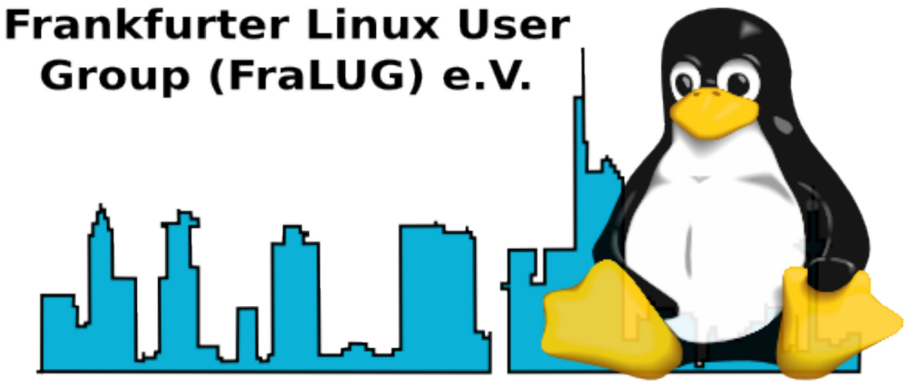
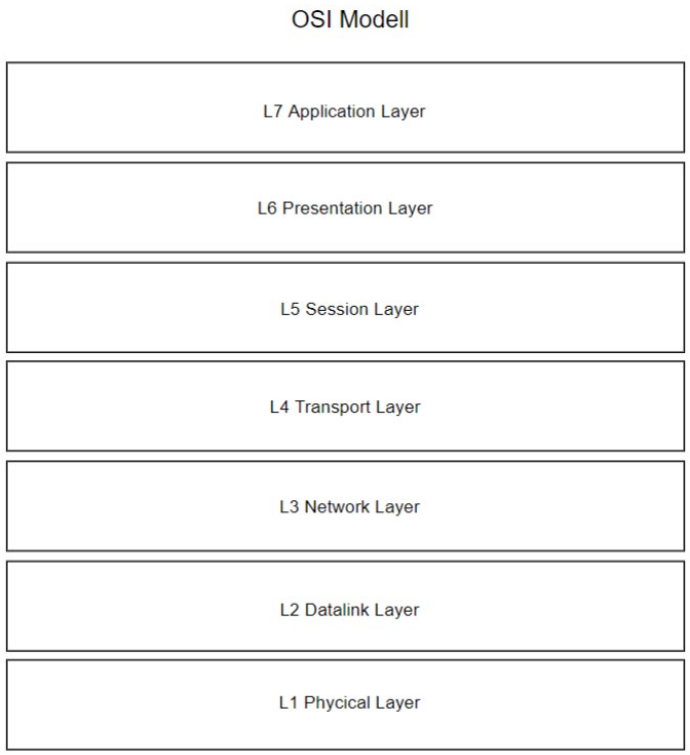
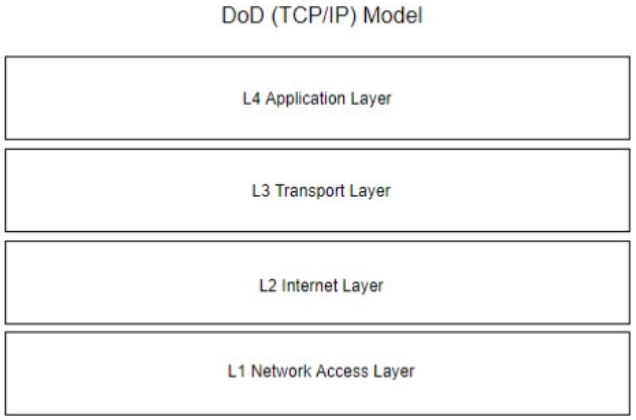
Linux und Netzwerke



