Internet, Principes et Protocoles (IPP)

Reliable transfer with Sliding Window

How to provide a reliable data transfer with a sliding window

- How to react upon reception of a control segment ?
- Sender's and receiver's behaviours

Basic solutions

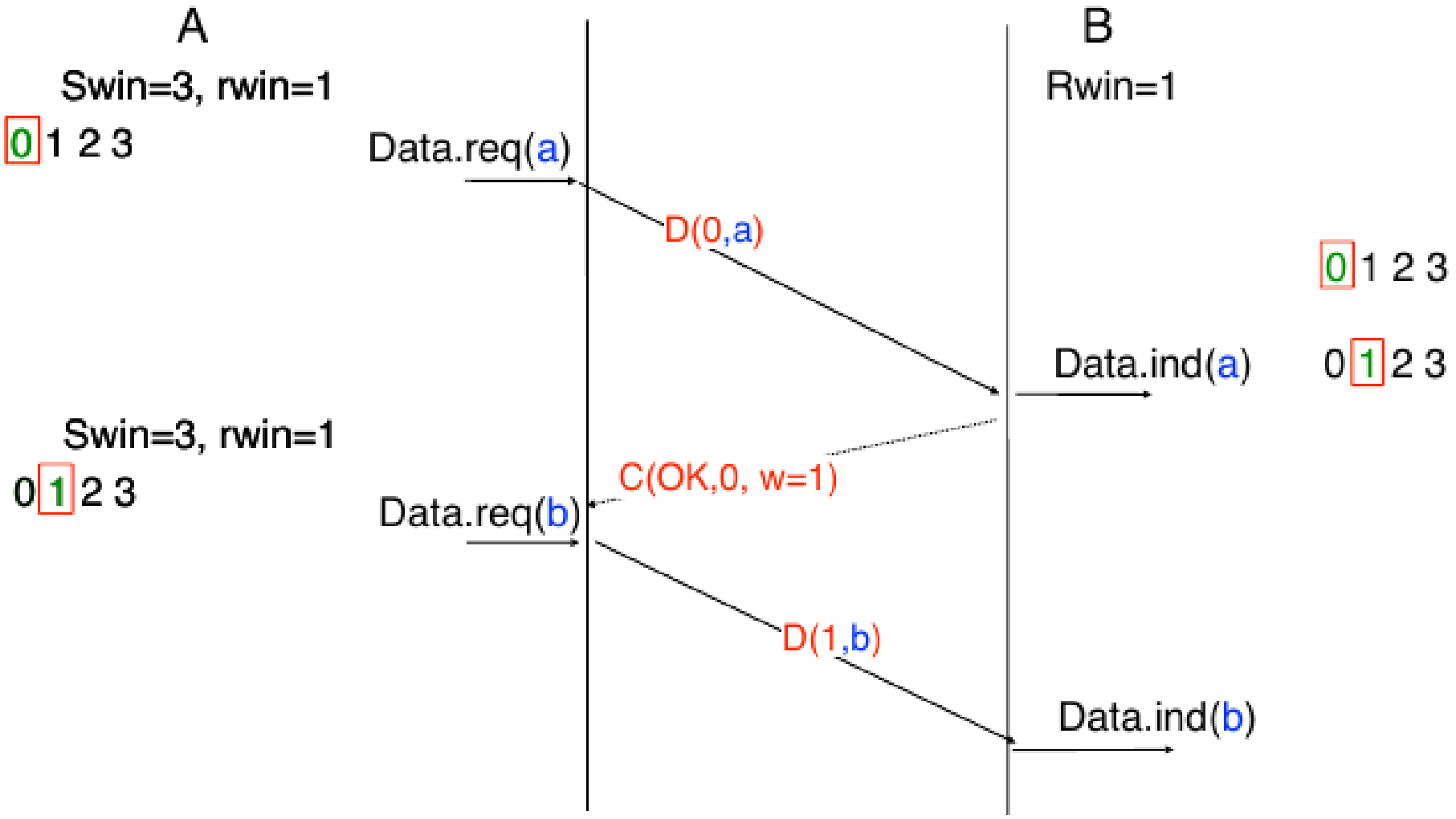
Go-Back-N

- simple implementation, in particular on receiving side
- throughput will be limited when losses occur

Selective Repeat

- more difficult from an implementation viewpoint
- throughput can remain high when limited losses occur

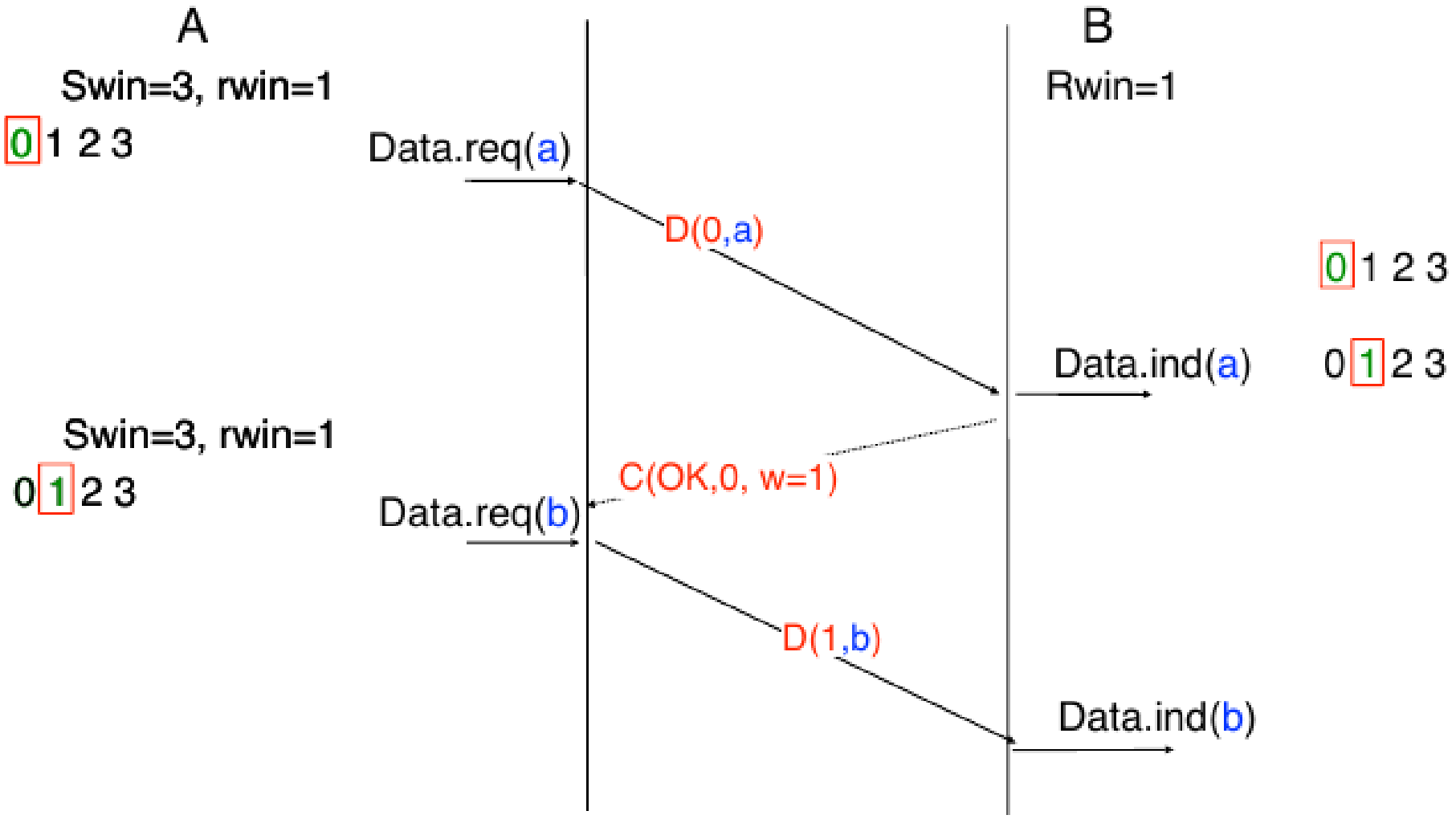
Window Size management



Transmition Errors

- Errors in Payload – CheckSum (CRC)
- Paquets can be lost – Timer
- Paquets can arrive out-of-order – Sequence Number
- Paquets can be duplicated – Sequence Number

