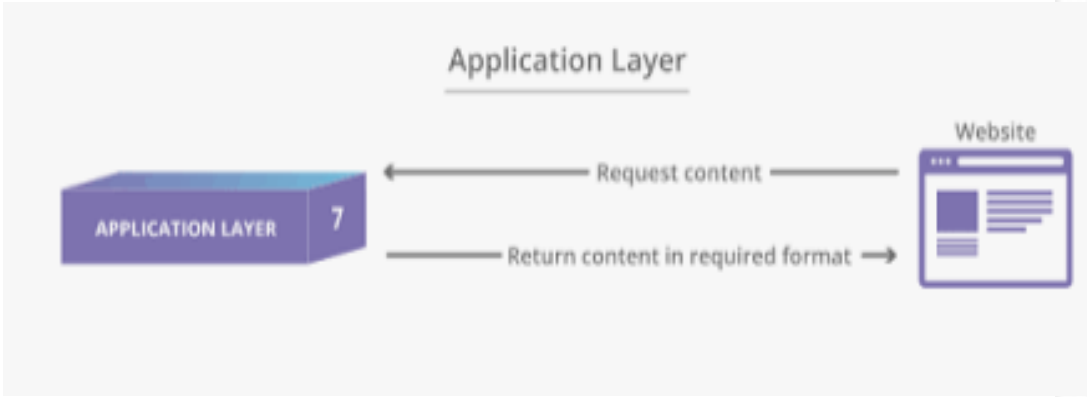
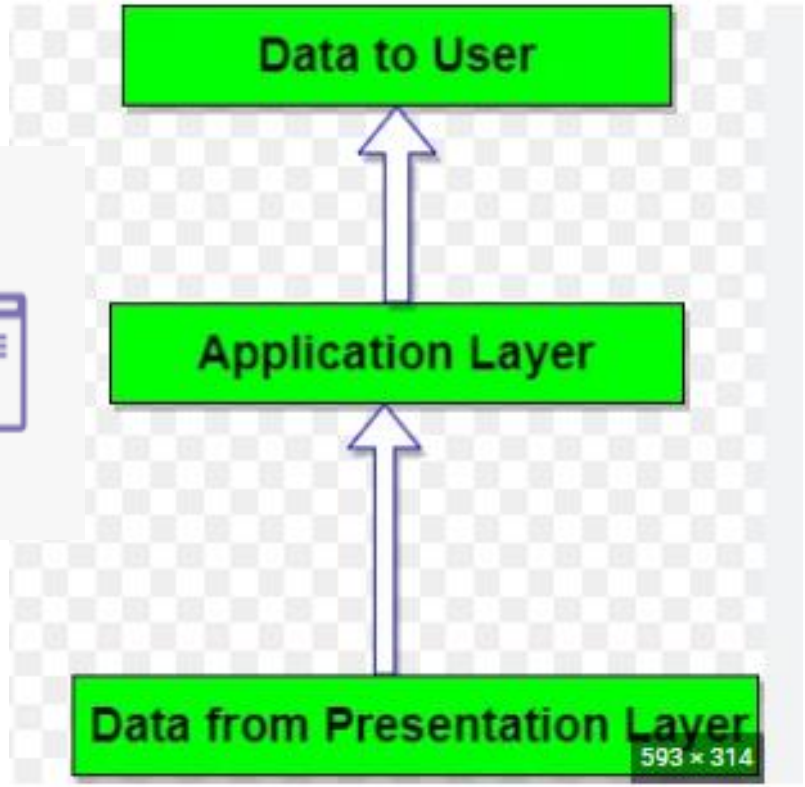
APPLICATION LAYER

Application Layer

- Topmost layer in OSI model that deals different application running over the system using different application layer protocols such as HTTP, FTP, TELNET etc.

- It ensures an application can effectively communicate with other applications on different computer systems and networks.

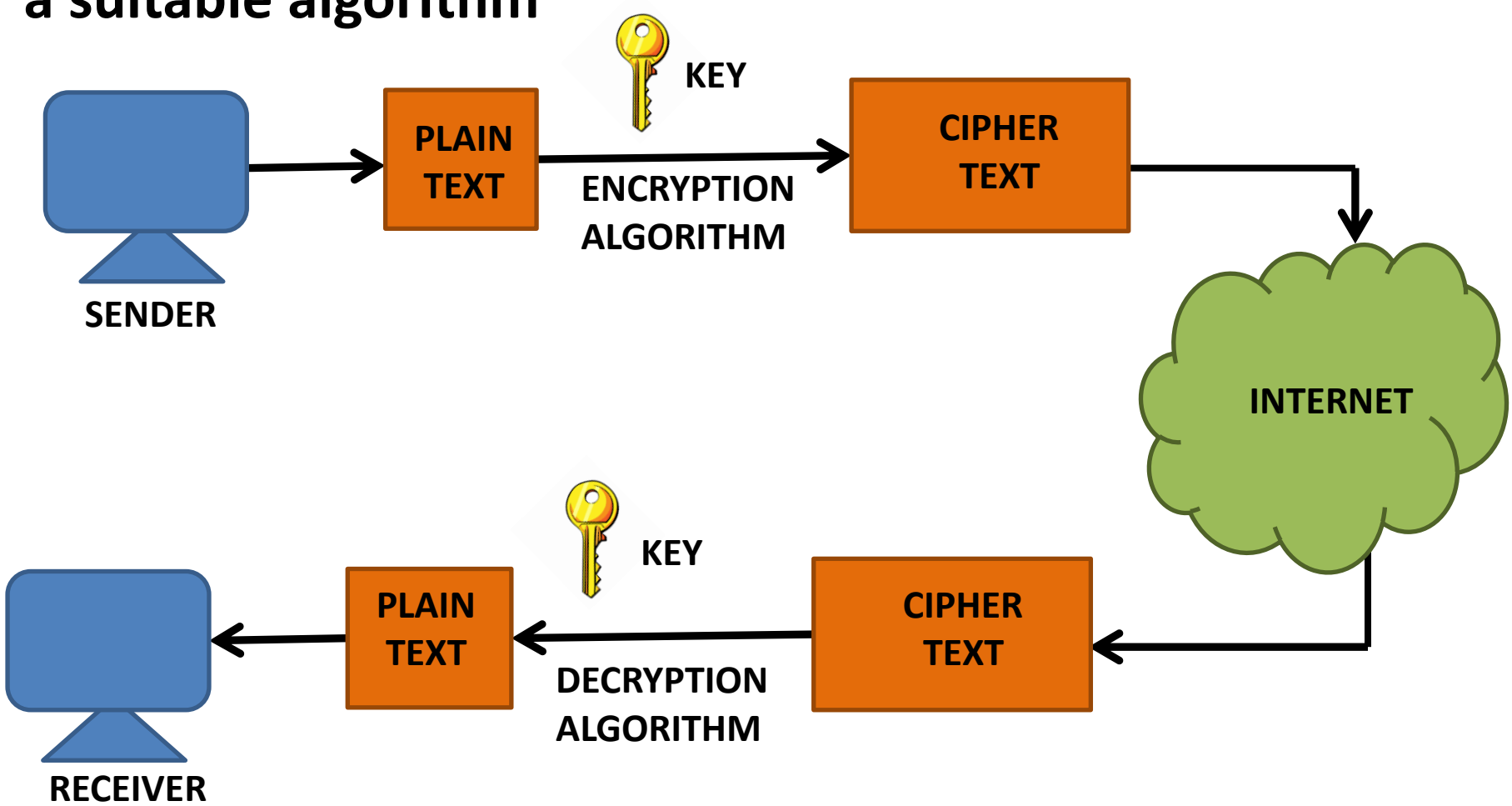




Data Encryption

- Conversion of our Readable form data (Plain Text) into non Readable form(Cipher Text) are known as Data Encryption

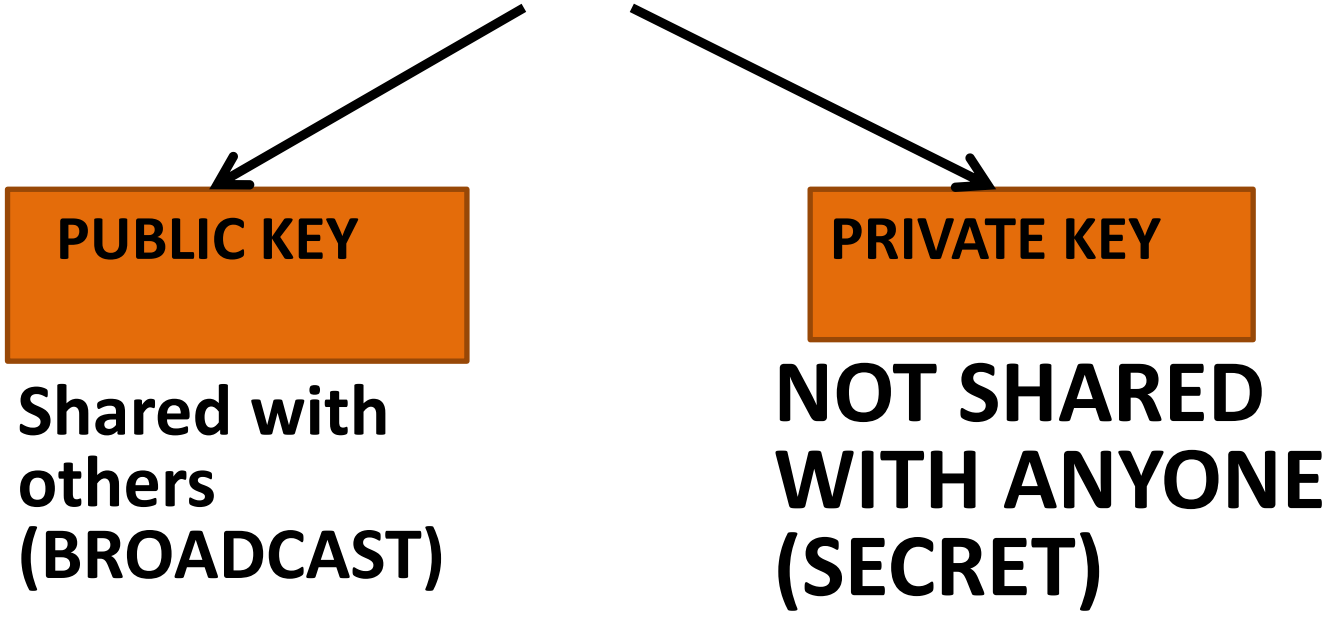
- Data Encryption can be performed by using keys and a suitable algorIthm



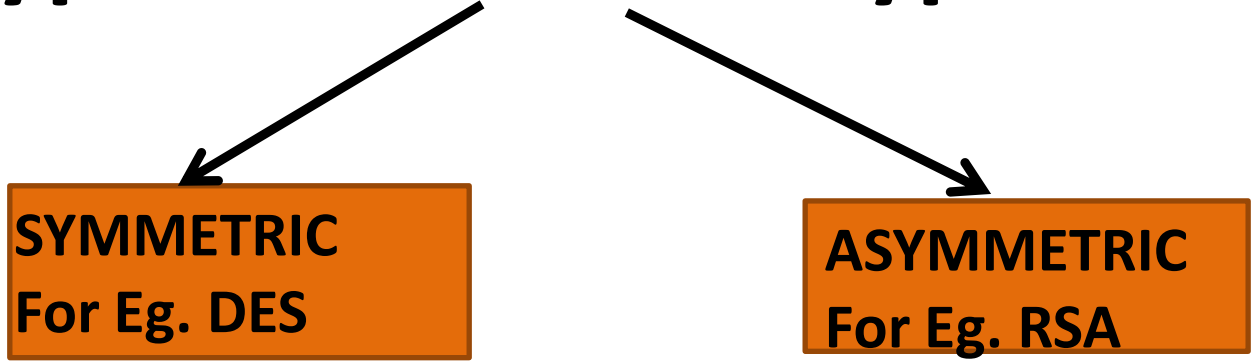
Data Encryption Keys

Data Encryption keys are value or property by which we can perform encryption and decryption.

