Linux Networking Namespaces

Mehrere IP-Stacks auf einem Host

Michael Dienert

Walther-Rathenau-Gewerbeschule Freiburg

20. Juli 2021

Inhalt

Network-Namespaces

Routing

Switching

Mailsever im Container

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell → *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* → *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell → *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* → *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell → *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* → *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel, Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

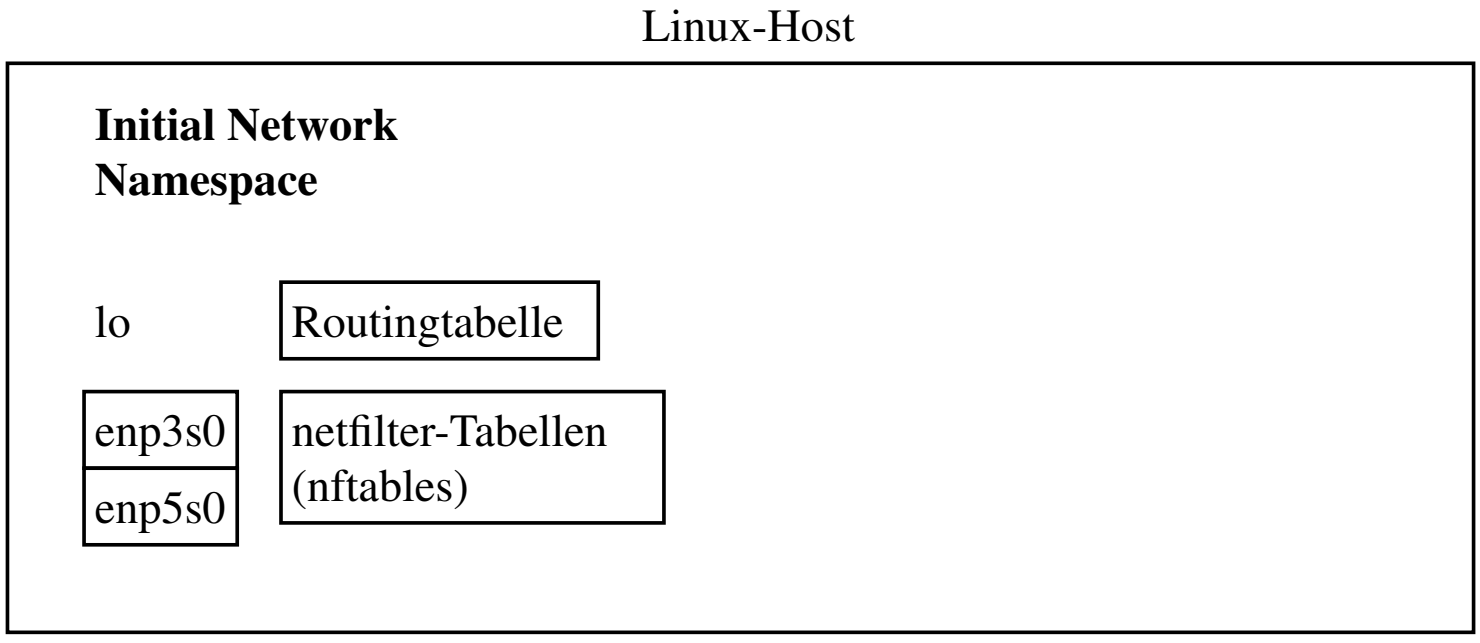
```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Erste Schritte

- Schichtenmodell $\rightarrow$ *Protokollstapel* (Network Stack) auf dem Hostrechner
- Für jedes physikalische Interface existiert ein Eintrag in der *Routingtabelle* $\rightarrow$ *eigenes Netz* (bzw. Netze).
- Network-Namespaces haben *eigene, unabhängige* Protokollstapel,Routingtabellen und netfilter-Tabellen

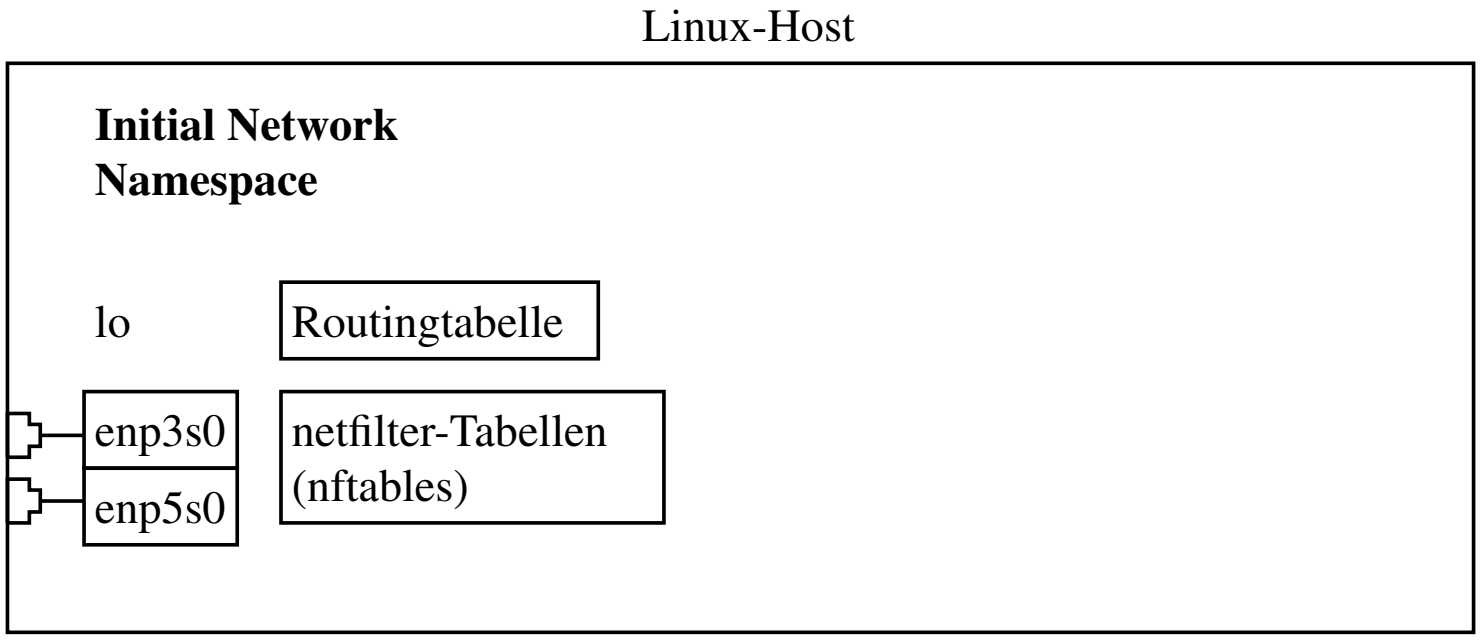
```
sudo su
ip netns add blau  #namespace hinzufuegen
ip netns add gruen #noch einer dazu
ip netns           #namespaces auflisten
ip netns list      #dito
ip netns del blau  #so kann man blau loeschen
ls /var/run/netns/
nsenter --net=/var/run/netns/blau #ns betreten
ip netns exec gruen bash          #alternativer beitritt
```

Wo sind wir jetzt?



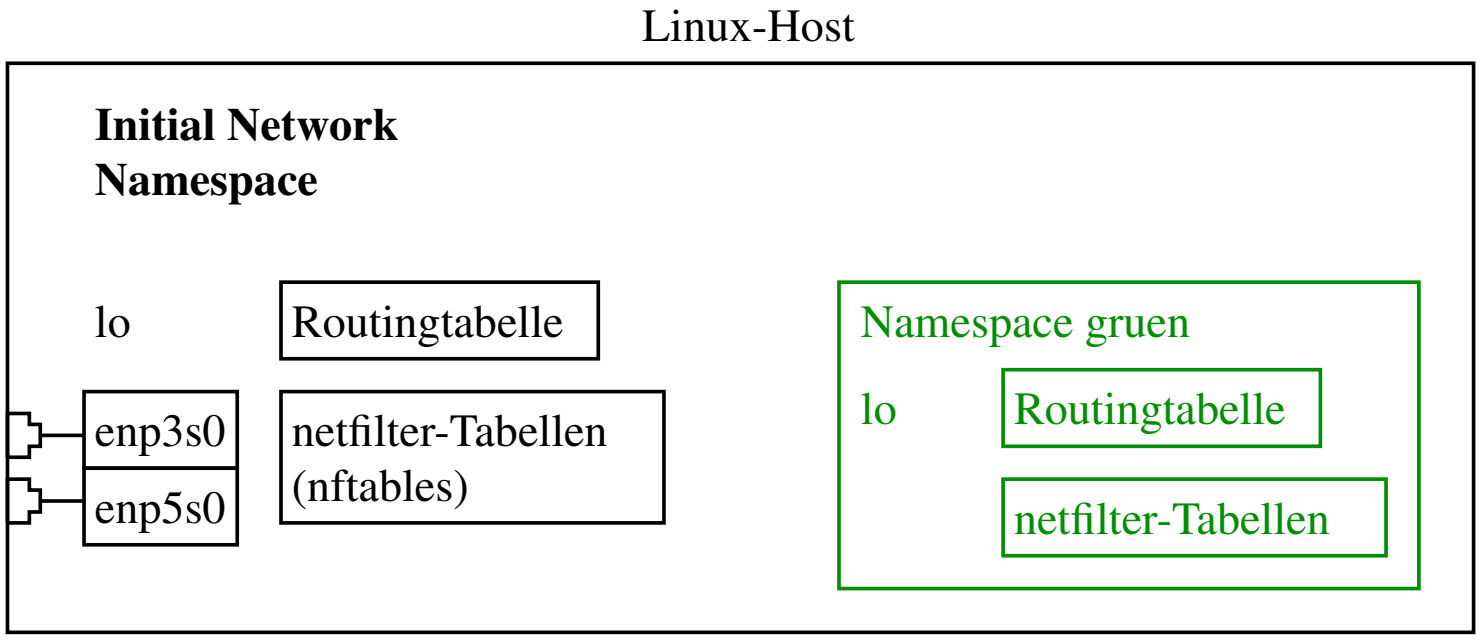
Namespace innerhalb eines Hosts

Wo sind wir jetzt?



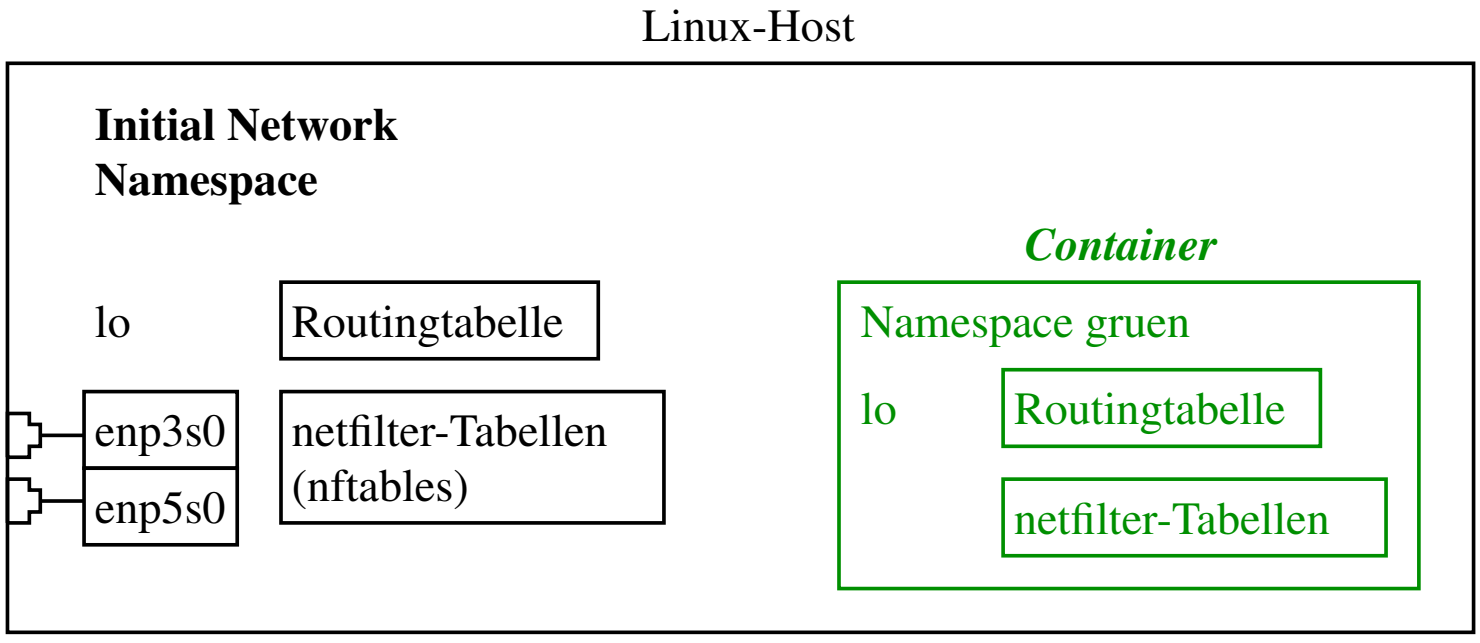
Namespace innerhalb eines Hosts

Wo sind wir jetzt?



Namespace innerhalb eines Hosts

Wo sind wir jetzt?



Namespace innerhalb eines Hosts

Virtuelle Interfaces

- Es ist möglich, die physikalischen Interfaces des Hosts mit dem Namespace zu verbinden.
- Aber (vgl. manpage): *A physical network device can live in exactly one network namespace.*
- Wenn man die physikalischen Interfaces in den Namespace *gruen* verschiebt, verschwinden sie aus dem *Initial Namespace*.
- Lösung: den Namespace mit *virtuellen Interfaces* mit der Aussenwelt verbinden.

Virtuelle Interfaces

- Es ist möglich, die physikalischen Interfaces des Hosts mit dem Namespace zu verbinden.
