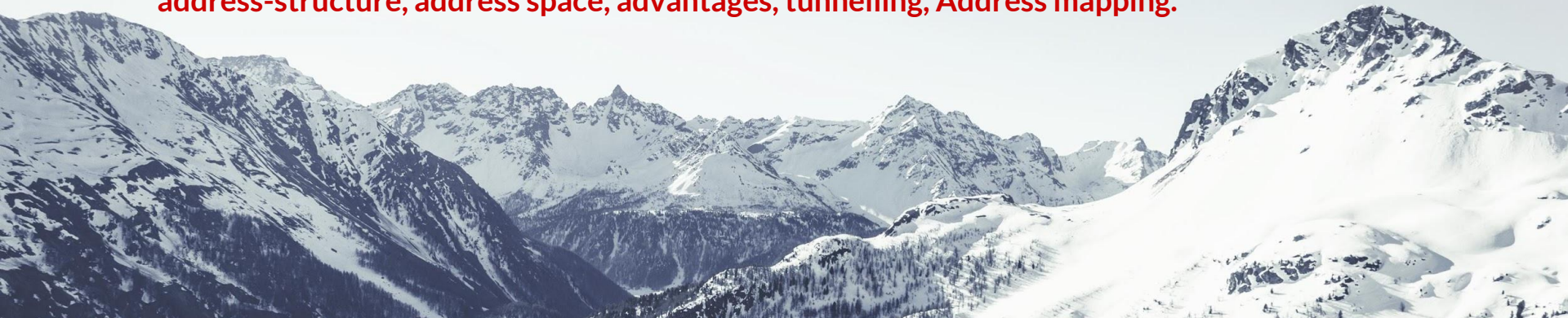




# Networking Fundamentals - Module 2

**Framing- Flow Control, Error Control, Noisy and Noiseless Channels. Network Layer: Logical Addressing, IPV4 Address-Address Space Notation, Network Address Translation. IPV6 address-structure, address space, advantages, tunnelling, Address mapping.**



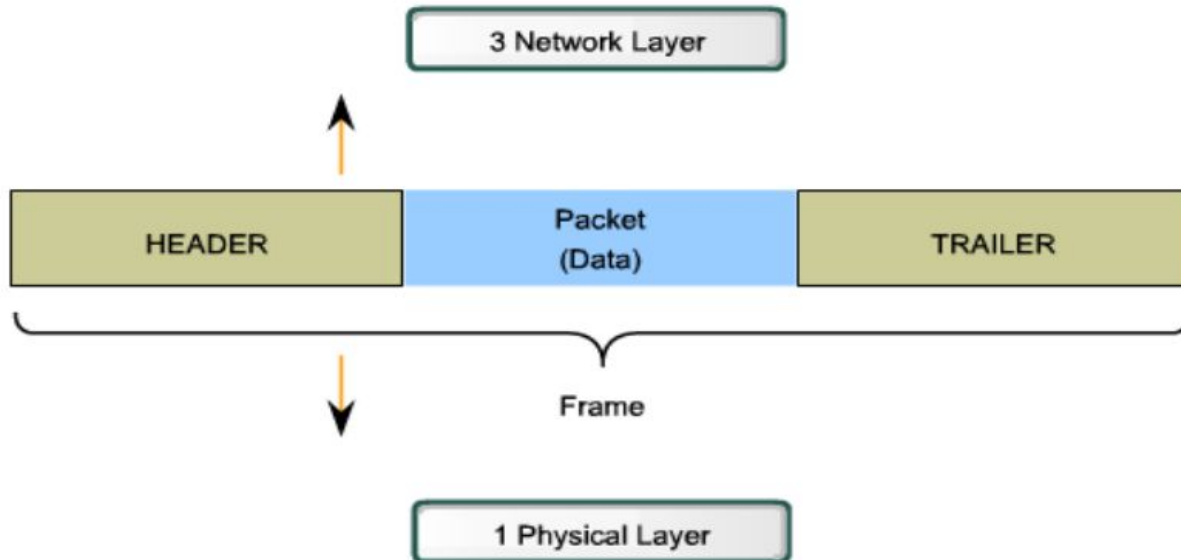
# Framing



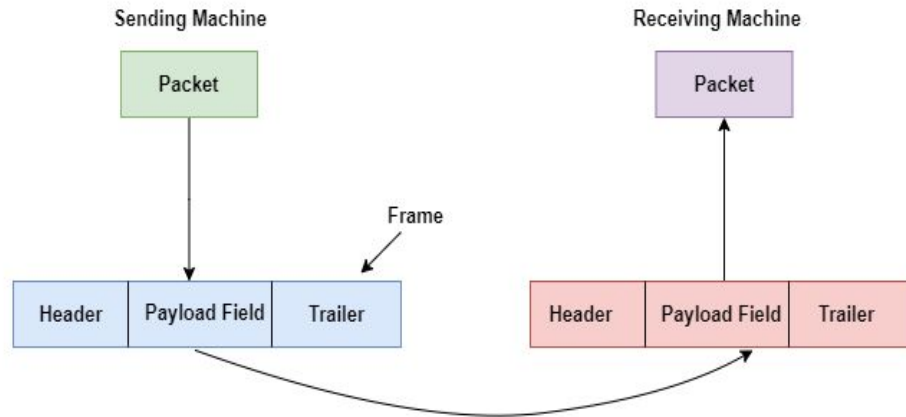
Framing is the primary function of the data link layer which provides a way for a sender to transmit data to the receiver. The data link layer receives packets from the network layer and then adds source and destination addresses & error detection or error correction redundant bits to the packets. They are converted into frames in the data link layer.

The Data Link layer frame includes:

- Data / Payload Field - The packet from the Network layer
- Header - Contains control information, such as addressing, and is located at the beginning of the PDU
- Trailer - Contains Error Control information added to the end of the PDU



- Our postal system practices a type of framing. The simple act of inserting a letter into an envelope separates one piece of information from another; the envelope serves as the delimiter. In addition, each envelope defines the sender and receiver addresses. The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt. The data link layer packs bits into frames such that each frame is distinguishable from another.
- When the frame size becomes large, a packet is divided into a small frame. These smaller-sized frames enable error control and flow control more efficiently.
- Then, it sends each frame bit-by-bit thru the hardware in the form of signals. At receiver's end, data link layer picks up signals from the hardware and assembles them into frames.
- The physical layer doesn't know anything about the frames. It takes individual bits and converts them into equivalent signals before sending them. Now, the receiving device's physical layer has to convert these bits into frames.



# Types of Framing



There are two types of framing that are used by the data link layer in computer networks. They are **fixed sized framing** and **variable sized framing**.

## Fixed Size Framing

**Here the size of the frame is fixed and the frame length acts as delimiter of the frame.** So there is no need for defining the boundaries of the frames to mark the beginning and end of a frame.

For example:- If a device is sending 200 bits of data and the 50 bits of frame size is fixed, then after receiving the 50 bits of data, the receiver will automatically know that the next 50 bits are of frame two and so on.

An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.

## Variable Size Framing

**In this type of framing, the size of each frame to be transmitted may vary.** So, we need a way to define the end of the frame and the beginning of the next.

For example, It is possible that out of 200 bits of data, 100 bits constitute frame1, 25 bits constitute frame2, and the rest bits constitute frame3. Hence additional mechanisms are needed to mark the end of a frame and the beginning of the next frame.

Variable-size framing is widely used in local- area networks.

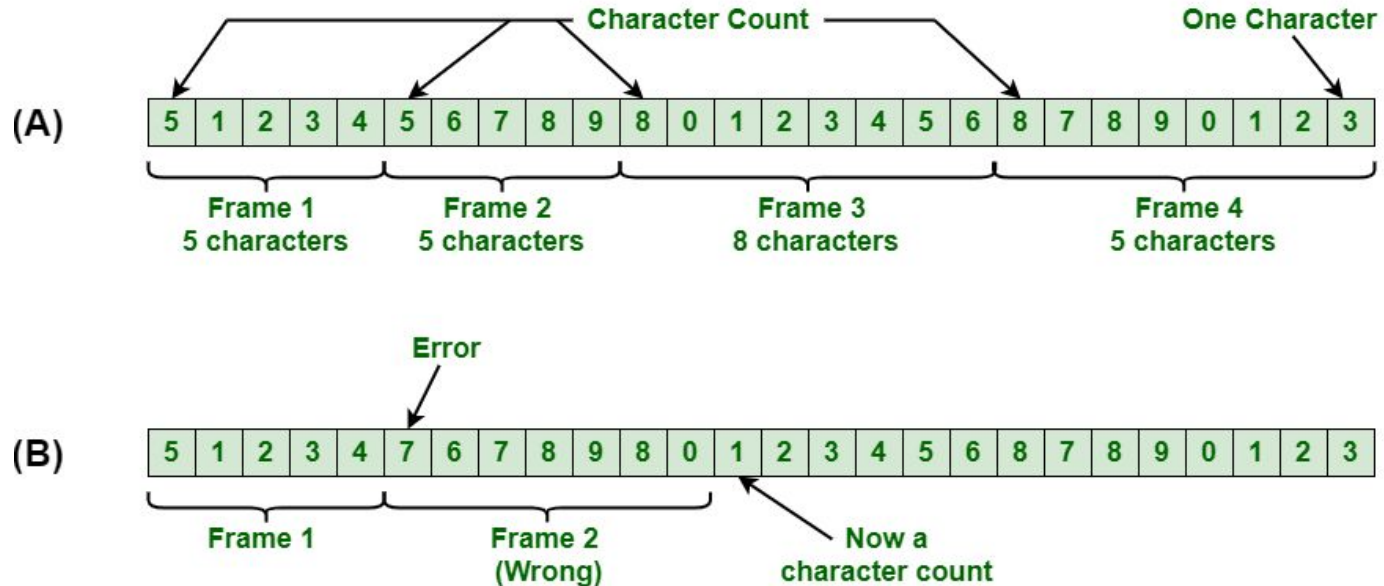
## Character count framing



In the first framing technique, a field is used in the header to specify the number of characters in the frame. By seeing the character count at the destination, the data link layer knows how many characters follow and hence where the end of the frame is.

The disadvantage of this method is if a character count in header is distorted by an error occurring during transmission, then the receiver might lose synchronization. The receiver might not be able to locate or identify beginning of next frame.

*For example, if the character count of 5 in the second frame, (see above fig ) becomes a 7, the destination will get out of synchronization and will not be able to locate the start of the next frame. **Due to this reason, the character count technique is rarely used anymore.***



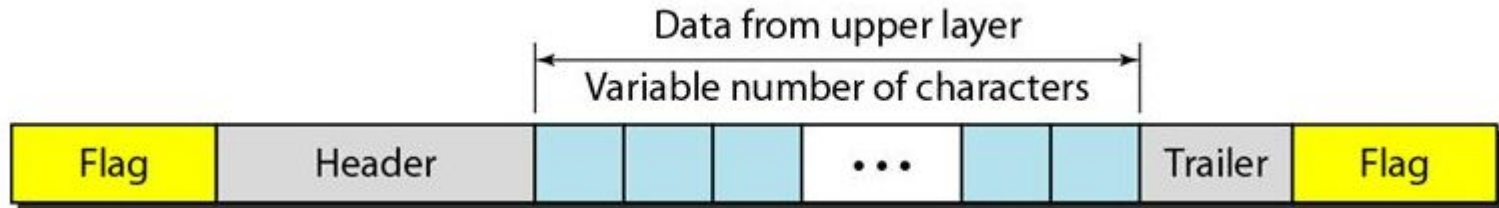
In the next technique of variable-size framing, we need to define the end of the frame and the beginning of the next. There are two approaches which are used for this purpose:

**A character-oriented approach** and **A bit-oriented approach**.

### Character-oriented framing or Byte-oriented framing

In a character-oriented protocol, data to be carried are 8-bit characters from a coding system such as ASCII. The header, which normally carries the source and destination addresses and other control information, and the trailer, which carries error detection or error correction redundant bits, are also multiples of 8 bits.

To separate one frame from the next, an 8-bit flag is added at the beginning and the end of a frame. The flag is a 1-byte special character, which marks the start or end of a frame. The following figure shows the format of a frame in a character-oriented protocol.



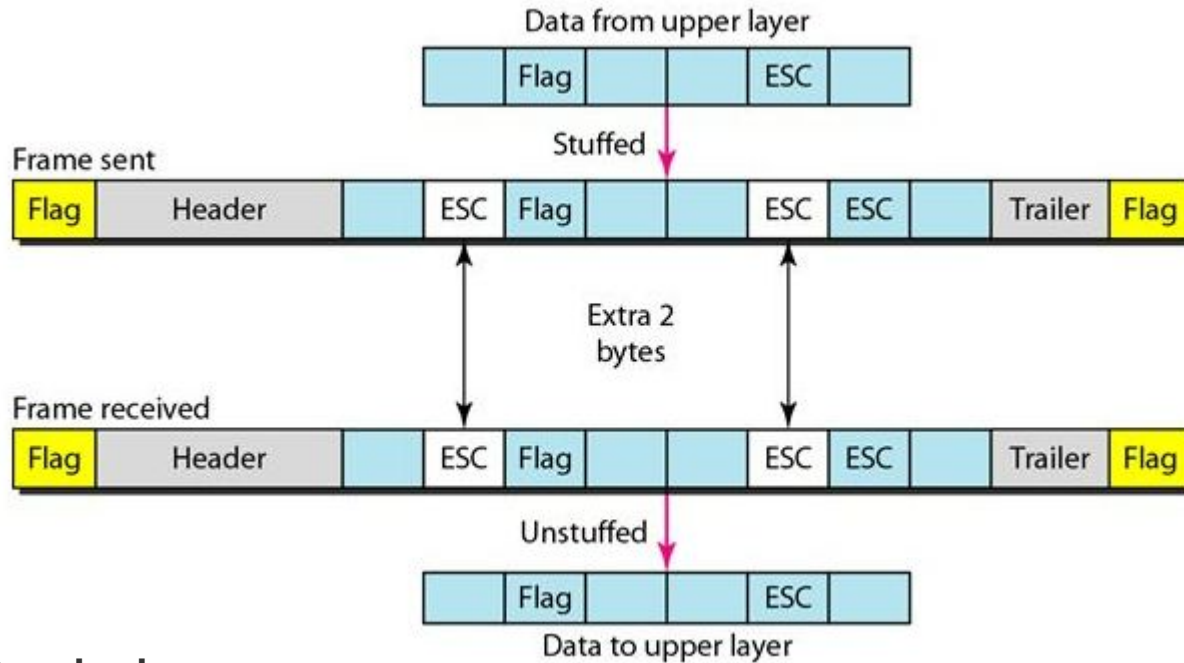
Character-oriented framing was popular when only text was exchanged by the data link layers. The flag could be selected to be any character not used for text communication.

