Notes for ST-2

Flow Control

BY:

DR. NAVNEET KAUR

Flow Control

Flow control coordinates the amount of data that can be sent before receiving an acknowledgment and is one of the most important duties of the data link layer.

In most protocols, flow control is a set of procedures that tells the sender how much data it can transmit before it must wait for an acknowledgment from the receiver.

The receiving device must be able to inform the sending device before those limits are reached and to request that the transmitting device send fewer frames or stop temporarily.

Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.

Error Control

Error control is both error detection and error correction.

It allows the receiver to inform the sender of any frames lost or damaged in transmission and coordinates the retransmission of those frames by the sender.

Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request (ARQ).

Error control in the data link layer is based on automatic repeat request, which is the retransmission of data.

Elementary Data Link Protocols

Protocols in the **data link layer** are designed so that this layer can perform its basic functions: framing, error control and flow control.

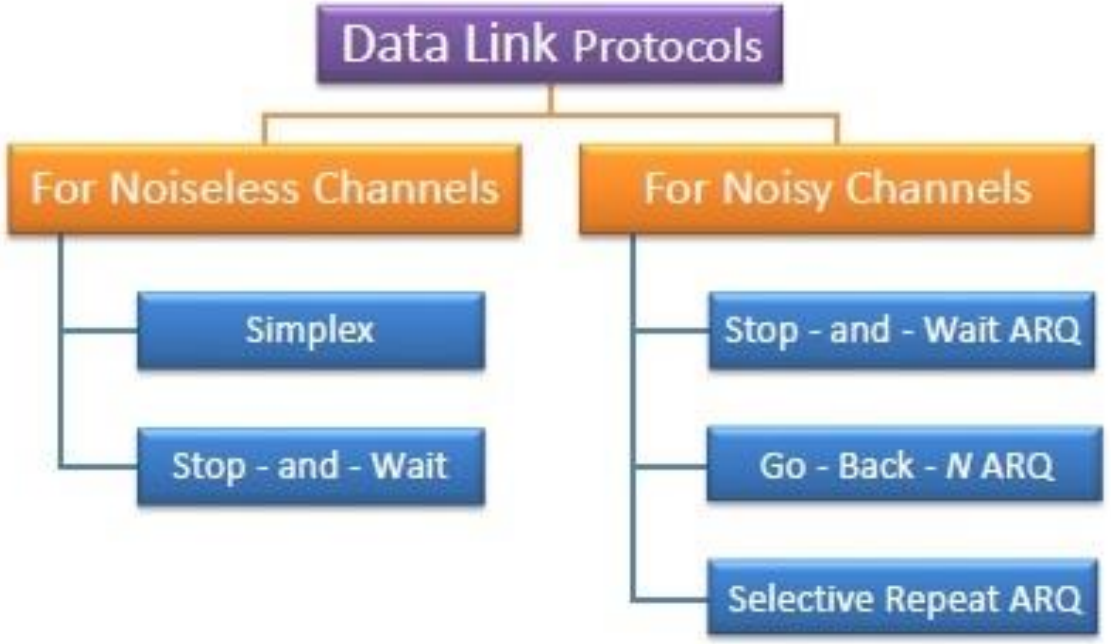
Framing is the process of dividing bit - streams from physical layer into data frames whose size ranges from a few hundred to a few thousand bytes.

Error control mechanisms deals with transmission errors and retransmission of corrupted and lost frames.

Flow control regulates speed of delivery and so that a fast sender does not drown a slow receiver.

Types of Data Link Protocols

Data link protocols can be broadly divided into two categories, depending on whether the transmission channel is noiseless or noisy.



Simplex Protocol

The Simplex protocol is hypothetical protocol designed for unidirectional data transmission over an ideal channel, i.e. a channel through which transmission can never go wrong.

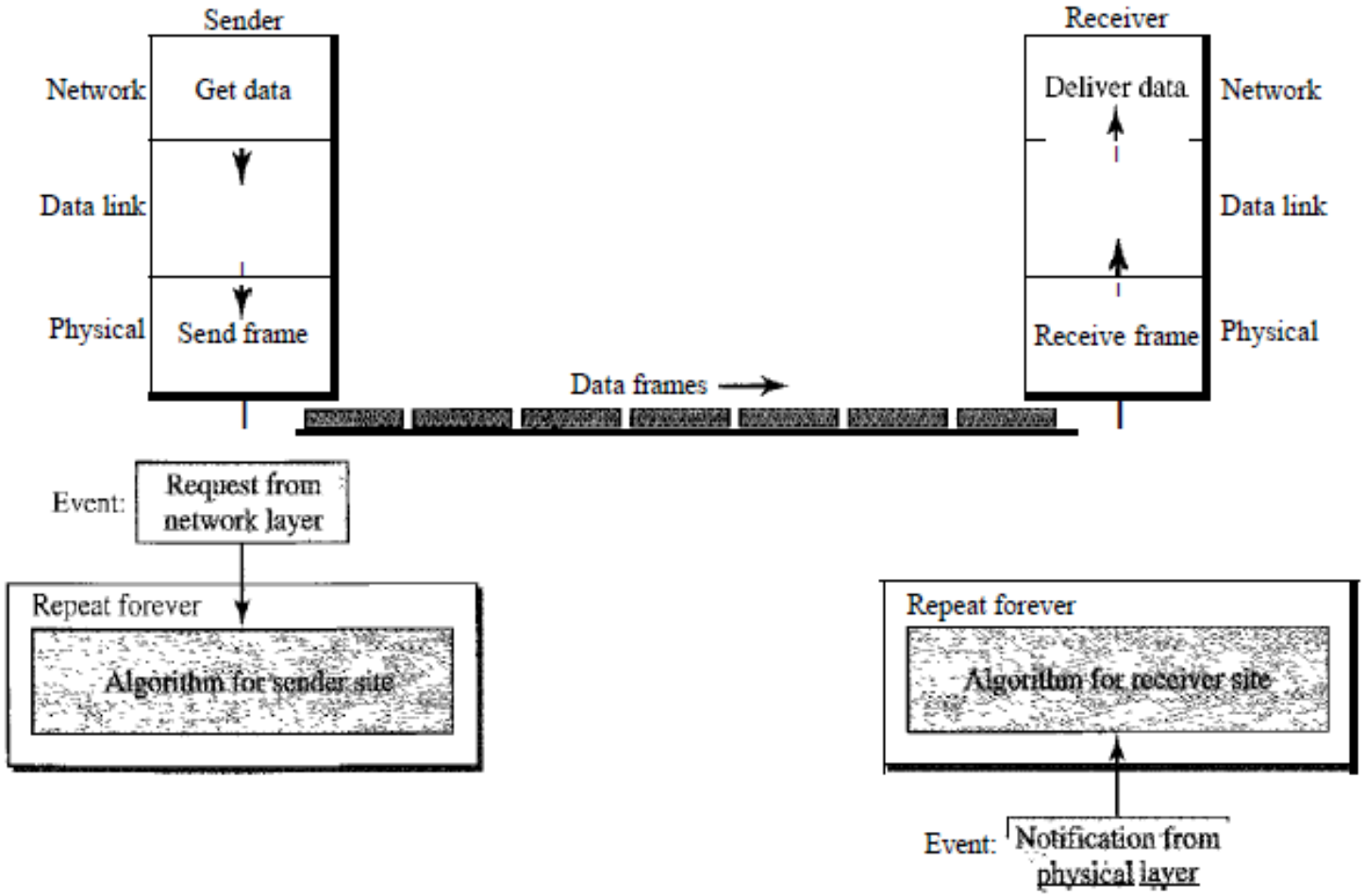
It has distinct procedures for sender and receiver.

The sender simply sends all its data available onto the channel as soon as they are available its buffer.

The receiver is assumed to process all incoming data instantly.

It is hypothetical since it does not handle flow control or error control.

Figure 11.6 *The design of the simplest protocol with no flow or error control*



There is no need for flow control in this scheme.

The data link layer at the sender site gets data from its network layer, makes a frame out of the data, and sends it.

The data link layer at the receiver site receives a frame from its physical layer, extracts data from the frame, and delivers the data to its network layer.

The data link layers of the sender and receiver provide transmission services for their network layers.

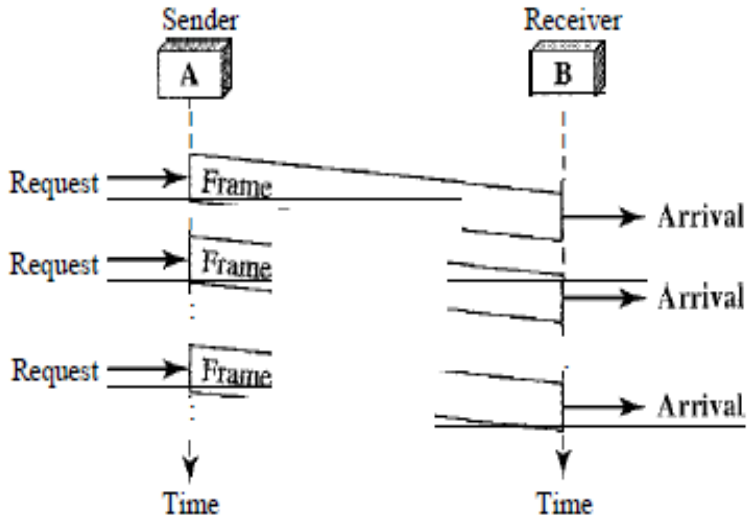
The data link layers use the services provided by their physical layers (such as signaling, multiplexing, and so on) for the physical transmission of bits. Figure 11.6 shows a design.

Example

Figure 11.7 shows an example of communication using this protocol. It is very simple. The sender sends a sequence of frames without even thinking about the receiver. To send three frames, three events occur at the sender site and three events at the receiver site.

Note that the data frames are shown by tilted boxes; the height of the box defines the transmission time difference between the first bit and the last bit in the frame.

Figure 11.7 *Flow diagram for Example 11.1*



Stop-and-Wait Protocol

If data frames arrive at the receiver site faster than they can be processed, the frames must be stored until their use. Normally, the receiver does not have enough storage space, especially if it is receiving data from many sources.

This may result in either the discarding of frames or denial of service.

To prevent the receiver from becoming overwhelmed with frames, we somehow need to tell the sender to slow down.

There must be feedback from the receiver to the sender.

The protocol we discuss now is called the Stop-and-Wait Protocol because the sender sends one frame, stops until it receives confirmation from the receiver (okay to go ahead), and then sends the next frame. We still have unidirectional communication for data frames, but auxiliary ACK frames (simple tokens of acknowledgment) travel from the other direction.

We add flow control to our previous protocol.

Stop-and-Wait Protocol

Working :

The sender sends data to the receiver.

The sender stops and waits for the acknowledgment.

The receiver receives the data and processes it.

The receiver sends an acknowledgment for the above data to the sender.

The sender sends data to the receiver after receiving the acknowledgment of previously sent data.

The process is unidirectional and continues until the sender sends the **End of Transmission (EoT)** frame.

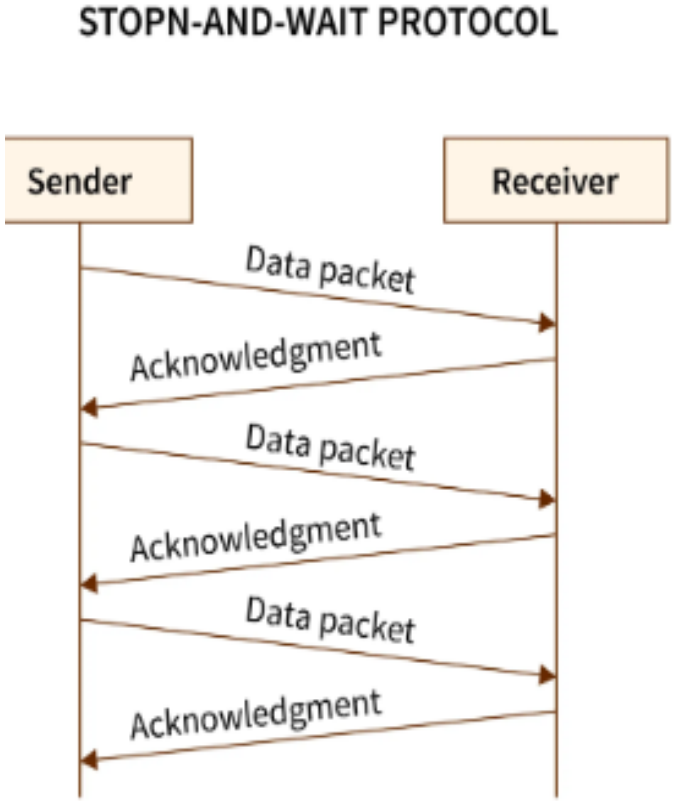
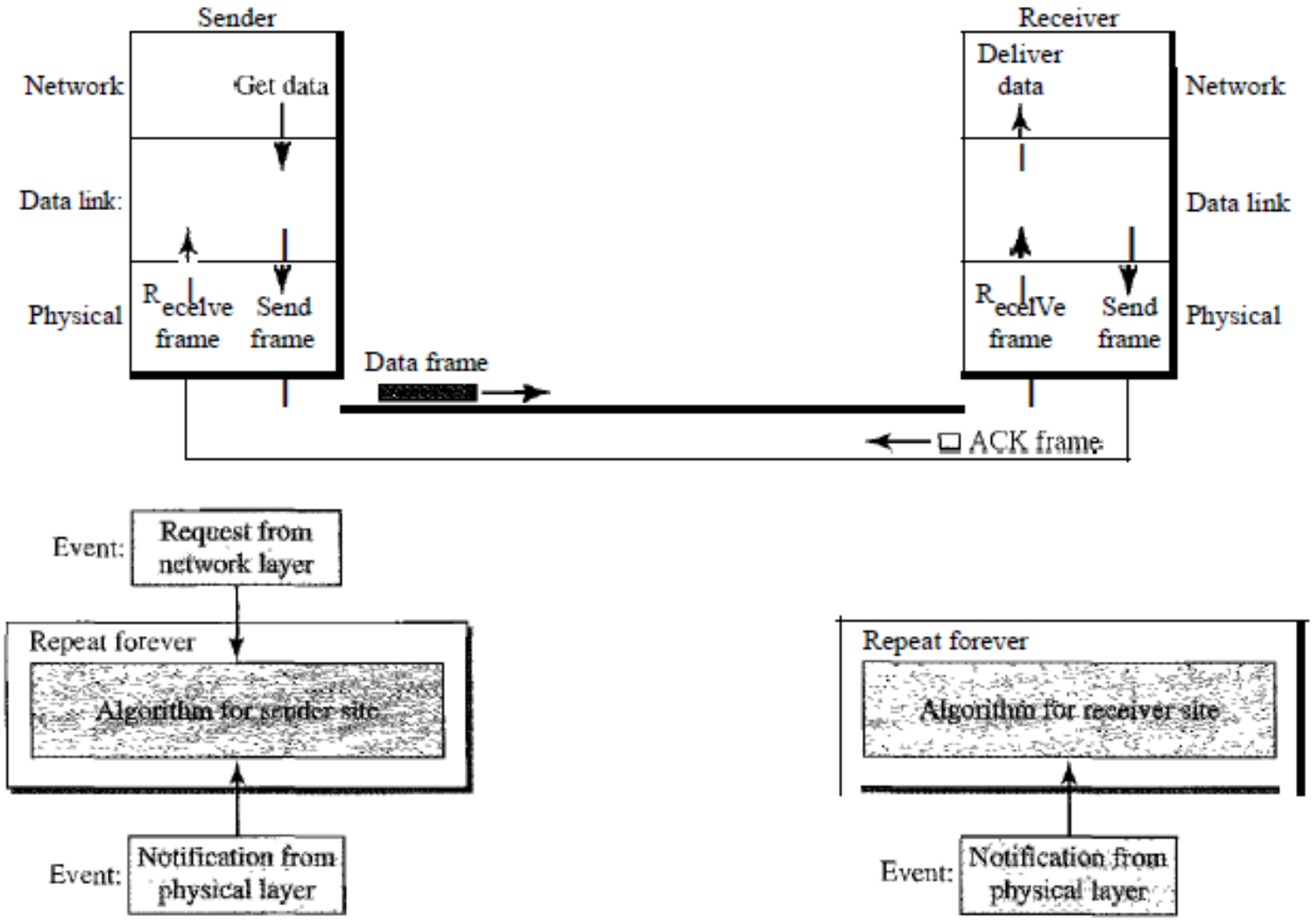


Figure 11.8 *Design of Stop-and-Wait Protocol*



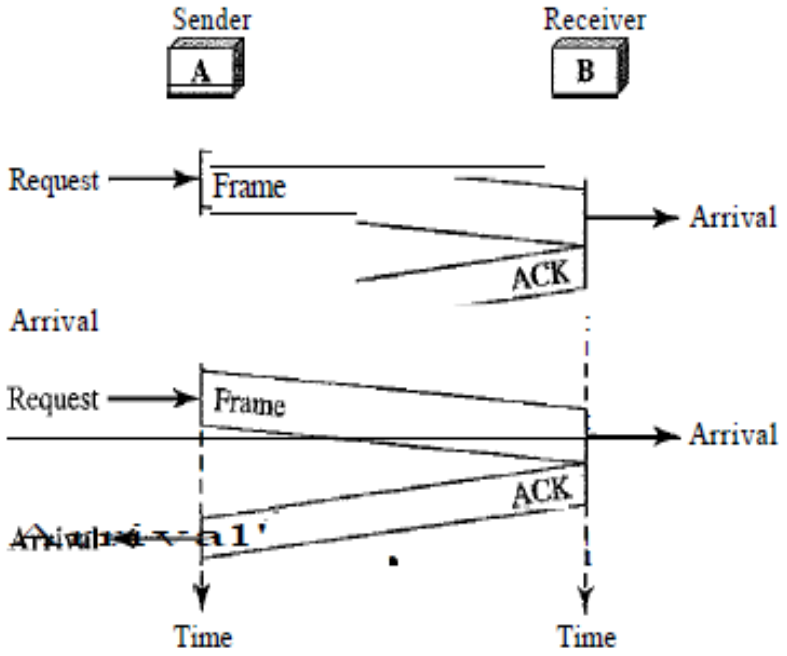
Example

Figure 11.9 shows an example of communication using this protocol. It is still very simple. The sender sends one frame and waits for feedback from the receiver.

When the ACK arrives, the sender sends the next frame.

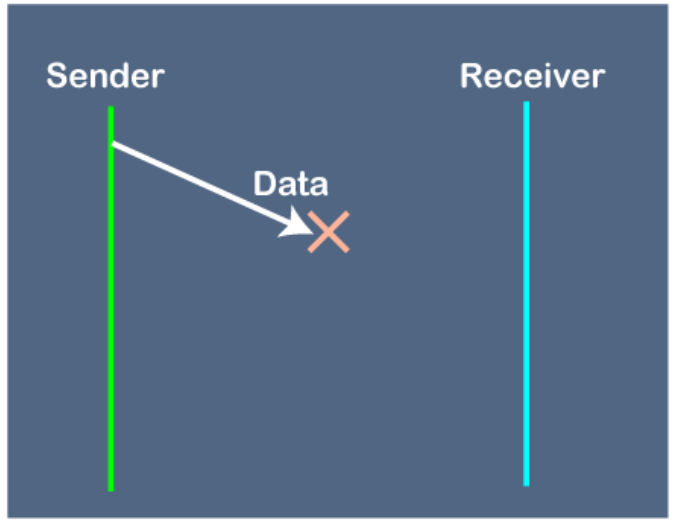
Note that sending two frames in the protocol involves the sender in four events and the receiver in two events.

Figure 11.9 *Flow diagramfor Example 11.2*



Disadvantages of Stop and Wait protocol

The following are the problems associated with a stop and wait protocol:



1. Problems occur due to lost data

Suppose the sender sends the data and the data is lost. The receiver is waiting for the data for a long time. Since the data is not received by the receiver, so it does not send any acknowledgment. Since the sender does not receive any acknowledgment so it will not send the next packet. This problem occurs due to the lost data.

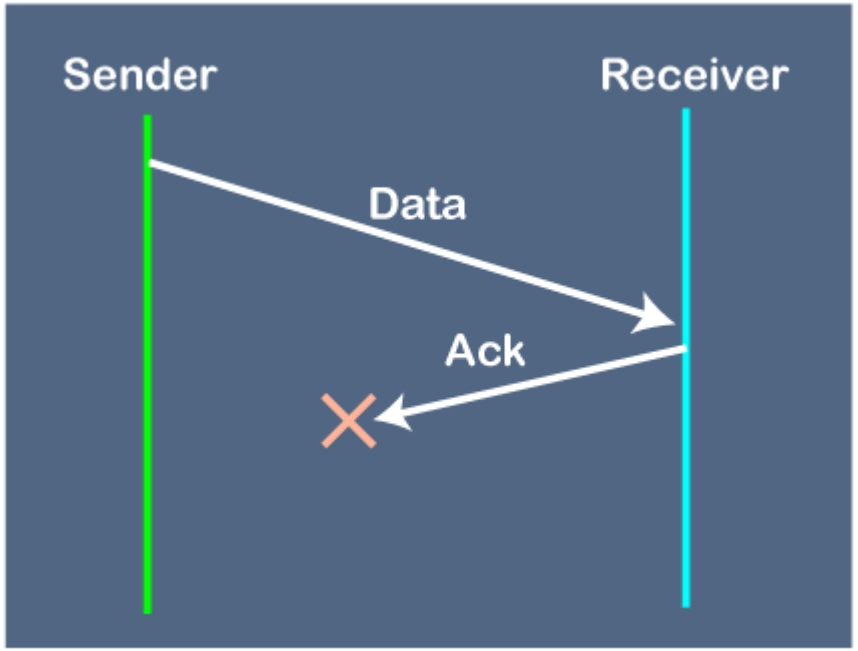
In this case, two problems occur:

Sender waits for an infinite amount of time for an acknowledgment.

Receiver waits for an infinite amount of time for a data.

2. Problems occur due to lost acknowledgment

Suppose the sender sends the data and it has also been received by the receiver. On receiving the packet, the receiver sends the acknowledgment. In this case, the acknowledgment is lost in a network, so there is no chance for the sender to receive the acknowledgment. There is also no chance for the sender to send the next packet as in stop and wait protocol, the next packet cannot be sent until the acknowledgment of the previous packet is received.

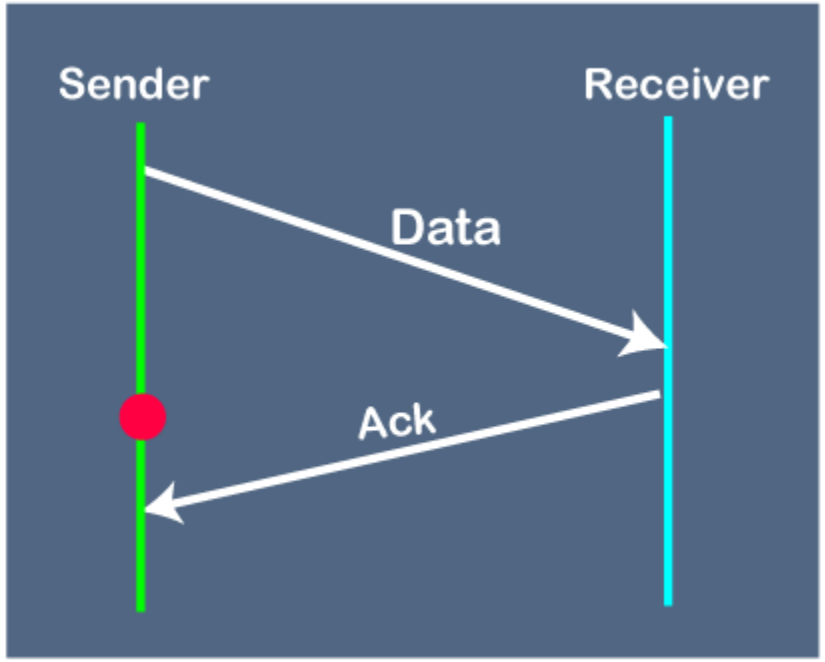


In this case, one problem occurs:

•Sender waits for an infinite amount of time for an acknowledgment.

3. Problem due to the delayed data or acknowledgment

Suppose the sender sends the data and it has also been received by the receiver. The receiver then sends the acknowledgment but the acknowledgment is received after the timeout period on the sender's side. As the acknowledgment is received late, so acknowledgment can be wrongly considered as the acknowledgment of some other data packet.



NOISY CHANNELS

Although the Stop-and-Wait Protocol gives us an idea of how to add flow control to its predecessor, noiseless channels are non-existent. We can ignore the error (as we sometimes do), or we need to add error control to our protocols. We discuss three protocols in this section that use error control.

Stop-and-Wait Automatic Repeat Request

Our first protocol, called the Stop-and-Wait Automatic Repeat Request (Stop-and Wait ARQ), adds a simple error control mechanism to the Stop-and-Wait Protocol.

To detect and correct corrupted frames, we need to add redundancy bits to our data frame. When the frame arrives at the receiver site, it is checked and if it is corrupted, it is silently discarded. The detection of errors in this protocol is manifested by the silence of the receiver.

Lost frames are more difficult to handle than corrupted ones. In our previous protocols, there was no way to identify a frame. The received frame could be the correct one, or a duplicate, or a frame out of order. The solution is to number the frames. When the receiver receives a data frame that is out of order, this means that frames were either lost or duplicated.

Stop-and-Wait Automatic Repeat Request

In the stop and wait ARQ protocol, ARQ stands for Automatic Repeat Request. ARQ is an error-control strategy that ensures that a sequence of information is delivered in order and without any error or duplications despite transmission errors and losses.

In the stop and wait ARQ, the sender also keeps a copy of the currently sending frame so that if the receiver does not receive the frame then it can retransmit it. In stop and wait ARQ, the sender sets a timer for each frame so whenever the timer is over and the sender has not received any acknowledgment for the frame, then the sender knows that the particular frame is either lost or damaged. So, the sender sends back the lost or damaged frame once the timer is out. So, we can see that the sender needs to wait for the timer to expire before retransmission

There is no negative acknowledgment of the lost or damaged frames. So, there is no NACK (negative acknowledgment) in case of stop and wait ARQ.

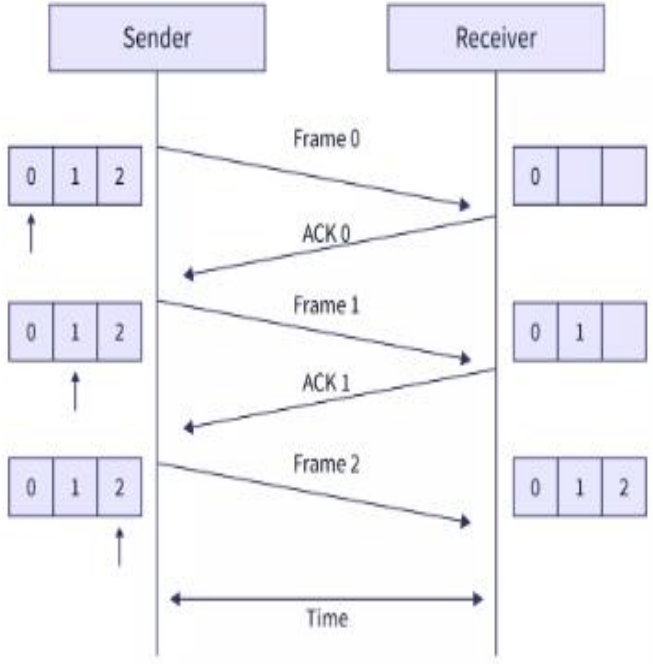
Working Principle of Stop and Wait ARQ

In the stop and wait ARQ, both the sender and the receiver have windows of the same size. The window on the sender's side covers the sequence of data packets that are sent (or to be sent). On the other hand, the window on the receiver's side covers the sequence of data packets that are received (or to be received).