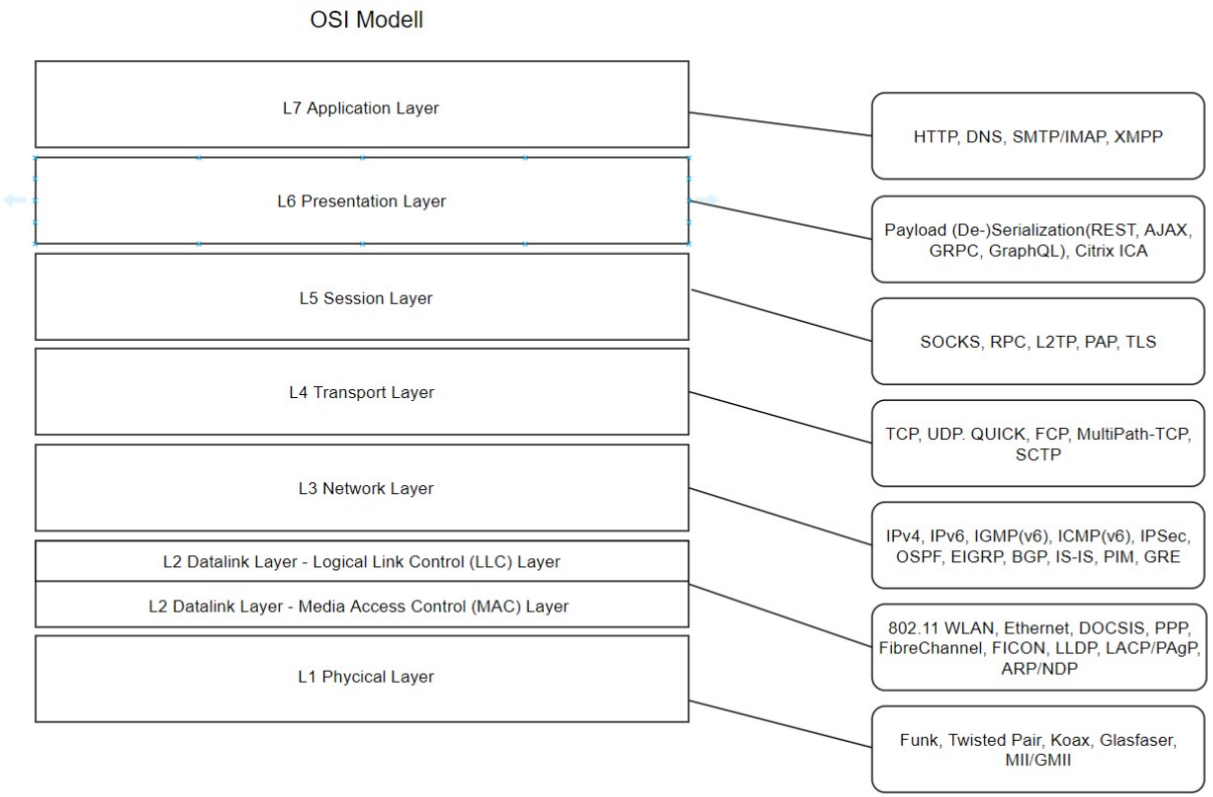
BASICS

Schichten und Kapselung





Schichten und Kapselung



L2 - Frames

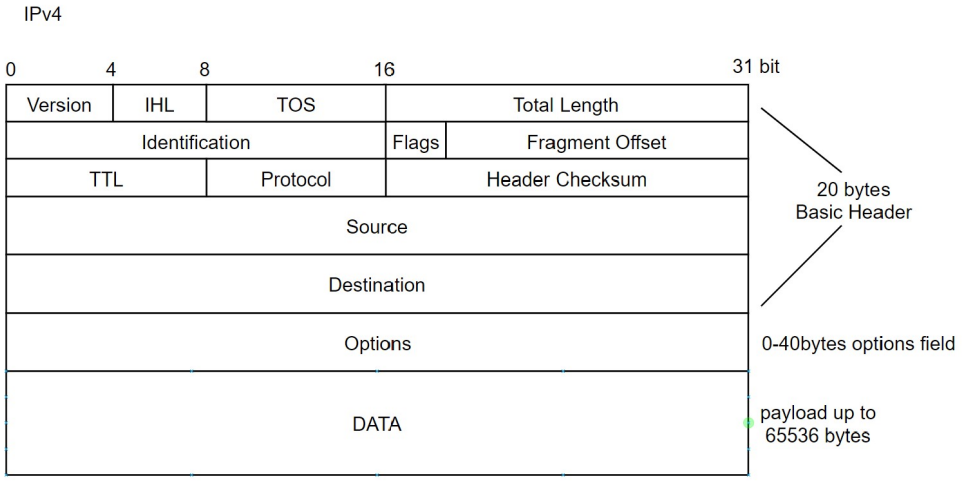
- PDUs auf L2 werden Frames genannt
- Aufbau, Länge und Nomenklatur eines Frames sind vom verwendeten Protokoll abhängig
- Im Allgemeinen haben Frames,Header, Payload sowie ein abschliessendens FCS Feld(Frame CheckSum - Prüfsumme) je nach Protokoll gibt es zusätzliche Control Frames
- je nach Norm werden IPG patterns (InterPacketGap) und RTS/CTS Mechaniken verwendet.