However, if we send other types of information such as graphs, audio, and video; any pattern used for the flag could also be part of the information. If this happens, the receiver, when it encounters this pattern in the middle of the data, thinks it has reached the end of the frame.

To fix this problem, a byte-stuffing (or character stuffing) strategy was added to character-oriented framing.

- In byte stuffing, a special character is added to the data section of the frame when there is a flag character in the data. The data section is stuffed with an extra byte. This byte is usually called the escape character (ESC), which has a predefined bit pattern.
- Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not a delimiting flag.

Byte stuffing by the escape character allows the presence of the flag in the data section of the frame, but it creates another problem. What happens if the text contains one or more escape characters followed by a flag? The receiver removes the escape character, but keeps the flag. To solve this problem, the escape characters that are part of the text must also be marked by another escape character. In other words, if the escape character is part of the text, an extra one is added to show that the second one is part of the text. The following figure shows the situation.



### Drawback:

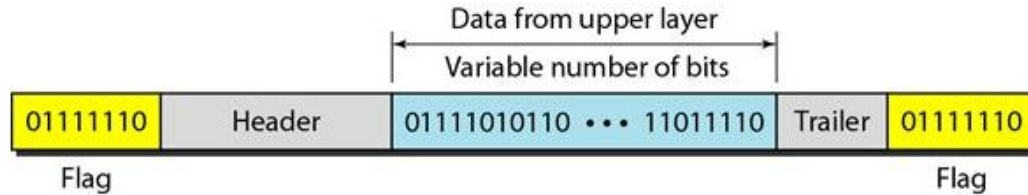
- A problem with character - oriented framing is that it adds too much overhead on the message, thus increasing the total size of the frame.
- Another disadvantage of using this framing method is that it is closely tied to the use of 8-bit characters. But the universal coding systems in use today, such as Unicode, have 16-bit and 32-bit characters that conflict with 8-bit characters.



## Bit-oriented framing



In a bit-oriented protocol, the data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video, and so on. Most protocols use a special 8-bit pattern flag 01111110 as the delimiter to define the beginning and the end of the frame, as shown in the following figure.



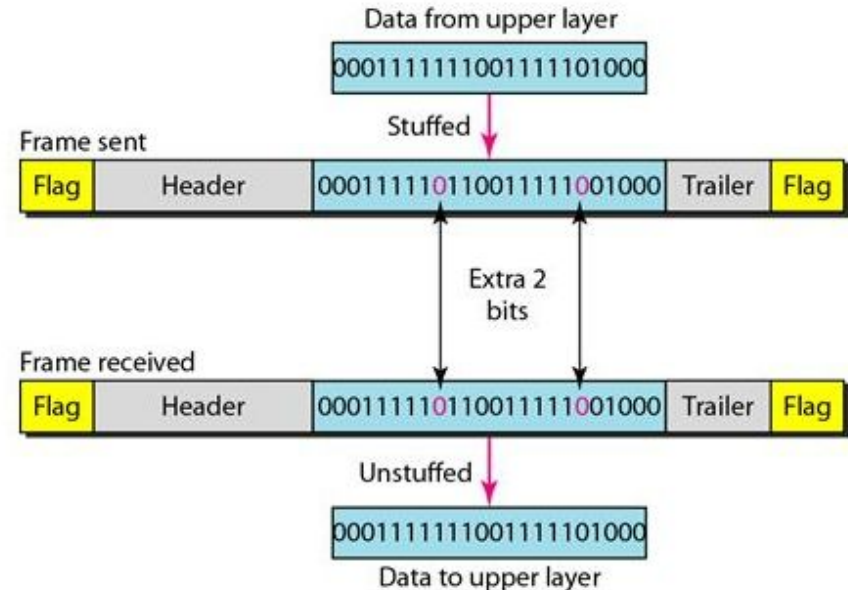
The parts of a frame in a bit-oriented framing are

- Frame Header – It contains bits denoting the source and the destination addresses of the frame.
- Payload field – It contains the message to be delivered. It is a variable sequence of bits.
- Trailer – It contains the error detection and error correction bits.
- Flags – Flags are a bit pattern that act as the frame delimiters signalling the start and end of the frame.

This flag can create the same type of problem we saw in the byte-oriented protocols. That is, if the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame.

We do this by stuffing a single bit to prevent the pattern from looking like a flag. The strategy is called bit stuffing. **In bit stuffing, if a 0 and five consecutive 1 bits are encountered, an extra 0 is added.** This extra stuffed bit is eventually removed from the data by the receiver. Note that the extra bit is added after one 0 followed by five 1s regardless of the value of the next bit.


The following figure shows bit stuffing at the sender and bit removal at the receiver. Note that even if we have a 0 after five 1s, we still stuff a 0. The 0 will be removed by the receiver.





# Flow Control & Error Control in DATA LINK LAYER

The most important responsibilities of data link layer are flow control & error control. Collectively these functions are called **data link control**

- 
- Flow control in the data link layer is a technique that controls the rate of data transmission between the sender and receiver. It allows two machines working at different speeds to communicate with each other.
  - If the sender is on a fast and powerful computer as compared to the receiver then the receiver may get overwhelmed by the frames transmitted by the sender.
  - i.e. Flow control refers to the set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgement.

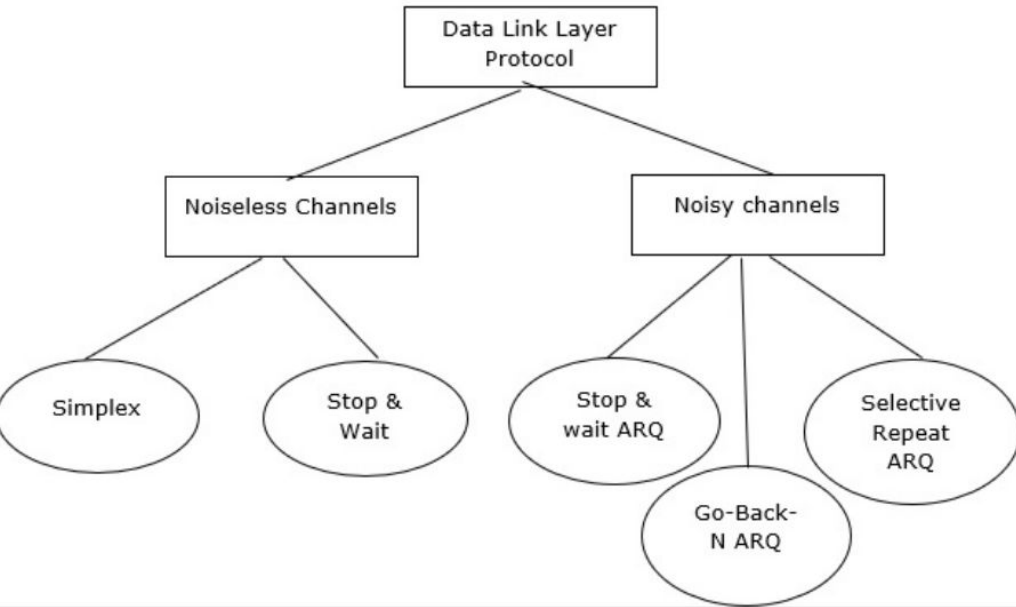
## Error Control

- Data link layer uses error control techniques to ensure that all data frames are transferred from sender to receiver with certain accuracy.
- Error control involves both error detection & error correction. Even though modern reliable equipments & transmission media are used, communication errors are inevitable. When an error is detected, the receiver can request for retransmission of the frame to the sender. This process is known as ARQ - Automatic Repeat reQuest
- The error control function of data link layer detects the errors in transmitted frames and re-transmits all the erroneous frames.
- **Error control in Data Link Layer is implemented by adding a CRC to the frame header by the sender and checked by the receiver.**

# Data Link Layer Flow Control Protocols



Data Link Layer (DLL) protocols are divided into two categories based on whether the transmission channel is noiseless or noisy. The data link layer protocol is diagrammatically represented below –



**Noiseless Channels** - These are ideal channels in which no frames are lost, duplicated or corrupted. The two protocols of this type of channel are -

- Simplex/ Simplest Protocol
- Stop & wait Protocol

## Noisy Channels

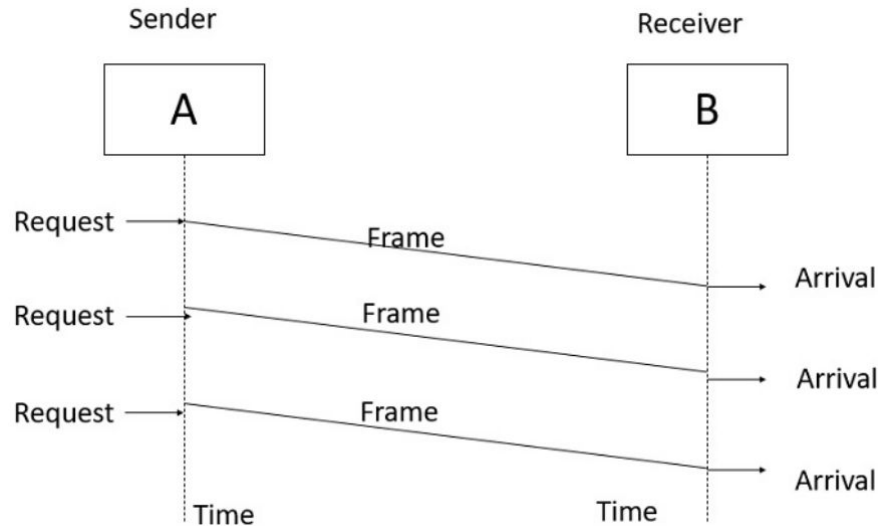
Noiseless channels are non-existent channels. The three types of noisy channel protocols are –

- Stop & wait Automatic Repeat Request.
- Go-Back-N Automatic Repeat Request.
- Selective Repeat Automatic Repeat Request.

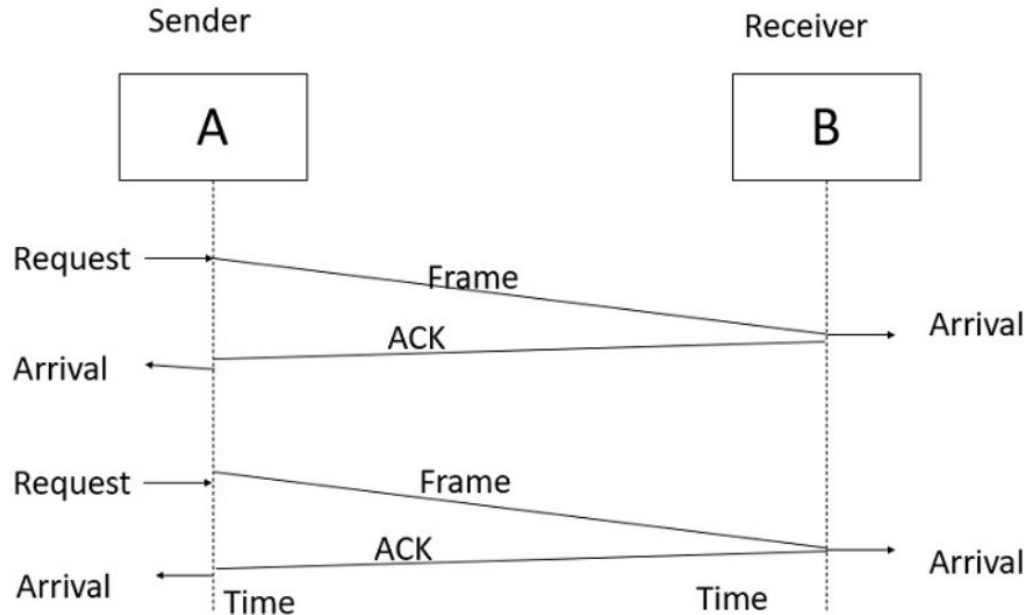
# Simplest / Simplex Protocol [ Noiseless]



- Simplest protocol does not have flow or error control. .ie. We assume that the receiver can handle any frame it receives with a processing time that is small enough to be negligible.
- It is a unidirectional protocol where data frames are traveling in one direction only, that is from the sender to receiver.
- The DLL at the sender gets packet from its network layer, makes a frame out of it & transmits it to the receiver. The DLL at receiver, extracts the packet from the frame and gives it to the network layer.



# Stop & Wait Protocol [ Noiseless]



- It is a DLL flow control protocol over noiseless channels.
- It provides unidirectional data transmission with flow control facilities but without error control facilities.
- The sender sends a single frame and waits for acknowledgment.
- Once the receiver receives the frame, it sends an acknowledgment frame, ACK back to the sender.
- On receiving the acknowledgment frame, the sender understands that the receiver is ready to accept the next frame. So it sends the next frame in queue.
- The main advantage of the Stop and Wait protocol is its simplicity,

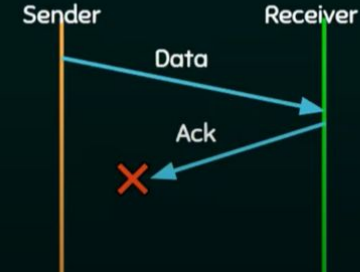
## 1. Problems due to lost data.

- ★ Sender waits for ack for an infinite amount of time
- ★ Receiver waits for data an infinite amount of time



## 2. Problems due to lost ACK.

- ★ Sender waits for an infinite amount of time for ack.

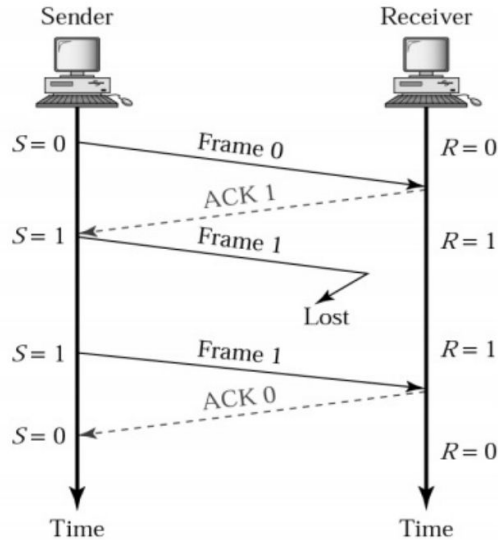


### Problems associated with Stop and Wait protocol -

1. The transmission process is extremely slow; Since only a single frame can be sent at a time & it has to wait for the ACK before sending the next packet.
2. If the data send by the sender is lost, the receiver waits for it infinitely; also the sender waits for its ACK infinitely.
3. If the ACK sent by the receiver is lost, the sender will be waiting for the ACK, and the receiver will be waiting for the next frame for infinite time.
4. It does not use the bandwidth entirely, because only a single frame or ACK is transferred at a time.



