

Internet routing within the network card

Asbjørn Sloth Tønnesen

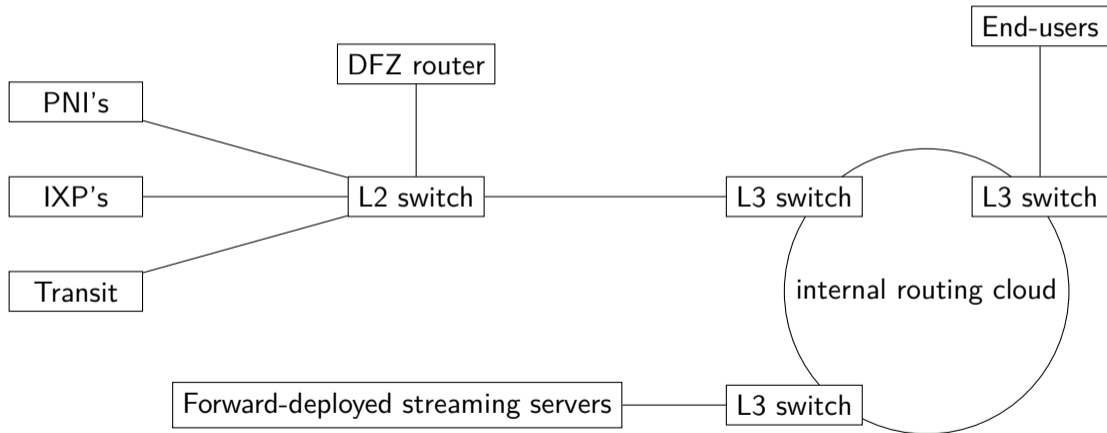
Fiberby ApS

July 22nd, 2024

Who are Fibercity / Fiberby

- We operate AS42541
- Based in Copenhagen, Denmark
- Using FTTB (Fiber to the Building) technology
- 1/1 Gbps connection for 17€/month
- 35k residential households connected
- Peak utilization: 140 Gbps / 15 Mpps (in+out)
- Transit sponsor for BornHack

Simple ISP network model



AS42541 routing timeline

2010

- Cisco 6500 (10G)



Credit: Cisco Systems

2015

- Brocade MLXe-4 (40G)



Credit: Extreme Networks

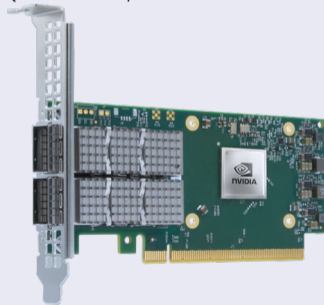
AS42541 routing timeline (cont.)

2020

- 2U server running Linux + BIRD v2

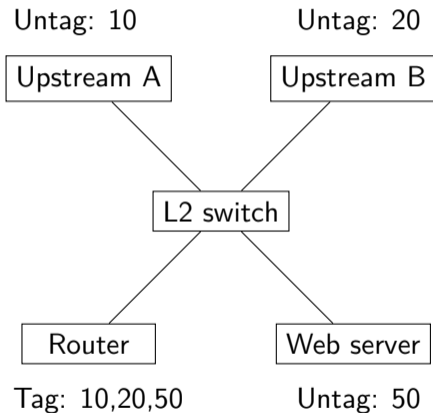
2021

- NIC upgraded to 100G
(Mellanox/Nvidia ConnectX-6 Dx)



Credit: Nvidia Corporation

One armed router / Router on a stick



- Router with only one active physical port
- Typically uses IEEE 802.1Q VLAN tags to reach multiple L2 networks.
- L3 router ports are expensive
- L2 switches are cheaper

Intro

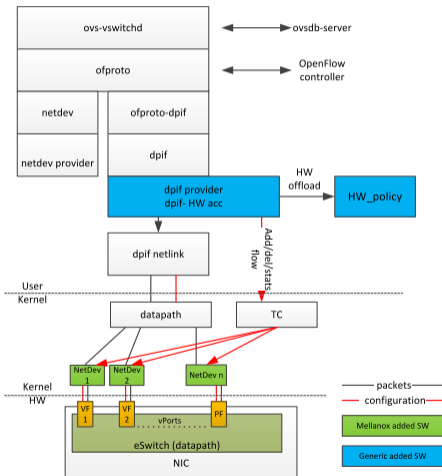
What is a ConnectX-5?

