While sending the data from the sender to the receiver the flow of data needs to be controlled.

Suppose a situation where the sender is sending the data at a rate higher than the receiver is able to receive and process it, then the data will get lost.

The [**Flow-control**](https://afteracademy.com/blog/what-is-flow-control-in-networking) methods will help in ensuring that the data doesn't get lost.

The flow control method will keep a check that the senders send the data only at a rate that the receiver is able to receive and process.

There are mainly two ways in which this can be achieved i.e. using **Stop-and-wait protocol** or sliding window protocol.

Stop and Wait ARQ

**Characteristics**

* Used in Connection-oriented communication.
* It offers error and flow control
* It is used in Data Link and Transport Layers
* Stop and Wait ARQ mainly implements Sliding Window Protocol concept with Window Size 1

**Useful Terms:**

* **Propagation Delay:**Amount of time taken by a packet to make a physical journey from one router to another router.

Propagation Delay = (Distance between routers) / (Velocity of propagation)

* RoundTripTime (**RTT**) = 2\* Propagation Delay
* TimeOut (**TO**) =  2\* RTT
* Time To Live (**TTL**) = 2\* TimeOut. (Maximum TTL is 180 seconds)

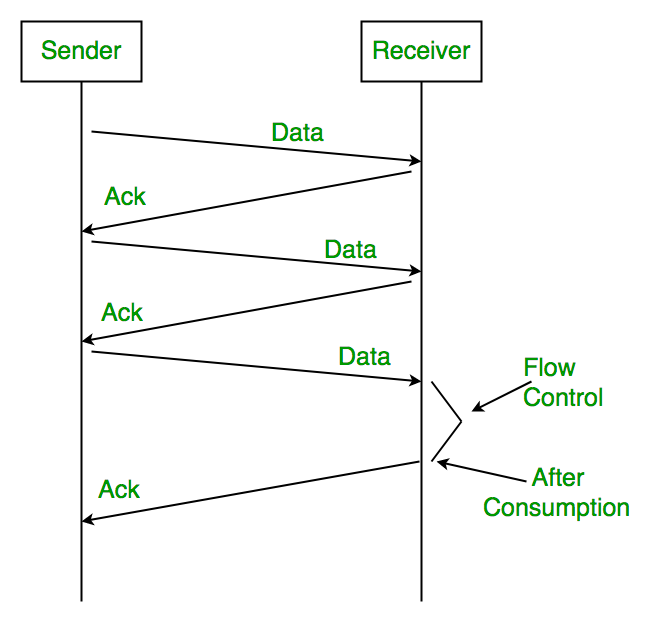
**Simple Stop and Wait**

**Sender:**

Rule 1) Send one data packet at a time.  
Rule 2) Send next packet only after receiving acknowledgement for previous.

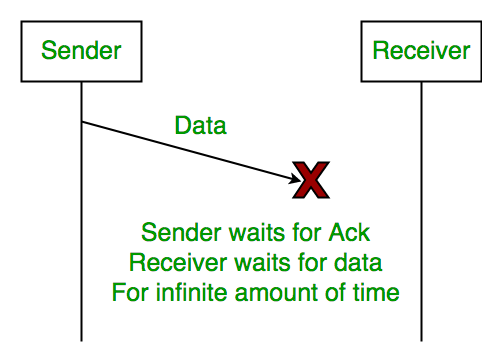
**Receiver:**

Rule 1) Send acknowledgement after receiving and consuming of data packet.  
Rule 2) After consuming packet acknowledgement need to be sent (Flow Control)

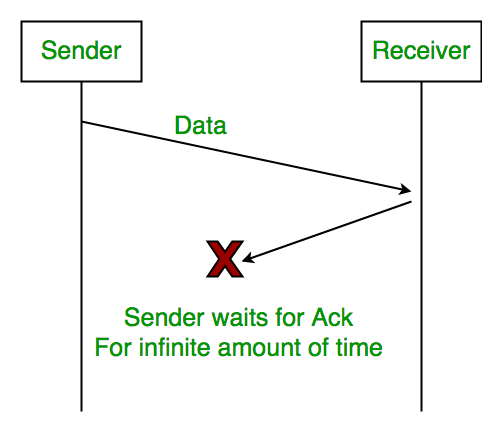
**[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ.png)**