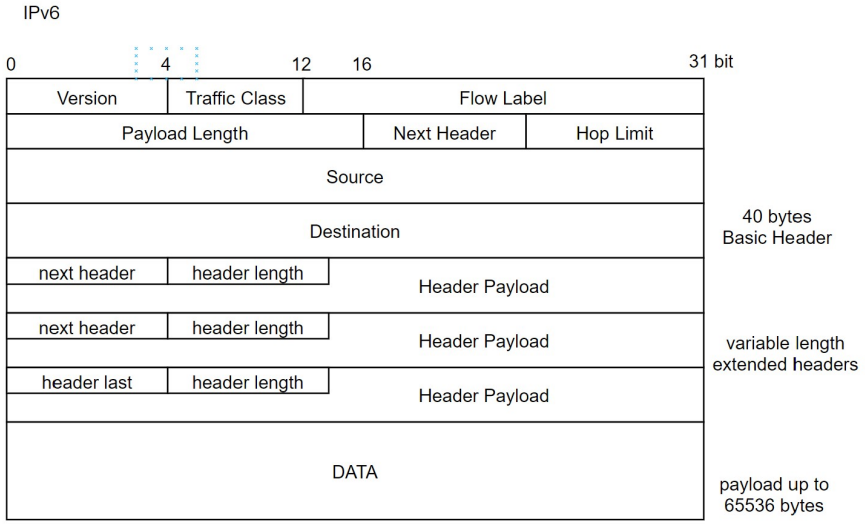
L2 - CSMA/CA und CSMA/CD

- CSMA/CA = Carrier Sense Multiple Access/Collision Avoidance
- CSMA/CD = Carrier Sense Multiple Access/Collision Detect
- Die meisten Netzwerke benutzen eine Bus-Topologie(shared media) das betrifft u.a. Ethernet, WLAN, DOCSIS
- Es darf immer nur ein Teilnehmer auf dem Bus „sprechen“ ansonsten kommt es zu Kollisionen
- Implementiert partitiell RTS/CTS Mechaniken
- Implementierung abhängig von Medium und Komponenten

Internet Layer IPv4/IPv6

- PDUs auf Layer 3 werden Pakete genannt.
- Sie enthalten Header und Payload
- unidirektional





IPv4

- IPv4 Addressen sind 32bit lang und werden dotted decimal notiert.
- Es gibt 4 Typen von Adressen.
- Network Address beschreibt die Addresse eines Subnet Segments
- Host Address beschreibt die Addresse eines Hosts innerhalb eines Subnets oder eines Hosts innerhalb einer Punkt-zu-Punkt Verbindung(/32)
- Die Network Broadcast Address wird verwendet um alle Hosts innerhalb eines Subnets zu erreichen, es handelt sich immer um die letzte Addresse eines Subnets.
- Global Broadcast Address 255.255.255.255, wird verwendet wenn noch keine Subnet Adresse bekannt ist. (DHCP eg.)

ICMP

- Das Internet Control Message Protocol generiert Fehler falls es bei der Vermittlung von Paketen zu Fehlern kommt und wird zum debuggen von Layer3 Problemen verwendet.
- Bekannteste Vertreter “ping“ und Traceroute
- Verwendet sogenannte Datagramme
- 8byte header und variable Datenlänge
- Wird meist auf Firewalls gefiltert.

IPv4 Dynamische Addresszuweisung
DHCP

- Je nach nach dem wie ein Client konfiguriert ist wird er entweder einen DHCP Server nach einer Addresse fragen.
- Dazu sendet er einen DHCP Discover an die Broadcast Address. (255.255.255.255)
- Wenn ein DHCP Server diesen Discover empfängt sendet er einen (je nach Konfig) einen Offer zurück.
- Der Client sendet darauf hin einen Lease Request und bittet um Zuweisung.
- Der Server weist einen Lease zu und übermittelt das im Lease-Acknowledgement.