# Stop & Wait ARQ Protocol [ Noisy Channel]

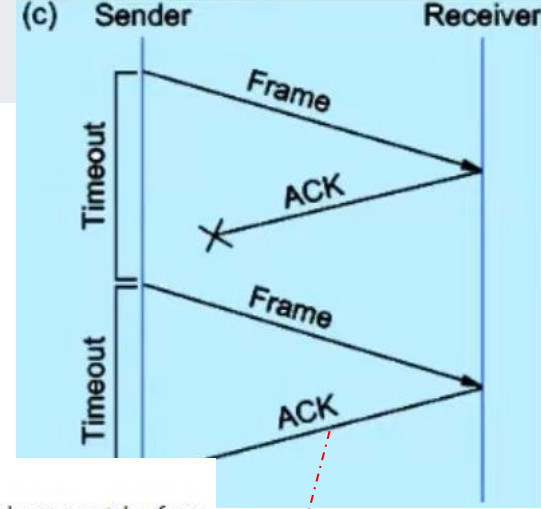
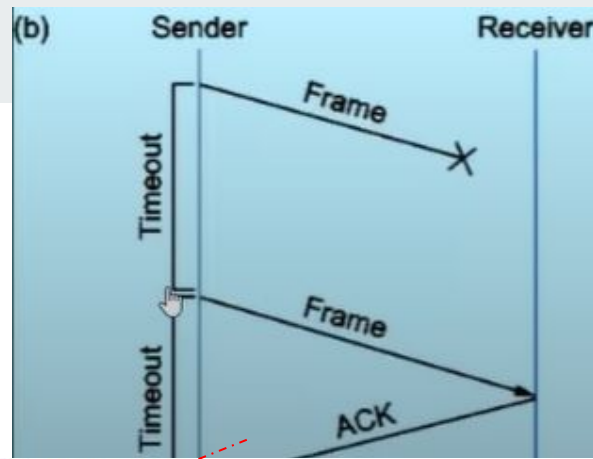
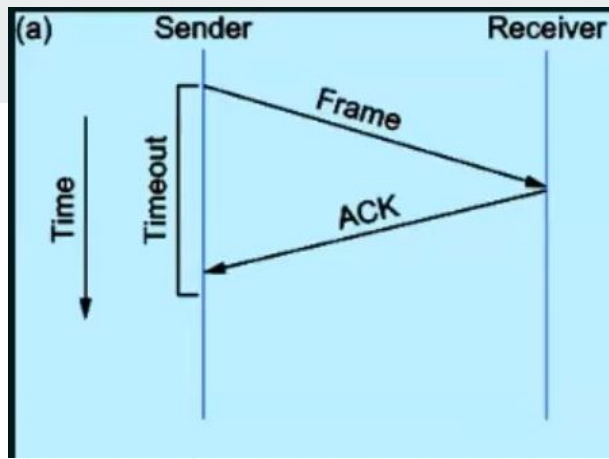


## Working of Stop and Wait for ARQ:

- 1) Sender sends a data frame or packet with sequence number 0.
- 2) Receiver, after receiving the data frame, sends an ACK with sequence number 1 (the sequence number of the next expected data frame or packet)

There is only a one-bit sequence number that implies that both sender and receiver have a buffer for one frame only.

- The main problem of Stop & Wait protocol is infinite waiting time by sender & receiver, when there is a loss of data frame/ACK.
- Stop & Wait ARQ is an improved version of Stop & Wait protocol. Here if the ACK does not arrive after a certain period of time, the sender times out and retransmits the same frame. This retransmission is automatic; so we call it Stop and Wait **ARQ (Automatic Repeat reQuest protocol)**.
- The sender waits for the ACK from the receiver. Every time the sender sends a frame, it starts a timer. If an ACK is received before the time expires, the timer is stopped & sender sends the next frame. If ACK is not received, the timer expires & the sender re-sends the previous frame assuming that it was lost or damaged.
- Another condition is, the data frame is received, but the ACK frame returned by receiver is lost. After waiting, the sender re-sends the same frame & this causes duplication. To solve this, sequence numbers are added to both data frames & ACK frames.
- **Stop-and- Wait ARQ = Stop-and- Wait + Timeout Timer + Sequence Number**



**The acknowledgement is received before the timer expires.**

The timeline diagram shows that the sender has sent the frame, and the receiver sent acknowledgement before the timeout. So, this is perfect as far as the Stop and Wait protocol is concerned

**The original frame is lost**

Here the sender is sending a frame, and the frame is lost. In this scenario, the original frame is lost. So if the sender has not received any acknowledgement before the timeout expires: the sender will again retransmit the frame after the timeout; this process is automatic. The Stop and Wait ARQ protocol differs from the Stop and Wait Protocol in this way.

**The acknowledgement is lost.**

Here the receiver is sending an acknowledgement, and the acknowledgement is lost. So, in this case, if the sender didn't receive any acknowledgement before the timeout expires: the sender will retransmit the frame after the timeout.

**Advantages :**

- The main advantage of stop & wait protocols is their accuracy. The next frame is transmitted only when the first frame is acknowledged. So there is no chance of the frame being lost.
- It is used in connection oriented communication; It offers error control & flow control
- It is used in data link & transport layer

## Disadvantages - Stop & Wait ARQ Protocol

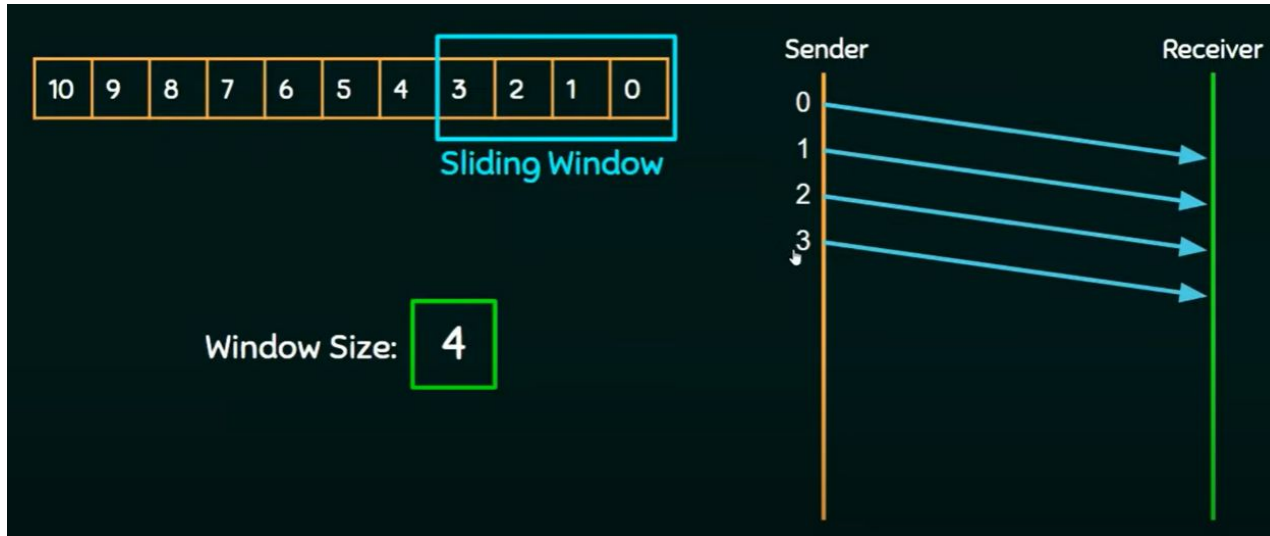
- It can send only one frame at a time. Since next frame is sent only after receiving ACK of the previous frame, waiting time is wasted.
- Only one frame can be in transmission at a time; this leads to inefficiency, if propagation delay is much longer than transmission delay.
  - Transmission delay : Time taken by a station to transmit a frame.
  - Propagation delay : Time taken by a frame to travel from the sender to the receiver
- If two devices are a distance apart, i.e. in the case of satellite communications, a lot of time is wasted waiting for ACKs leading to an increase in total transmission time. This causes very poor bandwidth utilization.
- Poor utilization of bandwidth and therefore Poor performance
- To improve link utilization, Sliding window protocol is used.

# Sliding Window Protocol



- The sliding window is a technique for sending multiple frames at a time, before receiving ACK from the receiver.
- Sliding window refers to an imaginary box that hold the frames on both sender & the receiver. Actually these are buffer areas which stores these frames temporarily.
- Each frame is numbered, which is called a sequence number.
- The windows have a specific size in which frames are numbered 0 to n-1.
- The number of frames that can be sent is based on the **window size**.

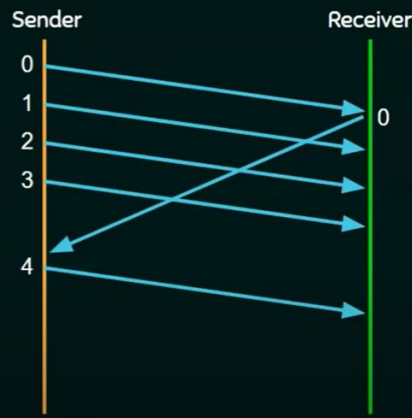
## Working of a Sliding Window Protocol :



- Suppose the Sender has to send 11 frames to the receiver; and the frames are numbered from 0 to 10.
- The number of frames to be sent is decided by the parameter **window size** and let it be 4.
- Here 4 frames can be sent before expecting an ACK. Therefore at the beginning, the sliding window contains frames 0 to 3



Window Size: **4**



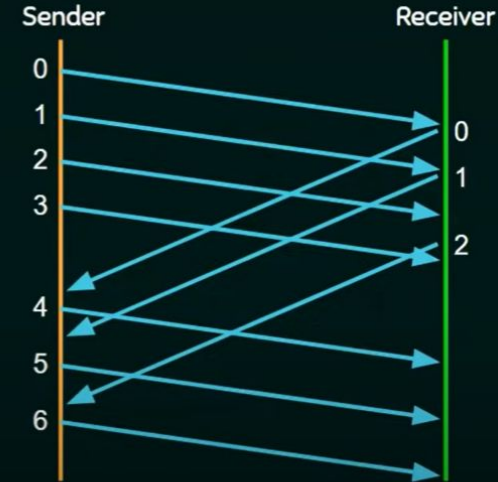
- When ACK of frame 0 is received, the sender can send the next frame, ie frame4.
- Now the sliding window moves to include frame4
- Like-wise, on receiving ACKs of sent frames, the sliding window moves to include all frames to be sent.



Not Yet Sent

Sent but not acknowledged

Already Sent and Acknowledged



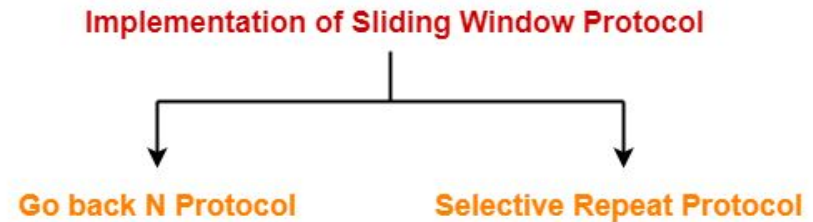
Window Size: **4**

The flow control protocols are generally divided into two categories i.e.

1. Stop and wait protocol
2. Sliding window protocol.

The difference between these two categories of flow control protocols is that in the stop and wait protocol, only one data frame is sent at a time from sender to receiver. While in sliding window protocol, multiple frames can be sent at a time from sender to receiver.

The two well known implementations of sliding window protocol are-



1. Go back N Protocol
2. Selective Repeat Protocol

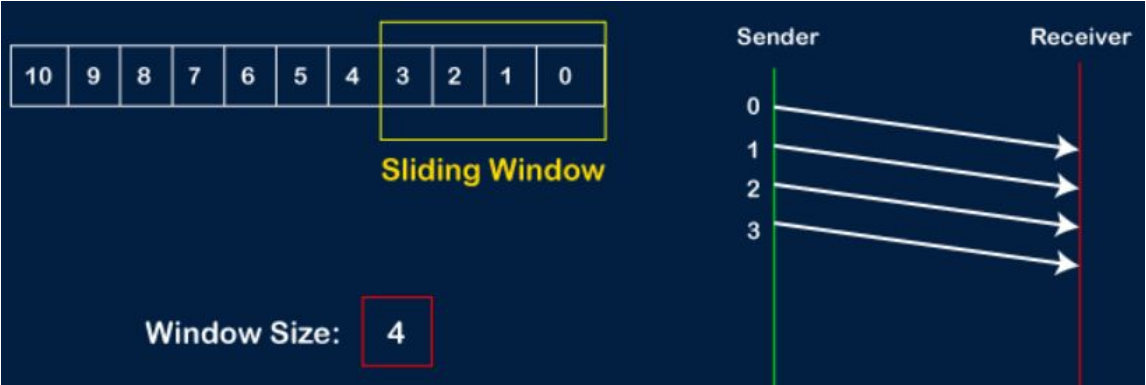
- Go Back N ARQ is a sliding window protocol which is used for flow control purposes. Multiple frames present in a single window are sent together from sender to receiver.
- In Go-Back-N ARQ, the N is the sender window size. Suppose if we say Go-Back-4, the sender window size is 4. It means it can send 4 frames simultaneously before expecting any acknowledgement from the receiver.
- Pipelining is allowed in the Go Back N ARQ protocol. Pipelining means sending a frame before receiving the acknowledgment for the previously sent frame. i.e. The sender can send multiple frames before receiving the acknowledgement for the first frame.
- There are finite number of frames, and the frames are numbered sequentially.
- The number of frames that can be sent depends on the sender's window size.
- If the acknowledgement of a frame is not received within a certain period of time, all the frames present in the current window will be transmitted.

# Working of Go-Back-N ARQ



Suppose there is a sender and a receiver. There are 11 frames to be sent, and the frames are numbered as 0,1,2,3,4,5,6,7,8,9,10. The sequence number of the frames is decided by the size of the window N. Let's take the sender's window size to be 4, which means the sender can send 4 frames before expecting any acknowledgement from the first frame, which is 0.

**Step 1:** Firstly, the sender will send the first four frames to the receiver, i.e., 0,1,2,3, and now the sender is expected to receive the acknowledgment of the 0<sup>th</sup> frame.



Let's assume that the receiver has sent the acknowledgment for the 0 frame, and the receiver has successfully received it.





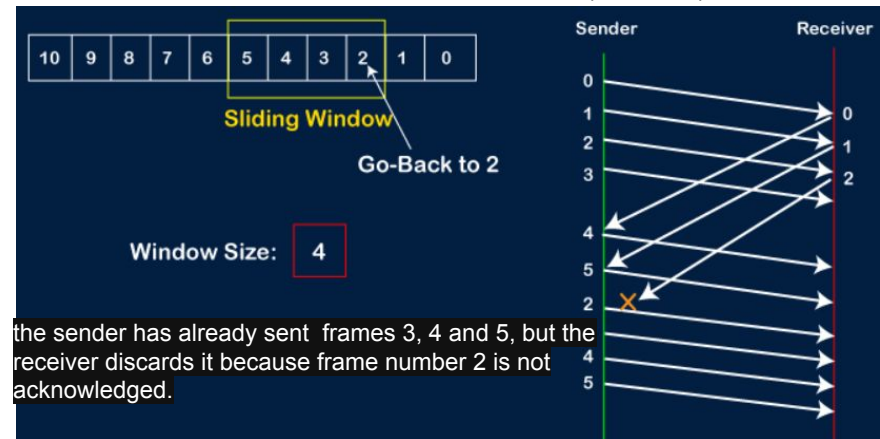
The sender will then send the next frame, i.e., 4, and the window slides containing four frames (1,2,3,4).

