Netzwerke und Datenkommunikation
NDK 02-050
Layer-4 UDP & TCP

rolf.schmutz@fhnw.ch

FHNW

18. April 2012

n|w

Ziele

- Sie kennen die Transportschichtprotokolle UDP und TCP und geeignete Anwendungen
- Sie kennen die Software-Abstraktion “Socket” und das dazugehörige demultiplexing auf dem System
- Sie können Verbindungen auf dem System identifizieren

n|w

Aufgaben der Schichten

- Layer-4: Prozess-zu-Prozess[1]
- Layer-3: Host-zu-Host[2]
- Layer-2: Host-zu-*lokalem-Host/Router*

[1] Programm-zu-Programm, z.B. Webbrowser-zu-Webserver
[2] end-to-end

Layer-4: Transportschicht 1/2

- Die Schicht 4 führt eine Abstraktion für Kommunikationskanäle ein, die die unterliegende Paketschicht verbirgt

1. Es gibt einen verbindungslosen "Telegrammdienst" (UDP) für kurze und/oder "einweg" Meldungen[3]
2. ... und einen verbindungsorientierten, bidirektionalen Dienst mit garantierter Sequenz[4]

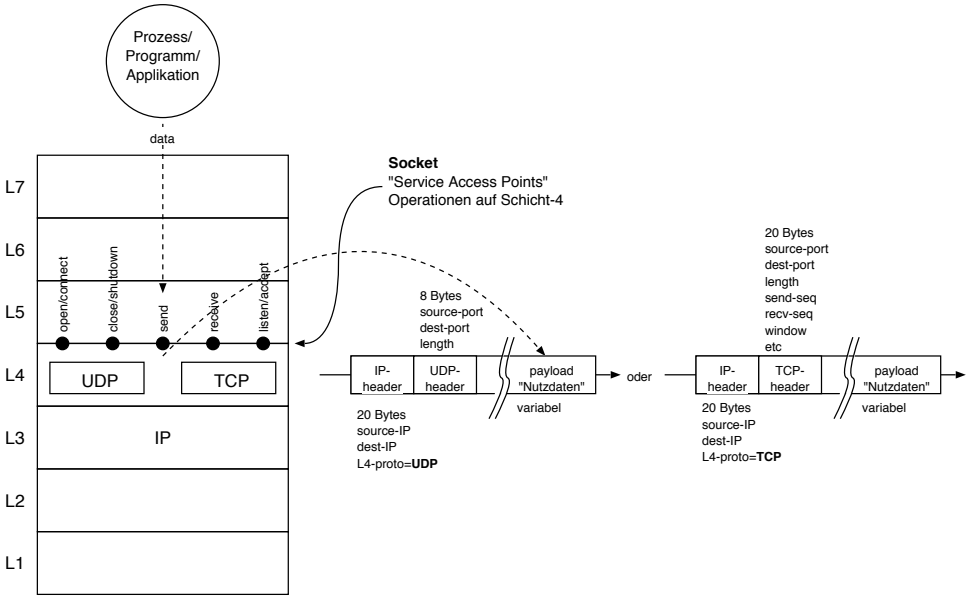
Abstraktion

beides sind "Illusionen", die die paketorientierte Arbeitsweise von IP verbergen
auf beiden Endgeräten muss ein Verbindungsstatus gepflegt werden

[3] . . .ziemlich genaue Analogie
[4] analog z.B. einer Telefonverbindung

Layer-4: Transportschicht 2/2

- Server: SAPI *passive-open* `accept` (warte auf Anfragen)
- Client: SAPI *active-open* `connect` (startet eine Anfrage)
- Beide: SAPIs *send*, *receive*, *close* (Datenkommunikation)



