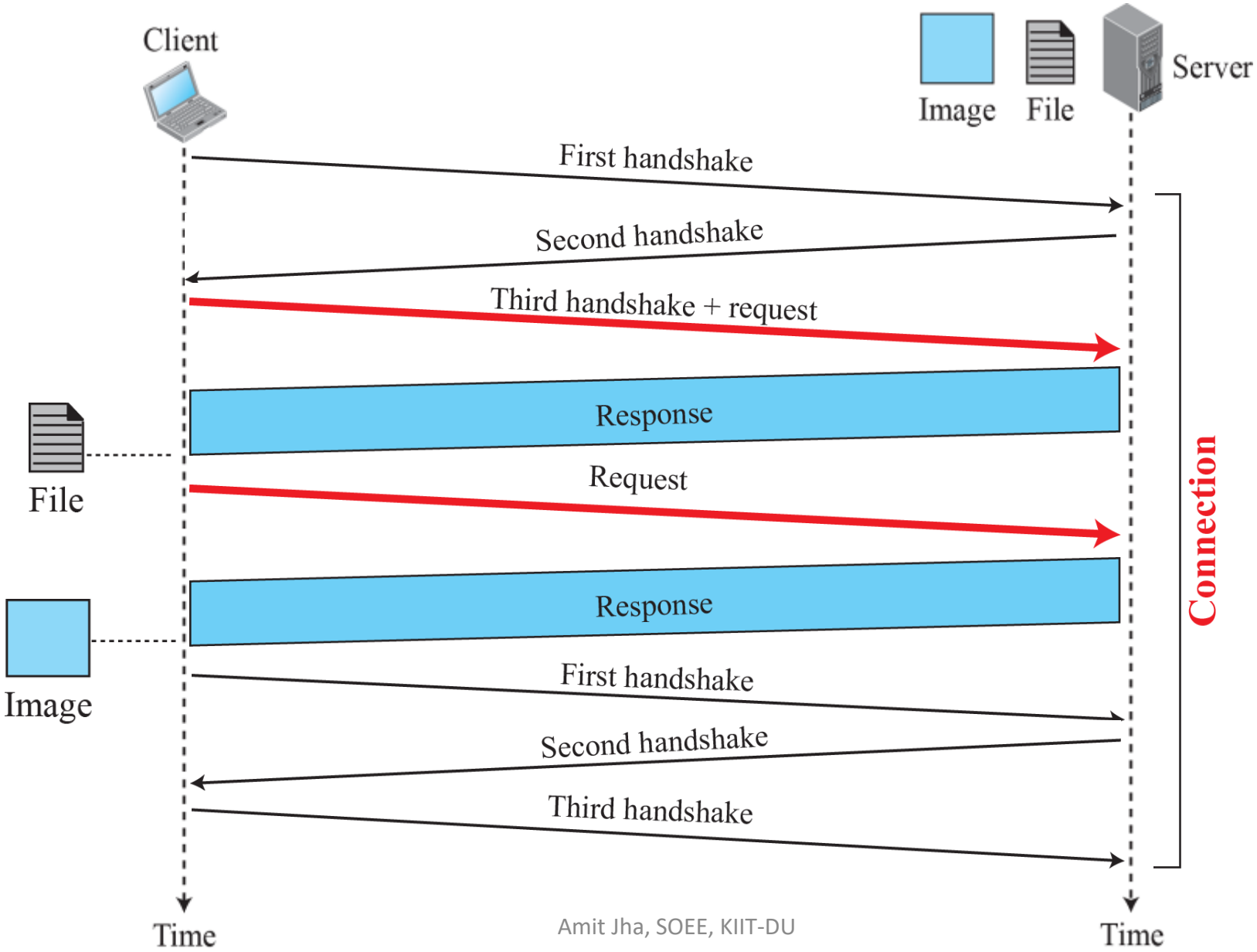
2. If objects are located on the same server, do we need to establish a TCP connection to the server
   1. Each time to access the object? → yes, this way is also possible
      → It is known as **nonpersistent method**
   2. Only once to access all the objects on the server? → yes, this way is also possible
      → It is known as **persistent method**

- **Nonpersistent connection:**
  - In this one TCP connection is made for each request/response in following stages.
    - Client opens a TCP connection and sends a request.
    - The server sends the response and closes the connection.
    - The client then reads the data until it encounters and end-of-file marker, it then closes the connection.
- **Persistent connection:**
  - In persistent connection, the server leaves the connection open for more requests after sending a response.
  - The server can close the connection either
    - at the request of a client or
    - If a time-out has been reached

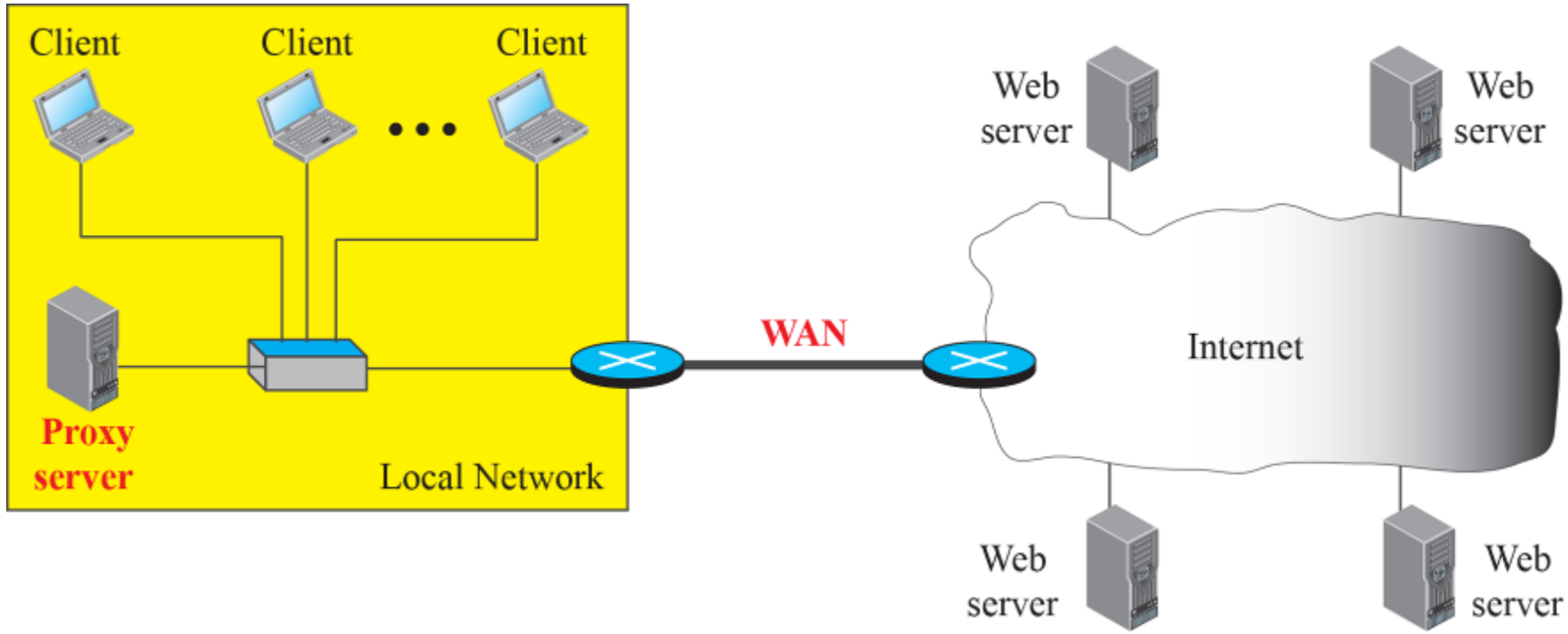
Scenario of Nonpersistent communication to get a file and an image from the same server



Scenario of *persistent communication* to get a file and an image from the same server



Proxy Server



- The proxy server is installed in the local network.
- When an HTTP request is created by any of the clients (browsers), the request is first directed to the proxy server.
- If the proxy server already has the corresponding web page, it sends the response to the client.
- Otherwise, the proxy server acts as a client and sends the request to the web server in the Internet.
- When the response is returned, the proxy server makes a copy and stores it in its cache before sending it to the requesting client.

Hone Your Understanding (HYU)

1. What is cache and cookies?
2. Why do we need it?
3. What are the advantages?
4. How does it work?
5. Are they same like proxy server?
6. How cookies are maintained at both client and server to behave the HTTP as a state-full protocol?

Client-Server Application: FTP

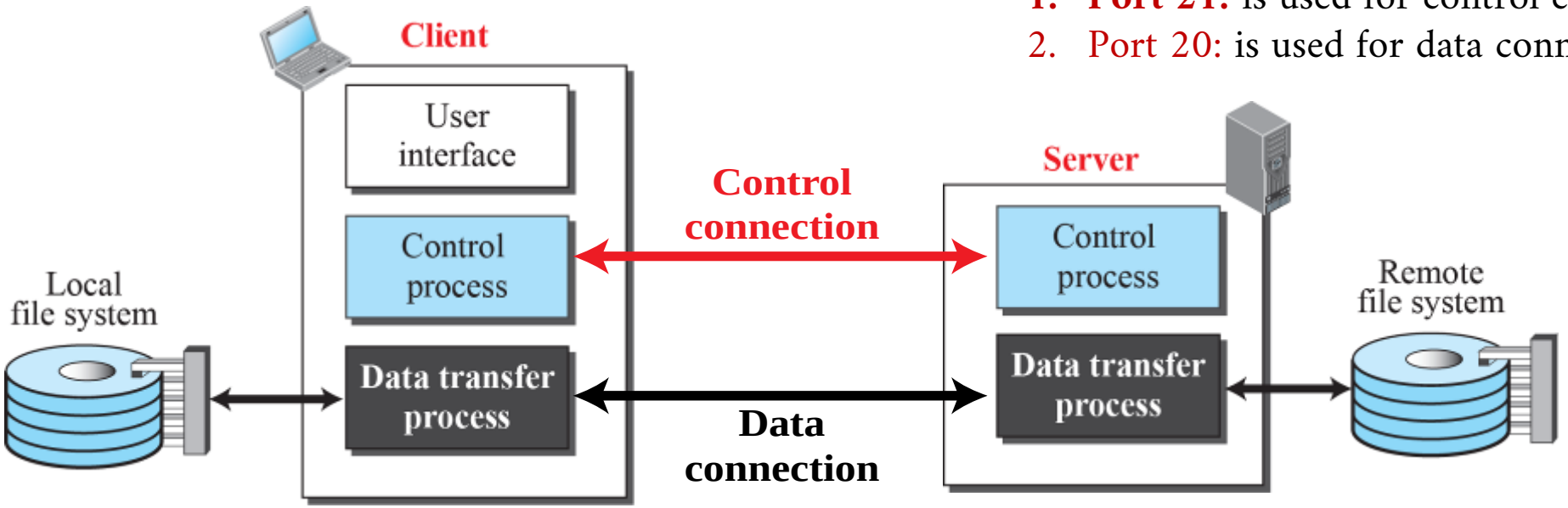
- It is standard protocol provide by TCP/IP to copy a file from one host to another.
- Although HTTP can be used to transfer the file; FTP is better choice for large file or file with different formats.
- FTP is not secure as data transfer takes place in the form of plain text, which is insecure.
- Although, FTP requires a password, the password is sent in plaintext (unencrypted), which means it can be intercepted.
- For security, one can add Secure Socket Layer (SSL) between the FTP application layer and TCP layer. Here this is called **SSL-FTP**.
- FTP can transfer → ASCII file, EBCDIC file, image file, etc.

