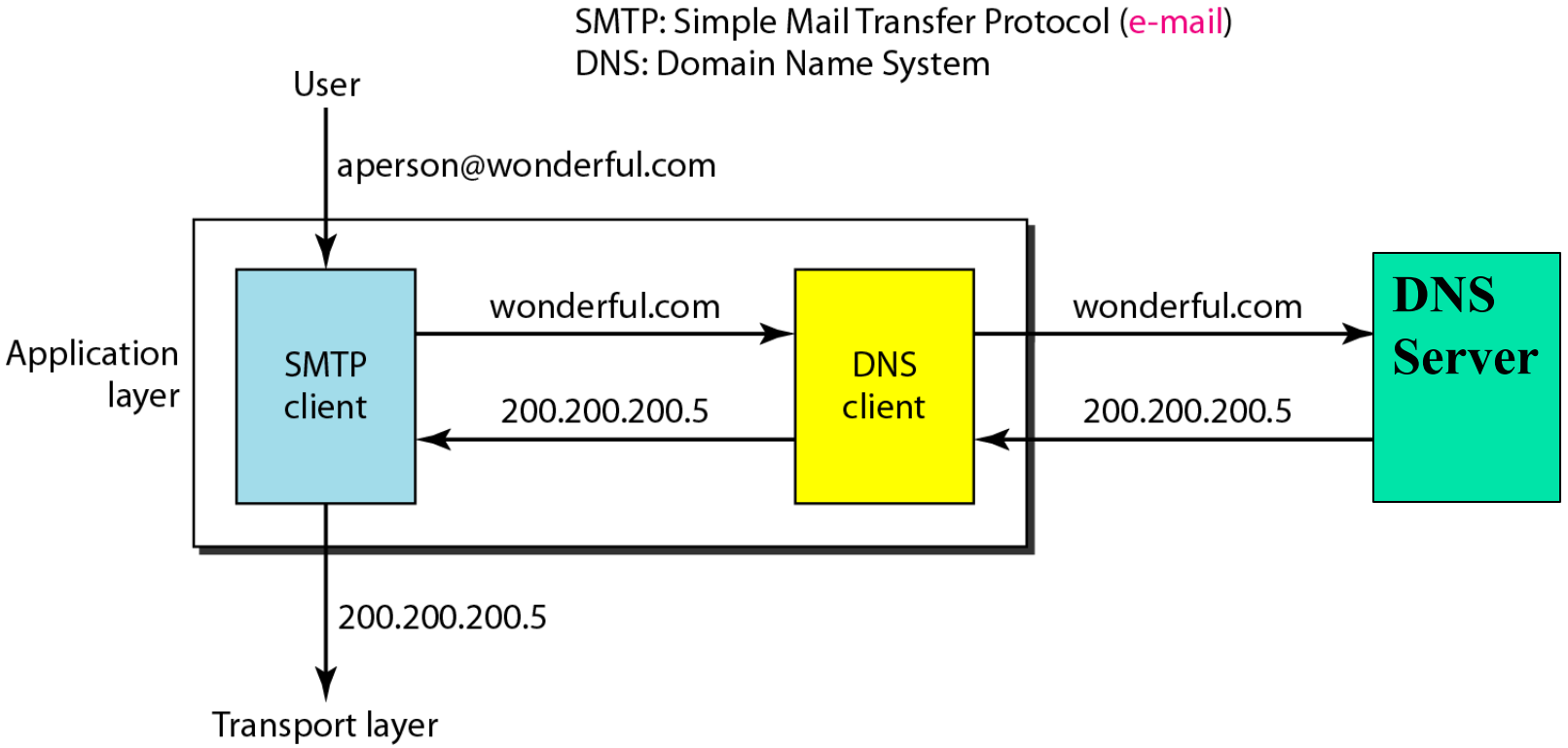
Domain Name System

Example of using the DNS service



Name Space

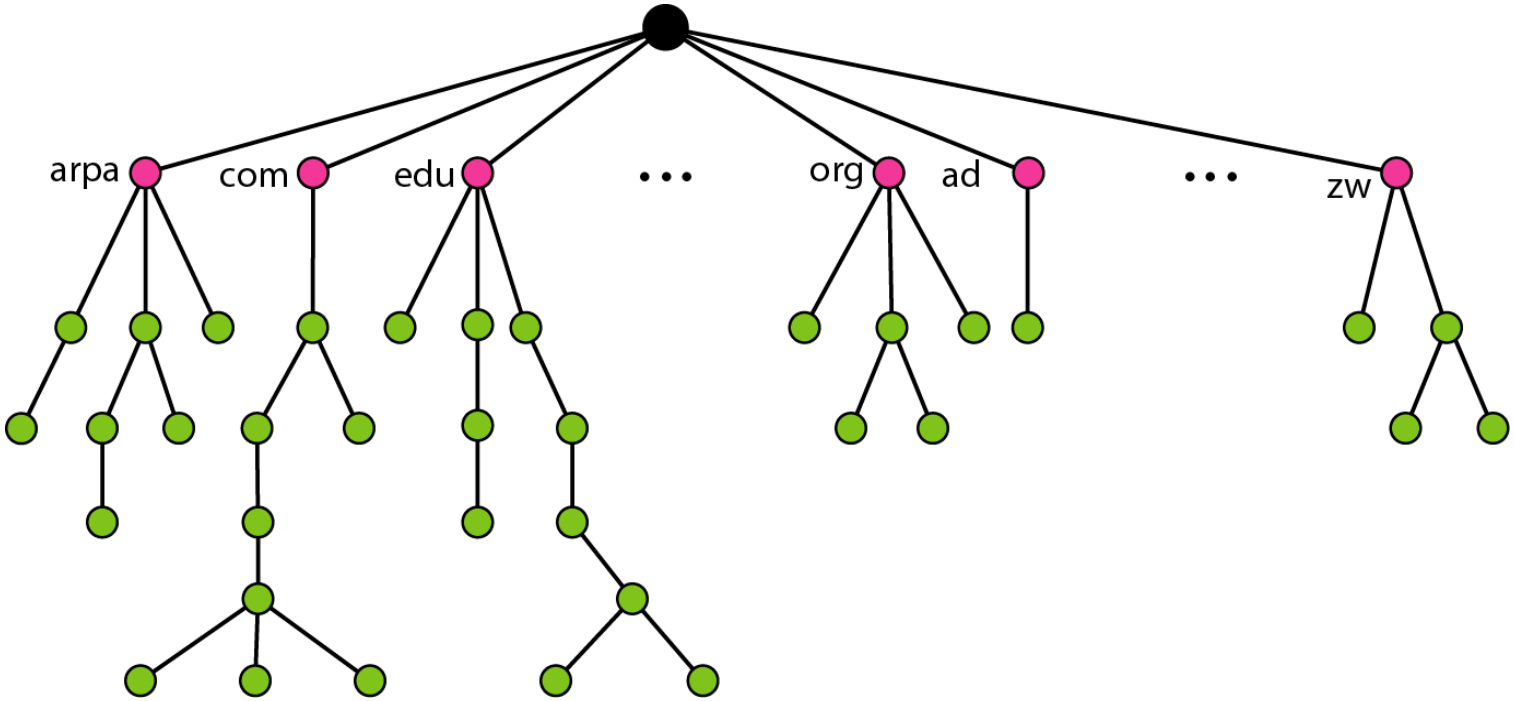
- To be unambiguous, the names assigned to machines must be carefully selected from a name space with complete control over the binding between the names and IP addresses.

- ***Name space can be organized in two ways:***
  - ***Flat name space***
    - ***Name is a sequence of character without structure***
    - ***Disadvantage: Can not be used for a large system***
  - ***Hierarchical name space***
    - ***Each name is made of several parts***

Domain Name Space

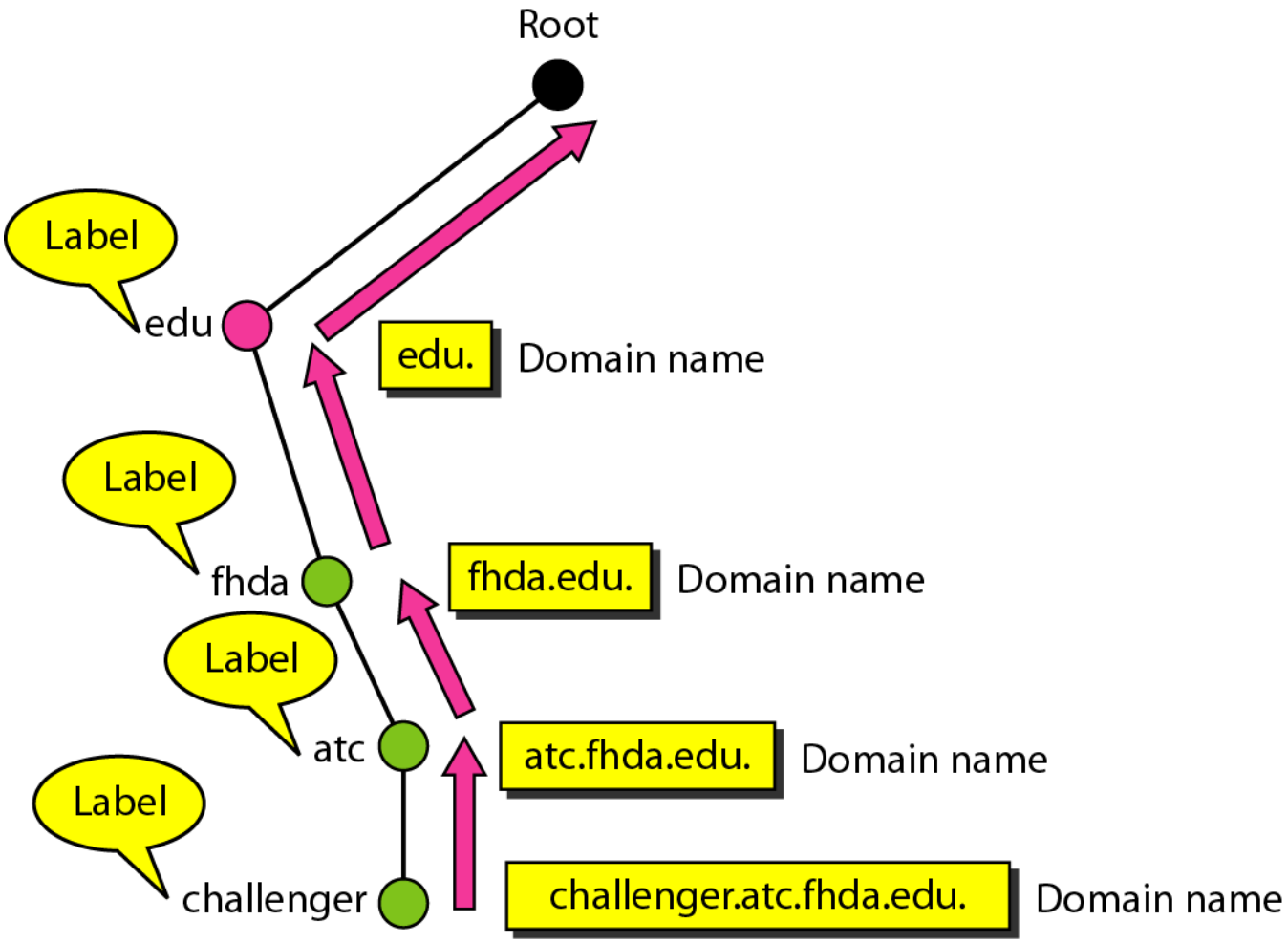
- To have a hierarchical name space, a domain name space was designed.
- In this design the names are defined in an *inverted-tree structure* with the root at the top.
- The tree can have only 128 levels: level 0 (root) to level 127.
- Each node in the tree has a label (max. 63 characters)
- The root label is a null string
- DNS requires that children of a node have different labels
  - **This guarantees the uniqueness of the domain name**

Domain name space



- Each node in the 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.
- The last label is the label of the root (null).

Domain names and labels



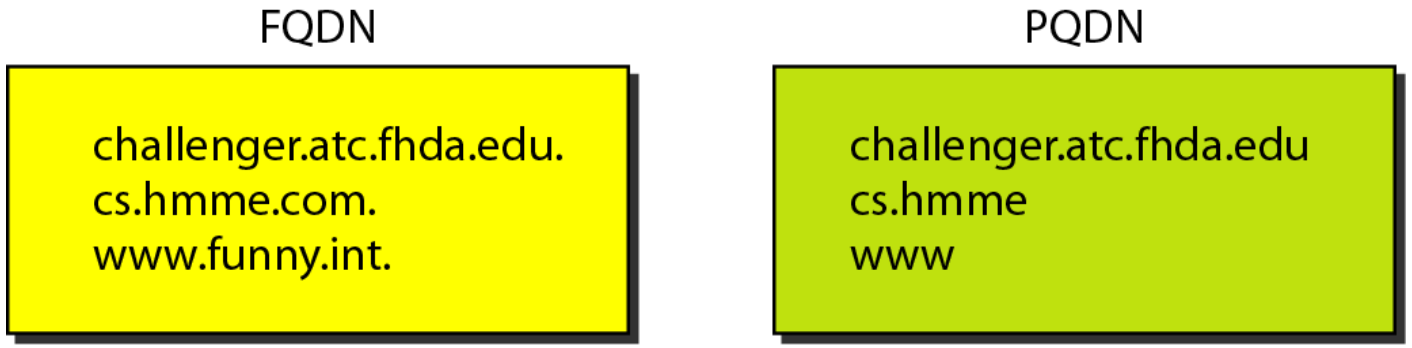
Fully Qualified Domain Name

- If a label is terminated by a null string, it is called a fully qualified domain name (FQDN).
- An FQDN is a domain name that contains the full name of a host.
- It contains all labels, from the most specific to the most general, that uniquely define the name of the host.
  - Example: *challenger.ate.tbda.edu.*
- A DNS server can only match an FQDN to an address.

Partially Qualified Domain Name

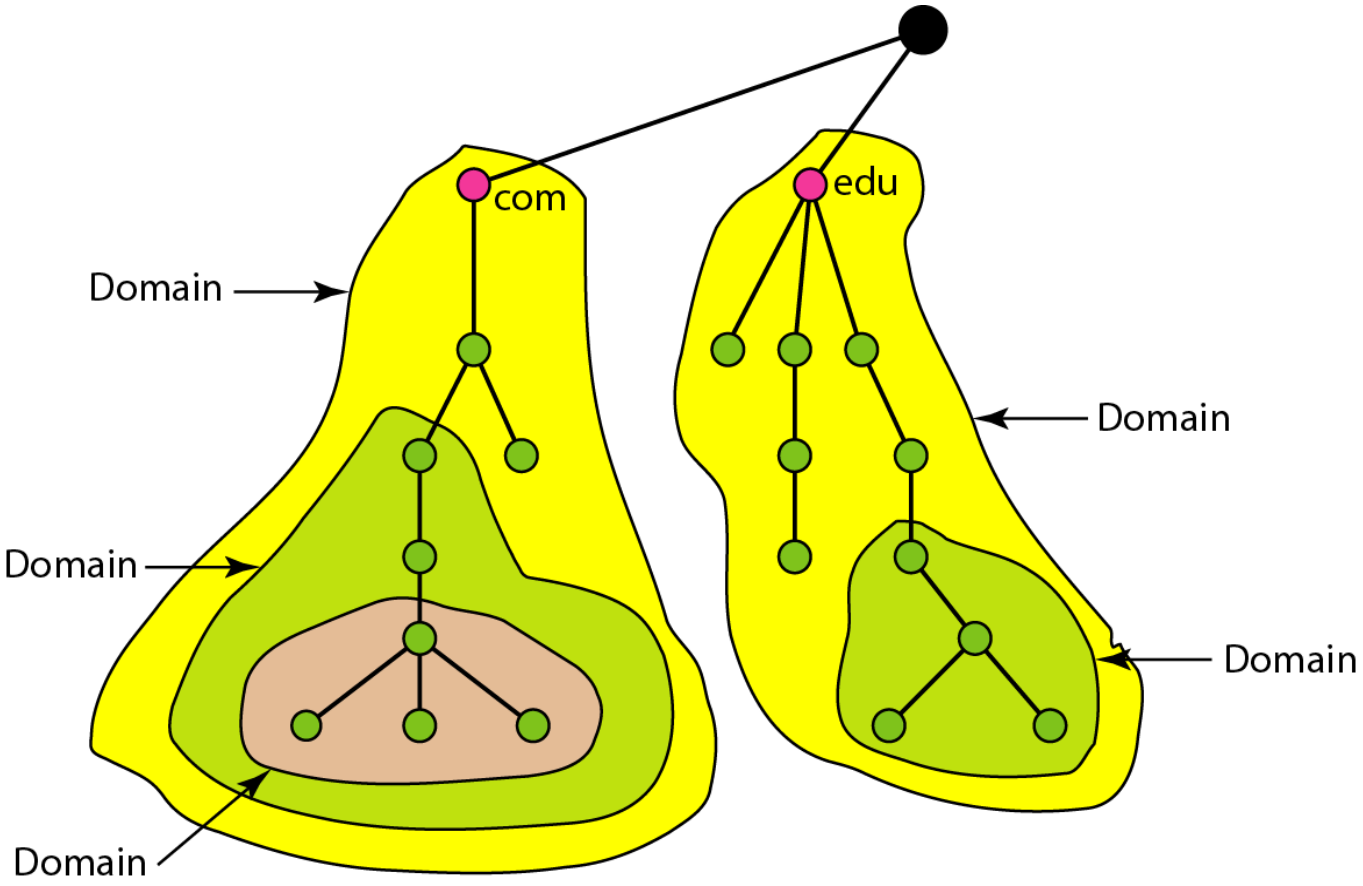
- If a label is not terminated by a null string, it is called a partially qualified domain name (PQDN).
- A PQDN starts from a node, but it does not reach the root.
- Here the resolver can supply the missing part, called the suffix, to create an FQDN.

FQDN and PQDN



Domains

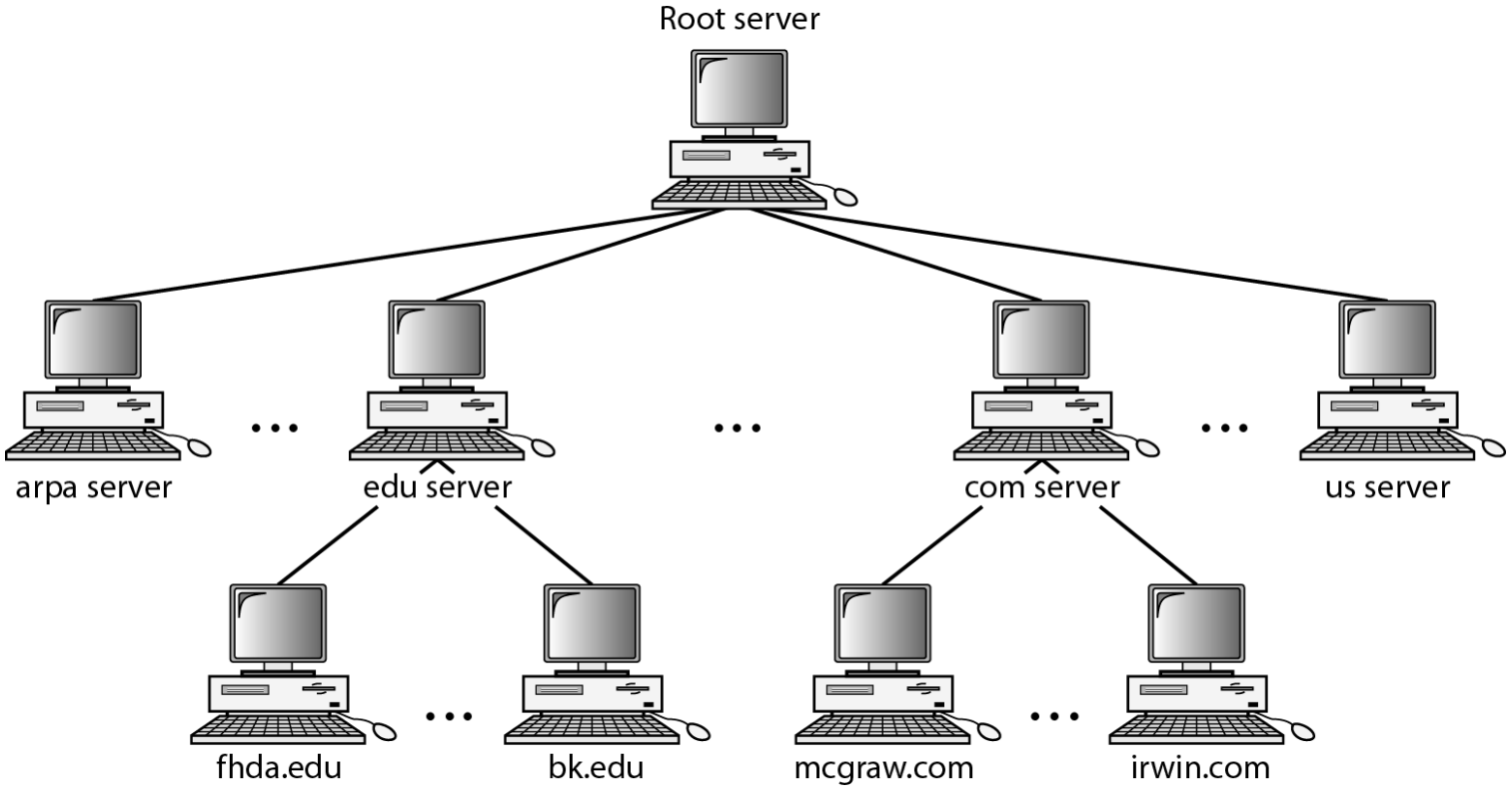
- A domain is a subtree of the domain name space.
- The name of the domain is the domain name of the node at the top of the subtree.



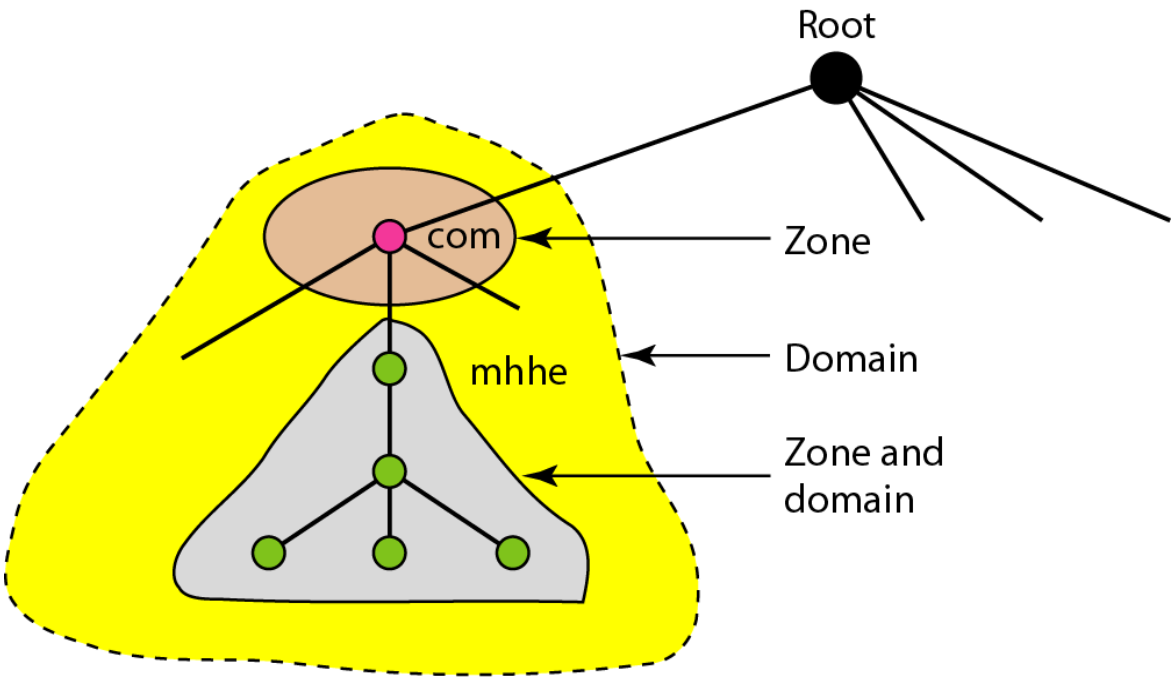
Distribution of Name Space

- The information contained in the domain name space must be stored.
- However, it is very inefficient and also unreliable to have just one computer store such a huge amount of information.

Hierarchy of name servers



Zones and domains



Root Server

- A root server is a server whose zone consists of the whole tree.
- A root server usually does not store any information about domains but delegates its authority to other servers, keeping references to those servers.
- There are several root servers, each covering the whole domain name space.
- The servers are distributed all around the world.

