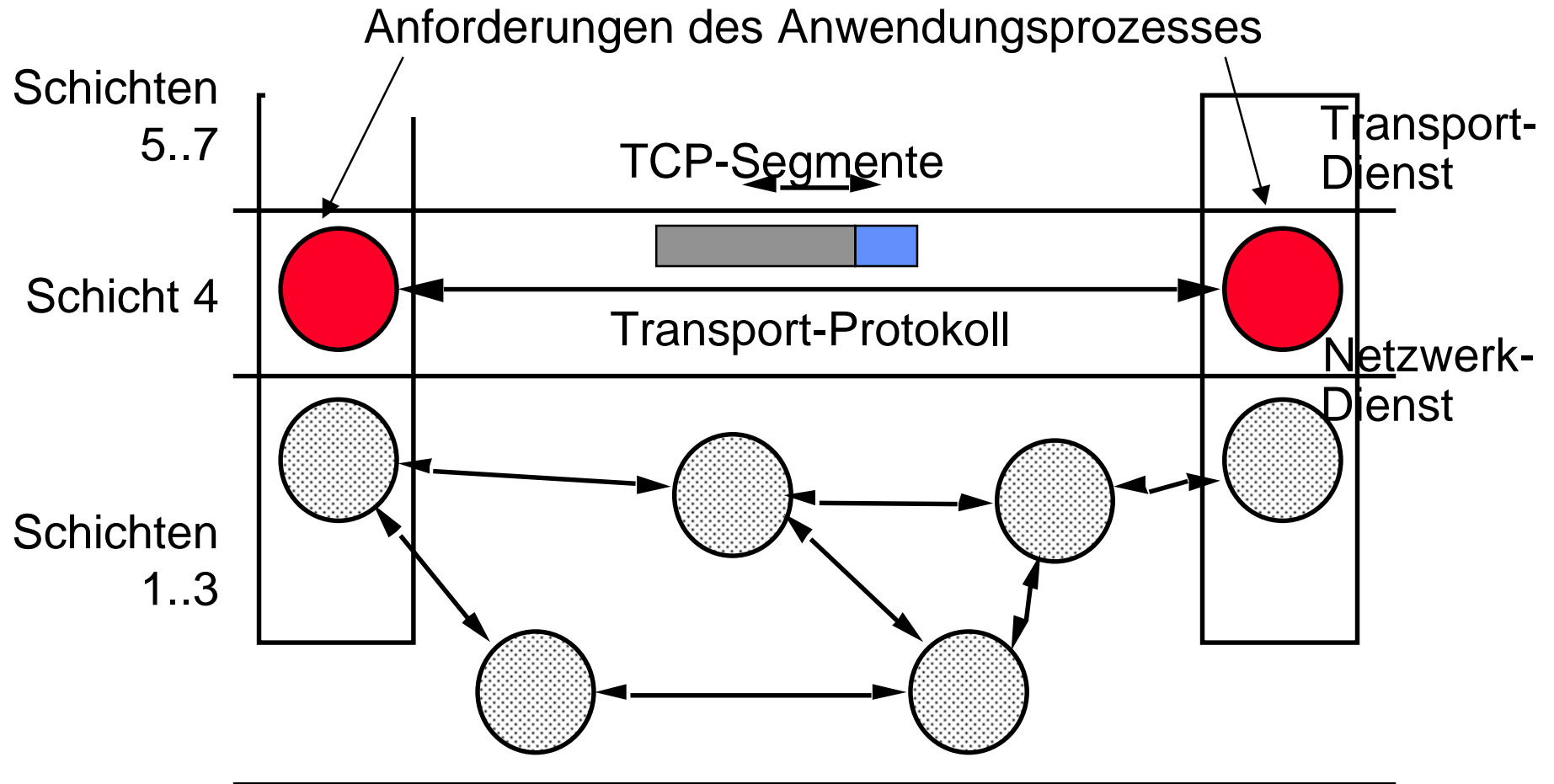


## **Die Transportprotokolle:**

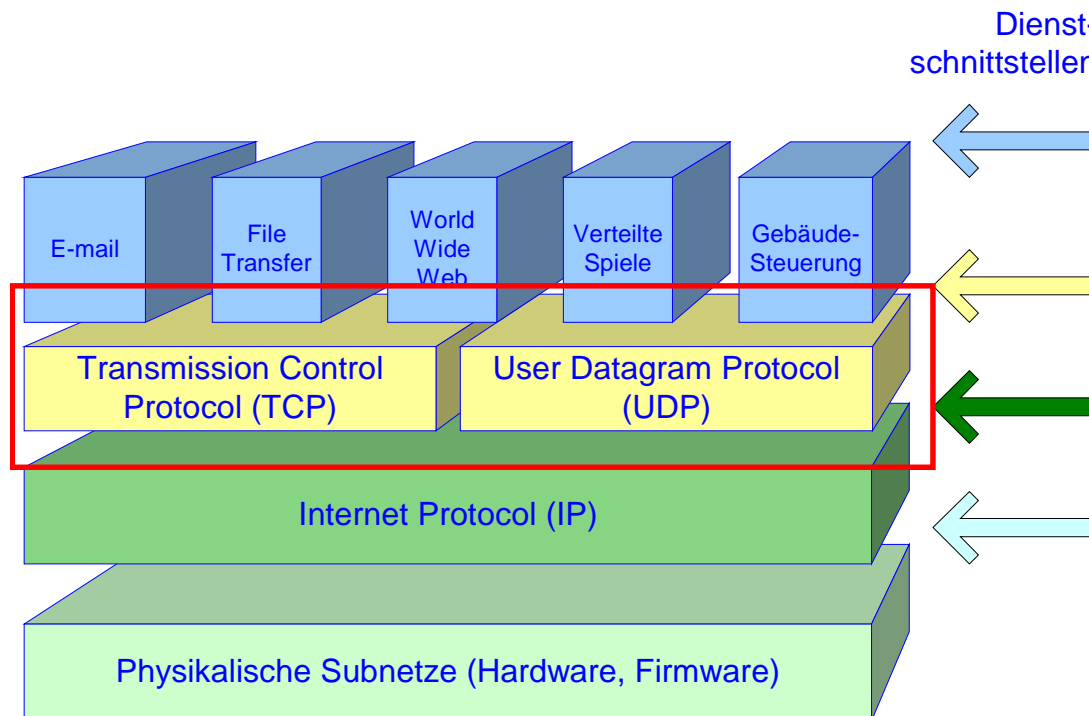
**Transmission Control Protocol (TCP) User  
Datagram Protocol (UDP)  
Die Socket-Schnittstelle**

