Netzwerk - Grundlagen

Inhalt

- Netzwerke
- TCP/IP-Schichtenmodell
- DHCP und DNS
- Netzwerkkonfiguration
- Netzwerkbefehle
- Netzwerkkonfiguration mit Puppet

Netzwerke

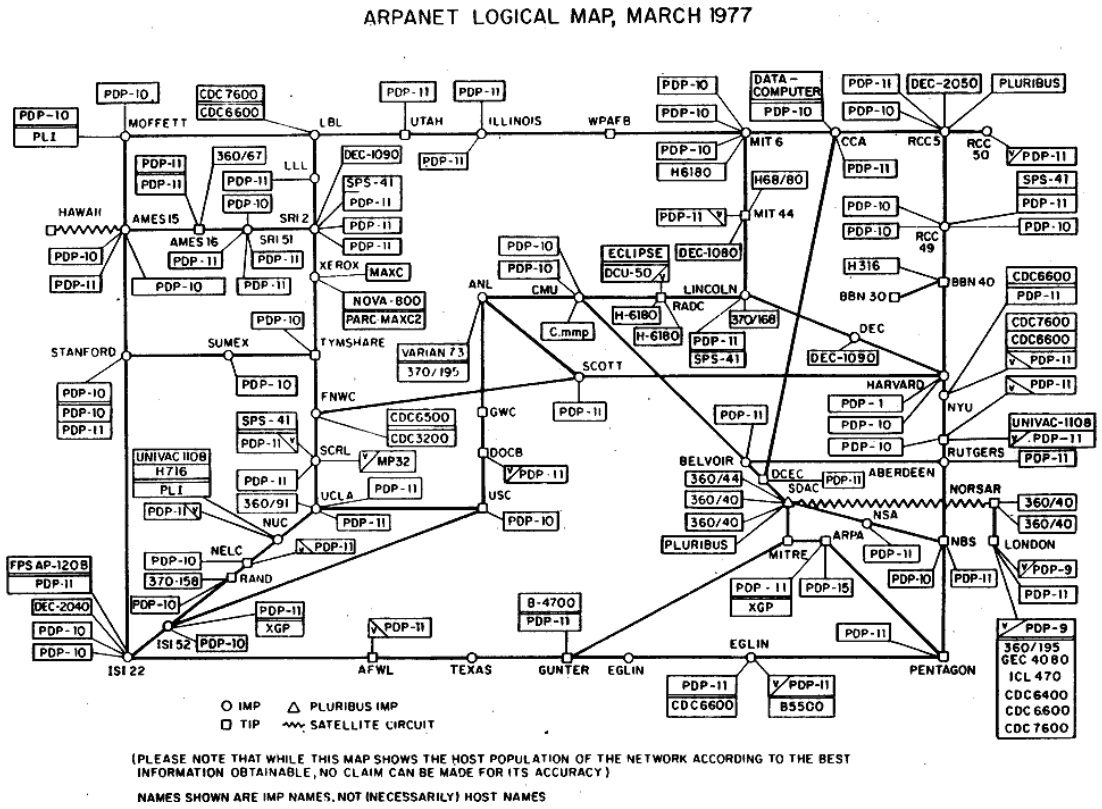
- Computernetzwerke dienen dem Austausch von Daten zwischen Computern. Die Daten können dabei über Kabel, Glasfaser, durch elektromagnetische Strahlung, ... übertragen werden (es gibt auch etwas unüblichere Übertragungsmedien - siehe RFC 1149).
- Eines der ersten großen Netzwerke war das ARPANET, der Vorläufer des heutigen Internets. Die US-Regierung (genauer, DARPA) wollte ein dezentrales und fehlertolerantes Netzwerk für die militärische Forschung schaffen.
- Anfang der 1970er Jahre wurde von Vint Cerf und Robert Kahn schließlich die Grundlage für das heutige Internet geschaffen und Anfang der 1980er Jahre löste dann das neue Internet das alte ARPANET ab.

Links: Wired - Vint Cerf: We Knew What We Were Unleashing on the World, Internet Hall of Fame

Netzwerke - Abkürzungen

- Wichtige Abkürzungen:
  - *Request For Comments* (RFC)
    Öffentliche Dokumente, die Internetstandards und -protokolle beschreiben. Einmal veröffentlicht, werden sie nicht mehr verändert (siehe auch Ubuntu-Pakete `doc-rfc*`).
  - *Internet Engineering Task Force* (IETF)
    Organisation, die sich mit der technische Weiterentwicklung des Internet befasst. Gibt die RFCs heraus.
  - *Internet Corporation for Assigned Names and Numbers* (ICANN)
    Zuständig z.B. für die Vergabe von Namen (z.B. DNS-Namen) und Adressen (z.B. IP-Adressen).
  - *Internet Assigned Numbers Authority* (IANA)
    Abteilung von ICANN, der die Vergabe von Nummern (z.B. Protokollnummern oder Port-Nummern) und Namen übernimmt.