Types of Encryption Keys

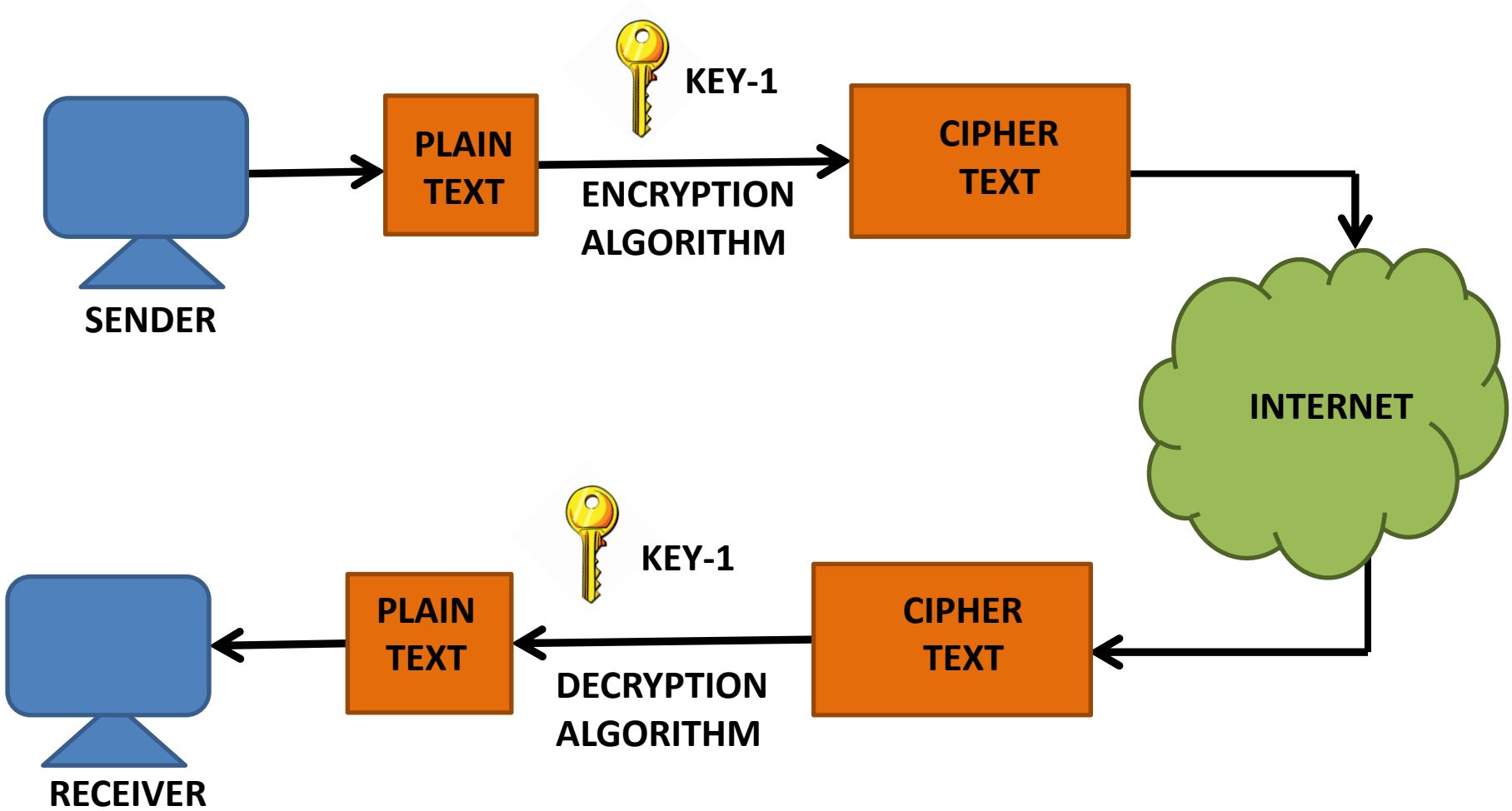


Types of Data Encryption

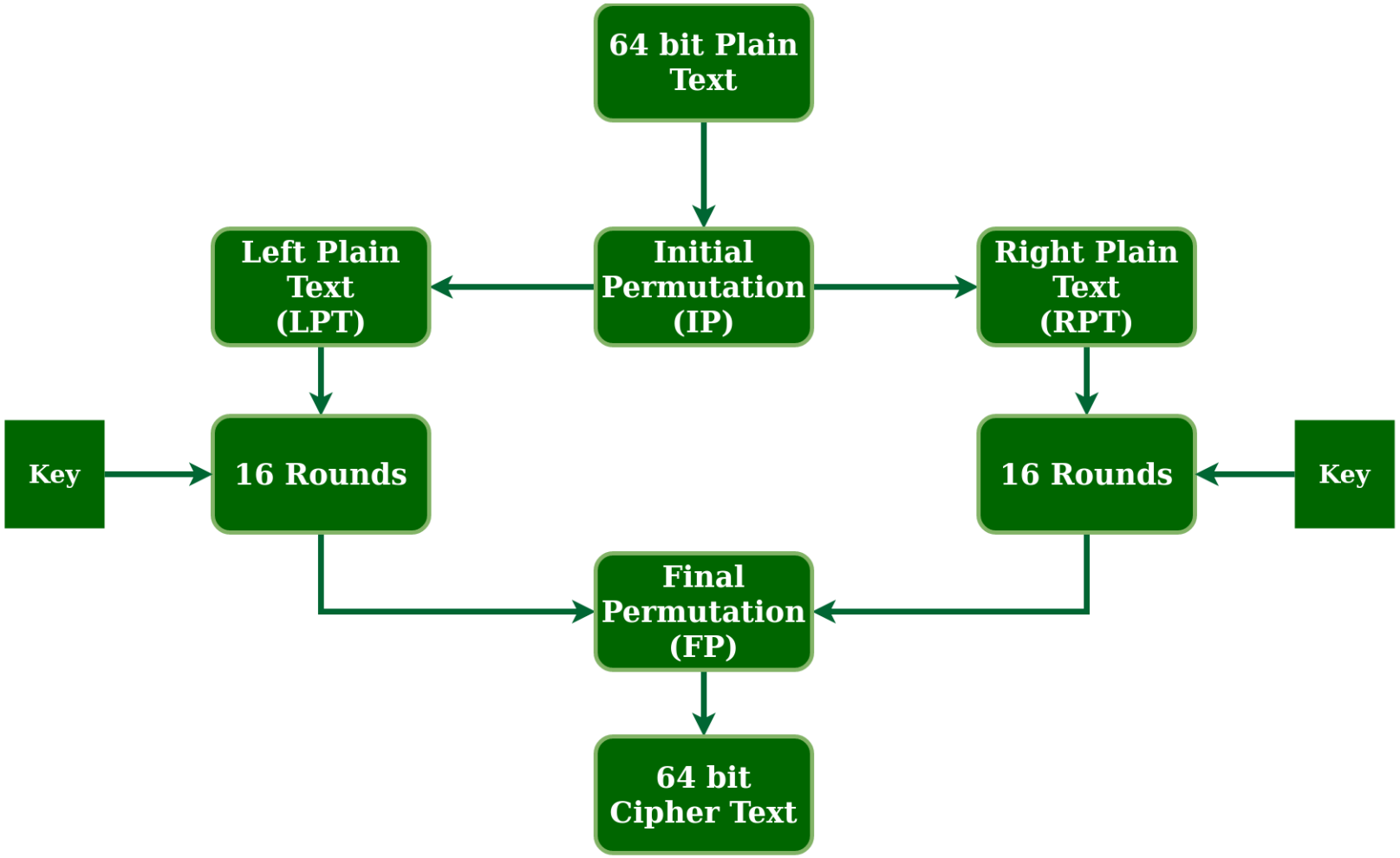


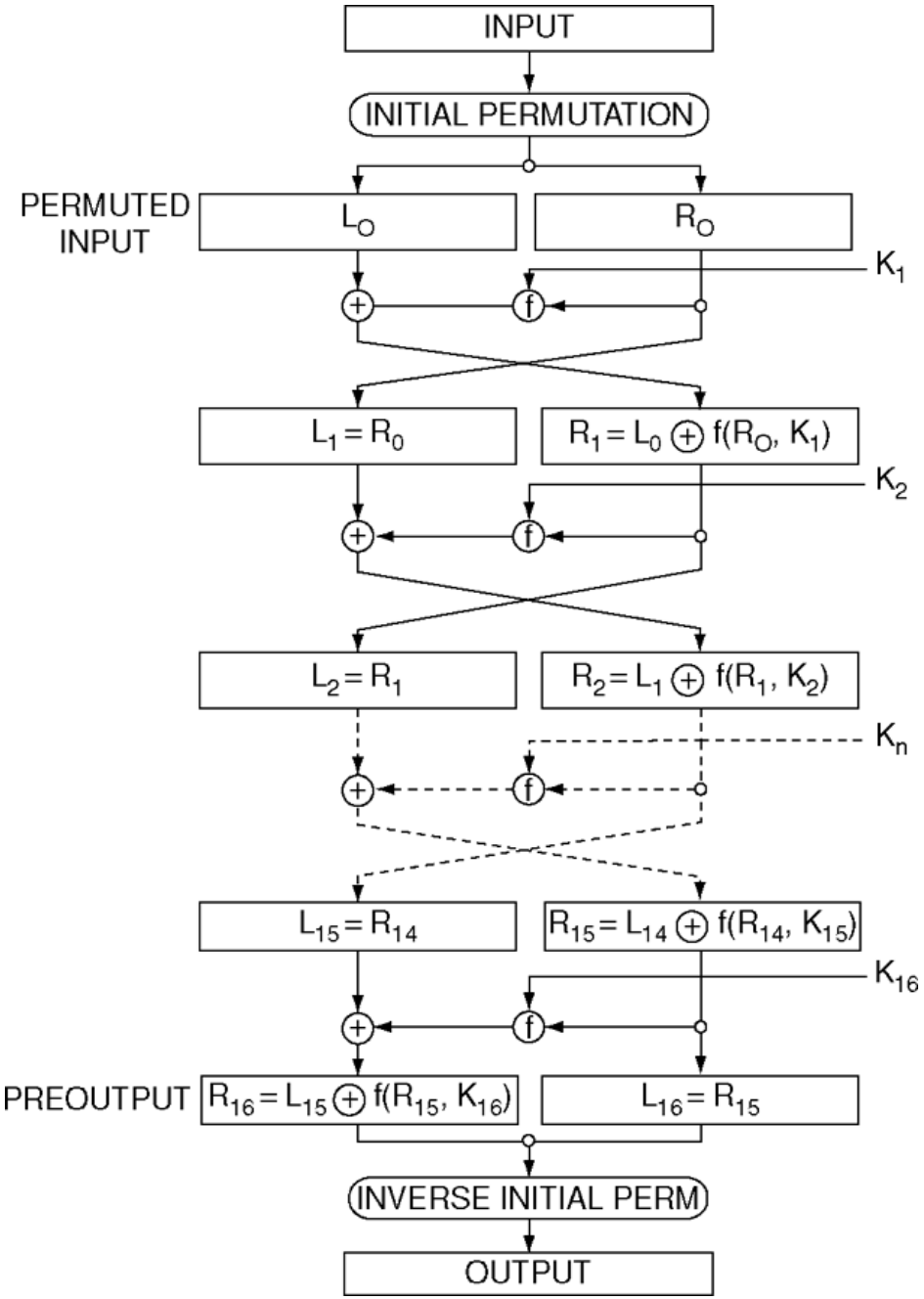
Symmetric Data Encryption

When Encryption and Decryption can be performed by using only same type of key



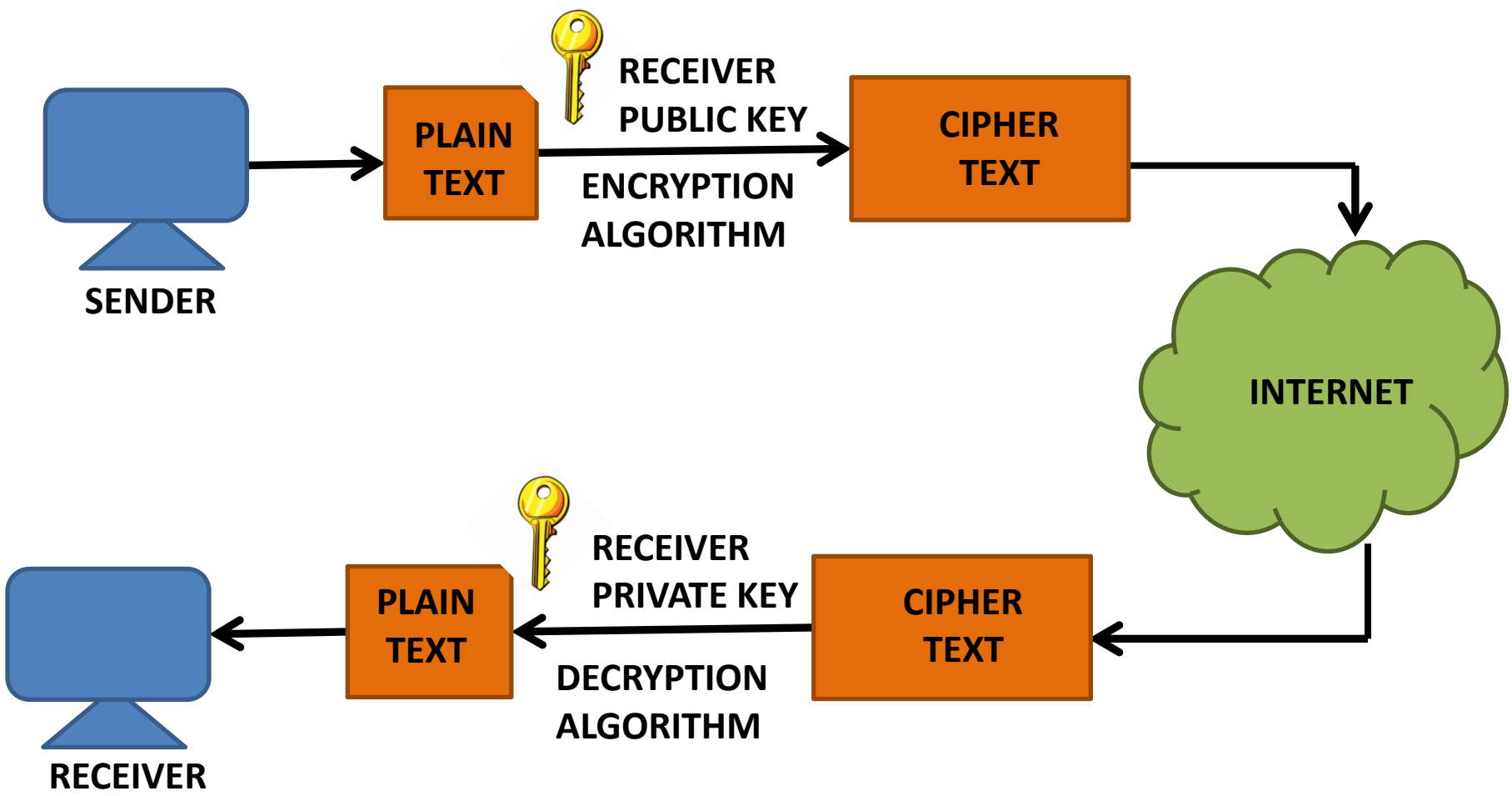
DES Data Encryption




INPUT
INITIAL PERMUTATION
PERMUTED INPUT
L_O
R_O
K_1
f
$L_1 = R_0$
$R_1 = L_0 \oplus f(R_O, K_1)$
K_2
$L_2 = R_1$
$R_2 = L_1 \oplus f(R_1, K_2)$
K_n
$L_{15} = R_{14}$
$R_{15} = L_{14} \oplus f(R_{14}, K_{15})$
K_{16}
PREOUTPUT
$R_{16} = L_{15} \oplus f(R_{15}, K_{16})$
$L_{16} = R_{15}$
INVERSE INITIAL PERM
OUTPUT

Asymmetric Data Encryption

When Encryption and Decryption can be performed by using two different type of key(Public or Private)



RSA Algorithm

1. Choose two large primes p and q (typically around 256 bits)

2. Compute: $n = p \times q$

3. Calculate totient function $\phi(n) = (p-1)(q-1)$

4. Choose a number 'd' relatively prime to 'z'
   $1 < e < \phi(n)$, e is co-prime to $\phi(n)$, gcd $(e, \phi(n)) = 1$

5. Find 'e' such that $e \times d \bmod (p-1)(q-1) = 1$

6. For encryption: $\boxed{C = P^e \pmod n}$

   For decryption: $\boxed{D = C^d \pmod n}$

Q. Let p=3 and q=11 and plain text message length M=31. Encrypt and Decrypt it using RSA and also calculate its Public Key 'e' and Private Key 'd'.

1. **Calculate n=p x q So, n= 3 x 11= 33**
2. **Calculate $\phi(n) = (p-1)x(q-1)$. So, $\phi(n)=2x10=20$**
3. **Calculate 'e', Let e=7 As gcd(7,20)=1 & 1<7<20.**
4. **Calculate 'd'. As dxe mod $\phi(n)=1$**

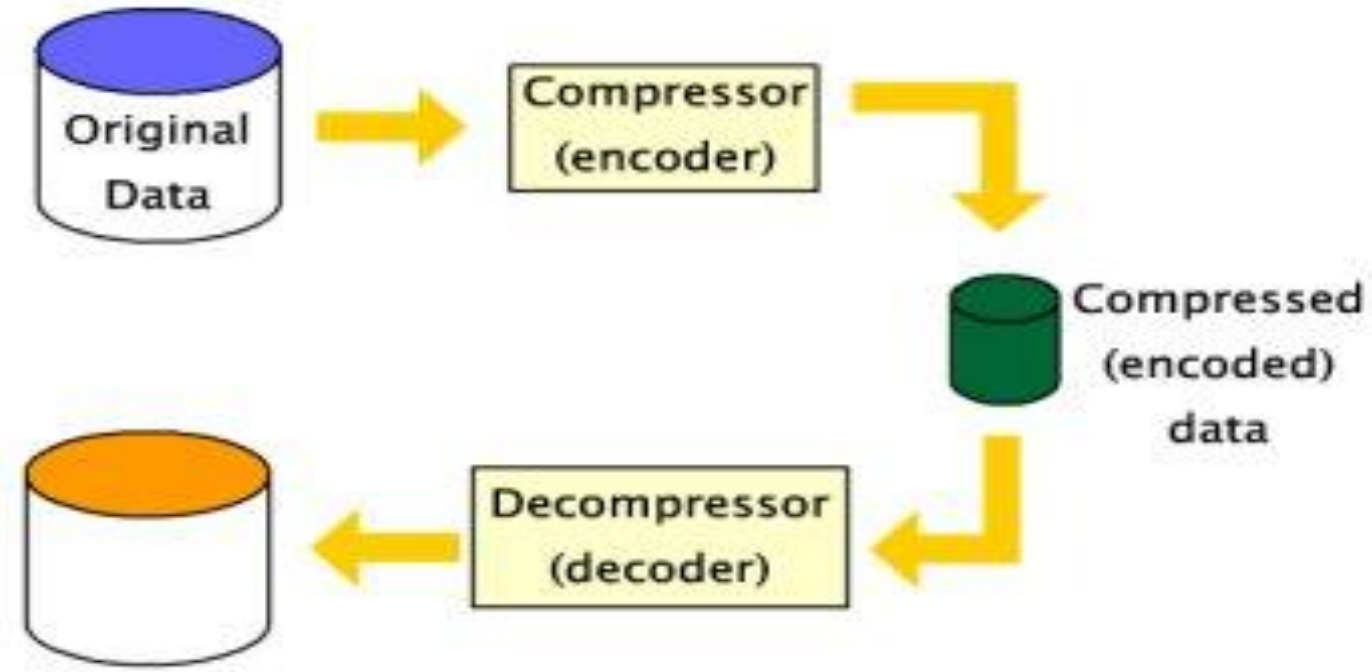
   d x 7 mod 20=1

   3 x 7 mod 20=1

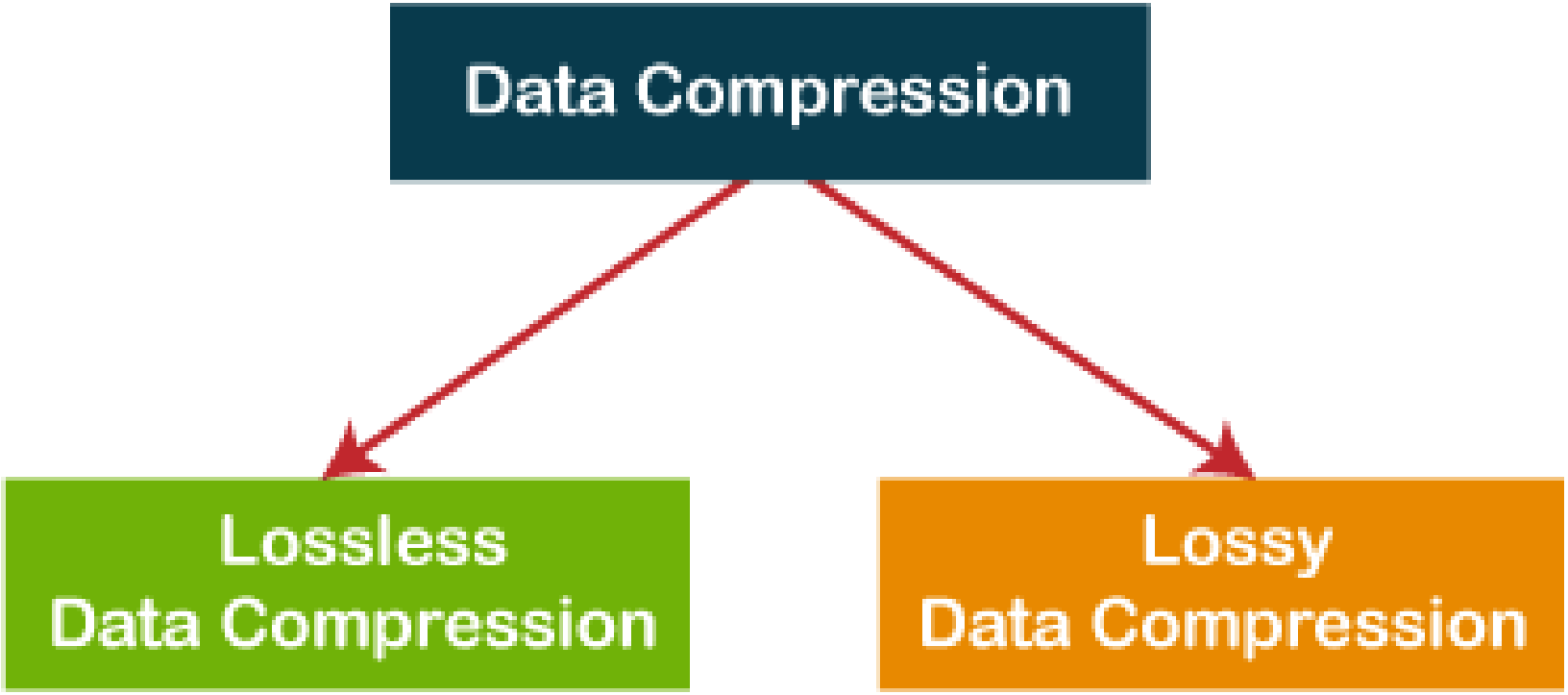
   So, d=3
5. **Encrypt Cipher C= M^emod n, C=31^7mod 33=4**
6. **Decrypt Plain M= C^dmod n, M=4^3mod 33=31**

