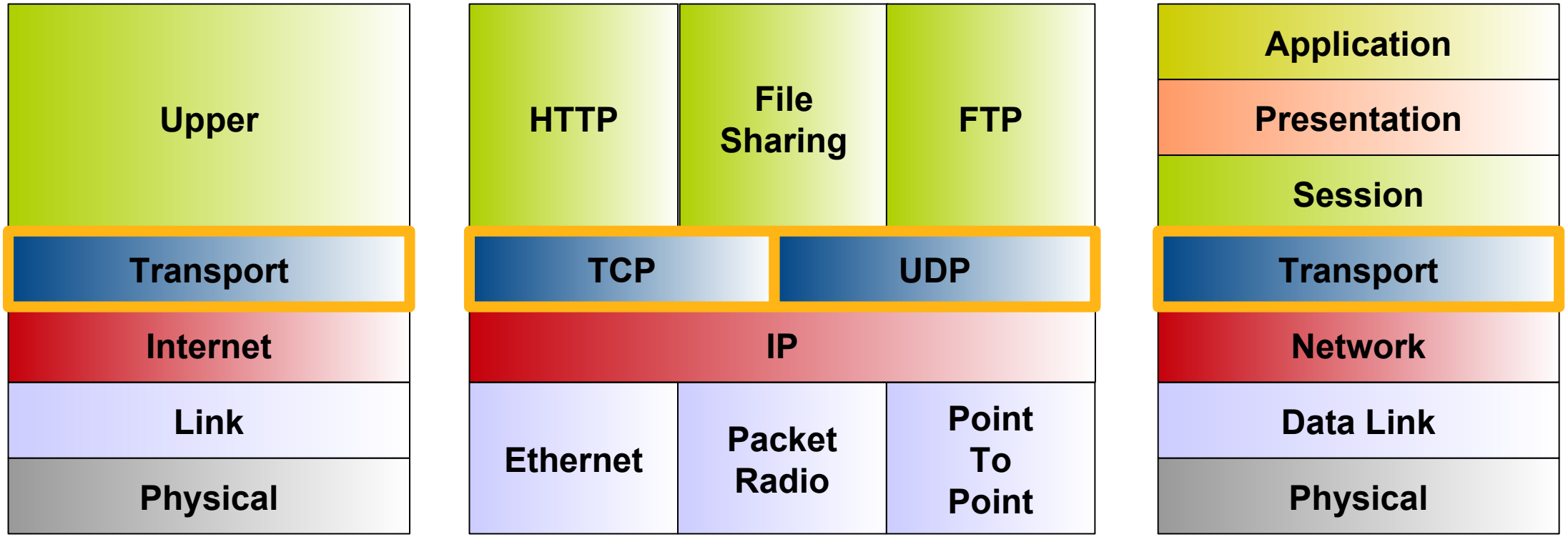
UDP & TCP

Fundamentos de Redes

Mestrado Integrado em
Engenharia de Computadores e Telemática
DETI-UA

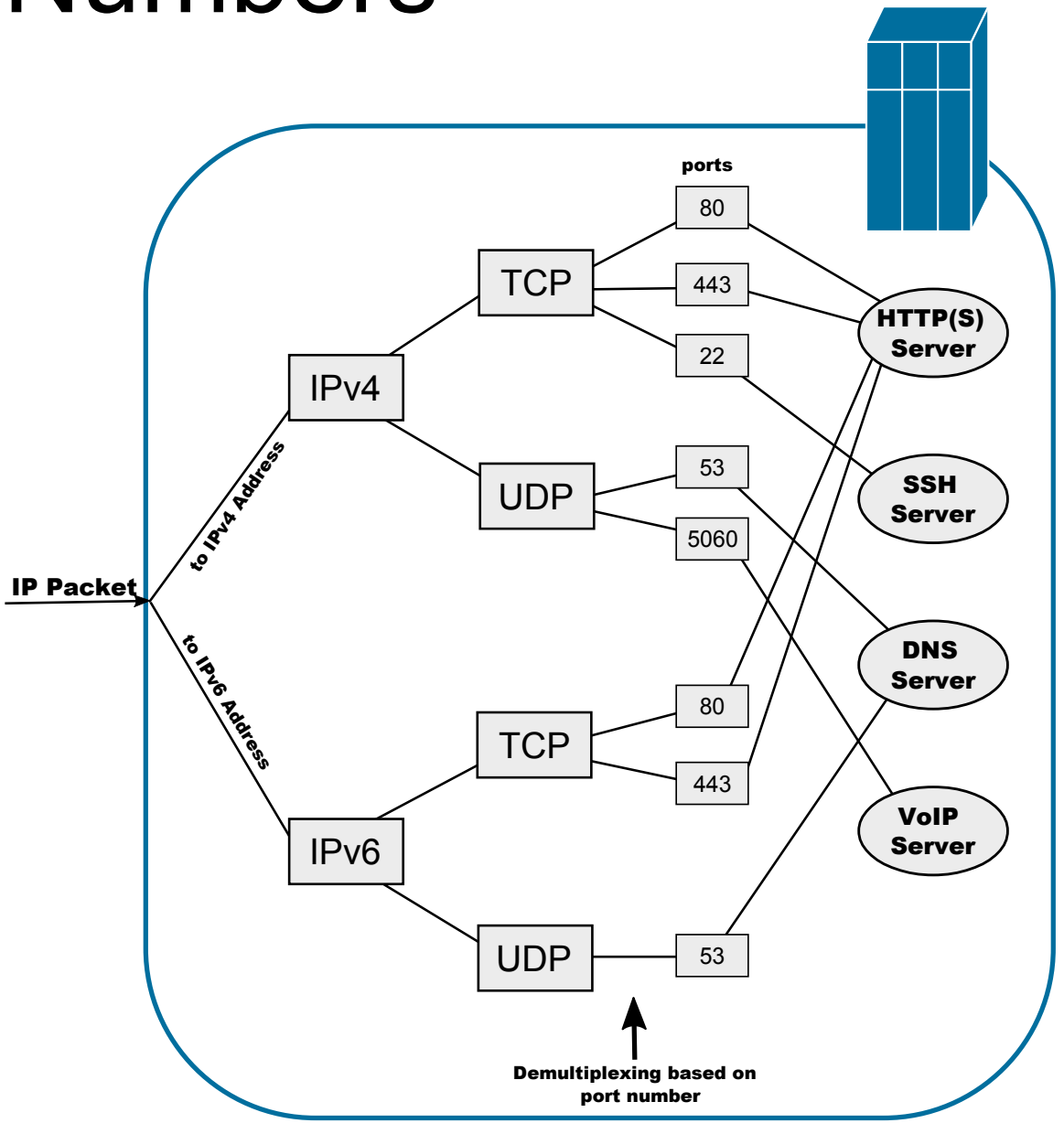
universidade de aveiro
deti.ua.pt

TCP/IP Reference Models



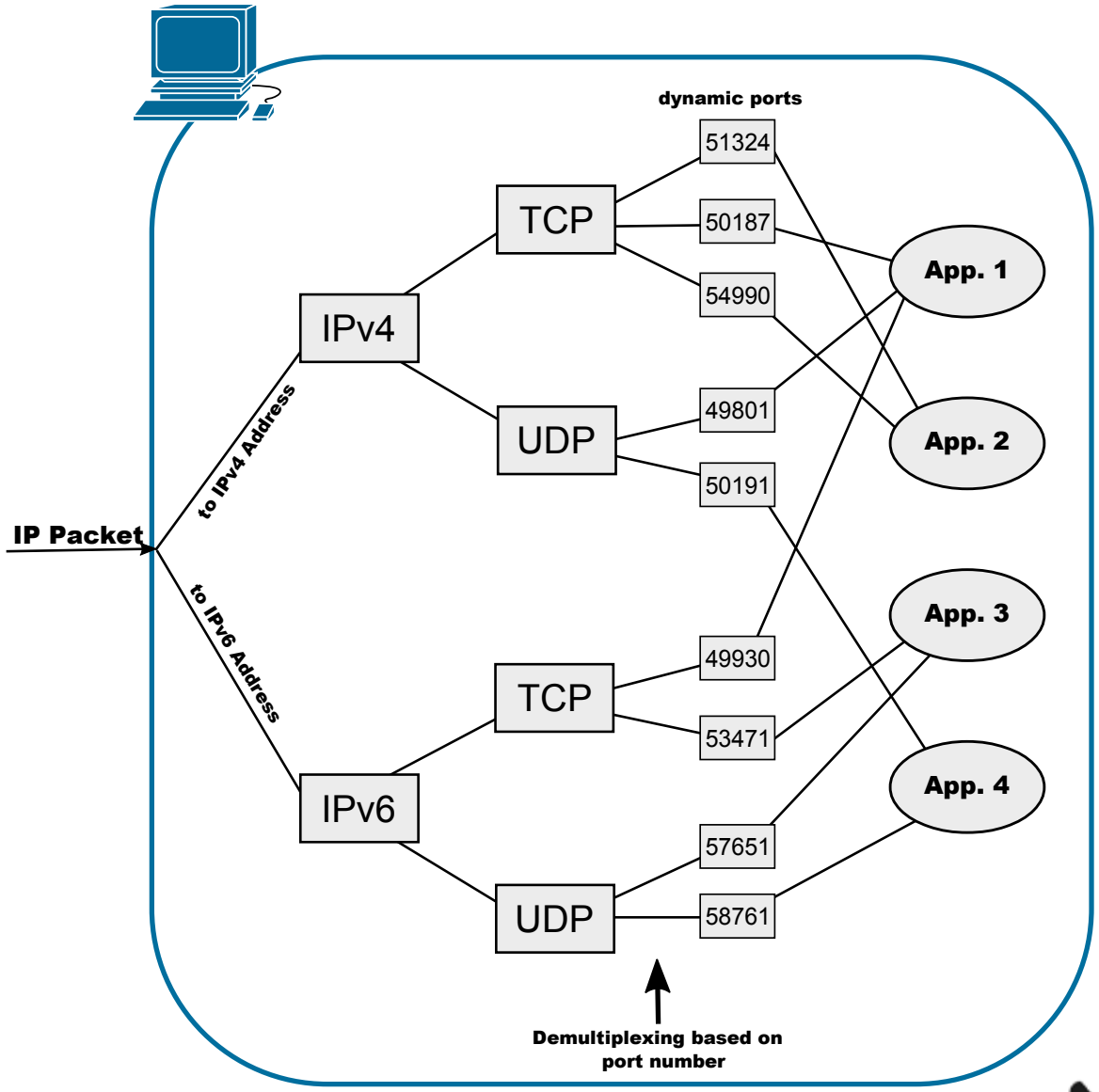
Port Numbers

- The packet destination IP address identifies a target host on the network.
- The transport protocol identifies the type of communication.
- The port number identifies the application running in the target host.
  - Chosen by the application/service.
    - Non-dynamic ports.
  - Each application/service may use more than one port.
  - The OS assures the assignment of different port numbers to each application.



Ephemeral/Dynamic Ports

- Ephemeral/Dynamic ports are used to identify a client application in a client-server communication.
- The Internet Assigned Numbers Authority (IANA) suggests the range 49152 to 65535.
- Randomly assigned by OS.



Well Known Port Numbers

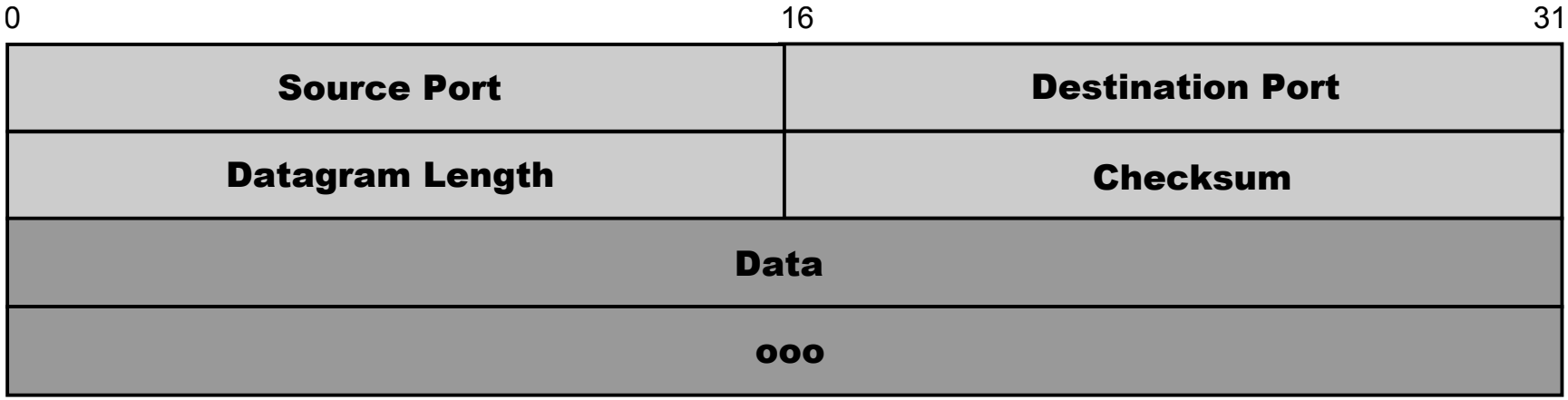
Decimal	Keyword	Protocol	Description
20	FTP-DATA	TCP	File Transfer Protocol (dados)
21	FTP-CONTROL	TCP	File Transfer Protocol (controlo)
22	SSH	TCP	Secure Shell (SSH) service
25	SMTP	TCP	Simple Mail Transport Protocol
67,68	BOOTP	UDP	Bootstrap Protocol (DHCP)
53	DNS	UDP/TCP	Domain Name System
69	TFTP	UDP	Trivial File Transfer Protocol
80	HTTP	TCP	Hypertext Transfer Protocol

- Many Internet IP services were the subject of study by IETF that proposed adequate support protocols.