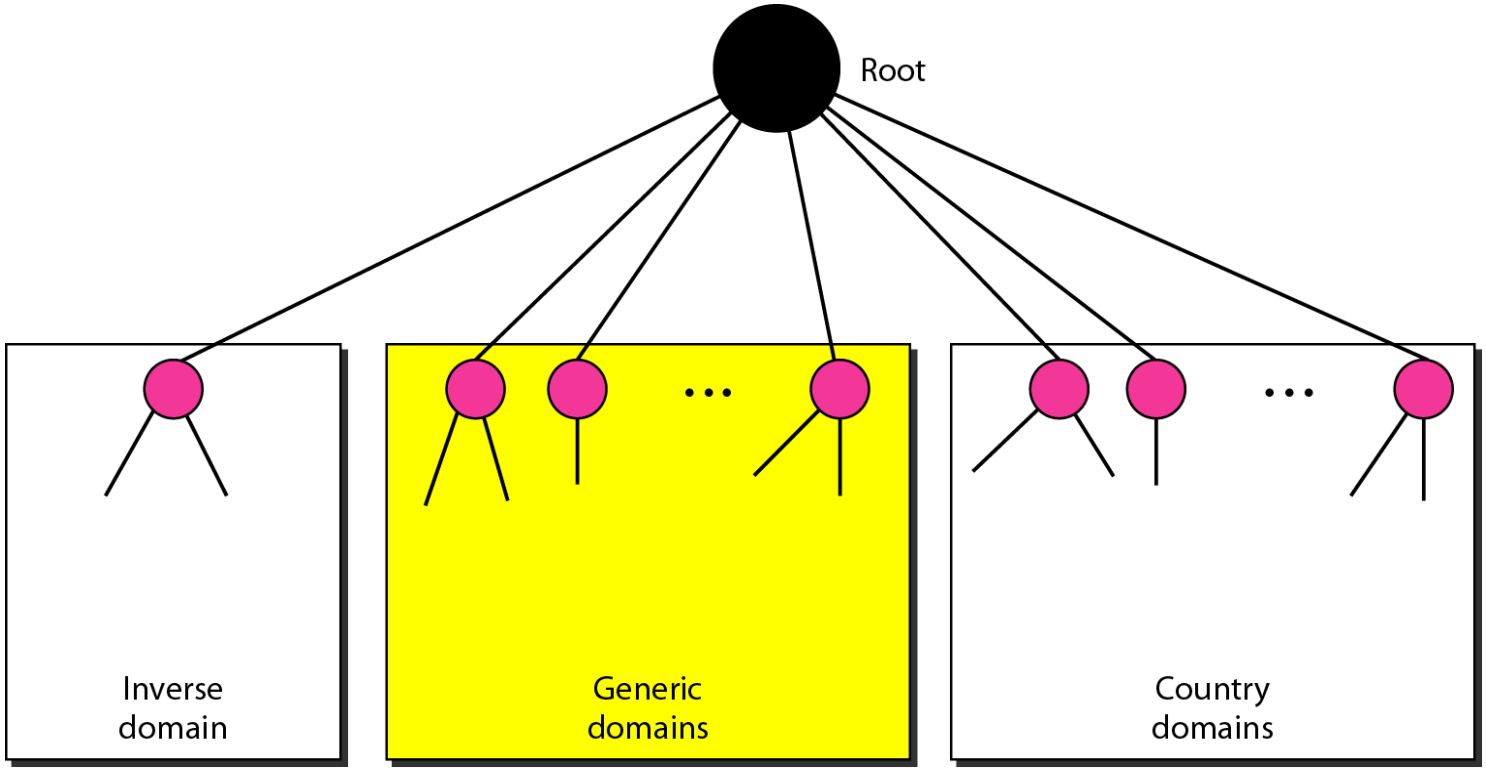
Primary and Secondary Servers

- **Primary server:**
  - It is a server that stores a file about the zone for which it is an authority.
  - It is responsible for ***creating, maintaining, and updating the zone file.*** It stores the zone file on a local disk.
- **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.
  - The secondary server neither creates nor updates the zone files.

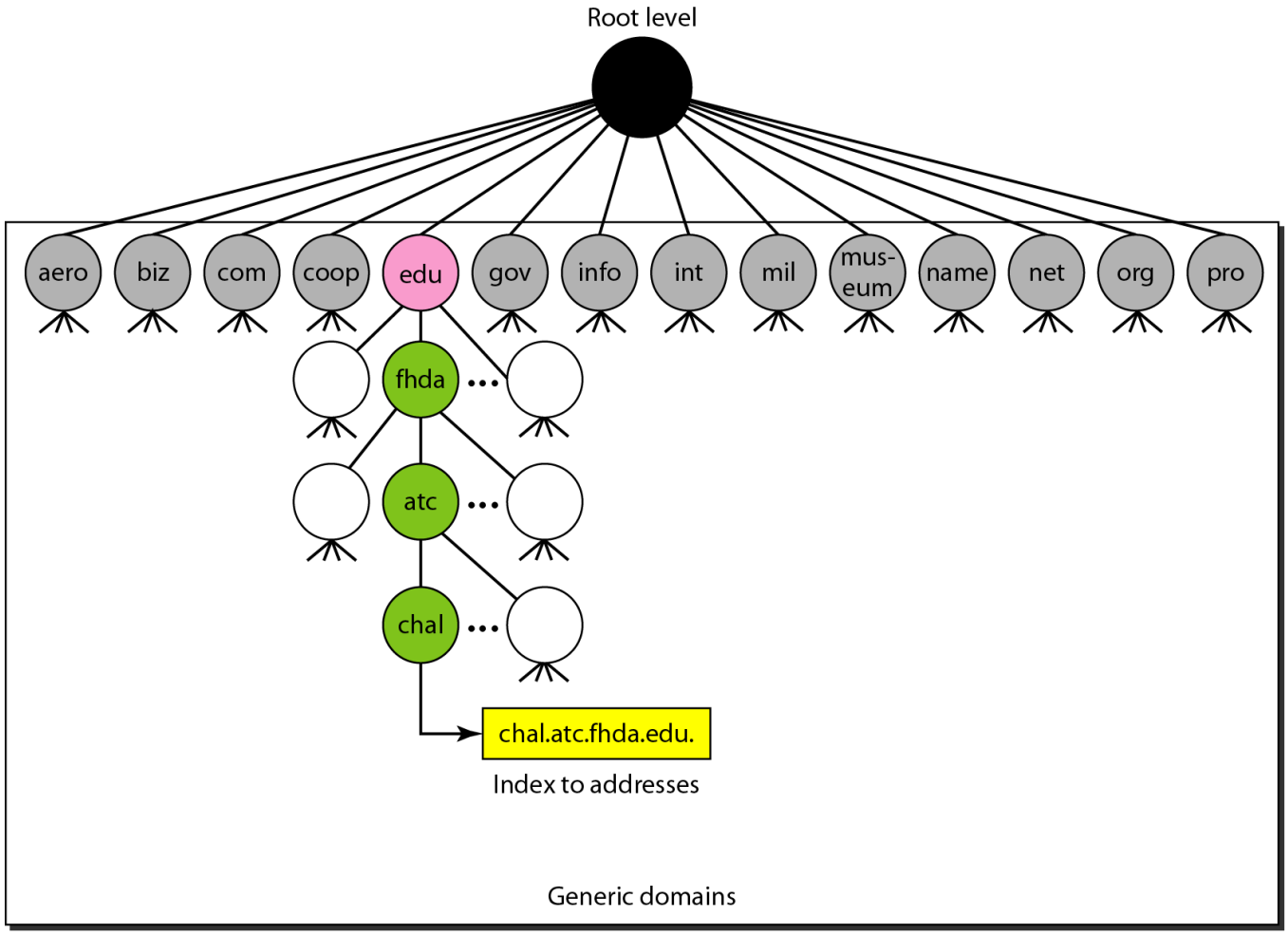
DNS In The Internet

- 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.

DNS IN THE INTERNET



Generic domains

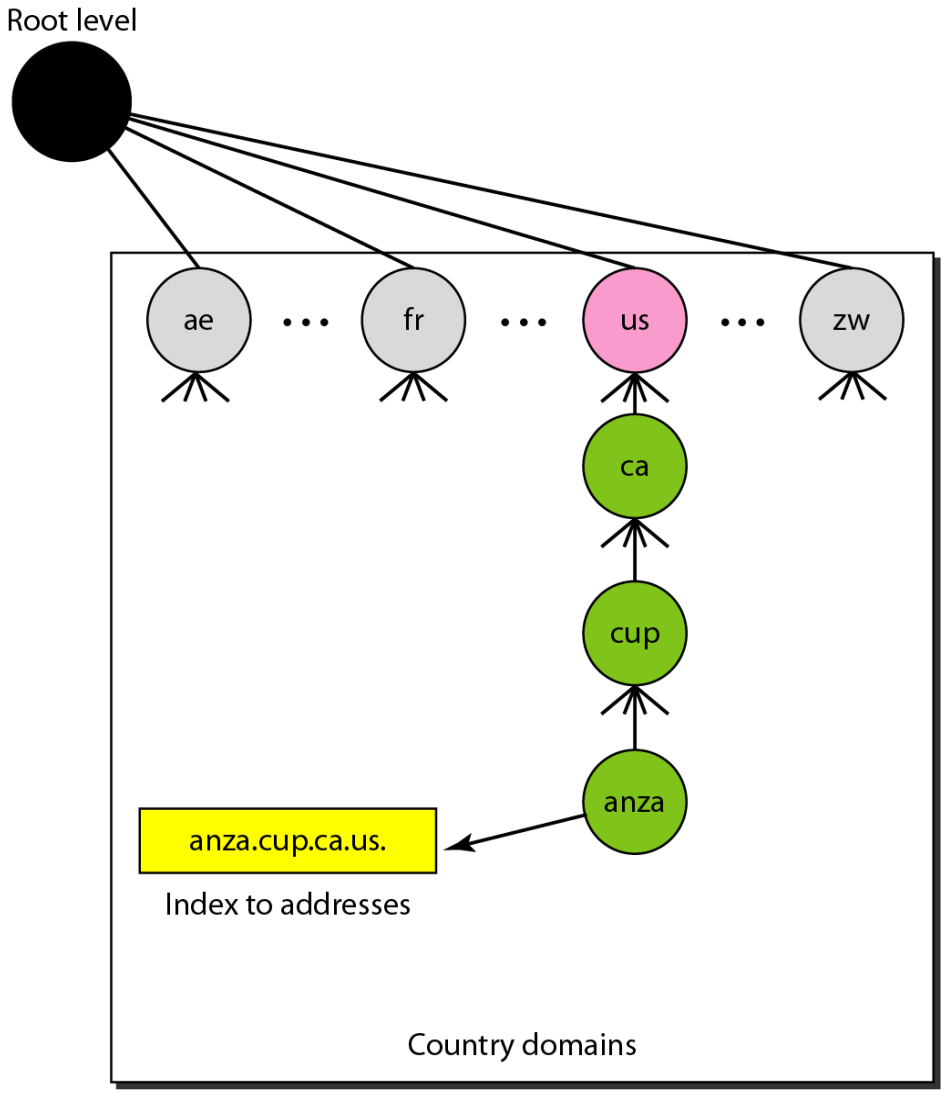


- The generic domains define registered hosts according to their generic behaviour.

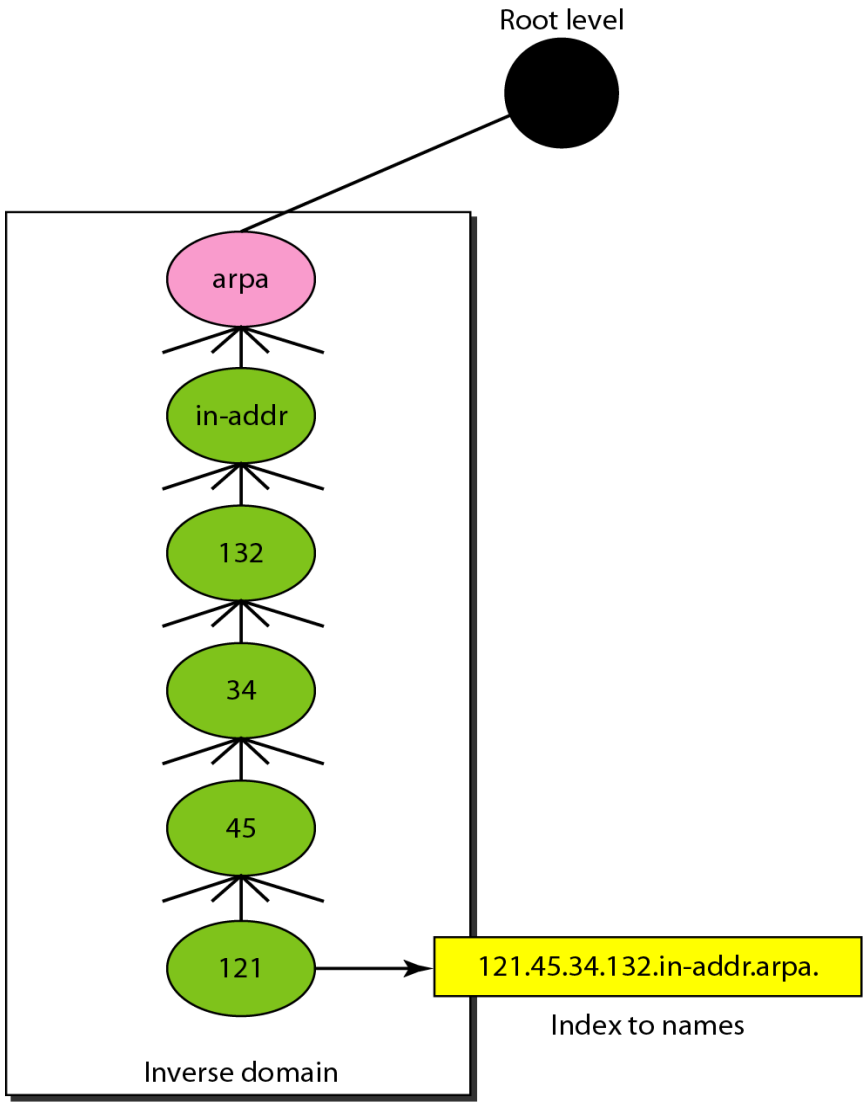
Generic domain labels

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 nonprofit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

Country domains



Inverse domain



Resolution

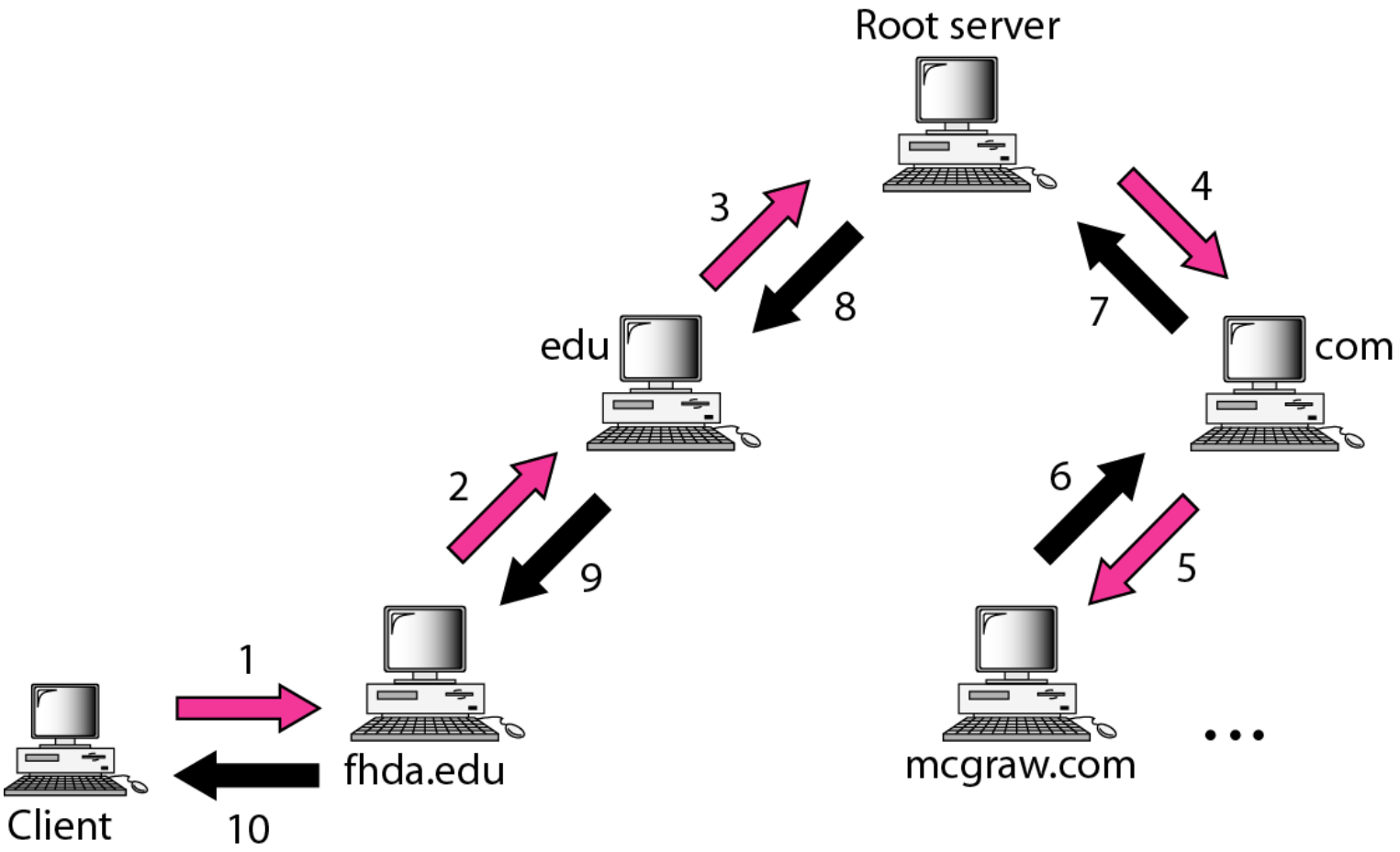
- Mapping a name to an address or an address to a name is called *name-address resolution*.

- **Resolver**
  - DNS is designed as a client/server application.
  - A host that needs to map an address to a name or a name to an address calls a **DNS client called a resolver.**
  - The resolver accesses the closest DNS server with a mapping request.
  - If the server has the information, it satisfies the resolver; otherwise, it either refers the resolver to other servers or asks other servers to provide the information.

Recursive Resolution

- The resolver expects the server to supply the final answer.
- If the server is the authority for the domain name, it checks its database and responds.
- If the server is not the authority, it sends the request to another server (the parent usually) and waits for the response.
- If the parent is the authority, it responds; otherwise, it sends the query to yet another server.
- When the query is finally resolved, the response travels back until it finally reaches the requesting client.

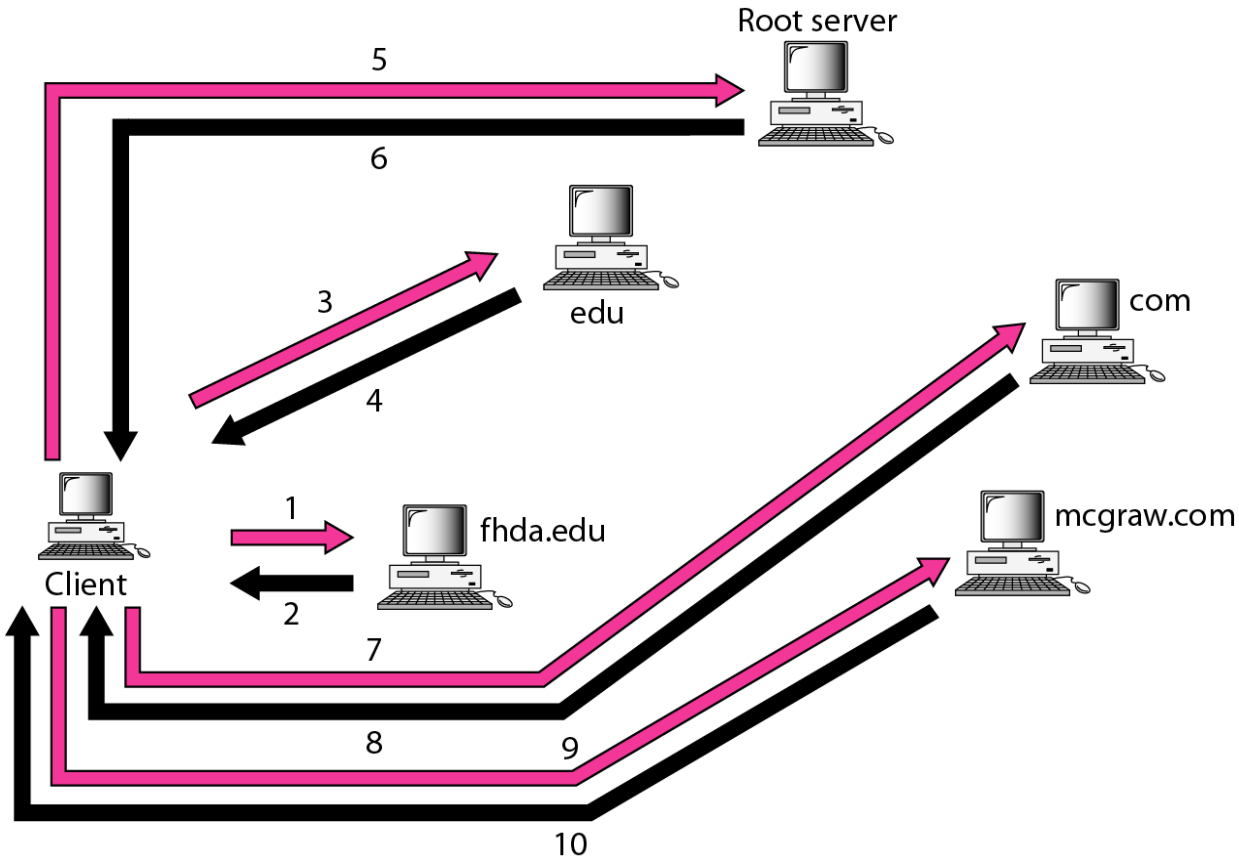
Recursive resolution



Iterative Resolution

- The server is an authority for the name, it sends the answer.
- If it is not, it returns (to the client) the IP address of the server that it thinks can resolve the query.
- The client is responsible for repeating the query to this second server.
- If the newly addressed server can resolve the problem, it answers the query with the IP address; otherwise, it returns the IP address of a new server to the client.

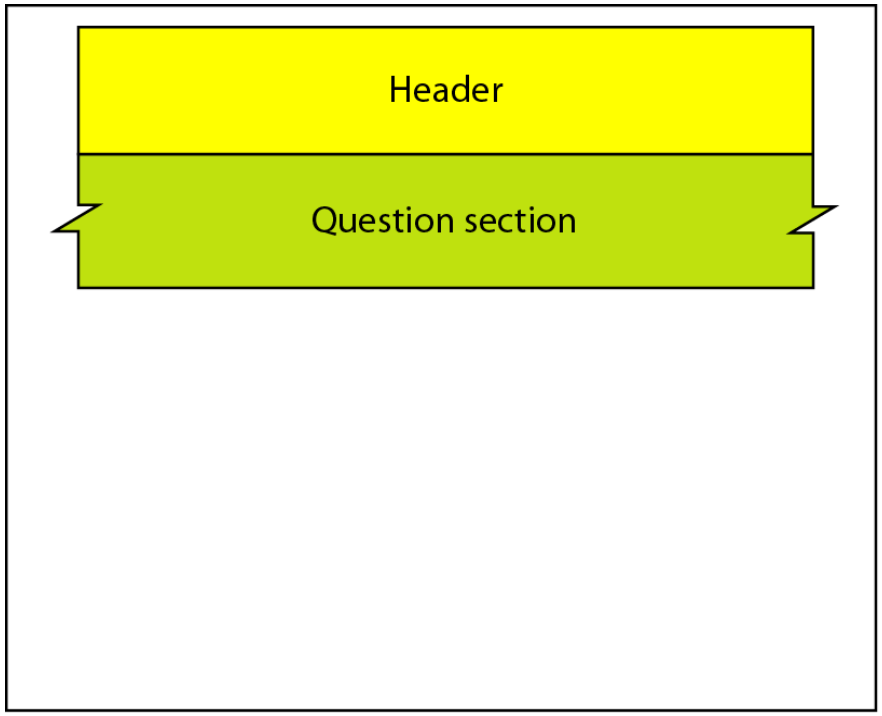
Iterative resolution



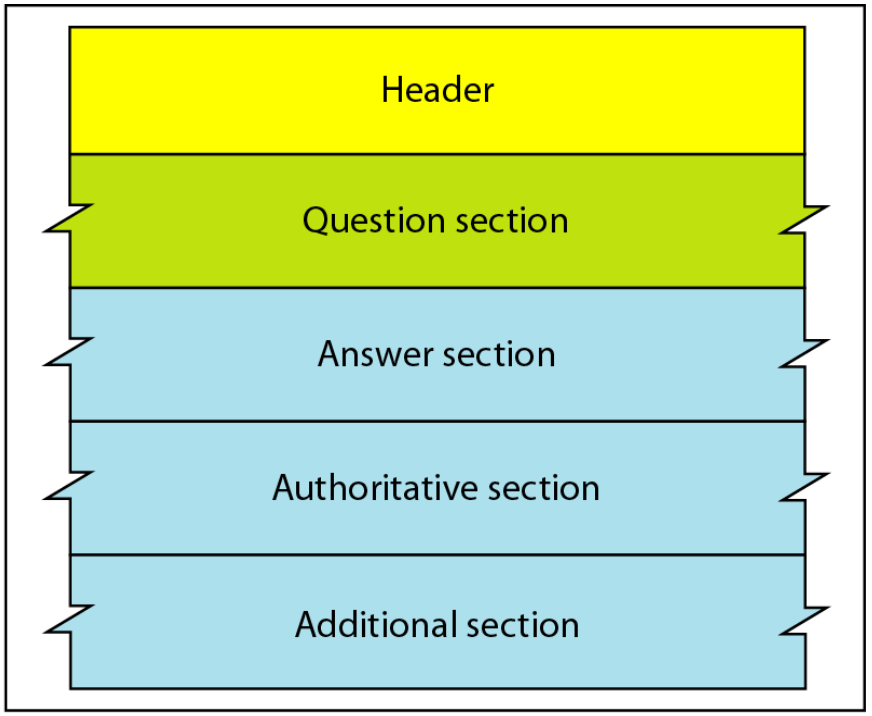
DNS Messages

- DNS has two types of messages: ***query and response.***
- Both types have the same format.
  - The *query messa*ge consists of a header and question records
  - The *response message* consists of a header, question records, answer records, authoritative records, and additional records.

Query and response messages



a. Query



b. Response

Header format

Identification	Flags
Number of question records	Number of answer records (all 0s in query message)
Number of authoritative records (all 0s in query message)	Number of additional records (all 0s in query message)

Types of Records

- Two types of records are used in DNS.
  - The *question records* are used in the question section of the query and response messages.
  - The *resource records* are used in the answer, authoritative, and additional information sections of the response message.

Registers

- How are new domains added to DNS?
  - This is done through a registrar, a commercial entity accredited by ICANN.
- A registrar first verifies that the requested domain name is unique and then enters it into the DNS database.
- **A fee is charged.**

Dynamic Domain Name System (DDNS)

- The DNS master file must be updated dynamically.
- The Dynamic Domain Name System (DDNS) therefore was devised to respond to this need.
- In DDNS, when a binding between a name and an address is determined, the information is sent, usually by DHCP to a primary DNS server.
- The primary server updates the zone. The secondary servers are notified either actively or passively.

Encapsulation

- DNS can use either UDP or TCP.
- In both cases the well-known port used by the server is port 53.
- UDP is used when the size of the response message is less than 512 bytes because most UDP packages have a 512-byte packet size limit.
- If the size of the response message is more than 512 bytes, a TCP connection is used.

Remote Logging, Electronic Mail, and File Transfer

Remote Logging

- It would be impossible to write a specific client/server program for each demand.

- The better solution is a general-purpose client/server program that lets a user access any application program on a remote computer.