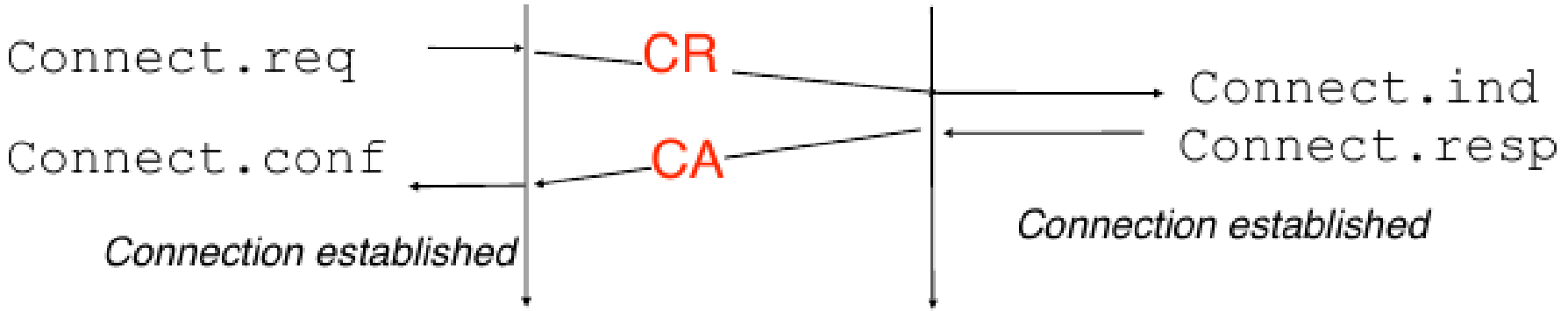
Window Size management



Network Layer

- ~~Reliable transfer~~
- Connection Establishment
- Connection Release
- TCP - UDP

Connection Establishment



Principle

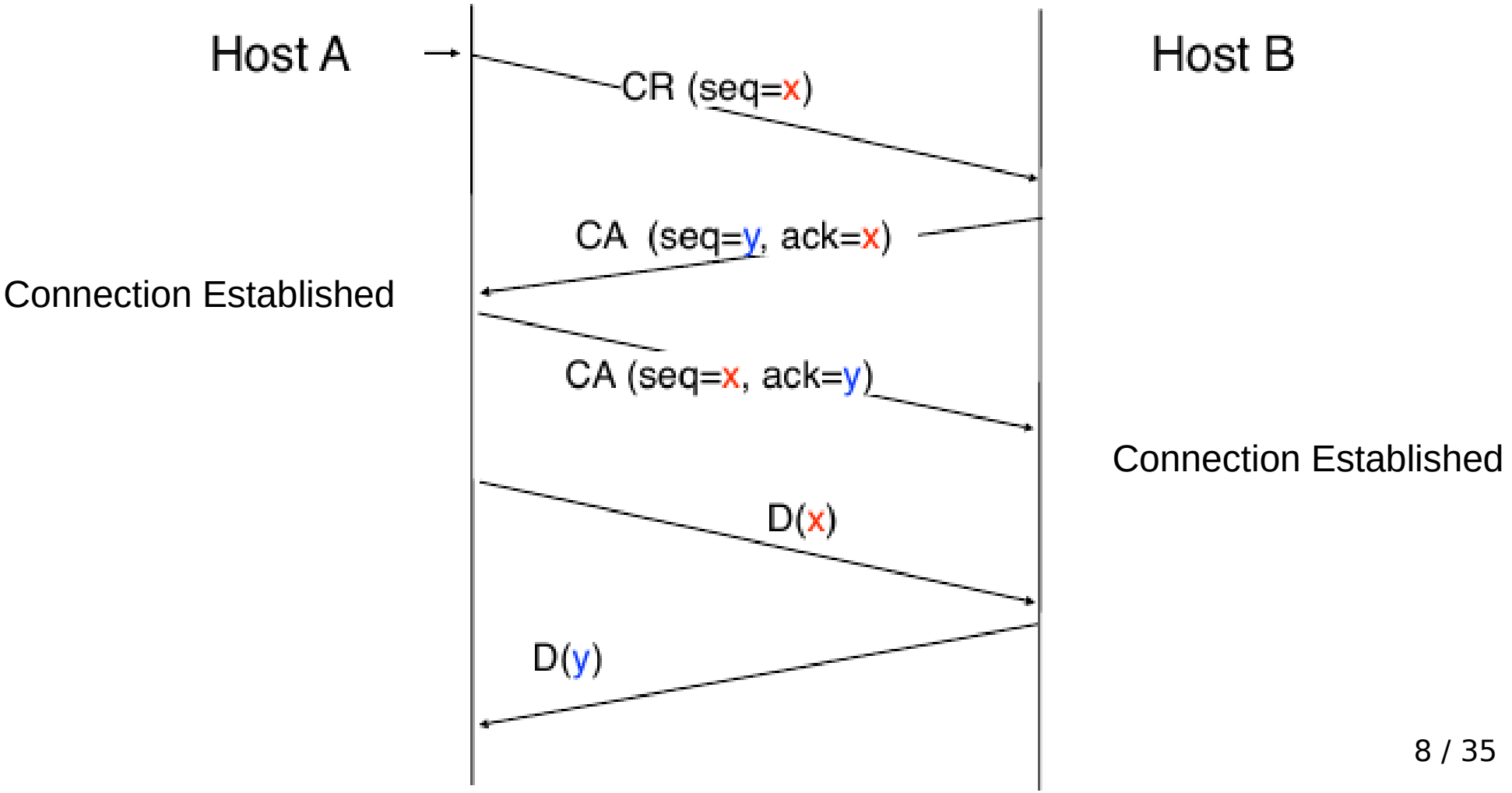
2 control segments

CR is used to request a connection establishment
CA is used to acknowledge a connection establishment

Is this sufficient with an imperfect network layer service ?

Connection Establishment

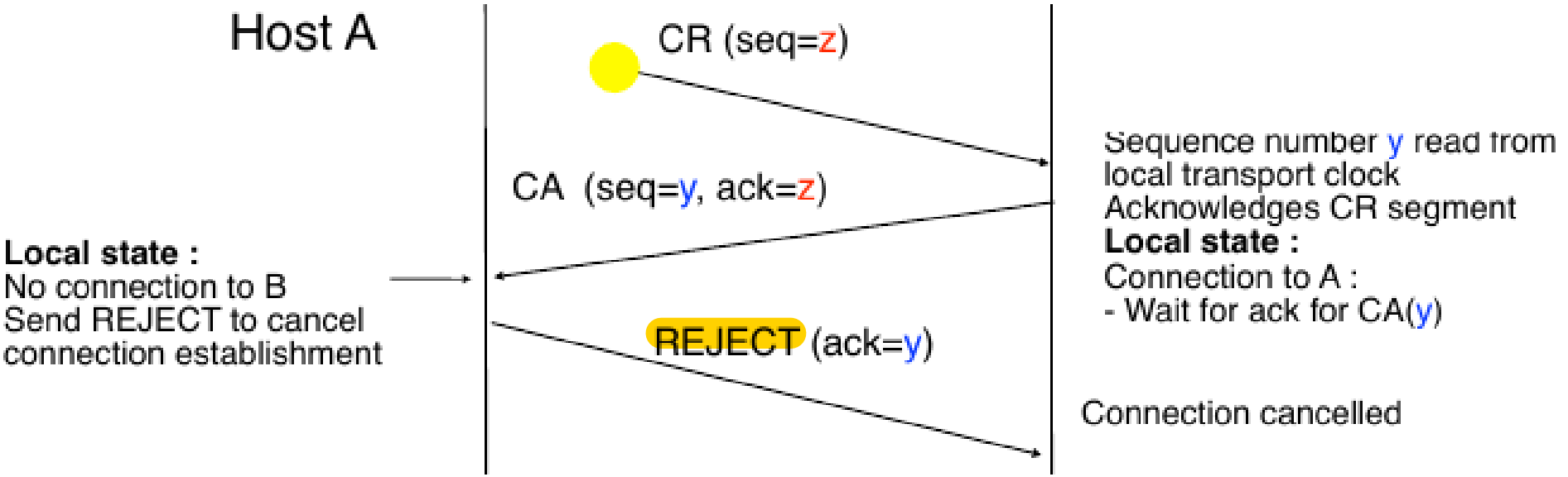
3-way handshake



Connection Establishment

3-way handshake

Rejecting a connection



Network Layer

- ~~Reliable transfer~~
- ~~Connection Establishment~~
- Connection Release
- TCP - UDP