Data Compression

- **Data compression means conversion of Large Data into Small reduced form (Reduction in the number of bits needed to represent data)**
- **Compressing data can save storage capacity, speed up file transfer, and decrease costs for storage hardware and network bandwidth.**



Data Compression Techniques



Lossless Data Compression

- **Lossless compression is a class of data compression that allows the original data to be perfectly reconstructed from the compressed data with no loss of information.**

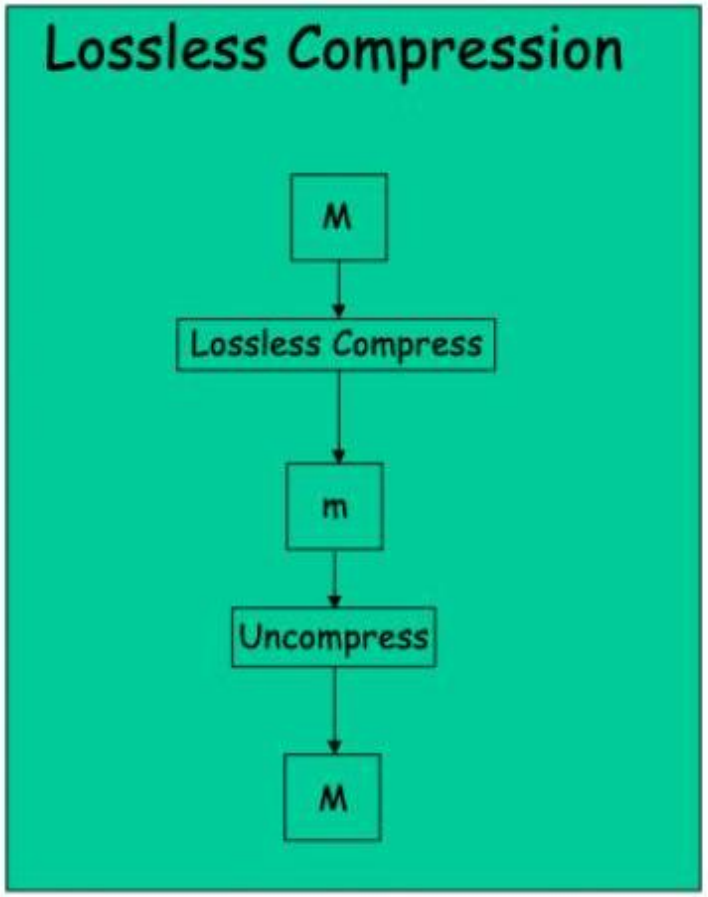
Let M= Original Data (Before Compression)

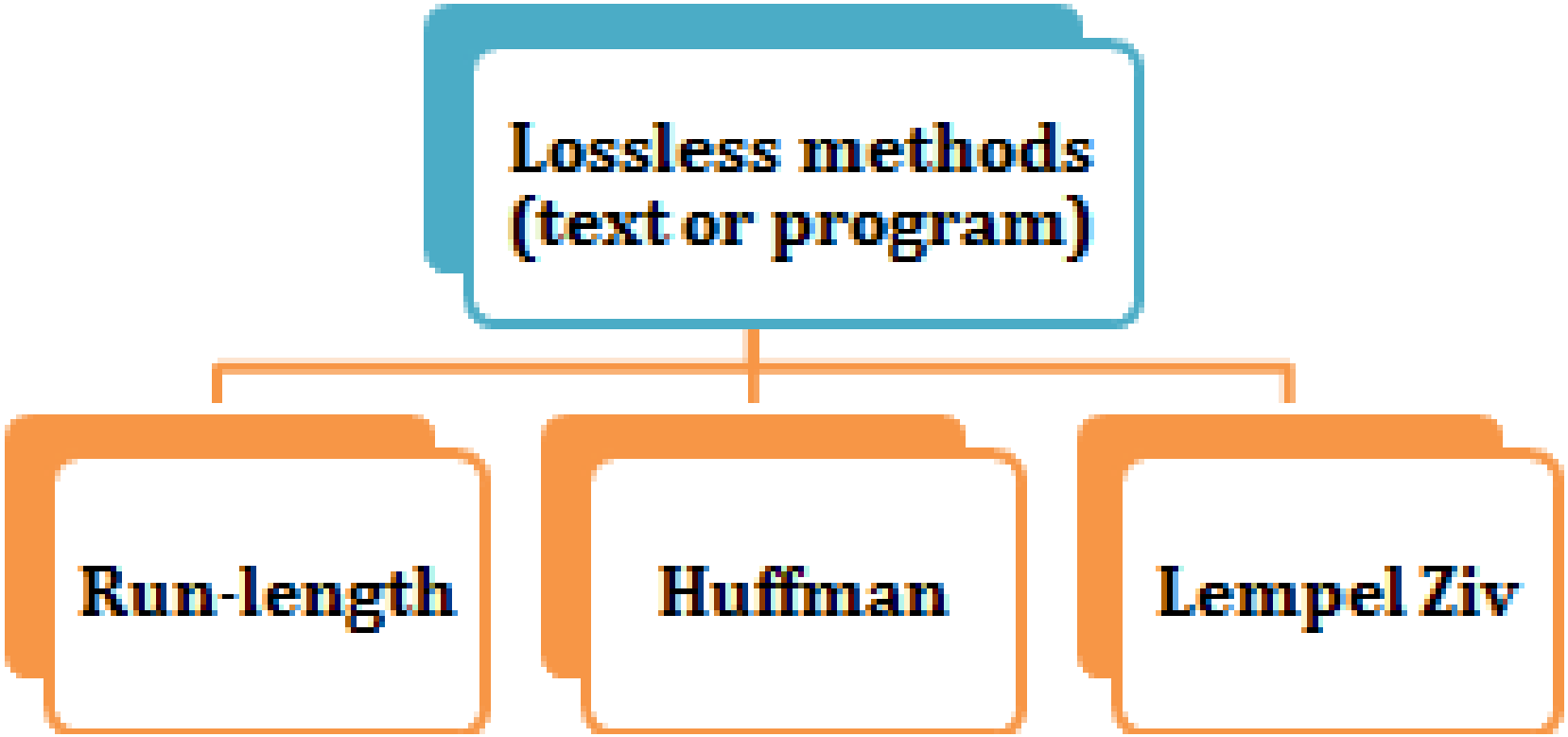
M'=Data Recovered (After Compression)

m=M-M'

where m=data lost

Here m=0




Lossless methods
(text or program)
Run-length
Huffman
Lempel Ziv

Lossy Data Compression

- **In such type of compression, some data is lost during recovery. Original data may not be recovered as it is.**

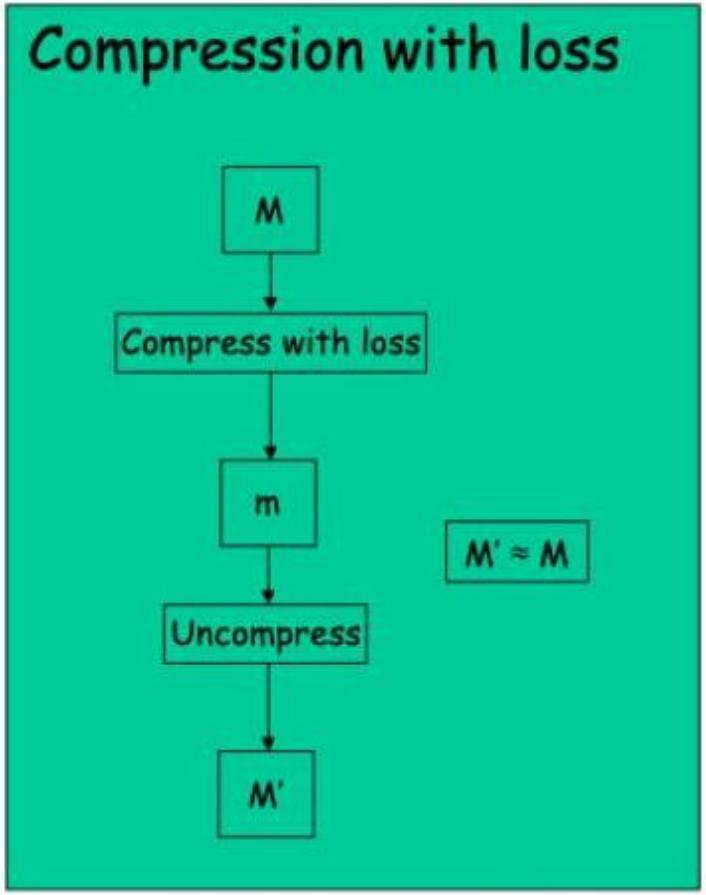
Let M= Original Data (Before Compression)

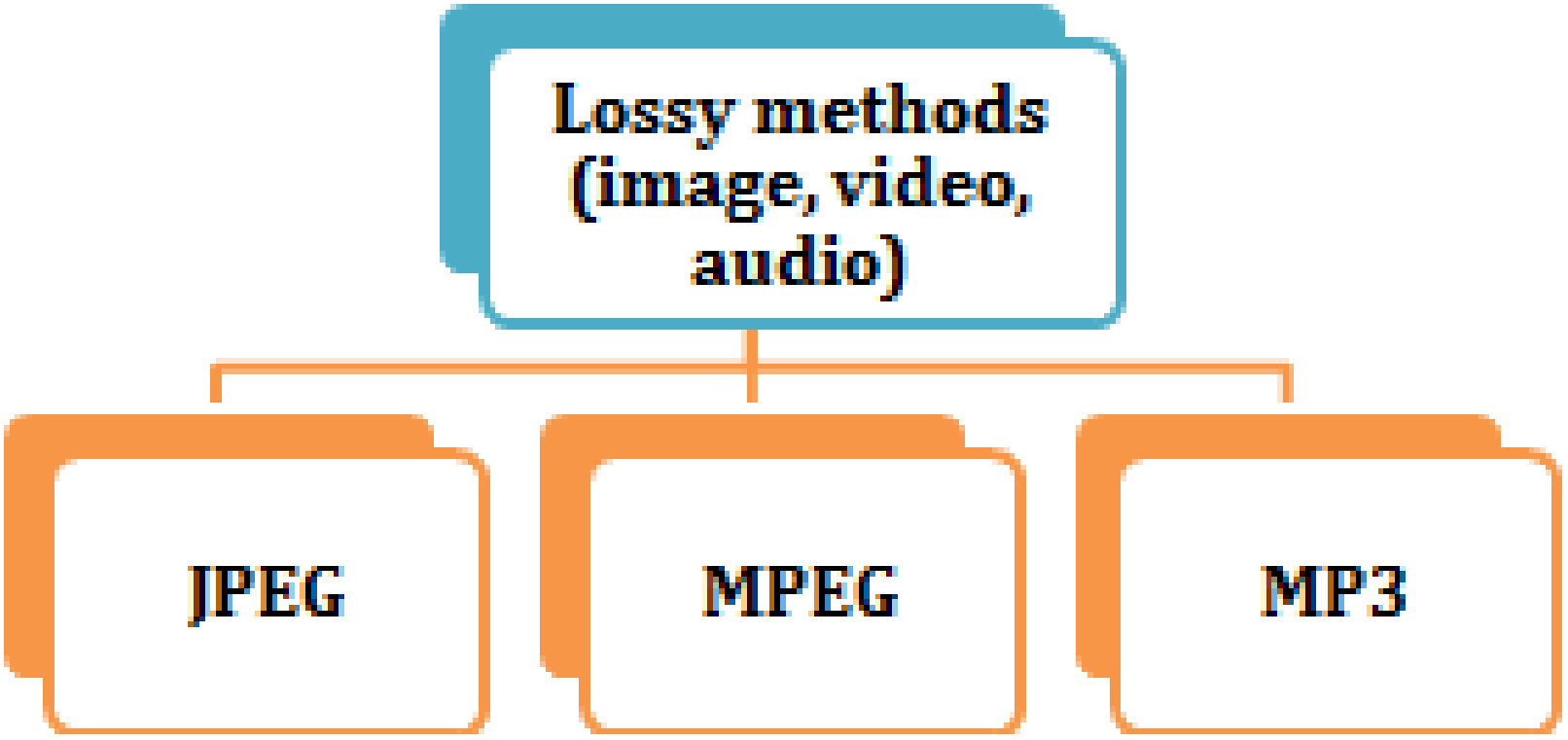
M'=Data Recovered (After Compression)

m=M-M'

where m=data lost

Here m>0




Lossy methods (image, video, audio)
JPEG
MPEG
MP3

Data Compression Terms

$$\text{Compression Ratio} = \frac{\text{Compressed Data}}{\text{Original Data}}$$

$$\text{Compression Factor} = \frac{1}{\text{Compression Ratio}}$$

Compression Time: Total Time Taken by Encoder to compress the data.

Decompression Time: Total Time Taken by Decoder to uncompress the data.

Fixed Length Encoding:-

Ascii

a → 97 → 7 bits (1100001) } ⟶ (3bit)

b
c
d
e
f
g

(7 x 7 bits) = 49 bits.

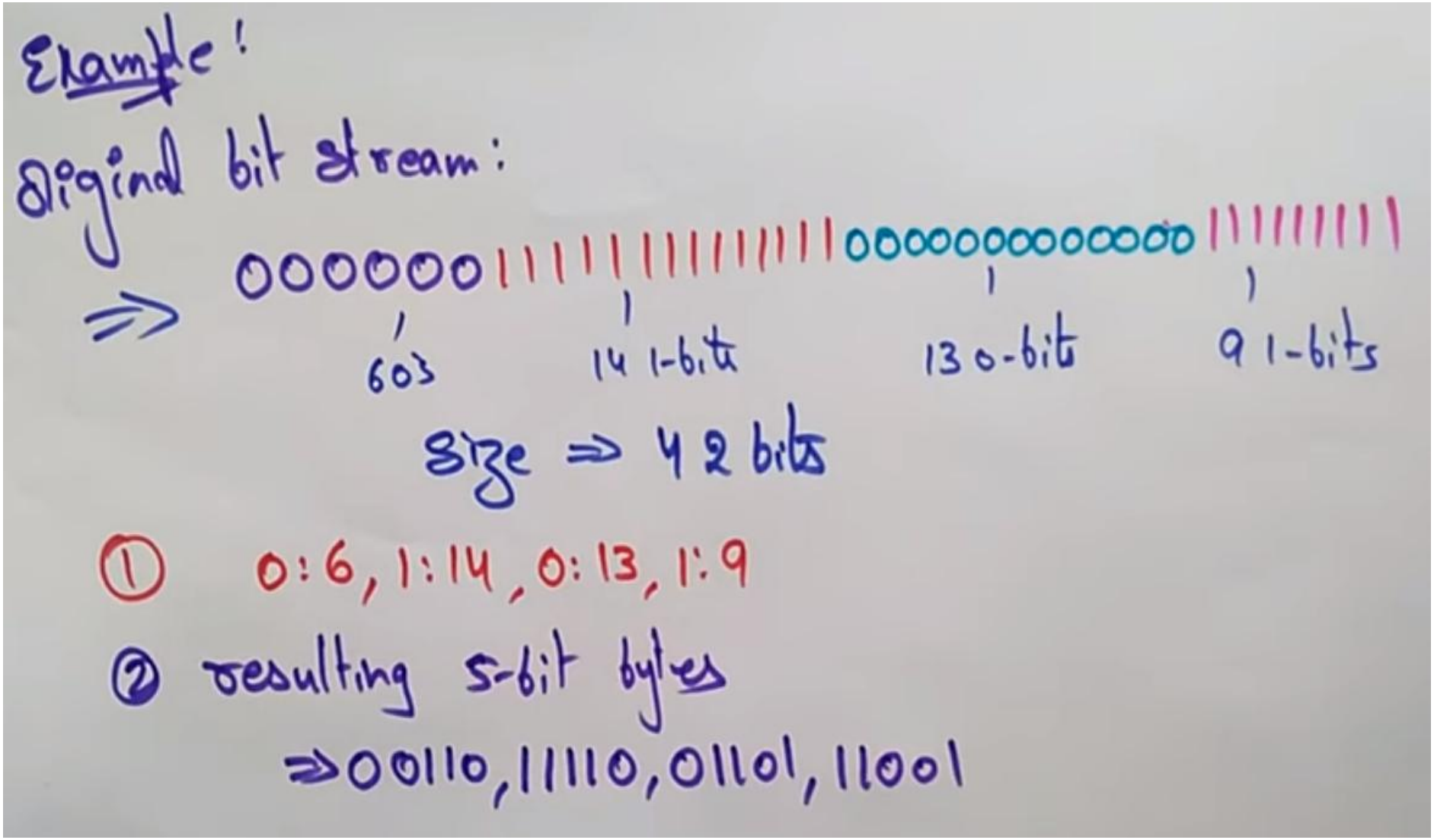
a → 000
b → 001
c → 010
d → 011
e → 100
f → 101
g → 110

(7 x 3 bits) = 21 bits.

