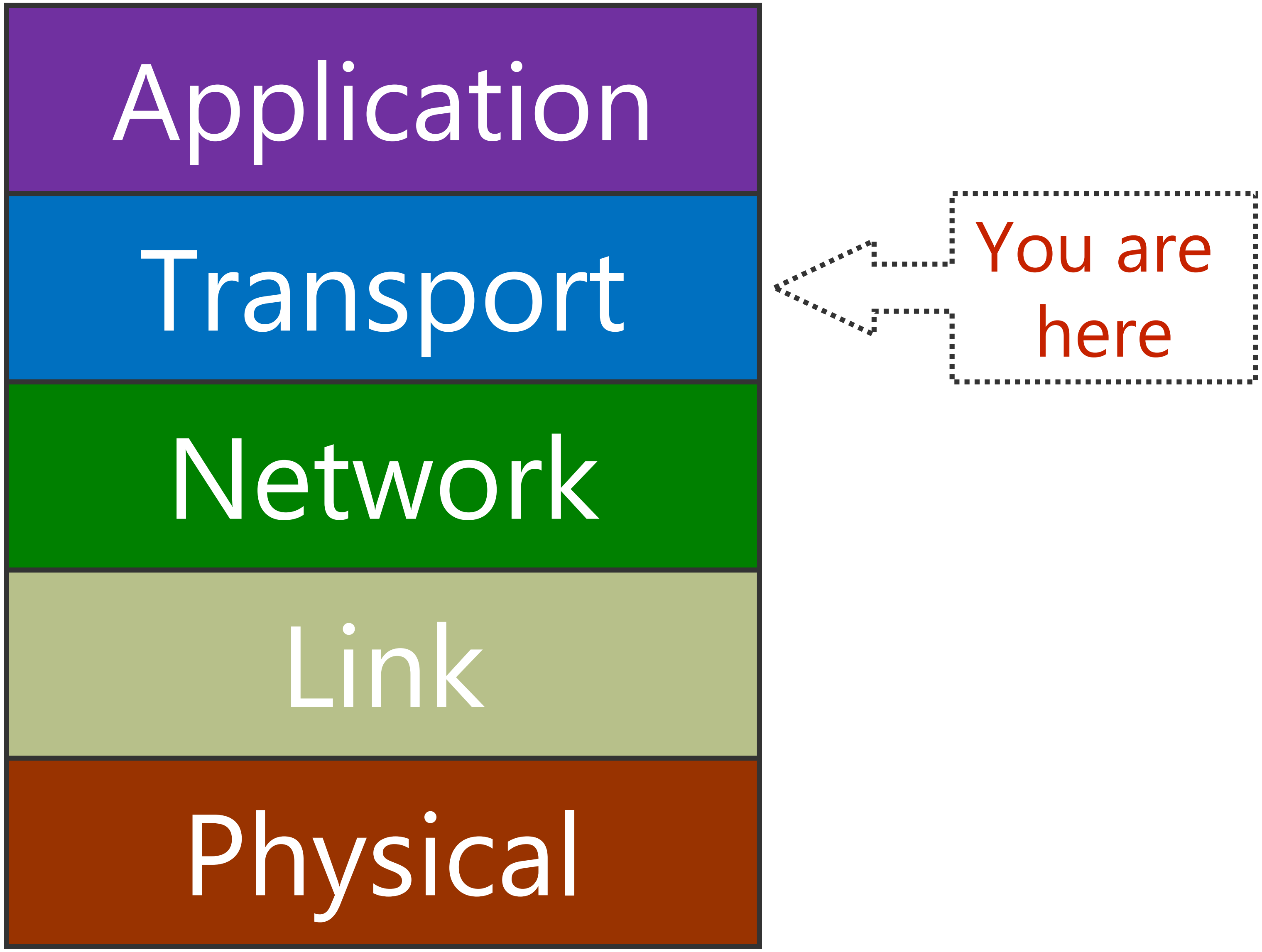
CS2105

An Awesome Introduction to Computer Networks

Lecture 4 discussion

NUS National University of Singapore | Department of Computer Science School of Computing


Application
Transport
Network
Link
Physical
You are here

Transport Layer Services

- Deliver messages between application processes running on different hosts
  - Two popular protocols: TCP and UDP