# Rolle der Transportschicht im OSI-Referenzmodell



# Einführung in TCP

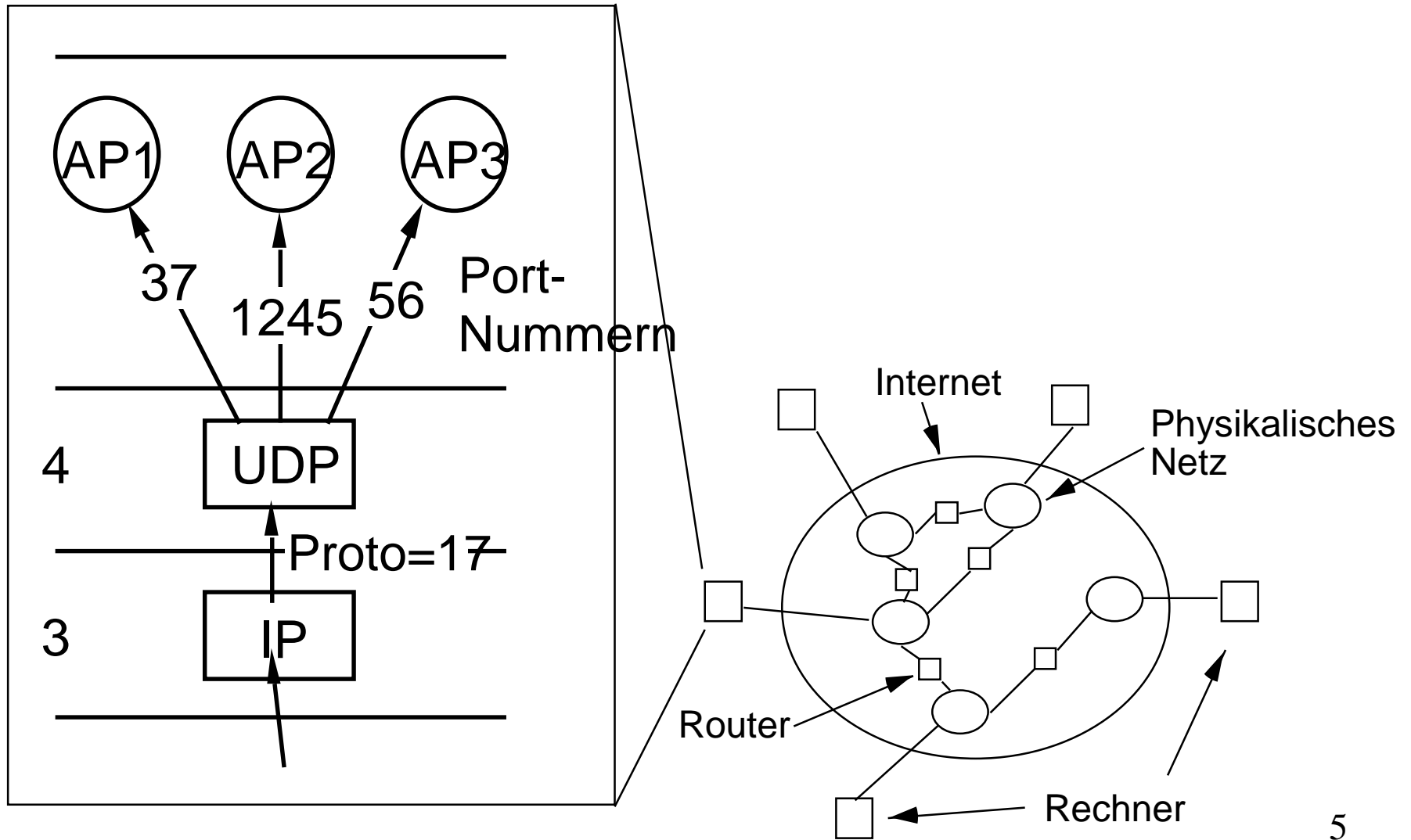
- **TCP implementiert ein verbindungsorientiertes, zuverlässiges Transport- Protokoll, aufbauend auf dem IP-Dienst.**



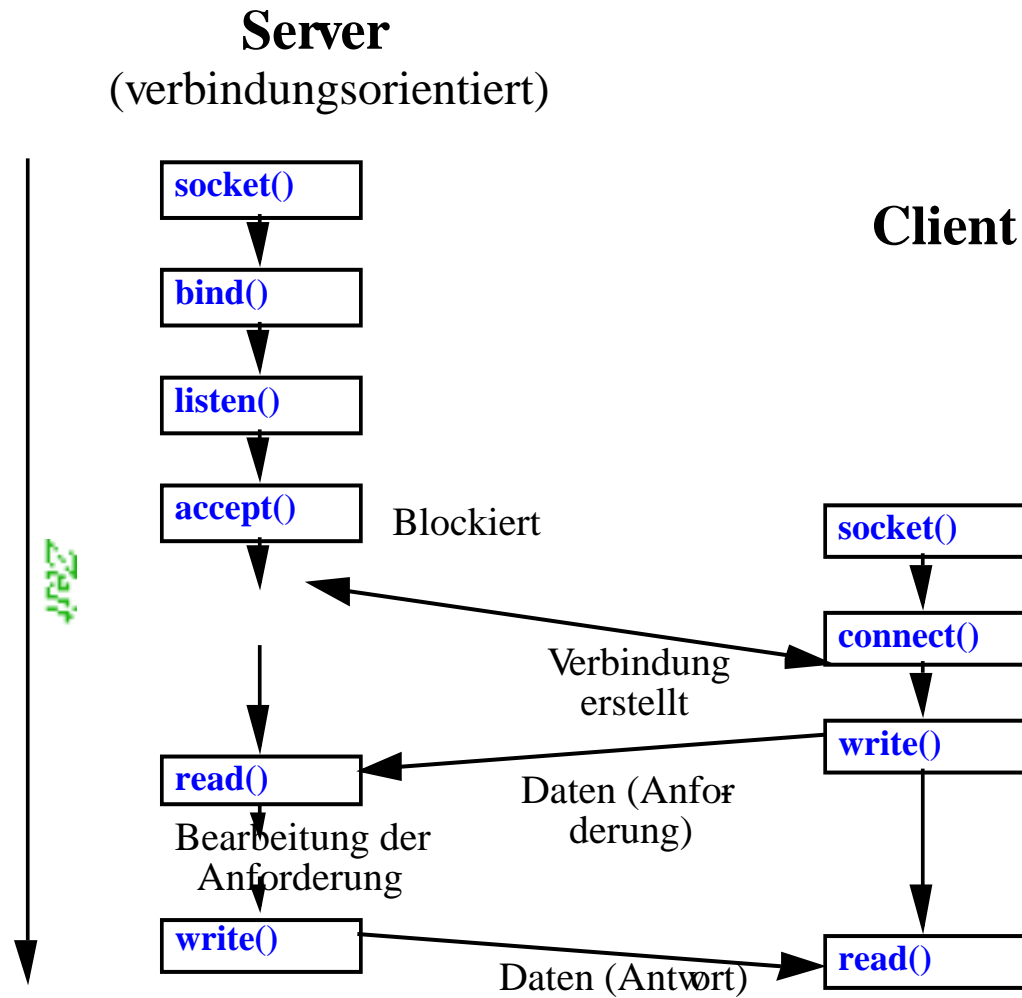
# Funktionen eines Transportdienstes

- Verbindungsaufbau (falls verbindungsorientiert)
- Datentransfer
  - normale Daten / Daten mit Priorität, Unterbrechungssignale
  - strom- oder paketorientiert
  - Fehlerbehandlung, Flusssteuerung
- Verbindungsabbruch
  - durch den Benutzer
  - durch den Dienstbringer, d.h. die Transportschicht
- Adressierung des "Benutzers" der Transportschicht, d.h. des Anwendungsprozesses
- Managementfunktionen
- Programmierschnittstelle (API)

# Adressierung von Anwendungsprozessen: Beispiel TCP/IP - Portnummern



# Verbindungserstellung (socket-Schnittstelle)



# Zuweisung von Portnummern

- Passive Seite: Server bindet sich an einen bestimmten Port (*bind()*)
- Aktive Seite: Client sendet Anfrage oder erstellt eine Verbindung zum Port des Servers (*connect()*).
- Portnummern können Anwendungsdiensten *statisch* zugeordnet sein: Eintrag in einer Datenbank, die den durch Server erbrachten Diensten bestimmte Ports zuordnet (*/etc/services*)
- Dynamische Zuordnung via Verzeichnisdienst, Nameserver (lokal oder verteilt) möglich.

