Computer Networks

Unit 5

Application layer

Traditional applications - electronic mail (SMTP, POP3, IMAP, MIME) - HTTP - File Transfer protocol - secure shell (SSH) - DNS

* HTTP - PPT-1

and World Wide Web

A. World Wide Web

→ proposed by Tim Berners-Lee at CERN.

→ The WWW is a repository of information in which documents called web pages are distributed all over the world and related documents are linked together.

Distribution - allows for the growth of the web, without overloading servers

Linking - allows one web page to refer another web page stored in another server somewhere else in the world.

→ Linking is done using the concept of hypertext - allows linked document to be retrieved when the link was clicked by the user.

* Architecture of WWW

→ in the form of a distributed client-server service, in which a client using a browser can access a service using a server.

→ The service provided is distributed over many locations called sites.

→ Each site holds one or more web pages. A web page can contain some links to other web pages in the same or other sites.

→ Each web page is a file with a name and address.

* Web Clients and Servers

(1) Web Clients (Browsers)

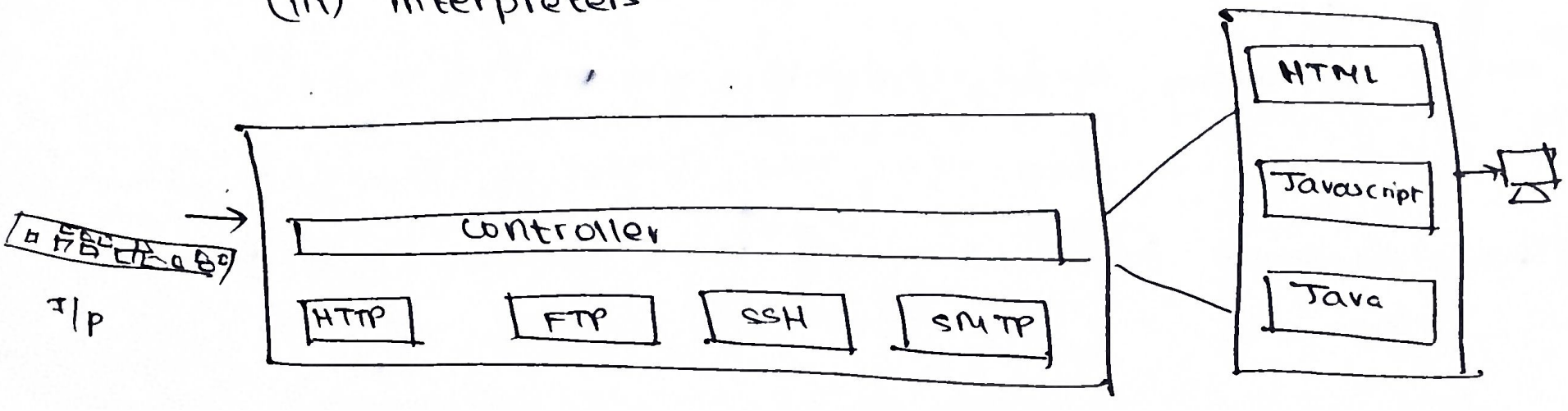
→ A variety of vendors offer commercial browsers that interpret and display a web page.

→ Each browser consists of three parts:

(i) a controller

(ii) client protocols

(iii) interpreters



(i) controller - receives input from the keyboard or mouse and uses the client programs to access the document

→ After the document has been accessed, the controller uses one of the interpreters to display the document on the screen.

(ii) interpreter - can be HTML, Java or Javascript

(iii) client protocols - HTTP, FTP

Examples of browsers - Google Chrome, Internet Explorer, Firefox

(2) Web Servers

→ The web page is stored in the server.

→ Each time a request arrives, the corresponding document is sent to the client.

→ To improve efficiency, servers normally store requested files in a cache in memory - memory is faster to access than a disk.

→ Efficiency of the server can also be increased through multithreading or multiprocessing. Then, the server can answer more than one request at a time.

→ Some web servers are Apache & Microsoft Internet Information Server.

* Uniform Resource Locator (URL)

→ a unique identifier to distinguish it from other web pages.

→ Three identifiers are required : host , port and path , as well as the protocol.

→ The protocol defines what client - server application we want to use.

(i) Protocol - the first identifier

- serves as the abbreviation for the client - server program needed to access the web page.

- can be HTTP , FTP etc.

(ii) Host - The host identifier can be the IP address (in dotted decimal notation) , or more commonly , the DNS is used

(iii) Port - A 16 bit integer - usually predefined for the client - server application

eg. if HTTP is used; the port is 80

→ The port number can also explicitly be given

(iv) Path - identifies the location and name of the file in the underlying operating system. (it lists directories from top to bottom)

→ These 4 pieces together constitute the URL. It can look like:

protocol : // host | path

protocol : // host : port | path

* Web Documents

→ The documents on the WWW can be grouped into 3 categories.

(i) static
(ii) dynamic
(iii) active

A. Static - fixed-content documents that are created and stored in a server

- client can only get a copy of the document
- user cannot change them
- prepared using one of many languages like HTML, XML, XSL, XHTML

B. Dynamic - a dynamic document is created by a web server whenever a browser requests the document.

- when a request arrives, the web server runs an application program / script that creates the dynamic document.

→ The server returns the result of the program or script as a response to the browser that requested the document

→ Because a fresh document is created at each request, the contents of a dynamic document may vary from one request to another (eg. retrieving date and time from the server)

→ To do this, scripting languages like Java Server Pages (JSP), Active Server Pages (ASP), Visual Basic etc. can be used.

c. Active Documents -

→ For many applications, a program or script must be run at the client site

→ for eg. when animated graphics are needed / a program that interacts with the user.

→ program needs to run on the client

→ on request, the browser sends a copy of the document / script - document run on client side.

→ made using Java applets / Javascript

B. Hyper Text Transfer Protocol (HTTP)

→ defines how the client - server programs can be used to retrieve web pages from the web.

→ A HTTP client sends a request - a HTTP server sends a response.

→ The server uses port no 80, the client uses a temporary port number.

→ HTTP uses the services of TCP, which is connection-oriented and reliable.