UDP: User Datagram Protocol

- kann für kurze Einwegmeldungen[5] wie z.B. Systemlog[6]
- oder auch für bidirektionale Konversation[7] wie z.B. DNS/Verzeichnisdienst[8] verwendet werden
- es ist Aufgabe der Applikation[9] Antwort-Datagramme zu senden – UDP selbst "kennt" das jeweilige Schicht-7 Protokoll nicht
- unterstütz *Multicasting* – senden von Daten an viele Hosts gleichzeitig
- die Bezeichnung für eine Dateneinheit (Telegramm) ist *datagram*

Telegrammdienst

Die jeweiligen Applikationen/Programme[a] müssen die eventuelle Quittierung oder Wiederholung von Meldungen selber sicherstellen

[a] *client* und *server*

[5] d.h. ohne Bestätigungsmeldung, best-effort

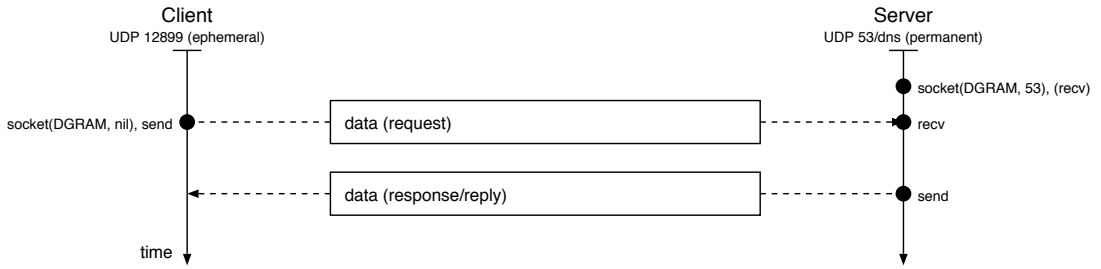
[6] Windows: *Eventlog*, Transkript

[7] *request* und *reply*

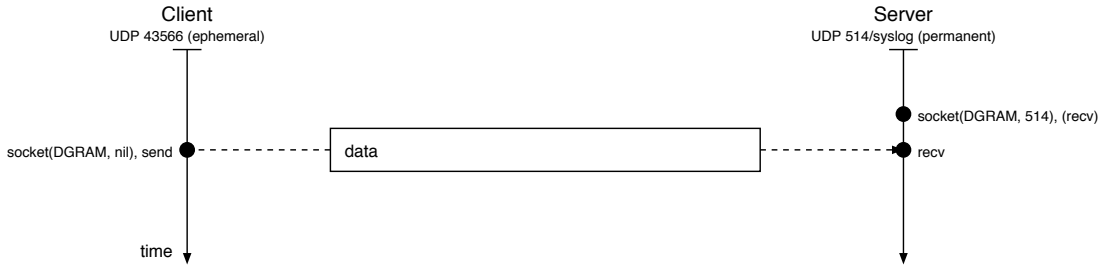
[8] . . .damit Sie `www.eff.org` eingeben können und DNS findet dann die IP-Adresse `64.147.188.3` dazu

[9] Prozess/"Programm", auf Server- und Client-Seite

UDP Communication



OR



TCP: Transmission Control Protocol 1/5

- Zweiweg[10] verbindungsorientierte Kommunikation
- garantierte Sequenz der Daten[11]
- verlorene Pakete werden neu gesendet
- *Flusskontrolle* – Empfänger kann "stop" oder "langsamer senden" verlangen
- die Bezeichnung für eine Dateneinheit ist *segment* – allerdings ist die Abstraktion für die Software ein *stream* (Datenstrom)

Verbindungsorientierter Dienst

Transparente[a] bidirektionale (Richtungsgetrennt) Verbindung[b]