- **TELNET is a general-purpose client/server application program.**

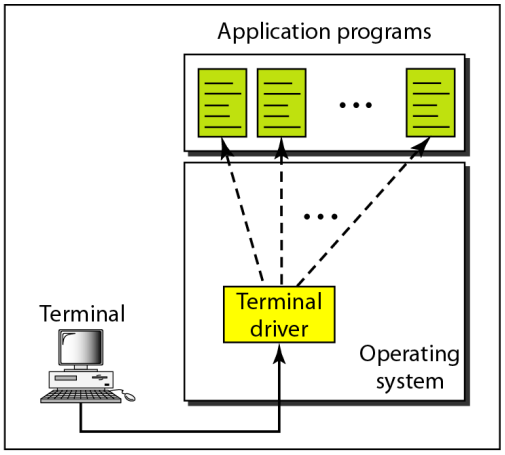
Local and remote log-in

- When a user wants to access an application program or utility located on a remote machine, she performs remote log-in.
- Here the TELNET client and server programs come into use.
- The user sends the keystrokes to the terminal driver, where the local operating system accepts the characters but does not interpret them.
- The characters are sent to the TELNET client, which transforms the characters to a universal character set called *network virtual terminal (NVT) characters* and delivers them to the local *TCP/IP* protocol stack.

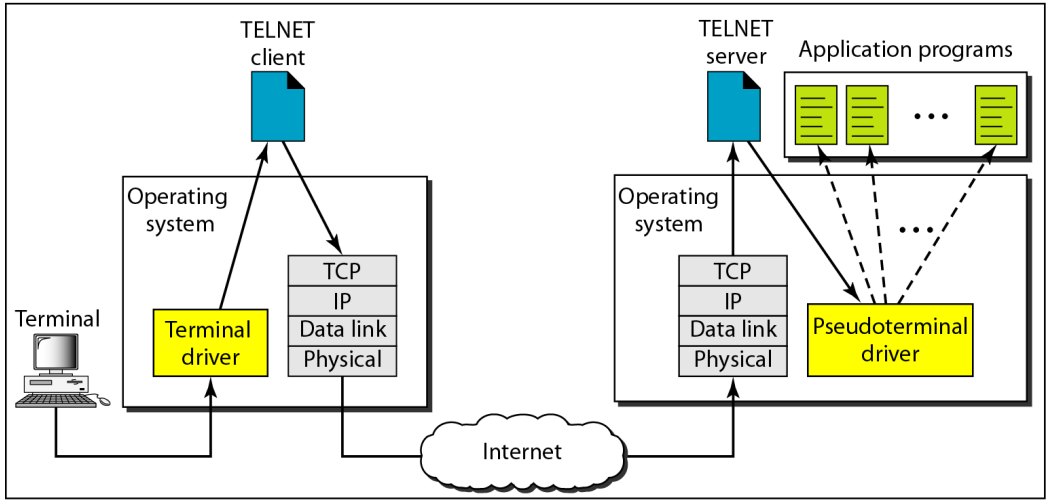
Local and remote log-in

- The commands or text, in NVT form, travel through the Internet reached at the remote machine.
- Here the characters are delivered to the operating system and passed to the TELNET server, which changes the corresponding characters understandable by the remote computer.
- However, the characters are then passed to a piece of software called a ***pseudoterminal driver*** which pretends that the characters are coming from a terminal.
- The operating system then passes the characters to the appropriate application program.

Local and remote log-in



a. Local log-in

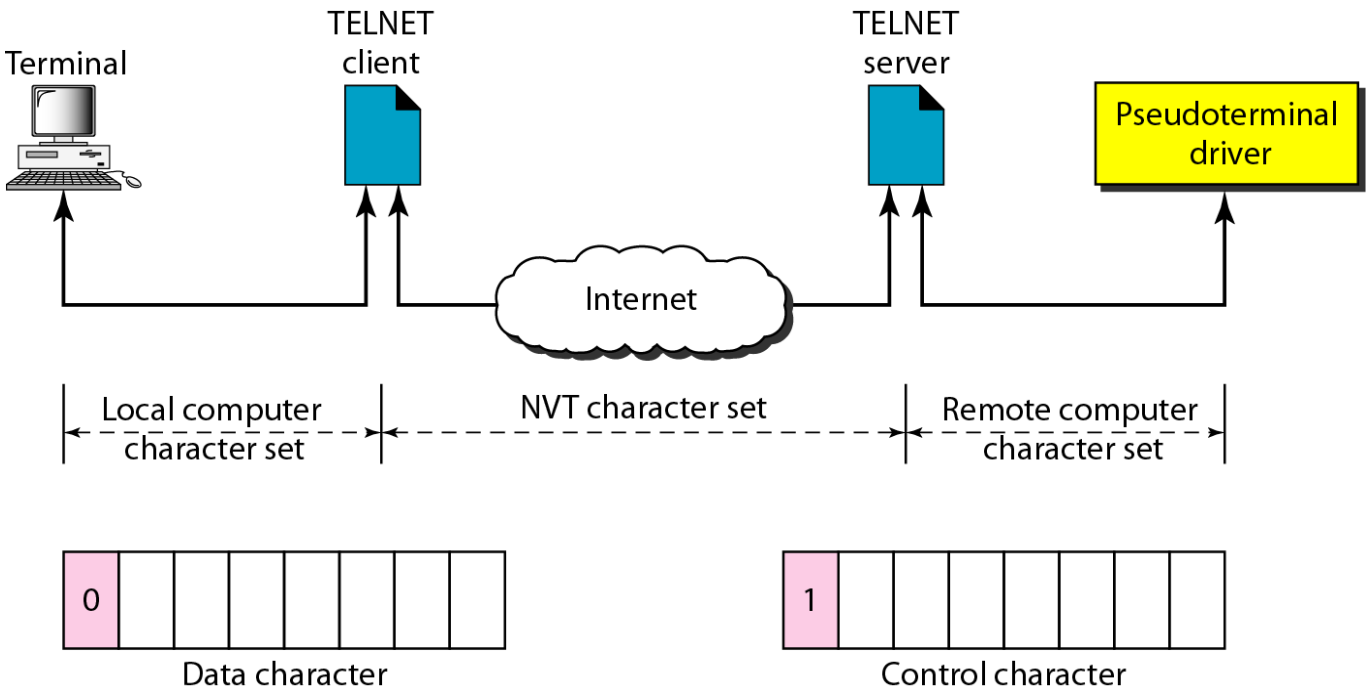


b. Remote log-in

Network Virtual terminal

- If we want to access any remote computer in the world, we must first know what type of computer we will be connected to, and
- TELNET solves this problem by defining a universal interface called the ***network virtual terminal (NVT)*** character set.
- Via this interface, the client TELNET translates characters (data or commands) that come from the local terminal into NVT form and delivers them to the network.
- The server TELNET, on the other hand, translates data and commands from NVT form into the form acceptable by the remote computer.

Concept of NVT



NVT Character Set

- NVT uses two sets of characters, one for data and the other for control.
- Both are 8-bit bytes.
- For data:
  - NVT is an 8-bit character set in which the 7 lowest-order bits are the same as ASCII and the highest-order bit is 0.
- For control:
  - NVT uses an 8-bit character set in which the highest-order bit is set to l.

Some NVT control characters

Character	*Decimal*	*Binary*	*Meaning*
EOF	236	11101100	End of file
EOR	239	11101111	End of record
SE	240	11110000	Suboption end
NOP	241	11110001	No operation
DM	242	11110010	Data mark
BRK	243	11110011	Break
IP	244	11110100	Interrupt process
AO	245	11110101	Abort output
AYT	246	11110110	Are you there?
EC	247	11110111	Erase character
EL	248	11111000	Erase line
GA	249	11111001	Go ahead
SB	250	11111010	Suboption begin
WILL	251	11111011	Agreement to enable option
WONT	252	11111100	Refusal to enable option
DO	253	11111101	Approval to option request
DONT	254	11111110	Denial of option request
IAC	255	11111111	Interpret (the next character) as control

An example of embedding

Embedding

- TELNET uses only one TCP connection. The server uses the well-known port 23, and the client uses an ephemeral port.
- The same connection is used for sending both data and control characters.

Options

- TELNET lets the client and server negotiate options before or during the use of the service.
- Options are extra features available to a user with a more sophisticated terminal.
- Users with simpler terminals can use default features.

Options

Code	*Option*	*Meaning*
0	Binary	Interpret as 8-bit binary transmission.
1	Echo	Echo the data received on one side to the other.
3	Suppress go ahead	Suppress go-ahead signals after data.
5	Status	Request the status of TELNET.
6	Timing mark	Define the timing marks.
24	Terminal type	Set the terminal type.
32	Terminal speed	Set the terminal speed.
34	Line mode	Change to line mode.

Option Negotiation

- To use any of the options mentioned in the previous section first requires option negotiation between the client and the server.
- Four control characters are used for this purpose

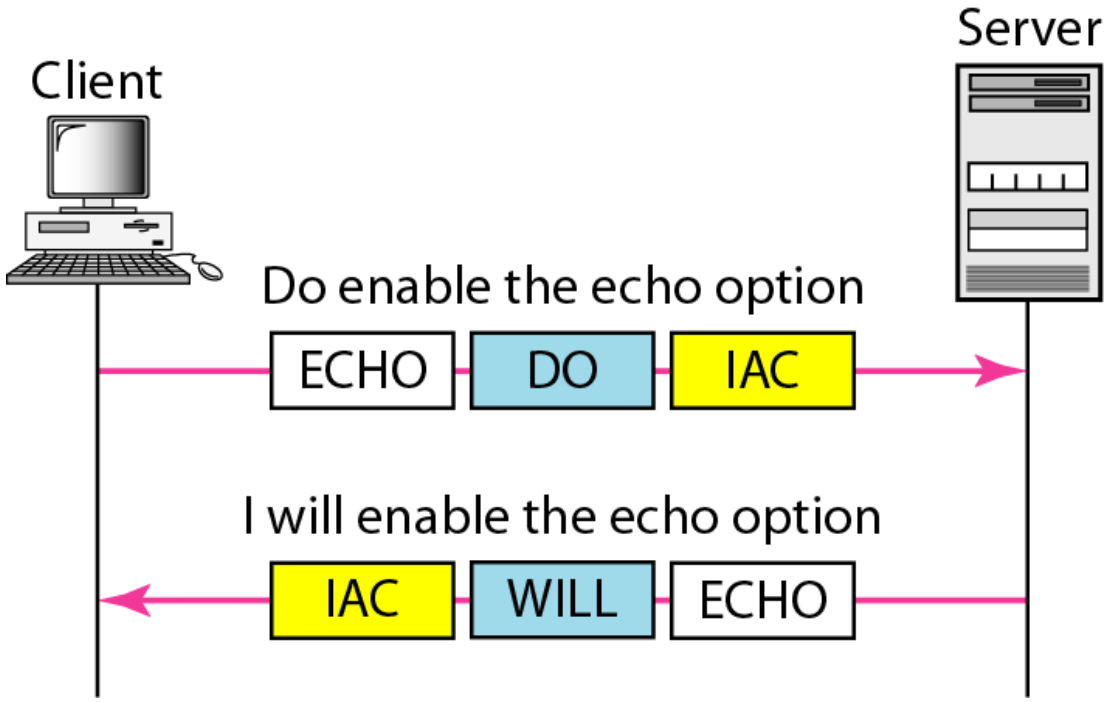
NVT character set for option negotiation

Character	*Decimal*	*Binary*	*Meaning*
WILL	251	11111011	1. Offering to enable 2. Accepting a request to enable
WONT	252	11111100	1. Rejecting a request to enable 2. Offering to disable 3. Accepting a request to disable
DO	253	11111101	1. Approving an offer to enable 2. Requesting to enable
DONT	254	11111110	1. Disapproving an offer to enable 2. Approving an offer to disable 3. Requesting to disable

Example

- Figure shows an example of option negotiation.
- In this example, the client wants the server to echo each character sent to the server.
- The echo option is enabled by the server because it is the server that sends the characters back to the user terminal.
- Therefore, the client should request from the server the enabling of the option using DO.
- The request consists of three characters: IAC, DO, and ECHO.
- The server accepts the request and enables the option.
- It informs the client by sending the three-character approval: IAC, WILL, and ECHO.

Example: Echo option

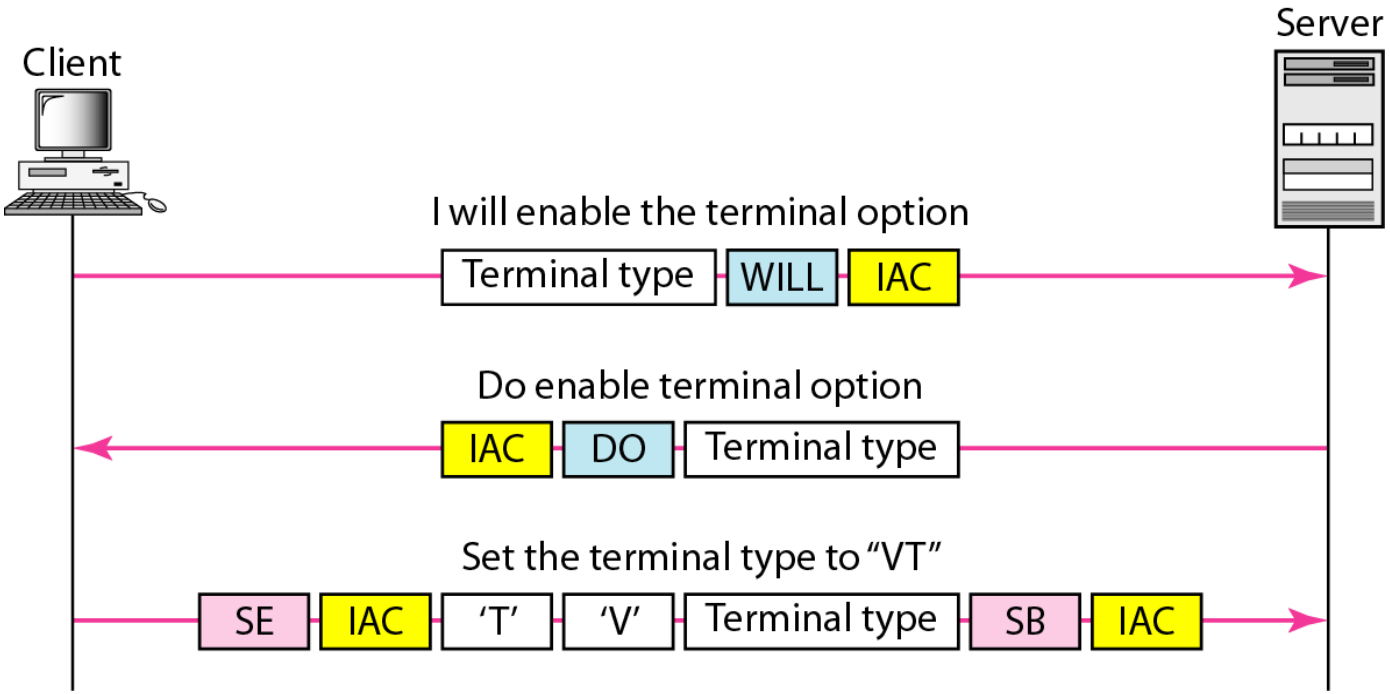


Character set for suboptions

Character	*Decimal*	*Binary*	*Meaning*
SE	240	11110000	Suboption end
SB	250	11111010	Suboption begin

Example : suboption negotiation

- Figure shows an example of suboption negotiation.
- In this example, the client wants to negotiate the type of the terminal.



Mode of Operation

- Most TELNET implementations operate in one of three modes: default mode, character mode, or line mode.
- Default Mode:
  - The default mode is used if no other modes are invoked through option negotiation.
  - In this mode, the echoing is done by the client.
  - The user types a character, and the client echoes the character on the screen (or printer) but does not send it until a whole line is completed.

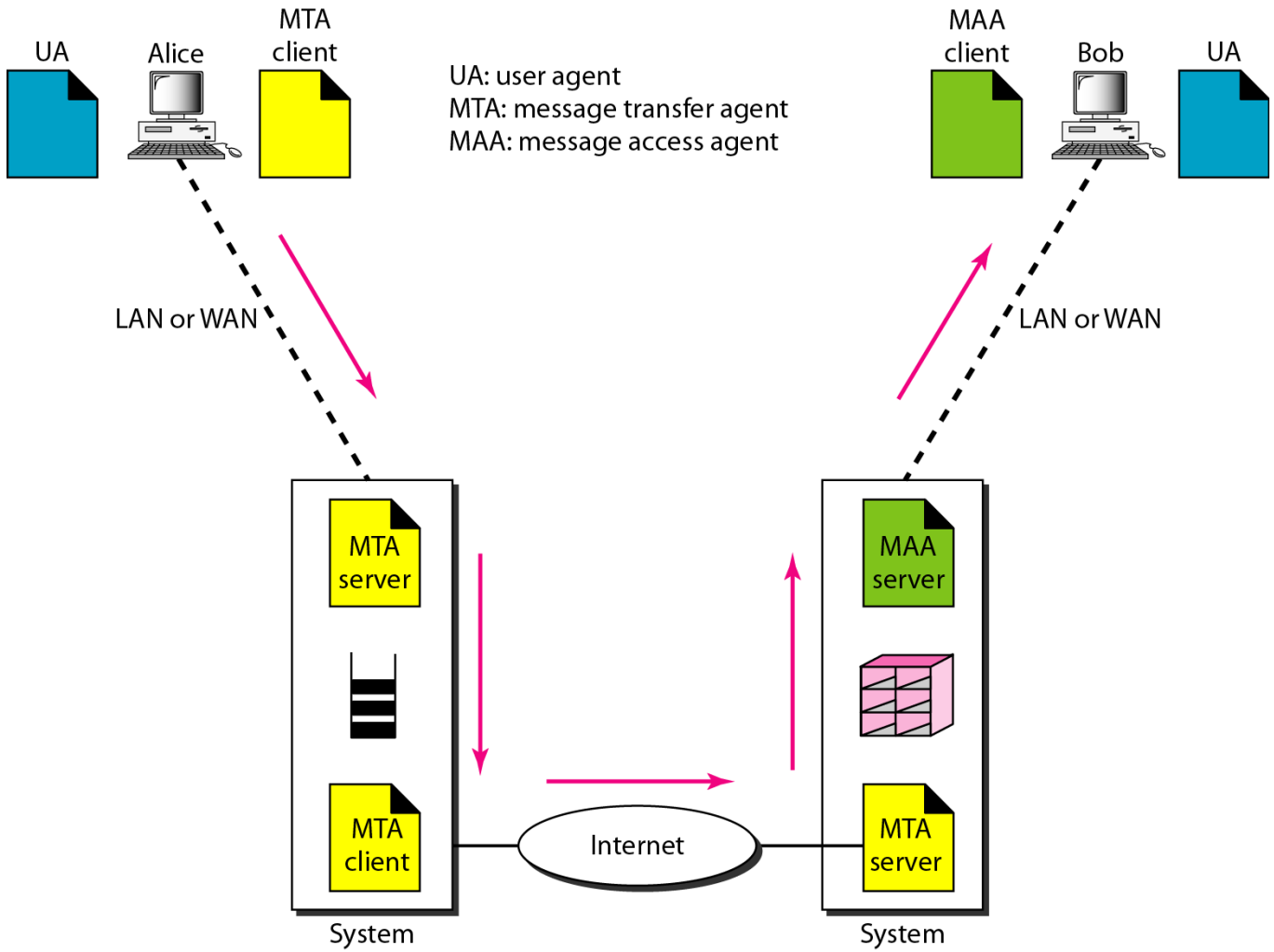
Mode of Operation

- Character Mode:
  - Each character typed is sent by the client to the server.
  - The server normally echoes the character back to be displayed on the client screen.
  - In this mode the echoing of the character can be delayed if the transmission time is long.
  - It also creates overhead (traffic) for the network because three TCP segments must be sent for each character of data.
- Line Mode:
  - A new mode has been proposed to compensate for the deficiencies of the default mode and the character mode.
  - In this mode line editing (echoing, character erasing, line erasing, and so on) is done by the client.
  - The client then sends the whole line to the server.

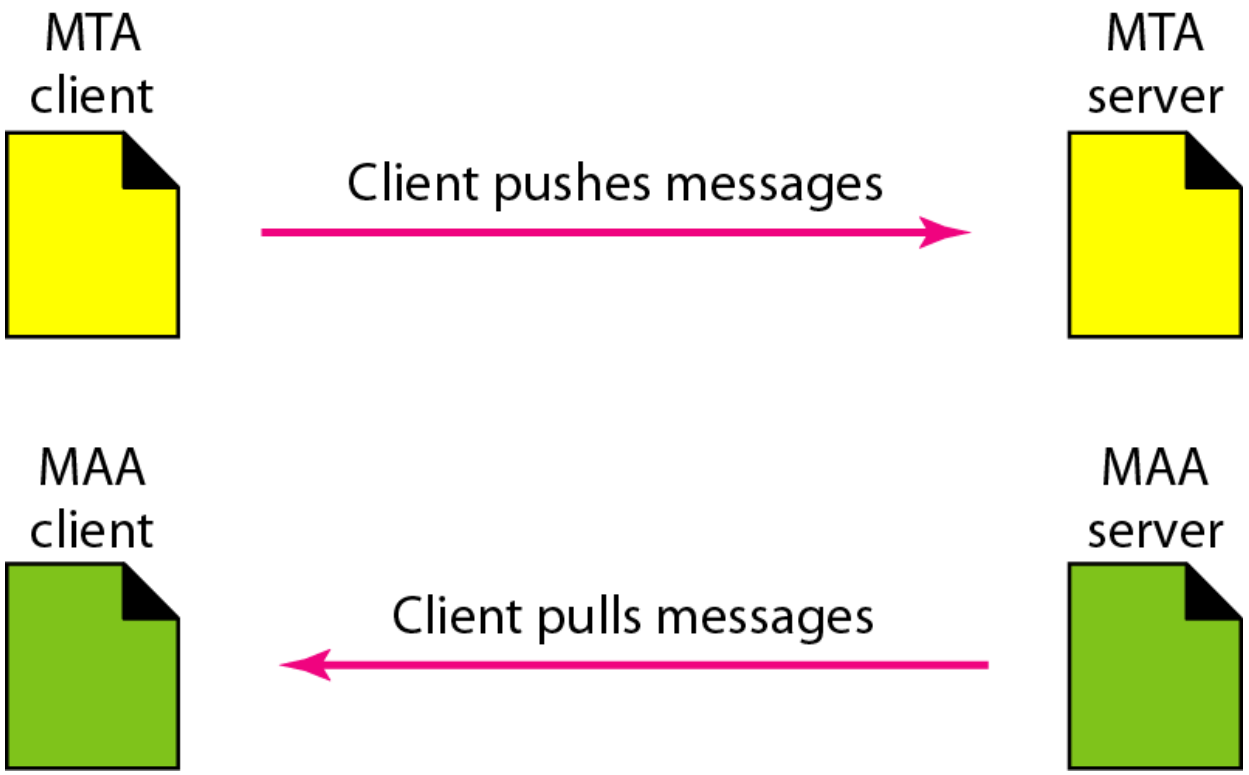
Electronic Mail

- One of the most popular Internet services is electronic mail (e-mail).
- The designers of the Internet probably never imagined the popularity of this application program.
- Its architecture consists of several components

Sending electronic mail



Push versus pull in electronic email

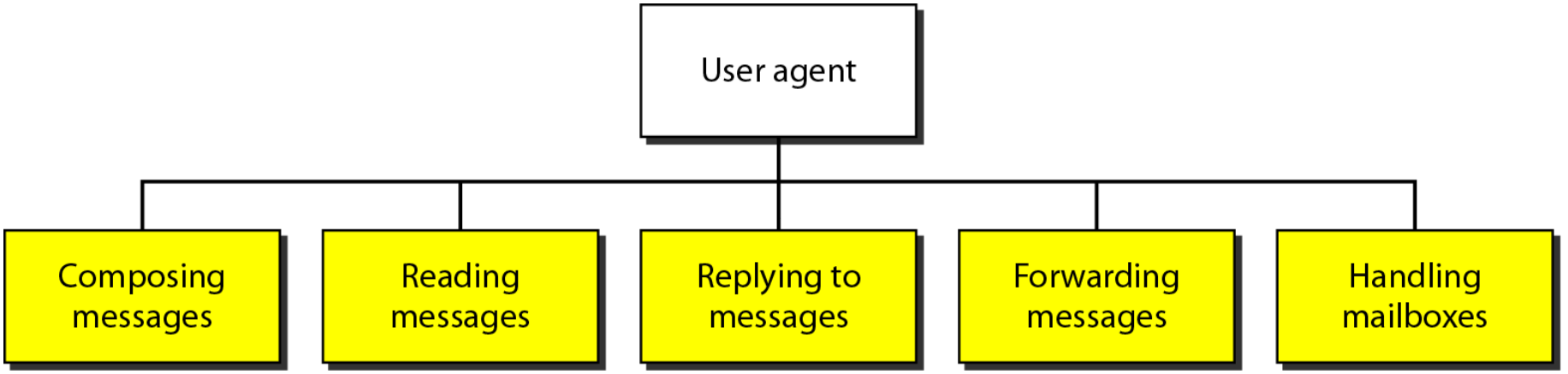


Note

When both sender and receiver are connected to the mail server via a LAN or a WAN, we need two UAs, two pairs of MTAs and a pair of MAAs.

This is the most common situation today.

