Introduction to Computer Networks

CS640

L2 Reliable Transmission

https://pages.cs.wisc.edu/~mgliu/CS640/S26/index.html

Ming Liu
mgliu@cs.wisc.edu

Outline

- Last
  - STP
  - Ethernet
- Today
  - Reliable transmission at L2
- Announcements
  - Lab2 released today

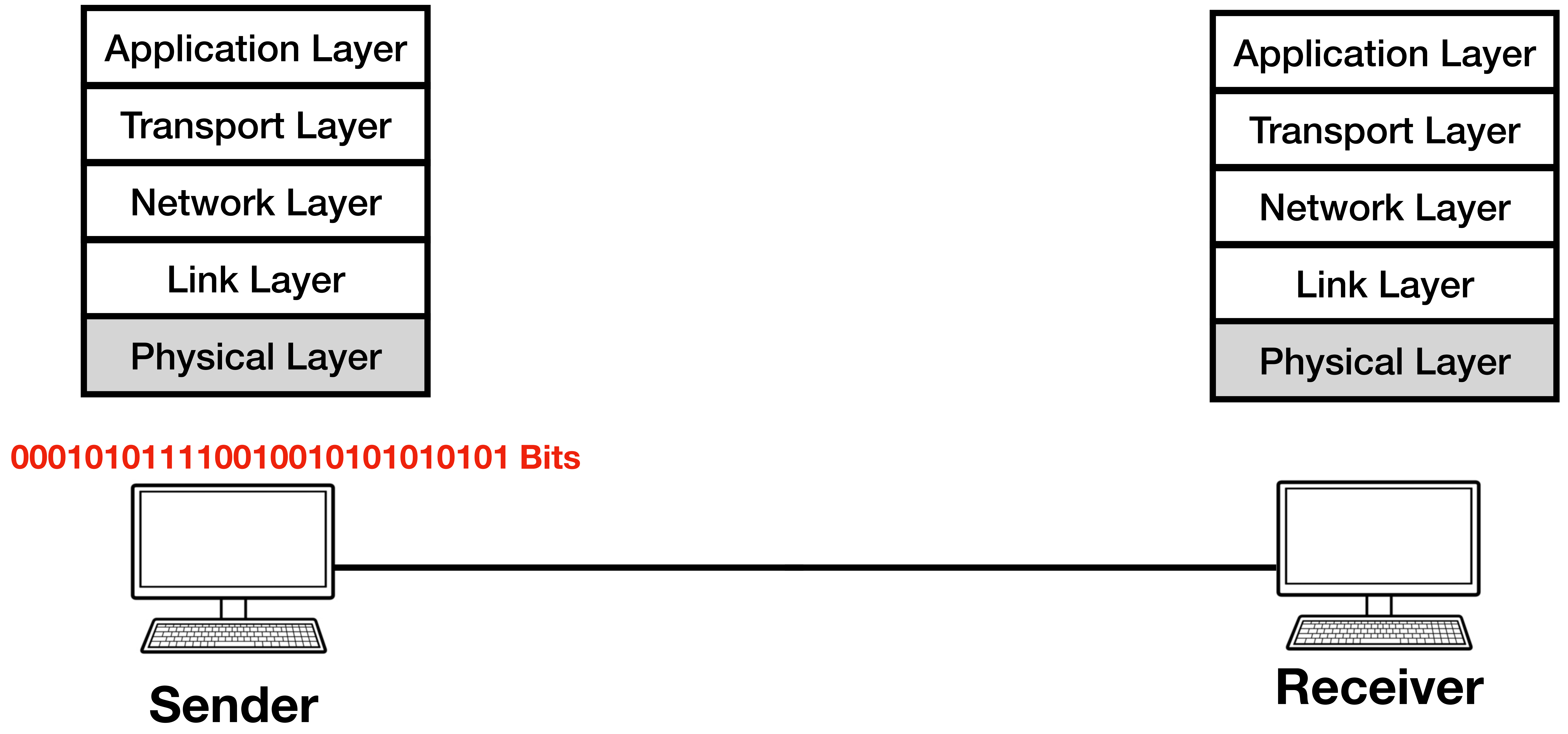
Recap

- Key Questions
  - How does STP work?
  - How has Ethernet evolved in the past decades?

- Terminology
  - CSMA/CD
  - Exponential back-off

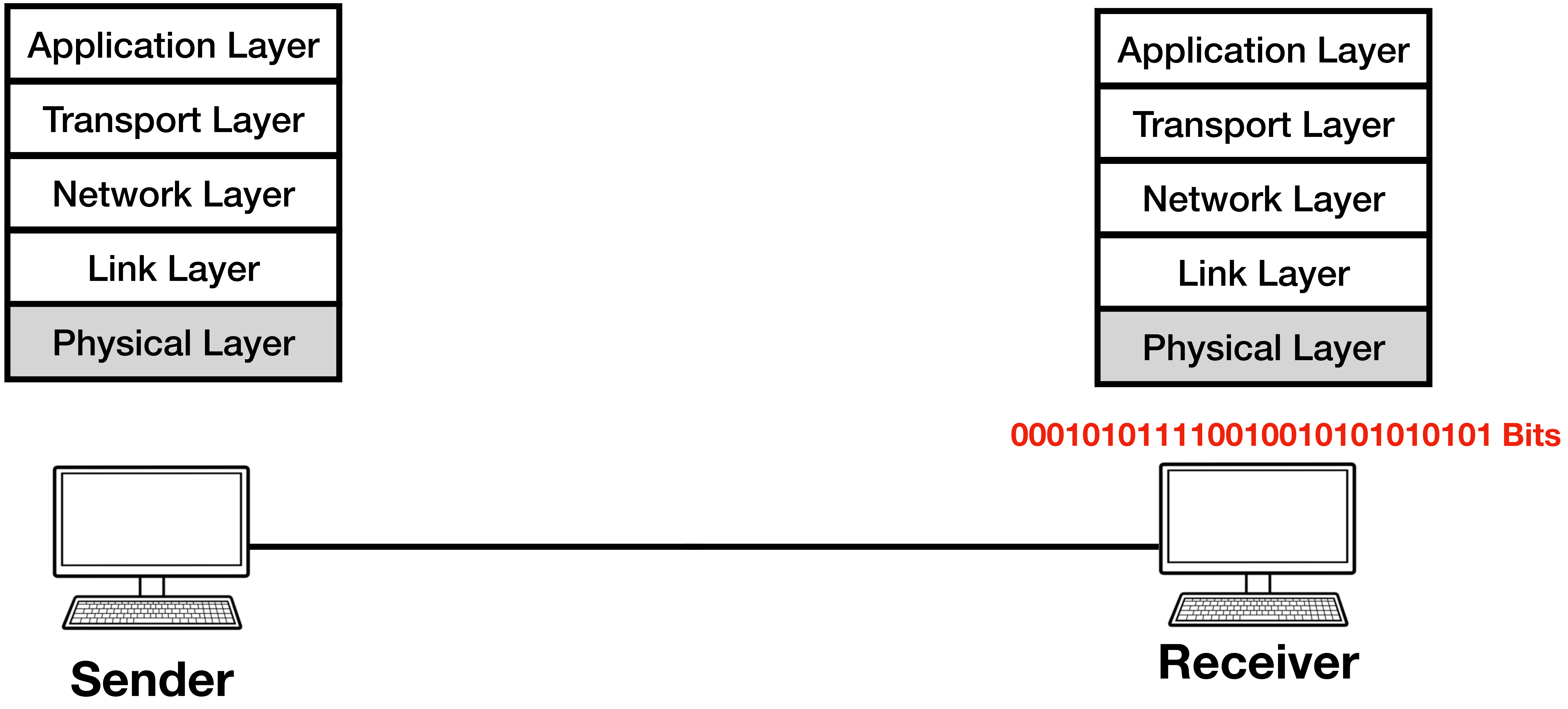
Transmission @Physical Layer

- Bitstreams transmitted between two directly connected hosts



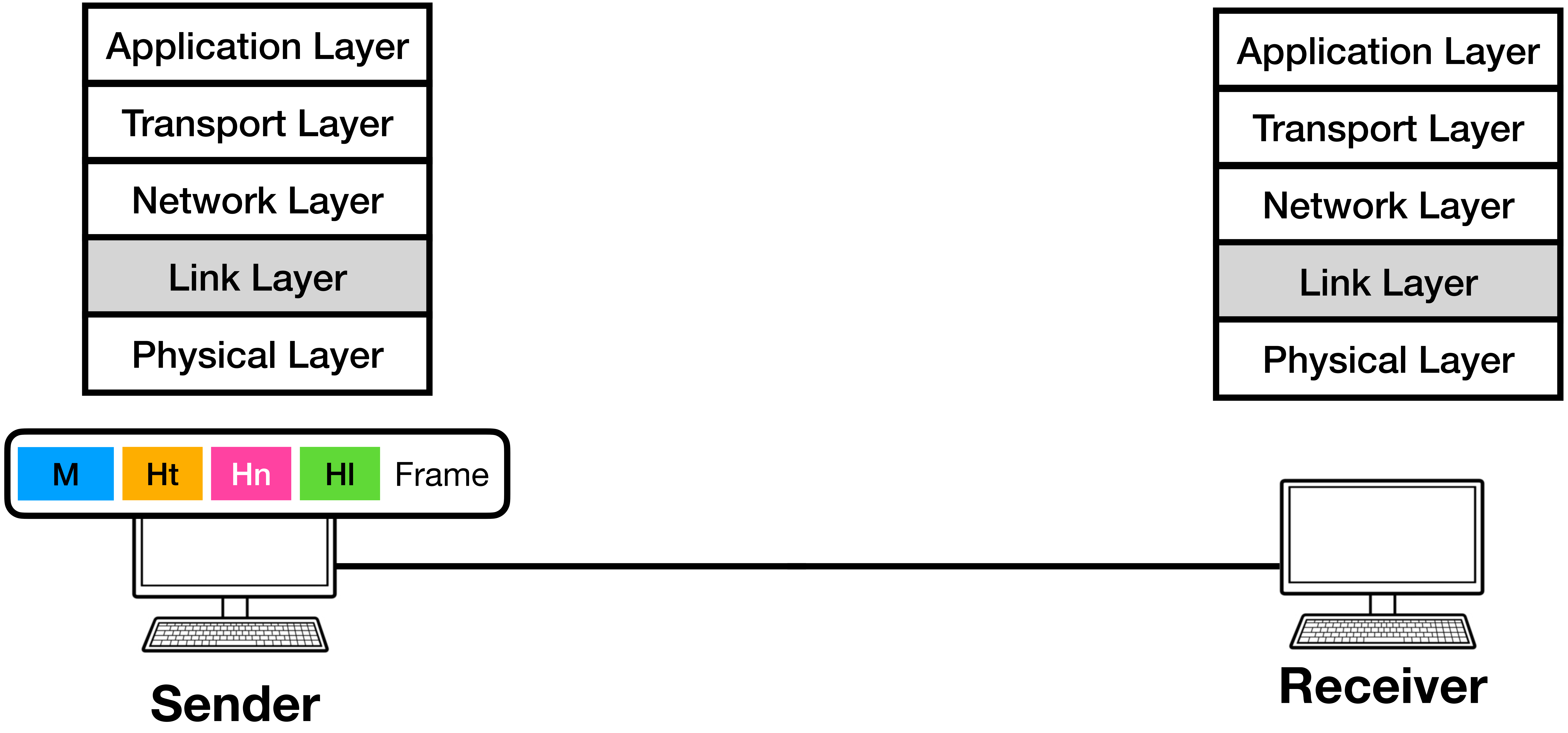
Reliable Transmission @Physical Layer

- Reliable transmission is essential



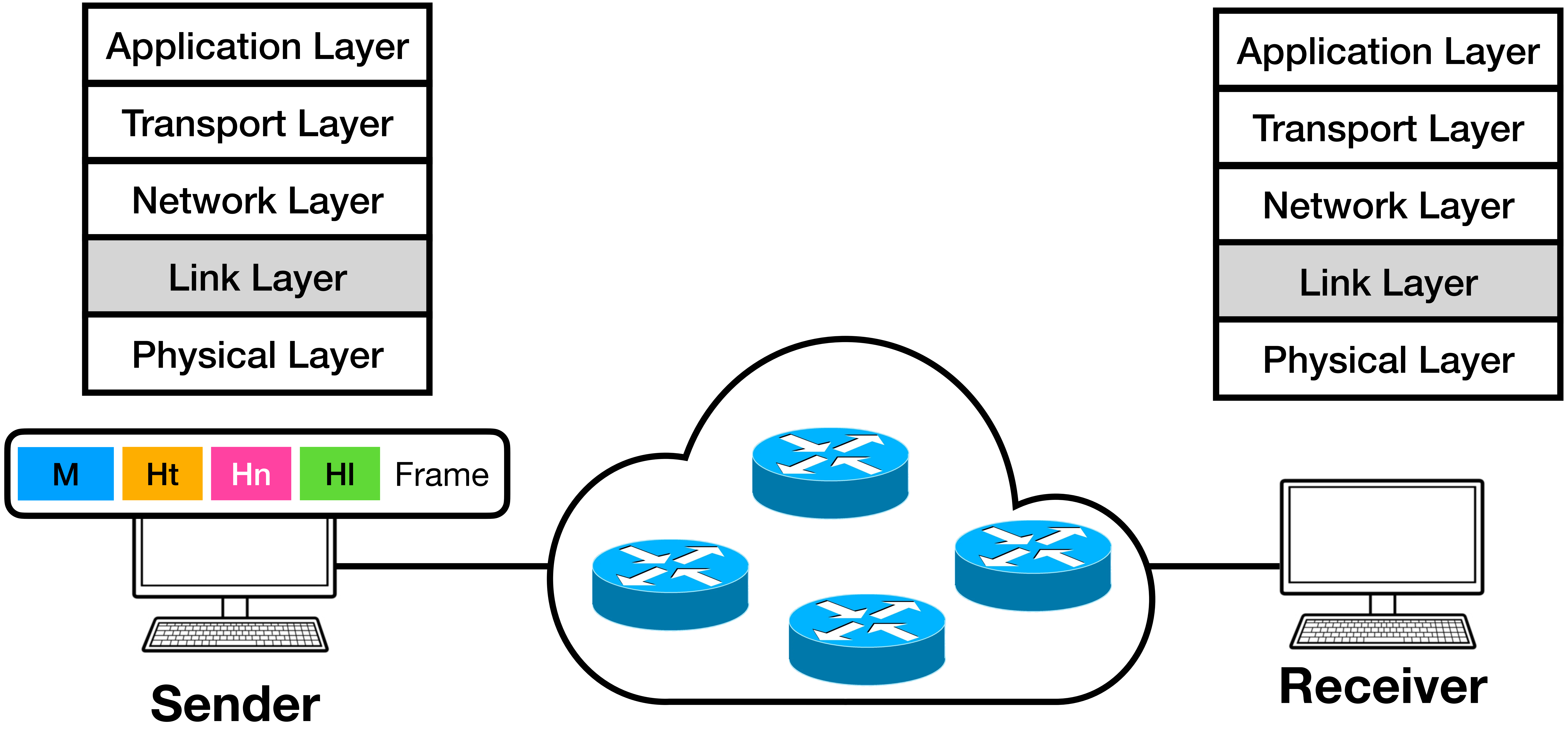
Transmission @Link Layer

- Frames transmitted between two (in)directly connected hosts



Transmission @Link Layer

- Frames transmitted between two (in)directly connected hosts



Reliable transmission is not necessary in the link layer!

Unreliable Link Layer

- Common errors
  - Frames are corrupted during transmission
  - Frames are dropped due to cable errors
  - Frames are dropped due to the SW/HW failures at the switch

Unreliable Link Layer

- Common errors
  - Frames are corrupted during transmission
  - Frames are dropped due to cable errors
  - Frames are dropped due to the SW/HW failures at the switch

- Best-effort transmission
  - For example, Ethernet does not handle reliability
  - Rely on the upper layer across the stack to take care

Unreliable Link Layer

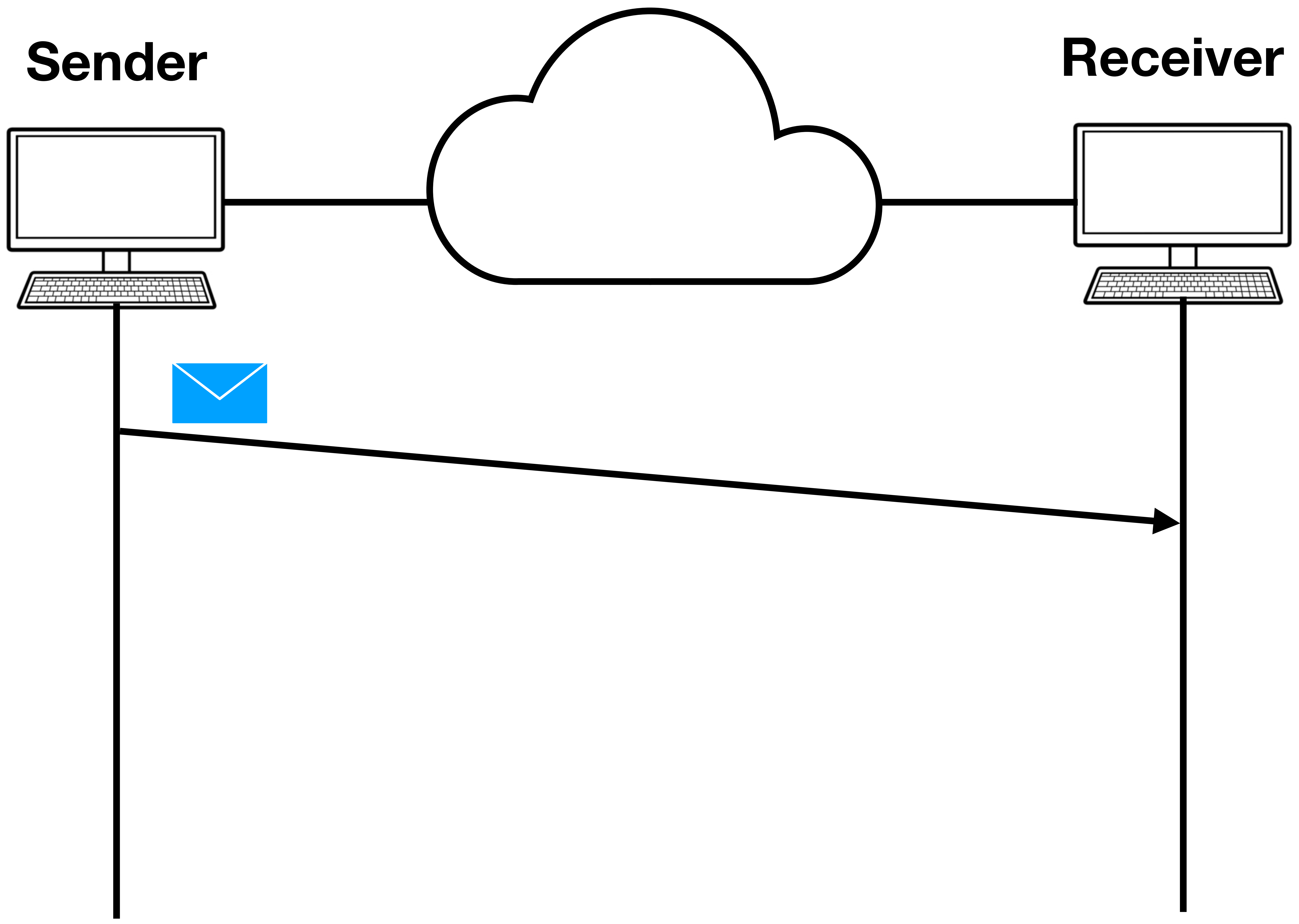
- Common errors
  - Frames are corrupted during transmission
  - Frames are dropped due to cable errors
  - Frames are dropped due to the SW/HW failures at the switch