RUN LENGTH ENCODING

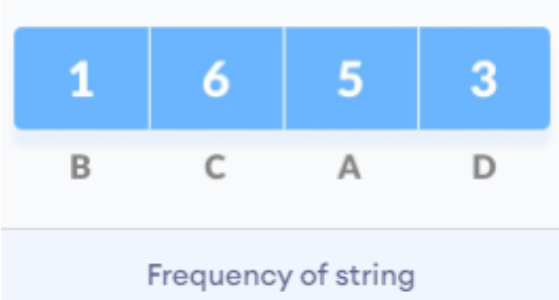
- Pick the first character from the source string.
- Append the picked character to the destination string.
- Count the number of subsequent occurrences of the picked character and append the count to the destination string.
- Pick the next character and repeat steps 2, 3 and 4 if the end of the string is NOT reached.

RUN LENGTH ENCODING

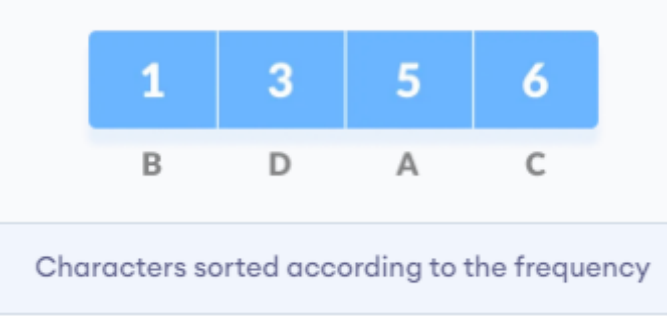


Huffman Encoding

- Calculate the frequency of each character in the string.

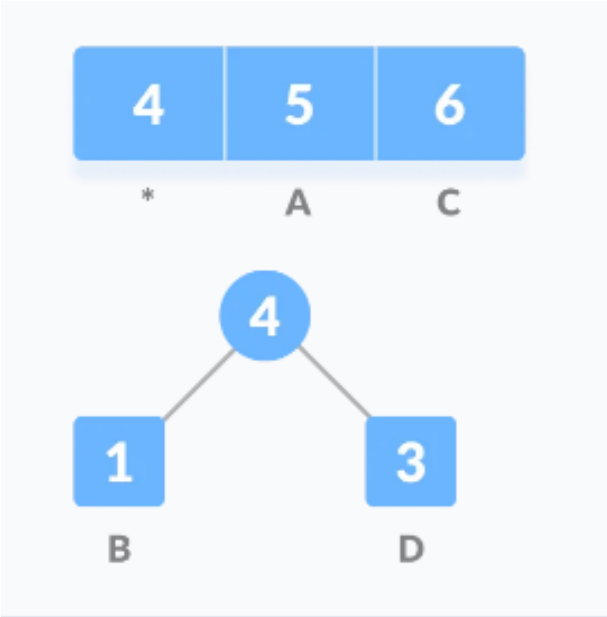


- Sort the characters in increasing order of the frequency. These are stored in a priority queue Q.



Huffman Encoding

- Make each unique character as a leaf node.
- Create an empty node z. Assign the minimum frequency to the left child of z and assign the second minimum frequency to the right child of z. Set the value of the z as the sum of the above two minimum frequencies. Getting the sum of the least numbers

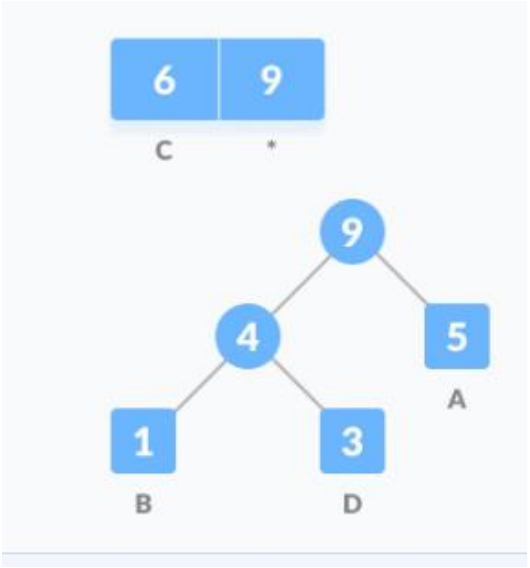


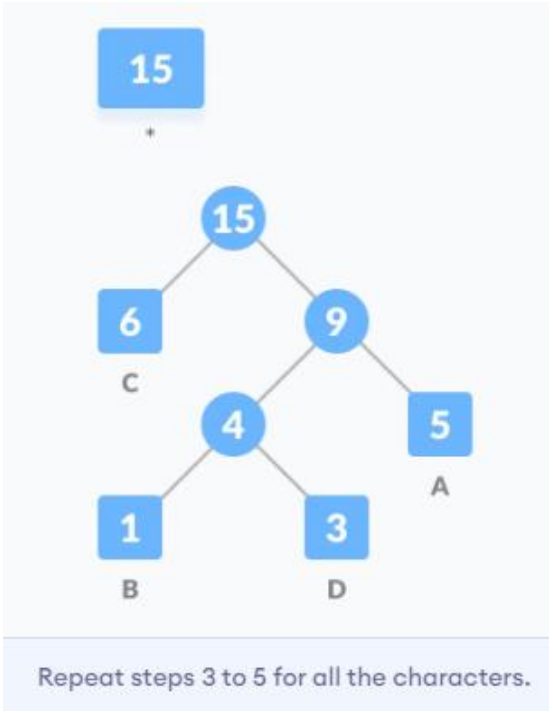
Getting the sum of the least numbers

Huffman Encoding

- Remove these two minimum frequencies from Q and add the sum into the list of frequencies (* denote the internal nodes in the figure above)
- Insert node z

  into the tree.

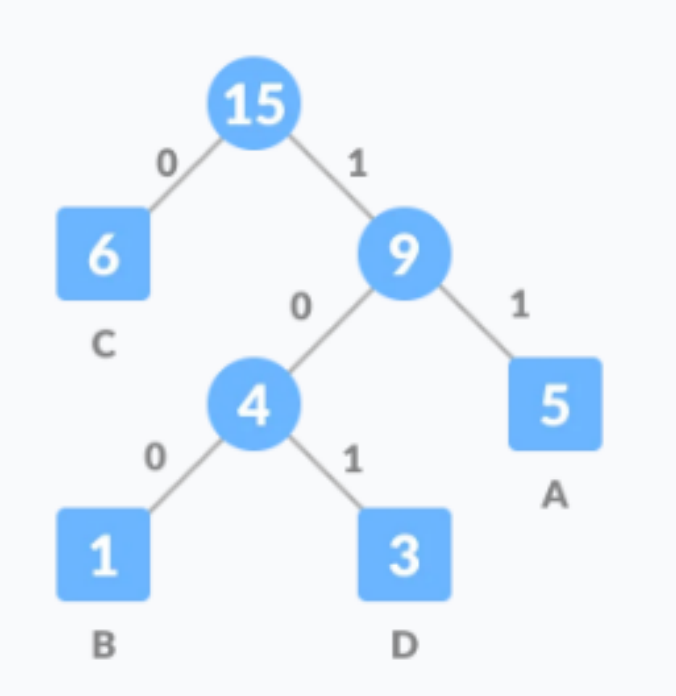




Huffman Encoding

- For each non-leaf node, assign 0 to the left edge and 1 to the right edge
- Assign 0 to the left edge

  and 1 to the right edge



Character	Frequency	Code	Size
A	5	11	5*2 = 10
B	1	100	1*3 = 3
C	6	0	6*1 = 6
D	3	101	3*3 = 9
4 * 8 = 32 bits	15 bits		28 bits

Without encoding, the total size of the string was 120 bits. After encoding the size is reduced to `32 + 15 + 28 = 75`.

Huffman Coding

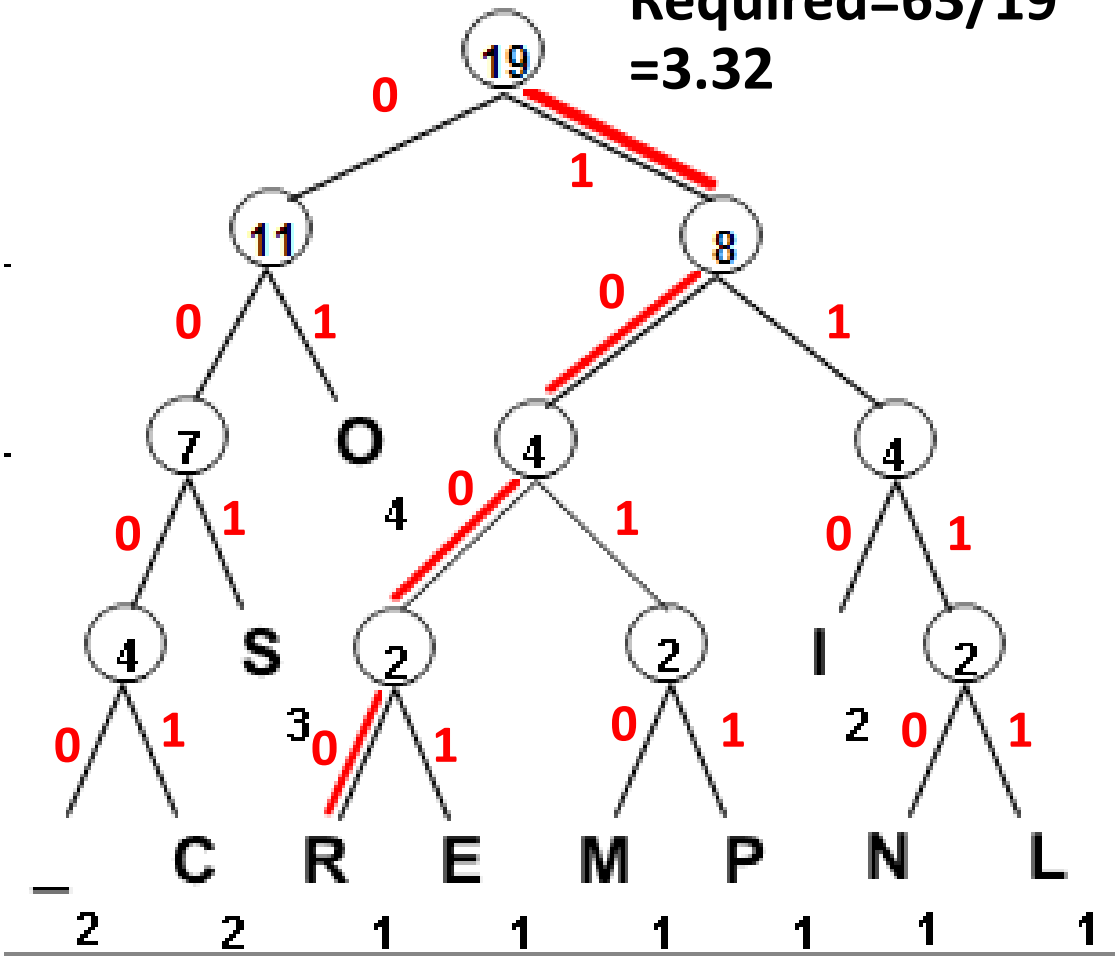
Example: "COMPRESSION_IS_COOL"

Here,
M=1,P=1,R=1, N=1, L=1
E=1,C=2, _=2, I=2, S=3,O=4,
Total Characters=19
19x7=133bits

Here,

C=0001=4	4x2=8
O=01=2	2x4=8
M=1010=4	4x1=4
P=1011=4	4x1=4
R=1000=4	4x1=4
E=1001=4	4x1=4
S=001=3	3x3=9
I=110=3	3x2=6
N=1110=4	4x1=4
_=0000=4	4x2=8
L=1111=4	4x1=4
	_____
Total Bits Reuired	63 bits

Average bits Required=63/19 =3.32



Lampel ZIV(LZ) Algorithm

Let we have data

101011011010101010

1,0,10,11,01,101,010,1010

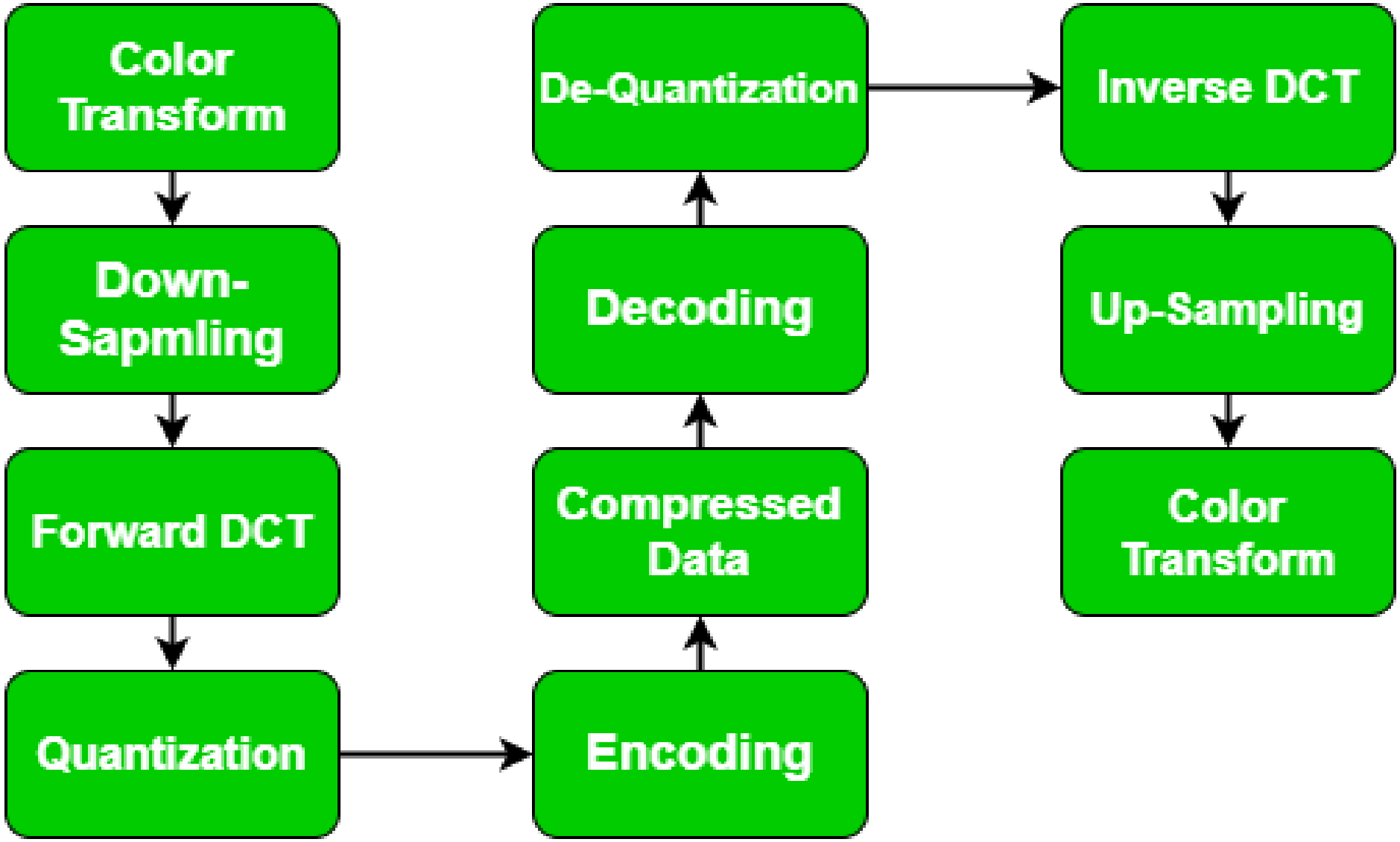
Dictionary Location	Content	Code word
001	1	0001
010	0	0000
011	10	0010
100	11	0011
101	01	0101
110	101	0111
111	010	1010
	1010	1100

Required Digital Code:

000100000010001101010111101011100

Image Compression (JPEG)

- JPEG, GIF and MPEG are more than compression algorithms, they also define the format for image or video data in the same way as XDR, NDR and ANS.1 define the format for numeric and string data.
- JPEG -Named after the Joint Photographic Experts Group that designed it.
- Used for digital imagery (ISO standard)
- Compression takes place in 3 phases: a discrete cosine transformation (DCT), quantization and encoding


Color Transform
Down-Sapmling
Forward DCT
Quantization
Encoding
Compressed Data
Decoding
De-Quantization
Inverse DCT
Up-Sampling
Color Transform

Algorithm of JPEG Data Compression :

1. **Splitting –**
   We split our image into the blocks of 8*8 blocks. It forms 64 blocks in which each block is referred to as 1 pixel.