Services of user agent



User Agent Types

- There are two types of user agents: command-driven and GUI-based.
- **Command-Driven:**
  - Command-driven user agents belong to the early days of electronic mail.
  - They are still present as the underlying user agents in servers.
  - A command-driven user agent normally accepts a one-character command from the keyboard to perform its task.
- Some examples of command-driven user agents are *mail, pine,* and *elm.*

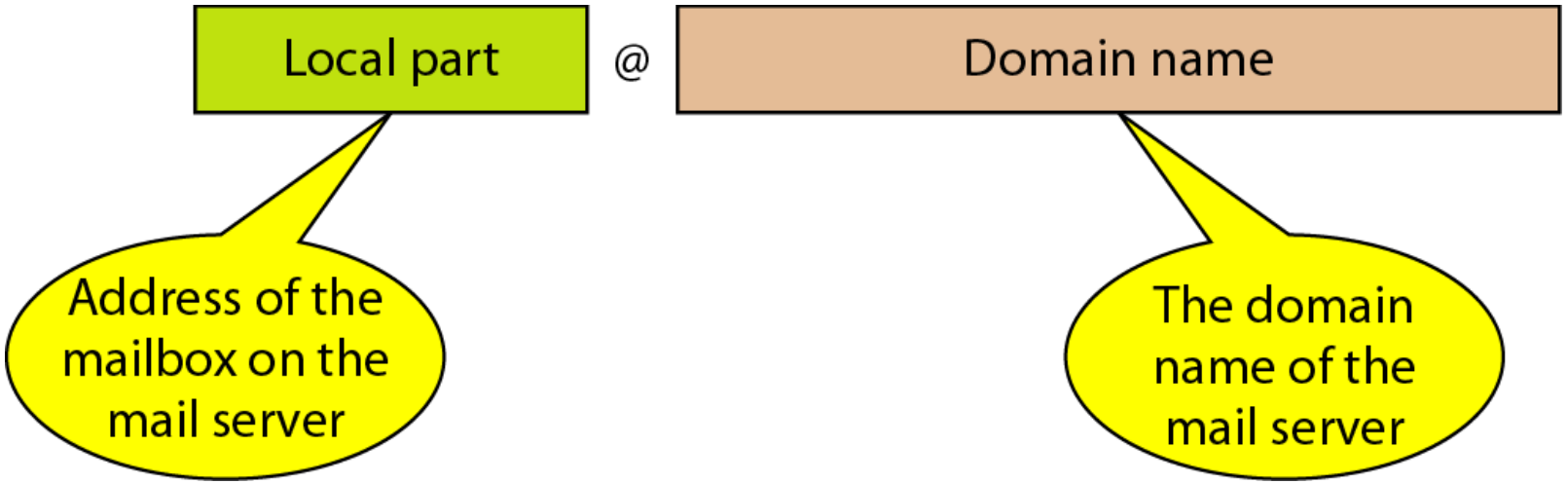
GUI-Based

- Modem user agents are GUI-based. They contain graphical-user inter- face (GUI) components that allow the user to interact with the software by using both the keyboard and the mouse.
- They have graphical components such as icons, menu bars, and windows that make the services easy to access.
- Some examples of GUI-based user agents are Eudora, Microsoft's Outlook, and Netscape.

Format of an e-mail

Address From: ---- To: …..	Message: *Body of the Message*

E-mail address



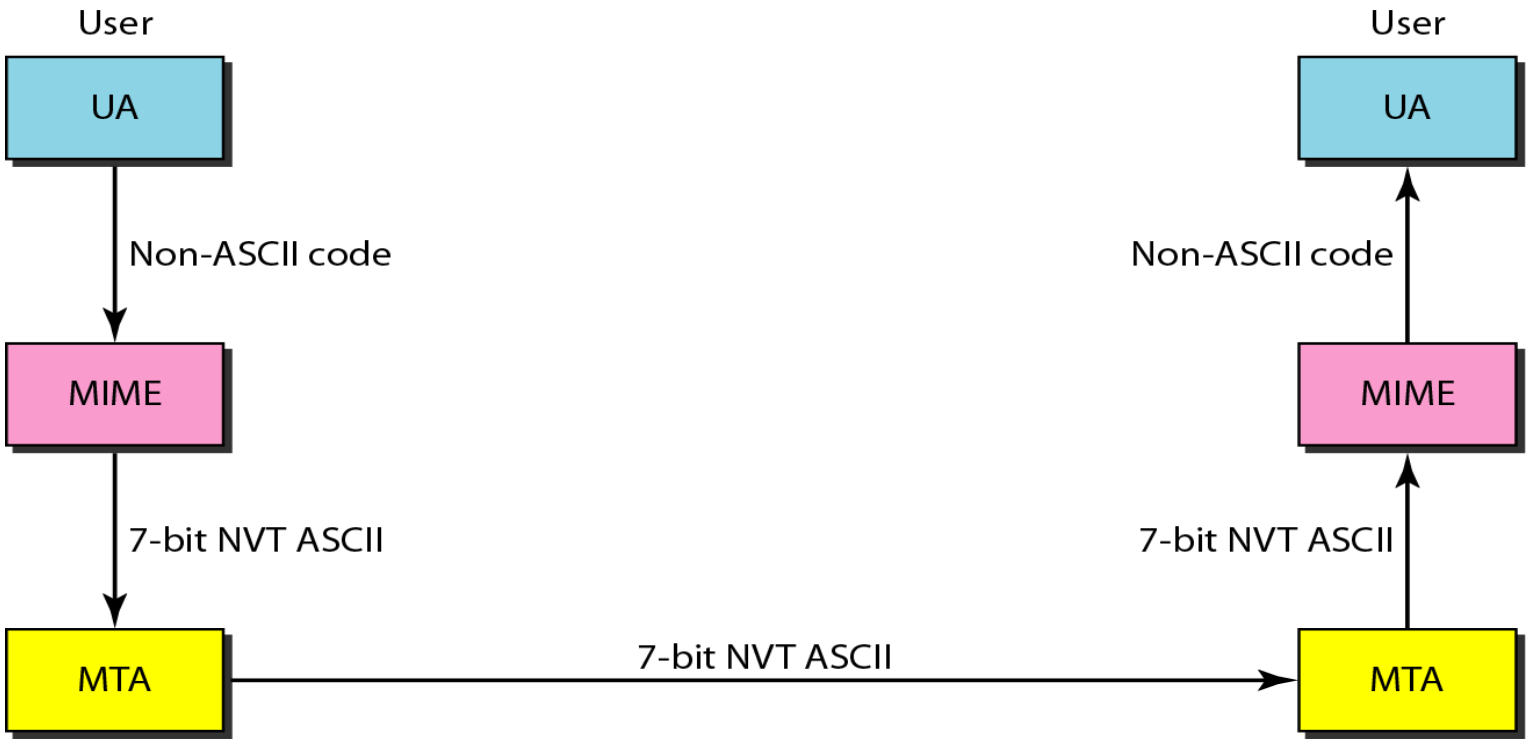
- **Local Part:** Defines the name of a special file, called the user mailbox., where all the mail received for a user is stored for retrieval by the message access agent.

MIME

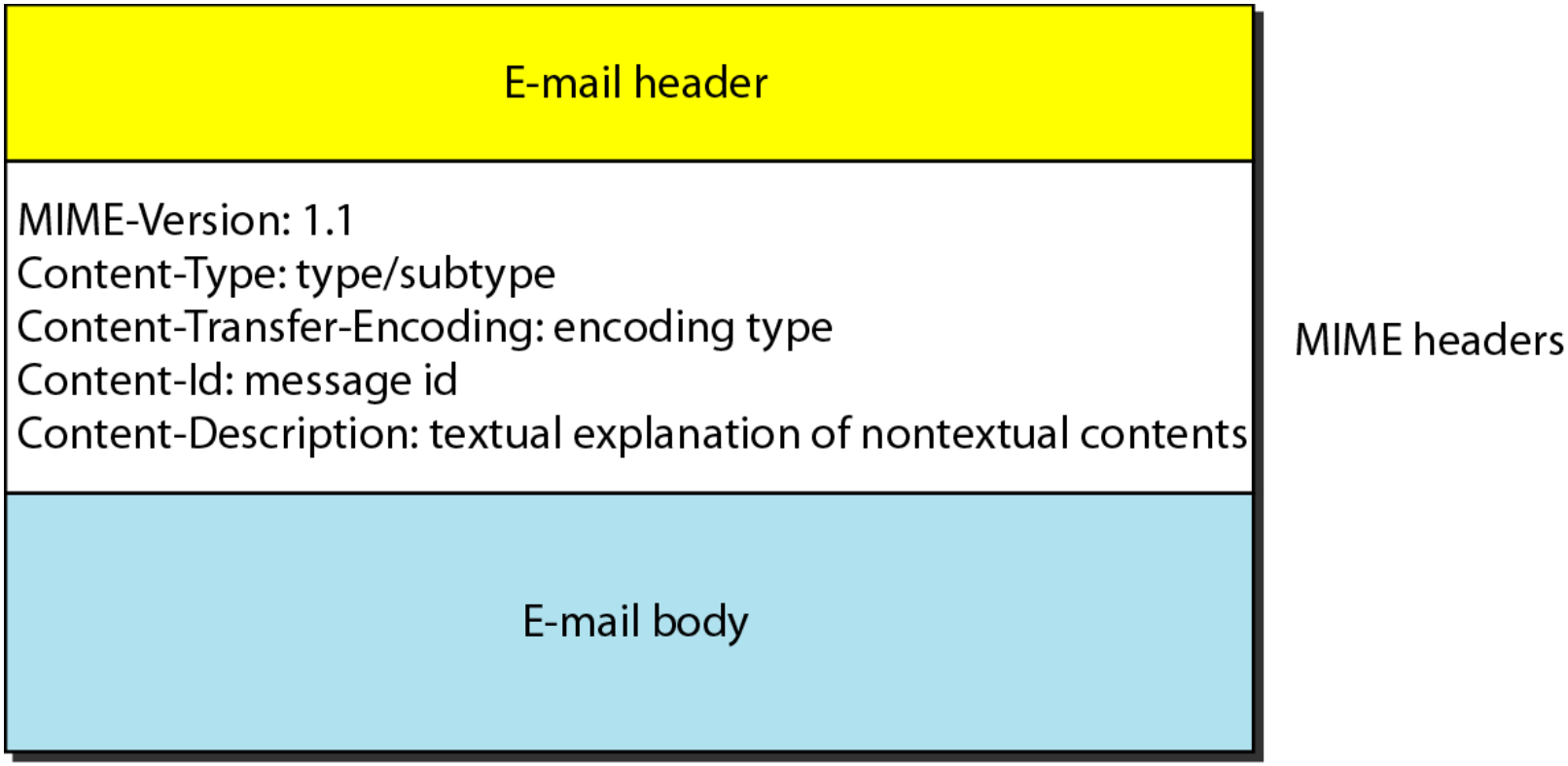
- Multipurpose Internet Mail Extensions (MIME) is a supplementary protocol that allows non-ASCII data to be sent through e-mail.

- It 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.

- The message at the receiving side is transformed back to the original data.



MIME header


E-mail 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
E-mail body
MIME headers

Data types and subtypes in MIME

Type	*Subtype*	*Description*
Text	Plain	Unformatted
	HTML	HTML format (see Chapter 27)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to mixed subtypes, but the default is message/RFC822
	Alternative	Parts are different versions of the same message
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 (8-bit bytes)

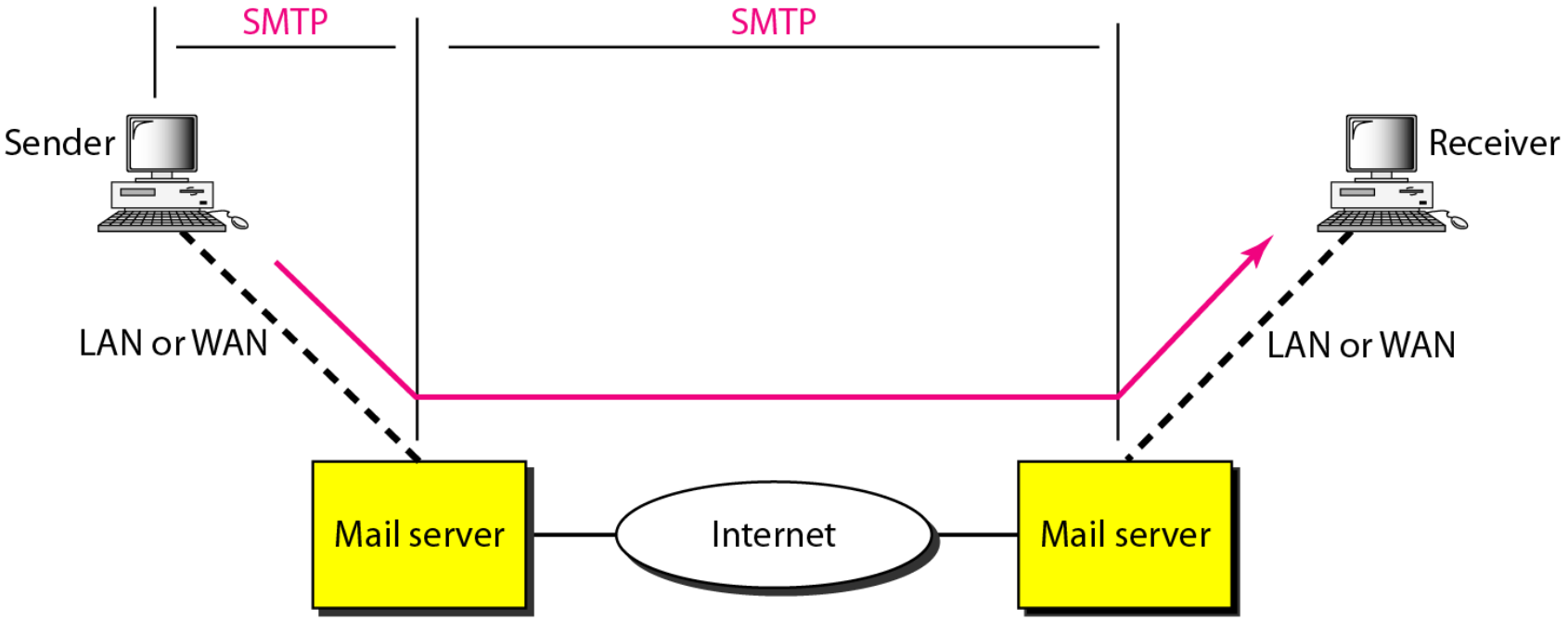
Content-transfer-encoding

Type	*Description*
7-bit	NVT ASCII characters and short lines
8-bit	Non-ASCII characters and short lines
Binary	Non-ASCII characters with unlimited-length lines
Base-64	6-bit blocks of data encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters encoded as an equals sign followed by an ASCII code

Message Transfer Agent: SMTP

- The actual mail transfer is done through message transfer agents.
- To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA.
- The formal protocol that defines the MTA client and server in the Internet is called the Simple Mail Transfer Protocol (SMTP).
- Two pairs of MTA client/server programs are used in the most common situation

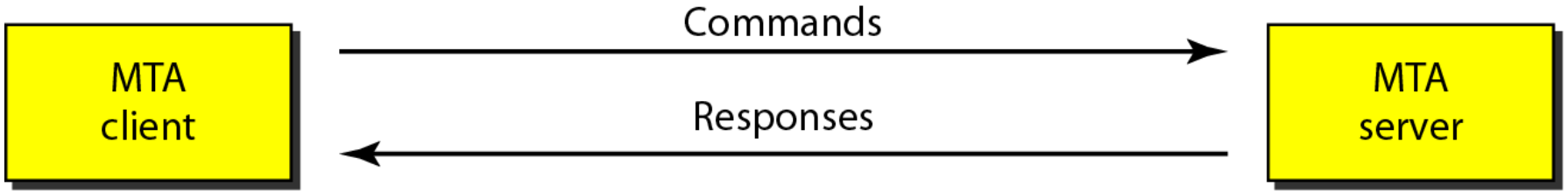
SMTP range


SMTP
SMTP
Sender
Receiver
LAN or WAN
LAN or WAN
Mail server
Internet
Mail server

Commands and responses

- SMTP uses commands and responses to transfer messages between an MTA client and an MTA server.

- Each command or reply is terminated by a two-character (carriage return and line feed) end-of-line token.



Command format

Commands

- SMTP defines 14 commands.
- The first five are mandatory
  - Every implementation must support these five commands.
- The next three are often used and highly recommended.
- The last six are seldom used.

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

- Responses are sent from the server to the client.
- A response is a three digit code that may be followed by additional textual information.

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

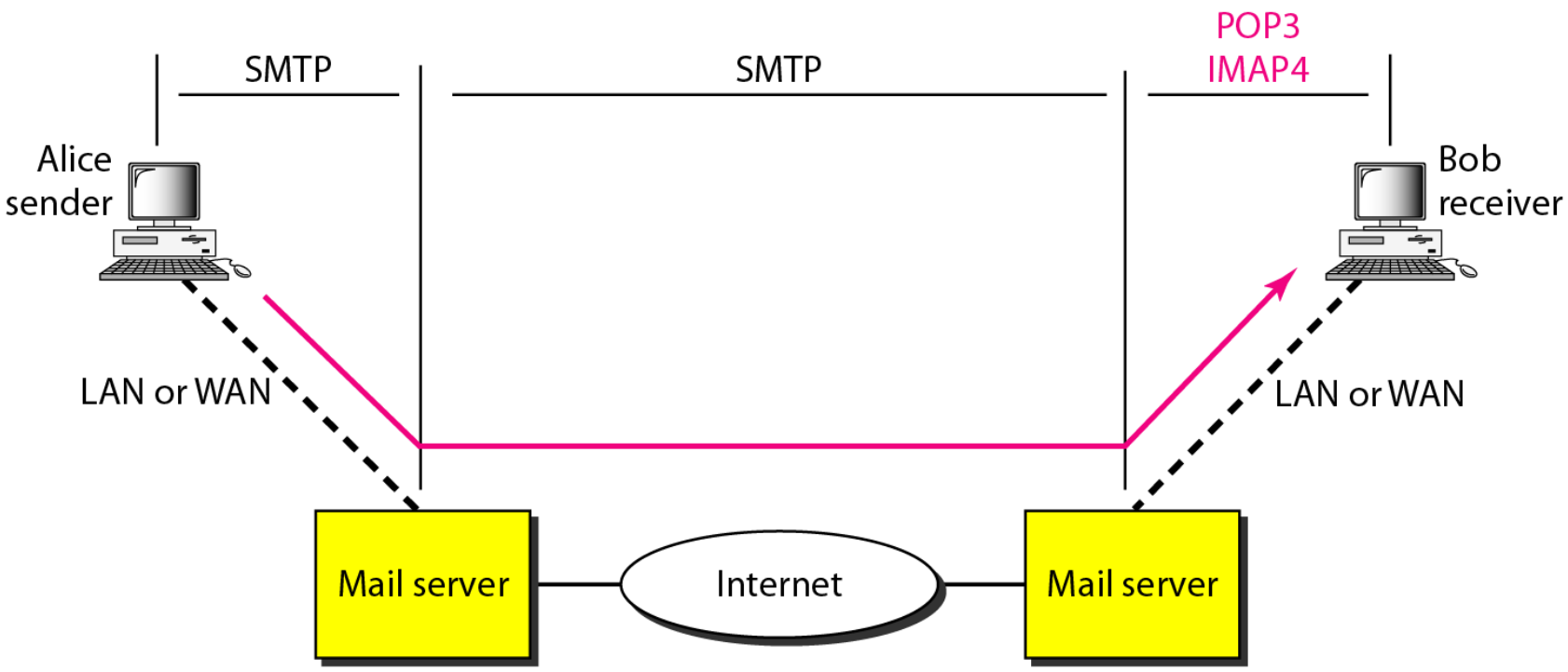
Responses (continued)

Code	*Description*
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

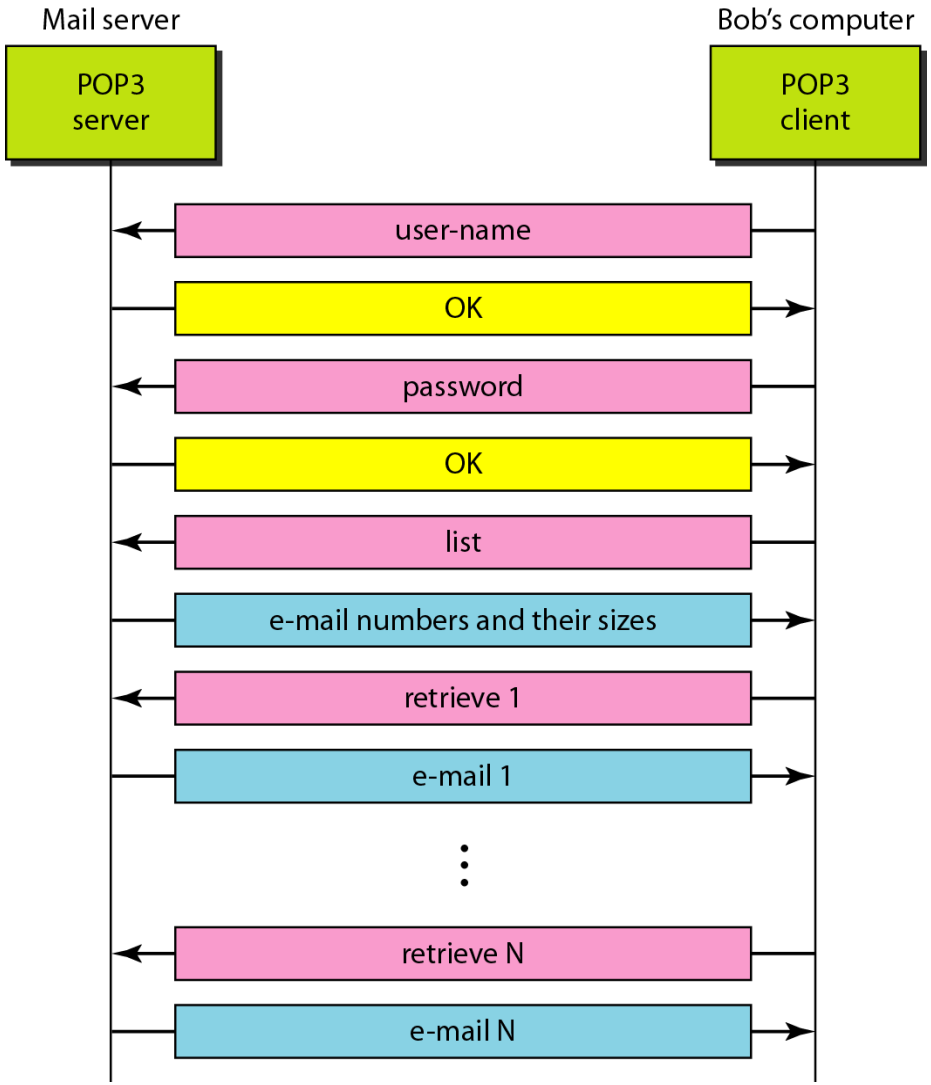
Message Access Agent: **POP3** and IMAP4

- The third stage needs a *pull* protocol
- The client must pull messages from the server.
- The direction of the bulk data is from the server to the client.
- The third stage uses a message access agent.

POP3 and IMAP4



The exchange of commands and responses in POP3



IMAP4

- IMAP4 provides the following extra functions:
  - A user can check the e-mail header prior to downloading.
  - A user can search the contents of the e-mail for a specific string of characters prior to downloading.
  - A user can partially download e-mail.
    - This is especially useful if bandwidth is limited
    - The e-mail contains multimedia with high bandwidth requirements.
  - A user can create, delete, or rename mailboxes on the mail server.
  - A user can create a hierarchy of mailboxes in a folder for e-mail storage.

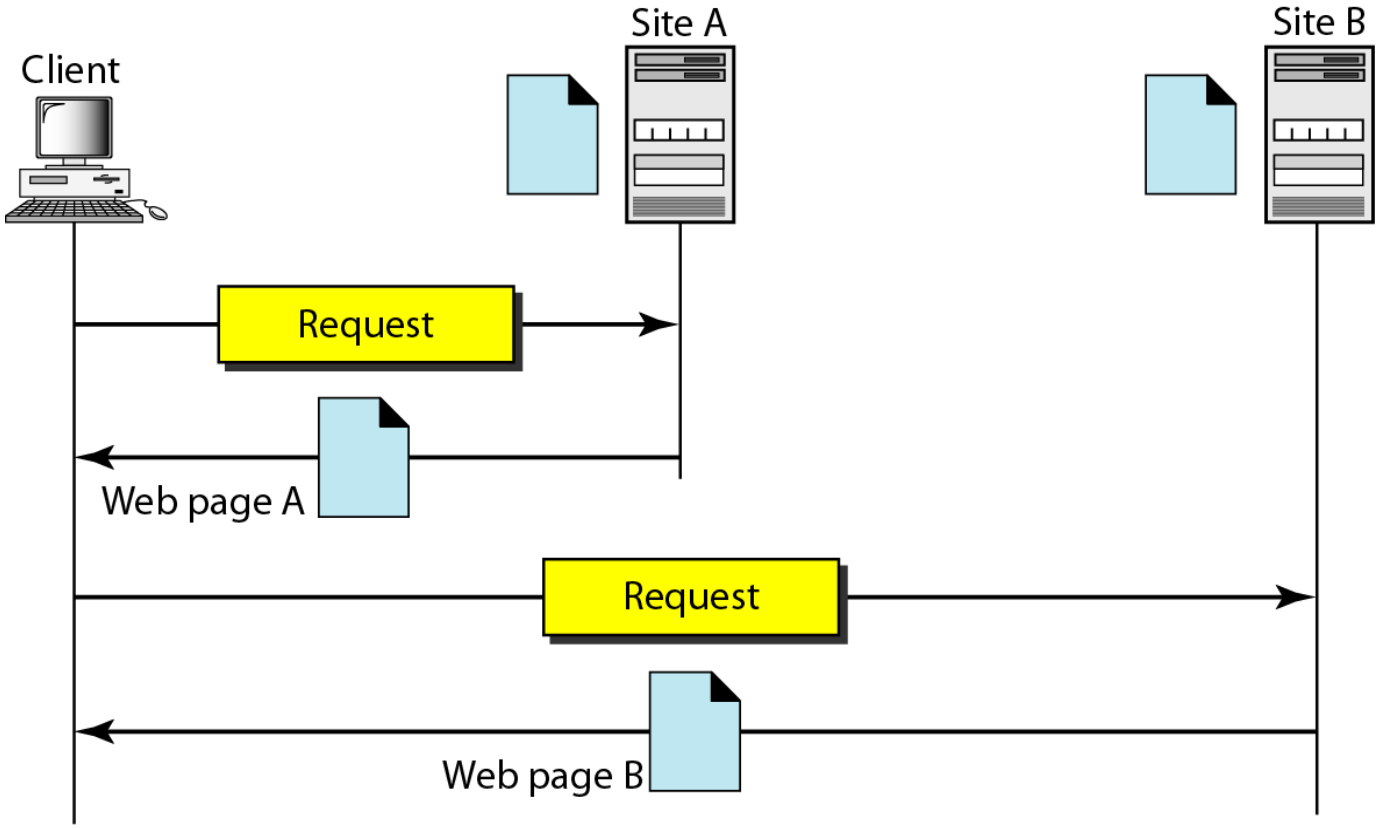
WWW and HTTP

ARCHITECTURE

- The WWW today is a distributed client/server service, in which a client using a browser can access a service using a server.

