Virtual switch HW acceleration

Rony Efraim

February 2016

Mellanox TECHNOLOGIES

Connect. Accelerate. Outperform.™

Agenda

- What is v-switch
- modern NIC capabilities
- Typical packet processing pipeline of a switch
- Classification & Actions
- RFC Tc HW offload of Classification, drop & flow id assignment
- HW classifying usage for openVswitch.
- Egress/Ingress QoS - rate limit and BW guaranty per VM/port
- Complex Use Cases: Nested Virtual Switch Offload
- vSwitch acceleration is it a NIC or a switch functionality

IGMP snooping

What is virtual-switch

- V-switch is a software element that connect few netdevs and can switch traffic between them like :
- Linux bridge
  - Switch based on MAC&VLAN
  - Dynamic MAC learning
  - Can run protocols like STP, IGMP snooping…
- openVswitch (OVS)
  - Flow based switch
  - Dynamic MAC learning
  - Open flow (OF)
  - Can run protocols like STP, …
- Mac-vlan
  - Switch based on static MAC&VLAN
  - mode types
    - private
    - vepa
    - bridge
    - passthru

Modern NIC capabilities

- Classification L2/L3/L4 and tunneling
- Action according to classification: Drop, Flow id assignment…
- MultiQ hundreds and more Queues.
- Queues rate limit
- Queues scheduling (DWRR BW guaranty)
- SR-IOV – multi NICs
- RDMA
- Packet pacing

- The NIC HW can function as a switch and thus can accelerate virtual switches

By doing some of the work like classification and Queues

Typical packet processing pipeline of a switch

- Packet parsing & classification/lookup,
- Push/pop vlan
- Encap/decap operations,
- QoS related functionality
  - Metering
  - Shaping
  - Marking
  - Scheduling
- Switching operation.

- Accelerate the Virtual Switch dataplane by decomposing the packet processing pipeline and offloading the various pipeline stages onto the NIC HW.

Classification & Actions

- Classification based on
  - L2 : S/D-MAC ,Ethertype, VLAN’s
  - L3 : IPv4/IPv6 s/d IP Protocol / Next header
  - L4 : S/D Port flags
  - Tunneling : vxlan VNI …
  - Inner packet L2/L3/L4
- Action
  - drop
  - Allow
  - flow id assignment
  - count
  - forward to ring
  - push/pop vlan,
  - encap/decap tunnel