Basic Model of FTP

- The client has three components:
  1. User interface
  2. Client control process→ remains connected for entire FTP session
  3. Client Data transfer process→ it is opened and then closed for each file transfer activity
- The server has two components:
  1. Server control process
  2. Server Data transfer process

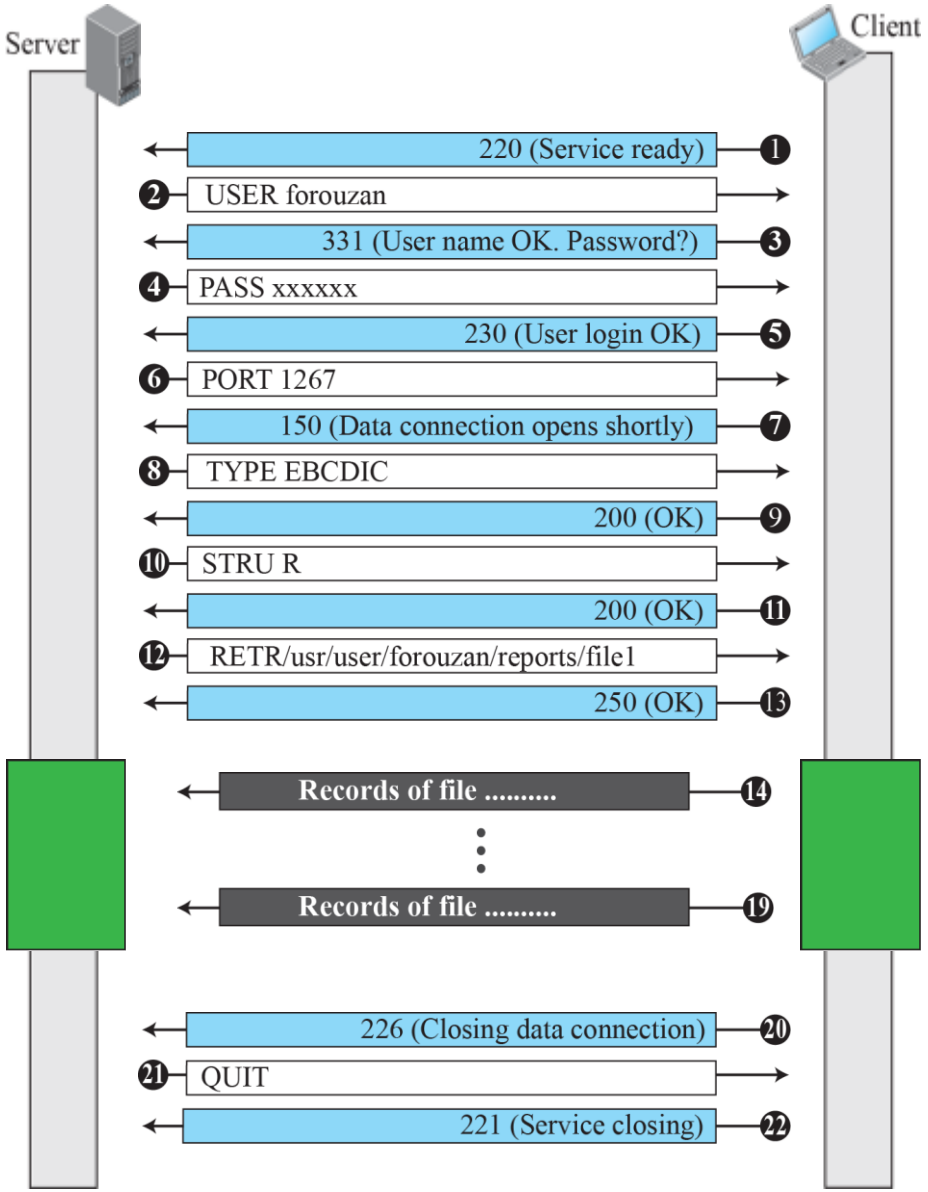
Note: FTP uses two well-known ports of TCP,

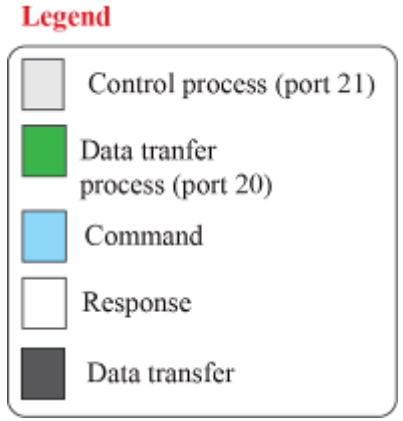
1. **Port 21:** is used for control connection
2. Port 20: is used for data connection



An example of using FTP for retrieving a file (server to client).

- *# of files to be transferred:* 1
- *Control connection:* remains open all the time
- *Data connection:* is opened and closed repeatedly.
- *File transfer:* We assume the file is transferred in six sections.
- After all records have been transferred, the server control process announces that the file transfer is done.
- Since the client control process has no file to retrieve, it issues the QUIT command, which causes the service connection to be closed.





For various ***commands*** and ***responses*** used in FTP refer Table 2.4 and 2.5 of the text book

Client-Server Application: Email

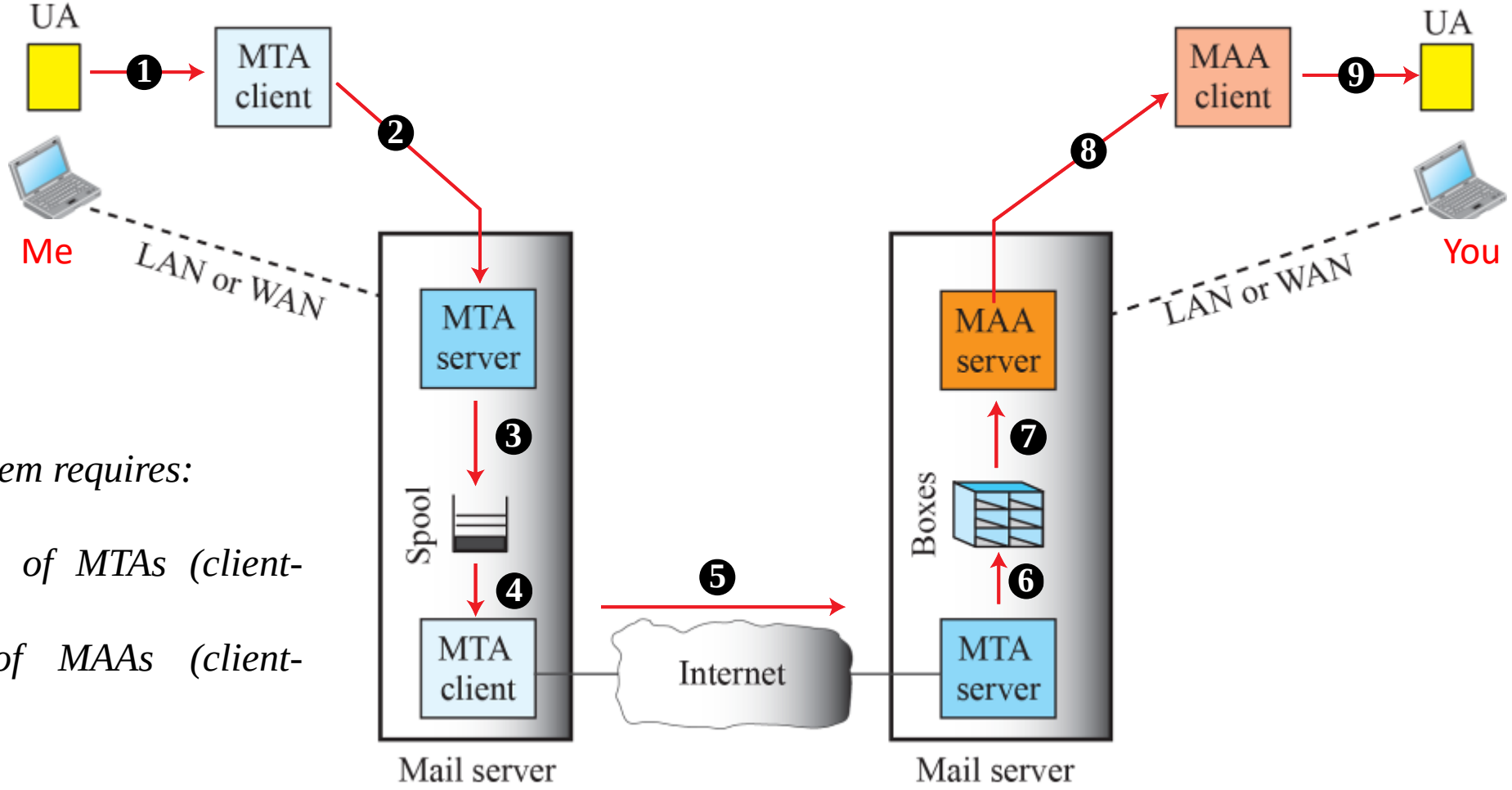
- In HTTP or FTP, the server program is running all the time. When a request come to the server, server provides service to the client→ Typical Client-server paradigm.
- Does e-mail works in the same fashion?
  - For e.g., if I (**client**) send you an email expecting a reply from you (**server**), then is it mandatory from you to reply me back?

Client-Server Application: Email

- In HTTP or FTP, the server program is running all the time. When a request come to the server, server provides service to the client→ Typical Client-server paradigm.
- Does e-mail works in the same fashion?
  - For e.g., if I (**client**) send you an email expecting a reply from you (**server**), then is it mandatory from you to reply me back?
  - **Answer** → is no, it is not mandatory for you to reply me back. i.e., your system should not be running all the time like a typical server.
- So, in email, the idea of client-server programming is implemented by different way:→ using some **intermediate computers (servers)**.
- Hence, the user run only client programs when they want and the intermediate servers apply the client-server paradigm.

Communication Architecture of email application

UA: user agent
MTA: message transfer agent
MAA: message access agent



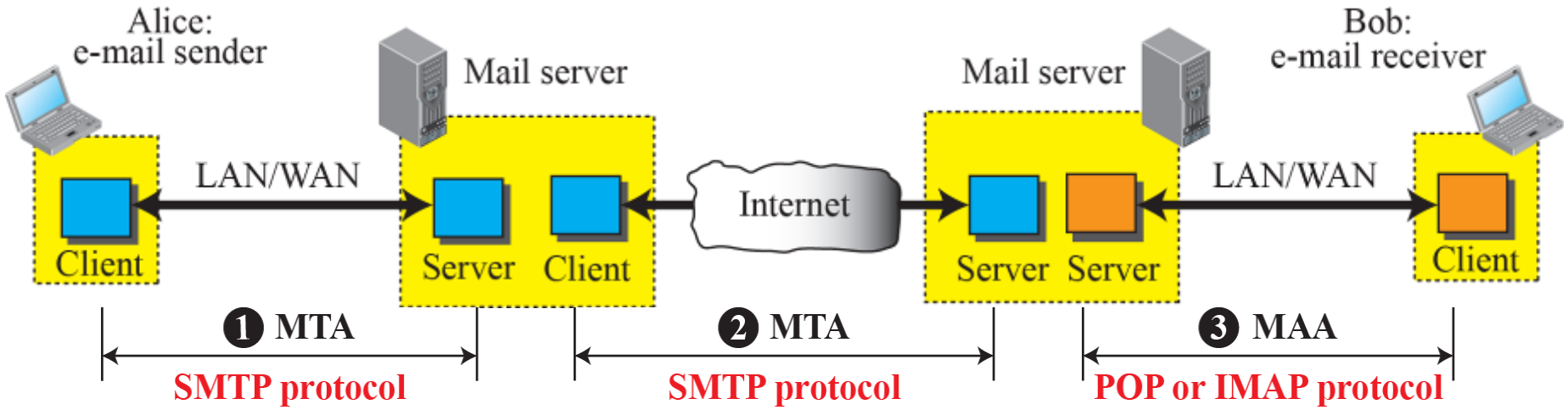
Note:
The email system requires:

1. *Two UAs*
2. *Two pairs of MTAs (client-server)*
3. *A pair of MAAs (client-server)*

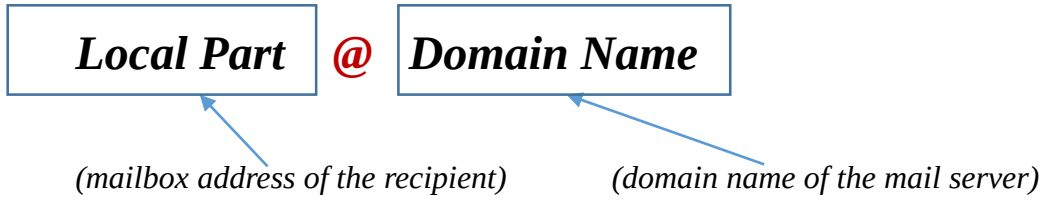
- I have some doubts !!
  1. Can you bypass the mail server and receive my email directly?
  2. Why I need MTA as client server program; whereas you need MAA client-server program?