- Best-effort transmission
  - For example, Ethernet does not handle reliability
  - Rely on the upper layer across the stack to take care

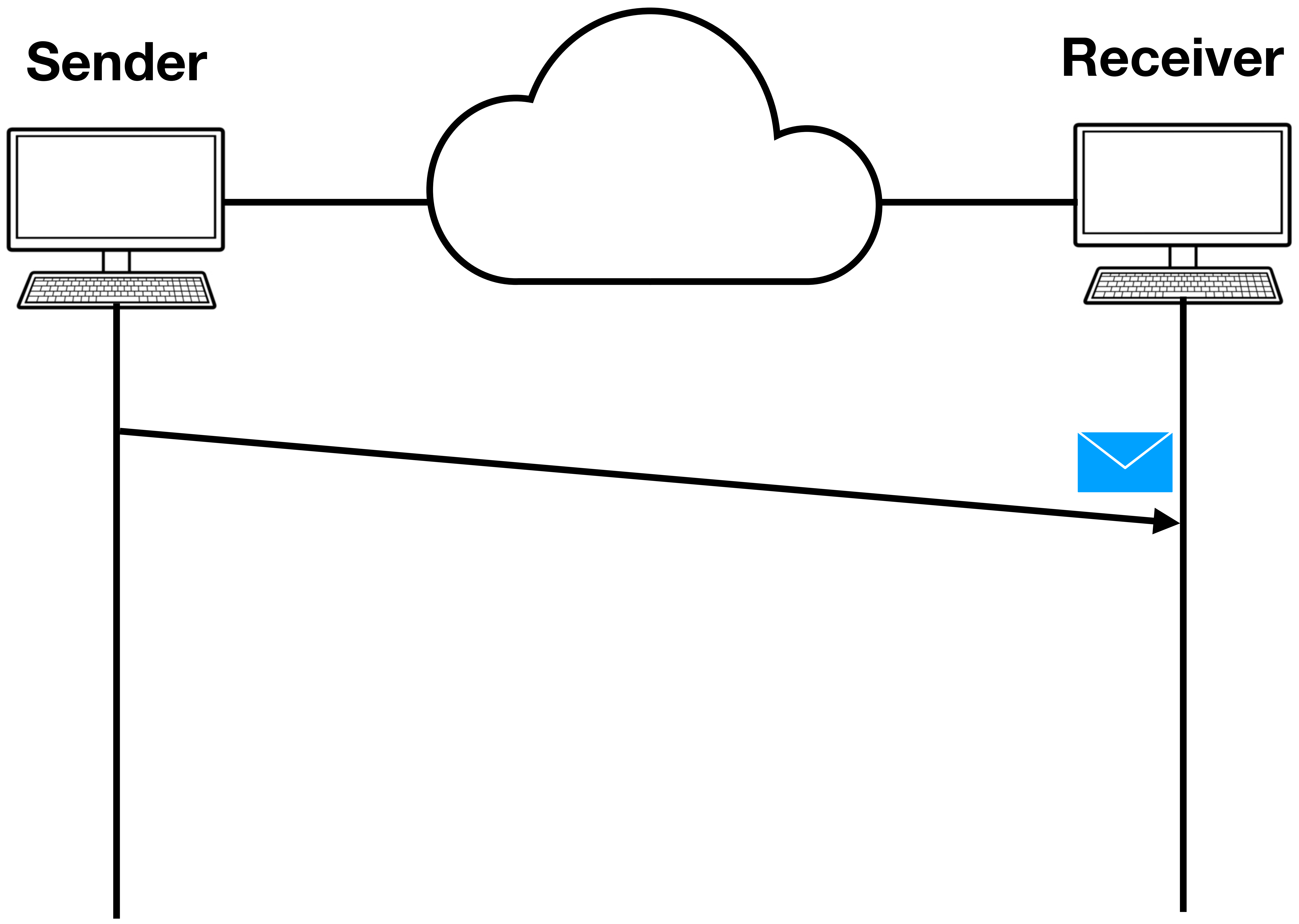
A reliable link layer can simplify the transport layer design (we'll discuss this later).

Why it is hard to achieve reliable transmission?

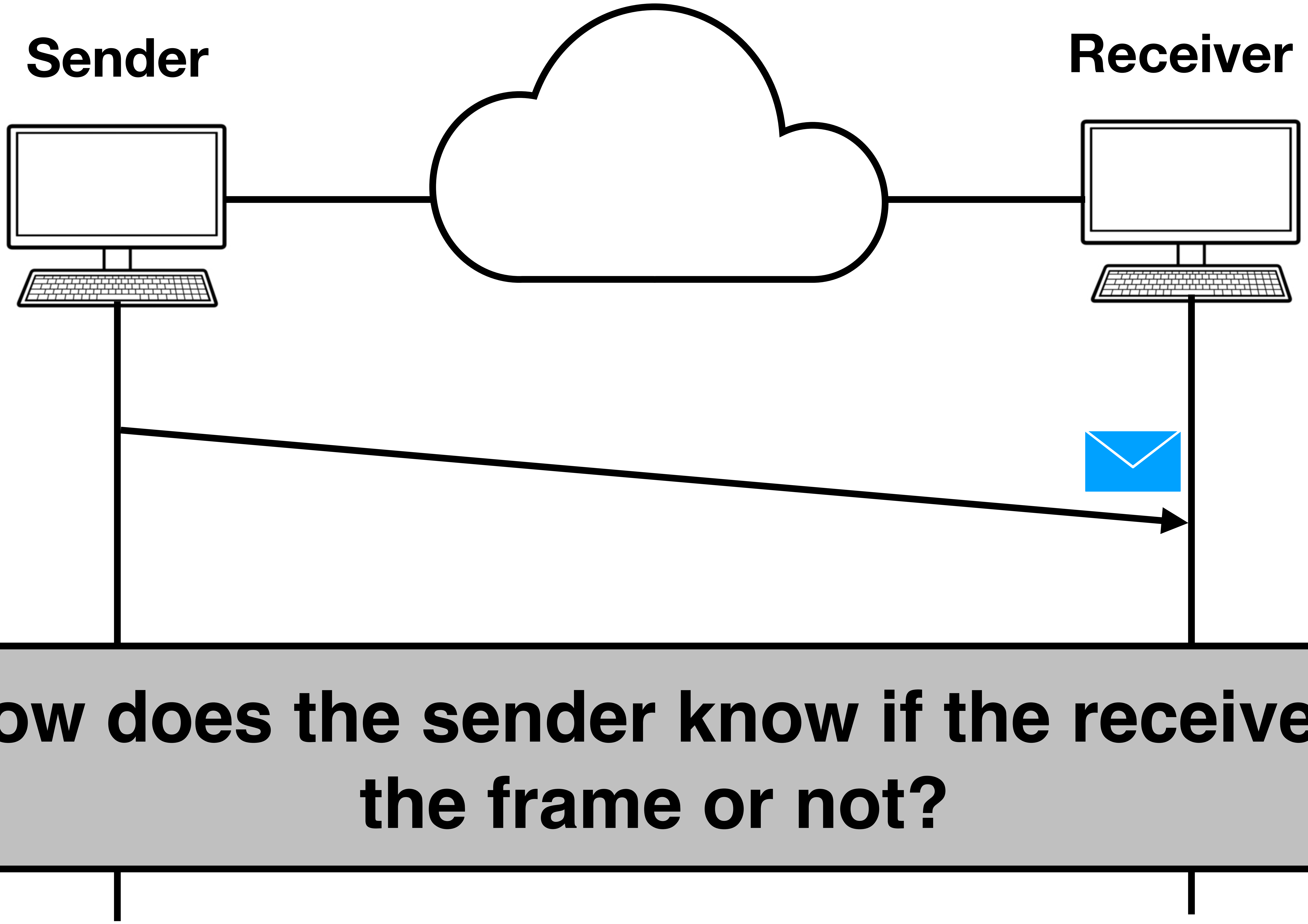
A Reliable Transmission Example



A Reliable Transmission Example

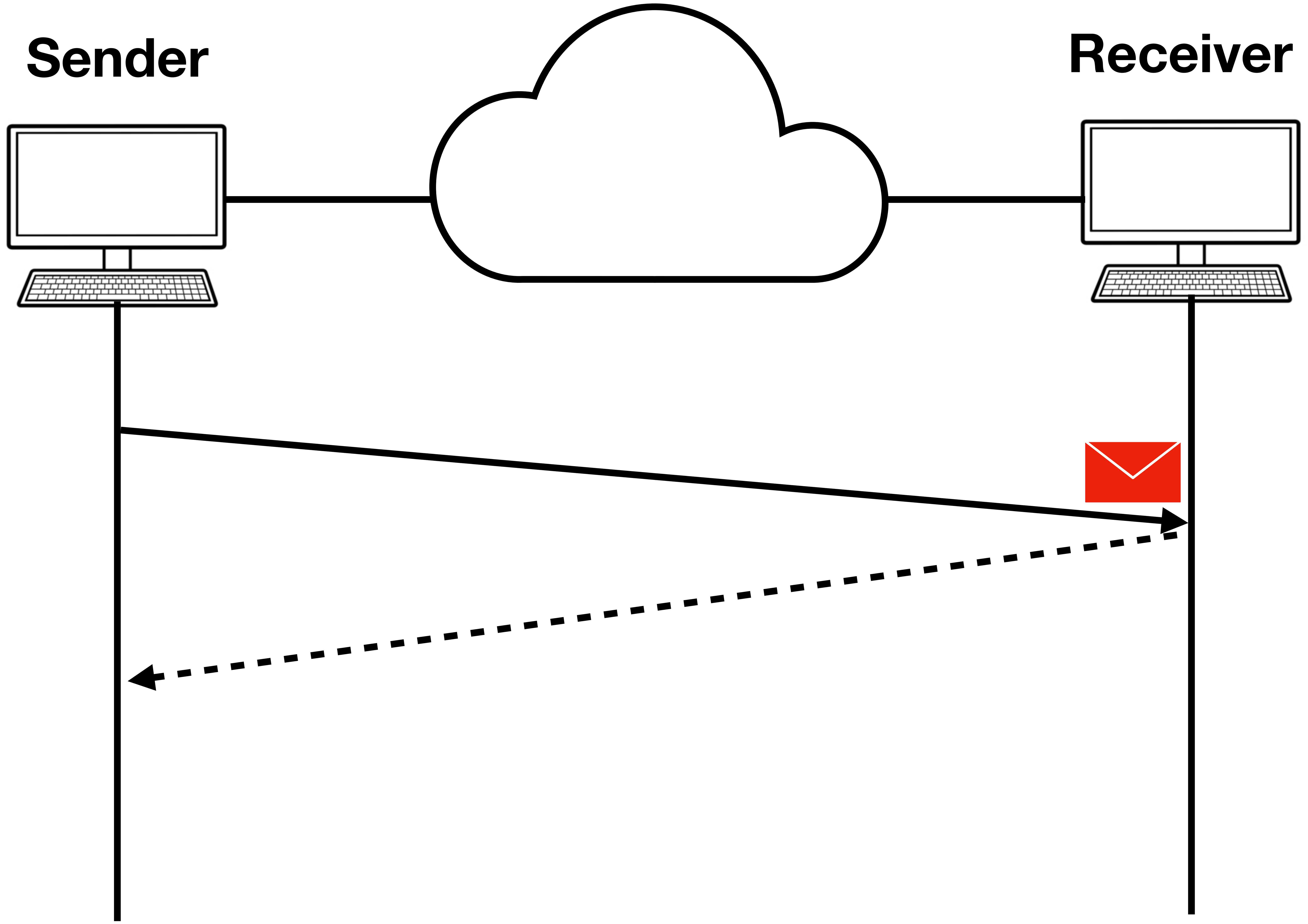


A Reliable Transmission Example

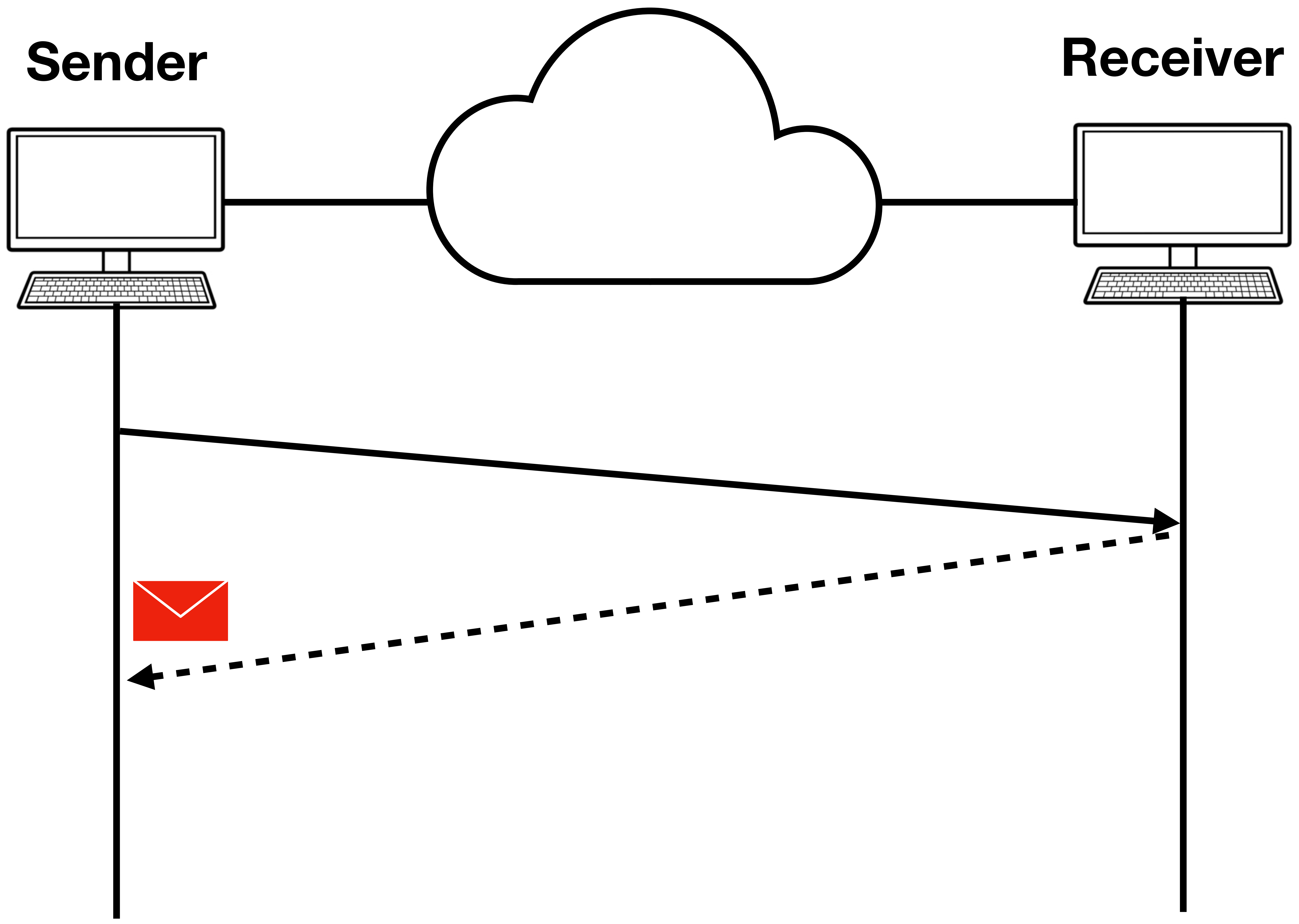


Q1: How does the sender know if the receiver gets the frame or not?