Connection Release

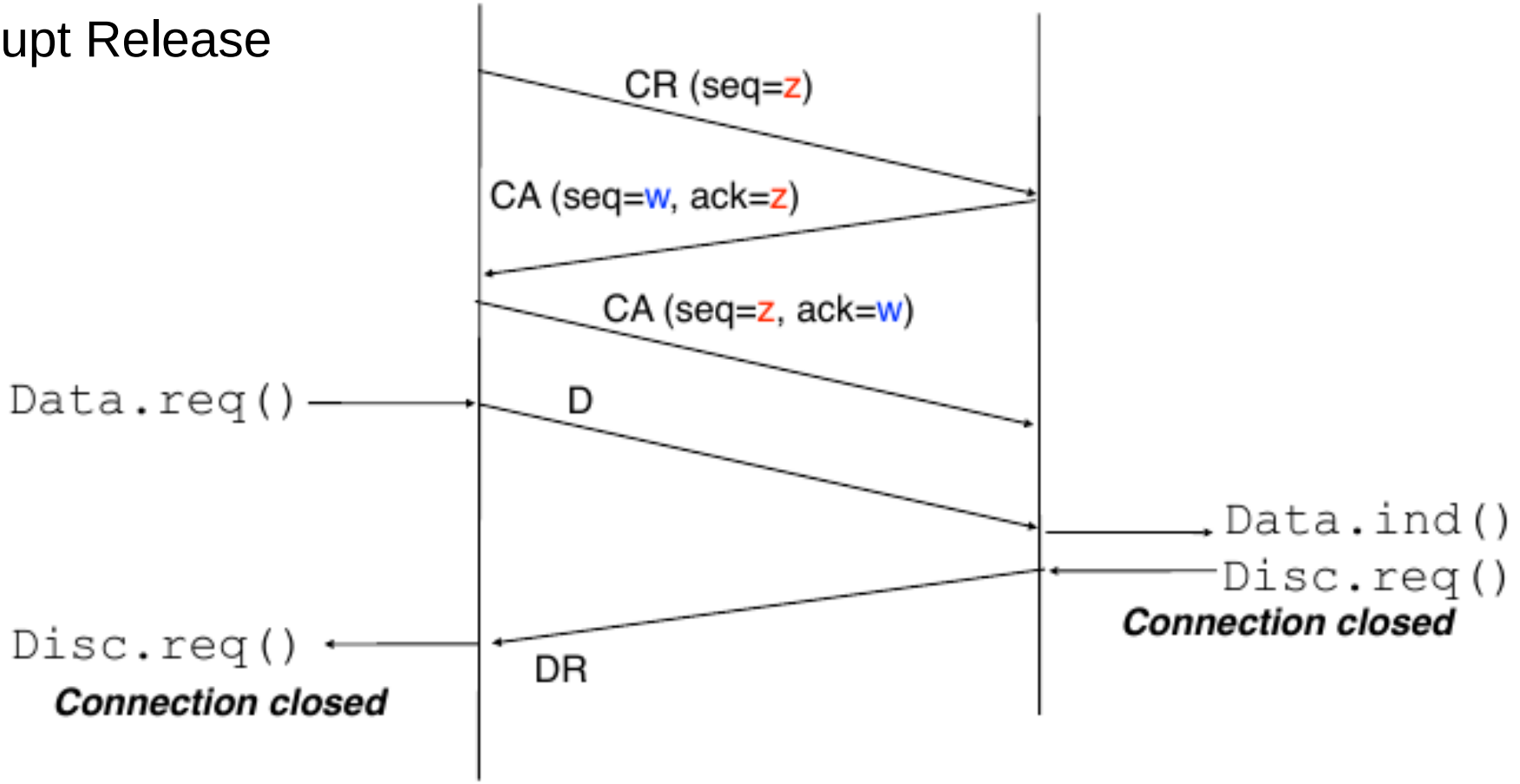
A transport connection can be used in both directions

Types of connection release

- Abrupt connection release
  - One of the transport entities closes both directions of data transfer
  - can lead to losses of data
- Graceful release
  - Each transport entity closes its own direction of data transfer
  - connection will be closed once all data has been correctly delivered