The size of the sender's window is 1. The window size of the receiver is the same as that of the sender i.e. 1. The sender's window size is represented using **Ws** and the receiver's window size is represented using **Wr**.

The overall working of the stop and wait ARQ is simple. Initially, the sender sends one frame as the window size is 1. The receiver on the other end receives the frame and sends the ACK for the correctly received frame. The sender waits for the ACK until the timer expires. If the sender does not receive the ACK within the timer limit, it re-transmits the frame for which the ACK has not been received.

Now, let us take an example to visualize the working of stop and wait ARQ or how the data frame is transmitted using the stop and wait ARQ protocol. The image below shows the transmission of frames.



Working Principle of Stop and Wait ARQ

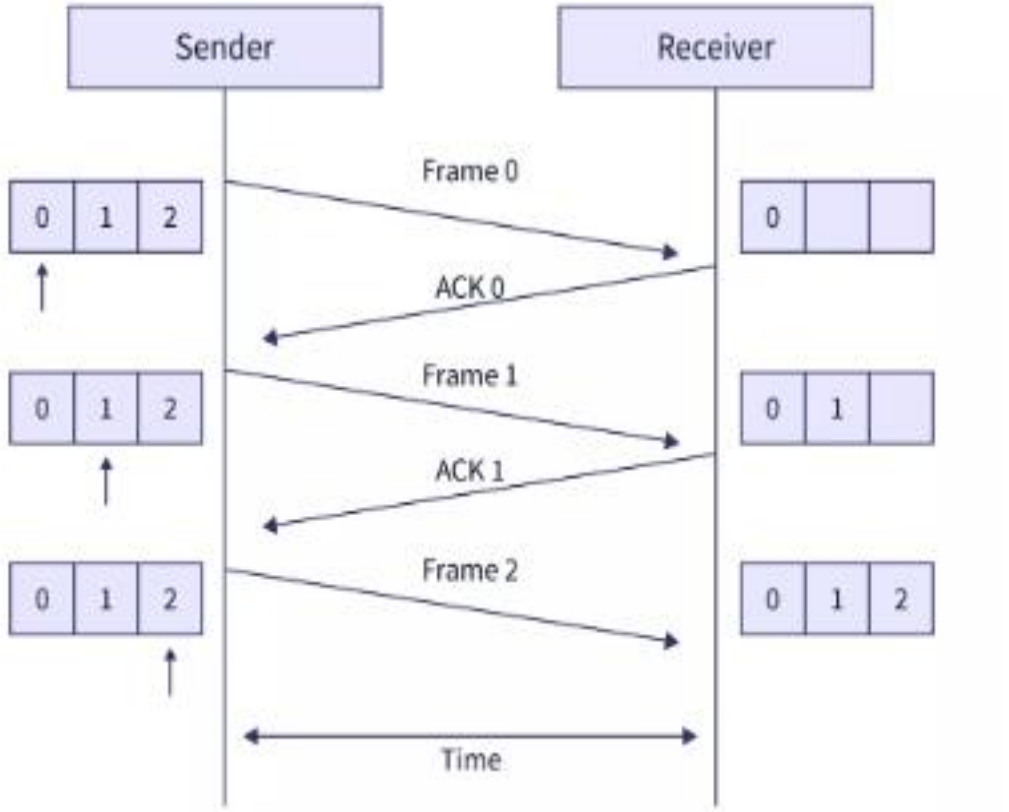
The steps of data transmission can be:

The sender sends frame 0.

The sender waits for ACK from the receiver.

The receiver receives the frame and sends back ACK 0.

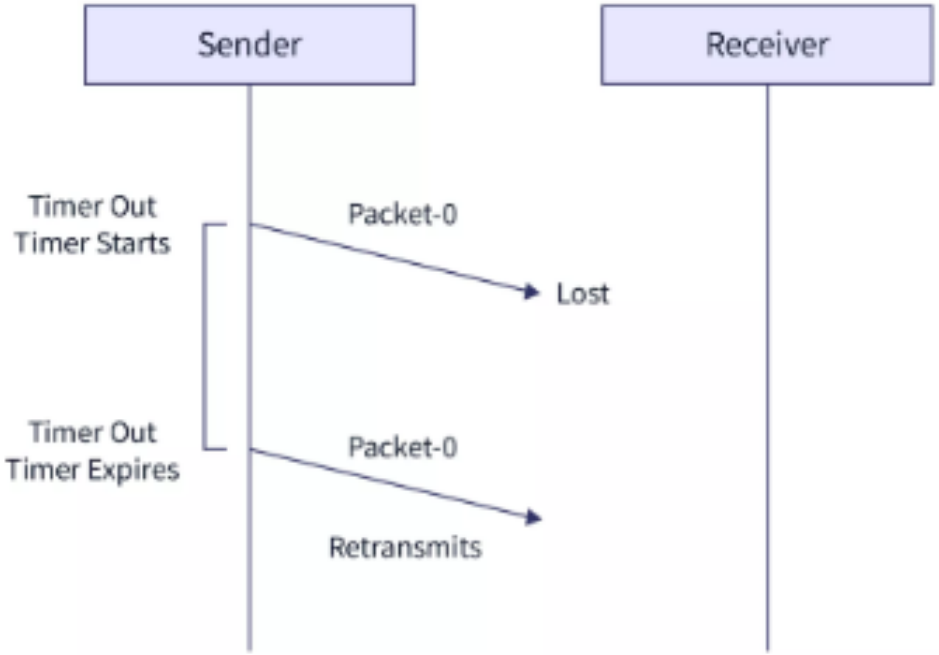
Again the sender sends the frame 1 and this process is continued till all the frames have been received by the receiver.



Let us discuss the various problems of the Stop and Wait ARQ -

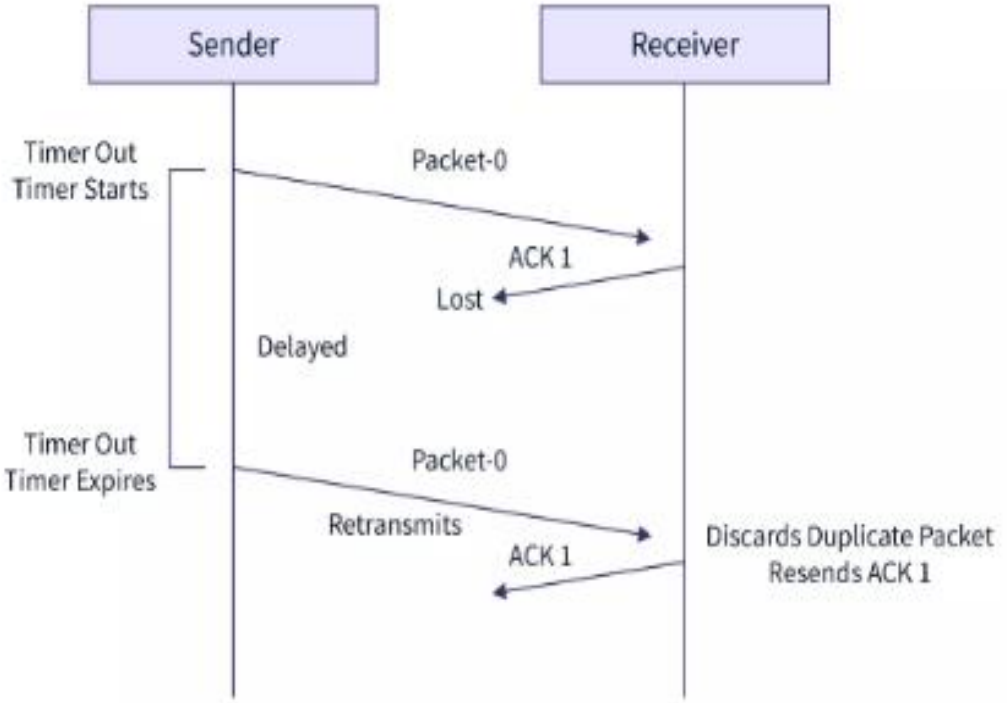
1. Problem of Lost Data Packet

When the sender sends the data packet and the receiver does not receive the data packet, it means that the data is lost in between the transmission. So, to overcome this type of problem, the sender uses a timer. When the sender sends the data packet, it starts a timer. If the timer goes off before receiving the acknowledgment from the receiver, the sender retransmits the same data packet. Refer to the image below for better visualization.



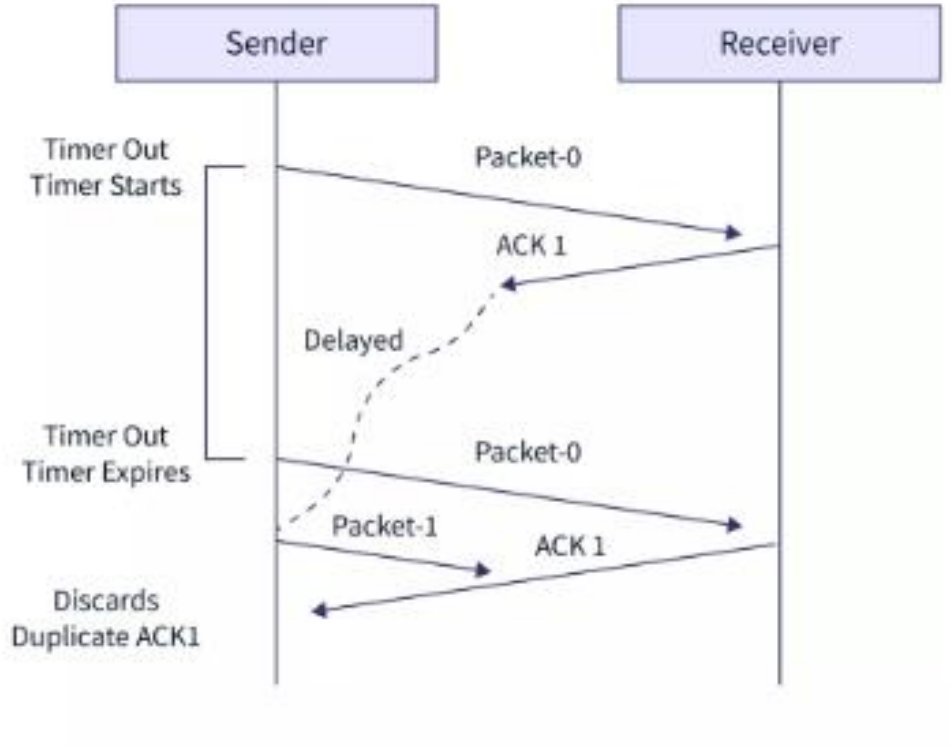
2. Problem of Lost Acknowledgement

When the sender sends the data packet and the receiver receives the data packet but the acknowledgment from the receiver is not received. It means that the acknowledgment is lost in between the transmission. So, to overcome this type of problem, the sender uses sequence numbering. When the sender sends the data packet, it attaches a certain sequence number which helps the receiver identify the data packet. If the timer goes off before receiving the acknowledgment from the receiver, the sender retransmits the same data packet. But in this case, the receiver already has the data packet, so it discards the data and sends it back an acknowledgment. This tells the sender that the certain data packet is now received correctly.



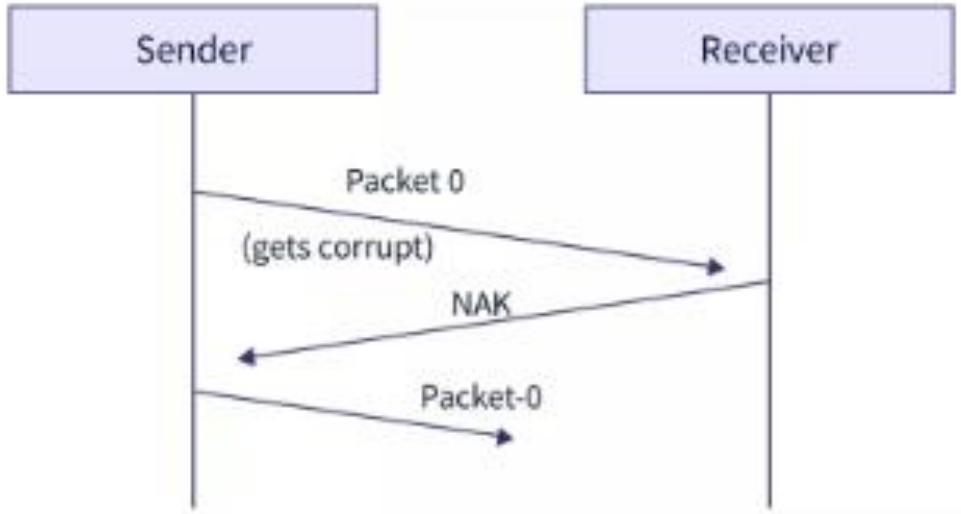
3. Problem of Delayed Acknowledgement

When the sender sends the data packet and the receiver receives the data packet but the acknowledgment from the receiver is not received. It means that the acknowledgment is lost or delayed in between the transmission. So, to overcome this type of problem, the sender uses sequence numbering. When the sender sends the data packet, it attaches a certain sequence number which helps the receiver identify the data packet. If the timer goes off before receiving the acknowledgment from the receiver, the sender retransmits the same data packet. But in this case, the receiver already has the data packet, so it discards the data and sends it back the acknowledgment again. Now, if the sender has received the previously sent acknowledgment then the newer acknowledgment is discarded by the sender. Refer to the image below for better visualization.



4. Problem of Damaged Packet

When the sender sends the data packet and the receiver receives the data packet the received data packet is corrupted. So, to overcome this type of problem, the receiver uses negative acknowledgment (NACK). So, if the sender receives a negative acknowledgment then the sender retransmits the data packet.



Advantages of Stop and Wait ARQ :

Simple Implementation: Stop and Wait ARQ is a simple protocol that is easy to implement in both hardware and software. It does not require complex algorithms or hardware components, making it an inexpensive and efficient option.

Error Detection: Stop and Wait ARQ detects errors in the transmitted data by using checksums or cyclic redundancy checks (CRC). If an error is detected, the receiver sends a negative acknowledgment (NAK) to the sender, indicating that the data needs to be retransmitted.

Reliable: Stop and Wait ARQ ensures that the data is transmitted reliably and in order. The receiver cannot move on to the next data packet until it receives the current one. This ensures that the data is received in the correct order and eliminates the possibility of data corruption.

Flow Control: Stop and Wait ARQ can be used for flow control, where the receiver can control the rate at which the sender transmits data. This is useful in situations where the receiver has limited buffer space or processing power.

Backward Compatibility: Stop and Wait ARQ is compatible with many existing systems and protocols, making it a popular choice for communication over unreliable channels.

Disadvantages of Stop and Wait ARQ :

Low Efficiency: Stop and Wait ARQ has low efficiency as it requires the sender to wait for an acknowledgment from the receiver before sending the next data packet. This results in a low data transmission rate, especially for large data sets.

High Latency: Stop and Wait ARQ introduces additional latency in the transmission of data, as the sender must wait for an acknowledgment before sending the next packet. This can be a problem for real-time applications such as video streaming or online gaming.

Limited Bandwidth Utilization: Stop and Wait ARQ does not utilize the available bandwidth efficiently, as the sender can transmit only one data packet at a time. This results in underutilization of the channel, which can be a problem in situations where the available bandwidth is limited.

Limited Error Recovery: Stop and Wait ARQ has limited error recovery capabilities. If a data packet is lost or corrupted, the sender must retransmit the entire packet, which can be time-consuming and can result in further delays.

Vulnerable to Channel Noise: Stop and Wait ARQ is vulnerable to channel noise, which can cause errors in the transmitted data. This can result in frequent retransmissions and can impact the overall efficiency of the protocol.

What is Sliding Window Protocol?

Sliding windows are a method of sending multiple frames at once. It manages data packets between the two devices, ensuring delivery of data frames reliably and gradually. It’s also found in TCP (Transmission Control Protocol).

Each frame is sent from the sequence number in this technique. The sequence numbers find use in the receiving end to locate missing data. The sequence number finds use in the sliding window technique in order to avoid duplicate data.

Working Principle of Sliding Window:

The sender's buffer is the sending window, and the receiver's buffer is the receiving window in these protocols.

Assignment of sequence numbers to frames is between the range 0 to 2n-1 if the frames' sequence number is an n-bit field. As a result, the sending window has a size of 2n-1. As a result, we choose an n-bit sequence number to accommodate a sending window size of 2n-1.

Modulo-n is used to number the sequence numbers. If the sending window size is set as 3, the sequence numbers will be 0, 1, 2, 0, 1, 2, 0, 1, 2 and so on.

The receiving window's size refers to the maximum number of frames the receiver can accept at one time. It establishes the maximum number of frames a sender can send before receiving an acknowledgement.

Sliding Window (Sender and Receiver side):

a. Sender Side:

The sequence number of the frame occupies a field in the frame. So, the sequence number should be kept to a minimum.

The sequence number ranges from 0 to 2k-1 if the frame header allows k bits.

The sender maintains a list of sequence numbers that are only allowed to be sent by the sender.

The sender window can only be 2k-1 in size.

For example, if the frame allows 4 bits, the window's size is 2 raised to the power 4 -1=((2^4)-1) =16-1=15.

The sender has a buffer with the same size as the window.

b. Receiver Side:

On the receiver side, the size of the window is always 1.

The receiver acknowledges a frame by sending an ACK frame to the sender, along with the sequence number of the next expected frame.

The receiver declares explicitly that it is ready to receive N subsequent frames, starting with the specified number.

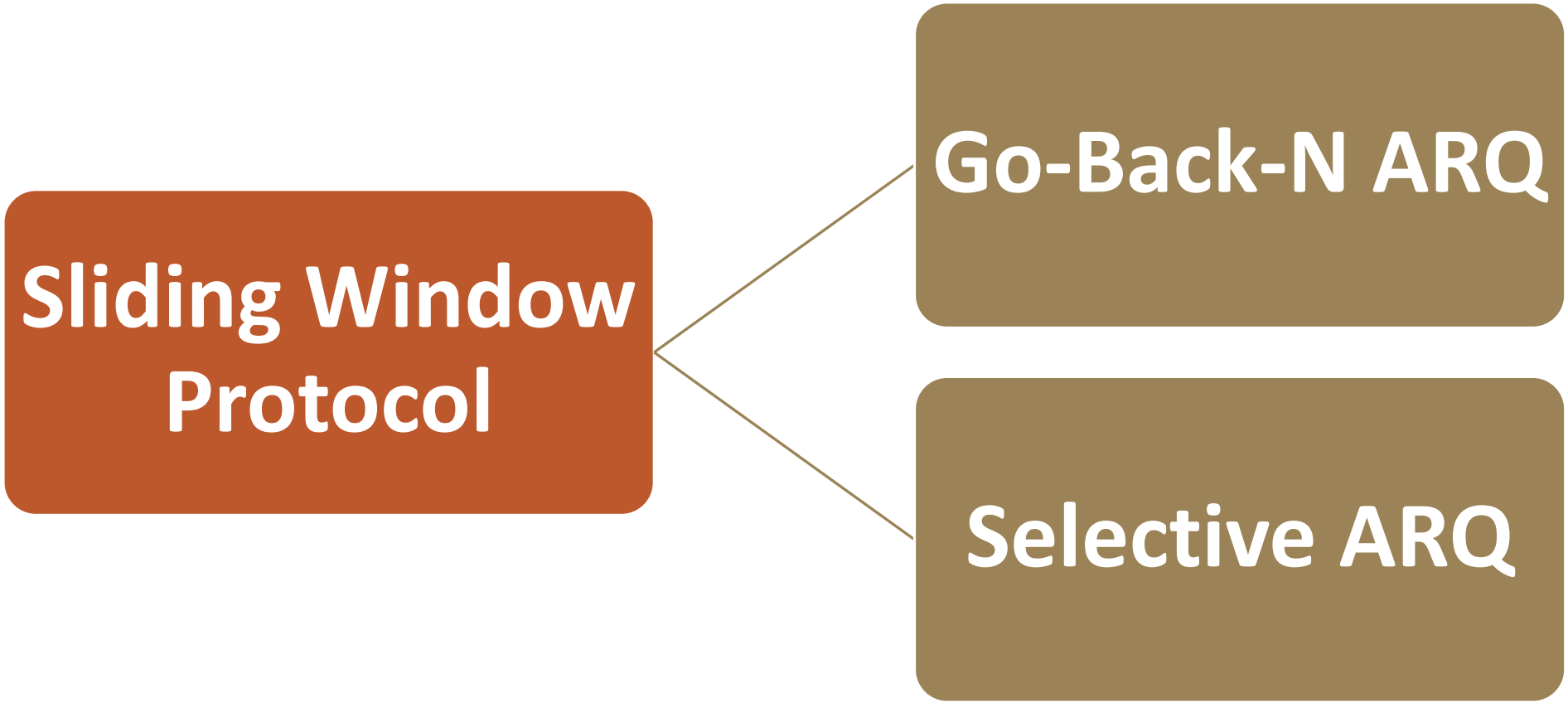
We use this scheme in order to acknowledge multiple frames.

The receiver's window can hold 2,3,4 frames, but the ACK frame will be held until frame 4 arrives.

It will send the ACK along with sequence number 5 after the arrival, with which the acknowledgment of 2,3,4 will be done one at a time.

The receiver requires a buffer size of one.

Types of Sliding Window Protocols:



Go-Back-N ARQ

Before understanding the working of Go-Back-N ARQ, we first look at the sliding window protocol. As we know that the sliding window protocol is different from the stop-and-wait protocol.

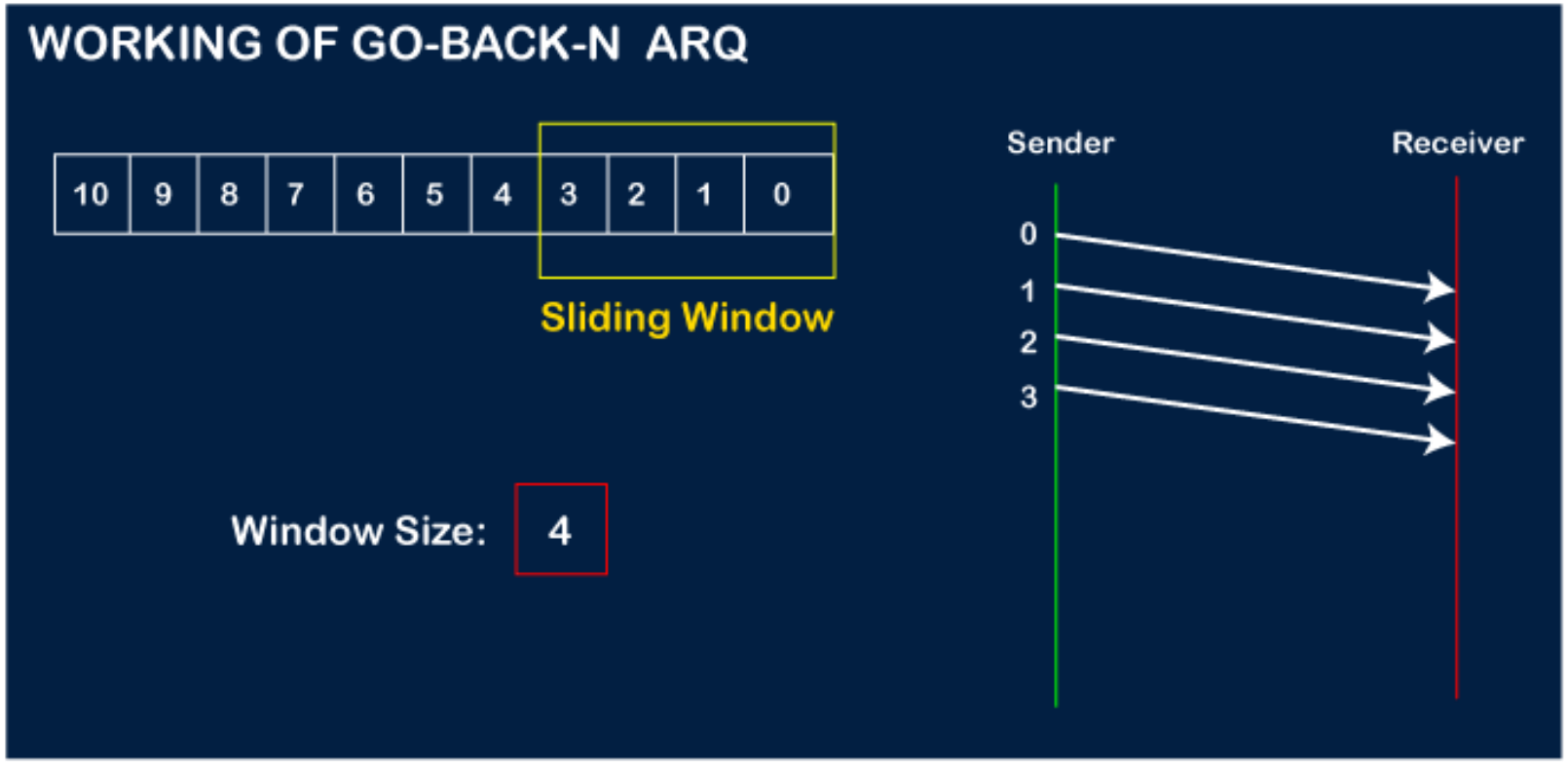
In the stop-and-wait protocol, the sender can send only one frame at a time and cannot send the next frame without receiving the acknowledgment of the previously sent frame, whereas, in the case of sliding window protocol, the multiple frames can be sent at a time.

The variations of sliding window protocol are Go-Back-N ARQ and Selective Repeat ARQ. Let's understand 'what is Go-Back-N ARQ'.

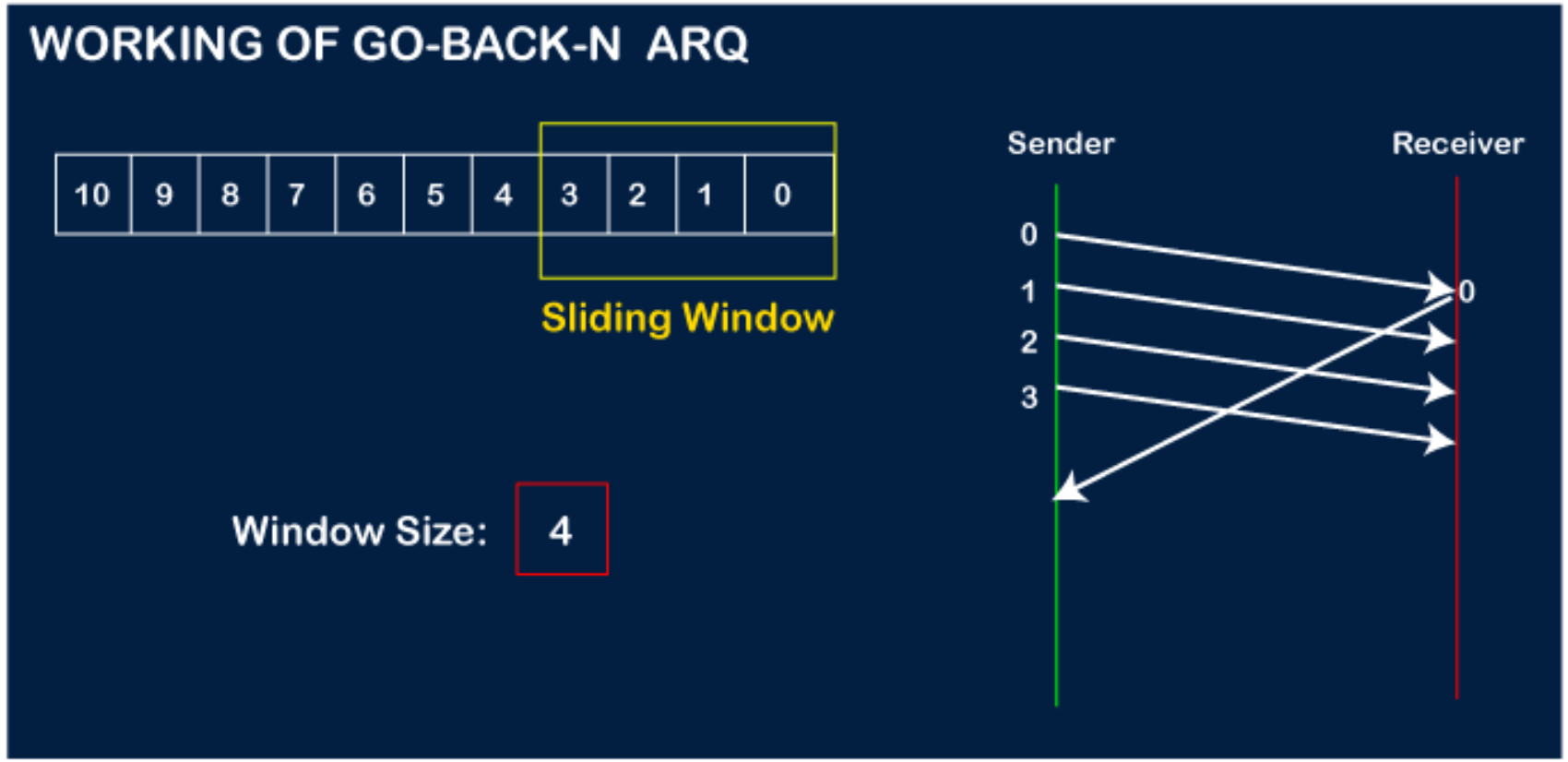
Working of Go-Back-N ARQ

Suppose there are a sender and a receiver, and let's assume that there are 11 frames to be sent. These frames are represented as 0,1,2,3,4,5,6,7,8,9,10, and these are the sequence numbers of the frames. Mainly, the sequence number is decided by the sender's window size. But, for the better understanding, we took the running sequence numbers, i.e., 0,1,2,3,4,5,6,7,8,9,10. Let's consider the window size as 4, which means that the four frames can be sent at a time before expecting the acknowledgment of the first frame.