Logische Karte von ARPANET, März 1977



Netzwerkmodelle

- Damit Computer über ein Netzwerk kommunizieren können, müssen sie sich auf eine gemeinsame Sprache einigen, d.h. es müssen **Protokolle** definiert sein, an die sich alle Beteiligten halten.
- Das OSI-Modell (*Open Systems Interconnection Model*) ist ein Referenzmodell für Netzwerkprotokolle. Es umfasst sieben Schichten und beschreibt, wie Geräte miteinandern kommunizieren können. Jede dieser Schichten erfüllt eine andere Aufgabe.
- Im OSI-Modell kommunizieren immer die Instanzen der selben Schicht über ein Protokoll miteinander, z.B. Schicht 3 auf dem ersten Gerät mit Schicht 3 auf dem zweiten Gerät.
- Die Daten werden von der obersten Schicht bis zur untersten Schicht auf dem sendenden Gerät durchgereicht (und ev. mit Meta-Informationen versehen), übertragen und auf der Empfangsseite von der untersten Schicht bis zur obersten Schicht durchgereicht (wobei die Meta-Informationen ausgelesen und entfernt werden).

TCP/IP-Modell

- Für das Internet wird ein einfacheres Netzwerkmodell verwendet, nämlich das vier Schichten umfassende **TCP/IP-Modell** (beschrieben im RFC 1122):
  - **Netzzugangsschicht**: Beschreibt die physische Kommunikation von Geräten (z.B. über Kabel, Glasfaser oder Funk)
  - **Internetschicht**: Beschreibt die Übermittlung von Datenpaketen über mehrere Netzwerkgrenzen hinweg.

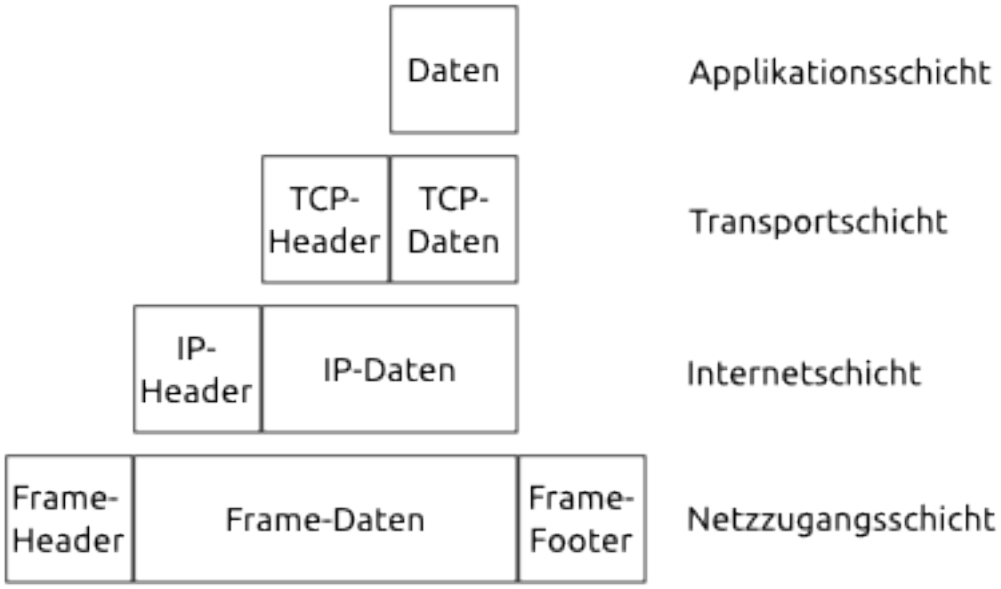
- **Transportschicht**: Beschreibt Ende-zu-Ende-Verbindungen, also wie zwei Applikationen miteinander kommunizieren können.
- **Anwendungsschicht**: Umfasst alle Protokolle, die Anwendungen für die Kommunikation miteinander verwenden.

In jeder Schicht können verschiedene Protokolle benutzt werden.

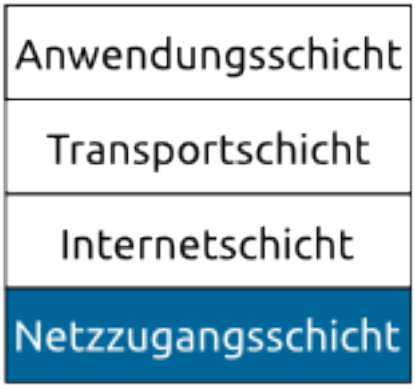
- Wir betrachten von jetzt an das TCP/IP-Modell, da andere Netzwerkmodelle auf Basis des OSI-Modells in der Praxis nur selten vorkommen.

Datenverschachtelung

- Beim Verschicken von Daten werden diese in ein Paket verpackt, in der nächsten Schicht wird das gesamte Paket in ein anderes Paket verpackt und so weiter bis zur untersten Schicht. Am Zielgerät wird in umgekehrter Reihenfolge wieder ausgepackt.
- Beispiel für die Datenverschachtelung:



Netzzugangsschicht (Link Layer)



- Die Netzzugangsschicht ist die unterste Schicht im TCP/IP-Modell und für die physische Übertragung der Daten zuständig.
- Die zuständige Hardware in einem Computer ist z.B. die Netzwerkkarte für kabelgebundene Verbindungen oder die WLAN-Karte für Funkverbindungen.
- 

  Standardprotokoll für kabelgebundene Datenübertragung ist Ethernet (weitgehend ident zur IEEE 802.3 Norm). Für die Verbindung von Geräten werden bestimmte Kabel (CAT-5/CAT-6) und Stecker benutzt.

  Für die Übertragung von Daten via Funk über ein WLAN wird meist ein Standard aus der IEEE 802.11-Familie benutzt.

