16.36 Communication Systems Engineering

Spring 2009

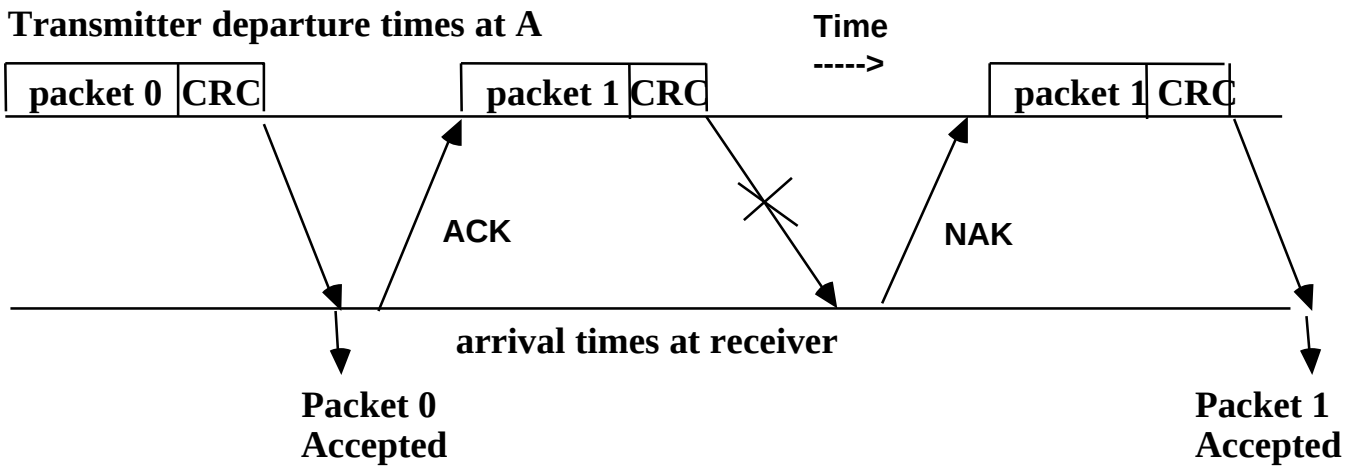
16.36: Communication Systems Engineering

ARQ Protocols: Stop & Wait

Automatic Repeat ReQuest (ARQ)

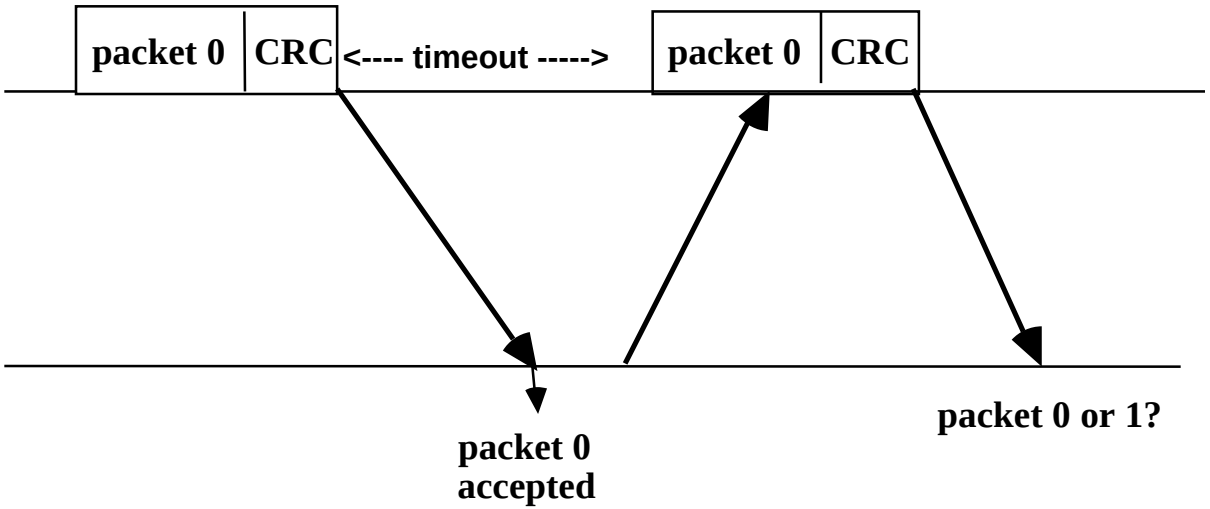
- When the receiver detects errors in a packet, how does it let the transmitter know to re-send the corresponding packet?
- Systems which automatically request the retransmission of missing packets or packets with errors are called ARQ systems.
- Three common schemes
  - Stop & Wait
  - Go Back N
  - Selective Repeat
- Byzantine Army Problem
  - Byzantine failure: the failure to correctly execute a step in an algorithm