The receiver will then send the acknowledgment for the frame no 1. After receiving the acknowledgment, the sender will send the next frame, frame no 5, and the window will slide having four frames (2,3,4,5).



Let's assume that the sender is not acknowledging frame number 2 because either the frame is lost or the acknowledgement is lost.

Now, the sender will wait for a certain time. If the ACK of frame 2 is not received, instead of sending the frame no 6, it will go back to 2, and whatever the frames are there in the current window, it will retransmit all of them (2, 3, 4, 5).



## Advantages of Go-Back-N ARQ

1. It can send multiple frames at once.
2. Pipelining is present in the Go-Back-N ARQ i.e. a frame can be sent by the sender before receiving the acknowledgment of the previously sent frame. This results in a lesser waiting time for the frame.
3. It handles corrupted as well as out-of-order frames which result in minimal frame loss.

## Disadvantages of Go-Back-N ARQ

1. If acknowledgment for a frame is not received, the whole window of frames is retransmitted instead of just the corrupted frame. This makes the Go Back N ARQ protocol inefficient.
2. Retransmission of all the frames on detecting a corrupted frame increases channel congestion and also increases the bandwidth requirement.
3. It is more time-consuming because while retransmitting the frames on detecting a corrupted frame, the error-free frames are also transmitted.

**Selective Repeat Automatic Repeat Request (ARQ)**, is a data link layer protocol that uses the **sliding window technique** for reliable data frame delivery. Only erroneous or lost frames are retransmitted in this case, while good frames are received and buffered.

The sender sends several frames specified by a window size in the selective repeat without waiting for individual acknowledgement from the receiver as in Go-Back-N ARQ.

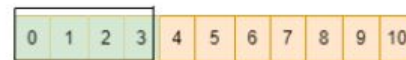
## Working of Selective Repeat ARQ

- In Selective Repeat ARQ, only the erroneous or lost frames are retransmitted, while correct frames are received and buffered.
- While keeping track of sequence numbers, the receiver buffers the frames in memory and sends **NACK** (negative acknowledgement) for only the missing or damaged frame.
- The sender will send/retransmit the packet for which NACK (negative acknowledgement) is received.

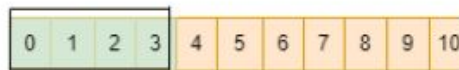
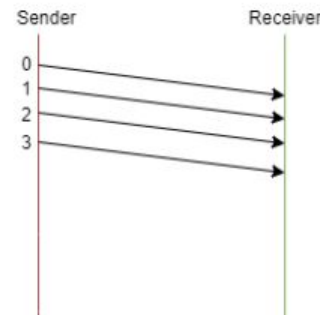
Example :

Suppose there is a sender and a receiver. There are 11 frames to be sent, and the frames are numbered as 0,1,2,3,4,5,6,7,8,9,10. The sequence number of the frames is decided by the size of the window N.

Let's take the sender's window size to be 4, which means the sender can send 4 frames before expecting any acknowledgement from the first frame, which is 0.

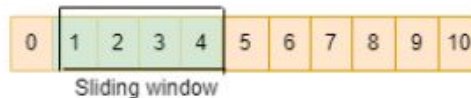
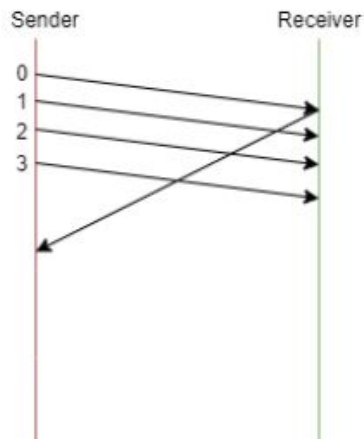


Window Size: 4



Window Size: 4

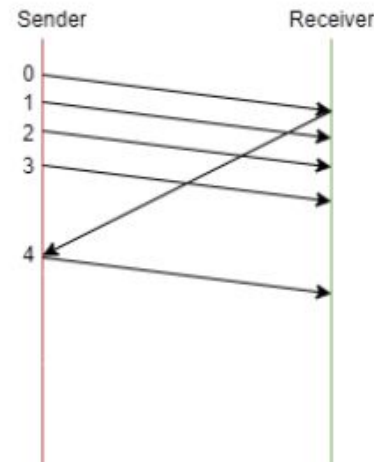
Let's assume the sender received an acknowledgement for frame 0 from the receiver.

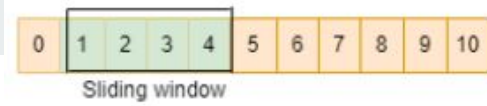


Sliding window

Window Size: 4

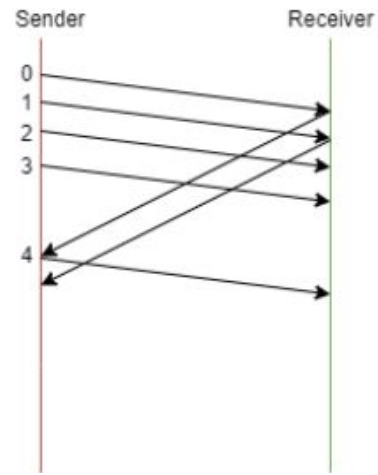
Frame 0 is sent and acknowledged. The current window size is 3; the sender will send the next frame from the buffer, which is 4, and the window slides.





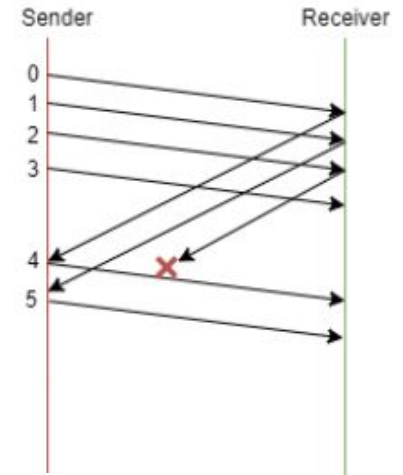
Window Size: 4

Now, the current window size is 4. So, the receiver acknowledges frame number 1.

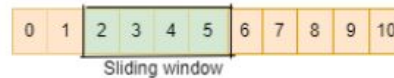


Window Size: 4

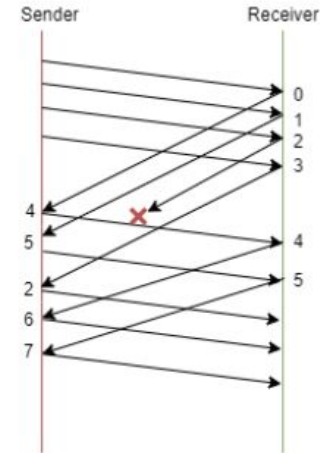
Suppose the sender is not acknowledging frame number 2 because either the frame is lost or the acknowledgement is lost.




In this case, the sender will not send frames 4 and 5 again, and it knows that frame 2 is missing because the receiver would have sent a negative acknowledgement (NACK) for frame 2. So, the sender will retransmit frame 2 alone and as usual other frames are transmitted.



Window Size: 4



# Network Layer: Logical Addressing

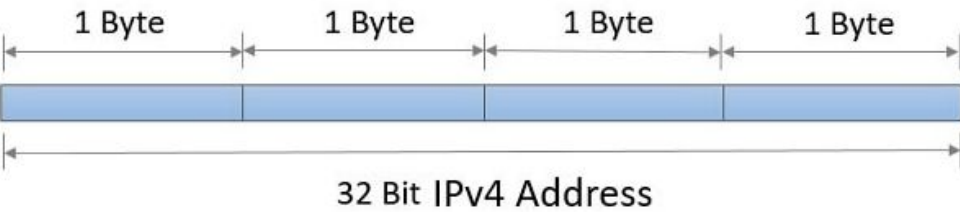
- 
- Source host to destination host communication needs a global addressing scheme - known as **Logical Address**
  - The logical addresses added by Network layer are known as IP (Internet protocol) address, it can be either IPv4(version 4) or IPv6 (version 6).
  - Internet protocol is a set of rules for the transfer of data on the internet. It is responsible for packetizing, forwarding and delivery of a packet at the network layer.

## IP Address -

**IPv4 Addresses:** IPv4 is the 4th version of the Internet Protocol. An IP version 4 address is a 32-bit address that uniquely and universally defines the connection of a device to the Internet. i.e. Two devices can never have the same address at the same time. This gives a maximum of  $2^{32}$  (4,294,967,296 -- more than 4 billion ) addresses. These addresses are unique in the sense that each address defines one, and only one, connection to the internet.

**IPv6 Addresses:** IP version 6 uses a 128-bit address, which allows  $2^{128}$  approximately  $3.4 \times 10^{38}$  addresses.

# IPv4 Addresses



- An IPv4 address is 32 bits long
- Universally & Uniquely identifies a system in the network
- There are two types of IPv4 addresses – Classful and Classless.

## Address Space:

- An address space is the total number of addresses used by the protocol. It defines the range of addresses used by the protocol. Each networking devices will get an address from this address space.
- If the protocol uses N-bits to define an address, the address space is  $2^N$ .
- IPv4 address is 32 bits in size and hence the total number of address possible is:  $2^{32} = 4,294,967,296$ . This means theoretically 4 billion devices can be connected to internet, but practically this number is way less because of some restrictions.

**Representation of IP address:** An IP address can be represented in two formats:

- **Dotted-Decimal Notation:** In this format, an IP address is represented as 4 octet, each octet of 8 bits(1 bytes) and are separated with a decimal point (dot). Example:192.168.0.1
- **Binary Notation:** In this IP address is represented as 32 bits. Example:

Dotted decimal : **192 . 168 . 0 . 1**

Binary Notation: 11000000 10101000 00000000 00000001

Arrows indicate the mapping from each octet of the dotted decimal notation to its corresponding 8-bit binary representation: 192 maps to 11000000, 168 maps to 10101000, 0 maps to 00000000, and 1 maps to 00000001.

## Classful Addressing:



- Classful addressing is an IPv4 addressing architecture that divides addresses into five groups. i.e The address space (0.0.0.0 - 255.255.255.255) is divided into five classes: A, B, C, D and E. Each class has some part of the address space.
- Class A, B, C are mostly used for unicast communication whereas class D is for multicast communication and class E is reserved.
- In dotted-decimal notation of IPv4 address the **value of the first byte recognizes the class** of the address.
- As shown in the figure, the first byte of each class denotes the range of addresses in each class.

Class A



Class B



Class C




Class D



Class E







In IPv4 addresses of class A, B & C the first part of the address is considered as **net-id** (Network id) and the second part of the address is called **host-id**. The size of these parts varies with the classes.

**Net-id:** The net-id denotes the address of the network.

**Host-id:** The hoist-id denotes the address of the host attached to the corresponding network.

In Class A, the **net-id** is defined by the **first byte** of the address. And the **rest 3 bytes** defines the **host-id**.

In Class B, the **first two bytes** of the address defines the **network address** and the **rest two bytes** defines the **host-id**.

In Class C the **first three bytes** defines the **network address** and the **last byte** defines the **host-id**.

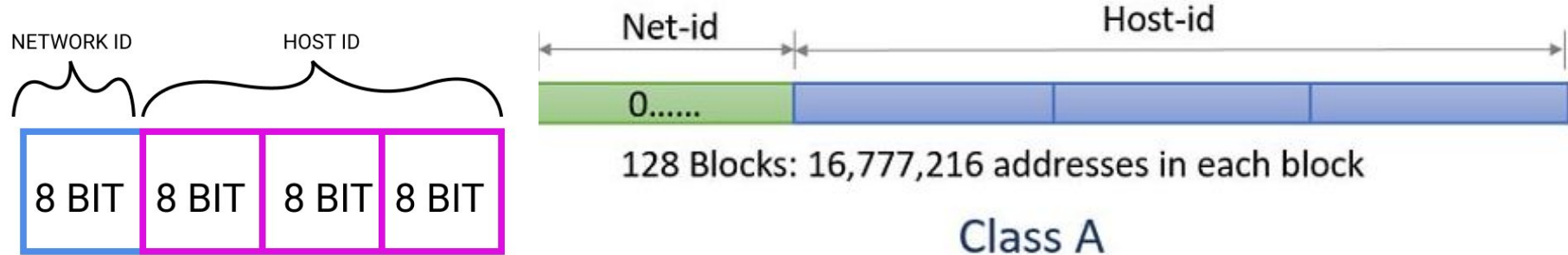
# Class A

Class A



- The **network id** of class A is defined by the **first byte** of the 32-bit IPv4 address.
- In class A, the **first bit** of the **net-id** stays '0' to define that the IPv4 address belongs to the class A. As the first bit is preserved the remaining seven bits calculate the number of blocks in the class A i.e.  $2^7 = 128$  **blocks**. There are 128 blocks in class A, as the addressing would start from 0 the range of blocks will be from 0-127.

The **host-id** in class A is defined by the **remaining three bytes** of the IPv4 address which is equal to 24 bits. So, we can calculate the **number of hosts for each block** as  $2^{24} = 1,67,77,216 \approx 1$  crore 70 lakhs. So, we conclude that we can assign 128 blocks from class A to 128 organizations where each organization can have 16,777,216 hosts connected to the network.



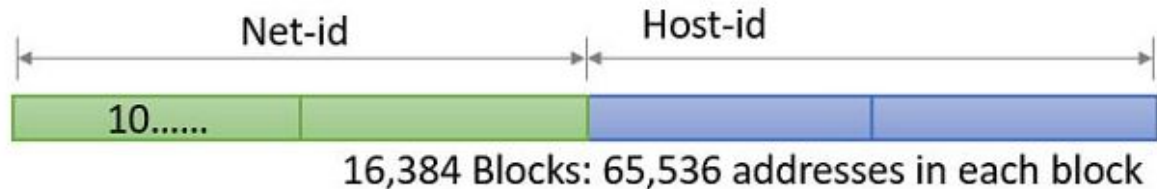
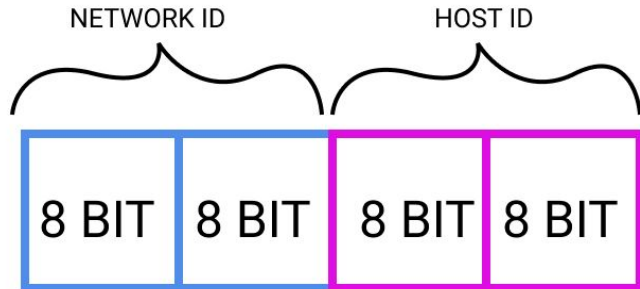
# Class B

Class B

128-191

- The **network id** or the net-id of **class B** is defined using the **first two bytes** of the IPv4 address.
- The first **two bits** of **net-id** stays '10' to define that the IPv4 address belongs to the **class B** and the remaining **14 bits** of net-id can be changed to calculate the number of **blocks** in class B i.e.  $2^{14} = 16,384$ .