- For these services, IETF together with the protocol specification proposed also a number (or numbers) to be used by that service at the server side.
- E.g., for protocol HTTP IETF recommends the usage of port 80. Therefore, all Web Browsers use port 80 as default for HTTP accesses.

User Datagram Protocol (UDP)

- Provides a data transport service with the performance characteristics offered by the IP network.
- Provides exchange of data between individual applications, and not only between hosts, with the introduction of a port identifier field.
- Does not provide any mechanism to recover from lost messages.
- Does not provide any mechanism to order received data.
  - Must rely on information at the application level.
- Allows to send data to multiple destinations simultaneously (point-to-multipoint communications).

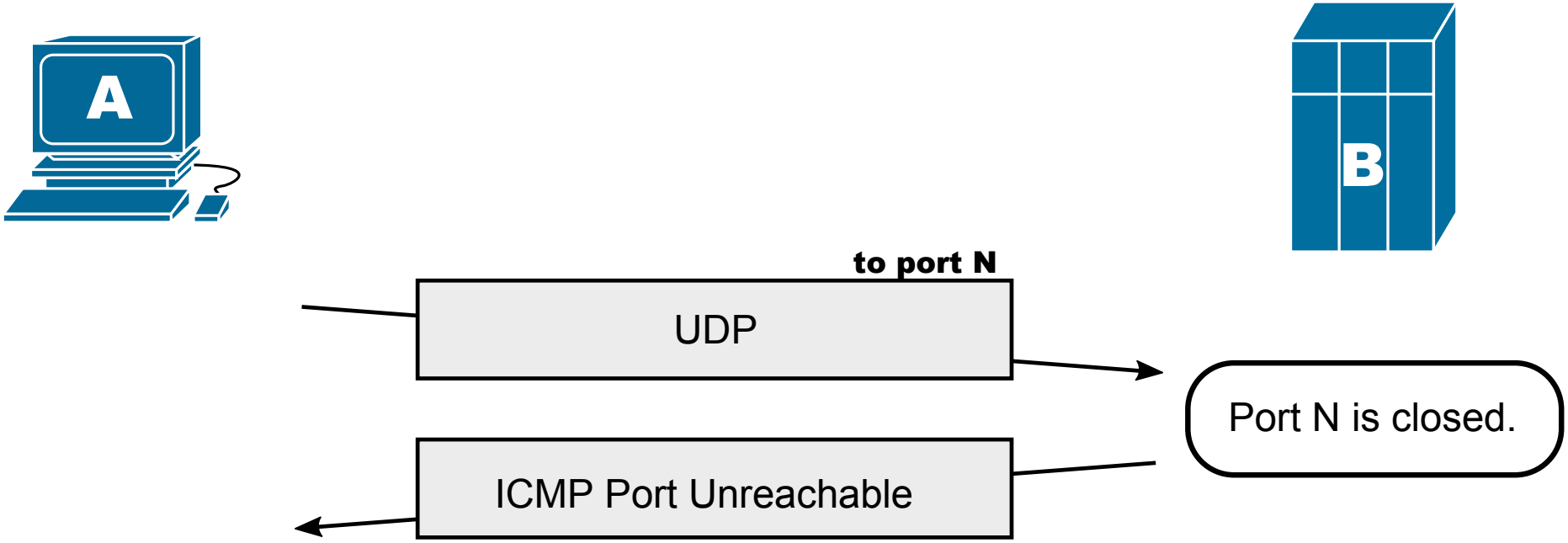
UDP Datagram (Header+Data)



- Source Port (2 bytes): defines the port number assign/chosen by the sender application. This field is optional, if not used should be field with zeros.
- Destination Port (2 bytes): defines the port number assign/chosen by the receiver application.
- Datagram Length (2 bytes): defines the size,in bytes, of the datagram (header+data).
- Checksum (2 bytes): used for data error detection and validation at the end-points. This field is optional, if not used should be field with zeros.
  - The checksum us calculated based on the UDP datagram and a pseudoheader IP (IP protocol identifier, source and destination IP addresses, and length of the IP datagram).
  - Can be used to verify if the end-points are the correct ones.