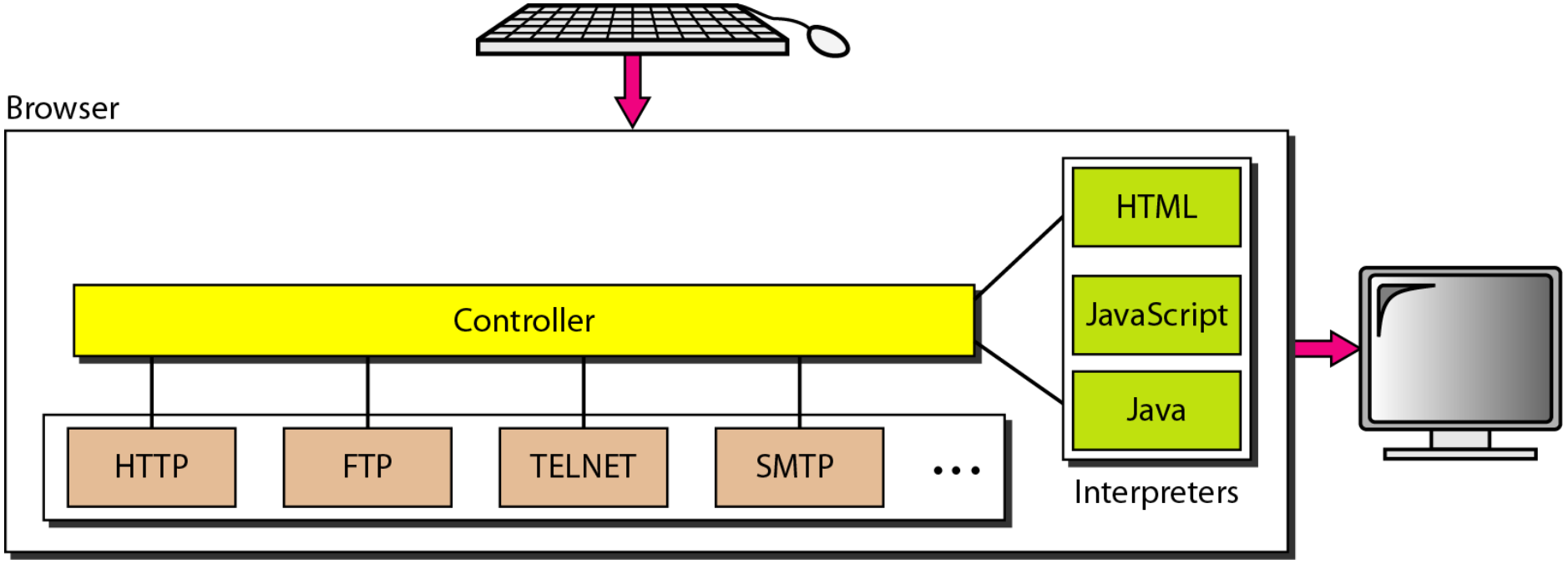
- However, the service provided is distributed over many locations called sites.

Architecture of WWW



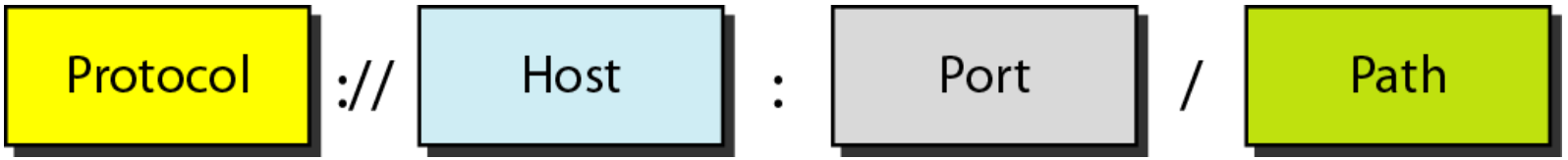
Browser

- Each browser usually consists of three parts:
  - *Controller, Client protocol, and Interpreters.*



Uniform Resource Locator (*URL)*

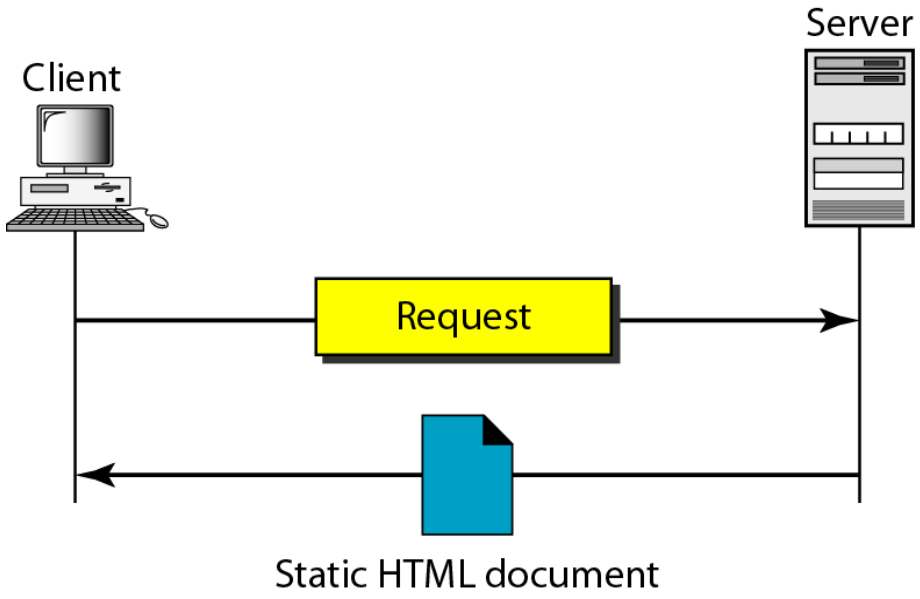
- A client that wants to access a Web page needs the address.
- To facilitate the access of documents distributed throughout the world, HTTP uses locators.
- The uniform resource locator (URL) is a standard for specifying any kind of information on the Internet.



WEB DOCUMENTS

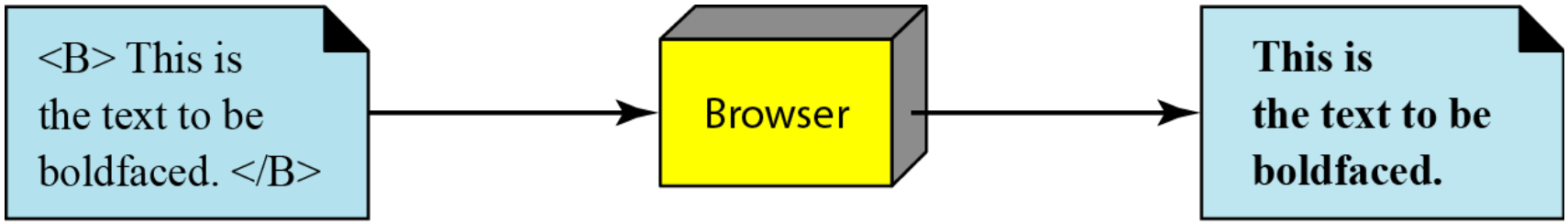
- The documents in the WWW can be grouped into three broad categories: static, dynamic, and active.
- The category is based on the time at which the contents of the document are determined.

Static document

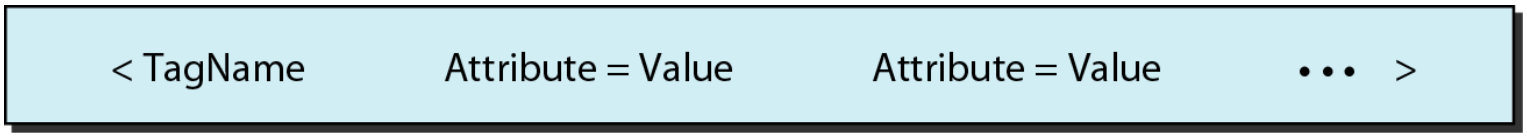


Boldface tags

Effect of boldface tags



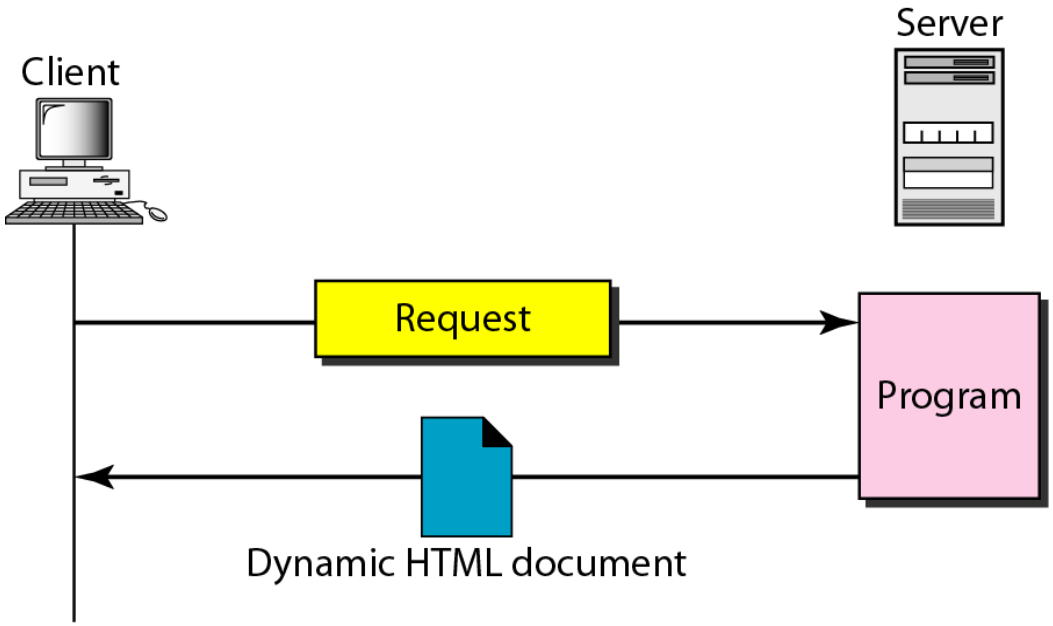
Beginning and ending tags



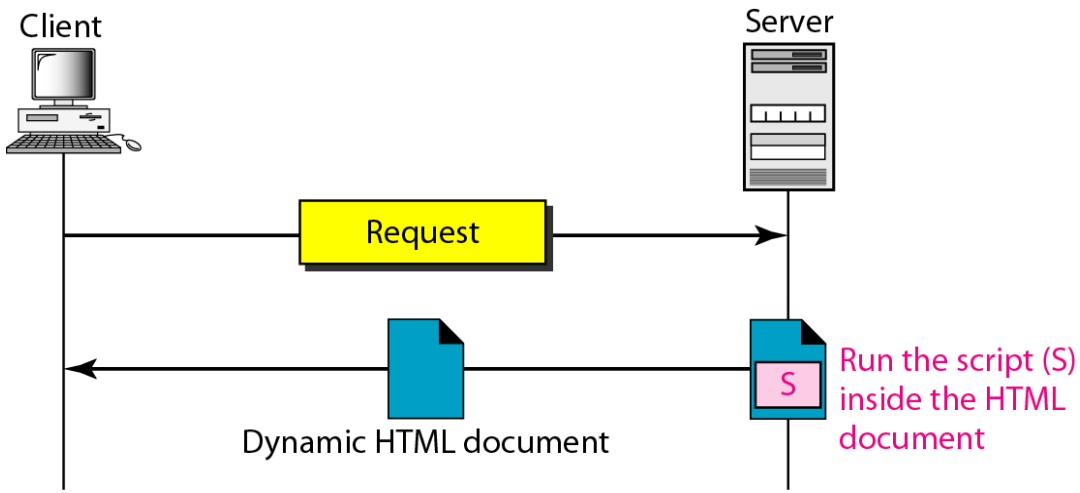
a. Beginning tag

b. Ending tag

Dynamic document using CGI



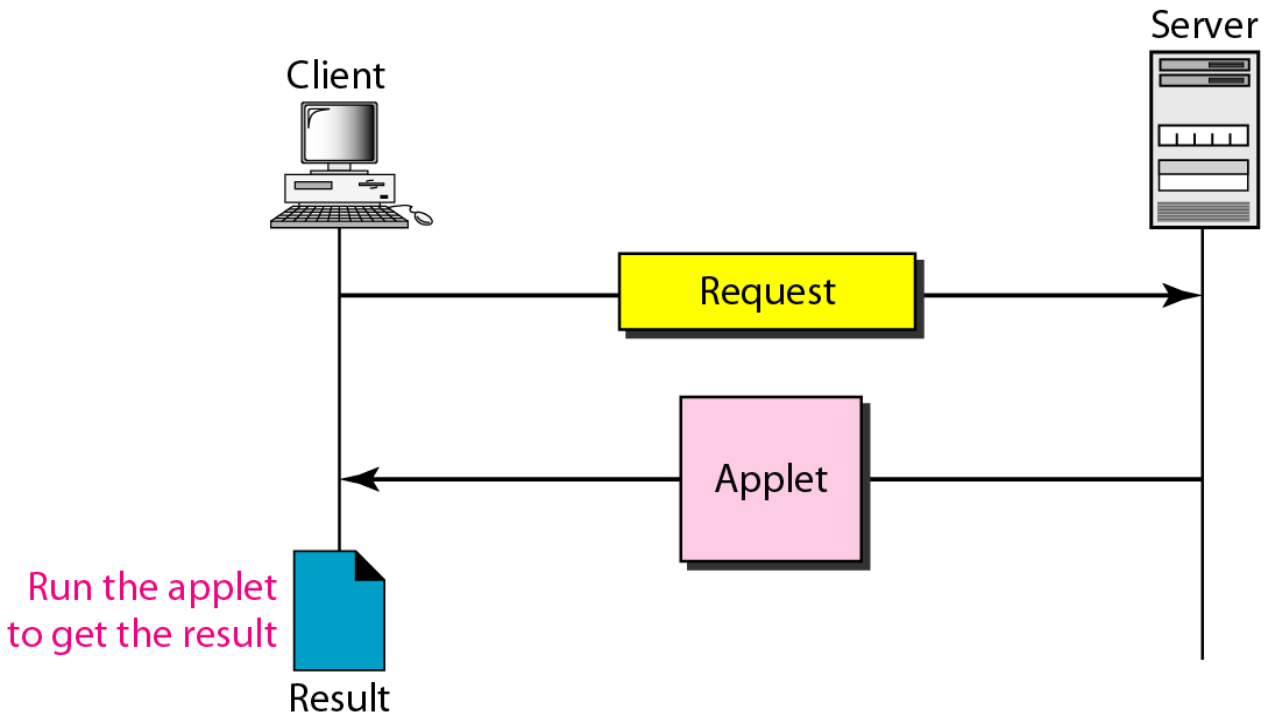
Dynamic document using server-site script



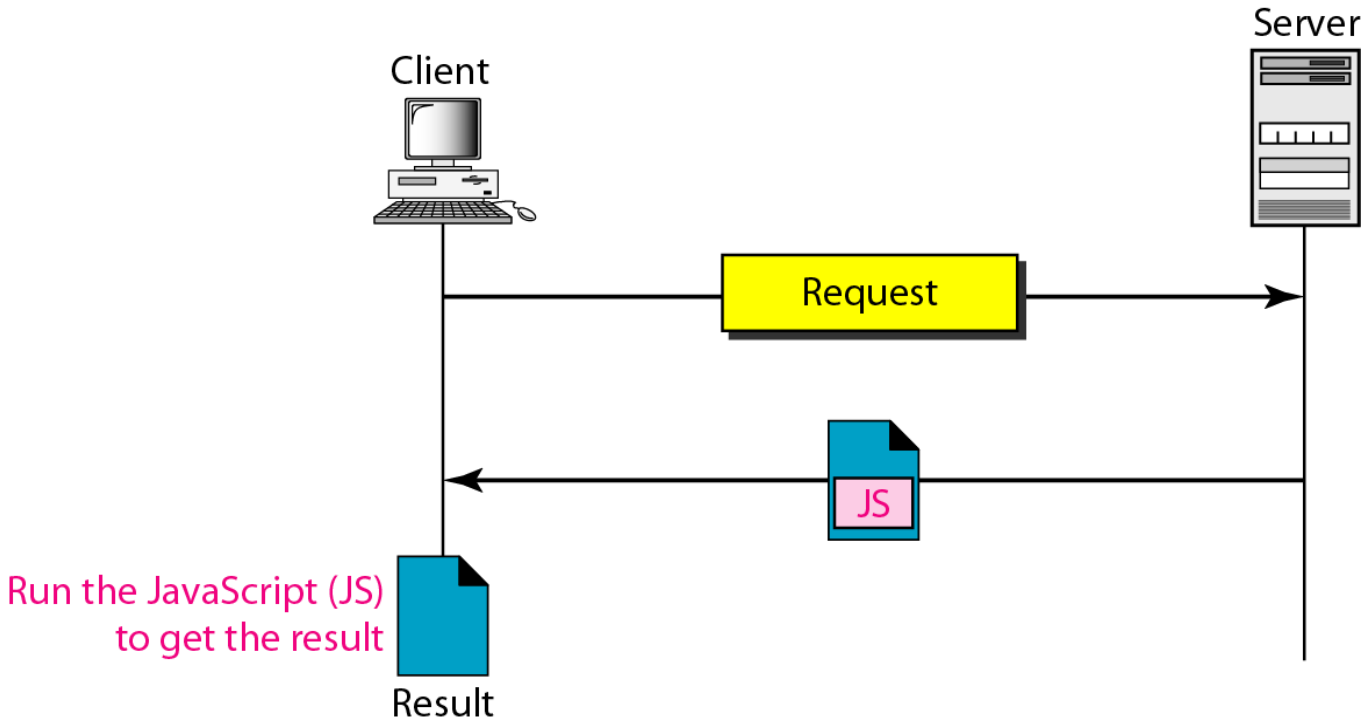
Note

Dynamic documents are sometimes referred to as server-site dynamic documents.

Active document using Java applet



Active document using client-site script



Note

Active documents are sometimes referred to as client-site dynamic documents.

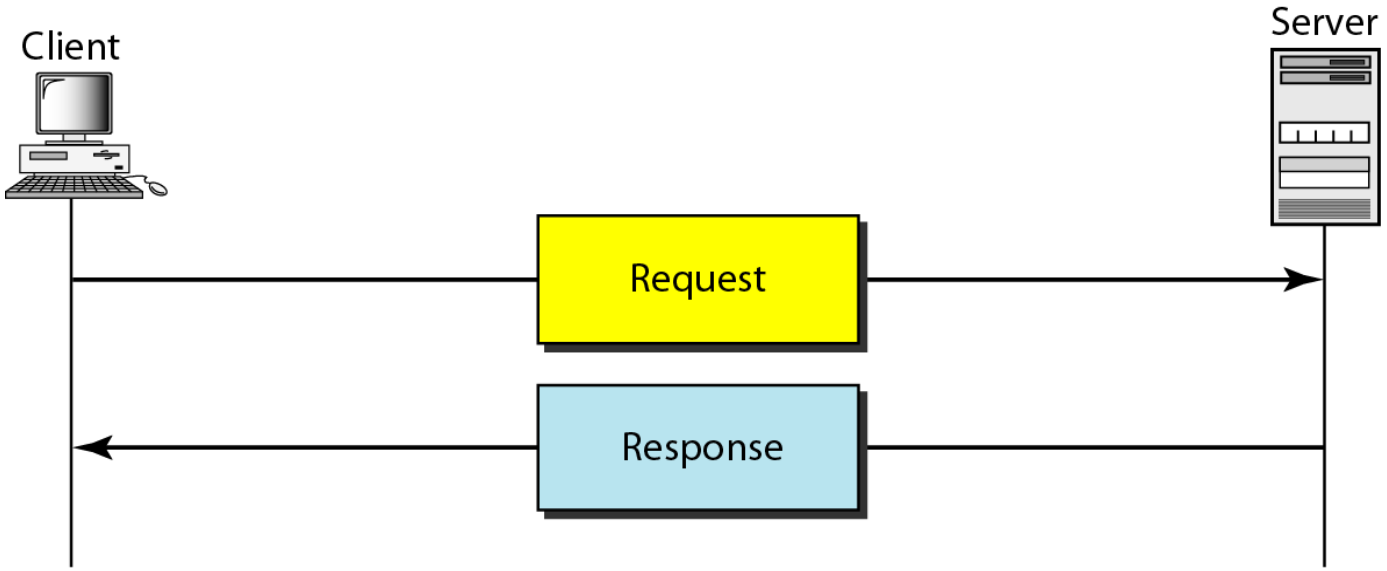
HTTP

- The Hypertext Transfer Protocol (HTTP) is a protocol used mainly to access data on the World Wide Web.
- HTTP functions as a combination of FTP and SMTP.

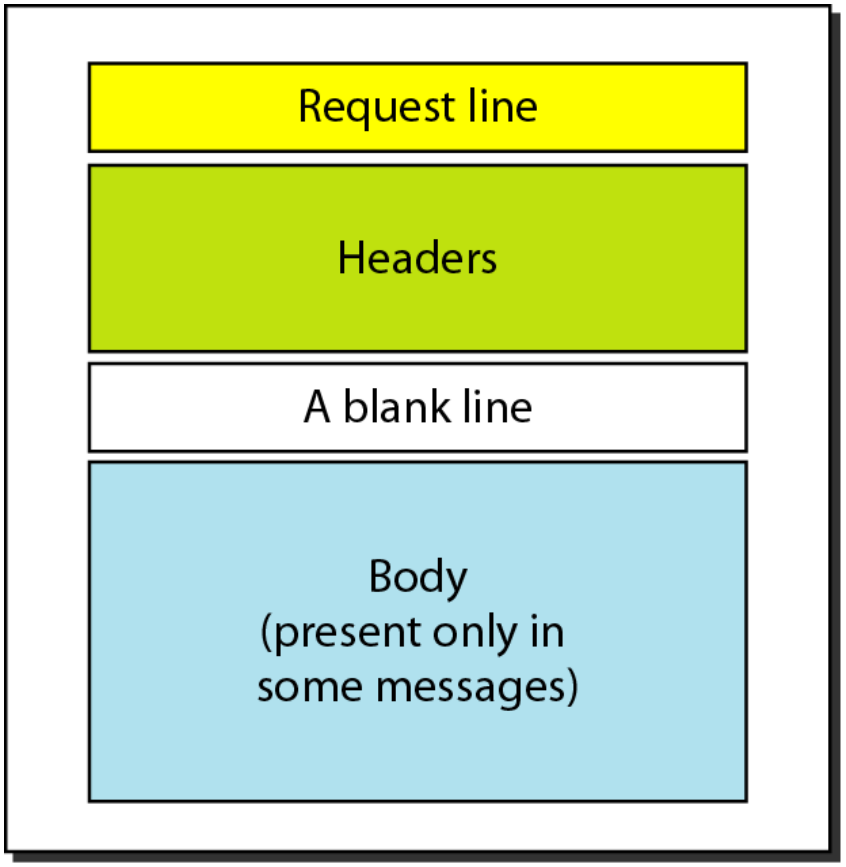
Note

HTTP uses the services of TCP on well-known port 80.

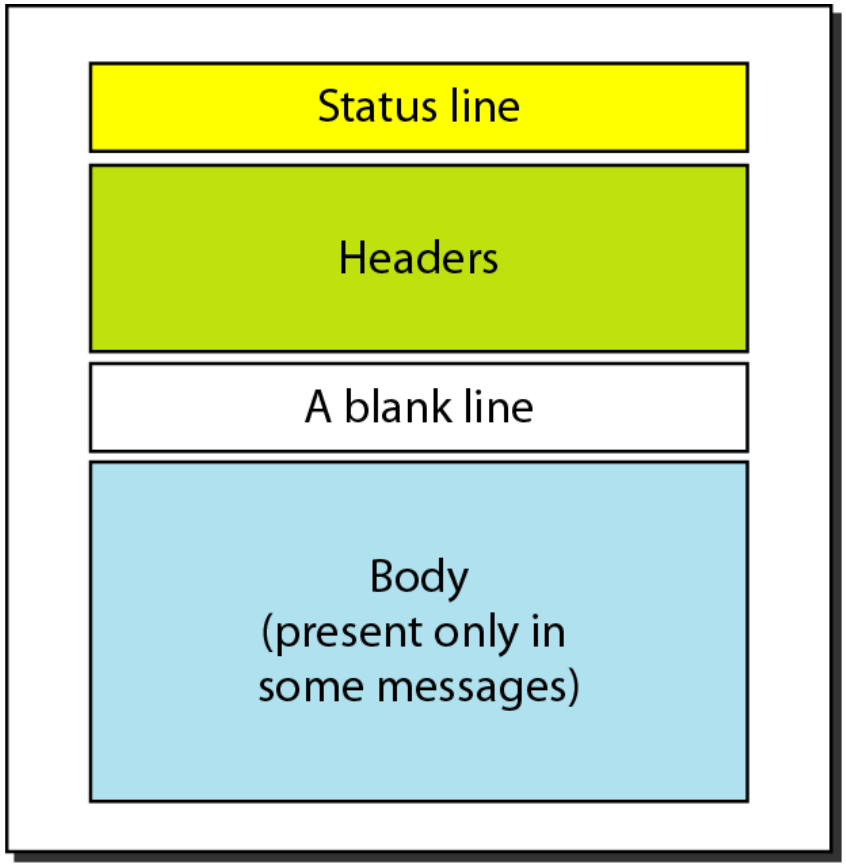
HTTP transaction



Request and response messages

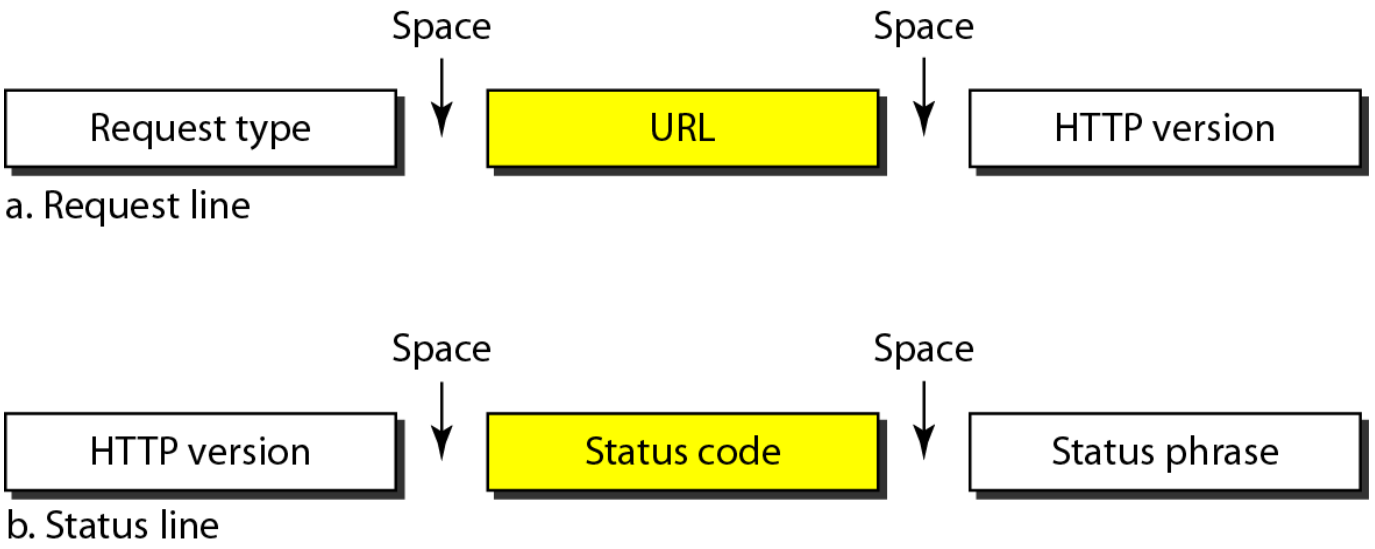


Request message



Response message

Request and status lines



Methods

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

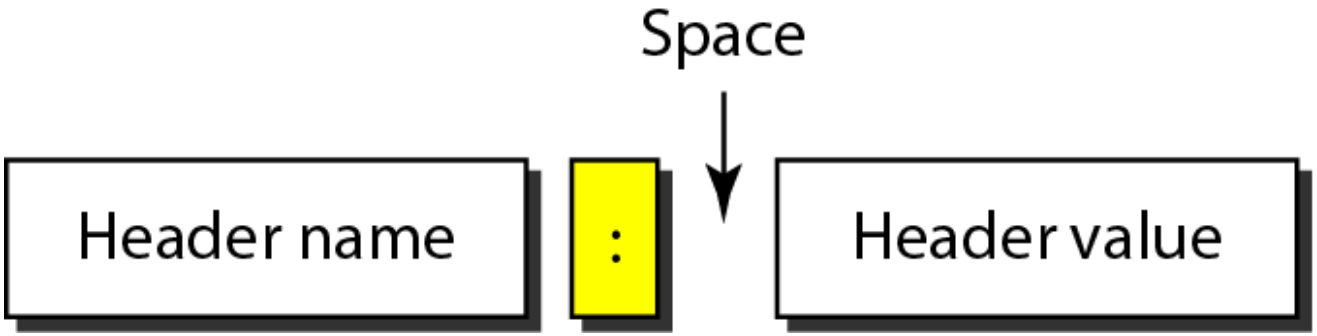
Status codes

Code	*Phrase*	*Description*
	Informational	
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
	Success	
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

Status codes (continued)

Code	*Phrase*	*Description*
	Redirection	
301	Moved permanently	The requested URL is no longer used by the server.
302	Moved temporarily	The requested URL has moved temporarily.
304	Not modified	The document has not been modified.
	Client Error	
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
	Server Error	
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable, but may be requested in the future.