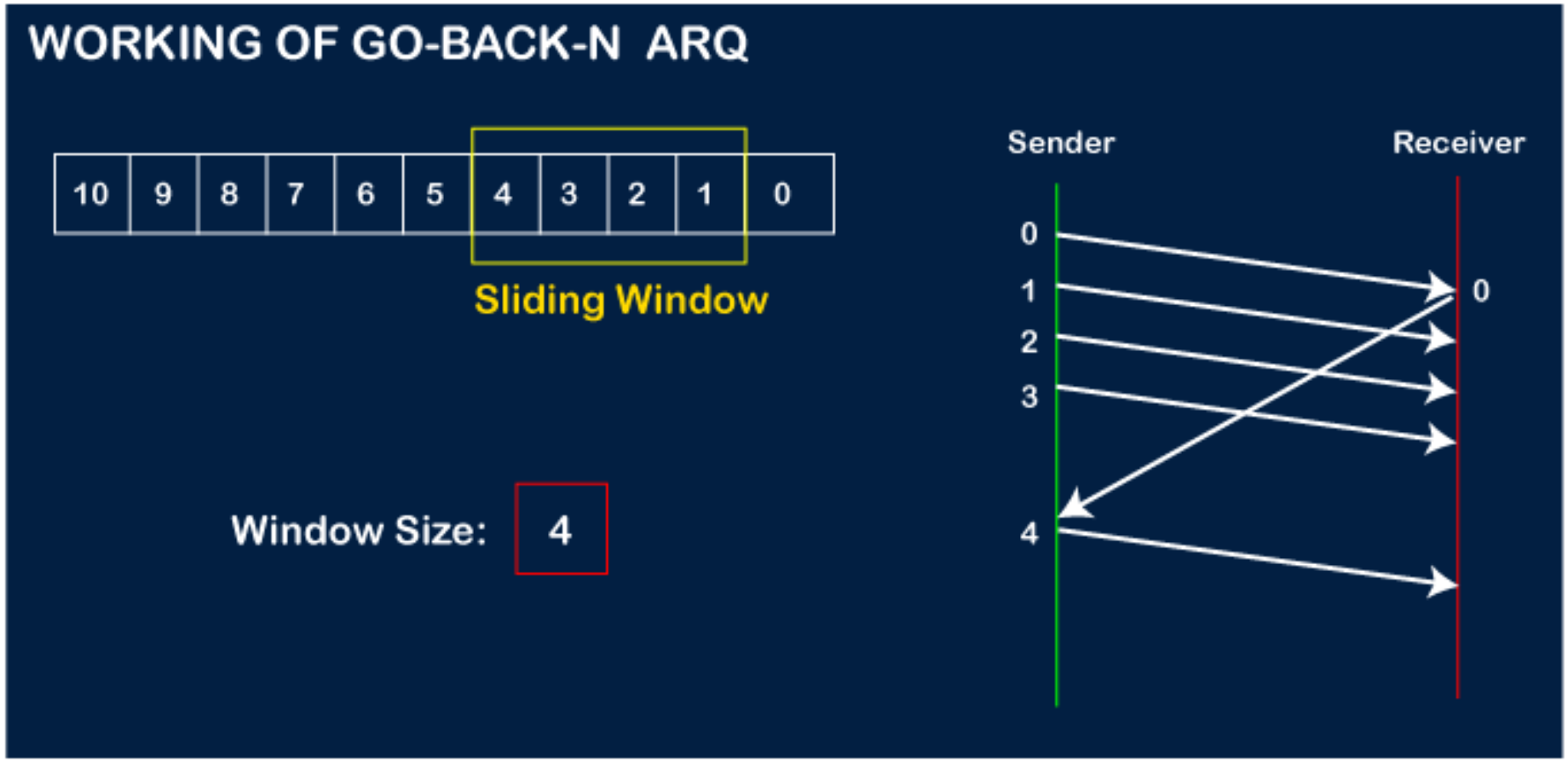
Step 1: Firstly, the sender will send the first four frames to the receiver, i.e., 0,1,2,3, and now the sender is expected to receive the acknowledgment of the 0^{th} frame.



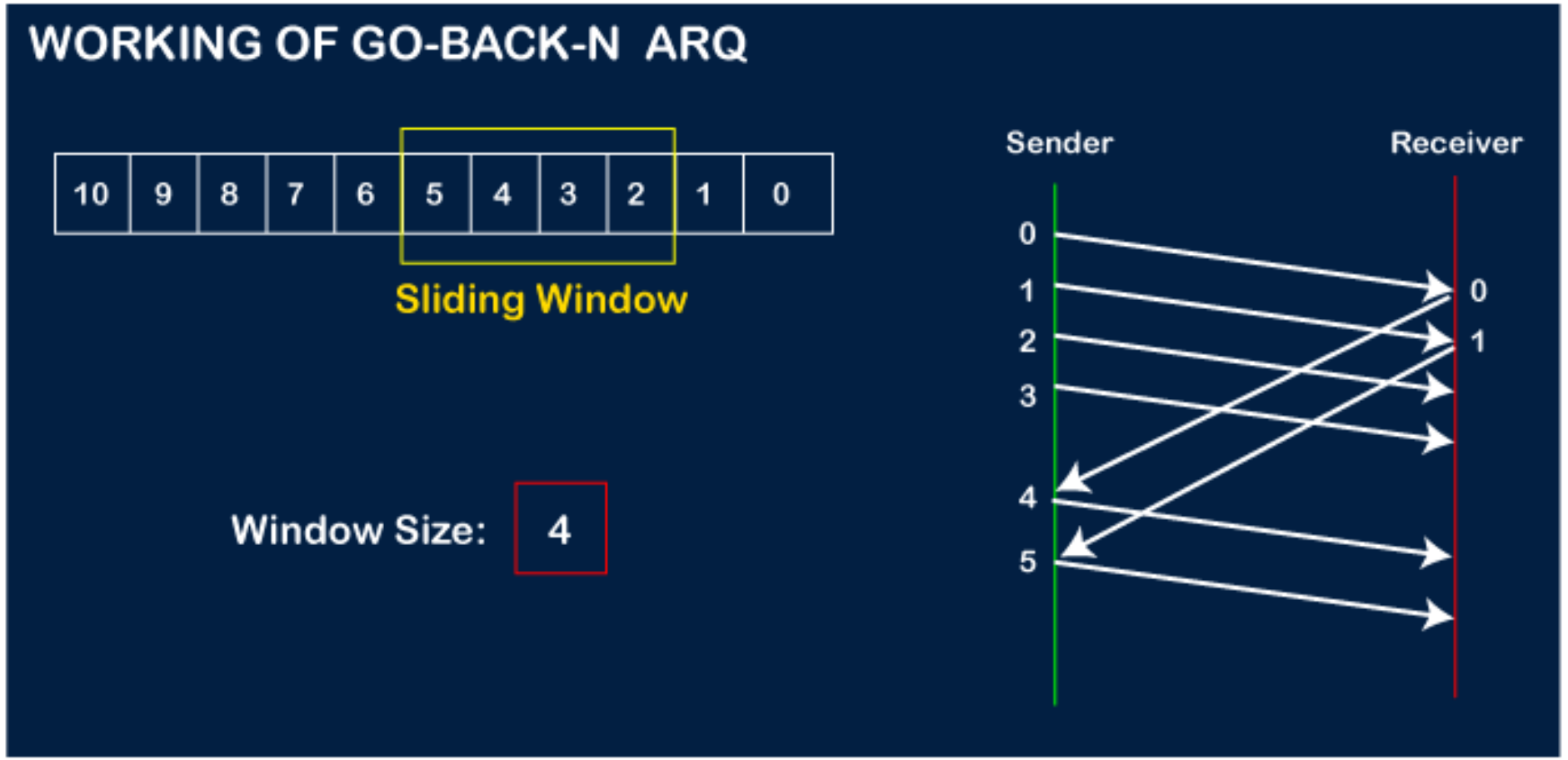
Let's assume that the receiver has sent the acknowledgment for the 0 frame, and the receiver has successfully received it.



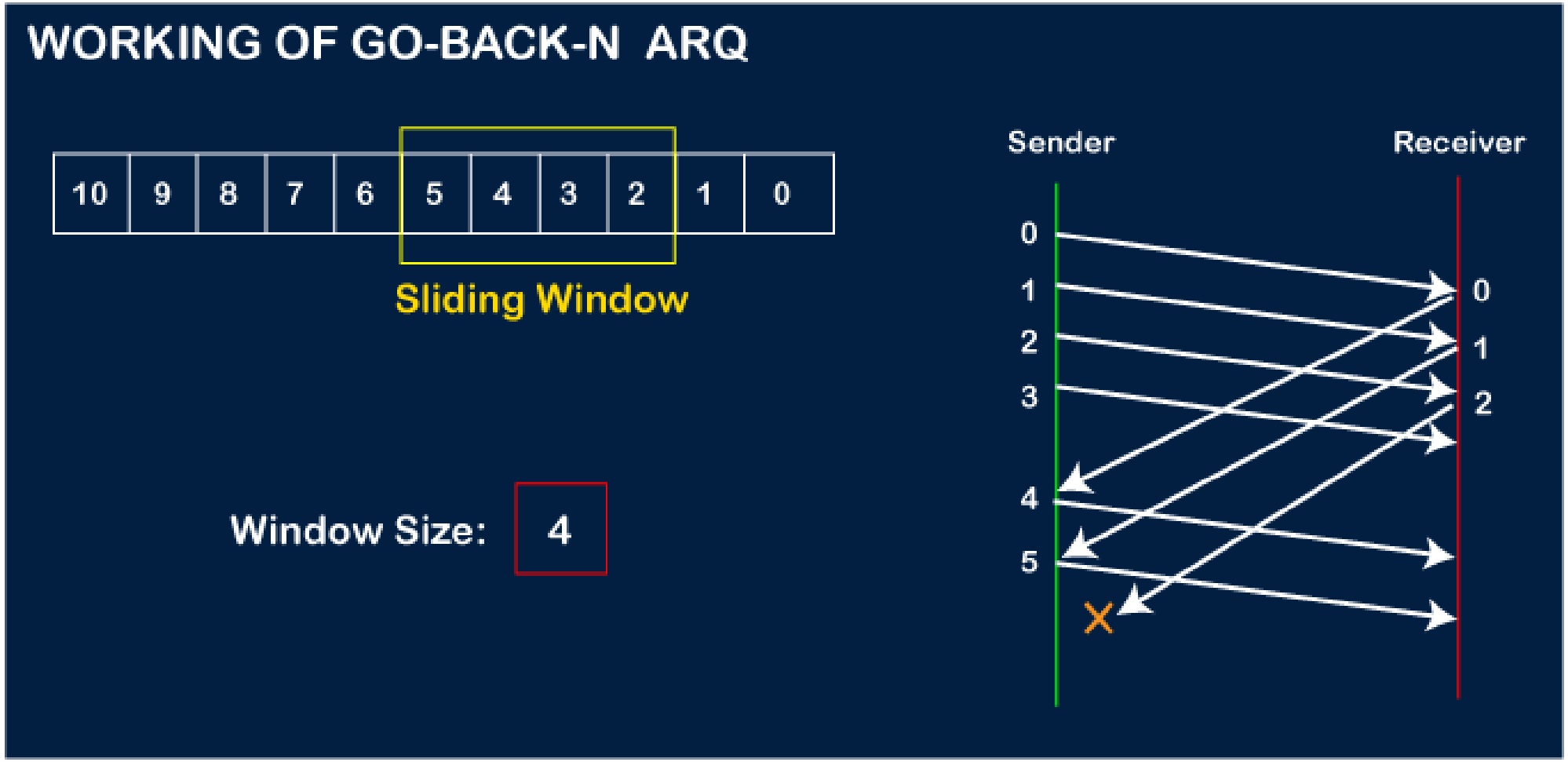
The sender will then send the next frame, i.e., 4, and the window slides containing four frames (1,2,3,4)

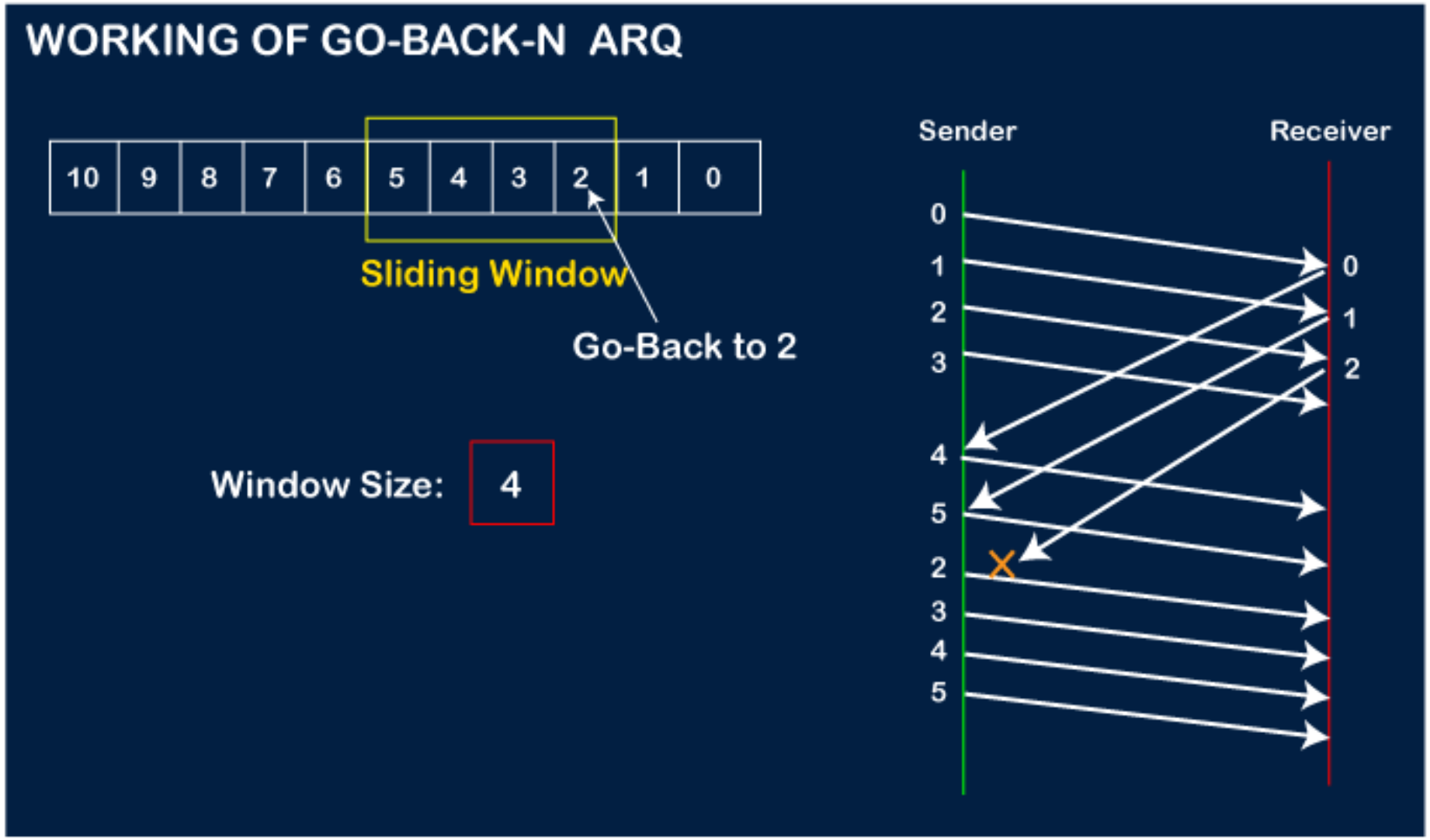


The receiver will then send the acknowledgment for the frame no 1. After receiving the acknowledgment, the sender will send the next frame, i.e., frame no 5, and the window will slide having four frames (2,3,4,5).



Now, let's assume that the receiver is not acknowledging the frame no 2, either the frame is lost, or the acknowledgment is lost. Instead of sending the frame no 6, the sender Go-Back to 2, which is the first frame of the current window, retransmits all the frames in the current window, i.e., 2,3,4,5.




WORKING OF GO-BACK-N ARQ
10
9
8
7
6
5
4
3
2
1
0
Sliding Window
Go-Back to 2
Window Size:
4
Sender
Receiver
0
1
2
3
4
5
2
3
4
5
0
1
2
X

Important points related to Go-Back-N ARQ:

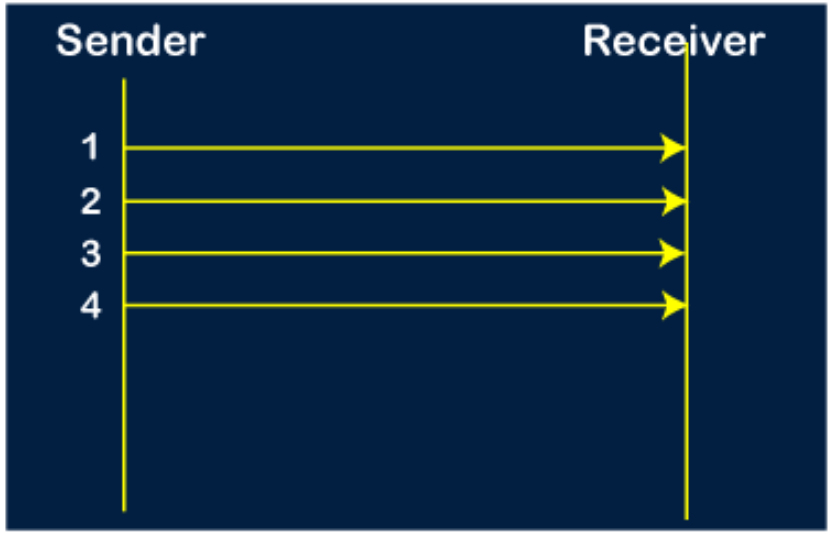
- In Go-Back-N, N determines the sender's window size, and the size of the receiver's window is always 1.
- It does not consider the corrupted frames and simply discards them.
- It does not accept the frames which are out of order and discards them.
- If the sender does not receive the acknowledgment, it leads to the retransmission of all the current window frames.

Example

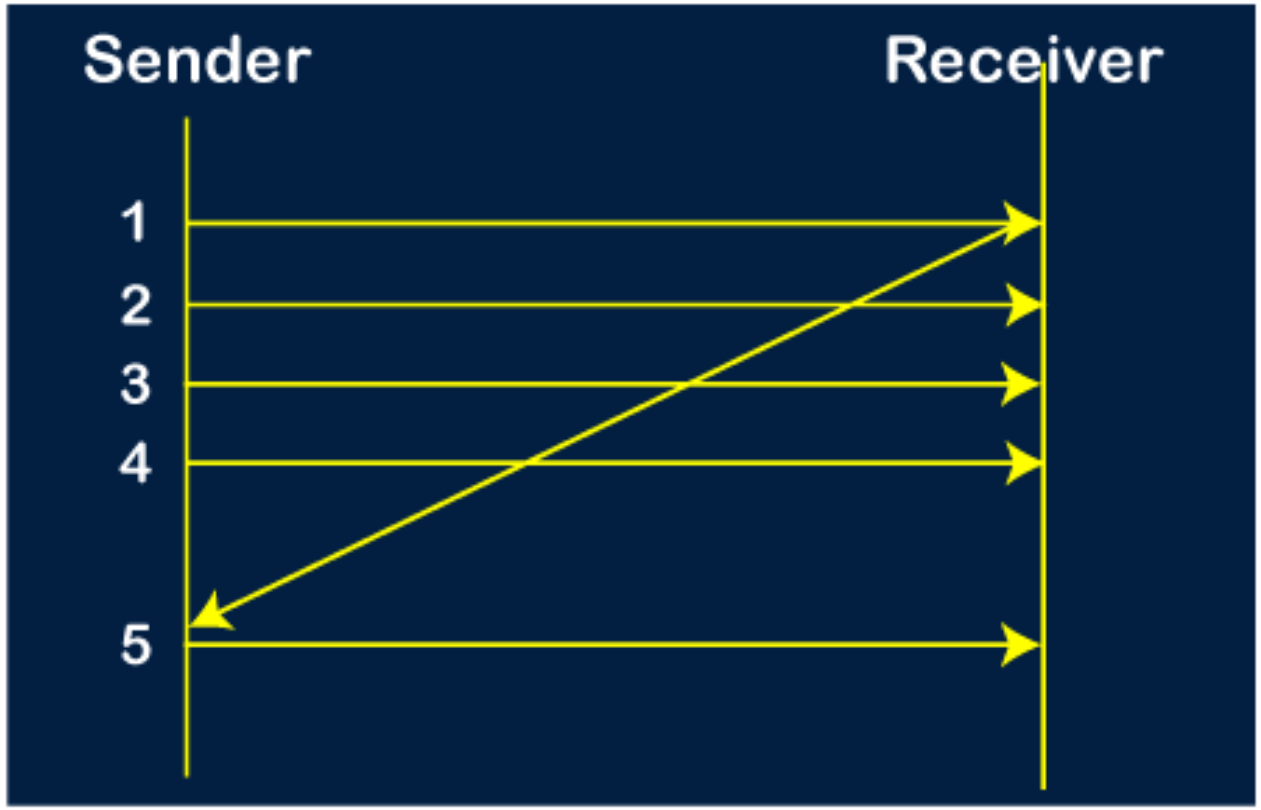
Example 1: In GB4, if every 6th packet being transmitted is lost and if we have to spend 10 packets then how many transmissions are required?

Solution: Here, GB4 means that N is equal to 4. The size of the sender's window is 4.

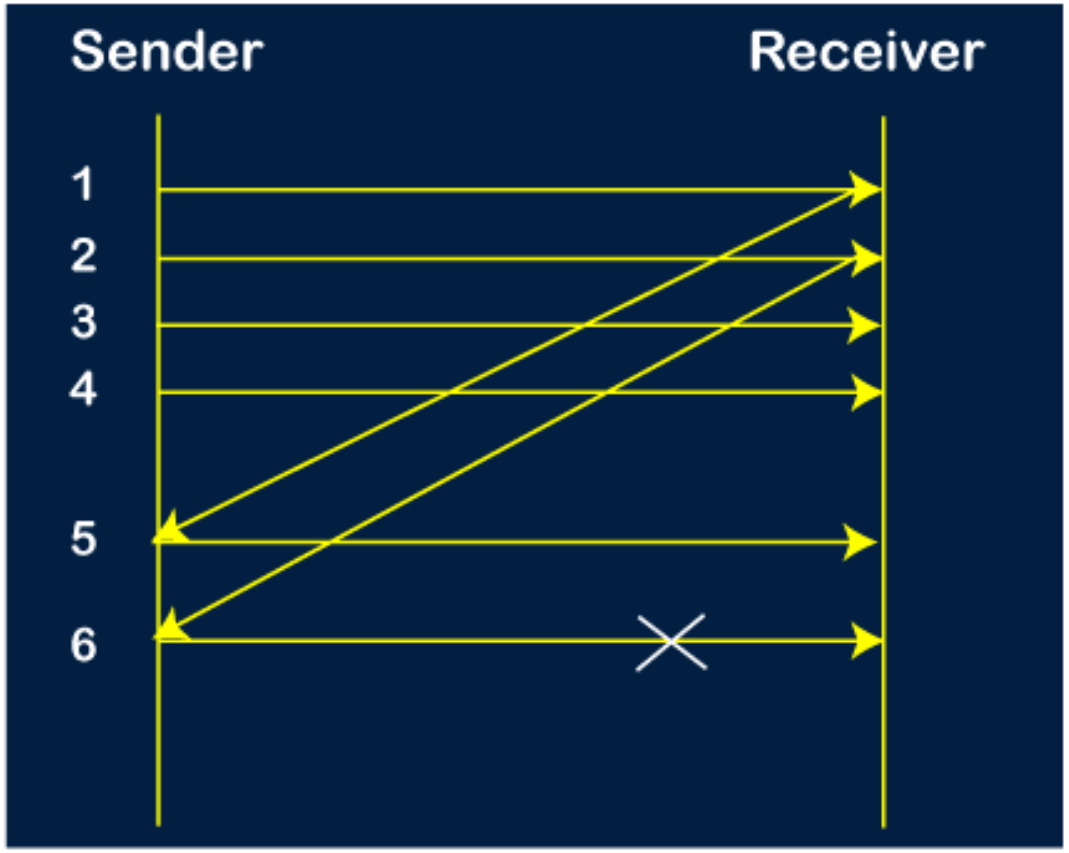
Step 1: As the window size is 4, so four packets are transferred at a time, i.e., packet no 1, packet no 2, packet no 3, and packet no 4.



