OVN Hardware offload

An effort review, performance and next steps

Ariel Levkovich
lariel@nvidia.com

Marcelo Ricardo Leitner
mleitner@redhat.com

Why is it needed?

25Gbs and 100Gbs are the standard today, yet 200Gbs and 400Gbs are coming soon. As a rule of thumb, an ovs-dpdk thread can push 16~18Mpps (64b), which is ~12.7Gbps.

Cloud native and modern data center applications such as AI, will continue to push for higher speed networking.

Such network speeds make the use of SDN very challenging, mostly due to the heavy CPU investment needed for SDN enforcement.

While OvS was already offloadable, including with Conntrack, having OVN added made some gaps evident.

OVN Datapaths are complexed

VF -> Uplink

recirc_id(0),in_port(pf0vf1),ct_state(-new-est-trk),ct_label(0/0x2),eth(src=0a:58:c0:a8:0b:3f,dst=0a:58:c0:a8:0b:01),eth_type(0x0800),ipv4(src=192.168.11.63,dst=192.168.2.35,proto=6,frag=no),tcp(dst=5201), packets:170575, bytes:6297030373, used:0.000s, flags:SP., actions:ct(zone=13,nat),recirc(0x26)

recirc_id(0x26),in_port(pf0vf1),ct_state(-new+est-rel-rpl-inv+trk),ct_label(0/0x3),eth(src=0a:58:c0:a8:0b:3f,dst=0a:58:c0:a8:0b:01),eth_type(0x0800),ipv4(src=192.168.11.32/255.255.255.224,dst=192.168.2.35,tos=0/0x3,ttl=64,frag=no), packets:170569, bytes:6297029908, used:0.000s, flags:P., actions:ct_clear,ct_clear,set(tunnel(tun_id=0xe,dst=10.0.14.18,ttl=64,tp_dst=6081,geneve({class=0x102,type=0x80,len=4,0x10003}),flags(df|csum|key))),set(eth(src=0a:58:c0:a8:02:01,dst=0a:58:c0:a8:02:23)),set(ipv4(ttl=63)),genev_sys_6081

recirc_id(0),in_port(vtep0),eth(src=36:84:00:e4:b7:98,dst=3e:8b:dc:1d:2f:30),eth_type(0x0800),ipv4(frag=no), packets:4528635, bytes:6847263479, used:0.000s, actions:push_vlan(vid=20,pcp=0),p0