UDP Closed Port

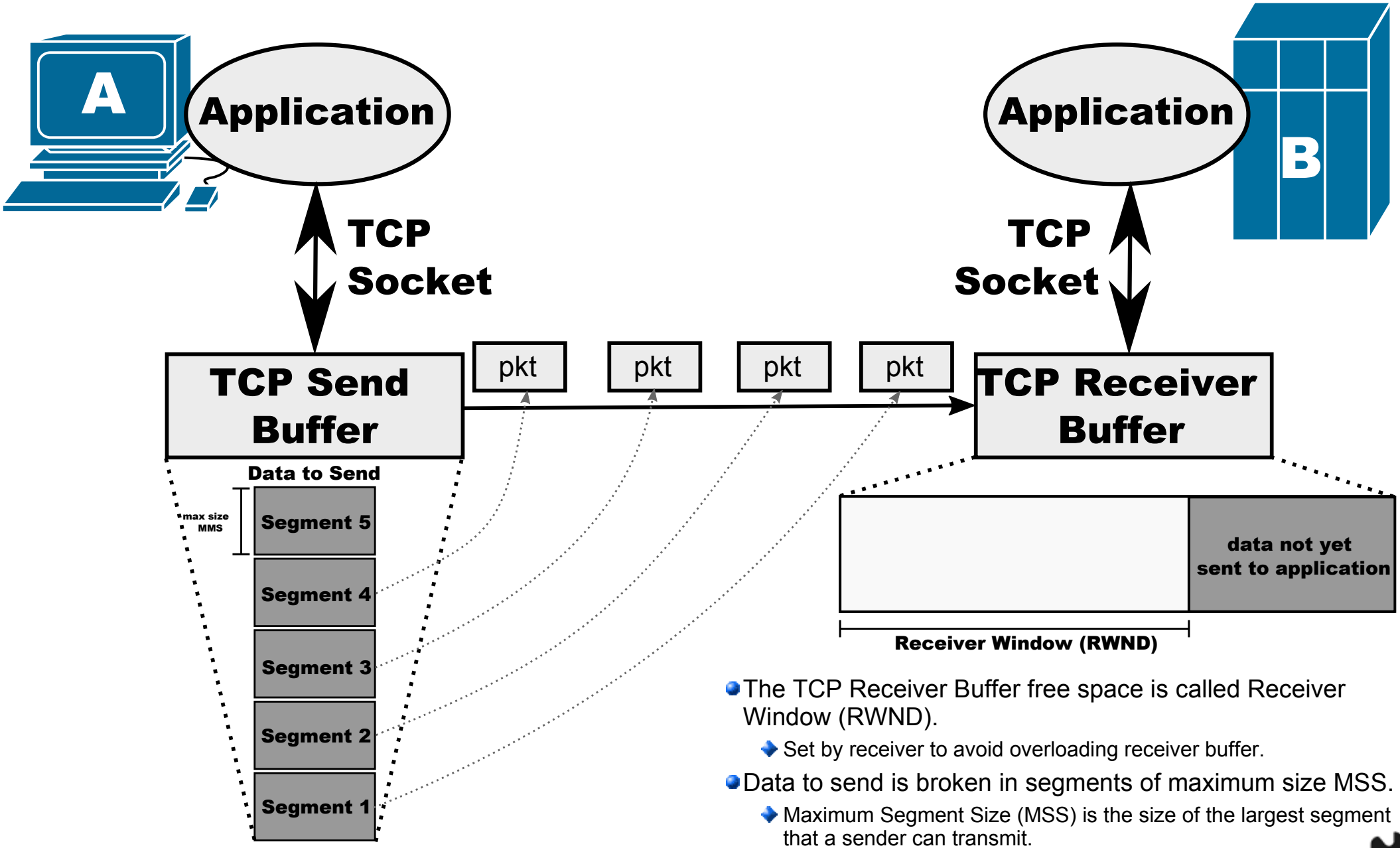
- When an UDP packet arrives to a host, but the UDP port is not open (no application listening):
  - The host responds with a packet ICMP *port unreachable*.



Transmission Control Protocol (TCP)

- Provides a reliable data transport service.
  - Data is received by the destination application without any losses and in order.
- Is connection-oriented protocol.
  - End-points establish a logical channel, to which are assigned applications' identifiers and the memory resources required to have a reliable transmission of data.
  - This connection is called Session.
- It is bi-directional.
  - Both end-points can send and receive data using the same logical channel.
- Traditional TCP supports only point-to-point connections.
  - See: Multipath TCP on last slide.
- Provides mechanisms to establish and terminate the connection.
- Network congestion and/or temporary lack of connectivity result in variable delays and consequent losses of packets (at transit or by timeout).
  - TCP includes algorithm that allows to efficiently react in this scenario.

TCP Buffers and Receiver Window



- The TCP Receiver Buffer free space is called Receiver Window (RWND).
  - Set by receiver to avoid overloading receiver buffer.
- Data to send is broken in segments of maximum size MSS.
  - Maximum Segment Size (MSS) is the size of the largest segment that a sender can transmit.
  - Defined by local configuration.

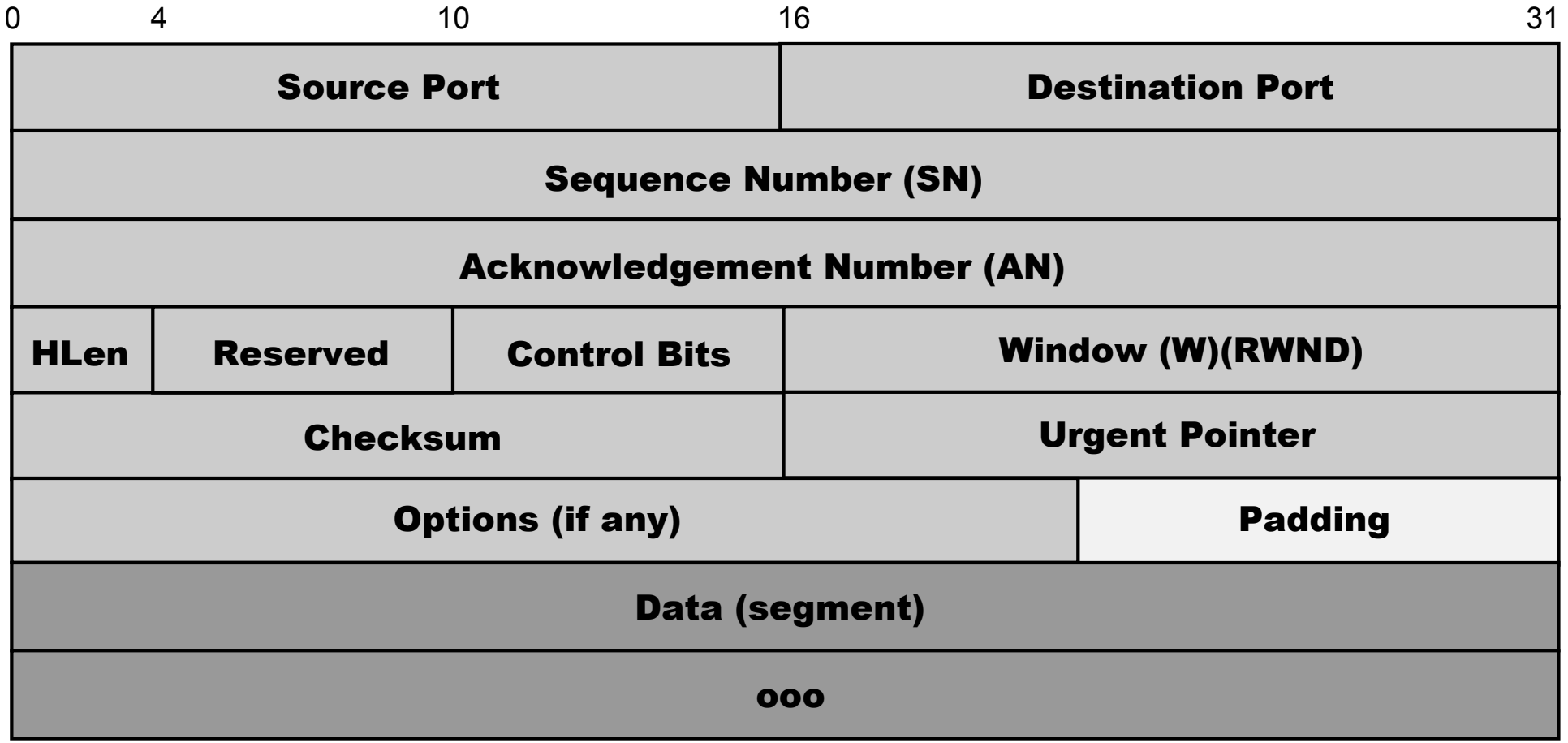
TCP: stream de dados, sequencia de bytes, tem que haver buffers: no fundo é um espaco de memoria que vai entrando coisas e saindo coisas, por ordem. fila de espera.
todo este controlo é feito pelo sistema operativo ou entre o driver e o sistema operativo.
temos que definir qual o tamanho do buffer em cada um dos dados.

o tamanho de buffer de envio nao é muito relevante mas o de envio sim. se nao tiver espaco no recepcao, ele avisa que nao tem mais espaco, para nao mandar, mais.