The **next two bytes** to of IPv4 address denotes the **host id** in class B which is **16 bits**. The number of hosts can be calculated as  $2^{16} = 65,536$ . So, we conclude that we can assign 16,384 blocks from class B to 16,384 organizations where each organization can have 65,536 hosts connected to the network.



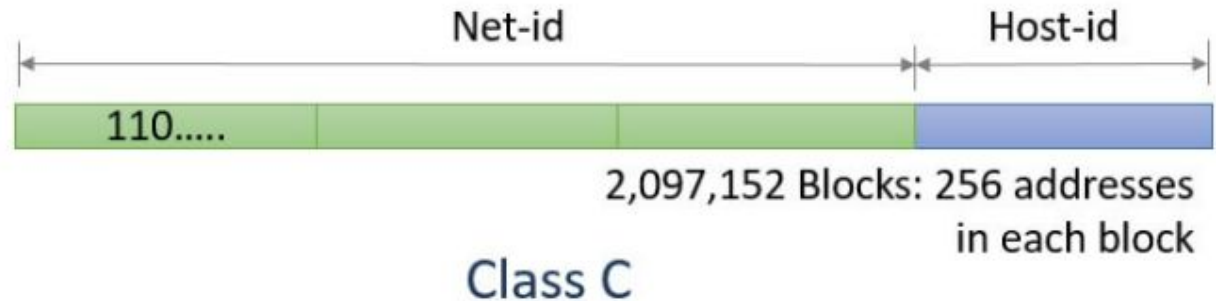
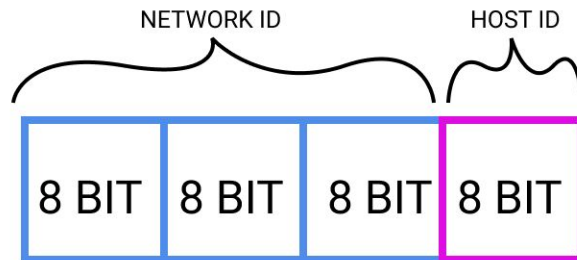
Class B

# Class C

Class C

192-223

- In class C the **network id** is defined by the **first 3 bytes** of the IPv4 address.
- The **first 3 bits** in **network id** stay '110' to define the **class** and the remaining **21 bits** defines the number of **blocks** in class B. The number of blocks can be calculated as  $2^{21} = 20,97,152 \approx 21$  lakhs.
- The **last byte** of the IPv4 address in class C defines the **host-id**. The **number of hosts** can be calculated as  $2^8 = 256$ . So, we conclude that we can assign 2,097,152 blocks from class C to 2,097,152 organizations where each organization can have 256 hosts connected to the network.

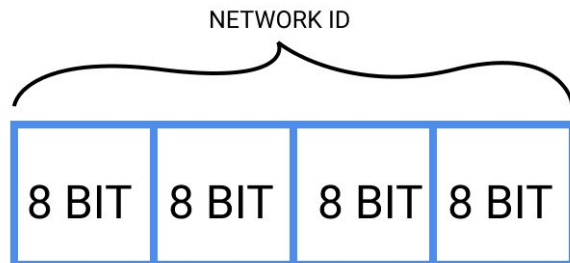


# Class D

Class D

224-239

- Like class A, B & C, class D does **not divide** IPv4 into **net-id** and **host-id**. In Class D, there are no hosts or networks.
- **All the addresses** of class D are of **one single block**.
- The class D addresses are designed for **multicasting**. The **first four-bit** of entire 32-bit addresses of **class D** stays '**1110**' to define the class.
- The remaining **28** bits from the 32-bit addresses of class D can be changed to define the **address space** of class D. So, the number of addresses in class D is  $2^{28}=26,84,35,456$ .



1 Block: 268,435,456 addresses

Class D

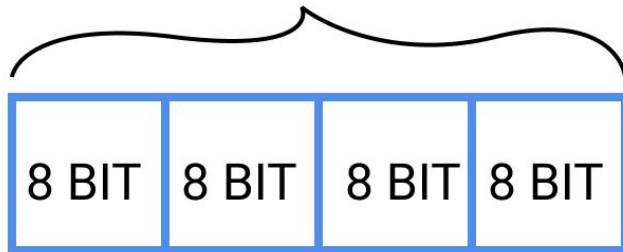
# Class E

Class E

240-255

- Like class D, Class E addresses are one block addresses.
- The addresses in class E are not split into net-id and host-id.
- The addresses in class E are **reserved for future** use.
- The **first four bits** of entire 32-bit IPv4 addresses of class E stays '1111'.
- The remaining **28-bit** changes to define the number of addresses in **class E** i.e.  $2^{28} = 268,435,456$ .

NETWORK ID



1 Block: 268,435,456 addresses

Class E

	First byte	Second byte	Third byte	Fourth byte
Class A	0			
Class B	10			
Class C	110			
Class D	1110			
Class E	1111			

a. Binary notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0-127			
Class B	128-191			
Class C	192-223			
Class D	224-239			
Class E	240-255			

b. Dotted-decimal notation

Find the class of each address.

- a. 00000001 00001011 00001011 11101111
- b. 11000001 10000011 00011011 11111111
- c. 14.23.120.8
- d. 252.5.15.111

Solution

- a. The first bit is 0. This is a class A address.
- b. The first 2 bits are 1; the third bit is 0. This is a class C address.
- c. The first byte is 14 (between 0 and 127); the class is A.
- d. The first byte is 252 (between 240 and 255); the class is E.



# Classes & Blocks



*Number of blocks and block size in classful IPv4 addressing*

<i>Class</i>	<i>Number of Blocks</i>	<i>Block Size</i>	<i>Application</i>
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

One Problem of classful addressing is that each class is divided into fixed number of blocks, with each block having a fixed size as shown in figure.

When an Organization requests for a block of addresses, it was granted one of the classes A, B or C. Class A addresses were designed for large organizations with large no. of hosts (more than one crore) in each block.

Class B addresses were designed for mid size organizations, with more than 65 thousand attached hosts. Class C were designed for small organizations with a small number of attached hosts.

A block in Class A address is too large for almost any organization. That means most of addresses in class A are wasted (ie not used). Similarly Class B is also too large for many of the organizations. But Class C is too small for many organizations. Class D addresses were designed for multi casting. Each address in this class is used to define one group of hosts on the Internet. The Internet designing authorities wrongly predicted the need for 26,84,35,456 (>26 crore) multicast addresses, which too were wasted. Finally Class E were reserved for future use and only a few were used, resulting another wastage of addresses.

## Disadvantages of Classful Addressing

1. If we consider class A, the number of addresses in each block is **more than enough** for almost any organization. So, it results in wastage of addresses.
2. Same is the case with class B, probably an organization receiving block from class B would not require that much of addresses. So, it also results in **wastage of addresses**.
3. A block in class C may be **too small** to **fulfil the addresses requirement** of an organization.
4. Each address in class D defines a group of hosts. Hosts need to **multicast** the address. So, the addresses are wasted here too.
5. Addresses of class E are **reserved for the future purpose** which is also wastage of addresses.
6. The main issue here is; we are **not assigning** addresses according to **user requirements**. We directly assign a **block of a fixed size** which has a **fixed number of addresses** which leads to wastage of address.

To overcome the flaws of classful addressing, these two solutions were introduced to compensate for the wastage of addresses. Let us discuss them one by one.

## **Subnetting :**

The class blocks of A & B are too large for any organization. So, the organization can divide their large network into the smaller subnetwork and share them with other organizations. This concept is known as subnetting. Subnetting is the strategy used to partition a single physical network into more than one smaller logical sub-networks (subnets). A subnet, or subnetwork, is a network inside a network. Subnets make networks more efficient. Subnets were designed for solving the shortage of IP addresses over the Internet.

## **Supernetting :**

As the blocks in class A and B were almost consumed, the new organizations consider class C. But, the block of class C is too small than the requirement of the organization. In this case, the solution which came out is supernetting which grants to join the blocks of class C to form a larger block which satisfies the address requirement of the organization.

# Classless Addressing in IPv4



**Classless addressing** is a concept of addressing the IPv4 addresses. It was adopted after the failure of classful addressing. The classful addressing leads to wastage of addresses as it assigns a fixed-size block of addresses to the customer. But, the classless addressing assigns a block of addresses to the customer according to its requirement which **prevents the wastage of addresses**.

The classless IPv4 addressing **does not divide** the address space into classes like classful addressing. It provides a **variable-length of blocks**, which have a range of addresses according to the need of users.

- Classless Addressing is an improved IP Addressing system.
- It makes the allocation of IP Addresses more efficient.
- It replaces the older classful addressing system based on classes.
- It is also known as **Classless Inter Domain Routing (CIDR)**.

**CIDR Block-** When a user asks for specific number of IP Addresses,

- CIDR dynamically assigns a block of IP Addresses based on certain rules.
- CIDR provides the flexibility of borrowing bits of Host part of the IP address and using them as Network
- This block contains the required number of IP Addresses as demanded by the user.
- This block of IP Addresses is called as a **CIDR block**.

# Network Address Translation



Any computer or network device requires an IP address to connect to the Internet. If your company has 20 devices, then 20 IP addresses will be required to connect to all these devices to internet. But using NAT, you can connect all network devices to Internetwork with only one Registered Public IP Address.

As we know that the Internet is growing at a very high speed and the number of IP addresses is becoming less than necessary, so **IPv6** is being developed to deal with this problem, but it will take time to be implemented. In such a Situation, NAT (Network Address Translation) is being used to fulfill the requirement of IP address.

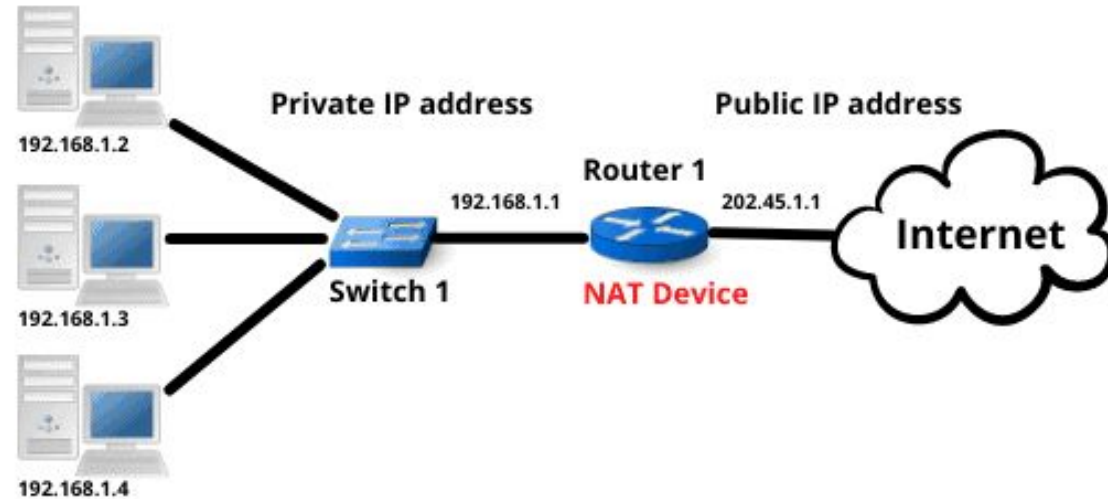
## Two different types of IPv4 addresses -

**Public IP :** These are IP addresses that are publicly registered on Internet. In order to get Internet connection you need a Public IP.

**Private IP :** Private IP addresses are not publicly registered in the Internet and hence we cannot directly get internet access using only Private IP address. These are used internally in home or business LANs. These IPs can be assigned manually or dynamically using routers.

Network Address Translation (NAT) is a process that enables one unique IP address to represent an entire group of computers. In network address translation, a network device, often a router or NAT firewall, assigns a computer or computers inside a private network a public address. In this way, NAT allows the single device, such as router to act as an intermediary or agent between the local, private network and the public network that is the internet. The private addresses will be translated to public & viceversa by NAT in the Router. NAT's main purpose is to conserve the number of public IP addresses in use, for both security and economic goals.

## Network Address Translation



When a host on the internal network with an internal IP address needs to access Internet, it would use the public IP address on the network's gateway(i.e. router) to identify itself to the rest of the world, and this translation of converting a private IP address to public is done by NAT.

For example a computer on an internal address of 192.168.1.3 wanted to communicate with a web server somewhere on the internet, NAT would translate the address 192.168.1.3 to the organization's public address i.e 202.45.1.1, so that the internal address is identified as the public address when communicating with the outside world.

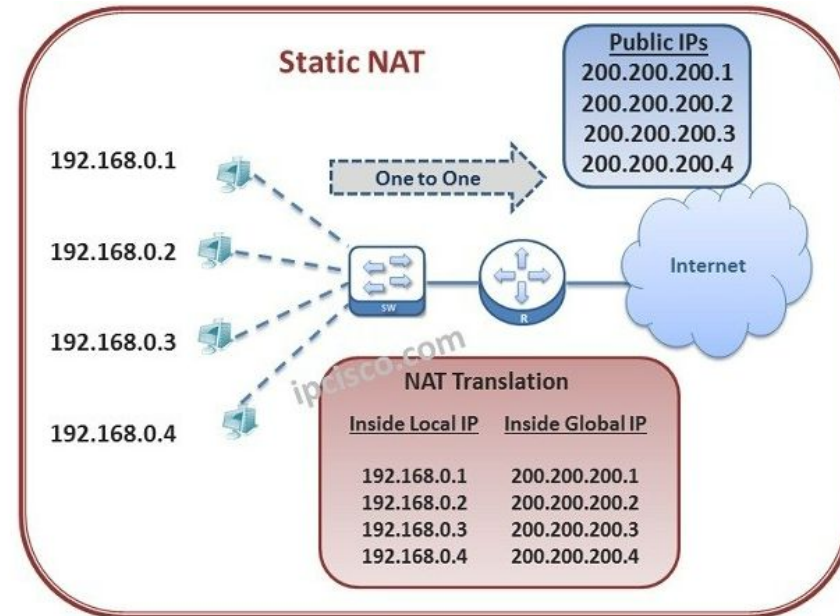
All outgoing packets go through the NAT router, which replaces source address in the packet with the global NAT address. Now, the web server would reply to the public address, 202.45.1.1. All incoming packets also pass through NAT router. NAT would then use its records to translate the packets received from the web server, back to the internal network address of 192.168.1.3, and thus the computer who requested the information, will receive the requested packets.

