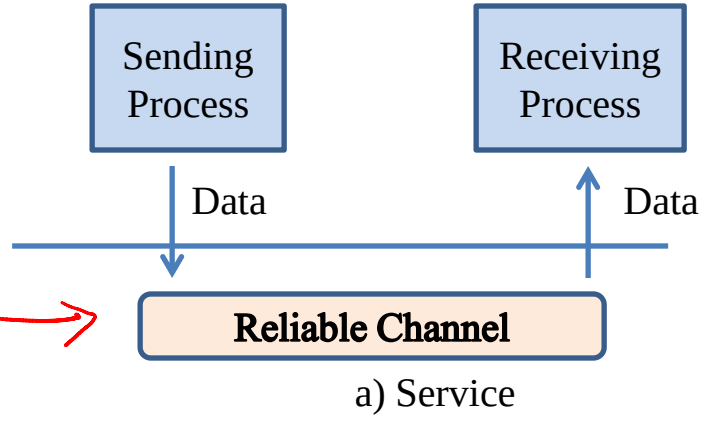
Reliable Data Transfer

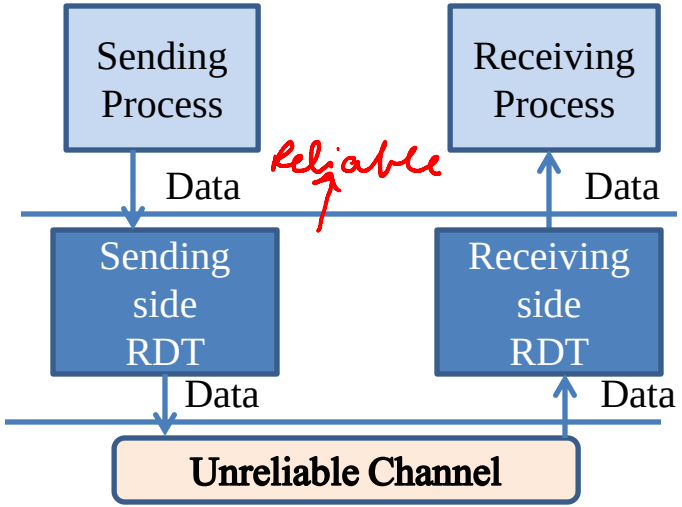
Kameswari Chebrolu

Outline

- Develop a Reliable Data Transfer protocol (RDT)
  - Unreliable channel with bit errors
  - Unreliable channel with bit errors and losses



a) Service



b) Implementation

RDTv1.0: Channel with bit errors

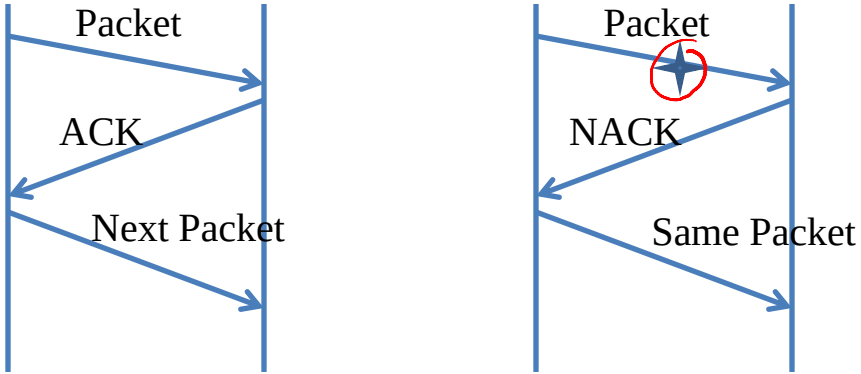
- Telephone Analogy
- Receiver Feedback
  - Positive: aha, ok, hmm → ACK
  - Negative: repeat that, didn't follow, what did you say? → NACK
  - Do we need both?
- Sender retransmits on NACK

Required Functionality:

- Error Detection mechanism
  - Checksum, CRC etc
- ?

Automatic Repeat Request (ARQ)

- Protocols based on Feedback and retransmissions



RDTv1.0

Required Functionality:

- Error Detection mechanism
  - Checksum, CRC etc
- Receiver Feedback
  - ACK + NACK

RDTv1.0 has a fatal flaw!