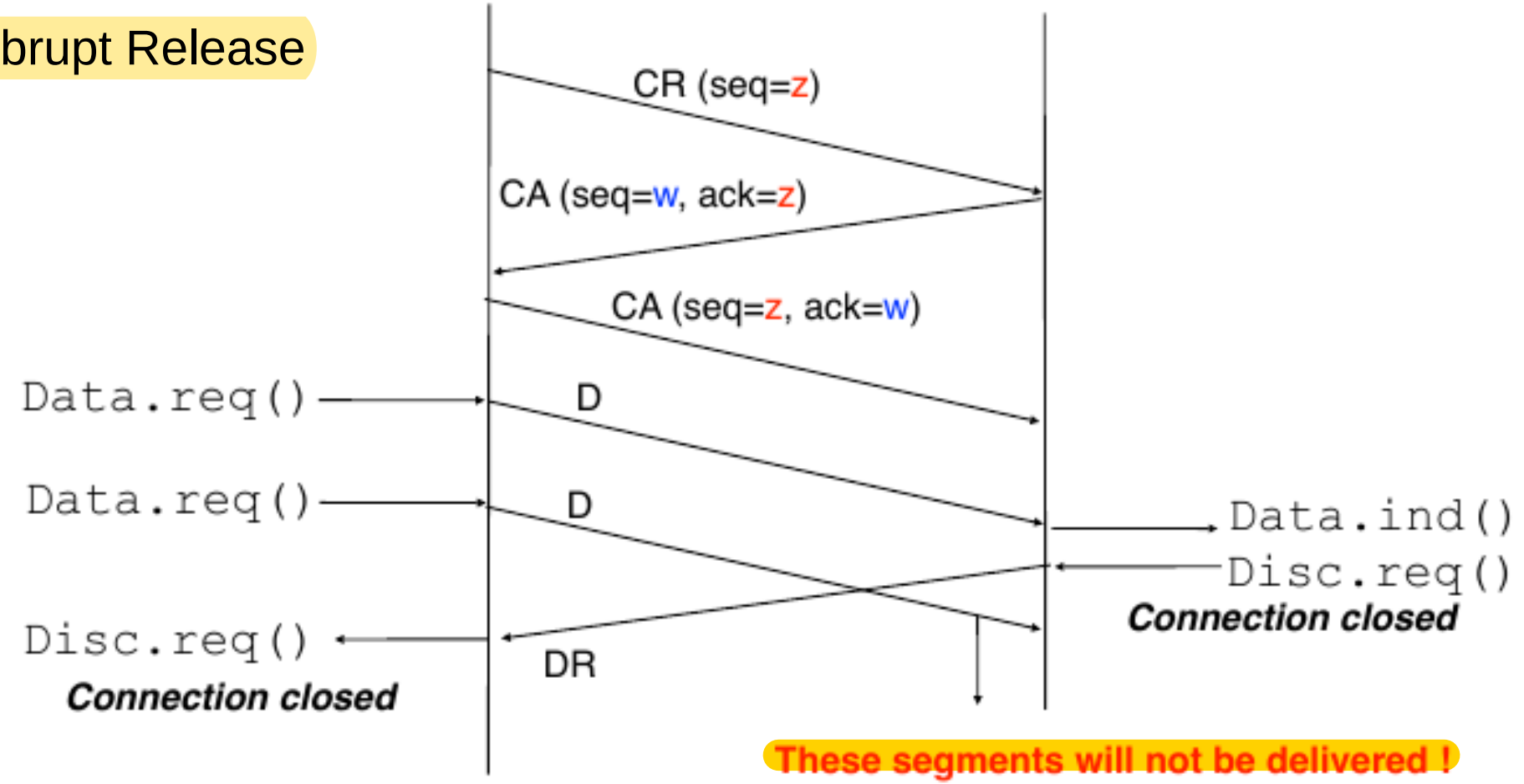
Connection Release

Abrupt Release



Connection Release

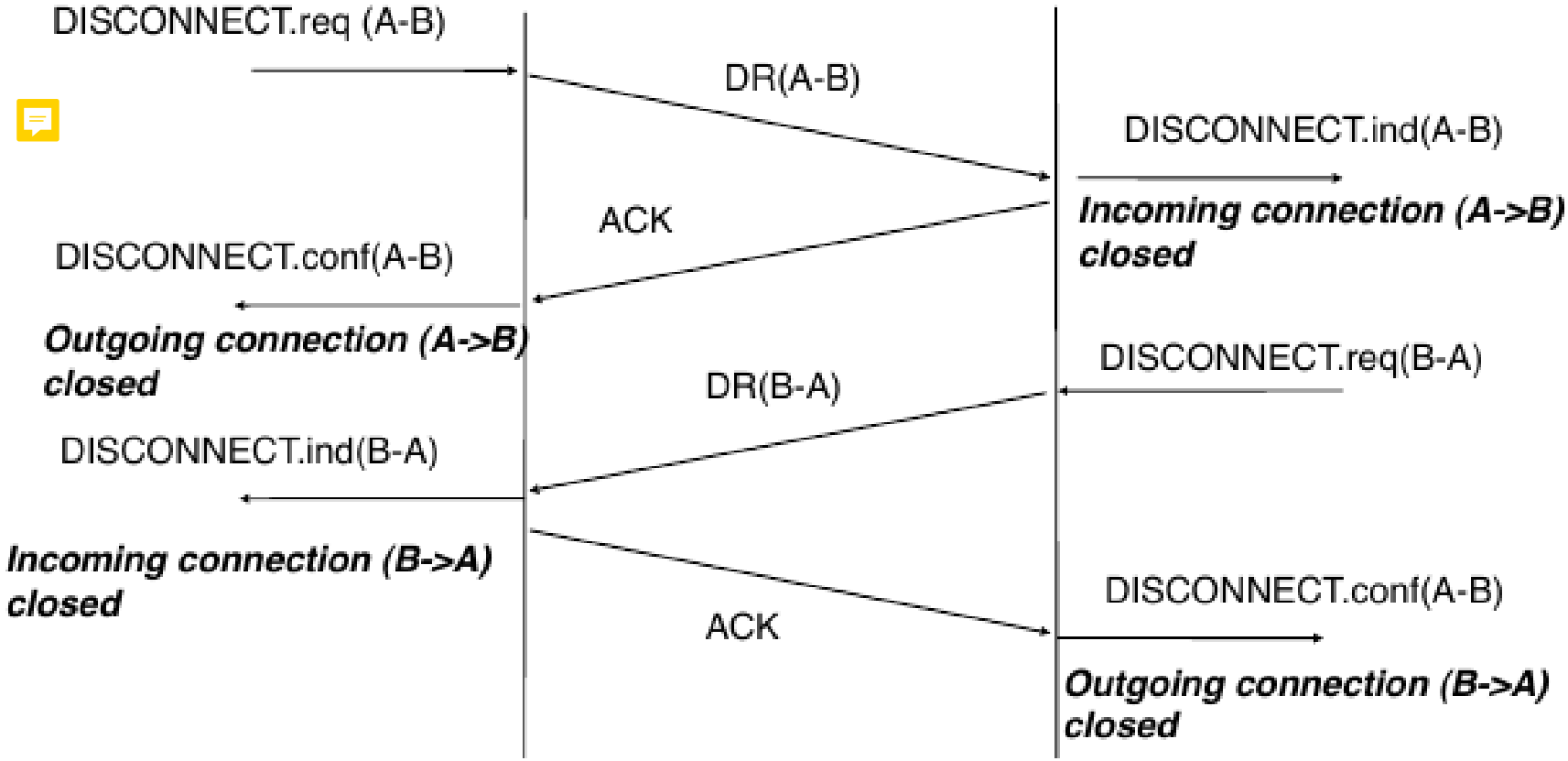
Abrupt Release



Graceful Release

Principle

Each entity closes its own direction of data transfer once all its data have been sent



Connection Release

- Transport layer does not recover itself from abrupt connection releases
  - Possible solutions
    - Application reopens the connection and restarts the data transfer
    - Session Layer
    - Transaction processing

Network Layer

- ~~Reliable transfer~~
- ~~Connection Establishment~~
- ~~Connection Release~~
- TCP - UDP

TCP

Transmission Control Protocol

Provides a reliable byte stream service

Characteristics of the TCP service

- TCP connections
- Data transfer is reliable
  - no loss
  - no errors
  - no duplications
- Data transfer is bidirectional
- TCP relies on the IP service
- TCP only supports unicast

TCP

How to identify a TCP connection

- Address of the source application
  - IP Address of the source host
  - TCP port number of the application on source host
- Address of the destination application
  - IP Address of the destination host
  - TCP port number of the application on destination host