- Aber (vgl. manpage): *A physical network device can live in exactly one network namespace.*
- Wenn man die physikalischen Interfaces in den Namespace *gruen* verschiebt, verschwinden sie aus dem *Initial Namespace*.
- Lösung: den Namespace mit *virtuellen Interfaces* mit der Aussenwelt verbinden.

Virtuelle Interfaces

- Es ist möglich, die physikalischen Interfaces des Hosts mit dem Namespace zu verbinden.
- Aber (vgl. manpage): *A physical network device can live in exactly one network namespace.*
- Wenn man die physikalischen Interfaces in den Namespace *gruen* verschiebt, verschwinden sie aus dem *Initial Namespace*.
- Lösung: den Namespace mit *virtuellen Interfaces* mit der Aussenwelt verbinden.

Virtuelle Interfaces

- Es ist möglich, die physikalischen Interfaces des Hosts mit dem Namespace zu verbinden.
- Aber (vgl. manpage): *A physical network device can live in exactly one network namespace.*
- Wenn man die physikalischen Interfaces in den Namespace *gruen* verschiebt, verschwinden sie aus dem *Initial Namespace*.
- Lösung: den Namespace mit *virtuellen Interfaces* mit der Aussenwelt verbinden.

Virtuelle Interfaces

- Es ist möglich, die physikalischen Interfaces des Hosts mit dem Namespace zu verbinden.
- Aber (vgl. manpage): *A physical network device can live in exactly one network namespace.*
- Wenn man die physikalischen Interfaces in den Namespace *gruen* verschiebt, verschwinden sie aus dem *Initial Namespace*.
- Lösung: den Namespace mit *virtuellen Interfaces* mit der Aussenwelt verbinden.

Virtuelle Interfaces

- Es ist möglich, die physikalischen Interfaces des Hosts mit dem Namespace zu verbinden.
- Aber (vgl. manpage): *A physical network device can live in exactly one network namespace.*
- Wenn man die physikalischen Interfaces in den Namespace *gruen* verschiebt, verschwinden sie aus dem *Initial Namespace*.
- Lösung: den Namespace mit *virtuellen Interfaces* mit der Aussenwelt verbinden.

Virtuelle Interfaces

- virtuelle Interfaces werden immer *paarweise* erzeugt.
- jedem der beiden Interfaces kann eine Adresse zugewiesen werden.
- man kann das Paar wie zwei Netzwerk-Schnittstellen (NIC) betrachten, die bereits mit einem LAN-Kabel verbunden sind
- jede dieser NICs kann in einen beliebigen Network-Namespace “gesteckt” werden.

Virtuelle Interfaces

