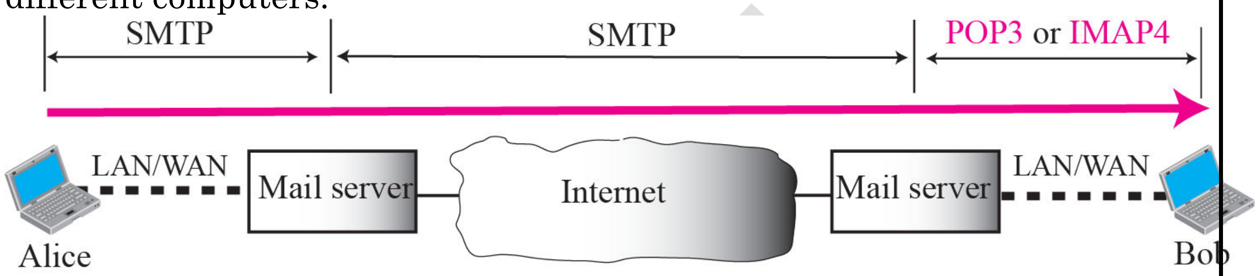
UNIT-V APPLICATION LAYER

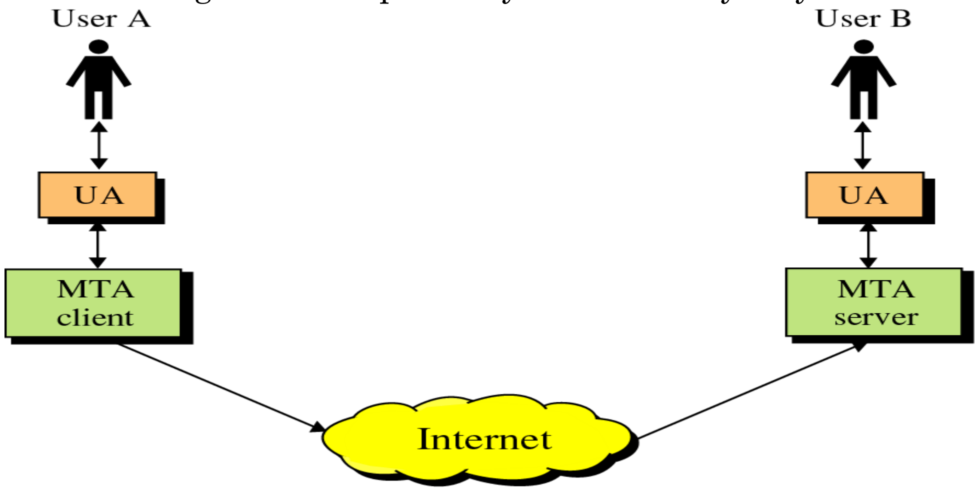
E-Mail (SMTP, MIME, POP3, IMAP), HTTP – DNS - FTP - Telnet – web services - SNMP – MIB– RMON.

SMTP - Simple Mail Transfer Protocol

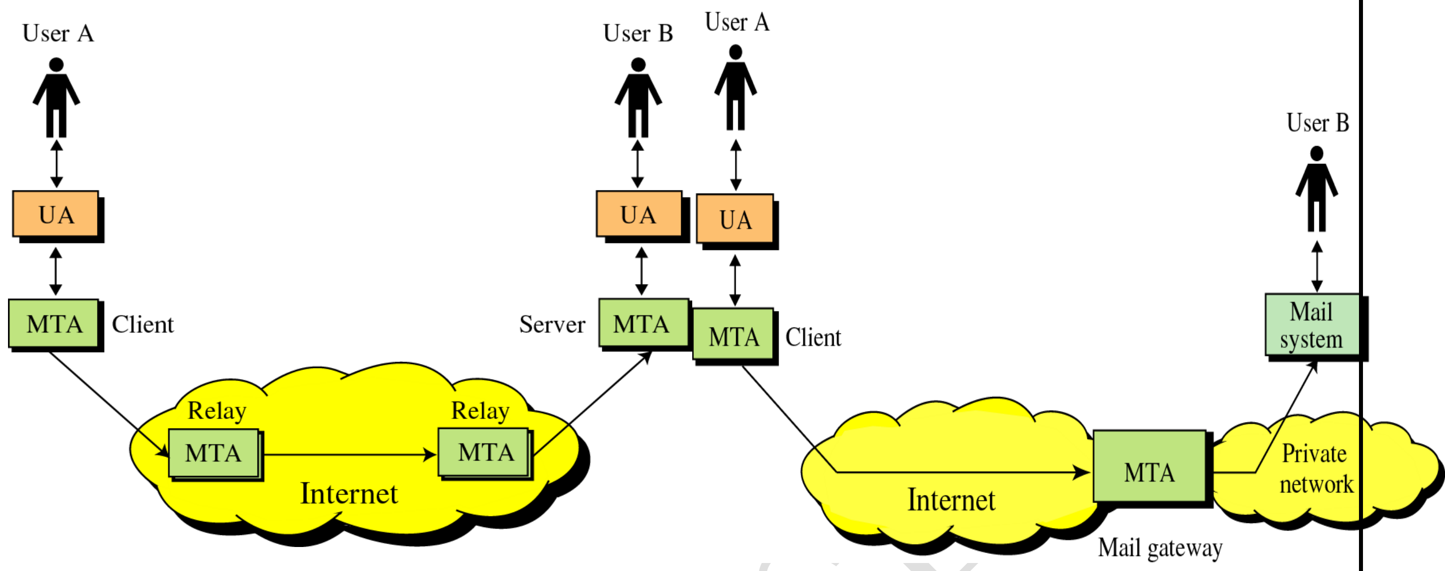
- The TCP/IP protocol suite supports electronic mail on the Internet through Simple Mail Transfer Protocol (SMTP).
- It is a protocol for sending messages to other computer users based on e-mail addresses.
- SMTP provides mail exchange between users on the same or different computers.



- SMTP originated in 1982 (RFC821, Jon Postel)
- Follows Standard message format (RFC822,2822, D. Crocker)
- Goal: To transfer mail reliably and efficiently.
- SMTP clients and servers have two main components
- User Agents – Prepares the message, encloses it in an envelope. (*Some examples of command-driven user agents are mail, pine, and elm*
  Some examples of GUI-based user agents are Eudora, Outlook, and Netscape.
- Mail Transfer Agent – Transfers the mail across the internet (ex. Sendmail, Exim)
- Analogous to the postal system in many ways.

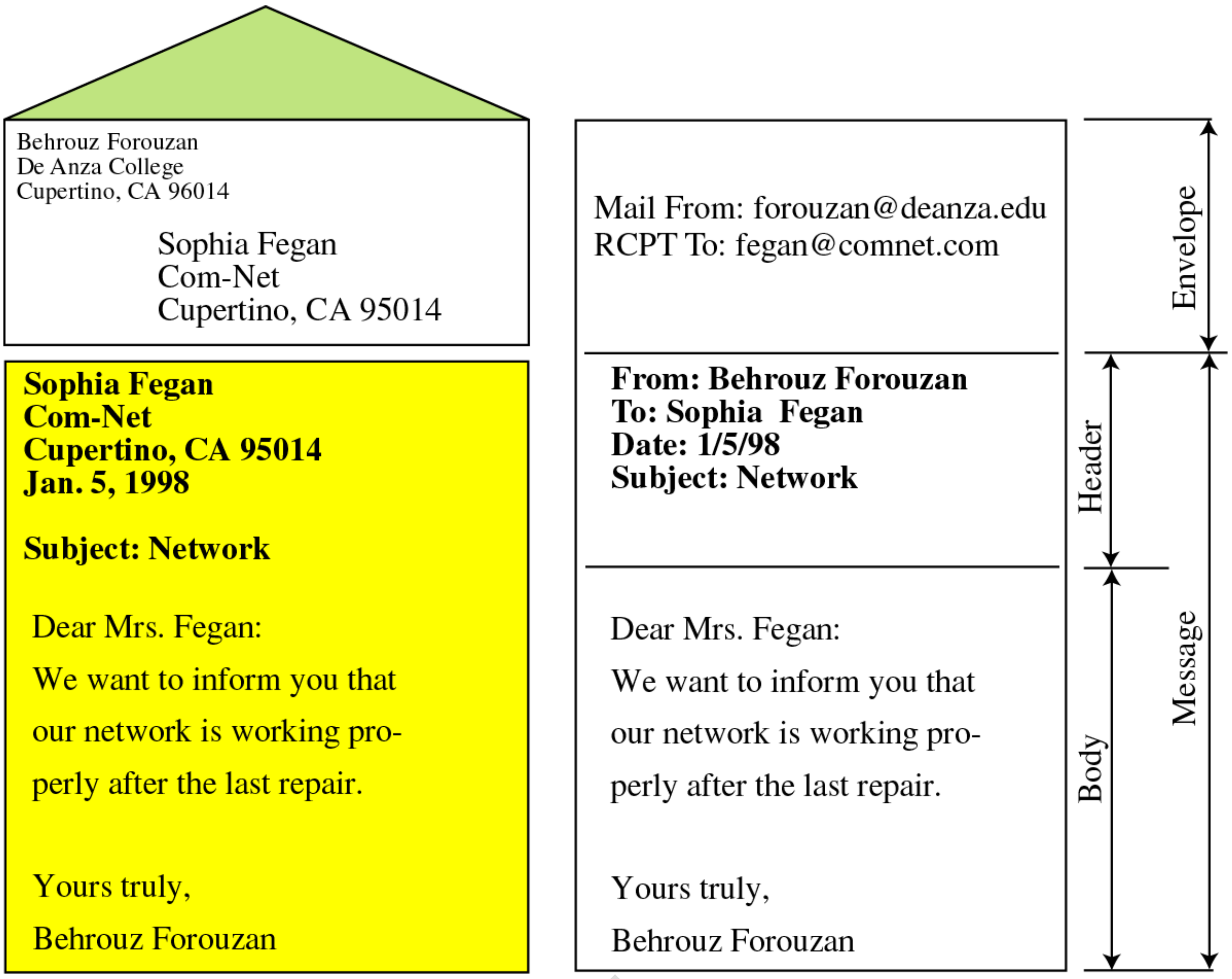


- SMTP also allows the use of Relays allowing other MTAs to relay the mail.
- Mail Gateways are used to relay mail prepared by a protocol other than SMTP and convert it to SMTP.

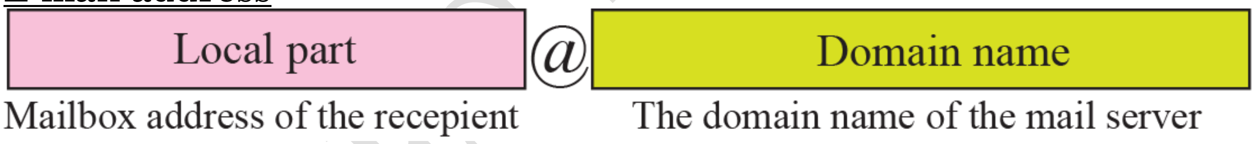


<u>Format of an email</u>

- Mail is a text file
- Envelope –
    - sender address
    - receiver address
- Message –
    - Mail Header – defines the sender, the receiver, the subject of the message, and other information
    - Mail Body – Contains the actual information in the message

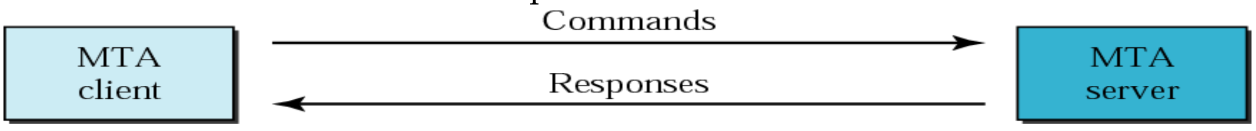


E-mail address

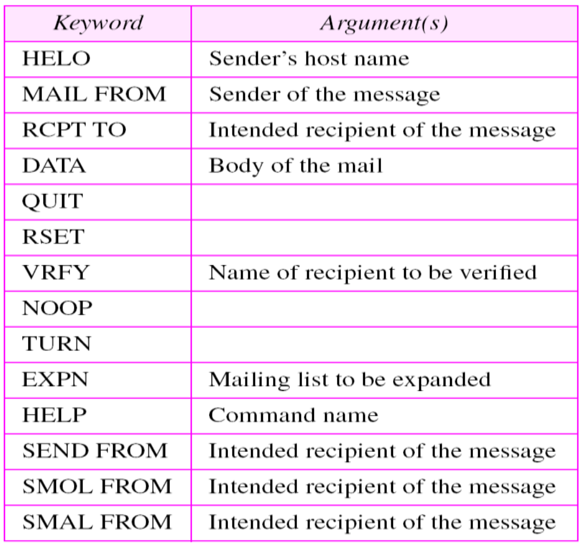

Local part
@
Domain name
Mailbox address of the recepient
The domain name of the mail server

Commands and responses

The Mail Transfer agents at the client side and server side communicate in the form of commands and responses.