Netzzugangsschicht - MAC-Address

- Jeder Netzwerkkarte ist eine eindeutige Nummer zur Identifikation, die MAC-Adresse (*Media Access Control Address*), fix zugeordnet. Diese sollte weltweit eindeutig sein, aber es reicht, wenn sie lokal (i.e. im gleichen Netz) eindeutig ist.

  Eine MAC-Adresse besteht aus sechs Bytes, die hexadezimal und mit Doppelpunkten getrennt geschrieben werden (z.B. `3C:97:0E:79:B8:0A`).

  Die spezielle MAC-Adresse `FF:FF:FF:FF:FF:FF` heißt **Broadcast**-Adresse, Pakete an diese Adresse werden von allen Geräten im Netzwerk verarbeitet.

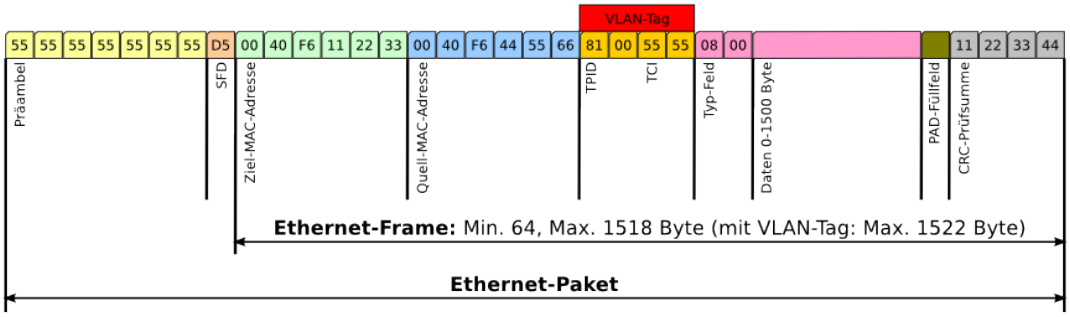
  MAC-Adressen können nicht beliebig von Herstellern vergeben werden. Jeder Hersteller bekommt einen oder mehrere 3-Byte-Prefixe zugeordnet (siehe die OUI-Liste der IEEE), die restliche drei

Bytes kann der Hersteller dann beliebig vergeben.

- Ethernet und WLAN verwenden die gleiche Art von MAC-Adressen.

Netzzugangsschicht - Ethernet-Frame

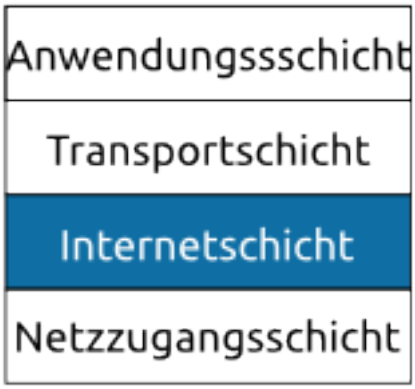
- Die zu sendenden Daten werden gemeinsam mit den nötigen Informationen (unter anderem die Quell- und Ziel-MAC-Adresse) in ein Paket (genannt *Ethernet-Frame*) verpackt und dann über die physische Verbindung verschickt.



- Müssen mehr Daten verschickt werden, als in ein Paket passen (bei Ethernet z.B. 1500 Byte), so müssen mehrere Pakete verschickt werden.
- Die Technologie VLAN (*Virtual Local Area Network*) ermöglicht es, mehrere logische Netze über ein physisches Netz zu betreiben, ohne dass die logischen Netze sich stören.

Siehe auch *Ethernet Theory of Operation*.

Internetschicht (Internet Layer)



- Die Internetschicht erlaubt den Transport von Paketen über mehrere physischen Netze und unterschiedliche Netzzugangsschichtprotokolle (z.B. Ethernet und WLAN) hinweg.
- Es werden dabei keine Garantien abgegeben, ob die Pakete in der richtigen Reihenfolge bzw. überhaupt am Zielgerät ankommen.
- Ein Gerät, das in zwei oder mehr logischen Netzen hängt, heißt **Router**. Dieses leitet Pakete von einem Netz in ein anderes weiter.
- Das Hauptprotokoll in dieser Schicht ist **IP** (*Internet Protokoll*, definiert in RFC 771). Dieses braucht, so wie das Ethernet-Protokoll, wieder Adressen, die sogenannten IP-Adressen. Jedes Gerät, das über das Internet mit anderen Geräten kommunizieren will, braucht eine eindeutige IP-Adressen (Notiz

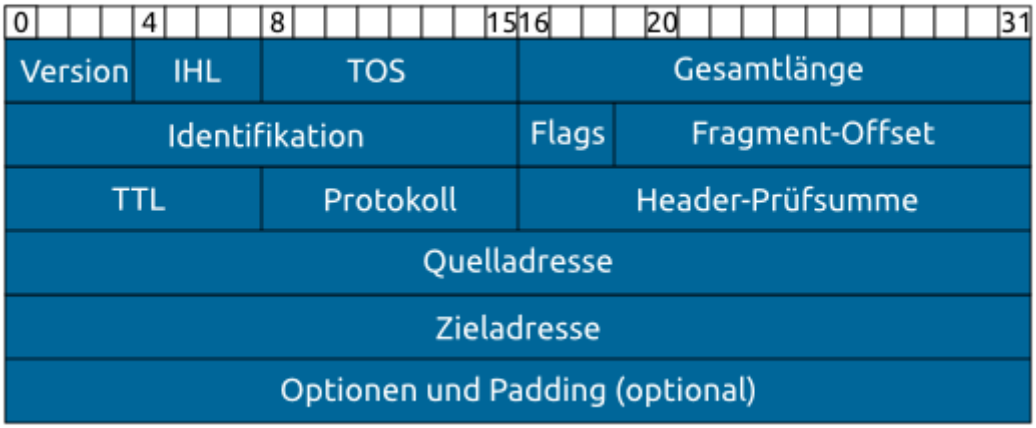
am Rande: Es gibt auch Möglichkeiten, das zu umgehen, z.B. durch NAT bei IPv4).