MSS: apesar de ser um stream de dados, ele vai ser partido aos bocados, ele vai recebendo dados, pega num maximo de 1000 bytes e manda os, por exemplo.

a particao de dados é feito a nivel da aplicacao.. i think

TCP Packet Header (1)



o porto de destino no cabecalho de TCP é diferente do cabecalho de destino do UDP-

Control bits:
URG: carregar os dados primeiro apontados pelo ponteiro na posicao de memoria.
PSH: para os segmentos de dados com o PUsh ativo, esses sao os primeiros a serem enviados… o proprio sistema operativo processa primeiro estes dados

SYN: bit que vai ativo no inicio para eles se sincronizar: estou neste porto TCP, vamos la sincronuzar.
FIN: fui a baixo, fecha a sessao TCP. se nao fecharmos, ao fechar a app, os protocolos sessoes de TCP.. podem estar a continuar a carregar, congestiona o tráfico tráfego e memoria. eles depois nao so criar uma interacao com o dispositivo que estavam a falar, mas tambem com a outra com que te vais ligar ,.

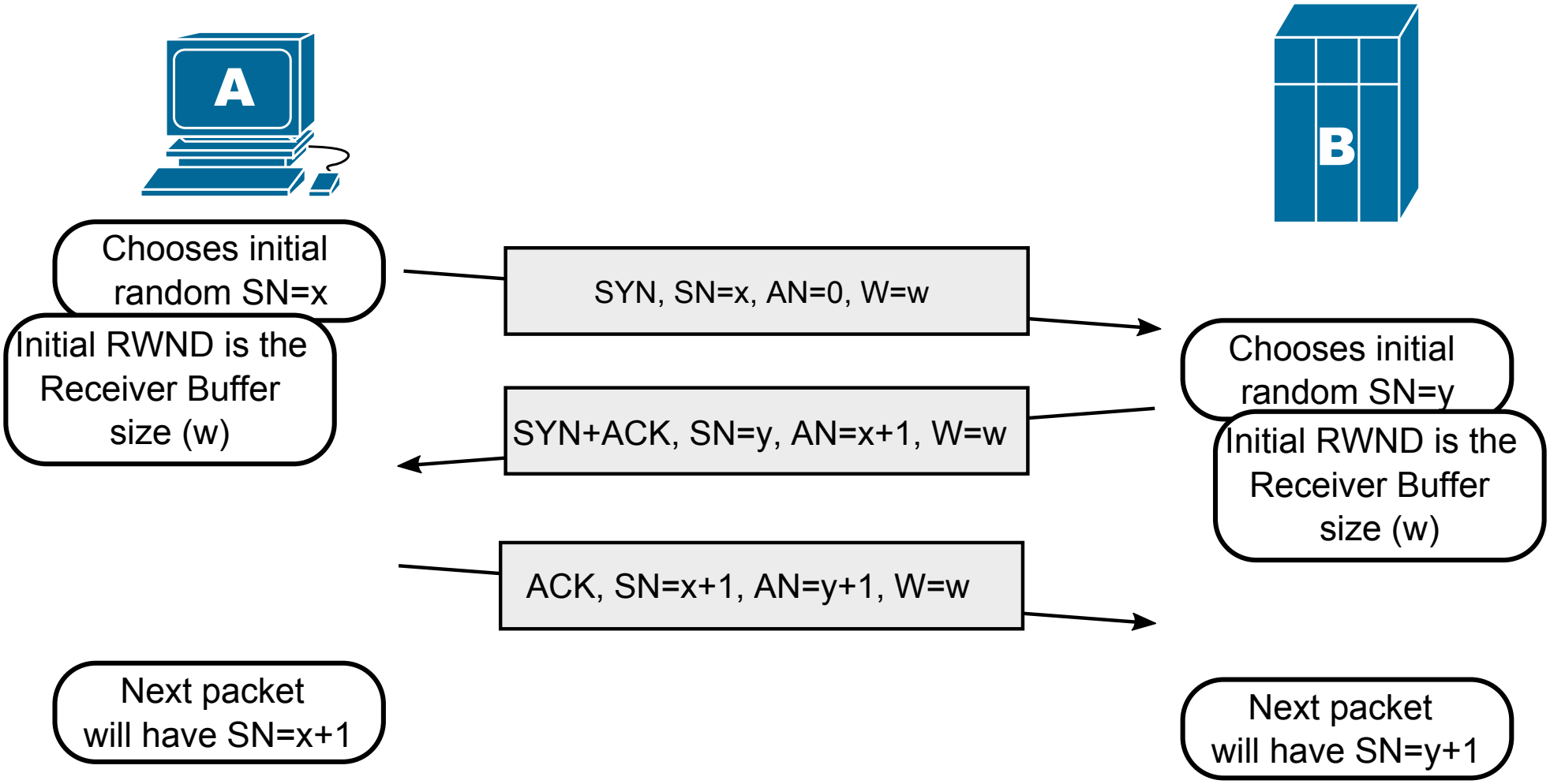
Reset RST: notifica o emissor e os do meio (firewalls e sistema de seguranca).

TCP Packet Header (2)

- The TCP Header has a variable length.
  - The field *Options* may contain additional fields.
- *Source Port* and *Destination Port* fields define the respective end-point ports.
- The *Sequence Number* defines the index of the first data byte in this segment.
- The *Acknowledge Number* defines the value of the next sequence number.
  - Acknowledges the good reception of data until byte with index (Acknowledge Number-1)
- *HLen* defines the size of the TCP header in bytes.
  - When *Options* are present, the field *Padding* is used to extend the header size to a multiple of 32 bytes.
- *Window* field defined the number of data bytes the sender of this segment is willing to accept (Receiver Window – RWND).
- *Control Bits* field is a binary set of flags
  - URG: Urgent Pointer is a valid field.
  - ACK: Acknowledgement is a valid field.
  - PSH: Data requires Push (receiver should push immediately data from TCP buffer to application).
  - RST: Connection Reset.
  - SYN: Sincronize Sequence Number.
  - FIN : Source closing connection.
- *Urgent Pointer* field defines the portion of data that should be considered urgent.
  - Not used by modern protocols.
- *Checksum* is used to detect errors.

Establishement of a TCP Connection

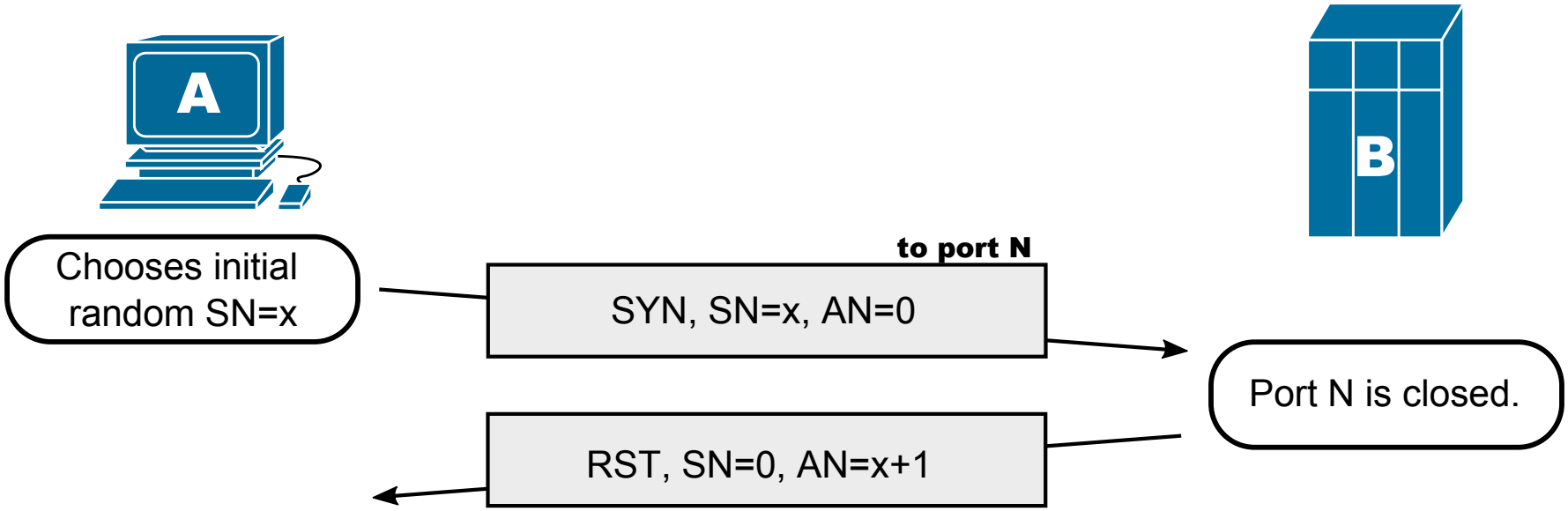
- 3-Way Handshake.
  - Synchronizes the both end-points initial *Sequence Numbers* (SYN), and acknowledges it (ACK flag).



envia um pacote SYN, escolhe um numero SN (random ou pseudo, random: antes era sequencial, mas por seguranca usa se random: havia intercepcao de sessoes de TCP, isto dava para saber qiual o SN que se iria usar aseguir.)

o B, ele vai responder com o SYN (os SYN sao indepentes nao sao iguais).