## Benefits of NAT :

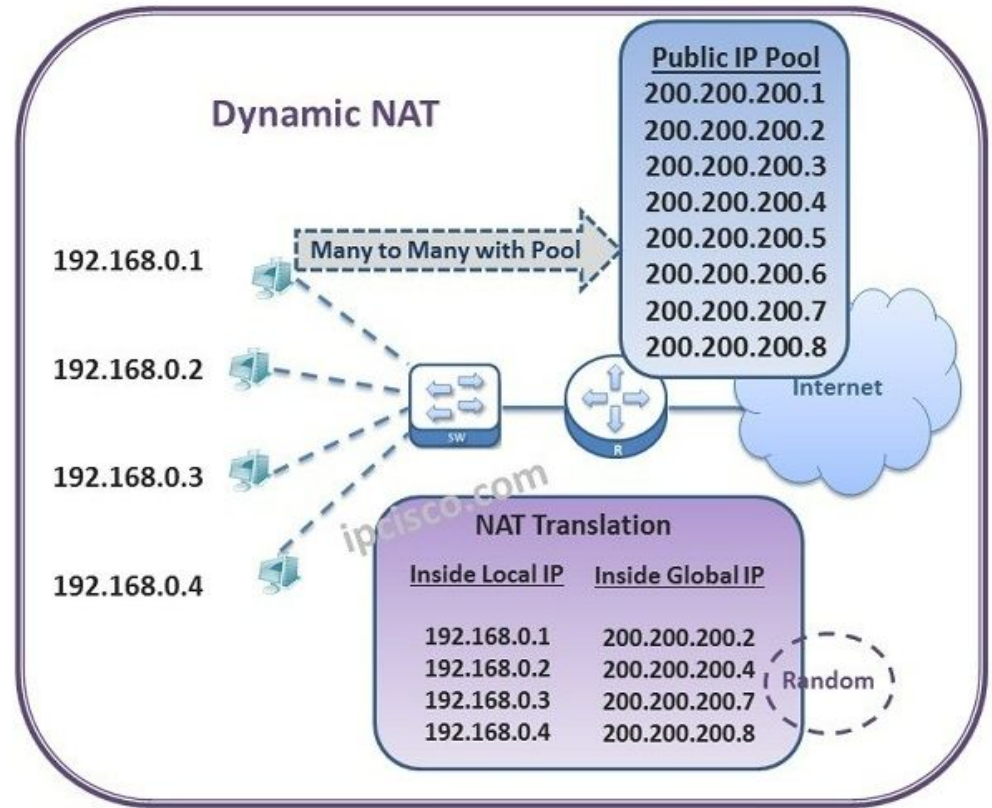
The two benefits of NAT are, firstly it would save the Public IP addresses , since every single computer does not need a public address and the next is the security factor, i.e. it would hide these private computers from the outside world. Everyone can only see the public address, the rest is hidden behind this public address.

## Types of NAT

1. **Static NAT** – This type of network translation is designed to allow one-to-one mapping between local and Global Addresses. That means, a single unregistered/ Private IP address is mapped with a legally registered/ Public IP address. For eg: If there are 300 devices in an organization, which requires internet access, this type of NAT requires to purchase 300 public IPs, which will be very costly. Static NAT are generally used for Web hosting. As shown in figure, using Static NAT, the computer with the IP address of 192.168.0.3 will always translate to 200.200.200.3



**2. Dynamic NAT** – Dynamic NAT is one of the NAT types that is used with a Public IP Address Pool. Here, Instead of choosing the same IP address every time, multiple Private IP Addresses are mapped to a Pool of Public IP Addresses. These Public IP Addresses are given to the Internal users randomly and hence it is difficult to reach any Internal user from outside. For eg: In dynamic NAT, the computer with the IP address 192.168.0.4 will translate to the first available public address in the range from 200.200.200.1 to 200.200.200.8



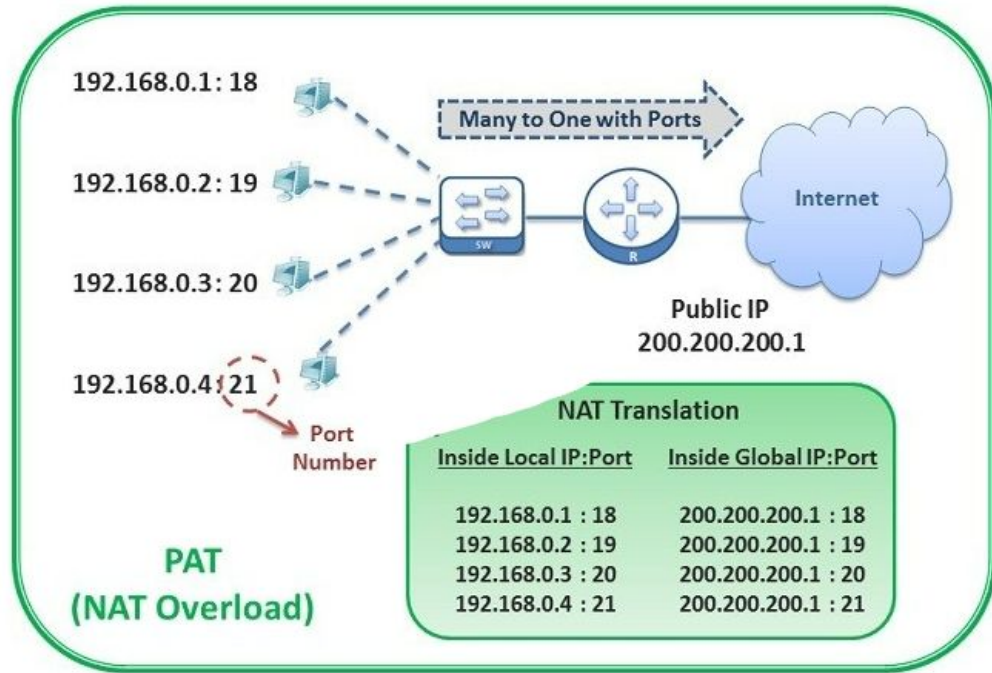


### 3. PAT (NAT Overload)

**PAT (Port Address Translation)** is one of the **NAT types** that is also known as **NAT Overload**. Here, many Private IP Addresses are translated to one Public IP Address.

Port numbers are used to distinguish the traffic. Each IP Address's traffic is determined by these ports. This is most frequently used type of NAT, as it is cost effective as thousands of users can be connected to internet by using only one public IP address.

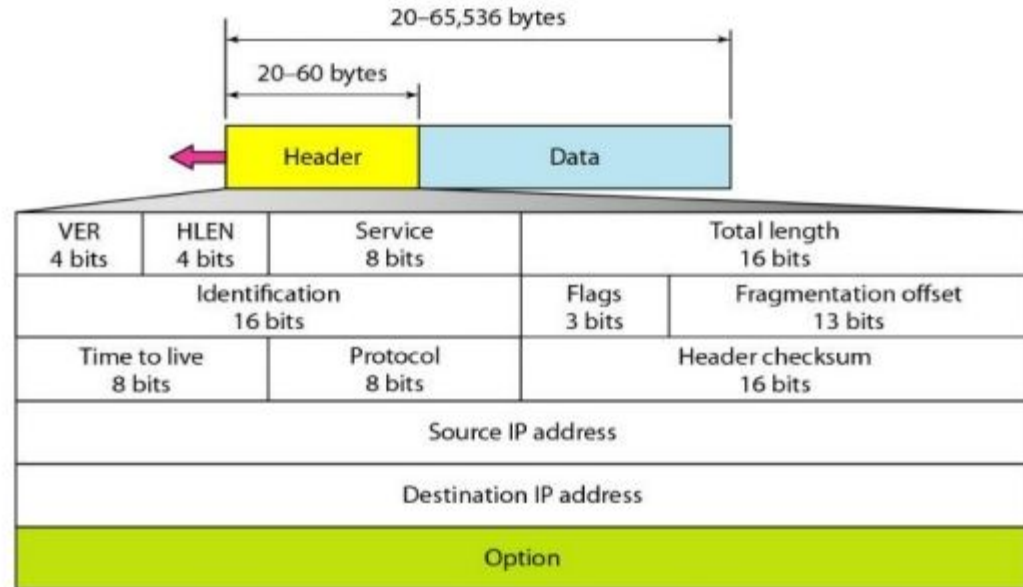
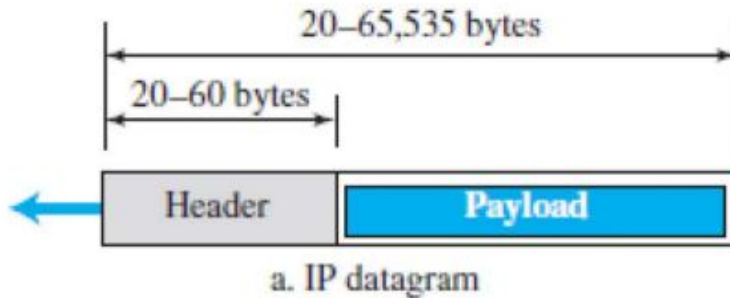
In this **PAT Example**, Private IP Addresses are translated to a specific Public IP Address with the help of Port Numbers. For example, 192.168.0.4:21 is translated to 200.200.200.1:21.



# IPv4 Datagram Format



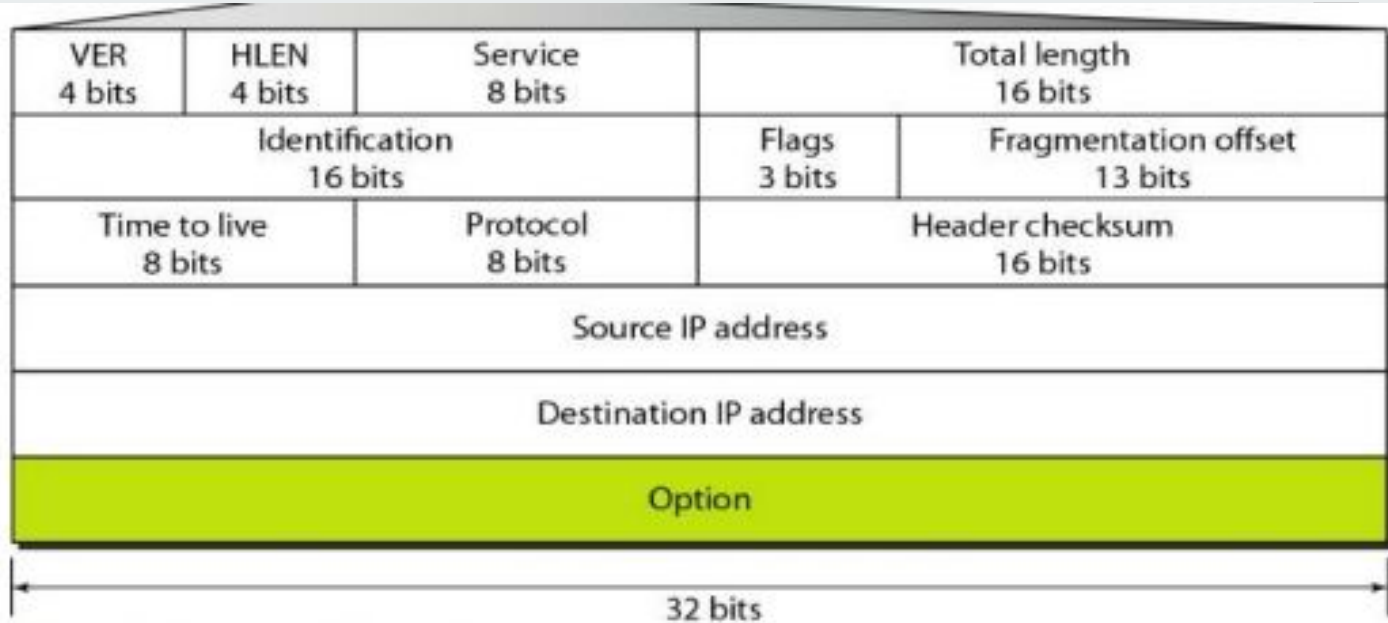
- The **IPv4** (Internet Protocol version 4) is the delivery mechanism used by the TCP/IP protocols.
- A TCP/IP Network-layer packet is referred to as a datagram.
- IPv4 is a connectionless datagram protocol. i.e. Each datagram is handled independently. Hence each datagram from the same source can follow a different route to the same destination and arrive out of order.
- **A datagram is a variable length packet consisting of two parts: header and payload (data).**
- A datagram can carry up to 65,535 bytes of data. The header is 20 to 60 bytes in length and contains information essential to routing and delivery.
- The first 20 bytes in the header are mandatory for all IPv4 datagrams. Hence, the minimum size of an IPv4 header is 20 bytes.



## IPv4 datagram format

### 1. VER (Version Number) :

This 4-bit field defines the version of the IPv4 protocol. Currently, the version is 4. However, IP version 6 may totally replace version 4 in the future.



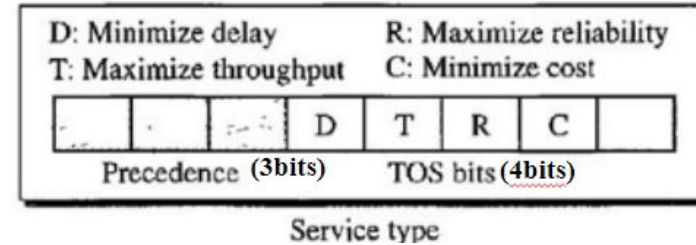
**2. HLEN (Header Length) :** Header length is a variable between 20 and 60 bytes. HLEN is the number of 32-bit words that make up the header field. The minimum available number is 5 ( $5 \times 4 = 20$ ), as the first 20 bytes are mandatory. When there are no options, header length = 20 bytes. When options field is at its maximum size, value in this field is 15.

*I.e. header length can be calculated by the formula : Header length = Header length field value  $\times$  4 bytes*

*Eg: If header length field contains decimal value 11 (represented in binary 1011) then Header length =  $11 \times 4 = 44$  bytes*

### 3. Service Type:

- It is an 8 bit field that is used to define the Quality of Service.
- The first 3 bits are called **Precedence bits**, and the next 4 bits are called **Type Of Service (TOS) bits**
- The last bit is reserved (usually set as 0).
- Precedence is a 3-bit sub-field ranging from 000 to 111 (ie 0-7). It defines the priority of the datagram in issues such as congestion. If a router is congested, the datagrams with lowest precedence will be discarded first.
- TOS is a 4-bit subfield with each bit having special meaning. They are DTRC (Delay, Throughput, Reliability, Cost (distance)). Each bit can be either 0 or 1, and **only one of the bits can have a value 1 in each datagram**. Bit patterns and their meaning is shown in the figure.
- This field previously called Service Type is now known as Differentiated Services.



