iproute2 vs. ifconfig

Warum habe ich das nicht schon viel früher benutzt?

Christian Perle

22. Juli 2015

Warum sollte man `iproute2` benutzen?

- Ersetzt u.a. die Kommandos `ifconfig` und `route`
- Kann deutlich mehr als nur Netzwerkinterfaces und Routen konfigurieren
- Moderne Linux-Distributionen setzen teilweise nur noch `iproute2` ein

Begriffsverwirrung und Sonstiges

- Das eigentliche Binary heißt `/bin/ip`, der Paket- und Projektname lautet `iproute2`
- Wurde schon mit Kernel 2.2 (1999) eingeführt
- Schnittstelle zum Kernel ist ein Netlink-Socket (im Gegensatz zu ioctl-Aufrufen auf „normalen“ Sockets)

Adresskonfiguration

Adressen können einzeln hinzufügt und gelöscht werden. *Echte* Konfiguration mehrerer Adressen auf einem Interface, kein Interface-Aliasing. Netzmasken werden grundsätzlich in CIDR-Schreibweise angegeben.

```
# ip addr add 10.0.80.2/24 dev eth0
# ip addr add 192.168.20.1/25 dev eth0

# ip addr del 10.0.80.2/24 dev eth0

# ip addr flush dev eth0
```

Link-Konfiguration

Adresskonfiguration setzt ein Interface nicht automatisch „up“, das wird von `ifconfig`-Umsteigern gerne vergessen. Zur Link-Konfiguration gehört auch das Setzen von MTU und MAC-Adresse, sowie das Umbenennen von Interfaces.

```
# ip link set eth0 up

# ip link set eth0 down

# ip link set eth0 mtu 1320

# ip link set eth0 address 00:23:42:af:fe:09

# ip link set eth0 name foobar0
```

CIDR – wie was wo?

- Steht für Classless Inter-Domain Routing
- Hebt die starre Unterteilung in nur drei Größenklassen von Netzen auf
- In einer Adresse `10.80.80.3/25` steht die 25 hinter dem Slash für die Anzahl der gesetzten Bits in der Netzmaske (25 entspricht 255.255.255.128)
- Die gesetzten Bits werden dabei als zusammenhängend und „linksbündig“ angenommen
- Macht es leichter, die Größe eines Netzes zu erkennen

ifconfig-Entsprechungen

```
# ifconfig eth0 10.0.0.1 netmask 255.255.0.0

# ip addr add 10.0.0.1/16 dev eth0
# ip link set eth0 up

# ifconfig eth0:1 10.20.20.1 netmask 255.255.254.0

# ip addr add 10.20.20.1/23 dev eth0
```

route-Entsprechungen

```
# route add -net 172.16.40.0 netmask 255.255.255.0 \
    gw 10.0.0.2

# ip route add 172.16.40.0/24 via 10.0.0.2

# route add -host 192.168.10.10 gw 10.0.0.10

# ip route add 192.168.10.10 via 10.0.0.10

# route add default gw 10.0.0.5

# ip route add default via 10.0.0.5
```

Neues

Der ARP-Cache (allgemeiner: Neighbor-Cache) kann abgefragt werden. Es kann eine Routing-Entscheidung abgefragt werden, ohne wirklich Traffic zu senden.

```
# ip neigh show
192.168.178.1 dev eth0 lladdr 00:16:1d:5d:97:40 STALE

# ip route get 173.194.32.223
173.194.32.223 via 192.168.178.1 dev eth0 \
  src 192.168.178.42
```

Neues *contd.*

Es können Tun/Tap-Interfaces und VLAN-Interfaces erzeugt werden. Advanced Routing (Policy Routing) kann konfiguriert werden. Dazu noch zig andere Sachen, die aus Faulheit nicht in diesen Folien stehen :-)

```
# ip tuntap add dev tap0 mode tap user 1000

# ip link add link eth0 name eth0.100 type vlan id 100

# ip rule add from all iif tap0 lookup 100
# ip route add default via 1.1.1.1 table 100
```