Networking Basics

04b - Transmission Control Protocol (TCP)

Wolfgang Tremmel
academy@de-cix.net

DE CIX

Where networks meet

Networking Basics

DE-CIX Academy

Internet Model

IP / Internet Layer

- Data units are called "Packets"
- Provides source to destination transport
  - For this we need addresses
- Examples:
  - IPv4
  - IPv6

Layer	Name
5	Application
4	Transport
3	Internet
2	Link
1	Physical

Internet Model

Transport Layer

- *May* provide flow control, reliability, congestion avoidance
- Also may contain information about the next layer up
- Examples:
  - UDP (none of the above)
  - TCP (flow control, reliability, congestion avoidance)

Layer	Name
5	Application
4	Transport
3	Internet
2	Link
1	Physical

DE CIX

TCP - Transmission Control Protocol

What does it do?

- Transports data - any kind of data
- Makes sure everything arrives at destination unchanged
  - And lets the sender know that it has arrived
- Takes care of speed of delivery - by adjusting sending rate
- TCP is complicated
  - ~~Oversimplify?~~ Or leave stuff out?

DE CIX

TCP - Transmission Control Protocol

Transport data: How does it do that?

- Establishing connections
- Not one - but two connections
  - Sender -> Receiver **and** Receiver -> Sender
- Each data received is **acknowledged** to the other side
  - So each side knows what the other side has already received
  - If anything gets lost, it is retransmitted (but this costs time!)

Encapsulation

Packets inside packets - headers after headers

- Encapsulation is like Russian dolls
- IP Packets have a payload
- This payload is usually UDP or TCP (there are others as well)
- So we have a TCP packet inside an IP packet

Attribution: Fanghong. derivative work: Greyhood
https://commons.wikimedia.org/wiki/File:Matryoshka_transparent.png

IPv4 Header

"Legacy" IP

- Starts with version and length
- Total length of packet
- Important: Time to live (TTL)
- **Protocol: Type of payload**
  - TCP = 6, UDP = 17
- Source / Destination address 32 bits
- Options (optional)

<table>
<tr><th>Byte</th><th>0</th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>0</td><td>Version | Header Length
always 4 | 5..15</td><td>DSCP / ECN</td><td colspan="2">Total Length
20..65535</td></tr>
<tr><td>4</td><td colspan="2">Identification</td><td colspan="2">Flags / Fragment Offset</td></tr>
<tr><td>8</td><td>Time To Live</td><td>Protocol</td><td colspan="2">Header Checksum</td></tr>
<tr><td>12</td><td colspan="4">Source IPv4 Address</td></tr>
<tr><td>16</td><td colspan="4">Destination IPv4 Address</td></tr>
<tr><td>20
24
28
32</td><td colspan="4">Optional (if HeaderLength > 5)</td></tr>
</table>

IPv6 Header

Modern IP

- Starts with version and some labels
- Payload length in bytes (0-65535)
- **Next Header** - you can chain more headers
  - replaces protocol field, same values
  - so this now points to the TCP header

Byte	0	1	2	3
0	Version = 6 / Traffic Class / Flow Label			
4	Payload Length in bytes		Next Header	Hop Limit
8	Source IPv6 Address			
12				
16				
20				
24	Destination IPv6 Address			
28				
32				
36				

Next header: Transport layer header

TCP, UDP, and more

- We already talked about UDP
- TCP is way more complex
- So, it is getting complicated
- Lets have a look at the header

Byte	0	1	2	3
0	Version = 6 / Traffic Class / Flow Label			
4	Payload Length in bytes		Next Header	Hop Limit
8	Source IPv6 Address			
12				
16				
20				
24	Destination IPv6 Address			
28				
32				
36				

TCP Header

- Source and destination port
  - 32 bit each, both mandatory
- Sequence number
  - Starts pseudo-random
- Acknowledge number
  - To tell the sender what is expected next
- Window size
  - Amount of data the sender can send without ACK from receiver

Byte	0	1	2	3
0	Source Port		Destination Port	
4	Sequence number			
	Acknowledge number			
	Data Offset (4 Bit) Reserved (3 Bit) Flags (9 Bit)		Window Size	
	Checksum		Urgent Pointer	

DE CIX

TCP Header

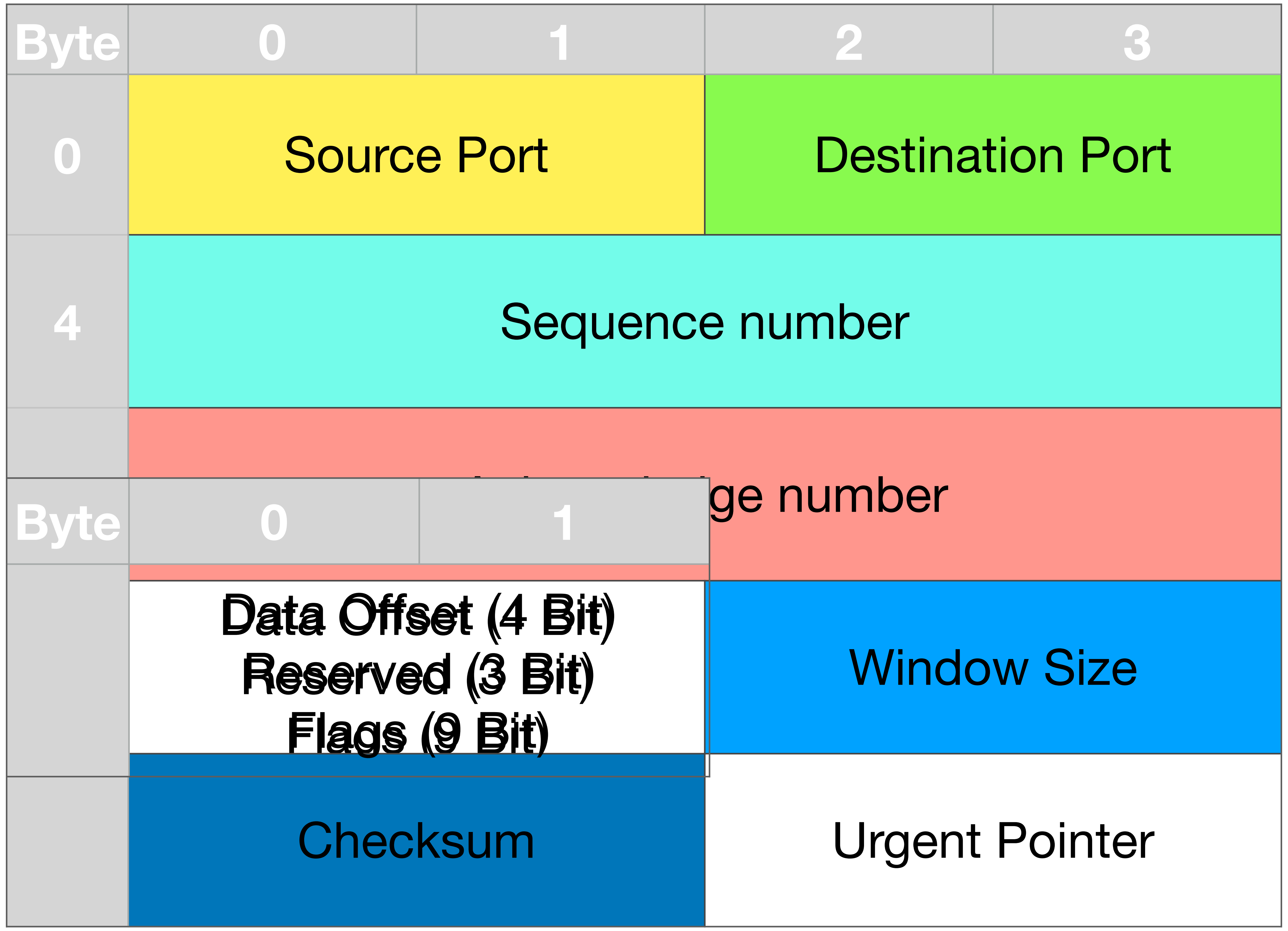
- Checksum
  - Parts of IP-header, TCP header, payload
- Urgent pointer
  - Mark part of the payload as urgent, not widely used.