2. **Color Space Transform –**
   In this phase, we convert R, G, B to Y, Cb, Cr model. Here Y is for brightness, Cb is color blueness and Cr stands for Color redness. We transform it into chromium colors as these are less sensitive to human eyes thus can be removed.

3. **Apply DCT –**
   We apply Direct cosine transform on each block. The discrete cosine transform (DCT) represents an image as a sum of sinusoids of varying magnitudes and frequencies.

4. **Quantization –**

   In the Quantization process, we quantize our data using the quantization table.

5. **Serialization –**

   In serialization, we perform the zig-zag scanning pattern to exploit redundancy.

6. **Vectoring –**

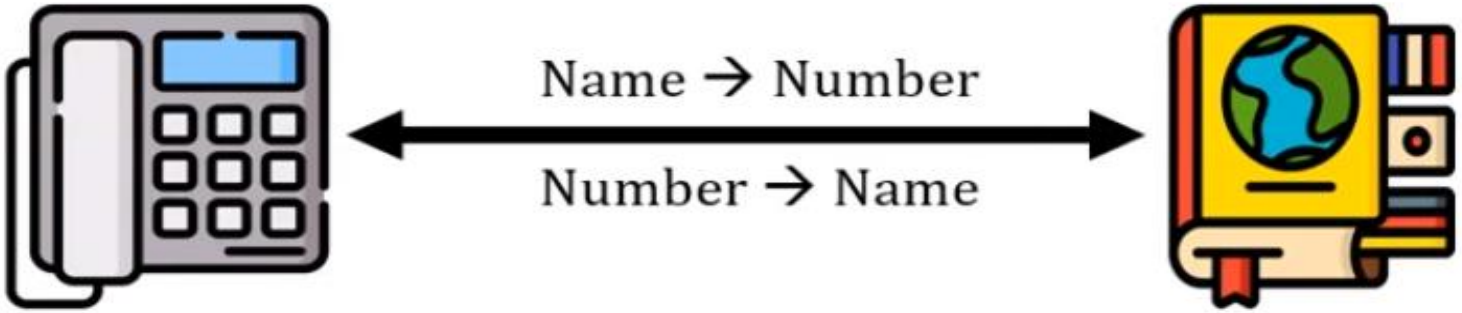
   We apply DPCM (differential pulse code modeling) on DC elements. DC elements are used to define the strength of colors.

7. **Encoding –**

   In the last stage, we apply to encode either run-length encoding or Huffman encoding. The main aim is to convert the image into text and by applying any encoding we convert it into binary form (0, 1) to compress the data.

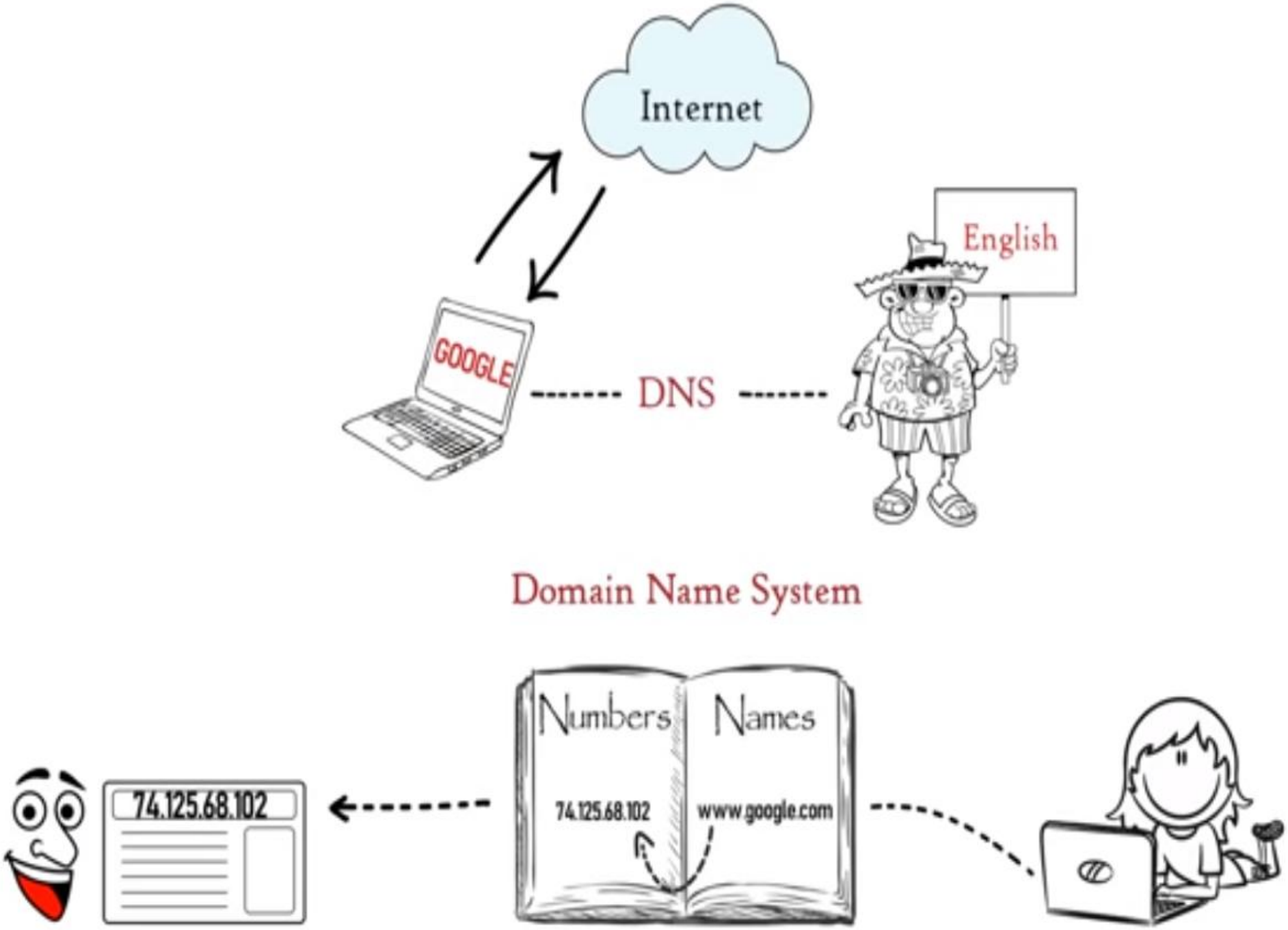
DNS Introduction

- **Domain** or **Domain name** is the location of a website.
- **DNS** is directory(phonebook) of internet.



- Connecting web browsers with web servers.

- IP Address → Naming || Naming → IP Address


Internet
GOOGLE
DNS
English
Domain Name System
Numbers
Names
74.125.68.102
www.google.com
74.125.68.102

Types of DNS Servers:

DNS recursive resolver/DNS resolver

Root name server

Top Level Domain/TLD name server

Authoritative name server

Root name server

13 sets ⇦

letter.root-servers.net

www.root-servers.org

operated by: 12 organizations

info page: letter.root-servers.org

letter: 'a' to 'm'

for 'g': https://disa.mil/g-root

TLD name server

.com TLD name server

.net TLD name server

domains: .com, .net, .in, .edu

websites: .com extension

websites: .net extension

Authoritative name server

Last server in DNS

stores the website's IP address

DNS Resolver

ISP

CACHE

www.google.com	172.217.167.46
www.amazon.com	176.32.98.166
www.twitter.com	104.244.42.193
www.youtube.com	172.217.166.238

www.facebook.com ‹IP address›

www.facebook.com

.com TLD name server

IP address

www.facebook.com

IP address

Root name server

13 sets ⇦

letter.root-servers.net

www.root-servers.org

TLD name server

.com TLD name server

.net TLD name server

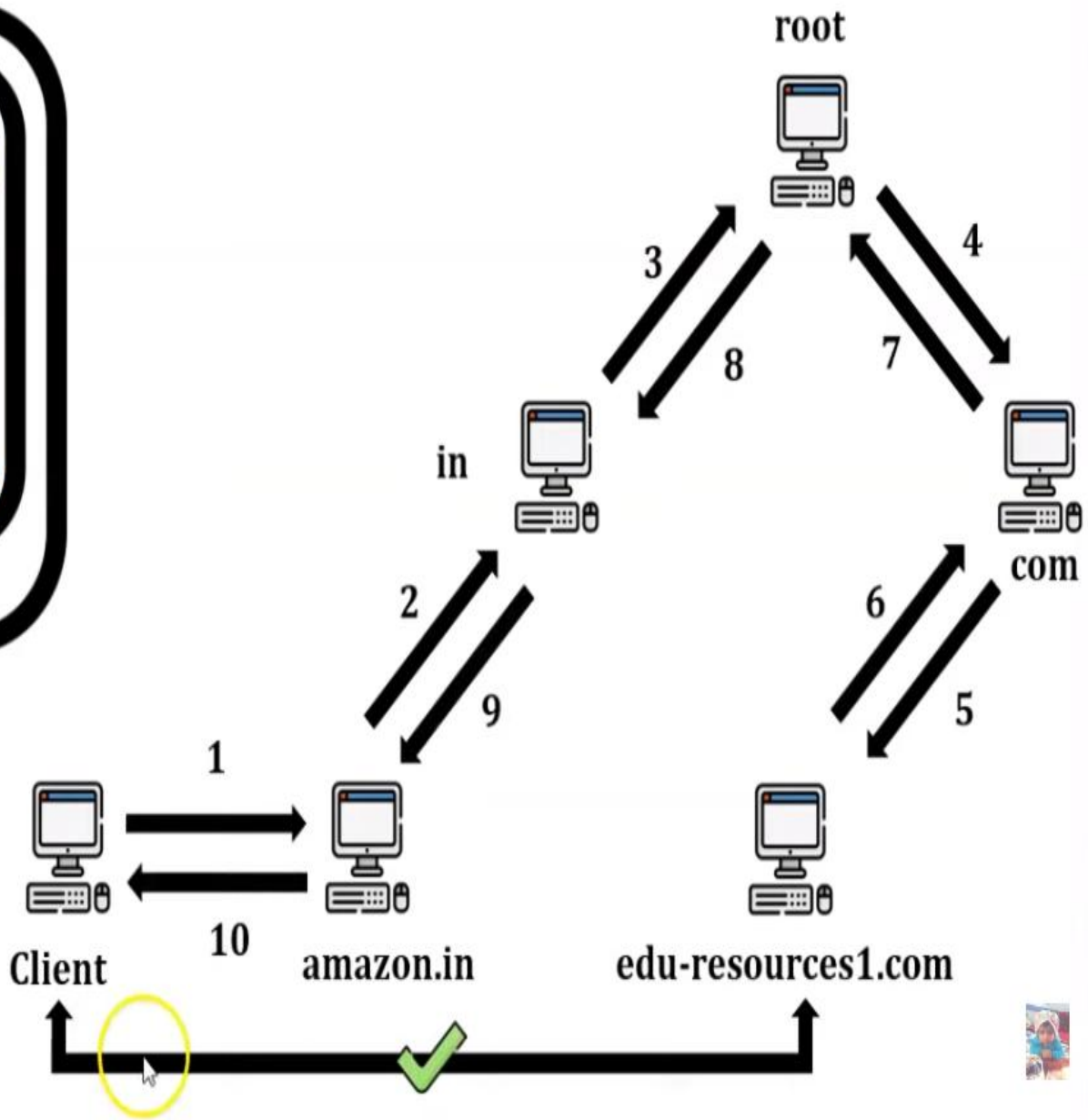
Authoritative name server

Last server in DNS

stores the website's IP address

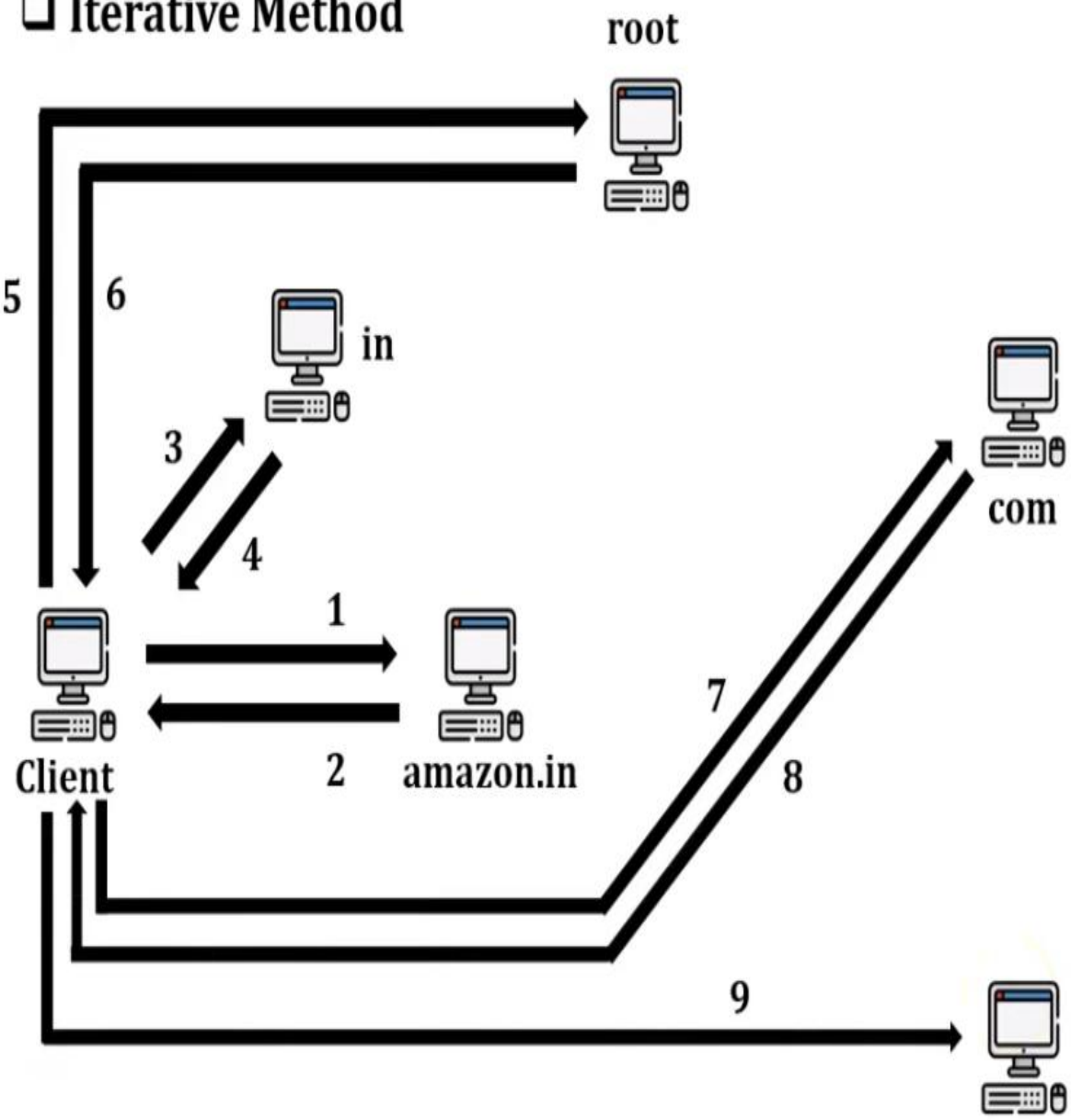
Name Resolver

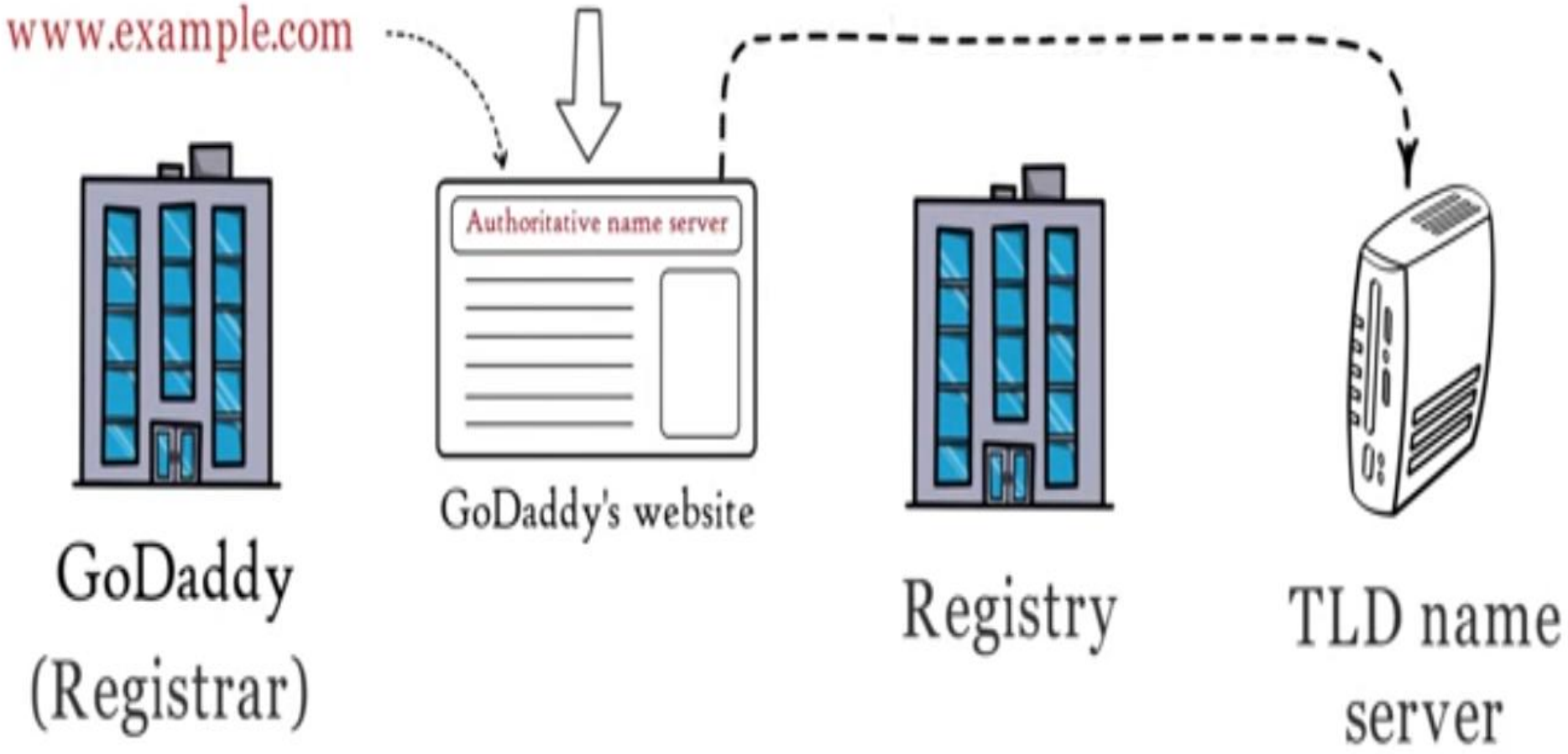
❑ Recursive Method



Name Resolver

❑ Iterative Method




www.example.com
Authoritative name server
GoDaddy's website
GoDaddy
(Registrar)
Registry
TLD name
server