Internetschicht - IP

- Es gibt zwei Arten des IP-Protokolls: IPv4 und IPv6. Einer der wichtigsten Unterschiede zwischen diesen beiden Protokollen ist die Größe des Adressraumes:
  - Bei IPv4 gibt es 2^32 IP-Adressen, also 4.294.967.296!
  - Bei IPv6 gibt es 2^128 IP-Adressen, also 340.282.366.920.938.463.463.374.607.431.768.211.456!
- IPv4-Adressen werden üblicherweise dezimal in vier Blöcken geschrieben, z.B. `131.130.16.250`. IPv6-Adressen werden hexadezimal in acht Blöcken zu zwei Bytes geschrieben, z.B. `2001:62a:4:2f00::22:250` (genau eine durchgehende Reihe von Blöcken mit Nullen kann durch zwei Doppelpunkte ersetzt werden).
- Derzeit wird fast ausschließlich IPv4 verwendet, aber IPv6 nimmt stetig an Bedeutung zu, da immer mehr Geräte eine IP-Adresse brauchen. Von Vorteil dabei ist, dass IPv6 gleichzeitig zu IPv4 betrieben werden kann.

Internetschicht - IPv4

- Wir werden nur IPv4 näher behandeln. IPv6 ist zwar ähnlich aufgebaut, aber doch in ein paar grundlegenden Gebieten anders.
- In dem Header eines IP-Pakets sind neben Quell- und Zieladresse auch verschiedene andere Felder vorhanden, z.B. das TTL-Feld (*time to live*), das angibt, wie viele Zwischenstationen das Paket passieren darf.



Internetschicht - IPv4-Netze 1

- Früher wurden IPv4-Adressen in Netzklassen (z.B. Klassen A, B und C) eingeteilt. Da diese Einteilung aber sehr unflexibel ist, kommt heutzutage fast ausschließlich das *Classless Inter-Domain Routing*-Verfahren (CIDR, definiert in RFC 1518 und RFC 4632) zum Einsatz.
- Bei den privaten Netzen (IPs daraus werden von Routern nicht weitergeleitet) hat sich die ursprünglichen Einteilung der Netzklassen gehalten, d.h. die Netze `10.0.0.0/8`, `172.16.0.0/12` und `192.168.0.0/16` sind weiterhin als private Netze deklariert.

- Beim CIDR wird durch die Angabe einer Netzwerkmaske die IP-Adresse in einen Netzwerk- und einen Hostteil aufgeteilt. Die Netzwerkmaske gibt an, wie viele Bits vom Anfang der IP-Adresse den Netzpräfix ausmachen.

  Beispiel: IP-Adresse `131.130.16.33` und Netzwerkmaske `255.255.252.0` (entspricht 22 Bits), d.h. das Netzwerk hat die IP-Adressen von `131.130.16.0` bis `131.130.19.255`.

Internetschicht - IPv4-Netze 2

- Jedes IP-Netzwerk hat zwei spezielle Adressen: die **Netzwerkadresse** (die erste Adresse; im Beispiel 131.130.16.0) und die **Broadcastadresse** (die letzte Adresse, im Beispiel 131.130.19.255).

  Die Netzwerkadresse zusammen mit der Netzwerkmaske bestimmt ein IP-Netzwerk eindeutig. Die Broadcastadresse kann benutzt werden, um Daten an alle Geräte des Netzes zu schicken.

- Damit ein Gerät Datenpakete an den richtigen Empfänger schicken kann, gibt es eine sogenannte **Routing-Tabelle**:

  - Pakete für das selbe logische Netz werden direkt an den Empfänger zugestellt
  - Pakete für andere Netze werden an einen Router geschickt, der sie weiterleitet

- Zusammen bedeutet das, dass man die IP-Adresse, die Netzwerkmaske als auch den Standardrouter (**Default-Gateway**) wissen muss, damit ein Gerät in einem IP-Netzwerk richtig funktioniert.

Netzzugangs-/Internetschicht - ARP

- Damit man Geräten im selben logischen Netz IP-Pakete schicken kann, muss deren MAC-Adresse bekannt sein, da diese ja für die Ethernet-Frames auf der Netzzugangsschicht gebraucht werden.

- Zum Zuordnen von IP- zu MAC-Adressen wird das *Address Resolution Protocol* (ARP, definiert in RFC 826) benutzt. Das Protokoll gehört technisch gesehen zur Netzzugangsschicht.

  IPv6 verwendet nicht ARP, sondern das *Neighbor Discovery Protocol* (NDP).