* Types of HTTP connections

(i) Non-persistent

(ii) Persistent

A. Non-Persistent Connections

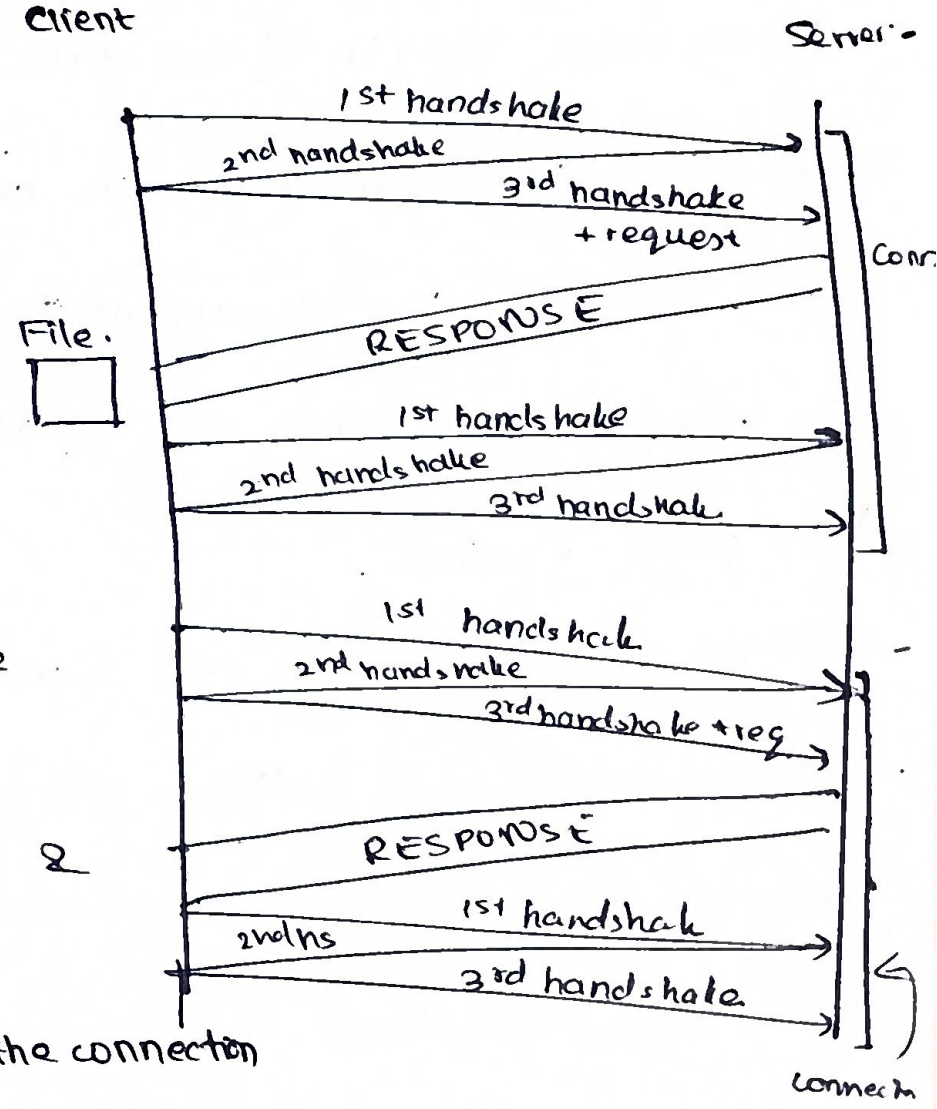
→ one TCP connection must be made for every request / response.

→ Steps : ① Client opens TCP connection & sends a request

② Server sends the response & closes the connection

③ Client reads data until it encounters an end of file marker.

→ In this strategy, if a file contains links to N pictures in N different files, the connection must be opened/closed N+1 times



→ imposes high overhead because the server needs N+1 different buffers each time a connection is opened.

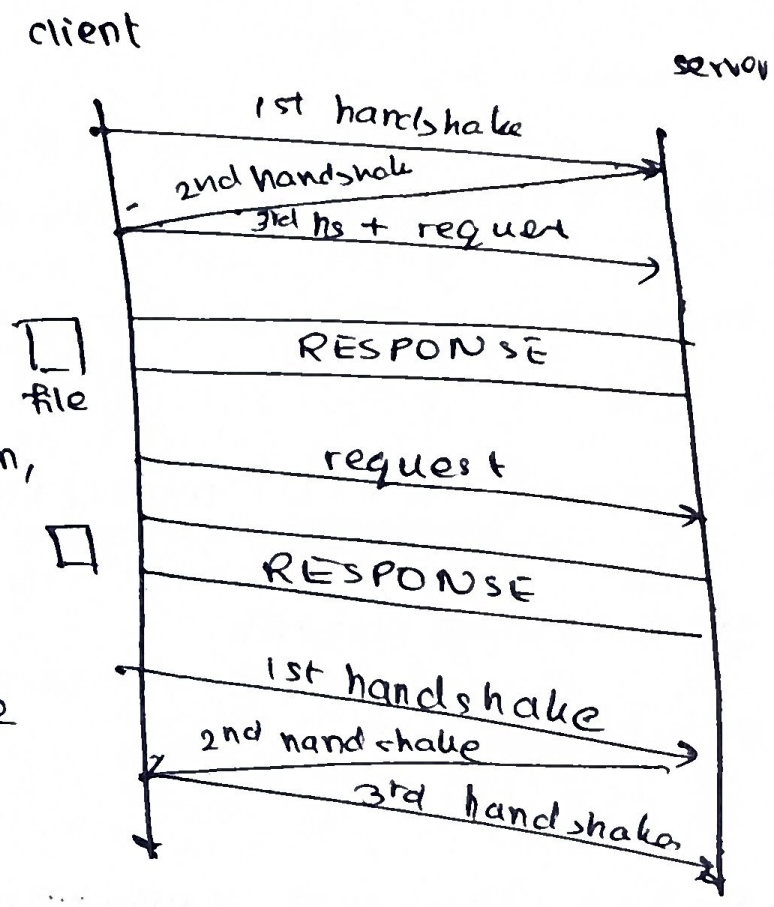
B. Persistent Connection

→ The server leaves the connection open for most requests after sending a response.

→ used in HTTP version 1.1

→ Server closes the connection at the request of a client or if a timeout has been reached.

→ Sometimes length of the data is known, otherwise, the server closes the connection after sending the document so that the client knows that the end of the data has been reached.



→ Time and resources are saved using persistent connections

→ Only one set of buffers and variables needs to be set for the connection at each site.

→ The round trip time for connection establishment & connection termination is saved.

* Message Formats

→ The HTTP protocol defines the format of the request and response messages.

A. Request Message

Line 1 in a request message is the request line.

It has: (i) method – defines request types

(ii) URL – defines address & name of web page

(iii) version – gives version of the protocol

Some methods that can be specified in (i) are:

GET → request document from serv

HEAD → requests info about document, but not document itself

PUT → send doc from client to server

POST → send info from client to the server.

Line 2.:— – Request Header Lines

→ can have zero or more request header lines

→ can send additional info from client to the server

→ some headers include - user-agent, cookie, if-modifiedsince

B. Response message

→ response message consists of a status line, header line, a blank line & sometimes a body

Line1 - Status line - has version, status code (3 digits) and phrase

The status code values have different meanings

100 range - informational

200 range - successful requests

300 range - redirect client to another URL

400 range - error at client side

500 range - error at server side

Line 2 - response header lines

→ sends additional info. from server to client (like date, set-cookie, last-modified)

* Conditional Requests

→ A client can add a condition in its request

→ Server sends webpage if condition is met, informs client otherwise

→ A common condition: time & date webpage is modified

* Cookies.