Protocols used in email

- *SMTP:* used two times,
  1. Between sender and sender's mail server
  2. Between two mail servers
- *POP and IMAP: used only one time*



Format of email address



Protocols used in email

1. *Simple Mail Transfer Protocol (SMTP):*
   - SMTP is message transfer agent.
   - It is a push protocol, which pushes the message from client to server.
   - It simply defines how commands and responses must be sent back and forth.
   - There are several standard commands and responses→ refer textbook, page no 67 and 68 for details.
2. *Post Office Protocol (POP): currently POP3→ 3rd version*
   - *POP is message access agent*
   - It is a pop protocol, which pulls the message from server to the client.
   - The client POP3 software is installed on recipient computer, whereas, the server POP3 software is installed on mail the server.
3. *Internet Mail Access Protocol (IMAP):currently IMAP4→ 4th version*
   - It is similar to POP3, but more powerful and complex than POP3.

POP3	IMAP4
User can not have different mail folder on the mail server	User can cerate, delete, rename mailboxes on the mail server
Does not allow user to partially check the content of the mail	Allows user to **check email header**, **search content of email prior to downloading**, **partial downloading email.**
Cannot create hierarchy of mailboxes in a folder for email storage	

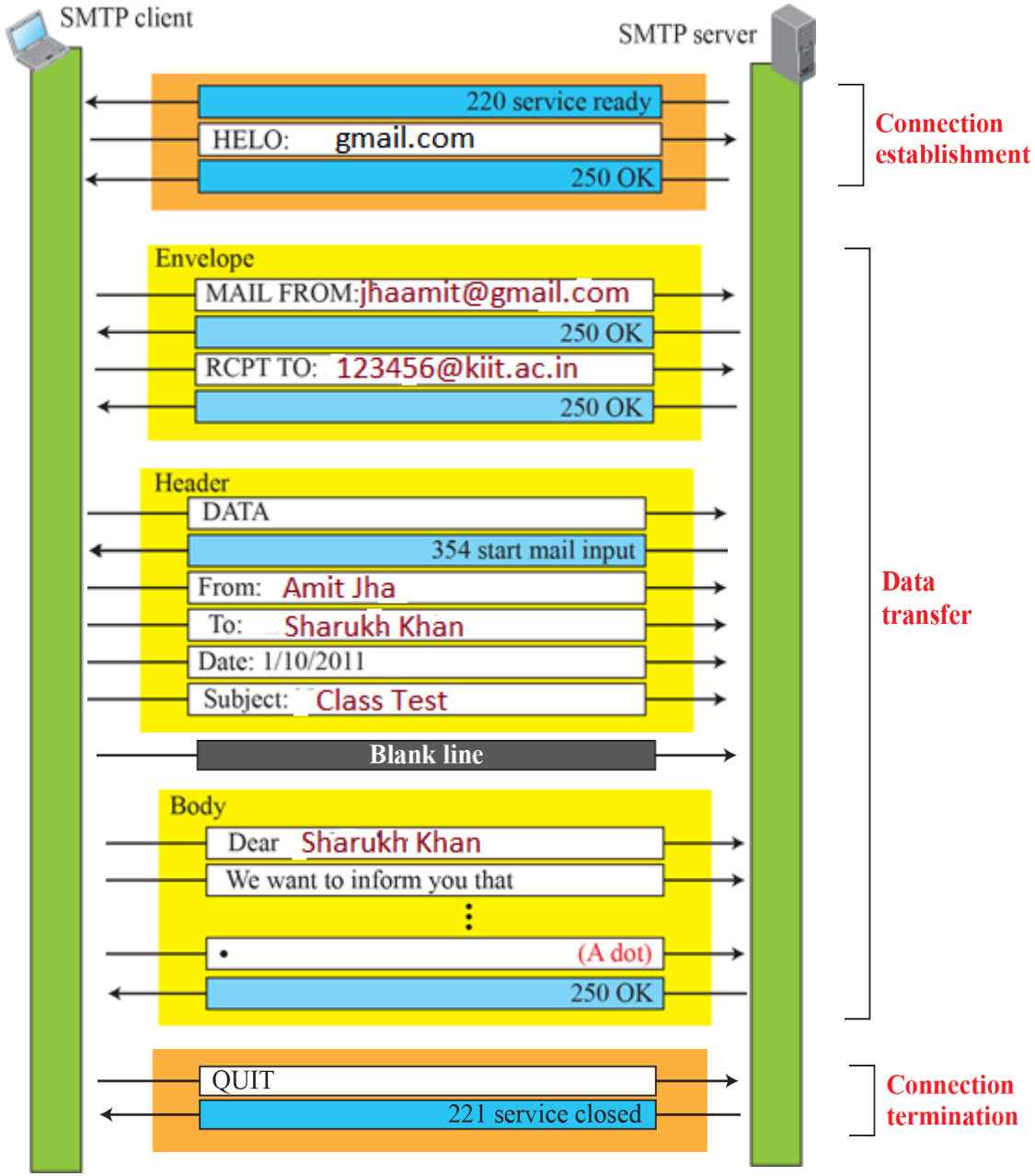
Mail Transfer Phase

- The process of transferring a mail occurs in following three phases:
  1. *Connection Establishment*: first client makes a TCP connection to well-known port 25, then SMTP server starts the connection phase.
     1. The Server sends code 220 (*service ready*).
     2. The client sends **HELO** message to identify itself using a domain name address.
     3. The server responds with code 250 (*request command completed*) or some other code depending upon the situation.
  2. *Message Transfer*: It takes total 8 number of steps for message transfer
     1. Refer the next diagram or page 69 of the text book.
  3. *Connection Termination*: happens in two phases
     1. The Client sends the QUIT command
     2. The Server responds with code 221 or some other appropriate code.

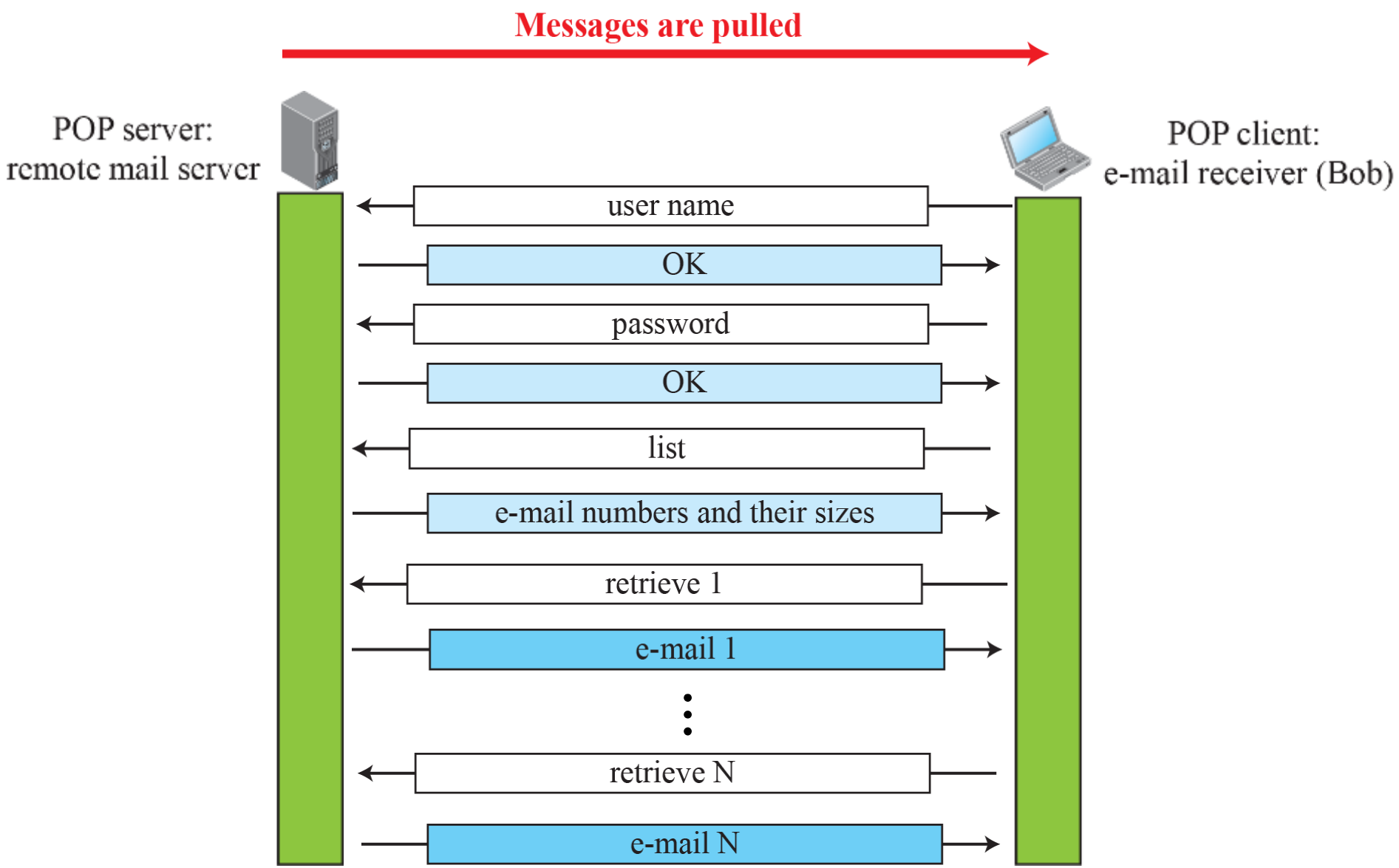
An example of using SMTP for mail transfer.

- Note: This process will be happened two times:
  - 1st, to transfer email sender to local mail server.
  - 2nd , from local mail server to remote mail server

- *Note: different commands and responses can be seen from Table 2.6 and 2.7 of the text book.*



Use of POP3 to receive the mail at the receiver



- I have some doubts !!
  1. Can we send email in language other than English?
  2. Can we send binary files, audio files, or video files using email?

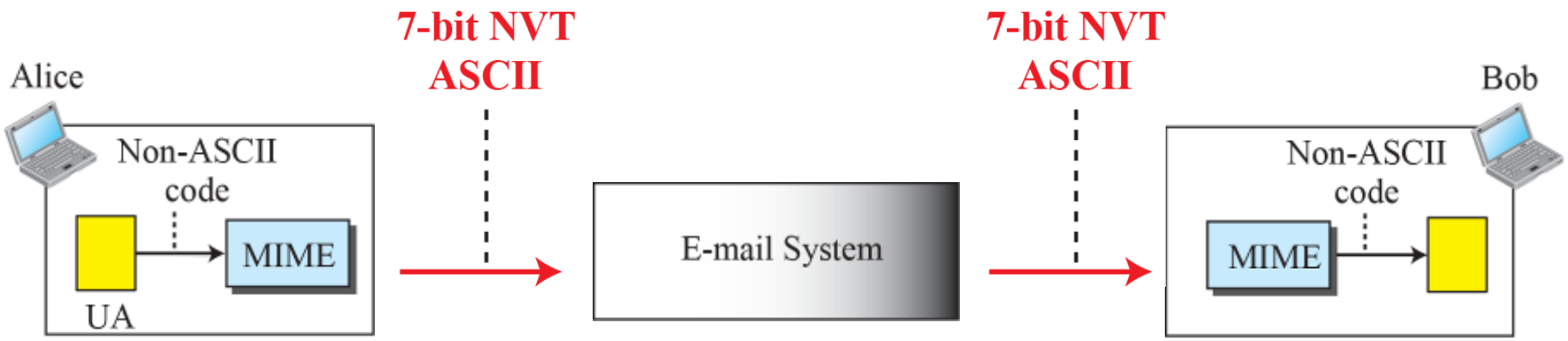
- I have some doubts !!
  1. Can we send email in language other than English?→ No
  2. Can we send binary files, audio files, or video files using email?→ No

Answer: This is because, email can send message only in NVT 7-bit ASCII format.