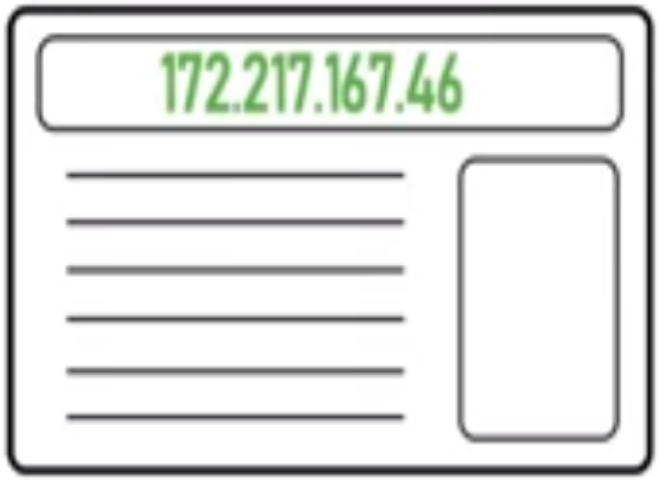
Is DNS necessary?

www.google.com

DNS

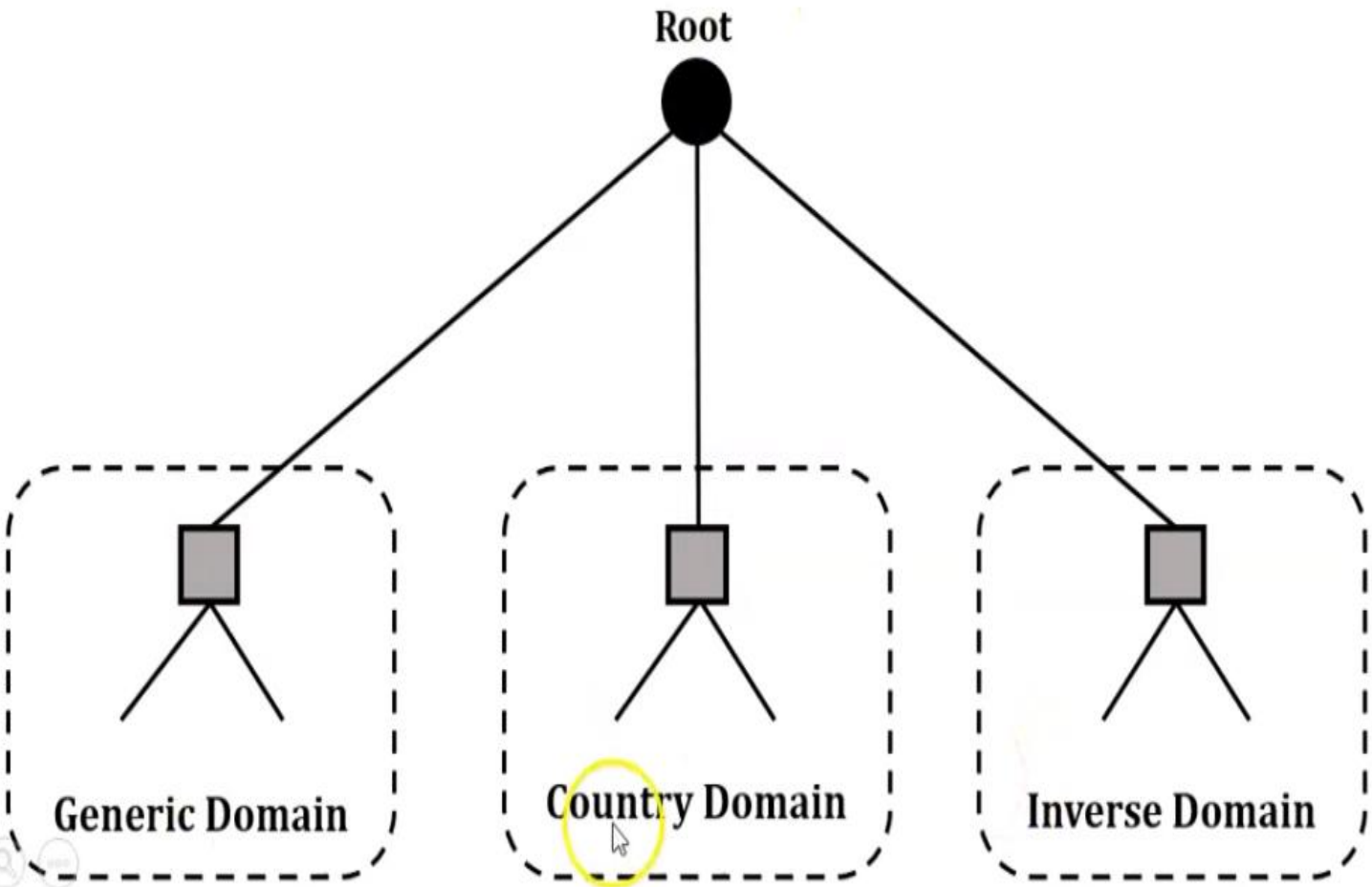
172.217.167.46

Web Browser



Domain Name

- Domain name is divided into main three different categories in the internet.
- i.e., Generic domain, Country domain and Inverse domain.
- Each node in tree defines a domain, which is an index to the domain name space database.



Domain Name

❑ Generic Domain

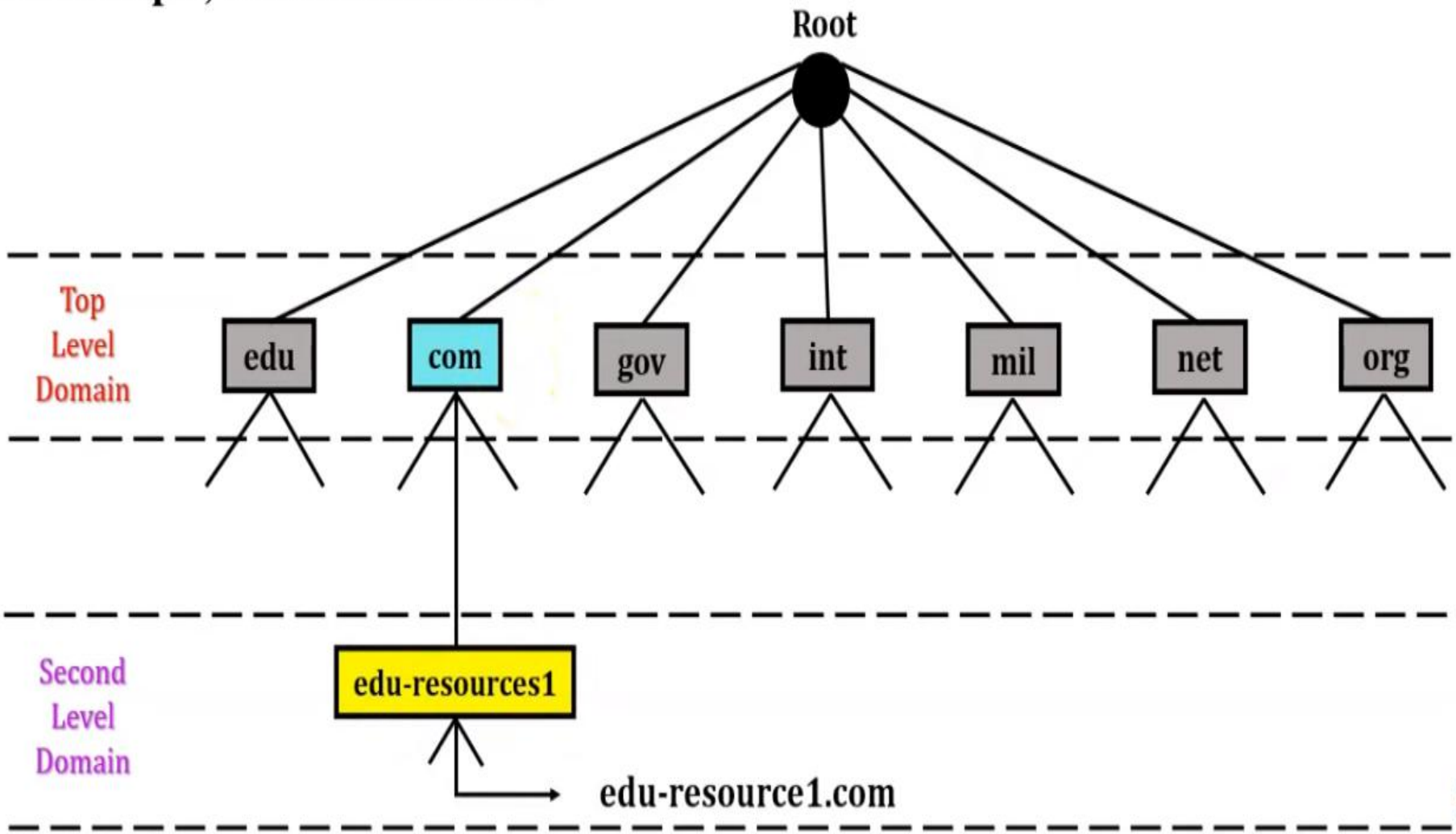
- The 3 character domains are called the *generic domains*.
- Generic domain labels are as follows

Sr. No.	Label	Description
1.	.com	Commercial Organization
2.	.edu	Educational Organization
3.	.gov	Government Organization
4.	.int	International Organization
5.	.mil	Military Group
6.	.net	Network Support Centers
7.	.org	Nonprofit Organization

Domain Name

- Generic Domain

- **For example,** edu-resources1.com



Domain Name

❑ Country Domain

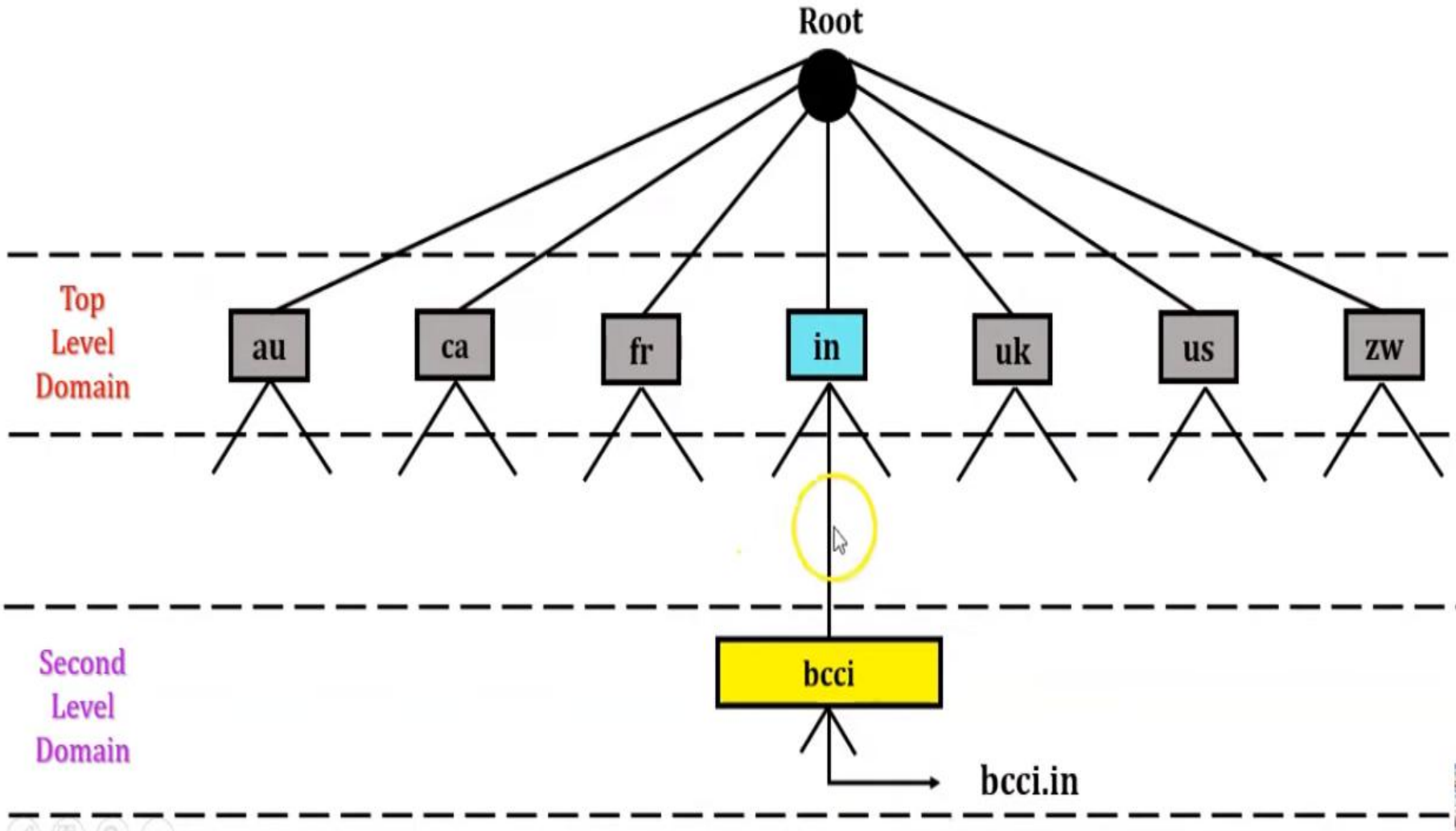
- The 2 character domains are called the *country domains*.