→ The web has functions that needs it to remember some info. about clients. - done with cookies

A. Creating and Storing Cookies

→ server receives a request from a client - it stores info about the client in a file or string

info may be: • client domain name
• contents of cookie
• timestamp

→ The server includes the cookie in the response it sends to the client.

→ When the client receives the response, the browser stores the cookie in the cookie directory, which is sorted by the server domain name.

B. Using Cookies

→ When a client sends a request to the server - the browser looks in the cookie directory to see if it can find a cookie sent by that server. If found, the cookie is included in the request

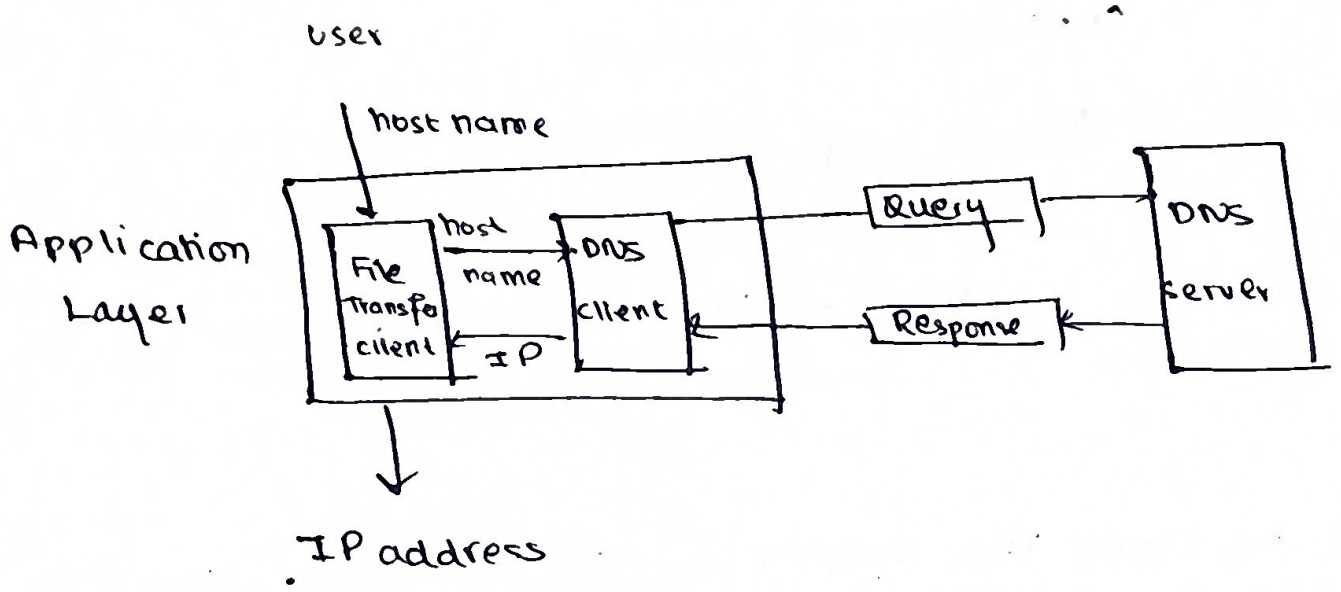
→ Cookies are used by e-commerce stores, registered clients, web portal, advertising agencies.

* DNS - PPT 2

→ Since the Internet is so large - a central directory system cannot hold all the mappings.

→ The information must be distributed among many computers in the world - the host that needs mapping can contact the computer closest to it that holds that info. This is how DNS works

* Usage of TCP/IP to map a name to an address using DNS



(1) User passes host name to file transfer client

(2) File transfer client passes host name to DNS client

(3) DNS client sends a message to a DNS server w/ a query that gives the FT server name using the known IP of the DNS server

(4) DNS server responds w/ IP of desired file transfer server

(5) DNS server passes IP to the file transfer client

(6) The file transfer client uses the received IP to access the file transfer server.

* Name Spaces

→ is a mapping of address to a unique name

→ can be 2 types:

(i) flat name space

(ii) hierarchical name space.

A. Flat Name Spaces

→ a name is directly assigned to an address - just a sequence of characters without structure

→ disadvantage: cannot be used in a large system like the Internet, as it would have to be centrally controlled to avoid ambiguity and duplication

B. Hierarchical Name Spaces

→ Each name is made of several parts

part 1 - nature of org

part 2 - name of org

part 3 - depts in org

→ The authority to assign & control name spaces can be decentralized.

→ Diff. prefixes / suffixes can be added to define its hosts or resources.

Domain Name Space - designed so that a hierarchical name space could be defined.

→ Names are defined in an inverted tree structure, with the root at the top

→ The tree can have only 128 levels

Labels - Each node in the tree has a label, which is a string with max 63 characters.

→ The root label is a null string

→ children that branch from the same node should have different labels, which guarantees the uniqueness of the domain names

Domain Name - each node in the tree has a domain name.

→ A full domain name is a sequence of labels separated by dots

→ Domain names are read from node up to the root

→ The last label is that of the root (null) ⇒ Domain name ends in a dot.

Types of Domain Names

a. Fully Qualified Domain Name (FQDN) - name ends with null label

b. Partially Qualified Domain Name (PQDN) - does not end w/ a null string - starts from a node - but does not end at root (resolver can help supply the missing part)

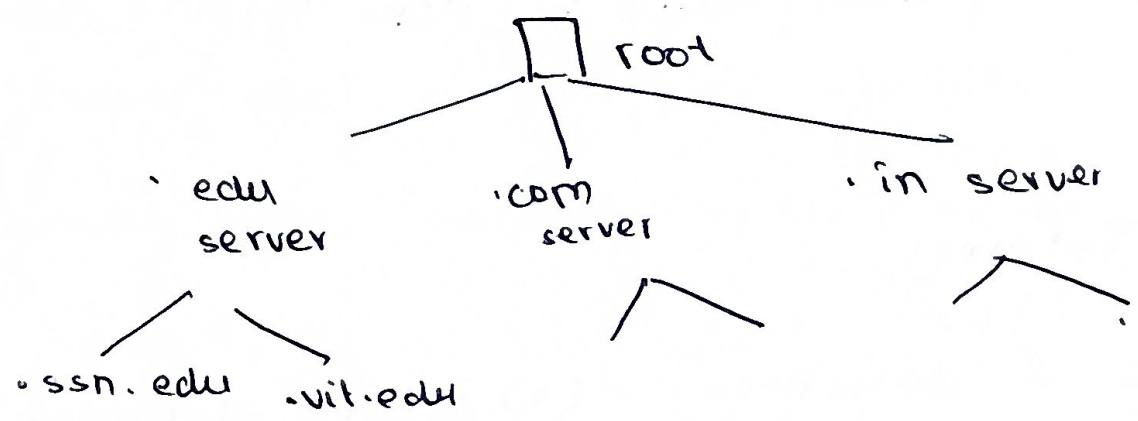
Domain - a subtree of the domain name space. The name of the domain is the name of the node at the top of the subtree.

