ZSL
Zentrum für Schulqualität
und Lehrerbildung
Baden-Württemberg

Networking Academy

Transport Layer

Andreas Grupp
Andreas.Grupp@zsl-rstue.de

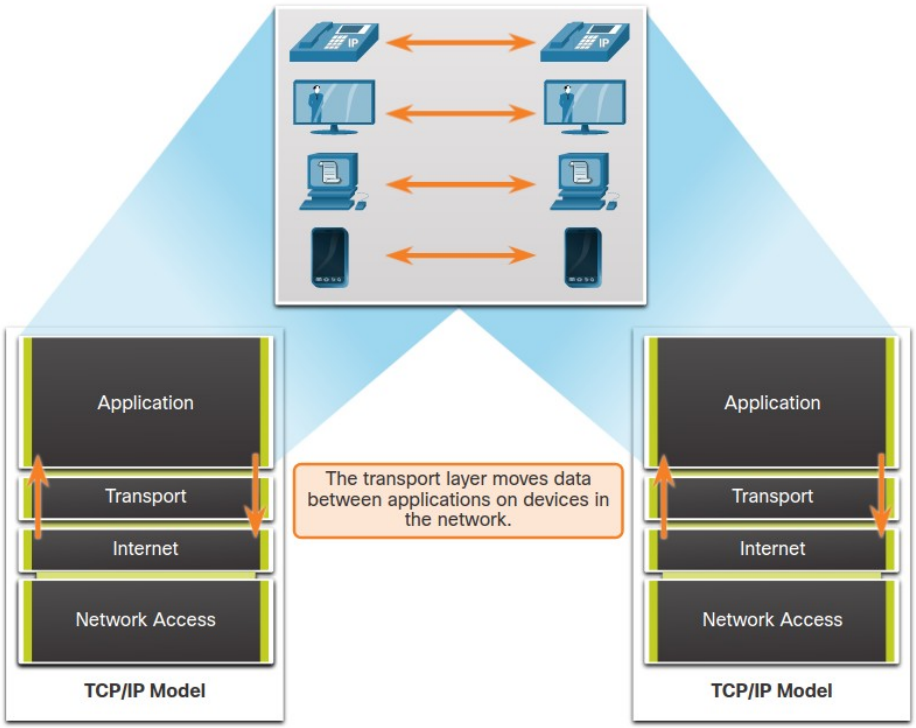
Carina Haag
carina.haag@zsl-rsma.de

Tobias Heine
tobias.heine@zsl-rsma.de

Uwe Thiessat
uwe.thiessat@gbs-sha.de

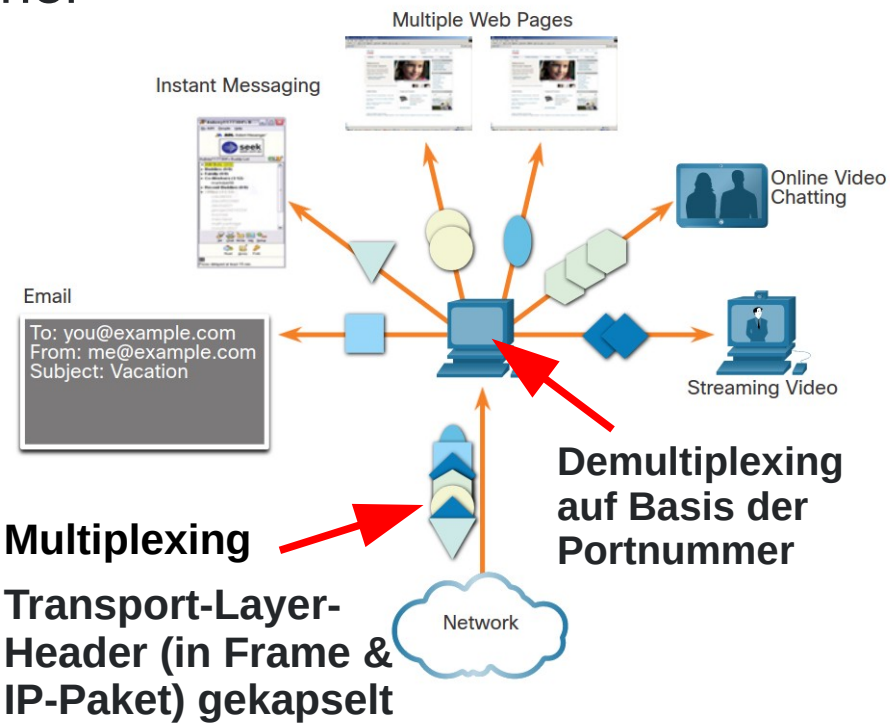
Rolle des Transport Layers

- Einrichtung und Verwaltung einer logischen Kommunikationsverbindung zwischen Anwendungsprozessen auf Quell- und Ziel-Host (z.B. Aufruf einer Webseite).
- *Feature:* Bietet eine zuverlässige Datenübertragung an, wenn man es benötigt (z.B. Korrektur einer fehlerhaften Datenübertragung).
- Die dominierenden Protokolle sind TCP und UDP. Grundsätzliches Auswahl- bzw. Entscheidungskriterium: *Geschwindigkeit ↔ Zuverlässigkeit*



Aufgaben des Transport Layers

- Verwaltung und Verfolgung unterschiedlicher Konversationen / Anwendungsprozessen.
- Segmentierung und Rekonstruktion des Datenstroms. Segmentgröße passend für tiefere Layer (MTU).
- Multiplexing und Demultiplexing ermöglicht gleichzeitige Kommunikation
- Identifizierung und Weiterleitung des Datenstroms an die Anwendung. Eine *Portnummer* dient als Kennung bzw. Adresse.
- Hinzufügung des Transport-Layer-Header - u.a. Portnummer.



Wichtige Protokolle des TCP/IP-Transport-Layer

Transmission Control Protocol (TCP)

- Verbindungsorientiert
- PDU: Segmente
- Zuverlässig: verlorene Segmente werden neu gesendet
- Datenrekonstruktion beim Empfänger (Reassemblierung)
- Flusskontrolle (Steuerung der Übertragungsgeschwindigkeit)
- Overhead min. 20 Octets
- Bsp.: HTTP(S), SMTP, FTP, SSH
- RFC 793

full-featured → Zuverlässig!

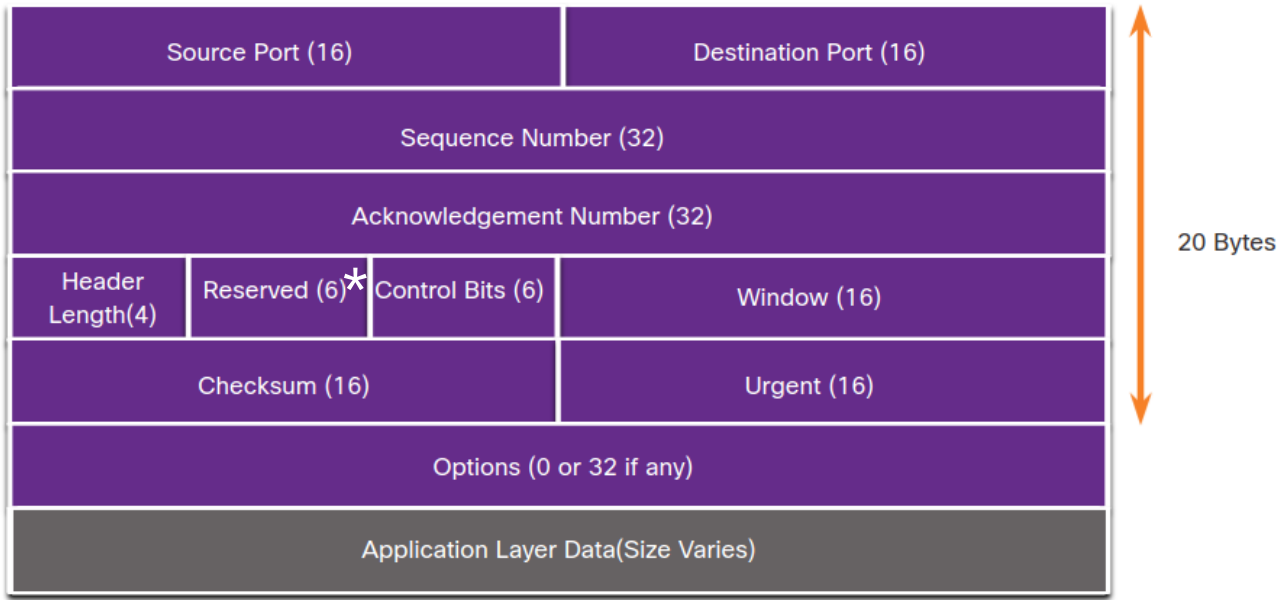
User Datagram Protocol (UDP)

