An introduction to iproute2 and friends

Davide Pesavento

davide.pesavento@lip6.fr

December 18th, 2014

Warning

You need to have superuser (root) privileges, or at least the CAP_NET_ADMIN and CAP_SYS_ADMIN capabilities, for the majority of the commands in these slides to work.

Linux network namespaces (netns)

- Network namespaces are isolated network stack instances within a single machine.
- They can be used for security domain separation, managing traffic flows between virtual machines or containers, and so on.
- Every namespace is a complete copy of the networking stack with its own interfaces, addresses, routes, etc…
- You can run processes inside a namespace.

Linux network namespaces (netns)

- Each namespace has at least a loopback interface (`lo`).
- You can bridge namespaces together or to a physical network interface.
- Physical interfaces cannot be assigned to a namespace, only virtual interfaces can.
- Virtual interfaces can be “moved” between namespaces.

Linux network namespaces (netns)

```
$ ip netns help
Usage: ip netns list
       ip netns add NAME
       ip netns delete NAME
       ip netns identify PID
       ip netns pids NAME
       ip netns exec NAME cmd ...
       ip netns monitor
```

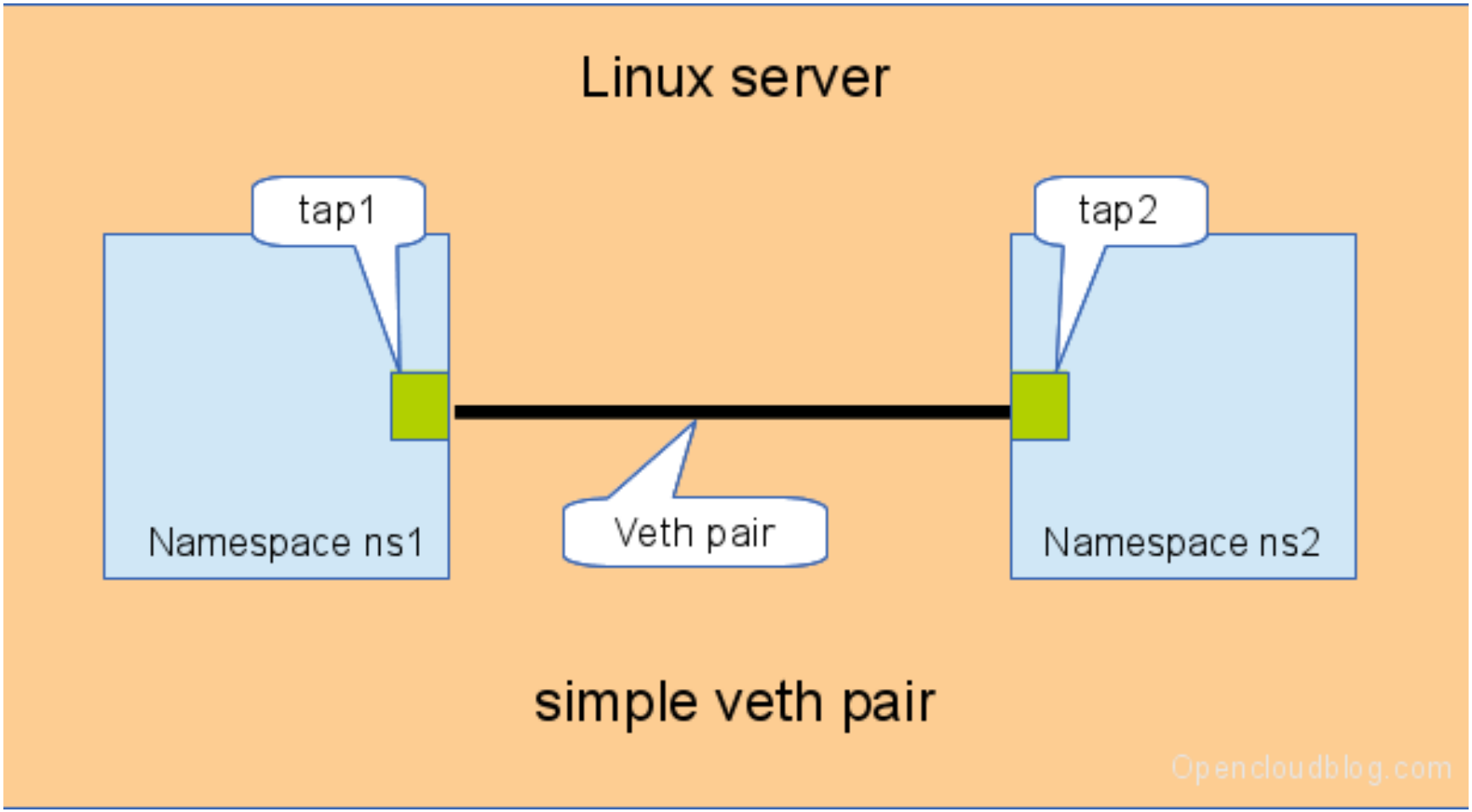
See also ip-netns(8) man page.