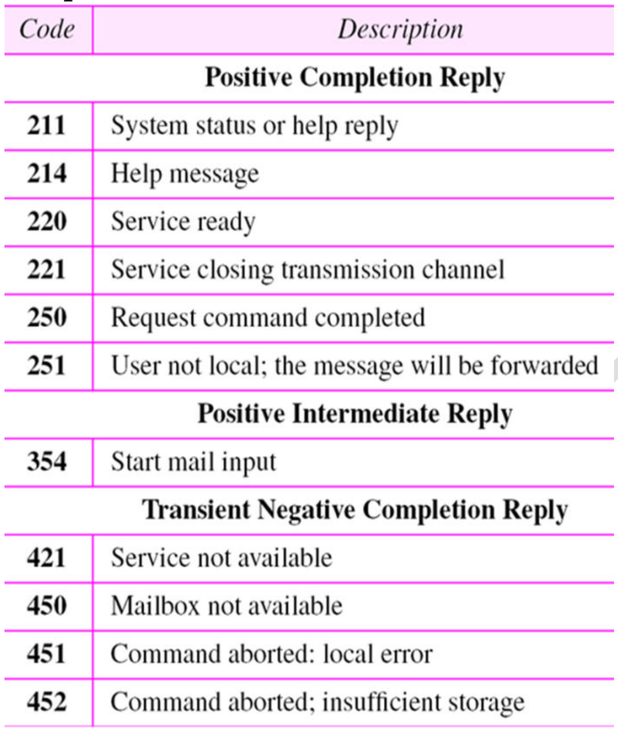


Commands

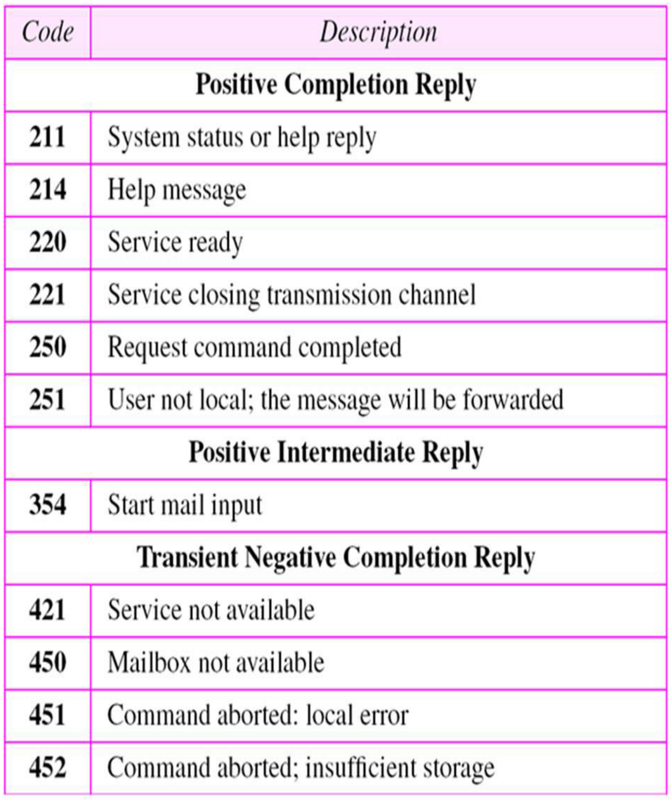
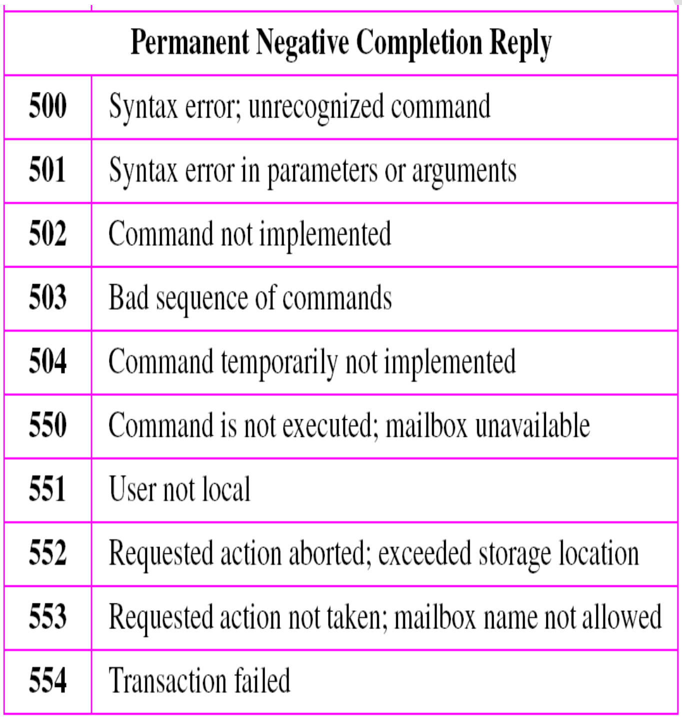


Keyword	*Argument(s)*
HELO	Sender's host name
MAIL FROM	Sender of the message
RCPT TO	Intended recipient of the message
DATA	Body of the mail
QUIT	
RSET	
VRFY	Name of recipient to be verified
NOOP	
TURN	
EXPN	Mailing list to be expanded
HELP	Command name
SEND FROM	Intended recipient of the message
SMOL FROM	Intended recipient of the message
SMAL FROM	Intended recipient of the message

Responses



Code	*Description*
	Positive Completion Reply
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
	Positive Intermediate Reply
354	Start mail input
	Transient Negative Completion Reply
421	Service not available
450	Mailbox not available
451	Command aborted: local error
452	Command aborted; insufficient storage

Code	Description
	Positive Completion Reply
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
	Positive Intermediate Reply
354	Start mail input
	Transient Negative Completion Reply
421	Service not available
450	Mailbox not available
451	Command aborted: local error
452	Command aborted; insufficient storage
	Permanent Negative Completion Reply
500	Syntax error; unrecognized command
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command temporarily not implemented
550	Command is not executed; mailbox unavailable
551	User not local
552	Requested action aborted; exceeded storage location
553	Requested action not taken; mailbox name not allowed
554	Transaction failed

How SMTP works (A-PDU's)

Keyword	Arguments
HELO	Sender's Host Domain Name
MAIL FROM:	Email Address of sender
RCPT TO:	Email of Intended recipient
DATA	Body of the message

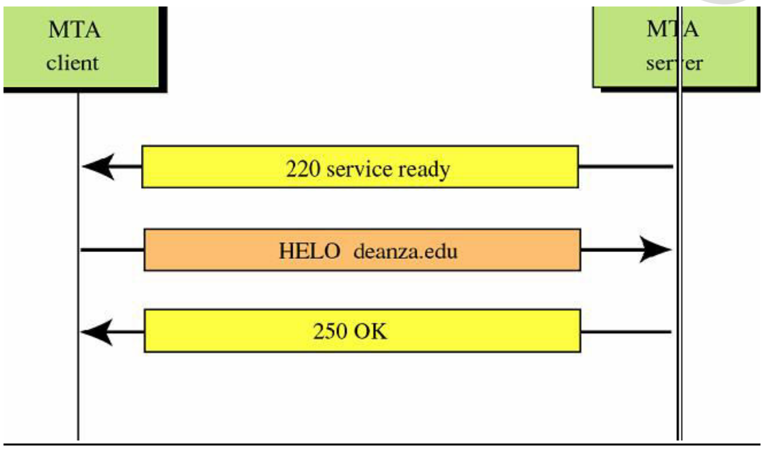
QUIT	
RSET	
VRFY	Name to be verified
NOOP	
TURN	
EXPN	Mailing list to expand
HELP	Command Name

Status Codes

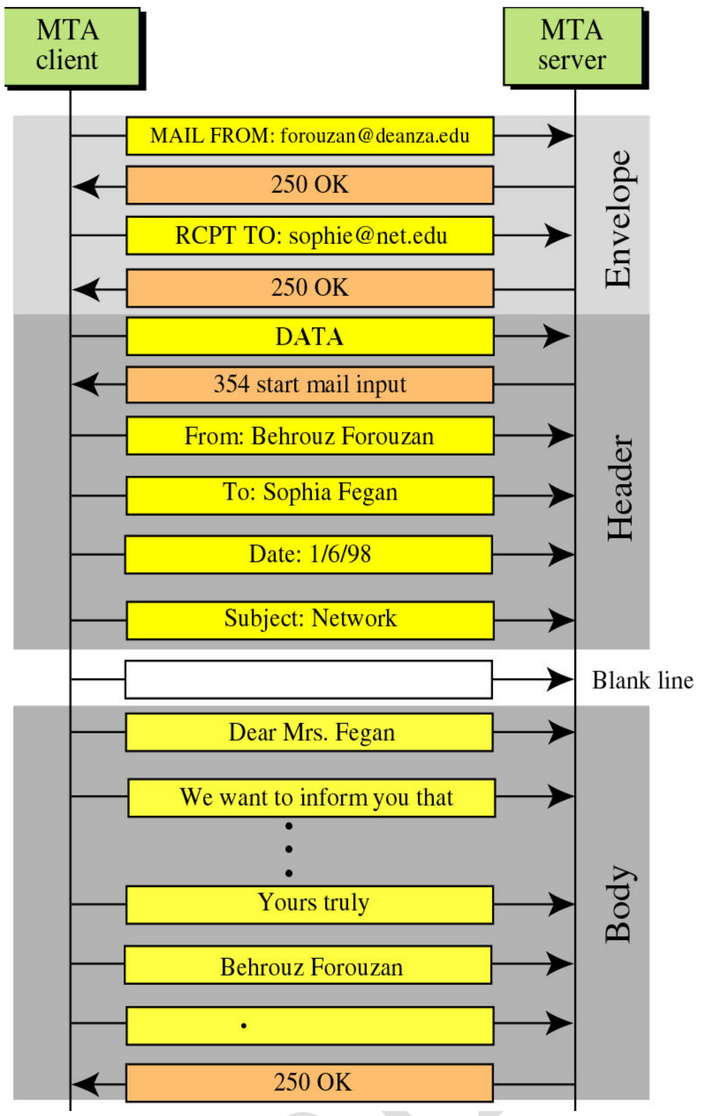
The Server responds with a 3 digit code that may be followed by text info

- 2## - Success
- 3## - Command can be accepted with more information
- 4## - Command was rejected, but error condition is temporary
- 5## - Command rejected, Bad User!

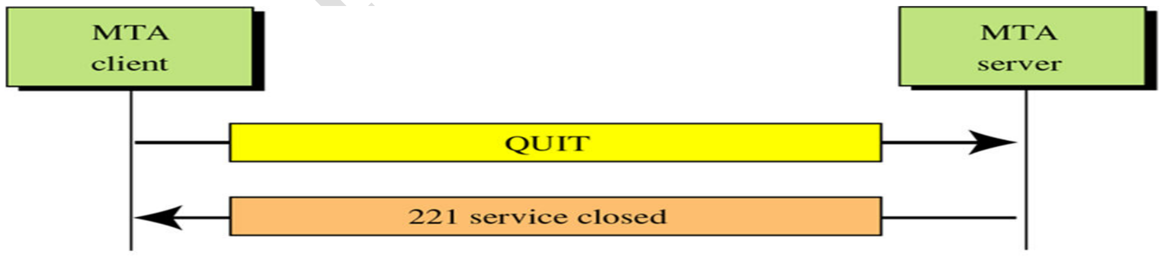
Connection Establishment



Message Progress



Connection Termination



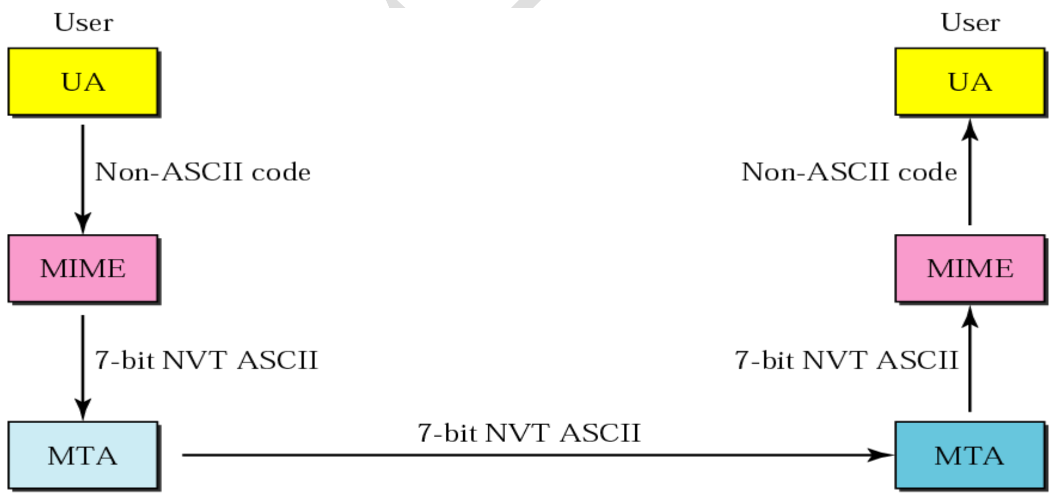
TCP Connection Termination

Limitations in SMTP

- Only uses NVT 7 bit ASCII format
    - How to represent other data types?
- No authentication mechanisms
- Messages are sent un-encrypted
- Susceptible to misuse (Spamming,faking sender address)

Multipurpose Internet Mail Extensions (MIME)