- Verbindungslos (Transaktionsorientiert)
- PDU: Datagramme
- "Best effort"-Ansatz, verlorene Segmente sind erst mal weg
- Keine Flusskontrolle oder geordnete Datenrekonstruktion
- Kleiner Overhead von 8 Octets
- Bsp.: DNS (i.d.R.), VoIP, Videostr., Spiele, TFTP, DHCPv4, SNMP
- RFC 768

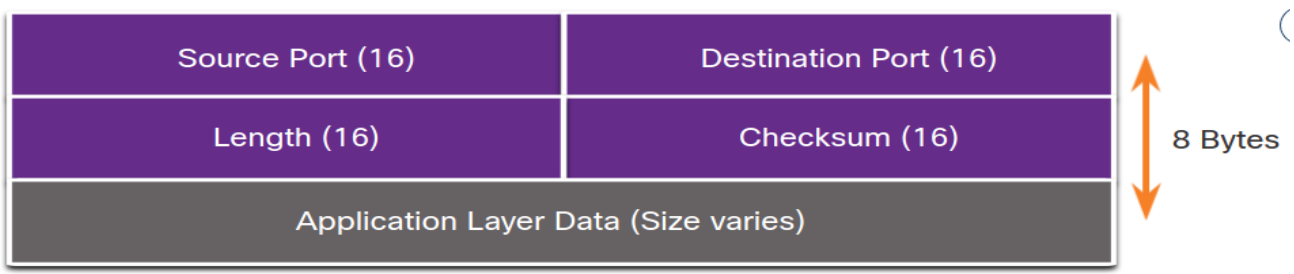
„keep it simple“ → Schneller!

Quiz 14.1.7

TCP- vs. UDP-Header



- TCP-Header gefolgt von den gekapselten Anwendungsdaten.
- Mehr Felder und komplexer als UDP-Header.
- TCP ist ein Stateful-Protokoll und trackt den Status der Verbindung (zustandsbehaftet).



- UDP-Header gefolgt von den gekapselten Anwendungsdaten.
- UDP ist ein Stateless-Protokoll (zustandslos).

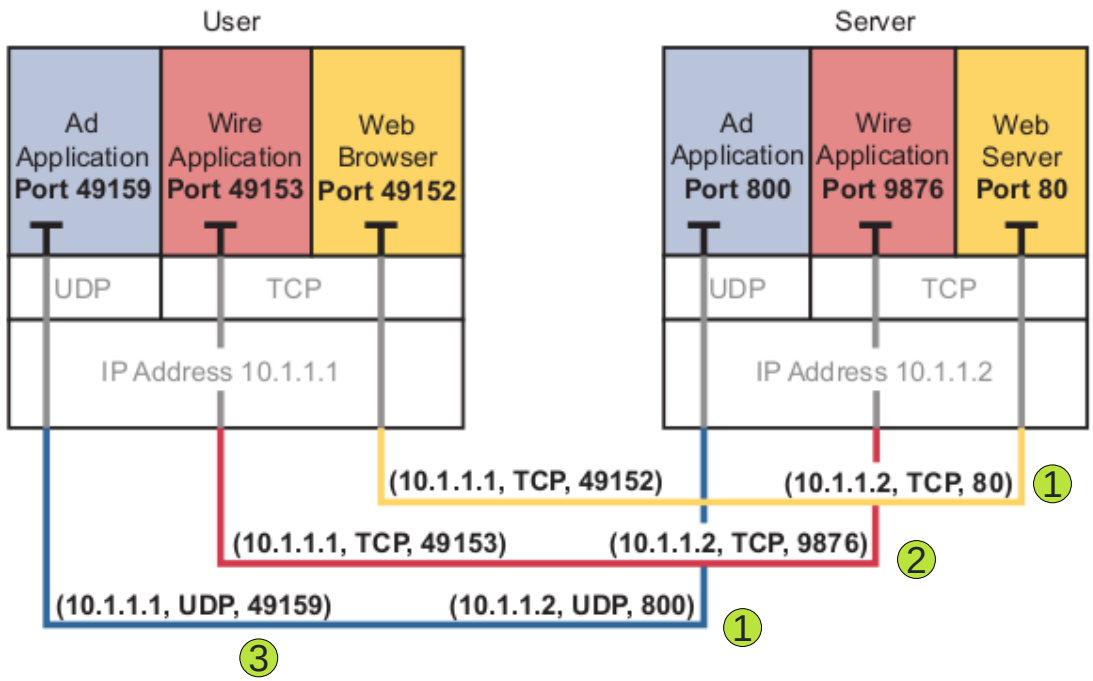
**) Aktuell gibt es neun Kontrollbits und drei reservierte Bits. Die zusätzlichen drei Kontrollbits werden zur Erkennung einer Überlastsituation im Netz verwendet.*

Quiz 14.2.5 & 14.3.4

Einführung Port-Nummern

	Port-Gruppe	Nr.	Beschreibung
1	Well-Known- Ports	0 – 1.023	Reservierte Ports durch IANA für standardisierte Serverdienste
2	Registrierte Ports	1.024 – 49.151	Auf Antrag zugewiesene Ports durch IANA oder freigewählte Ports für spezifische Server-Anwendungen (weniger restriktiv). Sie können auch dynamisch als Client-Ports verwendet werden.
3	Private und/oder dynamische Ports	49.152 – 65.535	Temporäre Ports, die dynamisch vom Client-Betriebssystem zugewiesen werden (auch ephemeral / kurzlebig genannt).