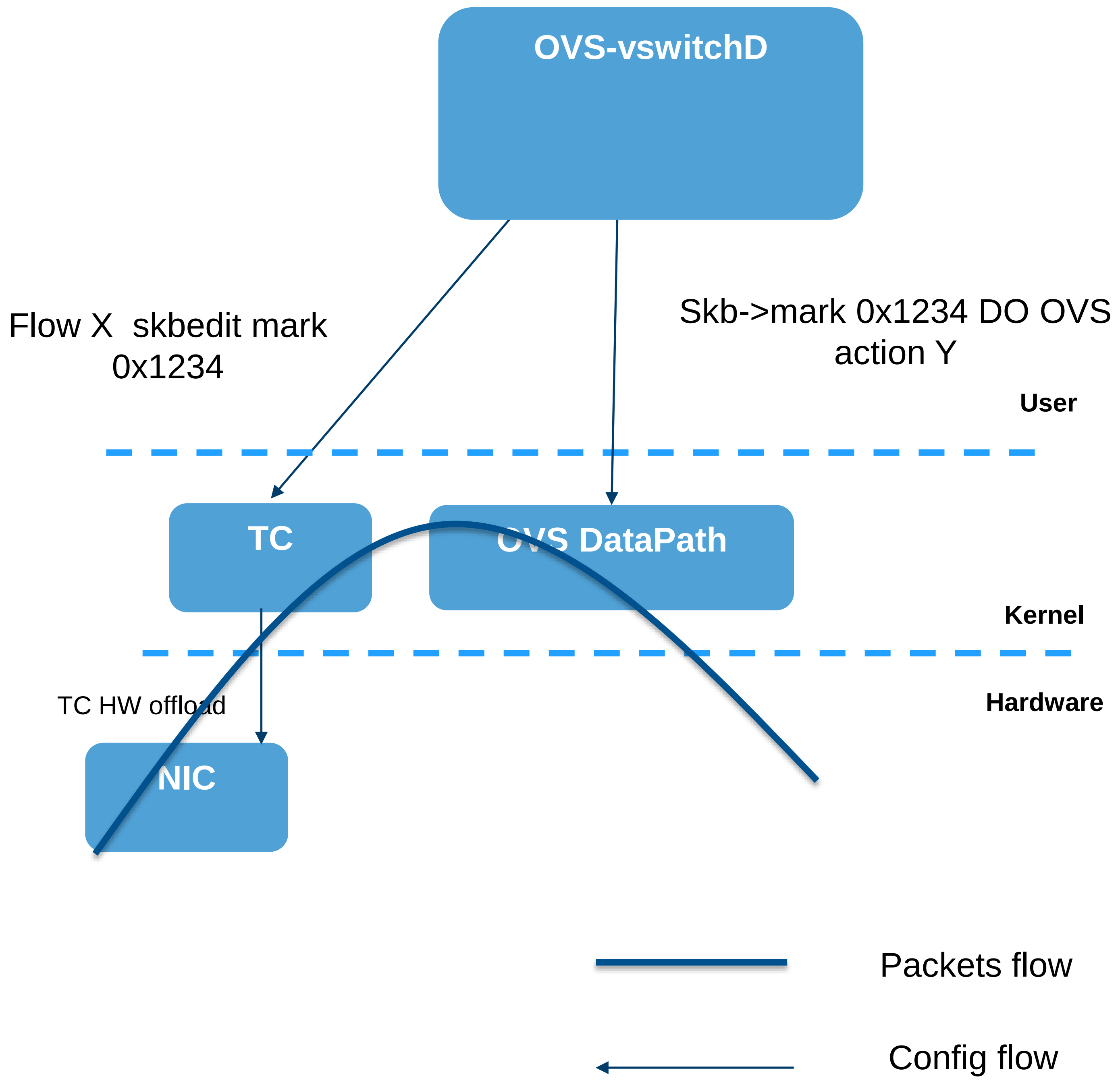
Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/net

RFC -**TC filter HW offloads by Amir Vadai**

```
# add an ingress qdisc
$TC qdisc add dev $ETH ingress

# Drop ICMP (ip_proto 1) packets
$TC filter add dev $ETH protocol ip prio 20 parent ffff: \
            flower eth_type ip ip_proto 1 \
            indev $ETH offload \
            action drop

# Mark (with 0x1234) TCP (ip_proto 6) packets
$TC filter add dev $ETH protocol ip prio 30 parent ffff: \
            flower eth_type ip ip_proto 6 \
            indev $ETH offload \
            action skbedit mark 0x1234

# A NOP filter for packets that are marked (0x1234)
$TC filter add dev $ETH protocol ip prio 10 parent ffff: \
            handle 0x1234 fw action pass
```

openVswitch using HW classifying

- OVS should config TC and OVS datapath
- for Example :
- OVS set action Y to flow X
  - Use TC to skbedit with 0x1234 for flow X
  - Config datapath do to action Y for skb->mark = 0x1234
- OVS action drop for flow Z
  - Use TC to drop flow Z
  - Use TC to get counter