Virtual Ethernet interfaces (veth)

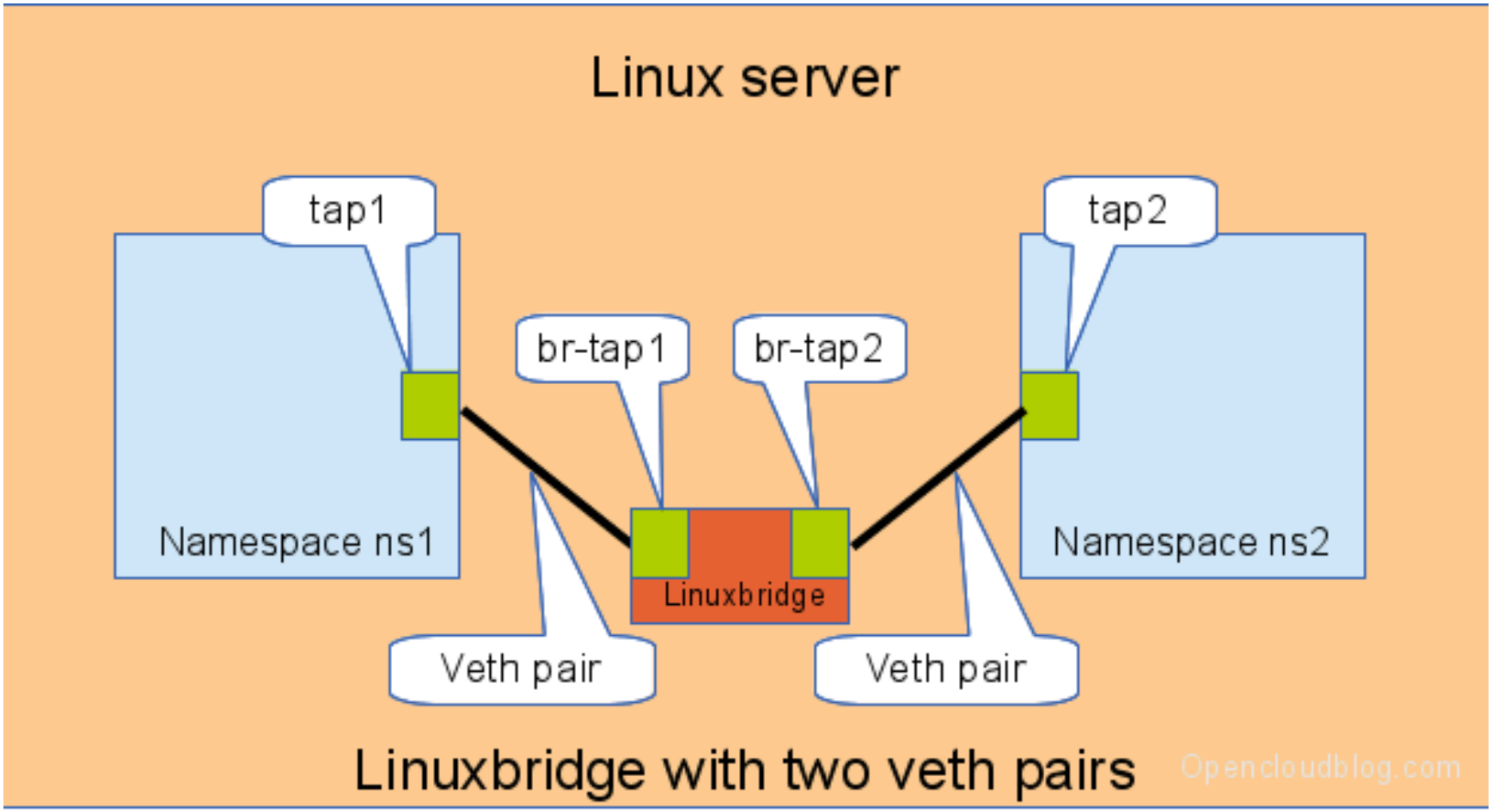
- veth interfaces are virtual Ethernet-like network interfaces, always created in pairs
- They have an auto-generated MAC address
- Packets sent on one interface exit from the other one, and vice versa
- They are created with: `ip link add type veth`
- E.g.: `ip link add ve1 type veth peer name ve2`

```
18: ve2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode
DEFAULT group default qlen 1000
    link/ether d2:82:a9:35:d7:f4 brd ff:ff:ff:ff:ff:ff
19: ve1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode
DEFAULT group default qlen 1000
    link/ether 7e:44:94:c5:19:d3 brd ff:ff:ff:ff:ff:ff
```