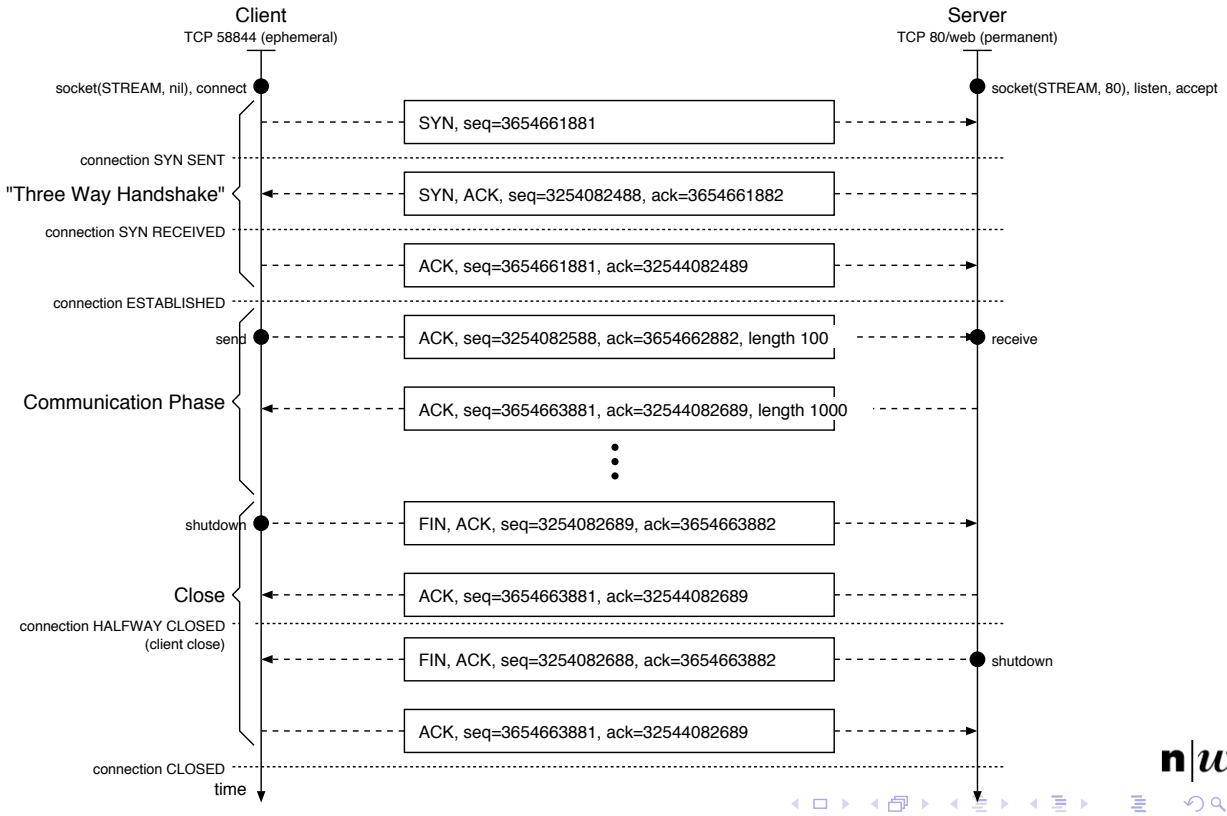
[a] d.h. die Client- und Server-Applikationen kümmern sich nicht um Paketwiederholungen, Sequenz, etc

[b] **das ist eine nur eine "Illusion" – die darunterliegende Schicht IP ist nicht Verbindungsorientiert**