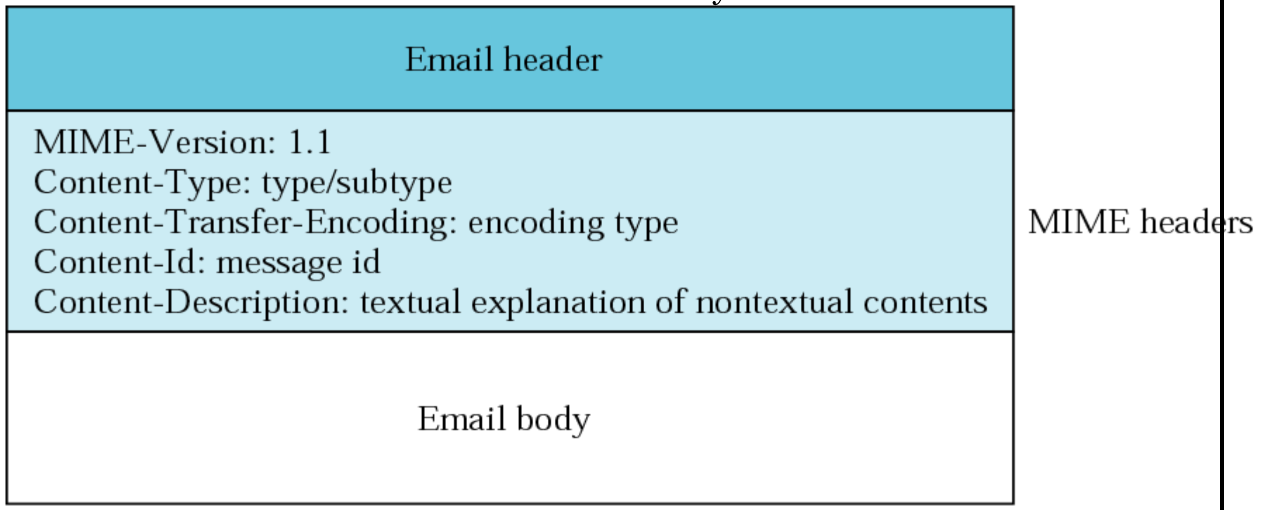
- Electronic mail has a simple structure. It can send messages only in NVT 7-bit ASCII format.
  It has some limitations. For example, it cannot be used for languages that are not supported by 7-bit ASCII characters (such as French, German, Hebrew, Russian, Chinese, and Japanese). Also, it cannot be used to send binary files or video or audio data.
- Multipurpose Internet Mail Extensions (MIME) is a 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 them to the client MTA to be sent through the Internet.
- In other words, MIME has a mechanism for sending multimedia data over e-mail.
- Transforms non-ASCII data to NVT (Network Virtual Terminal) ASCII data
  - ✓ Text
  - ✓ Application
  - ✓ Image
  - ✓ Audio
  - ✓ Video



- The message at the receiving side is transformed back to the original data. We can think of MIME as a set of software functions that transforms non-ASCII data (stream of bits) to ASCII data and vice versa, as shown

MIME Headers

- Located between the Email Header and Body


Email header
MIME-Version: 1.1
Content-Type: type/subtype
Content-Transfer-Encoding: encoding type
Content-Id: message id
Content-Description: textual explanation of nontextual contents
MIME headers
Email body

- Content-Type – Type of data used in the Body
  - Text: plain, unformatted text; HTML
  - Multipart: Body contains different data types
  - Message: Body contains a whole, part, or pointer to a message
  - Image: Message contains a static image (JPEG, GIF)
  - Video: Message contains an animated image (MPEG)
  - Audio: Message contains a basic sound sample (8kHz)
  - Application: Message is of data type not previously defined
- Content-Transfer-Encoding – How to encode the message
  - 7 bit – no encoding needed
  - 8 bit – Non-ASCII, short lines
  - Binary – Non-ASCII, unlimited length lines
  - Base64 – 6 bit blocks encoded into 8-bit ASCII
  - Quoted-printable – send non-ASCII characters as 3 ASCII characters, =##, ## is the hex representation of the byte

Data types and subtypes in MIME

Type	*Subtype*	*Description*
Text	Plain	Unformatted
	HTML	HTML format (see Chapter 22)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the same message

Type	*Subtype*	*Description*
Message	RFC822	Body is an encapsulated message
	Partial	Body is a fragment of a bigger message
	External-Body	Body is a reference to another message
Image	JPEG	Image is in JPEG format
	GIF	Image is in GIF format
Video	MPEG	Video is in MPEG format
Audio	Basic	Single channel encoding of voice at 8 KHz
Application	PostScript	Adobe PostScript
	Octet-stream	General binary data (eight-bit bytes)

Content-transfer-encoding

Type	*Description*
7bit	NVT ASCII characters and short lines
8bit	Non-ASCII characters and short lines
Binary	Non-ASCII characters with unlimited-length lines
Base64	6-bit blocks of data are encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code

MIME Messages

- MIME information is contained in the mail header using standard RFC 2822 format.
- MIME header specifies version, data type, encoding used to convert the data to ASCII.
- Example:
- From: bill@college.edu
- To: john@example.com
- MIME-Version: 1.0
- Content-Type: image/jpeg
- Content-Transfer-Encoding: base64
- ..data for the image..

MIME Multipart Messages

MIME multipart messages within the Content-Type adds considerable flexibility.

- There are four subtypes for a multipart message. The four subtypes are:

- mixed: allows a single message to contain multiple, independent sub-messages each having its independent type and encoding.
- alternative: allows a single message include multiple representations of the same data.
- parallel: allows a single message to include subparts that should be viewed together. (such as video and audio subparts)
- digest: allows a single message to contain a set of other messages.

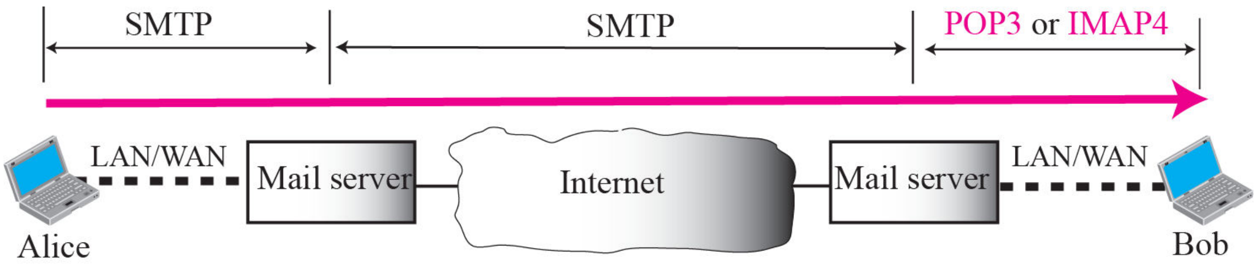
• To summarize, a multipart message can contain both a short text explaining the purpose of the message and some non-textual information

<u>Example:</u>

```
From: carrieerpdgns@yahoo.com
To: john@example.com
MIME-Version: 1.0
Content-Type: Multipart/Mixed; Boundary=StartOfNextPart
--StartOfNextPart
Content-Type: text/plain
Content-Transfer-Encoding: 7bit
John,
        Here is the photo of the carrier pigeons I saw last
week.
Sincerely,
Carrie Erpigeons
--StartOfNextPart
Content-Type: image/gif
Content-Transfer-Encoding: base64
        ..data for the image..
```

POP 3 (*Post Office Protocol)*

- Short for ***Post Office Protocol,*** a protocol used to retrieve e-mail from a mail server. Most e-mail applications (sometimes called an *e-mail client*) use the POP protocol, although some can use the newer IMAP (Internet Message Access Protocol).
- There are two versions of POP.
- The first, called *POP2,* became a standard in the mid-80's and requires SMTP to send messages. The newer version, POP3, can be used with or without SMTP. POP3 uses TCP/IP port 110.



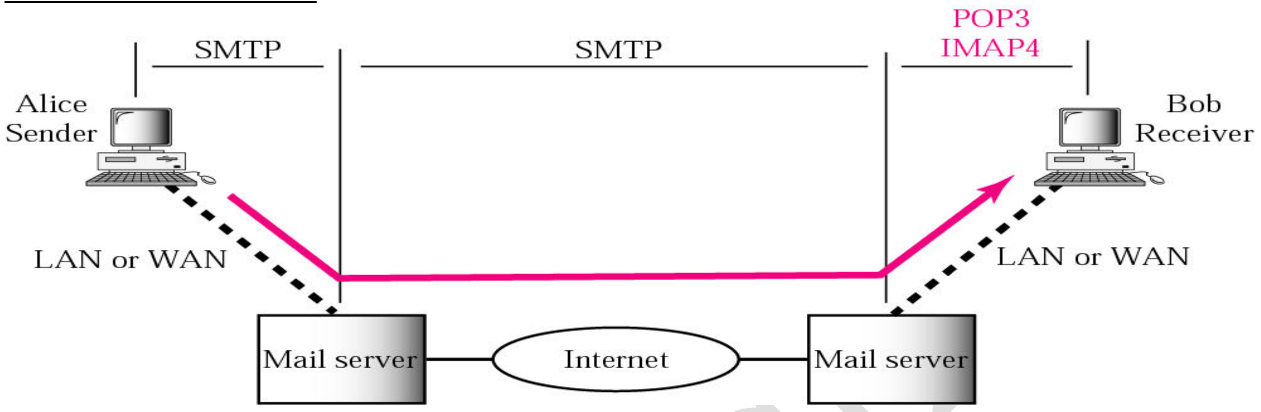
- Workstations interact with the SMTP host, which receives the mail on behalf of every host in the organization, to retrieve messages by using a client-server protocol such as Post Office Protocol, version 3(POP3). Although POP3 is used to download messages from the server, the SMTP client still needed on the desktop to forward messages from the workstation user to its SMTP mail server.

Post Office Protocol v3

- Simple
- Allows the user to obtain a list of their Emails
- Users can retrieve their emails
- Users can either delete or keep the email on their system
- Minimizes server resources
- Post Office Protocol, version 3 (POP3) is simple and limited in functionality.
- The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server.
- Mail access starts with the client when the user needs to download e-mail from the mailbox on the mail server.
- The client opens a connection to the server on TCP port 110. It then sends its user name and password to access the mailbox.
- The user can then list and retrieve the mail messages, one by one.
- Figure shows an example of downloading using POP3. POP3 has two modes: the delete mode and the keep mode.
- In the delete mode, the mail is deleted from the mailbox after each retrieval. In the keep mode, the mail remains in the mailbox after retrieval.

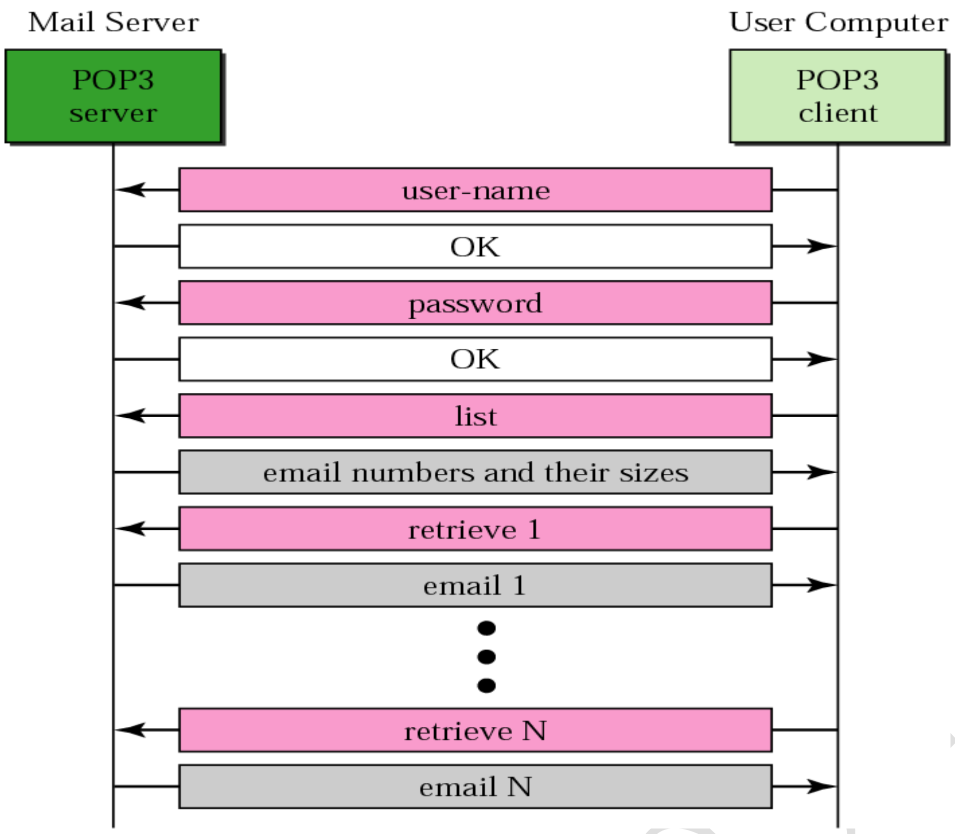
- The delete mode is normally used when the user is working at her permanent computer and can save and organize the received mail after reading or replying.
- The keep mode is normally used when the user accesses her mail away from her primary computer (e.g., a laptop). The mail is read but kept in the system for later retrieval and organizing