So, to achieve the above target, we need a supplementary protocol.

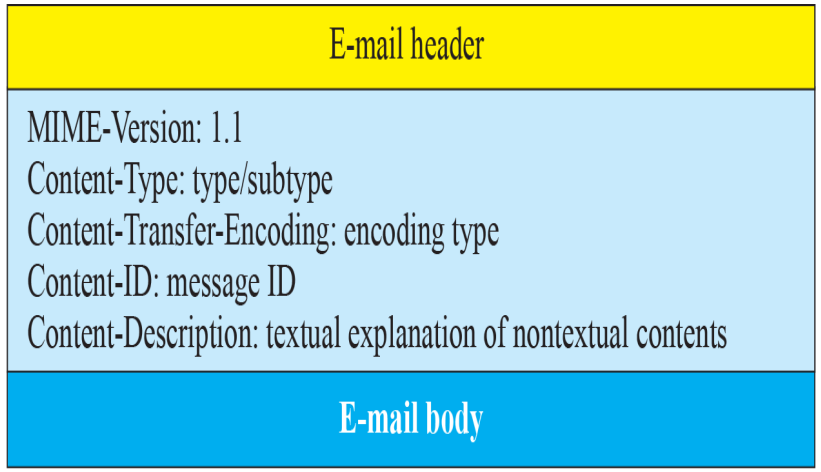
Multipurpose Internet Mail Extension (MIME): it is supplementary protocol that allows non-ASCII data to be sent through e-mail.

- MIME transforms **non-ASCII** data at the sender site to **NVT ASCII** data and delivers it to the client MTA to be sent through the Internet.
- The message at the receiving site is transformed back to the original data.



- **MIME Header:** It defines five headers, as shown below.

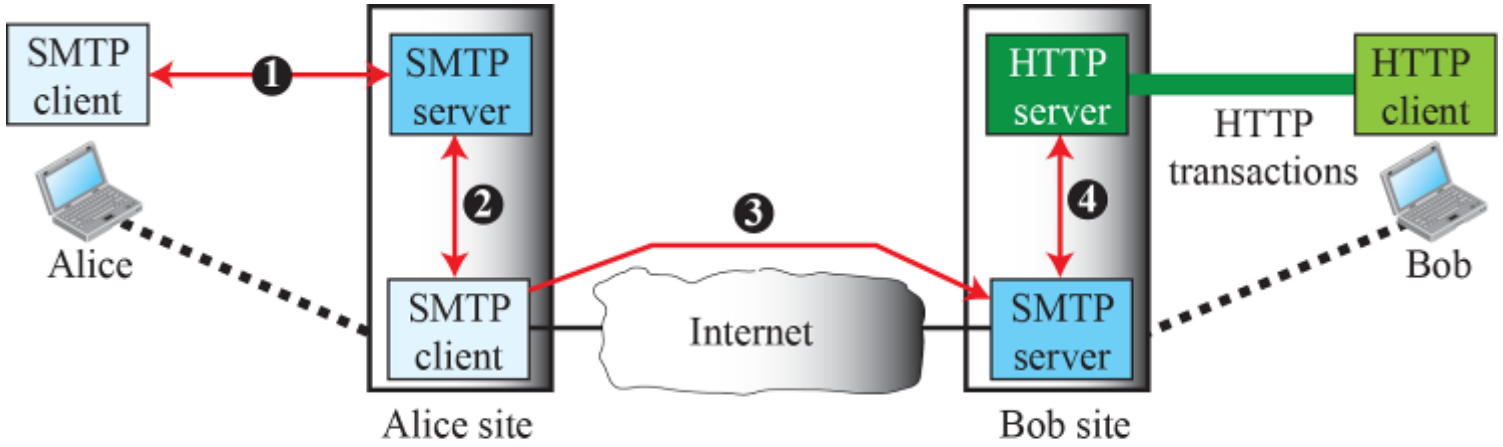
MIME headers



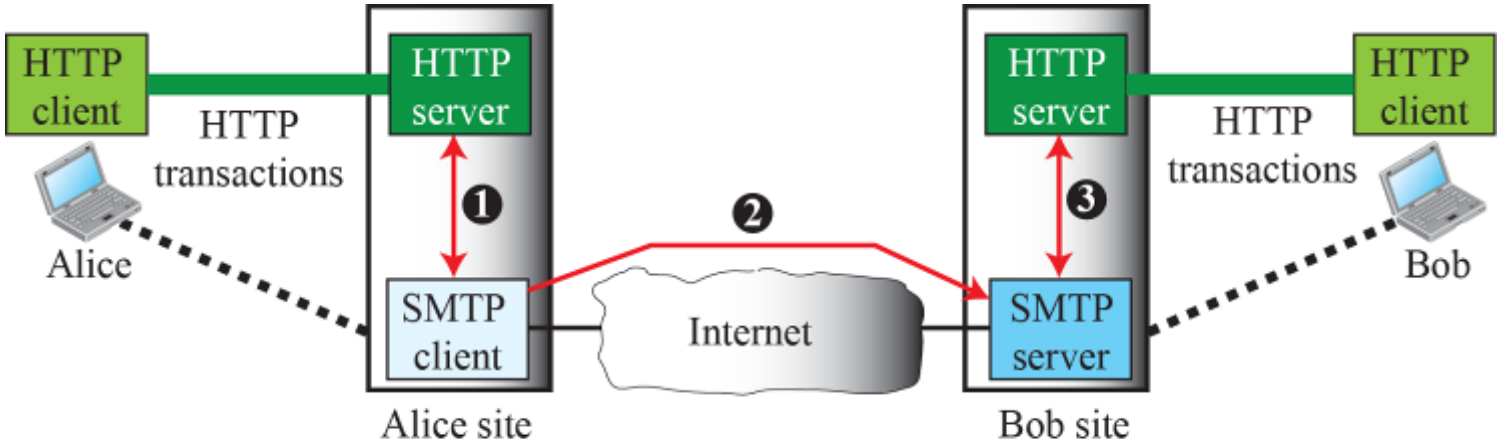
Description

Version	Current version is 1.1
Content Type/subtype	Text, multipart, message, image, video, audio, application
Content-Transfer-Encoding	7bit→ NVT ASCII 8-bit→ Non ASCII Binary, Base64, Quoted-printable
Content ID	To uniquely identify whole message in multi message environment
Content Description	Defines whether the body is image, , audio or video

Web-Based Email: Case-I and Case-II



Case 1: Only receiver uses HTTP

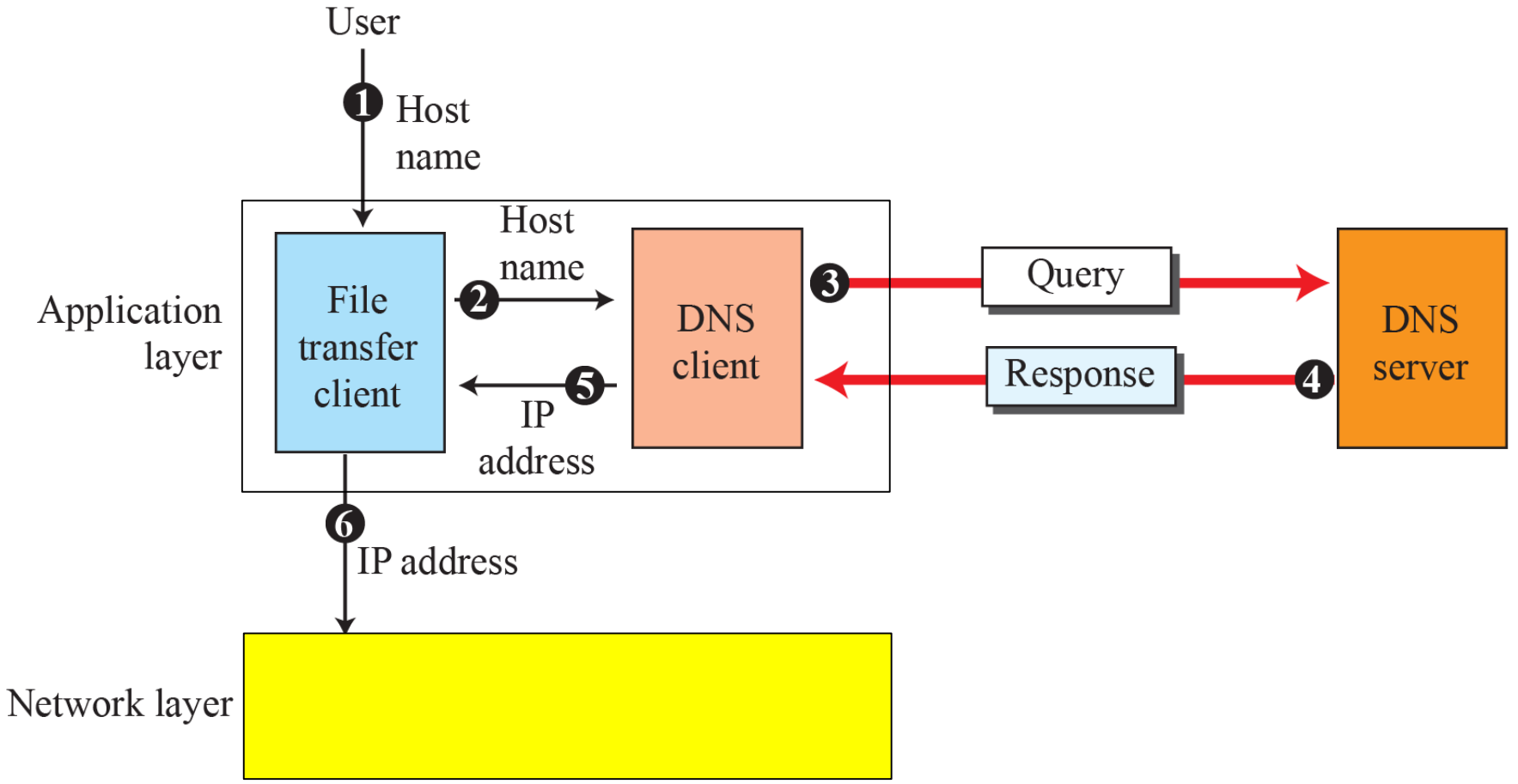


Case 2: Both sender and receiver use HTTP

Client-Server Application: DNS

- **Motivation:** To identify an entity, TCP/IP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name. It is done by DNS.

Operation of DNS Server



Domain Name Space

- **Name Space:** Like IP Address is unique, each machines has unique name from the available name space for unambiguous operation.

Classification of Name Space:

1. Flat name space
2. Hierarchical name space

Flat name Space: A name in this is a sequence of characters without structures. So, it must be centrally controlled to avoid the duplication of ambiguity.

Hierarchical name Space: It is in the form of structures. Name is consist of many parts, each having significance. So, it is not centrally controlled to avoid the duplication of ambiguity, thus decentralized from the administrative point-of-view.