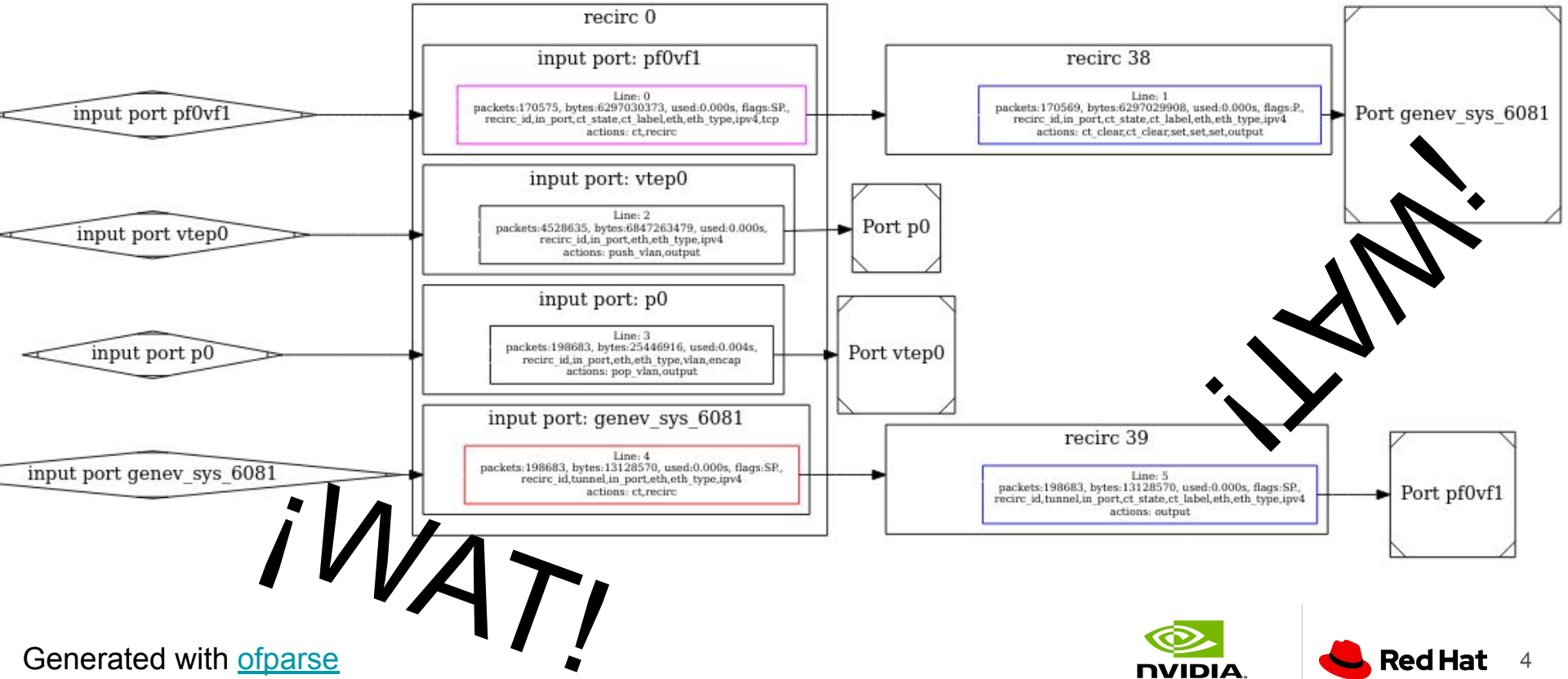
Uplink -> VF

recirc_id(0),in_port(p0),eth(src=3e:8b:dc:1d:2f:30,dst=36:84:00:e4:b7:98),eth_type(0x8100),vlan(vid=20,pcp=0),encap(eth_type(0x0800),ipv4(proto=17,frag=no),udp(dst=6081)), packets:198683, bytes:25446916, used:0.004s, actions:pop_vlan,vtep0

recirc_id(0),tunnel(tun_id=0x1d,src=10.0.14.18,dst=10.0.14.16,geneve({class=0x102,type=0x80,len=4,0x10003/0x7fffffff}),flags(-df+csum+key)),in_port(genev_sys_6081),eth(src=0a:58:c0:a8:0b:01),eth_type(0x0800),ipv4(proto=6,frag=no), packets:198683, bytes:13128570, used:0.000s, flags:SP., actions:ct(zone=13,nat),recirc(0x27)

recirc_id(0x27),tunnel(tun_id=0x1d,src=10.0.14.18,dst=10.0.14.16,geneve({}{}),flags(-df+csum+key)),in_port(genev_sys_6081),ct_state(-new+est-rel+rpl-inv+trk),ct_label(0/0x1),eth(dst=0a:58:c0:a8:0b:3f),eth_type(0x0800),ipv4(dst=192.168.11.63,frag=no), packets:198683, bytes:13128570, used:0.000s, flags:SP., actions:pf0vf1

OVN Datapaths are complexed



Generated with [ofparse]

When lower and upper layers work together...

… better results show up!

This was the first time we did changes to upper layers motivated by lower layers.

Nevertheless, SW datapath is now better too.

Lots, lots of pieces to put together!

- Conntrack Hardware Offload
  - Multi year effort from various parties
  - Based on netfilter flowtables
    - Added a second level of OFFLOADing
  - Required changes in:
    - driver
    - tc flower
    - flow dissector
    - the tc action ct
    - OvS
  - Discussed many times in previous conferences

Lots, lots of pieces to put together!

- On top of that, we added:
  - OVN
  - OVN kubernetes
  - OSP Neutron-ovn
- Which required changes to all those components again!

TC and CT

- Enhanced support for ct_state match (ovs and kernel)
  - Only a few key bits were supported initially: trk, new and est
  - But it was not enough for OVN
  - Added: rel, inv, repl
    - ea71a9d44316 ("netdev-offload-tc: Add support for ct_state flag rel.")
    - edcfd7176f8b ("netdev-offload-tc: Add support for ct_state flags inv and rpl")
    - And their counterparts in kernel
- TC was silently accepting and ignoring unsupported bits
  - Which led to bad matching!
  - A probe of the TC capability to match on the various ct_state flags was added to OVS - If TC can't match on a certain state OVS won't attempt to offload it to TC
    - 1e4aa061ac8b ("netdev-offload-tc: Probe for support for any of the ct_state flags")

TC and CT

- and IP fragments
  - In order to do CT on fragments, they must be reassembled
  - Once done, TC had no way to output the fragments
  - OVS datapath fallback didn’t/can't help here
  - TC mirred had to be improved.
    - c129412f74e9 ("net/sched: sch_frag: add generic packet fragment support.")