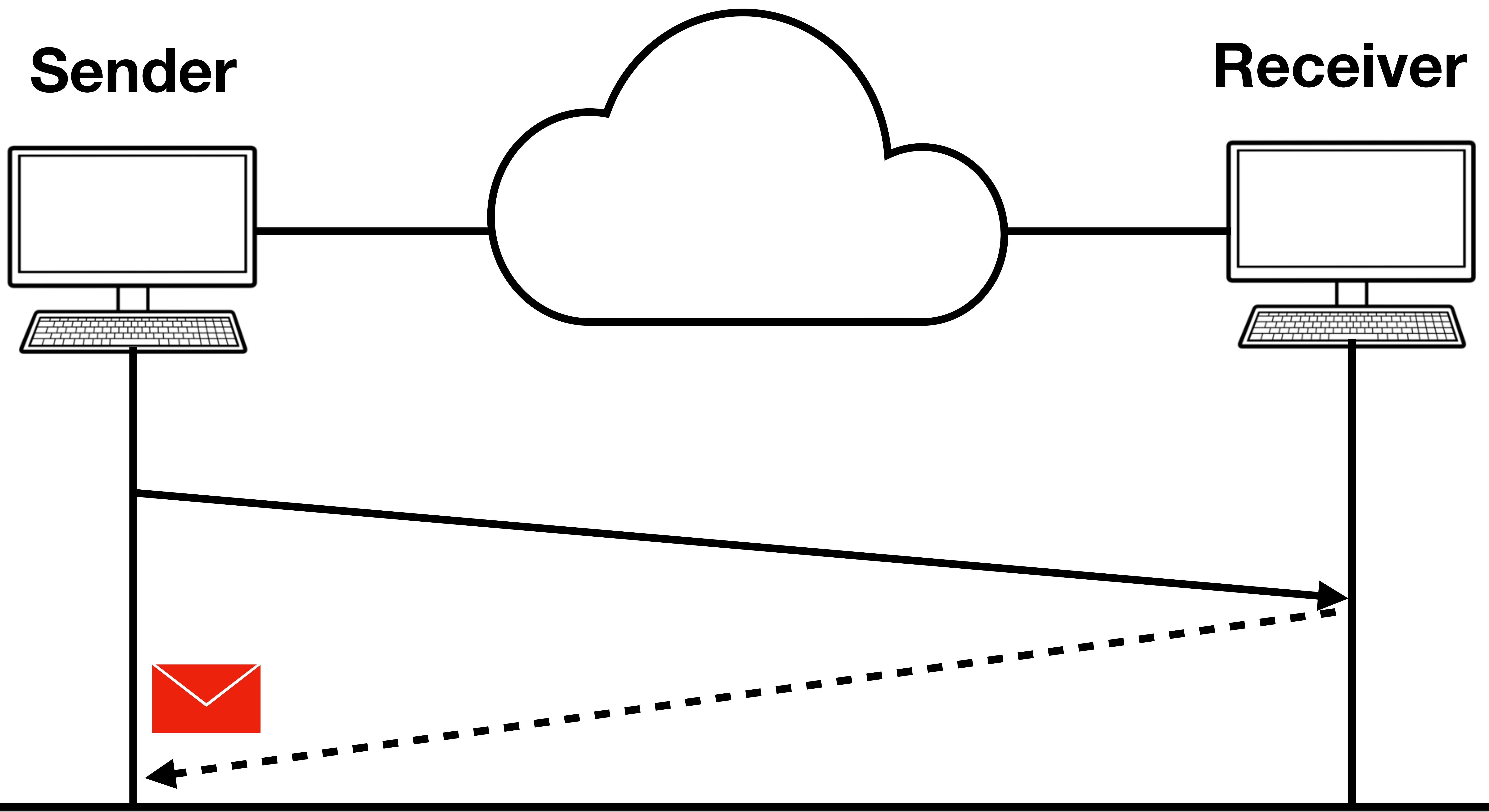
A Reliable Transmission Example



A Reliable Transmission Example

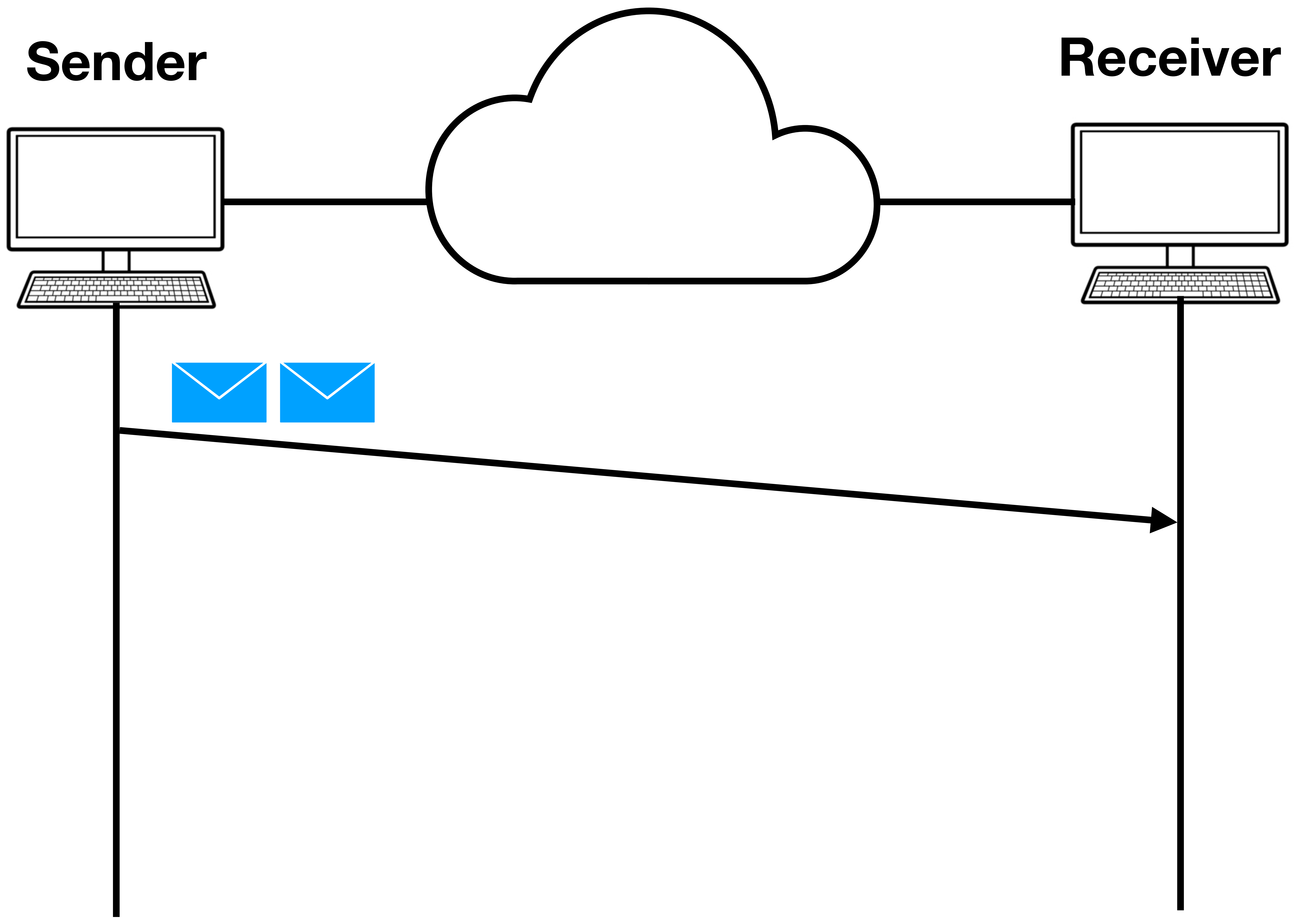


A Reliable Transmission Example

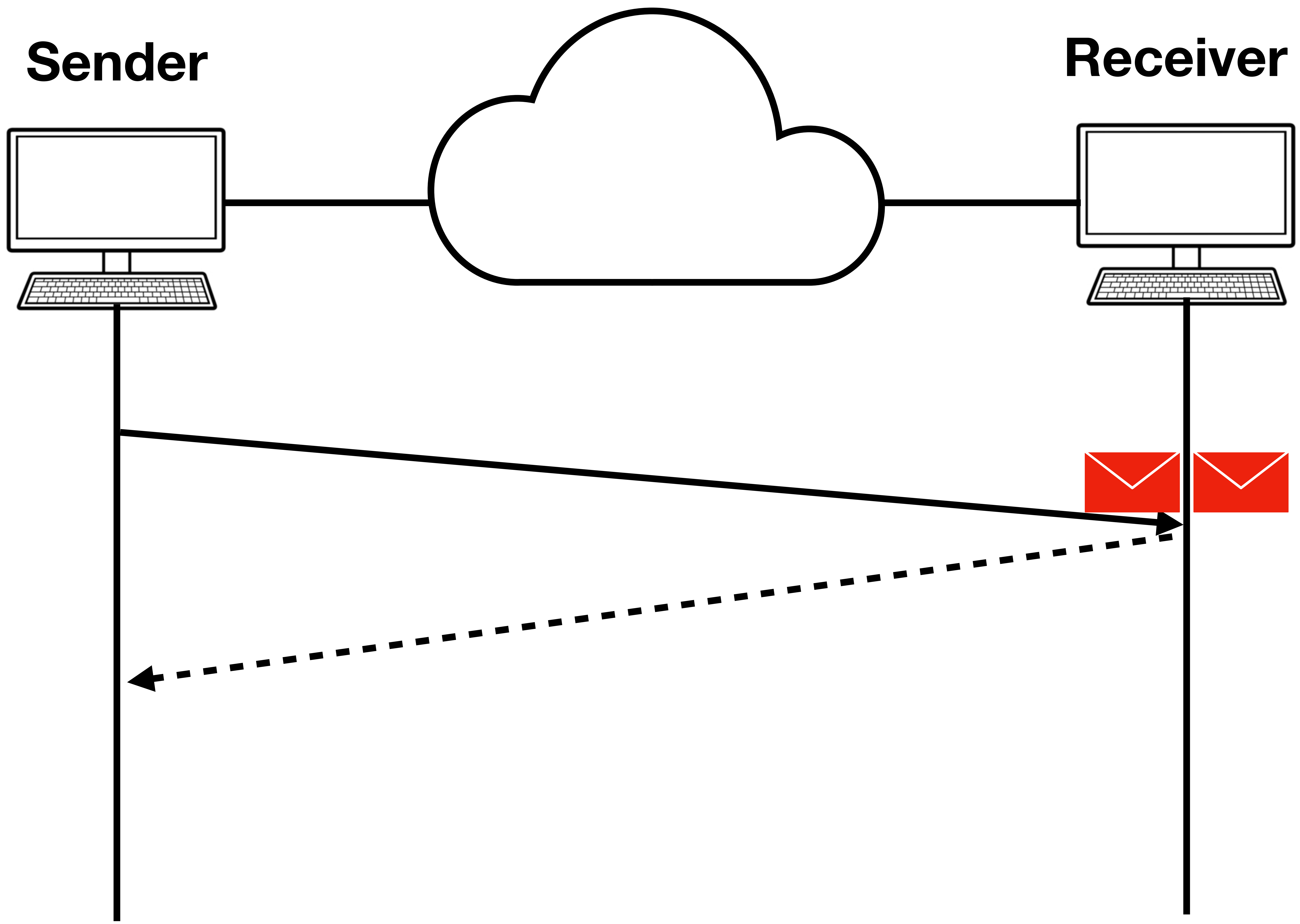


The receiver must explicitly tell the sender a frame is received — acknowledgment.

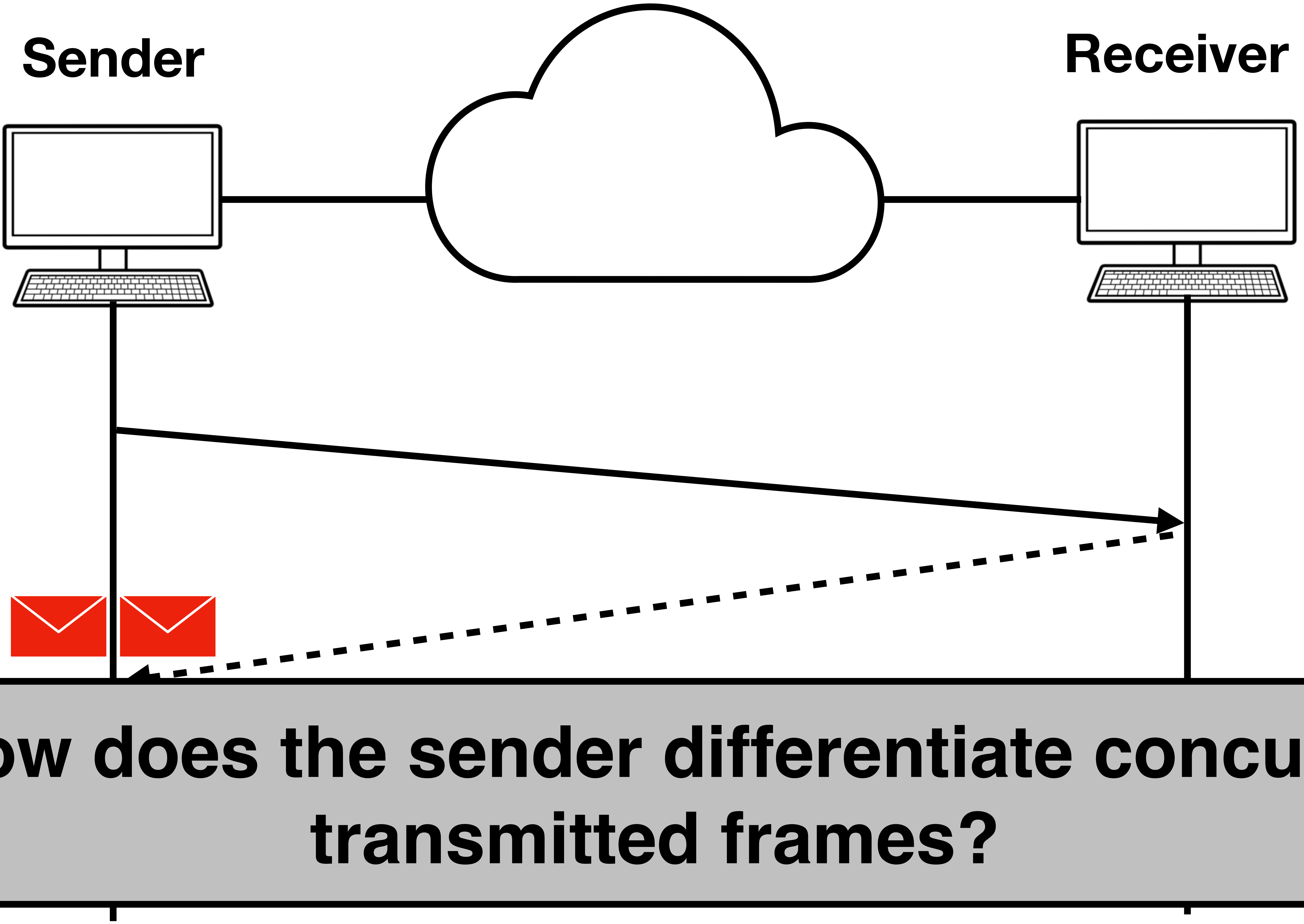
A Reliable Transmission Example



A Reliable Transmission Example

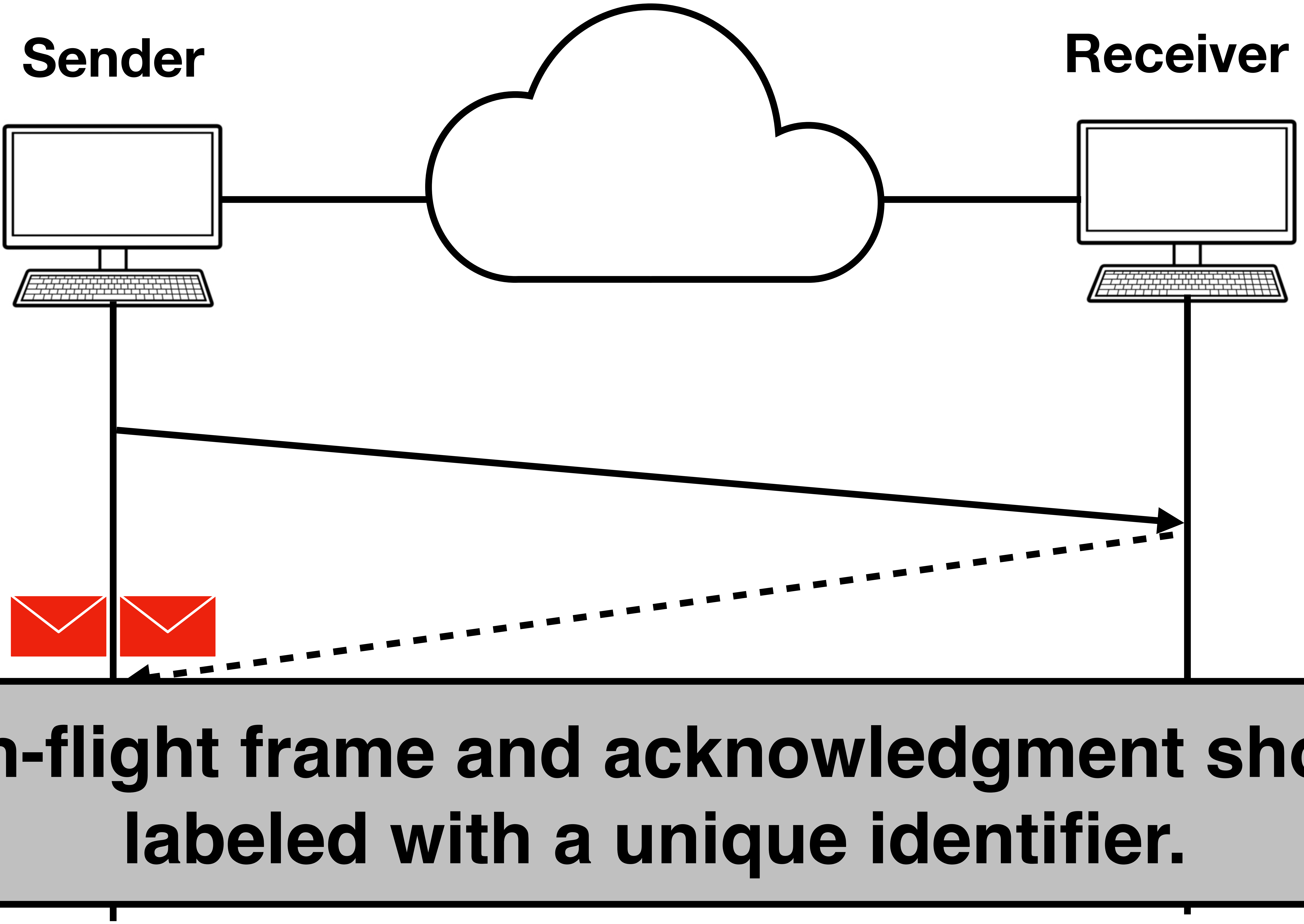


A Reliable Transmission Example



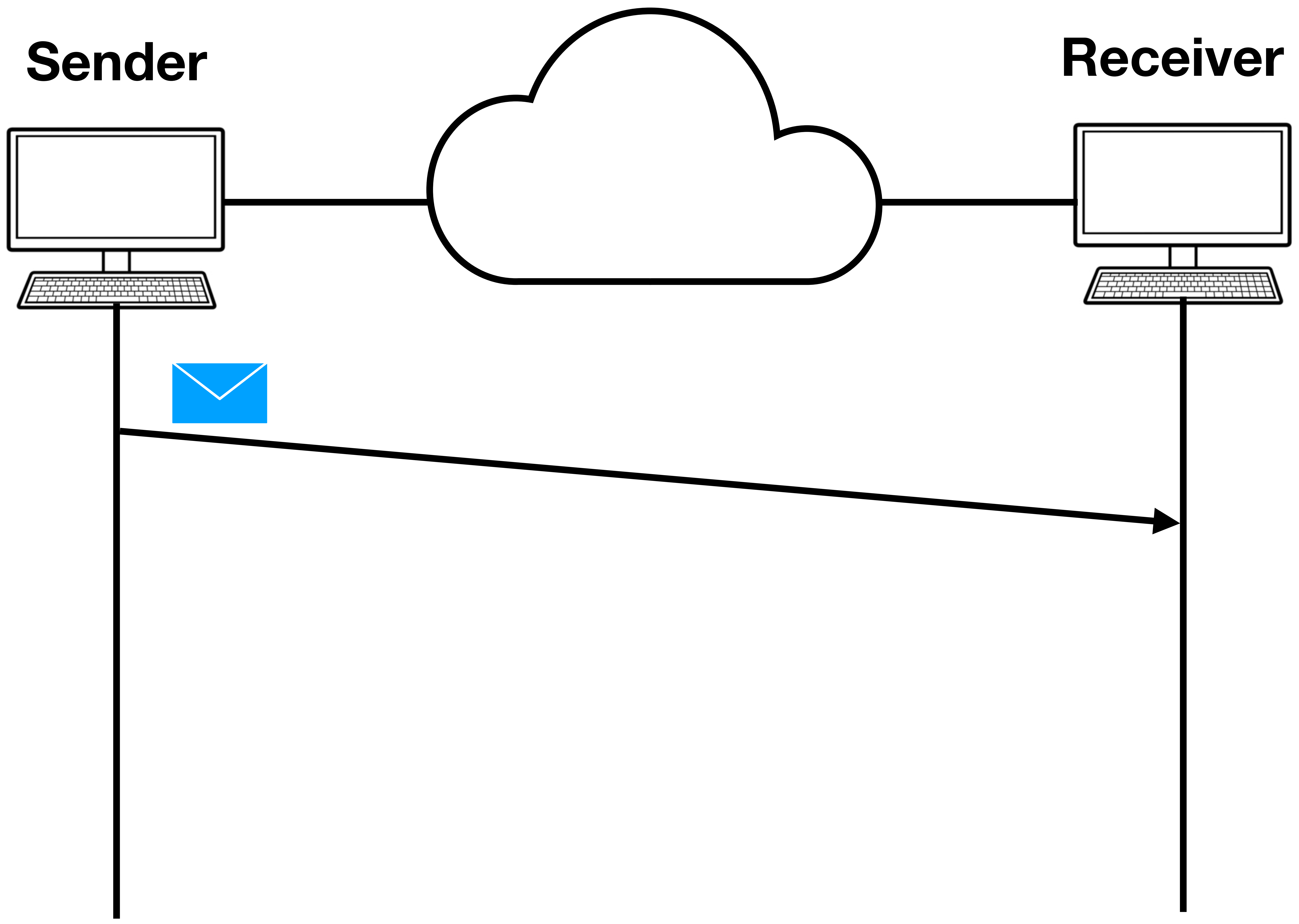
Q2: How does the sender differentiate concurrently transmitted frames?