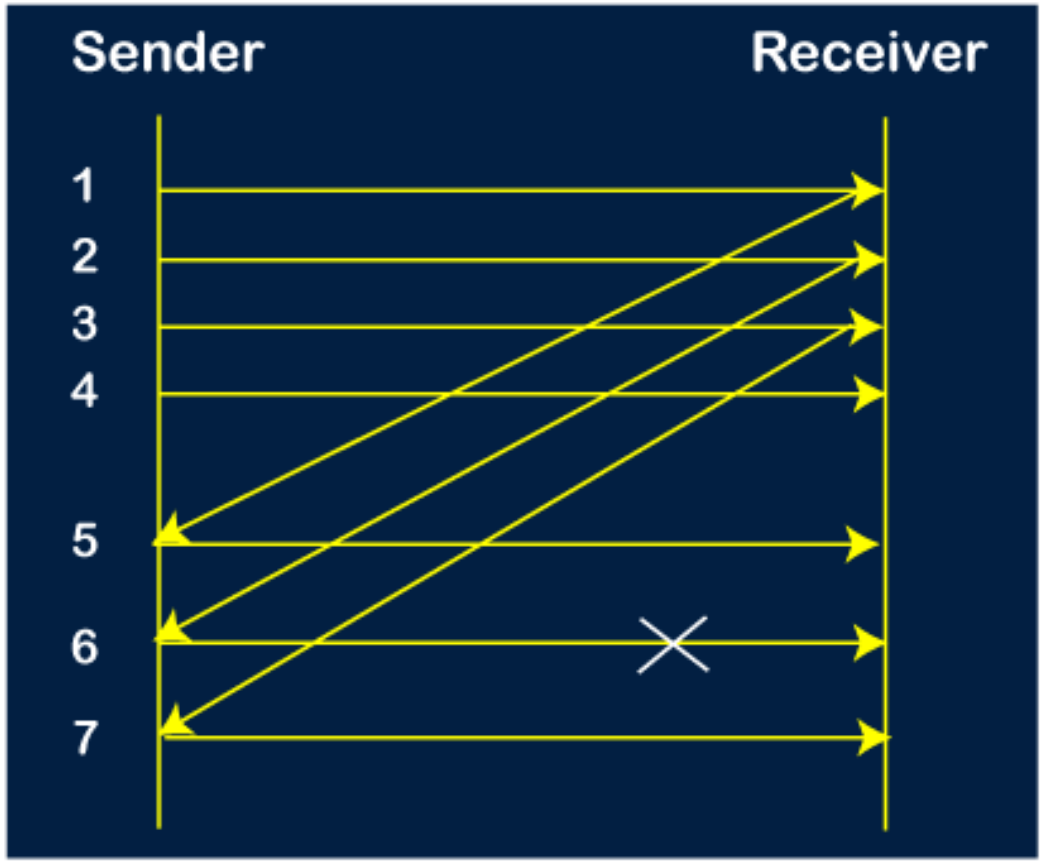
Step 2: Once the transfer of window size is completed, the sender receives the acknowledgment of the first frame, i.e., packet no1. As the acknowledgment receives, the sender sends the next packet, i.e., packet no 5. In this case, the window slides having four packets, i.e., 2,3,4,5 and excluded the packet 1 as the acknowledgment of the packet 1 has been received successfully.



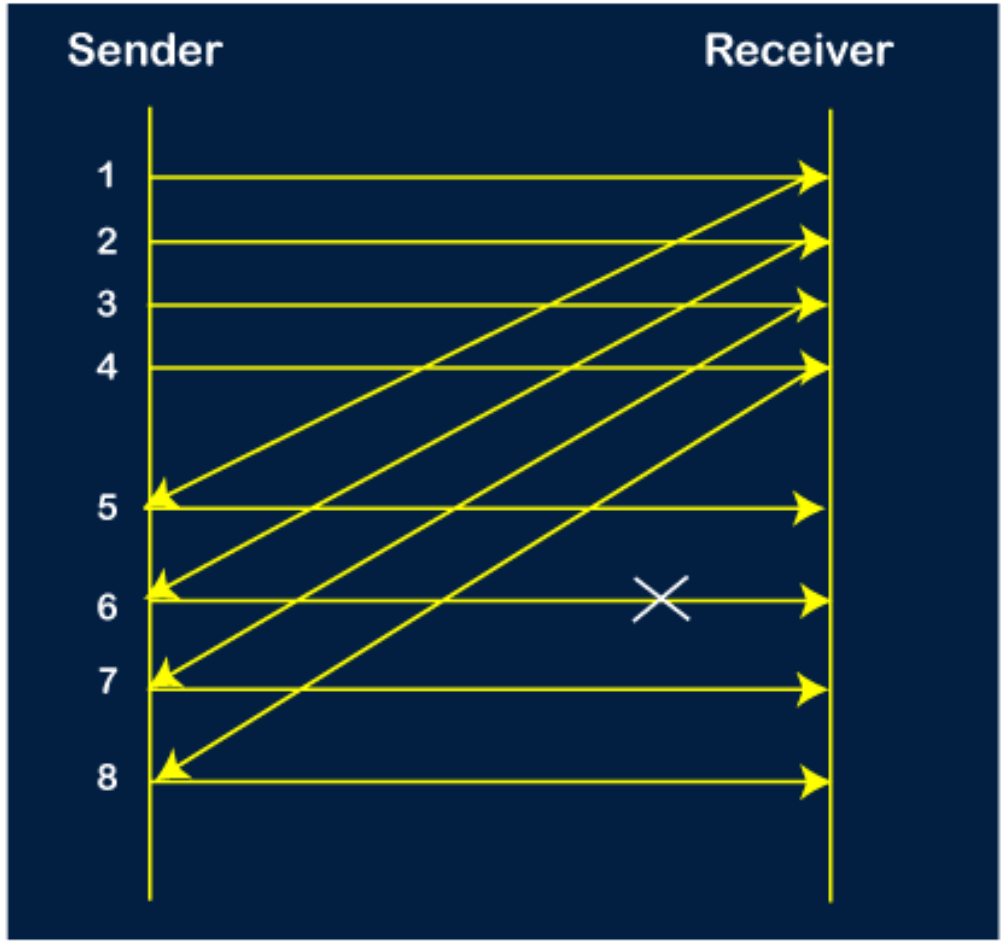
Step 3: Now, the sender receives the acknowledgment of packet 2. After receiving the acknowledgment for packet 2, the sender sends the next packet, i.e., packet no 6. As mentioned in the question that every 6^{th} is being lost, so this 6^{th} packet is lost, but the sender does not know that the 6^{th} packet has been lost.



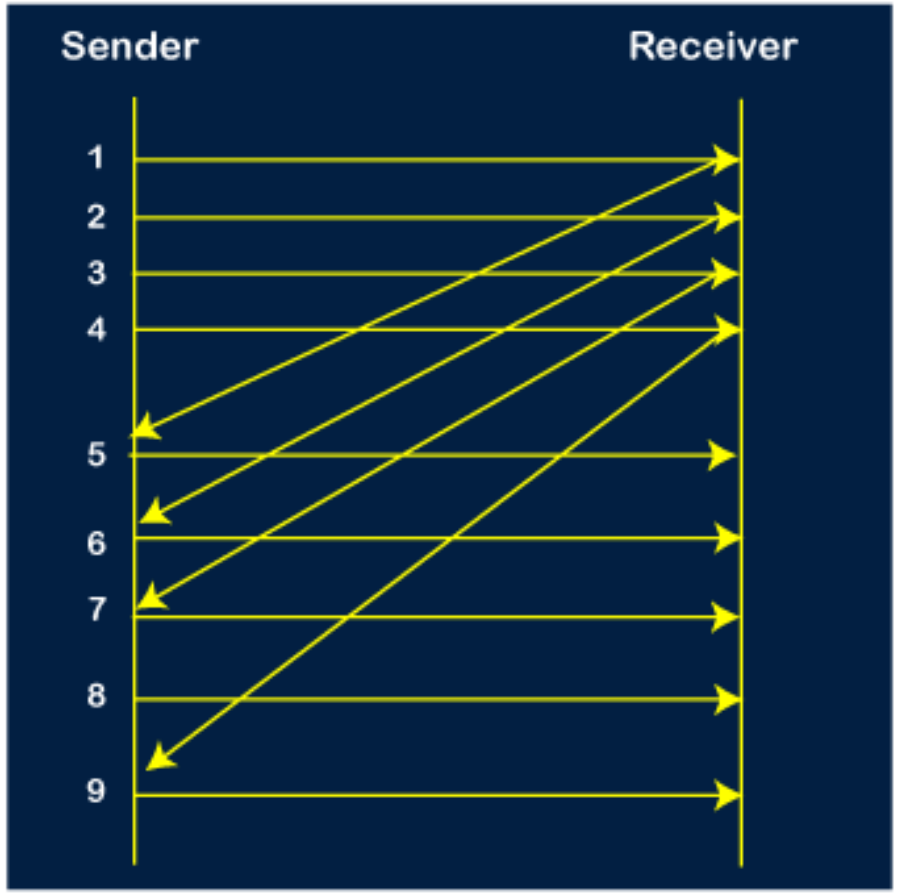
Step 4: The sender receives the acknowledgment for the packet no 3. After receiving the acknowledgment of 3rd packet, the sender sends the next packet, i.e., 7th packet. The window will slide having four packets, i.e., 4, 5, 6, 7.



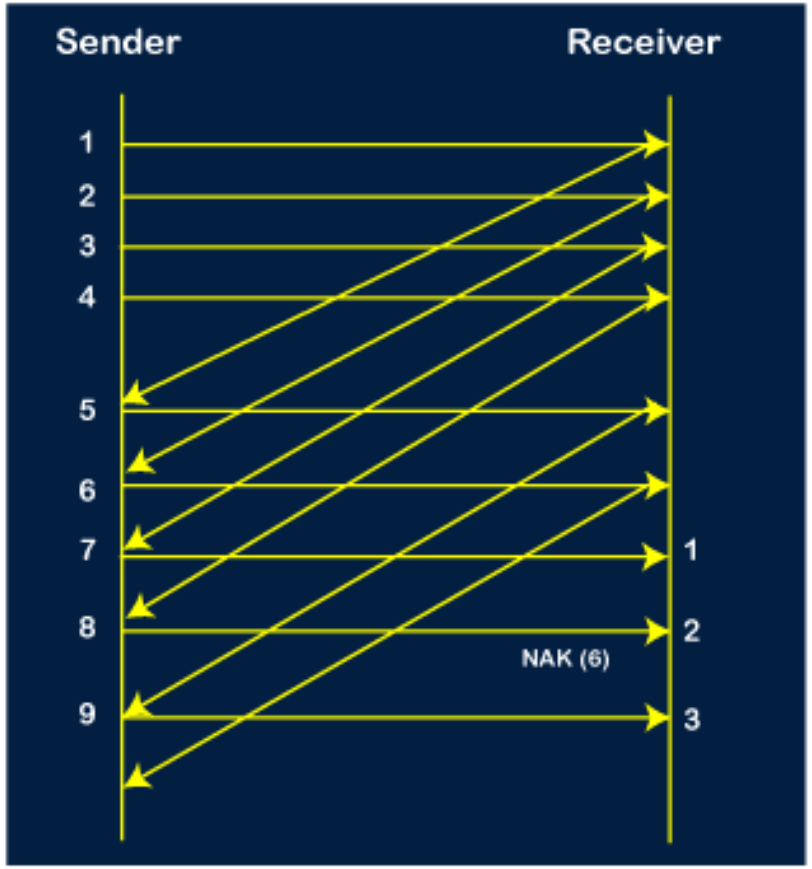
Step 5: When the packet 7 has been sent, then the sender receives the acknowledgment for the packet no 4. When the sender has received the acknowledgment, then the sender sends the next packet, i.e., the 8^{th} packet. The window will slide having four packets, i.e., 5, 6, 7, 8.



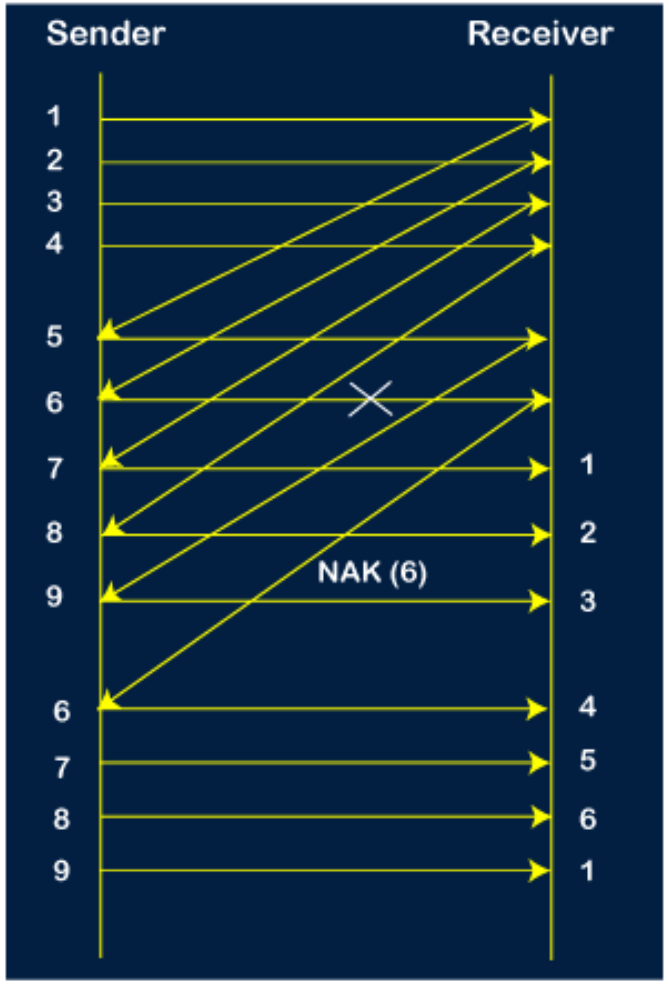
Step 6: When the packet 8 is sent, then the sender receives the acknowledgment of packet 5. On receiving the acknowledgment of packet 5, the sender sends the next packet, i.e., 9^{th} packet. The window will slide having four packets, i.e., 6, 7, 8, 9.



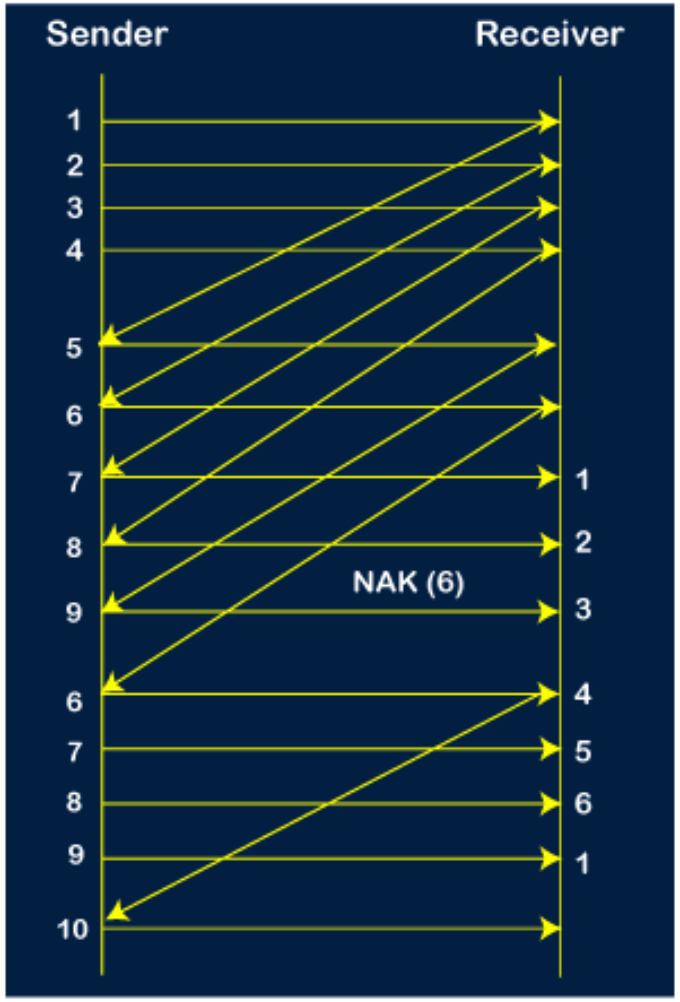
Step 7: The current window is holding four packets, i.e., 6, 7, 8, 9, where the 6^{th} packet is the first packet in the window. As we know, the 6^{th} packet has been lost, so the sender receives the negative acknowledgment NAK(6). As we know that every 6^{th} packet is being lost, so the counter will be restarted from 1. So, the counter values 1, 2, 3 are given to the 7^{th} packet, 8^{th} packet, 9^{th} packet respectively.



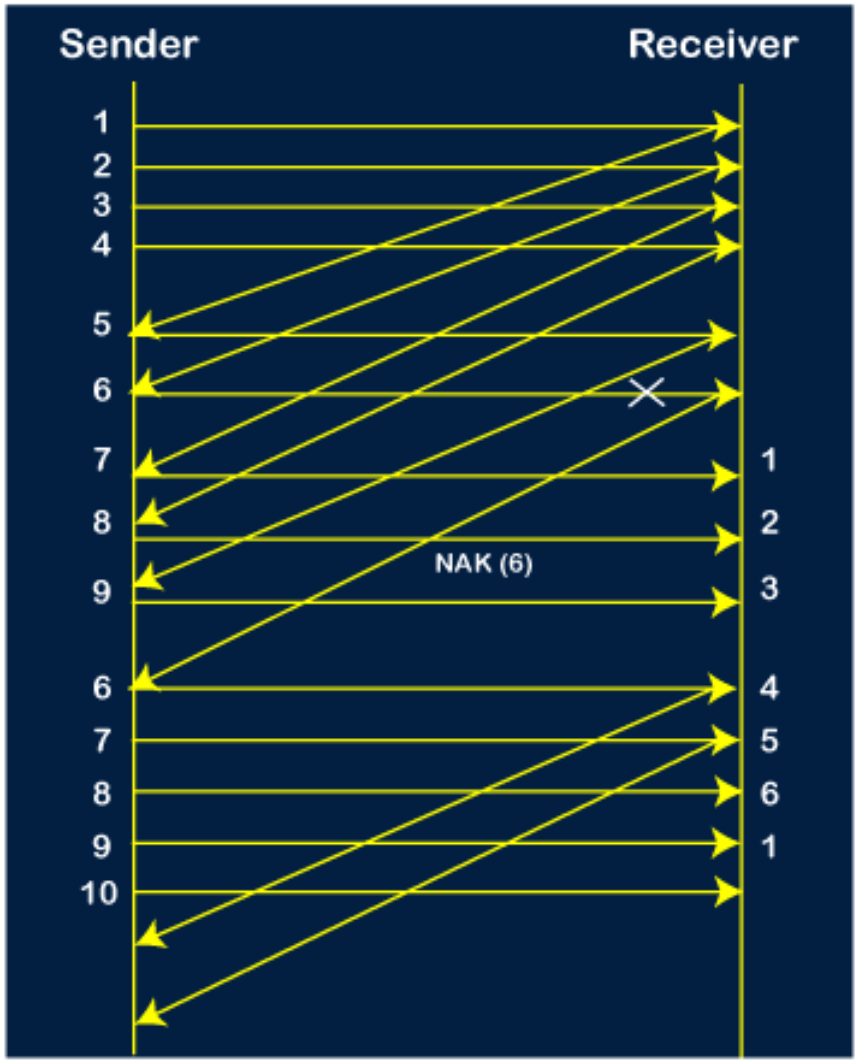
Step 8: As it is Go-BACK, so it retransmits all the packets of the current window. It will resend 6, 7, 8, 9. The counter values of 6, 7, 8, 9 are 4, 5, 6, 1, respectively. In this case, the 8^{th} packet is lost as it has a 6-counter value, so the counter variable will again be restarted from 1.



Step 9: After the retransmission, the sender receives the acknowledgment of packet 6. On receiving the acknowledgment of packet 6, the sender sends the 10^{th} packet. Now, the current window is holding four packets, i.e., 7, 8, 9, 10.

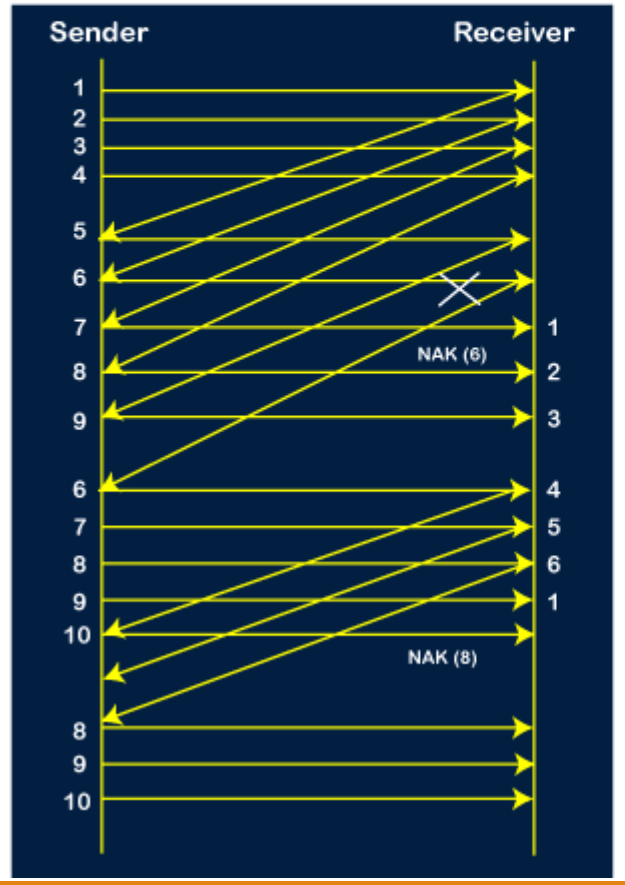


Step 10: When the 10^{th} packet is sent, the sender receives the acknowledgment of packet 7. Now the current window is holding three packets, 8, 9 and 10. The counter values of 8, 9, 10 are 6, 1, 2.



Step 11: As the 8^{th} packet has 6 counter value which means that 8^{th} packet has been lost, and the sender receives NAK (8).

Step 12: Since the sender has received the negative acknowledgment for the 8^{th} packet, it resends all the packets of the current window, i.e., 8, 9, 10.



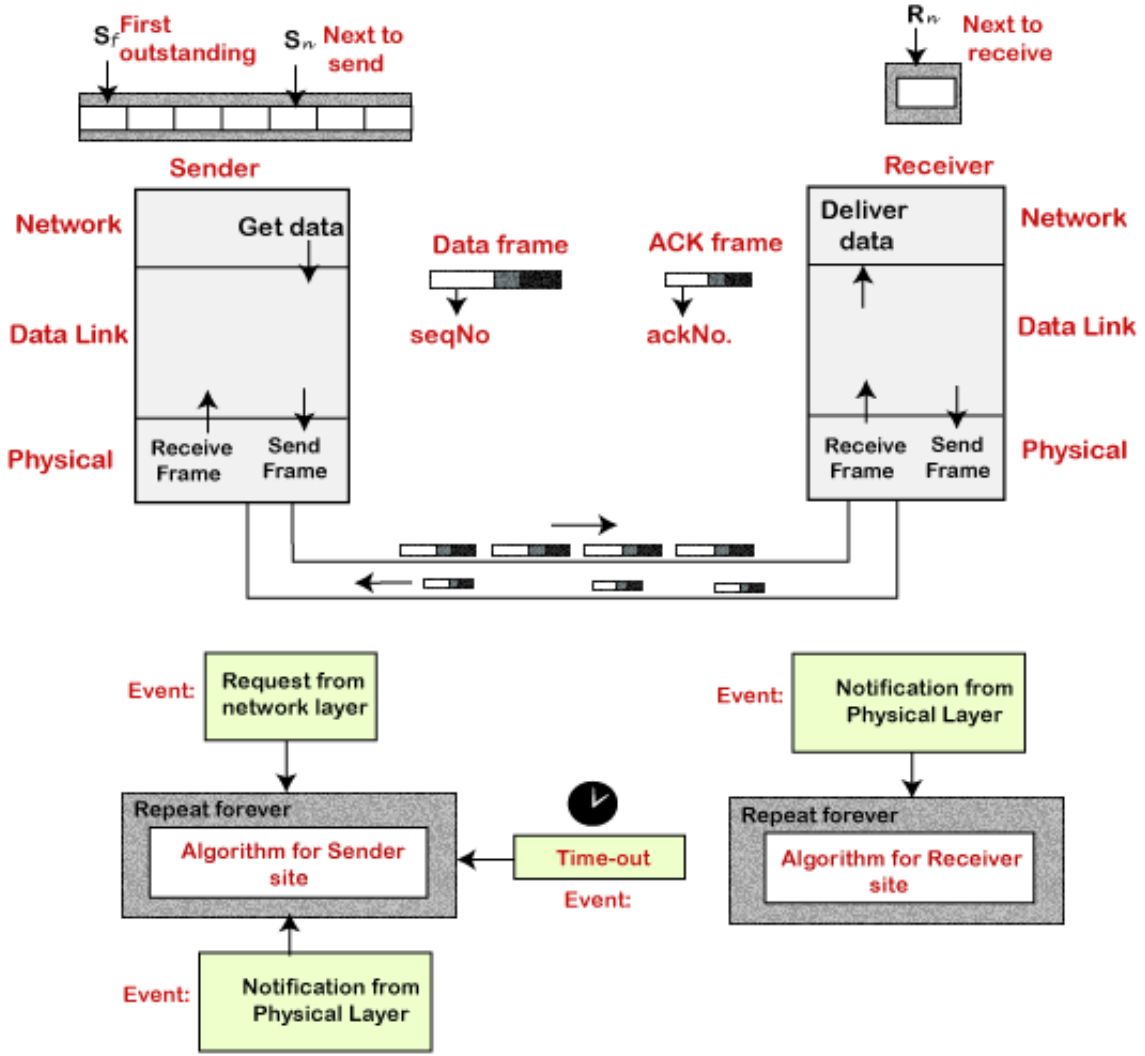
Step 13: The counter values of 8, 9, 10 are 3, 4, 5, respectively, so their acknowledgments have been received successfully.

Summary of Go-Back-N ARQ

Go-Back-N ARQ protocol is also known as Go-Back-N Automatic Repeat Request. It is a data link layer protocol that uses a sliding window method. In this, if any frame is corrupted or lost, all subsequent frames have to be sent again.

The size of the sender window is N in this protocol. For example, Go-Back-8, the size of the sender window, will be 8. The receiver window size is always 1.

If the receiver receives a corrupted frame, it cancels it. The receiver does not accept a corrupted frame. When the timer expires, the sender sends the correct frame again. The design of the Go-Back-N ARQ protocol is shown below.



Selective Repeat ARQ

Selective Repeat ARQ (Selective Repeat Automatic Repeat Request) is another name for Selective Repeat ARQ.

A sliding window method is used in this data link layer protocol. If the frame has fewer errors, Go-Back-N ARQ works well.

However, if the frame contains a lot of errors, sending the frames again will result in a lot of bandwidth loss.

As a result, we employ the Selective Repeat ARQ method.

The size of the sender window is always equal to the size of the receiver window in this protocol.

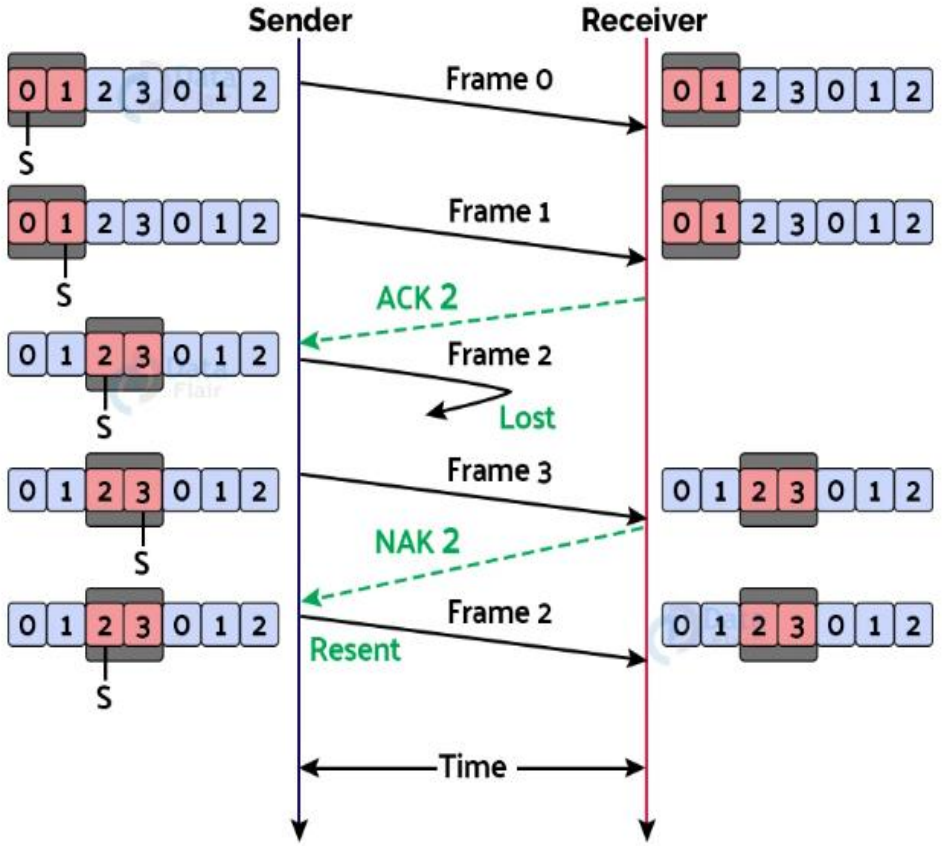
The sliding window's size is always greater than 1.