Byte	0	1	2	3
0	Source Port		Destination Port	
4	Sequence number			
	Acknowledge number			
	Data Offset (4 Bit) Reserved (3 Bit) Flags (9 Bit)		Window Size	
	Checksum		Urgent Pointer	

DE CIX

TCP Header Flags

- 9 Flags in the header



TCP Header Flags - Important ones

- **SYN** - Synchronize sequence numbers. Only at start of connection
- **ACK** - content of *Acknowledgement Number* field is valid. Should be set in all packets after initial handshake
- **PSH** - Tells TCP to push buffered data up to Application Layer
- **FIN** - Last packet from sender when closing connection
- **RST** - immediate reset / shutdown of connection

Byte	0								1							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Data offset				Reserved 0 0 0			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN

TCP Header Flags - the rest

- Flags for Explicit Congestion Notification - **RFC3168**
  - **NS** - ECN-Nonce - **RFC3540**
  - **CWR** - Congestion Window Reduced
  - **ECE** - ECN-Echo
- **URG** - Urgent data pointer field is significant

Byte	0								1							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	Data offset				Reserved 0 0 0			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN

TCP Options

Up to 40 bytes of optional header

- Each option has three fields
  - Option kind
  - Option length
  - Option data
- Please read the documentation about optional TCP header fields