A Reliable Transmission Example

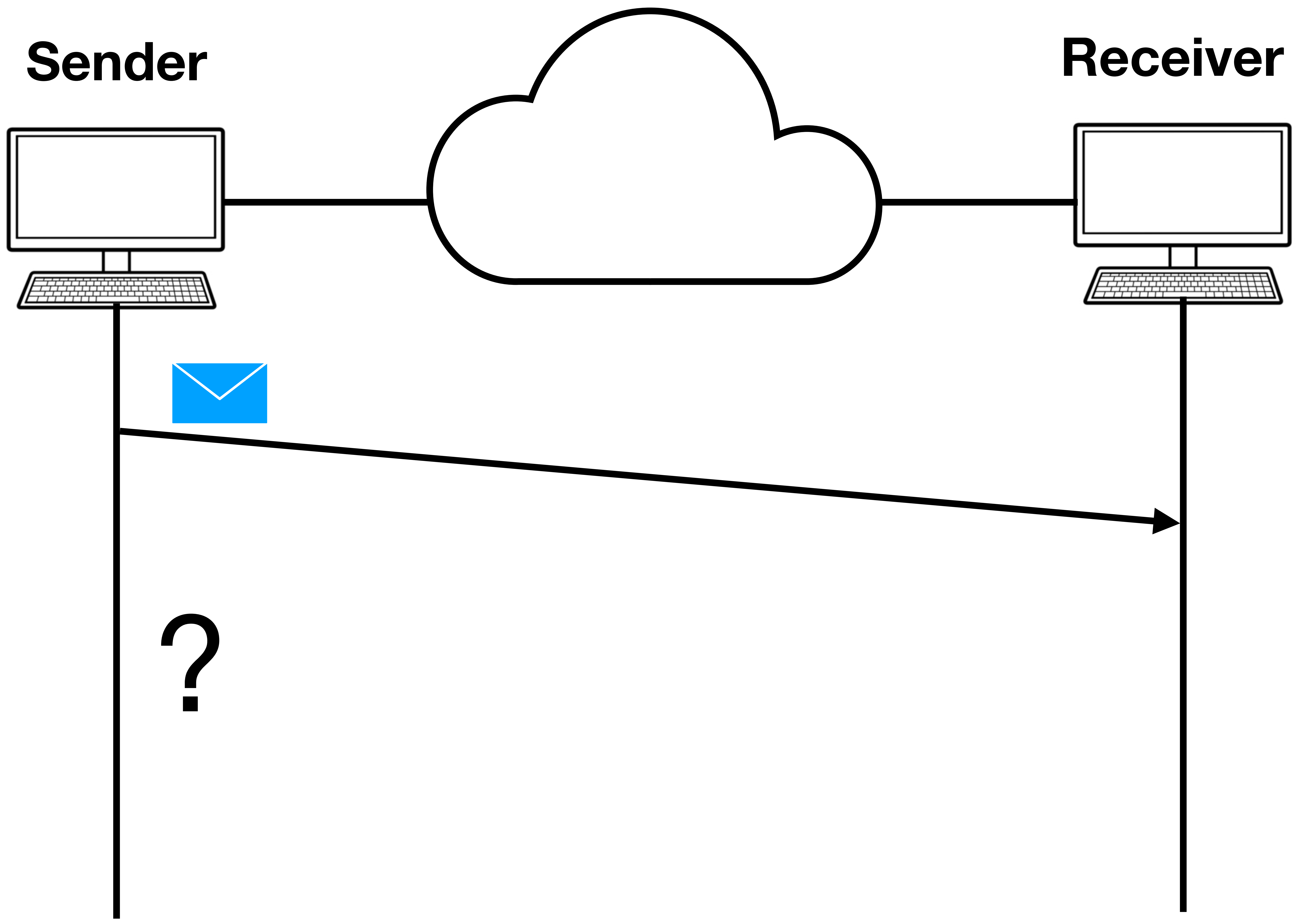


Each in-flight frame and acknowledgment should be labeled with a unique identifier.

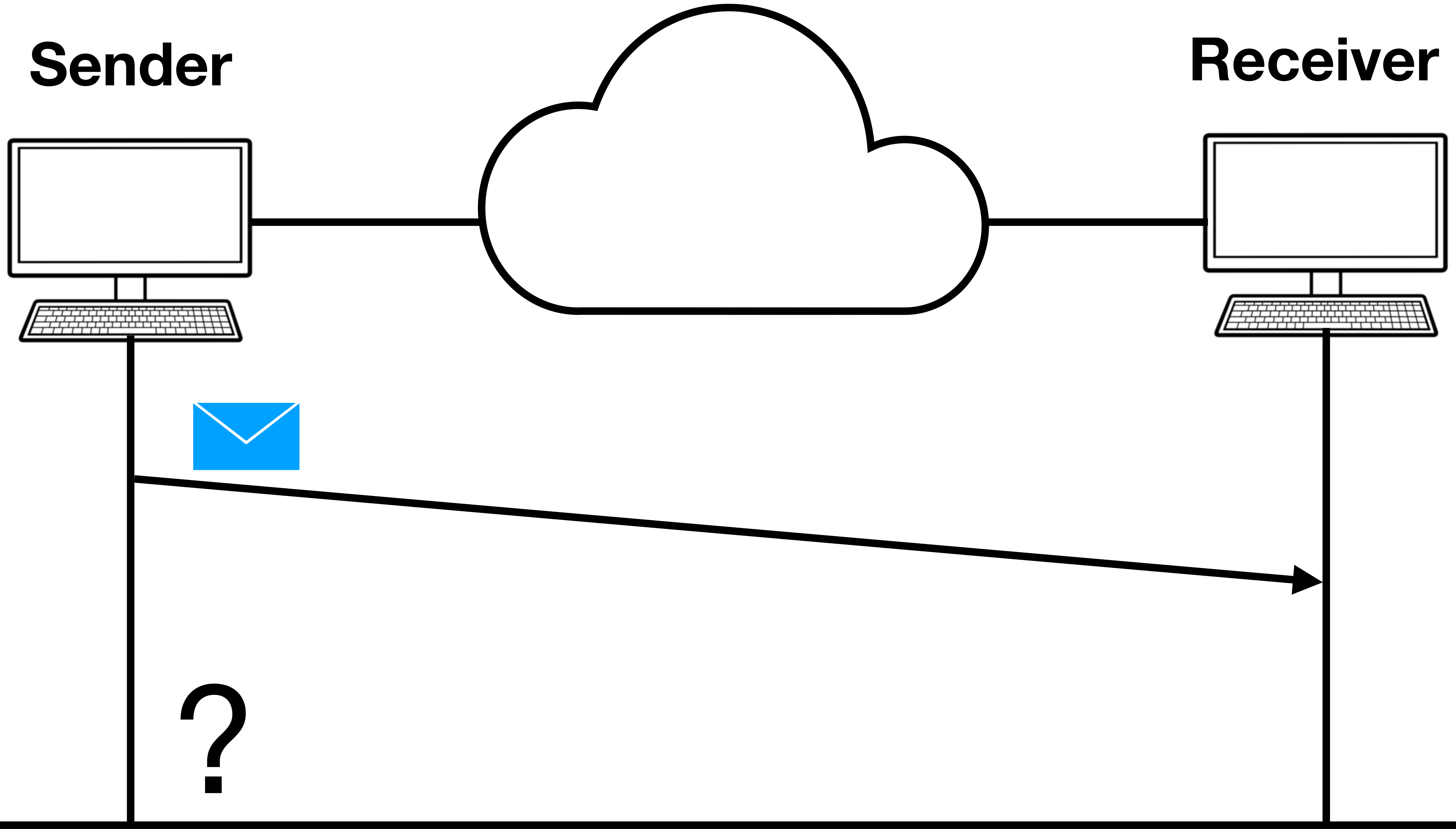
A Reliable Transmission Example



A Reliable Transmission Example

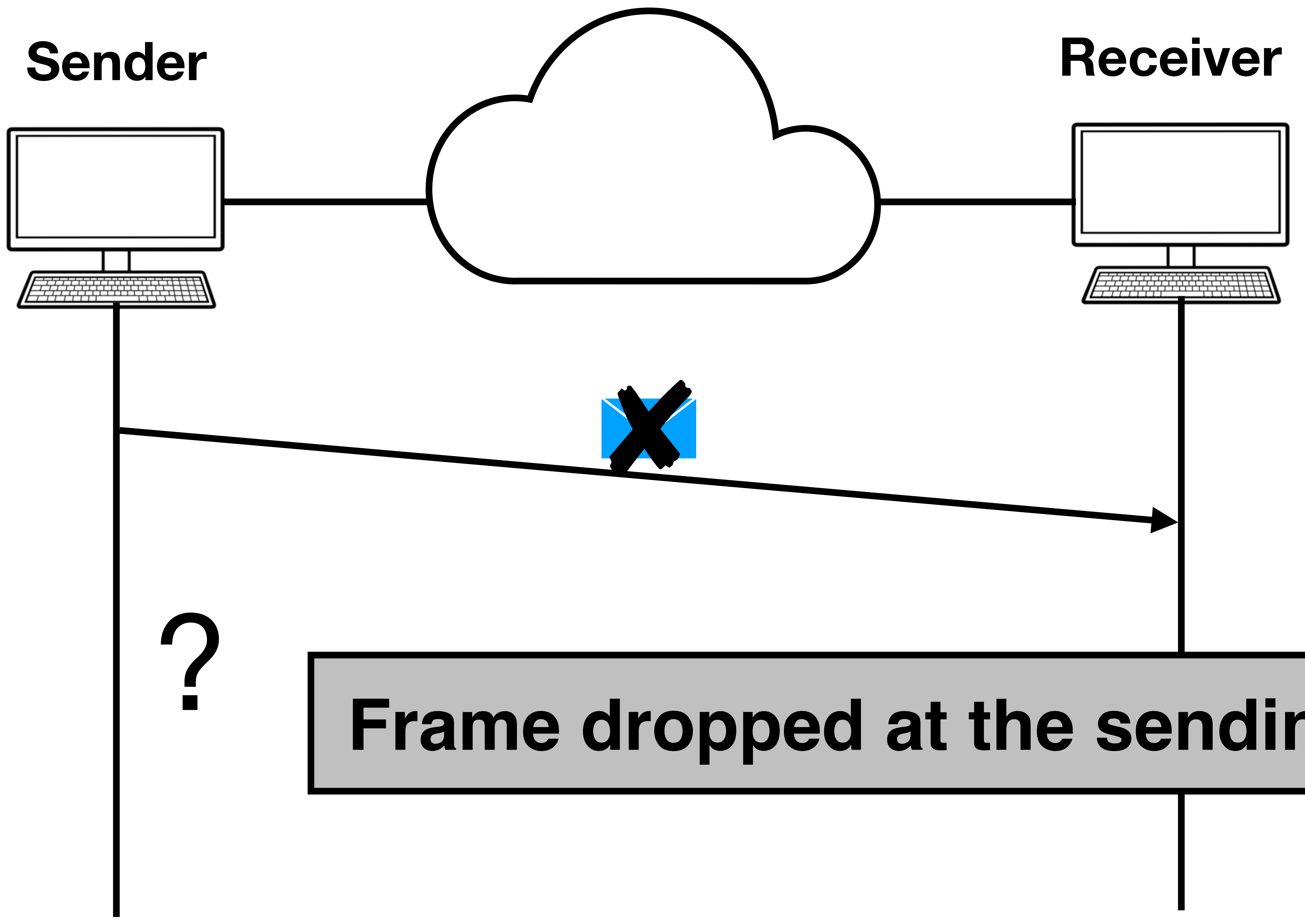


A Reliable Transmission Example



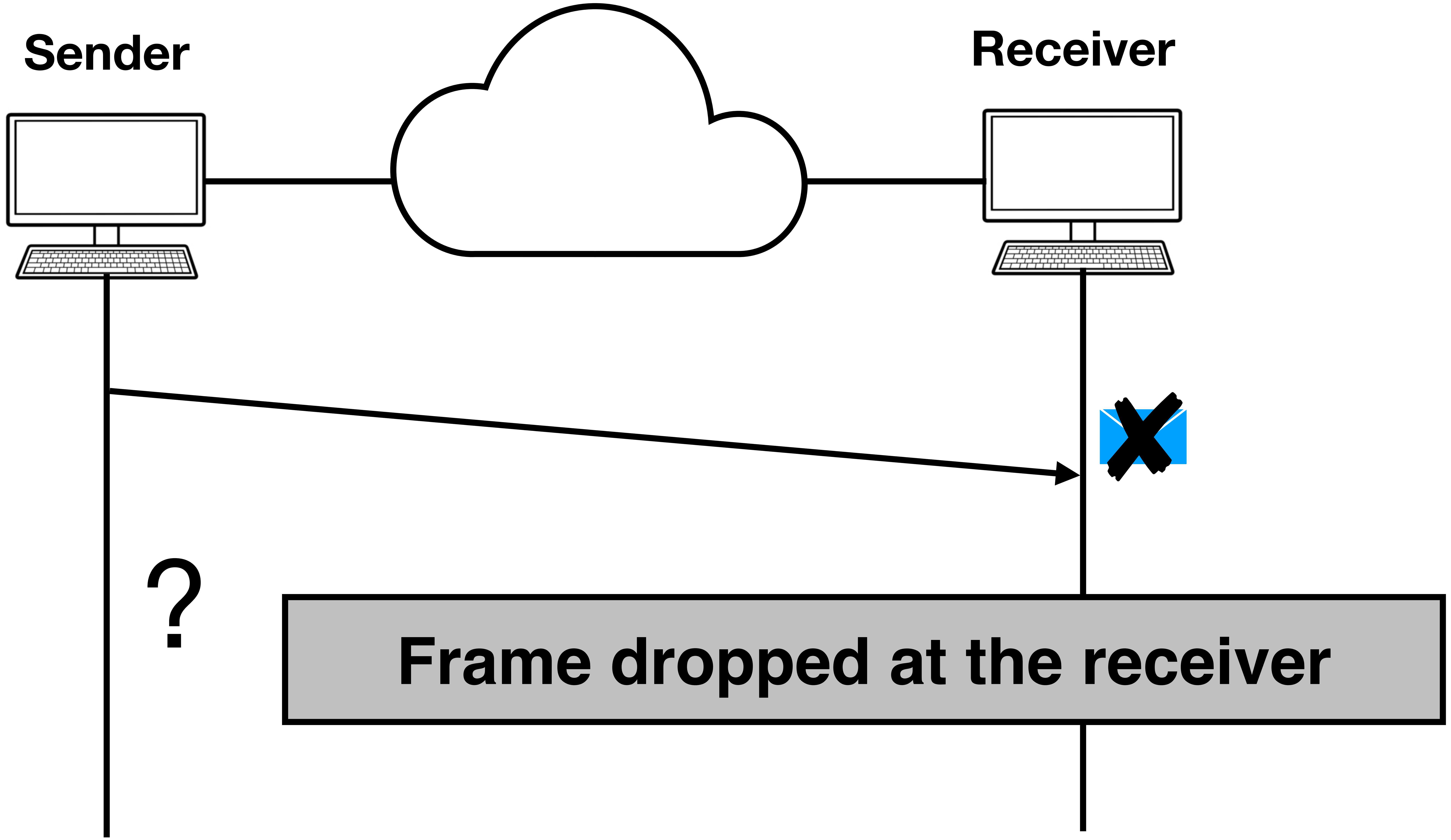
Q3: What happens if the sender doesn't receive the acknowledgment?