* Distribution of Name Spaces

→ The info. in the domain name space must be stored. It can be distributed among many computers called DNS servers.

→ The whole space is divided into many domains based on the first level., and into further subdomains

→ That is, there is a hierarchy of servers just like there is a hierarchy of names.



Zone - What a server is responsible for / has authority over is called a zone.

→ A zone is defined as a contiguous portion of the entire tree.

→ If a server does not divide the domain further => domain & zone are the same.

Root Server - A server whose zone has the whole tree
- does not store any info. about domains - delegates authority to other servers, keeping references.

Primary & Secondary Servers

A. Primary Server - a server that stores a file about the zone for which it is authority

→ responsible for creating, maintaining & updating the zone file

→ stores zone file on local disk

B. Secondary Server - a server that transfers the complete information about a zone from another (1° or 2°) and stores the file on its local disk

→ neither creates nor updates zone files

→ any updates if required, must be done by 1°, send updates to 2°.

* DNS in the Internet

→ divided into 3 sections : (i) generic domains
(ii) country domains
(iii) inverse domains - difficult to keep track of, not used (deprecated)

A. Generic Domains - define registered hosts based on generic behavior - eg. .aero, .edu, .gov, .net, .org

B. County Domains - use 2 character country abbreviations, second labels can be organizational / state-wise eg.

.ae , .fr , .ca.us

* Resolution

→ Mapping a name to an address is called name-address resolution.

→ A resolver accesses the closest DNS server with a mapping request. If the server has the information, it satisfies the resolver, otherwise, it refers the resolver to other servers or asks other servers to provide the information.

→ Resolution can be either recursive or iterative.

Recursive Resolution

Steps: ① Application program on source host calls the DNS resolver to find the IP address of destination host.

② Resolver does not IP- send query to local DNS server.

③ Assuming the local DNS server does not know either, send a query to the root DNS server.

④ The root server knows about one server at each top level domain (eg. a server for .com domain)

⑤ Send query to top level domain server - and then to local server.

⑥ Once the IP is found, trace path back to source

Iterative Resolution

use same example + pic

→ Each server that does not know the mapping sends the IP address of the next server back to the one that requested it.

* Caching

→ When a server asks for a mapping from another server and receives the responses, it stores this information in the cache memory before sending it back to the client.

→ If the same client or another client asks for the same mapping, it can check its cache memory

→ However, to inform the client that the response is coming from the cache memory and not from an authoritative source, the server marks the response as unauthoritative.

→ Caches speed up resolution, but also has drawbacks - can send an oudated mapping to the client. 2 Techniques can be used to counter this:

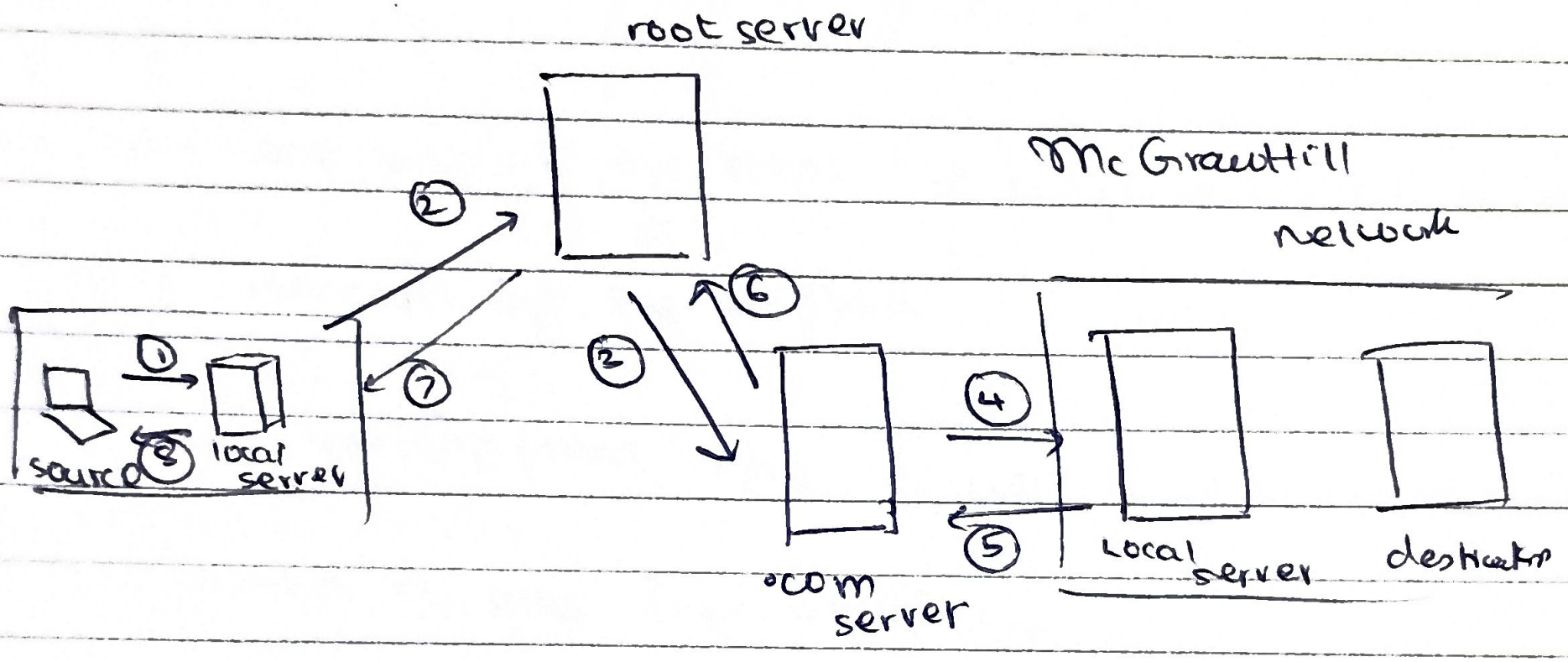
(i) Authoritative server adds TTL for caching

(ii) A TTL counter must be kept, and periodically purged.