# Koordination der global zugeordneten Ports

- Internet Assigned Numbers Authority (IANA):  
Zuständig für Vergabe von Konstanten in TCP/  
IP- Protokollen (port numbers, protocol numbers,  
...)
- Bereich 0.. 1023: Für globale "well known" ports,  
kontrolliert von der IANA
- Bereich 1024 .. 65535: Frei für dynamische  
Allozierung durch Prozesse oder für statische  
Allozierung mit lokaler Bedeutung
  - Registrierung durch IANA ist optional
- Aktuelle globale / statische Zuordnungen:  
<ftp://ftp.isi.edu/in-notes/iana/assignments>



# Well-known port numbers: /etc/services (Auszug)

# Note that it is presently the policy of IANA to assign a single well-known  
# port number for both TCP and UDP; hence, most entries here have two entries  
# even if the protocol doesn't support UDP operations.  
# Updated from RFC 1700, "Assigned Numbers"

echo	7/tcp		
echo	7/udp		
discard	9/tcp	sink null	
discard	9/udp	sink null	
ftp-data	20/tcp		
ftp	21/tcp		
telnet	23/tcp		
smtp	25/tcp	mail	
time	37/tcp	timserver	
time	37/udp	timserver	
nameserver	42/tcp	name	# IEN 116
whois	43/tcp	nickname	
domain	53/tcp	nameserver	# name-domain server
domain	53/udp	nameserver	

# Typischer Ablauf bei dynamisch zugeteilten Ports

1. Server-Prozess signalisiert Bereitschaft, eine Kommunikationsbeziehung entgegenzunehmen; erhält mit *bind()* seinen Port zugeteilt.
2. Server-Prozess trägt sich mit seinem Namen und dem Port in das Verzeichnis ein.
3. Client sucht im Verzeichnis nach dem Namen des Server-Prozesses und erhält dessen Port.
4. Client erstellt eine Verbindung (TCP) oder sendet ein Anfrage-Paket (UDP) zum Server-Prozess (*connect()*).
5. Server nimmt Verbindung an (TCP) oder sendet ein Antwortpaket (UDP).

# Eigenschaften des Transmission Control Protocol (TCP)

- verbindungsorientiert
- Vollduplex-Verbindung
- stellt eine “byte pipe” zur Verfügung - unstrukturierter Datenstrom
- Sliding Window-Protokoll
- Folgenummern sind Byte-nummern
- Maximale Fenstergrösse  $2^{16}$  Bytes
- Variable Grösse des Sendefensters bestimmt durch das Maximum von:
  - Angabe des Empfängers (receiver window size)
  - Congestion window size, abhängig von einer lokalen Schätzung der Netzbelastung -> “Slow Start” Algorithmus

# Basismechanismen von TCP

- Segmente variable Längen; maximale Segmentgrösse bei der Verbindungserstellung festgelegt.
- Jedes Segment hat eine Folgenummer, die seine Position im Datenstrom in Bytes spezifiziert.
- Abgesendete Segmente müssen innerhalb einer bestimmten Zeit bestätigt werden (adaptiv geschätzte Round Trip Time).
- Bestätigungen werden verzögert gesendet (ca. 200 ms)
- Jedes Segment hat eine Ende-zu-Ende-Prüfsumme; fehlerhaft empfangene Segmente werden ignoriert.
- Der Empfänger ordnet empfangene Segmente entsprechend ihrer Folgenummer; Duplikate werden ignoriert.

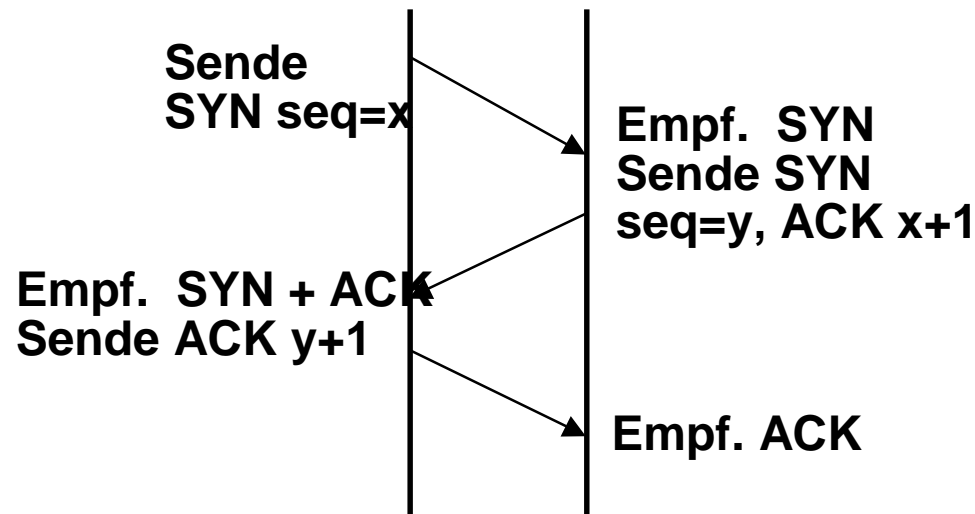
# TCP-Segmentformat

0	4	10	16	24	31
<b>Port des Senders</b>			<b>Port des Empfängers</b>		
<b>Sequenznummer im Bytestrom des Senders</b>					
<b>Bestätigungsnummer (ACK in Gegenrichtung)</b>					
<b>HLEN</b>	<b>Reserviert</b>	<b>Code Bits</b>	<b>Grösse des Empfängerfensters</b>		
<b>Prüfnummer (auch über Daten)</b>			<b>Dringlichkeitszeiger</b>		
<b>Optionen (falls vorhanden)</b>				<b>"Padding"</b>	
<b>Daten</b>					
...					