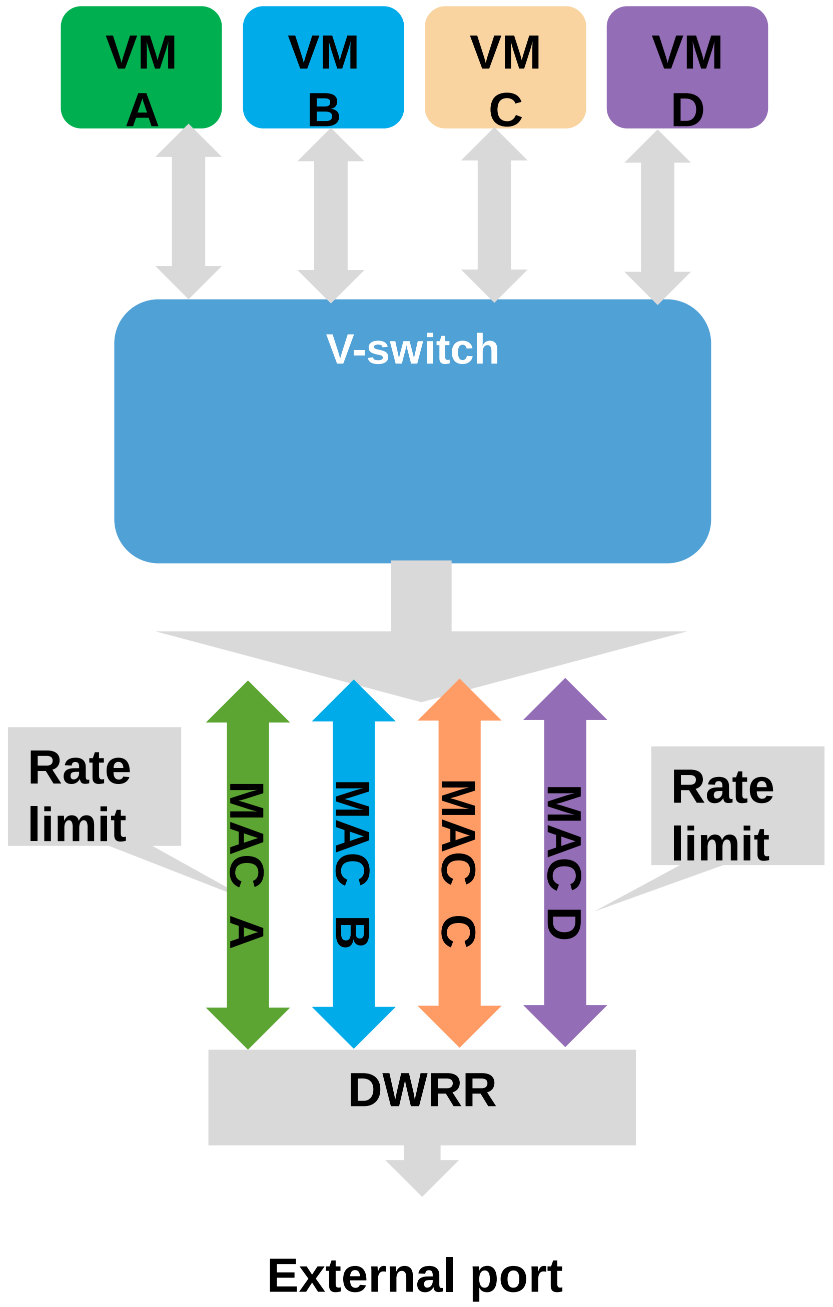


Egress/ingress QoS - rate limit and BW guaranty per VM/port

- Egress QoS per VM/inport port to the external port.
  - Weighted fairness between VM/ports on the external port.
  - Classify to Q
    - Very trivial for MACVTAP
    - bridge/OVS use TC Classify per in-port/src MAC with action set to Q
  - Rate limit the Q and set a weight to each Q