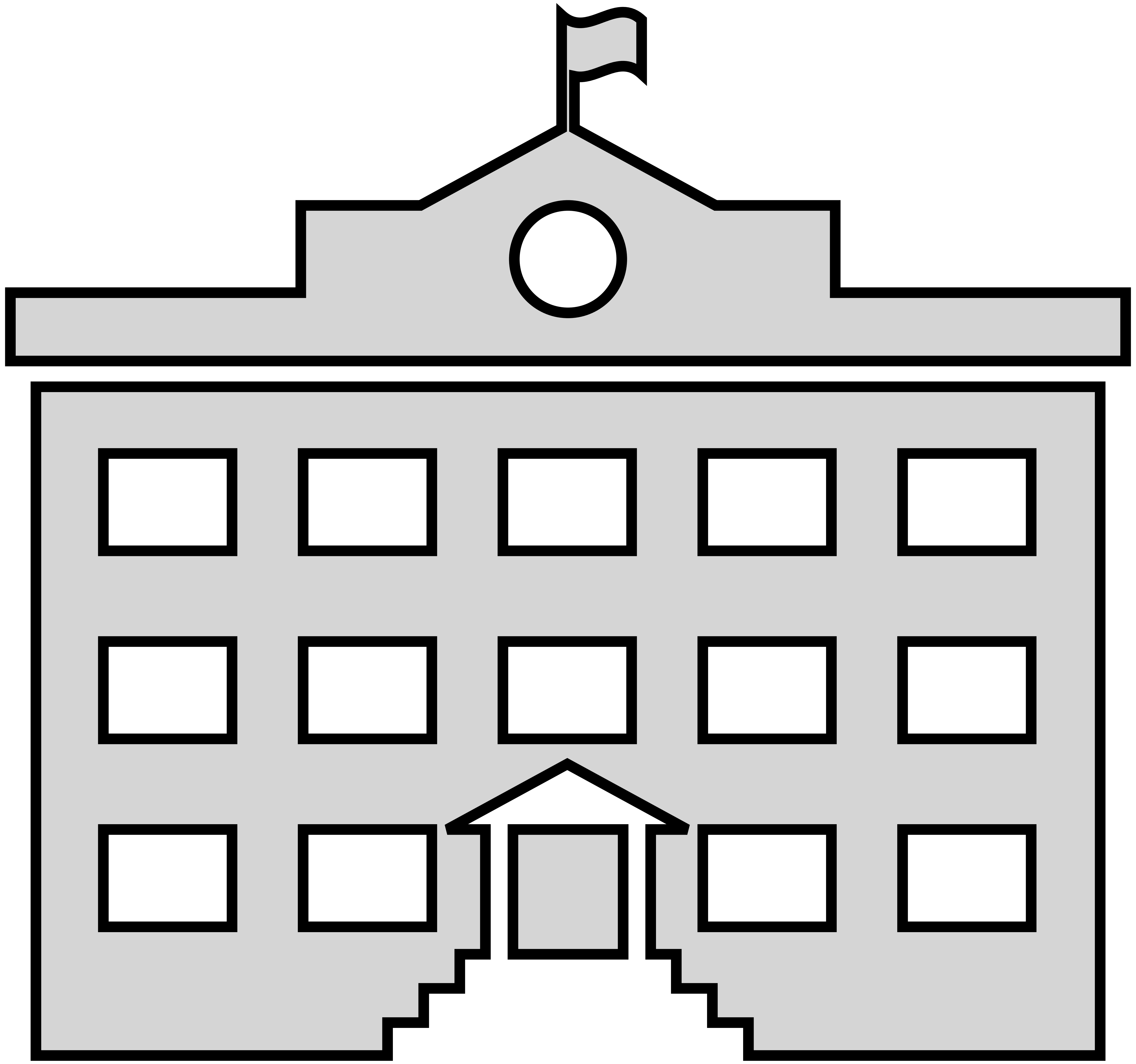
Features of TCP

DE CIX

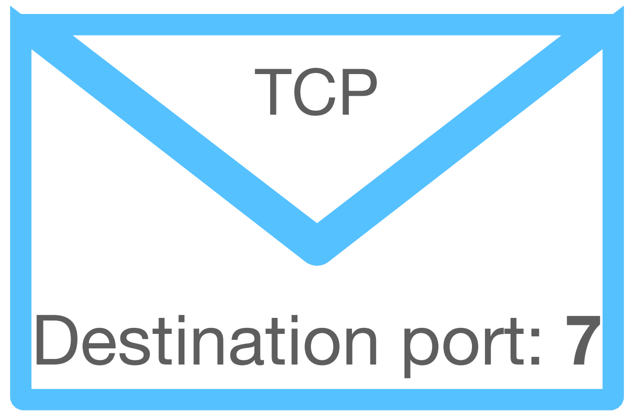
Features of TCP

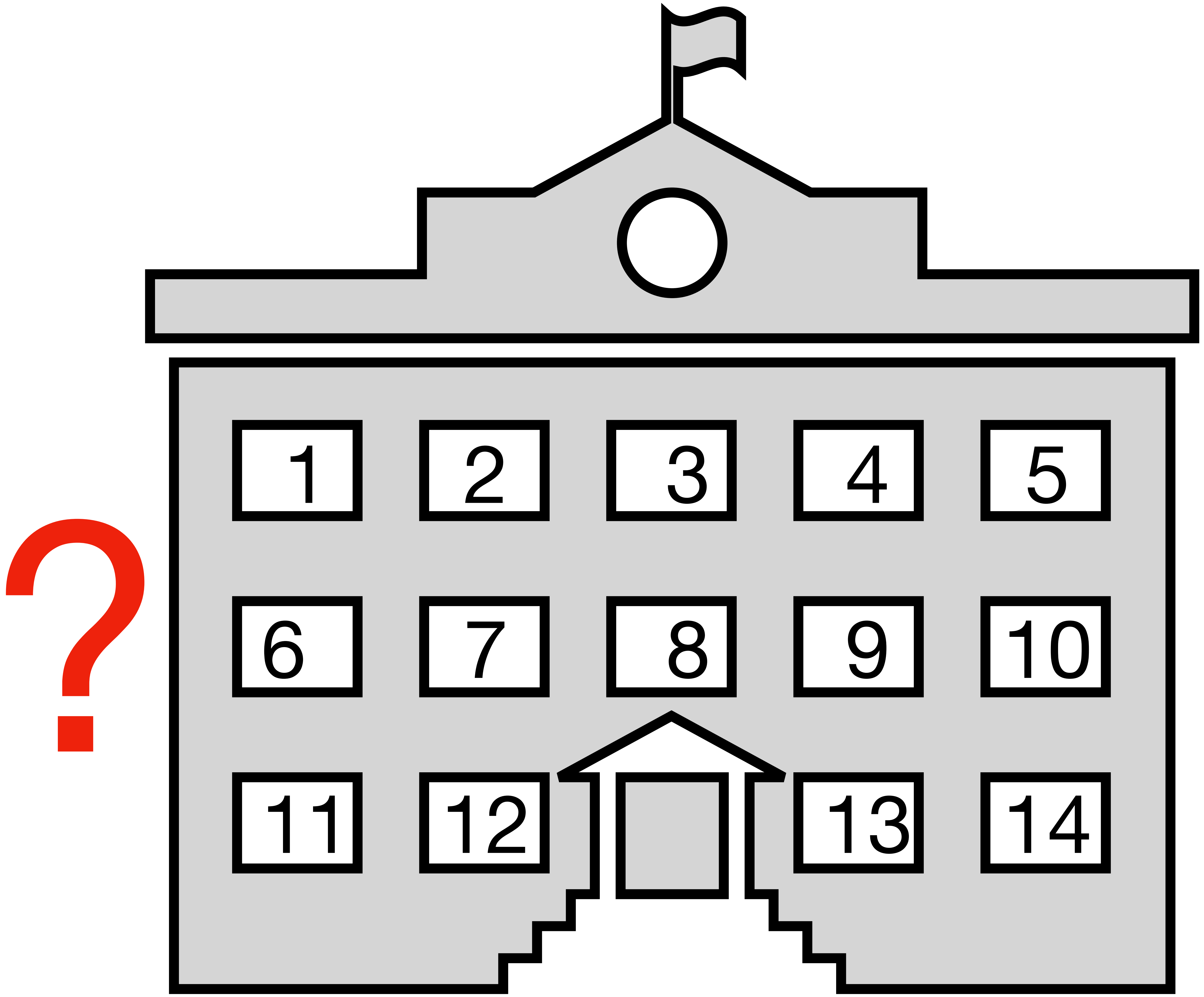
Do you remember this from UDP?

1. **Port numbers**
2. Connections
3. Reliability
4. Flow control
5. Congestion avoidance