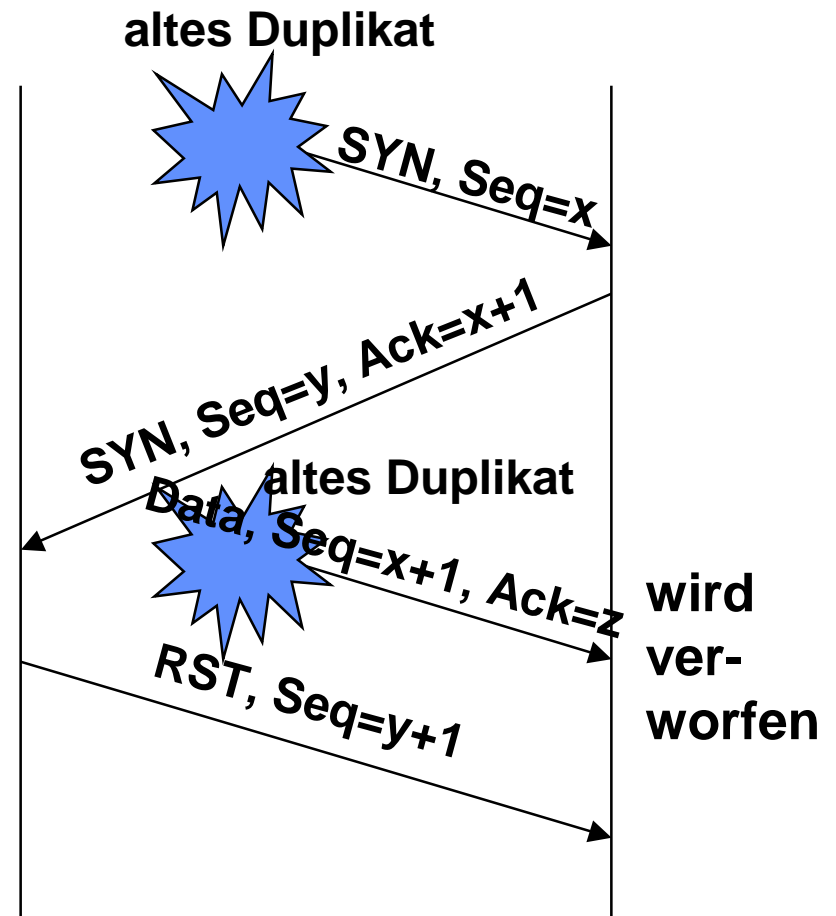
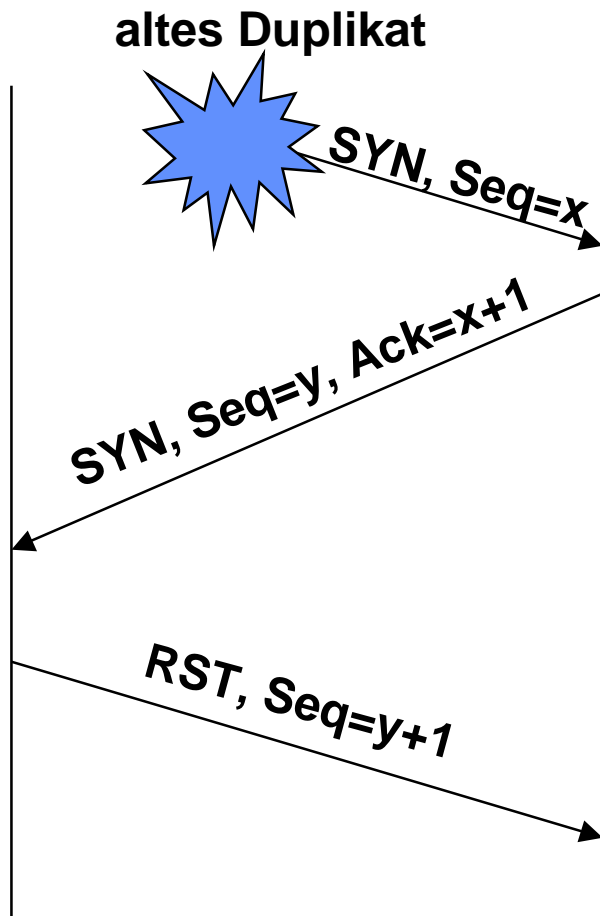
- Code Bits: URG, ACK, PSH, RST, SYN, FIN
- Dringlichkeitszeiger: zeigt auf das Ende der dringenden Daten im TCP-Datenfeld.

# Verbindungsaufbau

- Aktives Öffnen einer Verbindung (SYN)
- Passive Seite nimmt eine Verbindung auf einer bestimmten Port-Nummer entgegen
- Die initialen Sequenznummern werden auf jeder Seite zufällig gewählt und bestätigt.
- 3-fach-Handshake (nötig wegen des unzuverlässigen Dienstes von IP):

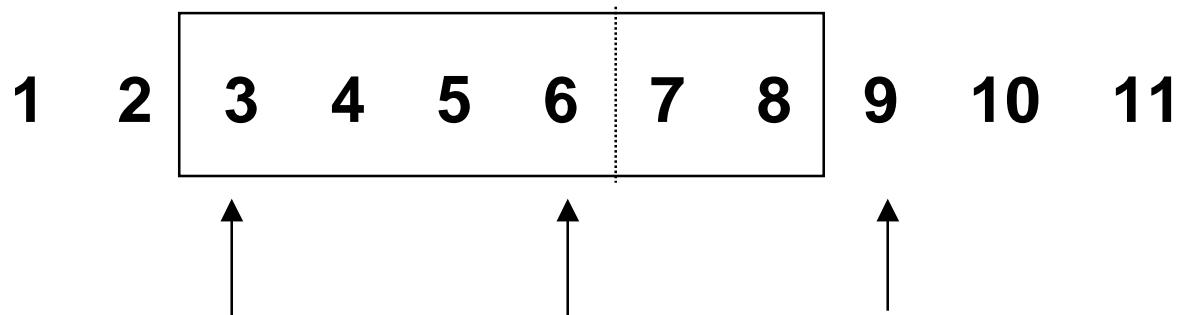


# Verbindungsaufbau, zwei Fehlerszenarien



# Segmente, Datenströme und Sequenznummern

- TCP betrachtet einen Datenstrom als Sequenz von Bytes, die für die Übertragung in TCP-Segmente eingeteilt werden. Jedes Segment wird dann in der Regel auf ein IP-Paket abgebildet. (Grösse eines Segmentes bei lokaler Übertragung gemäss physikalischem Netz, sonst 536 Bytes)
- TCP verwendet ein "sliding window" Protokoll, um möglichst effizient Daten zu übertragen, und Flusskontrolle zu ermöglichen. Bei einer Vollduplex-Verbindung müssen insgesamt 4 Fenster verwaltet werden.