Each TCP segment contains the identification of the connection it belongs to

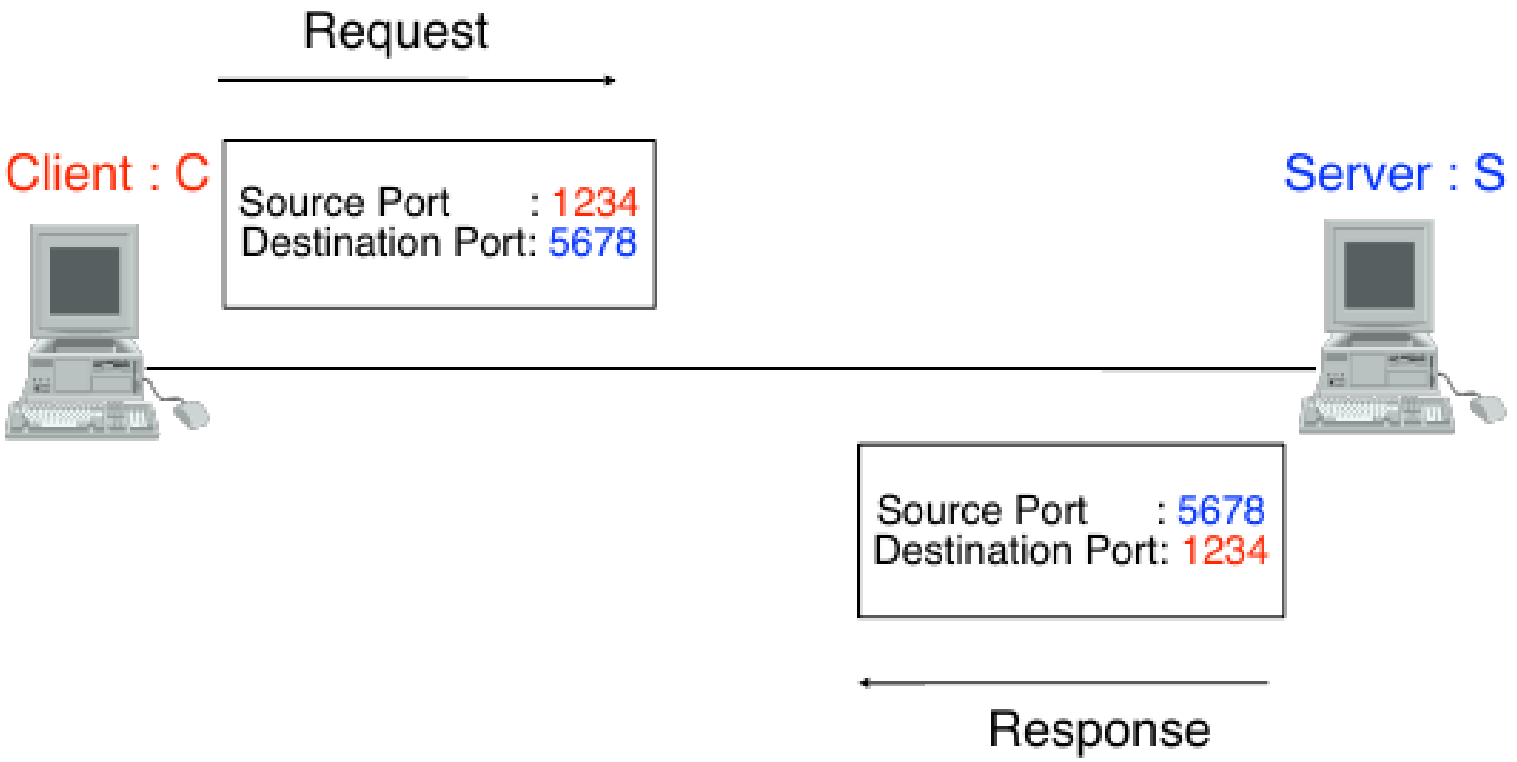
What is port

- When associated with an IP address, a port identifies a specific application on a given host.

- If a computer runs Skype and Youtube at the same time, we need to diferenciate the network flows between both applications. The computer has only one IP address, so we use Ports to identify the different applications.
- Port numbers go from 0 to 65535.

What is a port

Usage of TCP port numbers



Established TCP connections on client

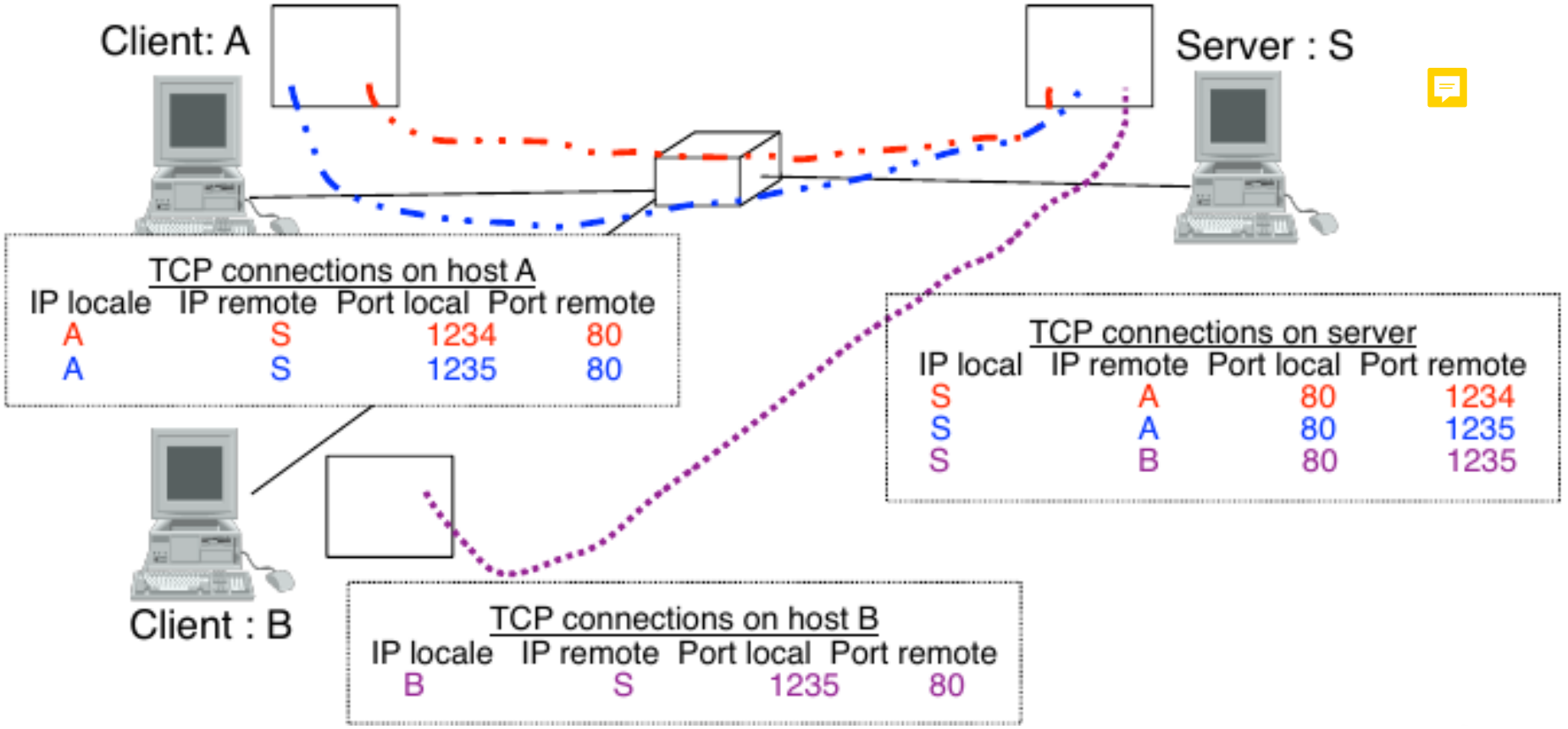
Local IP	Remote IP	Local Port	Remote Port
C	S	1234	5678