Header format



General headers

Header	*Description*
Cache-control	Specifies information about caching
Connection	Shows whether the connection should be closed or not
Date	Shows the current date
MIME-version	Shows the MIME version used
Upgrade	Specifies the preferred communication protocol

Request headers

Header	*Description*
Accept	Shows the medium format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
From	Shows the e-mail address of the user
Host	Shows the host and port number of the server
If-modified-since	Sends the document if newer than specified date
If-match	Sends the document only if it matches given tag
If-non-match	Sends the document only if it does not match given tag
If-range	Sends only the portion of the document that is missing
If-unmodified-since	Sends the document if not changed since specified date
Referrer	Specifies the URL of the linked document
User-agent	Identifies the client program

Response headers

Header	*Description*
Accept-range	Shows if server accepts the range requested by client
Age	Shows the age of the document
Public	Shows the supported list of methods
Retry-after	Specifies the date after which the server is available
Server	Shows the server name and version number

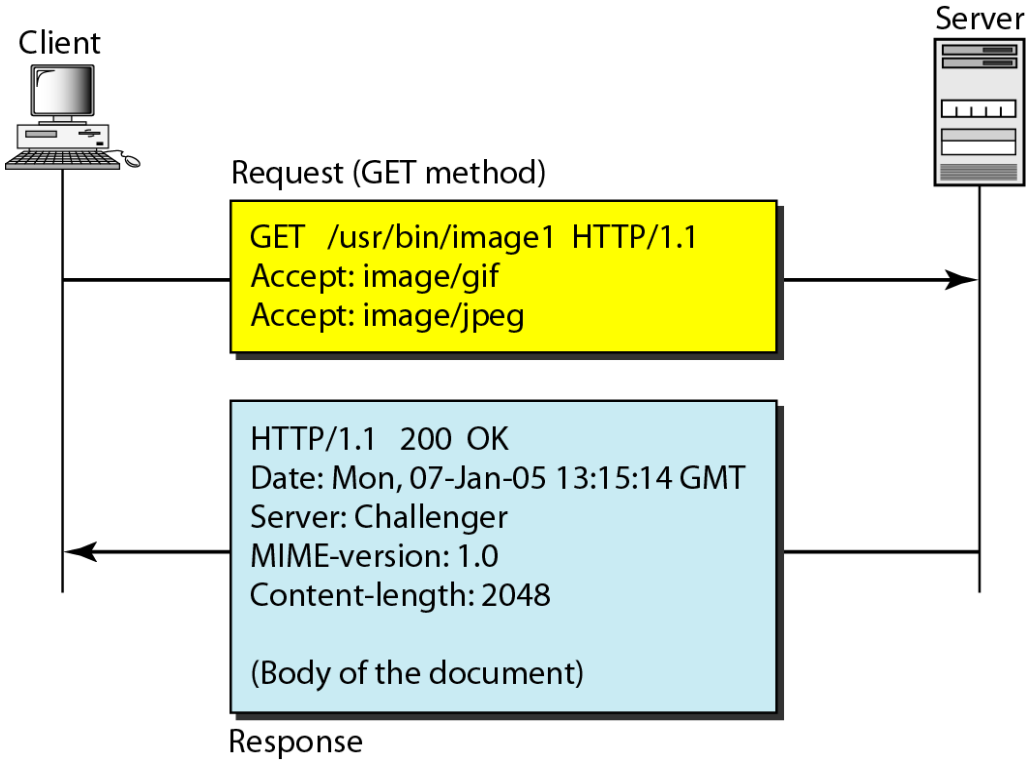
Entity headers

Header	*Description*
Allow	Lists valid methods that can be used with a URL
Content-encoding	Specifies the encoding scheme
Content-language	Specifies the language
Content-length	Shows the length of the document
Content-range	Specifies the range of the document
Content-type	Specifies the medium type
Etag	Gives an entity tag
Expires	Gives the date and time when contents may change
Last-modified	Gives the date and time of the last change
Location	Specifies the location of the created or moved document

Example

- This example retrieves a document.
- We use the GET method to retrieve an image with the path /usr/bin/image1.
  - The request line shows the method (GET), the URL, and the HTTP version (1.1).
  - The header has two lines that show that the client can accept images in the GIF or JPEG format.
  - The request does not have a body.
- The response message contains the status line and four lines of header.
  - The header lines define the date, server, MIME version, and length of the document.
  - The body of the document follows the header

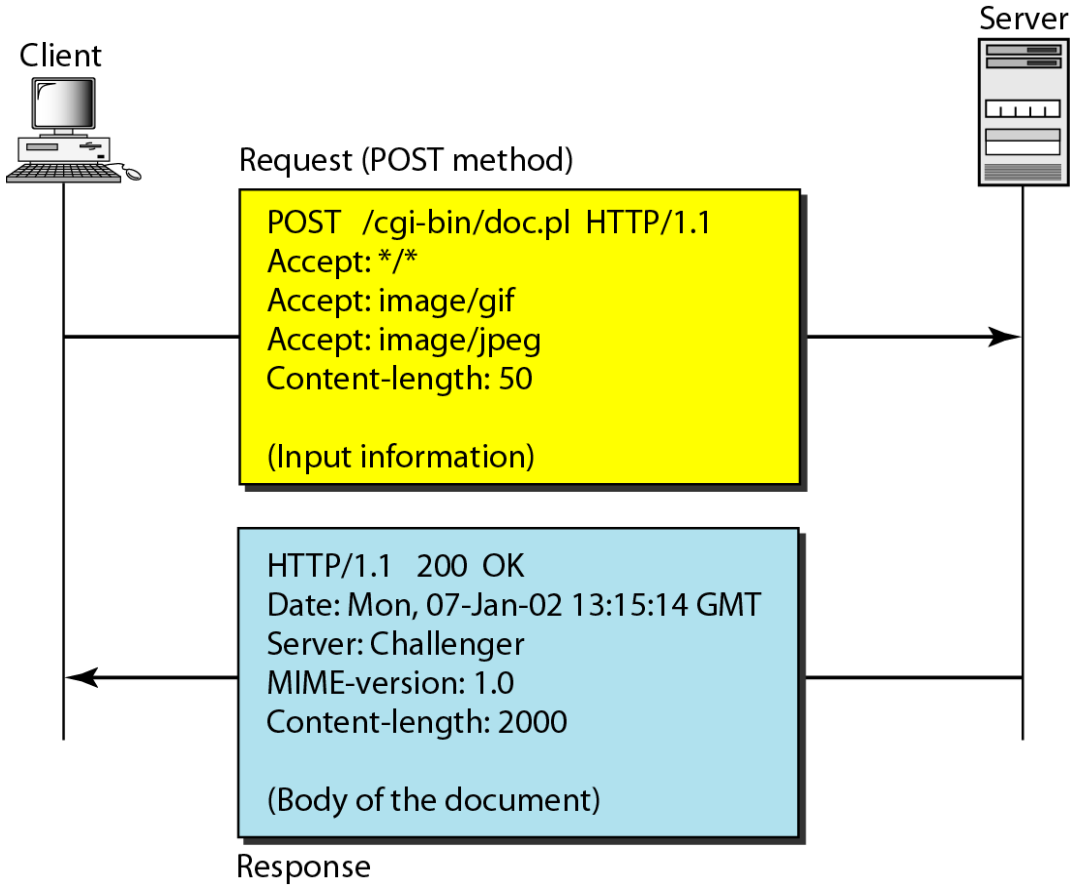
Example 27.1


Client
Server
Request (GET method)
GET /usr/bin/image1 HTTP/1.1
Accept: image/gif
Accept: image/jpeg
HTTP/1.1 200 OK
Date: Mon, 07-Jan-05 13:15:14 GMT
Server: Challenger
MIME-version: 1.0
Content-length: 2048
(Body of the document)
Response

Example

- In this example, the client wants to send data to the server.
- We use the POST method. The request line shows the method (POST), URL, and HTTP version (1.1).
- There are four lines of headers. The request body contains the input information.
- The response message contains the status line and four lines of headers.
- The created document, which is a CGI document, is included as the body.

Example 27.2


Client
Server
Request (POST method)
POST /cgi-bin/doc.pl HTTP/1.1
Accept: */*
Accept: image/gif
Accept: image/jpeg
Content-length: 50
(Input information)
HTTP/1.1 200 OK
Date: Mon, 07-Jan-02 13:15:14 GMT
Server: Challenger
MIME-version: 1.0
Content-length: 2000
(Body of the document)
Response

Persistent Versus Nonpersistent Connection

- **Nonpersistent Connection**
  - In a nonpersistent connection, one TCP connection is made for each request/response. The following lists the steps in this strategy:
    - The client opens a TCP connection and sends a request.
    - The server sends the response and closes the connection.
    - The client reads the data until it encounters an end-of-file marker; it then closes the connection.

Persistent Connection

- HTTP version 1.1 specifies a persistent connection by default.
- In a persistent connection, the server leaves the connection open for more requests after sending a response.
- The server can close the connection at the request of a client or if a time-out has been reached.
- The sender usually sends the length of the data with each response.
- But when a document is created dynamically or actively the server informs the client that the length is not known and closes the connection after sending the data so the client knows that the end of the data has been reached.

Note

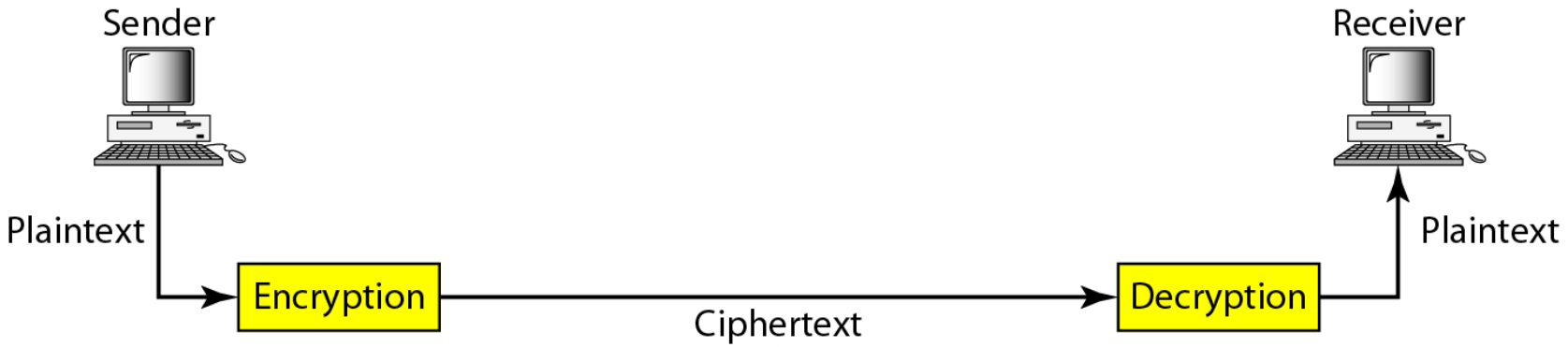
HTTP version 1.1 specifies a persistent connection by default.

Cryptography and Security

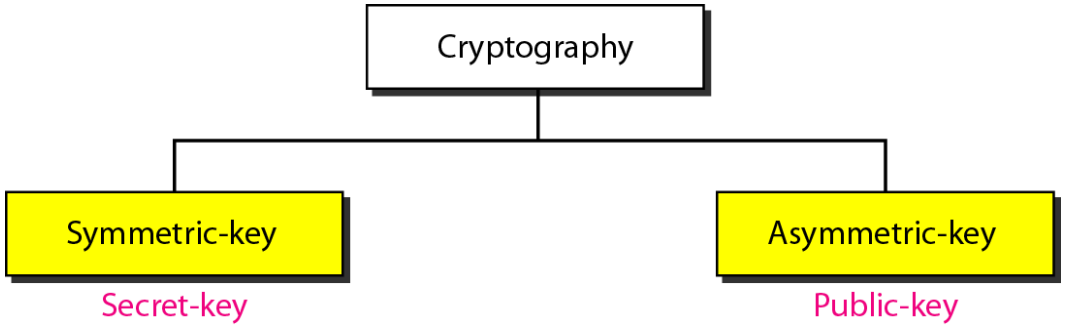
INTRODUCTION

Let us introduce the issues involved in cryptography. First, we need to define some terms; then we give some taxonomies.

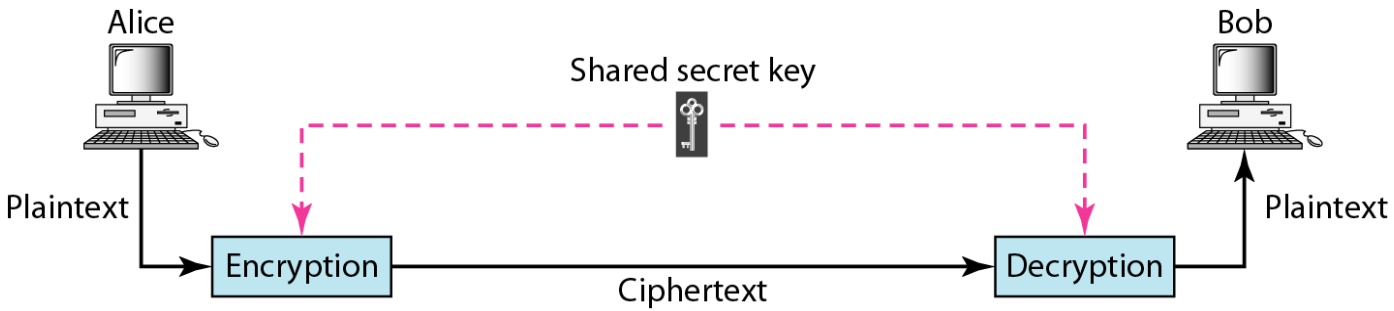
Cryptography components



Categories of cryptography



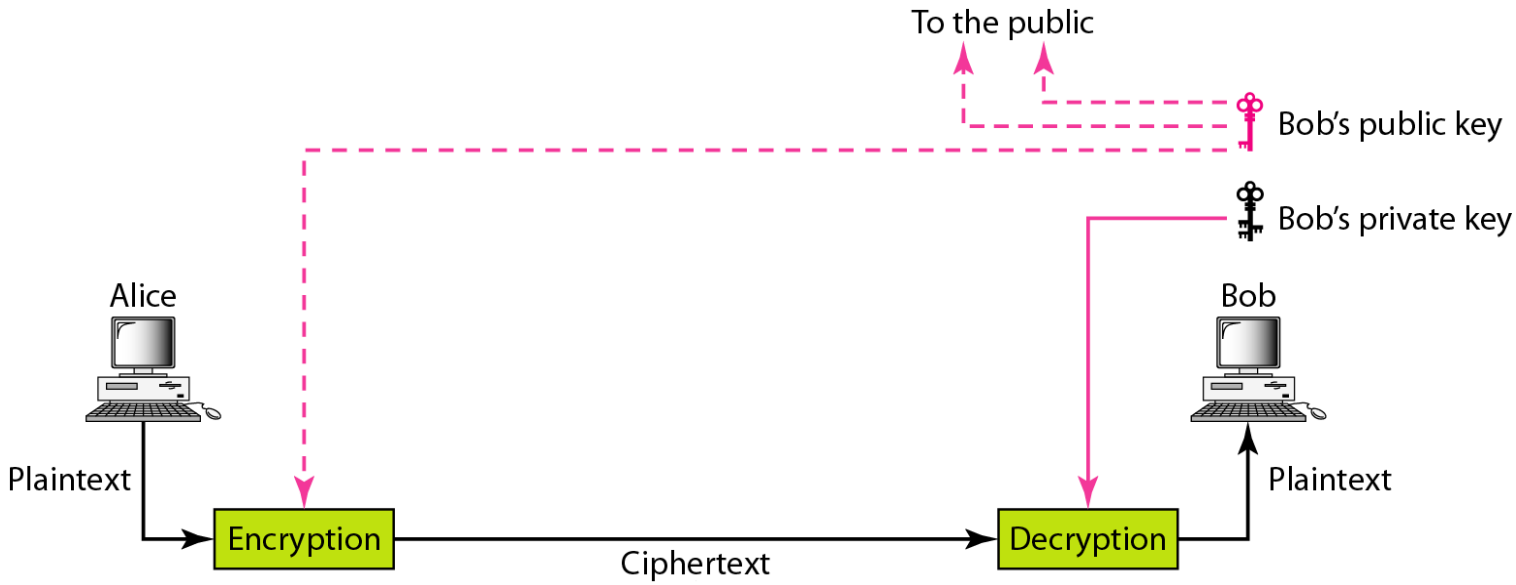
Symmetric-key cryptography



Note

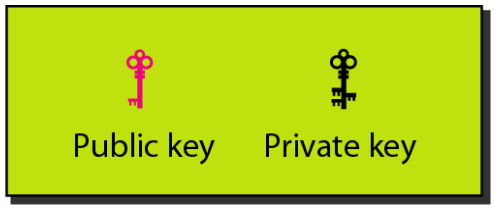
In symmetric-key cryptography, the same key is used by the sender (for encryption) and the receiver (for decryption). The key is shared.

Asymmetric-key cryptography



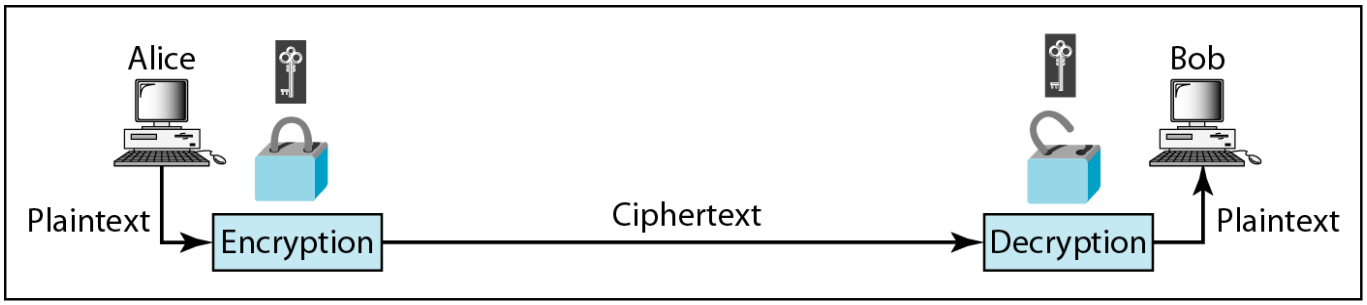
Keys used in cryptography

Symmetric-key cryptography

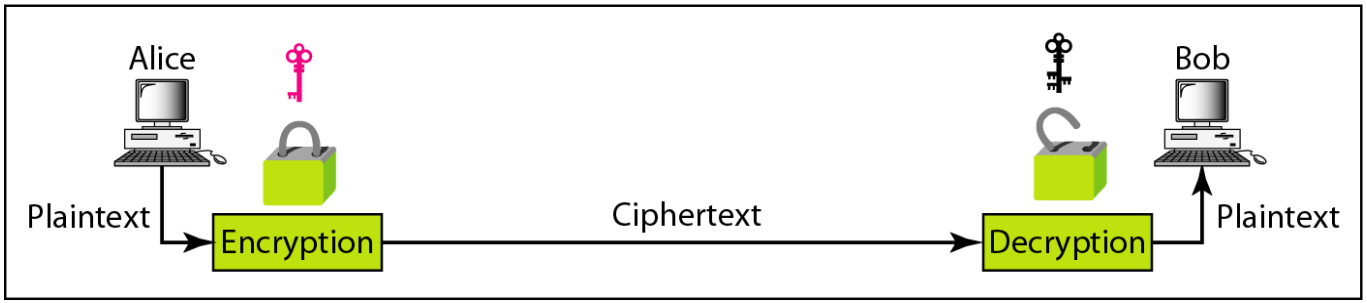


Asymmetric-key cryptography

Comparison between two categories of cryptography



a. Symmetric-key cryptography



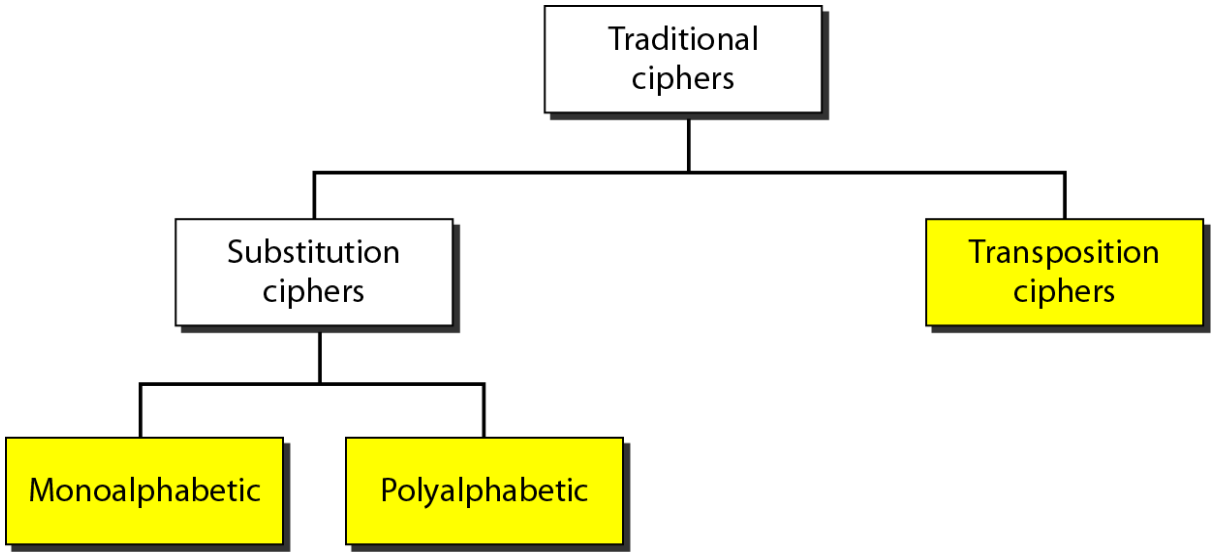
b. Asymmetric-key cryptography

SYMMETRIC-KEY CRYPTOGRAPHY

Symmetric-key cryptography started thousands of years ago when people needed to exchange secrets (for example, in a war). We still mainly use symmetric-key cryptography in our network security.

Traditional Ciphers
Simple Modern Ciphers
Modern Round Ciphers
Mode of Operation

Traditional ciphers



Note

A substitution cipher replaces one symbol with another.

Example

The following shows a plaintext and its corresponding ciphertext. Is the cipher monoalphabetic?

Plaintext: HELLO
Ciphertext: KHOOR

Solution

The cipher is probably monoalphabetic because both occurrences of L's are encrypted as O's.

Example

The following shows a plaintext and its corresponding ciphertext. Is the cipher monoalphabetic?

Plaintext: HELLO
Ciphertext: ABNZF

Solution

The cipher is not monoalphabetic because each occurrence of L is encrypted by a different character. The first L is encrypted as N; the second as Z.

Note

The shift cipher is sometimes referred to as the Caesar cipher.

Example

Use the shift cipher with key = 15 to encrypt the message "HELLO."

Solution

We encrypt one character at a time. Each character is shifted 15 characters down. Letter H is encrypted to W. Letter E is encrypted to T. The first L is encrypted to A. The second L is also encrypted to A. And O is encrypted to D. The cipher text is WTAAD.

Example

Use the shift cipher with key = 15 to decrypt the message "WTAAD."