**Problems :**

**1. Lost Data**

**[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-2.png)**

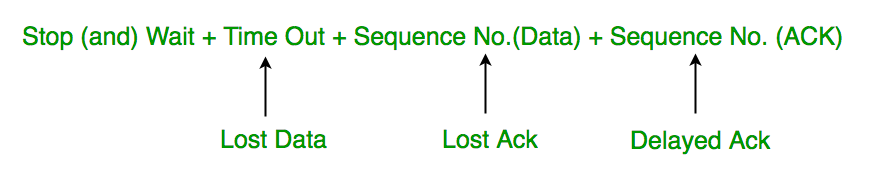
**2. Lost Acknowledgement:**

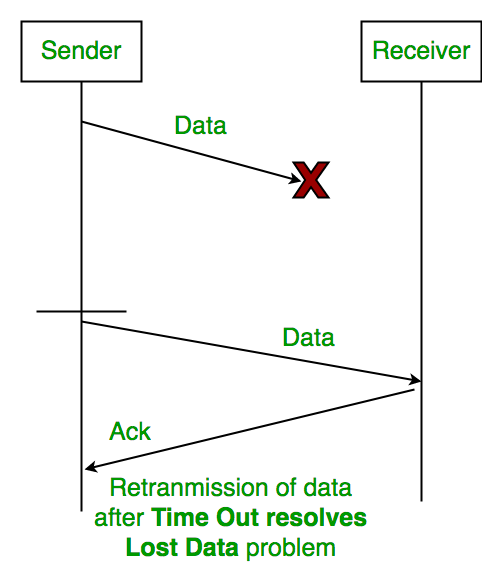
**[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-3.png)**

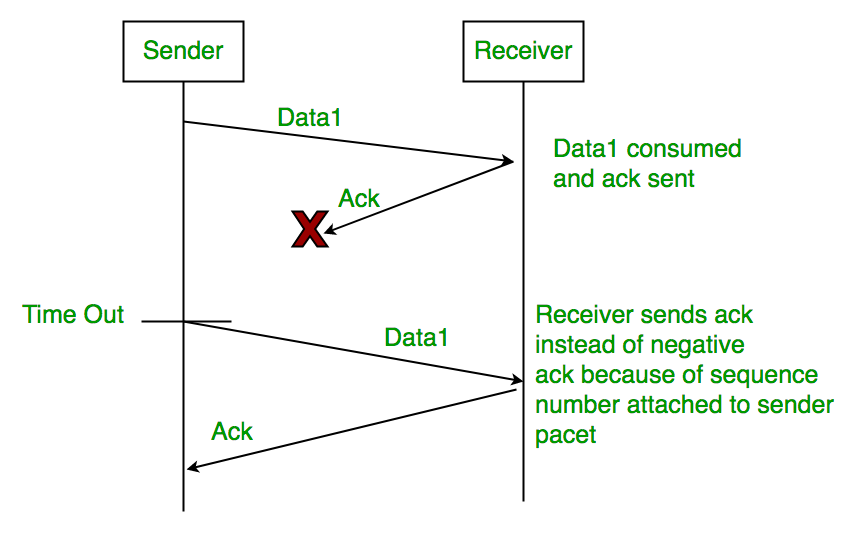
**3. Delayed Acknowledgement/Data:** After timeout on sender side, a long delayed acknowledgement might be wrongly considered as acknowledgement of some other recent packet.

**Stop and Wait ARQ (Automatic Repeat Request)**

Above 3 problems are resolved by Stop and Wait ARQ (Automatic Repeat Request) that does both error control and flow control.