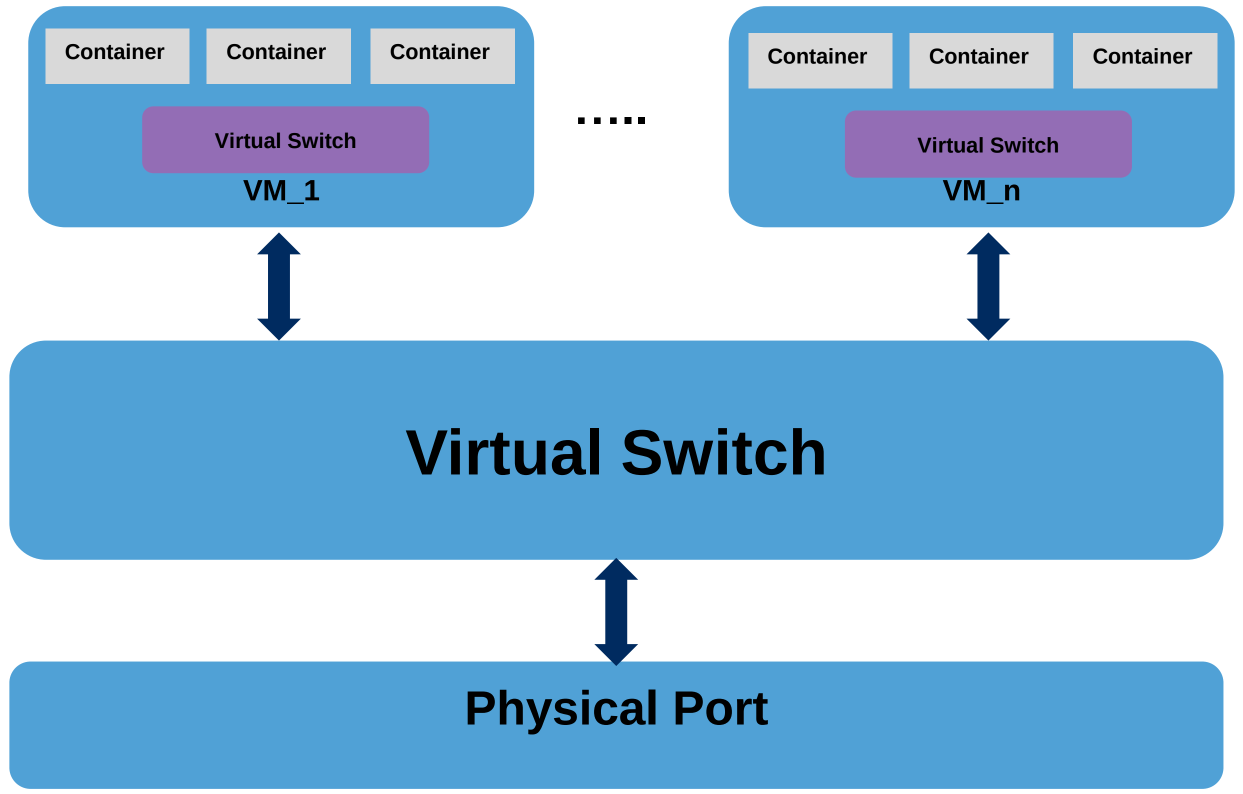
- Ingress QoS use a Q/ring per VM
  - Classify to Q
    - Very trivial for MACVTAP
    - bridge/OVS
      - use TC Classify per in-port/src MAC with action set to Q
      - supported today :ethtool -U eth2 flow-type **ether dst** *xx*:*yy*:*zz*:*aa*:*bb*:*cc* **action** Rx-queue

- logical Q can support RSS/TSS



Complex Use Cases: Nested Virtual Switch Offload

- Multiple VMs, each running multiple containers
- SW virtual Switch topology: Host “root” switch and above it multiple switches – one per VM
- Various :”Wiring” options:
  - Containers connected using SRIOV (VF)
    - Full Virtual Switch Offload.
  - Containers AND VMs connected via PV
    - Accelerated Virtual Switch
  - Hybrid: Container connected via PV, VMs are connected with VF (SRIOV)
    - Full Virtual Switch Offload for the VMs
    - Accelerated Virtual Switch within each VM



vSwitch acceleration: is it a NIC or a switch functionality?

- All the features mentioned provide vSwitch acceleration
- Neither of them requires switchdev.
- Full offload (SRIOV) reqwire the same api
- The same API can be use for switch/switchdev by adding action send to port X.
- We should have a single API for user (TC) for all actions.

Q&A