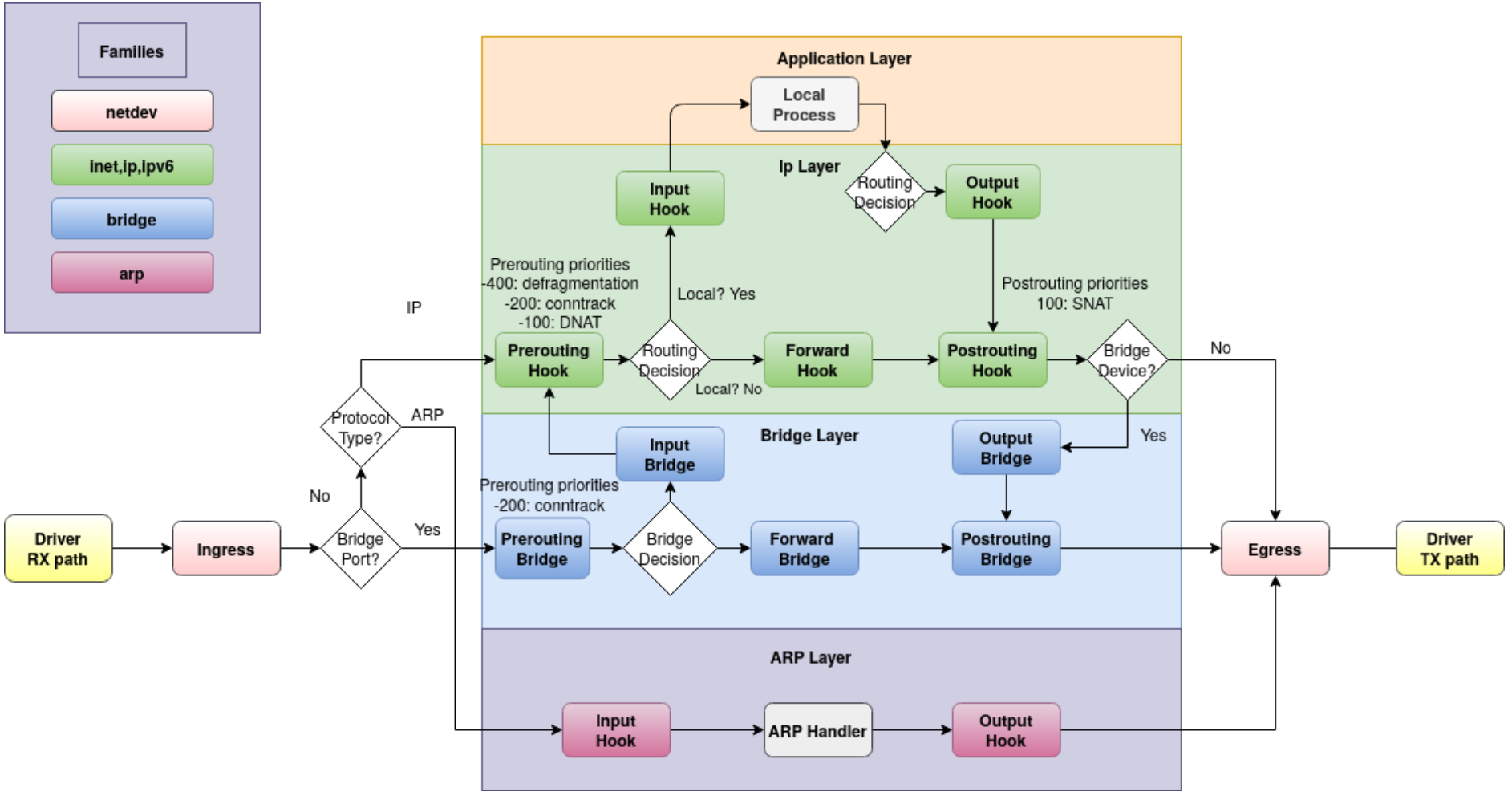
# Einführung in nftables

# Was ist nftables?

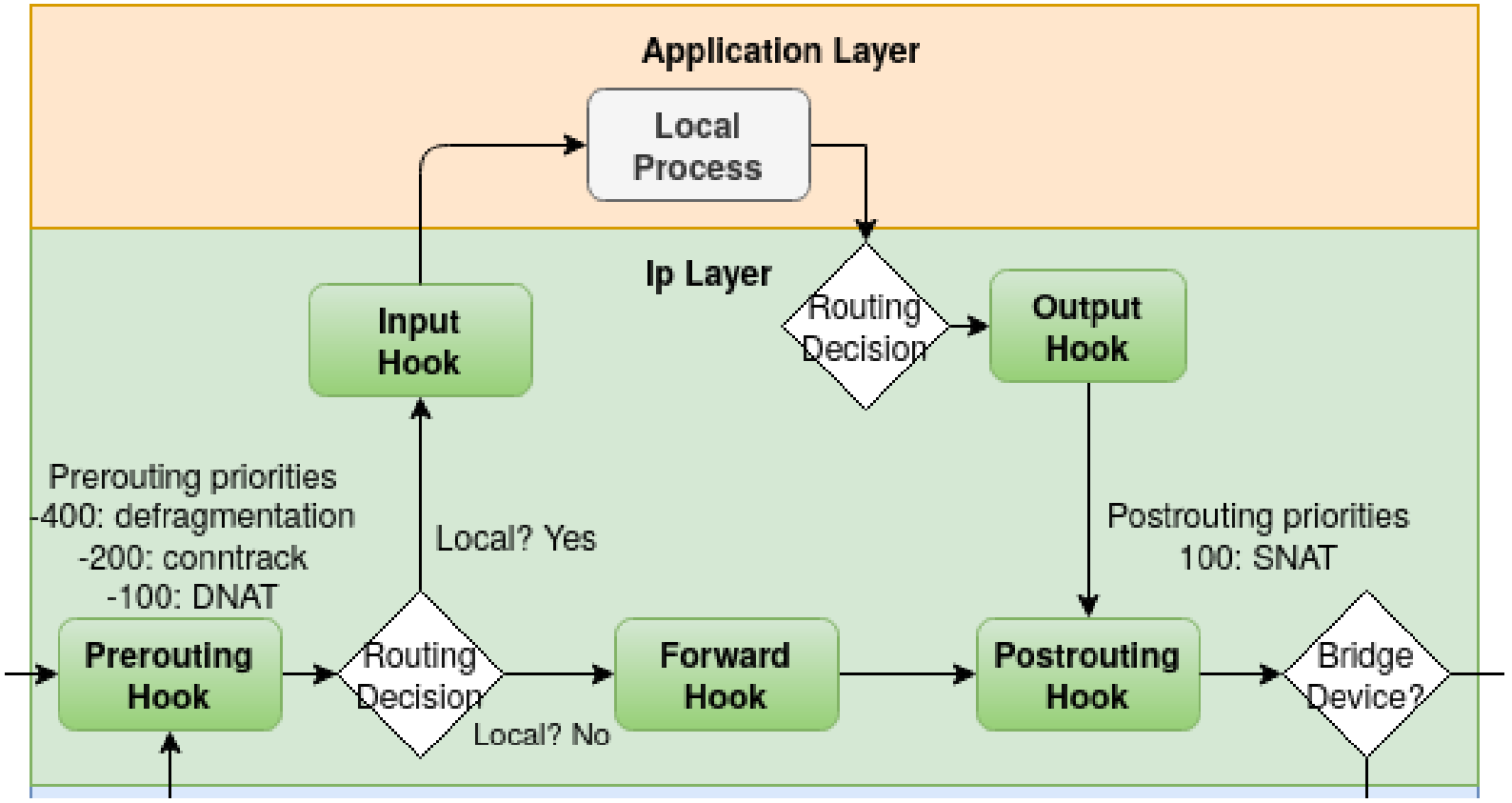
- Paketfilter für Linux
- Nachfolger von
  - iptables
  - ip6tables
  - arptables
  - ebtables

# Warum nftables?

- Einheitliches Interface (nft)
- Einfacherer Syntax
- Gemeinsame Regeln für IPv4 und IPv6
- Atomare Änderungen
- Bessere Performance