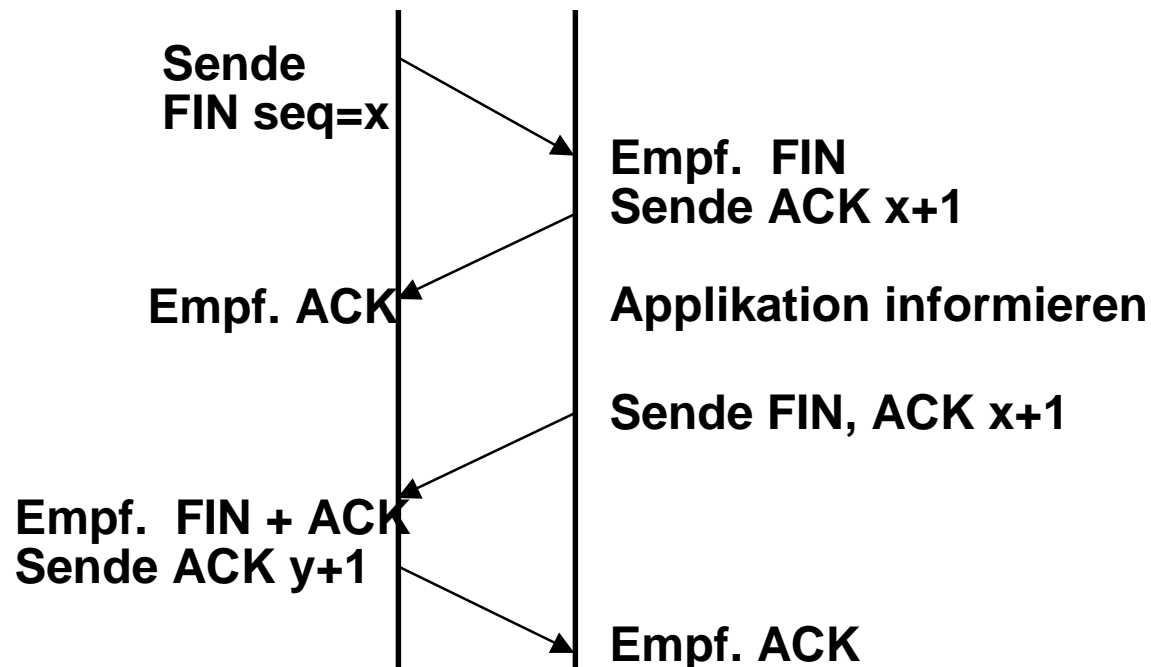


# Variable Fenstergrößen

- Die Fenstergröße im TCP "sliding window" Protokoll kann variiert, d.h. an den "Füllstand" des Netzes bzw. des Empfängers angepasst, werden.
- Flusssteuerung
  - Jedes Bestätigungspaket enthält einen "window advertisement" Wert, in dem der Empfänger angibt, für wieviele weitere Pakete er noch freie Kapazität hat (das Fenster kann also grösser oder kleiner werden).
- Verkehrssteuerung
  - Jacobsen's "slow start" Algorithmus variiert die Grösse des Sendefensters, um die Senderate an die Netzbelastung anzupassen (s. Folie 18).

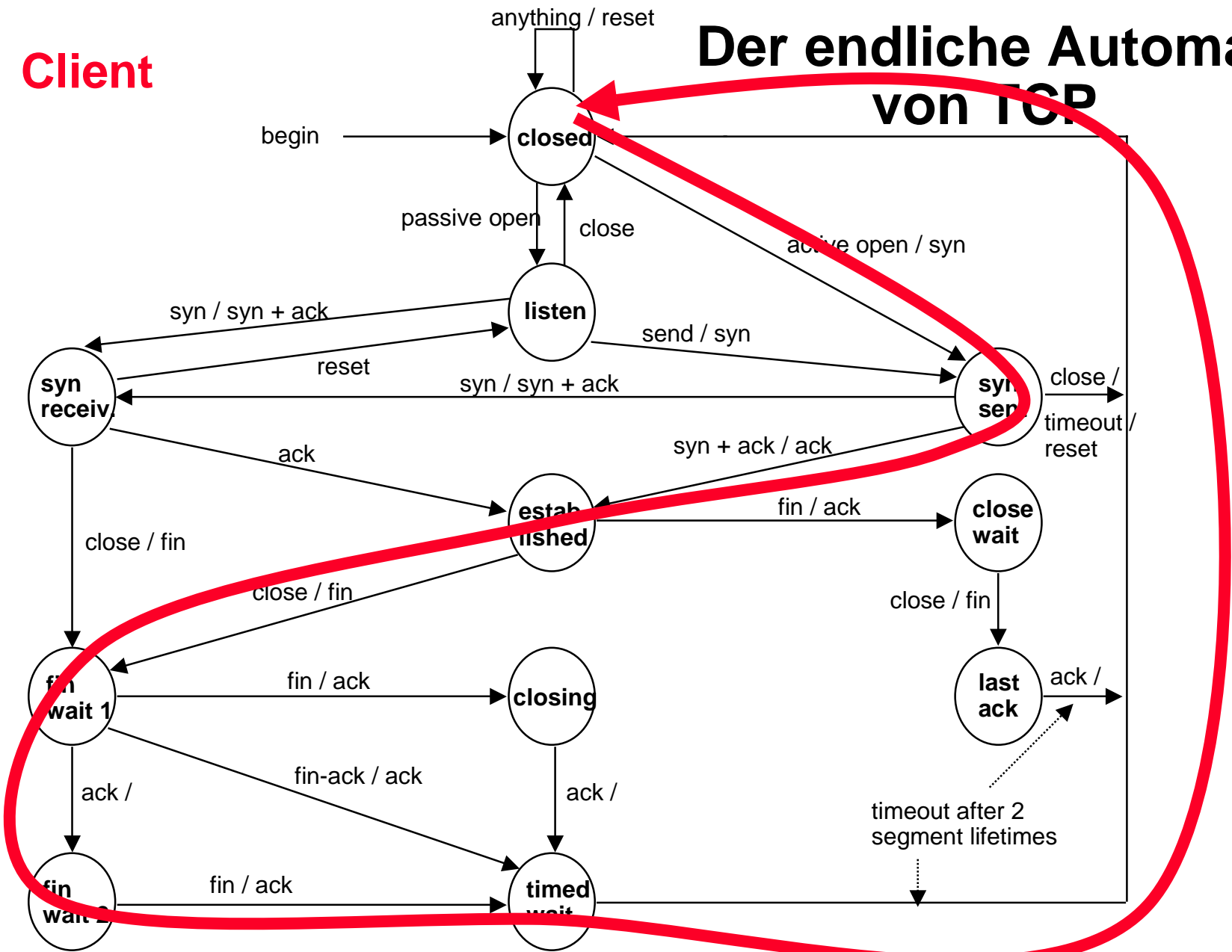
# Abbau einer TCP-Verbindung

- Aktive Seite (links) schliesst Verbindung mit FIN-Flag
- Neue Daten werden nicht mehr übertragen, von rechts ankommende Daten werden jedoch noch bestätigt.
- 4-fach-Handshake; jede Seite wird separat beendet (TCP half close)