- virtuelle Interfaces werden immer *paarweise* erzeugt.
- jedem der beiden Interfaces kann eine Adresse zugewiesen werden.
- man kann das Paar wie zwei Netzwerk-Schnittstellen (NIC) betrachten, die bereits mit einem LAN-Kabel verbunden sind
- jede dieser NICs kann in einen beliebigen Network-Namespace “gesteckt” werden.

Virtuelle Interfaces

- virtuelle Interfaces werden immer *paarweise* erzeugt.
- jedem der beiden Interfaces kann eine Adresse zugewiesen werden.
- man kann das Paar wie zwei Netzwerk-Schnittstellen (NIC) betrachten, die bereits mit einem LAN-Kabel verbunden sind
- jede dieser NICs kann in einen beliebigen Network-Namespace “gesteckt” werden.

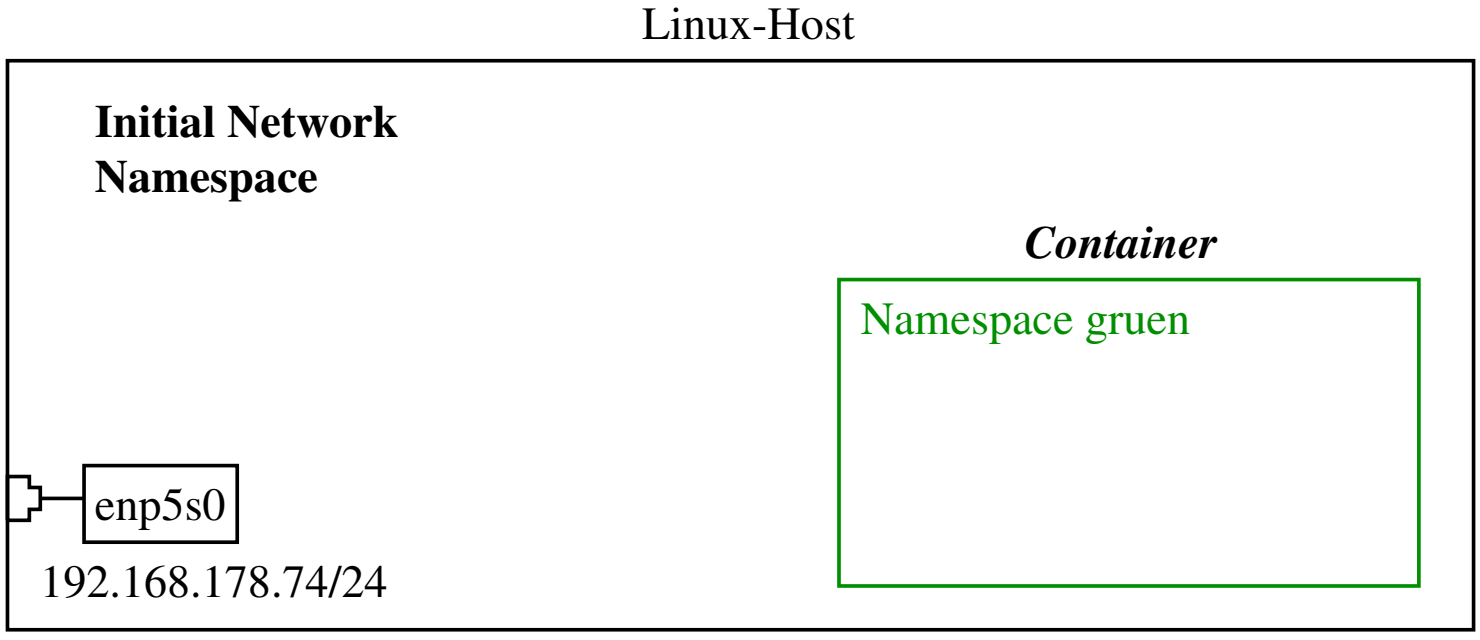
Virtuelle Interfaces

- virtuelle Interfaces werden immer *paarweise* erzeugt.
- jedem der beiden Interfaces kann eine Adresse zugewiesen werden.
- man kann das Paar wie zwei Netzwerk-Schnittstellen (NIC) betrachten, die bereits mit einem LAN-Kabel verbunden sind
- jede dieser NICs kann in einen beliebigen Network-Namespace “gesteckt” werden.

Virtuelle Interfaces

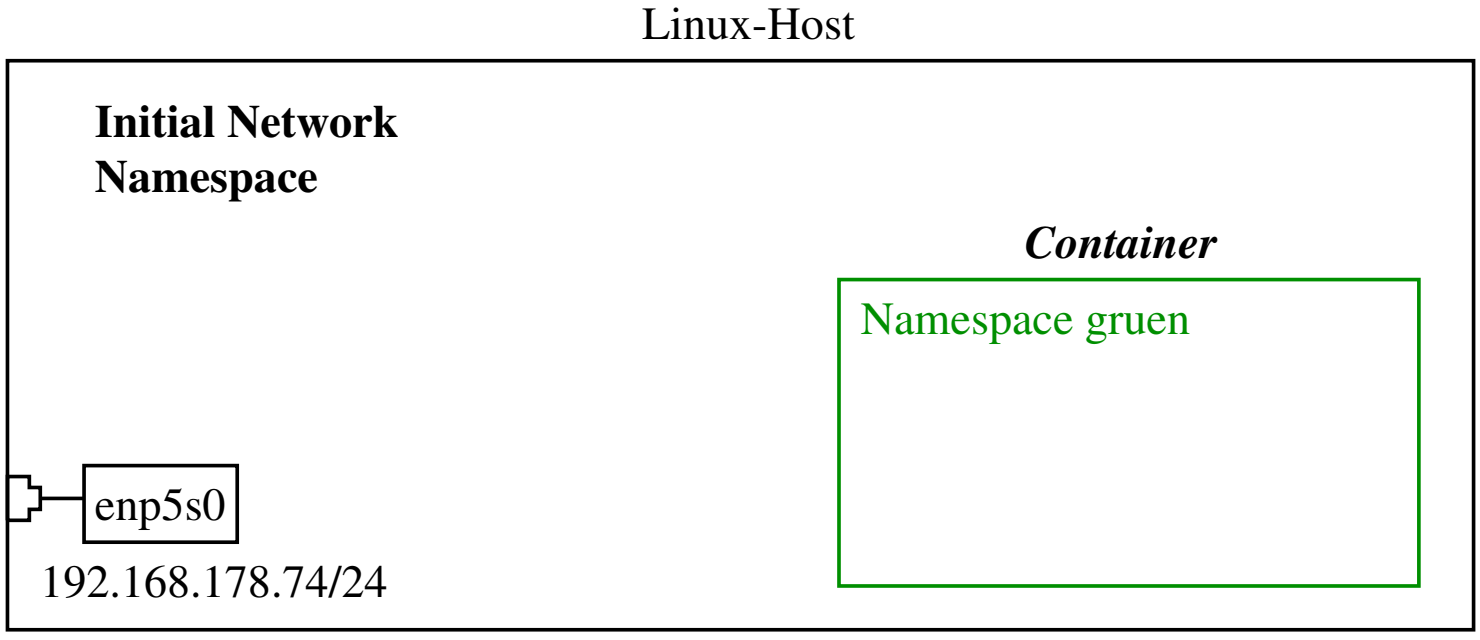
- virtuelle Interfaces werden immer *paarweise* erzeugt.
- jedem der beiden Interfaces kann eine Adresse zugewiesen werden.
- man kann das Paar wie zwei Netzwerk-Schnittstellen (NIC) betrachten, die bereits mit einem LAN-Kabel verbunden sind
- jede dieser NICs kann in einen beliebigen Network-Namespace “gesteckt” werden.

Virtuelle Interfaces



Erzeugen eines Paars virtueller Netzwerkschnittstellen

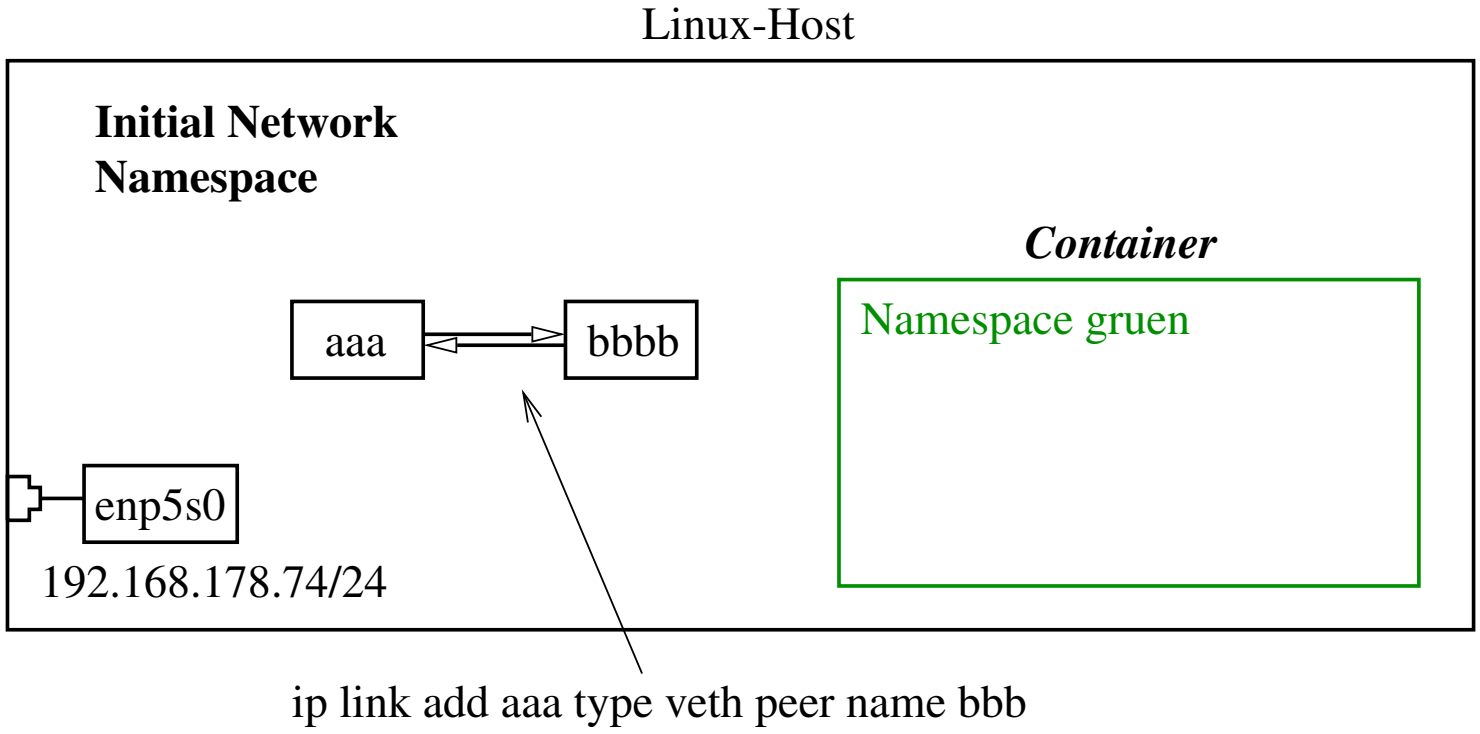
Virtuelle Interfaces



ip link add aaa type veth peer name bbb

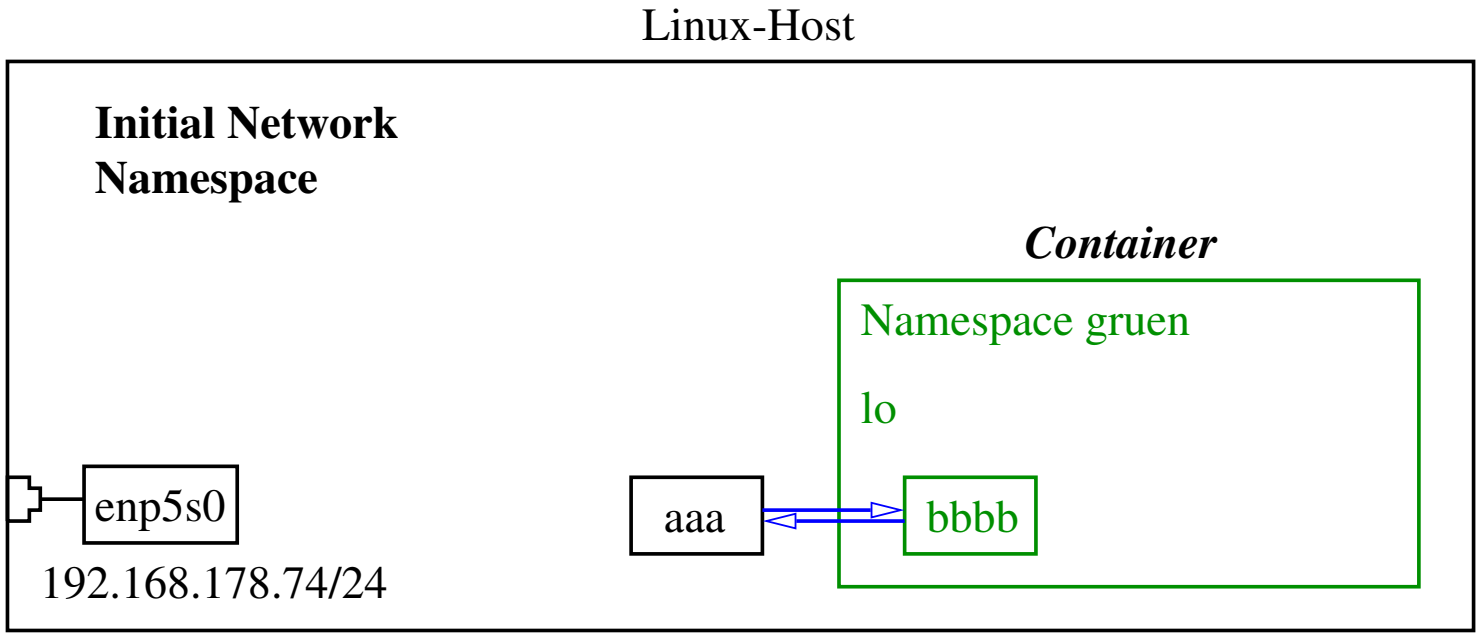
Erzeugen eines Paars virtueller Netzwerkschnittstellen

Virtuelle Interfaces



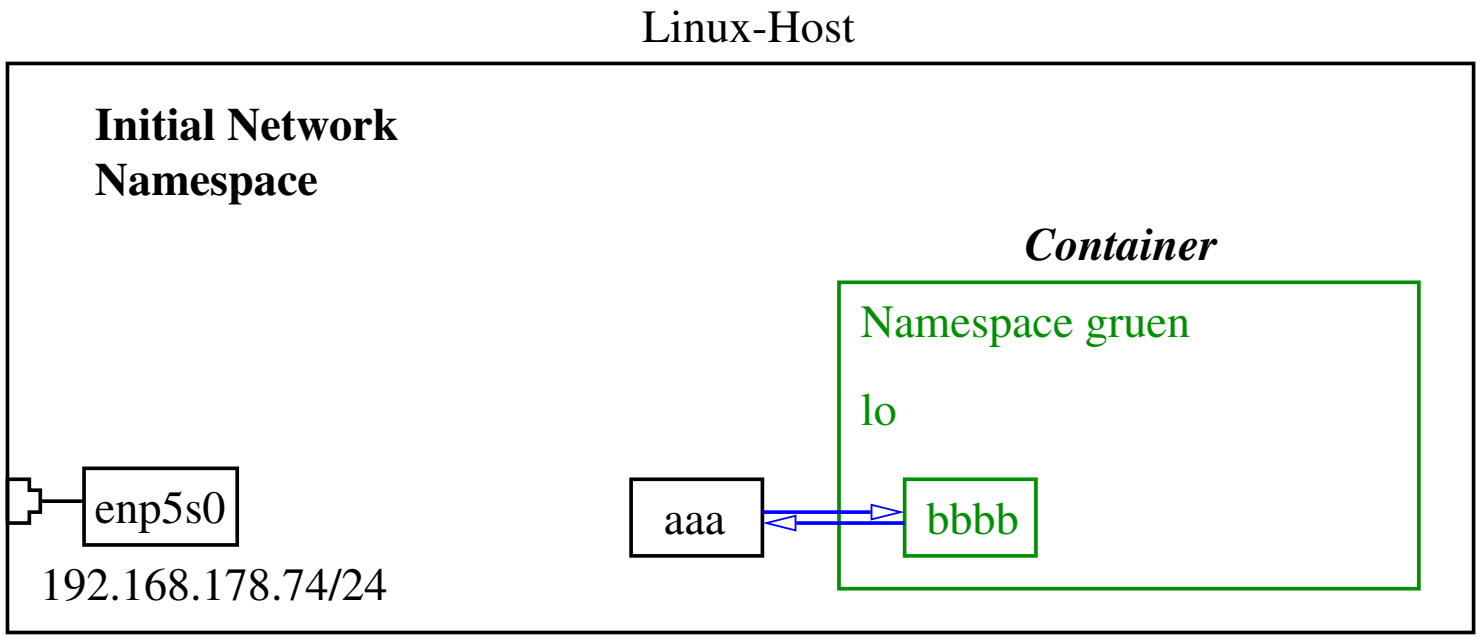
Erzeugen eines Paars virtueller Netzwerkschnittstellen

Virtuelle Interfaces



bbb wird nach gruen verschoben

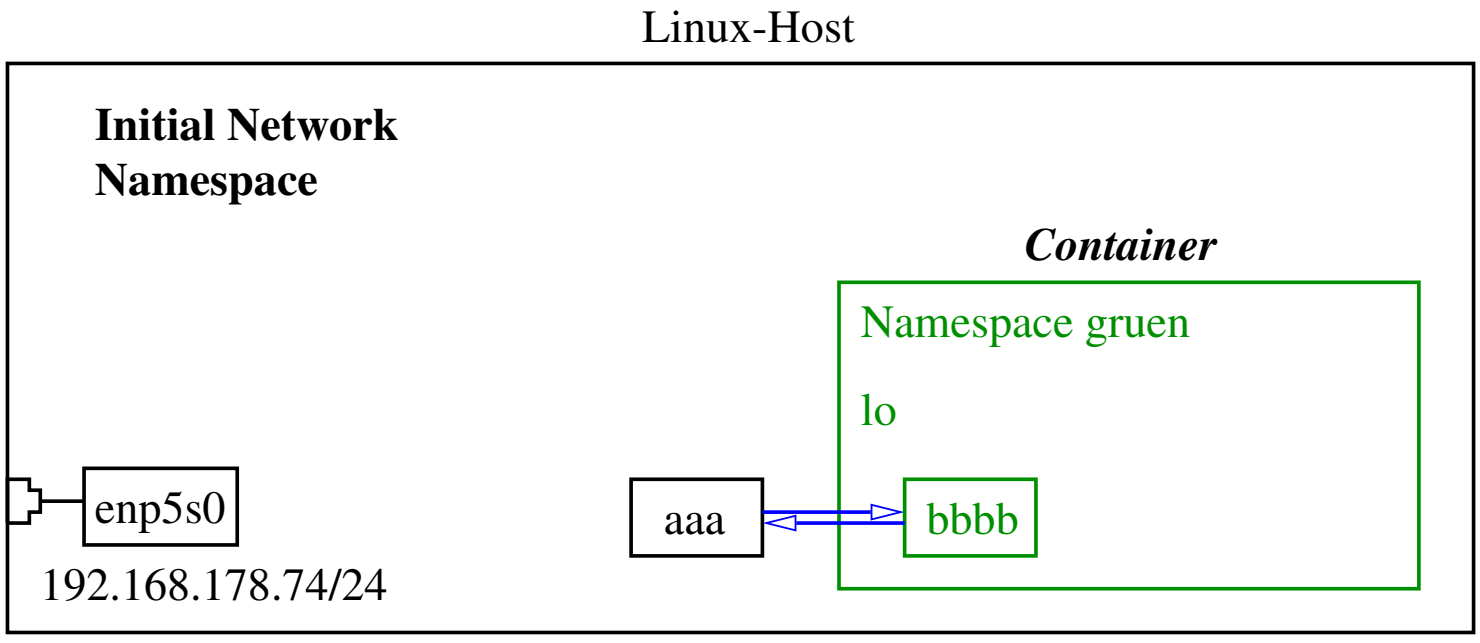
Virtuelle Interfaces



ip link set bbb netns gruen

bbb wird nach gruen verschoben

Virtuelle Interfaces

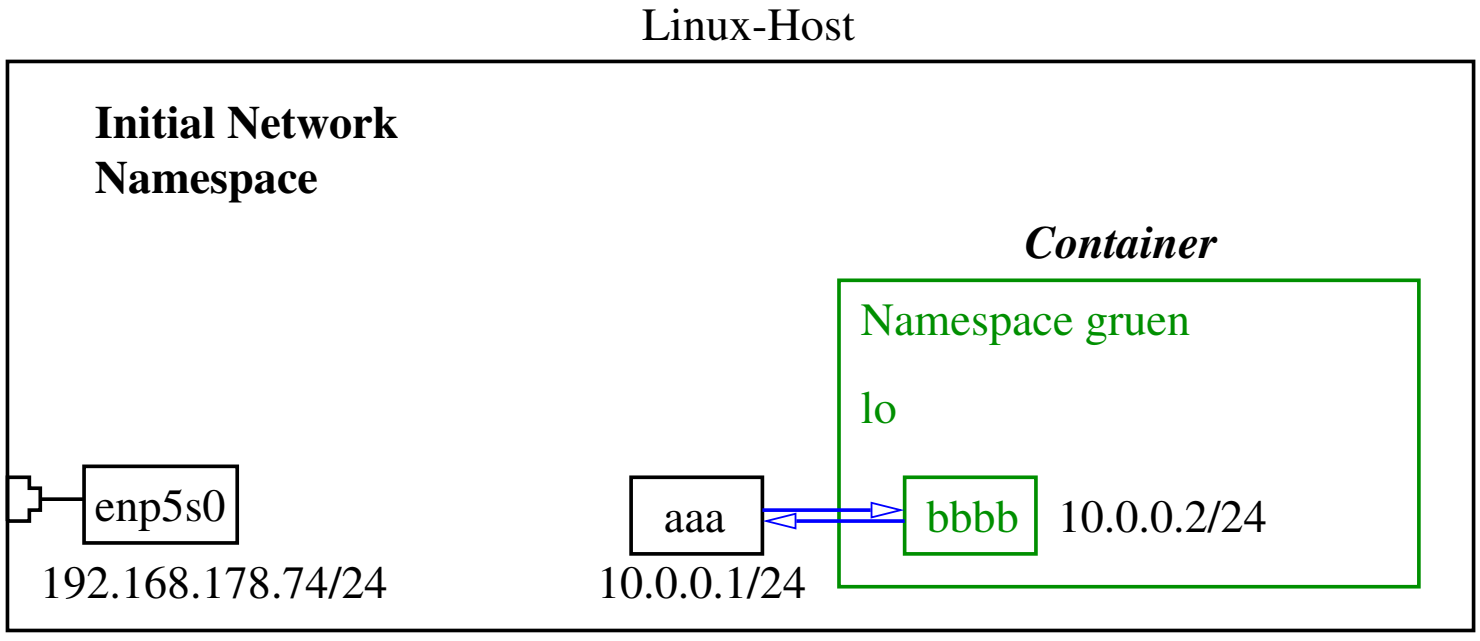


ip link set bbb netns gruen

ip addr add ... / ip link set up ...

bbb wird nach gruen verschoben

Virtuelle Interfaces



ip link set bbb netns gruen

ip addr add ... / ip link set up ...

bbb wird nach gruen verschoben

Virtuelle Interfaces