Beispiel eines Clients der drei Verbindungen zu Applikationen eines Servers hat:
Web Server
Wire (Videokonferenz)
Ad (Werbedienst)



Bildquelle: Odom, W. (2019) CCNA 200-301: Official Cert Guide, Volume 2, Cisco Press

Typische Ports mit Protokollen

Well-Known-Ports

Port Number	Protocol	Application
20	TCP	File Transfer Protocol (FTP) - Data
21	TCP	File Transfer Protocol (FTP) - Control
22	TCP	Secure Shell (SSH)
23	TCP	Telnet
25	TCP	Simple Mail Transfer Protocol (SMTP)
53	UDP, TCP	Domain Name Service (DNS)
67	UDP	Dynamic Host Configuration Protocol (DHCP) - Server
68	UDP	Dynamic Host Configuration Protocol - Client
69	UDP	Trivial File Transfer Protocol (TFTP)
80	TCP	Hypertext Transfer Protocol (HTTP)
110	TCP	Post Office Protocol version 3 (POP3)
143	TCP	Internet Message Access Protocol (IMAP)
161	UDP	Simple Network Management Protocol (SNMP)
443	TCP	Hypertext Transfer Protocol Secure (HTTPS)

Registered TCP/UDP Ports

1723 PPTP *(für VPN)*

1812 RADIUS

3306 MYSQL

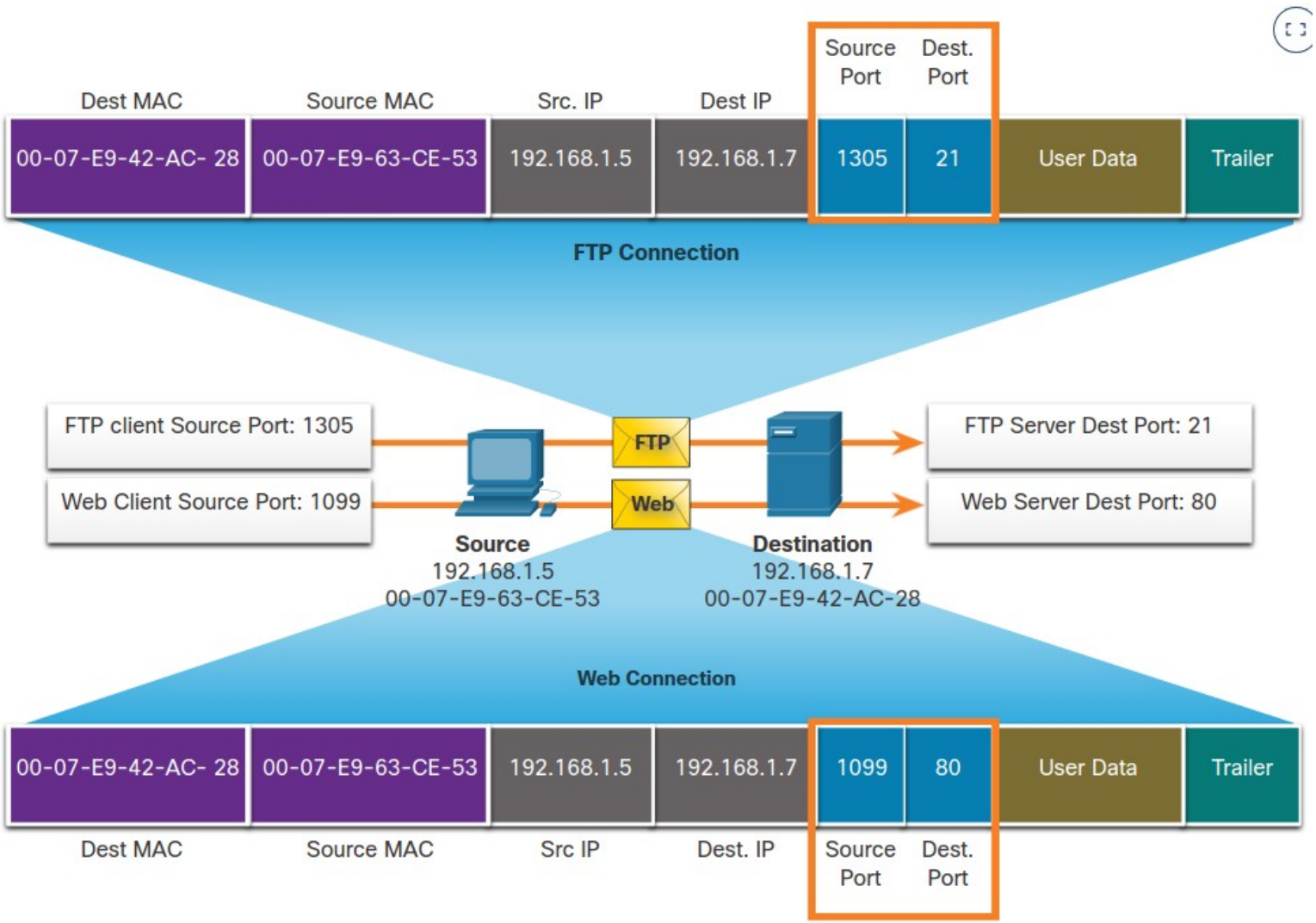
5004 RTP *(Voice and Video Transport Protocol)*

5040 SIP (VoIP)

8080 HTTP Alternate

etc.

Zwei Beispiele mit Layer 2, 3 und Layer 4 Adressen



Sockets

Zu jeder Kommunikations-Verbindung gehören immer die folgenden Adressen:

- Client-Netzadresse (IP-Adresse)
- Port des Client-Prozesses

} = Socket
Beispiel: 192.168.1.5:1305

- Server-Netzadresse (IP-Adresse)
- Port des Server-Prozesses

} = Socket
Beispiel: 192.168.1.7:80

"Socketpaar" identifiziert eine Verbindung:

- Eigene IP-Adresse und eigenen Anwendungsport
- Gegen-IP-Adresse und dortigen Anwendungsport

Offene Verbindungen anzeigen

```
C:\Users>netstat -f

Aktive Verbindungen

  Proto  Lokale Adresse          Remoteadresse          Status
  TCP    192.168.178.57:49698   93.184.220.29:http     HERGESTELLT
  TCP    192.168.178.57:50487   ec2-52-42-164-233.us-west-2.compute.amazonaws.com:https  HERGESTELLT
  TCP    192.168.178.57:50613   ip199.ip-51-38-2.eu:https  WARTEND
  TCP    192.168.178.57:50617   51.105.249.239:https   HERGESTELLT
  TCP    192.168.178.57:50957   40.79.65.78:https      WARTEND
  TCP    192.168.178.57:51081   20.186.48.46:https     WARTEND
  TCP    192.168.178.57:51149   40.79.66.209:https     WARTEND
  TCP    192.168.178.57:51239   40.79.65.78:https      WARTEND
  TCP    192.168.178.57:51292   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51302   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51305   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51310   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51311   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51319   64.4.54.18:https       WARTEND
  TCP    192.168.178.57:51320   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51329   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51335   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51351   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51357   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51361   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51369   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51375   www.heise.de:https     WARTEND
  TCP    192.168.178.57:51378   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51380   fra16s07-in-f19.1e100.net:https  HERGESTELLT
  TCP    192.168.178.57:51388   104.26.5.227:https     HERGESTELLT
  TCP    192.168.178.57:51394   192.229.233.4:https    HERGESTELLT
  TCP    192.168.178.57:51396   gzhls.at:https         HERGESTELLT
  TCP    192.168.178.57:51397   64.4.54.18:https       WARTEND
  TCP    192.168.178.57:51399   188.14.190.35.bc.googleusercontent.com:https  HERGESTELLT
  TCP    192.168.178.57:51400   ip206.ip-51-38-2.eu:https  HERGESTELLT
  TCP    192.168.178.57:51402   52.114.128.43:https    WARTEND
  TCP    192.168.178.57:51403   184.3.241.35.bc.googleusercontent.com:https  HERGESTELLT
  TCP    192.168.178.57:51418   server-13-224-197-99.fra2.r.cloudfront.net:https  HERGESTELLT
  TCP    192.168.178.57:51420   lehrerfortbildung-bw.de:https  WARTEND
  TCP    192.168.178.57:51421   lehrerfortbildung-bw.de:https  WARTEND
  TCP    192.168.178.57:51422   lehrerfortbildung-bw.de:https  WARTEND
  TCP    192.168.178.57:51437   server-13-225-87-126.fra2.r.cloudfront.net:https  HERGESTELLT
  TCP    192.168.178.57:51438   89.163.211.242:https   WARTEND
  TCP    192.168.178.57:51439   server-13-224-197-98.fra2.r.cloudfront.net:https  HERGESTELLT
  TCP    192.168.178.57:51442   104.27.162.128:https   HERGESTELLT
```

"**`netstat`**"- Kommando unter DOS-Shell bzw. Linux-Shell zeigt die derzeit offenen Verbindungen an!
Parameter **`-f`** für FQDN

Um Domain-Sockets unter Linux auszu-blenden =>
`netstat –tua`

Tipp Linux:
`lsof -i`

Quiz 14.4.5

Features von TCP

- Kontrollierter Verbindungsaufbau zwischen den beiden Hosts

  → **Three-Way-Handshake**

  und kontrollierter Verbindungsabbau.