- 100Gb/s or 2x100Gb/s QSFP
- Overlay offloading(VXLAN/GENEVE/...)
- SR-IOV/eSwitch (Switching between VMs on the same HV)
- Newer generations: CX6, CX7: TLS offloading, remote management, ...
- Price on Ebay: ~1000dkk
- Good open source support(Drivers, tooling, ...)
- Some public documentation



OVS Offload Using ASAP² Direct

- Open vSwitch (OVS)
- Embedded switch (eSwitch)
- SRIOV
- flow offload
- TC flower API

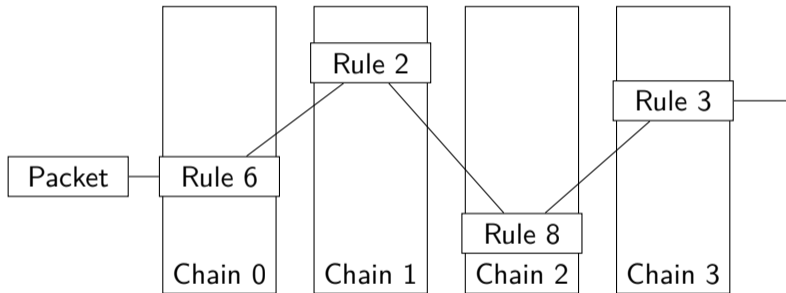


Credit: Nvidia Corporation

TC & flower primer

- Linux kernel subsystem for Traffic Control (TC)
- Chains, priorities and handles
- Actions: drop, trap, goto, redirect
- Hardware offload:
 - Opportunistic / `skip_sw` / `skip_hw`
 - Vendor agnostic-ish
 - Some drivers only support chain 0

TC chains, prios and handles



Packet fate:
drop
trap
redirect

Rule action can goto another chain

Routing with TC flower

The anatomy of routing a packet

- Change VLAN tag
- Change source and destination MAC address
- Decrement TTL / hoplimit
- Update checksum (only if IPv4)
- Push it back out again

TC flower by example

```
1 tc filter add dev "$dev" ingress chain 4 pref 1 \  
2   protocol 802.1Q flower skip_sw \  
3   vlan_ethertype ipv4 \  
4   action vlan modify id "$vlan_id" \  
5   pipe action pedit ex \  
6   munge eth dst set "$new_dst_mac" \  
7   munge eth src set "$new_src_mac" \  
8   munge ip ttl dec \  
9   pipe csum ip4h \  
10  pipe action mirrored egress redirect dev "$dev"
```

TC flower by example

```
tc filter show dev enp2s0f0np0 ingress chain 4
```

```
1 filter protocol 802.1Q pref 1 flower
2 filter protocol 802.1Q pref 1 flower handle 0x1
3   vlan_ethertype ip
4   eth_type ipv4
5   skip_sw
6   in_hw in_hw_count 1
7     action order 1: vlan modify id 620 protocol 802.1Q pipe
8     action order 2: pedit action pipe keys 5
9     action order 3: mirred (Egress Redirect to device enp2s0f0np0) \
10       stolen
11     index 1 ref 1 bind 1
12     used_hw_stats delayed
```

TC flower by example (cont.)

```
tc filter show dev enp2s0f0np0 ingress chain 4
```

```
1     action order 1: vlan modify id 620 protocol 802.1Q pipe
2         index 1 ref 1 bind 1
3         used_hw_stats delayed
4
5     action order 2:  pedit action pipe keys 5
6         index 1 ref 1 bind 1
7         key #0   at eth+0: val xxxxxxxx mask 00000000
8         key #1   at eth+4: val xxxx0000 mask 0000ffff
9         key #2   at eth+4: val 0000xxxx mask ffff0000
10        key #3   at eth+8: val xxxxxxxx mask 00000000
11        key #4   at ipv4+8: add ff000000 mask 00ffffff
12        used_hw_stats delayed
13
14        action order 3: mirred (Egress Redirect to device enp2s0f0np0) \
15            stolen
```

TC flower by example (cont.)

```
tc -s filter show dev enp2s0f0np0 ingress chain 4
```

```
1      action order 3: mirrored (Egress Redirect to device enp2s0f0np0) \
2          stolen
3          index 1 ref 1 bind 1 installed 1241229 sec used 0 sec
4          Action statistics:
5          Sent 2828989939137069 bytes 3904698201 pkt \
6              (dropped 0, overlimits 0 requeues 0)
7          Sent software 0 bytes 0 pkt
8          Sent hardware 2828989939137069 bytes 3904698201 pkt
9          backlog 0b 0p requeues 0
10         used_hw_stats delayed
```

Static hardware offload

Inbound traffic is simple

- Always known to be online next hop
- Prefixes don't change
- Next-hop is a L3 switch.

