Introduction to switchdev SR-IOV offloads

Or Gerlitz, Hadar Hen-Zion, Amir Vadai, Rony Efraim

Mellanox TECHNOLOGIES

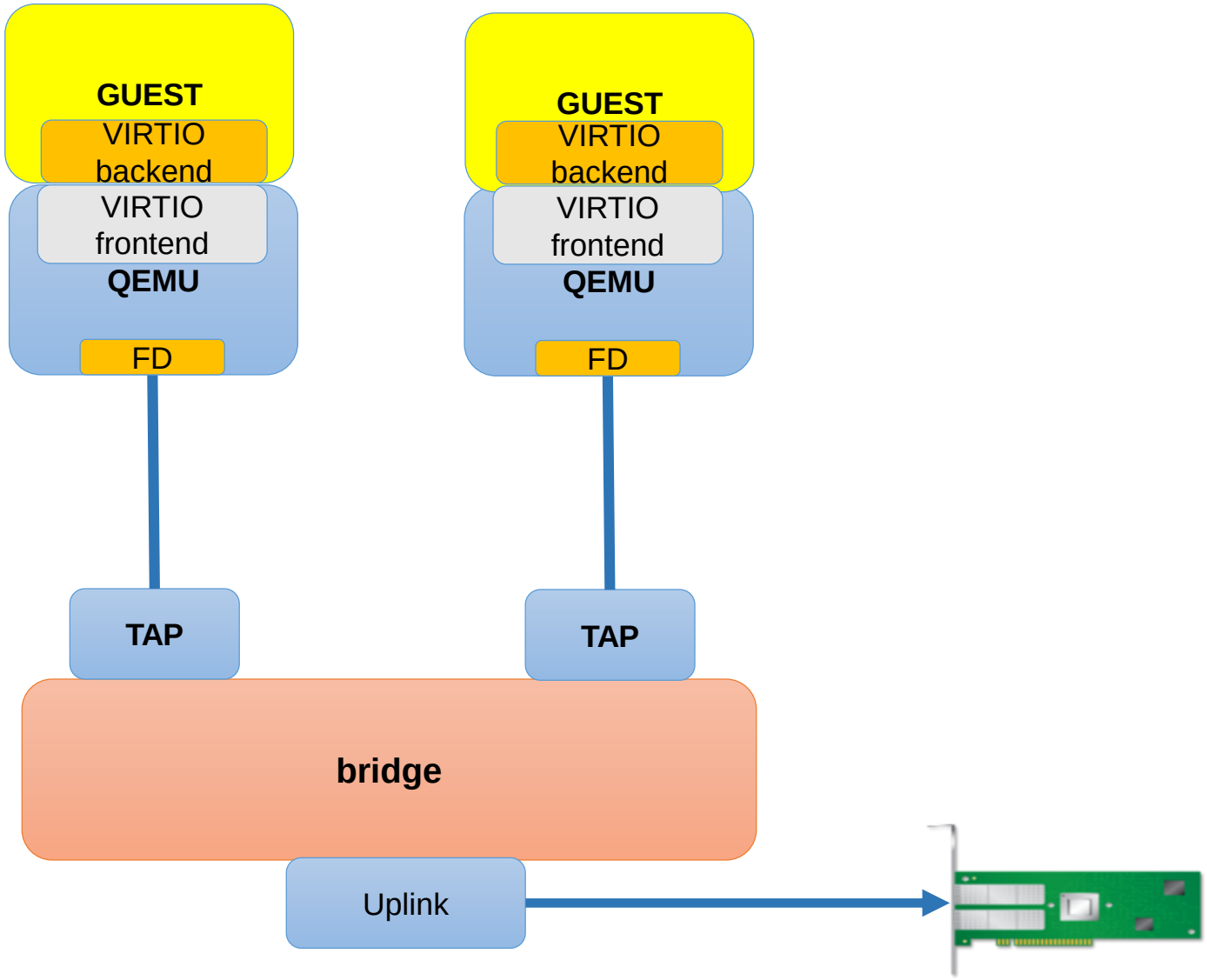
Connect. Accelerate. Outperform.™

Netdev 1.2, Tokyo, October 2016

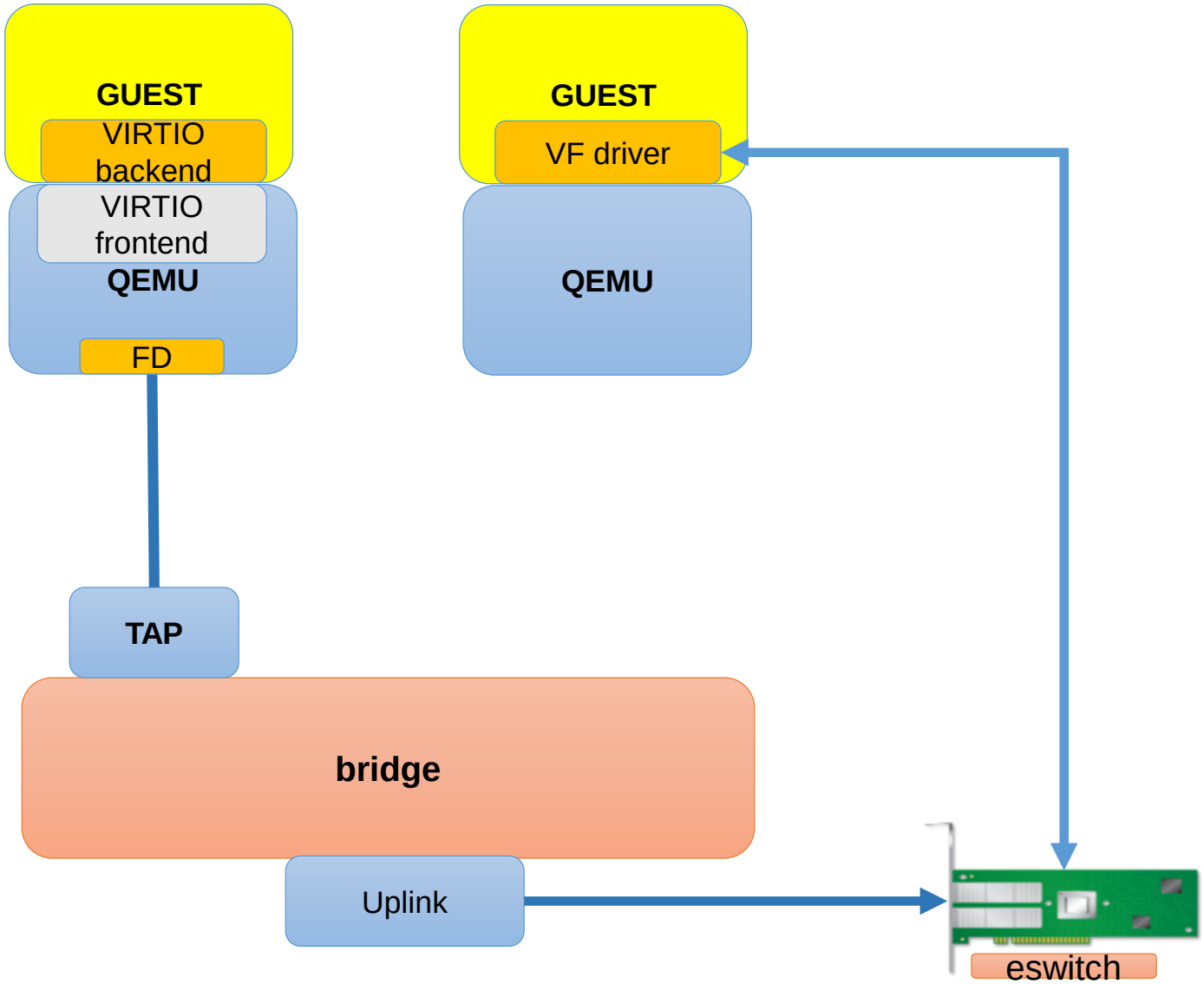
agenda

- background and needs
- SRIOV VF representors
  - concept and requirements
  - implementation
  - e-switch mode control
- setting up slow (non offloaded) SRIOV based switching
- offloading TC based SW datapath to e-switch through VF representors
- offloading IP tunneling to e-switch
- current state of upstreaming
- future challenges in e-switch offloads