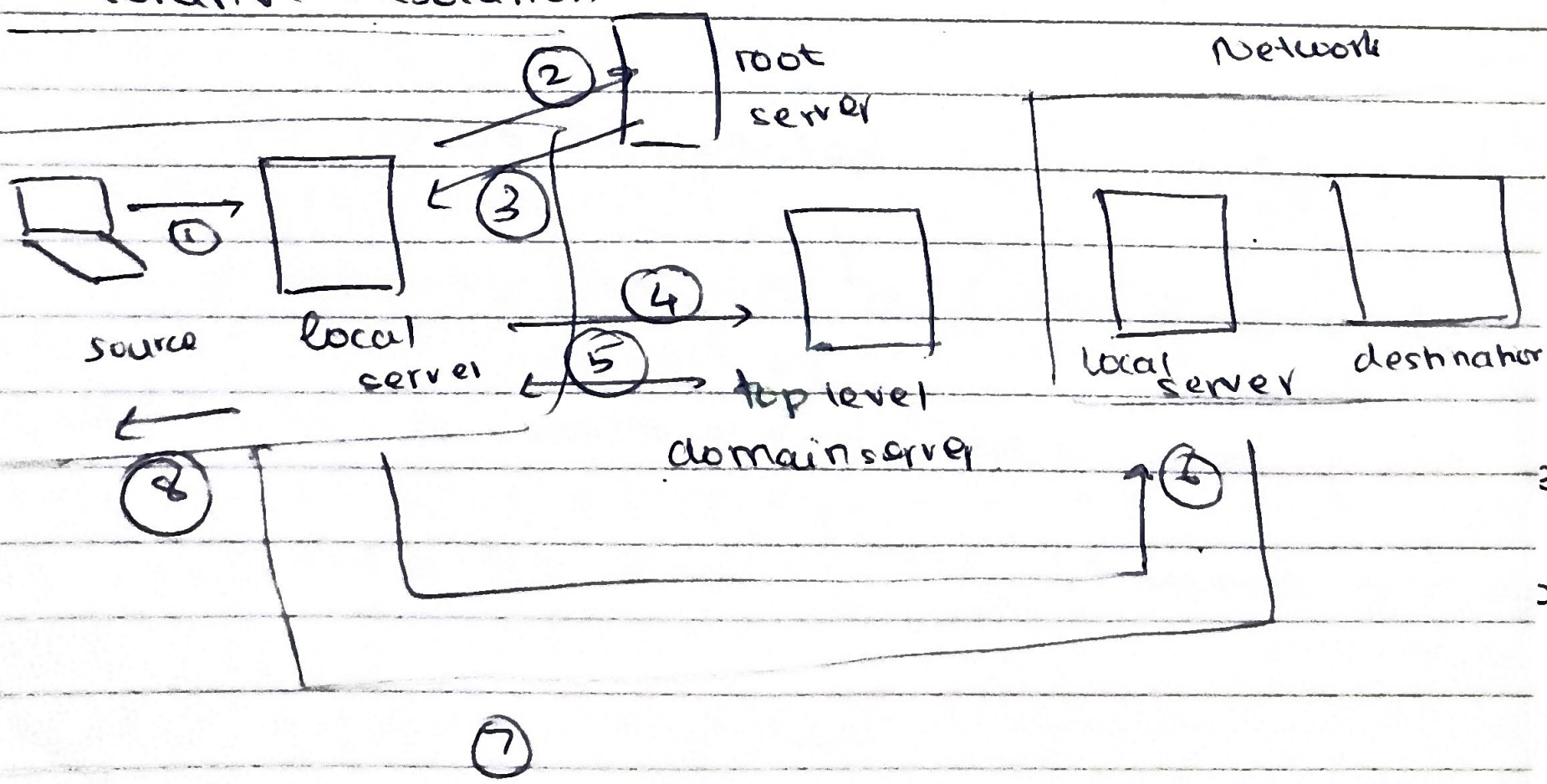
* Resource Records - A name server stores a database of records as a 5-tuple structure as:

(Domain Name, Type, Class, TTL, Value)

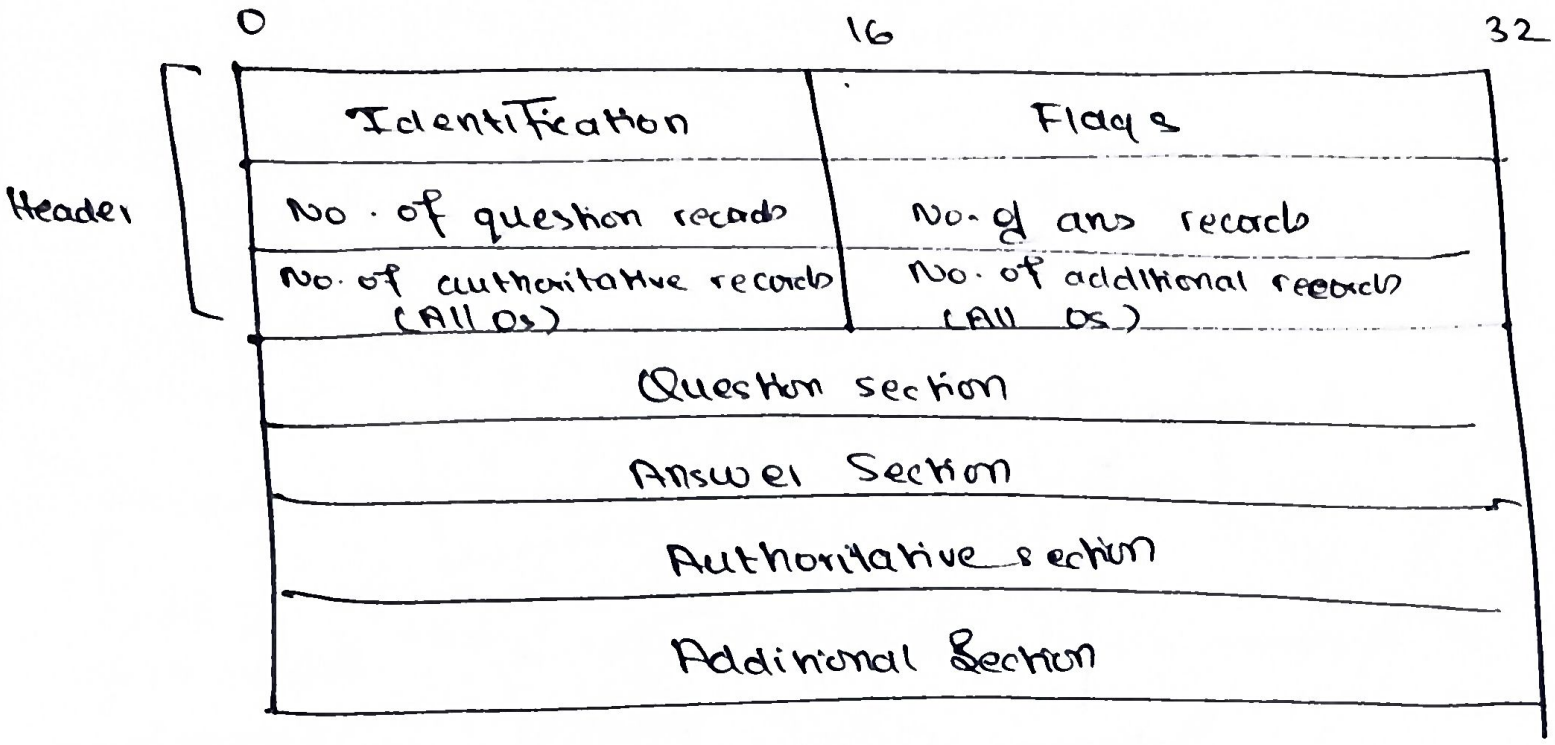
Recursive Resolution



Iterative Resolution



* DNS Message Format - has 2 types query & response with the following format:



* Registrar

→ New domains are added to DNS through a registrar, a commercial entity accredited by ICANN.

→ A registrar first verifies that the requested domain name is unique & then enters it into the DNS database.

* DDNS

→ The DNS master file must be updated dynamically.

→ In DDNS, when a binding between a name & an address is determined, the info is sent via DHCP to a primary DNS server.

→ The 1° server updates the zone. The 2° servers are notified either actively or passively.

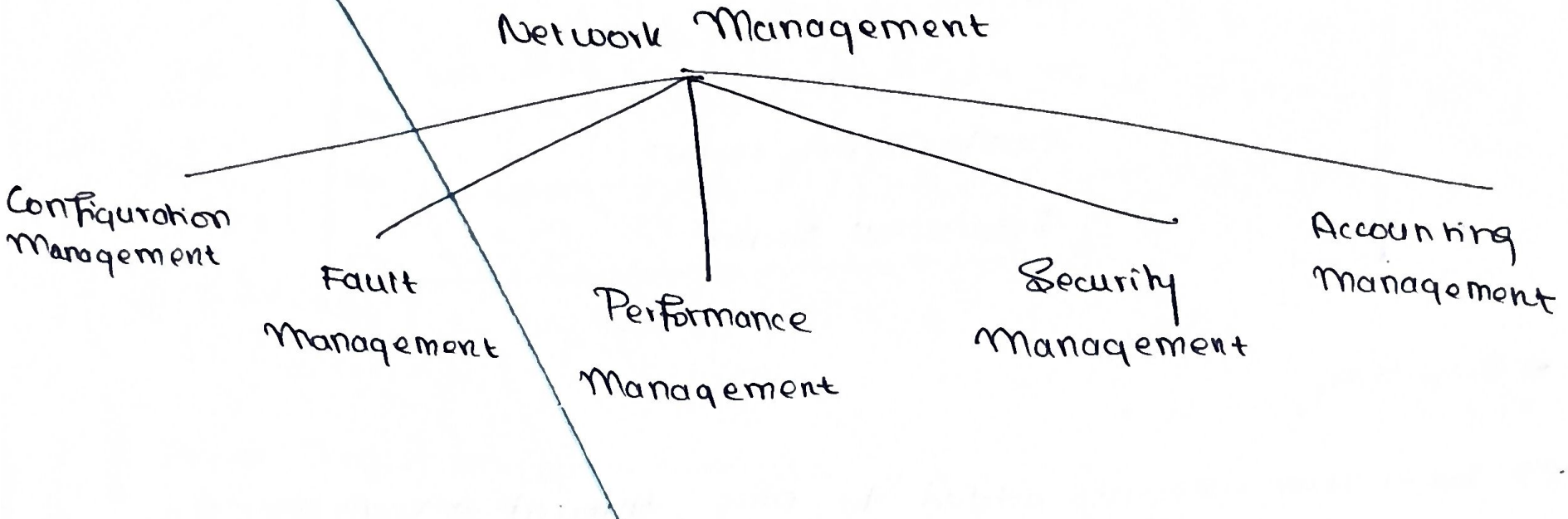
↓ actively: send a msg.

↓ passively: periodically check for changes

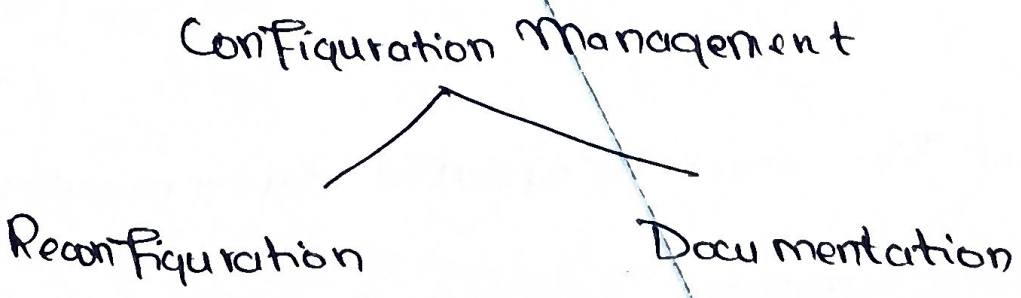
→ After noticing a change, the 2° server requests info on the entire zone (zone transfer)

* Network Management and SNMP - PPT-3

Network management = monitoring, testing, configuring and troubleshooting network components to meet a set of requirements defined by an organization. It is classified into 5 areas:



A. **Configuration Management** → the configuration mgmt. system must know, at any time, the status of each entity and its relation to other entities. Configuration mgmt. is divided into 2 subsystems:



Reconfiguration

(i) Hardware reconfiguration - deals with all changes to hardware, like replacement of peripherals - adding/ removing routers, must be done manually

(ii) Software Reconfiguration - changes in software like updates - can be automated

(iii) User - Account Reconfiguration - adding/ deleting users - also considering user priveleges - for individuals and groups

Documentation

(i) Hardware Documentation - includes maps & specifications

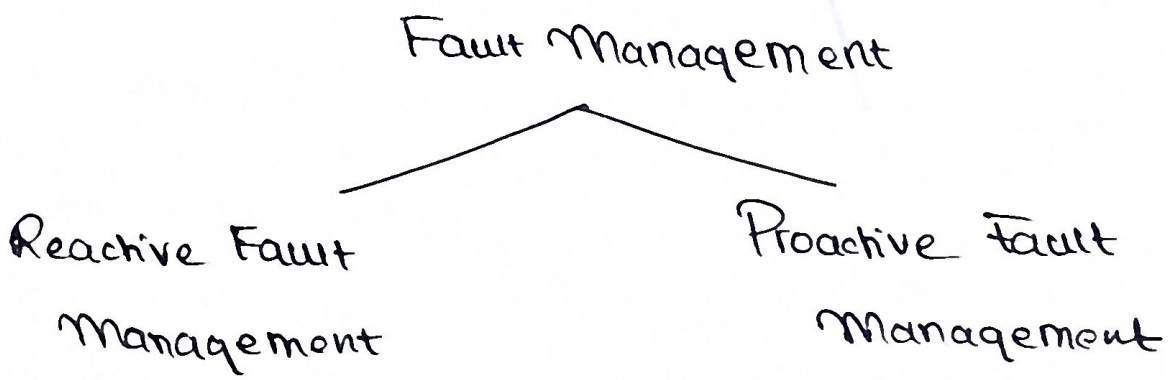
Maps - track each piece of h/w & its connection to the network

Specifications - has info. such as h/w type, serial no, vendor

(ii) Software Documentation - includes info. such as the type, version time installed and license agreement.