Pure Stop and Wait Protocol



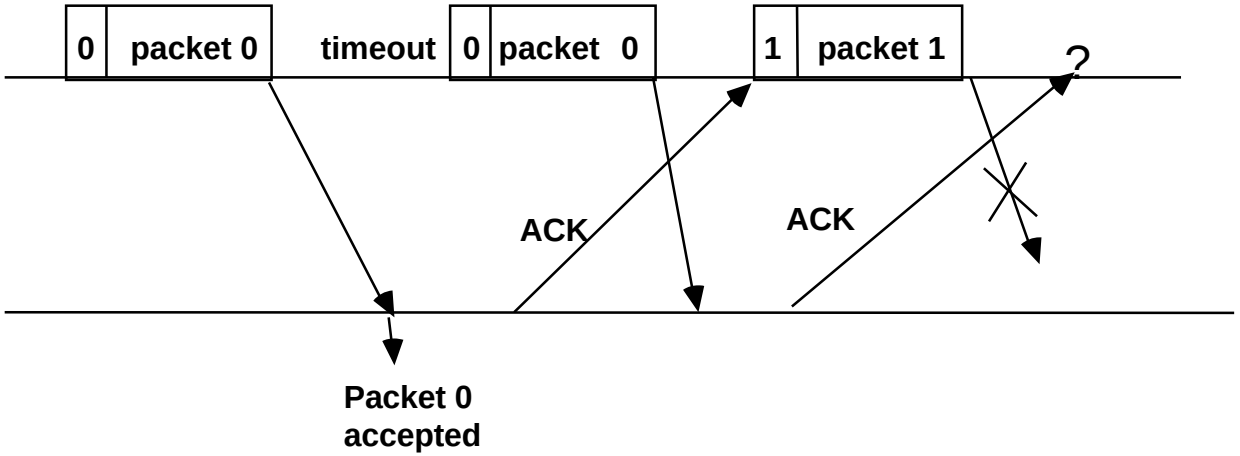
- **Problem: Lost Packets**
  - **Sender will wait forever for an acknowledgement**
- **Packet may be lost due to framing errors**
- **Solution: Use time-out (TO)**
  - **Sender retransmits the packet after a timeout**

The Use Of Timeouts For Lost Packets Requires Sequence Numbers



- **Problem: Unless packets are numbered the receiver cannot tell which packet it received**
- **Solution: Use packet numbers (sequence numbers)**

Request Numbers Are Required On ACKs To Distinguish Packet ACKed



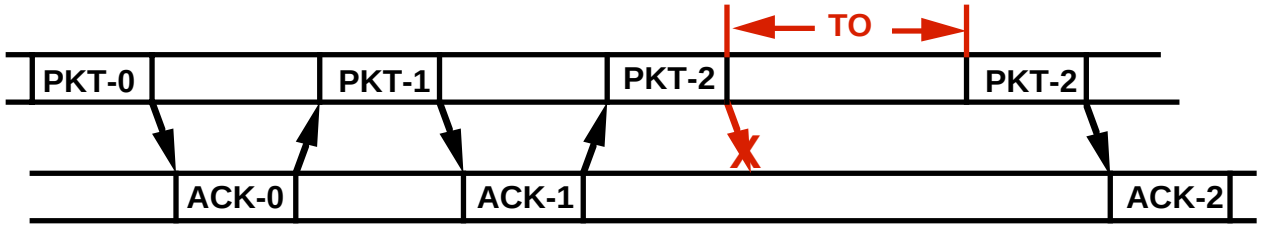
- **REQUEST NUMBERS:**
  - **Instead of sending "ack" or "nak", the receiver sends the number of the packet currently awaited.**
  - **Sequence numbers and request numbers can be sent modulo 2.**

    This works correctly assuming that

    1) Frames travel in order (FCFS) on links
    2) The CRC never fails to detect errors
    3) The system is correctly initialized.

The stop and wait protocol

- Original ARQ protocol
- Sender transmits one packet at a time and waits for an ACK
  - Receiver ACK's packets
  - Sender retransmits packet after a timeout



- Packet numbering
  - Sender numbers packets with sequence numbers (SN)
  - Receiver uses request numbers (RN) to ACK packets

    RN = j is the same as an ACK for packet j-1
- Note:
  - Transmitter idle while waiting for ACK
  - Efficiency limited by round trip delay time
  - Requires no storage of packets

Stop and Wait Protocol
Algorithm at sender (node A)

(with initial condition SN=0)