- What if the ACK/NACK got corrupted?
  - What should sender do then?
- Send next packet? If prev. pkt is lost, RDT not providing reliability
- Send previous packet? If prev. pkt is not lost, creates duplicates

Required Functionality:

- Error Detection mechanism
  - Checksum, CRC etc
- Receiver Feedback
  - ACK + NACK
  - Data Sequence Numbers

RDTv2.0

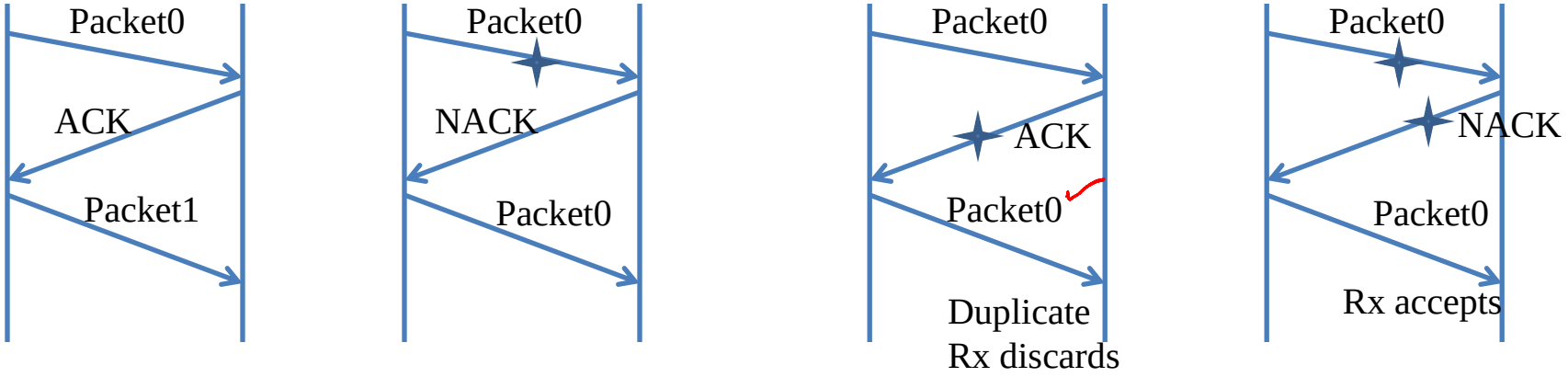
- Receiver gives feedback (ACK, NACK)
- Sender retransmits ‘sequenced’ packet on NACK, garbled ACK/NACK
- Receiver discards duplicates if any based on sequence number

Required Functionality:

- Error Detection mechanism
  - Checksum, CRC etc
- Receiver Feedback
  - ACK + NACK
  - Data Sequence Numbers

RDTv2.0

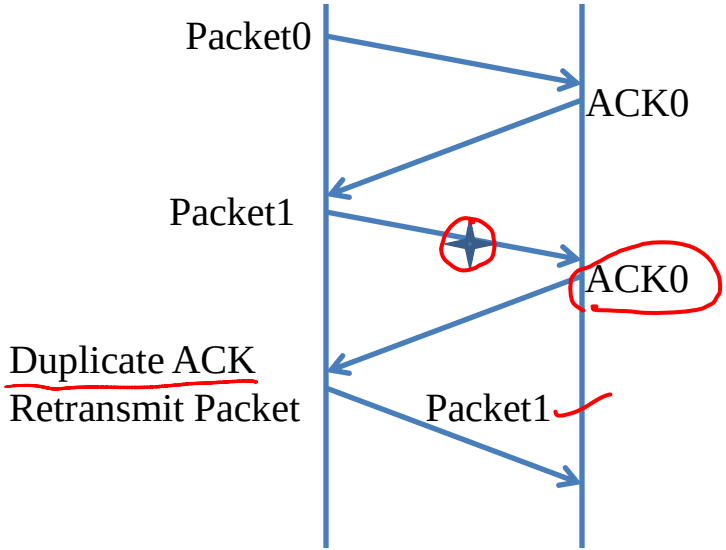
- What is the sequence number space?
  - Just two seq #s “0 , 1” will suffice



RDTv2.1

- Optimization: NACK free operation
  - Convey same information as NACK but through ACK. How?
- Instead of NACK, receiver sends ACK of last correctly received packet
  - Receiver must explicitly include seq # of packet being ACKed
- Duplicate ACK at sender results in same action as NACK: retransmit current packet

NACK Free Protocol



RDTv2.1

Required Functionality:

- Error Detection mechanism
  - Checksum, CRC etc
- Receiver Feedback
  - ACK + ~~NACK~~
- Data Sequence Numbers
- ACK carries sequence number of data packets