Chaining it together

Chain 0

- Send IPv4 to chain 1
- Send IPv6 to chain 2

Chain 1/2

- Rule 1: If TTL is expiring trap packet
- Rule 10-: Match some linknets, and trap em
- Rule 100-: Match inbound destinations, and goto chain 4/6

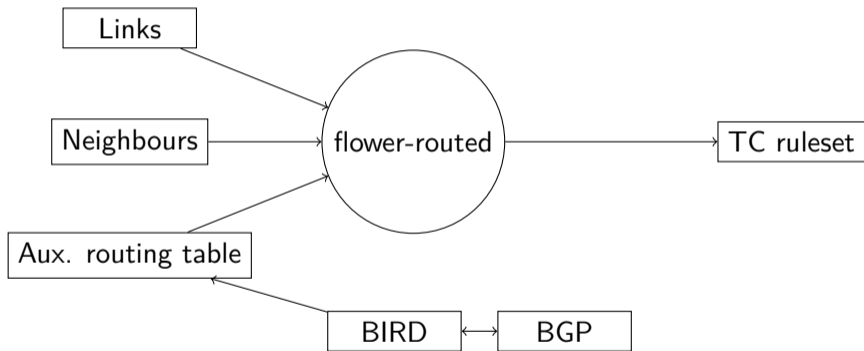
Chain 4/6

- Forward packet

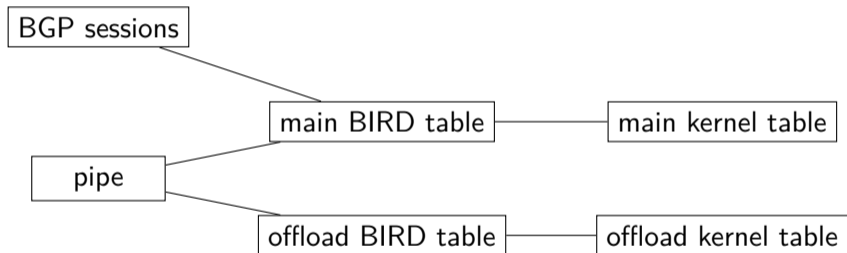
flower-route

- Small daemon to synchronize routes to hardware
- Runs an event loop, with two Netlink sockets
- Extracts links, neighbours, routes and TC ruleset from the kernel
- Learns routes from an aux. kernel routing table.
- Maintains TC ruleset in kernel
- Defensively coded, tries to only make minimal changes to TC
- Currently unit tests have 70+% code coverage
- Licensed as GPLv2+