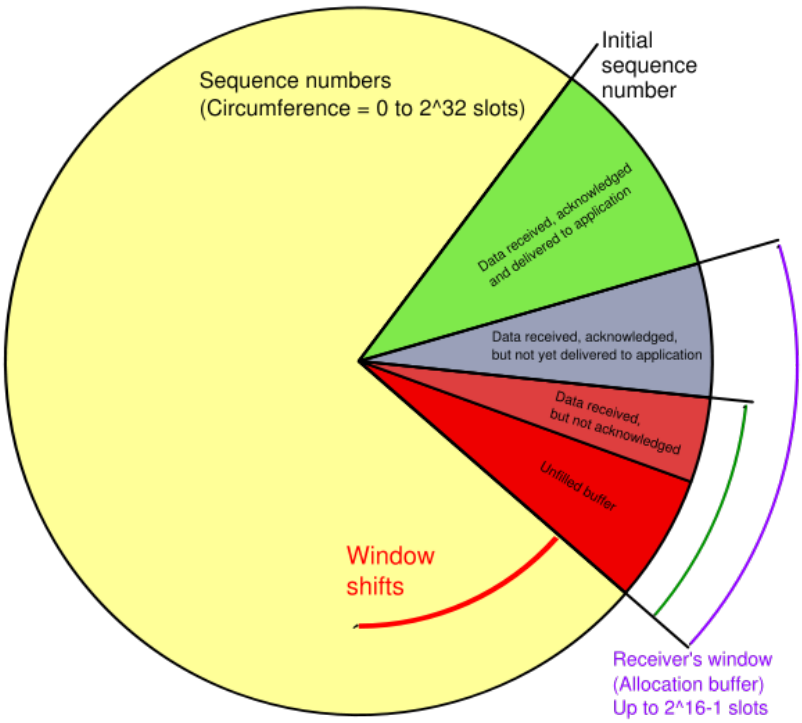
[10] bidirektional

[11] auch wenn sich Pakete im Internet "überholen"

TCP Handshake, Session 2/5

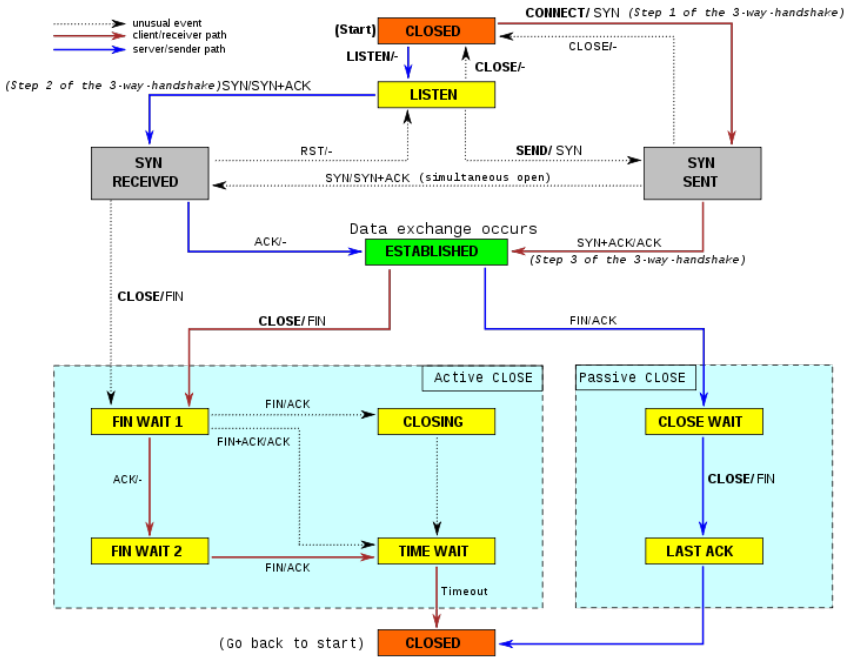


TCP Sequence, Window[12] 3/5



[12] http://upload.wikimedia.org/wikipedia/commons/thumb/d/db/Tcp.svg/600px-Tcp.svg.png

TCP Stati[13] 4/5



[13] http://upload.wikimedia.org/wikipedia/commons/thumb/a/a2/Tcp_state_diagram_fixed.svg/796px-Tcp_state_diagram_fixed.svg.png