POP3 and IMAP4

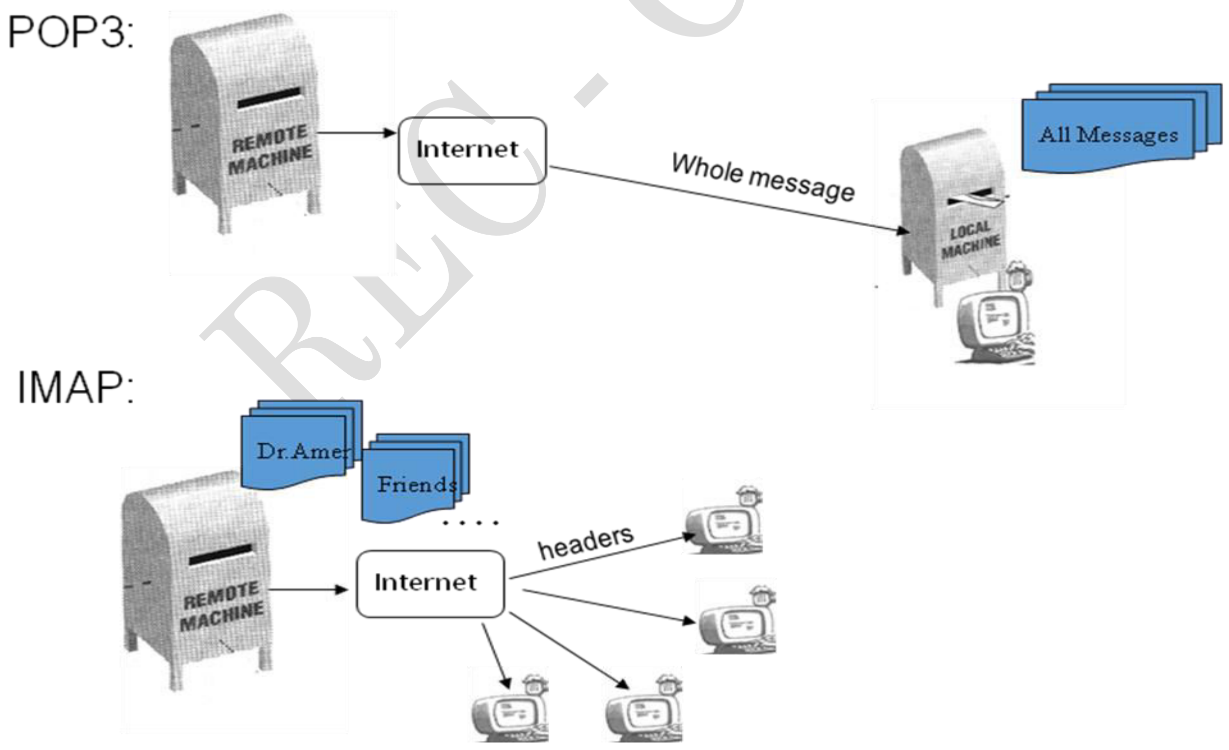


POP3 Commands

- **USER** name: User name for authentication
- **PASS** password: Password used for authentication
- **STAT:** Get number and total size of message
- **LIST:** [msg] get size of message
- **RETR:** msg Send message to client
- **DELE:** msg Delete message from mailbox
- **RSET:** Cancel previous delete requests.
- **QUIT:** Updates mailbox (deletes messages) and quits.



POP vs. IMAP



POP3 is deficient in several ways.

- It does not allow the user to organize her mail on the server; the user cannot have different folders on the server. (Of course, the user can create folders on her own computer.)
- POP3 does not allow the user to partially check the contents of the mail before downloading.

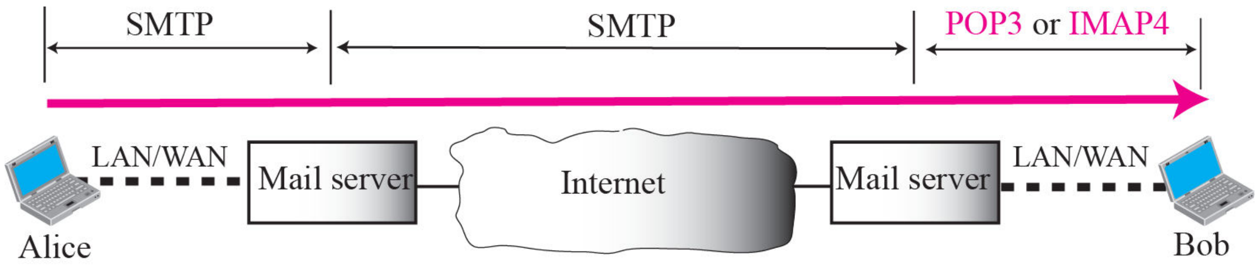
Pop vs IMAP

- Similarities
  - Mail delivered to a shared, constanly connected server
  - New mail accessible anywhere in network on a variety of platforms
  - For access only, Need SMTP to send mail
- Differences
  - POP simpler and more established (more clients and servers that support it)
  - IMAP is stateful protocol with more features

IMAP (Internet Message Access Protocol)

✓ IMAP is an **Internet Message Access Protocol**. It is a method of accessing electronic mail messages that are kept on a possibly shared mail server.

✓ For example, email stored on an IMAP server can be manipulated from a desktop computer at home, a workstation at the office, and a notebook computer while travelling, without the need to transfer messages or files back and forth between these computers. IMAP uses TCP/IP port 143.



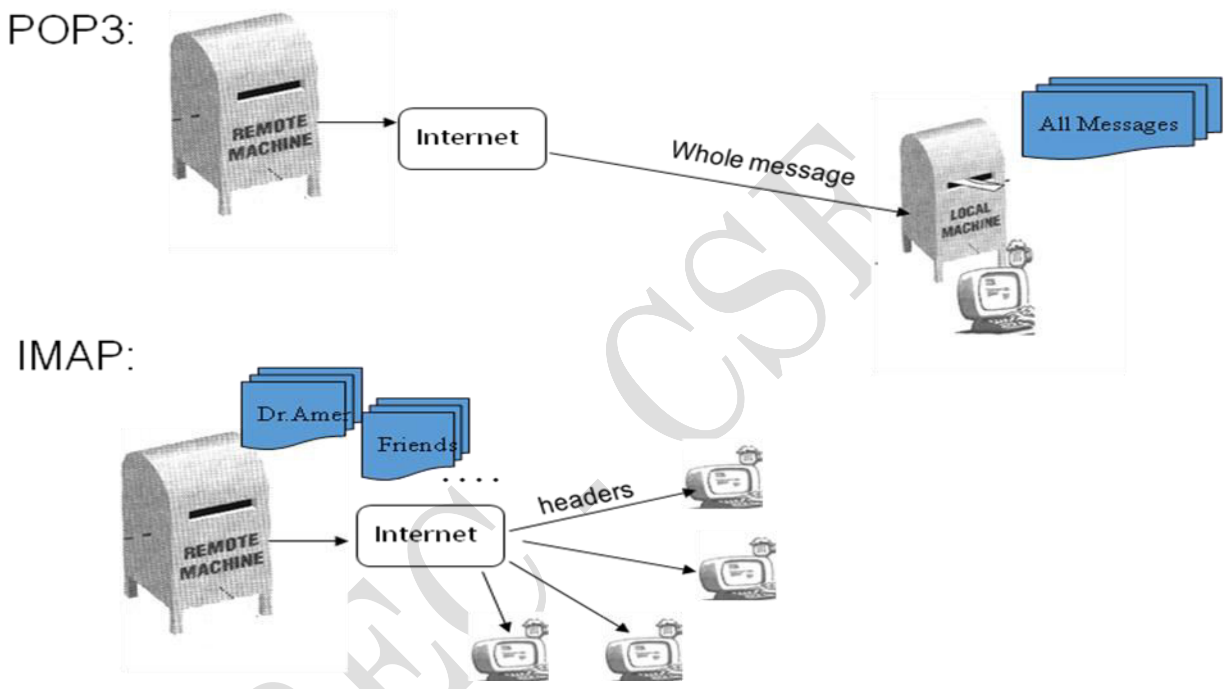
Internet Mail Access Protocol v4

- Similar to POP3 but as more features than it.
- User can check the email header before downloading
- Emails can be accessed from any location
- Can search the email for a specific string of characters before downloading
- User can download parts of an email
- User can create, delete, or rename mailboxes on a server
- It defines an abstraction known as a MAILBOX. Mailboxes are located on the same computer as a server.
- IMAP4 is a method for accessing electronic mail messages that are kept on a mail server. It permits a client e-mail program to view and manipulate those messages.
- Electronic mail stored on an IMAP server can be viewed or manipulated from a desktop computer at home,a notebook computer, or at a workstation. We can also say that mail messages can be accessed from multiple locations.

Functions of IMAP4

- IMAP provides extended functionality for message retrieval and processing.
- Users can obtain information about a message or examine header fields without retrieving the entire message.
- Users can search for a specified string and retrieve portions of a message. This is useful for slow-speed dialup connections since they wont need to download useless information.

POP vs. IMAP



Pop vs IMAP

Similarities

- ✓ Mail delivered to a shared, constanly connected server
- ✓ New mail accessible anywhere in network on a variety of platforms
- ✓ For access only, Need SMTP to send mail

Differences

- ✓ POP simpler and more established (more clients and servers that support it)
- ✓ IMAP is stateful protocol with more features

- With IMAP, all your mail stays on the server in multiple folders, some of which you have created. This enables you to connect to any computer and see all your mail and mail folders. In general, IMAP is great if you have a dedicated connection to the Internet or you like to check your mail from various locations.
- With POP3 you only have one folder, the Inbox folder. When you open your mailbox, new mail is moved from the host server and saved on your computer. If you want to be able to see your old mail

messages, you have to go back to the computer where you last opened your mail.
- With POP3 "leave mail on server" only your email messages are on the server, but with IMAP your email folders are also on the server.

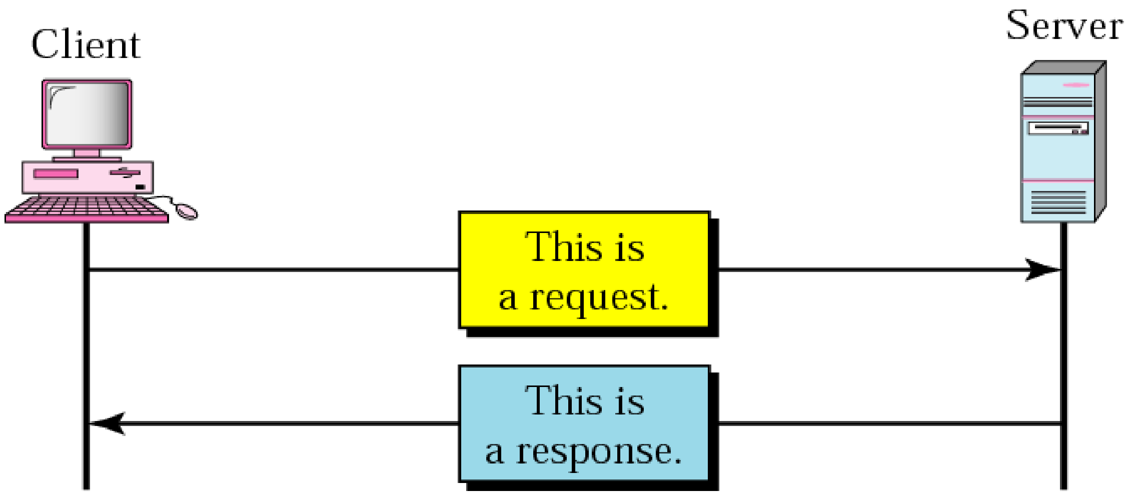
HTTP - Hypertext Transfer Protocol

- HTTP is an application layer network protocol built on top of TCP.
- *HTTP* - the Hypertext Transfer Protocol - provides a standard for Web browsers and Web servers to communicate. A "Web Server" is a HTTP server.
- HTTP clients (such as Web browsers) and servers communicate via HTTP messages.
- Most clients/servers today speak version 1.1, but 1.0 is also in use.
  - RFC 1945 (HTTP 1.0)
  - RFC 2616 (HTTP 1.1)
- Transport Independence
  - The HTTP protocol generally takes place over a TCP connection,
  - but the protocol itself is not dependent on a specific transport layer.
- HTTP allows transfer of various data formats between server and client
  - Plaintext
  - Hypertext
  - Images
  - Video
  - Sound
- Meta-information can also be transferred
- ***HTTP uses the services of TCP on well-known port 80.***

HTTP transaction

Two types of messages are

1. Request message
2. Response message

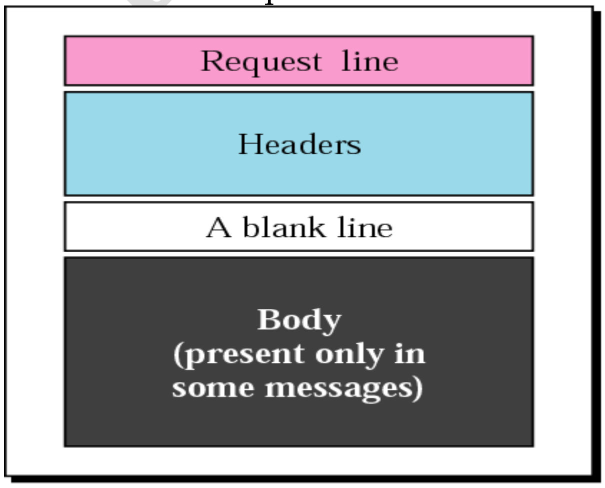


- HTTP has a simple structure:
  - client sends a request
  - server returns a reply.
  - HTTP can support multiple request-reply exchanges over a single TCP connection.