```
sudo su
ip link add aaa type veth peer name bbb
ip link show   #zwei neue interfaces werden angezeigt
#bbb wird in den NS gruen verschoben
ip link set bbb netns gruen
ip netns exec gruen bash          #zu gruen wechseln
ip link show
ip addr add 10.0.0.2/24 dev bbb
ip link set up dev bbb
exit
# nun sind wir wieder im inital NS;
# hier entsprechend aaa konfigurieren;
# gleiches netz, andere ip-adresse ...
ping 10.0.0.1 #hurra! echo replies kommen zurueck!
```

Inhalt

Network-Namespaces

Routing

Switching

Mailsever im Container

Routing zwischen Host und namespace

- Der Linux Host sieht 2 Interfaces mit je einer IP-Adresse: in seiner Routing-Tabelle tauchen beide Netze als *direkt verbunden* auf
- Innerhalb von *grün* ist das Netz 192.168.178.0/24 jedoch nicht bekannt.
- Lösung: Route hinzufügen:

```
sudo su

echo 1 > /proc/sys/net/ipv4/ip_forward
ip netns exec gruen bash
ip route add 192.168.178.0/24 via 10.0.0.1

ip route show
10.0.0.0/24 dev bbb proto kernel scope link src 10.0.0.2
192.168.178.0/24 via 10.0.0.2 dev bbb

ping 192.168.178.74 #IP des Linux-Hosts
```

Routing zwischen Host und namespace

- Der Linux Host sieht 2 Interfaces mit je einer IP-Adresse: in seiner Routing-Tabelle tauchen beide Netze als *direkt verbunden* auf
- Innerhalb von *grün* ist das Netz 192.168.178.0/24 jedoch nicht bekannt.
- Lösung: Route hinzufügen:

```
sudo su

echo 1 > /proc/sys/net/ipv4/ip_forward
ip netns exec gruen bash
ip route add 192.168.178.0/24 via 10.0.0.1

ip route show
10.0.0.0/24 dev bbb proto kernel scope link src 10.0.0.2
192.168.178.0/24 via 10.0.0.2 dev bbb

ping 192.168.178.74 #IP des Linux-Hosts
```

Routing zwischen Host und namespace

- Der Linux Host sieht 2 Interfaces mit je einer IP-Adresse: in seiner Routing-Tabelle tauchen beide Netze als *direkt verbunden* auf
- Innerhalb von *grün* ist das Netz 192.168.178.0/24 jedoch nicht bekannt.
- Lösung: Route hinzufügen:

```
sudo su

echo 1 > /proc/sys/net/ipv4/ip_forward
ip netns exec gruen bash
ip route add 192.168.178.0/24 via 10.0.0.1

ip route show
10.0.0.0/24 dev bbb proto kernel scope link src 10.0.0.2
192.168.178.0/24 via 10.0.0.2 dev bbb

ping 192.168.178.74 #IP des Linux-Hosts
```

Routing zwischen Host und namespace

- Der Linux Host sieht 2 Interfaces mit je einer IP-Adresse: in seiner Routing-Tabelle tauchen beide Netze als *direkt verbunden* auf
- Innerhalb von *grün* ist das Netz 192.168.178.0/24 jedoch nicht bekannt.
- Lösung: Route hinzufügen:

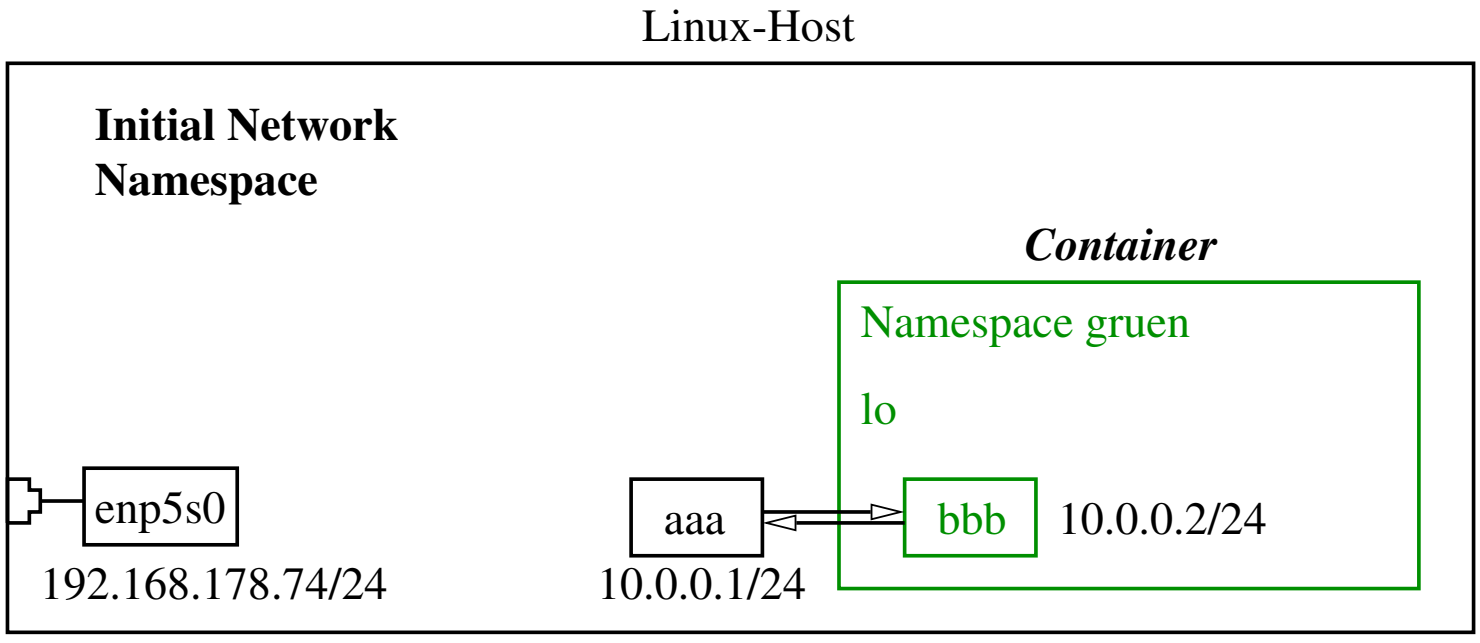
```
sudo su

echo 1 > /proc/sys/net/ipv4/ip_forward
ip netns exec gruen bash
ip route add 192.168.178.0/24 via 10.0.0.1

ip route show
10.0.0.0/24 dev bbb proto kernel scope link src 10.0.0.2
192.168.178.0/24 via 10.0.0.2 dev bbb

ping 192.168.178.74 #IP des Linux-Hosts
```

Routing zwischen Host und namespace

- Der Linux Host sieht 2 Interfaces mit je einer IP-Adresse: in seiner Routing-Tabelle tauchen beide Netze als *direkt verbunden* auf
- Innerhalb von *grün* ist das Netz 192.168.178.0/24 jedoch nicht bekannt.
- Lösung: Route hinzufügen:

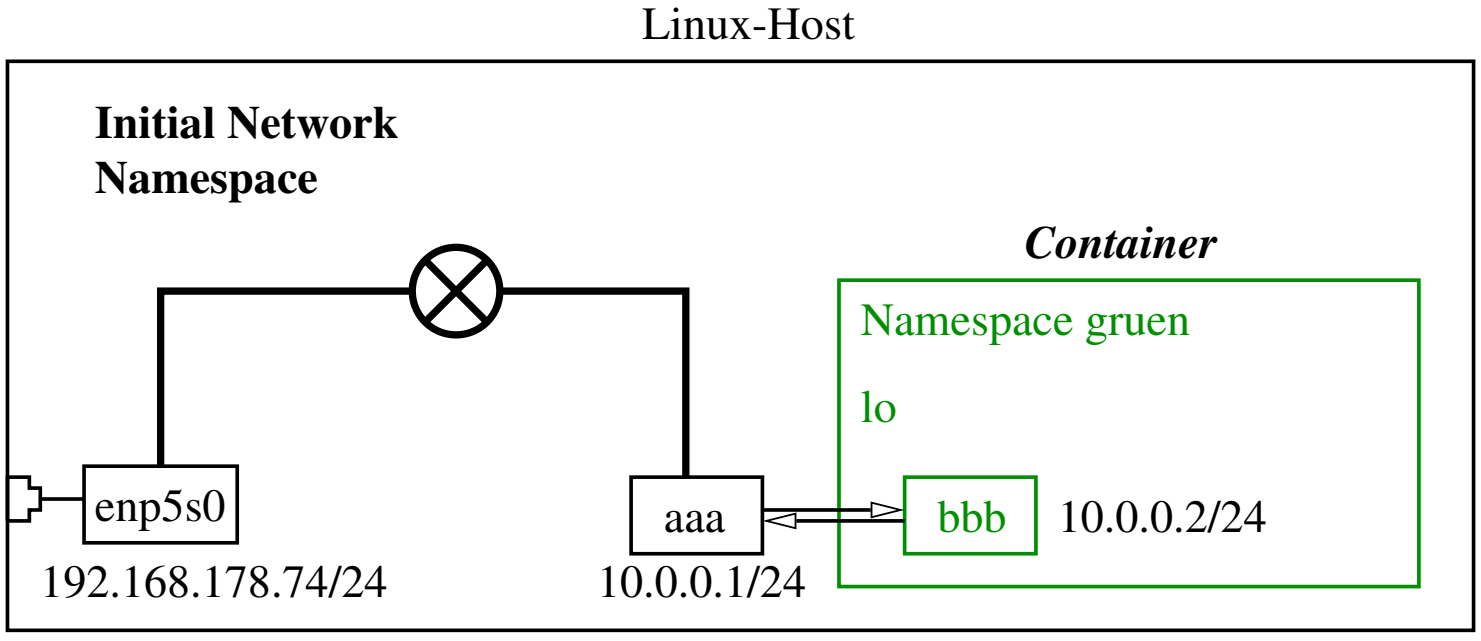
```
sudo su

echo 1 > /proc/sys/net/ipv4/ip_forward
ip netns exec gruen bash
ip route add 192.168.178.0/24 via 10.0.0.1

ip route show
10.0.0.0/24 dev bbb proto kernel scope link src 10.0.0.2
192.168.178.0/24 via 10.0.0.2 dev bbb