- Bestätigung erhaltener Segmente und ggf. erneute Übertragung

  → **Positive Acknowledgment and Retransmission (PAR)**

  Führt mit Three-Way-Handshake zu zustandsbezogener (stateful) Kommunikation.

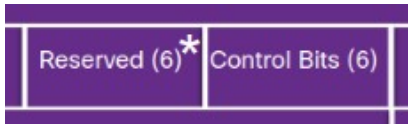
- Steuerung der Übertragungsgeschwindigkeit (Flusskontrolle)

  → **Sliding Window**

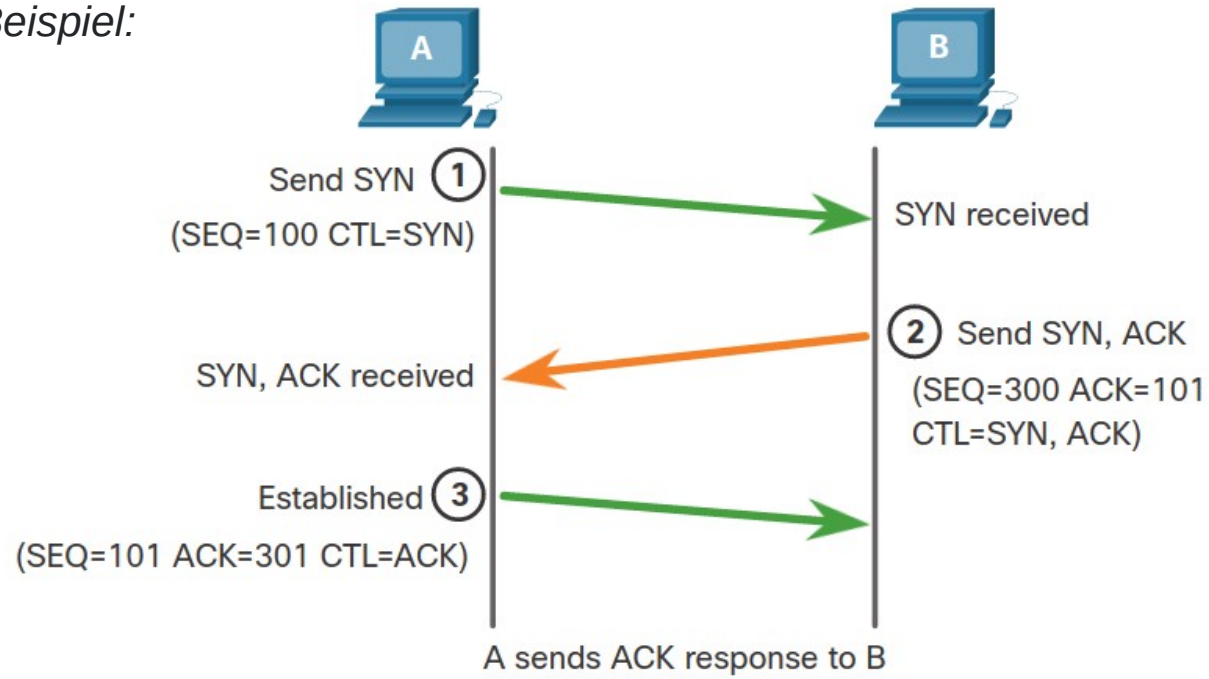
Three-Way-Handshake (Verbindungsaufbau)

Für den Verbindungsaufbau sind die Kontrollbits SYN und ACK verantwortlich.

Wireshark-Analyse →



Beispiel:



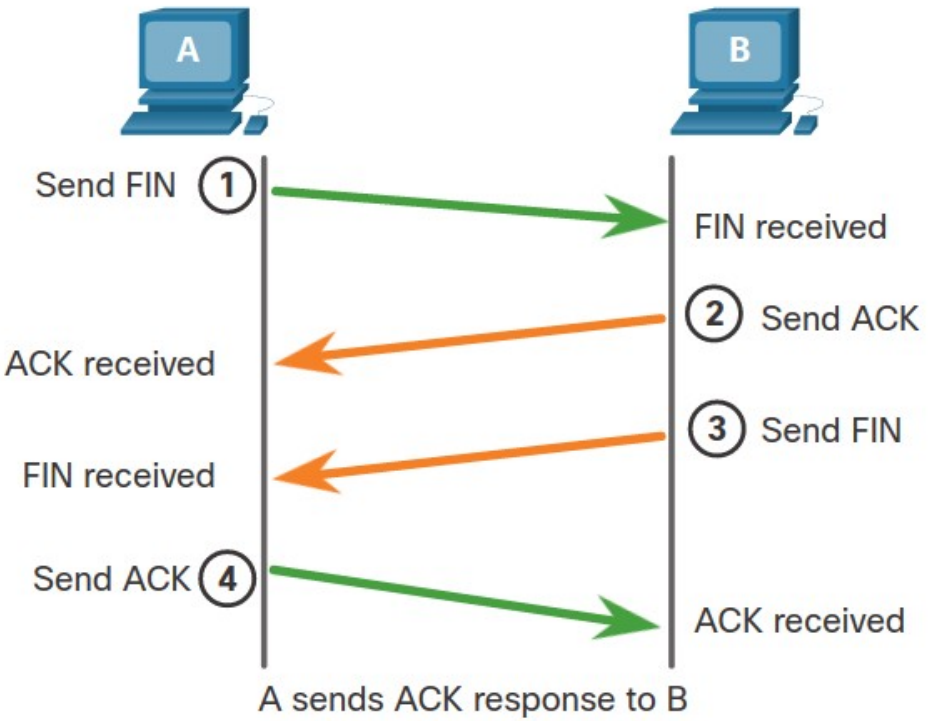
①
```
Flags: 0x002 (SYN)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Nonce: Not set
  .... 0... .... = Congestion Window
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ...0 .... = Acknowledgment: No
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
▸ .... .... ..1. = Syn: Set
  .... .... ...0 = Fin: Not set
```

②
```
Flags: 0x012 (SYN, ACK)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Nonce: Not set
  .... 0... .... = Congestion Window
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ...1 .... = Acknowledgment: Se
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
▸ .... .... ..1. = Syn: Set
  .... .... ...0 = Fin: Not set
```

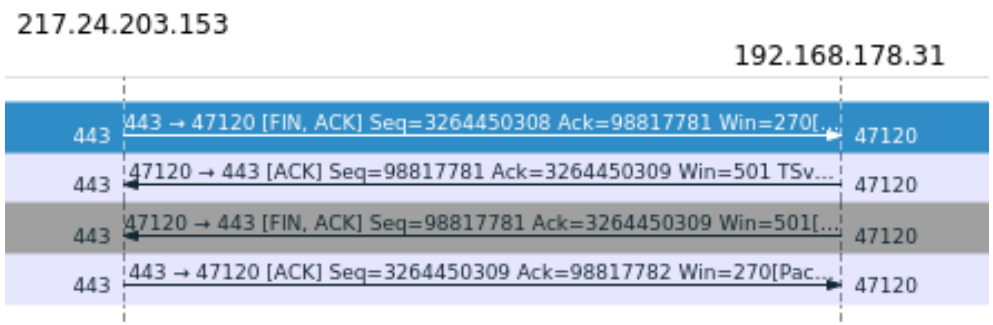
③
```
Flags: 0x010 (ACK)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Nonce: Not set
  .... 0... .... = Congestion Window
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ...1 .... = Acknowledgment: Se
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
  .... .... ..0. = Syn: Not set
  .... .... ...0 = Fin: Not set
```