- Ist für eine bestimmte IP-Adresse im gleichen Netz die MAC-Adresse noch nicht bekannt, so wird ein ARP-Request mit der Bitte um Bekanntgabe der zur IP-Adresse gehörigen MAC-Adresse an die Broadcastadresse der Netzzugangsschicht gesendet. Das Gerät mit der zugehörigen IP-Adresse antwortet mit einem ARP-Reply, der die Zuordnung MAC-Adresse zu IP-Adresse enthält, direkt an die MAC-Adresse des ersten Geräts.

- Diese Zuordnungen von IP- zu MAC-Adressen werden üblicherweise lokal in einem ARP-Cache gespeichert.

Internetschicht - ICMP

- Das *Internet Control Message Protocol* (ICMP, definiert in RFC 792) dient zum Austausch von Informationen und Fehlermeldungen. Es gehört zur Internetschicht, benutzt aber IP zur Datenübertragung (d.h. es repräsentiert sich selbst als Protokoll einer höheren Schicht).

- Eine typische Verwendung von ICMP ist das Pingen von Geräten um herauszufinden, ob ein bestimmtes Gerät verfügbar ist und wie groß die Paketumlaufzeit (*round trip time*, RTT) ist. Dazu werden ICMP-Pakete vom Typ *Echo Request* bzw. *Echo Reply* und das TTL-Feld des IP-Headers benutzt.

- Eine weitere, für normale Benutzer nützliche Anwendung von ICMP ist Traceroute, mit der man den

Weg eines Pakets durch das Internet nachvollziehen kann.

Transportschicht (Transport Layer)

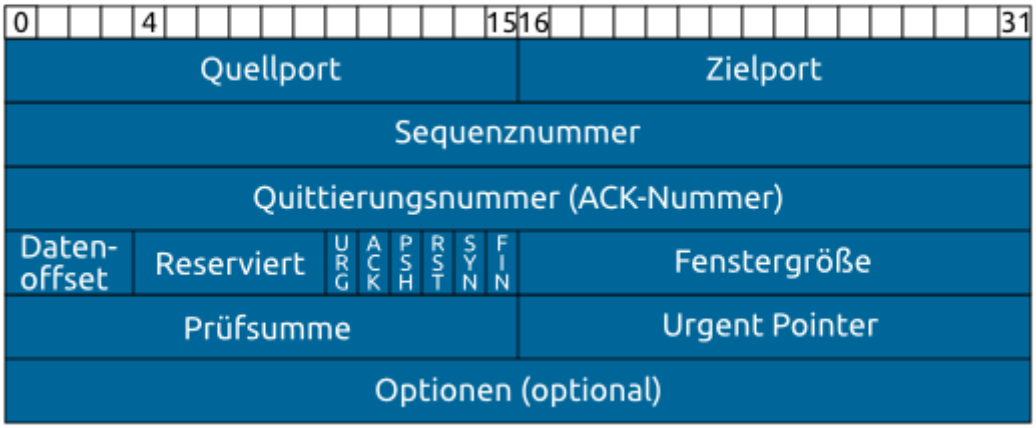
- Die Transportschicht ist für die Ende-zu-Ende-Übertragung von Daten von einem zu einem anderen Gerät zuständig, unabhängig von der Art der Daten und von der Art und Weise, wie sie übermittelt werden.
- Die Anwendungen, die die Daten schicken bzw. empfangen, werden über sogenannte **Port-Nummern** eindeutig identifiziert.

  Bevor Daten ausgetauscht werden können, muss eine Anwendung eine Port-Nummer reservieren. Damit wird sichergestellt, dass Daten immer zu richtigen Anwendung weitergeleitet werden.

  Port-Nummern unter 1024 können nur vom Benutzer *root* reserviert werden. Eine Liste von bekannten Port-Nummern findet man in der Datei `/etc/services`.
- Zwei Protokolle werden hauptsächlich in dieser Schicht verwendet: das *Transmission Control Protocol* (**TCP**, definiert in RFC 793) und das *User Datagram Protocol* (**UDP**, definiert in RFC 768). TCP ist ein verbindungsorientiertes, zuverlässiges Protokoll. UDP hingegen stellt nur sicher, dass die Pakete an die richtige Anwendung geschickt werden.

Transportschicht - TCP

- TCP ist neben IP der Namensgeber für das gesamte Modell (TCP/IP-Modell), weil es das am häufigsten verwendete Protokoll in der Transportschicht ist.
- Das Protokoll erlaubt den zuverlässigen Austausch von Daten. Es kümmert sich darum, dass die Pakete in der richtigen Reihenfolge an die Anwendung weitergegeben werden und dass verlorene oder fehlerhafte Pakete nochmal gesendet werden.
- Der TCP-Header wird an die Daten vorne angefügt.



Transportschicht - TCP-Verbindungen

- Damit zwei Anwendungen miteinander kommunizieren können, müssen zuerst sogenannte **Sockets** (Paare aus IP-Adresse und Port-Nummern) erzeugt werden.

  Durch ein Paar aus Server- und Client-Socket kann eine TCP-Verbindung eindeutig identifiziert werden.

- Bevor nun Daten ausgetauscht werden können, muss die Verbindung mittels des TCP-Handshakes aufgebaut werden:

  - Der Client schickt ein SYN-Paket an den Server.
  - Der Server antwortet mit einem SYN/ACK-Paket an den Client.
  - Der Client schließt den Verbindungsaufbau mit einem ACK-Paket an den Server ab.