**[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-4.png)**

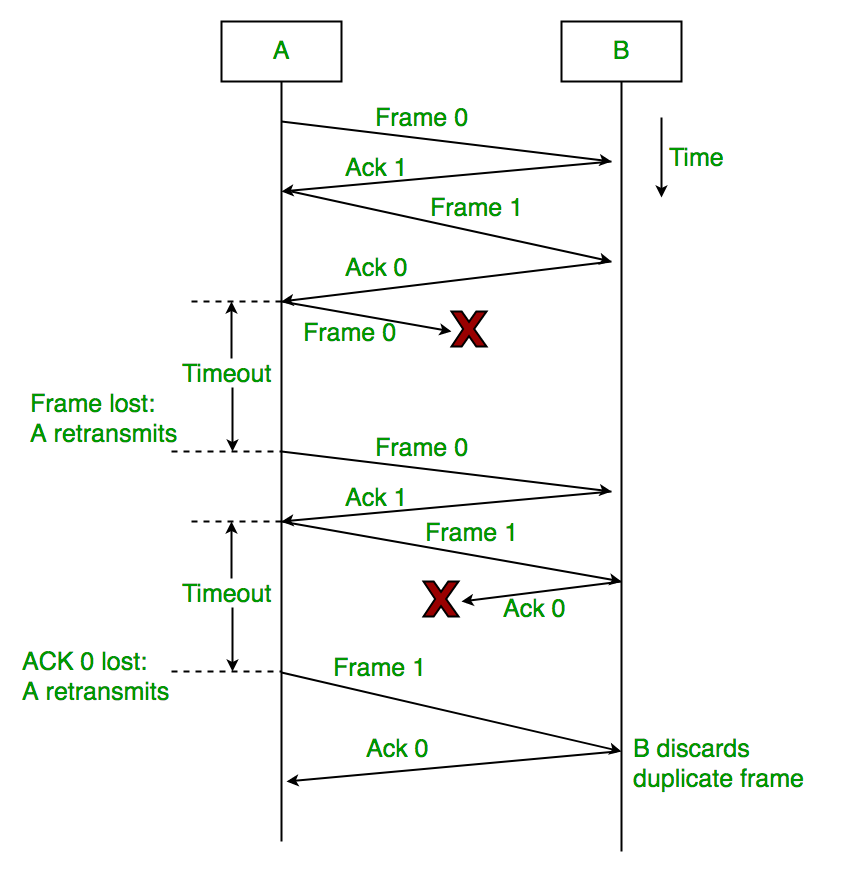
**1. Time Out:**  
[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-5.png)

**2. Sequence Number (Data)**  
 [[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-6.png)](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-6.png)

**3. Delayed Acknowledgement:**  
This is resolved by introducing sequence number for acknowledgement also.

**Working of Stop and Wait ARQ:**

1) Sender A sends a data frame or packet with sequence number 0.  
2) Receiver B, after receiving data frame, sends and acknowledgement with sequence number 1 (sequence number of next expected data frame or packet)  
There is only one bit sequence number that implies that both sender and receiver have buffer for one frame or packet only.

[](https://media.geeksforgeeks.org/wp-content/uploads/Stop-and-Wait-ARQ-7.png)

**Characteristics of Stop and Wait ARQ:**

* It uses link between sender and receiver as half duplex link
* Throughput = 1 Data packet/frame per  RTT
* If Bandwidth\*Delay product is very high, then stop and wait protocol is not so useful. The sender has to keep waiting for acknowledgements before sending the processed next packet.
* It is an example for “**Closed Loop OR connection oriented**“ protocols
* It is an special category of SWP where its window size is 1
* Irrespective of number of packets sender is having stop and wait protocol  requires only  2 sequence numbers 0 and 1

The Stop and Wait ARQ solves main three problems, but may cause big performance issues as sender always waits for acknowledgement even if it has next packet ready to send. Consider a situation where you have a high bandwidth connection and propagation delay is also high (you are connected to some server in some other country though a high speed connection). To solve this problem, we can send more than one packet at a time with a larger sequence numbers. We will be discussing these protocols in next articles.

So Stop and Wait ARQ may work fine where propagation delay is very less for example LAN connections, but performs badly for distant connections like satellite connection.

[**next →**](https://www.javatpoint.com/imap-protocol)[**← prev**](https://www.javatpoint.com/ospf-protocol)

Stop and Wait Protocol

Before understanding the stop and Wait protocol, we first know about the error control mechanism. The error control mechanism is used so that the received data should be exactly same whatever sender has sent the data. The error control mechanism is divided into two categories, i.e., Stop and Wait ARQ and sliding window. The sliding window is further divided into two categories, i.e., Go Back N, and Selective Repeat. Based on the usage, the people select the error control mechanism whether it is **stop and wait** or **sliding window**.