**Client**

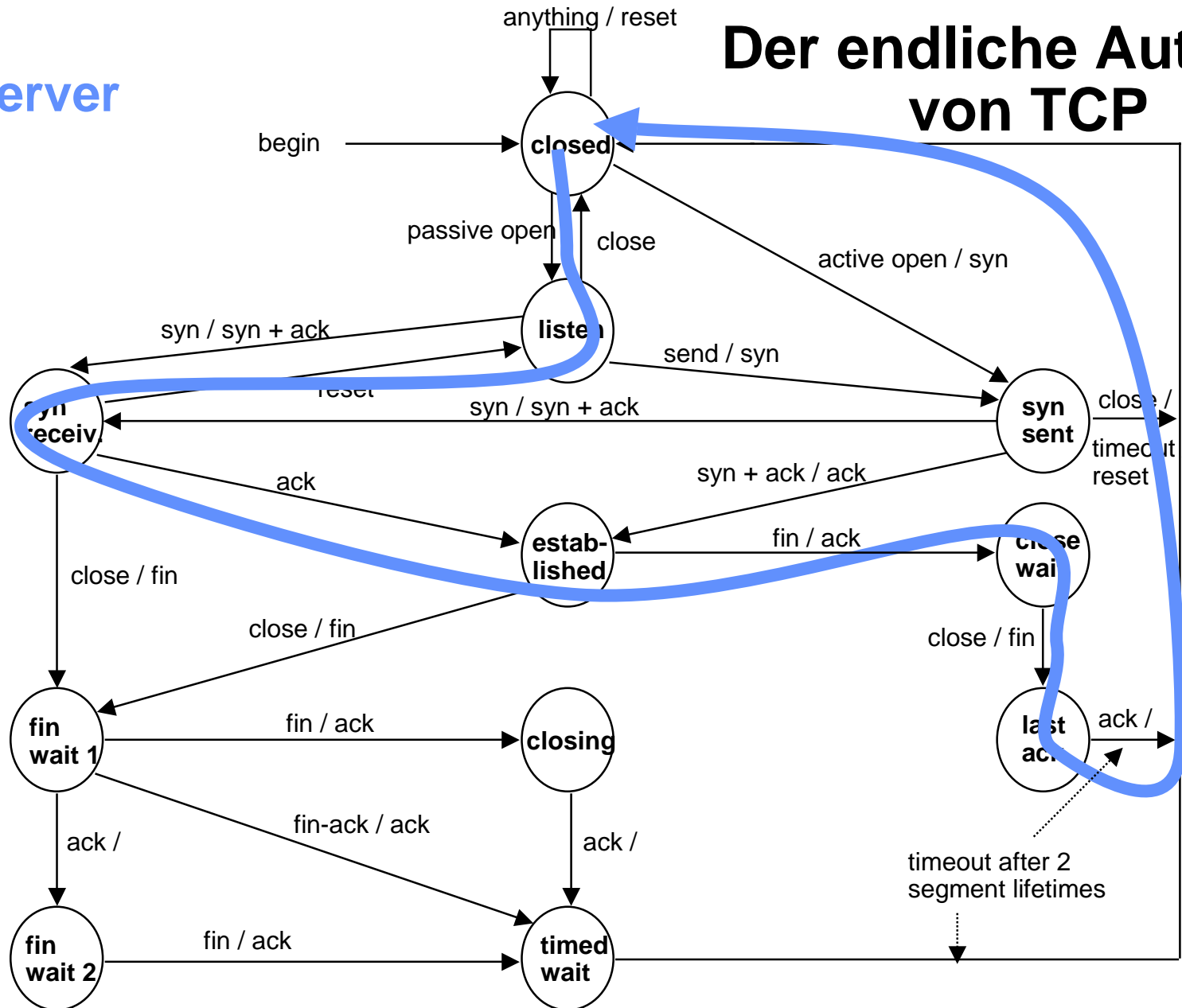
# Der endliche Automat von TCP



aus: D. Comer: Internetworking with TCP/IP, Volume 1, Prentice Hall, 1995, p. 220