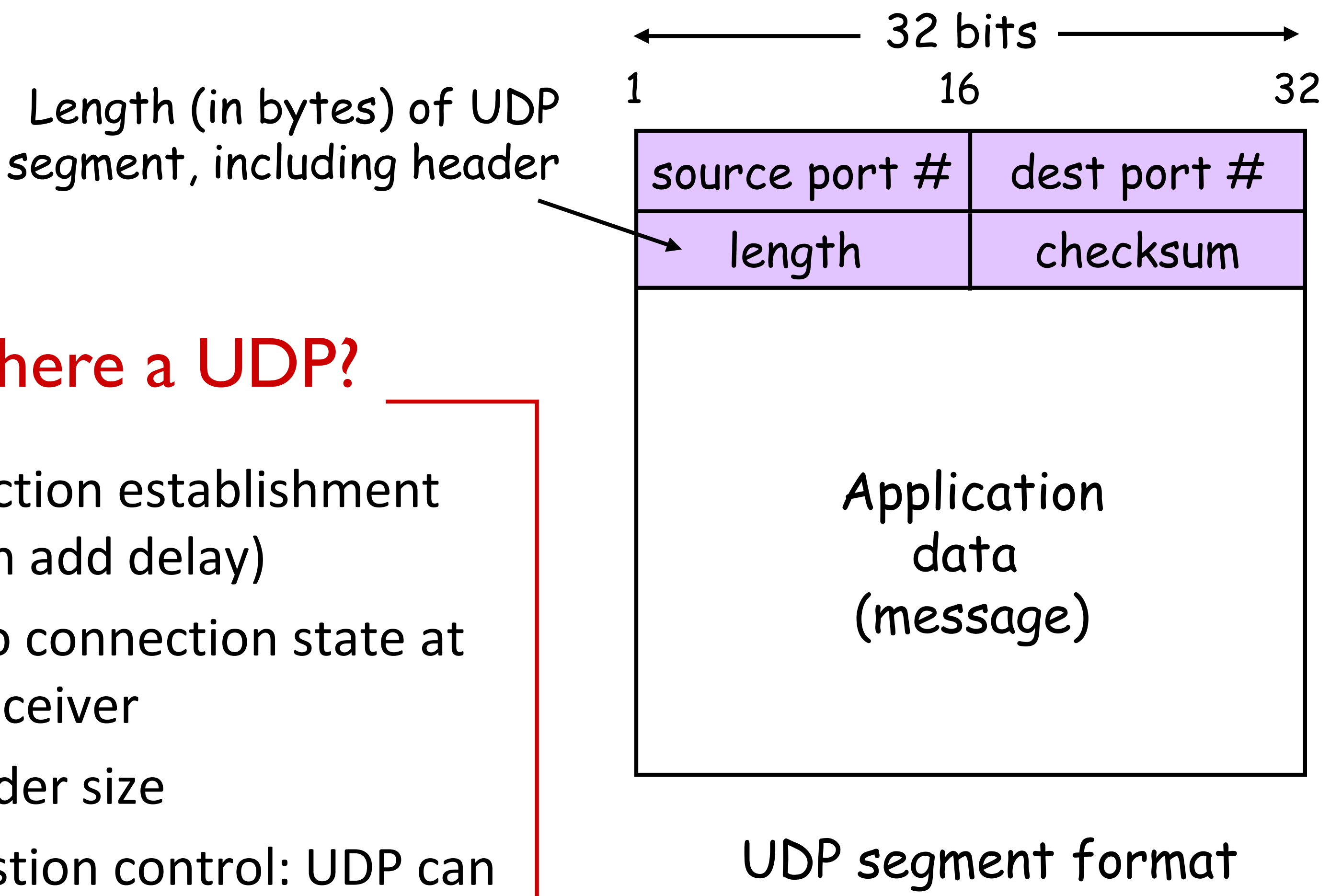
- Transport layer protocols run in hosts.
  - Sender side: breaks app message into *segments* (as needed), passes them to network layer (aka IP layer).
  - Receiver side: reassembles segments into message, passes it to app layer.
  - Packet switches (routers) in between: only check destination IP address to decide routing.

Lectures 4&5: Roadmap

UDP: User Datagram Protocol [RFC 768]

- UDP adds very little service on top of IP:
  - Multiplexing at sender: UDP gathers data from processes, forms packets and passes them to IP
  - De-multiplexing at receiver: UDP receives packets from lower layer and dispatches them to the right processes.
  - Checksum
- UDP transmission is unreliable
  - Often used by streaming multimedia apps (loss tolerant & rate sensitive)

UDP Header


32 bits
1
16
32
Length (in bytes) of UDP segment, including header
source port #
dest port #
length
checksum
Application data (message)
UDP segment format

Why is there a UDP?