ping 192.168.178.74 #IP des Linux-Hosts
```

Routing zwischen Host und namespace



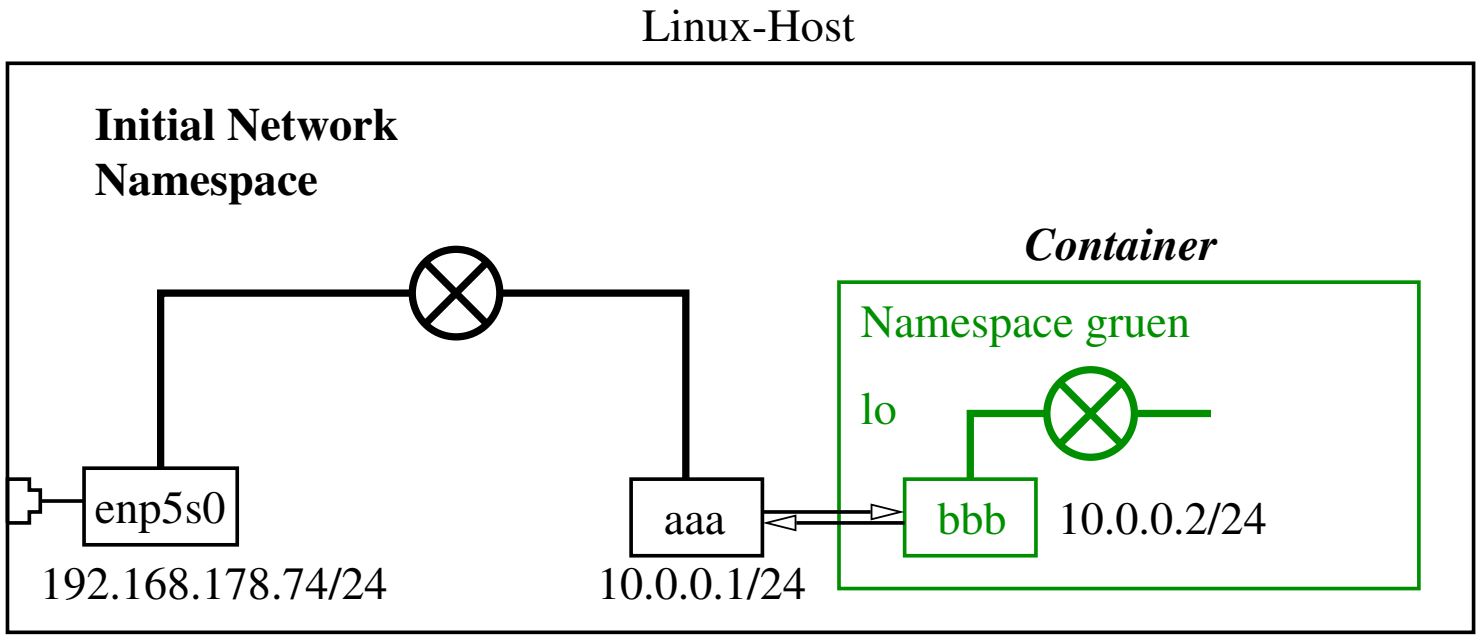
Routing zwischen Hostrechner und namespace

Routing zwischen Host und namespace



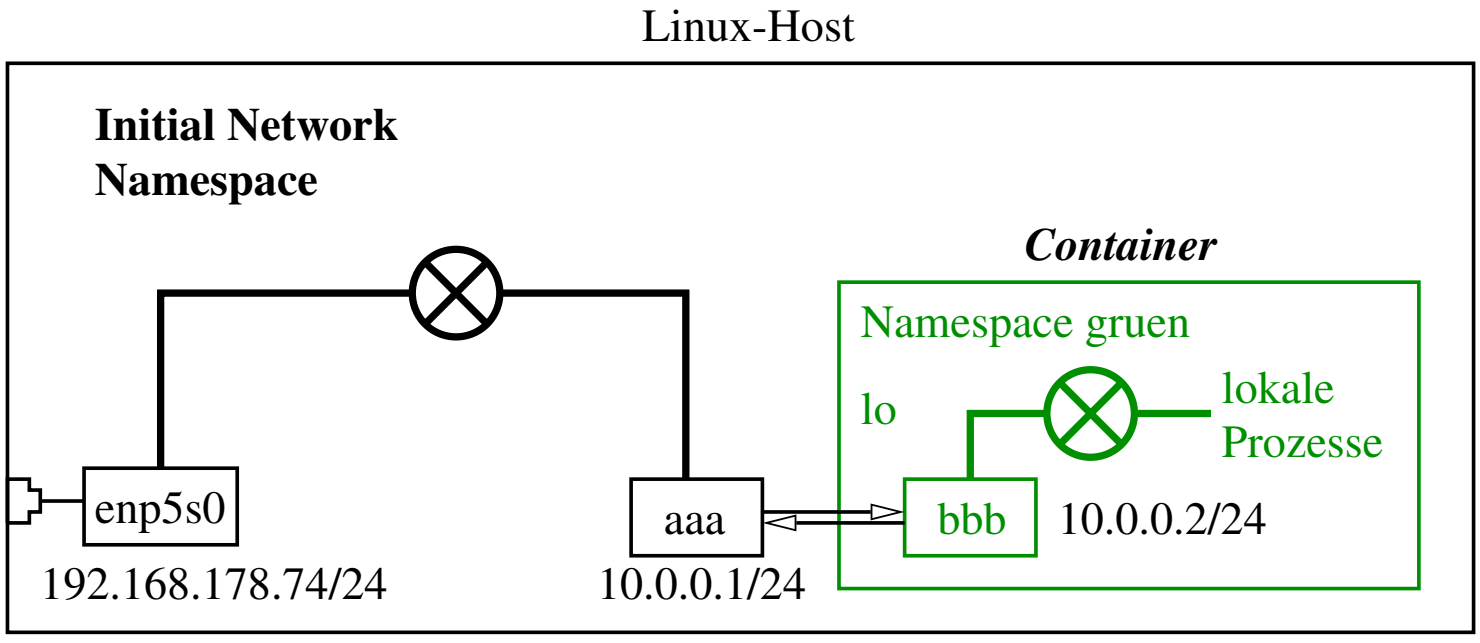
Routing zwischen Hostrechner und namespace

Routing zwischen Host und namespace



Routing zwischen Hostrechner und namespace

Routing zwischen Host und namespace



Routing zwischen Hostrechner und namespace

Routing zwischen Host und namespace

- Als Gateway-Adresse ist auch die 10.0.0.1 möglich!!
- Das Ziel-Netz muss an das Akademie-Netz des Hosts angepasst werden

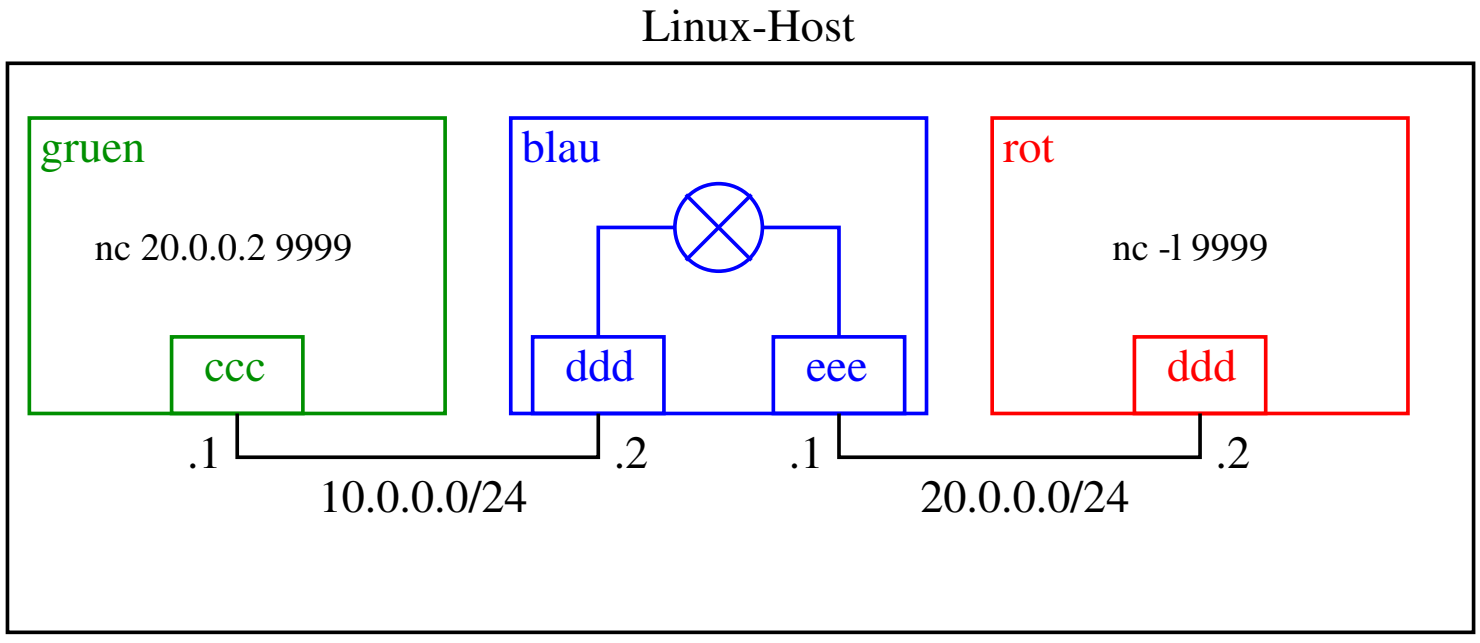
Routing zwischen Host und namespace

- Als Gateway-Adresse ist auch die 10.0.0.1 möglich!!
- Das Ziel-Netz muss an das Akademie-Netz des Hosts angepasst werden

Routing zwischen Host und namespace

- Als Gateway-Adresse ist auch die 10.0.0.1 möglich!!
- Das Ziel-Netz muss an das Akademie-Netz des Hosts angepasst werden

Routing zwischen mehreren namespaces



Daisy-Chain-Routing

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Aufgabe zu Routing

- namespaces anlegen
- Paare virtueller Interfaces erzeugen, Enden gemäss Bild in die ensprechenden namespaces verschieben, Interfaces hochfahren
- IP-Adressen vergeben
- Routingeinträge vornehmen
- Routing in den namespaces einschalten (`echo 1 > /proc/sys...`)
- mit ping testen und mit netcat (nc) einen Dienst in *rot* auf Port 9999 starten. (-l = listening)
- von *gruen* aus den Dienst in *rot* kontaktieren

Kommandosammlung

- Folgende Kommandos werden für die Aufgabe benötigt. Gezeigt sind Beispiele, Adressen müssen natürlich angepasst werden. Auch in welchem namespace das Kommando ausgeführt werden muss ist entscheidend.
- Superbequem ist es, in einem Terminal 4 nebeneinander liegende Tabs zu öffnen (shift-ctrl-t) und dort jeweils den entsprechenden namespace zu betreten.

Kommandosammlung

- Folgende Kommandos werden für die Aufgabe benötigt. Gezeigt sind Beispiele, Adressen müssen natürlich angepasst werden. Auch in welchem namespace das Kommando ausgeführt werden muss ist entscheidend.
- Superbequem ist es, in einem Terminal 4 nebeneinander liegende Tabs zu öffnen (shift-ctrl-t) und dort jeweils den entsprechenden namespace zu betreten.

Kommandosammlung

- Folgende Kommandos werden für die Aufgabe benötigt. Gezeigt sind Beispiele, Adressen müssen natürlich angepasst werden. Auch in welchem namespace das Kommando ausgeführt werden muss ist entscheidend.
- Superbequem ist es, in einem Terminal 4 nebeneinander liegende Tabs zu öffnen (shift-ctrl-t) und dort jeweils den entsprechenden namespace zu betreten.

Kommandosammlung

```
sudo su
#namespace erzeugen, namespaces auflisten
ip netns add rot
ip netns list

#veth erzeugen
ip link add ccc type veth peer name ddd

#enden verschieben
ip link set ccc netns gruen

#in ns wechseln; mit 'exit' wieder verlassen
ip netns exec rot bash

#veth hochfahren, adresse vergeben, route eintragen
ip link set up fff
ip addr add 20.0.0.2/24 dev fff
ip route add 10.0.0.0/24 via 20.0.0.1