(iii) User Account Documentation - record access of files, and hold a list of users with file permissions

B **Fault Management** - handles and checks the proper operation of each component individually and in relation to each other. Fault mgmt. has 2 subsystems:.



Reactive Fault Management – responsible for detecting, isolating, correcting and recording faults.

Proactive Fault Management – tries to prevent faults from occurring, keep monitoring vulnerable points in a network. – reconfigure

C. Performance Management – tries to monitor and control the network so that it runs as efficiently as possible.

Measures (i) capacity

(ii) Traffic

(iii) Throughput

(iv) Response Time

D. Security Management – responsible for controlling access to the network based on pre-defined policies. – works on encryption & authentication,

E. Accounting Management – controls users' access to the network resources through charges

→ prevents users from monopolizing limited network resources

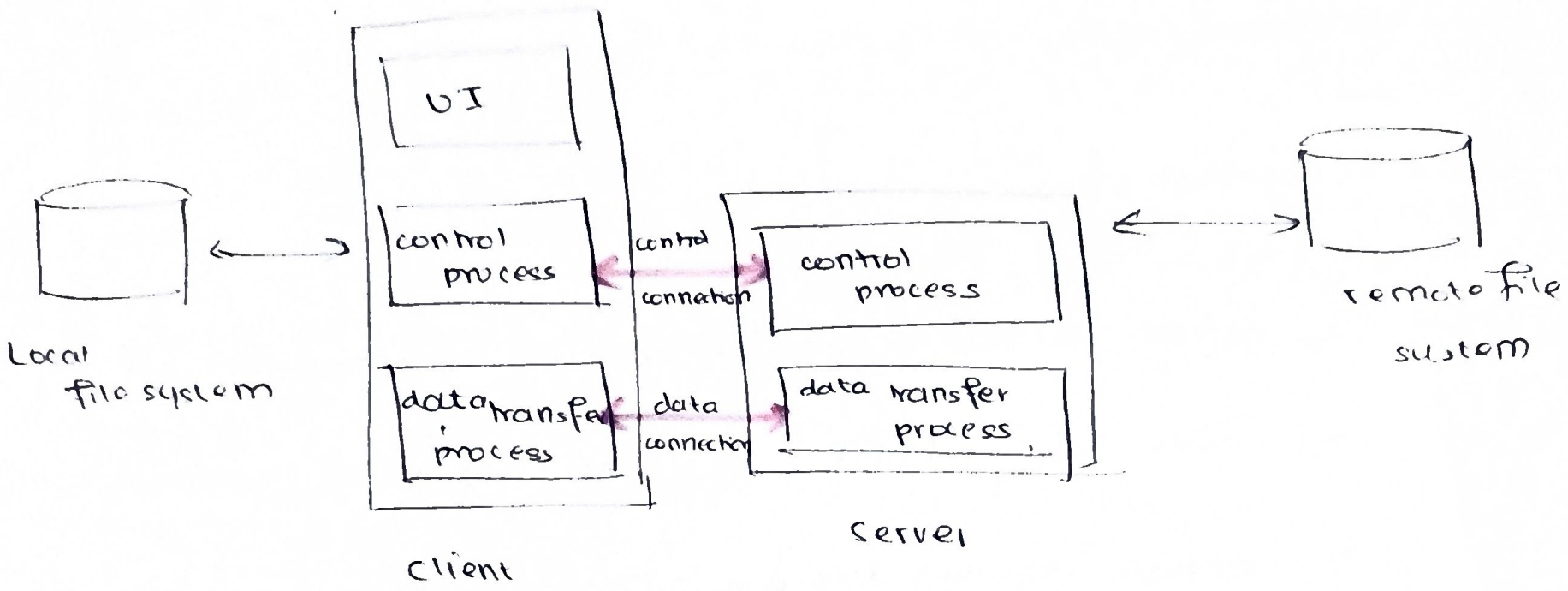
→ prevents users from using the system inefficiently

→ network managers can do short & long term planning based on the demand of network use.

* FTP

→ the standard protocol provided by TCP/IP for copying a file from one host to another.

→ The basic model of the FTP is as follows:



→ The control connection is made between the control process

→ The data connection is made between the data transfer processes

→ Separation of commands & data transfer makes FTP more efficient.

* Two Connections

→ The connections in FTP have different lifetimes.

→ The control connection remains connected during the entire session

→ The data connection is opened and closed for each file transfer activity.

→ | Port 21 – control connection
Port 20 – data connection

A. **Control Connection**

- send one command each, terminated by a carriage return & line feed.

- Each FTP response has a 3 digit no. followed by text

B. **Data Connection**

Steps:

1. client issues a passive open on an ephemeral port
2. client uses PORT to send port no. to the server
3. server receives the port no & issues an active open using the well known port 20 & the received ephemeral port number.

There are 3 attributes for communication

(1) File Type

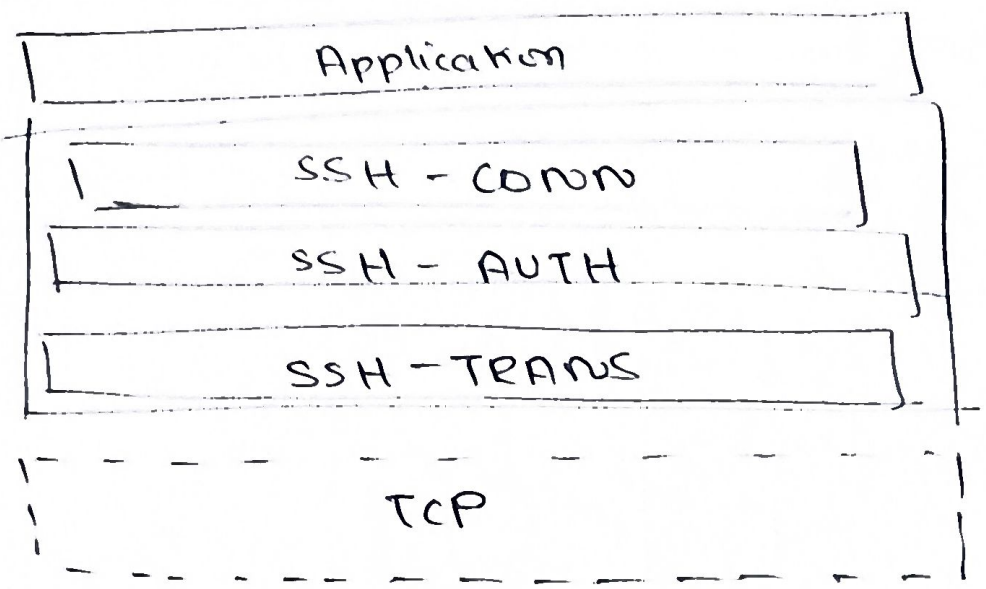
(2) Data Structure
- file (continuous)
- record (text)
- page structure

(3) Transmission
- stream
- block
- compressed mode.