- No connection establishment (which can add delay)
- Simple: no connection state at sender, receiver
- Small header size
- No congestion control: UDP can blast away as fast as desired

Checksum Example

- Example: add two 16-bit integers

```
                                                   carry
                   1 1 1 0   0 1 1 0   0 1 1 0   0 1 1 0
wraparound         1 1 0 1   0 1 0 1   0 1 0 1   0 1 0 1
carry            ----------------------------------------
                 1 1 0 1 1   1 0 1 1   1 0 1 1   1 0 1 1
                 ----------------------------------------
sum                1 0 1 1   1 0 1 1   1 0 1 1   1 1 0 0
Checksum           0 1 0 0   0 1 0 0   0 1 0 0   0 0 1 1
(1's complement)
```

Lectures 4&5: Roadmap

3.1 Transport-layer Services

3.3 Connectionless Transport: UDP

3.4 Principles of Reliable Data Transfer

3.5 Connection-oriented transport: TCP

Reliable Transfer over Unreliable Channel

- Underlying network may
  - corrupt packets
  - drop packets
  - re-order packets (not considered in this lecture)
  - deliver packets after an arbitrarily long delay

- End-to-end reliable transport service should
  - guarantee packets delivery and correctness
  - deliver packets (to receiver application) in the same order they are sent

RDT Summary

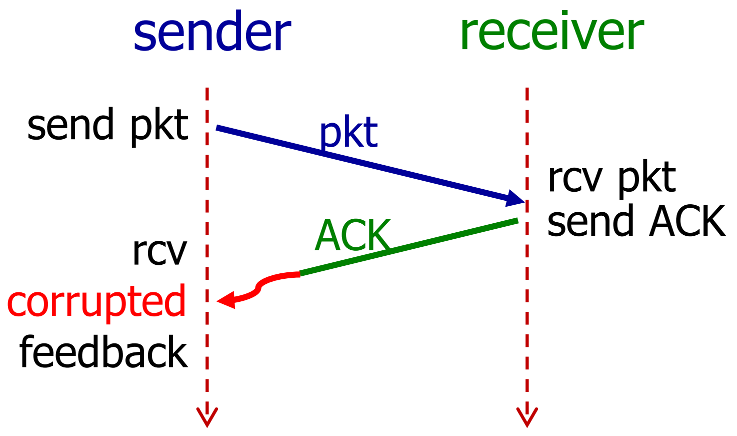
rdt Version	Scenario	Features Used
1.0	no error	nothing
2.0	data Bit Error	checksum, ACK/NAK
2.1	data Bit Error ACK/NAK Bit Error	checksum, ACK/NAK, sequence Number
2.2	Same as 2.1	NAK free
3.0	data Bit Error ACK Bit Error packet Loss	checksum, ACK, sequence Number, timeout/re-transmission

rdt 2.0: Channel with *Bit Errors*

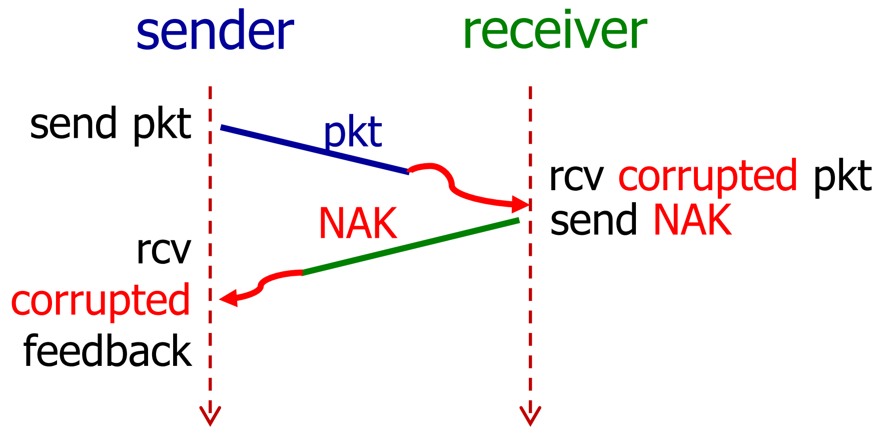
- Assumption:
  - underlying channel may flip bits in packets
  - other than that, the channel is perfect
- Q1: how to detect bit errors?
  - Receiver may use ***checksum*** to detect bit errors.
- Q2: how to recover from bit errors?
  - *Acknowledgements (ACKs):* receiver explicitly tells sender that packet received is OK.
  - *Negative acknowledgements (NAKs):* receiver explicitly tells sender that packet has errors.
    - Sender retransmits packet on receipt of NAK.

rdt 2.0 has a Fatal Flaw!