Port numbers

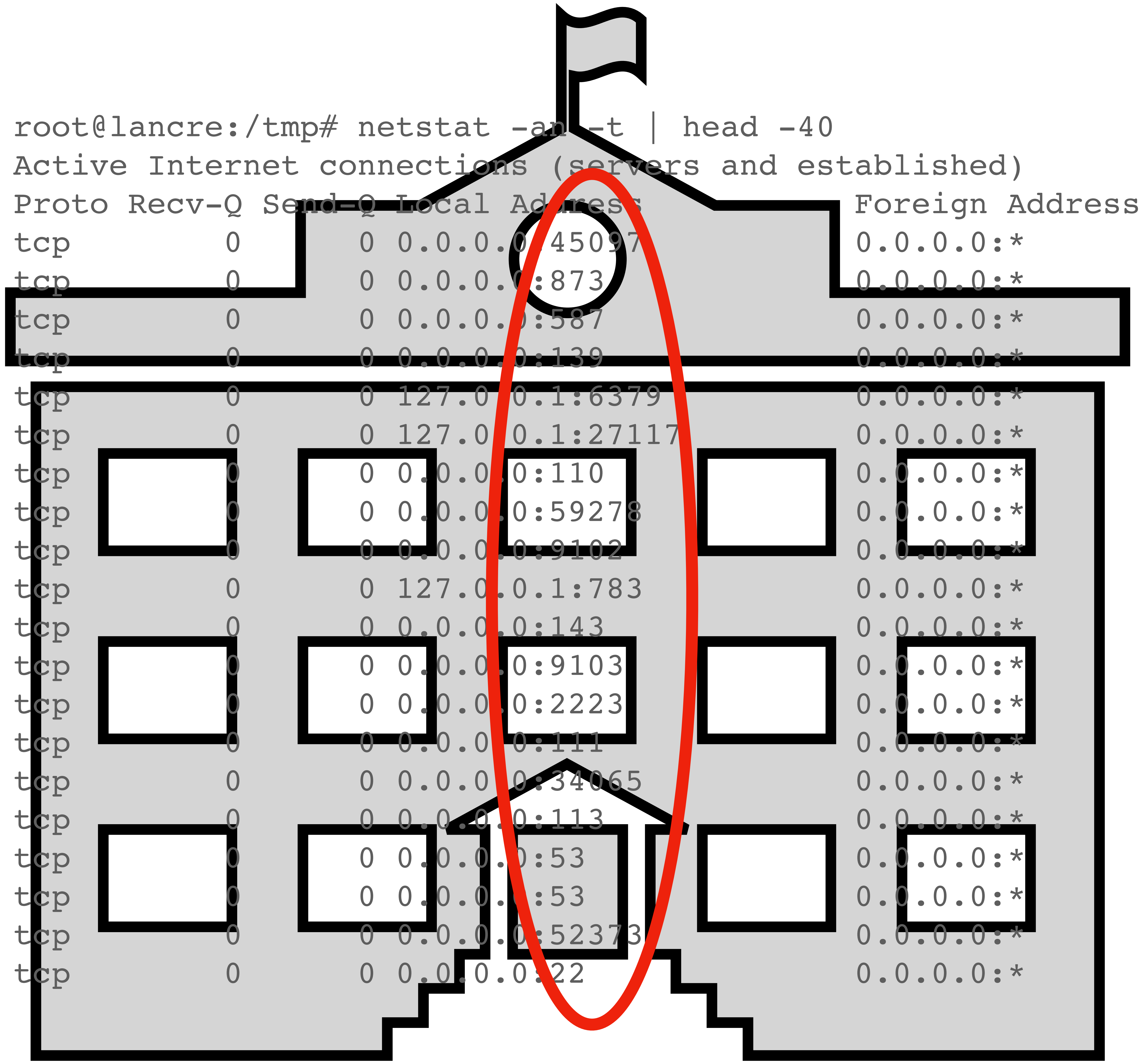




Port numbers

In reality...

- Of course we have not a building
- We have a computer system
- But we have port numbers
- Behind each port sits a piece of software
    - On some systems this software is called a "daemon"



Features of TCP

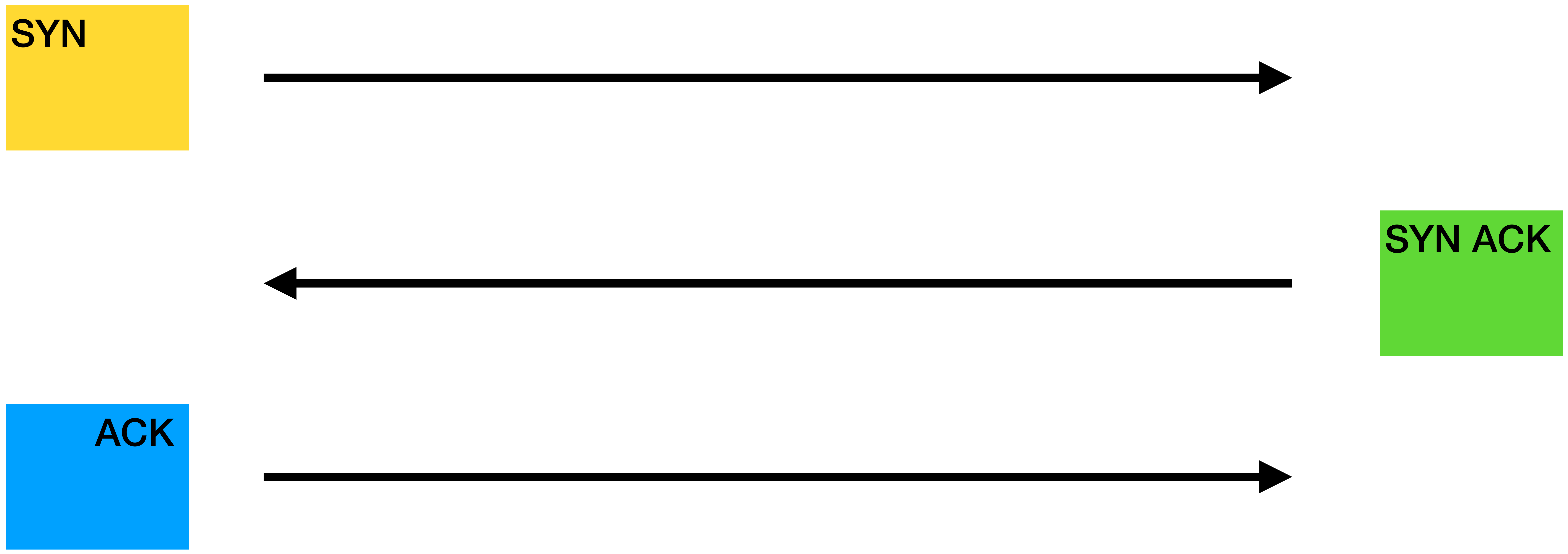
1. Port numbers
2. **Connections**
3. Reliability
4. Flow control
5. Congestion avoidance

Establishing Connections

Not so secret handshake

Establishing Connection

Not so secret handshake



Features of TCP

1. Port numbers
2. Connections
3. Reliability
4. Flow control
5. Congestion avoidance

Reliable Data Transfer

No need to wait for each packet

TCP

Data transfer

- Ordered transfer - receiver can re-arrange out-of-order packets
  - This is what the sequence number is for
- Retransmission of lost packets
  - This is what the acknowledgement number is for