Verbindungsabbau

Beispiel:



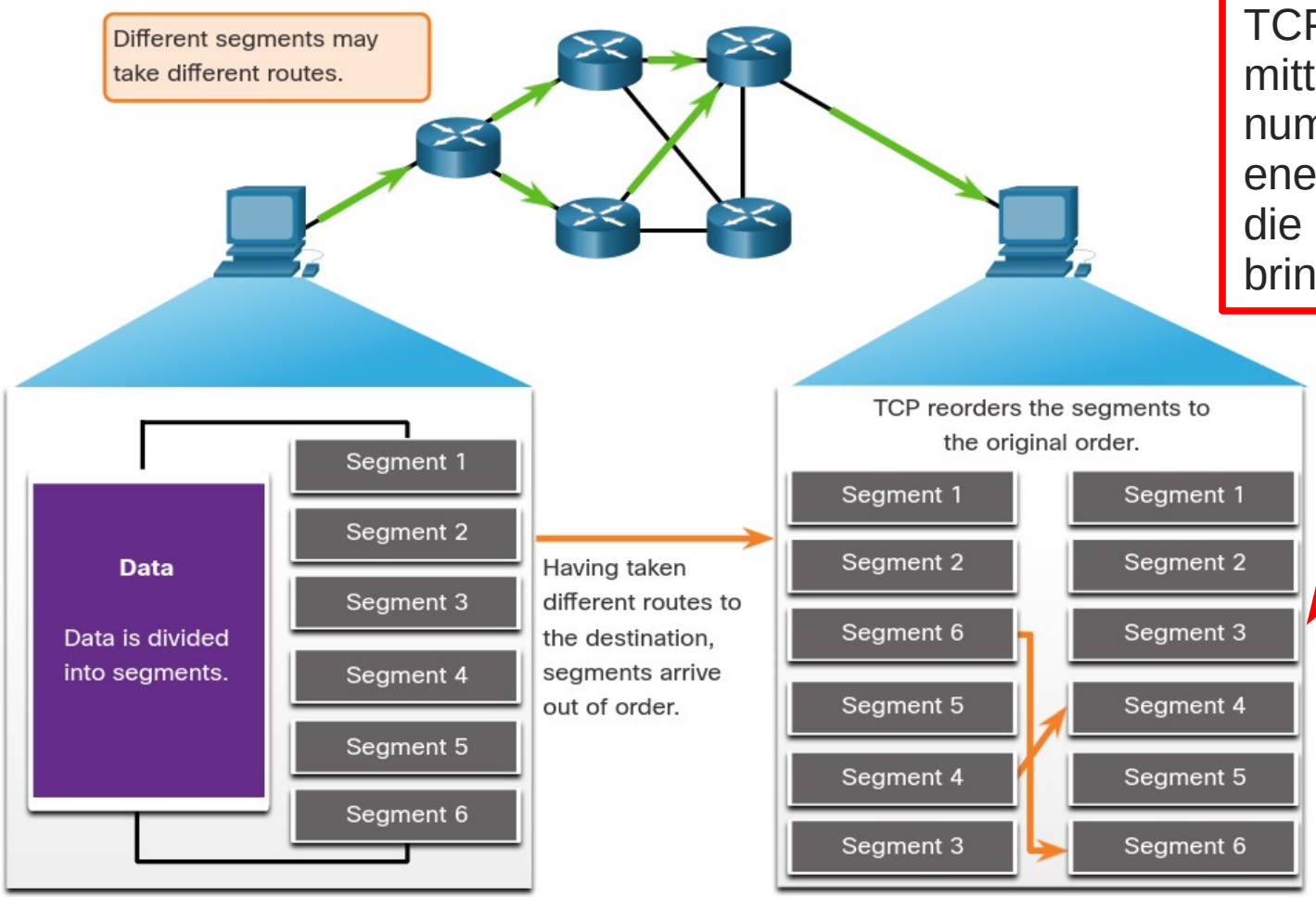
Wireshark-Analyse:



Verbindungen können auch durch das Kontrollbit RST „hart“ getrennt werden.

Quiz 14.5.6

Rekonstruktion der Daten

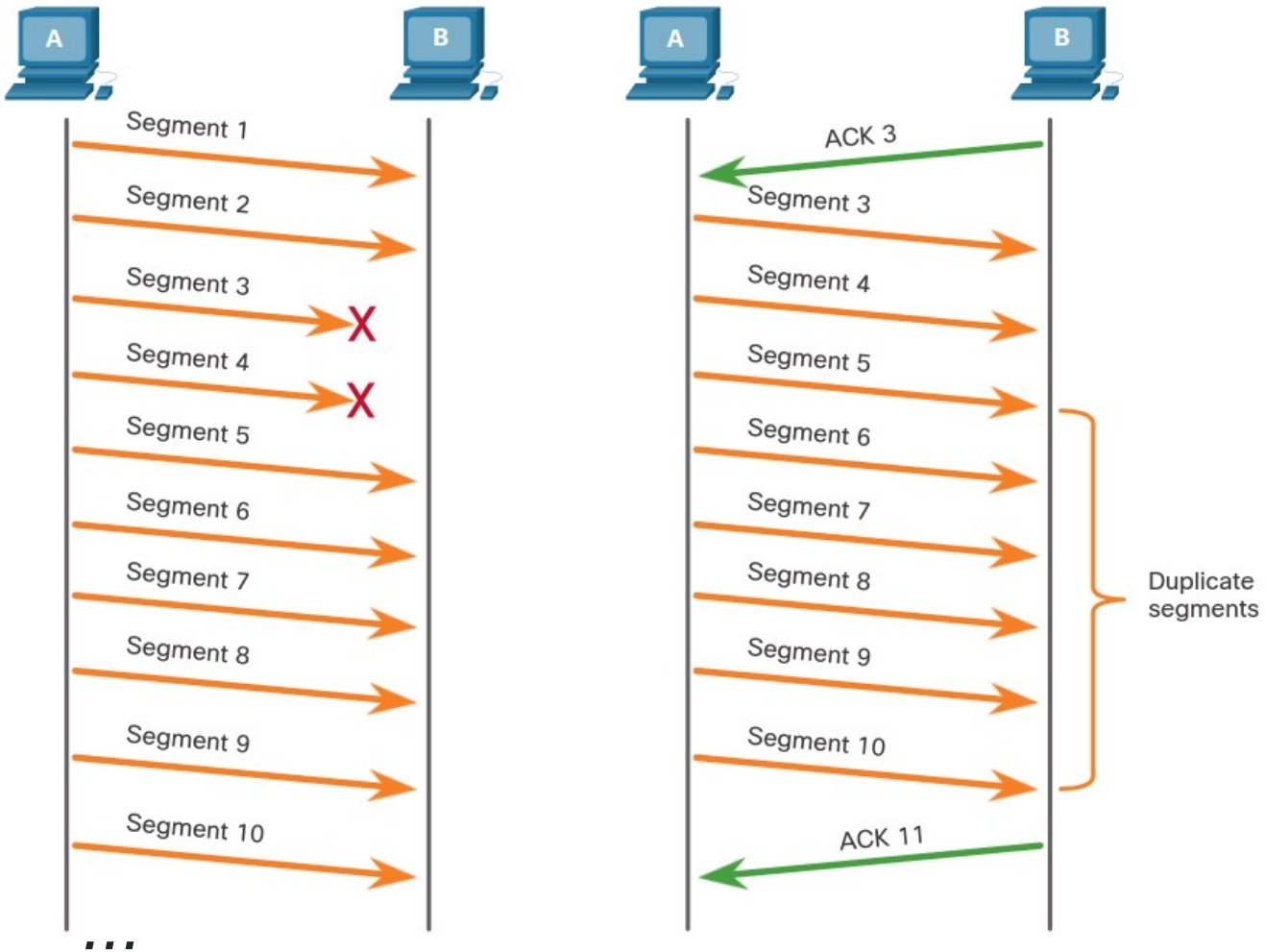


TCP kann durch die Übermittlung von Sequenznummern die empfangenen Segmente wieder in die ursprüngliche Form bringen.