1) **Accept packet from higher layer when available; assign number SN to it**

2) **Transmit packet SN**

3) **Wait for an error free packet from B**

   i. if received and it contains RN>SN in the request # field, set SN to RN and go to 1

   ii. if not received within given time (TO), go to 2

Stop and Wait
Algorithm at receiver (node B)

(with initial condition RN=0)

1) **Whenever an error-free frame is received from A with a sequence # equal to RN, release received packet to higher layer and increment RN.**

2) **At arbitrary times, but within bounded delay after receiving any error free frame from A, transmit a frame to A containing RN in the request # field.**

Correctness of Stop and Wait

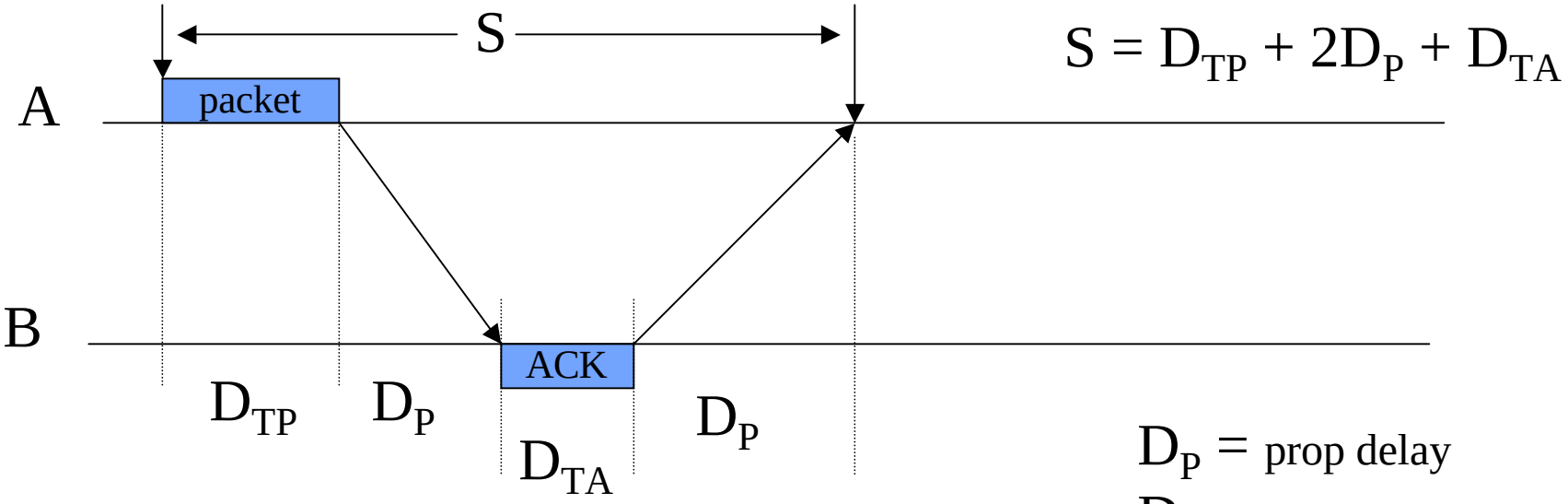
- **SAFETY: show that no packet is ever released out of order or more than once**
  - **This is immediately obvious from the algorithm. We start with packet '0' and receiver does not increment its RN until '0' is received; upon which it can only accept packet '1', etc.**
- **LIVENESS: show that every packet is eventually released**
  - **The sender keeps sending a packet until it gets an ack, so eventually every packet is correctly received**
- **Packet numbering: packets are numbered modulo 2 (0 or 1)**
  - **Start with packet 0, then 1, then 0, then 1, etc…**
  - **This works because at any time, the received packet can only be either the packet that the receiver is waiting for (I.e., SN = RN) or the previous packet which has already been received (I.e., SN = RN-1). Mod 2 numbering can be used to distinguish between these packets.**

Efficiency of stop and wait

Let S = total time between the transmission of a packet and reception of its ACK

D_{TP} = transmission time of the packet

Efficiency (no errors) = D_{TP}/S



$S = D_{TP} + 2D_P + D_{TA}$

D_P = prop delay

D_{TA} = ACK trans. Time

DTP = packet trans. time

$E = D_{TP}/(D_{TP} + 2D_P + D_{TA})$

Stop and wait in the presence of errors

Let P = the probability of an error in the transmission of a packet or in its acknowledgment

$S = D_{TP} + 2D_P + D_{TA}$

TO = the timeout interval

X = the amount of time that it takes to transmit a packet and receive its ACK. This time accounts for retransmissions due to errors

$E[X] = S + TO*P/(1-P)$, Efficiency = $D_{TP}/E[X]$

Where,

TO = D_{TP} in a full duplex system

TO = S in a half duplex system