Veth pair between two namespaces



Linux virtual bridge and veth pairs

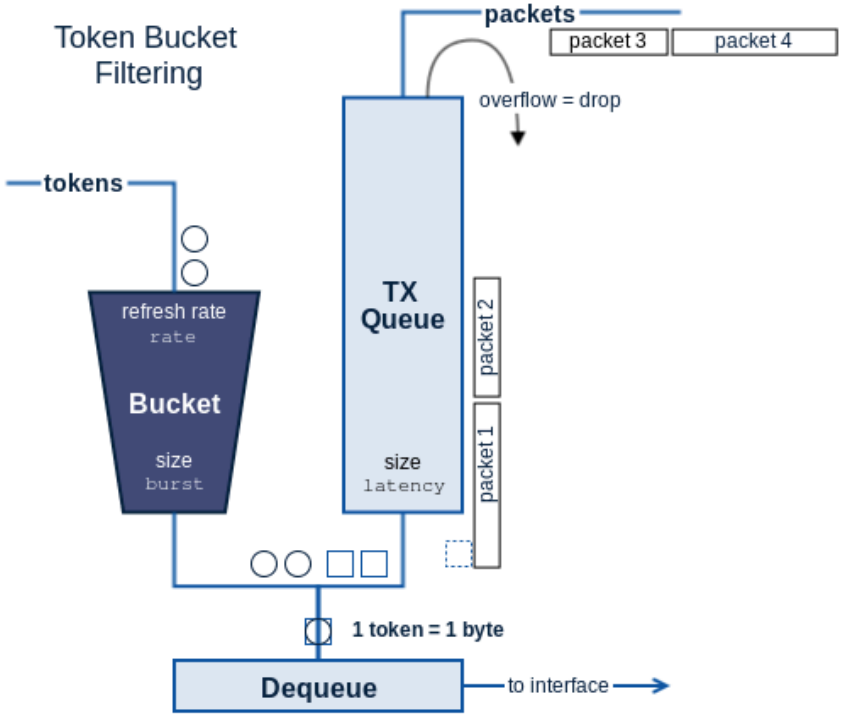


Network emulator (netem)

- Network delay
  - `tc qdisc add dev eth0 root netem delay 100ms 10ms`
- Packet loss
  - `tc qdisc add dev eth0 root netem loss 0.3% 25%`
- Packet duplication
  - `tc qdisc add dev eth0 root netem duplicate 1%`
- Packet reordering
  - `tc qdisc add dev eth0 root netem delay 10ms reorder 25% 50%`

Network emulator (netem)

- Rate control with Token Bucket Filter (TBF)
  - `tc qdisc add dev eth0 root tbf rate 5mbit burst 10kb latency 10ms`



The *Intermediate Functional Block* (IFB) pseudo-device

- IFB devices are used for traffic redirection and mirroring in conjunction with tc(8).

- `modprobe ifb`
- `ip link set dev ifb0 up`
- (*or, to create an IFB device:* `ip link add <name> type ifb`)
- `tc qdisc add dev eth0 ingress`
- `tc filter add dev eth0 parent ffff: protocol ip u32 match u32 0 0 flowid 1:1 action mirred egress redirect dev ifb0`
- `tc qdisc add dev ifb0 root netem delay 500ms`

References

- Linux man pages

  `ip(8), ip-link(8), ip-address(8), ip-netns(8), tc(8), tc-tbf(8), tc-netem(8), iptables(8), ebtables(8)`

- iproute2 cheat sheet
  http://baturin.org/docs/iproute2/
- Connect two netns using bridge and veth
  http://fosshelp.blogspot.fr/2014/08/connect-two-network-namespaces-using.html
- Linux Advanced Routing & Traffic Control
  http://lartc.org/howto/