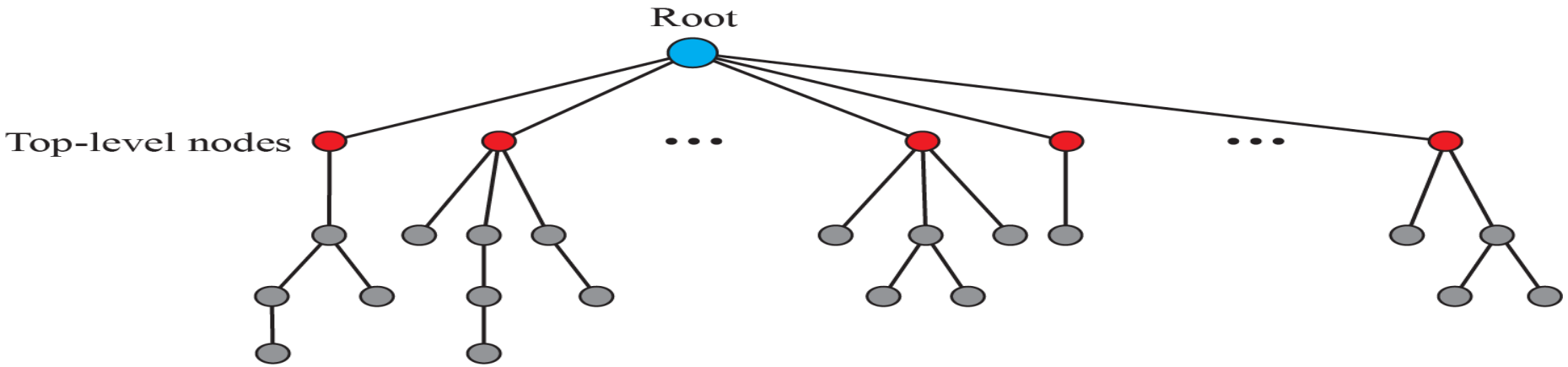
- Think Now !
  - Which structure we follow for domain name space?
    - **Flat** or **hierarchical method** ?

Answer: we use hierarchical domain name space.

- Where, a name is made of several parts and each has some significance.

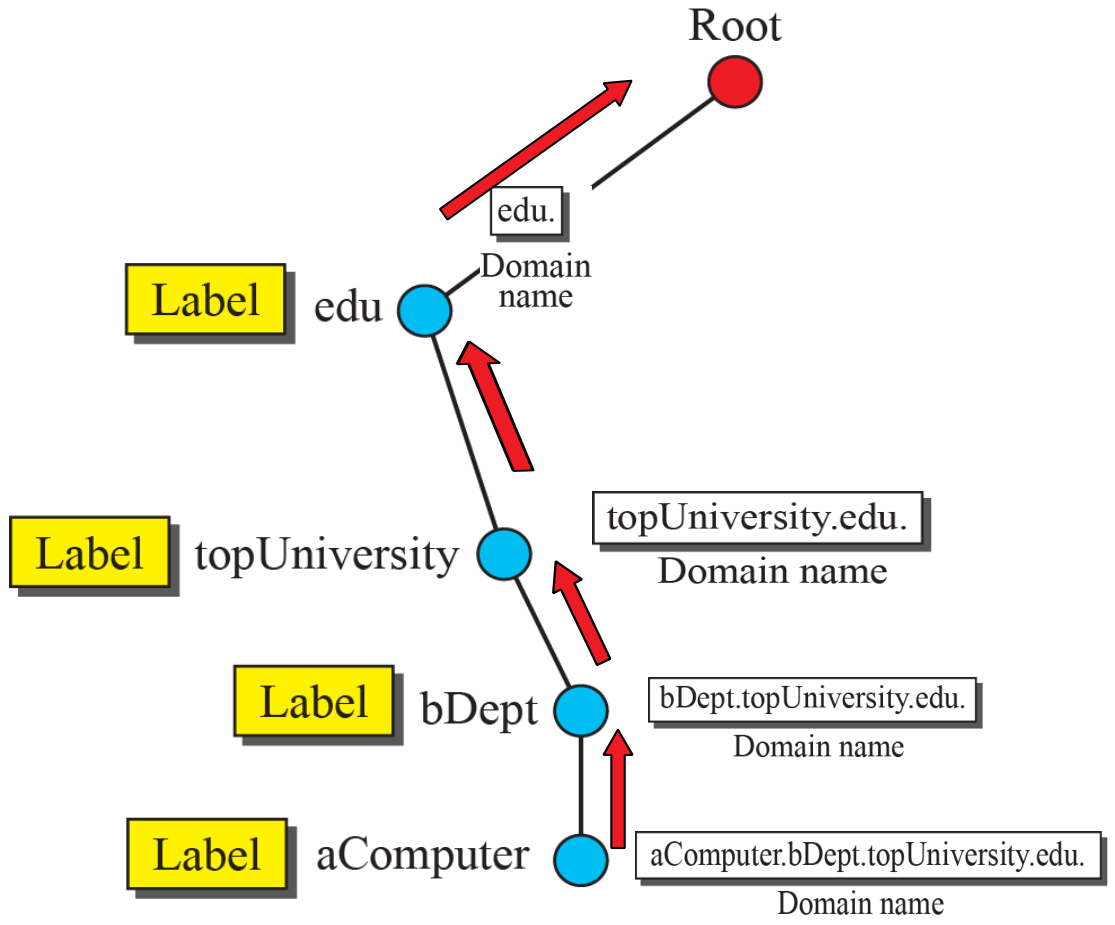
Domain Name space: To have hierarchical name space, domain name space was designed.

- In this design, names are defined in inverted-tree structure with the root at the top.
- The maximum possible levels can be up to 128; from level 0→ ***root*** to level 127 as shown figure below.



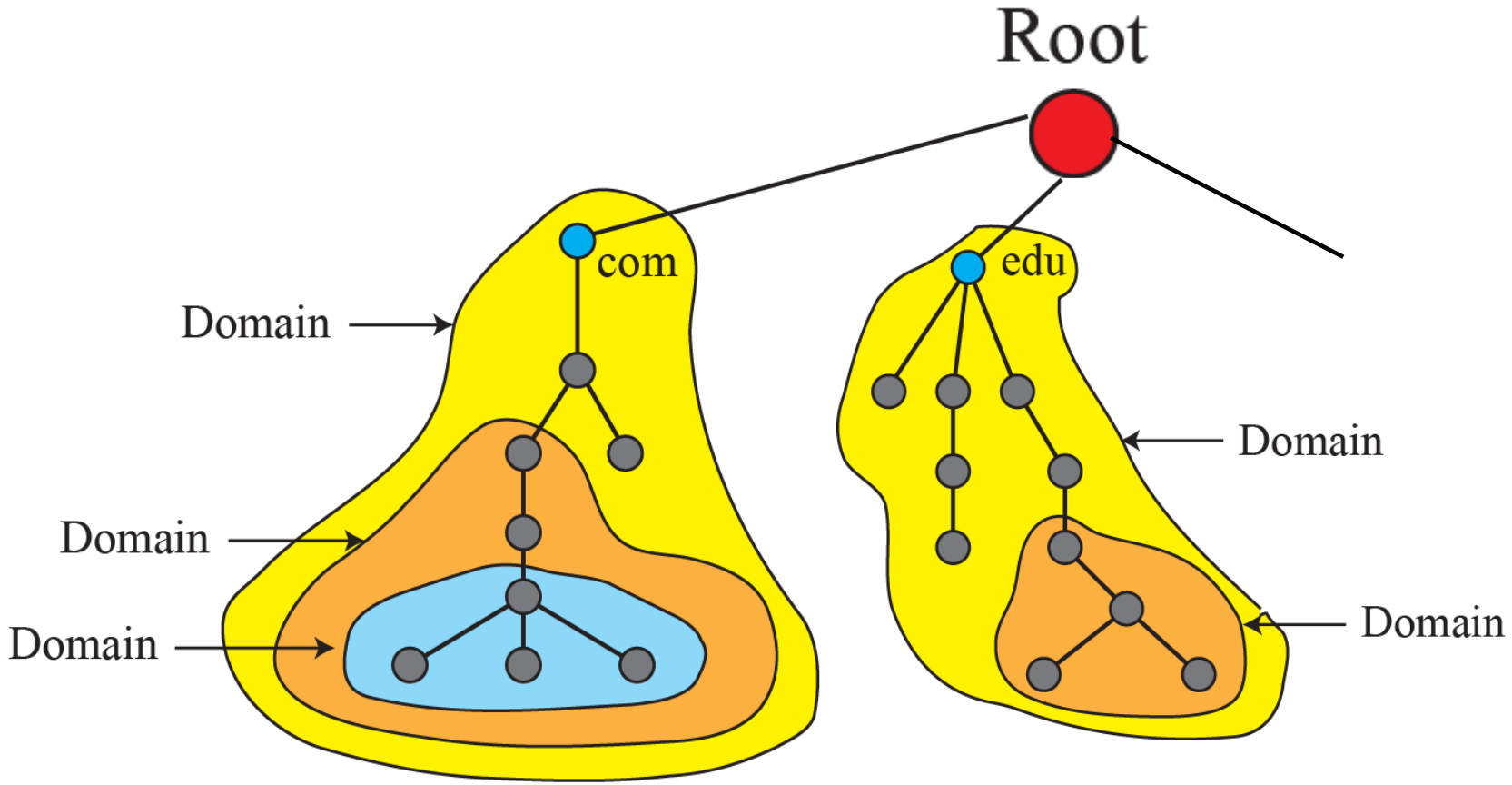
DNS: Domain Names and Labels

- **Label:**
  - Each node in the tree has a **label**; which is a string with a maximum of 63 characters.
  - The root label is null string (empty string).
  - The DNS requires the children of a node should have different labels.
- **Domain Name**:
  - Each node in a tree has a domain name.
  - A full domain name is a sequence of labels separated by dots (.).
  - The domain names are always read from the node up to the root.
- **Fully Qualified Domain Name (FQDN)**: is ended with null string, i.e., dot **(.)**. Otherwise, it is called **PQDN**



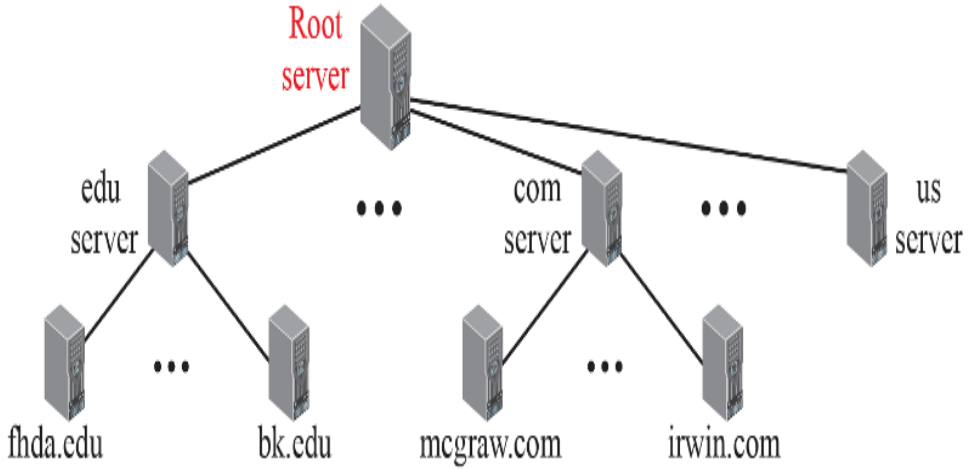
Domain: A **domain** is a subtree of the domain name space.

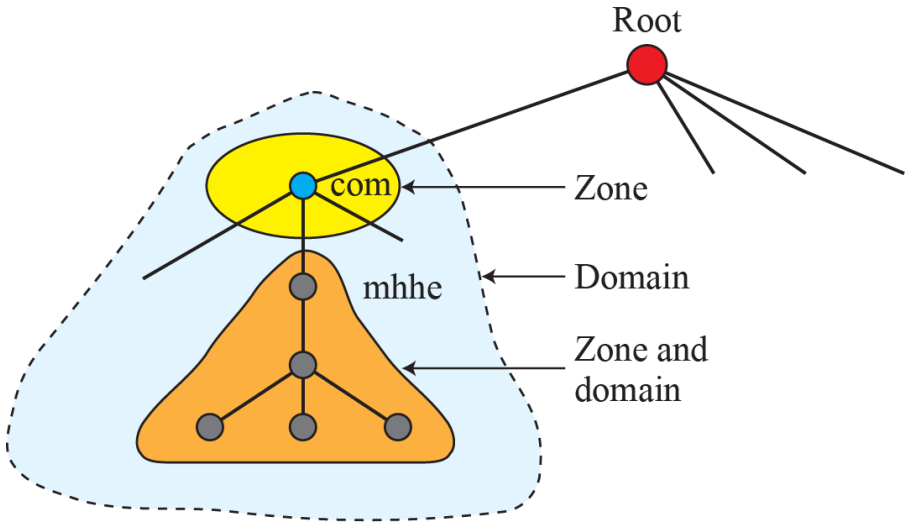
- **Name of the domain** is the name of the node at the top of the tree.