What is Stop and Wait protocol?

Here stop and wait means, whatever the data that sender wants to send, he sends the data to the receiver. After sending the data, he stops and waits until he receives the acknowledgment from the receiver. The stop and wait protocol is a flow control protocol where flow control is one of the services of the data link layer.

It is a data-link layer protocol which is used for transmitting the data over the noiseless channels. It provides unidirectional data transmission which means that either sending or receiving of data will take place at a time. It provides flow-control mechanism but does not provide any error control mechanism.

The idea behind the usage of this frame is that when the sender sends the frame then he waits for the acknowledgment before sending the next frame.

Primitives of Stop and Wait Protocol

**The primitives of stop and wait protocol are:**

**Sender side**

**Rule 1:** Sender sends one data packet at a time.

**Rule 2:** Sender sends the next packet only when it receives the acknowledgment of the previous packet.

Therefore, the idea of stop and wait protocol in the sender's side is very simple, i.e., send one packet at a time, and do not send another packet before receiving the acknowledgment.

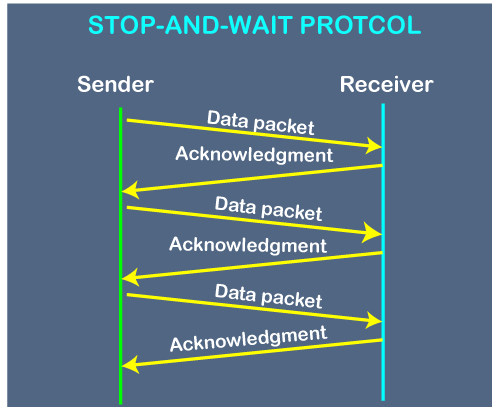
Receiver side

**Rule 1:** Receive and then consume the data packet.

**Rule 2:** When the data packet is consumed, receiver sends the acknowledgment to the sender.

Therefore, the idea of stop and wait protocol in the receiver's side is also very simple, i.e., consume the packet, and once the packet is consumed, the acknowledgment is sent. This is known as a flow control mechanism.

Working of Stop and Wait protocol



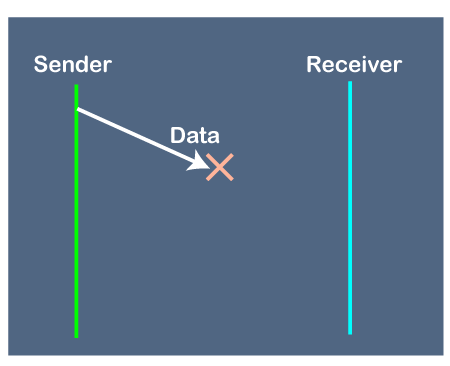
The above figure shows the working of the stop and wait protocol. If there is a sender and receiver, then sender sends the packet and that packet is known as a data packet. The sender will not send the second packet without receiving the acknowledgment of the first packet. The receiver sends the acknowledgment for the data packet that it has received. Once the acknowledgment is received, the sender sends the next packet. This process continues until all the packet are not sent.

The main advantage of this protocol is its simplicity but it has some disadvantages also. For example, if there are 1000 data packets to be sent, then all the 1000 packets cannot be sent at a time as in Stop and Wait protocol, one packet is sent at a time.

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**

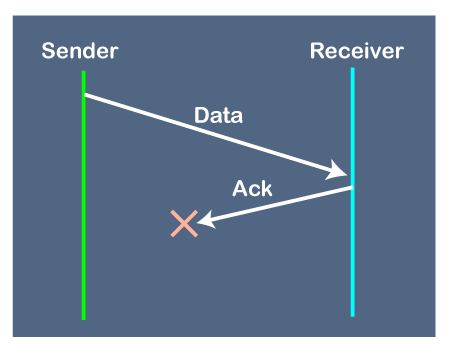


Suppose the sender sends the data and the data is lost. The receiver is waiting for the data for a long time. Since the data is not received by the receiver, so it does not send any acknowledgment. Since the sender does not receive any acknowledgment so it will not send the next packet. This problem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