Example of Selective Repeat ARQ:

a. First, the sender sends the contents of the first window, which are frames 0 and 1 (because the window size is 2).

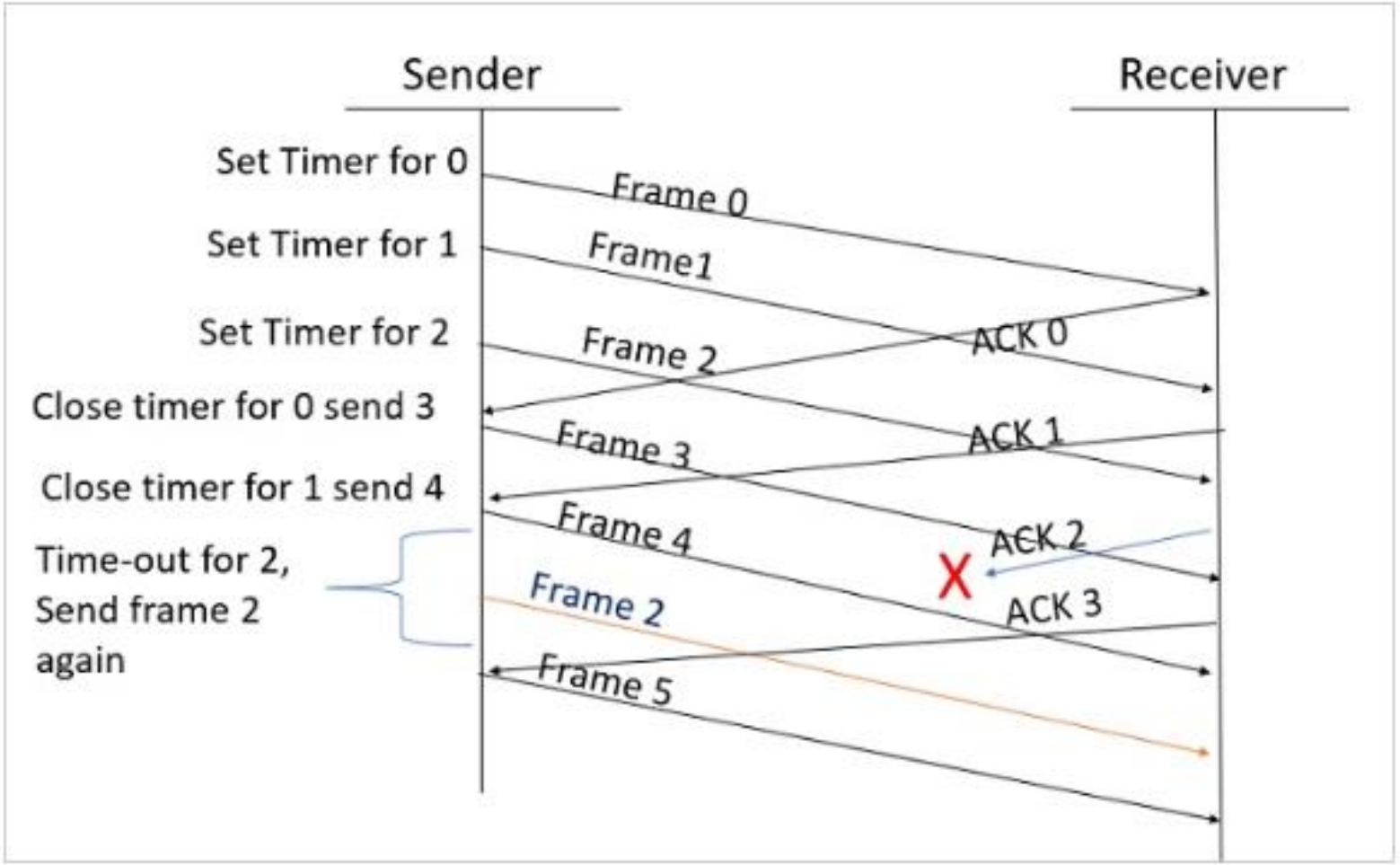
b. When the receiver receives the frames sent above, it sends an acknowledgment for frame 2 (because frame 2 is the frame it expects to receive next).

c. The sender then sends frames 2 and 3, however, frame 2 is lost on the way. The receiver thus sends back a “NAK” signal or a non-acknowledgment to let the sender know that frame 2 has been lost, and thus the sender retransmits frame 2.



Example

Given below is an example of the Selective Repeat ARQ –



Difference between the Go-Back-N ARQ and Selective Repeat ARQ

Go-Back-N ARQ	Selective Repeat ARQ
If a frame is corrupted or lost in it,all subsequent frames have to be sent again.	In this, only the frame is sent again, which is corrupted or lost.
If it has a high error rate,it wastes a lot of bandwidth.	There is a loss of low bandwidth.
It is less complex.	It is more complex because it has to do sorting and searching as well. And it also requires more storage.
It does not require sorting.	In this, sorting is done to get the frames in the correct order.
It does not require searching.	The search operation is performed in it.
It is used more.	It is used less because it is more complex.

Advantages of Sliding Window Protocols:

1. **Efficiency:** The sliding window protocol is an efficient method of transmitting data across a network because it allows multiple packets to be transmitted at the same time. This increases the overall throughput of the network.
2. **Reliable:** The protocol ensures reliable delivery of data, by requiring the receiver to acknowledge receipt of each packet before the next packet can be transmitted. This helps to avoid data loss or corruption during transmission.
3. **Flexibility:** The sliding window protocol is a flexible technique that can be used with different types of network protocols and topologies, including wireless networks, Ethernet, and IP networks.
4. **Congestion Control:** The sliding window protocol can also help control network congestion by adjusting the size of the window based on the network conditions, thereby preventing the network from becoming overwhelmed with too much traffic.
5. Piggybacking with full-duplex lines is possible.

Disadvantages of Sliding Window Protocols:

1. **Complexity:** The sliding window protocol can be complex to implement and can require a lot of memory and processing power to operate efficiently.
2. **Delay:** The protocol can introduce a delay in the transmission of data, as each packet must be acknowledged before the next packet can be transmitted. This delay can increase the overall latency of the network.
3. **Limited Bandwidth Utilization:** The sliding window protocol may not be able to utilize the full available bandwidth of the network, particularly in high-speed networks, due to the overhead of the protocol.
4. **Window Size Limitations:** The maximum size of the sliding window can be limited by the size of the receiver's buffer or the available network resources, which can affect the overall performance of the protocol.

What is Piggybacking in Networking?

In reliable full - duplex data transmission, the technique of hooking up acknowledgments onto outgoing data frames is called piggybacking.

Communications are mostly full – duplex in nature, i.e. data transmission occurs in both directions. A method to achieve full – duplex communication is to consider both the communication as a pair of simplex communication. Each link comprises a forward channel for sending data and a reverse channel for sending acknowledgments. However, in the above arrangement, traffic load doubles for each data unit that is transmitted. Half of all data transmission comprise of transmission of acknowledgments.

So, a solution that provides better utilization of bandwidth is piggybacking. Here, sending of acknowledgment is delayed until the next data frame is available for transmission. The acknowledgment is then hooked onto the outgoing data frame. The data frame consists of an *ack* field. The size of the *ack* field is only a few bits, while an acknowledgment frame comprises of several bytes. Thus, a substantial gain is obtained in reducing bandwidth requirement.

Suppose that there are two communication stations X and Y. The data frames transmitted have an acknowledgment field, *ack* field that is of a few bits length. Additionally, there are frames for sending acknowledgments, ACK frames. The purpose is to minimize the ACK frames.

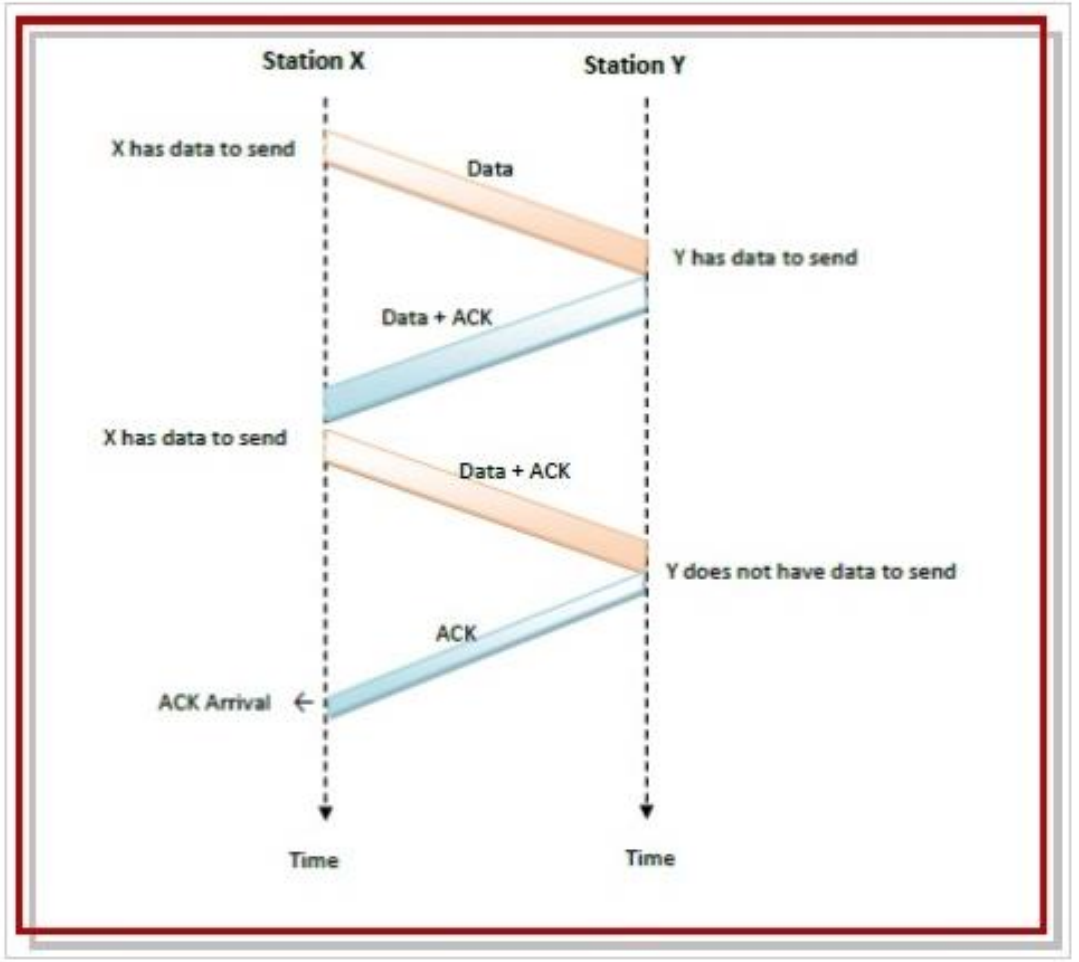
Working Principle

The three principles governing piggybacking when the station wants to communicate with station Y are −

If station X has both data and acknowledgment to send, it senc a data frame with the *ack* field containing the sequence numbe of the frame to be acknowledged.

If station X has only an acknowledgment to send, it waits for finite period of time to see whether a data frame is available t be sent. If a data frame becomes available, then it piggybacks th acknowledgment with it. Otherwise, it sends an ACK frame.

If station X has only a data frame to send, it adds the las acknowledgment with it. The station Y discards all duplicat acknowledgments. Alternatively, station X may send the dat frame with the *ack* field containing a bit combination denotin no acknowledgment.



Advantages of Piggybacking

1. The major advantage of piggybacking is the better use of available channel bandwidth. This happens because an acknowledgment frame needs not to be sent separately.
2. Usage cost reduction.
3. Improves latency of data transfer.
4. To avoid the delay and rebroadcast of frame transmission, piggybacking uses a very short-duration timer.

Disadvantages of Piggybacking

The disadvantage of piggybacking is the additional complexity.

If the data link layer waits long before transmitting the acknowledgment (blocks the ACK for some time), the frame will rebroadcast.

Multiple Access Protocols in Computer Network

DR. NAVNEET KAUR

Data Link Layer

The data link layer is used in a computer network to transmit the data between two devices or nodes. It divides the layer into parts such as **data link control** and the **multiple access resolution/protocol**.

The upper layer has the responsibility to flow control and the error control in the data link layer, and hence it is termed as **logical of data link control**.

Whereas the lower sub-layer is used to handle and reduce the collision or multiple access on a channel. Hence it is termed as media access control or the multiple access resolutions.

Data Link control

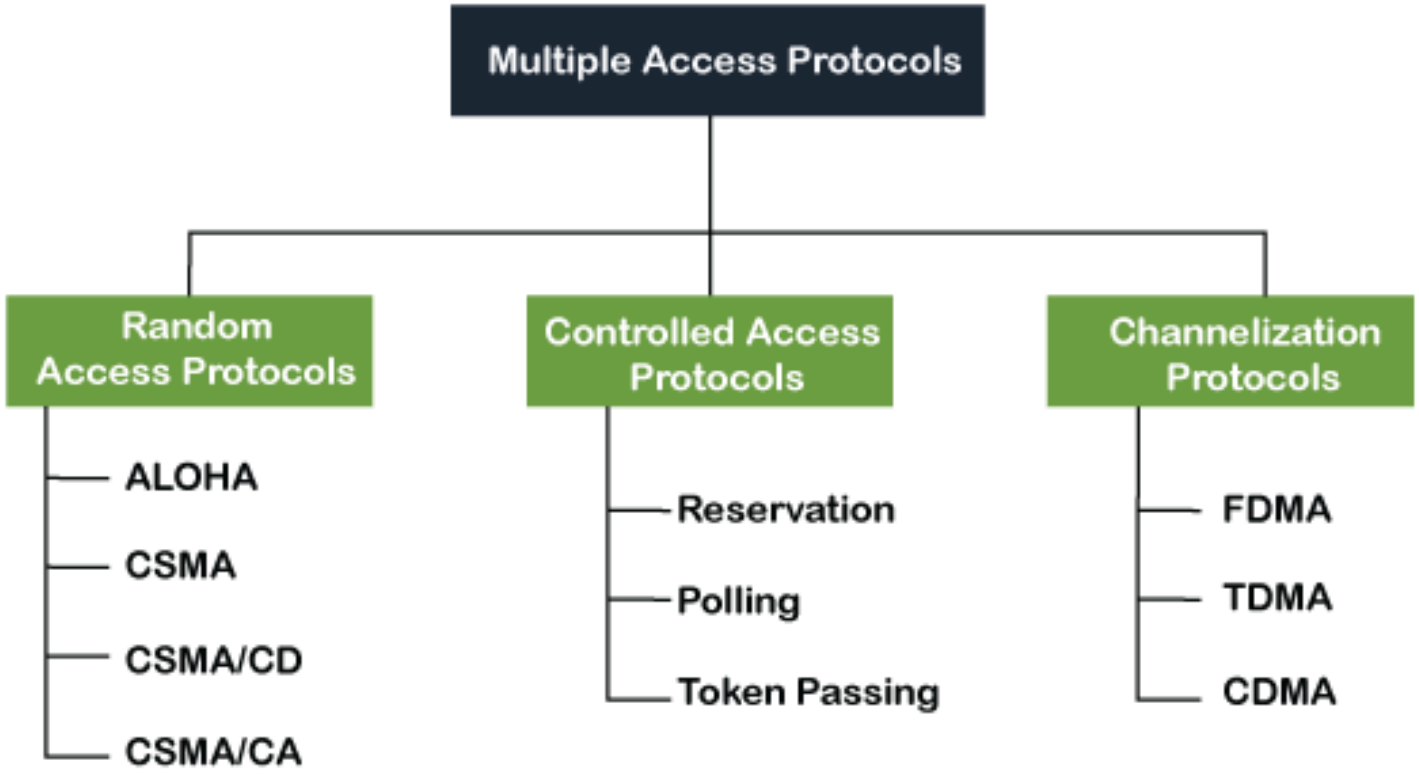
The data link control is responsible for reliable transmission of message over transmission channel by using techniques like framing, error control and flow control.

What is a multiple access protocol?

When a sender and receiver have a dedicated link to transmit data packets, the data link control is enough to handle the channel. Suppose there is no dedicated path to communicate or transfer the data between two devices. In that case, multiple stations access the channel and simultaneously transmits the data over the channel. It may create collision and cross talk. Hence, **the multiple access protocol is required to reduce the collision and avoid crosstalk between the channels.**

For example, suppose that there is a classroom full of students. When a teacher asks a question, all the students (small channels) in the class start answering the question at the same time (transferring the data simultaneously). All the students respond at the same time due to which data is overlap or data lost. Therefore it is the responsibility of a teacher (multiple access protocol) to manage the students and make them one answer.

Types of Multiple access protocols



A. Random Access Protocol

- In this protocol, all the station has the equal priority to send the data over a channel. In random access protocol, one or more stations cannot depend on another station nor any station control another station.
- Depending on the channel's state (idle or busy), each station transmits the data frame. However, if more than one station sends the data over a channel, there may be a collision or data conflict.
- Due to the collision, the data frame packets may be lost or changed. And hence, it does not receive by the receiver end.
- Following are the different methods of random-access protocols for broadcasting frames on the channel.
  - Aloha
  - CSMA
  - CSMA/CD
  - CSMA/CA

ALOHA Random Access Protocol

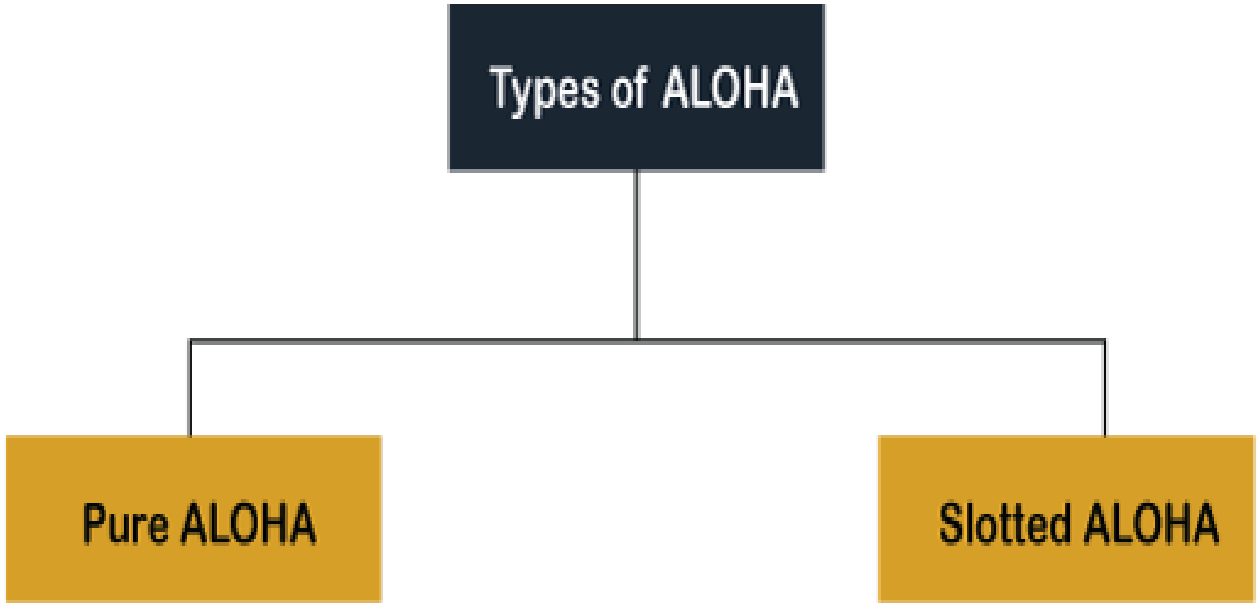
- It is designed for wireless LAN (Local Area Network) but can also be used in a shared medium to transmit data.

- Using this method, any station can transmit data across a network simultaneously when a data frameset is available for transmission.

Aloha Rules

- Any station can transmit data to a channel at any time.
- It does not require any carrier sensing.
- Collision and data frames may be lost during the transmission of data through multiple stations.
- Acknowledgment of the frames exists in Aloha. Hence, there is no collision detection.
- It requires retransmission of data after some random amount of time.

ALOHA Random Access Protocol



Pure ALOHA

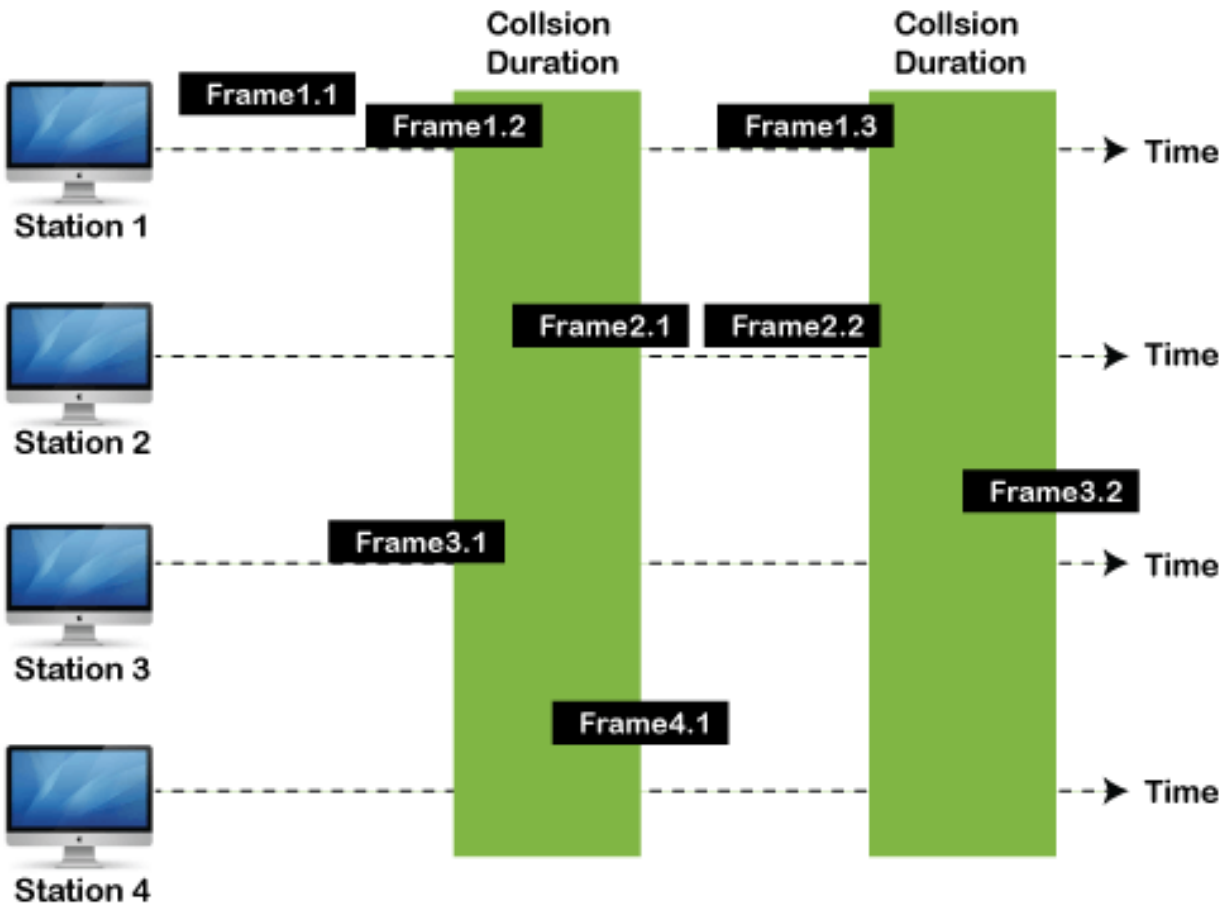
Whenever data is available for sending over a channel at stations, we use Pure Aloha. In pure Aloha, when each station transmits data to a channel without checking whether the channel is idle or not, the chances of collision may occur, and the data frame can be lost. When any station transmits the data frame to a channel, the pure Aloha waits for the receiver's acknowledgment. If it does not acknowledge the receiver end within the specified time, the station waits for a random amount of time, called the backoff time (Tb). And the station may assume the frame has been lost or destroyed. Therefore, it retransmits the frame until all the data are successfully transmitted to the receiver. (Suppose G is the average number of frames created by the system during a single-frame transmission period.)

The total vulnerable time of pure Aloha is 2 * Tfr (Frame transmission time).

Maximum throughput occurs when G = 1/ 2 that is 18.4%.

Successful transmission of data frame is S = G * e ^ - 2 G.

As we can see in the figure above, there are four stations for accessing a shared channel and transmitting data frames. Some frames collide because most stations send their frames at the same time. Only two frames, frame 1.1 and frame 3.2, are successfully transmitted to the receiver end. At the same time, other frames are lost or destroyed. Whenever two frames fall on a shared channel simultaneously, collisions can occur, and both will suffer damage. If the new frame's first bit enters the channel before finishing the last bit of the second frame. Both frames are completely finished, and both stations must retransmit the data frame.



Frames in Pure ALOHA

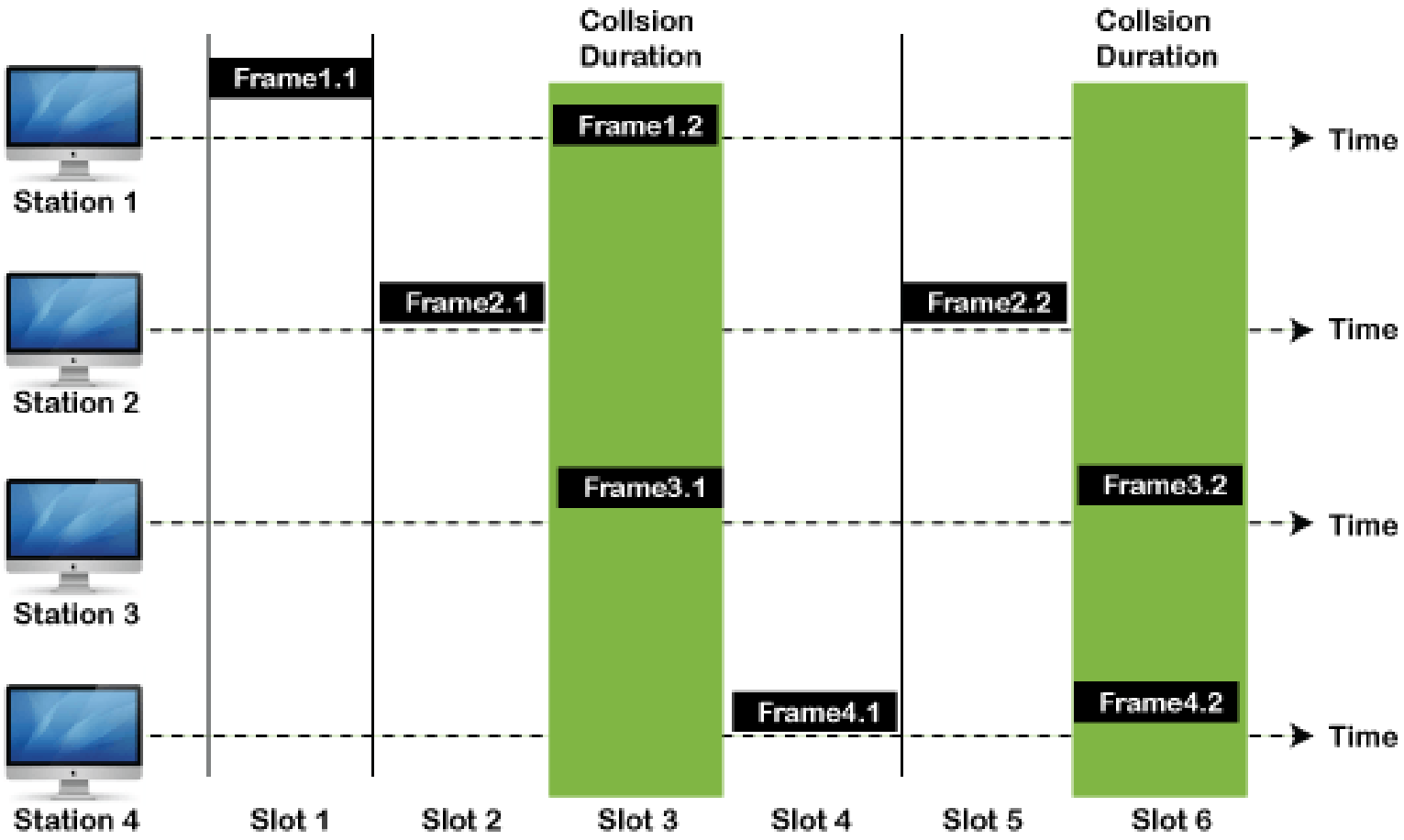
Slotted ALOHA

The slotted Aloha is designed to overcome the pure Aloha's defiency because pure Aloha has a very high possibility of frame hitting. In slotted Aloha, the shared channel is divided into a fixed time interval called **slots**. So that, if a station wants to send a frame to a shared channel, the frame can only be sent at the beginning of the slot, and only one frame is allowed to be sent to each slot. And if the stations are unable to send data to the beginning of the slot, the station will have to wait until the beginning of the slot for the next time. However, the possibility of a collision remains when trying to send a frame at the beginning of two or more station time slot.