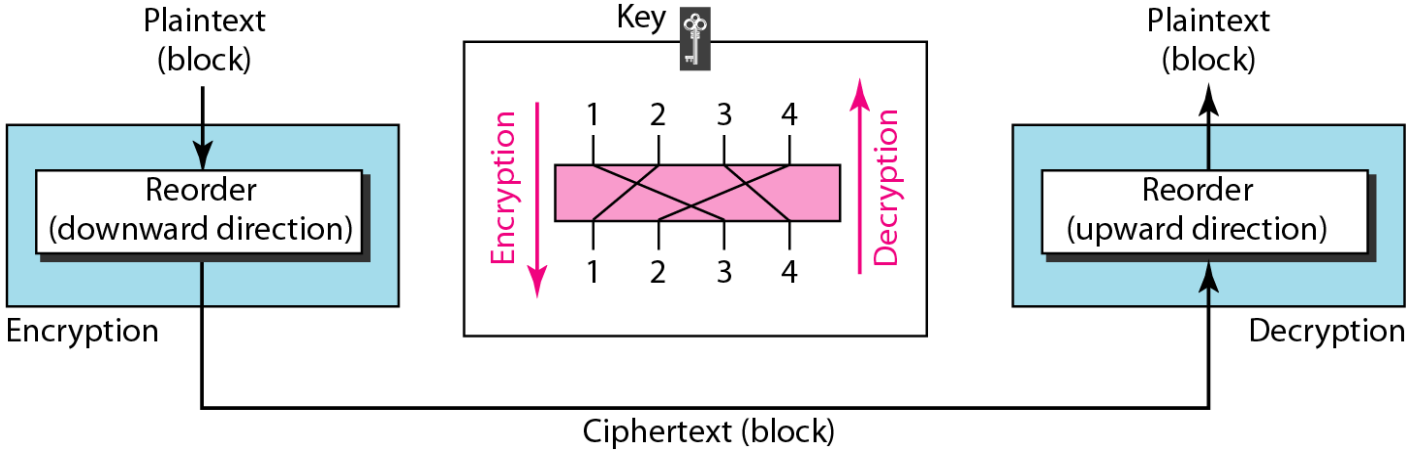
Solution

We decrypt one character at a time. Each character is shifted 15 characters up. Letter W is decrypted to H. Letter T is decrypted to E. The first A is decrypted to L. The second A is decrypted to L. And, finally, D is decrypted to O. The plaintext is HELLO.

Note

A transposition cipher reorders (permutes) symbols in a block of symbols.

Transposition cipher



Example

Encrypt the message "HELLO MY DEAR," using the key in previous slide.

Solution

We first remove the spaces in the message. We then divide the text into blocks of four characters. We add a bogus character Z at the end of the third block. The result is HELL OMYD EARZ. We create a three-block ciphertext ELHLMDOYAZER.

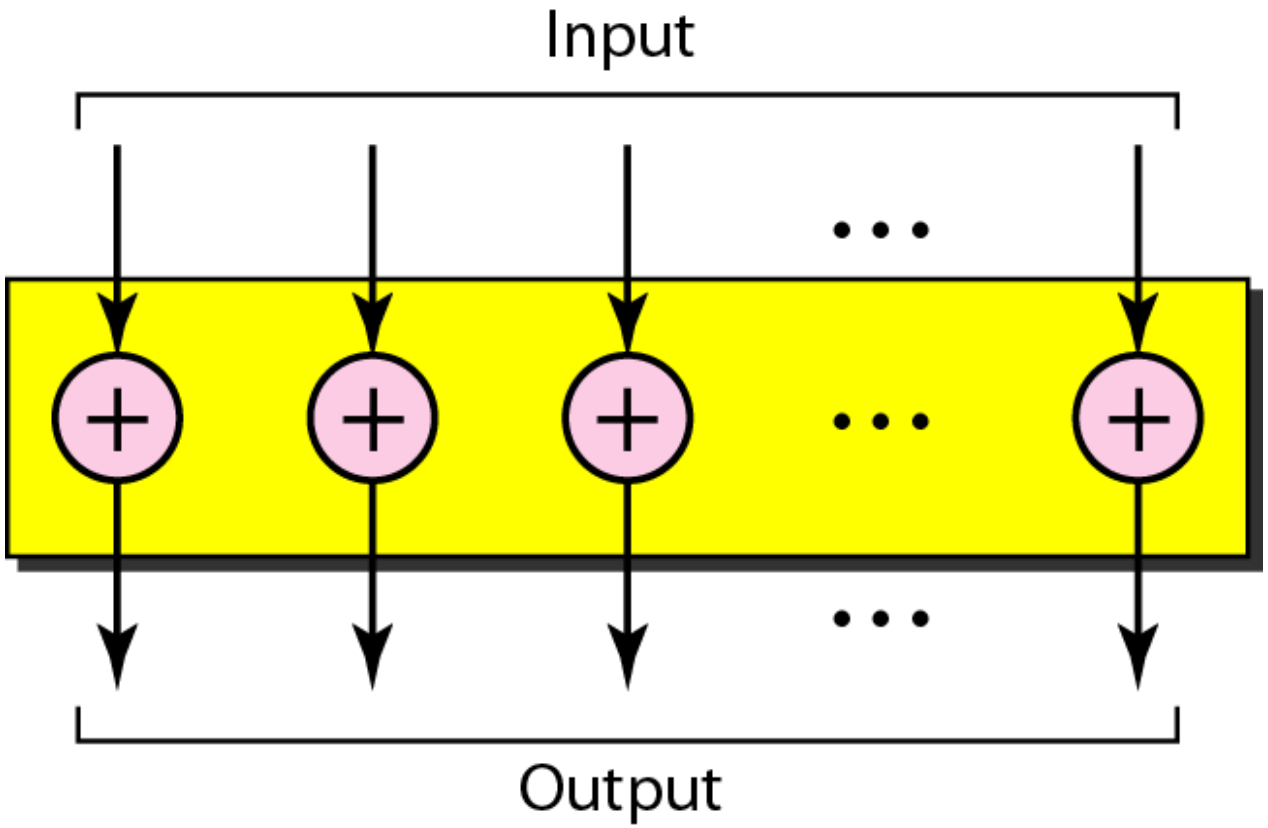
Example

Using previous Example, decrypt the message "ELHLMDOYAZER".

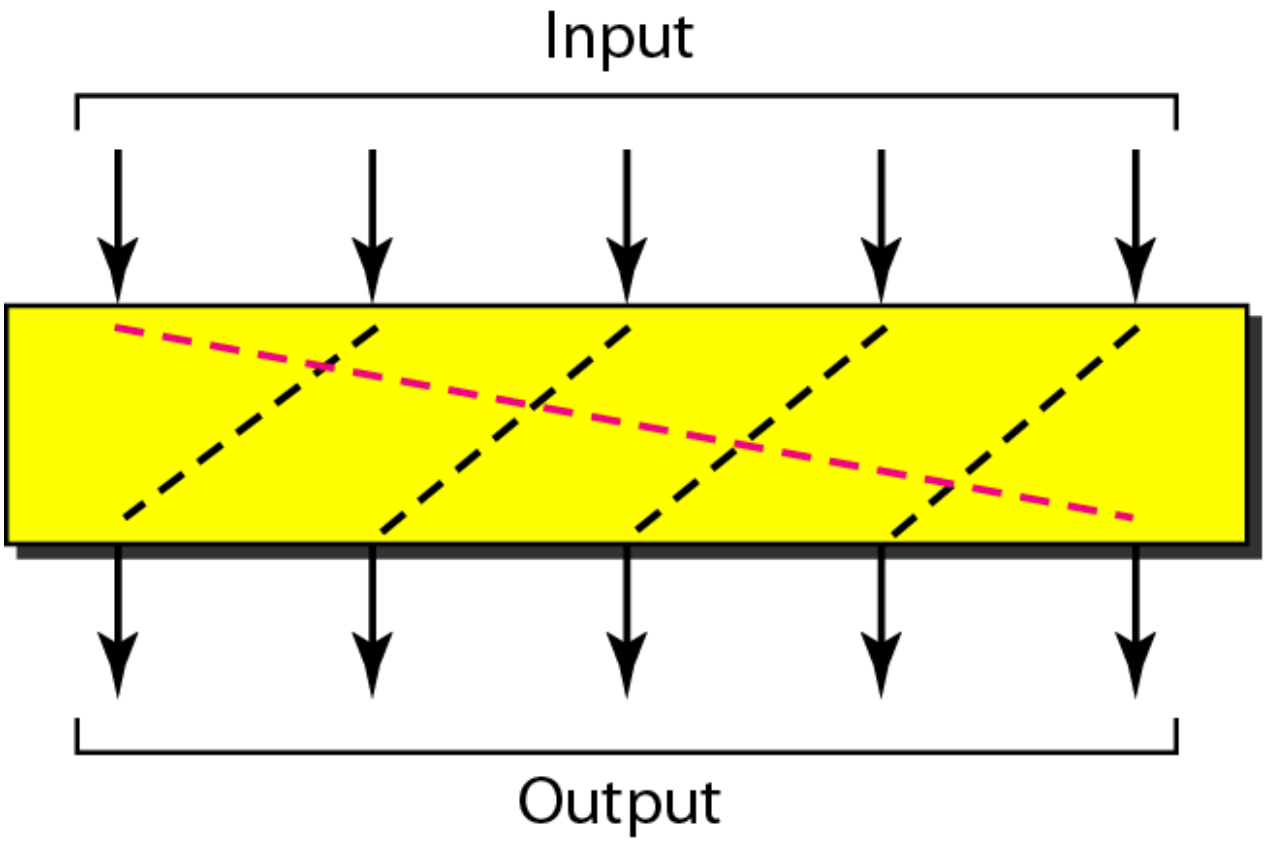
Solution

The result is HELL OMYD EARZ. After removing the bogus character and combining the characters, we get the original message "HELLO MY DEAR."

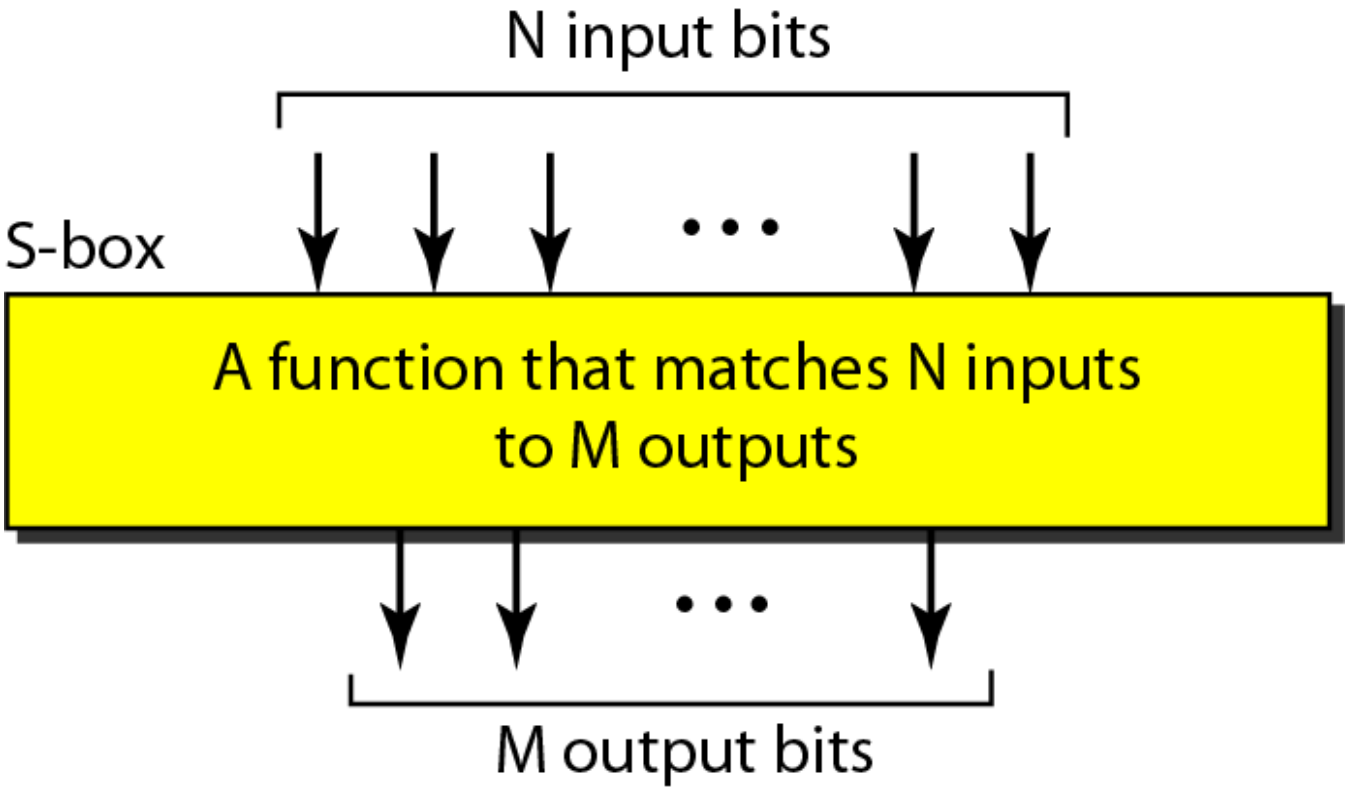
XOR cipher



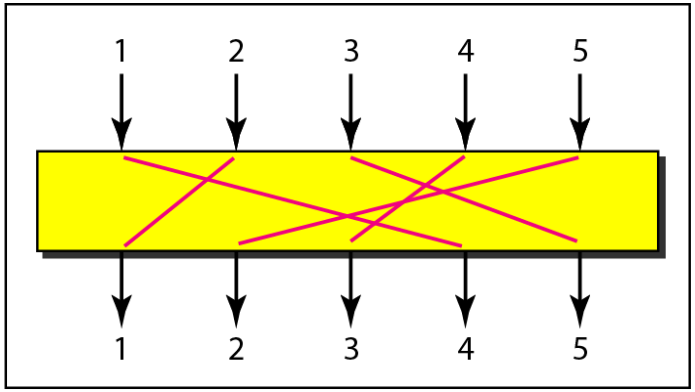
Rotation cipher



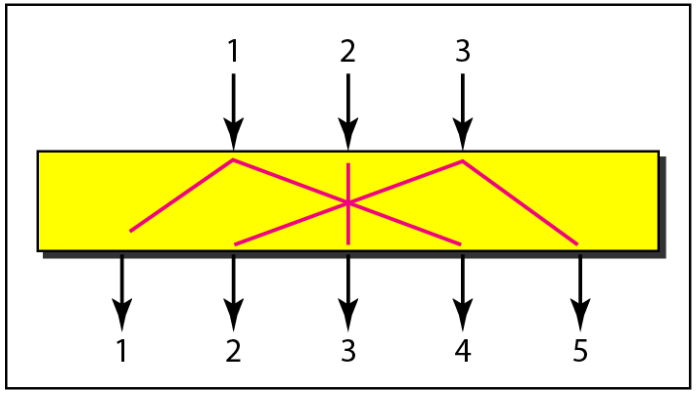
S-box


N input bits
S-box
A function that matches N inputs
to M outputs
M output bits

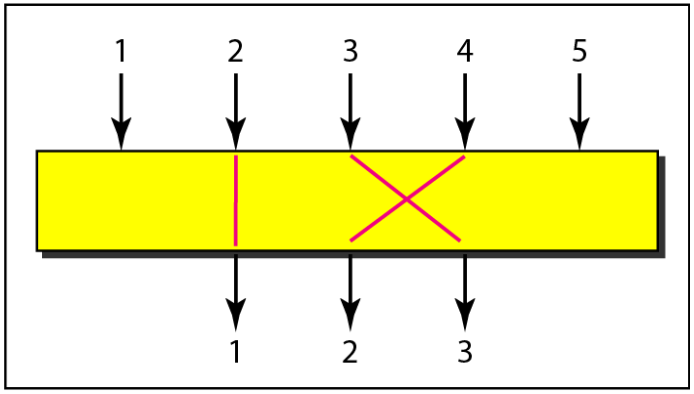
P-boxes: straight, expansion, and compression



a. Straight

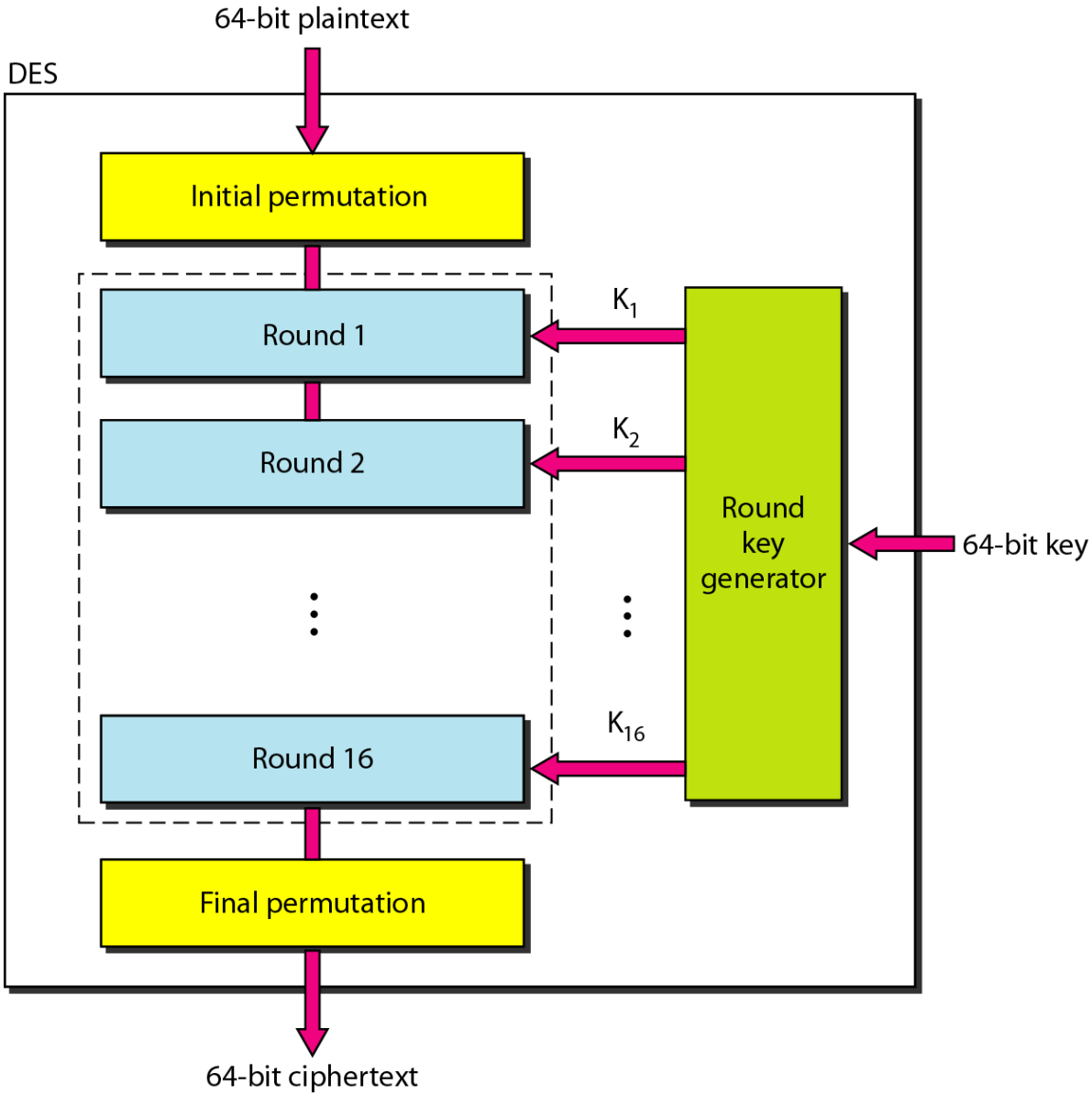


b. Expansion

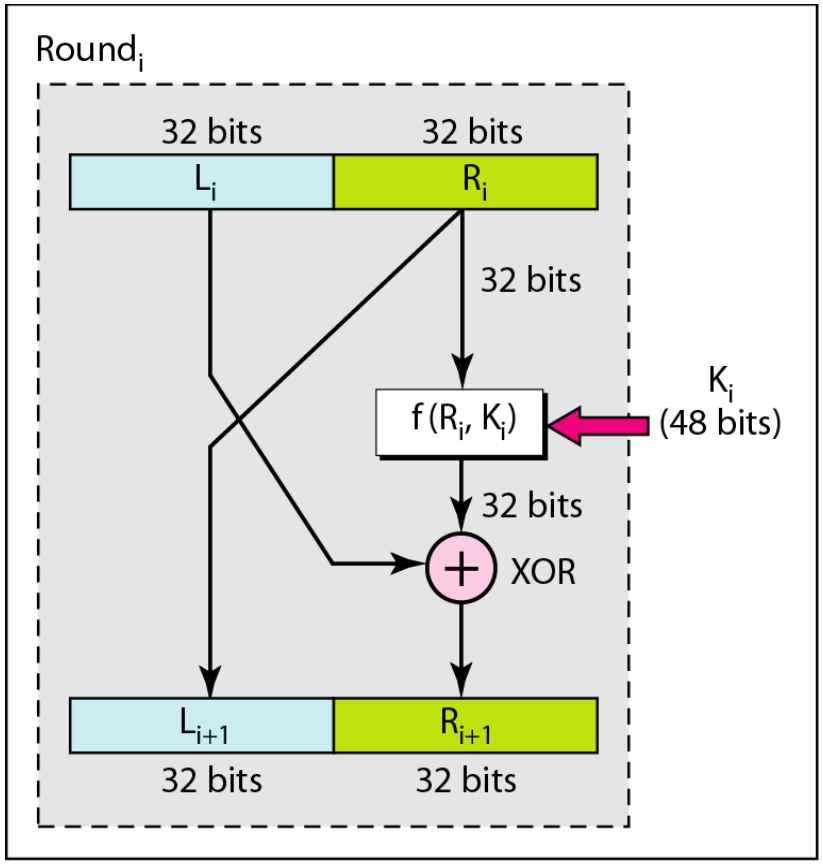


c. Compression

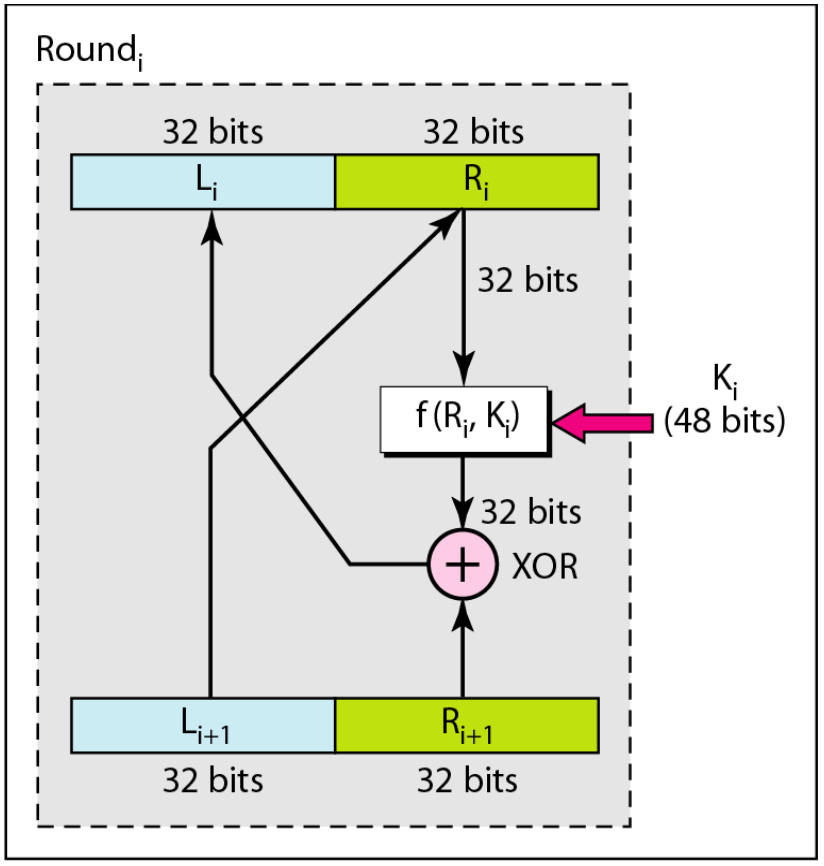
Data Encryption Standard (DES)