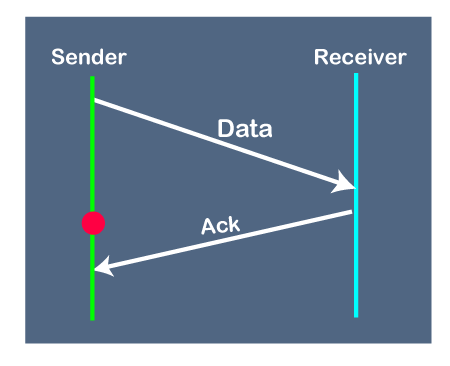


Suppose the sender sends the data and it has also been received by the receiver. On receiving the packet, the receiver sends the acknowledgment. In this case, the acknowledgment is lost in a network, so there is no chance for the sender to receive the acknowledgment. There is also no chance for the sender 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.

**Advantages of Stop and Wait Protocol**

1. It is very simple to implement.
2. The main advantage of this protocol is the accuracy. The next frame is sent only when the first frame is acknowledged. So, there is no chance of any frame being lost.

**Disadvantages of Stop and Wait Protocol**

1. We can send only one packet at a time.
2. If the distance between the sender and the receiver is large then the propagation delay would be more than the transmission delay. Hence, efficiency would become very low.
3. After every transmission, the sender has to wait for the acknowledgment and this time will increase the total transmission time. This makes the transmission process **slow**.

This is how the flow of data is controlled using the stop-and-wait protocol

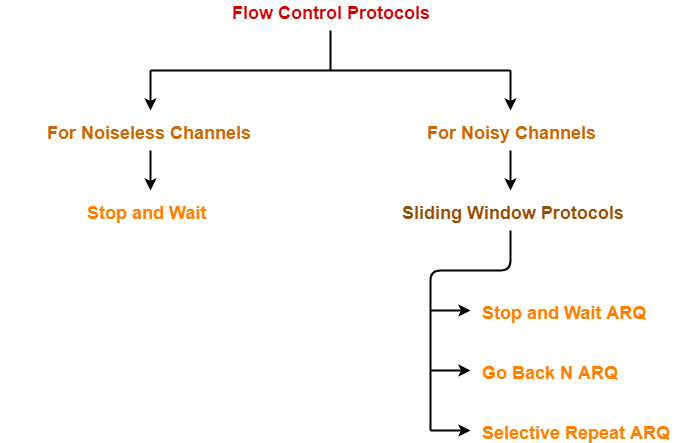
**Flow Control in Computer Networks-**

In computer networks, flow control is defined as-

|  |
| --- |
| A set of procedures which are used for restricting the amount of data that a sender can send to the receiver. |

**Flow Control Protocols-**

There are various flow control protocols which are classified as-



**Stop and Wait Protocol-**

|  |
| --- |
| Stop and Wait Protocol is the simplest flow control protocol. |

It works under the following assumptions-

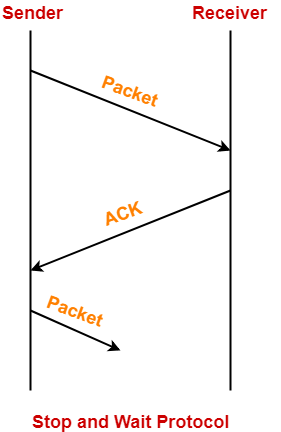
* Communication channel is perfect.
* No error occurs during transmission.

**Working-**

The working of a stop and wait protocol may be explained as-

* Sender sends a data packet to the receiver.
* Sender stops and waits for the acknowledgement for the sent packet from the receiver.
* Receiver receives and processes the data packet.
* Receiver sends an acknowledgement to the sender.
* After receiving the acknowledgement, sender sends the next data packet to the receiver.