Distribution of Name Space: Name space is distributed in hierarchy manner. The name space is distributed among many computers called **DNS servers**.
DNS allows to be divided further into smaller domains (subdomains).

Zone: Since the complete domain name hierarchy cannot be stored on a single server, it is divided among many servers. What a server is responsible for or has authority over is called a **zone**.





- **Root Server:** It is a server whose zone consists of the whole tree. The root server does not store any information about domain but delegates its authority to other servers, keeping references to those servers.

DNS defines two types of servers:

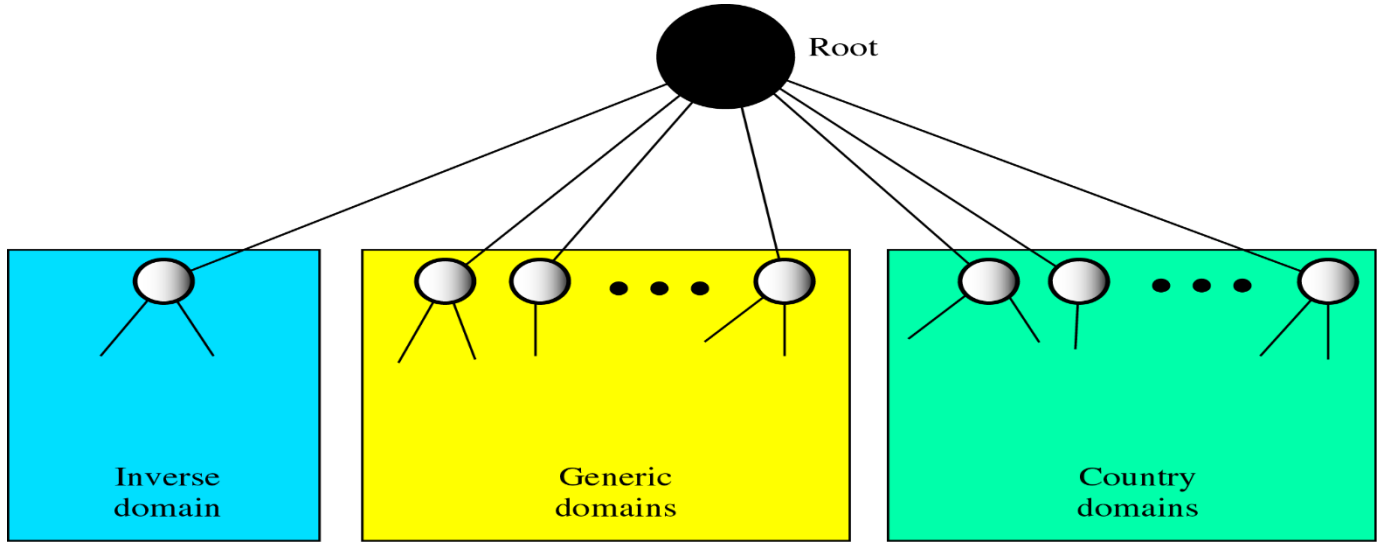
1. **Primary Server:** It is server that stores a file about the zone for which it has an authority. It is responsible for creating, maintaining, and updating the zone file. It stores the zone file on local disk.
2. **Secondary Server:** It is a server that transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. It neither creates nor updates the zone files. If update is required, it must be done by the primary server, which sends the updated version to secondary.

Note: *A primary server loads all information from the disk files; the secondary server loads all information from the primary server.*

DNS in the Internet

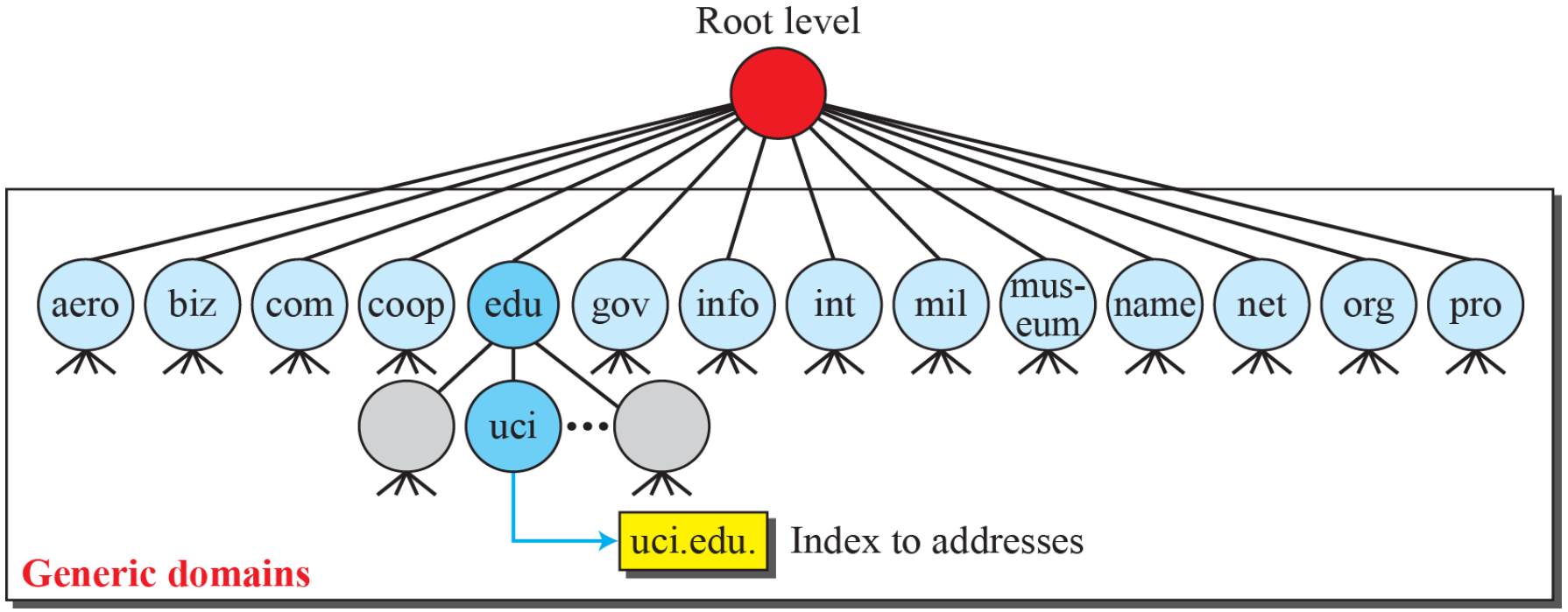
- DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) was originally into three different sections:

1. Generic domains
2. Country domains
3. Inverse domains.



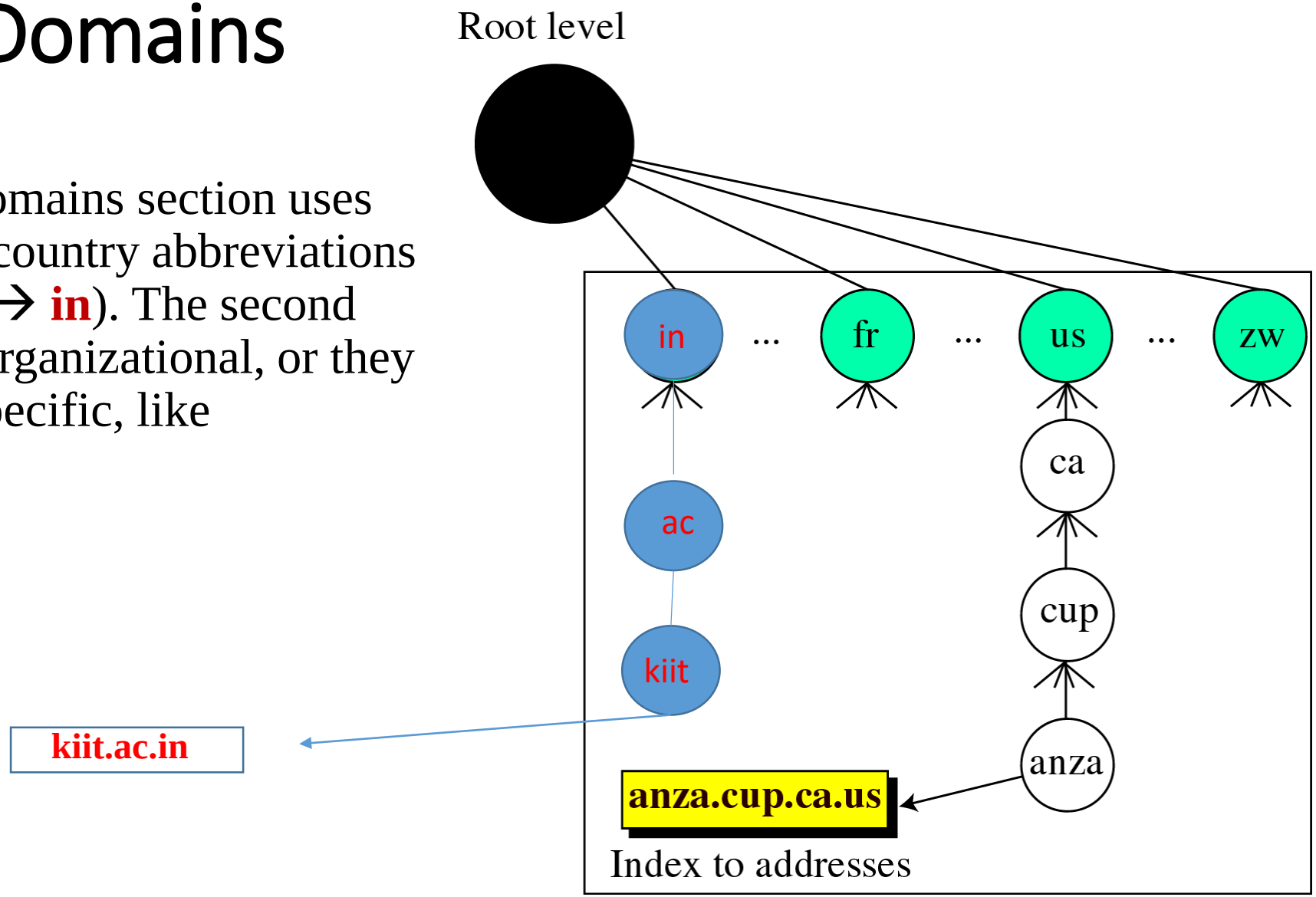
Generic Domains

- The generic domains define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database.