Request Messages

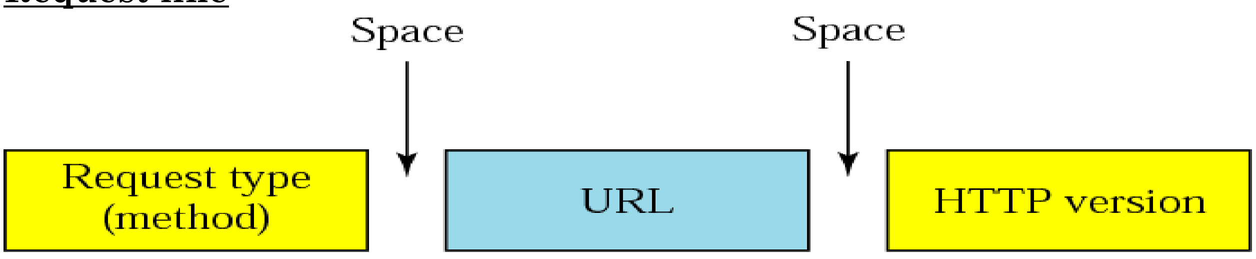
The first line of an HTTP request message specifies three things:
- the operation to be performed,
- the Web page the operation should be performed on
- and the version of HTTP being used.

Although HTTP defines a wide assortment of possible request operations—including "write" operations that allow a Web page to be posted on a server—the two most common operations are GET (fetch the specified Web page) and HEAD (fetch status information about the specified Web page).
- Lines of text (ASCII).
- Lines end with CRLF **"\r\n"**
- First line is called "Request-Line"

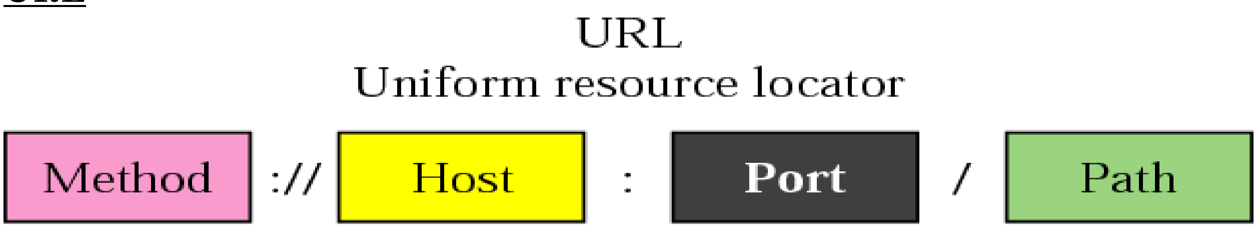


Request message

Request line



URL



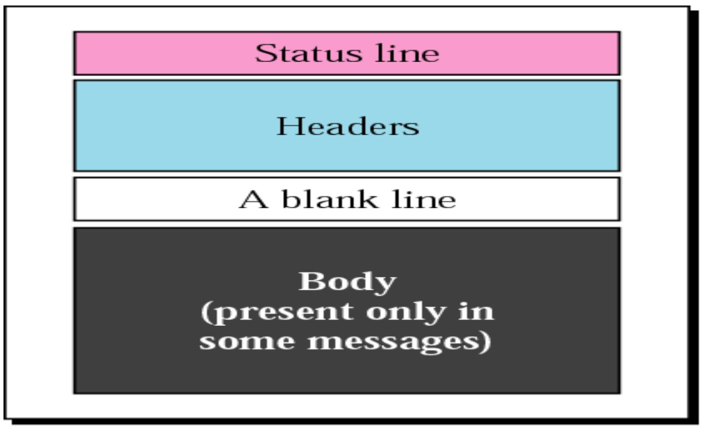
HTTP request operations

Operation	Description
OPTIONS	Request information about available options
GET	Retrieve document identified in URL
HEAD	Retrieve metainformation about document identified in URL
POST	Give information (e.g., annotation) to server
PUT	Store document under specified URL
DELETE	Delete specified URL
TRACE	Loopback request message
CONNECT	For use by proxies

Response Messages

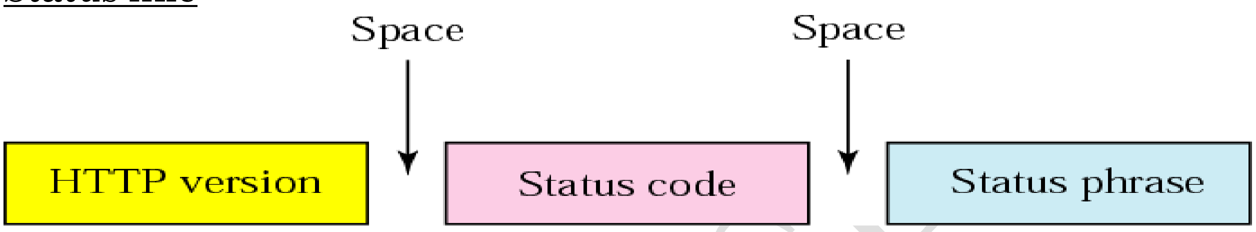
Like request messages, response messages begin with a single START LINE.

- In this case, the line specifies the version of HTTP being used, a three-digit code indicating whether or not the request was successful, and a text string giving the reason for the response.
- ASCII Status Line
- Headers Section
- Content can be anything (not just text)
    - typically an HTML document or some kind of image.

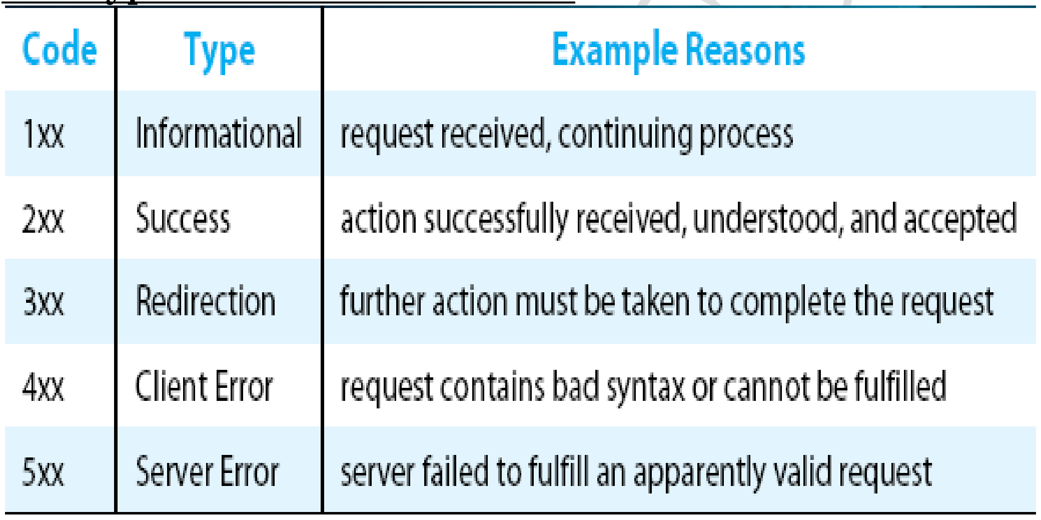


Response message

Status line



Five types of HTTP result codes



Code	Type	Example Reasons
1xx	Informational	request received, continuing process
2xx	Success	action successfully received, understood, and accepted
3xx	Redirection	further action must be taken to complete the request
4xx	Client Error	request contains bad syntax or cannot be fulfilled
5xx	Server Error	server failed to fulfill an apparently valid request

Uniform Resources

- URL
  - Uniform Resource Locator
  - Refers to an existing protocol
    - http:, wais:, ftp:, mailto:, gopher:, news:
  - Points to a document on a specific server
- URN
  - Uniform Resource Name
  - Globally unique, persistent identifier
    - Independent of location
- URI
  - Uniform Resource Identifier
  - Collection of URL's and URN's

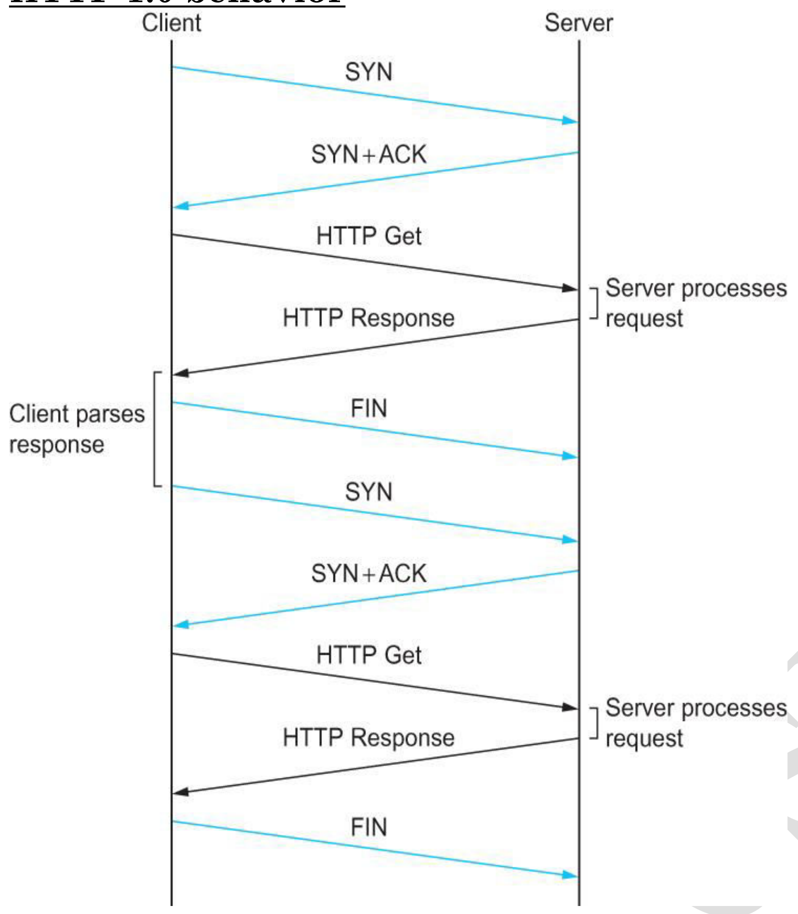
TCP Connections

- The original version of HTTP (1.0) established a separate TCP connection for each data item retrieved from the server.
- It's not too hard to see how this was a very inefficient mechanism: connection setup and teardown messages had to be exchanged between the client and server even if all the client wanted to do was verify that it had the most recent copy of a page.
- Thus, retrieving a page that included some text and a dozen icons or other small graphics would result in 13 separate TCP connections being established and closed.
- To overcome this situation, HTTP version 1.1 introduced *persistent connections*— the client and server can exchange multiple request/response messages over the same TCP connection.

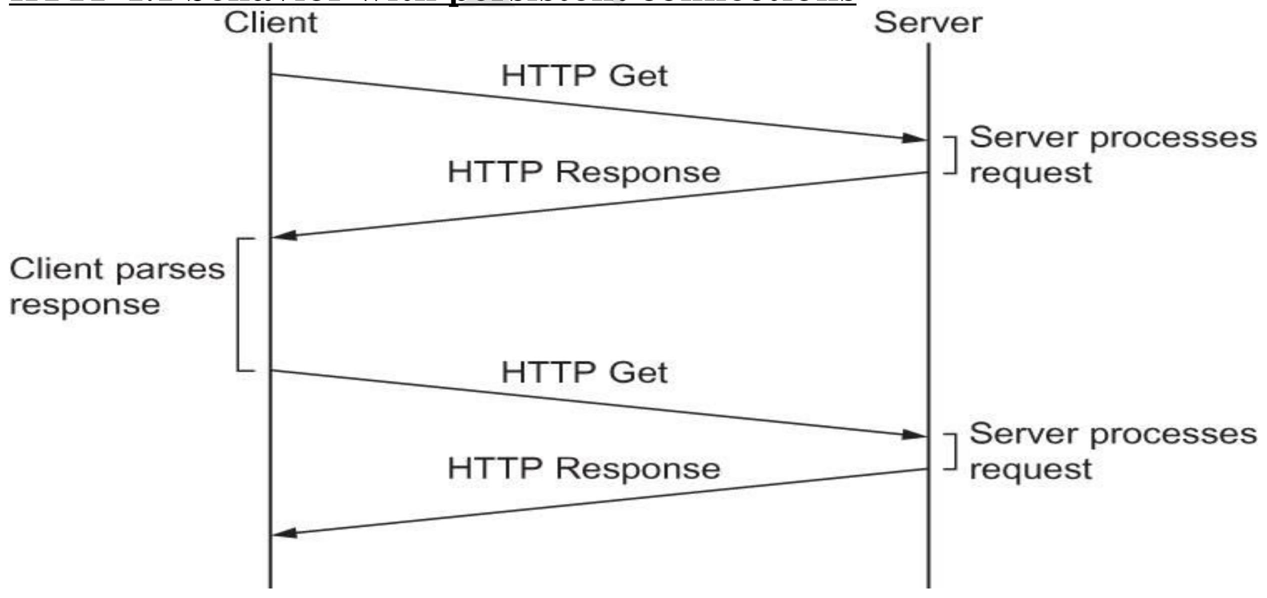
Persistent connections have many advantages.