IPv4 RFC 1918 - Private Addresses

- Da die Anzahl an IPv4 begrenzt ist hat man recht früh bestimmte Addressräume für private/nicht-öffentliche Nutzung reserviert.
- Diese Netze sollen nicht ins public Internet geroutet werden.
- für IPv4 umfasst das die folgenden Bereiche
  10.0.0.0/8 - 10.0.0.0 - 10.255.255.255
  172.16.0.0/12 - 172.16.0.0 - 172.31.255.255
  192.168.0.0/16 - 192.168.0.0 - 192.168.255.255
- Traffic von und zu diesen Addressen wird via Address Translation ins Internet vermittelt.

IPv4 Dynamische Addresszuweisung Zeroconf/APIPA/Bonjour

- Je nach Konfiguration wird versucht eine link local Addresse festzulegen wenn weder ein DHCP zur Verfügung steht noch eine statische Addresse konfiguriert wurde. Dies dient dem Zweck der Kommunikation im lokalen Netz.

- Für diesen Zweck wurde laut RFC3927 das Subnetz 169.254.x.x/16 für link local Adressen zugewiesen. Dieses Netz ist nicht routable.

IPv6

- Im Gegensatz zu IPv4 sind IPv6 Adressen 128bit lang und werden hexadezimal notiert.
- verkürzte Darstellung möglich.
- Interfaces haben immer zwei IP Adressen. Eine Link Global und eine Link Local Addresse.
- Die Link Local Adresse wird für den NDP Sublayer benötigt, selbst wenn Link Global Adressen konfiguriert sind. (NDP Neighbor Discovery Protocol -> ARP Nachfolger)
- NDP kann verwendet werden um Routing Prefixe zu an einzelne Teilnehmer zu übermitteln.

IPv6 Link Global und Link Local (RFC4193)

- Analog zu RFC3927 wurden in RFC4193 link local Adressen zugewiesen.
- Im Gegensatz zu IPv4 erfüllen IPv6 LL Adressen zusätzliche Zwecke.
- der prefix für link local Adressen ist fe80::/10
- die Default Route wird normalerweise via link local zugewiesen.

L3 - Static Routing

- Jede IP Adress Konfiguration besteht aus Adresse und Netzmaske
  Angegeben wird das im Format Adresse/<Netmask|CIDR>
- Bsp: 192.168.1.1/24 ist eine IPv4 Adresse und hat 32bit.
  Der Netprefix ist 24bit lang. Alle Adressen im Hostanteil(blau) sind lokal erreichbar und brauchen keinen Router.
- Adressen die eine Änderung im grünen Bereich(Netz) erfordern müssen geroutet werden.

11111111.11111111.11111111.00000000
11000000.10101000.00000001.00000001

L4 - Transport Protocol

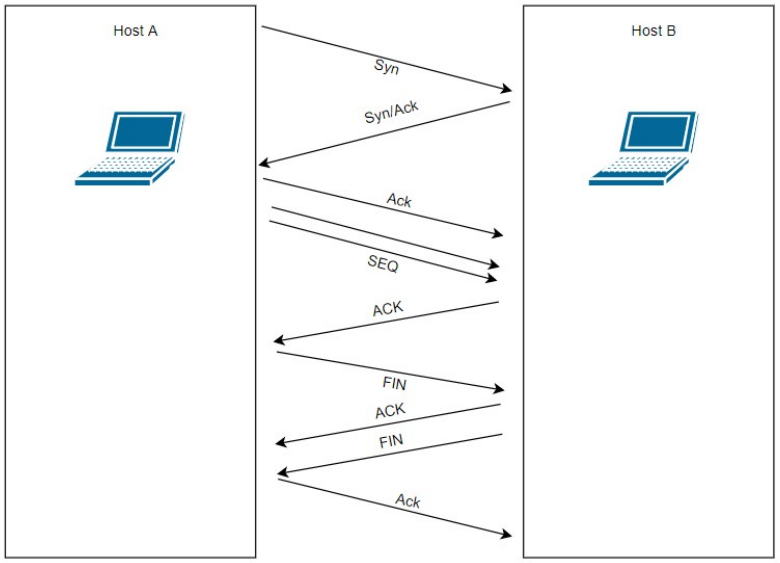
- Auf L4 erfolgt der eigentliche Datentransfer.
- L4 ist normalerweise Portbasiert.
- Ein Service auf einem Host lauscht auf einem Port auf eingehenden Traffic. Der Client öffnet einen Port in seiner ephemeral Range kommuniziert auf diesem Port mit dem Server und dessen Port um entsprechende Daten auszutauschen.
- Ggfs. werden im Laufe der Kommunikation Seitenkanäle auf weiteren Ports geöffnet.