A Reliable Transmission Example



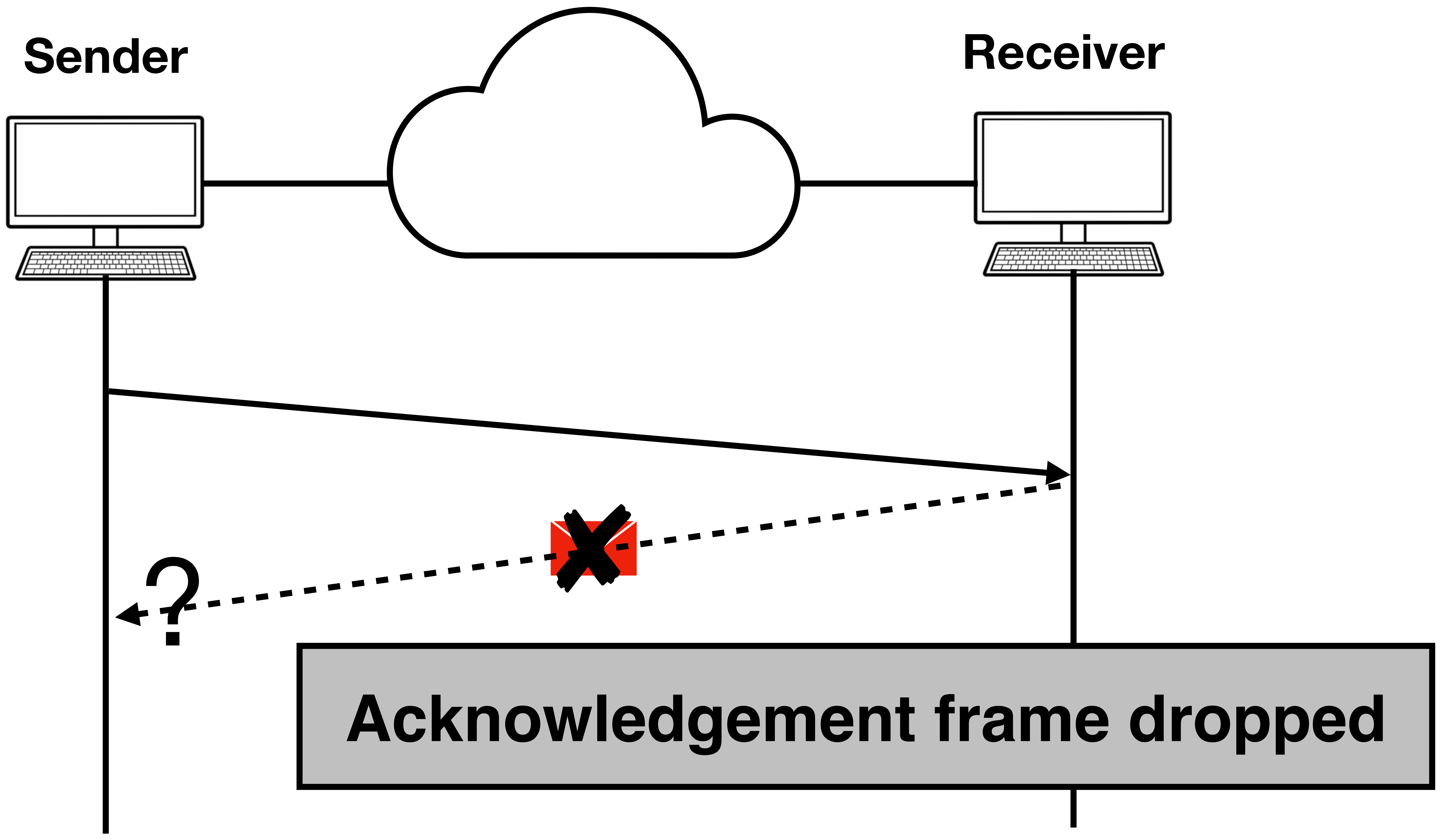
Frame dropped at the sending path

A Reliable Transmission Example

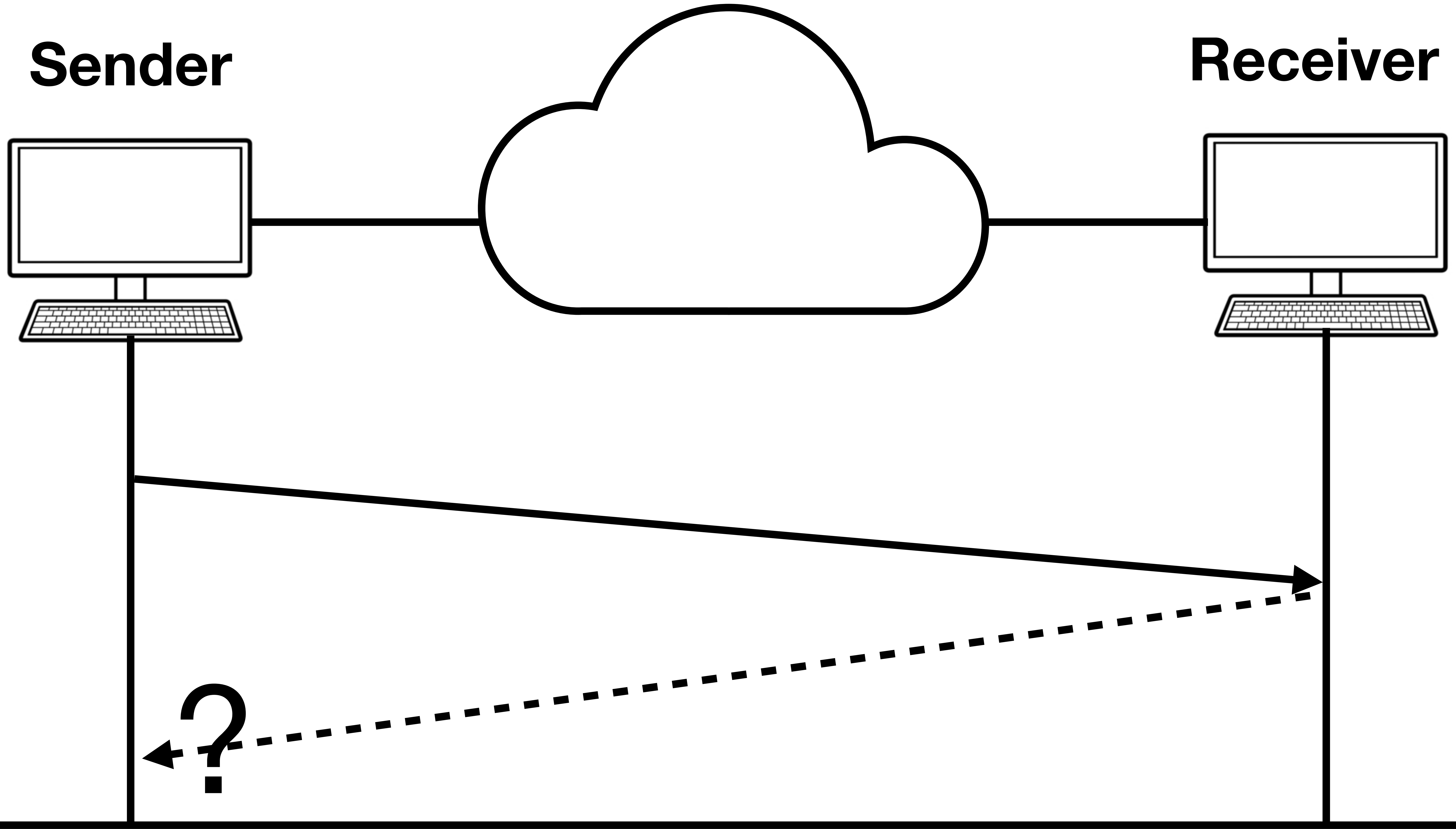


Frame dropped at the receiver

A Reliable Transmission Example

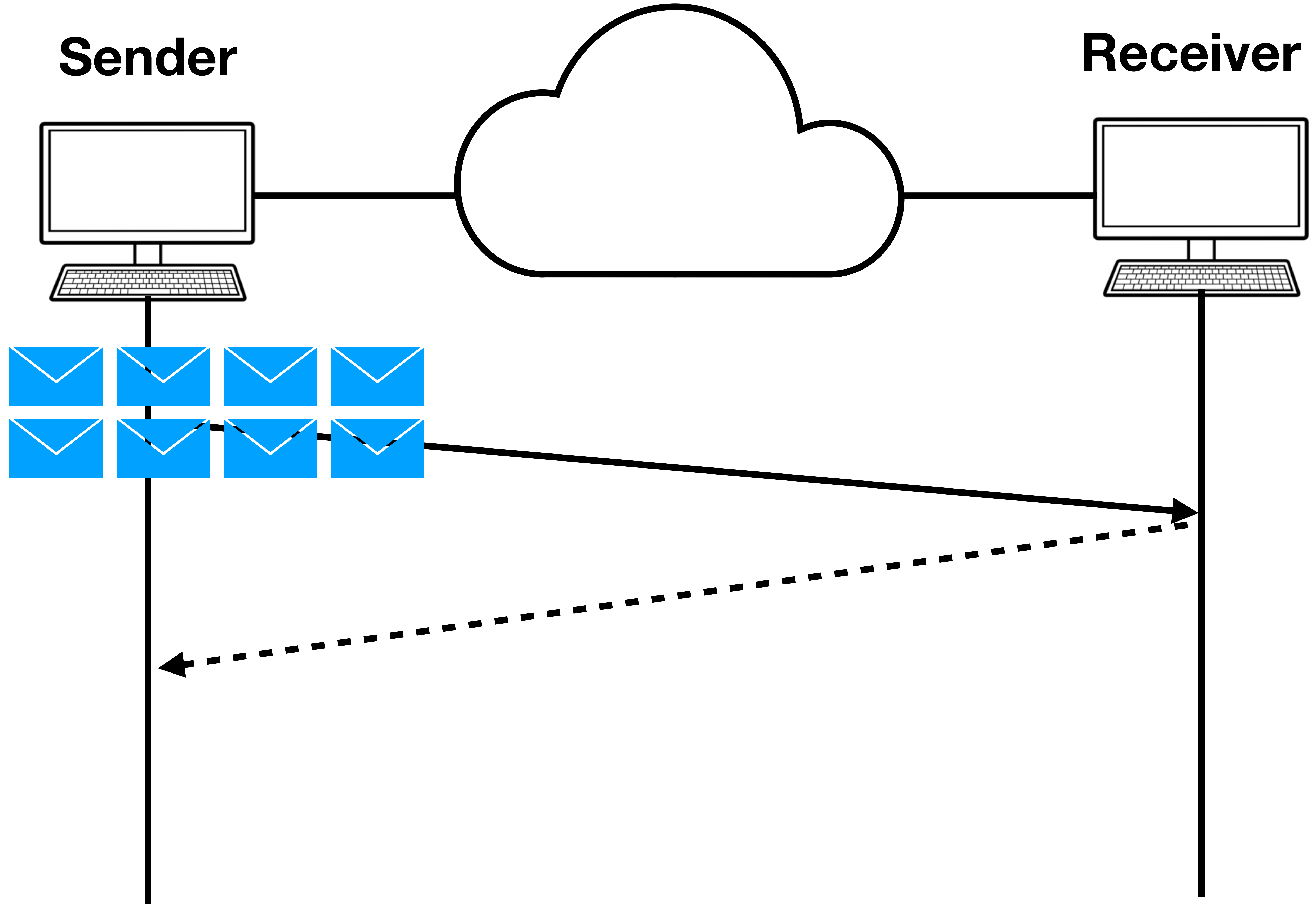


A Reliable Transmission Example

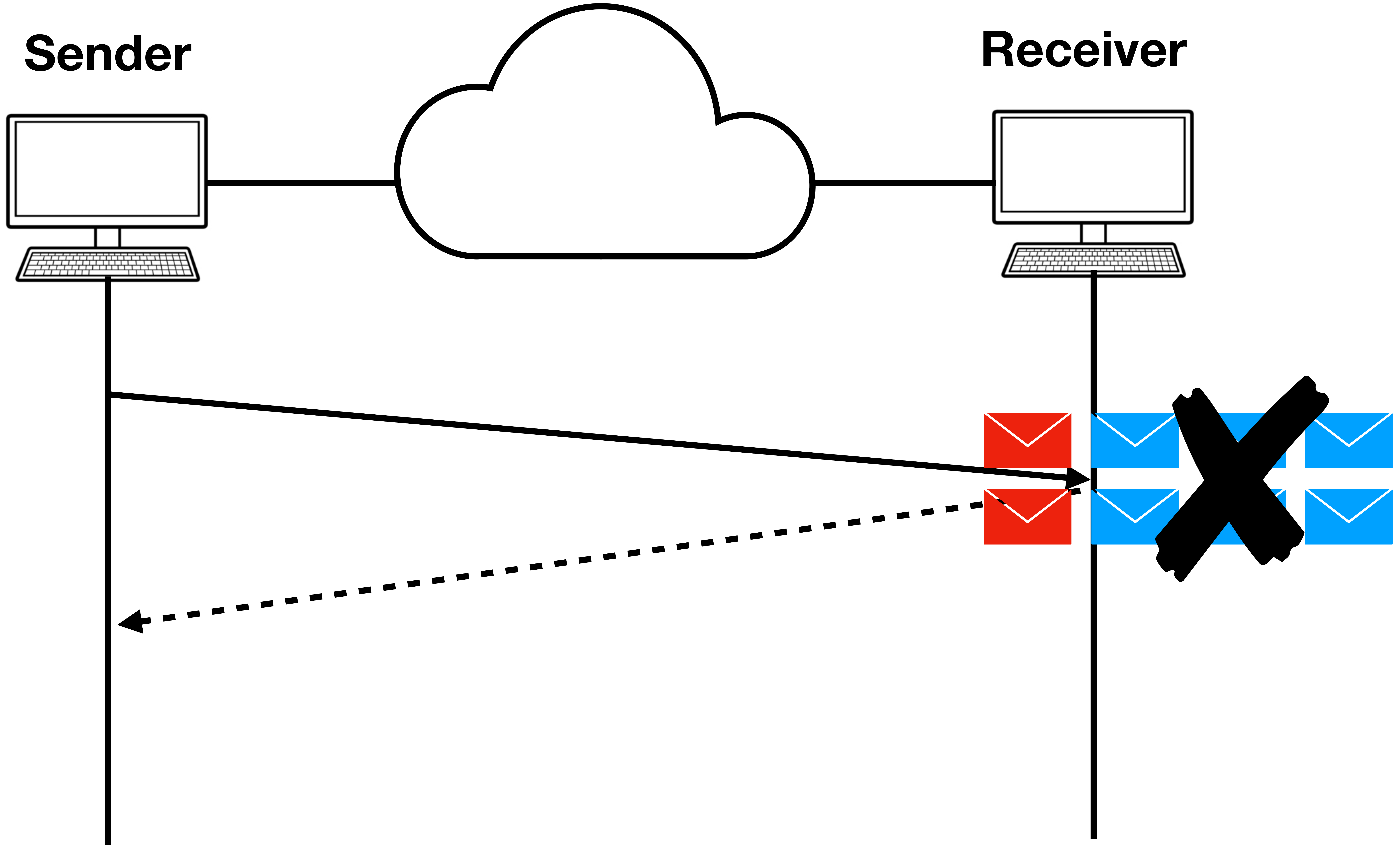


The sender has to retransmit the frame sometime later — timeout.

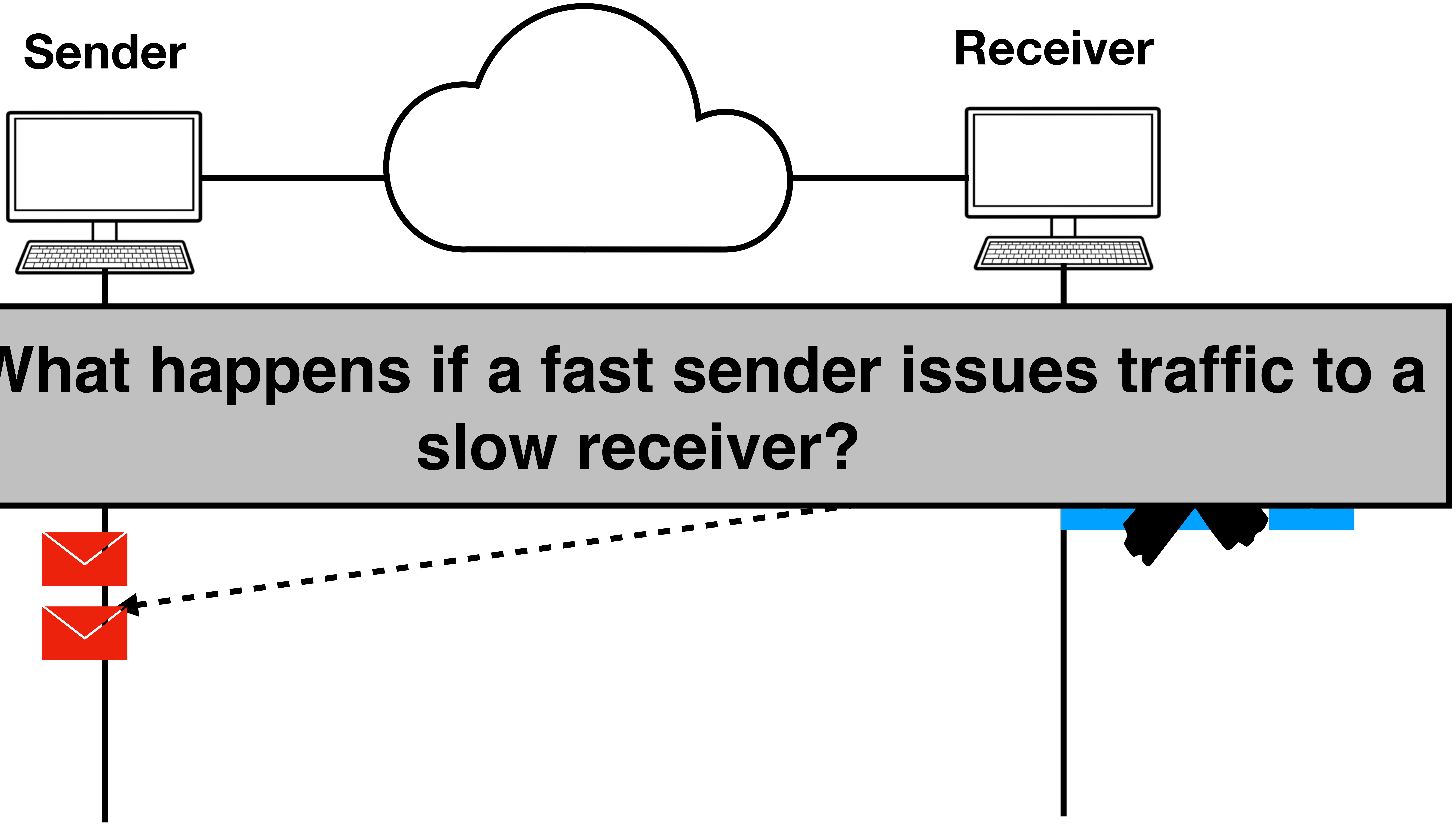
A Reliable Transmission Example



A Reliable Transmission Example

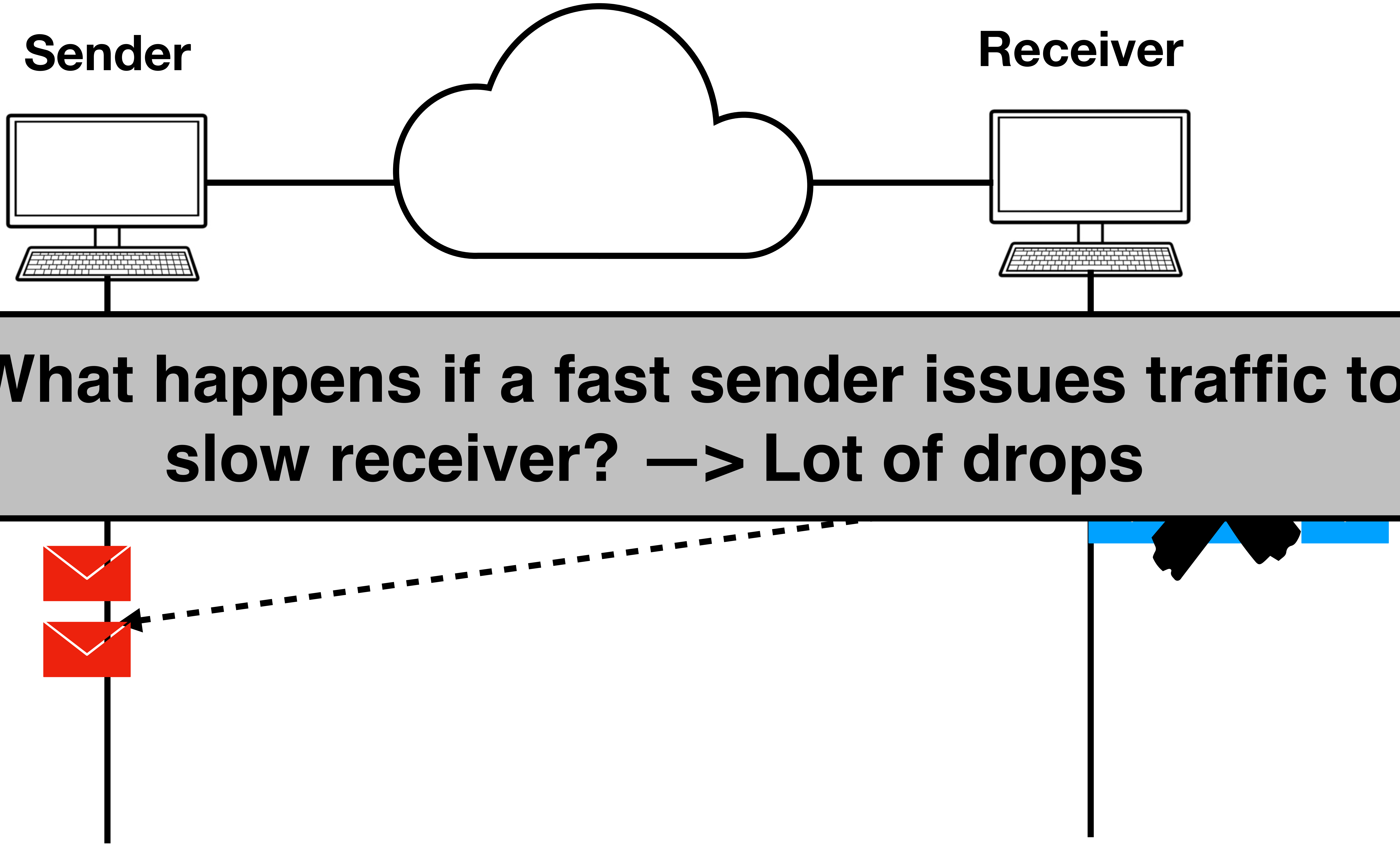


A Reliable Transmission Example



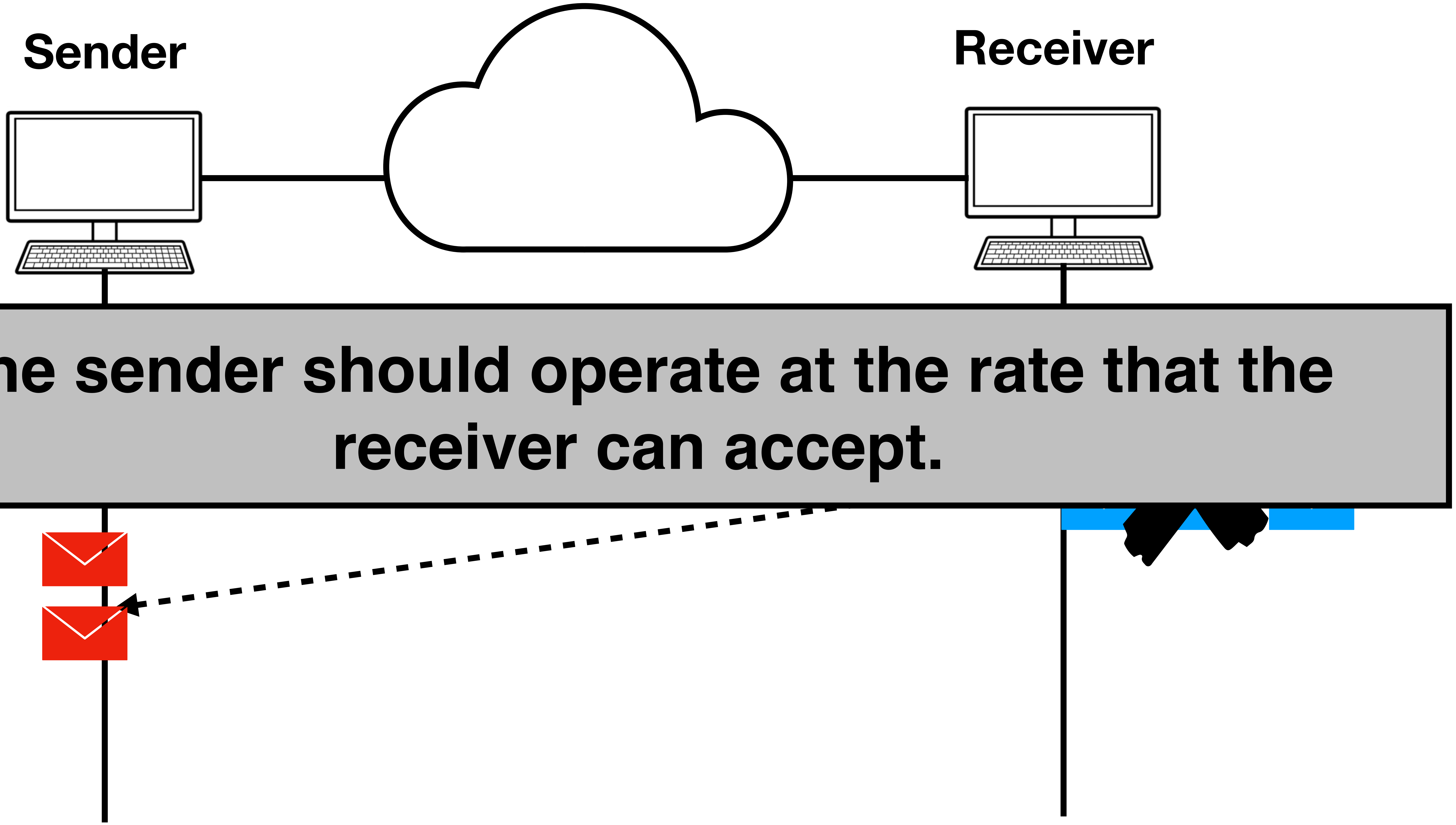
Q4: What happens if a fast sender issues traffic to a slow receiver?