TCP Data Transfer

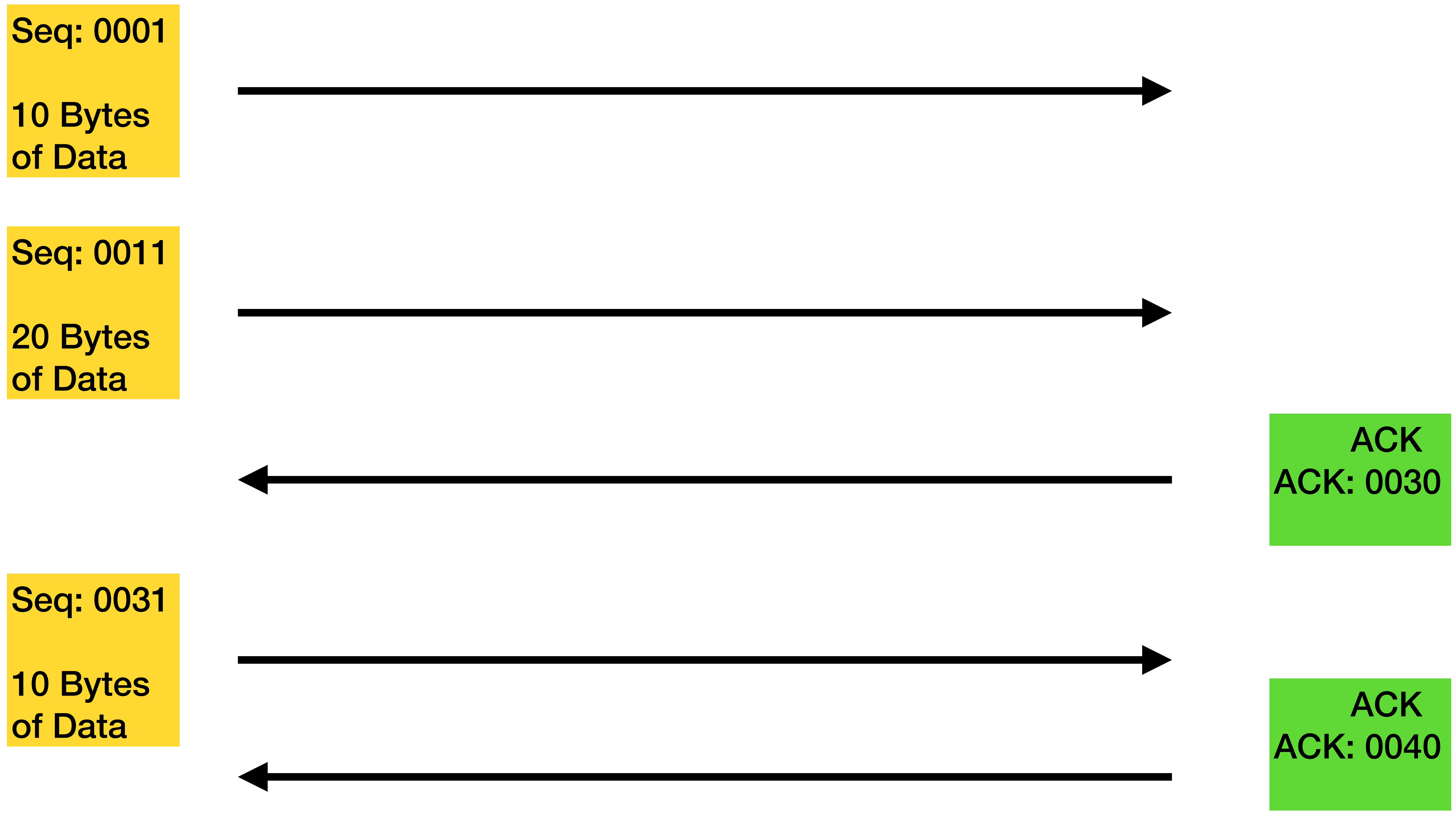
Sequence number and acknowledgement number

- Initial sequence number is random
- It notes the byte number of the first payload byte
- Acknowledgements are sent by the receiver
  - Acknowledgement Number = "I have received data up to this sequence number"
- In case of packet loss, the Acknowledgement Number is not increased