TCP - Segments

- Stateful Connection mit Fehlerkorrektur und Flußkontrolle
- Viel Overhead

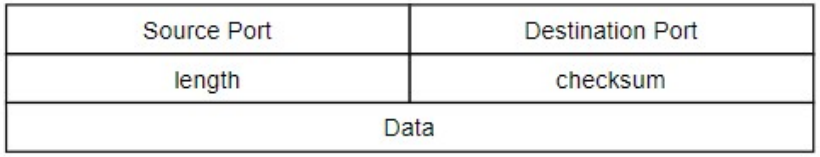
TCP Segment

Source Port								Destination Port
Sequence Number								
Ackknowledgement Number								
offset		U	A	P	R	S	F	Window Size
Checksum								Urgent Pointer
Options								
Payload								



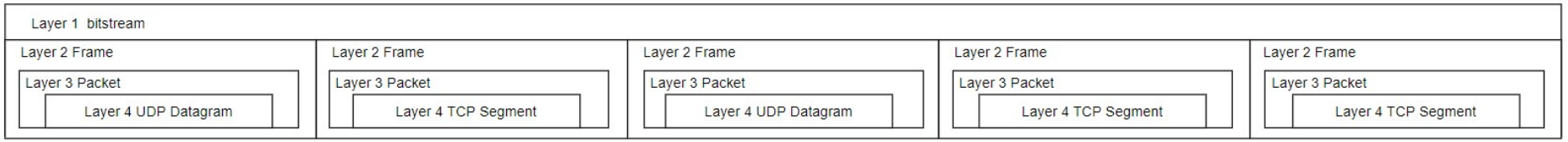
UDP - DATAGRAMS

- UDP ist unzuverlässig by Design (Keine Handshakes, kein retransmit - Fire and Forget)
- UDP kennt keine Flußkontrolle(muss auf Applikationsebene implementiert werden)
- geringerer overhead, geringere Latenz.
- präferiert für Protokolle die keine 100% Datenkonsistenz benötigen wie voice oder gaming-traffic.



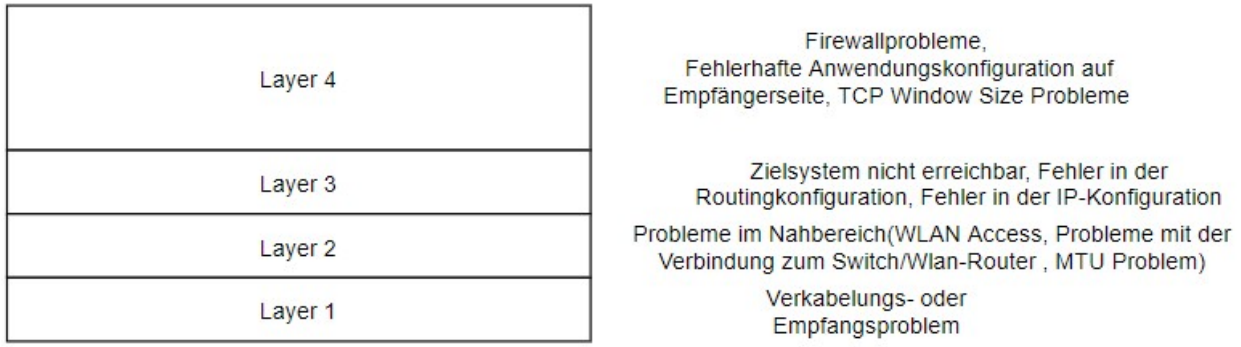
Encapsulation

- Layer(Schichten) bieten saubere logische Trennung
- Höhere Layer werden in niedrigeren Layern gekapselt