UDP kann dies mangels Sequenznummern nicht!

Positive Acknowledgment and Retransmission (PAR)

Fortsetzung...

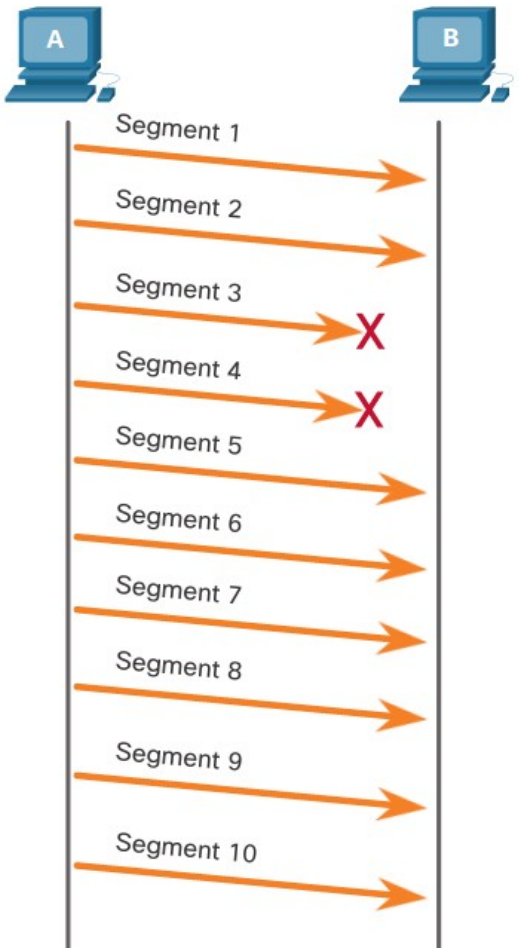


ACK 3 *bedeutet, dass Segment 1 und 2 angekommen sind und hiermit bestätigt werden und als nächstes Segment 3 erwartet wird.*

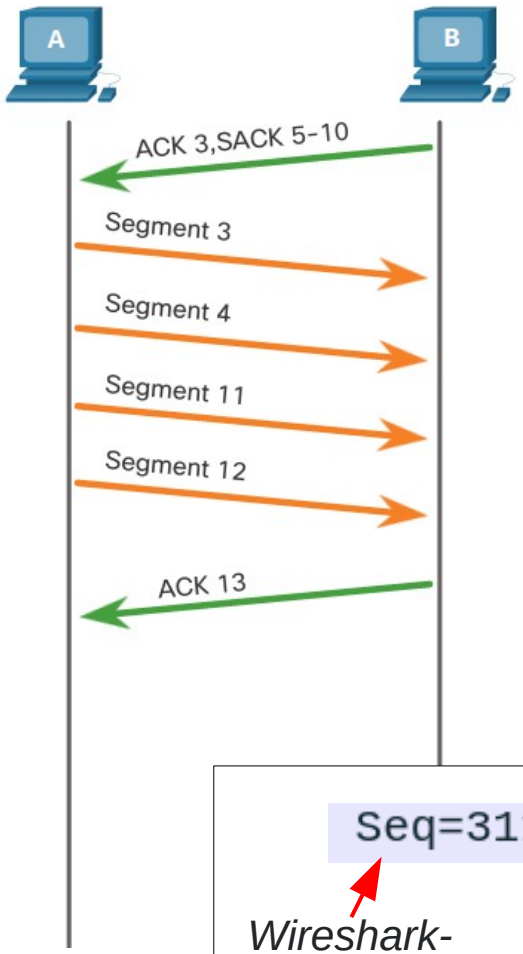
Fehlende Segmente werden neu übertragen.

Problem: Sender überträgt mehrfach die gleichen Segmente, obwohl nur zwei Segmente gefehlt hätten.

Selective Acknowledgment (SACK)



Fortsetzung...



ACK 3, SACK 5-10
bedeutet, dass Segment 1 und 2 sowie Segment 5 bis 10 angekommen sind. Als nächstes werden Segment 3 und Segment 11 erwartet.

<u>Vorteil:</u> Präzisere Beschreibung der fehlenden Segmente möglich.

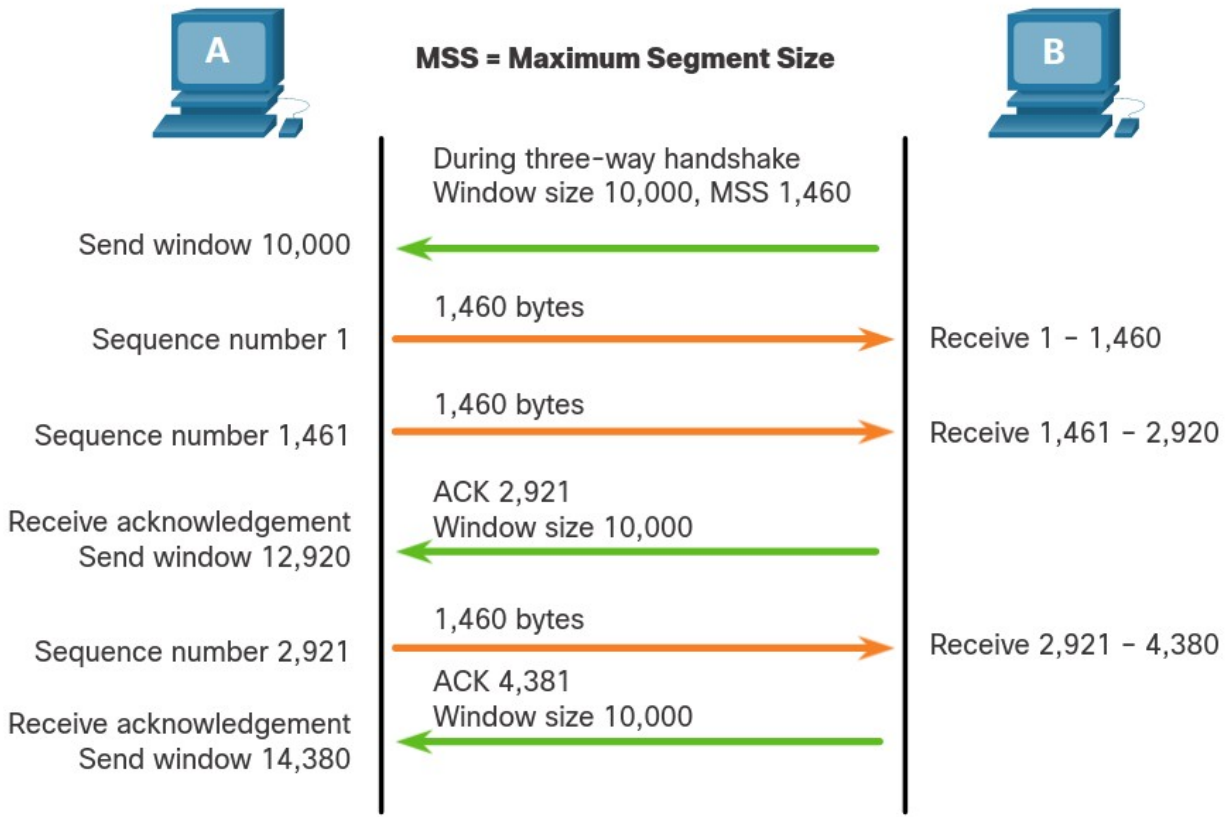
```
Seq=3114 Ack=1138 ... SLE=2586 SRE=2787
```