TCP tcpdump[14] (edited) 5/5

```
--- three-way handshake ---
19:09:56.361262 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [S], seq 1704735491, win 65535, length 0
19:09:56.384815 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [S.], seq 4146040110, ack 1704735492,
                win 5792, length 0
19:09:56.384871 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [.], ack 4146040111, win 33304, length 0

--- communication phase ---
19:10:00.891376 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [P.], seq 1704735492:1704735509,
                ack 4146040111, win 33304, length 17
19:10:00.915173 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [.], ack 1704735509, win 46, length 0
19:10:06.987161 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [P.], seq 1704735509:1704735533,
                ack 4146040111, win 33304, length 24
19:10:07.010497 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [.], ack 1704735533, win 46, length 0
19:10:07.531102 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [P.], seq 1704735533:1704735535,
                ack 4146040111, win 33304, length 2
19:10:07.555122 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [.], ack 1704735535, win 46, length 0
19:10:07.555127 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [P.], seq 4146040111:4146040348,
                ack 1704735535, win 46, length 237
19:10:07.555182 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [.], ack 4146040348, win 33185, length 0

--- shutdown ---
19:10:12.792188 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [F.], seq 4146040348,
                ack 1704735535, win 46, length 0
19:10:12.792244 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [.], ack 4146040349, win 33304, length 0
19:10:12.792341 IP 10.202.5.121.63505 > 188.40.65.199.80: Flags [F.], seq 1704735535,
                ack 4146040349, win 33304, length 0
19:10:12.815841 IP 188.40.65.199.80 > 10.202.5.121.63505: Flags [.], ack 1704735536, win 46, length 0
```

[14] sudo tcpdump -v -nnn -S -i en1 tcp port 80 and ip host zaphod und dann telnet zaphod 80

Kommunikationsendpunkt "Socket"

- am weitesten verbreitete Software-Abstraktion eines Kommunikationsenpunkts "Berkeley Socket"[15]
- ein Verbindungsversuch auf *closed ports*[16] wird bei TCP mit einem RESET bei UDP mit einem ICMP-Port-Unreachable beantwortet
- der Kommunikationskanal wird von der Software wie eine Datei angesprochen[17]

```
root@zaphod:~# netstat -tunap4
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:993             0.0.0.0:*               LISTEN      7720/imap-login
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      19994/lighttpd
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      32004/named
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      3097/sshd
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      1602/master
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      19994/lighttpd
tcp        0      0 188.40.65.199:22        77.56.89.75:45753       ESTABLISHED 18655/sshd: tunnel
tcp        0      0 188.40.65.199:22        212.60.51.243:40469     ESTABLISHED 5604/sshd: tunnel
tcp        0      0 188.40.65.199:22        212.60.51.243:46973     ESTABLISHED 24007/sshd: tunnel
tcp        0   3248 188.40.65.199:22        77.56.89.75:52550       ESTABLISHED 24992/sshd: rschmutz
tcp        1      0 188.40.65.199:80        77.56.89.75:51856       CLOSE_WAIT  19994/lighttpd
udp        0      0 188.40.65.199:53        0.0.0.0:*                           32004/named
udp        0      0 188.40.65.199:123       0.0.0.0:*                           3057/ntpd
```