Maximum throughput occurs in the slotted Aloha when G = 1 that is 37%.

The probability of successfully transmitting the data frame in the slotted Aloha is S = G * e ^ - 2 G.

The total vulnerable time required in slotted Aloha is Tfr.

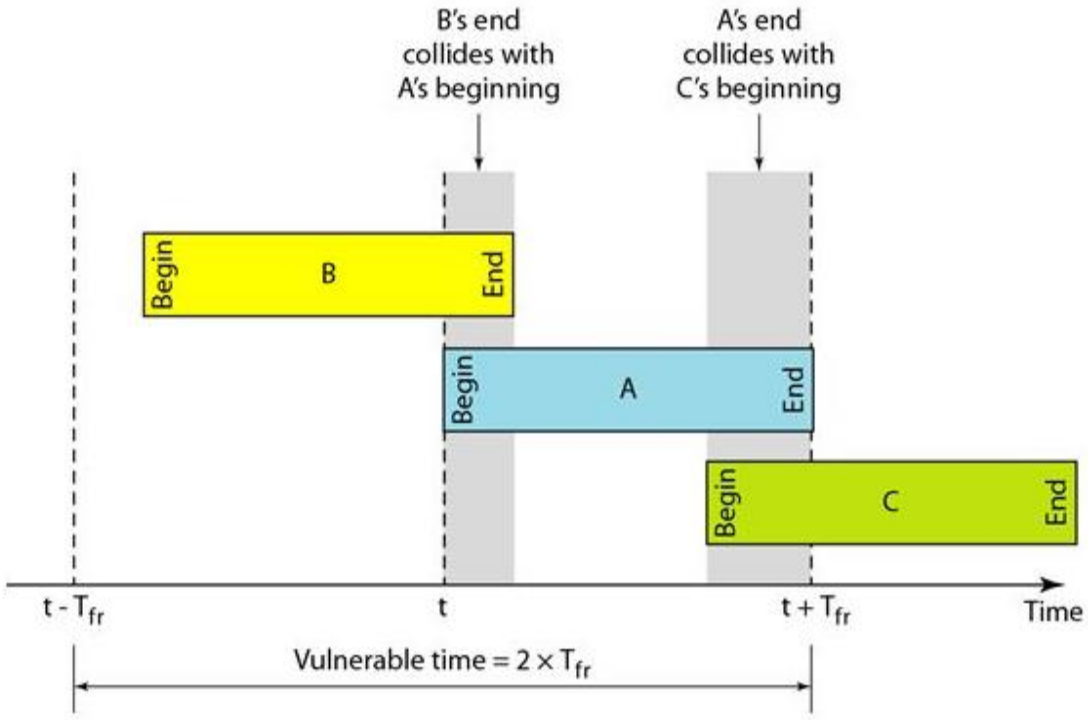


Frames in Slotted ALOHA

Vulnerable time:

The vulnerable time is in which there is a possibility of collision. We assume that the stations send fixed-length frames with each frame taking Tfr S to send. The following figure shows the vulnerable time for station A.

Station A sends a frame at time t. Now imagine station B has already sent a frame between t - Tfr and t. This leads to a collision between the frames from station A and station B. The end of B's frame collides with the beginning of A's frame. On the other hand, suppose that station C sends a frame between t and t + Tfr . Here, there is a collision between frames from station A and station C. The beginning of C's frame collides with the end of A's frame.



Key	Pure Aloha	Slotted Aloha
Time Slot	In Pure Aloha, any station can transmit data at any time.	In Slotted Aloha, any station can transmit data only at the beginning of a time slot.
Time	In Pure Aloha, time is continuous and is not globally synchronized.	In Slotted Aloha, time is discrete and is globally synchronized.
Vulnerable time	The vulnerable time or susceptible time in Pure Aloha is equal to (2×Tt).	In Slotted Aloha, the vulnerable time is equal to (Tt).
Probability	The probability of successful transmission of a data packet $S = G \times e^{-2G}{}_G$	The probability of successful transmission of data packet $S = G \times e^{-G}$
Maximum efficiency	Maximum efficiency = 18.4%.	Maximum efficiency = 36.8%.
Number of collisions	Does not reduce the number of collisions.	Slotted Aloha reduces the number of collisions to half, thus doubles the efficiency.

CSMA (Carrier Sense Multiple Access)

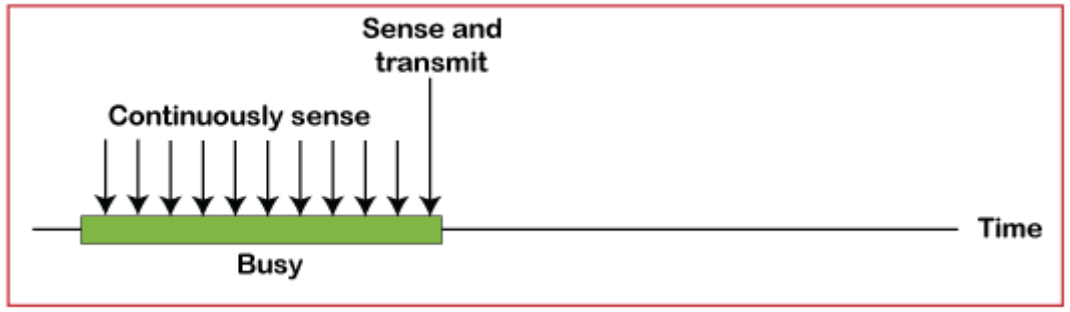
It is a **carrier sense multiple access** based on media access protocol to sense the traffic on a channel (idle or busy) before transmitting the data.

It means that if the channel is idle, the station can send data to the channel. Otherwise, it must wait until the channel becomes idle.

Hence, it reduces the chances of a collision on a transmission medium.

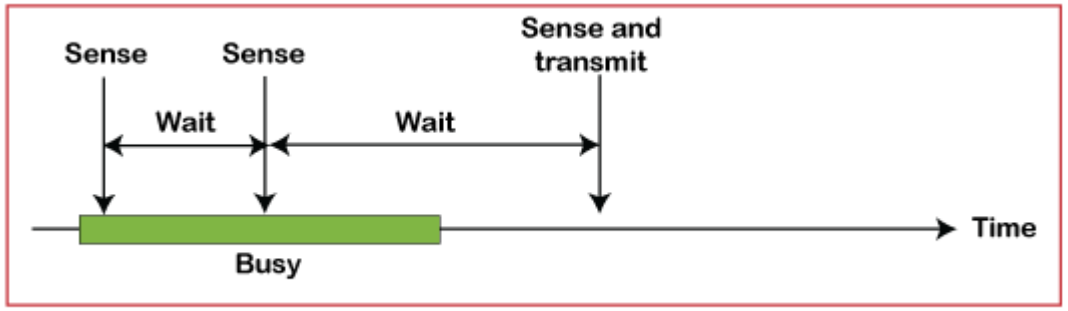
CSMA Access Modes

- **1-Persistent:** In the 1-Persistent mode of CSMA that defines each node, first sense the shared channel and if the channel is idle, it immediately sends the data. Else it must wait and keep track of the status of the channel to be idle and broadcast the frame unconditionally as soon as the channel is idle.



a. 1-persistent

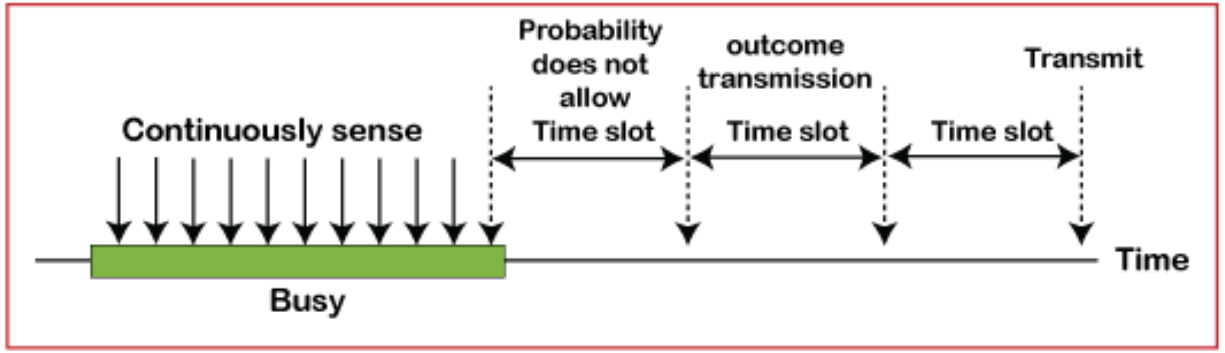
- **Non-Persistent:** It is the access mode of CSMA that defines before transmitting the data, each node must sense the channel, and if the channel is inactive, it immediately sends the data. Otherwise, the station must wait for a random time (not continuously), and when the channel is found to be idle, it transmits the frames.



b. Nonpersistent

CSMA Access Modes

- **P-Persistent:** It is the combination of 1-Persistent and Non-persistent modes. The P-Persistent mode defines that each node senses the channel, and if the channel is inactive, it sends a frame with a **P** probability. If the data is not transmitted, it waits for a (**q = 1-p probability**) random time and resumes the frame with the next time slot.



c. p-persistent

- **O- Persistent:** It is an O-persistent method that defines the superiority of the station before the transmission of the frame on the shared channel. If it is found that the channel is inactive, each station waits for its turn to retransmit the data.

CSMA/ CD

-It is a **carrier sense multiple access/ collision detection** network protocol to transmit data frames.

-The CSMA/CD protocol works with a medium access control layer. Therefore, it first senses the shared channel before broadcasting the frames, and if the channel is idle, it transmits a frame to check whether the transmission was successful.

-If the frame is successfully received, the station sends another frame.

-If any collision is detected in the CSMA/CD, the station sends a jam/ stop signal to the shared channel to terminate data transmission.

-After that, it waits for a random time before sending a frame to a channel.

CSMA/ CA

-It is a **carrier sense multiple access/collision avoidance** network protocol for carrier transmission of data frames. It is a protocol that works with a medium access control layer.

-When a data frame is sent to a channel, it receives an acknowledgment to check whether the channel is clear.

-If the station receives only a single (own) acknowledgments, that means the data frame has been successfully transmitted to the receiver.

-But if it gets two signals (its own and one more in which the collision of frames), a collision of the frame occurs in the shared channel.

-Detects the collision of the frame when a sender receives an acknowledgment signal.

Following are the methods used in the CSMA/ CA to avoid the collision:

Interframe space: In this method, the station waits for the channel to become idle, and if it gets the channel is idle, it does not immediately send the data. Instead of this, it waits for some time, and this time period is called the **Interframe** space or IFS. However, the IFS time is often used to define the priority of the station.

Contention window: In the Contention window, the total time is divided into different slots. When the station/ sender is ready to transmit the data frame, it chooses a random slot number of slots as **wait time**. If the channel is still busy, it does not restart the entire process, except that it restarts the timer only to send data packets when the channel is inactive.

Acknowledgment: In the acknowledgment method, the sender station sends the data frame to the shared channel if the acknowledgment is not received ahead of time.

B. Controlled Access Protocol

It is a method of reducing data frame collision on a shared channel. In the controlled access method, each station interacts and decides to send a data frame by a particular station approved by all other stations.

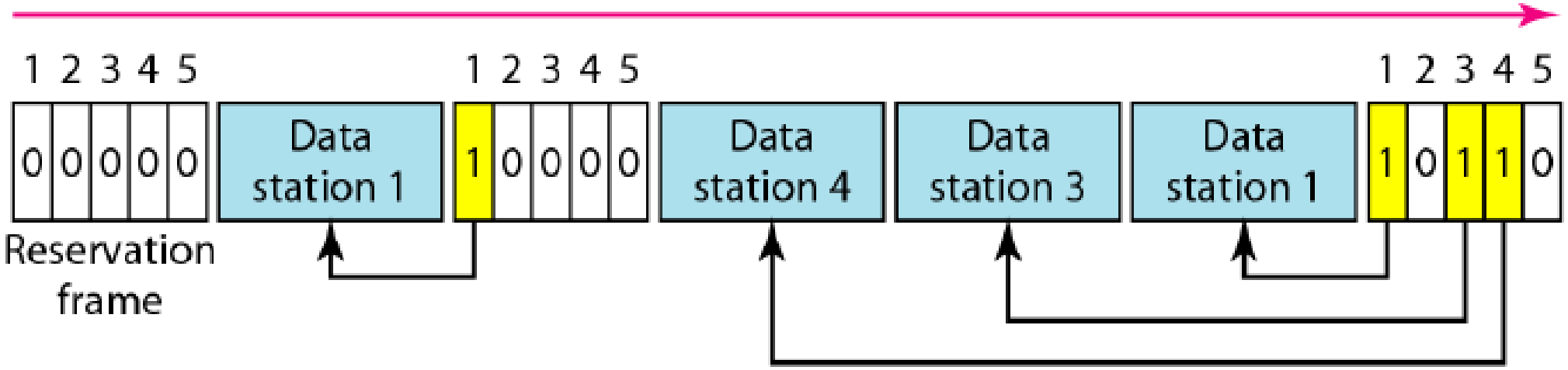
It means that a single station cannot send the data frames unless all other stations are not approved. It has three types of controlled access: **Reservation, Polling**, and **Token Passing**.

Reservation

In the reservation method, a station needs to make a reservation before sending data. Time is divided into intervals. In each interval, a reservation frame precedes the data frames sent in that interval.

If there are *N* stations in the system, there are exactly *N* reservation mini slots in the reservation frame. Each mini slot belongs to a station. When a station needs to send a data frame, it makes a reservation in its own mini slot. The stations that have made reservations can send their data frames after the reservation frame.

Figure shows a situation with five stations and a five-mini slot reservation frame. In the first interval, only stations 1, 3, and 4 have made reservations. In the second interval, only station 1 has made a reservation.



Advantages of Reservation:

- The main advantage of reservation is *high rates and low rates of data accessing* time of the respective channel can be predicated easily. Here time and rates are fixed.
- Priorities can be set to provide speedier access from secondary.
- Predictable network performance: Reservation-based access methods can provide predictable network performance, which is important in applications where latency and jitter must be minimized, such as in real-time video or audio streaming.
- **Reduced contention:** Reservation-based access methods can reduce contention for network resources, as access to the network is pre-allocated based on reservation requests. This can improve network efficiency and reduce packet loss.
- **Quality of Service (QoS) support**: Reservation-based access methods can support QoS requirements, by providing different reservation types for different types of traffic, such as voice, video, or data. This can ensure that high-priority traffic is given preferential treatment over lower-priority traffic.
- **Efficient use of bandwidth**: Reservation-based access methods can enable more efficient use of available bandwidth, as they allow for time and frequency multiplexing of different reservation requests on the same channel.
- **Support for multimedia applications:** Reservation-based access methods are well-suited to support multimedia applications that require guaranteed network resources, such as bandwidth and latency, to ensure high-quality performance.

Disadvantages of Reservation:

- Highly trust on controlled *dependability.*
- *Decrease in capacity* and channel data rate under light loads; increase in turn-around time.

Polling

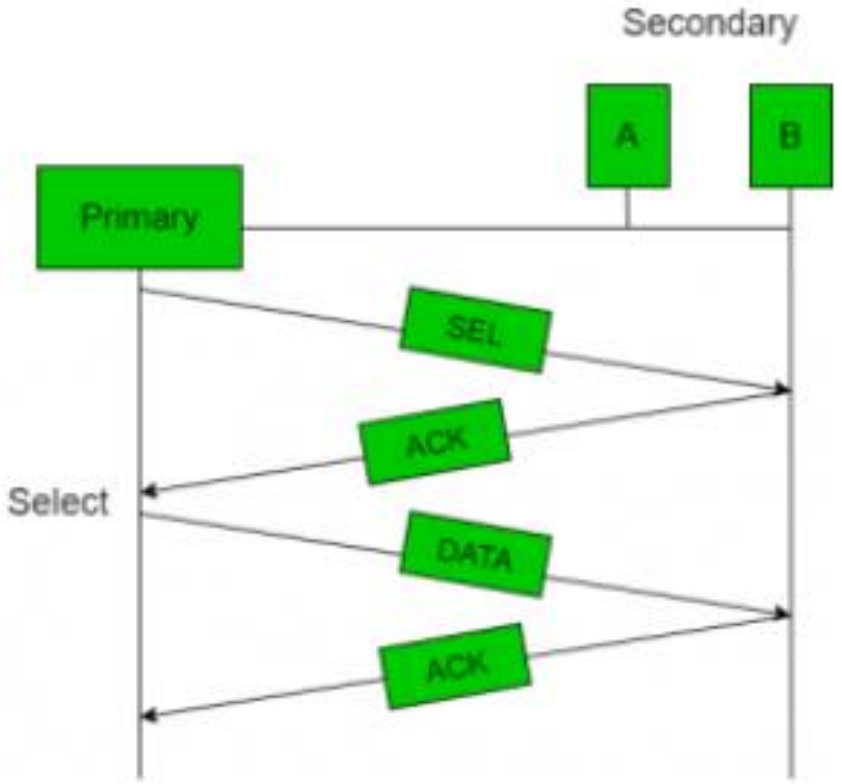
- Polling process is similar to the roll-call performed in class. Just like the teacher, a controller sends a message to each node in turn.
- In this, one acts as a primary station(controller) and the others are secondary stations. All data exchanges must be made through the controller.
- The message sent by the controller contains the address of the node being selected for granting access.
- Although all nodes receive the message the addressed one responds to it and sends data if any. If there is no data, usually a “poll reject”(NAK) message is sent back.
- Problems include high overhead of the polling messages and high dependence on the reliability of the controller.

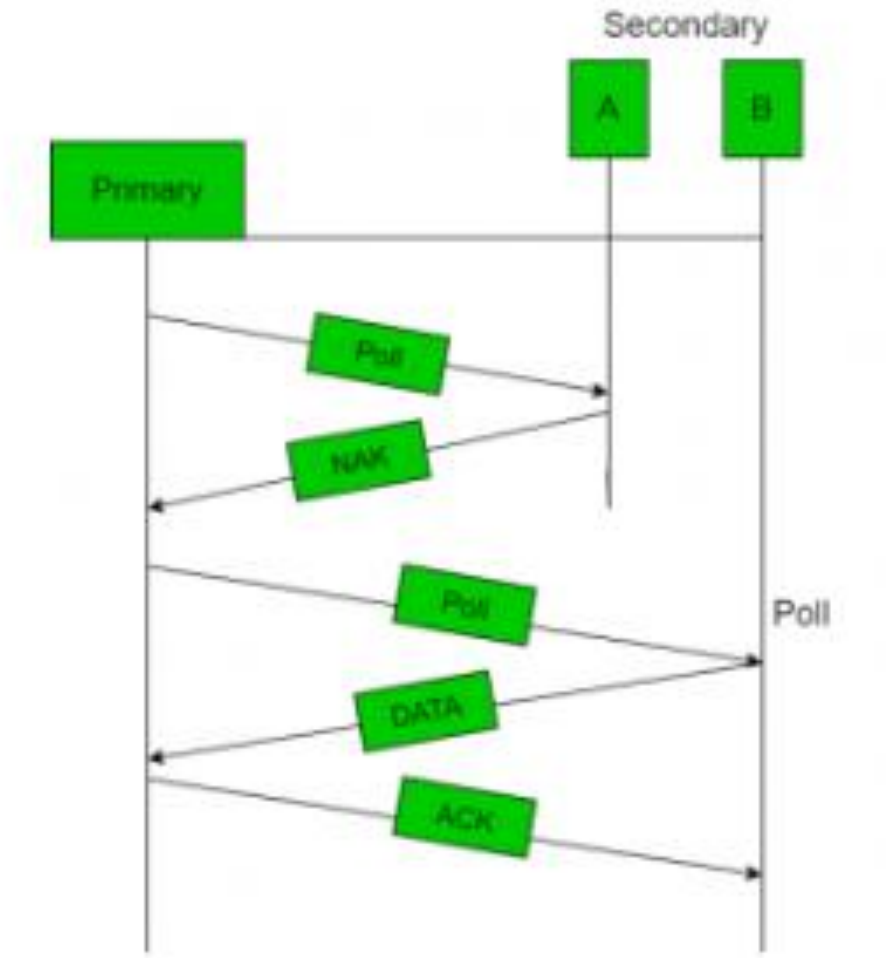
Select

The *select* function is used whenever the primary device has something to send. Remember that the primary controls the link. If the primary is neither sending nor receiving data, it knows the link is available.

If it has something to send, the primary device sends it. What it does not know, however, is whether the **target device is prepared to receive**.

So the primary must **alert the secondary to the upcoming** transmission and **wait for an acknowledgment of the secondary's** ready status. Before sending data, the primary creates and transmits a select (SEL) frame, one field of which includes the address of the intended secondary.



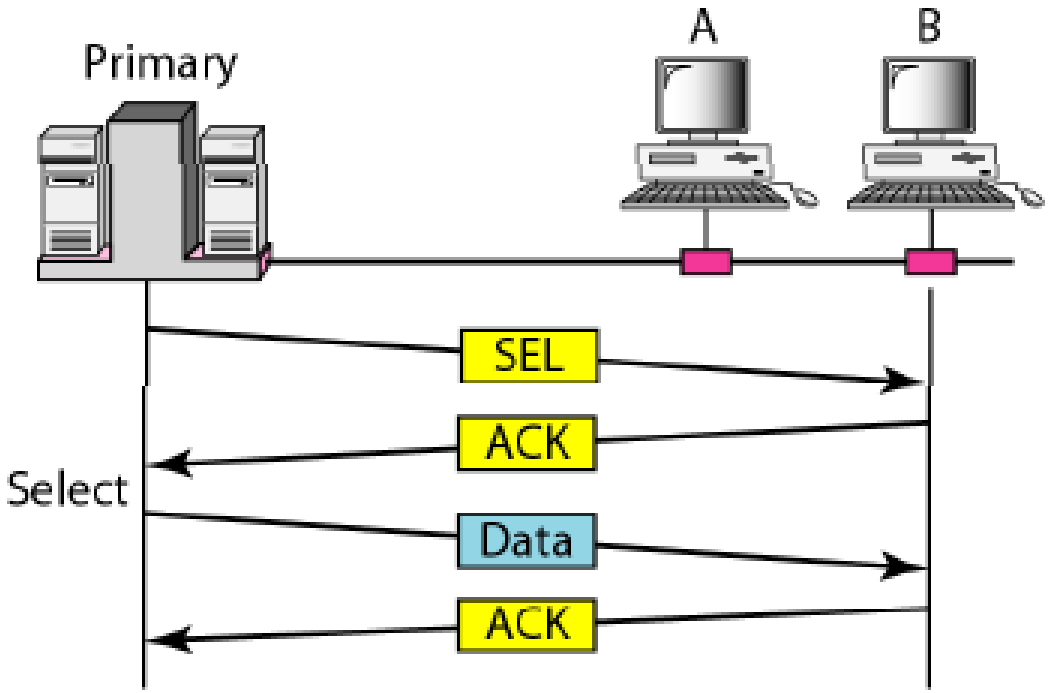


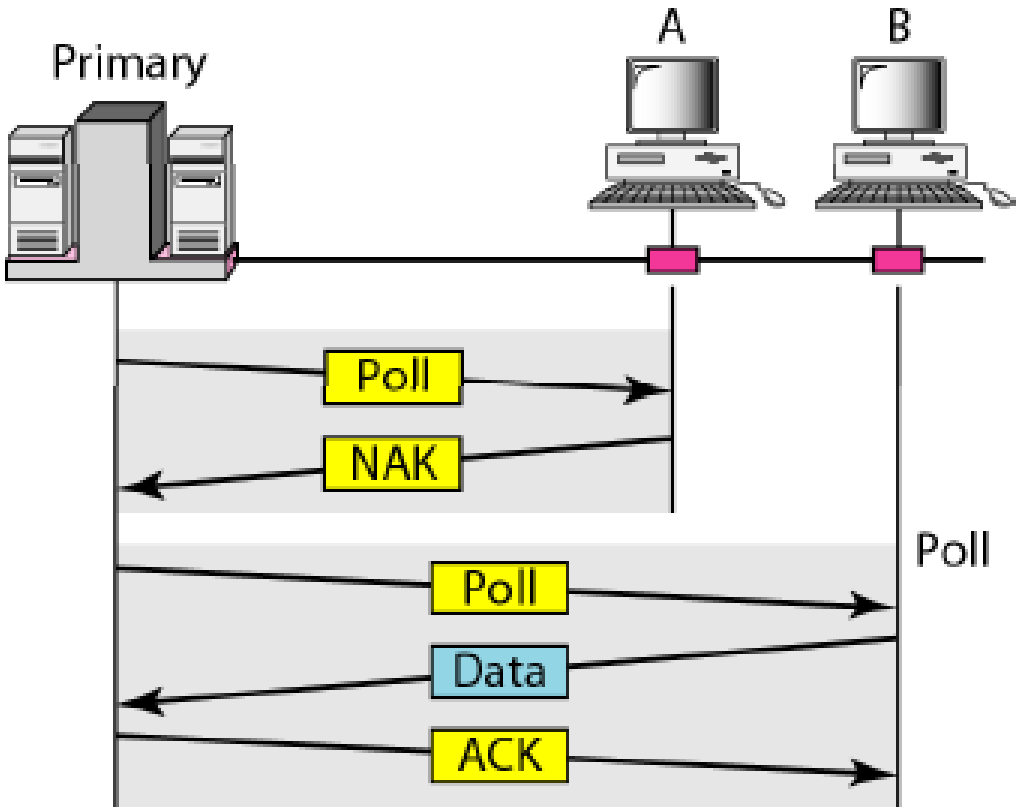
Poll

The *poll* function is used by the primary device to solicit transmissions from the secondary devices.

When the primary is ready to receive data, it must ask (poll) each device in turn if it has anything to send. When the first secondary is approached, it responds either with a NAK frame if it has nothing to send or with data (in the form of a data frame) if it does.

If the response is negative (a NAK frame), then the primary polls the next secondary in the same manner until it finds one with data to send. When the response is positive (a data frame), the primary reads the frame and returns an acknowledgment (ACK frame), verifying its receipt.