Established TCP connections on server

Local IP	Remote IP	Local Port	Remote Port
S	C	5678	1234

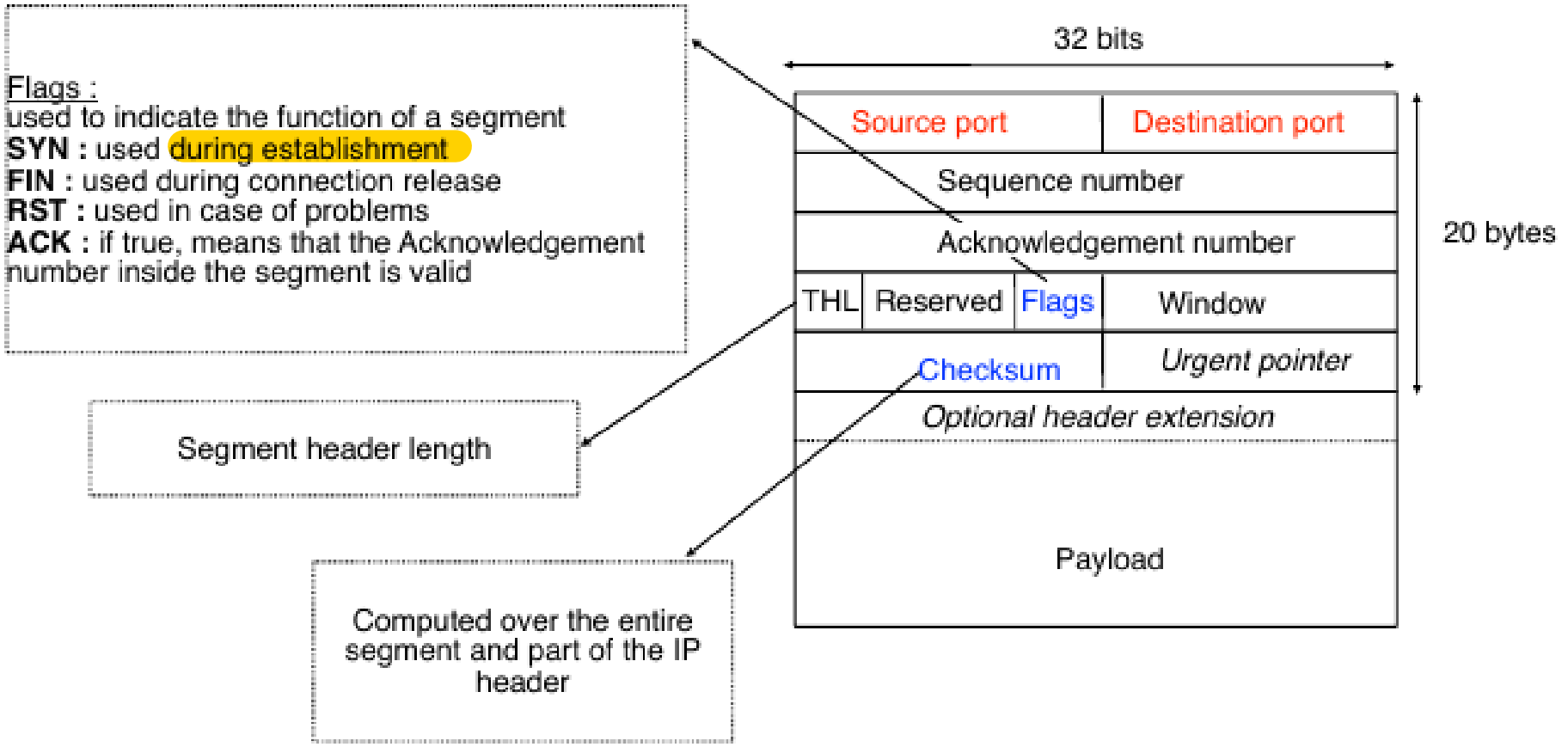
Ports Examples

How to open several TCP connections at the same time ?