Block diagram

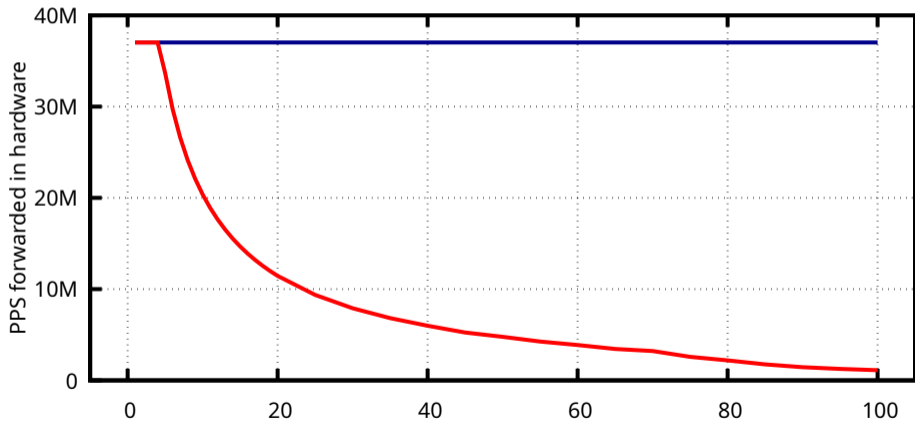


BIRD config



Offload performance tests

The scalability of skip_sw rules

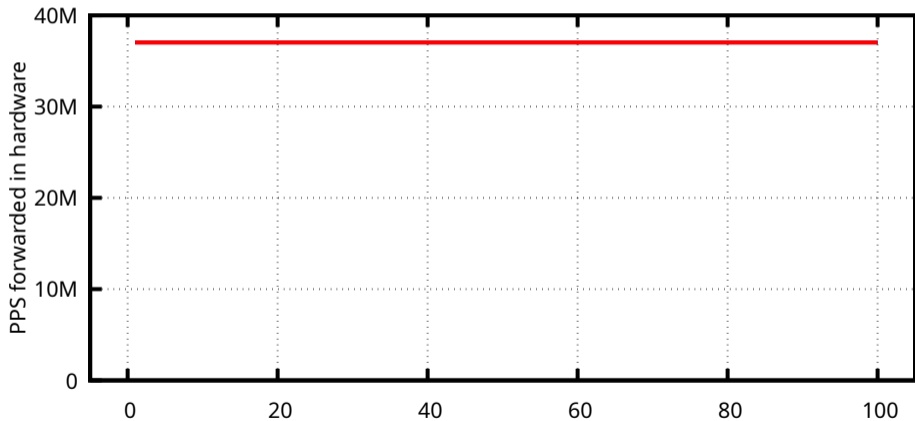


<n> non-matching rules (based on priority), followed by one matching rule

pktgen — forwarded

Offload performance tests

The scalability of skip_sw rules

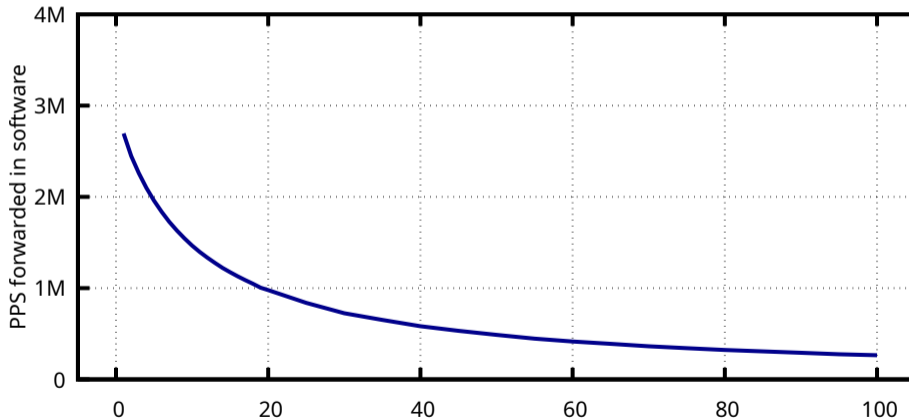


<n> non-matching rules (based on handles), followed by one matching rule

pktgen — forwarded —

Offload performance tests

The effect of skip_sw rules on software processing

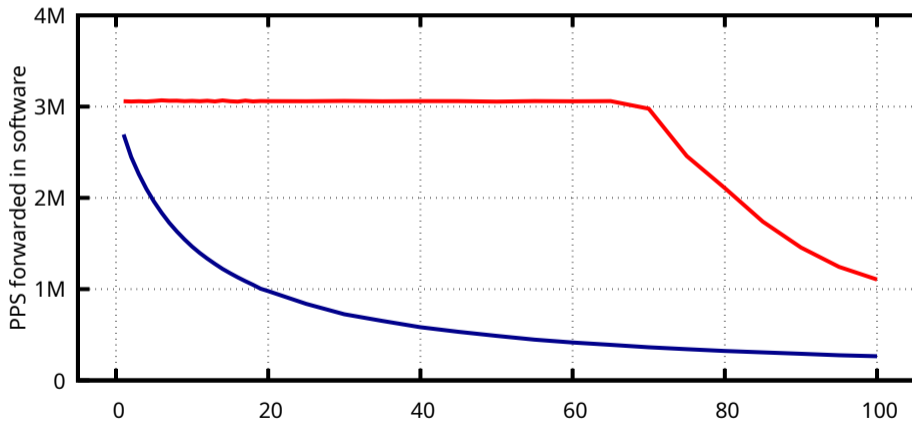


<n> non-matching rules, forcing the software path to take over

Linux v6.9 —

Offload performance tests

The effect of skip_sw rules on software processing



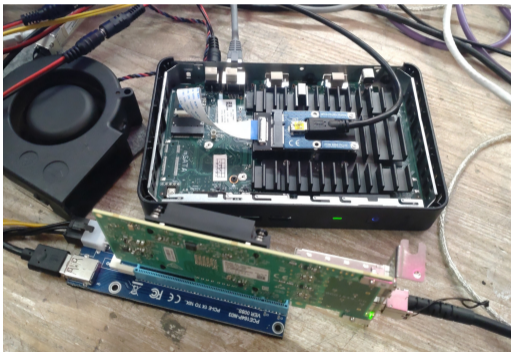
<n> non-matching rules, forcing the software path to take over

Linux v6.9 — Linux v6.10 —

Future work on `flower-route`

- Better TC ruleset debug and simulation tooling
- More flexible configuration:
 - Reverse path filtering (BCP38)
 - Offload to directly connected hosts
- Use TC handles (for better scalability)
- ECMP support
- Handle MTU differences
- Verify that bonding works (eg. 2x100G)

Alternative applications - Energy efficient routing



25G router @ ~15W

- Dell Wyse 3030LT (~3W, peak: 5W)
- M.2 (E-key) PCIe v2 x1
- -> mini-PCIe
- -> PCIe riser
- -> Mellanox ConnectX-5 2x25G (~7W w/ DAC)
- and a fan

Q&A

flower-route

- Patches welcome <https://github.com/fiberby-dk/flower-route>
- Next update:
 - Blog post with deployment details from Labitat (AS205235, Copenhagen hackerspace)