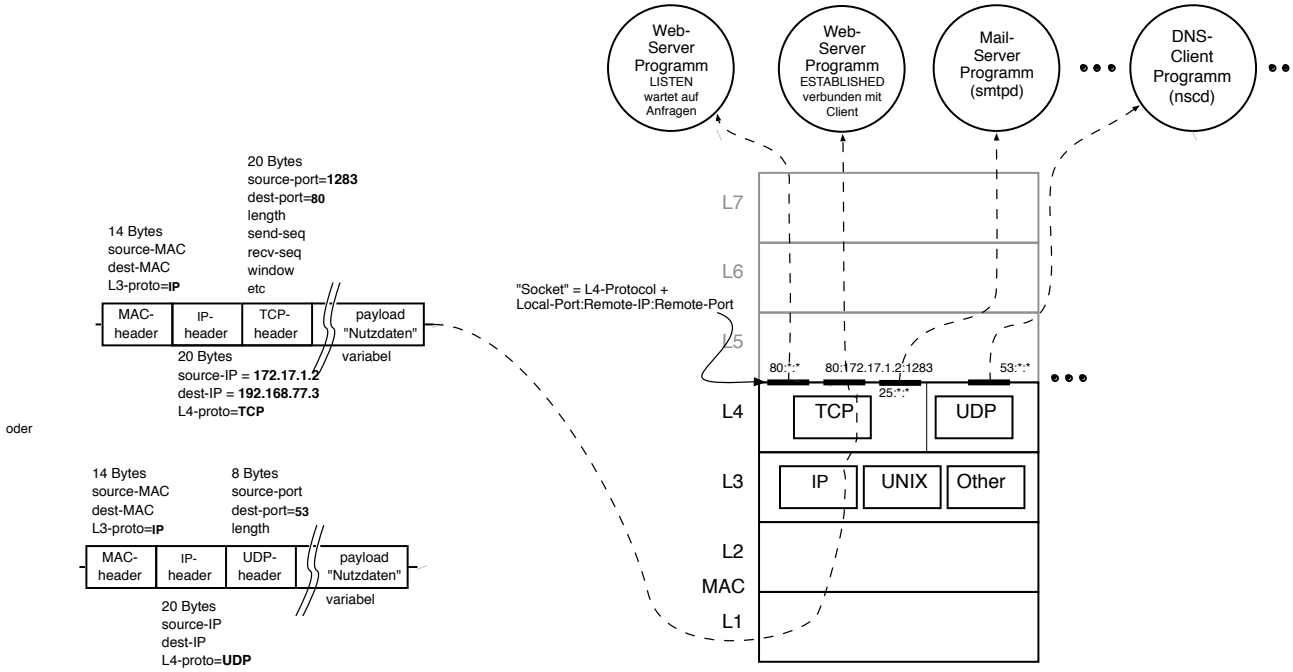
[15] von UC Berkley, BSD "Berkeley Software Distribution" UNIX

[16] kein Serverprozess

[17] *read* und *write*. Bei TCP zusätzlich *open* und *close*

Layer-4: *Demultiplexing*

für das Demultiplexing wird das 5-Tuple { `protocol`, `local-ip`, `local-port`, `remote-ip`, `remote-port` } verwendet



Socket Stati[18]

http://www-01.ibm.com/support/docview.wss?uid=isg1II12449

[18] "Statüsser"

Port Nummern[23] ≈ Dienst

- um einen bestimmten Dienst[19] anzusprechen müssen die die entsprechenden Portnummern bekannt sein
- Systemseitig werden anstatt Portnummern oft symbolische Namen benutzt[20], Windows: `C:`
- Portnummern werden von IANA[21] verwaltet
  es gibt die

1. *well-known-services*[22] 0 bis 1023
2. *registered ports* 1024 bis 49151: darin finden sich bekannte Dienste ("Server-side", *permanent*) aber auch "Client-side" (*ephemeral* Ports

[19] z.B. Web oder Mail

[20] UNIX: `/etc/services` und `getent services mail` oder `getent services 25`

[21] Internet Assigned Numbers Authorithy, http://www.iana.org/assignments/port-numbers

[22] "WKS" auch bekannt als "low-ports"

[23] http://en.wikipedia.org/wiki/TCP_and_UDP_port_numbers

Command Line Tools

- Socket Status, "offene Ports": `netstat -an` (alle sockets)
- TCP Verbindungstest: `telnet` *`host port`*

References

- Internet Standards: http://tools.ietf.org/html/rfc1280
- UDP:
  RFC http://tools.ietf.org/html/rfc768 und
  Standard http://tools.ietf.org/html/std6
- TCP:
  RFC http://tools.ietf.org/html/rfc793 und
  Standard http://tools.ietf.org/html/std7
- Socket: http://en.wikipedia.org/wiki/Internet_socket
- Port Nummern: http://en.wikipedia.org/wiki/TCP_and_UDP_port_numbers und
  http://www.iana.org/assignments/port-numbers