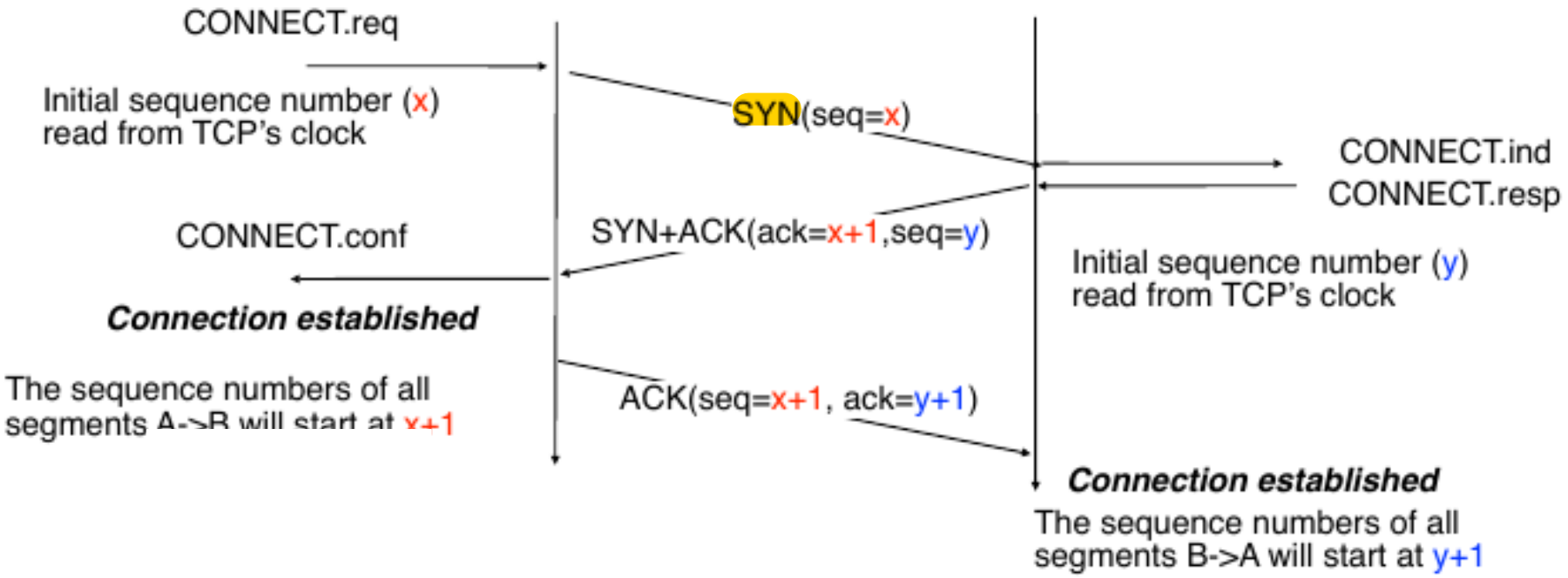


TCP Paquet

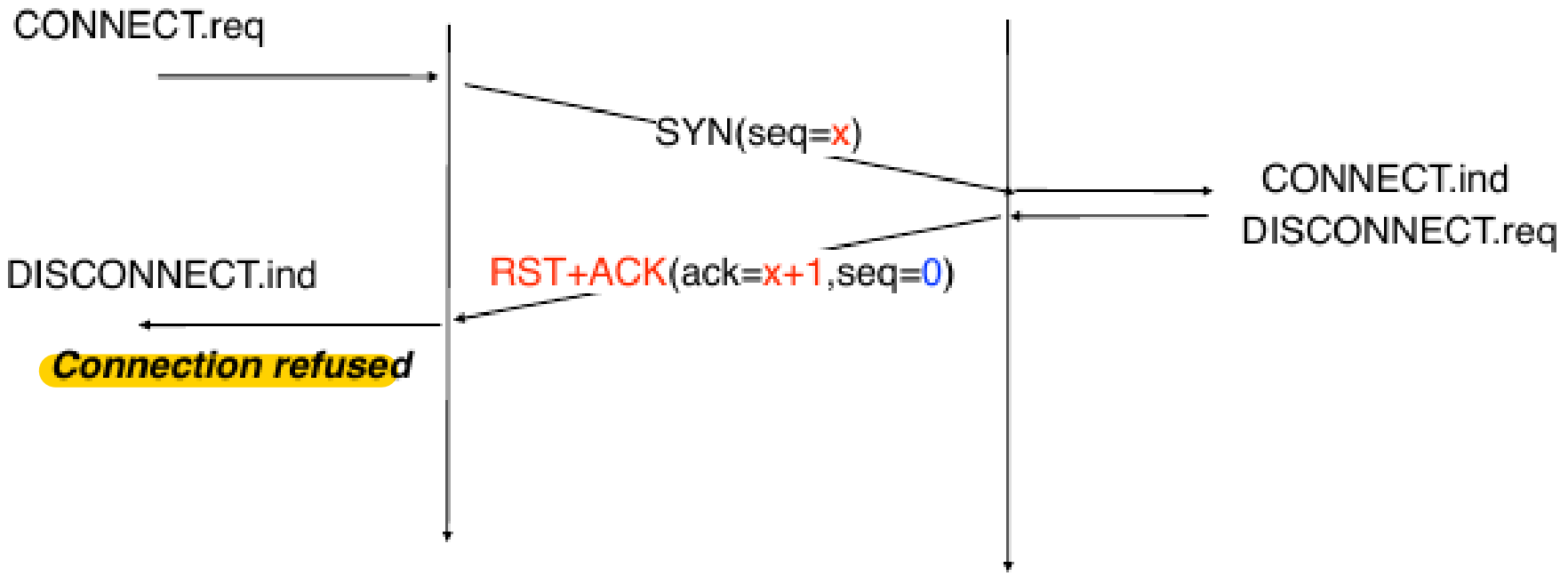
Single segment format



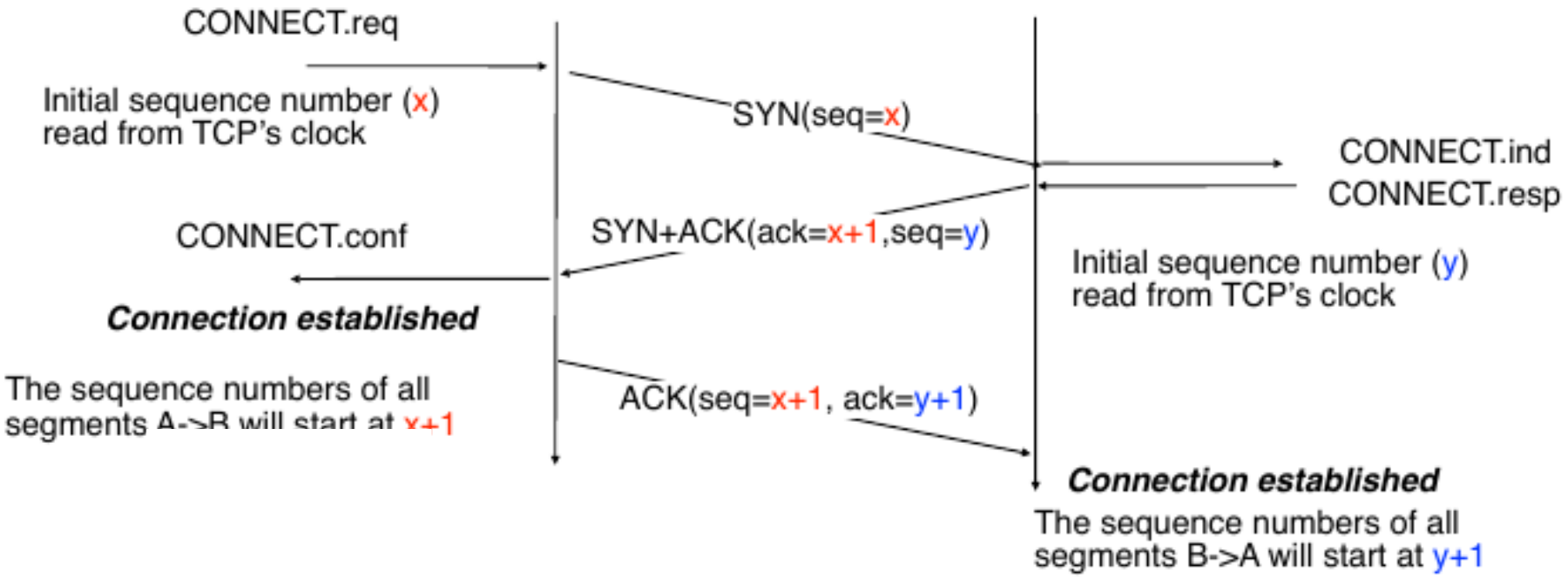
TCP 3-way handshake



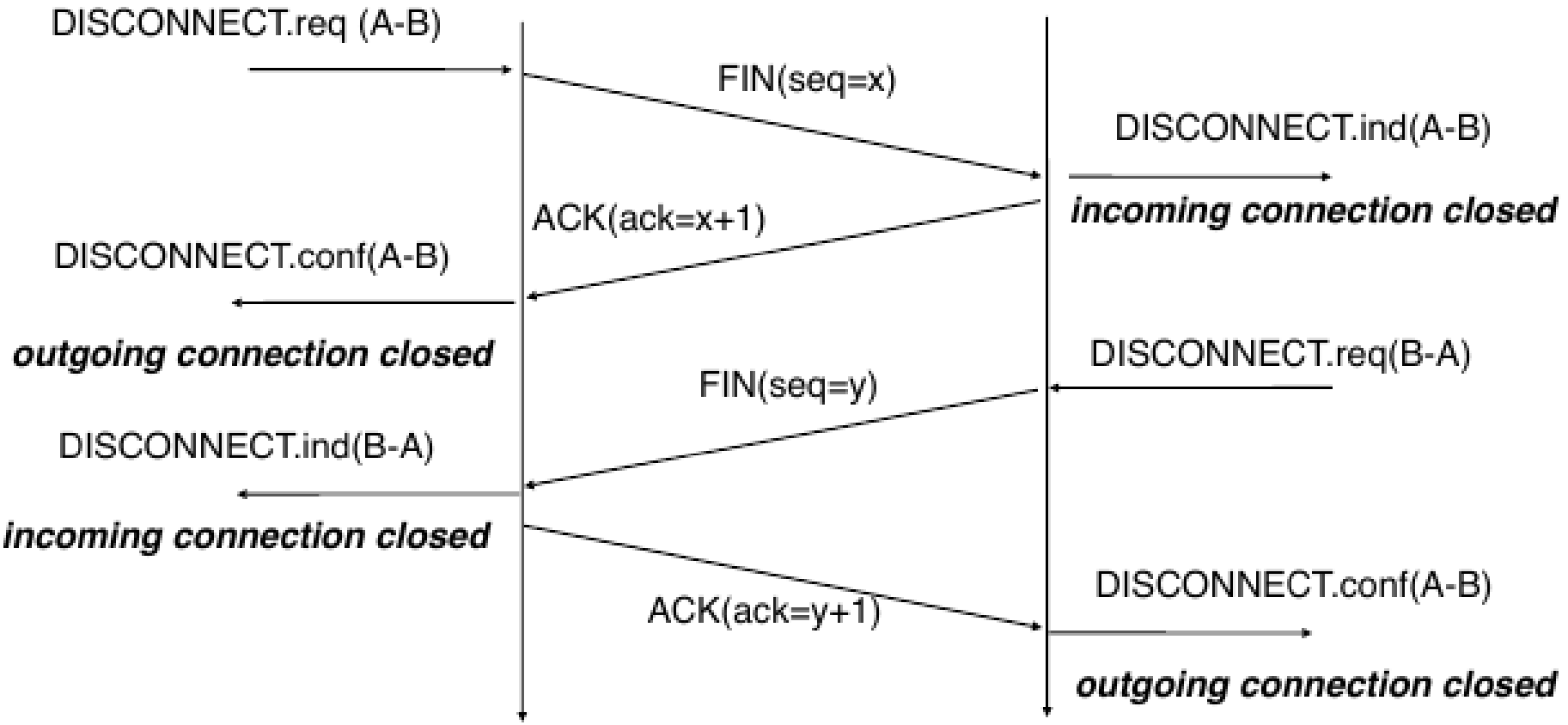
TCP Connection Rejection



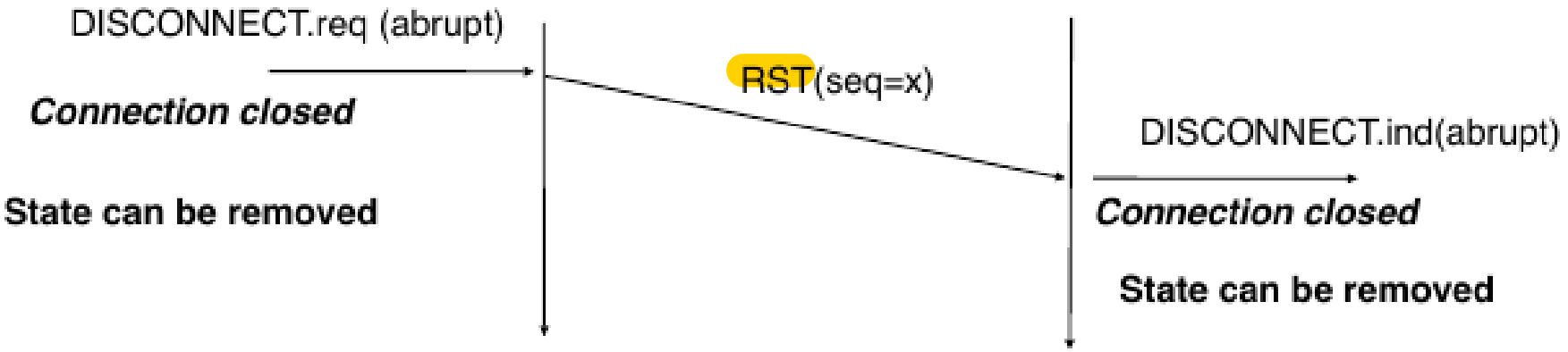
TCP 3-way handshake



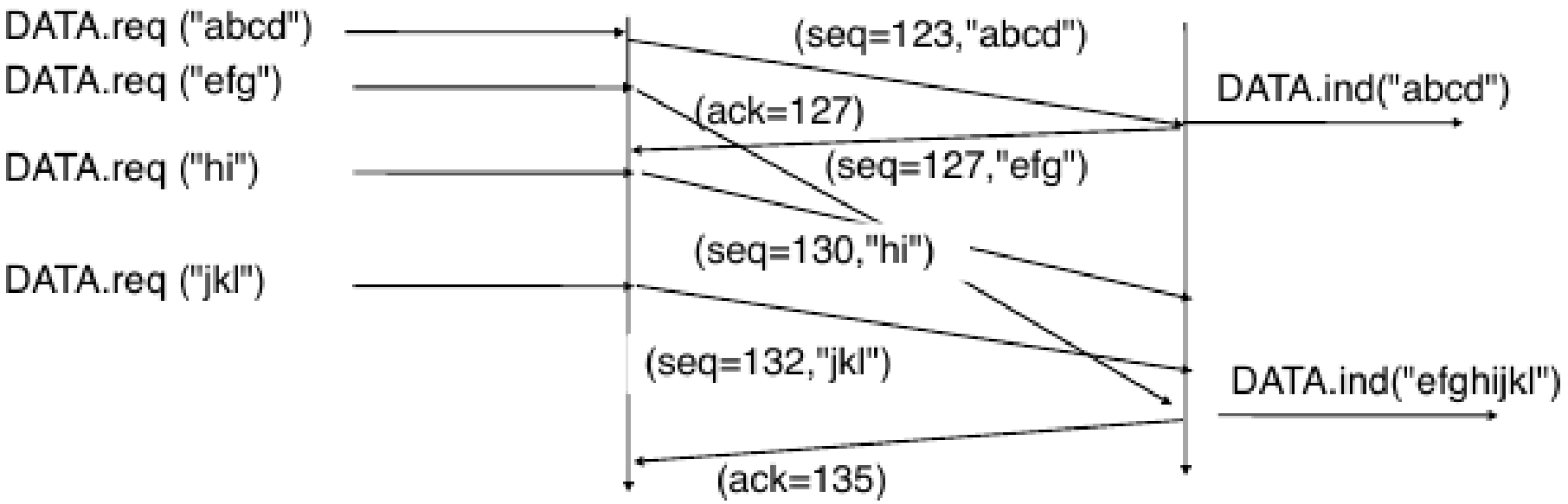
TCP Graceful Release



TCP Abrupt Release



TCP Reliable transfer



Note: The Ack number = sequence number + length of payload
("I have received XXX bytes so far.")

Network Layer

- ~~Reliable transfer~~
- ~~Connection Establishment~~
- Connection Release
- ~~TCP~~ - UDP

UDP

User Datagram Protocol (UDP)
- The simplest transport protocol

Goal
- Allow applications to exchange small SDUs by relying on the IP service
  - on most operating systems, sending raw IP packets requires special privileges while any application can use directly the transport service

Constraint
- The implementation of the UDP transport entity should remain as simple as possible

UDP

Which mechanisms inside UDP ?

Application identification

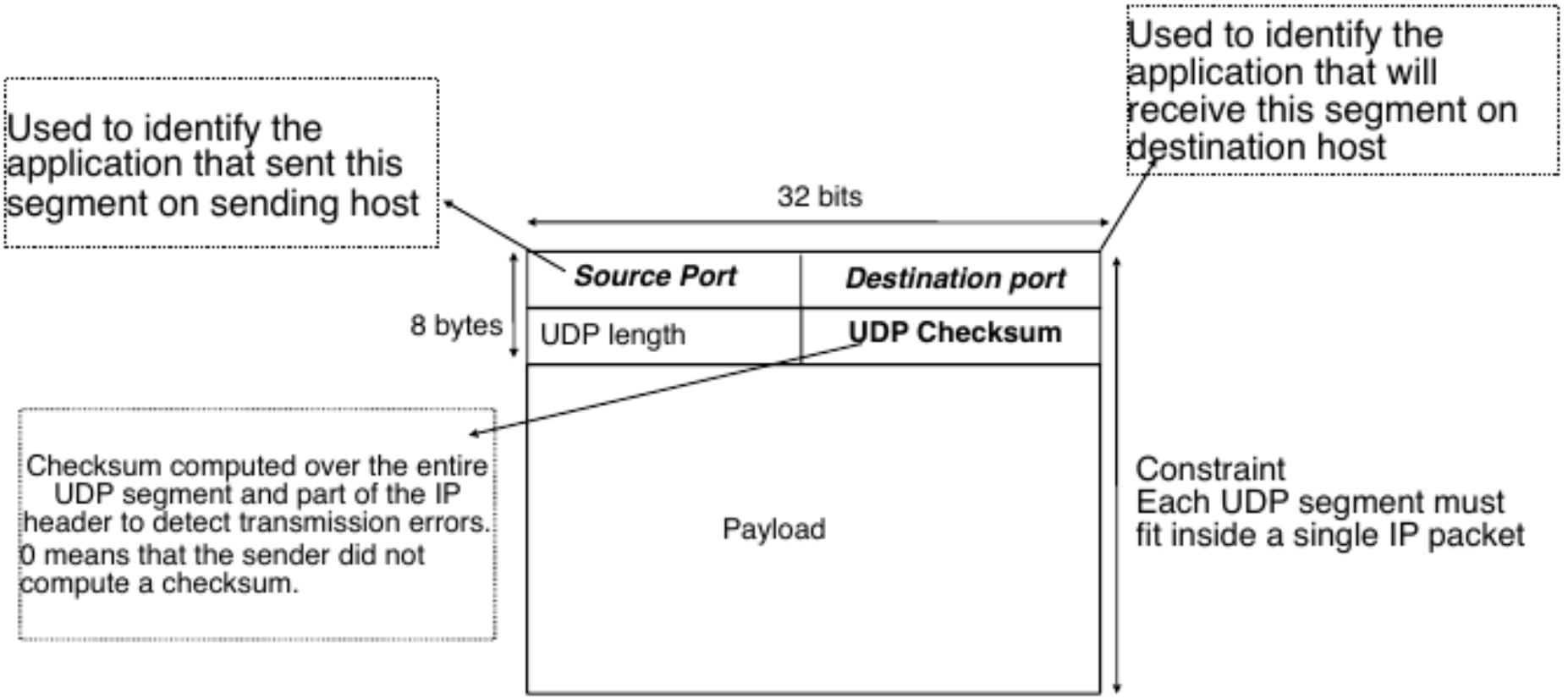
Several applications running on the same host must be able to use the UDP service

Solution

Source port to identify sending application
Destination port to identify receiving application
Each UDP segment contains both the source and the destination ports

Detection of transmission errors

UDP Segment


Used to identify the application that sent this segment on sending host
Used to identify the application that will receive this segment on destination host
32 bits
Source Port
Destination port
8 bytes
UDP length
UDP Checksum
Checksum computed over the entire UDP segment and part of the IP header to detect transmission errors. 0 means that the sender did not compute a checksum.
Payload
Constraint
Each UDP segment must fit inside a single IP packet

UDP

Limitations

Maximum length of UDP SDUs depends on maximum size of IP packets

Unreliable connectionless service

SDUs can get lost but transmission errors will be detected

UDP does not preserve ordering

UDP does not detect nor prevent duplication

UDP Usage

Request-response applications where requests and responses are short and short delay is required or used in LAN environments

- DNS
- Remote Procedure Call
- NFS
- Games

Multimedia transfer were reliable delivery is not necessary and retransmissions would cause too long delays

- Voice over IP
- Video over IP

TCP end + udp + NMAP + dDOS server opeing connections+exercice