Quelle: https://people.netfilter.org/pablo/nf-hooks.png




Quelle: https://people.netfilter.org/pablo/nf-hooks.png

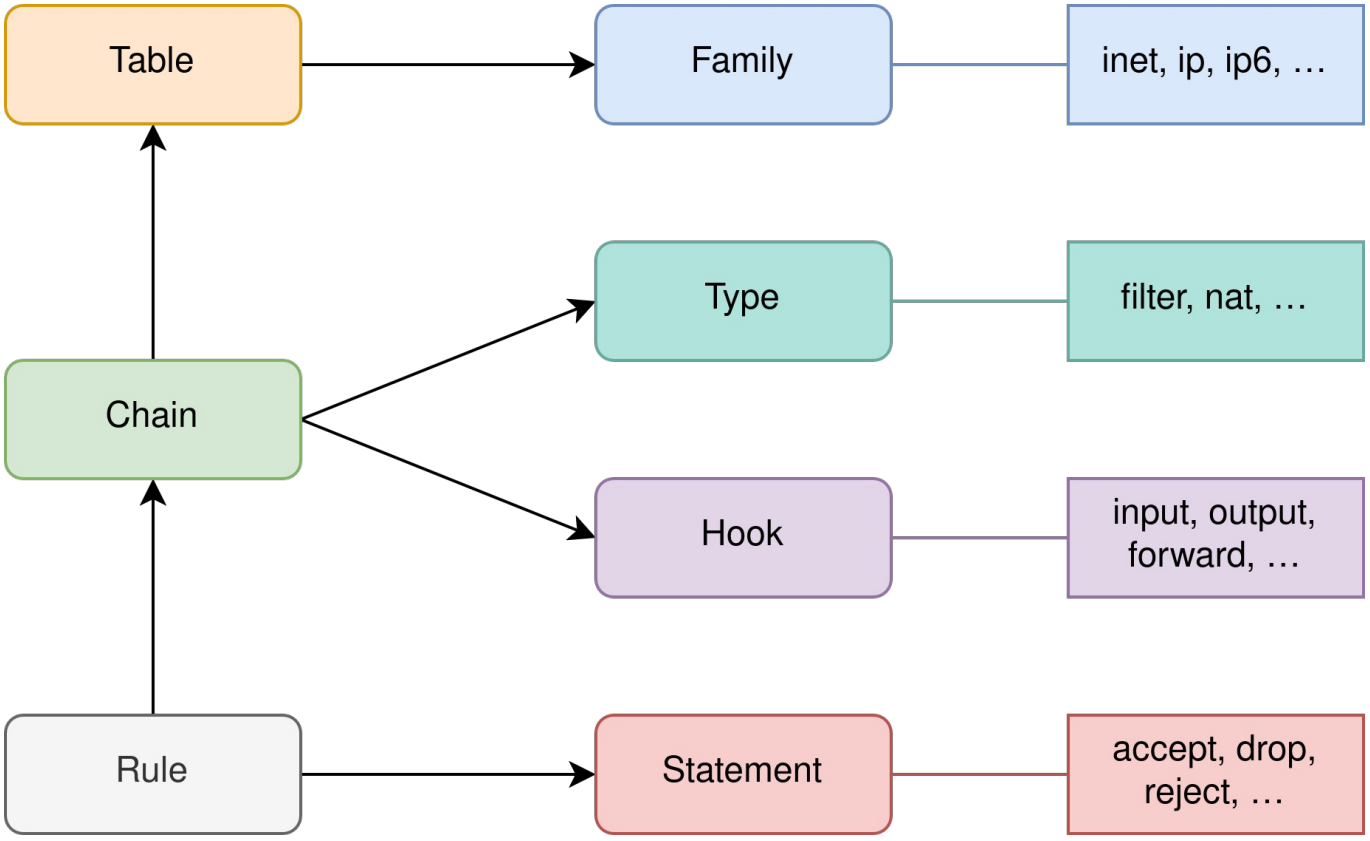


Table
Family
inet, ip, ip6, …
Type
filter, nat, …
Chain
Hook
input, output,
forward, …
Rule
Statement
accept, drop,
reject, …

# nftables verwenden

# Table anlegen

```
nft add table inet filter
```

```
nft add table <family> <name>
```

family: ip, ip6, inet (ip + ip6), arp, bridge, netdev

# Table anzeigen

```
nft list tables

table inet filter
```

# Chain anlegen

```
nft add chain inet filter input↵
{ type filter hook input priority 0\; }

nft add chain <family> <table> <name>↵
{ type <type> hook <input> priority 0\; }
```

types: filter, nat, route

hooks: prerouting, input, forward, output, postrouting

# Chain anzeigen

```
nft list chains

table inet filter {
   chain input {
      type filter hook input priority filter;
      policy accept;
   }
}
```