#netcat als server und client starten
nc -l 9999
nc 20.0.0.2 9999

#bischen was tippen und ein bier aufmachen,
#wenn es auf der gegenseite ankommt (der text, nicht das bier)
```

Inhalt

Network-Namespaces

Routing

Switching

Mailsever im Container

Durchreichen des physikalischen Netzwerks

- Mit einem virtuellen Switch (Bridge) kann man Namespaces mit dem äusseren, physikalischen Netzwerk verbinden
- Die Interfaces im namespace können dhcp-Requests versenden
- Dienste im namespace können von aussen direkt erreicht werden

Durchreichen des physikalischen Netzwerks

- Mit einem virtuellen Switch (Bridge) kann man Namespaces mit dem äusseren, physikalischen Netzwerk verbinden
- Die Interfaces im namespace können dhcp-Requests versenden
- Dienste im namespace können von aussen direkt erreicht werden

Durchreichen des physikalischen Netzwerks

- Mit einem virtuellen Switch (Bridge) kann man Namespaces mit dem äusseren, physikalischen Netzwerk verbinden
- Die Interfaces im namespace können dhcp-Requests versenden
- Dienste im namespace können von aussen direkt erreicht werden

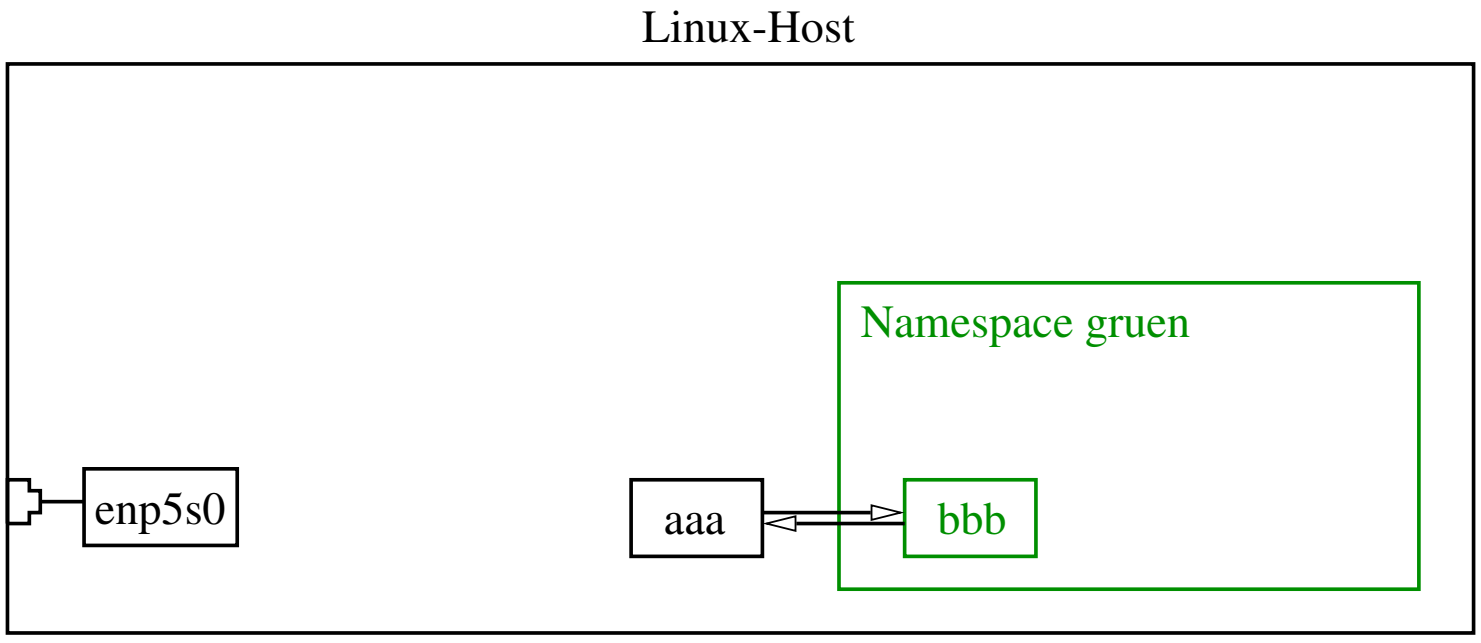
Durchreichen des physikalischen Netzwerks

- Mit einem virtuellen Switch (Bridge) kann man Namespaces mit dem äusseren, physikalischen Netzwerk verbinden
- Die Interfaces im namespace können dhcp-Requests versenden
- Dienste im namespace können von aussen direkt erreicht werden

Durchreichen des physikalischen Netzwerks

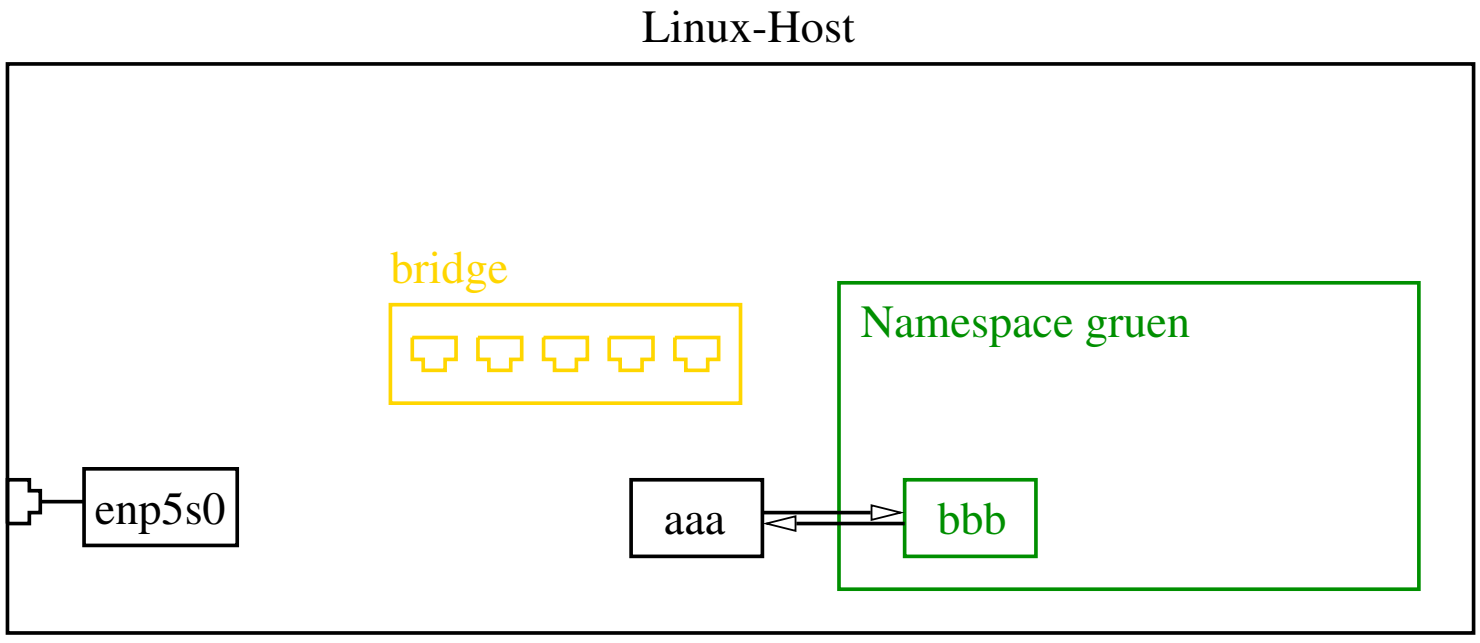
```
sudo su
#alte adressen ggfs. loeschen
ip addr del <ip> dev <schnittstelle>

ip link add name bridge type bridge
ip link set bridge up
ip link add zzz type veth peer name yyy

ip link set aaa master bridge
ip link set zzz master bridge
ip link set enp5s0 master bridge
#wieder entfernen: ip link set <if-name> nomaster
bridge link   #auflisten, was eingestoepselt ist