DE CIX

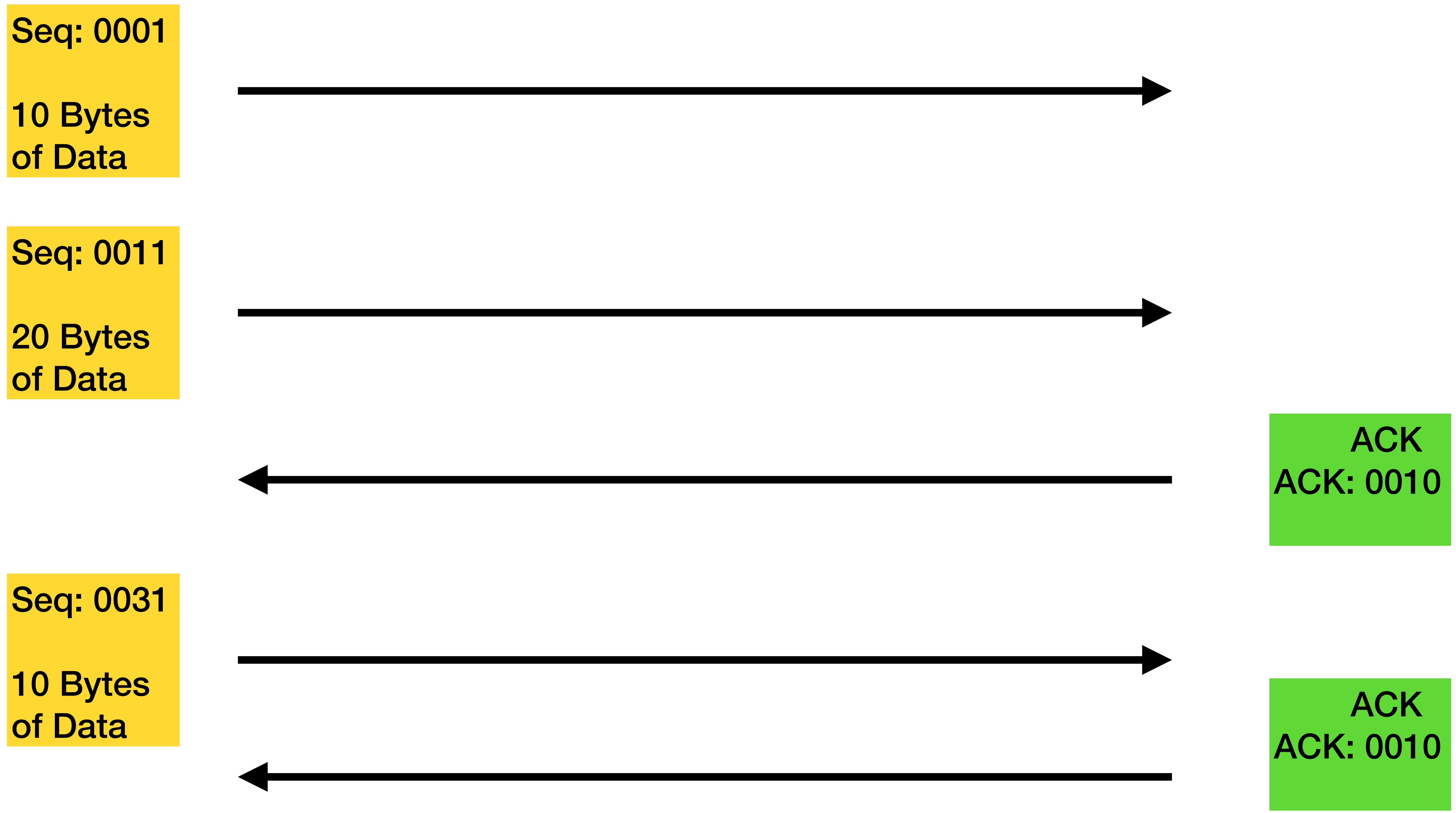
TCP Data Transfer

Acknowledgement of received data



TCP Data Transfer

Handling of packet loss



Features of TCP

1. Port numbers
2. Connections
3. Reliability
4. **Flow control**
5. Congestion avoidance

Flow control

How "fast senders" can deal with "slow receivers"

- Flow control is about end-to-end communication
- The sender should not "overload" the receiver
- Remember "window size" in the TCP header?
  - "*Amount of data the sender can send without ACK from receiver*"
- So if receivers tends to get overloaded, it simply reduced the window size
- If sender has sent <*window size*> data, it stops until it received an ACK from the receiver

DE CIX

Flow control

When window size was designed, networks were way slower

- Original window size field is 16 bit - that means 64kByte of data
  - This is too small for todays (fast) networks
- Solution - see RFC7323 (first introduced 1988 in RFC1072)
  - "Window Scale" - uses a TCP option field when setting up a connection
  - Possible values: 0 (no scale) - 14
  - Typical value on Linux: 1
  - A scale of *n* means: multiply window value with 2^n

DE CIX

Features of TCP

1. Port numbers
2. Connections
3. Reliability
4. Flow control
5. **Congestion control**

Source: Public Domain
https://commons.wikimedia.org/wiki/File:Zeichen_124_-_Stau,_StVO_1992.svg

Congestion control

The bottleneck between sender and receiver

- Goal: Do not send more than the network can transport
- TCP uses four algorithms for that:
    - Slow start
    - Congestion avoidance
    - Fast retransmit
    - Fast recovery

Source: Public Domain
https://commons.wikimedia.org/wiki/File:Zeichen_124_-_Stau,_StVO_1992.svg

"Slow Start" & "Congestion Avoidance"

...because a sender does not know anything about the network

- Two algorithms to limit sender of data
- Sender keeps variables (per connection):
  - *Congestion Window* - limit of data that can be sent before receiving an ACK
  - *Slow Start Threshold* - switch from "Slow Start" to "Congestion Avoidance"
- Slow Start: Used at beginning or after packet loss

DE CIX

"Slow Start"

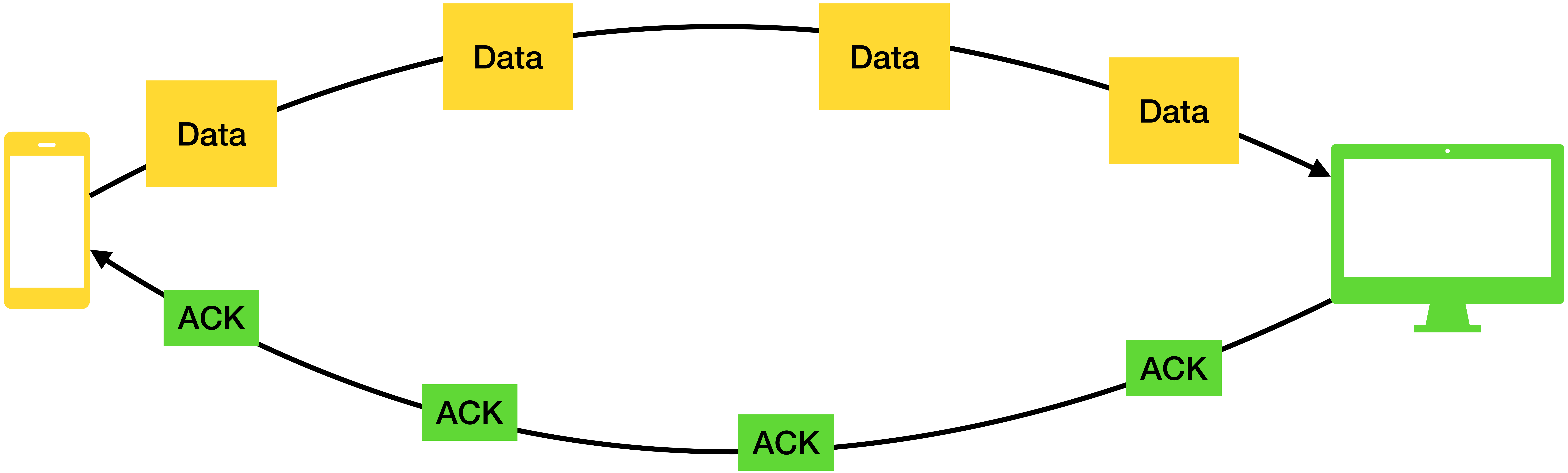
At the beginning of a connection or after packet loss

- Initial value of *Congestion Window* is set depending on maximum packet size of sender
  - Usually its 2-4 segments (packets)
- Once ACKs are received "*Congestion Window*" is increased
  - Each ACK increases *Congestion Window*
  - For details, see RFC5681
- Once *Congestion Window* > *Slow Start Threshold* **Congestion Avoidance** takes over

"Congestion Avoidance"

Do not send more than the network can transport

- Increase Congestion Window by one segment per each round-trip-time
  - **This is the reason low-latency is so important**
- This continues until congestion is detected



"Fast Retransmit"

Try again Sam...