- What happens if ACK/NAK is corrupted?
    - Sender doesn't know what happened at receiver!
- So what should the sender do?
    - Sender just retransmits when receives garbled ACK or NAK.
    - Questions: does this work?



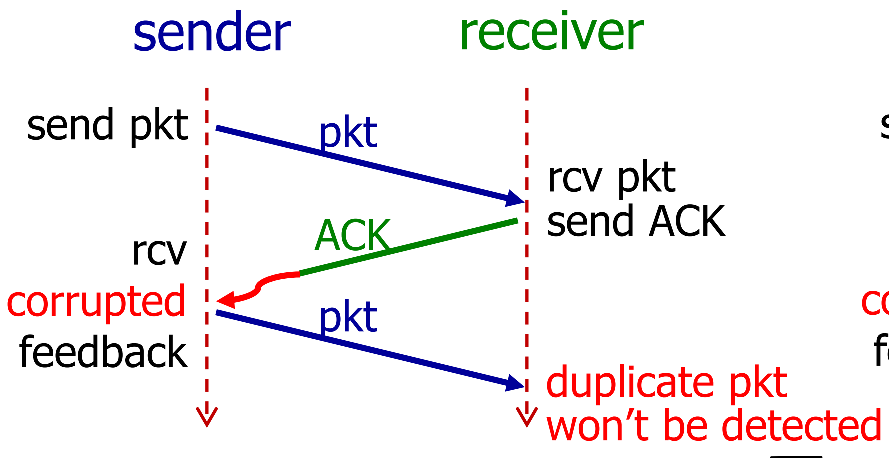
a) corrupted ACK



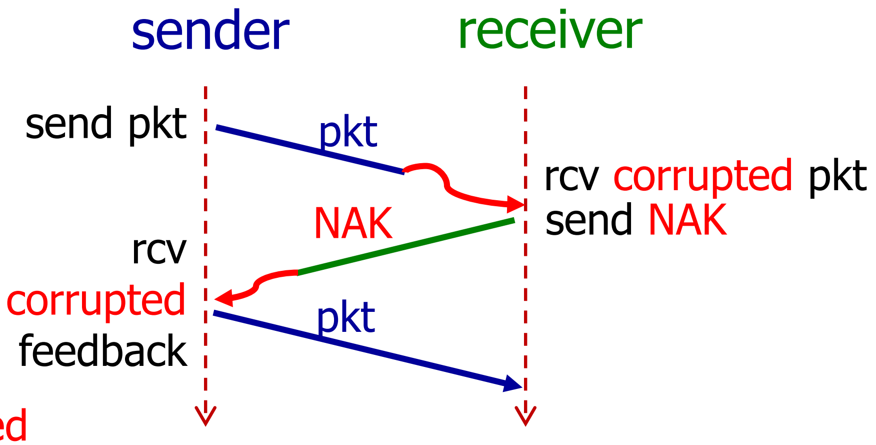
b) corrupted NAK

rdt 2.0 has a Fatal Flaw!

- Sender just retransmits when it receives garbled feedback.
  - This may cause retransmission of correctly received packet!
  - Question: how can receiver identify duplicate packet?