Debugging der unteren Layer

- Typische Fehler

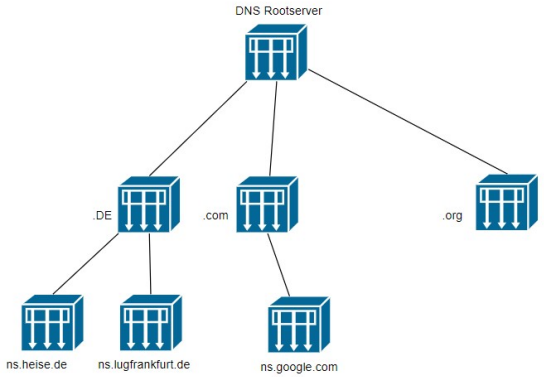


DNS Resolver

- Internet Routing ist Nummern basiert und kann per se nichts mit Hostnamen oder Domains anfangen.
- Um Maschinen per Hostnamen anzusprechen bedarf es eines DNS Servers der ein entsprechendes Namensverzeichnis bereitstellt. (analog zum klassischen Telefonbuch)
- per default werden diese Server in /etc/resolve.conf eingetragen.
- DNS arbeitet auf Layer 7

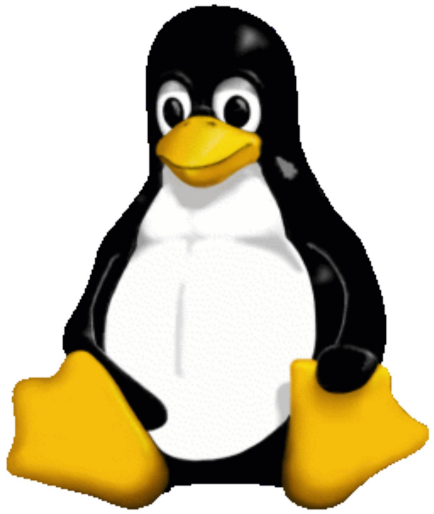
DNS Zones

- DNS ist in Zonen organisiert. Einzelne Zonen können hierarchisch delegiert werden.
- Zonen enthalten Resource Records für Domains, Subdomains und einzelne Server und weitere Informationen (DKIM/DMARC eg.)



DNS Queries & Records

- Wenn ein Linux-Client einen Hostnamen auflösen möchte prüft er zu erst sein lokales hosts-file(/etc/hosts)
- Wenn kein Eintrag vorhanden ist sendet er einen Query an seinen DNS Server. Wenn dieser einen gültigen Cache-Eintrag für den Host hat oder die Zone selbst verwaltet gibt er die entsprechende Antwort mit passender IP Addresse.
- Hat er keinen Eintrag delegiert er an die nächst höhere Hierarchieebene.
- Der nächst höhere Server schaut ob er einen gültigen Cache-Eintrag für den Hostnamen hat und falls nicht prüft er ob er den Request weiter nach oben delegieren muss oder sich ein Authoritativer Server innerhalb seiner Zone befindet und sendet den Request entsprechend weiter.



LINUX

Linux driver modules

- Wenn der Kernel Bus Treiber zur Laufzeit neue Geräte feststellt triggert er driver_register()
- Damit erstellt der Kernel für das Gerät Strukturen im /sys filesystem und löst ein uevent(udev event) vom Typ „add“ aus
- Das BUS Device identifiziert sich mit seiner HW ID
- Der Kernel wandelt die ID in eine MODALIAS ID um.

Linux driver modules

- Modalias IDs sind Bestandteil der Linux Treiber Module und werden durch depmod indiziert. Der Index wird im /lib/modules/ Verzeichnis des Kernels im modules.alias file hinterlegt.
- udev lädt entsprechend seiner Regeln (/lib/udev/rules.d/ && /etc/udev/rules.d/) das entsprechende Modul via modprobe $modalias
- udev erstellt die nötigen nodes in /dev um das Gerät im System verfügbar zumachen

Linux driver modules - Boot

- Bei Systemstart steht udevd noch nicht zur Verfügung sondern der Kernel lädt nur Module die festeinkompiliert oder in seiner Ramdisk(initrd) vorhanden sind
- für alle anderen Devices schreibt der Kernel beim Boot files mit udev events in /sys/devices/ in die Geräteunterverzeichnisse die abgearbeitet werden sobald udevd gestartet wurde.