# Chain anzeigen

```
nft list table inet filter

table inet filter {
   chain input {
      type filter hook input priority filter;
      policy accept;
   }
}
```

# Regel anlegen

```
nft add rule inet filter input↵
   tcp dport ssh accept

nft add rule <family> <table>↵
<chain> <matches> <statements>
```

# Regel anzeigen

```
nft list chain inet filter input

table inet filter {
   chain input {
      type filter hook input priority filter;
      policy accept;
      tcp dport 22 accept
   }
}
```

# Alle Regeln anzeigen

```
nft list ruleset

table inet filter {
   chain input {
      type filter hook input priority filter;
      policy accept;
      tcp dport 22 accept
   }
}
```

# Weitere Regeln anlegen

```
nft add rule inet filter input tcp↵
dport smtp accept
```

# Weitere Regeln anlegen

```
nft add rule inet filter input tcp
dport smtp accept
```

```
nft add rule inet filter input↵
tcp dport { http, https } accept
```

# Alle Regeln anzeigen (mit Handles)

```
nft -a list ruleset

table inet filter { # handle 1
  chain input { # handle 1
    type filter hook input priority filter;
    policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 25 accept # handle 3
    tcp dport { 80, 443 } accept # handle 5
  }
}
```

# Regel löschen

```
nft delete rule inet filter input handle 3

nft -a list ruleset

table inet filter { # handle 1
  chain input { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport { 80, 443 } accept # handle 5
  }
}
```

# Regeln speichern und laden

```
nft list ruleset > nftables.conf

nft -f nftables.conf
```

# Alle Regeln anzeigen

```
nft list ruleset

table inet filter {
  chain input {
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept
    tcp dport { 80, 443 } accept
    tcp dport 22 accept
    tcp dport { 80, 443 } accept
  }
}
```

# nftabeles.conf

```
flush ruleset

table inet filter {
  chain input {
    type filter hook input priority filter;
    policy accept;
    tcp dport 22 accept
    tcp dport { 80, 443 } accept
  }
}
```

# nftabeles.conf

```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
  chain input {
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept
    tcp dport { 80, 443 } accept
  }
}
```

# Regeldatei ausführen

```
chmod +x nftables.conf

./nftables.conf
```

# Regeln beim Systemstart laden

```
mv nftables.conf /etc/nftables.conf

systemctl enable nftables
```

# Beispiel: Webserver

# Beispiel: Webserver

```
#!/usr/sbin/nft -f

flush ruleset
```

# Beispiel: Webserver

```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
  chain input {

  }
}
```

# Beispiel: Webserver

```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
  chain input {
    type filter hook input priority 0;
  }
}
```

# Beispiel: Webserver

```
table inet filter {
   chain input {
      type filter hook input priority 0;
      policy drop;
   }
}
```

# Beispiel: Webserver

```
table inet filter {
   chain input {
      type filter hook input priority 0;
      policy drop;
      iif lo accept
   }
}
```

# Beispiel: Webserver

```
table inet filter {
  chain input {
    type filter hook input priority 0;
    policy drop;
    iif lo accept
    ct state { established, related } accept
  }
}
```

# Beispiel: Webserver

```
table inet filter {
  chain input {
    […]
    iif lo accept
    ct state { established, related } accept
    icmp type echo-request accept
  }
}
```

# Beispiel: Webserver

```
table inet filter {
  chain input {
    […]
    icmp type echo-request accept
    icmpv6 type {
      echo-request, nd-router-advert,
      nd-neighbor-solicit, nd-neighbor-advert
    } accept
  }
}
```

# Beispiel: Webserver

```
table inet filter {
   chain input {
      […]
      icmpv6 type { […] } accept
      tcp dport ssh accept
   }
}
```

# Beispiel: Webserver

```
table inet filter {
   chain input {
      […]
      tcp dport ssh accept
      tcp dport { http, https } accept
   }
}
```

# Beispiel: Webserver

```
table inet filter {
  chain input {
    […]
    tcp dport ssh accept
    tcp dport { http, https } accept
    ip saddr 203.0.113.5 tcp dport nrpe accept
  }
}
```