These steps are illustrated below-

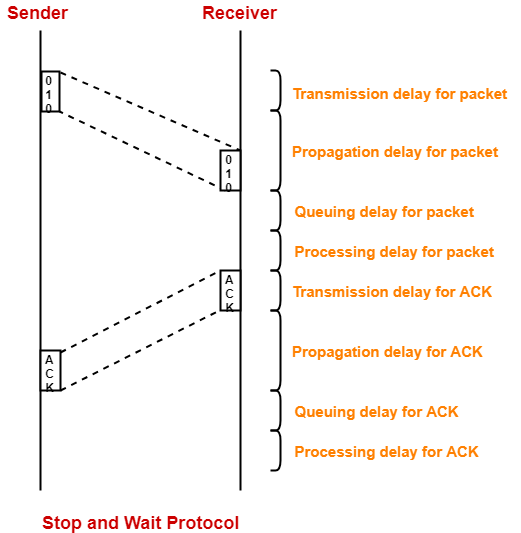


**Analysis-**

Now, let us analyze in depth how the transmission is actually carried out-

* Sender puts the data packet on the transmission link.
* Data packet propagates towards the receiver’s end.
* Data packet reaches the receiver and waits in its buffer.
* Receiver processes the data packet.
* Receiver puts the acknowledgement on the transmission link.
* Acknowledgement propagates towards the sender’s end.
* Acknowledgement reaches the sender and waits in its buffer.
* Sender processes the acknowledgement.

These steps are illustrated below-



**Advantages-**

The advantages of stop and wait protocol are-

* It is very simple to implement.
* The incoming packet from receiver is always an acknowledgement.

**Limitations-**

The limitations of stop and wait protocol are-

**Point-01:**

It is extremely inefficient because-

* It makes the transmission process extremely slow.
* It does not use the bandwidth entirely as each single packet and acknowledgement uses the entire time to traverse the link.

**Point-02:**

If the data packet sent by the sender gets lost, then-

* Sender will keep waiting for the acknowledgement for infinite time.
* Receiver will keep waiting for the data packet for infinite time.

**Point-03:**

If acknowledgement sent by the receiver gets lost, then-

* Sender will keep waiting for the acknowledgement for infinite time.
* Receiver will keep waiting for another data packet for infinite time.

**Important Notes-**

**Note-01:**

Efficiency may also be referred by the following names-

* Line Utilization
* Link Utilization
* Sender Utilization
* Utilization of Sender

**Note-02:**

Throughput may also be referred by the following names-

* Bandwidth Utilization
* Effective Bandwidth
* Maximum data rate possible
* Maximum achievable throughput

**Note-03:**

Stop and Wait protocol performs better for LANs than WANs.

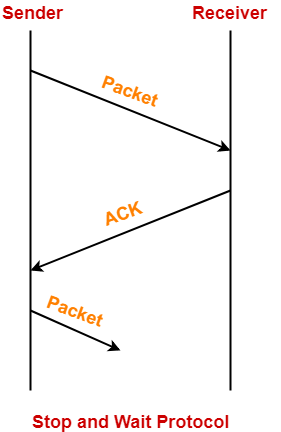
This is because-

* Efficiency of the protocol is inversely proportional to the distance between sender and receiver.
* So, the protocol performs better where the distance between sender and receiver is less.
* The distance is less in LANs as compared to WANs.

**Stop and Wait Protocol-**

In stop and wait protocol,

* Sender sends one data packet and then waits for its acknowledgement.
* Sender sends the next packet only after it receives the acknowledgement for the previous packet.



The main problem faced by the Stop and Wait protocol is the occurrence of deadlock due to-

1. Loss of data packet
2. Loss of acknowledgement

**Stop and Wait ARQ-**

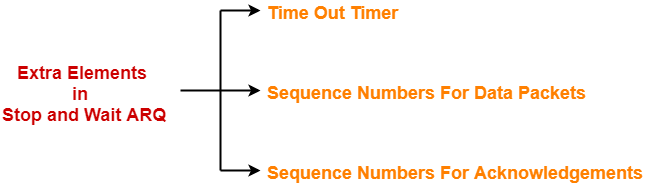
|  |
| --- |
| Stop and Wait ARQ is an improved and modified version of Stop and Wait protocol. |

Stop and Wait ARQ assumes-

* The communication channel is noisy.
* Errors may get introduced in the data during the transmission.

**Working-**

* Stop and wait ARQ works similar to stop and wait protocol.
* It provides a solution to all the limitations of stop and wait protocol.
* Stop and wait ARQ includes the following three extra elements.