Primary
A
B
SEL
ACK
Select
Data
ACK


Primary
A
B
Poll
NAK
Poll
Poll
Data
ACK

Advantages of Polling:

- The maximum and minimum access time and data rates on the channel are fixed predictable.
- It has maximum *efficiency.*
- It has maximum *bandwidth.*
- No slot is wasted in polling.
- There is assignment of priority to ensure faster access from some secondary.

Disadvantages of Polling:

- It consume *more time.*
- Since every station has an equal chance of winning in every round, link sharing is *biased.*
- Only some station might run out of data to send.
- An increase in the turnaround time leads to a drop in the data rates of the channel under low loads.

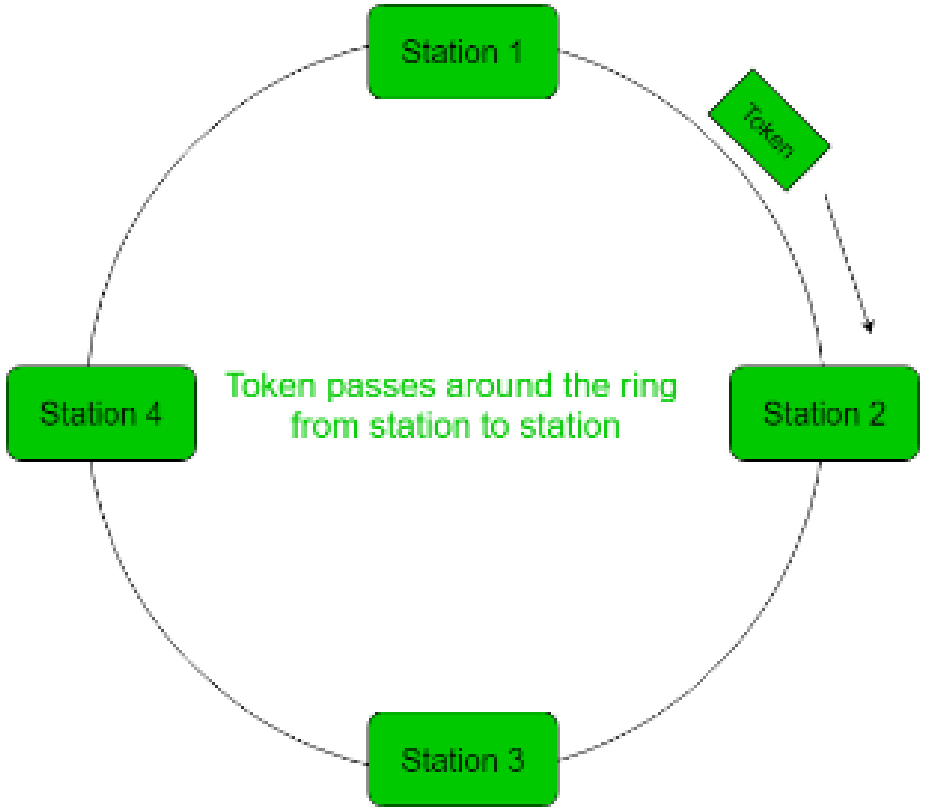
Efficiency Let T_{poll} be the time for polling and T_t be the time required for transmission of data. Then,

Efficiency = $T_t/(T_t + T_{poll})$

Token Passing

- In token passing scheme, the stations are connected logically to each other in form of ring and access to stations is governed by tokens.
- A token is a special bit pattern or a small message, which circulate from one station to the next in some predefined order.
- In Token ring, token is passed from one station to another adjacent station in the ring whereas in case of Token bus, each station uses the bus to send the token to the next station in some predefined order.
- In both cases, token represents permission to send. If a station has a frame queued for transmission when it receives the token, it can send that frame before it passes the token to the next station. If it has no queued frame, it passes the token simply.
- After sending a frame, each station must wait for all N stations (including itself) to send the token to their neighbours and the other N – 1 stations to send a frame, if they have one.

- There exists problems like duplication of token or token is lost or insertion of new station, removal of a station, which need be tackled for correct and reliable operation of this scheme.



Performance of token ring can be concluded by 2 parameters:-

1.Delay, is a measure of time between when a packet is ready and when it is delivered. So, the average time (delay) required to send a token to the next station = a/N.

2.Throughput, which is a measure of successful traffic.
Throughput, $S = 1/(1 + a/N)$ for $a<1$
and
$S = 1/\{a(1 + 1/N)\}$ for $a>1$.
where N = number of stations
$a = T_p/T_t$
(T_p = propagation delay and T_t = transmission delay)

Advantages of Token passing:

- It may now be applied with routers cabling and includes built-in debugging features like *protective relay and auto reconfiguration.*
- It provides *good throughput* when conditions of high load.

Disadvantages of Token passing:

- Its cost is *expensive.*
- Topology components are more expensive than those of other, more widely used standard.
- The hardware element of the token rings are designed to be tricky. This implies that you should choose on manufacture and use them exclusively.

C. Channelization Protocols

It is a channelization protocol that allows the total usable bandwidth in a shared channel to be shared across multiple stations based on their time, distance and codes. It can access all the stations at the same time to send the data frames to the channel.

Following are the various methods to access the channel based on their time, distance and codes:

- FDMA (Frequency Division Multiple Access)
- TDMA (Time Division Multiple Access)
- CDMA (Code Division Multiple Access)

FDMA (Frequency Division Multiple Access)

It is a frequency division multiple access (**FDMA**) method used to divide the available bandwidth into equal bands so that multiple users can send data through a **different frequency** to the subchannel. Each station is reserved with a particular band to prevent the crosstalk between the channels and interferences of stations.

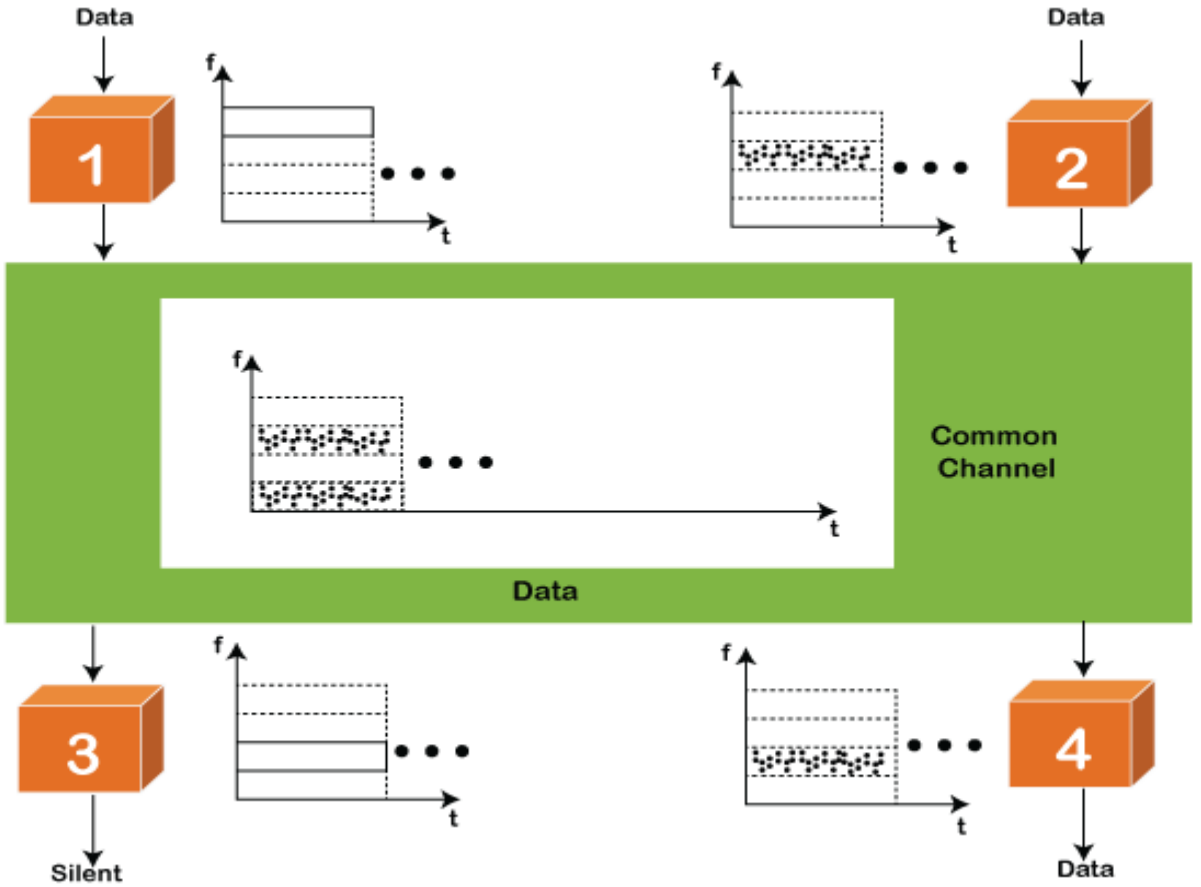
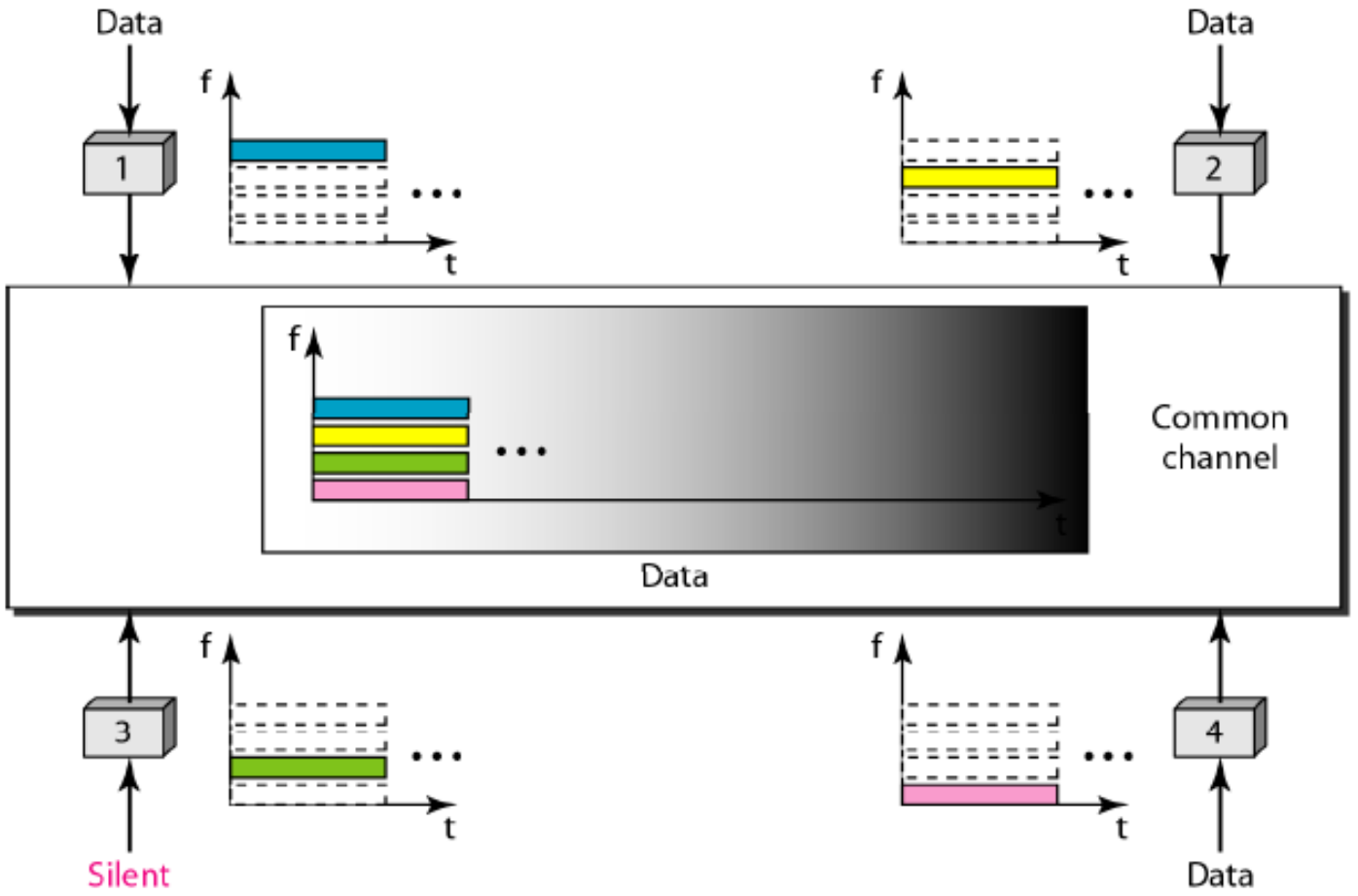


Figure 12.21 *Frequency-division multiple access (FDMA)*



TDMA (Time Division Multiple Access)

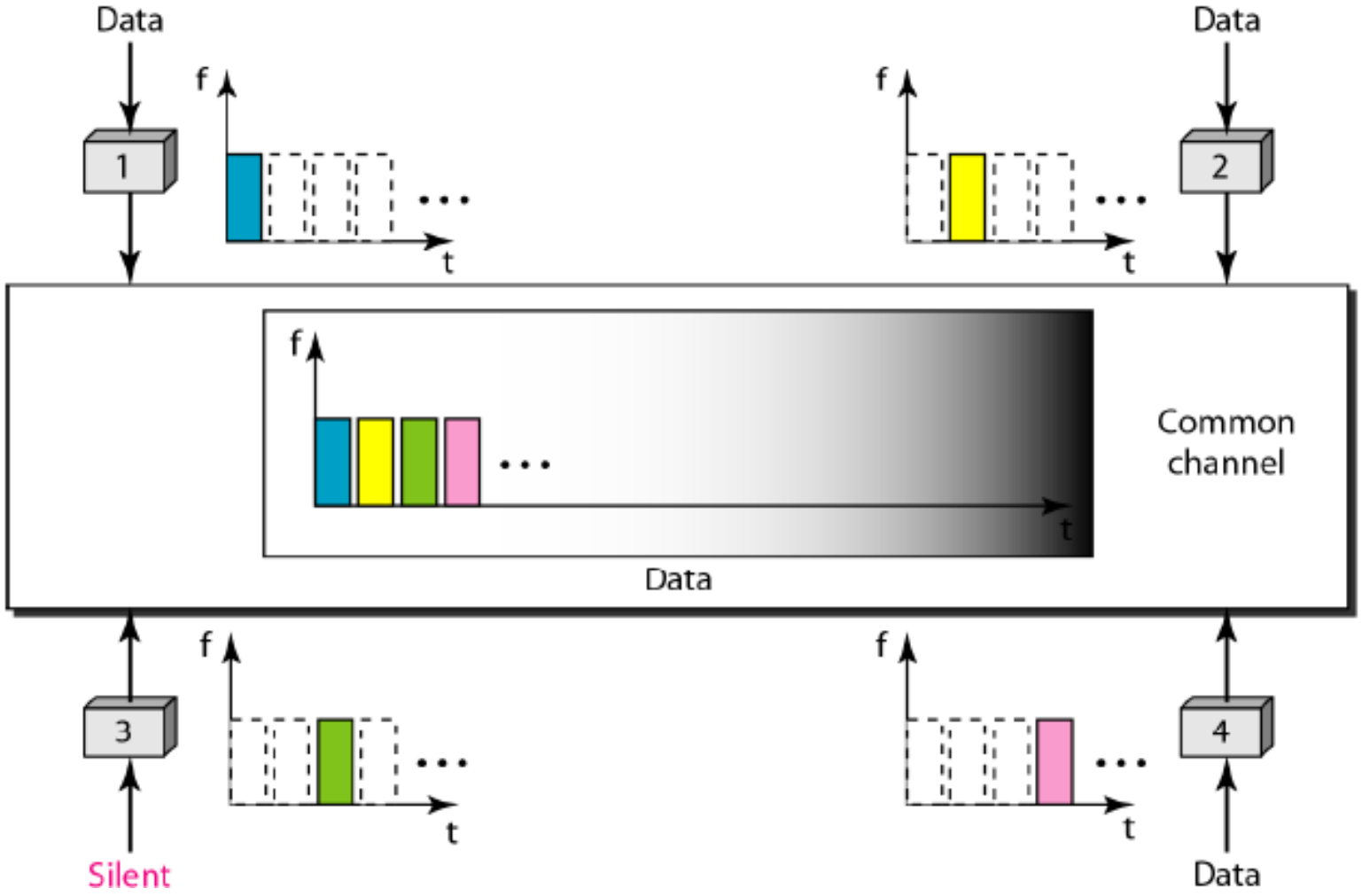
Time Division Multiple Access (**TDMA**) is a channel access method. It allows the same frequency bandwidth to be shared across multiple stations.

And to avoid collisions in the shared channel, it divides the channel into **different frequency slots** that allocate stations to transmit the data frames.

The same **frequency** bandwidth into the shared channel by dividing the signal into various time slots to transmit it.

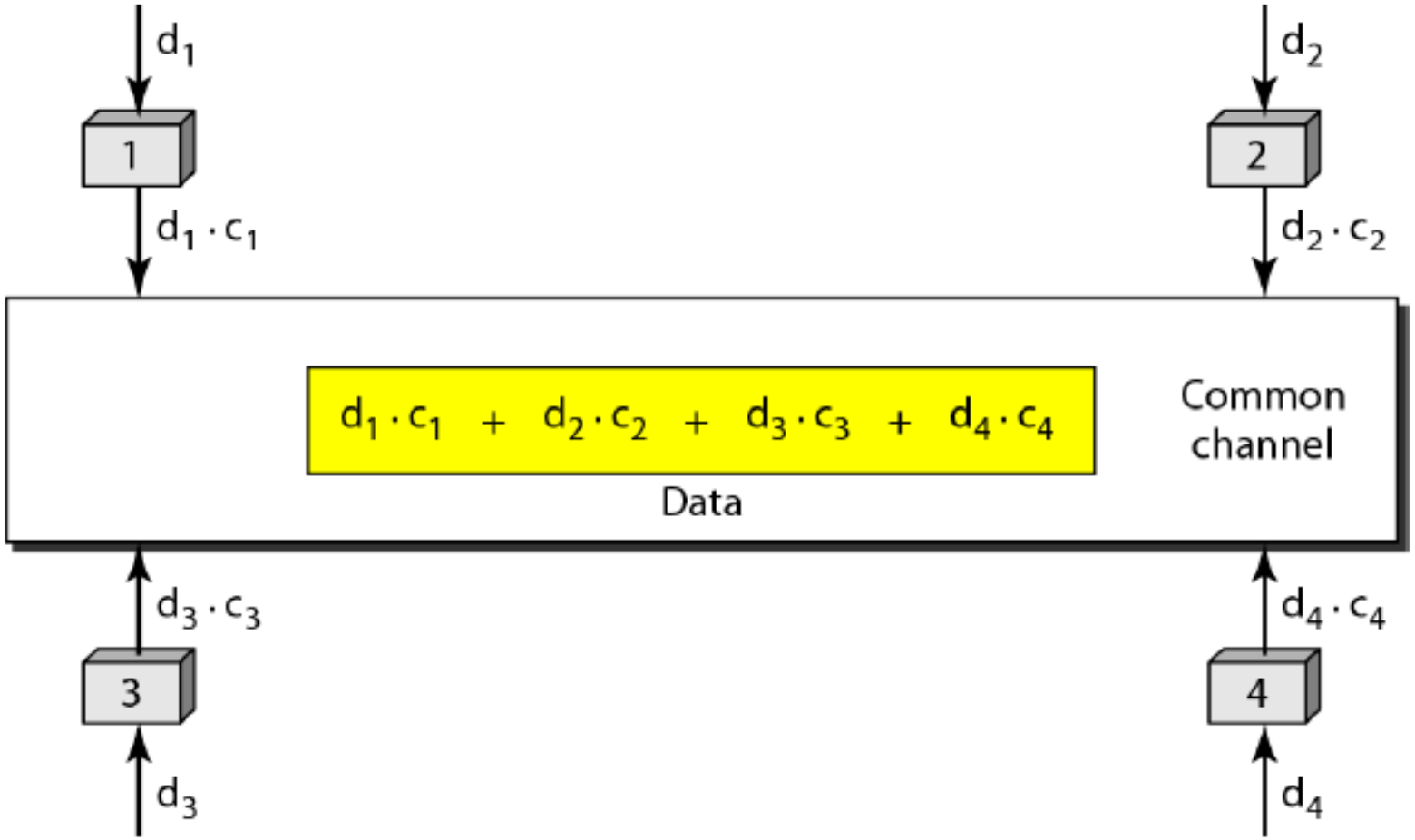
However, TDMA has an overhead of synchronization that specifies each station's time slot by adding synchronization bits to each slot

Figure 12.22 *Time-division multiple access (TDMA)*



CDMA (Code Division Multiple Access)

The code division multiple access (CDMA) is a channel access method. In CDMA, all stations can simultaneously send the data over the same channel. It means that it allows each station to transmit the data frames with full frequency on the shared channel at all times. It does not require the division of bandwidth on a shared channel based on time slots. If multiple stations send data to a channel simultaneously, their data frames are separated by a unique code sequence. Each station has a different unique code for transmitting the data over a shared channel. For example, there are multiple users in a room that are continuously speaking. Data is received by the users if only two-person interact with each other using the same language. Similarly, in the network, if different stations communicate with each other simultaneously with different code language.


d_1
1
$d_1 \cdot c_1$
d_2
2
$d_2 \cdot c_2$
$d_1 \cdot c_1 + d_2 \cdot c_2 + d_3 \cdot c_3 + d_4 \cdot c_4$
Data
Common channel
$d_3 \cdot c_3$
3
d_3
$d_4 \cdot c_4$
4
d_4

Bridges

- A bridge operates in both the physical and the data link layer.
- As a physical layer device, it regenerates the signal it receives.
- As a data link layer device, the bridge can check the physical (MAC) addresses (source and destination) contained in the frame.

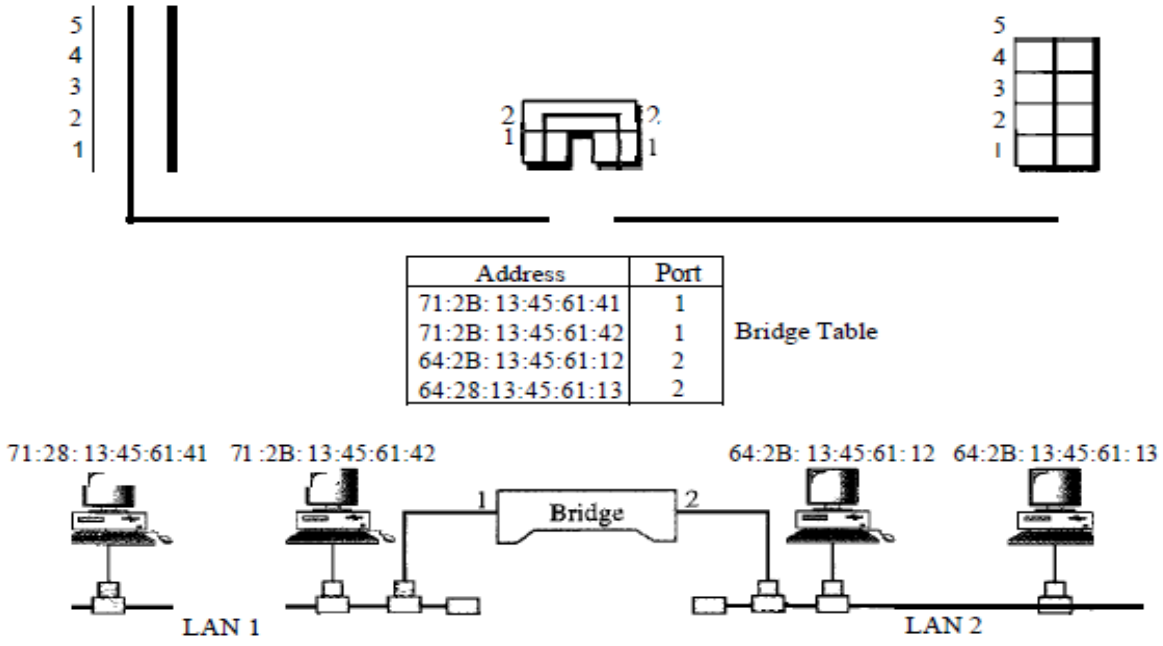
Filtering

One may ask, What is the difference in functionality between a bridge and a repeater? A bridge has filtering capability. It can check the destination address of a frame and decide if the frame should be forwarded or dropped. If the frame is to be forwarded, the decision must specify the port. A bridge has a table that maps addresses to ports.

A bridge has a table used in filtering decisions.

Let us give an example. In Figure 15.5, two LANs are connected by a bridge. If a frame destined for station 712B13456142 arrives at port 1, the bridge consults its table to find the departing port. According to its table, frames for 7l2B13456142 leave through port 1; therefore, there is no need for forwarding, and the frame is dropped. On the other hand, if a frame for 712B13456141 arrives at port 2, the departing port is port 1 and the frame is forwarded. In the first case, LAN 2 remains free of traffic; in the second case, both LANs have traffic. In our example, we show a two-port bridge; in reality a bridge usually has more ports. Note also that a bridge does not change the physical addresses contained in the frame.

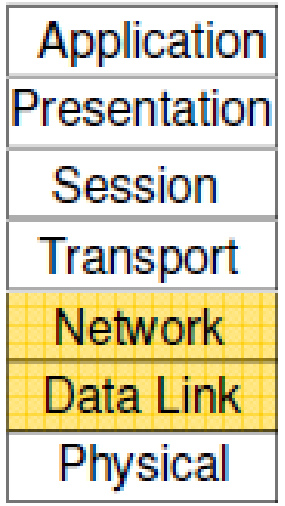
Figure 15.5 *A bridge connecting two LANs*



Bridges, Spanning Trees & Switches

- Focus:
  - What to do when one shared LAN isn' t big enough?

- Interconnecting LANs
  - Hubs
  - LAN bridges/switches
  - A preview of the Network layer

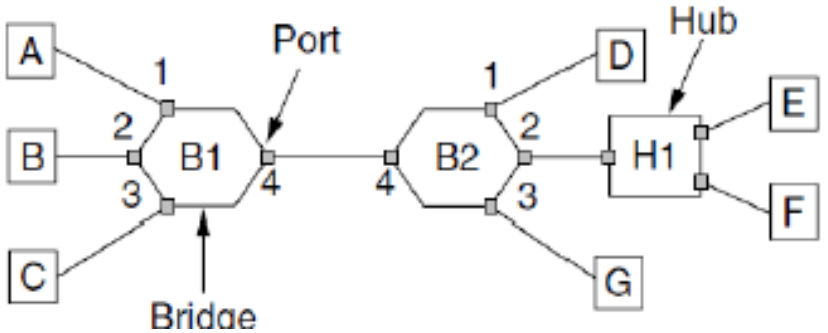


Why do we need a bridge/switch?