Sr. No.	Label	Country Name
1.	.at	Austria
2.	.au	Australia
3.	.bb	Barbados
4.	.be	Belgium
5.	.br	Brazil
6.	.ca	Canada
7.	.ch	Switzerland
8.	.cn	China
9.	.de	Germany
10.	.es	Spain
11.	.fr	France
12.	.hk	Hong Kong

Sr. No.	Label	Country Name
13.	.in	India
14.	.il	Israel
15.	.it	Italy
16.	.jp	Japan
17.	.kr	South Korea
18.	.nz	New Zealand
19.	.pt	Portugal
20.	.qa	Qatar
21.	.ru	Russia
22.	.us	United States
23.	.uk	United Kingdom
24.	.zw	Zimbabwe

Domain Name

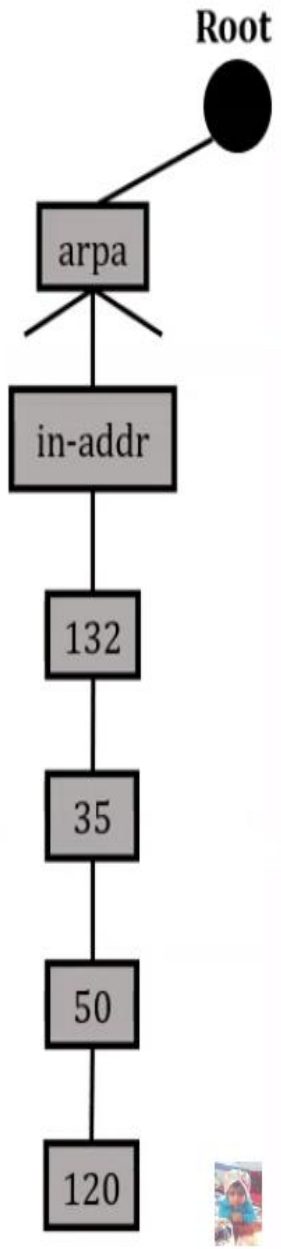
- **Country Domain**
  - **For example,** bcci.in



Domain Name

❑ Inverse Domain

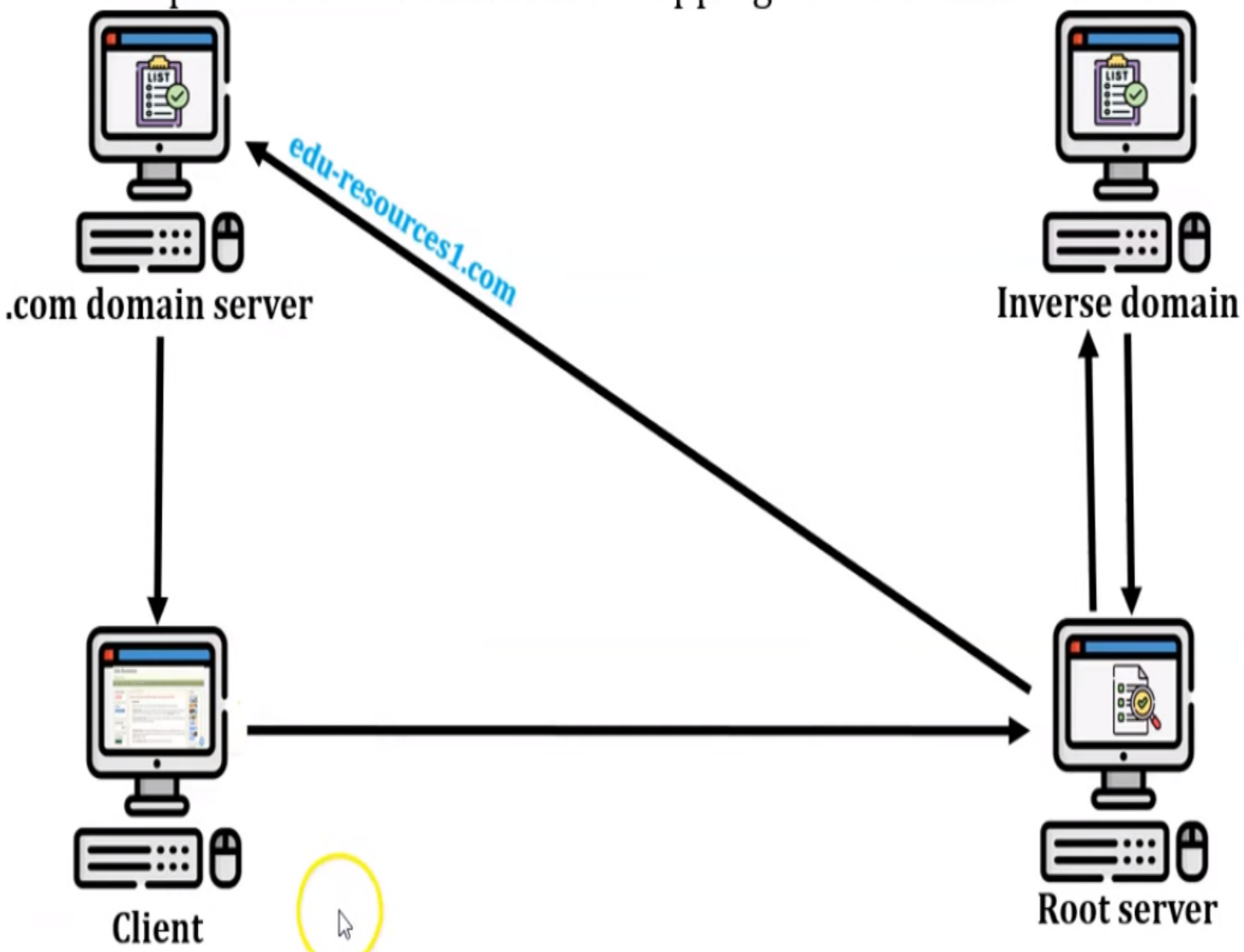
- Purpose of invers domain is mapping an address to a name.
- Example: When a client send a request to the server for doing particular task, server finds the list of authorized client.
- The list contains only IP address of the client.
- Server send a query to the inverse DNS server and ask for a mapping address to name for authorized client list.
- The above query is called an inverse or pointer query.
- The pointer query is handled by the first level node called arpa. The second level is also one single node named in-addr. The rest of the domain defines IP addresses.

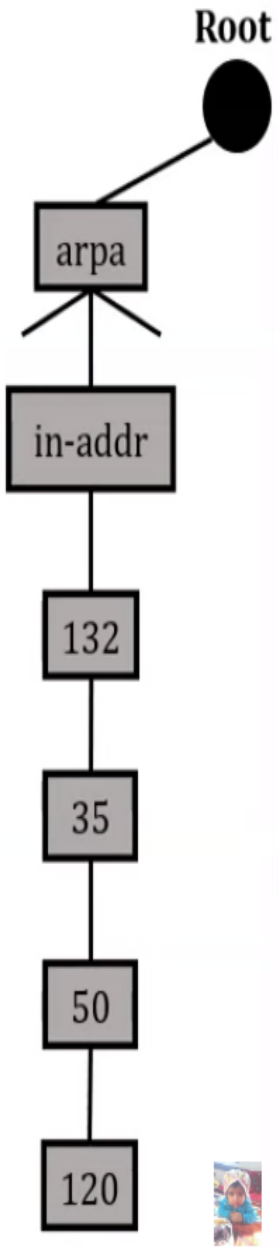


Domain Name

❑ Inverse Domain

- Purpose of inverse domain is mapping an address to a name.





132.35.50.120 ⇒ 120.50.35.132 - in-addr - arpa

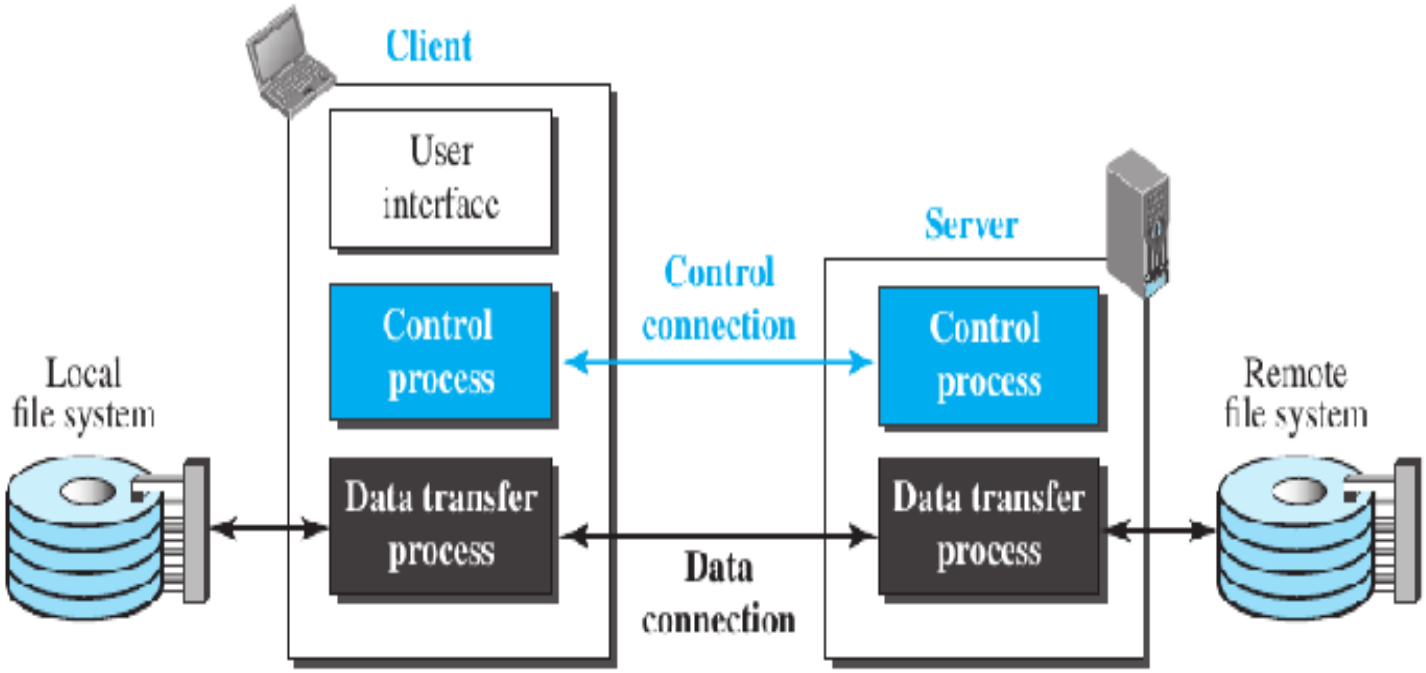
File Transfer

Transfer of data from Sender to Receiver in OSI Model is possible using FTP Protocol

FILE TRANSFER PROTOCOL (FTP)

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

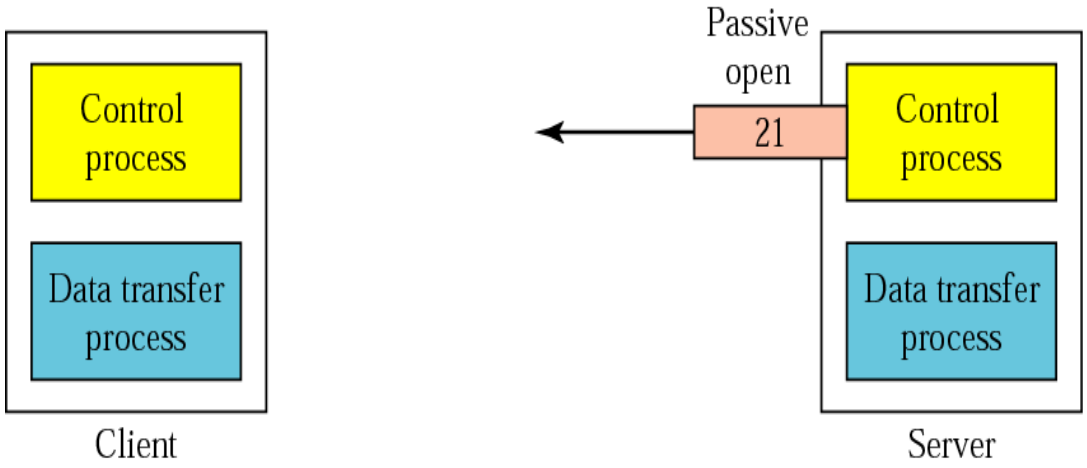
FTP


Client
User interface
Control process
Data transfer process
Server
Control process
Data transfer process
Control connection
Data connection
Local file system
Remote file system

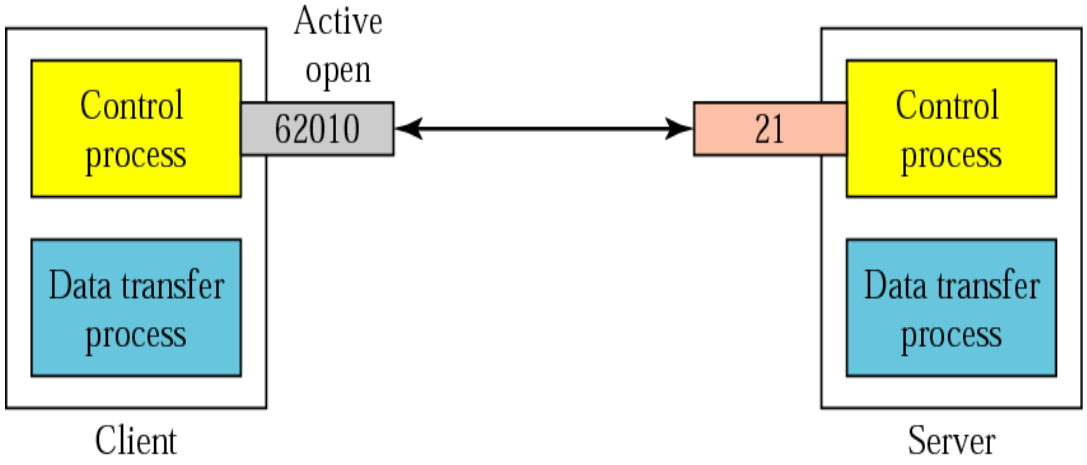
FTP uses the services of TCP. It needs two TCP connections.

The well-known port 21 is used for the control connection and the well-known port 20 for the data connection.