Thus, we can say-

|  |
| --- |
| Stop and Wait ARQ  = Stop and Wait Protocol + Time Out Timer + Sequence Numbers for Data Packets and Acknowledgements |

**Number of Sequence Numbers Required-**

|  |
| --- |
| **NOTE**  For any sliding window protocol to work without any problem,  the following condition must be satisfied-  Available Sequence Numbers >= Sender Window Size + Receiver Window Size |

Stop and wait ARQ is a one bit sliding window protocol where-

* Sender window size = 1
* Receiver window size = 1

Thus, in stop and wait ARQ,

Minimum number of sequence numbers required

= Sender Window Size + Receiver Window Size

= 1 + 1

= 2

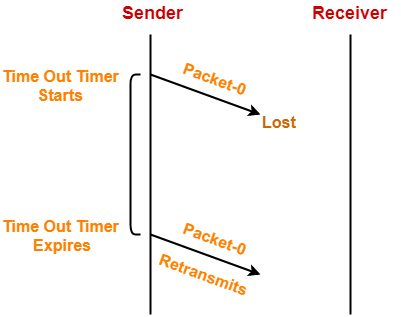
Thus,

* Minimum number of sequence numbers required in Stop and Wait ARQ = 2.
* The two sequence numbers used are 0 and 1.

**How Stop and Wait ARQ Solves All Problems?**

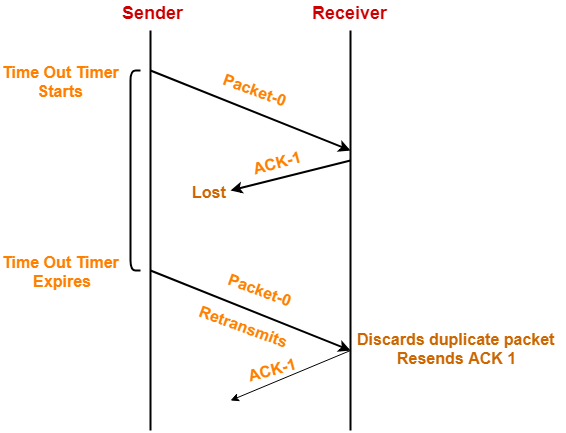
**1. Problem of Lost Data Packet-**

* Time out timer helps to solve the problem of lost data packet.
* After sending a data packet  to the receiver, sender starts the time out timer.
* If the data packet gets acknowledged before the timer expires, sender stops the time out timer.
* If the timer goes off before receiving the acknowledgement, sender retransmits the same data packet.
* After retransmission, sender resets the timer.
* This prevents the occurrence of deadlock.



**2. Problem of Lost Acknowledgement-**

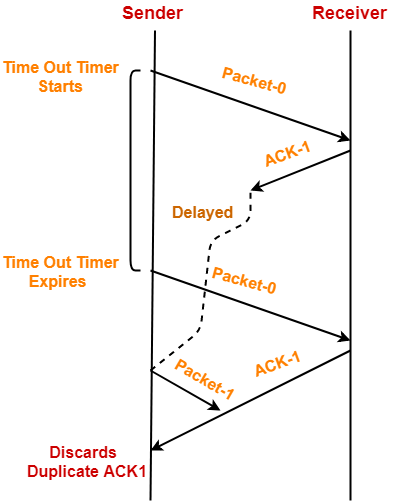
* Sequence number on data packets help to solve the problem of delayed acknowledgement.
* Consider the acknowledgement sent by the receiver gets lost.
* Then, sender retransmits the same data packet after its timer goes off.
* This prevents the occurrence of deadlock.
* The sequence number on the data packet helps the receiver to identify the duplicate data packet.
* Receiver discards the duplicate packet and re-sends the same acknowledgement.



|  |
| --- |
| **Role of Sequence Number on Data Packets**    Consider the above example-    **Step-01:**     * Sender sends a data packet with sequence number-0 to the receiver.     **Step-02:**     * Receiver receives the data packet correctly. * Receiver now expects data packet with sequence number-1. * Receiver sends the acknowledgement ACK-1.     **Step-03:**     * Acknowledgement ACK-1 sent by the receiver gets lost on the way.     **Step-04:**     * Sender receives no acknowledgement and time out occurs. * Sender retransmits the same data packet with sequence number-0. * This will be a duplicate packet for the receiver.     **Step-05:**     * Receiver receives the data packet and discovers it is the duplicate packet. * It expects the data packet with sequence number-1 but receiving the data packet with sequence number-0. * It discards the duplicate data packet and re-sends acknowledgement ACK-1. * ACK-1 requests the sender to send a data packet with sequence number-1. * This avoids the inconsistency of data.     **Conclusion-**     * Had the sequence numbers not been allotted to the data packets, receiver would have accepted the duplicate data packet thinking of it as the new data packet. * This is how sequence numbers allotted to the data packets prove to be useful for identifying the duplicate data packets and discarding them. |