- First, they obviously eliminate the connection setup overhead, thereby reducing the load on the server, the load on the network caused by the additional TCP packets, and the delay perceived by the user.
- Second, because a client can send multiple request messages down a single TCP connection, TCP's congestion window mechanism is able to operate more efficiently.
- This is because it's not necessary to go through the slow start phase for each page.

HTTP 1.0 behavior



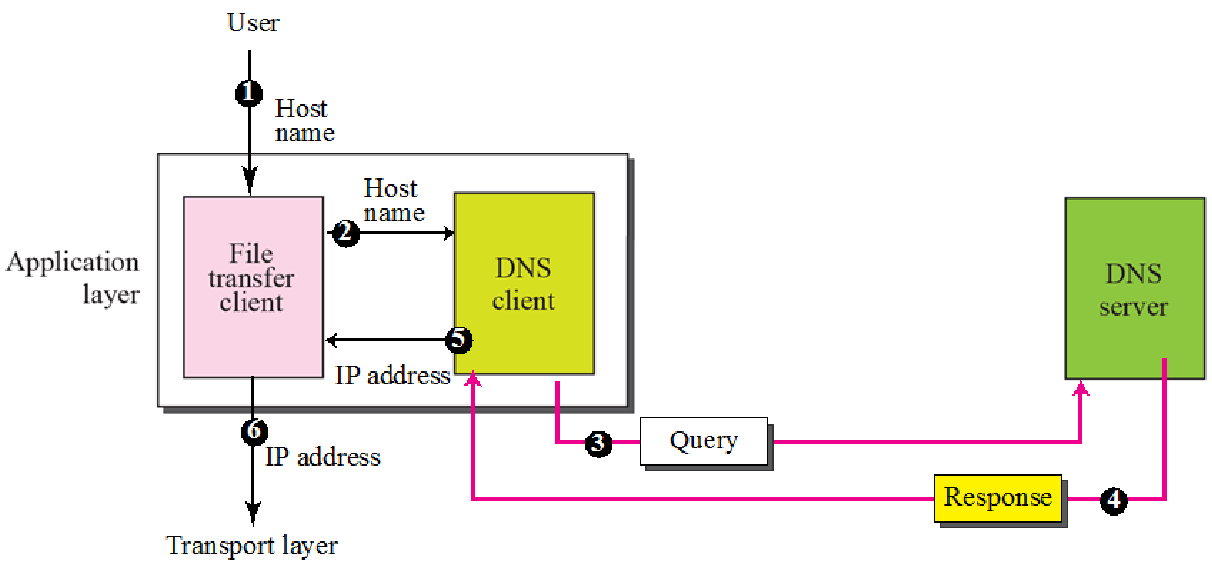
HTTP 1.1 behavior with persistent connections



DNS – Domain Name System

- To identify an entity (device), 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.
- Domain Name System can map a name to an address and conversely an address to name.
- DNS is a distributed database implemented in a hierarchy of name servers

Purpose of DNS



How does it work?

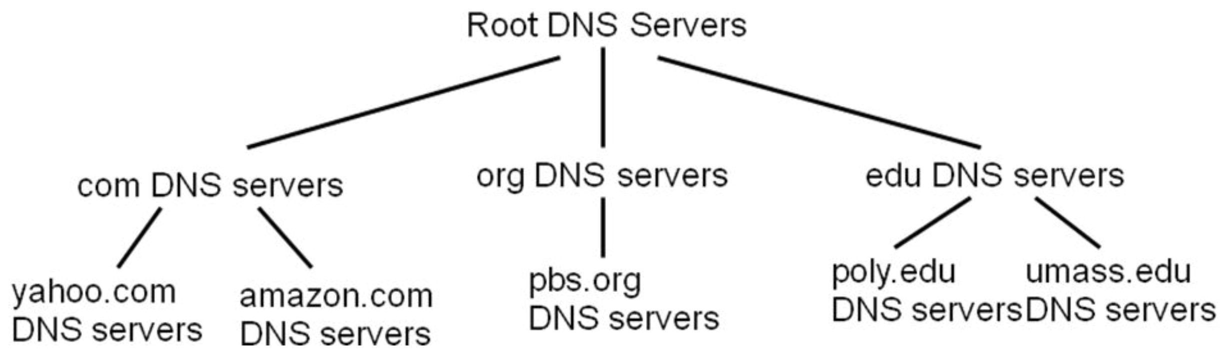
- DNS works by exchanging messages between client and server machines.
- A client application will pass the destination host name to the DNS process to get the IP address.
- Then the DNS client sends a query to the DNS Server to get the IP address of the corresponding host name.
- The DNS server replies the IP address through a response to the DNS client.
- The DNS client gives the IP address to the client application.

Why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance
- doesn't *scale!*

So use Distributed, Hierarchical Database

Distributed, Hierarchical Database



Client wants IP for www.amazon.com; 1st approx:

- client queries a 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

DNS Components

There are 3 components:

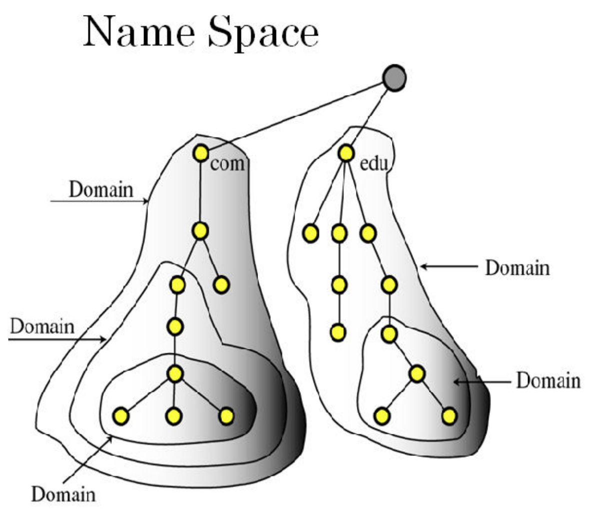
- Name Space: Specifications for a structured name space and data associated with the names
- Name Servers: Server programs which hold information about the structure and the names.
- Resolvers: Client programs that extract information from Name Servers.

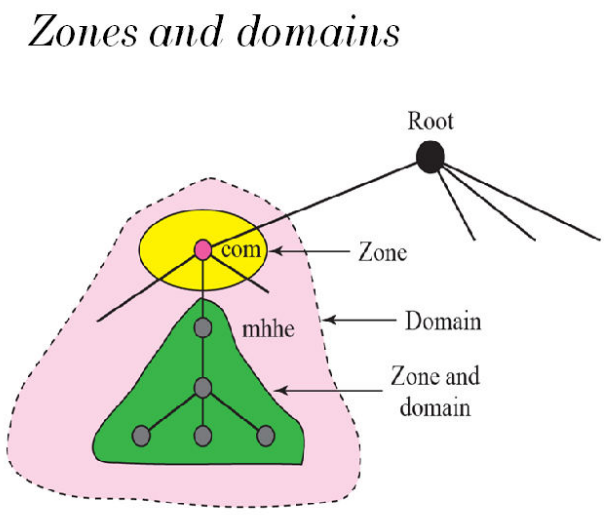
Name space

- To be unambiguous, the names assigned to machines must be carefully selected from a name space with complete control over the binding between the names and IP addresses. In other words, the names must be unique because the addresses are unique.
- A name space that maps each address to a unique name can be organized in two ways:

  flat or hierarchical.

  - A name space can be either
  - flat (names are not divisible into components)
  - hierarchical (Unix file names are an obvious example).



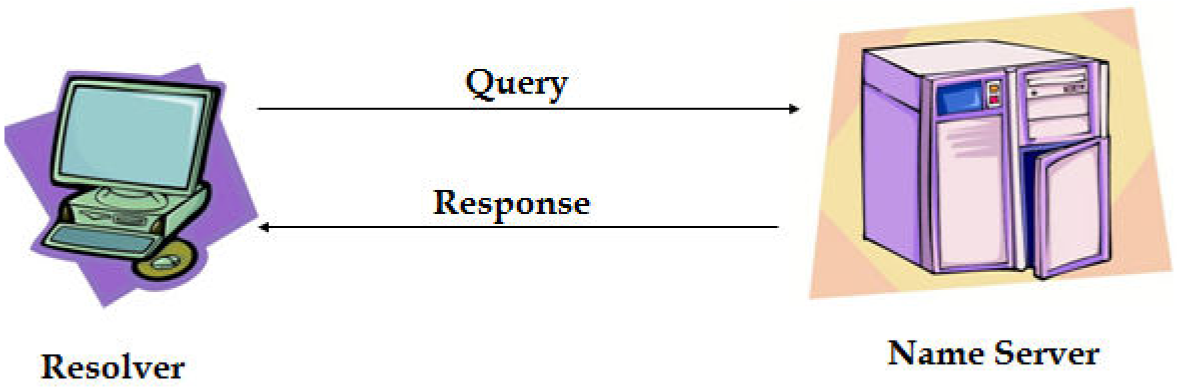


- ***The naming system (Name servers)*** maintains a collection of *bindings of names(URL) to values(IP address). The value can be anything we* want the naming system to return when presented with a name; in many cases it is an address.
- ***Resolution mechanism (Resolver)*** *is a procedure that, when invoked with a* name, returns the corresponding value. A *name server implements a specific* resolution mechanism that is available on a network and that can be queried by sending it a message.

<u>Resolvers</u>

A Resolver maps a name to an address and vice versa.

Resolvers ask the questions to the DNS system on behalf of the application.