A Reliable Transmission Example



Q4: What happens if a fast sender issues traffic to a slow receiver? —> Lot of drops

A Reliable Transmission Example



The sender should operate at the rate that the receiver can accept.

Reliable Transmission Consideration

- #1: Acknowledgement
  - Notify the sender of the receipt of a frame from the receiver
- #2: Unique Frame ID
  - Differentiate concurrent frame transmission
- #3: Timeout
  - Emulate errors in a pragmatic way
  - False negatives cannot be avoided, e.g., a slow receiver
- #4: Pacing
  - Reduce the number of unnecessary retransmissions

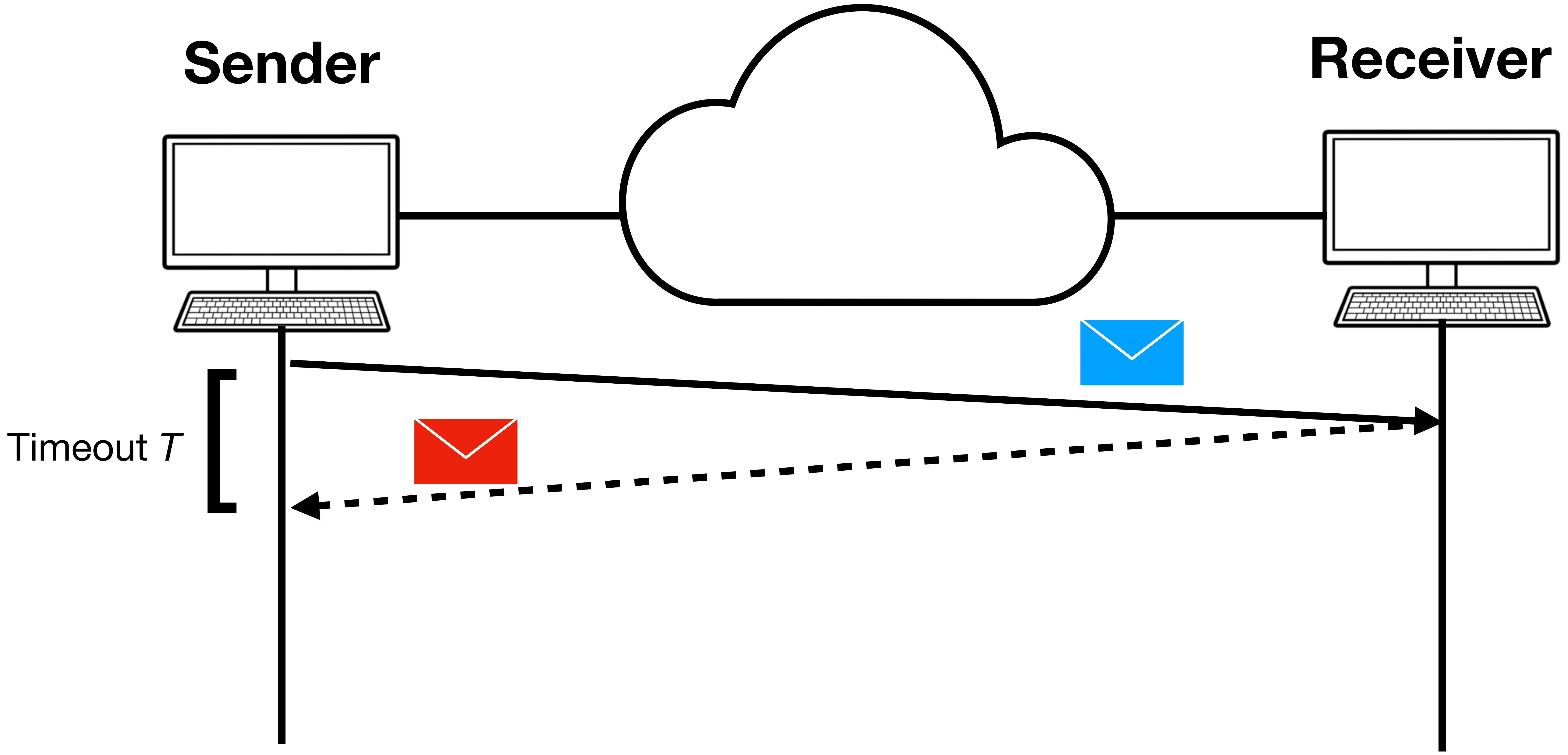
How do we design a reliable transmission mechanism?

Technique #1: Stop-and-Wait

- Key idea: 1 outstanding frame + ACK + Timeout
  - Send the next frame only if the last one is successfully delivered
  - When the timeout is signaled, the sender issues another frame

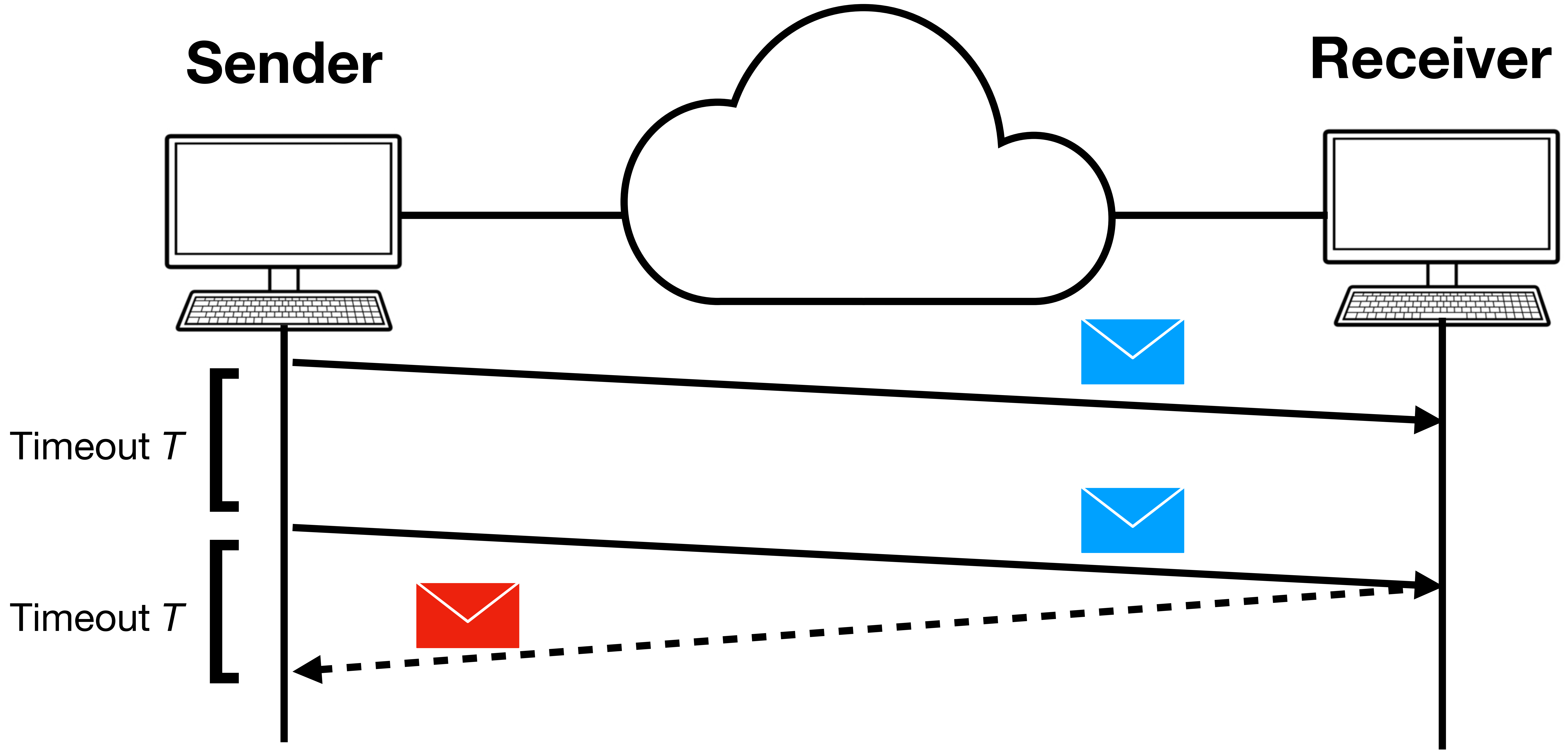
Technique #1: Stop-and-Wait

- Key idea: 1 outstanding frame + ACK + Timeout
  - Send the next frame only if the last one is successfully delivered
  - When the timeout is signaled, the sender issues another frame



Technique #1: Stop-and-Wait

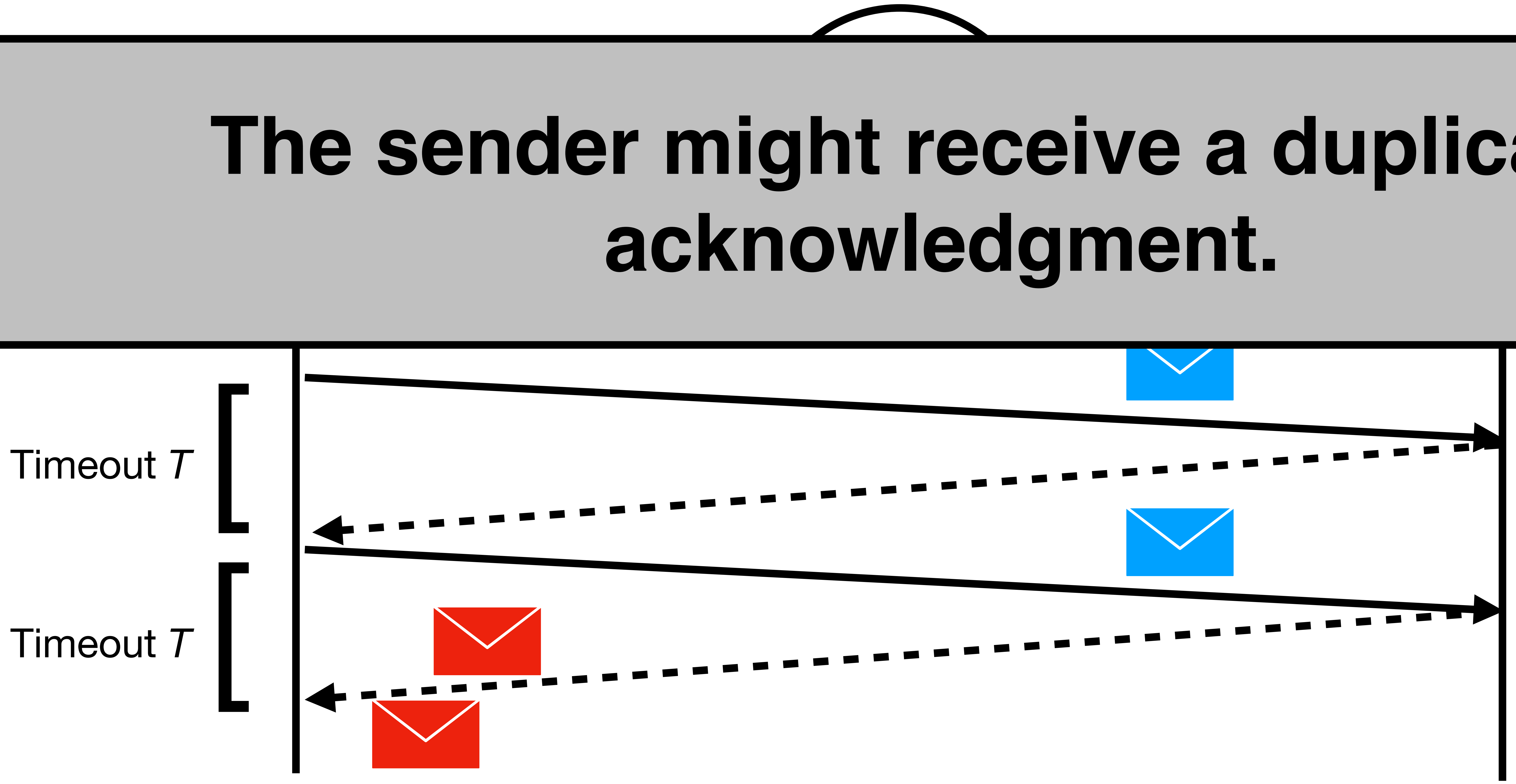
- Key idea: 1 outstanding frame + ACK + Timeout
  - Send the next frame only if the last one is successfully delivered
  - When the timeout is signaled, the sender issues another frame



Technique #1: Stop-and-Wait

- Key idea: 1 outstanding frame + ACK + Timeout
  - Send the next frame only if the last one is successfully delivered
  - When the timeout is signaled, the sender issues another frame

The sender might receive a duplicated acknowledgment.



Technique #1: Stop-and-Wait

- Key idea: 1 outstanding frame + ACK + Timeout
  - Send the next frame only if the last one is successfully delivered
  - When the timeout is signaled, the sender issues another frame

- Discussion:
  - Simple to implement
  - Low performance — cannot fully utilize the bandwidth

Technique #2: Concurrent Logical Channels

- Key idea: partition a physical link into multiple logical channels
  - Each channel works independently
  - Each channel can operate using the stop-and-wait or sliding window mechanism (discussed next)
  - Concurrent outstanding frames per link = Channel # X Concurrent outstanding frames per channel

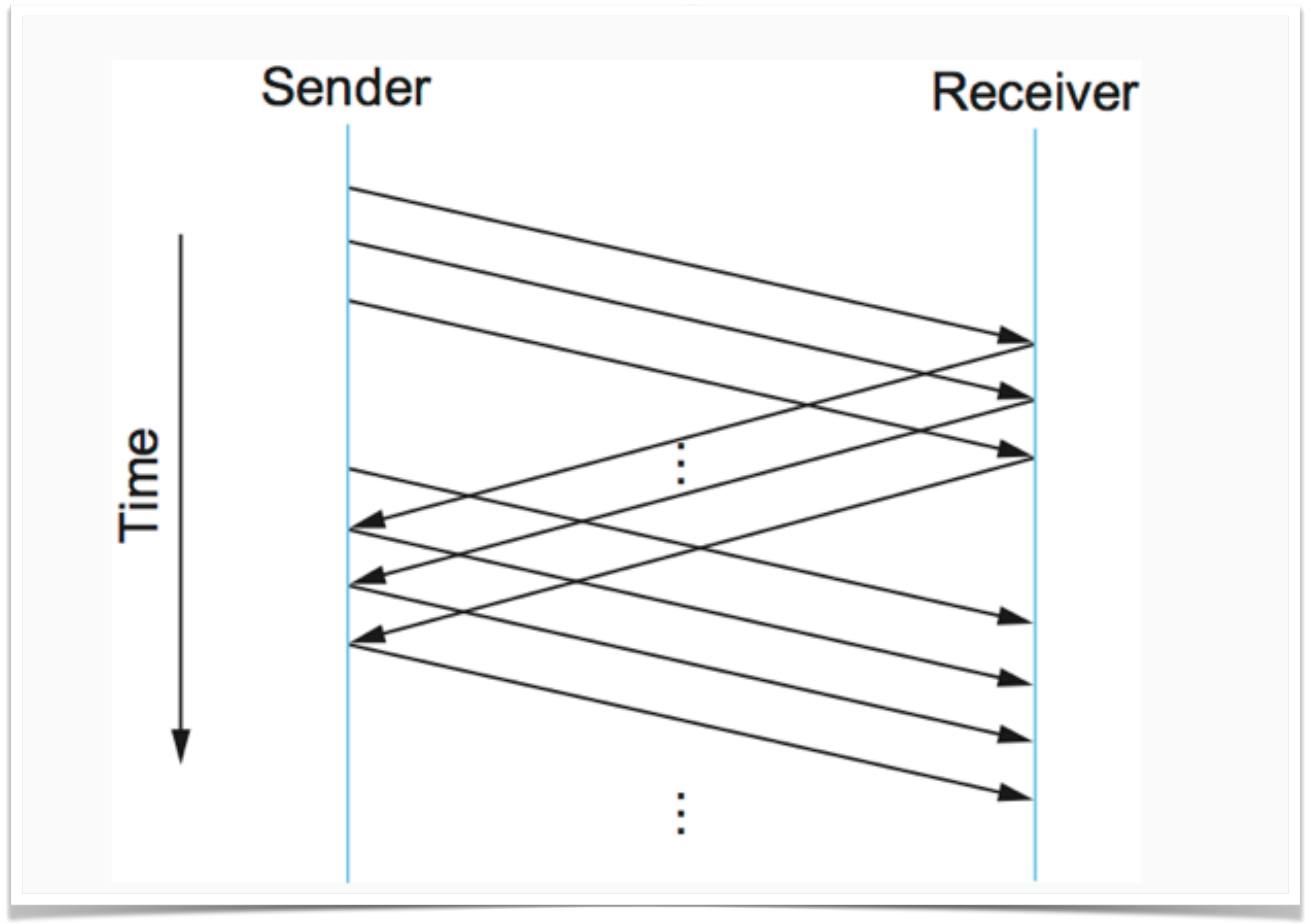
Technique #2: Concurrent Logical Channels