**3. Problem of Delayed Acknowledgement-**

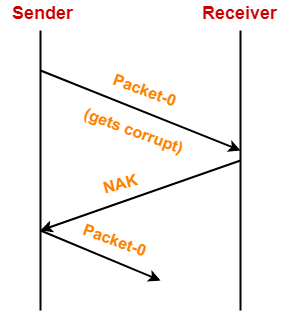
* Sequence number on acknowledgements help to solve the problem of delayed acknowledgement.



|  |
| --- |
| **Role of Sequence Number on Acknowledgements**    Consider the above example-    **Step-01:**     * Sender sends a data packet with sequence number-0 to the receiver.     **Step-02:**     * Receiver receives the data packet correctly. * Receiver now expects data packet with sequence number-1. * Receiver sends the acknowledgement ACK-1.     **Step-03:**     * Acknowledgement ACK-1 sent by the receiver gets delayed in reaching the sender.     **Step-04:**     * Sender receives no acknowledgement and time out occurs. * Sender retransmits the same data packet with sequence number-0. * This will be a duplicate packet for the receiver.     **Step-05:**     * Receiver receives the data packet and discovers it is the duplicate packet. * It expects the data packet with sequence number-1 but receiving the data packet with sequence number-0. * It discards the duplicate data packet and re-sends acknowledgement ACK-1. * ACK-1 requests the sender to send a data packet with sequence number-1.     **Step-06:**     * Two acknowledgements ACK1 reaches the sender. * When first acknowledgement ACK1 reaches the sender, sender sends the next data packet with sequence number 1. * When second acknowledgement ACK1 reaches the sender, sender rejects the duplicate acknowledgement. * This is because it has already sent the data packet with sequence number-1 and now sender expects the acknowledgement with sequence number 0 from the receiver.     **Conclusion-**     * Had the sequence numbers not been allotted to the acknowledgements, sender would have accepted the duplicate acknowledgement thinking of it as the new acknowledgement for the latest data packet sent by it. * This is how sequence numbers allotted to the acknowledgements prove to be useful for identifying duplicate acknowledgements and discarding them. |

**4. Problem of Damaged Packet-**

* If receiver receives a corrupted data packet from the sender, it sends a negative acknowledgement (NAK) to the sender.
* NAK requests the sender to send the data packet again.



**Stop and Wait Protocol Vs Stop and Wait ARQ-**

The following comparison table states the differences between the two protocols-

|  |  |
| --- | --- |
| **Stop and Wait Protocol** | **Stop and Wait ARQ** |
| It assumes that the communication channel is perfect and noise free. | It assumes that the communication channel is imperfect and noisy. |
| Data packet sent by the sender can never get corrupt. | Data packet sent by the sender may get corrupt. |
| There is no concept of negative acknowledgements. | A negative acknowledgement is sent by the receiver if the data packet is found to be corrupt. |
| There is no concept of time out timer. | Sender starts the time out timer after sending the data packet. |
| There is no concept of sequence numbers. | Data packets and acknowledgements are numbered using sequence numbers. |

**Limitation of Stop and Wait ARQ-**

|  |
| --- |
| The major limitation of Stop and Wait ARQ is its very less efficiency. |

To increase the efficiency, protocols like [**Go back N**](https://www.gatevidyalay.com/go-back-n-sliding-window-protocol/) and [**Selective Repeat**](https://www.gatevidyalay.com/selective-repeat-sliding-window-protocol/) are used.

**Explanation-**

In stop and wait ARQ,

* Sender window size is 1.
* This allows the sender to keep only one frame unacknowledged.
* So, sender sends one frame and then waits until the sent frame gets acknowledged.
* After receiving the acknowledgement from the receiver, sender sends the next frame.