ethtool & iwconfig

- zuständig für Treiber und Hardwareeinstellungen
- ethtool <devicename> bzw iwconfig zeigen die aktuellen Einstellungen an.

```
$ iwconfig eth1

eth1      IEEE 802.11g  ESSID:"OSU_PUB"
          Mode:Managed  Frequency:2.427 GHz  Access Point: 00:0D:9D:C6:38:2D
          Bit Rate=48 Mb/s   Tx-Power=20 dBm   Sensitivity=8/0
          Retry limit:7   RTS thr:off   Fragment thr:off
          Power Management:off
          Link Quality=91/100  Signal level=-39 dBm  Noise level=-87 dBm
          Rx invalid nwid:0  Rx invalid crypt:860  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:39   Missed beacon:8
```

```
[testhost ~] ethtool  ens160
Settings for ens160:
        Supported ports: [ TP ]
        Supported link modes:   1000baseT/Full
                                10000baseT/Full
        Supported pause frame use: No
        Supports auto-negotiation: No
        Supported FEC modes: Not reported
        Advertised link modes:  Not reported
        Advertised pause frame use: No
        Advertised auto-negotiation: No
        Advertised FEC modes: Not reported
        Speed: 10000Mb/s
        Duplex: Full
        Port: Twisted Pair
        PHYAD: 0
        Transceiver: internal
        Auto-negotiation: off
        MDI-X: Unknown
        Supports Wake-on: uag
        Wake-on: d
        Link detected: yes
```

iproute2

- iproute2 ist die derzeitige Standardtoolssammlung für TCP/IP(v6) für Netzwerk und Traffic Kontrolle in Linux
- Es löst die Tools der älteren BSD TCP/IP Suite ab.(netstat, ifconfig, etc)
- Es stellt Werkzeuge für die Konfiguration von IP Adressen, Device, Tunneling sowie die Verwaltung von ARP/Neighbor Einträgen und Routing zur Verfügung

iproute2 - ip link und ip neigh

- „ip link set“ bietet Funktionalitäten zur Konfiguration des Links(MTU Size, Multicast, ARP/NDP, <UP|DOWN> State etc)
- „ip neigh <add|delete|show|change|replace|flush>“ dient zur Kontrolle der Einträge in der ARP/Neighbor Table
- debugging commands:
  - ip link show
  - ip neigh show

iproute2 - ip addr und ip route

- „ip addr“ bietet Funktionalitäten zur Konfiguration der IP-Konfiguration
- „ip route“ dient der Verwaltung der Routingkonfiguration
- debugging commands:
  - ip --br add
  - ip route show
  - ip route get <destination>

iproute2 - ip rule und ip tunnel

- „ip rule“ bietet Funktionalitäten zur Konfiguration der Policy-based Routingkonfiguration
- „ip tunnel“ verwaltet gekapselten Traffic
- debugging commands:
  - ip rule list
  - ip tun list
  - ip tun list <tunnel-dev>

Network Manager

- aktuelles Standardkonfigurationswerkzeug der meisten Distributionen für die Netzwerkkonfiguration.
- standard command: nmcli
- standard debug commands:
  nmcli con show
  nmcli radio
  nmcli wifi rescan
  nmcli wifi list

Network Manager - WLAN

- Connect to a wifi network
  nmcli <device>
  nmcli <device> wifi rescan
  nmcli <device> wifi list
  nmcli <device> wifi connect <ssid> --ask

Network Manager - Ethernet

- Connect to ethernet network with DHCP
  nmcli con show
  nmcli con modify “<con-name>“ ipv4.method auto
- connect to ethernet with static ip
  nmcli con show
  nmcli con modify “<con-name>“ ipv4.method manual ipv4.addresses x.x.x.x/x
  nmcli con modify “<con-name“> ipv4.gateway x.x.x.x
  nmcli con modify “<con-name“> ipv4.dns x.x.x.x

nützliche Troubleshooting Tools

- ss(früher netstat)
  “ss -tulpenmZ“ zeigt alle listener an inklusive Speicherverbrauch und SELinux Context(netstat -tulpn)
- dig (früher nslookup)
  “dig +short lugfrankfurt.de“
- telnet (tcp port connectivity checking)
  “telnet <host> <port> “
- tcpdump(traffic auf einem Interface sniffen)
  “tcpdump -v -i <interface> -s 0 -w <output-file>“

nützliche Troubleshooting Tools

- netcat connectivity testing
  receiver einrichten :
  “netcat -l <port>“ für tcp und “netcat -ul <port>“ für einen udp listener
  traffic senden:
  “netcat <targethost> <port>“ für tcp und “netcat -u <targethost> <port>“
- iperf/iperf3 bandwidth test
  receiver starten :
  iperf -s
  traffic senden:
  iperf -c <target ip>