- Nach dem TCP-Handshake können Daten ausgetauscht werden. Die Verbindung bleibt auch dann bestehen, wenn keine Daten ausgetauscht werden. Erst der geordnete Verbindungsabbau (ähnlich zum Verbindungsaufbau) beendet eine Verbindung.

- Folgene Anwendungen verwenden beispielsweise TCP: HTTP, FTP, SSH

Transportschicht - UDP

- Im Gegensatz zu TCP ist UDP ein verbindungsloses Protokoll, d.h. es wird keine Verbindung aufgebaut, sondern es werden einfach nur Daten verschickt.

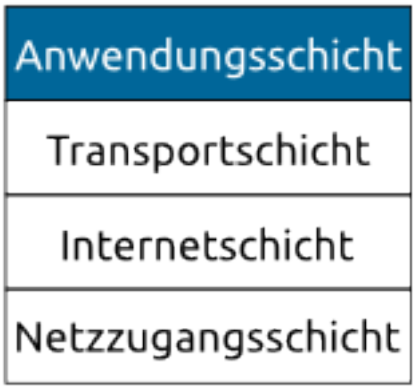
- Es werden auch keinerlei sonstige Garantien gemacht, d.h. man hat hier die gleichen Probleme wie bei IP-Datagrammen:

  - Die Reihenfolge der ankommenden Pakete kann sich ändern.
  - Es können Pakete mehrfach oder gar nicht ankommen.

- Anwendungen, die UDP benutzen, müssen also robust gegenüber solchen Problem sein, haben aber den Vorteil, dass Daten schneller verschickt werden können (der Verbindungsaufbau fällt ja komplett weg).

- Folgende Anwendungen verwenden beispielsweise UDP: DNS, DHCP, SNMP

Anwendungsschicht



- Die Anwendungsschicht umfasst alle Protokolle, die Anwendungen für die Kommunikation miteinander verwenden.
- Dazu gehören standardisierte Protokolle wie HTTP, DHCP oder DNS, aber auch alle anderen, nicht-standardisierten Protokolle, die beliebige Anwendungen verwenden.

DHCP (Dynamic Host Configuration Protocol)

- In größeren Netzwerken ist es mühsam, wenn jedes Gerät einzeln konfiguriert werden muss, d.h. IP-Adresse, Netzwerkmaske und Default-Gateway zugewiesen werden müssen. Abhilfe schafft DHCP (definiert in RFC 2131).
- Bei DHCP wird eine Anfrage an einen DHCP-Server geschickt (mittels der Broadcastadresse 255.255.255.255), in der um eine IP-Adresse und zusätzliche Konfiguration gebeten wird.
- Der DHCP-Server antwortet mit einer noch nicht vergebenen IP-Adresse sowie der Netzwerkmaske. Zusätzlich werden normalerweise das Default-Gateway sowie DNS-Server mitgeteilt. Die vergebene IP-Adresse kann zudem zeitlich beschränkt gültig sein, vor Ablauf der Gültigkeit muss das Gerät erneut den DHCP-Server fragen.
- Die meisten WLAN-Router und z.B. auch VirtualBox (für NAT- und interne Netzwerke) verwenden DHCP zur Konfiguration der angeschlossenen Geräte.

DNS (Domain Name System)

- Das *Domain Name System* (DNS, definiert in RFC 1034 und RFC 1035) erlaubt die Verbindung eines Namens mit einer IP-Adresse. Erst dadurch konnte das Internet so richtig erfolgreich werden, da sich Namen viel leichter merken lassen, als die IP-Adressen dahinter.
- Die einzelnen Teile eines Domainnamens werden mit Punkten voneinander getrennt geschrieben, z.B. `www.mat.univie.ac.at`.
- Das DNS ist hierarchisch aufgebaut. Es gibt nicht ein paar DNS-Server, die alles wissen, sondern viele, die Teile wissen und wissen, wo sie nachschauen können.

  Ganz oben in dieser Hierarchie stehen die **Root-Server**, die die DNS-Server der *Top Level Domains* (TLDs, im Domainnamen ganz rechts zu finden) kennen.
- Soll ein Domainname zu einer IP-Adresse aufgelöst werden, wird dieser (im schlimmsten Fall) an

einen Root-Server geschickt. Dieser antwortet mit der IP-Adresse des DNS-Servers, der die TLD des Domainnamens verwaltet. Dann wird dieser DNS-Server gefragt, der dann den nächsten Teil des Domainnamens (von rechts nach links) auflöst und so weiter (iterative Auflösung).

Netzwerkkonfiguration

- Bei einem Ubuntu-Desktop-System wird für die Konfiguration der Netzwerkschnittstellen das Programm NetworkManager verwendet. Dieses erlaubt eine einfache Konfiguration der vorhandenen Schnittstellen (über die CLI oder GUI) und ist mittlerweile schon tief in verschiedene Desktop-Umgebungen wie Gnome oder KDE eingebunden.

  Für die CLI gibt es z.B. die Kommandos `nm-tool` (zur Statusanzeige) und `nmcli` (zur Kontrolle des NetworkManager-Dämons).

- Auf einem Ubuntu-Server-System wird die Netzwerkkonfiguration in der Datei `/etc/network/interfaces` vorgenommen. Aus mehreren Gründen (z.B. Sicherheit) wird der NetworkManager hier nicht verwendet.

- Fixe Zuordnungen von IP-Adressen zu Hostnamen lassen sich über die Datei `/etc/hosts` bewerkstelligen, die dynamische Auflösung via DNS kann über `/etc/resolv.conf` konfiguriert werden.