Wireshark-Auszug

RFC 2018 SLACK Auszug

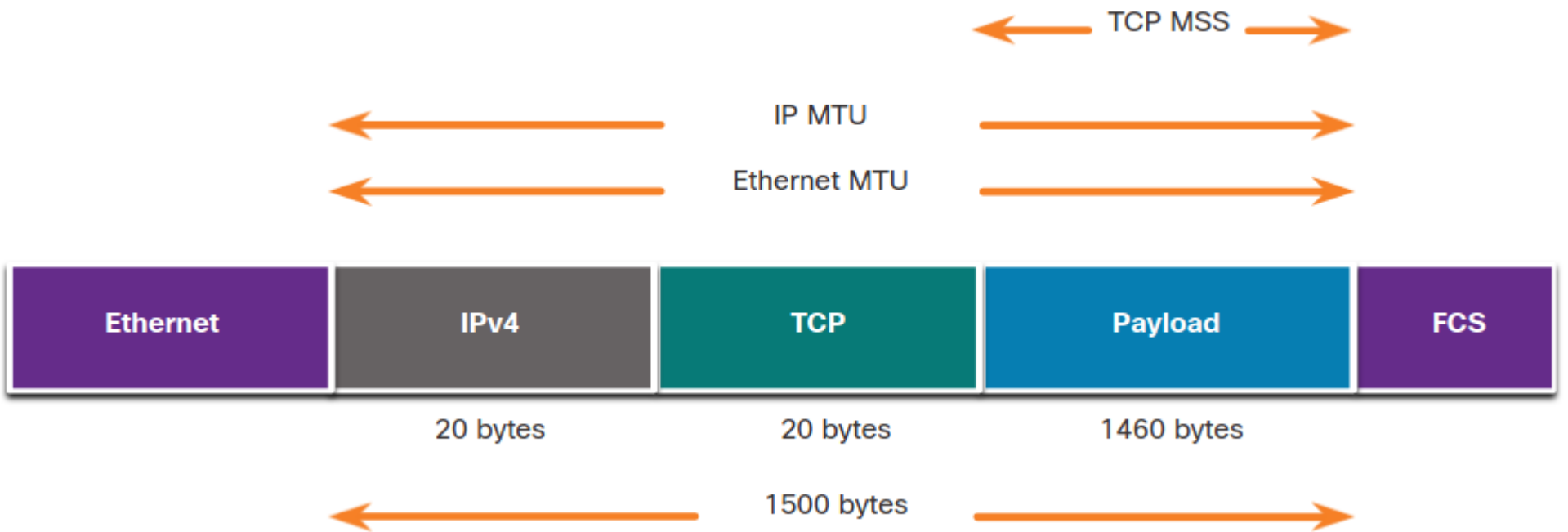
```
+--------+--------+--------+--------+
|      Left Edge of 1st Block       |
+--------+--------+--------+--------+
|      Right Edge of 1st Block      |
+--------+--------+--------+--------+
```

TCP Flow Control – Window Size & Acknowledgements

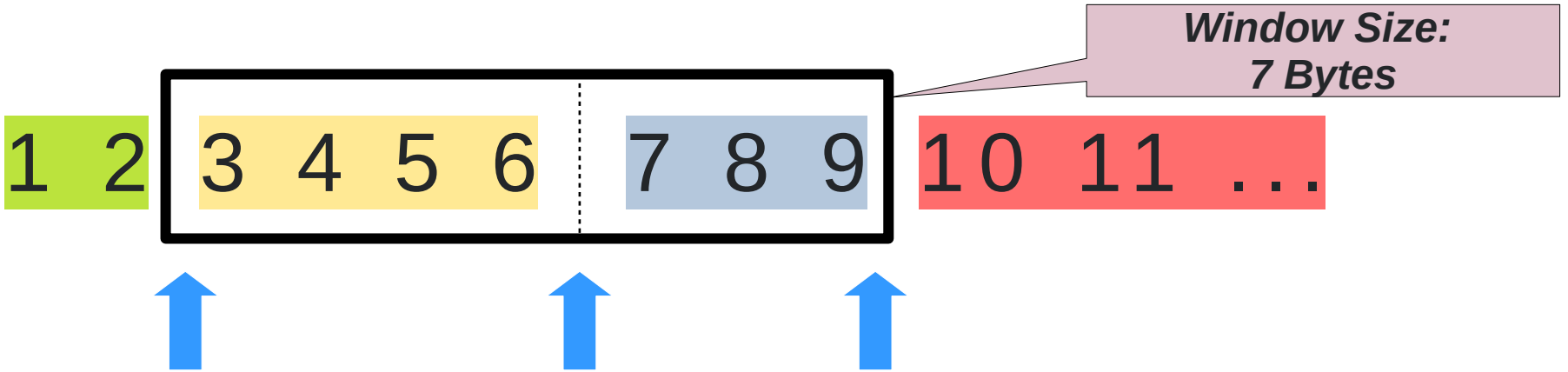


Mit der „Window Size“ legt der Empfänger die Anzahl der Bytes fest, die vom Sender gesendet werden dürfen, ohne vorher auf ein Acknowledgement der zuvor gesendeten Daten warten zu müssen. Window Size wird während der Kommunikation ggf. angepasst → Flusskontrolle (Flow Control)

Maximum Segment Size (MSS)



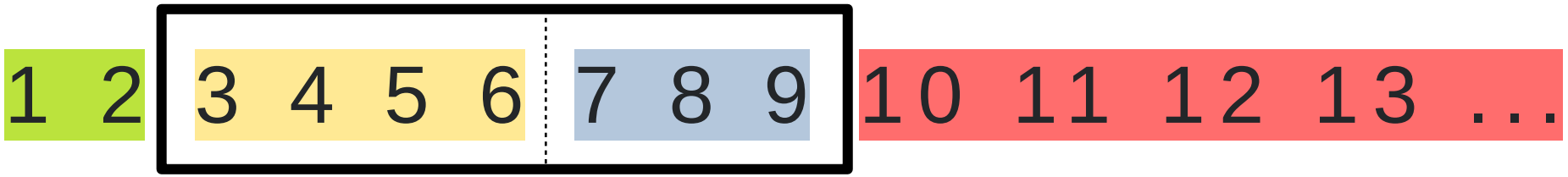
Sliding Window (Schiebe-Fenster-Prinzip)



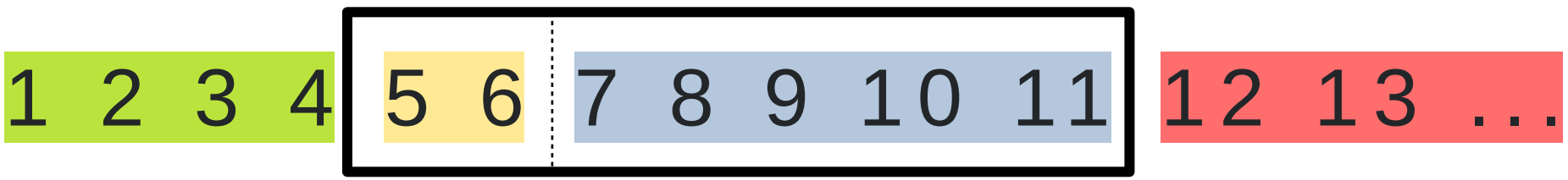
Sliding Window mit drei Pointern:

- 1. - 2. Byte wurden versandt und sind bestätigt.
- 3. - 6. Byte wurden gesendet, aber sind noch nicht bestätigt.
- 7. - 9. Bytes können noch versendet werden.
- Ab dem 10. Byte darf die Versendung erst nach Erhalt einer Bestätigung erfolgen.