One round in DES ciphers

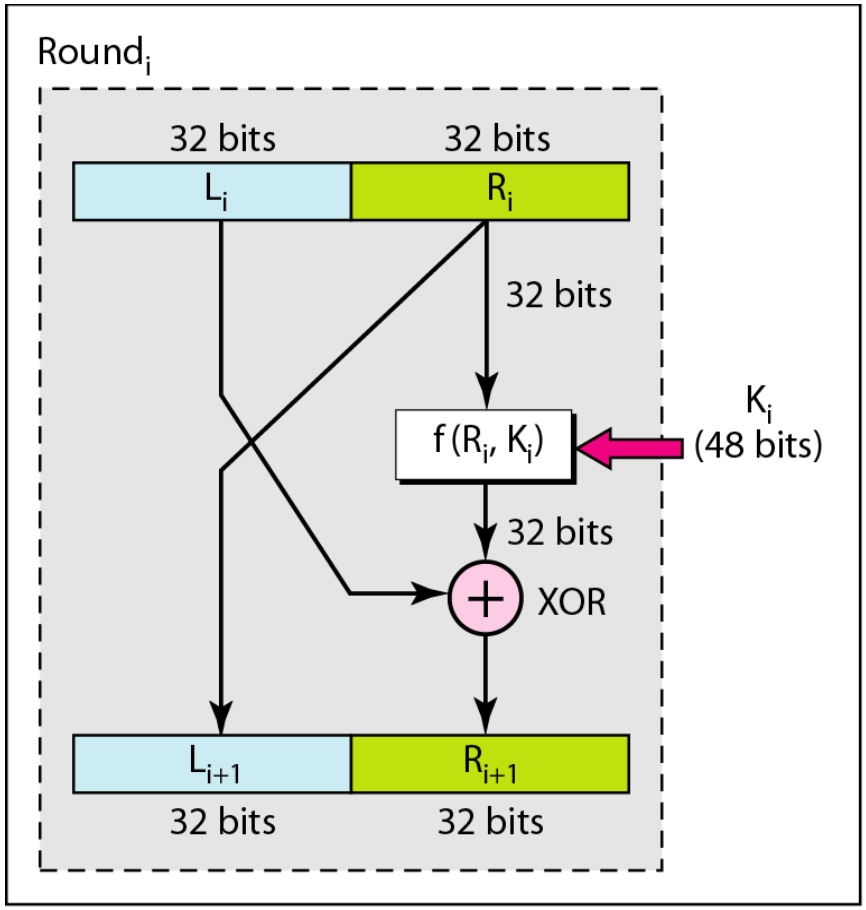


a. Encryption round

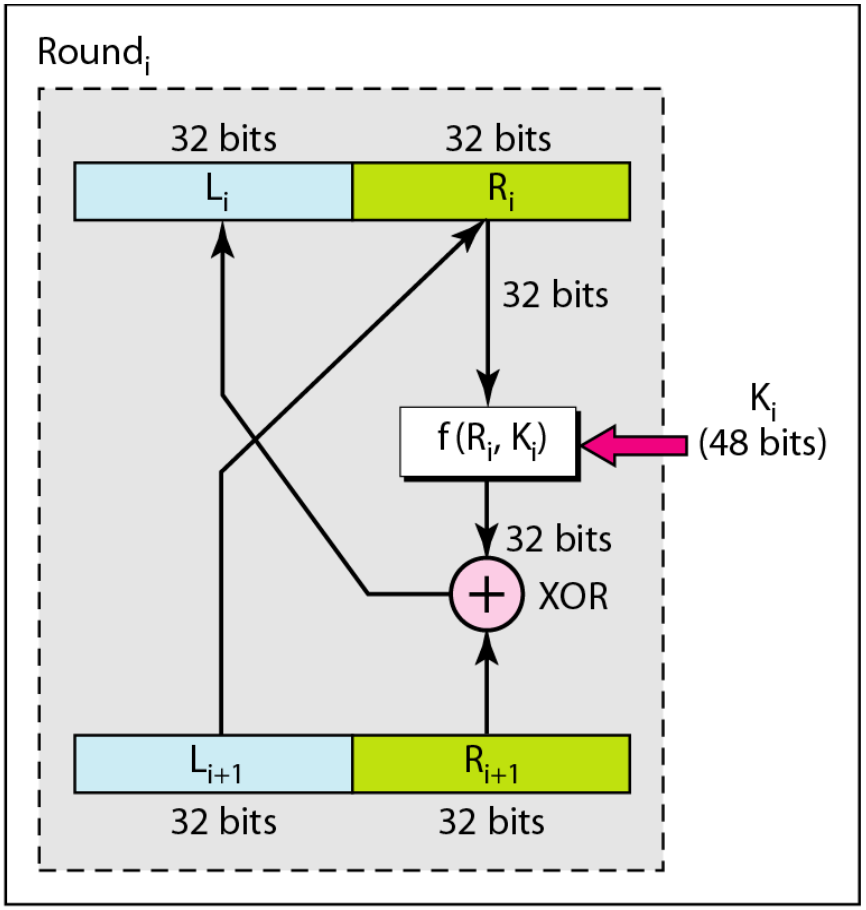


b. Decryption round

DES function

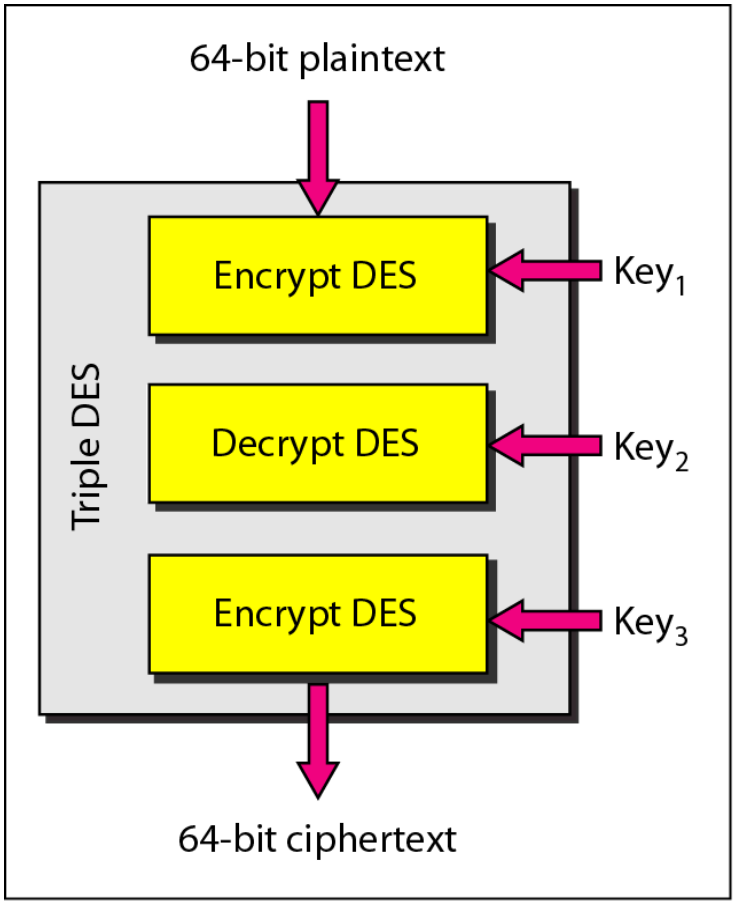


a. Encryption round

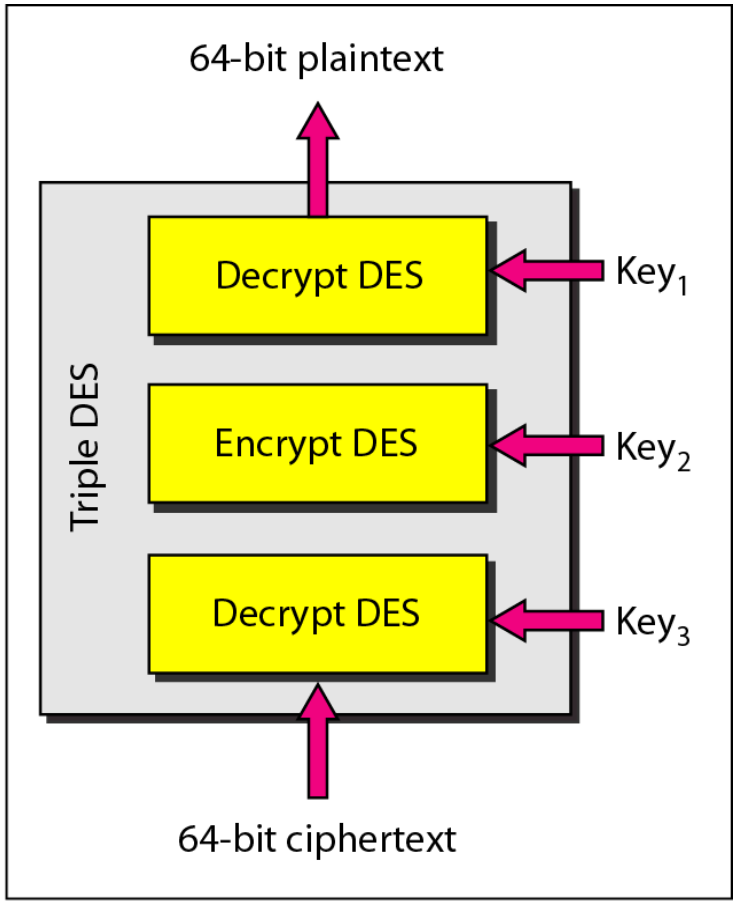


b. Decryption round

Triple DES



a. Encryption Triple DES



b. Decryption Triple DES

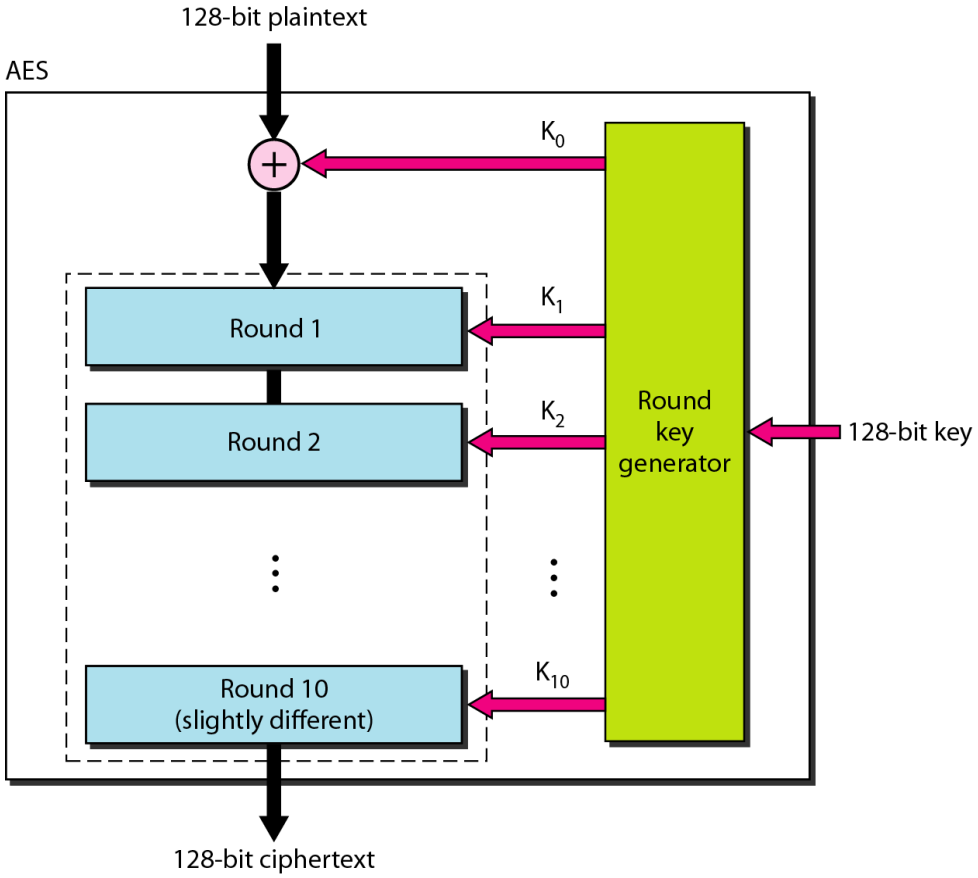
AES configuration

Size of Data Block	*Number of Rounds*	*Key Size*
128 bits	10	128 bits
	12	192 bits
	14	256 bits

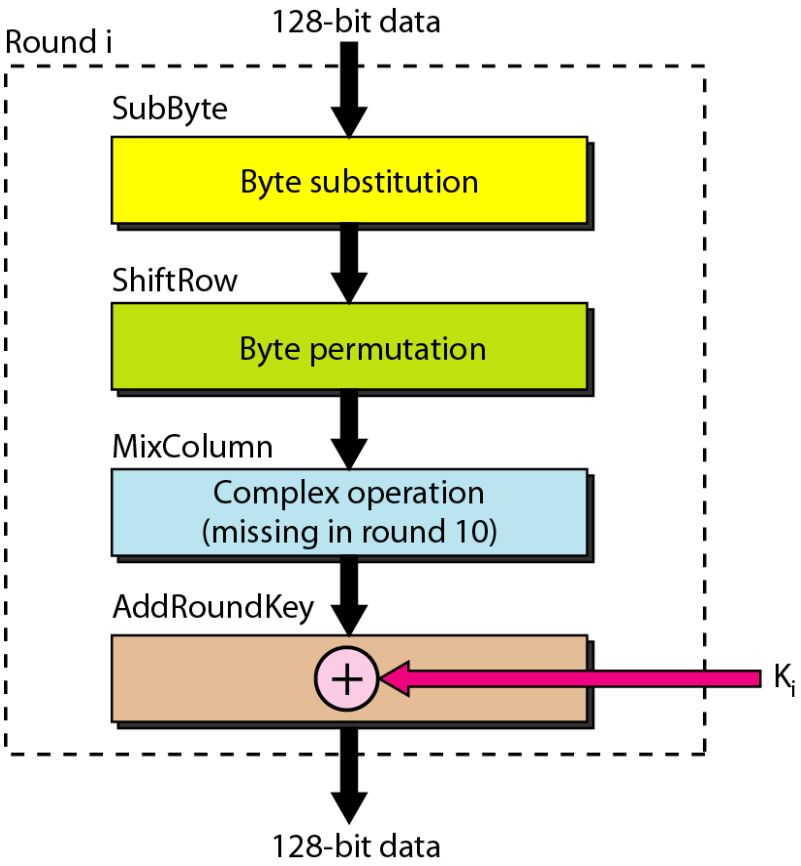
Note

AES has three different configurations with respect to the number of rounds and key size.

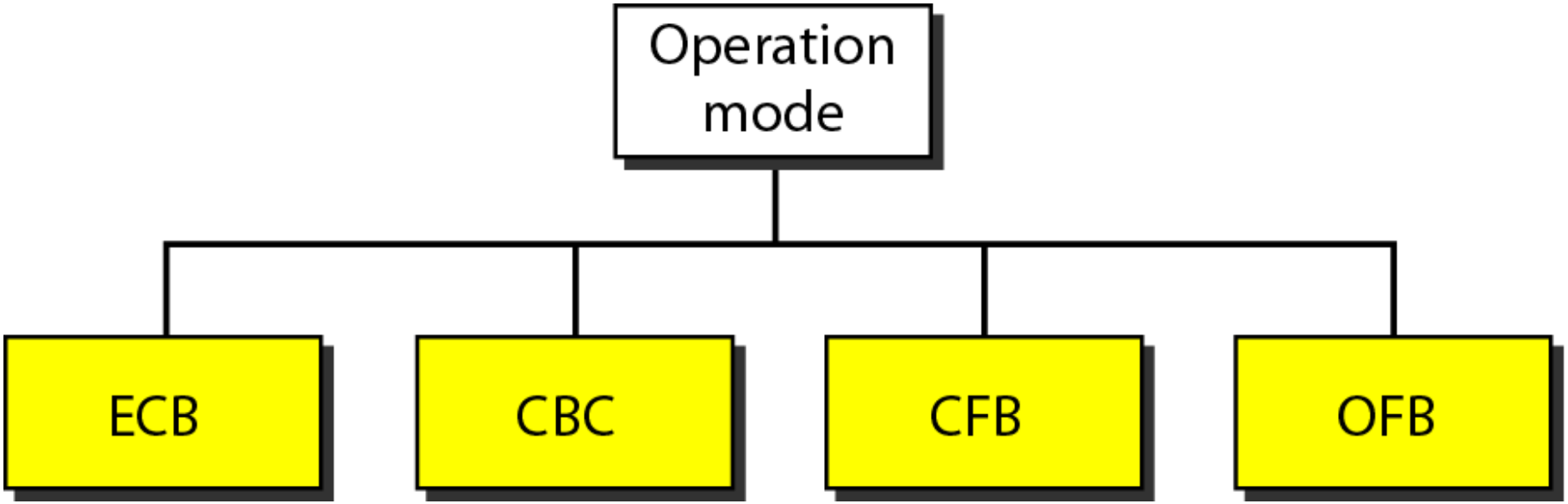
AES


128-bit plaintext
AES
K_0
K_1
K_2
K_{10}
Round 1
Round 2
Round 10
(slightly different)
Round
key
generator
128-bit key
128-bit ciphertext

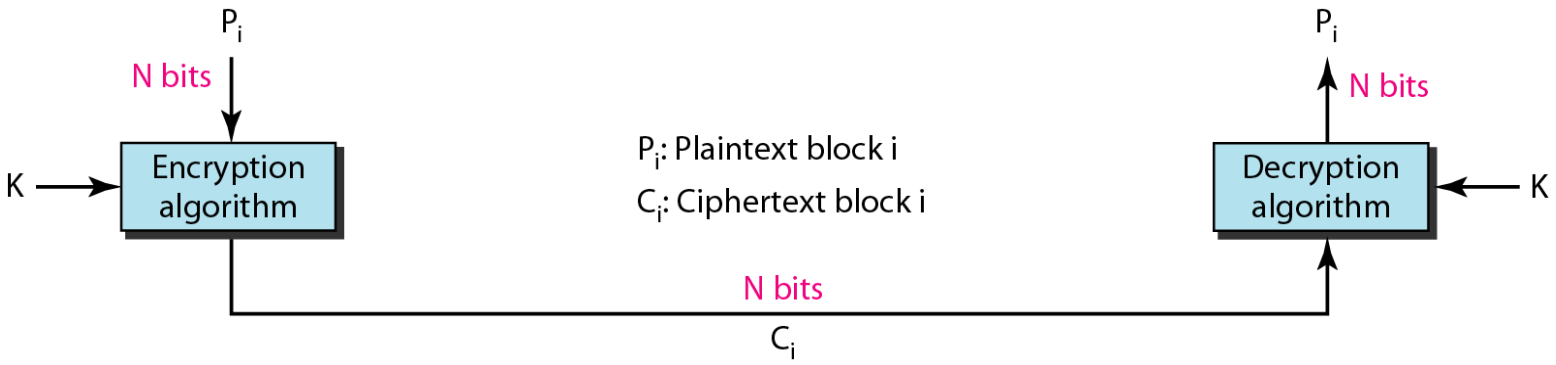
Structure of each round



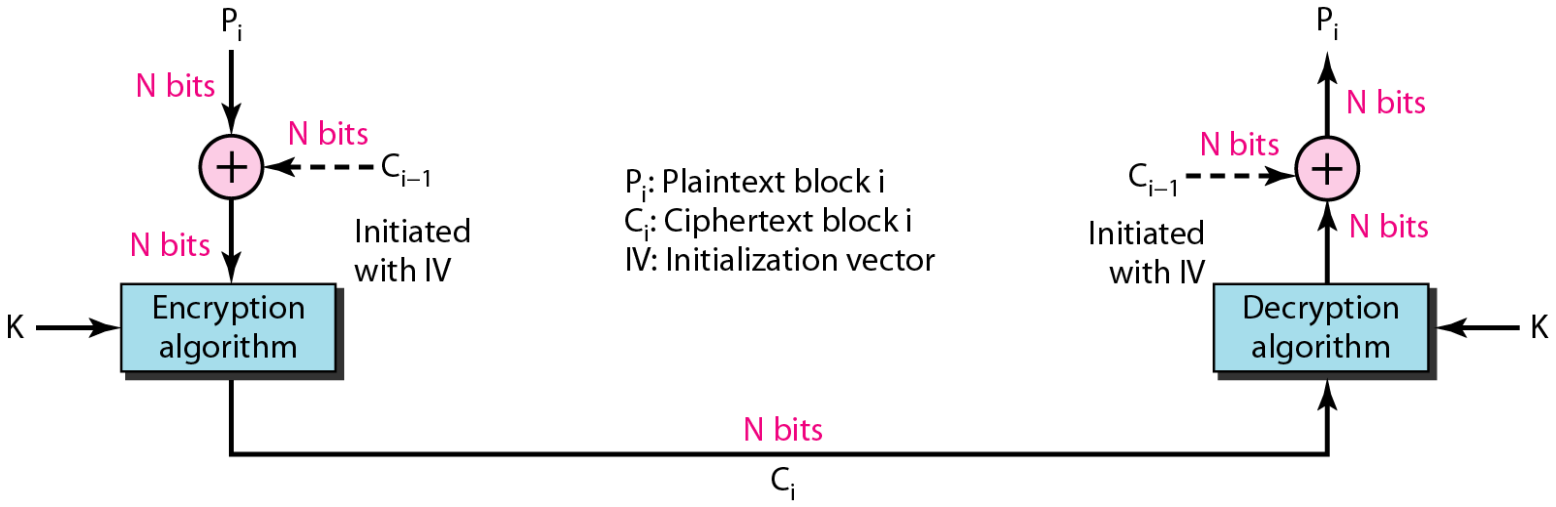
Modes of operation for block ciphers



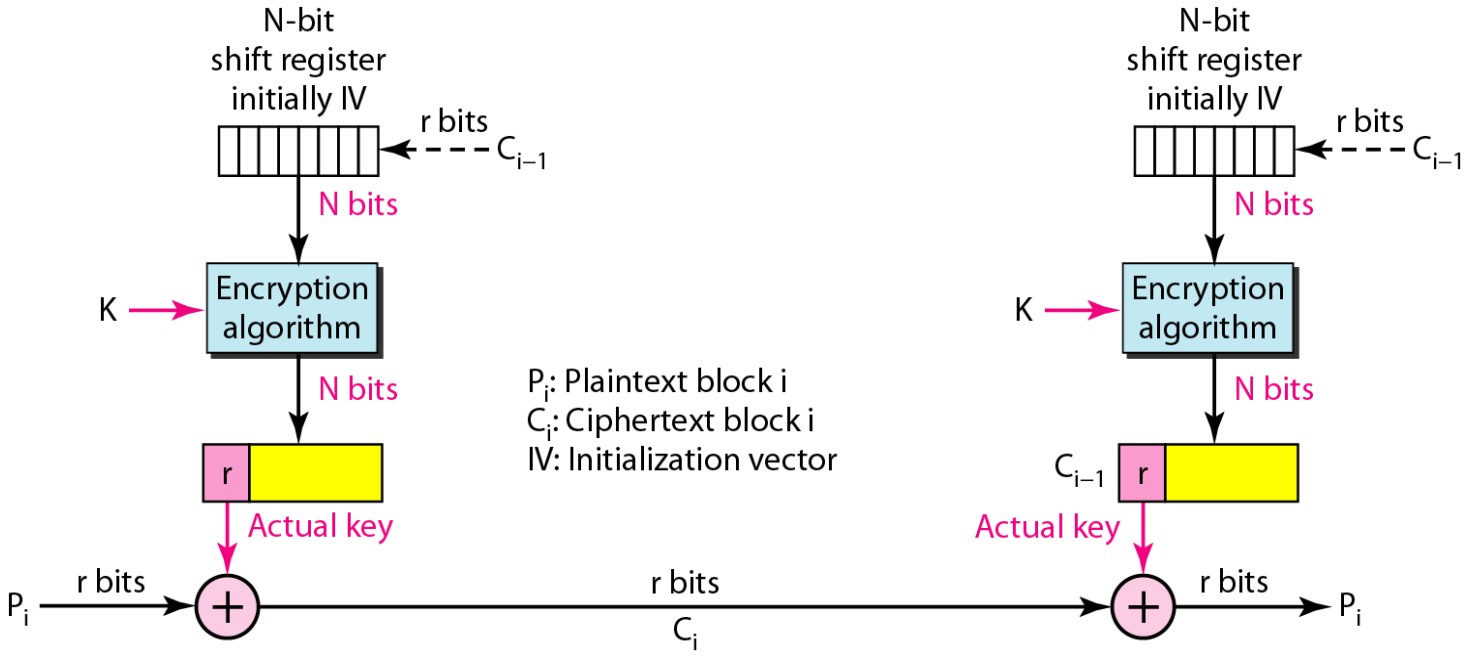
ECB mode


P_i
N bits
K
Encryption algorithm
P_i: Plaintext block i
C_i: Ciphertext block i
N bits
C_i
P_i
N bits
Decryption algorithm
K

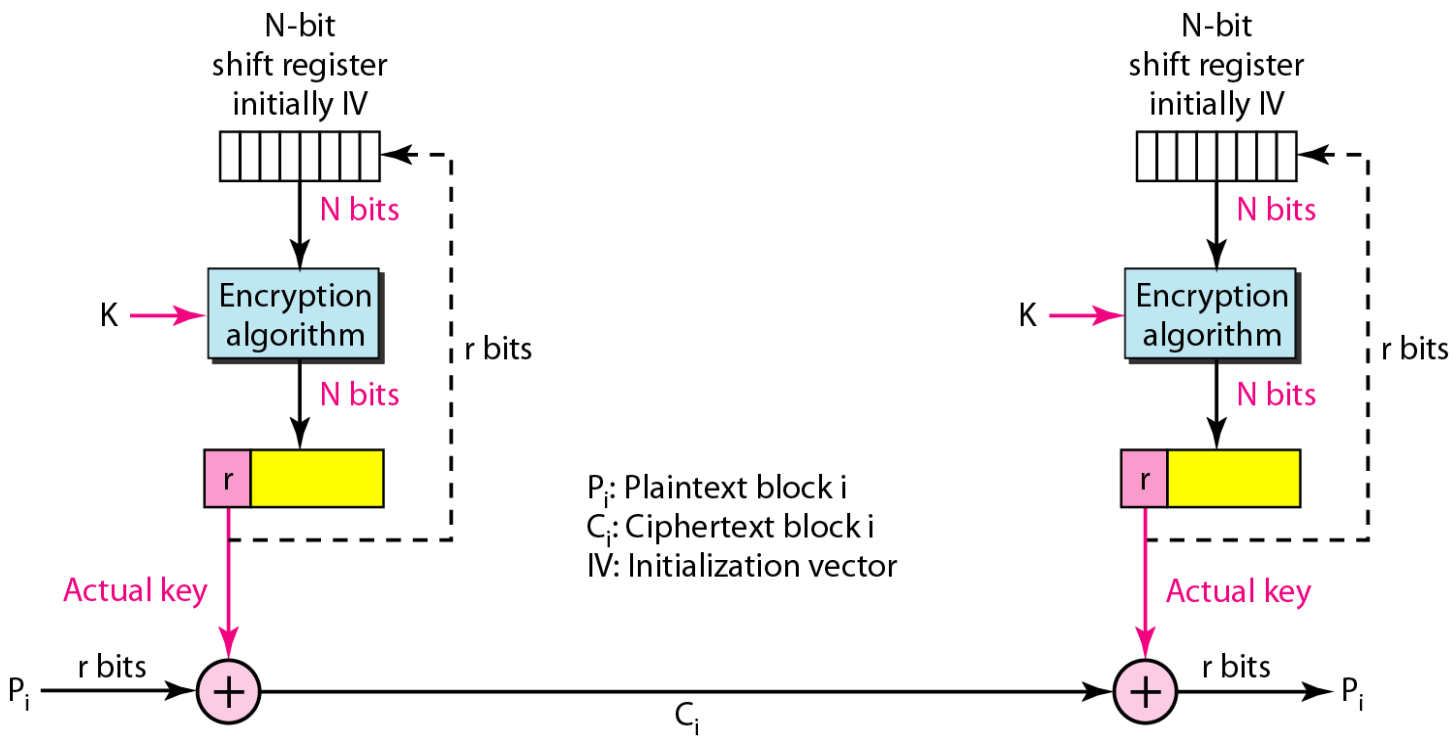
CBC mode


P_i
N bits
N bits
C_{i-1}
N bits
Initiated with IV
K
Encryption algorithm
P_i: Plaintext block i
C_i: Ciphertext block i
IV: Initialization vector
P_i
N bits
N bits
C_{i-1}
Initiated with IV
N bits
Decryption algorithm
K
N bits
C_i

CFB mode


N-bit
shift register
initially IV
r bits
C_{i-1}
N bits
K
Encryption
algorithm
N bits
r
Actual key
P_i
r bits
P_i: Plaintext block i
C_i: Ciphertext block i
IV: Initialization vector
r bits
C_i
N-bit
shift register
initially IV
r bits
C_{i-1}
N bits
K
Encryption
algorithm
N bits
C_{i-1}
r
Actual key
r bits
P_i

OFB mode


N-bit
shift register
initially IV
N bits
K
Encryption
algorithm
N bits
r bits
r
Actual key
P_i
r bits
C_i
P_i: Plaintext block i
C_i: Ciphertext block i
IV: Initialization vector
N-bit
shift register
initially IV
N bits
K
Encryption
algorithm
N bits
r bits
r
Actual key
r bits
P_i