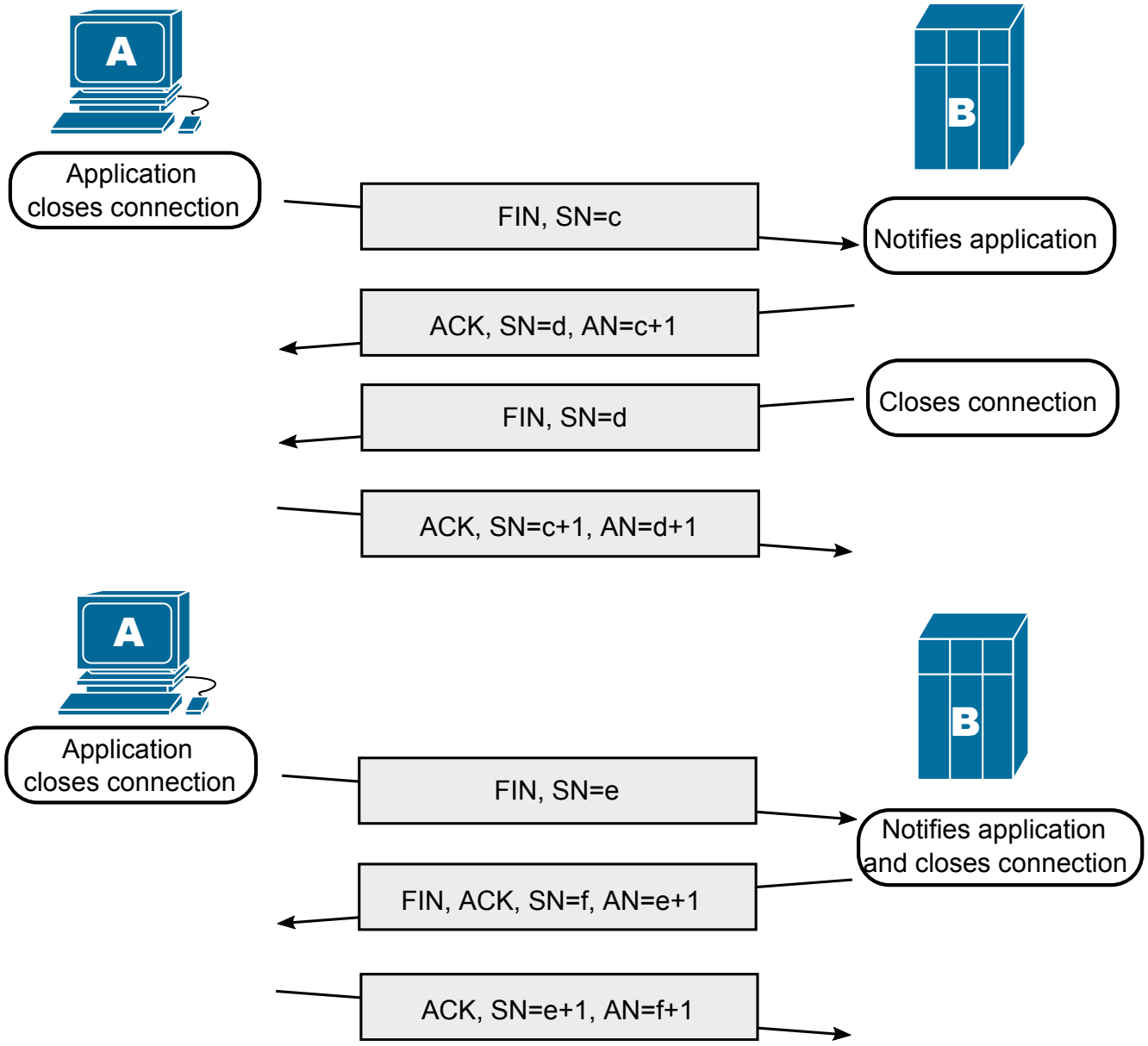
Establishement of a TCP Connection to a Closed Port



nao esta a sincronisar, entao vai a zero. SYN.

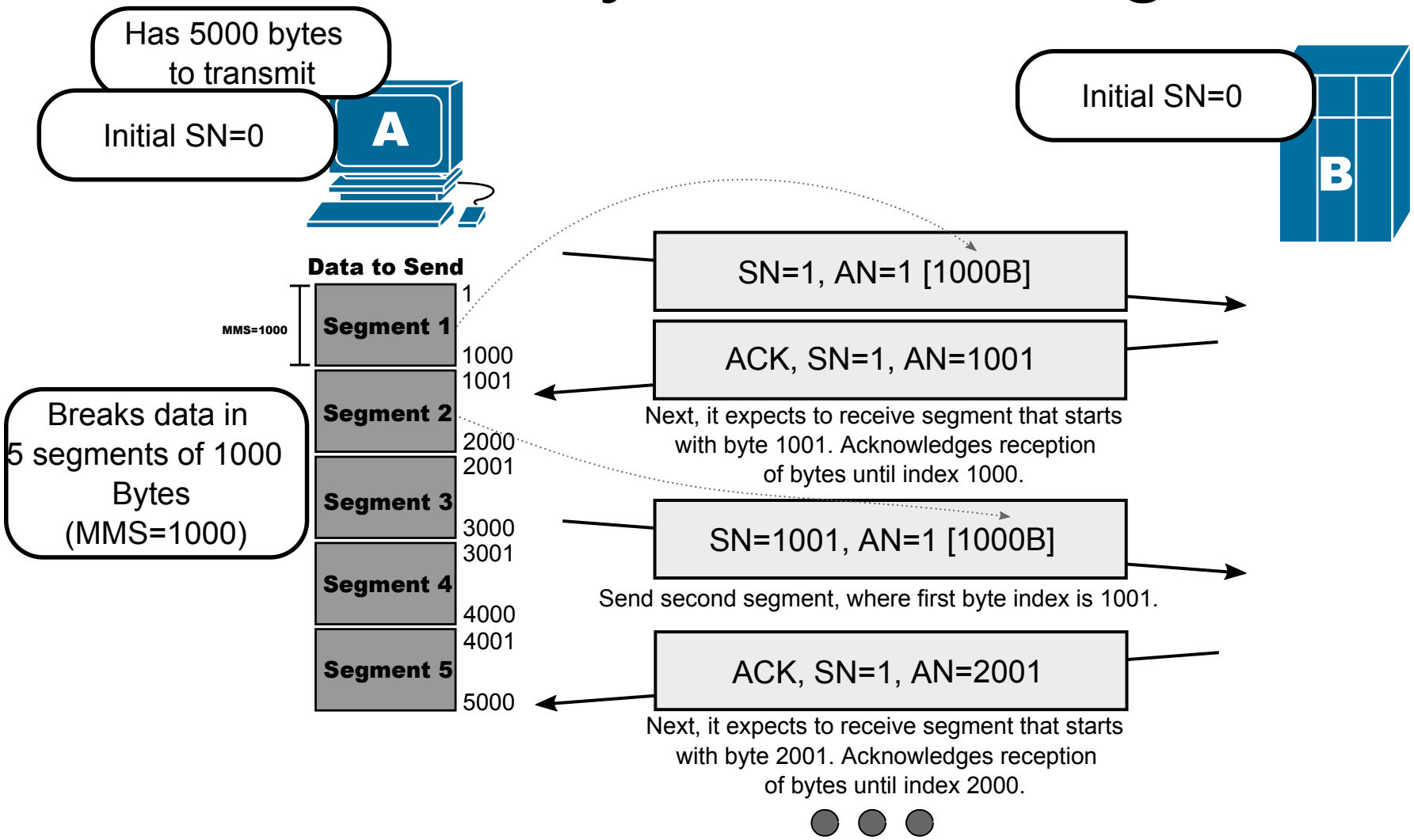
nao se usa mt isto porque da para opter informacoes, entao nao costumam, responder.

Closing a TCP Connection



Text

Data Delivery Acknowledgment



- Sender can (usually) send more than one packet before receiving the ACK.
  - Depends on the congestion window (next slide).
- Both end-points can send data.
  - A packet with a data segment, can acknowledge the reception of data a segment received from the other end-point.