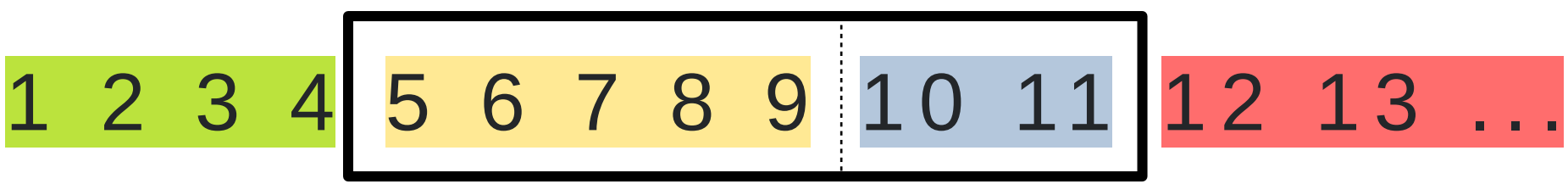
Sliding Window (Schiebe-Fenster-Prinzip) - Beispiel



✉ → ***3. - 4. Byte werden bestätigt [ACK].***



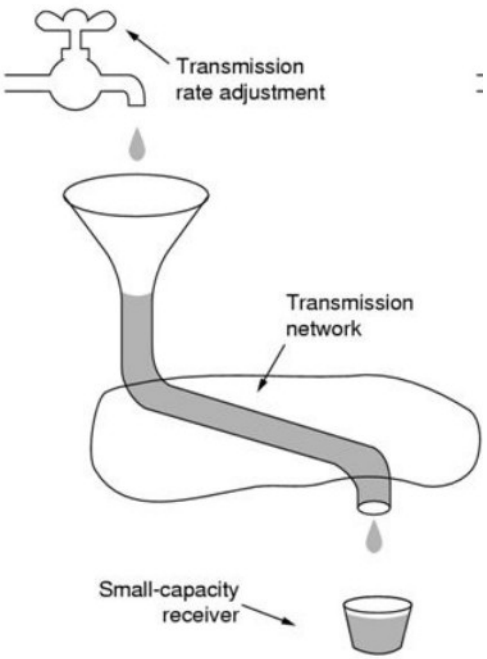
7. - 9. Byte werden versendet. → ✉



TCP Flow Control & Congestion Control

Die Sendegeschwindigkeit wird durch zwei Faktoren beeinflusst:

a) schnelles Netz, das an einen Empfänger mit geringer Kapazität weiterleitet



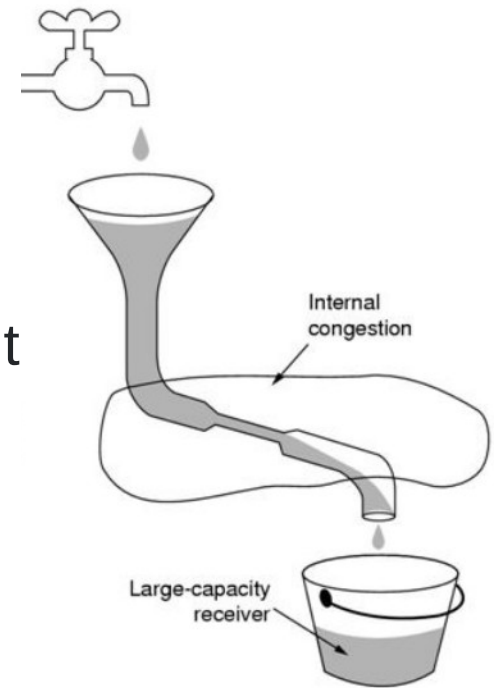
Lösung:

Flusskontrolle - Empfänger legt die „window size“ fest.

b) langsames Netz, das an einen Empfänger mit hoher Kapazität weiterleitet

Lösung:

Überlastkontrolle - Sender reduziert die Sendegeschwindigkeit auch, wenn die „window size“ noch nicht erreicht wurde.



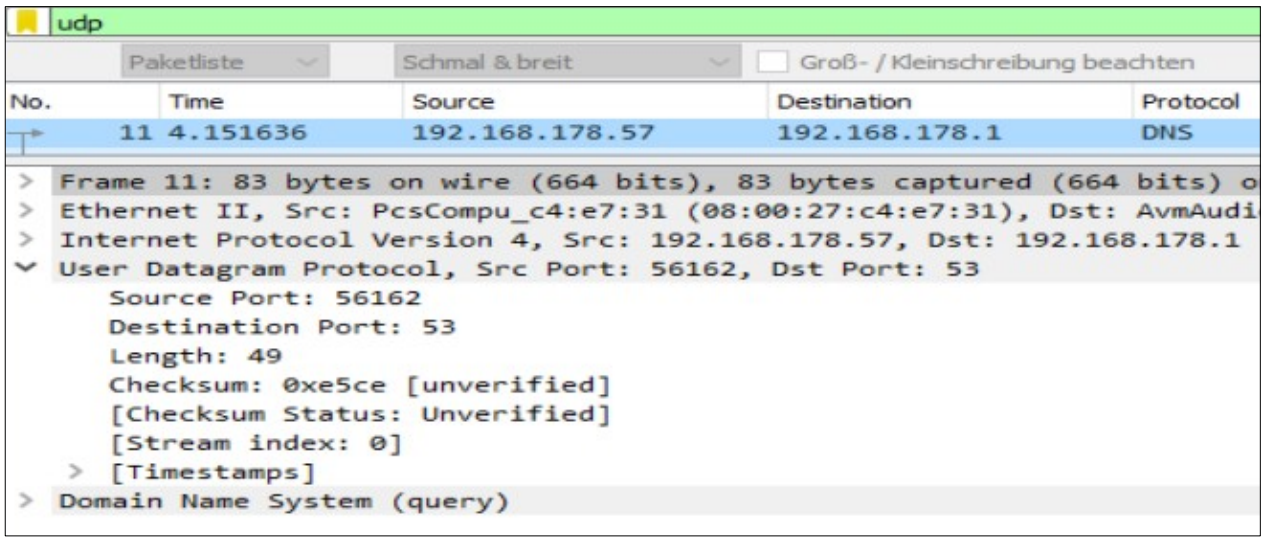
Siehe auch: Congestion Window, RFC 5681, Slow Start Algorithmus, Congestion-Avoidence-Algorithmus …

Quiz 14.6.8

User Datagram Protocol (UDP) - WDH

- Verbindungslos (Transaktionsorientiert)
- Zustandslos – kein geordneter Verbindungsaufbau und -abbau
- "Best effort" Ansatz, verlorene Segmente sind weg
- Keine Flusskontrolle oder geordnete Datenrekonstruktion

Wireshark-Analyse:



Quiz 14.7.5

Modulabschluss

- Lernziel-Zusammenfassung – 14.8.2
- Modul-Quiz – 14.8.3

Fragen ...