Country Domains

- The country domains section uses two-character country abbreviations (for e.g., India→ **in**). The second labels can be organizational, or they can be more specific, like academic→ **ac**



- Do we need to study 3rd domain name?
  → Inverse domain name

Answer→ NO,

Explanation:

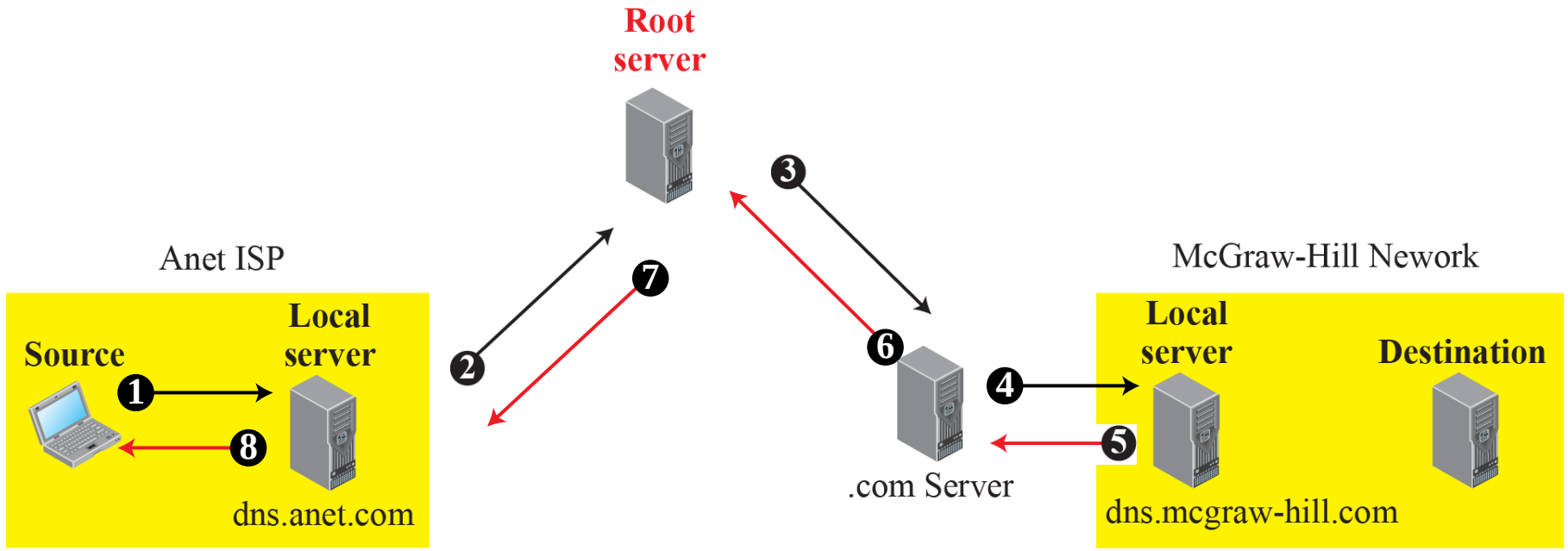
However, due to rapid growth of the Internet, it becomes extremely difficult to keep track of the inverse domains, which could be used to find out name of the host when IP address is known.

DNS Resolution:

- **Resolution:** Mapping a name to an address is called ***name-address resolution***.
- **Resolver:** A host that needs to map an address to a name or a name to an address calls a DNS client known as **resolver**.
- A resolution can be of two types:

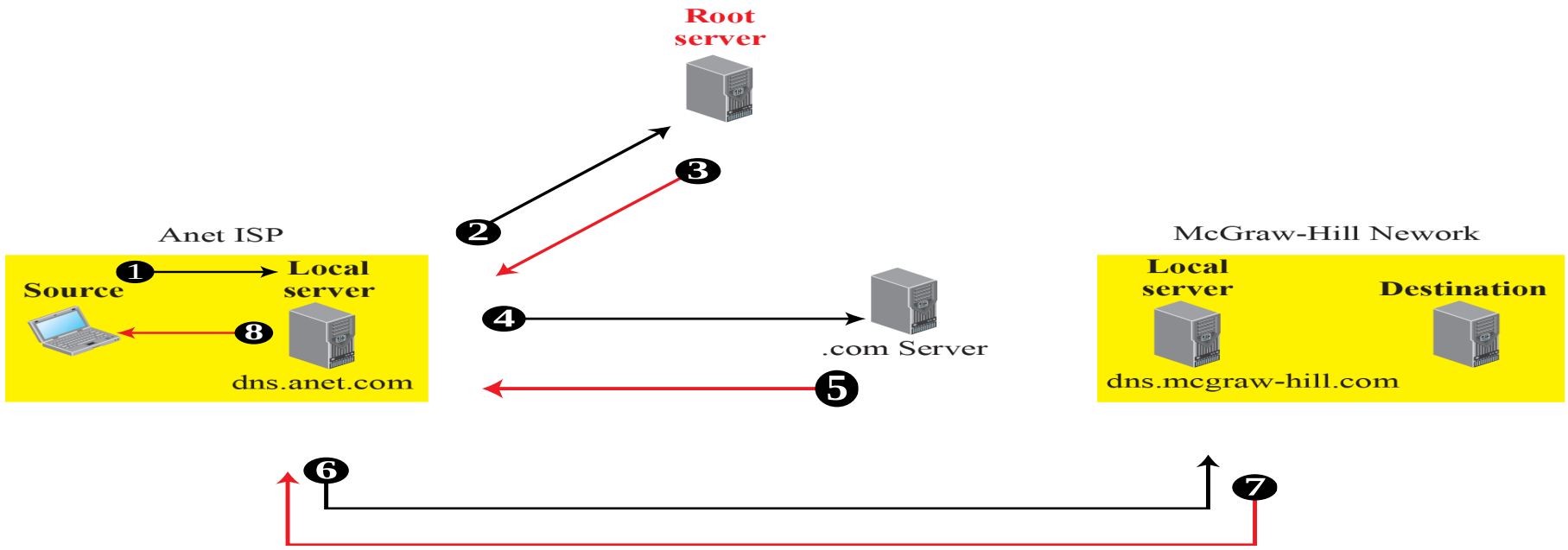
1. **Recursive Resolution:** In this, each server that does not know the mapping sends the query to
   →the root DNS server as root DNS server does not keep the mapping information
   →It sends the query to this top-level-domain server.
   → process goes on as shown in the diagram, until mapping is done.
2. **Iterative Resolution:** In this, each server that does not know the mapping sends the IP address of the next server back to one that requested it.

Architecture of DNS: Using Recursive Resolution



Source: some.anet.com
Destination: engineering.mcgraw-hill.com

Architecture of DNS: Using Iterative Resolution



Source: some.anet.com
Destination: engineering.mcgraw-hill.com

HYU……..

- Can we use *caching* for the domain name as well?

Resource Records

- The zone information associated with a server is implemented as a set of *resource records*. A *resource record* is a 5-tuple structure as shown below→ **Domain Name, Type, Class, TTL, Value**

- **Domain Name:** It identifies the resource records.
- **Type:** defines how the value should be interpreted.
- **Class:** defines the type of network. For Internet, it is **IN.**
- **TTL:** defines the number of seconds for which the information is valid.
- **Value:** The information kept about the domain name.

Type	*Interpretation of value*
A	A 32-bit IPv4 address (see Chapter 4)
NS	Identifies the authoritative servers for a zone
CNAME	Defines an alias for the official name of a host
SOA	Marks the beginning of a zone
MX	Redirects mail to a mail server
AAAA	An IPv6 address (see Chapter 4)

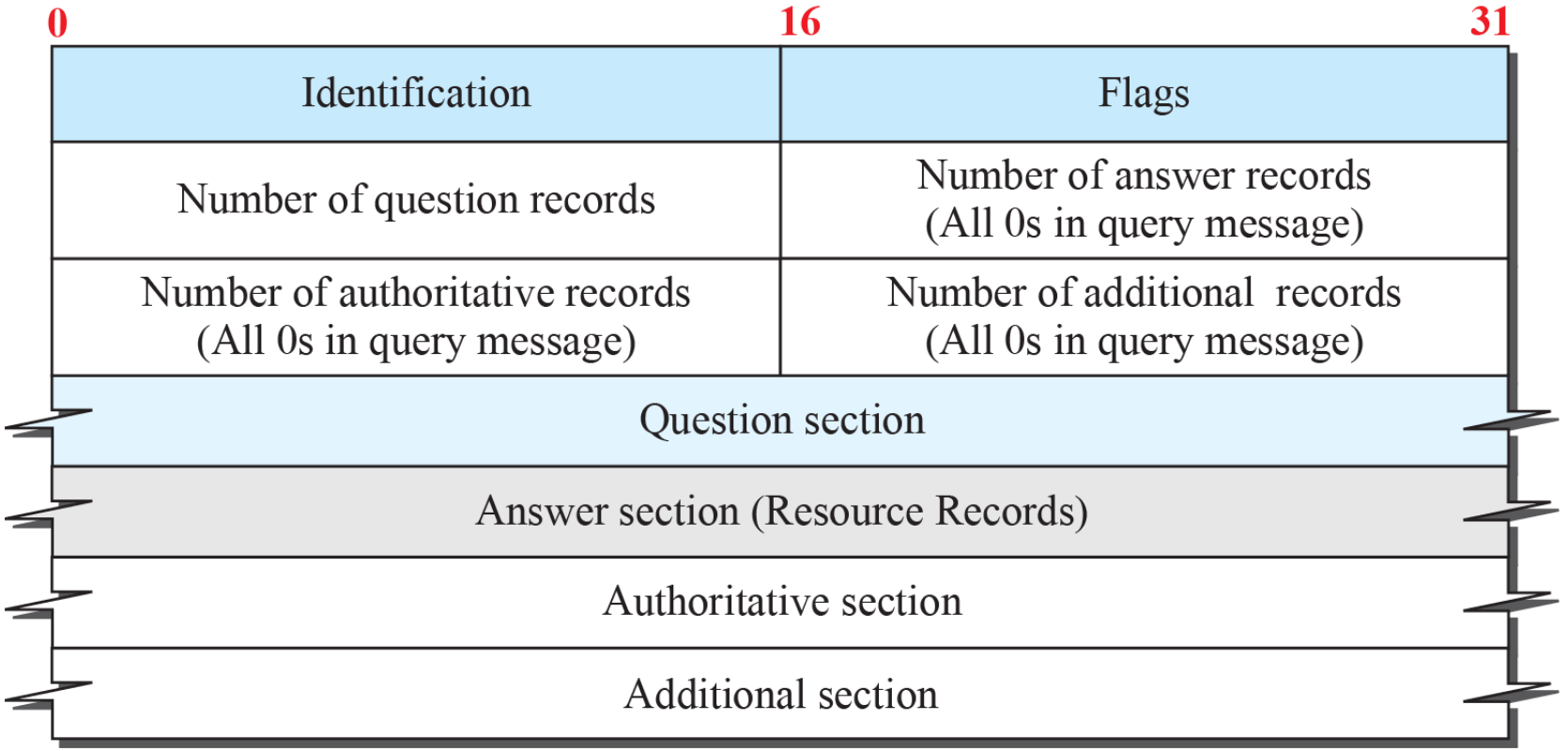
Various Types of Records Maintained by DNS

- **Host A:** It is used to translate human friendly domain names such as "www.example.com" into IP-addresses such as 23.211.43.53 (machine friendly numbers). For **IPv4 it is A**, whereas, for IPv6, it **is AAAA** record.
- **NS:** NS-records identify the **DNS servers responsible (authoritative) for a zone**. An NS-record identifies the name of a DNS server - not the IP-address.
- **MX:** MX-records are used to specify the e-mail server(s) responsible for a domain name. It redirects mail to mail server.
  - If a domain name is handled by multiple e-mail servers (for backup/redundancy), a separate MX-record is used for each e-mail server, and the preference numbers then determine in which order (lower numbers first) these servers should be used by other e-mail servers.
  - If a domain name is handled by a single e-mail server, only one MX-record is needed and the preference number does not matter.
  - When sending an e-mail to "**user@example.com**", your e-mail server must first look up any MX-records for "**example.com**" to see which e-mail servers handles incoming e-mail for "example.com".
- **CNAME:**CNAME-records are **domain name aliases**.
  - Computers on the Internet often performs multiple roles such as web-server, ftp-server, chat-server etc.
  - To mask this, CNAME-records can be used to give a single computer multiple names (aliases).
  - For example, the computer "computer1.xyz.com" may be both a web-server and an ftp-server, so two CNAME-records are defined:
    - **"www.xyz.com" = "computer1.xyz.com"** and **"ftp.xyz.com" = "computer1.xyz.com"**.
  - The most common use of the CNAME-record type is to provide access to a web-server using both the standard **"www.domain.com"** and "**domain.com**" (with and without the **www** prefix).
    - This is usually done by creating an A-record for the short name (without www), and a CNAME-record for the www name pointing to the short name.