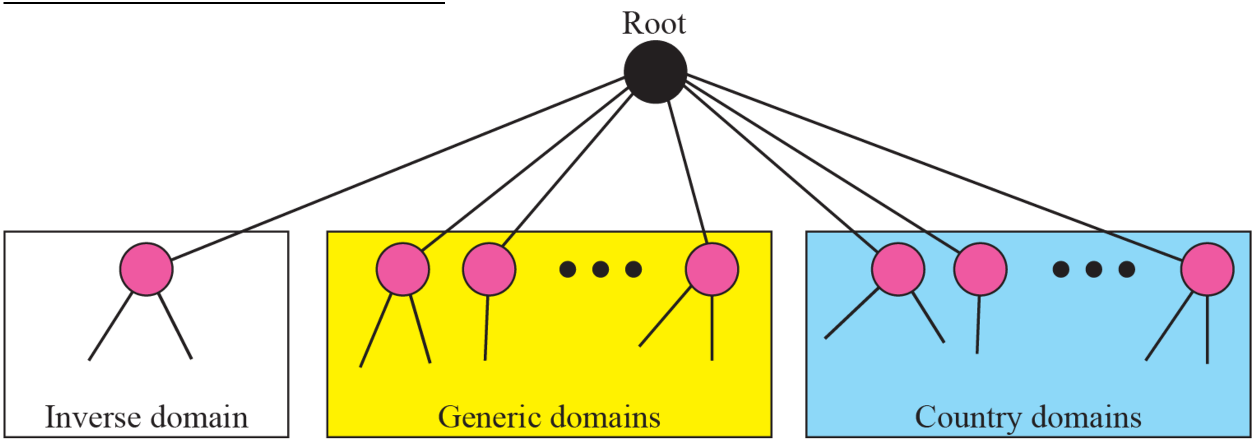
<u>DNS in The Internet</u>

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain

Three domains are,

1. Generic Domains
2. Country Domains
3. Inverse Domain

DNS used in the Internet



Generic domains

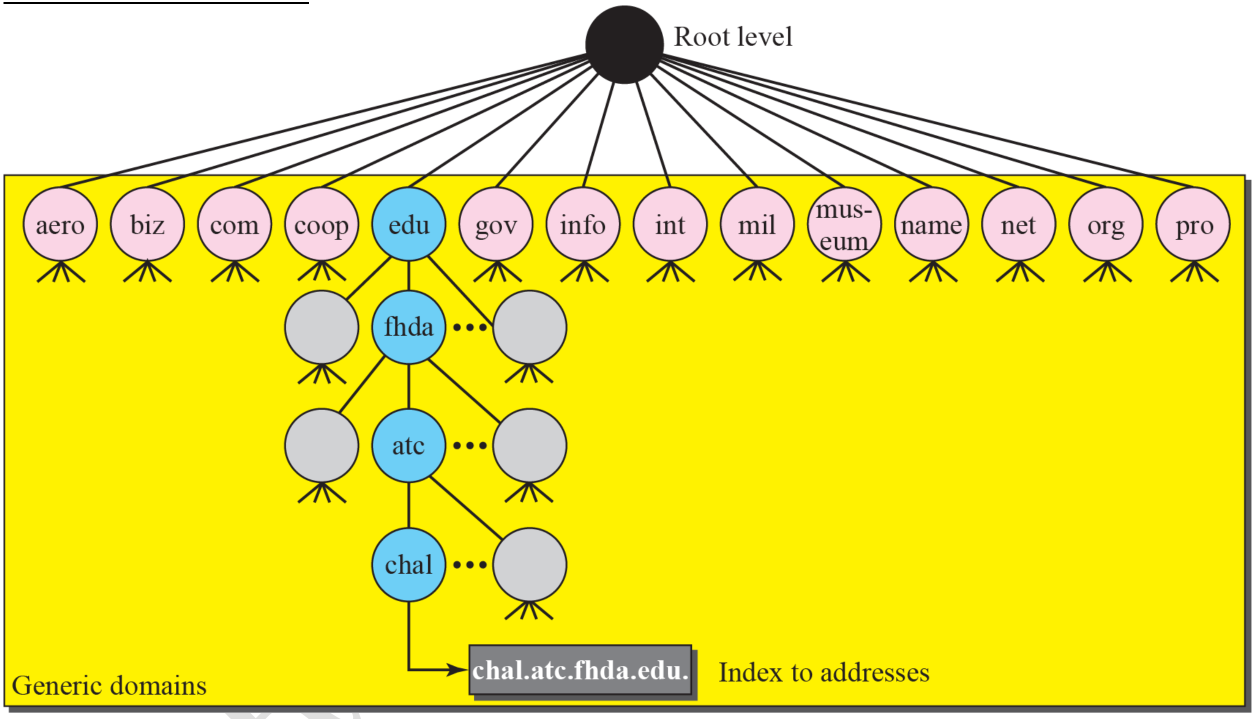
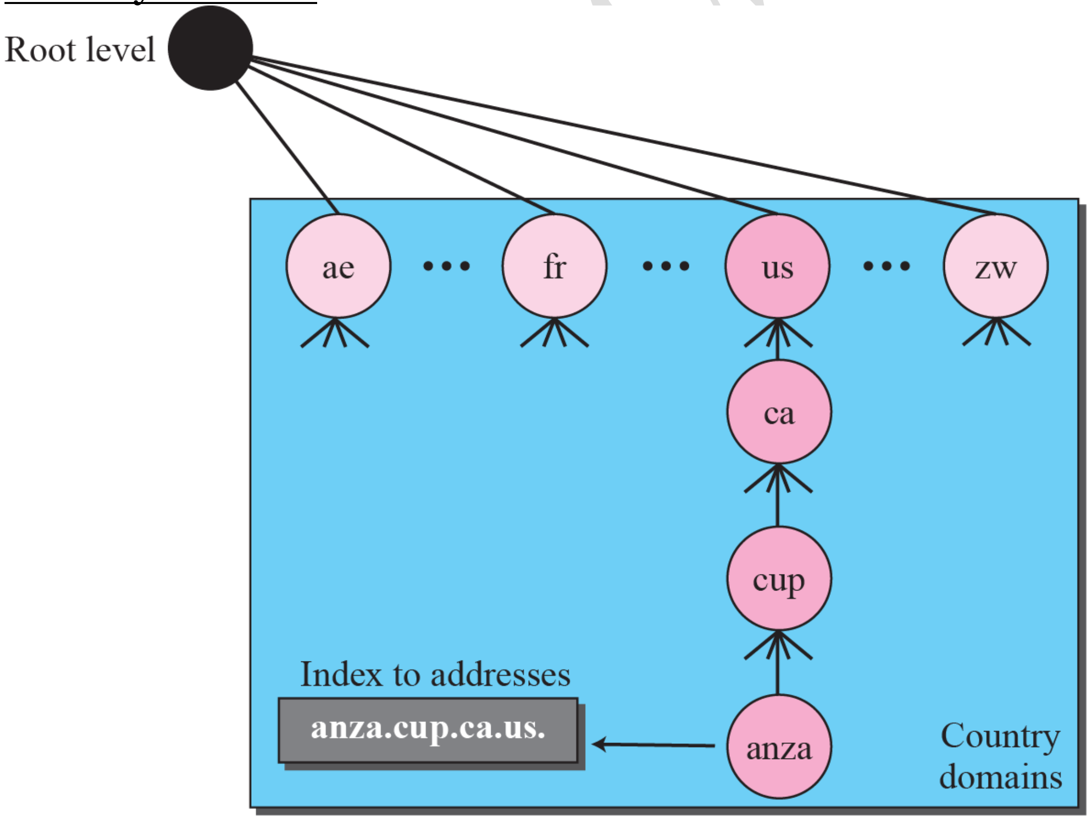


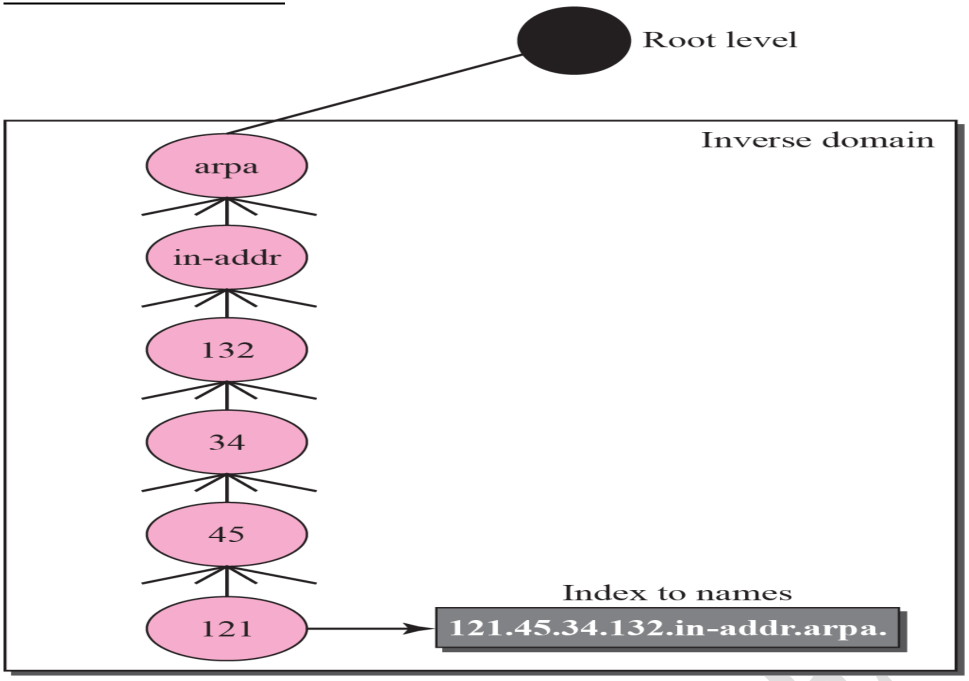
Table 19.1 *Generic domain labels*

Label	*Description*
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to "com")
com	Commercial organizations
coop	Cooperative business organizations
edu	Educational institutions
gov	Government institutions
info	Information service providers
int	International organizations
mil	Military groups
museum	Museums and other non-profit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

Country domains



Inverse domain



RESOLUTION

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

DNS MESSAGES

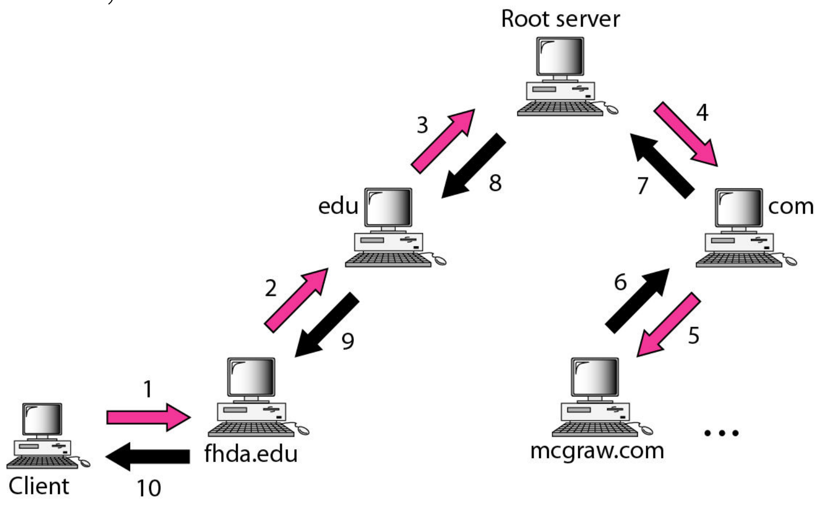
DNS has two types of messages:

1. Query messages
2. Response messages

Query messages can be of two formats

a. Recursive:

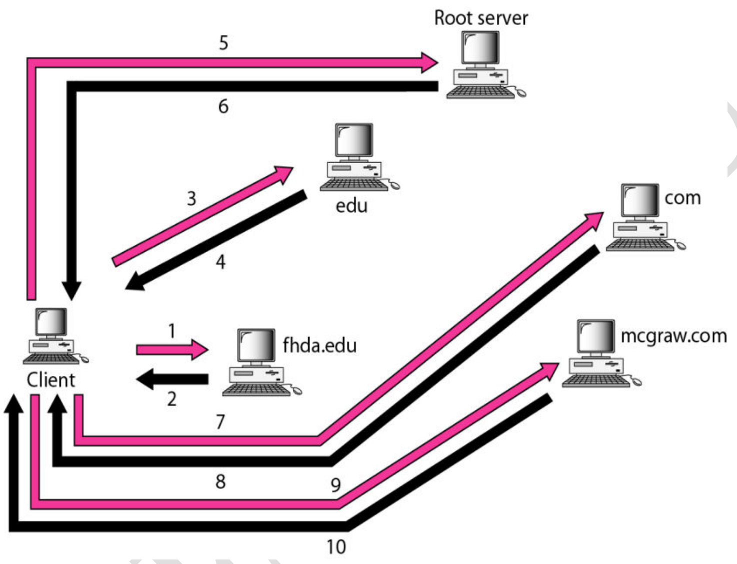
A recursive query is a kind of query, in which the DNS server, who received your query will do all the job of fetching the answer, and giving it back to you. During this process, the DNS server might also query other DNS server's in the internet on your behalf, for the answer.



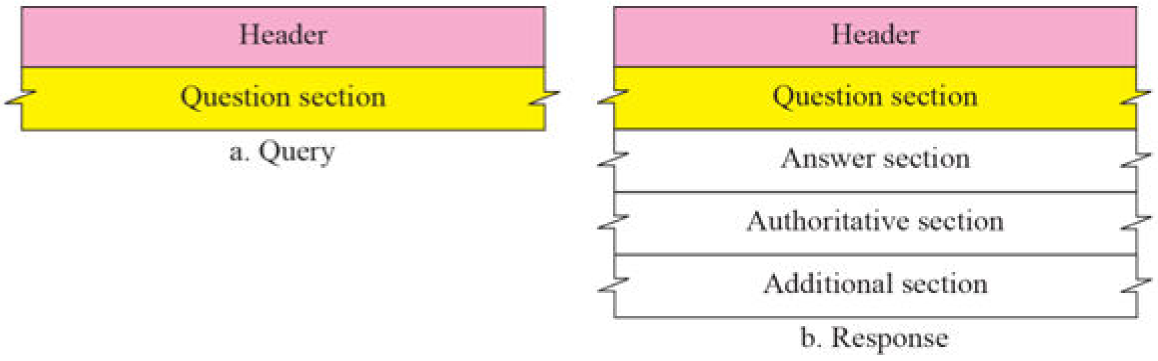
b. Iterative:

In an iterative query, the name server, will not go and fetch the complete answer for your query, but will give back a referral to other DNS server's, which might have the answer.

If the DNS server is not a recursive name server(which means its iterative), it will give us the answer if it has in its records. Otherwise will give us the referral to the root servers(it will not query the root server's and other servers by itself.).Now its the job of our resolver to query the root server, .COM TLD servers, and authoritative name server's, for the answer.



Query and response messages format



WEB SERVICES

- Much of the motivation for enabling direct application-to-application communication comes from the business world.
- Historically, interactions between enterprises—businesses or other organizations—have involved some manual steps such as filling out an order form or making a phone call to determine whether some product is in stock.
- Even within a single enterprise it is common to have manual steps between software systems that cannot interact directly because they were developed independently.
- Increasingly such manual interactions are being replaced with direct application-to application interaction.
- An ordering application at enterprise A would send a message to an order fulfillment application at enterprise B, which would respond immediately indicating whether the order can be filled.
- Perhaps, if the order cannot be filled by B, the application at A would immediately order from another supplier, or solicit bids from a collection of suppliers.
- Two architectures have been advocated as solutions to this problem.
- Both architectures are called *Web Services,* taking their name from the term for the individualapplications that offer a remotely-accessible service to client applications to form network applications.
- The terms used as informal shorthand to distinguish the two Web Services architectures are *SOAP and REST (as in, "the SOAP vs. REST debate").*
- The SOAP architecture's approach to the problem is to make it feasible, at least in theory, to generate protocols that are customized to each network application.
- The key elements of the approach are a framework for protocol specification, software toolkits for automatically generating protocol implementations from the specifications, and modular partial specifications that can be reused across protocols.

Custom Application Protocols (WSDL, SOAP)

- The architecture informally referred to as SOAP is based on *Web Services Description Language (WSDL) and SOAP.4*
- Both of these standards are issued by the World Wide Web Consortium (W3C).
- This is the architecture that people usually mean when they use the term Web Services without any preceding qualifier.
- Just like the traditional applications described earlier in this chapter, multimedia applications such as telephony and videoconferencing need their own protocols.
- We have already seen a number of protocols that multimedia applications use.