background: Para-virtual networking model



background: SRIOV has its own management

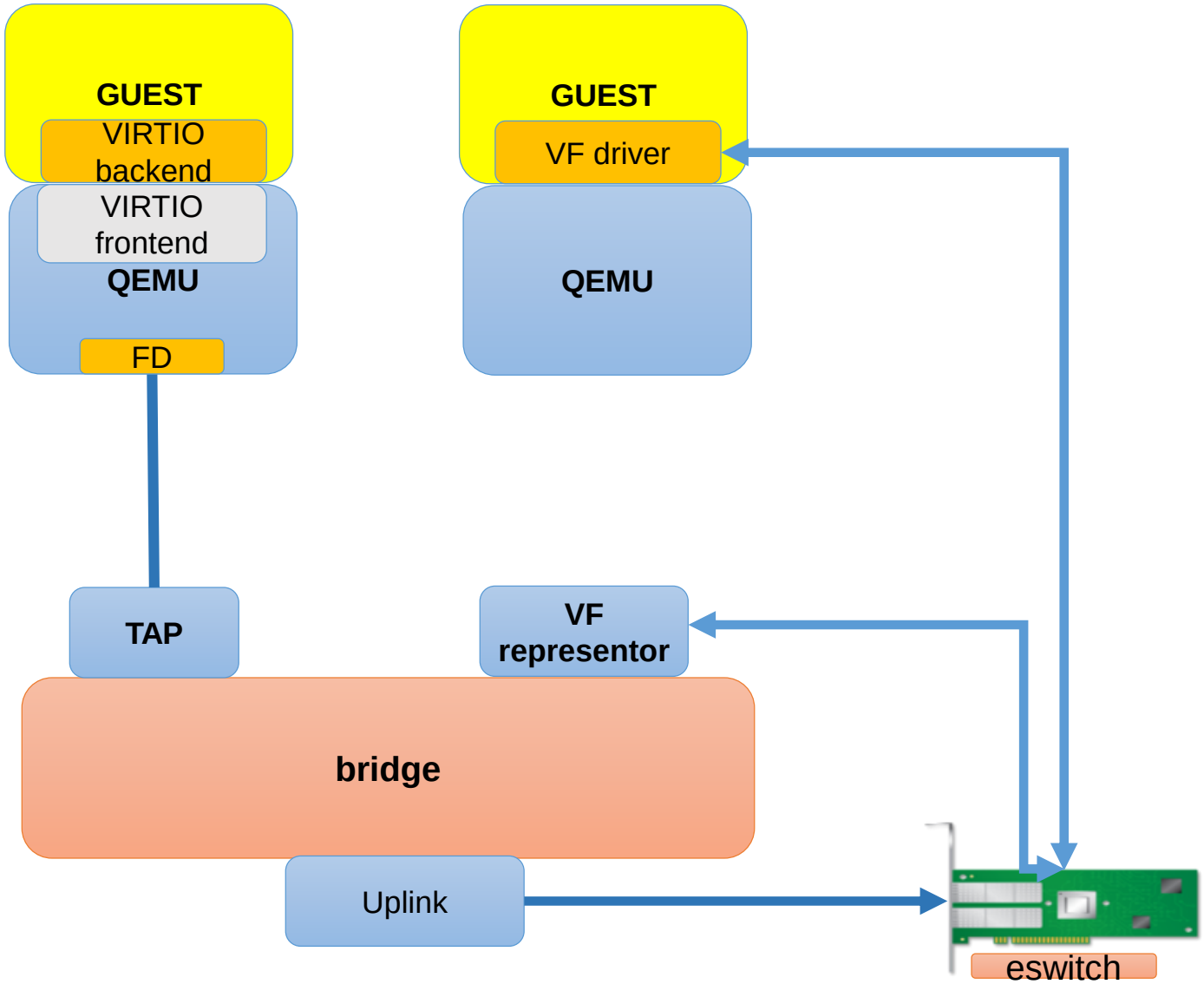


problem description and … a solution

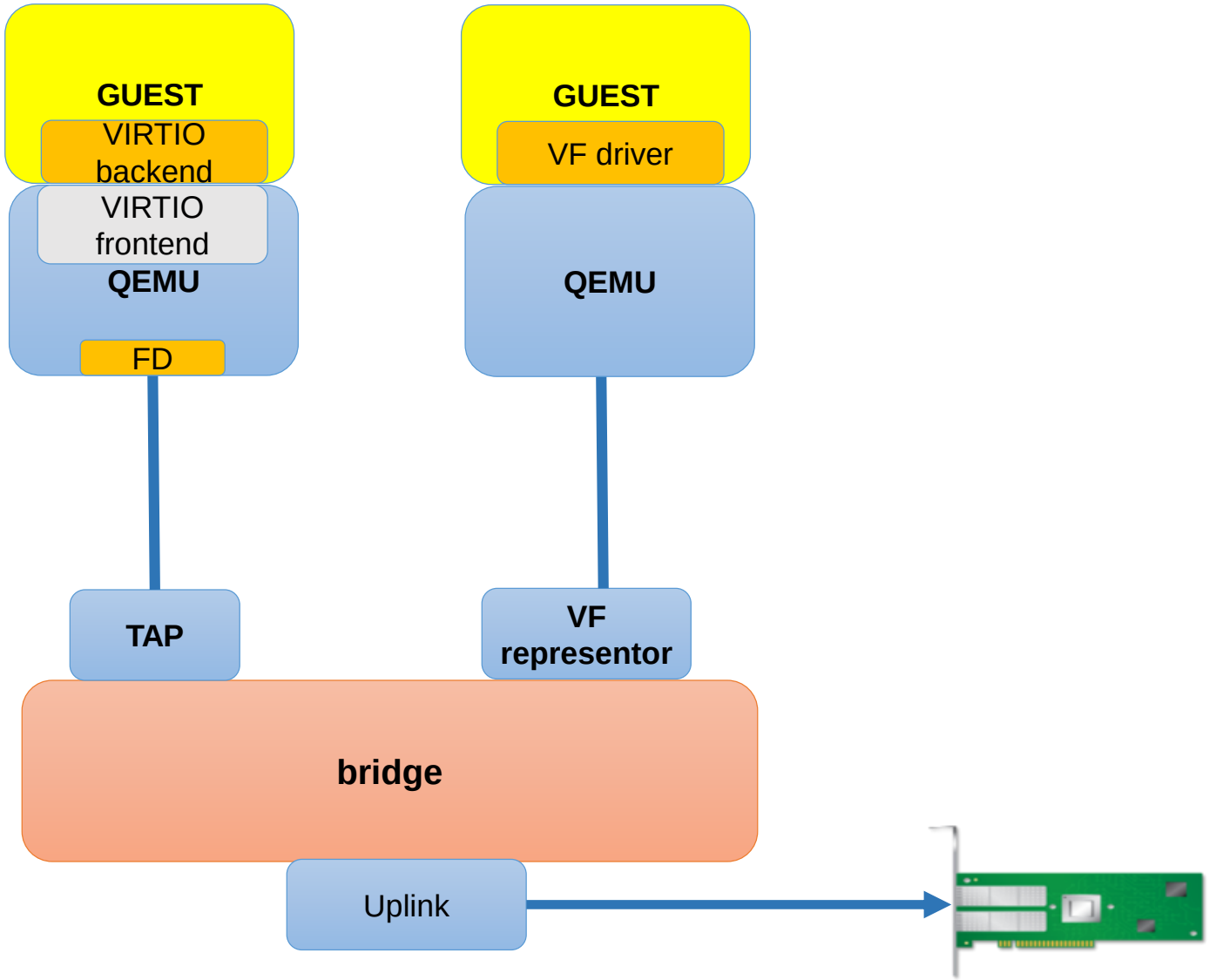
- the way SR-IOV embedded switches were dealt with in Linux was limited in its expressiveness and flexibility
- the kernel model for controlling the SR-IOV e-switch did not allow the configuration of anything beyond MAC/VLAN based forwarding
- we refer to this model as the legacy mode

- and… we've changed that! introduced an upstream SW representation model for the SR-IOV E-Switch port
- the representation plugs nicely into existing kernel SW switching subsystems such as TC, OVS and allows to offload SW traffic rules to HW on SRIOV environment

adding VF representors



VF representors – the kernel SW data-path view



e-switch vport / VF representor basics

- Linux net-device on the host OS that represents the e-switch vport (per VF and the uplink)
- packet sent through the VF representor on the host arrives to the VF (guest)
- packet sent through the VF on the guest is received by its representor into the host OS

→ hooking the representor net-device into a kernel switching data-path (Bridge, TC, OVS) causes VF traffic to be subject to steering (matching and actions) of that software component

- admin operations on the VF rep link state controls the VF link state
- default statistics of the VF rep are VF HW statistics (new NL for SW stats)