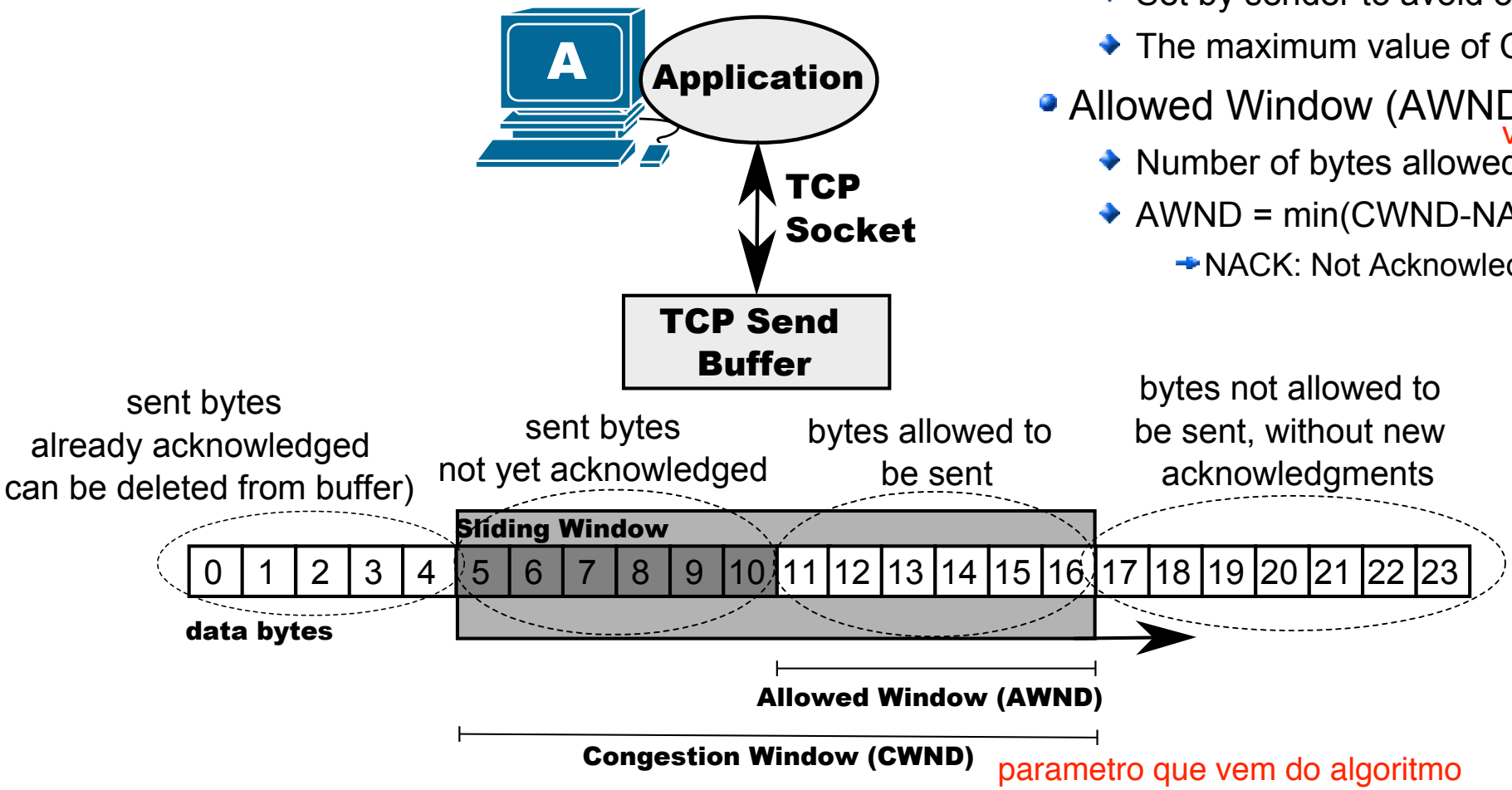
na verdade o SN nunca é zero.

TCP Congestion Control (1)

- Uses a sliding window to determine the number of packets/bytes the sender is allowed to transmit.

- Congestion Window (CWND)
  - Set by sender to avoid overloading network.
  - The maximum value of CWND is RWND.
- Allowed Window (AWND)
  - Number of bytes allowed to be sent.
  - AWND = min(CWND-NACK,RWND)
    - NACK: Not Acknowledged bytes

vem nos pacotes/



parametro que vem do algoritmo

TCP Congestion Control (2)

Has 4000 bytes to transmit

Both have:
MSS=1000 bytes (by configuration)
RWND=2500 (Window value after establishment)
CWND=RWND

CWND=2500, NACK=0, AWND=2500

CWND=2500, NACK=1000, AWND=1500

CWND=2500, NACK=2000, AWND=500

CWND=2500, NACK=2500 AWND=0

CWND=2500, NACK=1500 AWND=0 (RWND=1500)

CWND=2500, NACK=500 AWND=0 (RWND=500)

CWND=2500, NACK=0 AWND=0 (RWND=0)

SN=x+1, AN=y+1 [1000B]

SN=x+1001, AN=y+1 [1000B]

SN=x+2001, AN=y+1 [500B]

ACK, SN=y+1, AN=y+1001, W=1500 [0B]

ACK, SN=y+1, AN=y+2001, W=500 [0B]

ACK, SN=y+1, AN=y+2501, W=0 [0B]

Transfers 2500B to application.

CWND=2500, NACK=0 AWND=2500 (RWND=2500)

CWND=2500, NACK=1000, AWND=1500

CWND=2500, NACK=1500, AWND=1000

ACK, SN=y+1, AN=y+2501, W=2500 [0B]

SN=x+2501, AN=y+1 [1000B]

SN=x+3501, AN=y+1 [500B]

ACK, SN=y+1, AN=y+3501, W=1500 [0B]

ACK, SN=y+1, AN=y+4001, W=500 [0B]

Janela de Congestionamento (CWND)
janela de recepcao RWND

numero de bits permitidos para mandar = min(CWND - menosuns bits,
RWND)

teste que saiu: tenho um bit em PT e quero para moscovo: é mais rapido mandar por radio que por fibra optica: fotao é mais lento.

TCP Congestion Control (3)

- A packet with a data segment is considered lost when:
  - Timeout: After some time no ACK is received for that data segment
  - With Fast Retransmit/Recovery: After 3 or more duplicate ACKs are received for the previous data segment.
- When a packet is lost, TCP automatically decreases transmission rate to adapt to network conditions.
  - i.e., Decreases CWND size.
- When data delivery is acknowledged, TCP increases transmission rate.
  - i.e., Increases CWND size.
- The way the CWND size varies depend on the TCP Algorithms used:
  - Tahoe (Original TCP, 1988) uses:
    - Fast Retransmit, Slow Start, and Congestion Avoidance.
  - Reno (1990) uses:
    - Uses Fast Recovery, and Congestion Avoidance.
- At the beginning, the initial CWND value is usually 2, 3, 4, or 10 MSS and then the terminal starts the *Slow Start* with SSThresh=RWND.

o TCP vai adaptar se se os pacotes chegaram ou nao com o ACK.

pode ser perdido os dados ou o ACK.

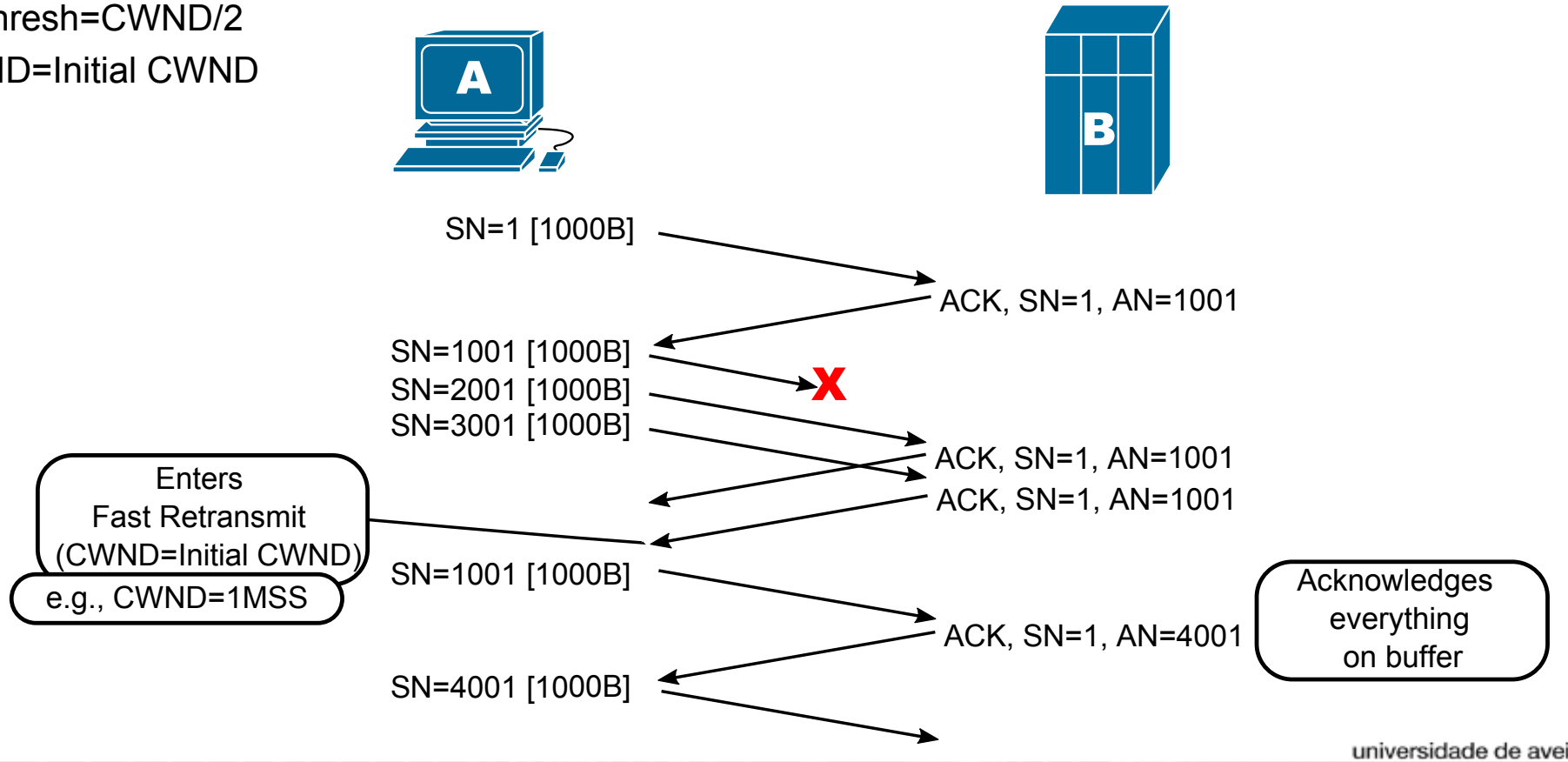
Nem todos os sistemas operativos utilizam o mesmo algoritmo de TCP. quando se tem uma aplicacao que use TCP, podemos mudar o algoritmo do TCP de acordo com essa app para ser mais eficiente.