- Key idea: partition a physical link into multiple logical channels
  - Each channel works independently
  - Each channel can operate using the stop-and-wait or sliding window mechanism (discussed next)
  - Concurrent outstanding frames per link = Channel # X Concurrent outstanding frames per channel

- In practice:
  - PCIe: x1, x2, x4, x8, x16
  - NVLink: x16, x32, x64, x96, x128

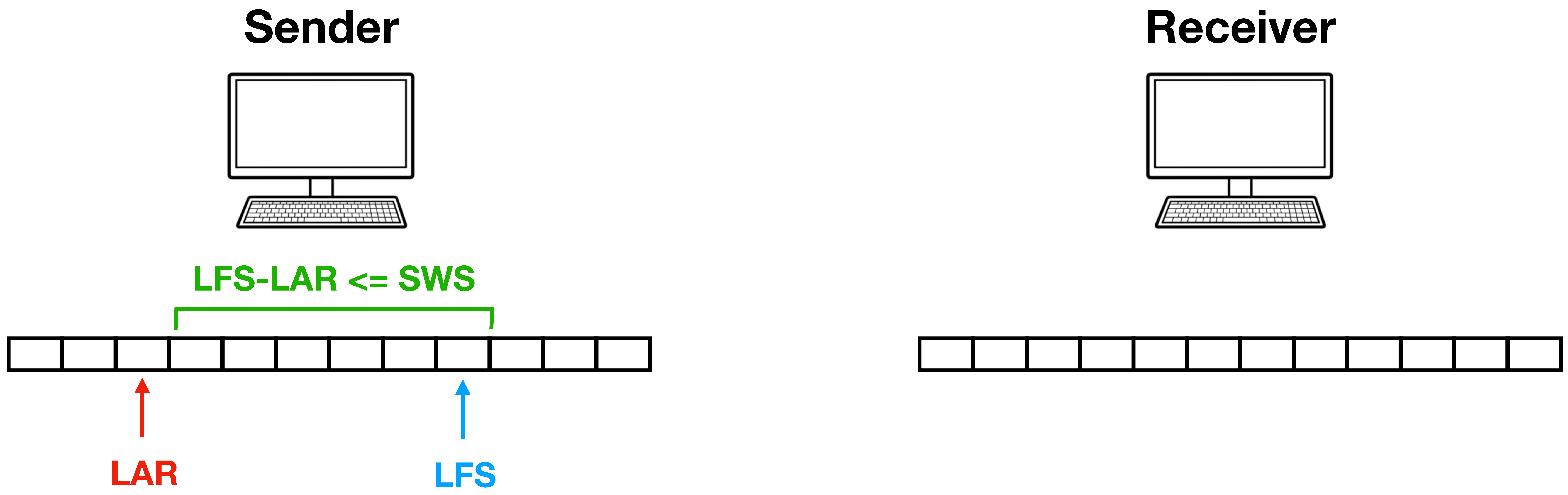
Technique #3: Sliding Window

- Key idea: keep the communication channel full with N consecutive frames
  - Driven by the bandwidth-delay product (BDP)
  - Seems simple, but…



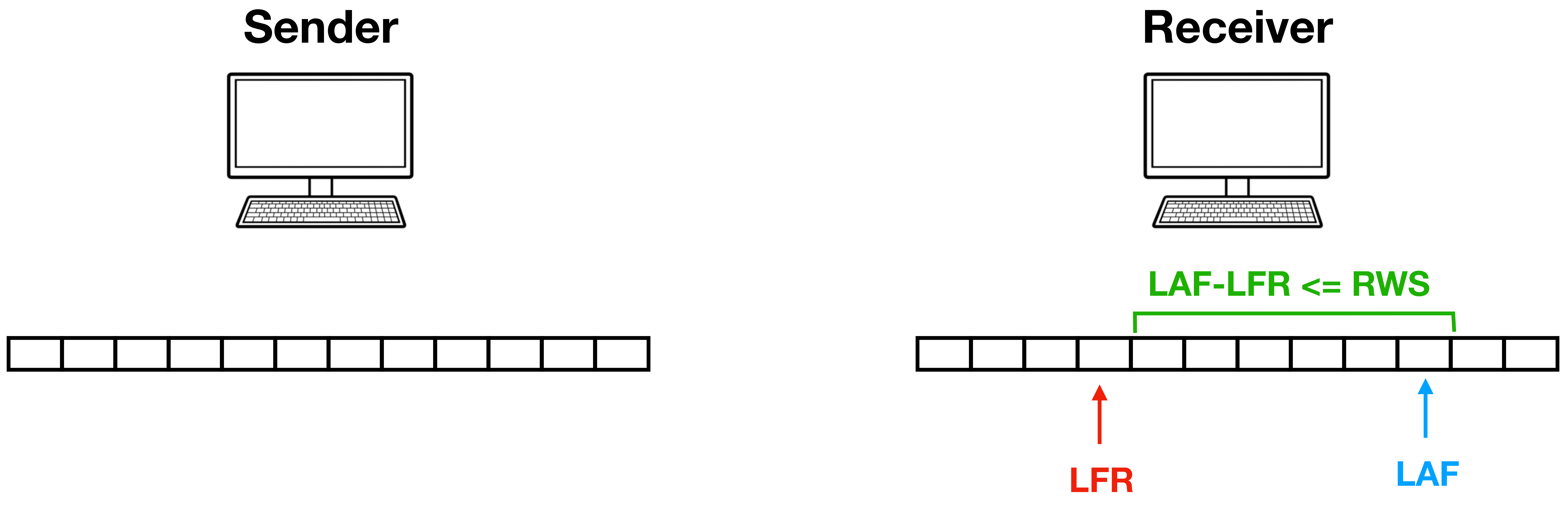
States Maintained by the Sender

- SWS: Send Window Size
- LAR: the sequence number of the last acknowledgment received
- LFS: the sequence number of the last frame sent



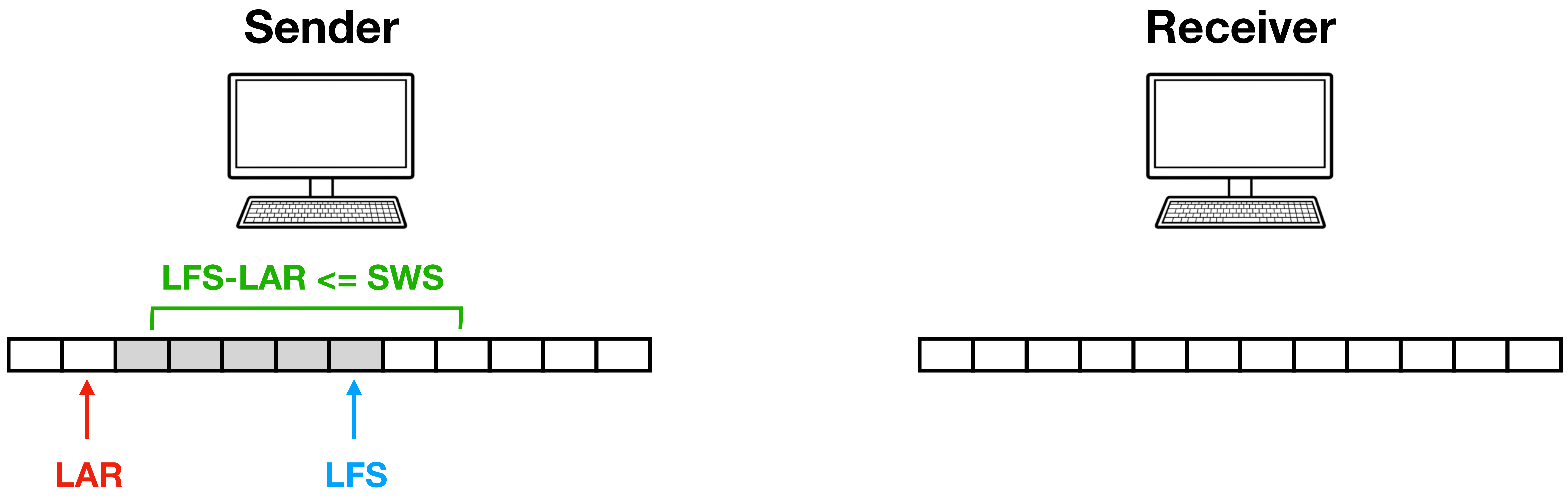
States Maintained by the Receiver

- RWS: Receive Window Size
- LAF: the sequence number of the largest acceptable frame
- LFR: the sequence number of the last frame received



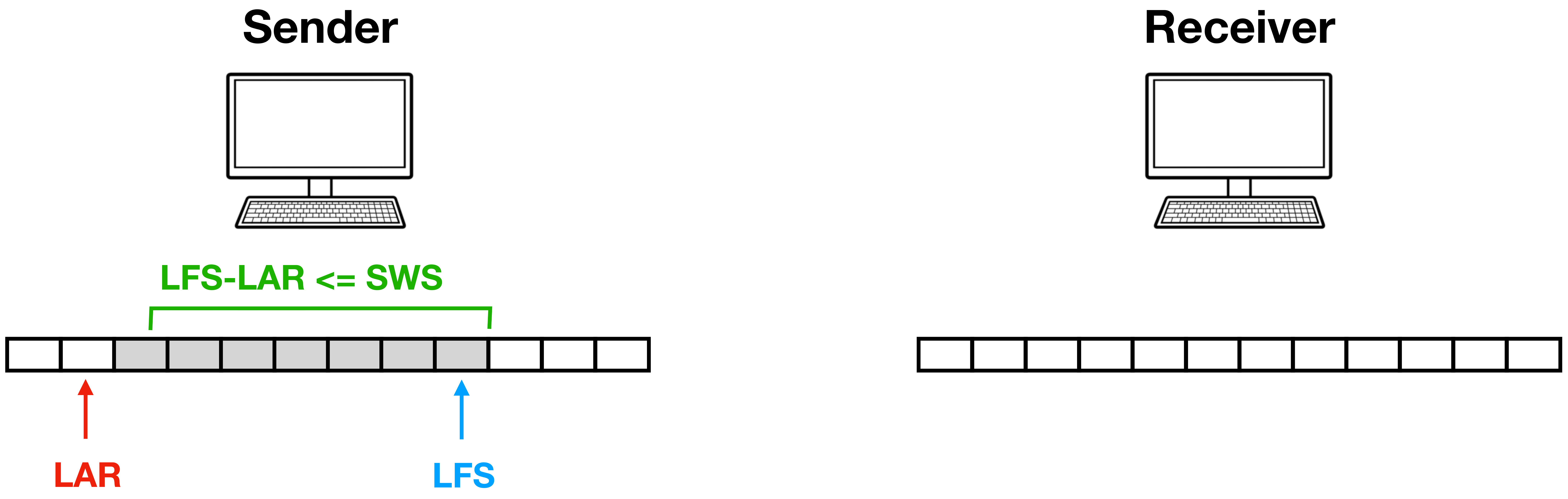
Sender Logic — Sending a Frame

- Logic #1: send unacknowledged frames within the SWS
  - Keep the invariant: LFS - LAR <= SWS



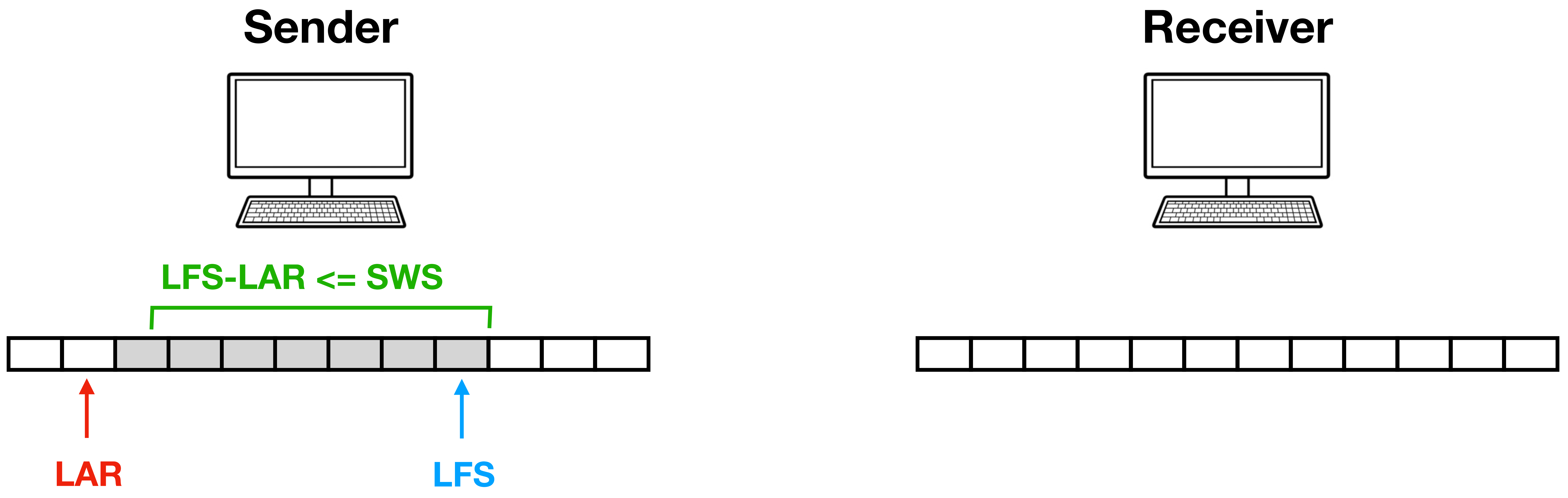
Sender Logic — Sending a Frame

- Logic #1: send unacknowledged frames within the SWS
  - Keep the invariant: LFS - LAR <= SWS



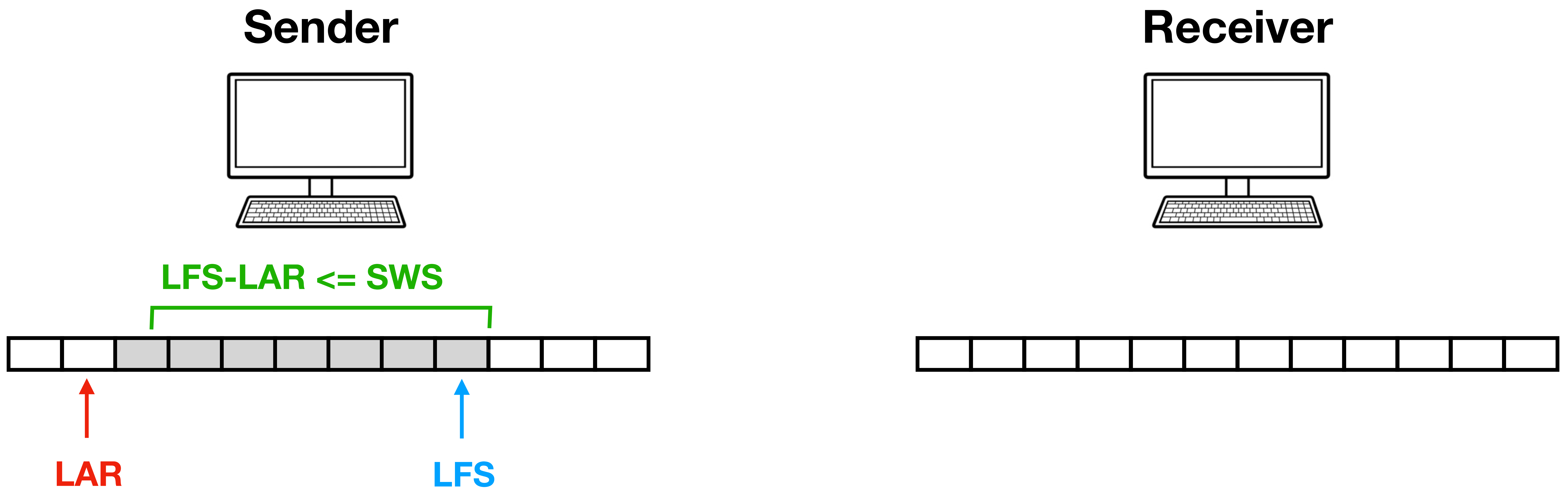
Sender Logic — Receiving an Acknowledgment

- Logic #2: receive acknowledgments from the receiver
  - Only update LAR if the SeqNum of the acknowledgment is LAR
  - Out-of-order acknowledgment is possible, which can be further optimized



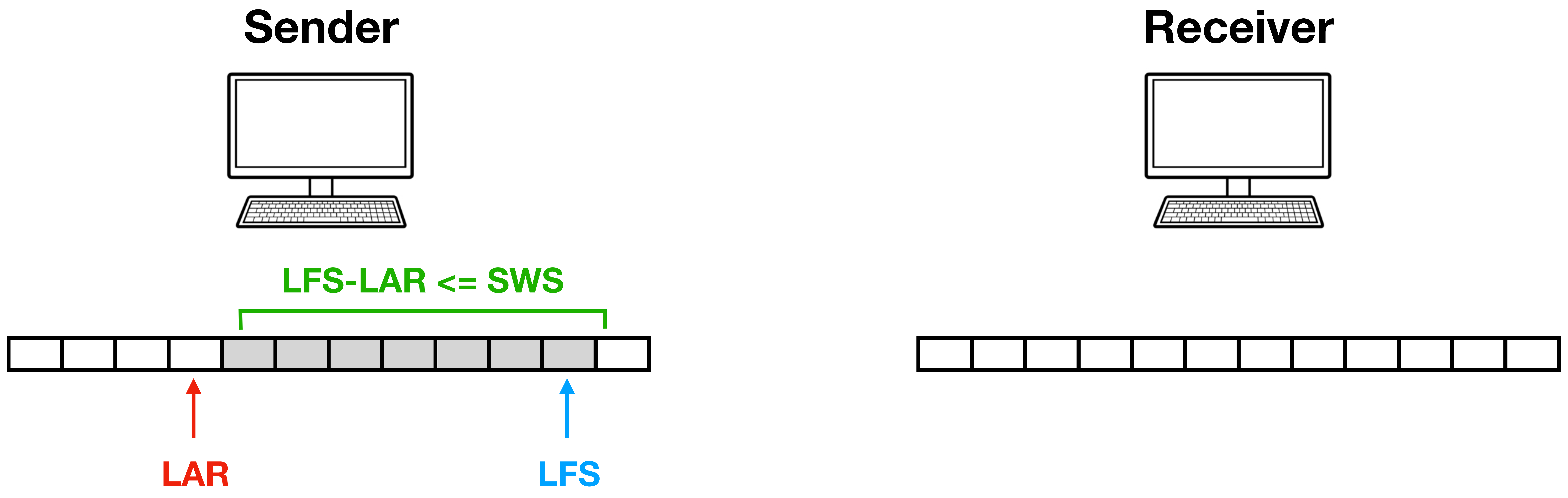
Sender Logic — Receiving an Acknowledgment

- Logic #2: If LAR is updated
    - Free the frame buffer
    - Send more frames within the SWS