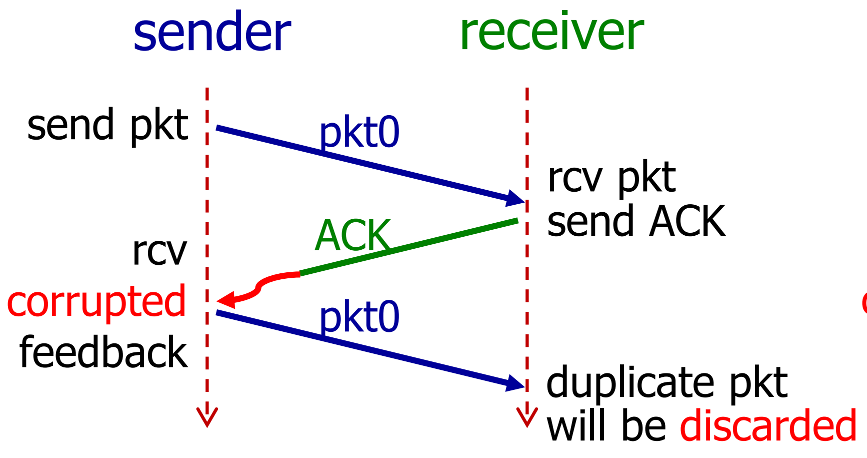
a) corrupted ACK



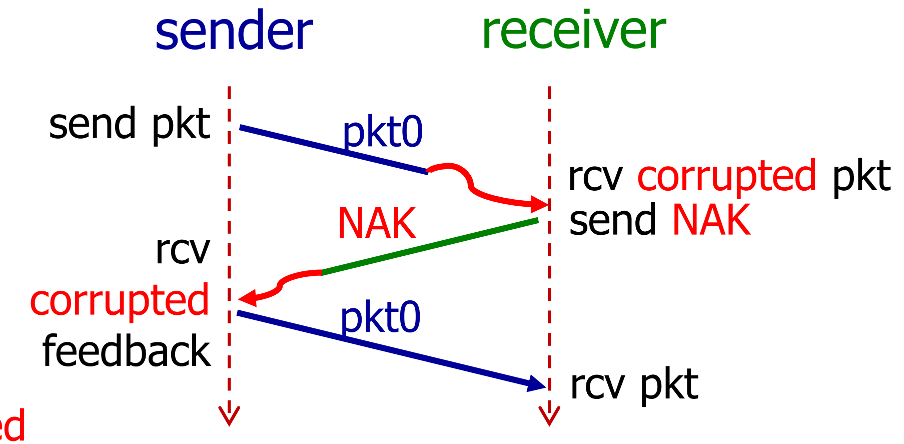
b) corrupted NAK

rdt 2.1: rdt 2.0 + Packet Seq.

- To handle duplicates:
  - Sender retransmits current packet if ACK/NAK is garbled.
  - Sender adds *sequence number* to each packet.
  - Receiver discards (doesn't deliver up) duplicate packet.
- This gives rise to protocol rdt 2.1.



a) corrupted ACK



b) corrupted NAK

RDT Summary

rdt Version	Scenario	Features Used
1.0	no error	nothing
2.0	data Bit Error	checksum, ACK/NAK
2.1	data Bit Error ACK/NAK Bit Error	checksum, ACK/NAK, sequence Number
2.2	Same as 2.1	NAK free
3.0	data Bit Error ACK Bit Error packet Loss	checksum, ACK, sequence Number, timeout/re-transmission

rdt 3.0: Channel with *Errors* and *Loss*

- Assumption: underlying channel
  - may flip bits in packets
  - may lose packets
  - may incur arbitrarily long packet delay
  - but won't re-order packets

- Question: how to detect packet loss?
  - checksum, ACKs, seq. #, retransmissions will be of help… but not enough

rdt 3.0: Channel with *Errors* and *Loss*

- To handle packet loss:
  - Sender waits "reasonable" amount of time for ACK.
  - Sender retransmits if no ACK is received till *timeout*.

- Question: what if packet (or ACK) is just delayed, but not lost?
  - Timeout will trigger retransmission.
  - Retransmission will generate duplicates in this case, but receiver may use seq. # to detect it.
  - Receiver must specify seq. # of the packet being ACKed (check scenario (d) two pages later).

RDT Summary

rdt Version	Scenario	Features Used
1.0	no error	nothing
2.0	data Bit Error	checksum, ACK/NAK
2.1	data Bit Error ACK/NAK Bit Error	checksum, ACK/NAK, sequence Number
2.2	Same as 2.1	NAK free
3.0	data Bit Error ACK Bit Error packet Loss	checksum, ACK, sequence Number, timeout/re-transmission

Q1

A DNS query is directly encapsulated in a __________ segment.

A. TCP
B. UDP
C. IP

Q2

In the Internet protocol stack, network layer is responsible for _________ communication while transport layer is responsible for _________ communication.

A. host-to-host; process-to-process
B. process-to-process; host-to-host

Q3

Suppose two hosts are connected by a direct link of 1 Mbps. A stop-and-wait protocol is used to transfer 10 packets from the sender to the receiver. Each packet is 1000 bytes long. RTT is 24 milliseconds. Assume that transmission is error-free and ACK packets are of negligible size.

When will the sender receive the last ACK?

A. 32 milliseconds
B. 160 milliseconds
C. 320 milliseconds

Q4

Suppose we want to design a stop-and-wait, reliable protocol for communication between a sender and a receiver over a channel with the following characteristics: data packets may be lost or corrupted but will not be reordered. Feedback packets are always received in good order. Moreover, the maximum RTT between sender and receiver is known.

Which of the following statements about the reliable protocol is TRUE?

A. Sender must attach a sequence number to every data packet.

B. If sender set the timer properly, receiver definitely won't receive duplicate packets.

C. Receiver should discard corrupted data packet but must acknowledge it.