Befehle - ifconfig

ifconfig - Konfiguriert eine Netzwerkschnittstelle.

» Wird benutzt, um Netzwerkschnittstellen zu (de)aktivieren und IP-Address-Konfigurationen zuzuweisen bzw. um die aktuelle Konfiguration anzuzeigen.

» Die Datei `/proc/net/dev` zeigt alle verfügbaren Kernel-Netzwerkschnittstellen an.

» Option `-a` zeigt alle Schnittstellen (auch deaktivierte) an.

»
```
$ # Konfiguration von eth0
$ ifconfig eth0 up 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
$ ifconfig eth0
eth0      Link encap:Ethernet  Hardware Adresse 08:00:27:f2:94:b9
          inet Adresse:10.0.2.15  Bcast:10.0.2.255  Maske:255.255.255.0
          inet6-Adresse: fe80::a00:27ff:fef2:94b9/64 Gültigkeitsbereich:Verbindung
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metrik:1
          RX packets:31 errors:0 dropped:0 overruns:0 frame:0
          TX packets:43 errors:0 dropped:0 overruns:0 carrier:0
          Kollisionen:0 Sendewarteschlangenlänge:1000
          RX-Bytes:3811 (3.8 KB)  TX-Bytes:4111 (4.1 KB)
```

Befehle - route

route - Zeigt/Ändert die Routing-Tabelle.

» Wird üblicherweise nach `ifconfig` benutzt, um eine statische Route für einen Host/ein Netzwerk anzulegen (z.B. um das Default-Gateway einzutragen).

» Optionen: `-n` → keine Namensauflösung von IP-Adressen, `-net` → Route für ein Netzwerk statt einem Host anlegen.

» Für das Löschen einer Route `del` statt `add` verwenden.

»
```
$ # Anlegen einer Netzwerk-Route
$ route add -net 10.0.2.0 netmask 255.255.255.0 dev eth0
$ # Anlegen der Route für das Default-Gateway
```

```
$ route add default gw 10.0.2.3
$ route
Ziel            Router          Genmask         Flags Metric Ref    Use Iface
default         10.0.2.3        0.0.0.0         UG    0      0        0 eth0
10.0.2.0        *               255.255.255.0   U     0      0        0 eth0
192.168.56.0    *               255.255.255.0   U     0      0        0 eth1
```

Befehle - arp/arping

arp - Manipuliert den ARP-Cache.

» Wird hauptsächlich für die Anzeige des ARP-Caches benutzt, man kann aber auch gespeicherte Zuordnungen löschen bzw. neue hinzufügen.

» In der Datei `/proc/net/arp` sieht man auch den ARP-Cache.

»
```
$ arp
Adresse Hardware-Typ Hardware-Adresse Optionen Maske Schnittstelle
192.168.56.1             ether   0a:00:27:00:00:00   C                     eth1
10.0.2.3                 ether   52:54:00:12:35:03   C                     eth0
$ # Entfernen eines Eintrags
$ arp -i eth0 -d 10.0.2.3
$ # Hinzufügen eines Eintrags
$ arp -i eth0 -s 10.0.2.3 52:54:00:12:35:03
```

arping - Schickt ARP-Requests an einen Host.

» Nützlich, um die MAC-Adresse für eine bestimmte IP-Adresse herauszufinden.

»
```
$ arping -c 1 10.0.2.3
ARPING 10.0.2.3 from 10.0.2.15 eth0
Unicast reply from 10.0.2.3 [52:54:00:12:35:03]  1.332ms
Sent 1 probes (1 broadcast(s))
Received 1 response(s)
```

Befehle - ip

ip - Zeigt an und verändert Netzwerkschnittstellen, Routing-Tabellen und mehr.

» Die Programme `ifconfig`, `route` und `arp` werden nach wie vor benutzt, sind allerdings schon sehr alt und haben auch Schwächen. Das `ip`-Programm (Teil des `iproute2`-Pakets) vereint die Funktionalitäten dieser Programe und bietet noch viel mehr.

» Links:

- Vergleich von ifconfig und ip
- iproute2 auf Wikipedia
- IPRoute Howto
- Linux Advanced Routing & Traffic Control HOWTO
- Deprecated Linux networking commands and their replacements.

»
```
$ ip link set eth0 up
$ ip addr add 10.0.2.15/24 broadcast 10.0.2.255 dev eth0
$ ip route add 10.0.2.0/24 dev eth0
$ ip route add default via 10.0.2.3 dev eth0
$ ip neigh add 10.0.2.3 lladdr 52:54:00:12:35:03 dev eth0
```

Befehle - ping

ping - Schickt einen Ping (ICMP-Echo-Request) an einen Host.

- » Zum Testen, ob ein bestimmter Host erreichbar ist. **Achtung**: Manche Hosts sind aus Sicherheitsgründen so konfiguriert, dass sie nicht antworten!
- » `$ ping -c 1 www.orf.at`

```
PING www.orf.at (194.232.104.139) 56(84) bytes of data.
64 bytes from orf.at (194.232.104.139): icmp_req=1 ttl=56 time=1.19 ms

--- www.orf.at ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.195/1.195/1.195/0.000 ms
```

Befehle - traceroute

traceroute - Zeigt die Route eines Pakets zu einem Host an.