VF representor implementation

- Linux net-device on the host OS
- per vendor implementation
- all the VF representors are running over the e-switch management port
- should set the grounds for typical kernel SW data-path requirements
  - send/receive
  - miss rule – allow to plug into SW data-path that uses learning
  - send-to-vports rules – allow to bypass HW data-path when re-injecting packets
- in mlx5, the VF representors implement a functional subset of mlx5 Ethernet net-devices. This design bought us code reuse and sharing
- currently the PF serves as the uplink representor

e-switch mode control

- the VF representors are running over the e-switch management port
- e-switch is assumed to be controlled through vendor PCI driver
- admin uses the `devlink` tool over the PCI device to set the e-switch mode
- the new mode is called (surprise) switchdev
- note: setup is **not** done through the PF net-device
  - not assuming that PF is the e-switch manager allows to handle more use-cases such as multiple PFs, e-switch with multiple uplinks, dedicated non-PF e-switch management (function) ports

```
# devlink dev eswitch set pci/0000:21:00.1 mode switchdev
```

e-switch modes – cont’

- so why we named it switchdev mode?

- because this is essentially the same approach taken on switch HW ASIC
- SW net-device representation of switch ports + offloading
- The VF representors expose switchdev ops and implement the parent ID attribute
- in the future more attributes and objects (FDB, FIB) might be supported by SRIOV VF reps switchdev drivers and plug into the existing kernel switchdev offload infrastructure

setting up slow (non offloaded) SRIOV based switching

- enable SRIOV with N VFs
- assign VFs to VMs
- set SRIOV mode to switchdev → VF reps are created
- attach VF representors to SW switching in the host: Bridge, OVS, TC
- and this is it ☺

- but the performance isn't that of SRIOV
- solution: enhance the networking stack to allow offloading of the SW switching datapath

offload TC SW datapath to e-switch with VF reps

- enable SRIOV, assign VFs to VMs, set e-switch to switchdev mode [..]
- use TC HW offloads to program ingress rules to VF rep → e-switch HW
- typical rule format: <ingress port, matching, action>

```
# ethtool -K enp4s0f1_0 hw-tc-offload on
# tc qdisc add dev enp4s0f1_0 ingress
```

```
# tc filter add dev enp4s0f1_0 protocol ip parent ffff:
flower skip_sw src_mac e4:11:22:33:44:50 dst_mac
e4:1d:2d:a5:f3:9d action mirred egress redirect dev
enp4s0f1
```

offload TC SW datapath e-switch - cont’

- matching on L2/L3/L4 headers
- actions: push/pop vlan, forward (mirred redirect), drop
- offloaded flow statistics (packets, bytes, last-use) to allow aging

offloading IP tunneling to e-switch

- new TC matching and actions to support IP tunneling with shared tunnel device (VXLAN, GRE)
- flower matching on outer headers: src/dst IP and tunnel key
- tunnel key set (encap) action, tunnel key release (decap) action

- here offloading is less straight forward and involves route lookup to determine HW net-device, neigh lookups, etc

current state of upstreaming

- 4.6 – 4.7
  - framework for TC classifier/action offloads by NIC drivers
  - U32 classifier with drop action offload (ixgbe)
  - Flower classifier with mark/drop action offload (mlx5)
- 4.8
  - e-switch switchdev mode, control through devlink
  - mlx5 SRIOV VF representors
  - mlx5 HW offload TC/Flower forward/drop actions & statistics by VF representors
- 4.9
  - TC action for IP tunnel key set / release with shared tunneling device (VXLAN, GRE)
  - TC/Flower with vlan push/pop action offload (mlx5)
- plans for 4.10
  - mlx5 HW offload TC/Flower with IP tunnel key actions (vxlan encap/decap)
  - i40e VF representors ?
  - any other goodies? yeah get me more! ☺

next steps / building blocks for switchdev SRIOV networking

- multiple uplinks (hair-pin)
  - offloading SW gateways
- LAG
  - HW LAG for SRIOV guest networking
- offload flow based routing
  - support L3 based SRIOV guest networking
- **enable faster transition from legacy mode**
  - FDB offload to set HW steering similar to what is used there, more ideas?
- more

conclusions

- switchdev mode for NICs provides SR-IOV performance with Para-virt like flexibility using TC flow offloads
- putting that upstream set the grounds for Open-VSwitch offloading in SRIOV environments, including tunneling
- next… lets see the demo or a script of it
- and later… come tomorrow to the OVS talk

Thank You

Mellanox TECHNOLOGIES

Connect. Accelerate. Outperform.™