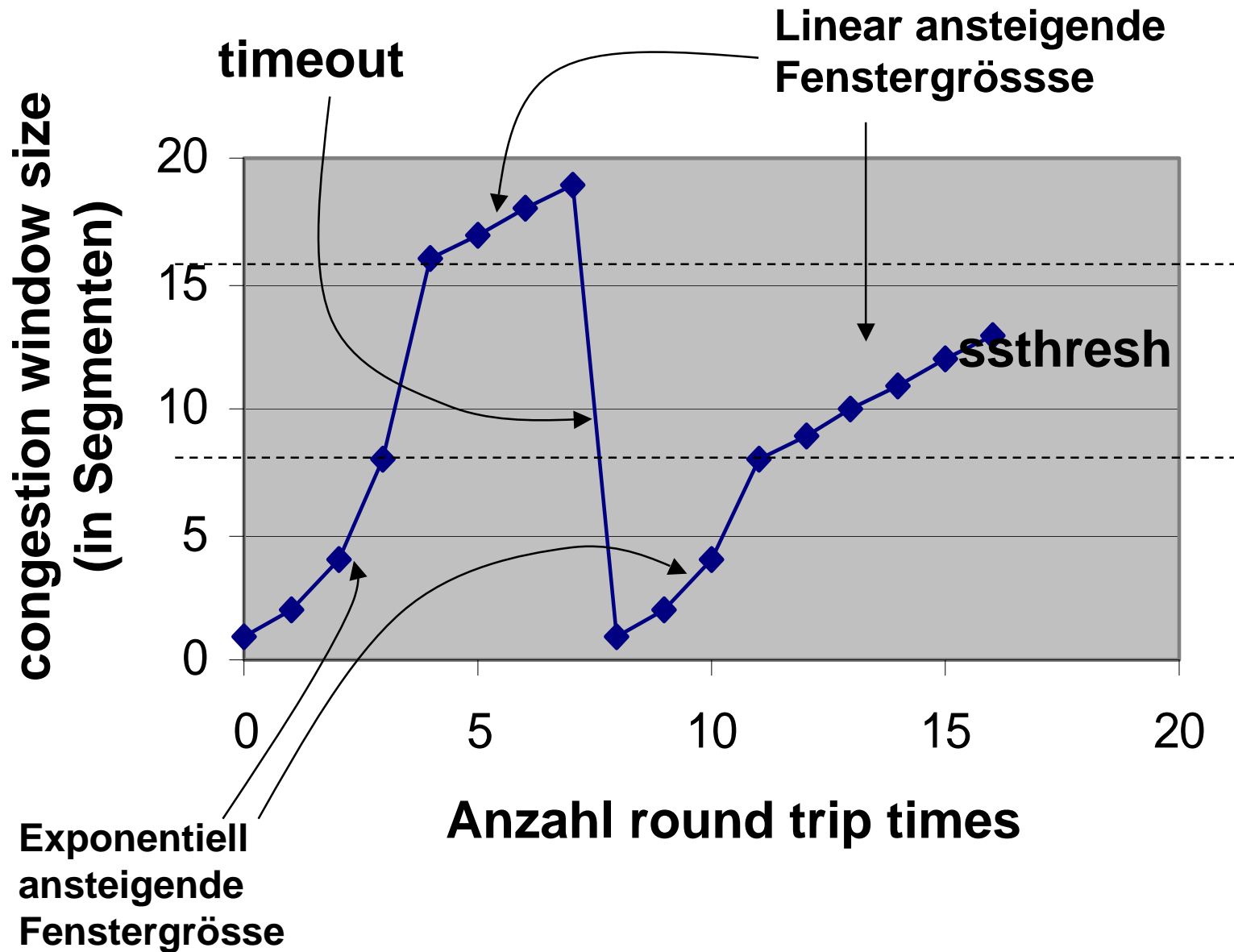
Server

# Der endliche Automat von TCP

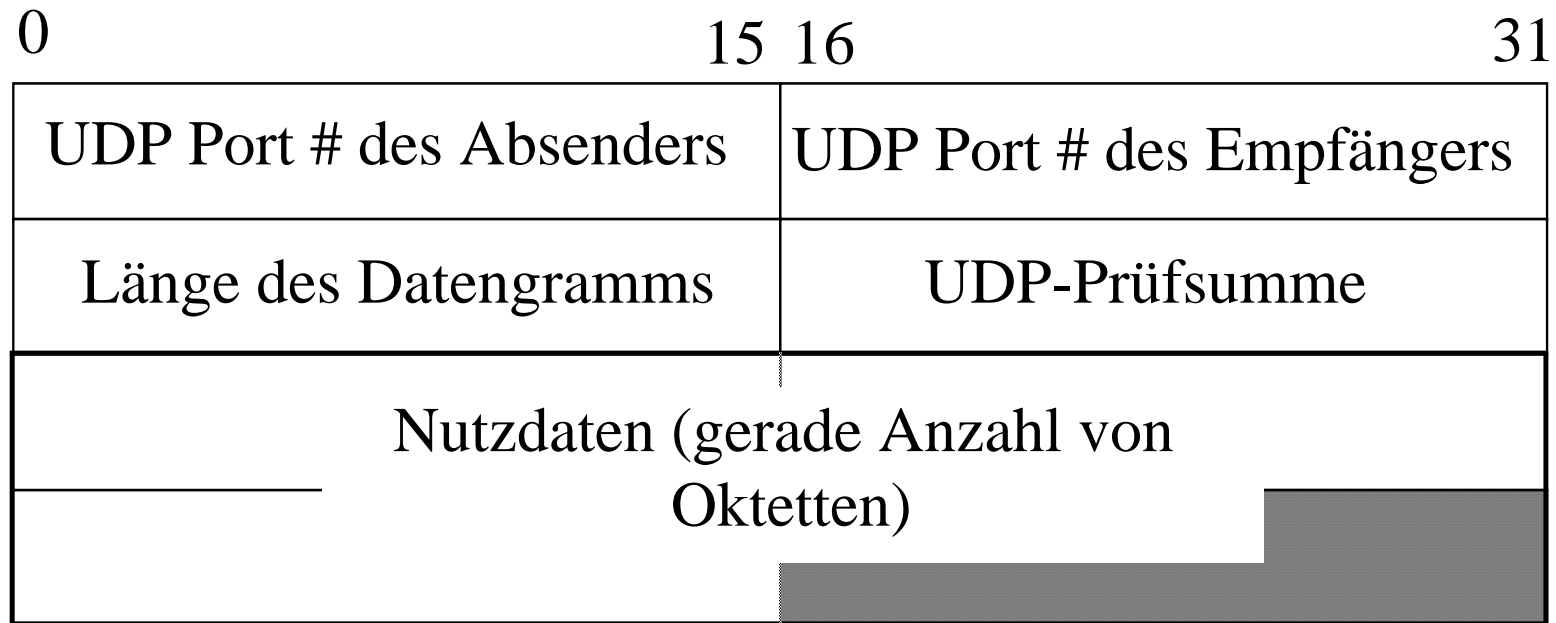


aus: D. Comer: Internetworking with TCP/IP, Volume 1, Prentice Hall, 1995, p. 220

# Slow Start Algorithmus



# User Datagram Protocol (UDP)



Prüfsummenberechnung schliesst Pseudo-Header mit ein (gilt für UDP und TCP).

# Die Socket-Schnittstelle

- De-facto-Standard für TCP/IP Programmierschnittstelle
- Zugang zu TCP, UDP und (eingeschränkt) IP
- Unterstützung verschiedener Protokolle
  - Protocol family
  - Address family
- Abstraktion für Kommunikationsendpunkte
  - sockets
- ... mit verschiedenen Kommunikationseigenschaften
  - socket types (stream socket, datagram socket)
- Benennung/Adressierung von Kommunikationsendpunkten
  - name binding

# Socket-Datenstrukturen für die Adressierung

allgemein: 

family	Adresse (variabel lang)
--------	-------------------------

für TCP/IP: 

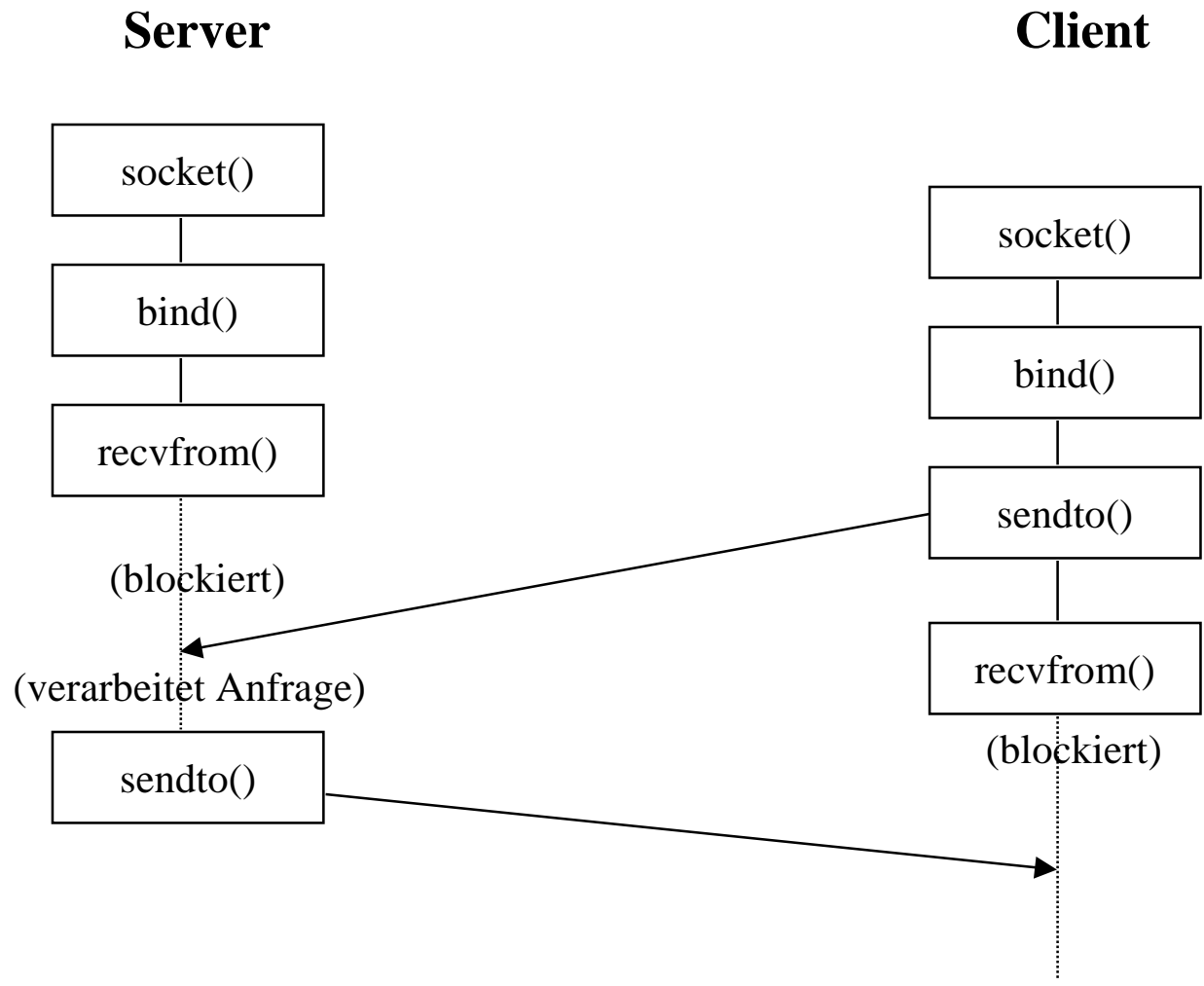
AF-INET	Port	IP-Adresse
---------	------	------------

Für UNIX-  
Filesystem: 