Opening the control connection

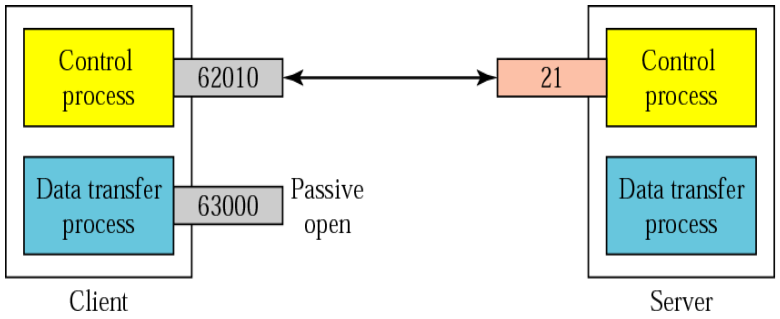


a. Passive open by server

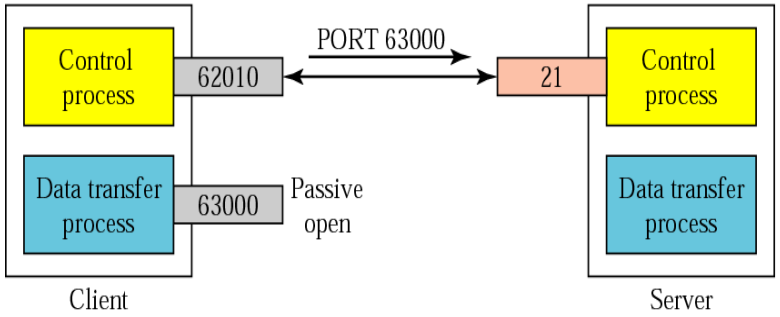


b. Active open by client

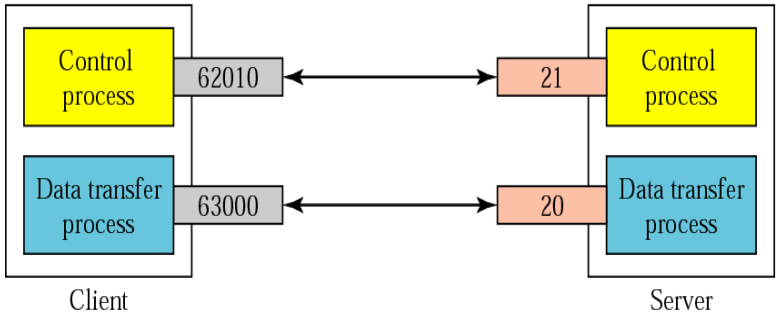
Creating the data connection



a. Passive open by client

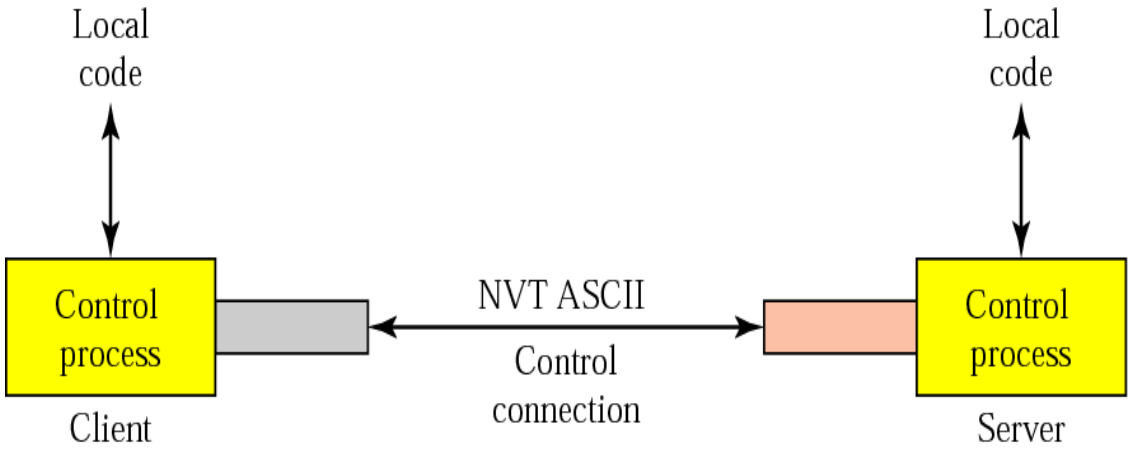


b. Sending ephemeral port number to server

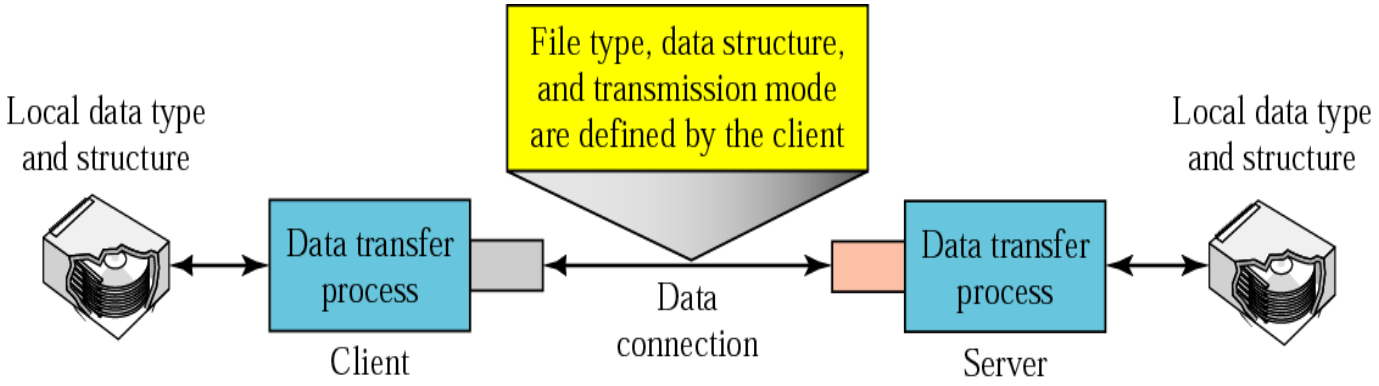


c. Active open by server

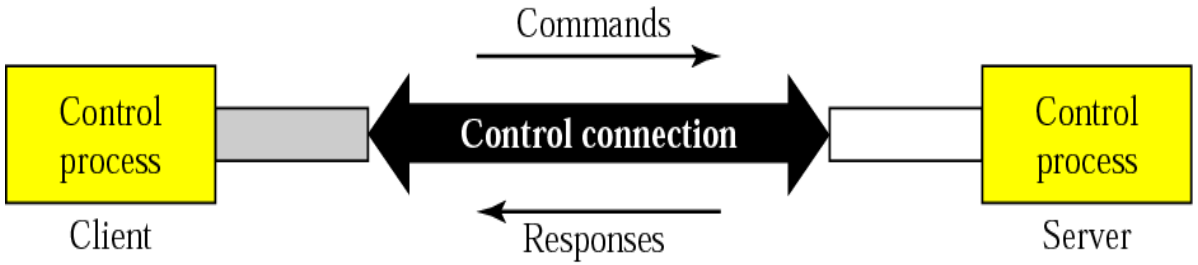
Using the control connection



Using the data connection



Command processing



Access commands

Command	*Argument(s)*	*Description*
USER	User id	User information
PASS	User password	Password
ACCT	Account to be charged	Account information
REIN		Reinitialize
QUIT		Log out of the system
ABOR		Abort the previous command

File management commands

Command	*Argument(s)*	*Description*
CWD	Directory name	Change to another directory
CDUP		Change to the parent directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
NLIST	Directory name	List the names of subdirectories or files without other attributes
MKD	Directory name	Create a new directory
PWD		Display name of current directory
RMD	Directory name	Delete a directory
RNFR	File name (old file name)	Identify a file to be renamed
RNTO	File name (new file name)	Rename the file
SMNT	File system name	Mount a file system

Data formatting commands

Command	*Argument(s)*	*Description*
TYPE	A (ASCII), E (EBCDIC), I (Image), N (Nonprint), or T (TELNET)	Define the file type and if necessary the print format
STRU	F (File), R (Record), or P (Page)	Define the organization of the data
MODE	S (Stream), B (Block), or C (Compressed)	Define the transmission mode

Port defining commands

Command	*Argument(s)*	*Description*
PORT	6-digit identifier	Client chooses a port
PASV		Server chooses a port

File transfer commands

Command	*Argument(s)*	*Description*
RETR	File name(s)	Retrieve files; file(s) are transferred from server to the client
STOR	File name(s)	Store files; file(s) are transferred from the client to the server
APPE	File name(s)	Similar to STOR except if the file exists, data must be appended to it
STOU	File name(s)	Same as STOR except that the file name will be unique in the directory; however, the existing file should not be overwritten

File transfer commands (continued)

Command	*Argument(s)*	*Description*
ALLO	File name(s)	Allocate storage space for the files at the server
REST	File name(s)	Position the file marker at a specified data point
STAT	File name(s)	Return the status of files

Miscellaneous commands

Command	*Argument(s)*	*Description*
HELP		Ask information about the server
NOOP		Check if server is alive
SITE	Commands	Specify the site-specific commands
SYST		Ask about operating system used by the server

Responses

Code	*Description*
Positive Preliminary Reply	
120	Service will be ready shortly
125	Data connection open; data transfer will start shortly
150	File status is OK; data connection will be open shortly

Responses (continued)

Code	*Description*
Positive Completion Reply	
200	Command OK
211	System status or help reply
212	Directory status
213	File status
214	Help message
215	Naming the system type (operating system)
220	Service ready
221	Service closing
225	Data connection open
226	Closing data connection
227	Entering passive mode; server sends its IP address and port number
230	User login OK
250	Request file action OK

Responses (continued)

Code	*Description*
	Positive Intermediate Reply
331	User name OK; password is needed
332	Need account for logging
350	The file action is pending; more information needed

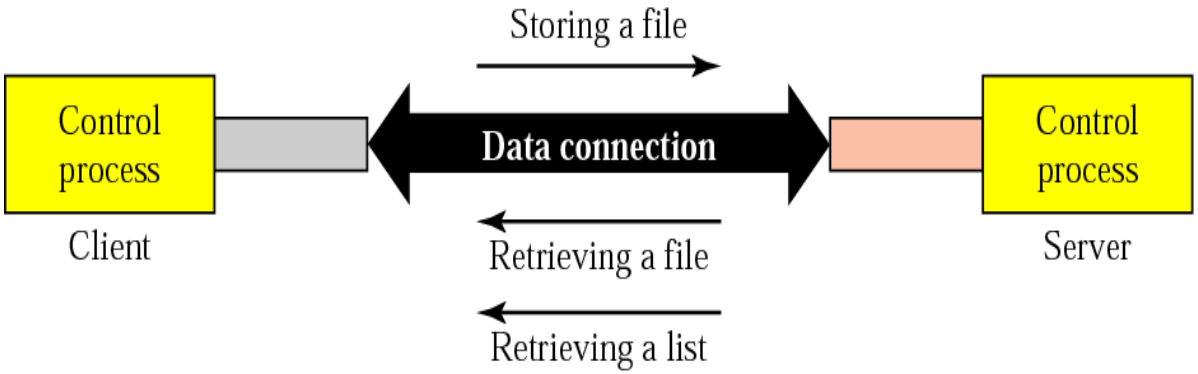
Responses (continued)

Code	*Description*
Transient Negative Completion Reply	
425	Cannot open data connection
426	Connection closed; transfer aborted
450	File action not taken; file not available
451	Action aborted; local error
452	Action aborted; insufficient storage
Permanent Negative Completion Reply	
500	Syntax error; unrecognized command

Responses (continued)

Code	*Description*
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command parameter not implemented
530	User not logged in
532	Need account for storing file
550	Action is not done; file unavailable
552	Requested action aborted; exceeded storage allocation
553	Requested action not taken; file name not allowed

File transfer



EXAMPLE 1

Next Figure shows an example of using FTP for retrieving a list of items in a directory.

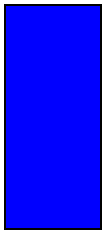
1. After the control connection to port 21 is created, the FTP server sends the 220 (service ready) response on the control connection.

2. The client sends the USER command.

3. The server responds with 331 (user name is OK, password is required).

4. The client sends the PASS command.

5. The server responds with 230 (user login is OK)



EXAMPLE 1 (CONTINUED)

6. The client issues a passive open on an ephemeral port for the data connection and sends the PORT command (over the control connection) to give this port number to the server.

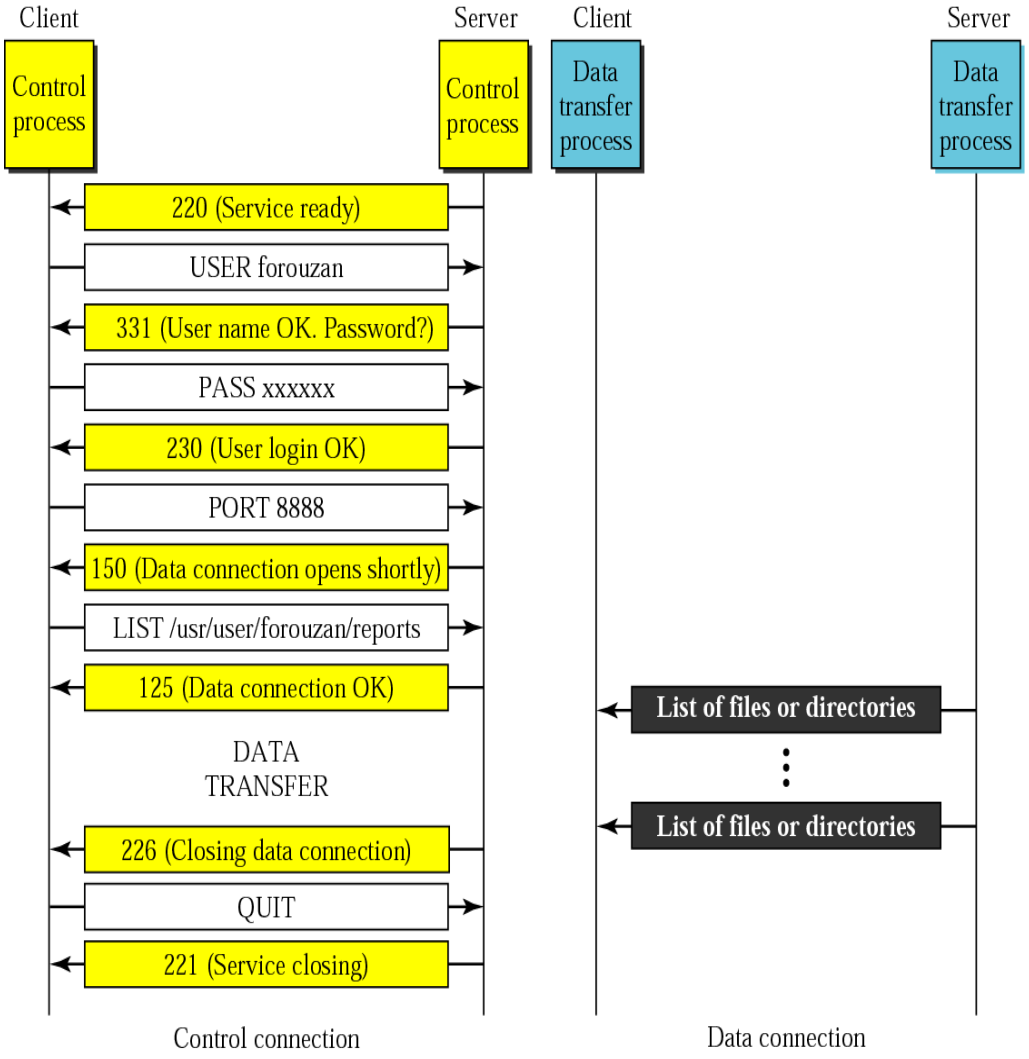
7. The server does not open the connection at this time, but it prepares itself for issuing an active open on the data connection between port 20 (server side) and the ephemeral port received from the client. It sends response 150 (data

EXAMPLE 1 (CONTINUED)

10. The server then sends the list of the files or directories (as a file) on the data connection. When the whole list (file) is sent, the server responds with 226 (closing data connection) over the control connection.

11. The client now has two choices. It can use the QUIT command to request the closing of the control connection or it can send another command to start another activity (and eventually open another data connection). In our example, the client

Figure 19.8 *Example 1*



EXAMPLE 2

The following shows an actual FTP session that parallels Example 1. 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 black background shows data transfer.

```
$ 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.
```

EXAMPLE 2

```
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.
```

EXAMPLE 3

Next Figure shows an example of how an image (binary) file is stored.

1. After the control connection to port 21 is created, the FTP server sends the 220 (service ready) response on the control connection.

2. The client sends the USER command.

3. The server responds with 331 (user name is OK, a password is required).

4. The client sends the PASS command.

5. The server responds with 230 (user login is OK).

6. The client issues a passive open on an ephemeral port for the data connection and sends the PORT command (over

EXAMPLE 3 (CONTINUED)

7. The server does not open the connection at this time, but
   prepares itself for issuing an active
   open on the data connection between port 20 (server side)
   and the ephemeral port received
   from the client. It sends the response 150 (data connection
   will open shortly).
8. The client sends the TYPE command.
9. The server responds with the response 200 (command OK).
10. The client sends the STRU command.
11. The server responds with 200 (command OK).
12. The client sends the STOR command.

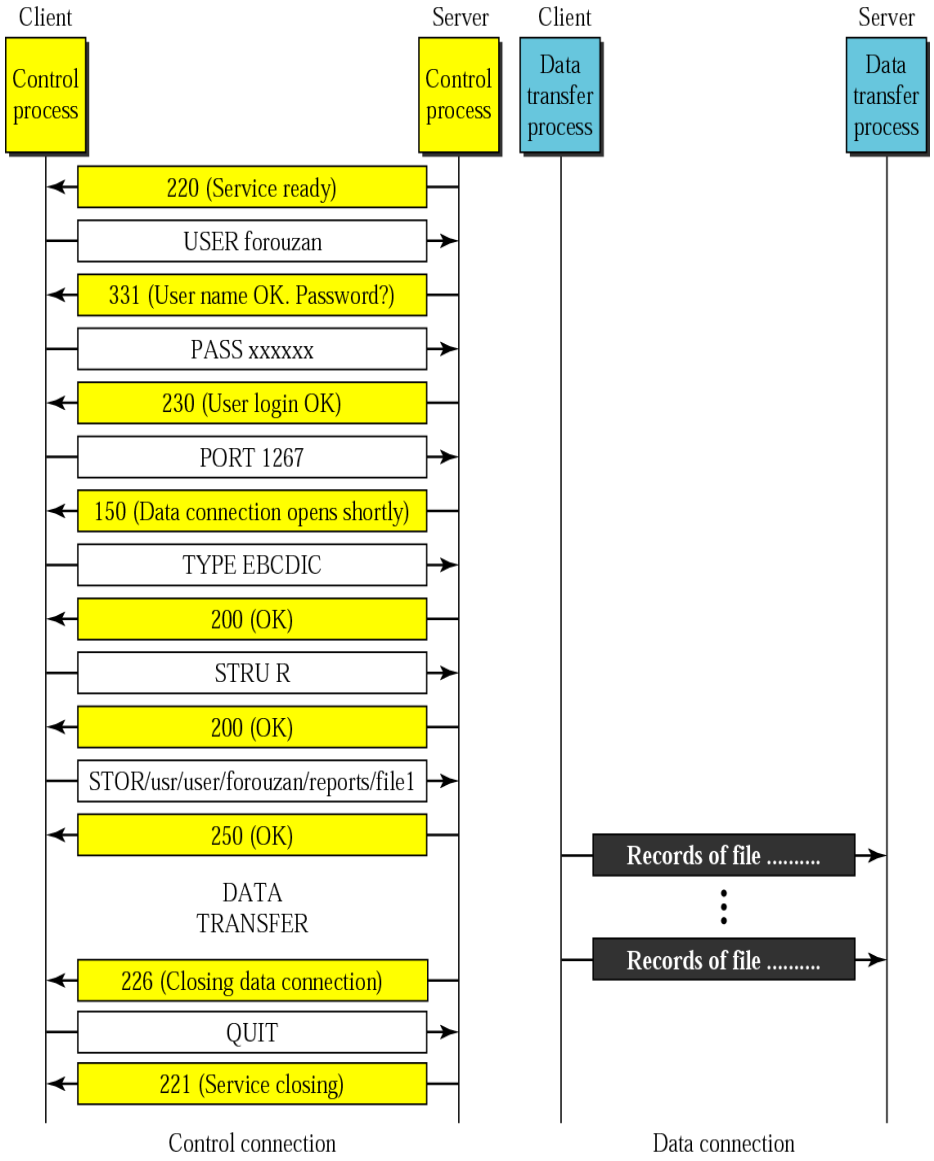
EXAMPLE 3 (CONTINUED)

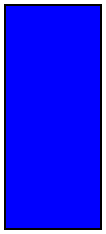
14. The client sends the file on the data connection. After the entire file is sent, the data connection is closed. Closing the data connection means end-of-file.

15. The server sends the response 226 on the control connection.

16. The client sends the QUIT command or uses other commands to open another data connection for transferring another file. In our example, the QUIT command is sent.

Figure 19.9 *Example 3*





EXAMPLE 4

We show an example of anonymous FTP. We assume that some public data are available at internic.net.

```
$ ftp internic.net
Connected to internic.net
220 Server ready
Name: anonymous
331 Guest login OK, send "guest" as password
Password: guest
ftp > pwd
257 '/' is current directory
```

EXAMPLE 4

```
bin
. . .
. . .
. . .
```

```
ftp > close
221 Goodbye
ftp > quit
```

Figure 17.19 *Resource record format*