TC and CT

- TC CT action and zone 0
  - ovn-kubernetes uses zone 0 in the middle of the pipeline
  - But tc ct action wasn't setting the template in such case, which led to skipping this zone and traffic breakage
  - Upstream fix: fb91702b743d ("net/sched: act_ct: Fix ct template allocation for zone 0")

TC, CT and OVS

- OVN datapath involves flows which forward to/from OVS internal ports so offloading such rules is required to fully offload the datapath.
- New behavior - OVS is adding internal port rules as TC egress rules.
- Full TC support for egress rules had gaps:
  - TC couldn't miss to a specific chain in egress path:
    - Which broke TC CT usage from OVS bridge (OVS internal port)
    - Fixed by 3aa260559455 ("net/sched: store the last executed chain also for clsact egress")
  - Redirecting to egress didn’t reset skb ct state:
    - Fixed by d09c548dbf3b ("net: sched: act_mirred: Reset ct info when mirror/redirect skb")
  - Egress to ingress didn't reset skb dst:
    - Fixed by f799ada6bf23 ("net: sched: act_mirred: drop dst for the direction from egress to ingress")

TC and OVN

- max_reclassify_loop
  - Needed a bump, as 4 was too low for OVN pipeline
    - Upstream commit 05ff8435e505 ("net/sched: cls_api: increase max_reclassify_loop")

OVN pipeline and CT HWOL design imposed restrictions

- By design, only TCP/UDP conntrack entries in EST state get offloaded:
  - Asymmetric routing
    - OVN had asymmetric paths for rx and tx
      - Mismatched zone numbers
    - Resulted in entries never going into Established state (remain in unreplied state), thus never getting offloaded.
    - Too many commits to mention one by one :-)
  - Overlay traffic going via CT (ovn-kubernetes)
    - Tunnel headers are asymmetric (udp dport always the same, for ingress and egress) and conntrack will not see them as established connections - therefore they are not offloadable.
    - Avoid sending overlay traffic (before tunnel decapsulation) to CT by matching on the unique dport.

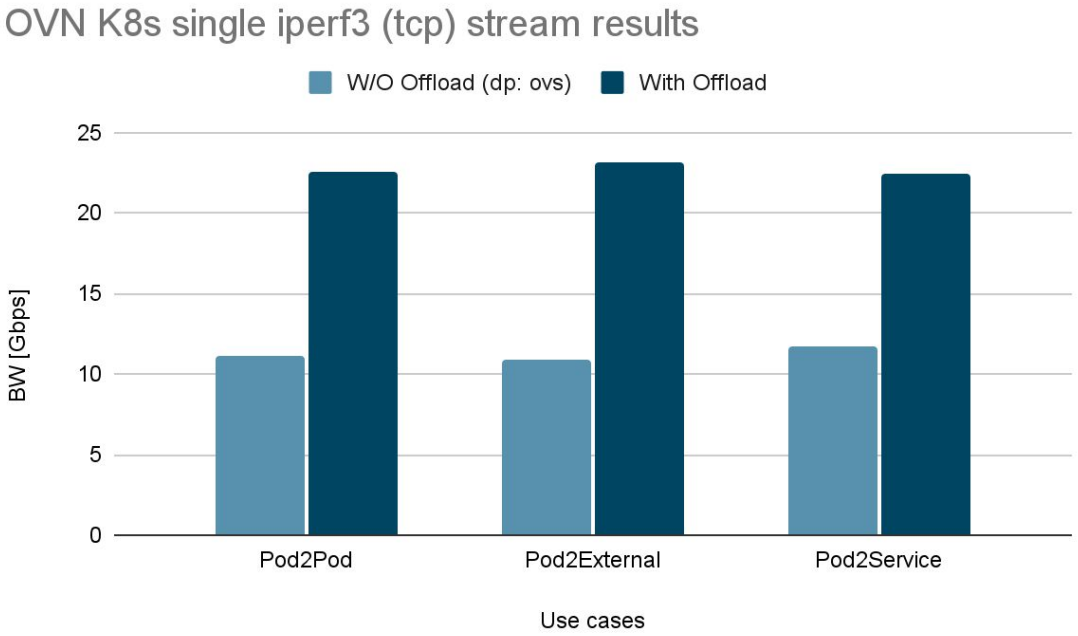
OVN pipeline optimization

- ct/ct(nat) pattern
  - OVN used to first go through conntrack, to then go again to apply the NAT operation.
  - Not needed, and not optimal, for both HW and SW
  - OVN was patched in a couple of places to avoid this pattern

OVS HWOL and NetworkManager

- Two (many) cooks for one pot...
  - They still weren't playing very nice.
  - NetworkManager was patched to avoid mangling the qdiscs if not asked to.

Performance? Boosted!



*Tested on NVIDIA ConnectX-6Dx device
*CPU: AMD Ryzen Threadripper PRO 3955WX

Future challenges

- Check Packet larger action
  - No offload path. New type of behavior not supported by TC today (Action splitting and calling to other actions).
  - Some optimizations were done to avoid this action when not required (E-W traffic) but other cases may hit it and not get offloaded.
- New dec_ttl optimization OVS (pending patches)
  - How to offload to TC?
- Support for matching on the original tuple (ct_tuple4/6)
  - Match avoided today by using a different logic in the flows
- CI

Thanks!
Questions?