* Secure Shell (SSH)

→ an application program that is used for remote logging and file transfer.

→ The 3 major components are:



A. SSH-TRANS

→ since TCP is not secure, SSH uses a protocol that creates a secure channel on top of TCP. - an independent protocol called SSH-TRANS

→ Its services include:

(i) privacy or confidentiality of the message exchanged

(ii) data integrity

(iii) server authentication

B. SSH - AUTH

→ a procedure to authenticate the client or server.

→ Auth. starts with the client, which sends a request message to the server. - includes user name, server name, method of auth

→ server responds with success/failure.

c. SSH-CONN

→ helps in multiplexing

→ allows client to create multiple logical channels

→ Each channel can be used for a different purpose such as remote logging, file transfer.

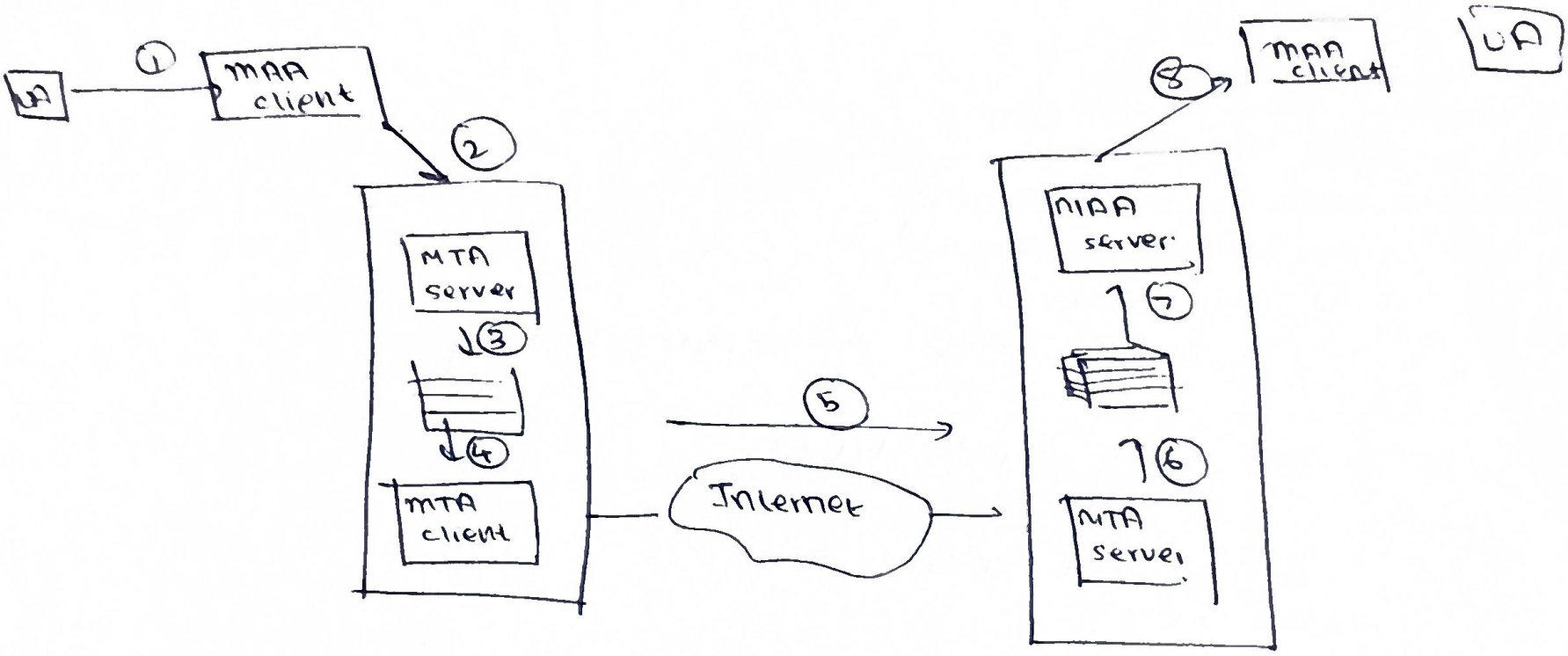
* Email

→ Email is considered a one-way transaction.

→ The client-server paradigm should be implemented with intermediate computers (servers)

→ The users run the client program when they want to.

Architecture



There are 3 different agents:

(i) User agent (UA)

(ii) message transfer agent (MTA)

(iii) message access agent (MAA)

Steps: (1) Sender runs UA program → can be command | GUI based — create an envelope & message

(2) Mail server uses a spool (queue) w/ messages to be sent.

(3) The MTA server runs all the time, the MTA client runs when used.

(4) To receive mail, the user agent is triggered by the timer.

(1) SMTP

→ The formal protocol that defines the MTA client and servers in the Internet

→ SMTP is used twice : (i) once between the sender's mail server

(ii) once between the 2 mail servers

Mail Transfer Phases

A. Connection Establishment

server

1. sends 220 - service ready
2. client sends HELO
3. server sends 250

B. Message Transfer

1. client sends MAILFROM
2. server sends 250
3. client sends RCPT TO
4. server sends 250
5. client sends DATA
6. server responds with 354 (start input)
7. client sends mail in lines
8. server responds w/ 250

c. Connection Termination

- client sends QUIT
- server responds with 221

(2) POP3

→ used when user needs to download email from mailbox on server.

→ POP3 has 2 modes

(i) delete mode - delete from mailbox after each retrieval

(ii) keep mode - mail remains in mailbox after retrieval

(3) IMAP

has POP3 functions +

(i) check email header prior to downloading

(ii) can check email for a particular string

(iii) can partially download an email

(iv) can create, delete or rename mail

(v) can create a hierarchy of emails

(4) MIME

→ Multipurpose Internet Mail Extensions

Russian, Chinese, Japanese

→ a supplementary protocol that allows non-ascii data to be sent through email.