- » Kann neben ICMP-Echo-Requests auch andere Methoden verwenden (z.B. TCP und UDP).
- » Optionen: `-I` → ICMP-Methode, `-T` → TCP-Method, `-U` → UDP-Methode.
- » `$ traceroute www.google.com`

```
traceroute to www.google.com (173.194.44.244), 30 hops max, 60 byte packets
 1  selene.cc.univie.ac.at (131.130.254.145)  1.499 ms  1.522 ms  1.480 ms
 2  hekate.cc.univie.ac.at (131.130.254.37)  1.486 ms  1.459 ms  1.724 ms
 3  ares.cc.univie.ac.at (131.130.253.113)  1.434 ms  1.399 ms  1.707 ms
 4  vlan1501.wien1.aco.net (193.171.13.1)  1.365 ms *  1.659 ms
 5  * * *
 6  * * *
 7  * * *
 8  nixcz.net.google.com (91.210.16.211)  7.347 ms  7.313 ms  7.309 ms
 9  209.85.241.79 (209.85.241.79)  7.607 ms  7.863 ms  7.520 ms
10  173.194.44.244 (173.194.44.244)  7.086 ms  7.106 ms  7.097 ms
```

`$ traceroute6 www.google.com 2>/dev/null | tail -n 1`

```
15  bk-in-x93.1e100.net (2a00:1450:4008:c01::93)  29.472 ms  29.337 ms  29.492 ms
```

Befehle - netstat

netstat - Zeigt Netzwerkinformationen an.

- » Sehr vielseitiges Programm mit vielen Optionen zur Anzeige von Routing-Tabellen, offenen Verbindungen, Statistiken, ...
- » Anzeigen von: keine Angabe → offene Sockets, `-r` → Routing-Tabellen, `-i` → Netzwerkschnittstellen, `-s` → Statistiken.
- » Optionen: `-n` → keine Namensauflösung, `-l` → nur Sockets im Status LISTEN anzeigen, `-a` → alle Sockets anzeigen, `-t` → TCP-Sockets anzeigen, `-u` → UDP-Sockets anzeigen, `-p` → Prozess anzeigen.
- » `$ netstat -t -n -a`

```
Aktive Internetverbindungen (Server und stehende Verbindungen)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:6010          0.0.0.0:*               LISTEN
tcp        0      0 192.168.56.101:22       192.168.56.1:34711      VERBUNDEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:6010                :::*                    LISTEN
```

Befehle - ss

ss - Zeigt Socket-Statistiken an.

» Ähnlich wie `netstat`, gehört zu dem selben Paket wie `ip`.

» Optionen: `-l` → nur Sockets im Status LISTEN anzeigen, `-a` → alle Sockets anzeigen, `-r` → Adressen und Ports zu Namen auflösen, `-n` → keine Namensauflösung, `-p` → Prozess anzeigen, `-t` → TCP-Sockets anzeigen, `-u` → UDP-Sockets anzeigen, `-x` → Unix-Sockets anzeigen.

» `$ ss -t -n -a`

```
State      Recv-Q Send-Q        Local Address:Port      Peer Address:Port
LISTEN     0      128                       *:22                    *:*
LISTEN     0      128                      :::22                   :::*
LISTEN     0      128           127.0.0.1:6010                    *:*
LISTEN     0      128                  ::1:6010                   :::*
ESTAB      0      0          192.168.56.101:22       192.168.56.1:54330
```

Befehle - Netzwerkmonitoring

Zur Überwachung eines Netzwerks oder zum Untersuchen von Netzwerkpaketen gibt es mehrere Programme unter Linux:

- **tcpdump**: CLI; Pakete, gefiltert nach Bedingungen, werden ausgegeben.
- **wireshark**: GUI; Ähnlich wie tcpdump, aber einfacher. Kann tcpdump Ausgaben verwenden.
- **iptraf**: CLI (ncurses); Zeigt unter anderem Statistiken für verschiedene Protokolle an.

Netzwerkkonfiguration mit Puppet

- Puppet bietet keine inkludierte Möglichkeit, Netzwerkschnittstellen zu verwalten.
- Der Wiki-Eintrag *Network Interface Templates* zeigt aber einige Möglichkeiten auf, wie über die eingebauten Ressourcetypen die Netzwerkschnittstellen verwaltet werden können.

Hostverwaltung mit Puppet

- Hostnamen und deren IP werden in Puppet über die Ressource *host* verwaltet.
- Unter Ubuntu wird standardmäßig der Provider *parsed* verwendet (der einzig mögliche).
- Wichtige Attribute:
  - **name**: Der Name des Computers (falls nicht angegeben, wird der Titel verwendet).
  - **ensure**: Der gewünschte Zustand (*present* oder *absent*).
  - **comment**: Ein Kommentar zum Computer.
  - **host_aliases**: Ein Aliasname oder ein Array von Aliasnamen.
  - **ip**: Die IP-Adresse des Computers.
  - **target**: Die Zieldatei, in der die Informationen gespeichert werden (Standardwert: `/etc/hosts`).

Hostverwaltung mit Puppet - Beispiele

- Anlegen oder Modifizieren eines Hosteintrags:

```
host {'ict-infrastruktur.home:
```

```
  ensure => present,
  host_aliases => 'ict-infrastruktur',
  ip => '127.0.0.1',
}
```

- Löschen eines Hosteintrags:

```
host {ict-infrastruktur.home': ensure => absent}
```

Copyright und Lizenz