AF-UNIX	UNIX-Pfadname
---------	---------------

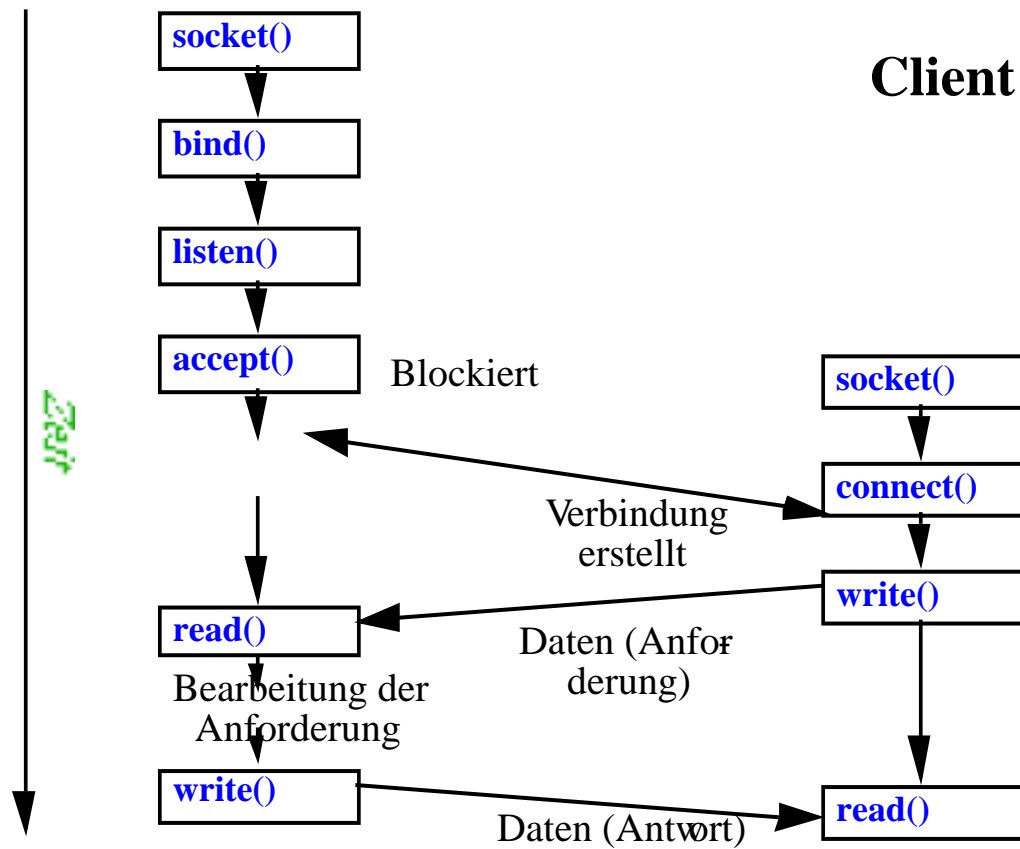


# Verbindungslose Kommunikation über Sockets



# Verbindungserstellung

**Server**  
(verbindungsorientiert)



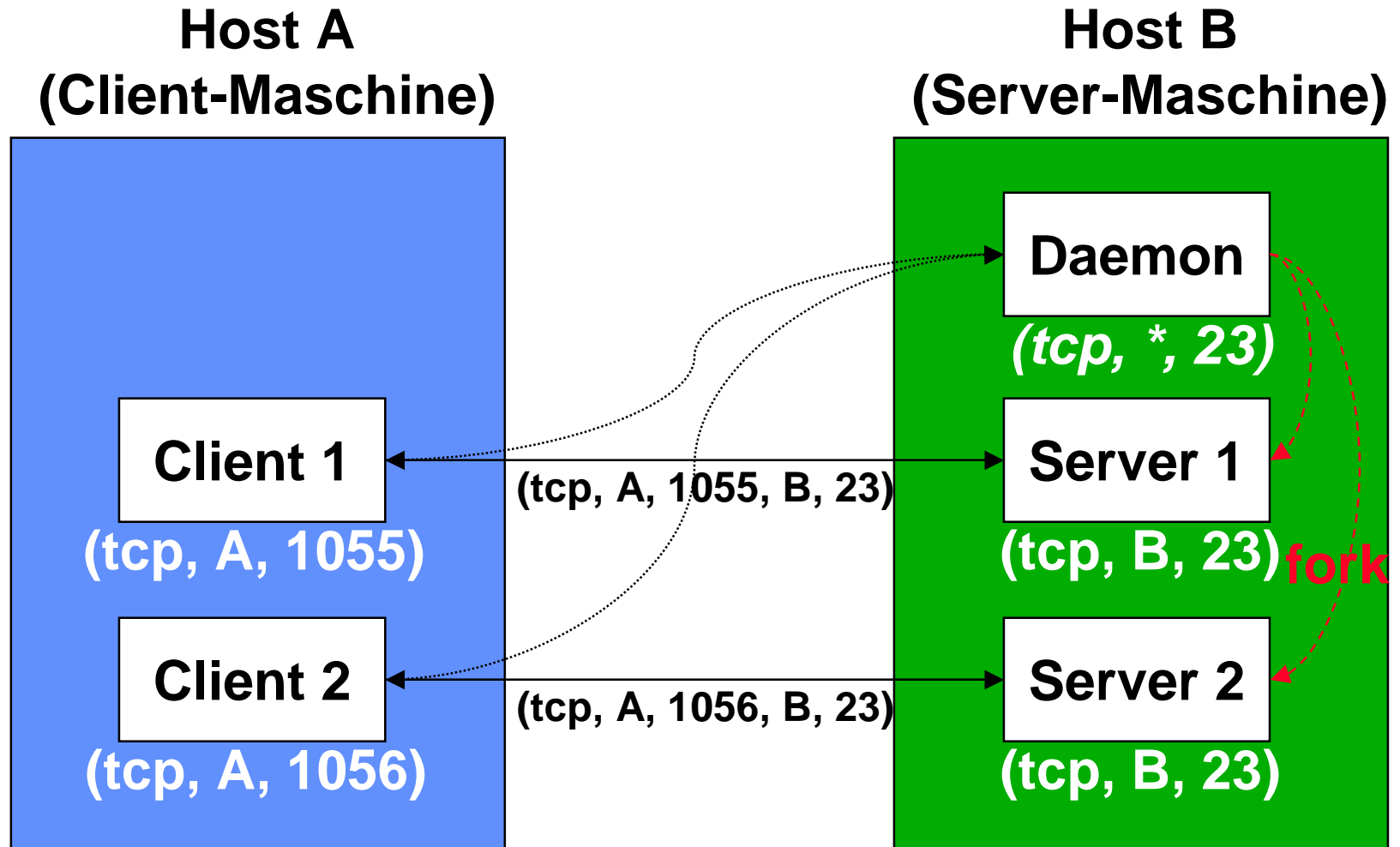
# Verbindungen und Verbindungsendpunkte

Eine TCP-Verbindung wird durch ein Paar von Adressen und Port-Nummern identifiziert (Verbindungsendpunkte):

- IP-Adresse und Port-Nummer Host A
- IP-Adresse und Port-Nummer Host B

Jede Verbindung wird durch ein Paar von Verbindungsendpunkten eindeutig identifiziert -> mehrere Verbindung zwischen den gleichen Hosts sind dadurch gleichzeitig möglich.

# Identifikation von Verbindungen



## Weiterführende Literatur

- **Postel, Jon, "Transmission Control Protocol - DARPA Internet Program Protocol Specification", RFC 793, Network Information Center, SRI International, Menlo Park, Calif., September 1981**
- **W. Richard Stevens, "UNIX Network Programming", Prentice Hall, Engelwood Cliffs, NJ, 1990, ISBN 0-13-949876-1**
- **"man 4 tcp"**