# Beispiel: Webserver

```
table inet filter {
  chain input {
    […]
    tcp dport { http, https } accept
    ip saddr 203.0.113.5 tcp dport nrpe accept
    ip6 saddr 2001:db8::5 tcp dport nrpe accept
  }
}
```

# Beispiel: Webserver

```
table inet filter {
  chain input {
    […]
    ip saddr 203.0.113.5 tcp dport nrpe accept
    ip6 saddr 2001:db8::5 tcp dport nrpe accept
    udp dport 33434-33523 reject
  }
}
```

# Beispiel: Webserver

```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
 chain input {
  type filter hook input priority 0; policy drop;
  iif lo accept
  ct state { established, related } accept
  icmp type echo-request accept
  icmpv6 type { echo-request, nd-router-advert, nd-neighbor-solicit, nd-neighbor-advert } accept
  tcp dport ssh accept
  tcp dport { http, https } accept
  ip saddr 203.0.113.5 tcp dport nrpe accept
  ip6 saddr 2001:db8::5 tcp dport nrpe accept
  udp dport 33434-33523 reject
 }
}
```

# Beispiel: Netzwerk-Firewall

# Beispiel: Netzwerk-Firewall

```
eth0 = extern
eth1 = intern (192.0.2.0/24)
eth2 = DMZ    (198.51.100.0/24)
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain input {
    type filter hook input priority 0;
    policy drop;
    iif lo accept
    ct state { established, related } accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain input {
    […]
    ct state { established, related } accept
    icmpv6 type {
      nd-router-solicit, nd-router-advert,
      nd-neighbor-solicit, nd-neighbor-advert
    } accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain input {
    […]
    ct state { established, related } accept
    […]
    iifname "eth0" drop
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
   chain input {
      […]
      iifname "eth0" drop
      udp dport domain accept
      tcp dport domain accept
   }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain input {
    […]
    udp dport domain accept
    tcp dport domain accept
    iifname "eth1" tcp dport ssh accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain forward {
    type filter hook forward priority 0;
    policy drop;
    ct state { established, related } accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain forward {
    […]
    ct state { established, related } accept
    iifname "eth1" oifname "eth0" accept
    iifname "eth1" oifname "eth2" accept
    iifname "eth2" oifname "eth0" accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain forward {
    […]
    iifname "eth2" oifname "eth0" accept
    iifname "eth0" oifname "eth2"↵
      ip daddr 198.51.100.17↵
      tcp dport { http, https } accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain forward {
    […]
    iifname "eth0" oifname "eth2" ip […] accept
    iifname "eth0" oifname "eth2"↵
      ip6 daddr 2001:db8::a1↵
      tcp dport { http, https } accept
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain forward {
    […]
    ct state { established, related } accept
    iifname { "eth1", "eth2" } tcp dport smtp reject
    iifname "eth1" oifname "eth0" accept
    […]
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table inet filter {
  chain forward {
    […]
    ct state { established, related } accept
    ip saddr 198.51.100.32 tcp dport smtp accept
    iifname { "eth1", "eth2" } tcp dport smtp reject
    iifname "eth1" oifname "eth0" accept
    […]
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table ip nat {
  chain prerouting {
    type nat hook prerouting priority -100;
    policy accept;

    iifname "eth0" tcp dport {
      80, 443
    } dnat 198.51.100.17
  }
}
```

# Beispiel: Netzwerk-Firewall

```
table ip nat {
  chain postrouting {
    type nat hook postrouting priority 100;
    policy accept;

    oifname "eth0" ip saddr {
      192.0.2.0/24, 198.51.100.0/24
    } masquerade
  }
}
```

# Beispiel: IP-Adressen blockieren

# Beispiel: IP-Adressen blockieren

```
table ip prefilter {
  set blocklist {
    type ipv4_addr
  }

  chain input {
    type filter hook input priority -1; policy accept;
    ip saddr @blocklist reject
  }
}
```

# Beispiel: IP-Adressen blockieren

```
nft add element ip prefilter blocklist { 192.0.2.25 }

nft list set ip prefilter blocklist

table ip prefilter {
  set blocklist {
    type ipv4_addr
    elements = { 192.0.2.25 }
  }
}
```

# Fragen?

# Weitere Informationen zu nftables:

https://wiki.nftables.org/