dhclient yyy
#das gleiche auch im NS gruen fuer bbb wiederholen
```

Durchreichen des physikalischen Netzwerks



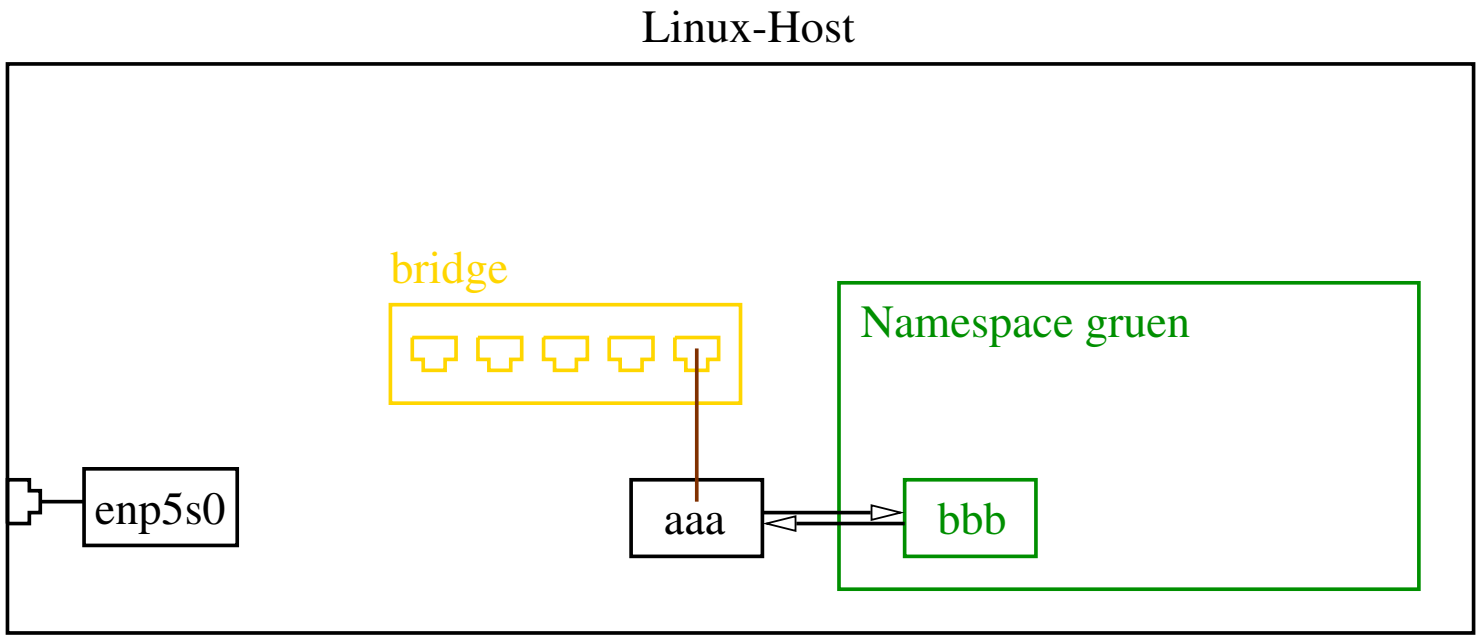
Routing zwischen Hostrechner und namespace

Durchreichen des physikalischen Netzwerks



Routing zwischen Hostrechner und namespace

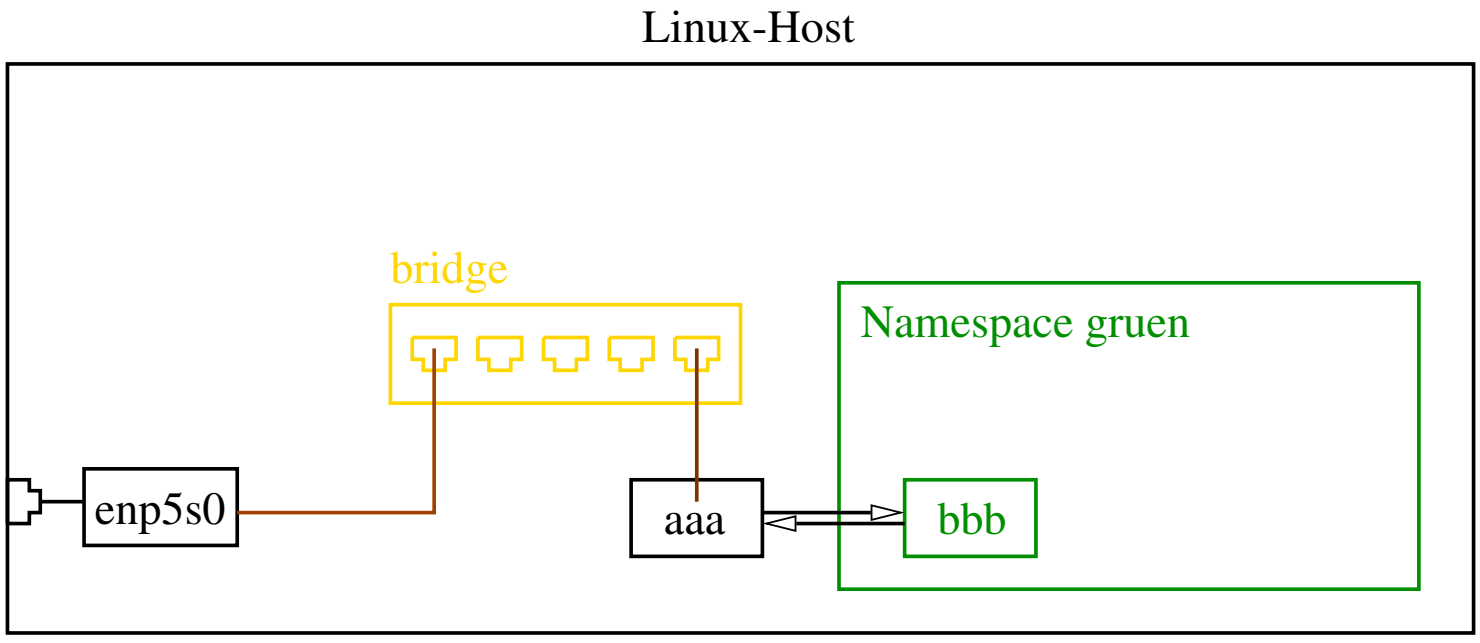
Durchreichen des physikalischen Netzwerks



ip link set aaa master bridge

Routing zwischen Hostrechner und namespace

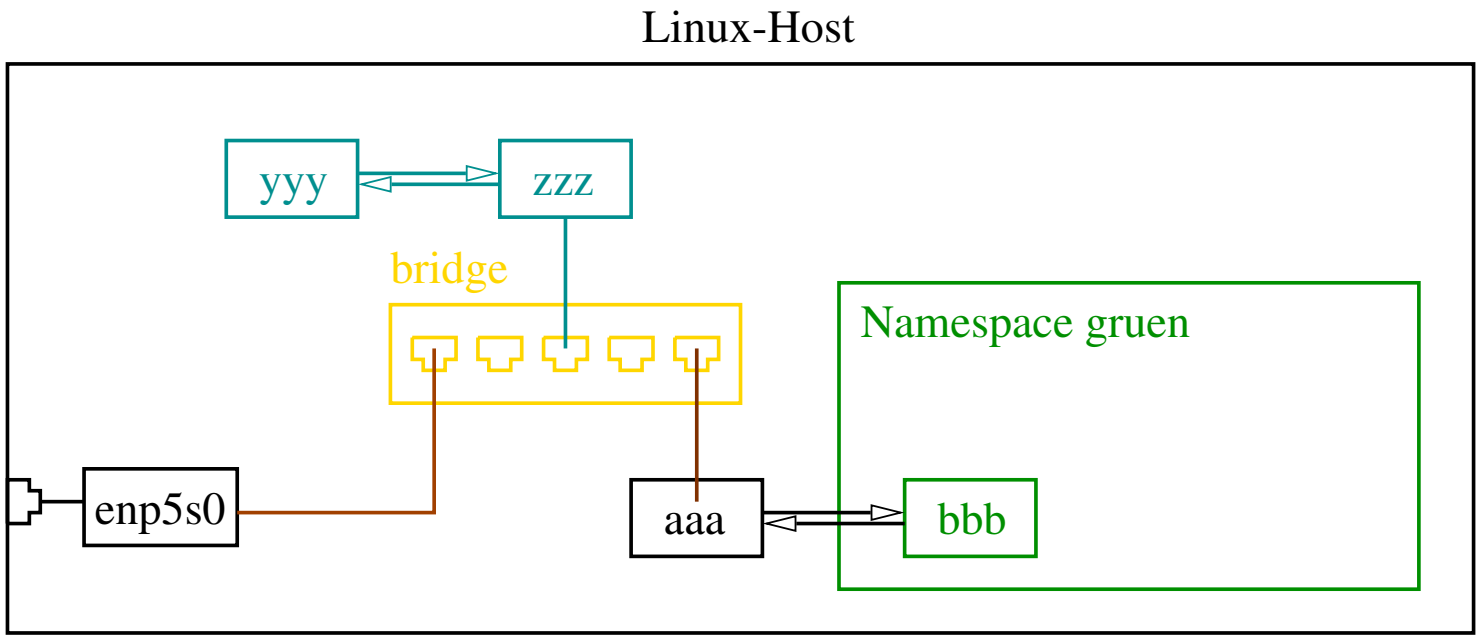
Durchreichen des physikalischen Netzwerks



ip link set aaa master bridge

Routing zwischen Hostrechner und namespace

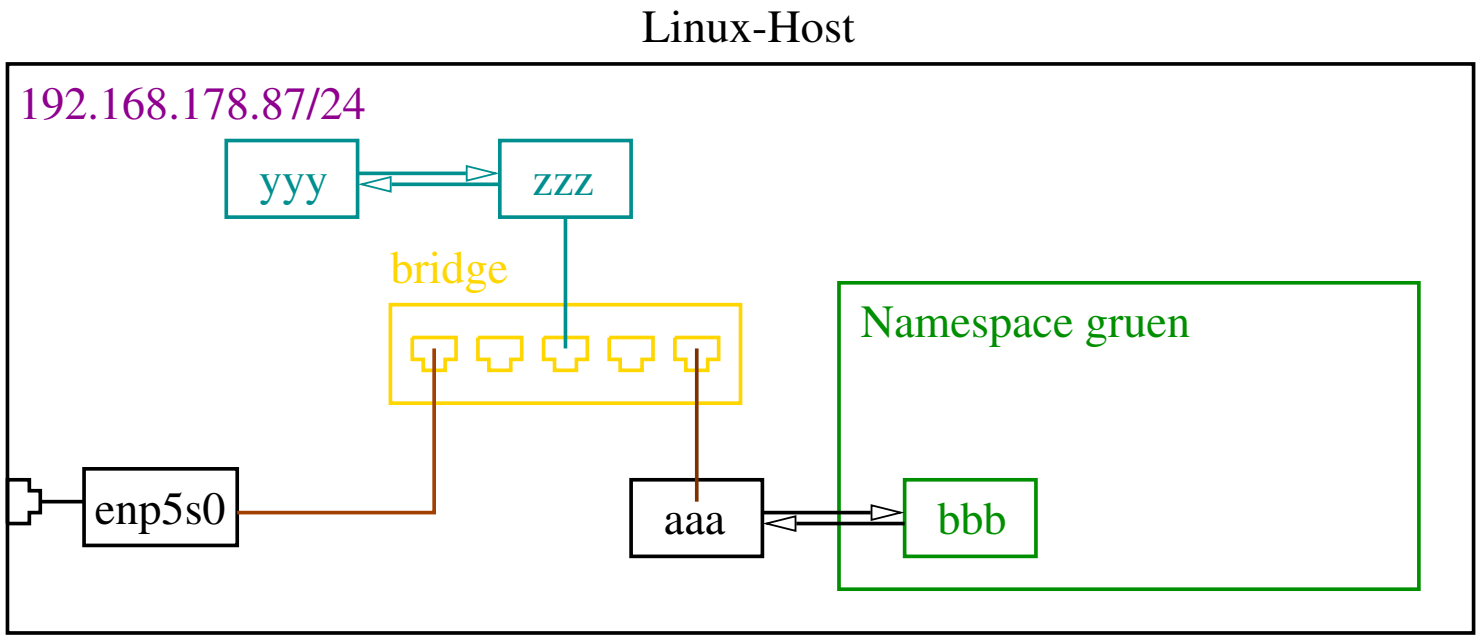
Durchreichen des physikalischen Netzwerks



ip link set aaa master bridge

Routing zwischen Hostrechner und namespace

Durchreichen des physikalischen Netzwerks

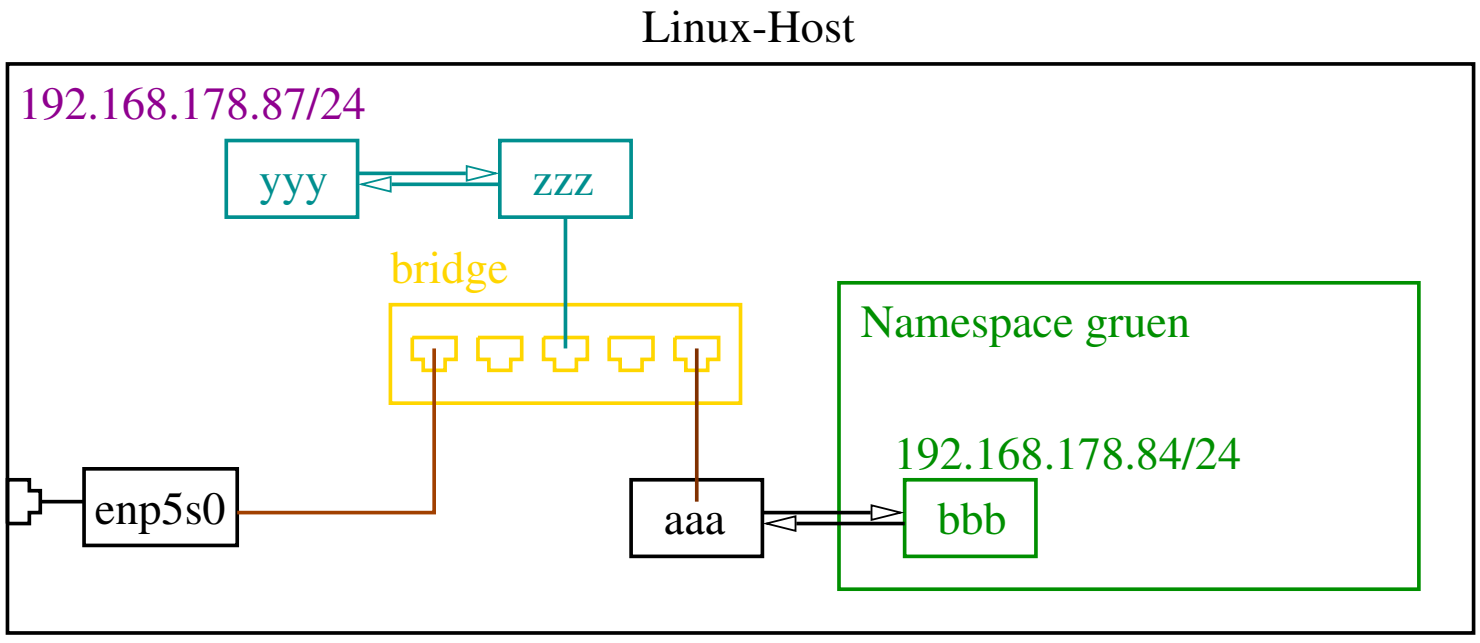


ip link set aaa master bridge

dhclient yyy

Routing zwischen Hostrechner und namespace

Durchreichen des physikalischen Netzwerks



ip link set aaa master bridge

dhclient yyy

dhclient bbb

Routing zwischen Hostrechner und namespace

Inhalt

Network-Namespaces

Routing

Switching

Mailsever im Container

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Nur zu Demonstationszwecken opensmtpd

- Zu Demonstrationszwecken soll ein smtp-Server im namespace gestartet werden
- der Autor hat sich fuer *opensmtpd* entschieden, da der problemlos im Vordergrund gestartet werden kann
- *opensmtpd* installieren, Maildomain festlegen
- *opensmtpd* Dienst stoppen, wir starten später im Vordergrund (s.u.)
- einfachste config-datei:

```
listen on bbb port 25
accept from any for local deliver to maildir\
"/tmp/maildir"
```

- Serverstart (-T -> trace):

```
smtpd -d -f smtpd.conf -T smtp
```

Durchreichen des physikalischen Netzwerks

```
nc 10.0.0.2 25
220 debbie10.dienert.eu ESMTP OpenSMTPD
HELO defaultNS
250 debbie10.dienert.eu Hello defaultNS [10.0.0.1],
    pleased to meet you
MAIL FROM: abgelascht@ausgebufft
553 5.1.0: Sender address syntax error
MAIL FROM: <abgelascht@ausgebufft>
250 2.0.0: Ok
RCPT TO: <micha@dienert.eu>    #abhaengig von maildomain
250 2.1.5 Destination address valid: Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Subject: ohne header tut er's nicht

der abgelaschte armin und der ausgebuffte markus:
solche helden braucht das land!
.
250 2.0.0: baabedc4 Message accepted for delivery
quit
221 2.0.0: Bye
```