Which service DNS uses of Transport Layer?

- It can use either UPD or TCP.
- **UDP**
  - It is used when the size of the response packet is less than 512 bytes because most of the UDP packets has limitation **of 512** bytes packet size.
- **TCP**
  - If size of the response packet is more than 512 bytes, it uses TCP service of the Transport Layer.
- **Note:** *In both the cases (UDP and TCP), the well-known port used by the server is* ***53***.

DNS Message Format



Note:

The query message contains only the question section.
The response message includes the question section,
the answer section, and possibly two other sections.

P2P Networks

- **Peer:** In this, internet users which are ready to share their information becomes peers.
- **P2P Network:** Different peers are connected over the Internet to create a network.
- **How it works:** When a peer in the network has a file to share, it makes it available to the rest of the peers. An interested peer can connect itself to the computer where the file is stored and download it. After a peer downloads a file, it can make it available for other peers to download. As more peers join and download that file, more copies of the file become available to the group.
- Think..
  - Since, the list peers may grow and shrink, then how the P2P paradigm keeps tracks of the loyal peers and the location of the file?

To answer the last question, the P2P network is divided into two categories: **Centralized** and **Decentralized**

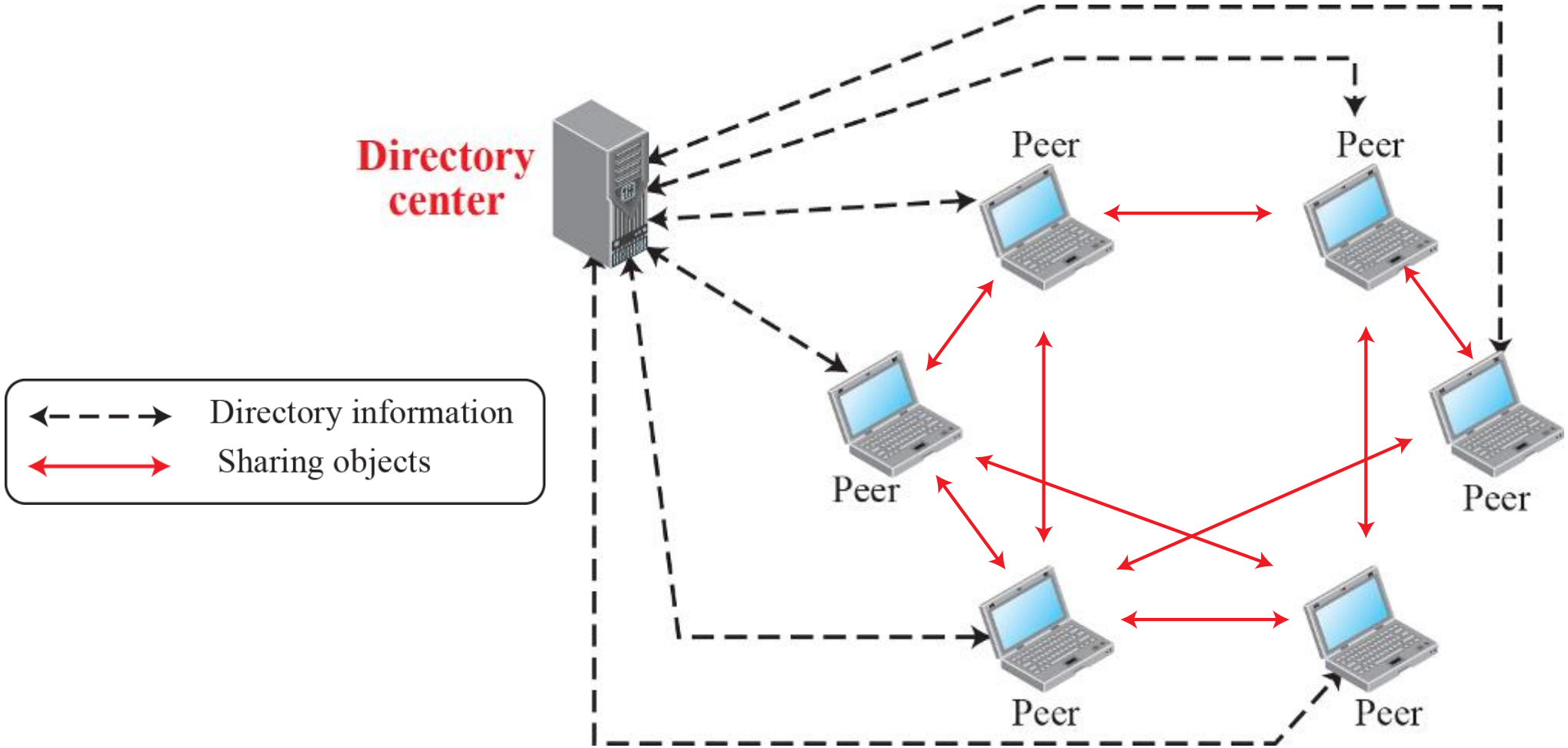
- **Centralized P2P Networks:**
  - In this a directory system is used which works on **client-server paradigm**.
  - It has list of all the peers and the information about (files) what they can offer.
  - Since, listing of the peers and the file information is done by directory system which uses a client-server paradigm, however, the files download and upload is achieved using peer-to-peer paradigm, this architecture is also referred to as **hybrid P2P networks**.

  - **How one can register as a peer?**
    - In this, a peer first register itself with the directory system (central server) by providing its IP address and the files which it has to share.

  - **How file is downloaded?**
    - A peer, looking for a particular file, sends a query to a central server. The server searches its directory and responds with the IP address of the node (peer) that have a copy of the file. The peer contacts on of the node and downloads the file.

Note: *The directory is constantly updated as nodes join or leave the network.*

Working of centralized P2P Networks

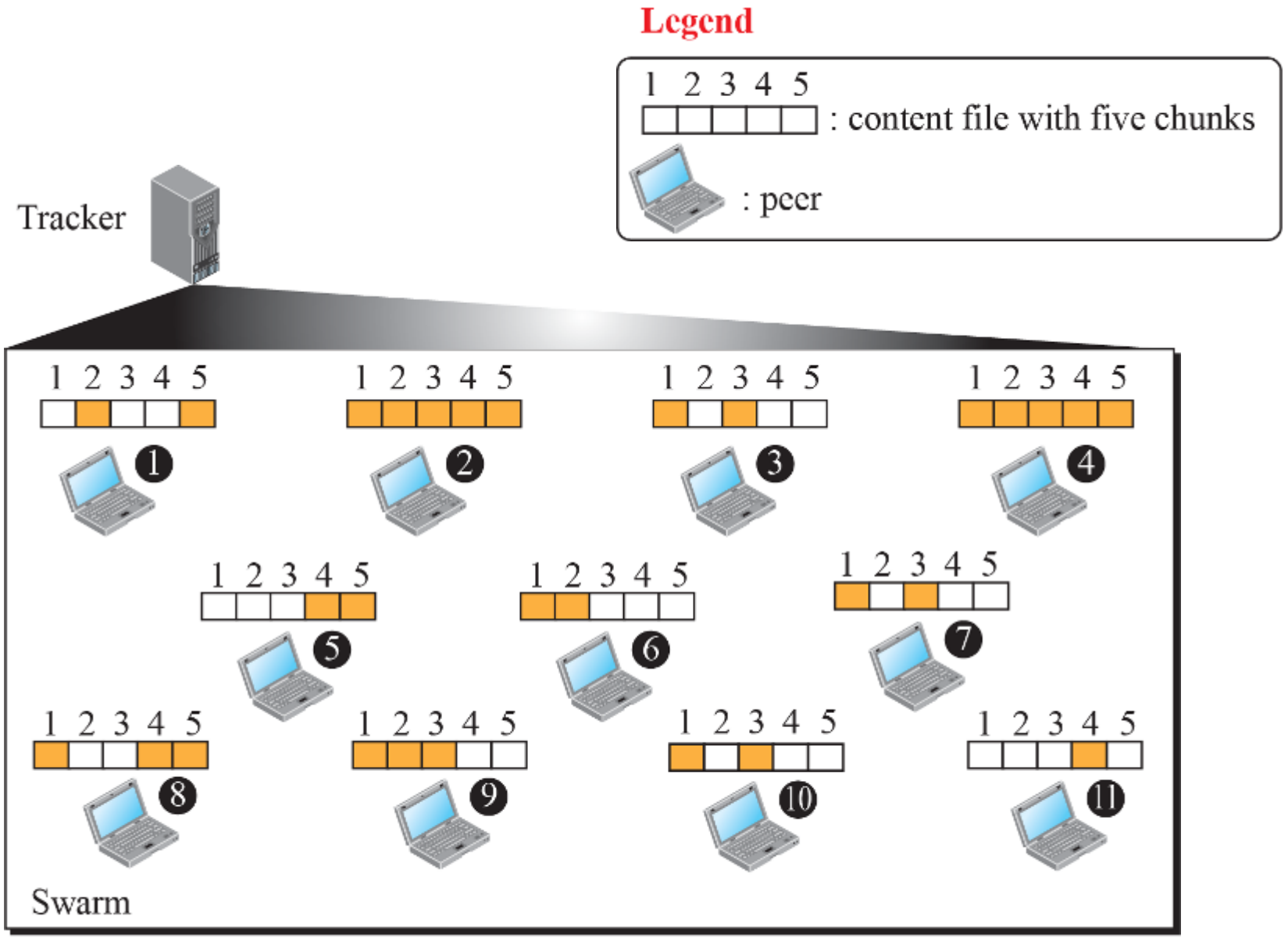


- **Decentralized P2P Networks:**
  - In this, there is no need of a centralized directory system.
  - Here, peers arrange themselves into an overlay network, which is a logical network made on top of the physical network.
  - Depending upon how the nodes in the overlay network are linked, a decentralized P2P network is classified as either unstructured or structured.
- **Unstructured P2P Networks:** In this, nodes are linked randomly. A search in this network is not very efficient because a query to find a file must be flooded through the network, which produces significant traffic and still the query may not be resolved.
- **Structured P2P Networks:** In this, nodes are linked through a predefined set of rules. Thus, a search in this network is very efficient because a query to find a file must not be flooded through the network, which produces less traffic to resolve the query. The most common technique used for created structured networks is using Distributed Hash Table (DHT).

P2P Application: BitTorrent

- In original BitTorrent, there is another entity in a torrent, called the tracker, which as the name implies, tracks the operation of the swarm. Swarm is the group of peers in the torrent.
- **Let us assume:**
  - The file is to be shared, is divided into 5 pieces (chunks).
  - Peer 2 and peer 4 already have all the pieces; other peers have some pieces.
  - The pieces that each peer has are shaded.
  - Some peers may leave the torrent; some may join the torrent.
- Think…w.r.t given statement as above. *Explain the process that how does a peer can download the same file?*

- The new peer accesses the BitTorrent server with the name of the content file (or simply file).
- It receives a metafile (called torrent file), that contains the information about the pieces in the content file and address of the tracker that handles that specific torrent.
- The new peer now accesses the tracker and receives the addresses of some peers in the torrent.
- The new peer is now part of the torrent and can download and upload the pieces of the content file.



Note: Peers 2 and 4 are seeds; others are leeches.

Remember……

- *Also, there exist trackerless BitTorrent.*
- Importantly, BitTorrent clients never actually download files from the tracker itself. The tracker participates in the torrent only by keeping track of the BitTorrent clients connected to the swarm, not actually by downloading or uploading data.

- *Note: BitTorrent may be primarily used for piracy at the moment, as its decentralized and peer-to-peer nature are a direct response to efforts to crack down on Napster and other peer-to-peer networks with central points of failure. However, BitTorrent is a tool with legitimate uses in the present — and many other potential uses in the future.*

End of Module-2

You are advised to go through the text book as mentioned in the class for more clarity and details.