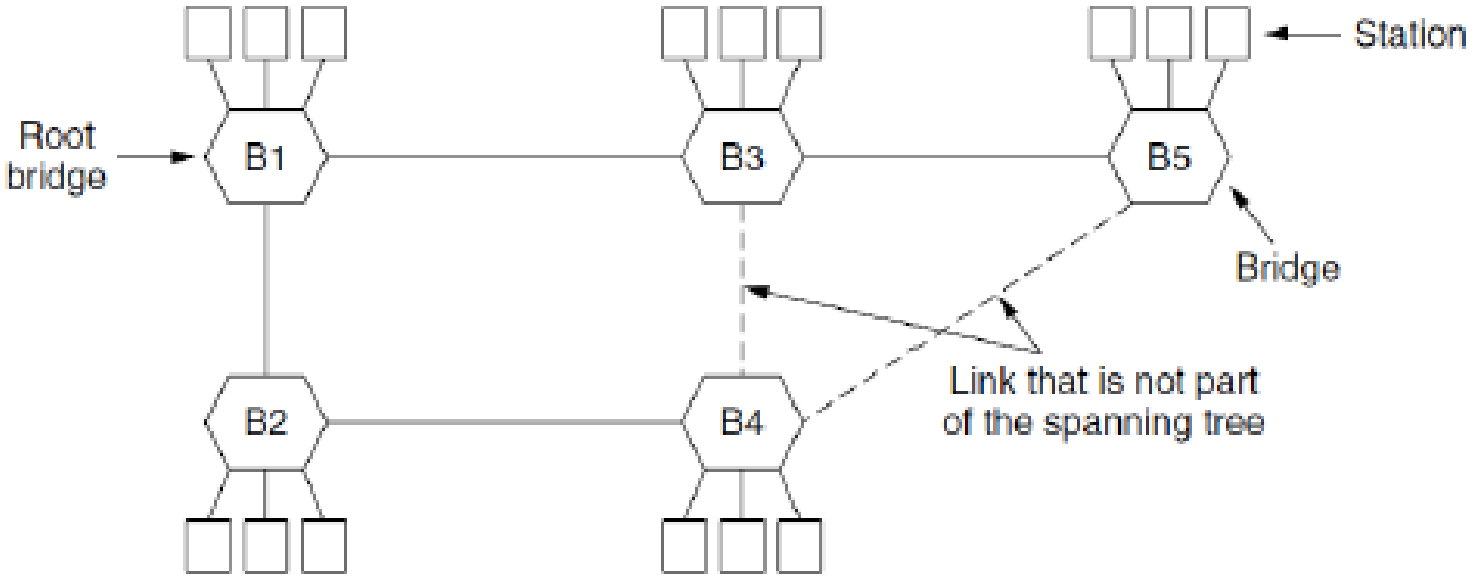
- One shared LAN can limit us in terms of:
  - Distance *(why?)*
  - Number of nodes *(why?)*
- How do we scale to a larger, more efficient networks?
  - We must be able to interconnect LANs

Bridges

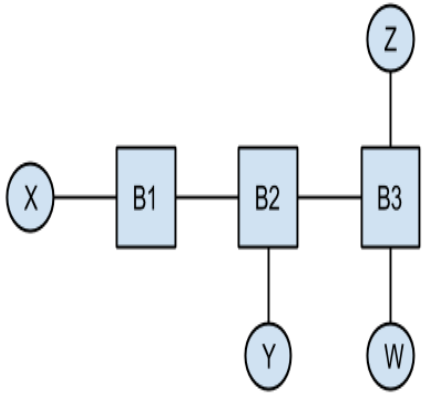
- Connected to different LANs over *ports*
- Operate in "promiscuous mode"
  - receives packet on one port and forwards it to the outgoing port
    - Example. From A to D
  - is "never" a communication endpoint itself
- What should each bridge do with an incoming frame?



Spanning Tree Example: Transients



Ex: Consider the hosts W, X, Y, Z and the learning bridges B1, B2, B3. Assume, that they have empty forwarding tables to begin with and keep on adding entries based on the events



•Suppose host X sends a packet to W. Which bridges learn the location of host X (i.e., which of the bridge's interfaces should traffic destined to X be directed)? Also, does host Y see this packet? Also, which bridges learn the location of host W?

Bridges B1, B2, B3 learn of the location of host X. Yes, the host Y will see this packet since each of the bridges will flood this packet since they do not know the location of W. No, the bridges do not learn the location of W since the bridges flood the packet but do not know flooding on which interface made W get the packet.

- Now, suppose host Y sends to X. Which bridges learn the location of host Y? Also, does host Z see this packet? Also, which bridges learn the location of host X from this transmission?

  Bridge B1, B2 will learn the location of Y. No, host Z will not see this packet. As before, this transmission does not tell us anything about the host that receives the packet.

- Now, finally host W sends to Y. Which bridges learn the location of host W? Does host Z see this packet? Also, which bridges learn the location of host Y from this transmission?

  Bridges B2, B3 will learn of the location of W. Yes, host Z will see this packet. As before, this transmission does not tell us anything about the host that receives the packet.

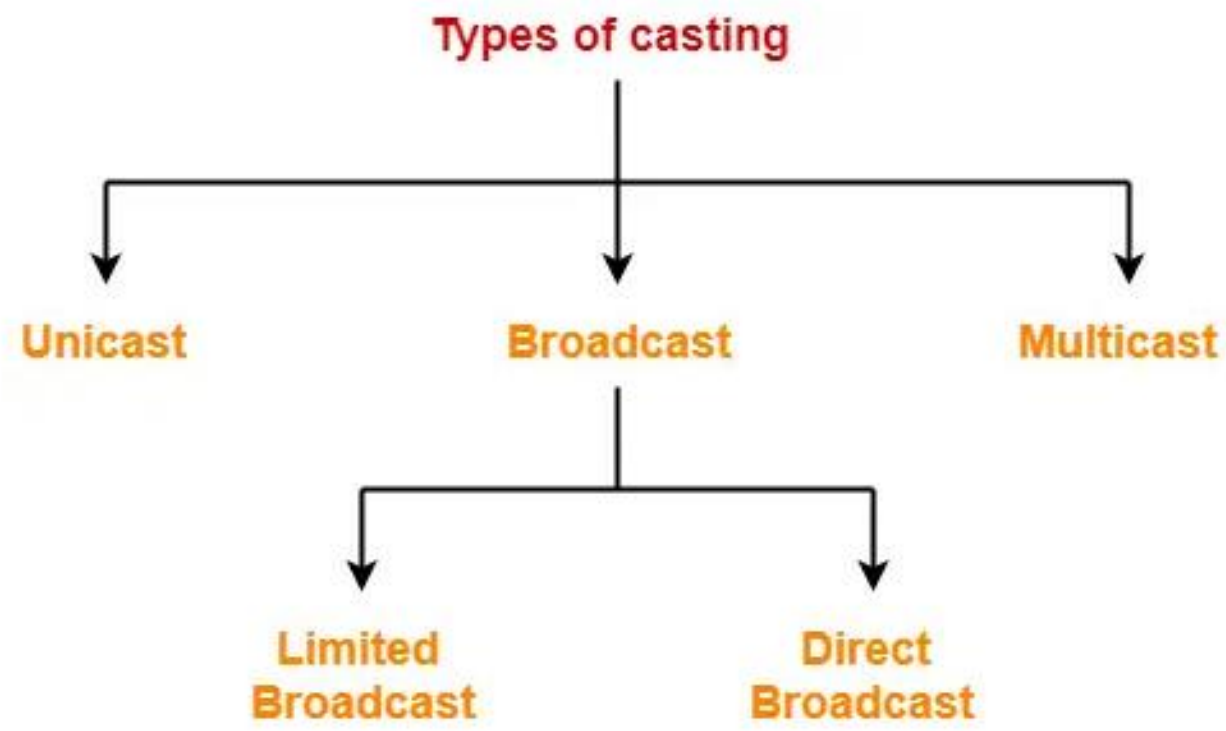
Two-Layer Switches

When we use the term *switch,* we must be careful because a switch can mean two different things. We must clarify the term by adding the level at which the device operates. We can have a two-layer switch or a three-layer switch. A **three-layer switch** is used at the network layer; it is a kind of router. The **two-layer switch** performs at the physical and data link layers.

A two-layer switch is a bridge, a bridge with many ports and a design that allows better (faster) performance. A bridge with a few ports can connect a few LANs together. A bridge with many ports may be able to allocate a unique port to each station, with each station on its own independent entity. This means no competing traffic (no collision, as we saw in Ethernet).

Logical addressing

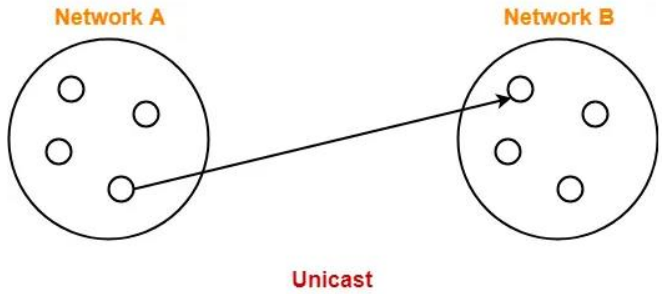
Casting in Networking-



1. Unicast-

Transmitting data from one source host to one destination host is called as **unicast**.

It is a one to one transmission.



Example-

Host A having IP Address 11.1.2.3 sending data to host B having IP Address 20.12.4.2.

Here,

Source Address = IP Address of host A = 11.1.2.3

Destination Address = IP Address of host B = 20.12.4.2

2. Broadcast-

Transmitting data from one source host to all other hosts residing in the same or other network is called as **broadcast**.

It is a one to all transmission.

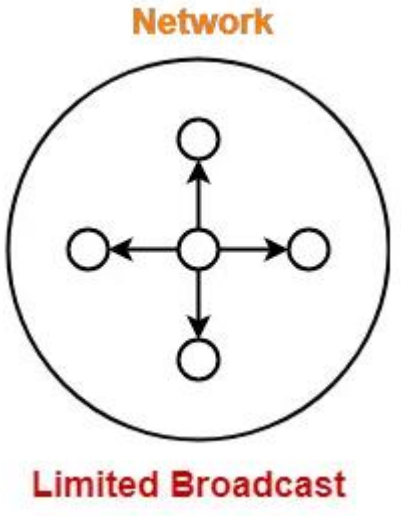
Based on recipient's network, it is classified as-

Limited Broadcast

Direct Broadcast

A. Limited Broadcast-

Transmitting data from one source host to all other hosts residing in the same network is called as **limited broadcast**.



NOTE

Limited Broadcast Address for any network

= All 32 bits set to 1

= 11111111.11111111.11111111.11111111

= 255.255.255.255

Example-

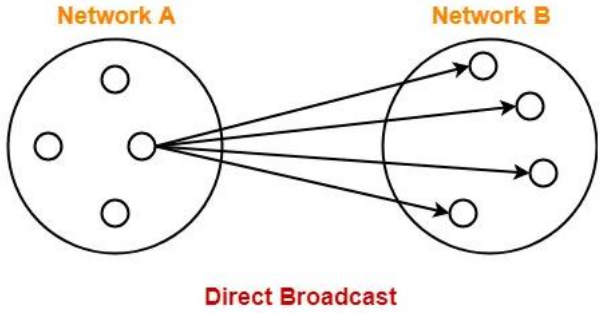
Host A having IP Address 11.1.2.3 sending data to all other hosts residing in the same network.

Here,

- Source Address = IP Address of host A = 11.1.2.3
- Destination Address = 255.255.255.255

B. Direct Broadcast-

Transmitting data from one source host to all other hosts residing in some other network is called as **direct broadcast**.



NOTE

Direct Broadcast Address for any network is the IP Address where-

- Network ID is the IP Address of the network where all the destination hosts are present.
- Host ID bits are all set to 1.

Example-

Host A having IP Address 11.1.2.3 sending data to all other hosts residing in the network having IP Address 20.0.0.0

Here,

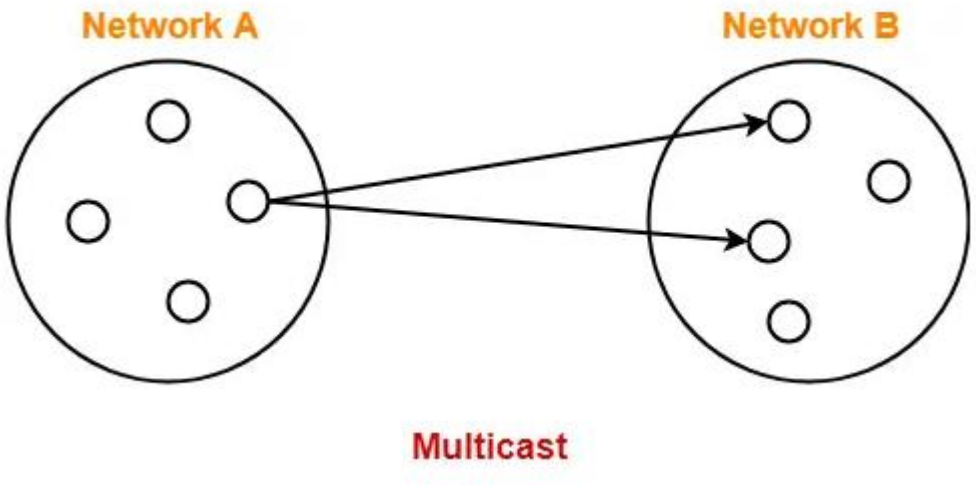
Source Address = IP Address of host A = 11.1.2.3

Destination Address = 20.255.255.255

3. Multicast-

Transmitting data from one source host to a particular group of hosts having interest in receiving the data is called as **multicast**.

It is a one to many transmission.



Examples-

- Sending a message to a particular group of people on whatsapp
- Sending an email to a particular group of people
- Video conference or teleconference

Basic internetworking (IP, CIDR, ARP, RARP, DHCP, ICMP),

IP Datagram Header Format

Unlike the post office, a router or computer cannot determine the size of a package without additional information.

A person can look at a letter or box and determine how big it is, but a router cannot. Therefore, additional information is required at the IP layer, in addition to the source and destination IP addresses.

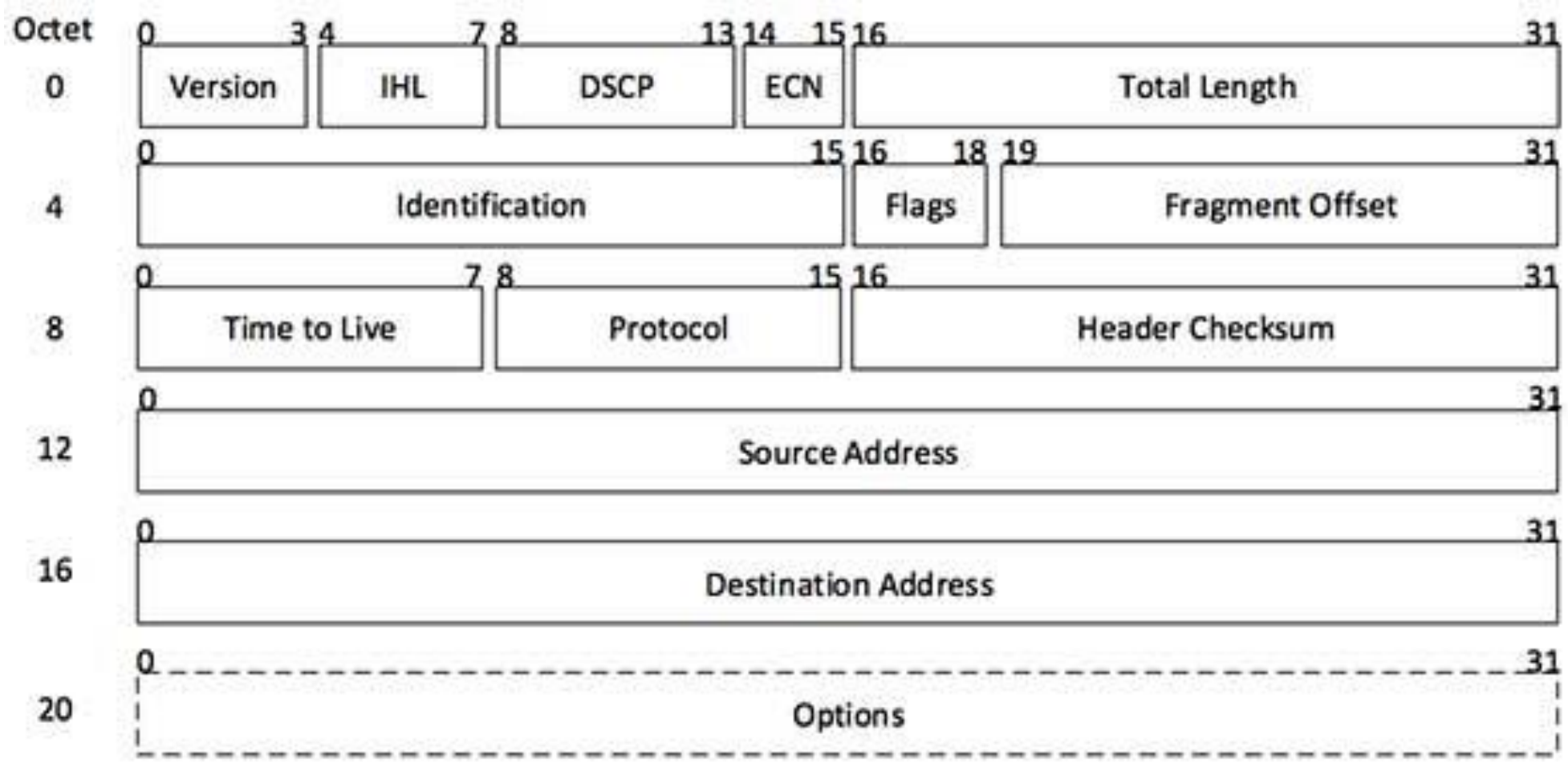
Figure on next slide is a logical representation of the information that is used at the IP layer to enable the delivery of electronic data.

This information is called a header, and is analogous to the addressing information on an envelope.

A header contains the information required to route data on the Internet, and has the same format regardless of the type of data being sent.

This is the same for an envelope where the address format is the same regardless of the type of letter being sent.

IP Datagram Header Format



[Image: IP Header]

IP header includes many relevant information including Version Number, which, in this context, is 4. Other details are as follows −

Version − A 4-bit field that identifies the IP version being used. The current version is 4, and this version is referred to as IPv4.

IHL − Internet Header Length; Length of entire IP header. A 4-bit field containing the length of the IP header in 32-bit increments. The minimum length of an IP header is 20 bytes, or five 32-bit increments. The maximum length of an IP header is 24 bytes, or six 32-bit increments. Therefore, the header length field should contain either 5 or 6.

DSCP − Differentiated Services Code Point; this is Type of Service (ToS). A 6-bit field used to identify the level of service a packet receives in the network. DSCP is a 3-bit expansion of IP precedence with the elimination of the ToS bits.

ECN − Explicit Congestion Notification; It carries information about the congestion seen in the route.

Total Length − Length of entire IP Packet (including IP header and IP Payload). The size of the field is 16 bits. Total length of the datagram = Length of the header + Length of the data

Identification − If IP packet is fragmented during the transmission, all the fragments contain same identification number to identify original IP packet they belong to.

Flags − As required by the network resources, if IP Packet is too large to handle, these ‘flags’ tells if they can be fragmented or not. In this 3-bit flag, the MSB is always set to ‘0’.

Fragment Offset − This offset tells the exact position of the fragment in the original IP Packet.

As an IP packet moves through the Internet, it might need to cross a route that cannot handle the size of the packet. The packet will be divided, or fragmented, into smaller packets and reassembled later. These fields are used to fragment and reassemble packets.

Time to Live − To avoid looping in the network, every packet is sent with some TTL value set, which tells the network how many routers (hops) this packet can cross. At each hop, its value is decremented by one and when the value reaches zero, the packet is discarded.

Protocol − Tells the Network layer at the destination host, to which Protocol this packet belongs to, i.e. the next level Protocol. For example protocol number of ICMP is 1, TCP is 6 and UDP is 17.

Header Checksum − This field is used to keep checksum value of entire header which is then used to check if the packet is received error-free.

Source Address − 32-bit address of the Sender (or source) of the packet.

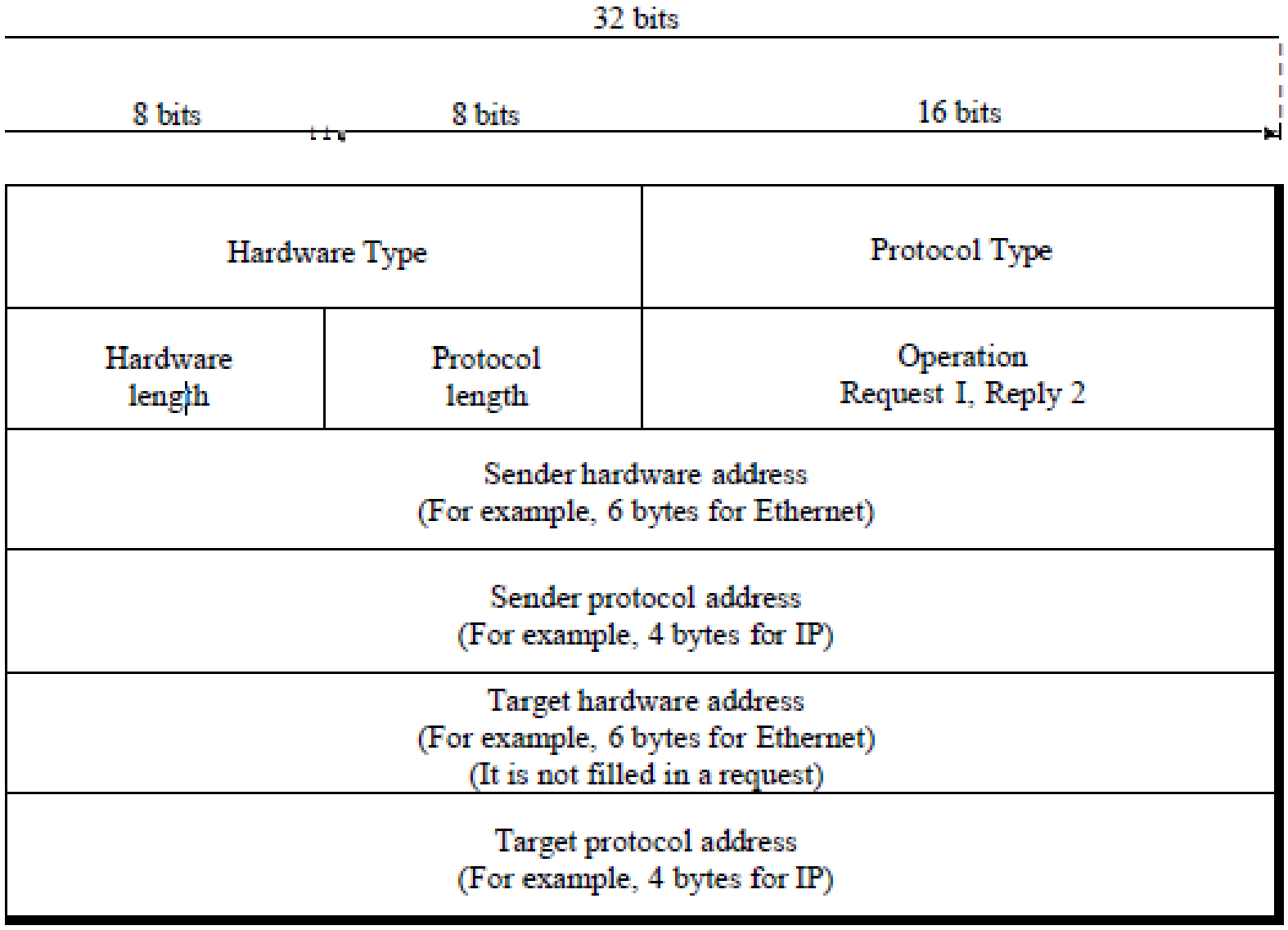
Destination Address − 32-bit address of the Receiver (or destination) of the packet.

Options − This is optional field, which is used if the value of IHL is greater than 5. These options may contain values for options such as Security, Record Route, Time Stamp, etc.

Table 3.2 Values the Protocol Component Can Take

Value (Decimal)	Protocol
1	Internet Control Message Protocol (ICMP)
2	Internet Group Management Protocol (IGMP)
3	Gateway-to-Gateway Protocol (GGP)
4	Internet Protocol (IP)
6	Transmission Control Protocol (TCP)
8	Exterior Gateway Protocol (EGP)
9	Interior Gateway Protocol (IGP)
17	User Datagram Protocol (UDP)
41	Internet Protocol Version 6 (IPv6)
86	Dissimilar Gateway Protocol (DGP)
88	Interior Gateway Routing Protocol (IGRP)
89	Open Shortest Path First (OSPF)

Address Resolution Protocol (ARP)



The fields are as follows:

Hardware type. This is a 16-bit field defining the type of the network on which ARP is running. Each LAN has been assigned an integer based on its type. For example, Ethernet is given type 1. ARP can be used on any physical network.

Protocol type. This is a 16-bit field defining the protocol. For example, the value of this field for the IPv4 protocol is 080016, ARP can be used with any higher-level protocol.

Hardware length. This is an 8-bit field defining the length of the physical address in bytes. For example, for Ethernet the value is 6.

Protocol length. This is an 8-bit field defining the length of the logical address in bytes. For example, for the IPv4 protocol the value is 4.

Operation. This is a 16-bit field defining the type of packet. Two packet types are defined: ARP request (1) and ARP reply (2).

Sender hardware address. This is a variable-length field defining the physical address of the sender. For example, for Ethernet this field is 6 bytes long.

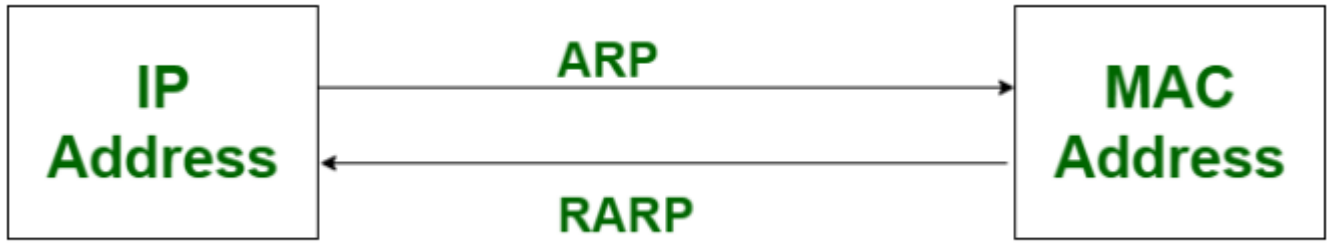
Sender protocol address. This is a variable-length field defining the logical (for example, IP) address of the sender. For the IP protocol, this field is 4 bytes long.

Target hardware address. This is a variable-length field defining the physical address of the target. For example, for Ethernet this field is 6 bytes long. For an ARP request message, this field is all 0s because the sender does not know the physical address of the target.

Target protocol address. This is a variable-length field defining the logical (for example, IP) address of the target. For the IPv4 protocol, this field is 4 bytes long.

Difference between ARP and RARP

In **Address Resolution Protocol (ARP)**, Receiver's MAC address is fetched. Through ARP, (32-bit) IP address mapped into (48-bit) MAC address. Whereas, In **Reverse Address Resolution Protocol (RARP)**, IP address is fetched through server. Through RARP, (48-bit) MAC address of 48 bits mapped into (32-bit) IP address.



ARP	RARP
A protocol used to map an IP address to a physical (MAC) address	A protocol used to map a physical (MAC) address to an IP address
To obtain the MAC address of a network device when only its IP address is known	To obtain the IP address of a network device when only its MAC address is known
Client broadcasts its IP address and requests a MAC address, and the server responds with the corresponding MAC address	Client broadcasts its MAC address and requests an IP address, and the server responds with the corresponding IP address
Widely used in modern networks to resolve IP addresses to MAC addresses	Rarely used in modern networks as most devices have a pre-assigned IP address
ARP stands for Address Resolution Protocol.	Whereas RARP stands for Reverse Address Resolution Protocol.
Through ARP, (32-bit) IP address mapped into (48-bit) MAC address.	Whereas through RARP, (48-bit) MAC address of 48 bits mapped into (32-bit) IP address.
In ARP, broadcast MAC address is used.	While in RARP, broadcast IP address is used.
In ARP, ARP table is managed or maintained by local host.	While in RARP, RARP table is managed or maintained by RARP server.
In Address Resolution Protocol, Receiver's MAC address is fetched.	While in RARP, IP address is fetched.
In ARP, ARP table uses ARP reply for its updation.	While in RARP, RARP table uses RARP reply for configuration of IP addresses .
Hosts and routers uses ARP for knowing the MAC address of other hosts and routers in the networks.	While RARP is used by small users having less facilities.
ARP is used in sender's side to map the receiver's MAC address.	RARP is used in receiver's side to map the sender's IP.