- When a receiver gets a packet out-of-order it sends a "duplicate ACK" immediately
  - So the sender knows something might have been lost (or re-ordered?)
- If three duplicate ACKs come back to the sender, **Fast Retransmit** kicks in
  - Sender re-sends missing data immediately (without waiting for the retransmission timer)
- Then **Fast Recovery** takes over

DE CIX

"Fast Recovery"

Getting things back on track

- New data is being sent again now
- At a slower speed
- Runs until non-duplicate ACK arrives
- Then "Congestion Avoidance" takes over again

Is that all?

TCP has many more extensions

Some overview…

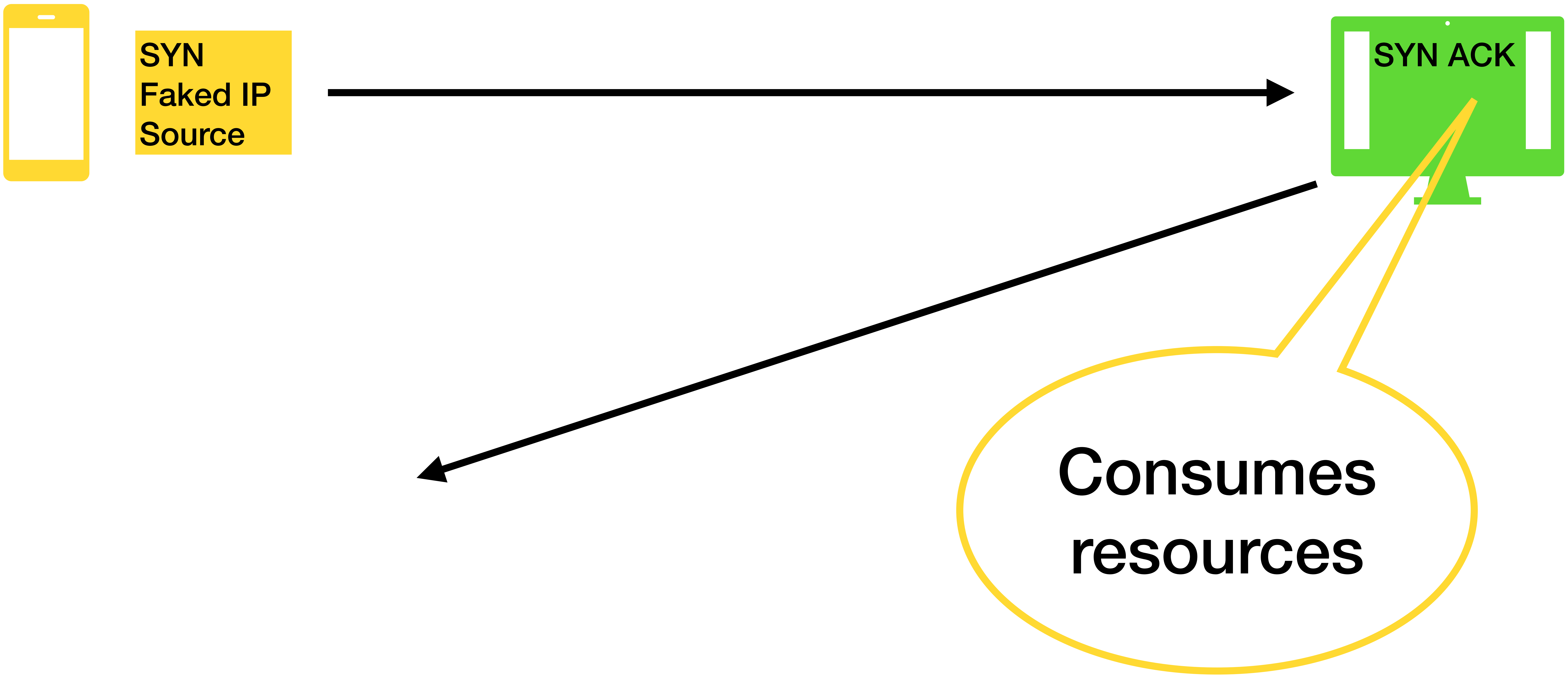
- Selective ACK (SACK) - RFC2018 - allows ACKs of single segments
- There are several security extensions, like defense against SYN flooding (RFC4987)
- TCP Authentication Option allows cryptographic signing of TCP segments (RFC5925)
- See RFC7414 for all TCP-related documents

What about security?