TOS Bits	Description
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

**4. Total Length:** This 16-bit field defines the total length of the IP datagram in bytes. Total length = Header length + Payload length. A 16-bit number can define a total length of up to 65,535 (when all bits are 1s).

**5. Identification 6. Fragmentation Offset and 7. Flags :** These three fields are related to the fragmentation of the IP datagram when the size of the datagram is larger than the underlying network can carry .

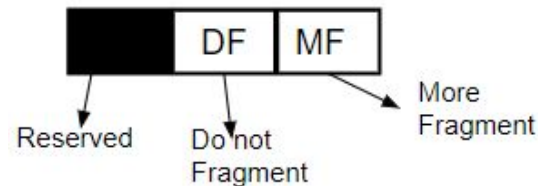
**Identification** is a 16-bit field. It is helpful for the identification of the fragments of an original datagram. When an IP datagram is fragmented, each fragmented datagram is assigned the same identification header number. This header-number is useful during the reassembly of fragmented datagrams.

**Fragmentation Offset:** It is a 13 bit field. It tells the relative position of a fragmented datagram in the original un-fragmented IP datagram.

**Flags** - 3 flags of 1 bit each is used.

**First flag bit** is Reserved. **Second Flag bit** -DF (Do Not Fragment) Bit - value may be 0 or 1; 0 means intermediate devices such as routers can fragment the datagram if required & 1 means datagrams should not be fragmented.

**Third flag bit** - MF(More Fragments) - values may be 0 or 1. MF=0, tells the receiver that the current datagram-fragment is the last fragment of the same datagram. MF =1, tells more fragments are still to come after this fragment. So MF bit is set to 1 for all the fragments except the last one.



**8. Time to live:** Datagram's lifetime . The time-to-live (TTL) field is an 8-bit field included to ensure that datagrams do not circulate forever in the network. It prevents the datagram to loop in the network. This field is decremented by one each time the datagram is processed by a router. If the TTL field reaches 0, the datagram must be dropped.

**9. Protocol:** it is an 8-bit number that defines the protocol in the transport layer, to which the data is to be passed. This field is used only when an IP datagram reaches its final destination. The value of this field indicates the specific transport-layer protocol to which the data portion of this IP datagram should be passed. For example, Protocol no. 6 indicates that the data portion is passed to TCP, while a value of 17 indicates that the data is passed to UDP.

**10. Header Checksum :** This is a 16 bit field used to keep the checksum value of the entire header for checking errors in the datagram header. The header checksum helps a router in detecting bit errors in the header portion of the received IP datagram.

**11. Source IP address:** This 32-bit field defines the IPv4 address of the source. This field must remain unchanged during the time the IPv4 datagram travels from the source host to the destination host.

**12. Destination IP address:** 32 bits IP address of the receiver

**13. Options :** This is an optional field and is rarely used. Additional options such as security, timestamp, route etc. are stored in this field when present. Due to this field, datagram-header- size can be of variable length (20 bytes to 60 bytes).

# IPv6 Addresses



The network layer protocol in the TCP/IP protocol suite is currently IPv4. Although IPv4 is well designed, data communication has evolved since the formation of IPv4 in the 1970s. The numbers of users of the internet are increasing day by day and the services offered to these users are also increasing. IPv4 has some deficiencies (listed below) that make it unsuitable for the fast-growing Internet.

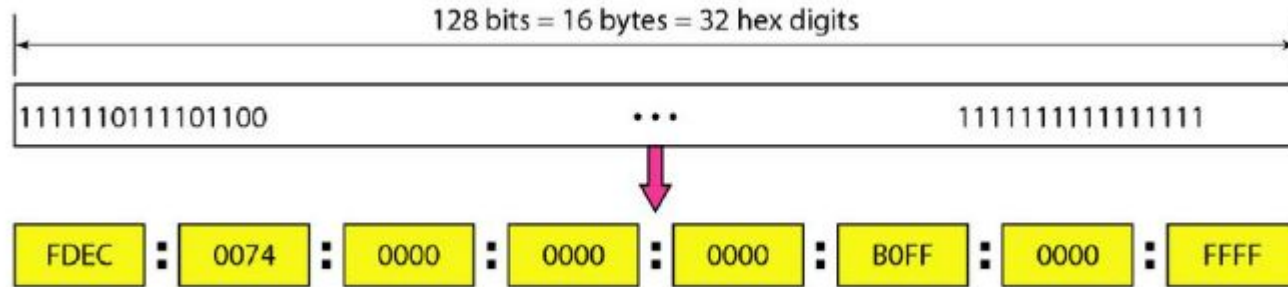
- A huge number of addresses are needed to accommodate such a vast number of users. Despite all short-term solutions, such as subnetting, classless addressing, and NAT, **address depletion** is still a long-term problem in the Internet.
- The Internet must accommodate **real-time audio and video transmission**. This type of transmission requires minimum delay strategies and reservation of resources that are not provided in the IPv4 design.
- The Internet must accommodate **encryption and authentication of data**. No encryption or authentication is provided by IPv4.

IPv6 was developed to overcome the shortcomings of IPv4. The problem of address depletion, lack of accommodation for real-time audio and video transmission, and encryption and authentication of data for some applications motivated the development of IPv6.

IPv6 (Internetworking Protocol, version 6) is also known as IPng (Internetworking Protocol, next generation)

## Hexadecimal Colon Notation

- An IPv6 address is made up of 128 bits. Therefore Address space is  $2^{128}$ .
- 128 bits are divided into eight blocks.
- Each block is 2 Bytes = 16 bits. I.e. An IPv6 address is made of 128 bits divided into eight 16-bit blocks.
- Each block is converted into 4-digit hexadecimal number separated by colon symbols.
- An IPv6 address example: **2001:0000:3238:DFE1:0063:0000:0000:FEFB**



*The dotted decimal notation used for IPv4 does not make IPv6 addresses sufficiently compact*

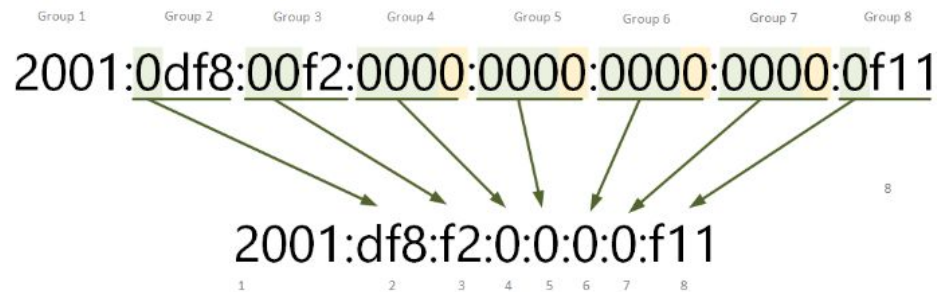
104.230.140.100.255.255.255.255.0.0.17.128.150.10.255.255



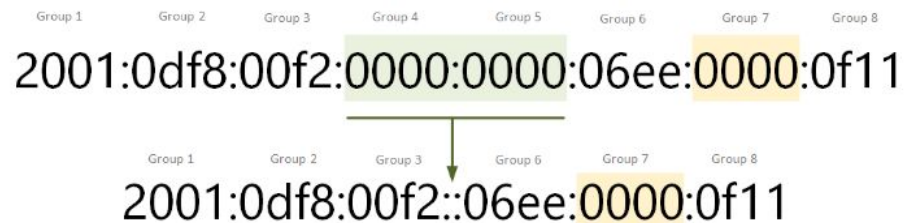
# Shortening IPv6 addresses

IPv6 provides two rules to shorten the address. The rules are as follows:≡

**Rule 1: Omit Leading zeros** - Omit leading zeros in any group of 4 hexadecimal digits. The rule applies only to leading zeros and no trailing zeros. Even if a group consists of 4 zeros 0000 we can only omit the leading 3 - 0000.

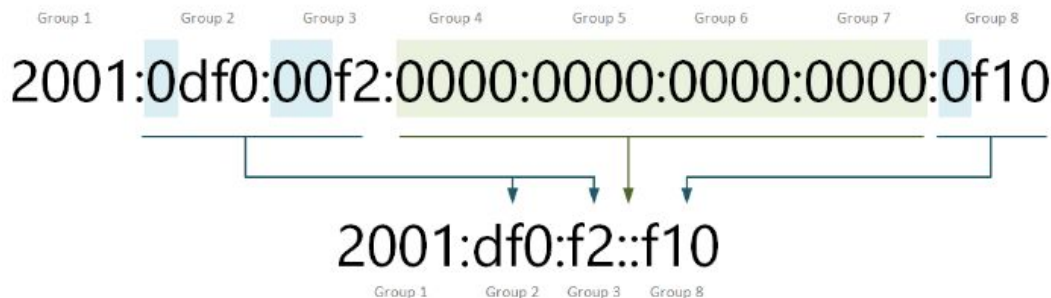


**Rule 2: Omit groups of all zeros** : This says that a double colon (::) can replace a single, contiguous string of one or more groups consisting of all 0s. This zero compression with a double colon can be applied only once!



## Combining Rule 1 and 2

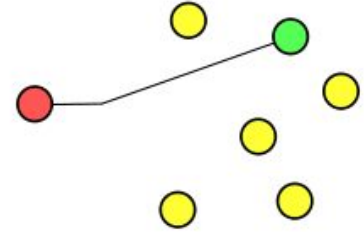
The shortest possible representation is achieved by combining both rules



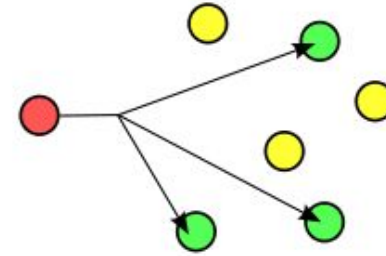
# Types of IPv6 Addresses

The three types of IPv6 addresses are: unicast, multicast and anycast.

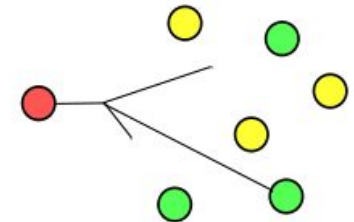
**Unicast** : A unicast address identifies a single interface or computer. When a network device sends a packet to a unicast address, the **packet goes only to the specific recipient** identified by that address.



**Multicast** : A **multicast address** can be used to deliver a package to a **group of destinations**. Any packet sent to a multicast address, will be delivered to every host that has joined that particular group.

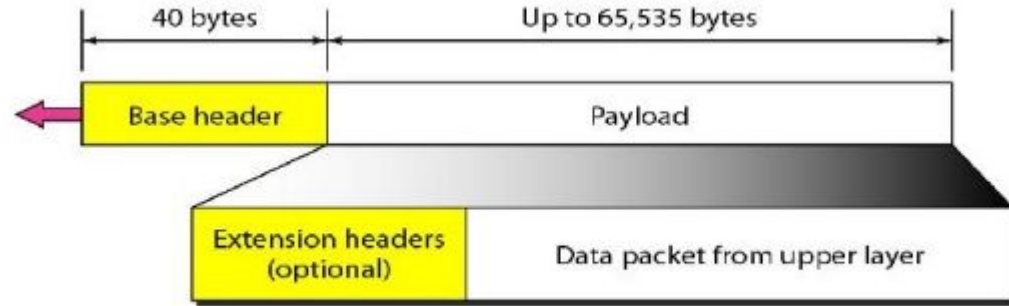


**Anycast** : The **anycast address** is very similar to the multicast address, but packets will be delivered to **only one member of the group which is the most nearest host**, instead of the entire group.



IPv6 has **no support for the broadcast address**, any function that used to rely on broadcasts will be considered as a special case of multicasting.

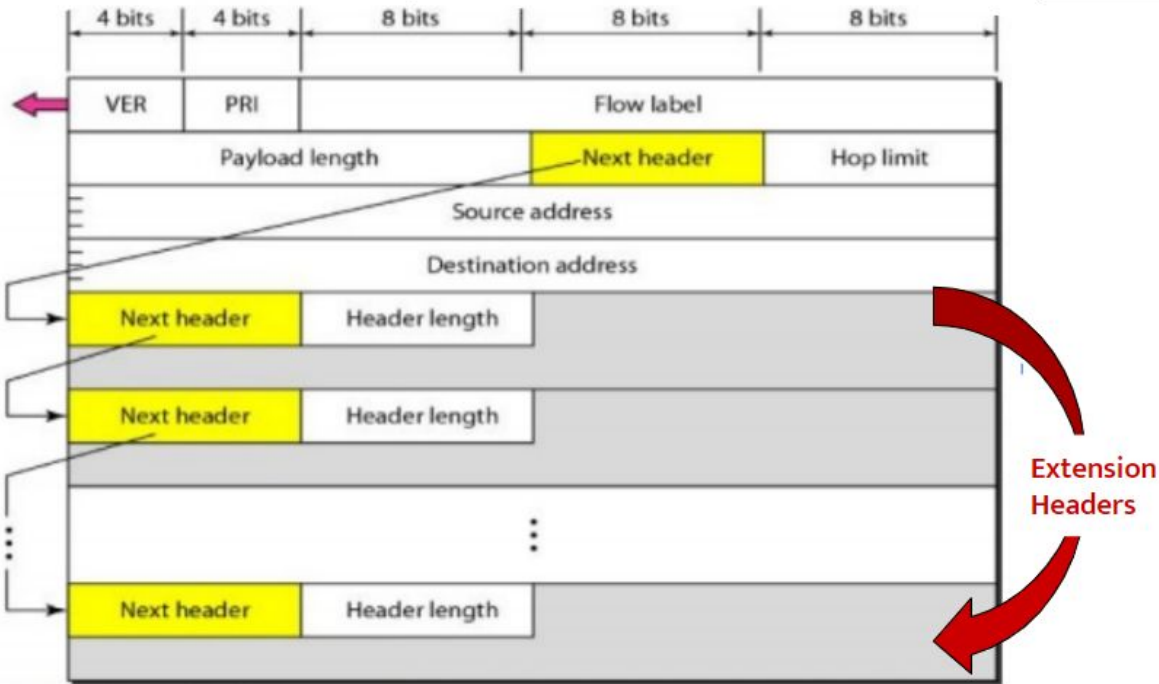
# IPv6 - Datagram Format



**IPv6 datagram header and payload**

- The IPv6 packet is shown in Figure.
- Each packet is composed of a mandatory base header followed by the payload.
- The payload consists of two parts: optional extension headers and data from an upper layer.
- The base header occupies 40 bytes, whereas the extension headers and data from the upper layer can contain up to 65,535 bytes of information.