–The Real-Time Transport Protocol (RTP) provides many of the functions that are common to multimedia applications such as conveying timing information and identifying the coding schemes and media types of an application.

Defining Application Protocols

WSDL has chosen a procedural operation model of application protocols. An abstract Web Service interface consists of a set of named operations, each representing a simple interaction between a client and theWeb Service An operation is analogous to a remotely callable procedure in an RPC system. An example from W3C's WSDL Primer is a hotel reservation Web Service with two operations, CheckAvailability and MakeReservation.

- Each operation specifies a Message Exchange Pattern (MEP) that gives the sequence in which the messages are to be transmitted, including the fault messages to be sent when an error disrupts the message flow.
- MEPs are templates that have placeholders instead of specific message types or formats, so part of the definition of an operation involves specifying which message formats to map into the placeholders in the pattern.
- WSDL nicely separates the parts of a protocol that can be specified abstractly—operations, MEPs, abstract message formats—from the parts that must be concrete. WSDL's concrete part specifies an underlying protocol, how MEPs are mapped onto it, and what bit-level representation is used for messages on the wire.

Defining Transport Protocols

Although SOAP is sometimes called a protocol, it is better thought of as a framework for defining protocols. As the SOAP 1.2 specification explains, "SOAP provides a simple messaging framework whose core functionality is concerned with providing extensibility." SOAP uses many of the same strategies as WSDL, including message formats defined using XML Schema, bindings to underlying protocols, Message Exchange Patterns, and reusable specification elements identified using XML namespaces.

- SOAP is used to define transport protocols with exactly the features needed to support a particular application protocol. SOAP aims to make it feasible to define many such protocols by using reusable components. Each component captures the header information and logic that go into implementing a particular feature. To define a protocol with a certain set of features, just compose the corresponding components. Let's look more closely at this aspect of SOAP.

– The second and more flexible way to implement features involves header blocks. A SOAP message consists of an Envelope, which contains a Header that contains header blocks, and a Body, which contains the payload destined for the ultimate receiver. This message structure is illustrated in Figure

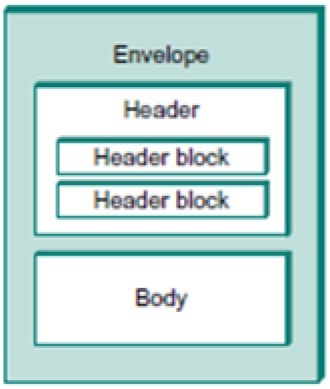


Figure : SOAP message structure

<u>Standardizing Web Services Protocols:</u>

– As we've said, WSDL and SOAP aren't protocols; they are standards for specifying protocols. For different enterprises to implementWeb Services that interoperate with each other, it is not enough to agree to use WSDL and SOAP to define their protocols; they must agree on—standardize— specific protocols.

– The broadest and most widely adopted profile is known as the WS-I Basic Profile. It was proposed by the Web Services Interoperability Organization

(WS-I), an industry consortium, while WSDL and SOAP are specified by the World Wide Web Consortium (W3C). The Basic Profile resolves some of the most basic choices faced in defining a Web Service Most notably it requires that WSDL be bound exclusively to SOAP and SOAP be bound exclusively to HTTP and use the HTTP POST method. It also specifies which versions of WSDL and SOAP must be used.

– The payload is a representation of the abstract state of a resource. For example, a GET could return a representation of the current state of the resource, and a POST could send a representation of a desired state of the resource.

<u>A Generic Application Protocol (REST):</u>

– The WSDL/SOAP Web Services architecture is based on the assumption that the best way to integrate applications across networks is via protocols that are customized to each application.

– This model, articulated byWeb architect Roy Fielding, is known as Representational State Transfer (REST). There is no need for a new REST architecture for Web Services—the existing Web architecture is sufficient, although a few extensions are probably necessary.
– An area where WSDL/SOAP may have an advantage is in adapting or wrapping previously written, "legacy" applications to conform to Web Services. This is an important point since most Web Services will be based on legacy applications for the near future at least. These applications usually have a procedural interface that maps more easily into WSDL's operations than REST states.
– The online retailer Amazon.com, as it happens, was an earlyadopter (2002) of Web Services. Interestingly, Amazon made its systems publicly accessible via *both* of the Web Services chitectures, and according to some reports a substantial majority of developers use the REST interface. Of course, this is just one data point and may well reflect factors specific to Amazon.

FILE TRANSFER PROTOCOL (FTP)

Transferring files from one computer to another is one of the most common tasks expected from a networking or internetworking environment. As a matter of fact, the greatest volume of data exchange in the Internet today is due to file transfer.

✓ File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another.

✓ Although transferring files from one system to another seems simple and straightforward, some problems must be dealt with first. For example, two systems may use different file name conventions.

✓ Two systems may have different ways to represent text and data. Two systems may have different directory structures. All these problems have been solved by FTP in a very simple and elegant approach

✓ FTP differs from other client/server applications in that it establishes two connec tions between the hosts. One connection is used for data transfer, the other for control information (commands and responses). Separation of commands and data transfer makes FTP more efficient.

✓ The control connection uses very simple rules of communi cation. Wc need to transfer only a line of command or a line of response at a time. The data connection, on the other hand, needs more complex rules due to the variety of data types transferred. However, the difference in complexity is at the FTP level, not TCP. For TCP, both connections are treated the same.

- **FTP uses the services of TCP. It needs two TCP connections.**
- **The well-known port 21 is used for the control connection and the well-known port 20 for the data connection.**

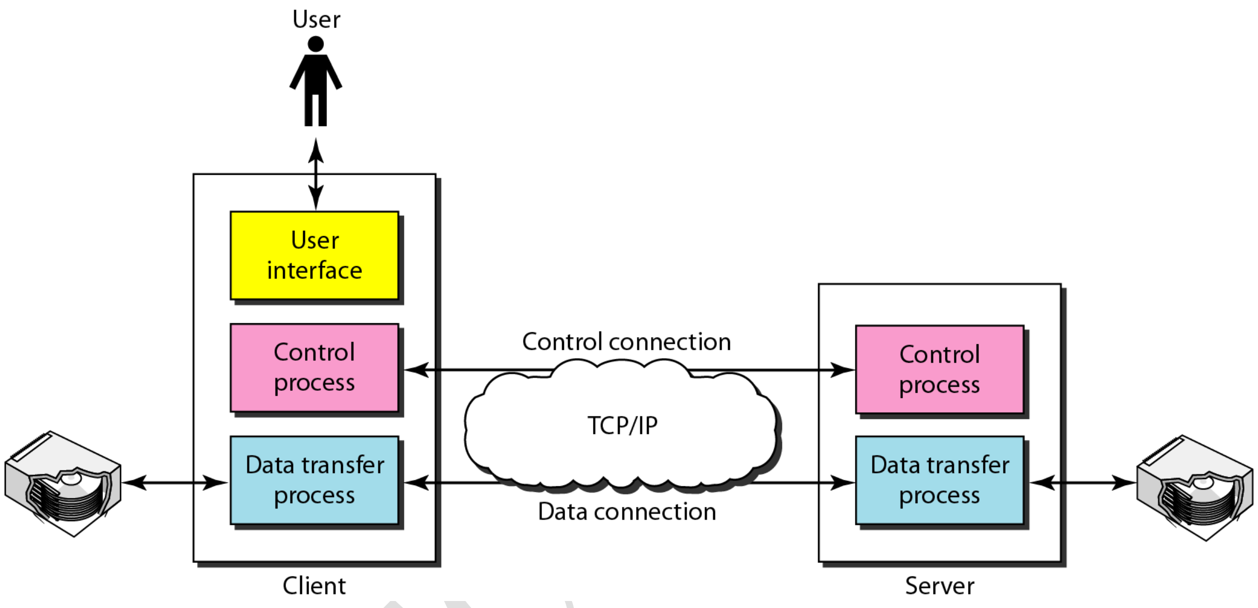


Fig: FTP

<u>Communication over Control Connection</u>

✓ FTP uses the same approach as SMTP to communicate across the control connection. It uses the 7-bit ASCII character set. Communication is achieved through commands and responses.

✓ This simple method is adequate for the control connection because we send one command (or response) at a time.

✓ Each command or response is only one short line, so we need not worry about file format or file structure. Each line is terminated with a two-character (carriage return and line feed) end-of-line token.

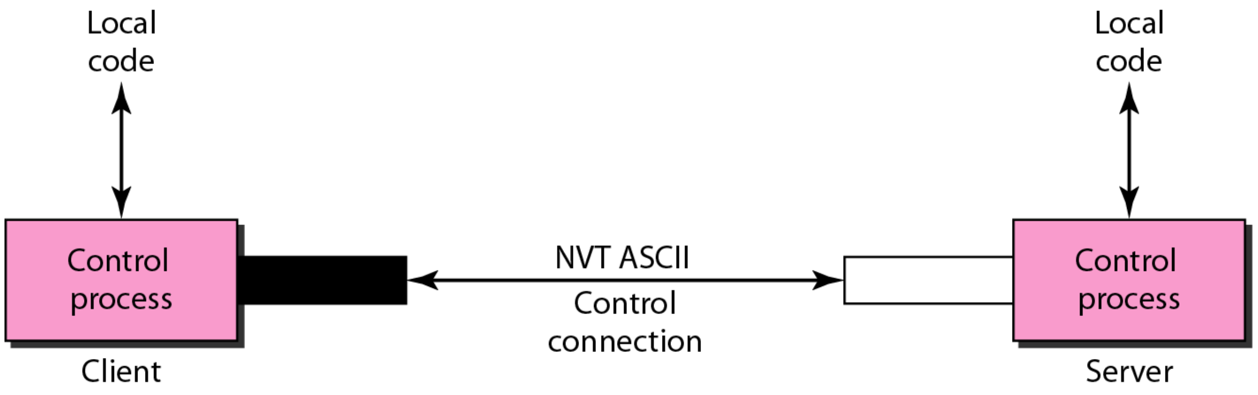


Fig: ***Using the control connection***

Communication over Data Connection

- ✓ **The purpose of the data connection is different from that of the control connection. We want to transfer files through the data connection.**
- ✓ **File transfer occurs over the data connection under the control of the commands sent over the control connection. However, we should remember that file transfer in FTP means one of three things.**

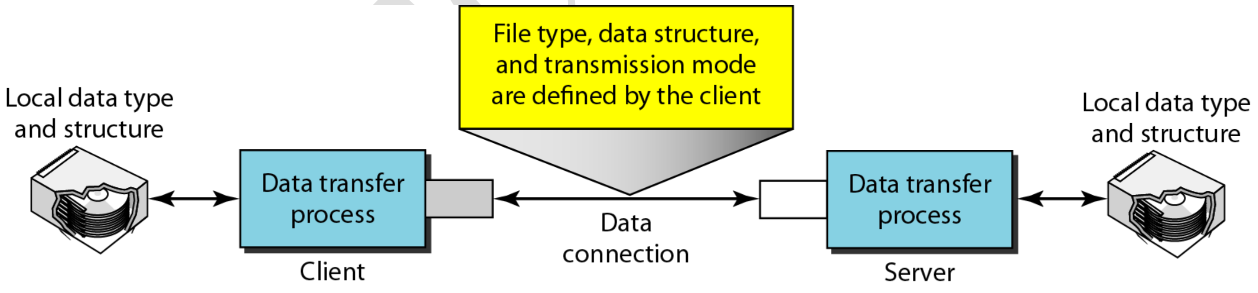


Fig: Using the data connection

- ➢ A file is to be copied from the server to the client. This is called retrieving a file. It is done under the supervision of the RETR command,
- ➢ A file is to be copied from the client to the server. This is called storing a file. It is done under the supervision of the STOR command.

- A list of directory or file names is to be sent from the server to the client. This is done under the supervision of the LIST command. Note that FTP treats a list of directory or file names as a file. It is sent over the data connection.

Example

The following shows an actual FTP session for retrieving a list of items in a directory. The colored lines show the responses from the server control connection; the black lines show the commands sent by the client. The lines in white with a black background show data transfer.

1. After the control connection is created, the FTP server sends the 220 response.

2. The client sends its name.

3. The server responds with 331.

4. The client sends the password (not shown).

5. The server responds with 230 (user log-in is OK).

6. The client sends the list command (ls reports) to find the list of files on the directory named report.

7. Now the server responds with 150 and opens the data connection.

8. The server then sends the list of the files or directories on the data connection.

9. The client sends a QUIT command.

10. The server responds with 221.

```
$ ftp voyager.deanza.fhda.edu
Connected to voyager.deanza.fhda.edu.
220 (vsFTPd 1.2.1)
530 Please login with USER and PASS.
Name (voyager.deanza.fhda.edu:forouzan): forouzan
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls reports
227 Entering Passive Mode (153,18,17,11,238,169)
150 Here comes the directory listing.
drwxr-xr-x    2 3027      411           4096 Sep 24  2002 business
drwxr-xr-x    2 3027      411           4096 Sep 24  2002 personal
drwxr-xr-x    2 3027      411           4096 Sep 24  2002 school
226 Directory send OK.
ftp> quit
221 Goodbye.
```