DE CIX

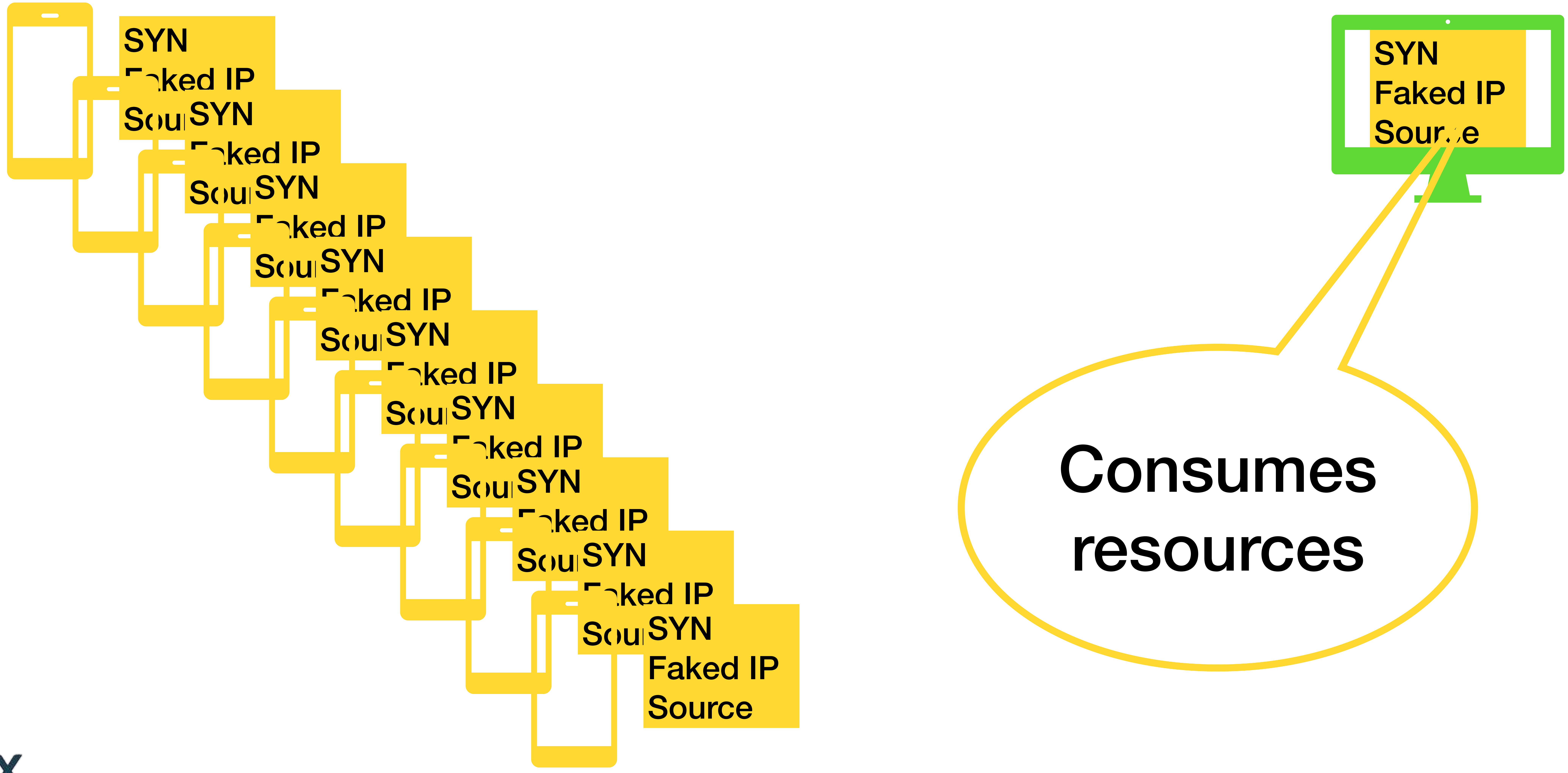
Remember "establishing a connection"?

What if the handshake is incomplete?



Remeber "establishing a connection"?

What if the handshake is incomplete?



This is the "TCP SYN Flood" attack

Overloading the receiver

- Send lots of TCP SYN packets (with faked IP source addresses)
- Each received SYN triggers a SYN/ACK and consumes resources (memory etc.)
- Until the connection attempt expires
- So if enough SYN packets are received, the receiver's tables fill up and no new (real) connections can be accepted. The receiver appears offline.
- Several mitigation strategies exist - see RFC4987

Guessing the Sequence Number

Disrupting connections

- To immediately shut down a TCP connection, the RST (reset) flag can be used
- A RST is only valid if it's sequence number is in the window
  - So the attacker must know (or guess) the sequence number, port numbers etc. correctly and fake the IP source address.
- RFC4953 describes the attack and counter measures

TCP - what it is used for

TCP
What is it used for?

- The list is too long
- Most application protocols use TCP
  - HTTP - browsing the web
  - SMTP - transporting email
  - SSH - secure log in to remote systems
  - ... and many many more

DE CIX

Conclusion

Conclusion

TCP - Transmission Control Protocol

- TCP is a connection oriented protocol on the transport layer
- It uses sophisticated algorithms for...
  - ...**reliable** data transfer
    - by end-to-end acknowledgement of data transferred
  - ...**efficient** use of available bandwidth
    - by increasing sending rate step by step
  - ...**fair**ness to other connections
    - by not overcrowding other TCP connections

Layer	Name
5	Application
4	Transport
3	Internet
2	Link
1	Physical

DE CIX

Thank you!

academy@de-cix.net

Interested in more webinars? Please subscribe to our mailing list at https://lists.de-cix.net/wws/subscribe/academy

DE-CIX Management GmbH | Lindleystr. 12 | 60314 Frankfurt | Germany
Phone + 49 69 1730 902 0 | sales@de-cix.net | www.de-cix.net

Links and further reading

Links and further reading

- Internet protocol - https://en.wikipedia.org/wiki/Internet_Protocol
- Protocol stack - https://en.wikipedia.org/wiki/Protocol_stack
  - Transport Layer: https://en.wikipedia.org/wiki/Transport_layer
  - Datagram: https://en.wikipedia.org/wiki/Datagram
- IP Network Model: https://en.wikipedia.org/wiki/Internet_protocol_suite
- IPv4
  - IPv4 - https://en.wikipedia.org/wiki/IPv4
- IPv6
  - IPv6 itself - https://en.wikipedia.org/wiki/IPv6
  - IPv6 header - https://en.wikipedia.org/wiki/IPv6_packet
- History of Internet and IP
  - Internet Hall of Fame - https://internethalloffame.org
  - Defense Advanced Research Projects Agency (DARPA) - https://www.darpa.mil
  - ARPANET - https://www.darpa.mil/about-us/timeline/arpanet
  - The "Protocol Wars" - https://en.wikipedia.org/wiki/Protocol_Wars

Internet RFCs (Standards)

- There are too many RFCs dealing with IPv4 and IPv6 to be listed here
- Just go to https://tools.ietf.org/html/ and use the search field
- How does something become RFC? https://www.rfc-editor.org/pubprocess/
- The IETF - Internet Engineering Task Force

DE CIX

Internet RFCs and other links about TCP

- TCP on Wikipedia:
  - https://en.wikipedia.org/wiki/Transmission_Control_Protocol
  - TCP Window Scale Option: https://en.wikipedia.org/wiki/TCP_window_scale_option
  - TCP Congestion Control: https://en.wikipedia.org/wiki/TCP_congestion_control
- Presentations and interesting links about TCP:
  - A great presentation about TCP: https://www.potaroo.net/presentations/2019-09-05-bbr.pdf

- Notable RFCs about TCP:
  - Initial definition (1974): RFC675
  - Roadmap of documents (start here!): RFC7414
  - Initial Sequence Number calculation: RFC6528
  - Window size RFCs:
    - RFC7323: TCP Extensions for High Performance
  - Congestion control RFCs:
    - TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms: RFC2001 (obsolete)
    - TCP Congestion Control: RFC5681

DE CIX