# IPv6 Datagram Format



Format of an IPv6 datagram

The base header has eight fields. These fields are as follows:

1. **Version** : This 4-bit field defines the version number of the IP. For IPv6, the value is 6 (0110).
2. **Priority** : The 4-bit priority field defines the priority of each packet. For example, if one of two consecutive datagrams must be discarded due to congestion, the datagram with the lower packet priority will be discarded. It helps routers to handle the traffic based on the priority of the packet. Congestion-controlled data are assigned priorities from 0 to 15, 0 being the lowest & 15 being the highest

**3. Flow label :** The flow label is a 3-byte (24-bit) field that is designed to provide special handling for a particular flow of data. A sequence of packets (such as real-time audio/ video streaming), sent from a particular source to a particular destination, that needs special handling by routers is called a flow of packets. To a router, a flow is a sequence of packets that share the same characteristics, such as traveling the same path, using the same resources, having the same kind of security, and so on.

A router that supports the handling of flow labels has a **flow label table**. The table has an entry for each active flow label; each entry defines the services required by the corresponding flow label. When the router receives a packet, it consults its flow label table to find the corresponding entry for the flow label value defined in the packet. It then provides the packet with the services mentioned in the entry. **Thus a flow label can be used to speed up the processing of a packet by a router.** When a router receives a packet, instead of consulting the routing table and going through a routing algorithm to define the address of the next hop, it can easily look in a flow label table for the next hop. Thus real-time data will not be delayed due to a lack of resources.

**4. Payload length :** The 2-byte (16-bit) payload length field defines the length of the IP datagram excluding the base header.

**5. Next header :** This is an 8 bit field, used to indicate either the type of Extension Header, or if the Extension Header is not present then it indicates the protocols present inside the upper layer packet. i.e. It identifies the protocol to which the payload data of this datagram will be delivered - UDP (17) and TCP (6). Note that this field uses the same values as the protocol field in the IPv4 header.

## Extension Headers

When the sender wants to add additional functionalities to the packets being sent, Extension headers are used. For example if security encapsulation option is to be added, the value 50 is given in the Next Header field. If there is one more Extension Header, then the first Extension Header's 'Next-Header' field points to the second one, and so on.

IPv6 extension headers are not examined or processed by any router along a packet's delivery path until the packet arrives at its final destination. This feature is a major improvement in router performance for packets that contain options. In IPv4, the presence of any options requires the router to examine all options.

### *Values of the Next Header Field*

Value	Header
0	Hop-by-Hop Options Header
6	TCP
17	UDP
41	Encapsulated IPv6 Header
43	Routing Header
44	Fragment Header
50	Encapsulating Security Payload
51	Authentication Header
58	ICMPv6
59	No next header
60	Destination Options Header

**6. Hop limit :** This 8-bit hop limit field serves the same purpose as the TTL field in IPv4. This field makes sure that the packet does not go into an infinite loop. Every time the packet passes through a router, this field is decremented by 1 and finally package is discarded. It allows a maximum of 255 hops between the nodes, and anything after that will get discarded.

**7. Source address :** The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram.

**8. Destination address :** The destination address field is a 16-byte (128-bit) Internet address that usually identifies the final destination of the datagram. However, if source routing is used, this field contains the address of the next router.

The internet protocol version 6 has been designed for the future needs of the internet. IPv6 has following new features to make the network ready for next generation of the internet.

## Advantages

The next-generation IP, or IPv6, has some advantages over IPv4 that can be summarized as follows:

- **Larger address space.** An IPv6 address is 128 bits long and therefore it provides  $2^{128}$  unique addresses. Compared with the 32-bit address of IPv4, this is a huge increase ( $2^{96}$ ) in the address space.
- **Better header format.** IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the upper-layer data. This simplifies and speeds up the routing process because most of the options do not need to be checked by routers.
- **New options.** IPv6 has new options to allow for additional functionalities.
- **Allowance for extension.** IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.
- **Support for resource allocation.** In IPv6, the type-of-service field has been removed, but a mechanism (called flow label) has been added to support traffic such as real-time audio and video.
- **Support for more security.** The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.



## Difference between IPv4 & IPv6.

Sr.no	IPv4	IPv6
1	IPv4 has 32-bit address length.	IPv6 has 128-bit address length.
2	IPv4 has header of 20-60 bytes.	IPv6 has header of 40 bytes fixed.
3	Address representation of IPv4 in decimal.	Address representation of IPv6 in hexadecimal.
4	Fragmentation performed by Sender and forwarding routers.	In IPv6 fragmentation performed only by sender.
5	In IPv4 Encryption and Authentication facility not provided.	In IPv6 Encryption and Authentication are provided.

# Comparison between IPv4 and IPv6 packet headers



1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The TTL field is called hop limit in IPv6.
5. The protocol field is replaced by the next header field.
6. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

# Transition from IPv4 to IPv6

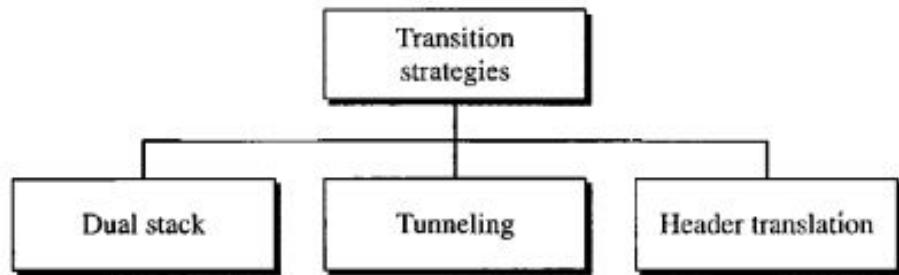


Because of the huge number of systems on the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. IPv6 is not backward compatible with IPv4, since these two are different protocols. It takes a considerable amount of time to move from IPv4 to IPv6. The transition must be smooth to prevent any problems between IPv4 and IPv6 systems.

## Transition Techniques :

The 3 popular transition techniques are

- Dual Stack
- Tunneling
- Header Translation

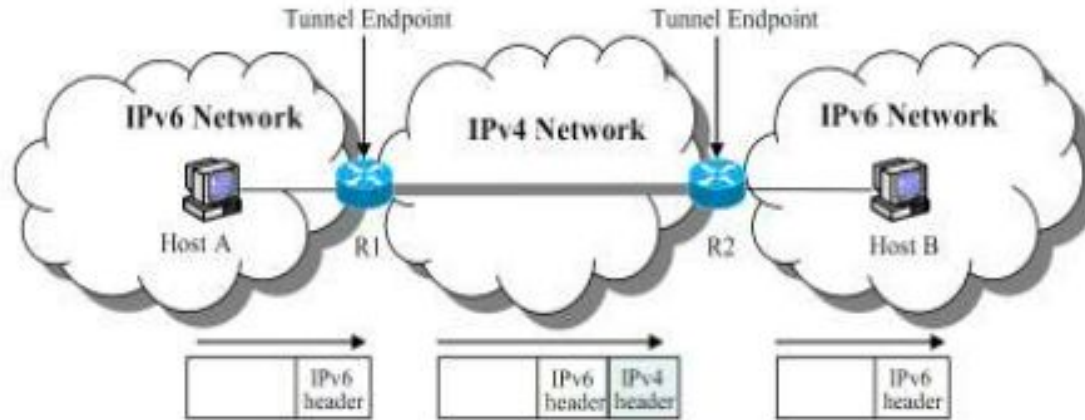


## Dual Stack :

It is recommended that all hosts, before migrating completely to version 6, have a dual stack of protocols. In other words, a host must run IPv4 and IPv6 simultaneously until all the Internet uses IPv6. To determine which version to use when sending a packet to a destination, the source host queries the DNS. If the DNS returns an IPv4 address, the source host sends an IPv4 packet. If the DNS returns an IPv6 address, the source host sends an IPv6 packet.

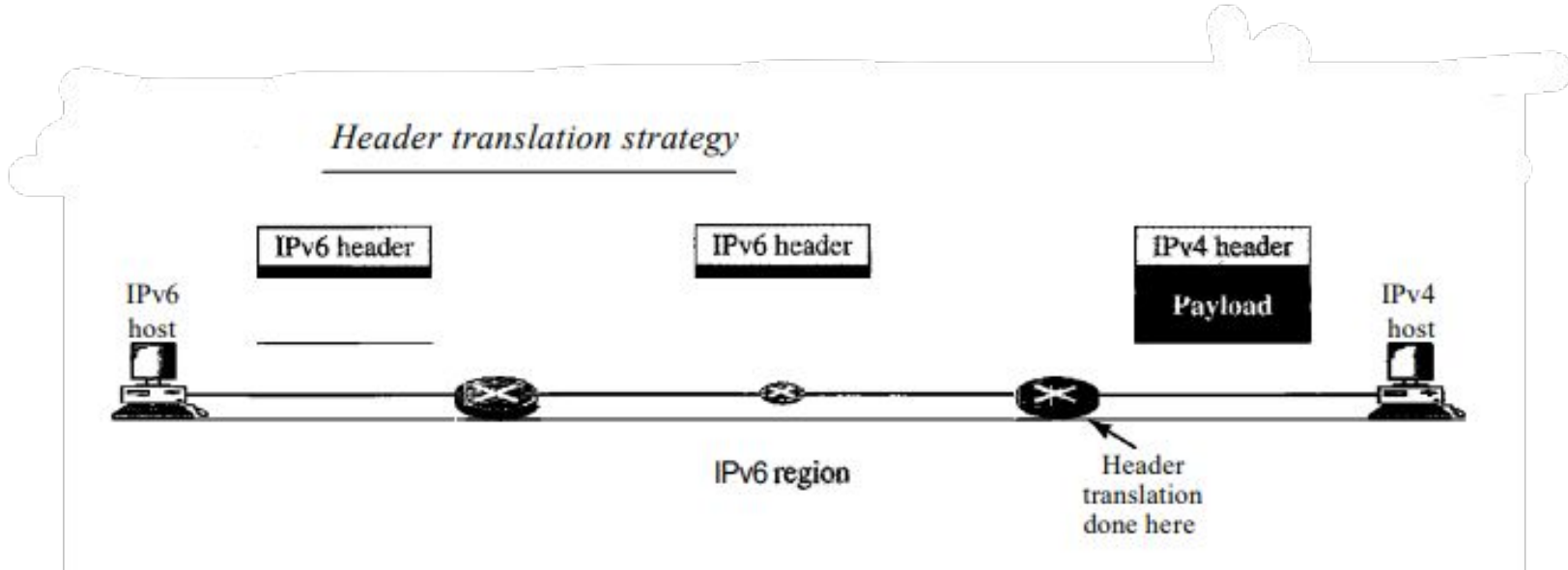
## Tunneling :

Tunneling is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4. ie Two IPv6 nodes want to communicate via IPv4 router. To pass through this region, the packet must have an IPv4 address. So the IPv6 packet is encapsulated in an IPv4 packet, when it enters the region, and it leaves its capsule when it exits the region. It seems as if the IPv6 packet goes through a tunnel at one end and emerges at the other end.



## Header Translation

Header translation is necessary when the majority of the Internet has moved to IPv6 but some systems still use IPv4. The sender wants to use IPv6, but the receiver does not understand IPv6. Tunneling does not work in this situation because the packet must be in the IPv4 format to be understood by the receiver. **In this case, the header format must be totally changed through header translation. The header of the IPv6 packet is converted to an IPv4 header**



# ADDRESS MAPPING



- An internet is made of a combination of physical networks connected by internetworking devices such as routers.
- The hosts and routers are recognized at the network level by their logical addresses or IP address. At the physical level, the hosts and routers are recognized by their physical addresses or MAC address. MAC addresses are usually fabricated in the hardware.
- A packet starting from a source host may pass through several different physical networks before finally reaching the destination host. So, the delivery of a packet to a host or a router requires two levels of addressing : logical and physical.
- We need to be able to map a logical address to its corresponding physical address and vice versa. So **Address mapping is a process of determining a logical address knowing the physical address of the device and determining the physical address by knowing the logical address of the device.**

## Types of Address Mapping -

There are two kinds of address mapping, static address mapping, and dynamic address mapping.

1. **Static mapping** involves in the creation of a table that associates a logical address with a physical address. This table is stored in each machine on the network. But this has some limitations because physical addresses may change in the following ways:
  - If a device changes its Network Interface Card (NIC), the physical address of the device also changes. As the physical address is hardcoded on the NIC card at the time of its manufacturing.
  - Some local networks such as Apple's LocalTalk compel the connected device to change its physical address each time the device turns on.
  - A mobile device can move from one physical network to another which results in change of MAC address.

To implement these changes, a static mapping table must be updated periodically. This overhead could affect the network performance.

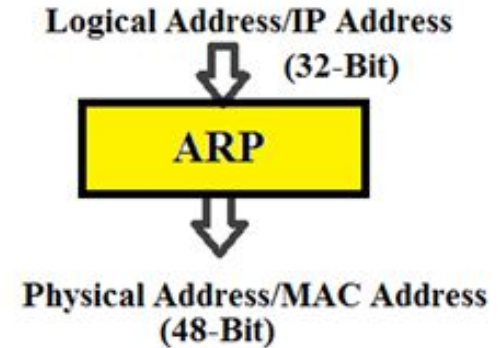
## 2. Dynamic Mapping

The source host knows the logical address of the destination host but to deliver the packet to the destination host, its physical address is required. In dynamic mapping, when a machine knows one of the two addresses (logical or physical), it can use a protocol to find the other one. Two protocols are designed for dynamic mapping - **ARP (Address Resolution Protocol)** and **RARP (Reverse Address Resolution Protocol)**.

### ARP - Mapping Logical to Physical Address:

When the host/router has to send an IP datagram, the IP address of the receiver is obtained from the DNS or routing table. But the IP datagram must be encapsulated in a frame to be able to pass through the physical network. This means that the sender needs the physical address of the receiver. For this, the **source host broadcasts the ARP query packet** to all the hosts in the network.

The ARP query packet contains the logical & physical address of the source and the logical address of the destination. All the hosts present in the network receive this query packet and but only the destination host recognize its logical address and returns an ARP reply packet. **ARP response is unicast.** Other hosts discard the ARP query packet.



## RARP - Mapping Physical to Logical Address:



There are occasions in which a host knows its physical address, but needs to know its logical address.

**Reverse Address Resolution Protocol (RARP)** finds the logical address for a machine that knows only its physical address. RARP's working is similar to the working of ARP, but basically the reverse of the ARP protocol.

The host which requires the logical address, broadcasts the RARP request packet. The RARP request packet contains the physical address of the host. All hosts in the network receive this RARP request packet, but only the one who is the authorized host completes the RARP service. This host sends the RARP response packet to the client in which its logical address is stored.

*RARP is not used nowadays, it was replaced by BOOTP (Bootstrap protocol), and now BOOTP has been replaced by DHCP (dynamic host configuration protocol).*