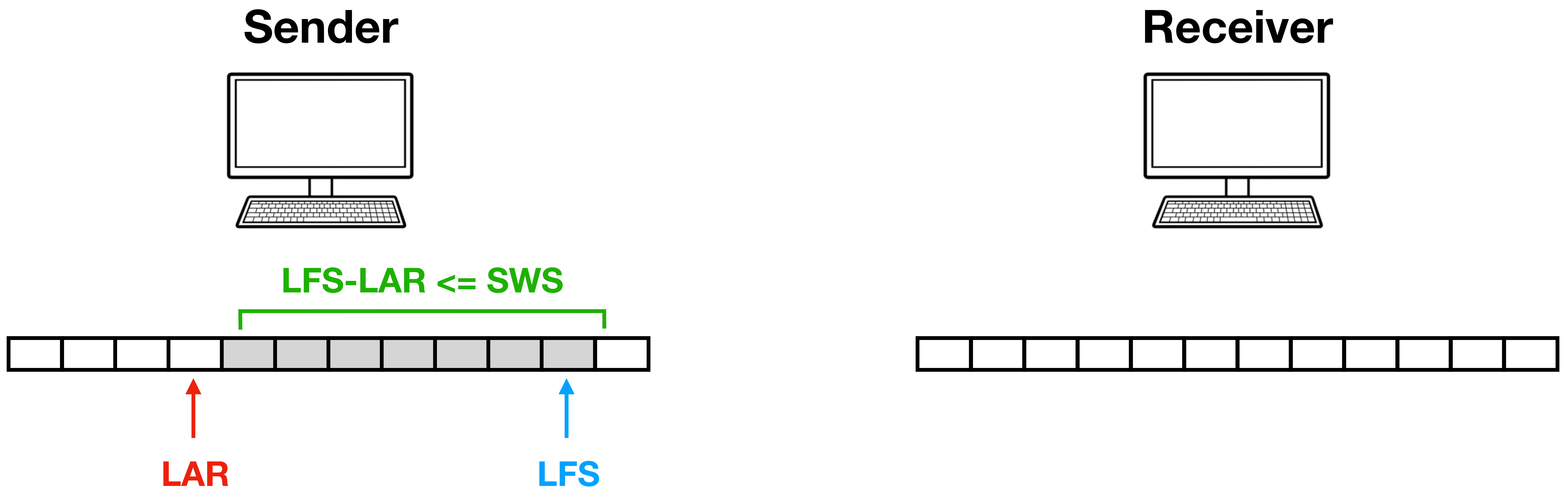
Sender Logic — Receiving an Acknowledgment

- Logic #2: If LAR is updated
  - Free the frame buffer
  - Send more frames within the SWS



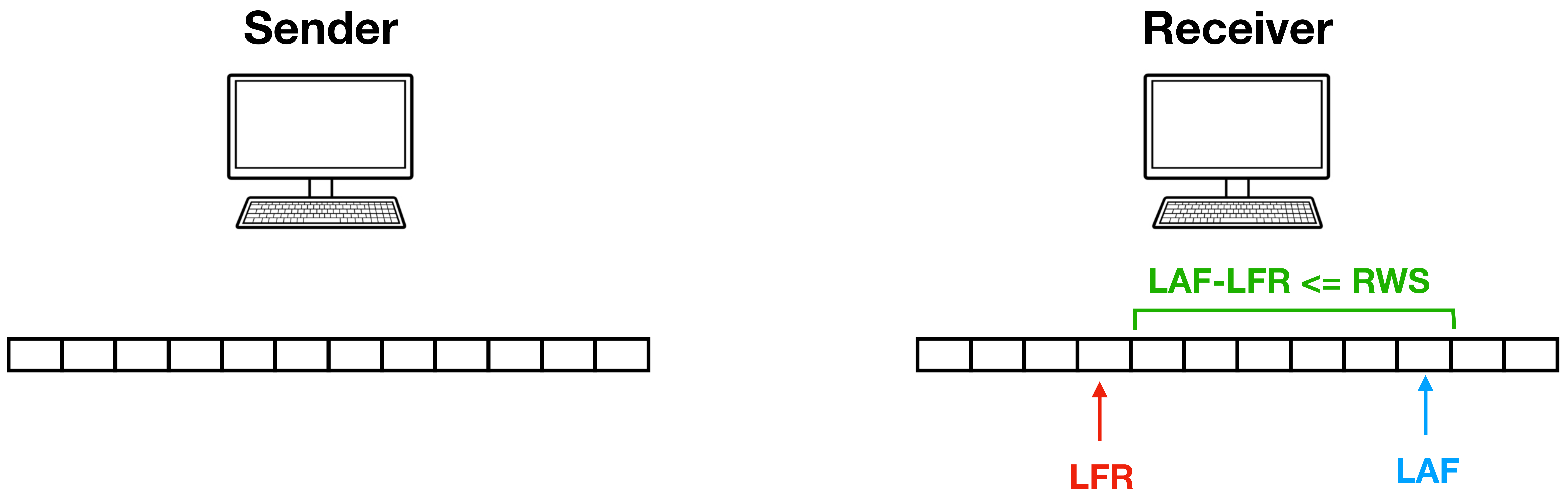
Sender Logic — Timeout

- Logic #3: retransmit frames when a timeout signal is generated
  - Each frame should maintain its own timeout variable
  - LAR and LFS cannot be changed!



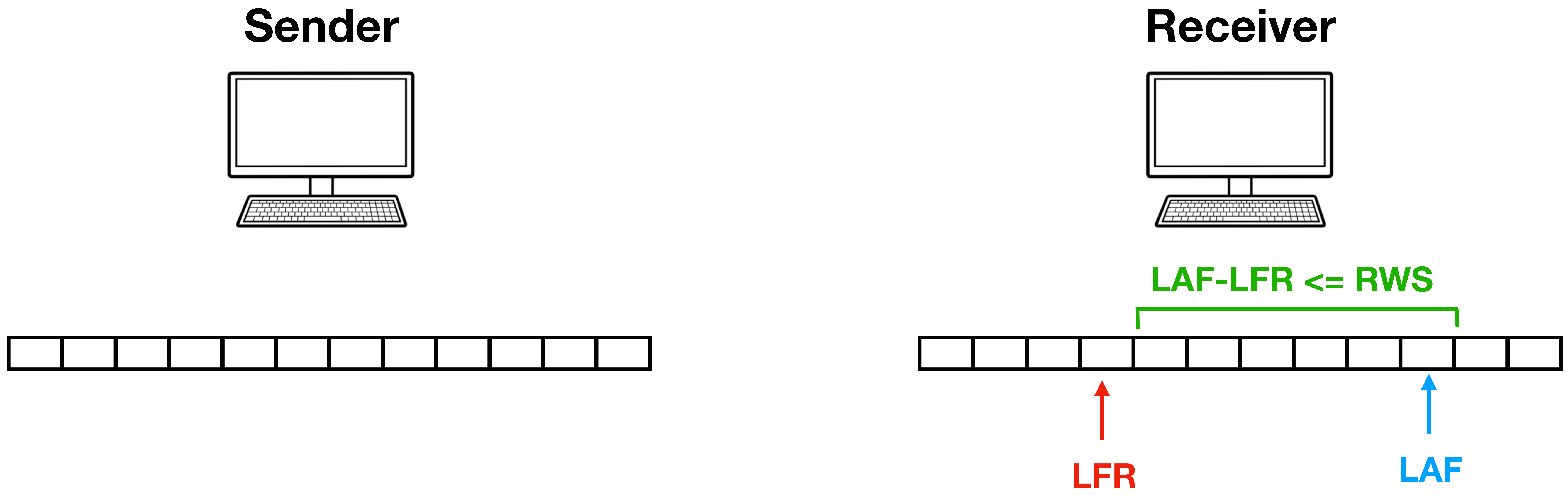
Receiver Logic — Receiving a Frame

- Logic #1: examine the sequence number (SeqNum) of the frame



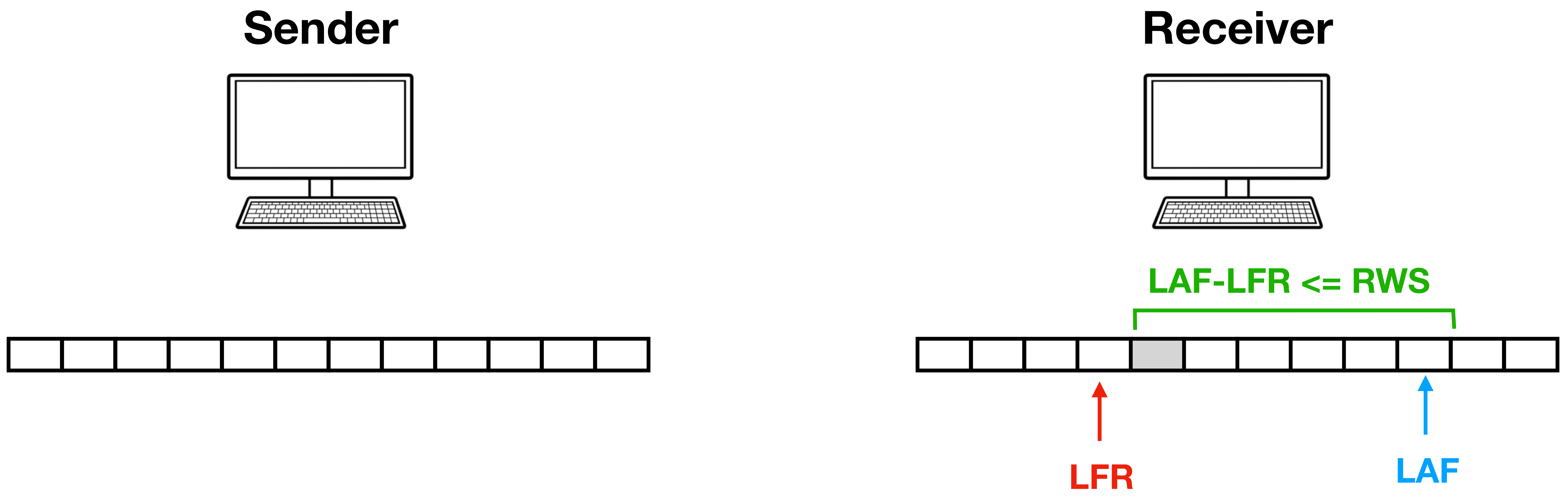
Receiver Logic — Receiving a Frame

- Logic #1: examine the sequence number (SeqNum) of the frame
  - If seqNum <= LFR, the frame has been acked and sent the ack again



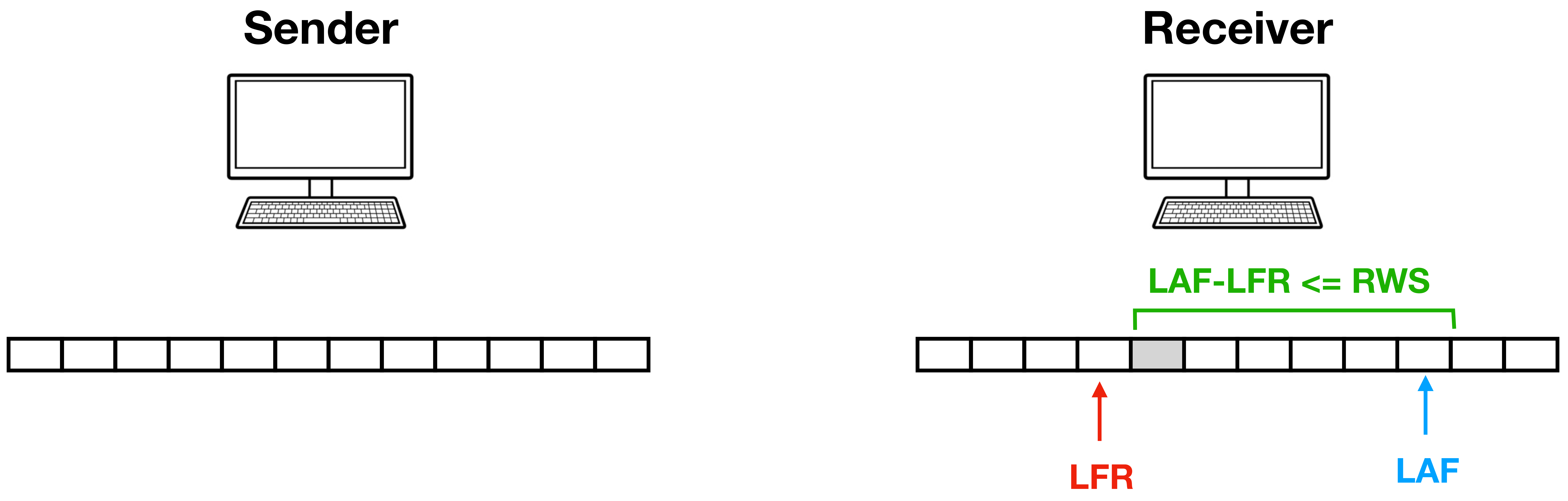
Receiver Logic — Receiving a Frame

- Logic #1: examine the sequence number (SeqNum) of the frame
    - If seqNum <= LFR, the frame has been acked and sent the ack again
    - if seqNum == LFR+1, LFR = LFR+1, LAF=LFR+RWS, and send the ack



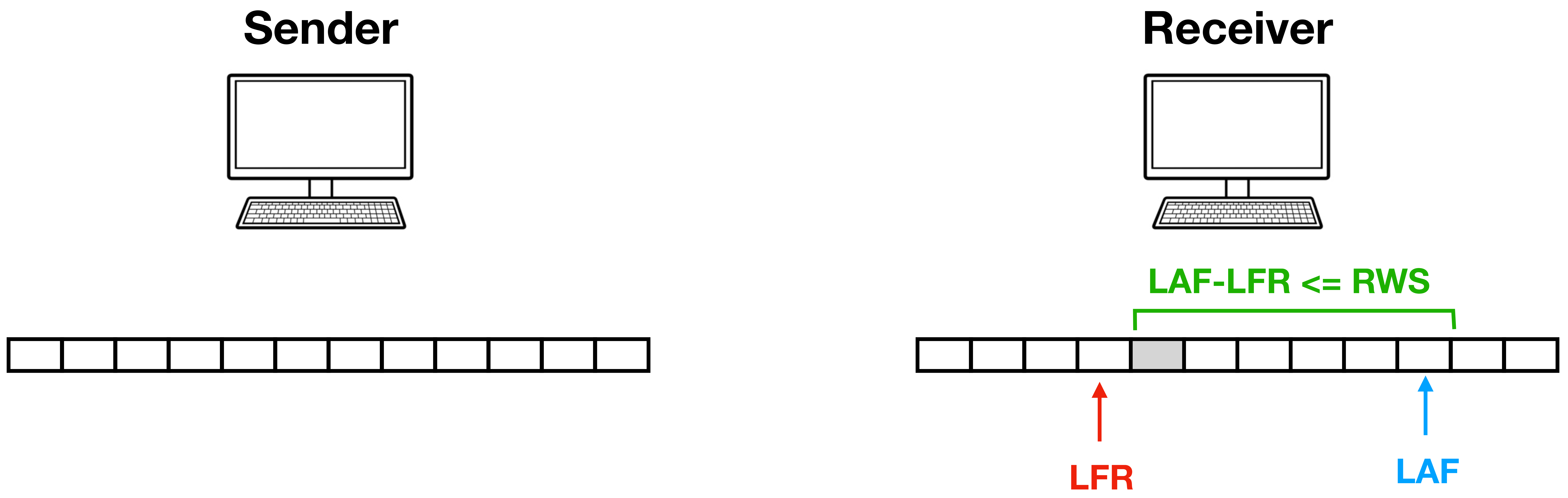
Receiver Logic — Receiving a Frame

- Logic #1: examine the sequence number (SeqNum) of the frame
  - If seqNum <= LFR, the frame has been acked and sent the ack again
  - if seqNum == LFR+1, LFR = LFR+1, LAF=LFR+RWS, and send the ack
  - If seqNum - LFR <= RWS, send the ack



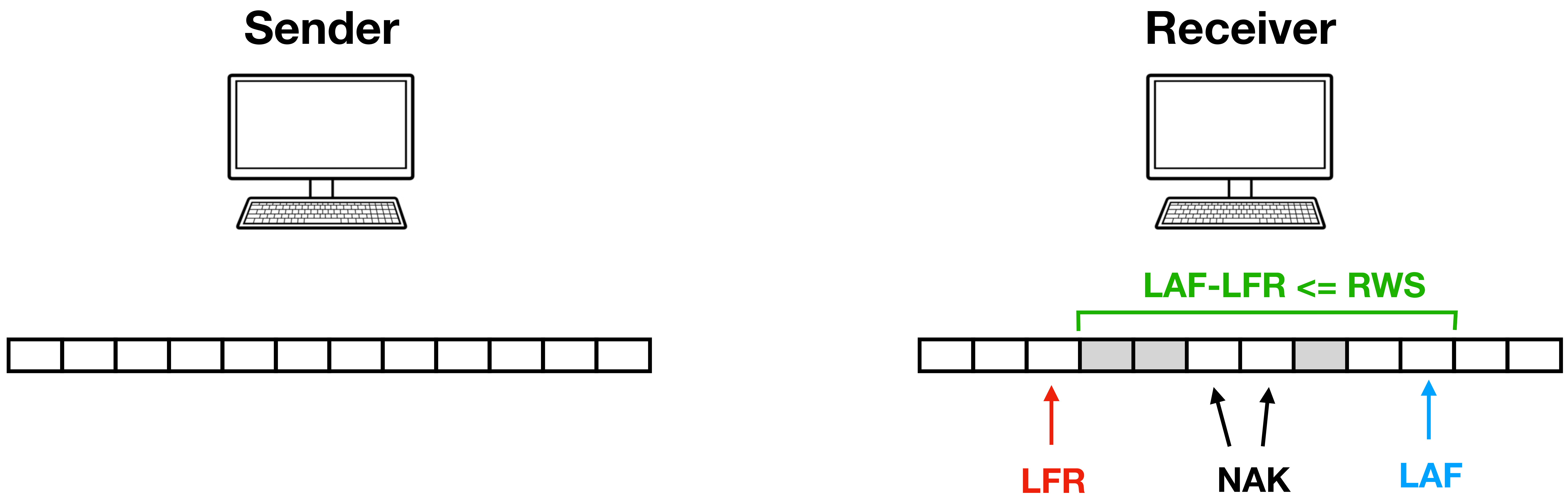
Receiver Logic — Receiving a Frame

- Logic #1: examine the sequence number (SeqNum) of the frame
  - If seqNum <= LFR, the frame has been acked and sent the ack again
  - if seqNum == LFR+1, LFR = LFR+1, LAF=LFR+RWS, and send the ack
  - If seqNum - LFR <= RWS, send the ack
  - If seqNum - LFR > RWS, discard the frame and don't send the ack



Receiver Logic — Sending a Negative Acknowledgment

- Logic #2 (optional): send an NAK to accelerate retransmission
  - If seqNum - LFR <= RWS && seqNum > LFR



Sliding Window Discussion

- The sender and receiver can be implemented via state machines

- Tricky details
  - SWS and RWS are based on BDP and can be adjusted online
  - The frame buffer is a ring
  - SeqNum can be rounded up
  - ……

Link Layer Summary

Physical layer — A reliable (and efficient) bit delivery channel over a link

Link Layer Summary

Link layer	A frame delivery channel between directly connected or switched hosts
Physical layer	A reliable (and efficient) bit delivery channel over a link

Link Layer Summary

Q1: How can we identify a frame from bit streams?
Q2: How can we handle transmission errors?
Q3: How can we achieve scaled transmission using switches?
Q4: How can we coordinate transmission between two hosts?

Link layer	A frame delivery channel between directly connected or switched hosts
Physical layer	A reliable (and efficient) bit delivery channel over a link

Link Layer Summary

Q1: How can we identify a frame from bit streams?
=> Framing
Q2: How can we handle transmission errors?
=> Error handling
Q3: How can we achieve scaled transmission using switches?
=> L2 switching
Q4: How can we coordinate transmission between two hosts?
=> Reliable transmission

Link layer	A frame delivery channel between directly connected or switched hosts
Physical layer	A reliable (and efficient) bit delivery channel over a link

Summary

- Today
  - L2 Reliable Transmission

- Next lecture
  - IP Introduction