ele *o algoritmo* vai alterando o numero de pacotes que envia: envia 5 e foi sucedido, entao envia 10 (slow start)

internets de hoje em dia com muitas gigas, requerem ou podem requerer parametros mais altos.

TPC Fast Retransmit

- A segment is considered lost if 3 or more duplicate ACKs are received for the previous data segment.
  - Faster detection than waiting for a timeout.
  - Requires receiver to work.
- TCP retransmits immediately the lost segment.
- The TCP algorithm enters Slow-Start, with:
  - SSThresh=CWND/2
  - CWND=Initial CWND



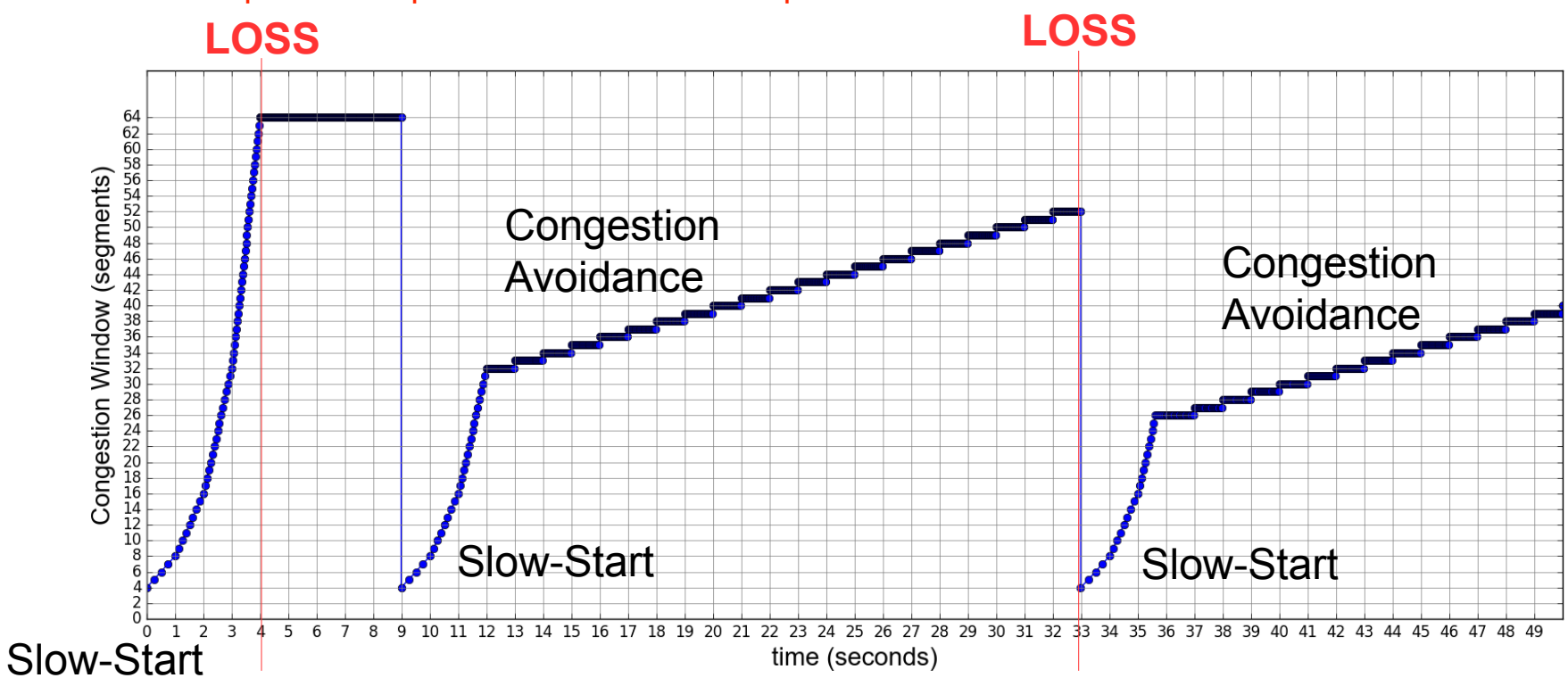
TCP Slow-Start

- At the beginning, the initial CWND value is usually 2, 3, 4, or 10 MSS and the terminal starts the *Slow Start* with SSThresh=RWND.
- CWND size grows very fast while smaller than the predefined threshold (CWND<SSThresh).
  - When a ACK arrives the CWND is updated: CWND=CWND+N,
    - N is the number of bytes acknowledged in the ACK.
    - Results that the window size (approximately) doubles each round-trip time.
      - Exponential growth.
    - Continues until a loss occurs or CWND reaches RWND.
- When a loss occurs, SSThresh is defined as CWND/2 and Slow-Start begins again from its initial CWND.
- Once the CWND reaches the SSThresh, it changes to congestion avoidance algorithm.
  - Linear growth.

TCP Congestion Avoidance

- When a ACK arrives the CWND is updated: CWND=CWND+N/CWND, linear
  - This results in a linear increase of the CWND.

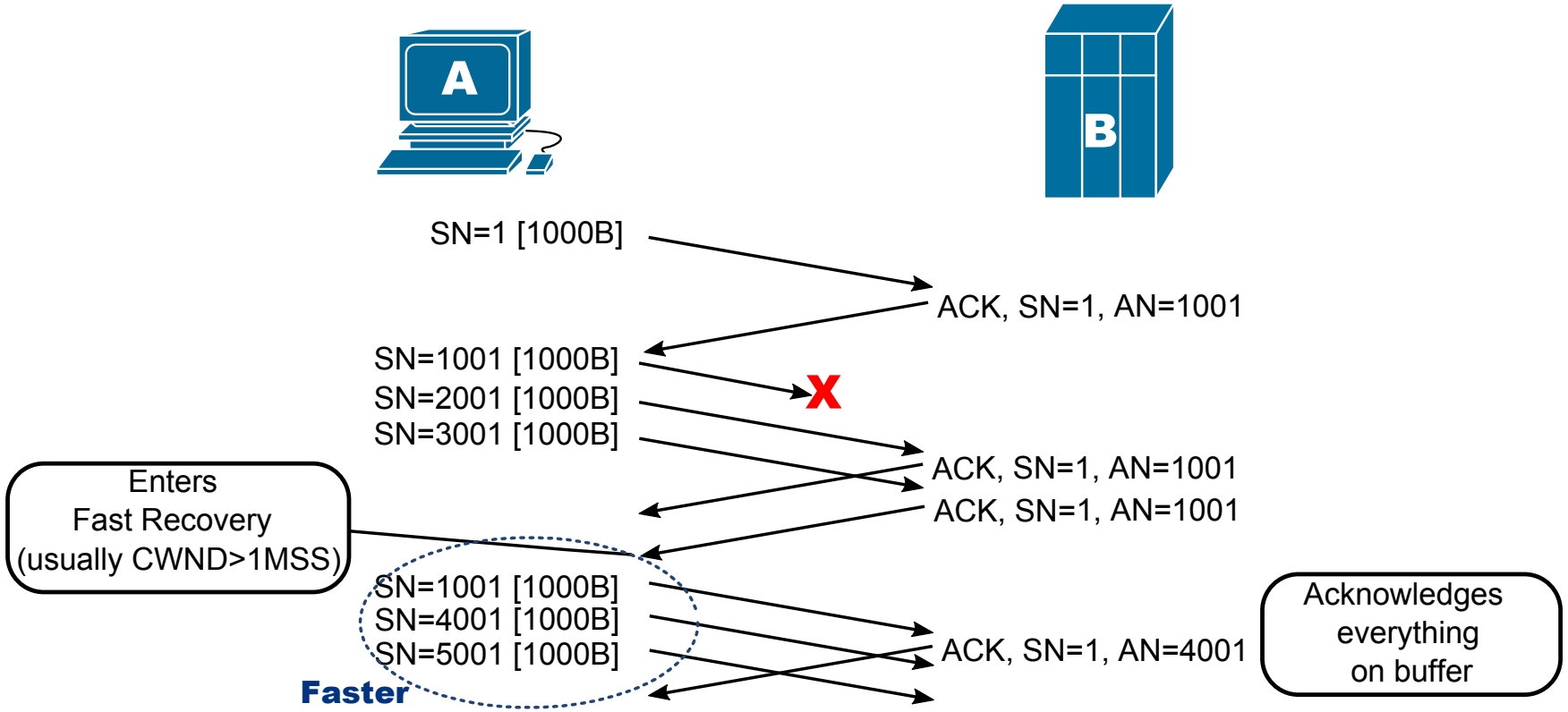
se perder um pacote so vai crescer exponencialmente ate meio



- RWND is 64, and initial CWND is 4 MSS (segments).
- Initial SSThresh=RWND=64. After first loss: SSThresh=CWND/2=32. After second loss: SSThresh=CWND/2=26.

TPC Fast Recovery

- The same as TPC Fast Retransmit.
- However when a loss occurs enters directly to Congestion Avoidance with:
  - SSThresh=CWND/2
  - CWND=SSThresh
    - Some implementation have: CWND=SSThresh+3*MSS.

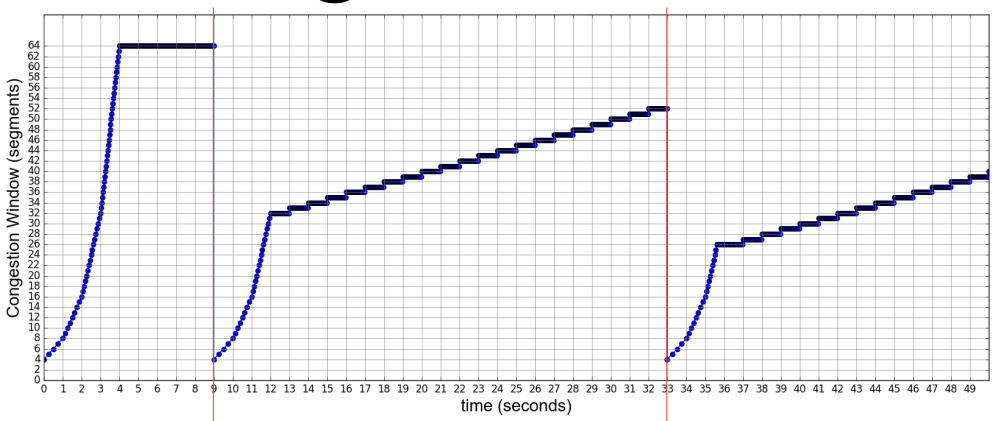


Other TCP Algorithms

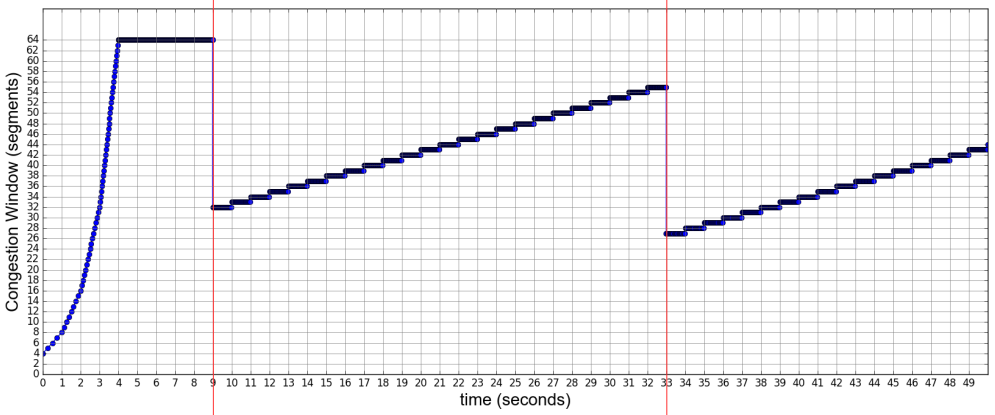
- NewReno (1996)
  - Allows for partial ACK.
  - When a loss occurs, CWND is defined as β*CWND, with β=0.5. When a ACK arrives, CWND is updated as CWND=CWND+α, with α=1 MMS.
  - Used by default in Windows and supported by Mac OS X.
    - Used in Windows XP and earlier.
    - After Windows Vista, Compound TCP can also be enabled.
- CUBIC (2005) expressao geometrica que cresce mais rapido que a exponencial (x^3)
  - Uses a cubic function to control the CWND.
  - Used by Linux (kernel 2.6.19 and later) and supported by Mac OS X.
- Compound TCP (2006)
  - Adapts its behavior by use of a scalable delay-based component. T
    - Increases throughput more quickly in the congestion avoidance phase.
  - The AWND depend on the RTT measurements from successfully acknowledged packets.
  - Windows OS supports it as an option.
- Low Extra Delay Background Transport (LEDBAT)
  - Delay-based congestion control algorithm that uses all the available bandwidth while limiting the increase in delay. Measures one-way delay.
  - Supported by Windows 10 and latest versions of Mac OS X.

TCP Algorithms Comparison

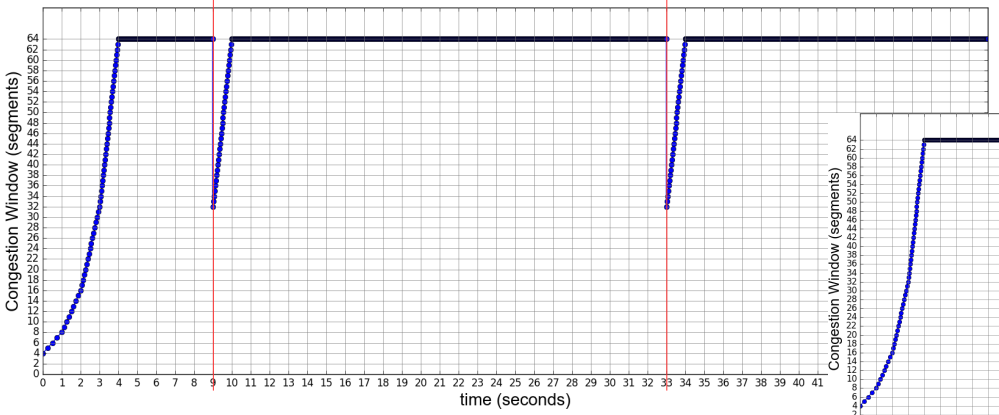
- Tahoe



- Reno

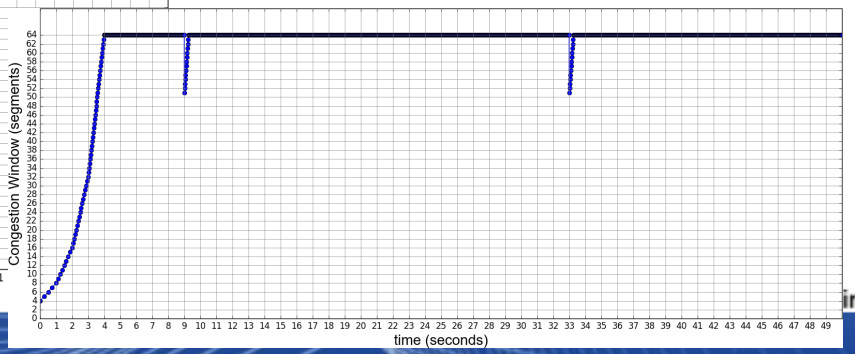


- NewReno



- Behavior and performance depends greatly on traffic and losses patterns, and end-points' behaviors
- In these examples:
  - Using a traffic pattern that sends a number of packets equal to the CWND, in one second interval, equally spaced over the interval.
  - Last packet of interval 8s-9s and 32s-33s are lost.
  - ACWD is 64, and initial CWND is 4 MSS (segments).
  - Receiver sends one ACK per packet.
  - Each dot represents a received ACK.
  - Tahoe sends 1585 packets, Reno 2017 packets and NewReno 2938 packets.

- Cubic



Multipath TCP (MPTCP)

- TCP is essentially a single-path protocol.
  - When a TCP connection is established, the transmission is bound to the IP addresses of the two end-points.
  - If one address changes the TCP session will fail.
  - TCP can not load balance segments using more than um TCP session.
    - This load balancing must be done at the application level.
- Multipath TCP allows multiple subflows within a single MPTCP session.
  - A MPTCP session starts with an initial subflow, using the traditional 3-Way Handshake.
  - After the first MPTCP subflow is established, additional subflows can also established similar to the tradional TCP 3-Way Handshake. However, but rather than being a separate session, all subflows are bounded to the same MPTCP session.
  - Data can then be sent over any of the active subflows, using joint *Sequence* and *Acknowledgment Numbers.*
- Apple's Siri application uses Multipath TCP.