Look Mom - No Patches in our Blazing Fast and Smart Telco Cloud

Anita Tragler
Product Manager, Networking/NFV, Red Hat

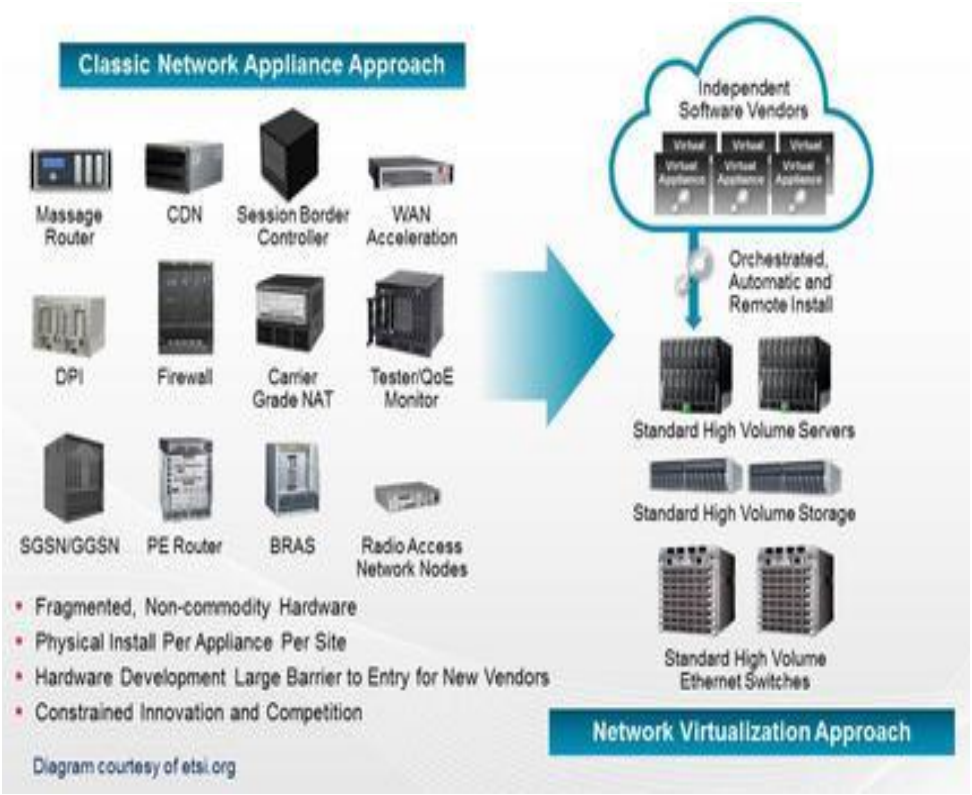
Ash Bhalgat
Senior Director, Cloud Marketing, Mellanox Technologies

Mark Iskra
Partner Group TME, Nuage Networks

nuagenetworks
From Nokia

May 23, 2018

SDN & NFV Vision: Disaggregation and Virtualization



SDN & NFV - Tsunami of Disruption

Bigger than Circuit Switched to IP transformation

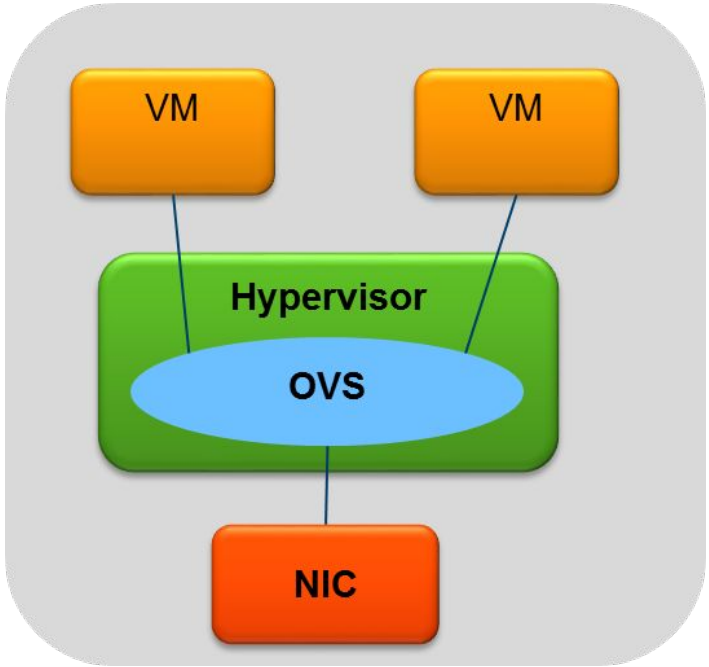
Closed → Open Network Infrastructure

1. Vertically integrated to open source

- High → Low Capex
  1. Custom products to Industry standard virtualized servers
- Vendor locked → Ecosystem
  1. Incumbent dominated to multi-vendor solutions
-
- But.. Achieving bare metal performance is challenging
- Open, Disaggregated and Virtualized Networking degrades packet performance significantly
-

Open Virtual Switch (OVS) – Love and Hate Relationship!

- Virtual switches such as Open vSwitch (OVS) are used as the forwarding plane in the hypervisor
- Virtual switches implement extensive support for SDN (e.g. enforce policies) and are widely used by the industry
- Supports L2-L3 networking features:
  - L2 & L3 Forwarding, NAT, ACL, Firewall, Load Balancer, etc.
  - Flow based
- OVS Challenges:
  - Packet Performance: <1M PPS w/ 2-4 cores
  - Significant CPU: Even w/ 12 cores, can't get 1/3rd 100G NIC
  - User Experience: High and unpredictable latency, packet drops
- **Solution**
  - **Hardware Acceleration without compromising Software Programmability**



Why Is Hardware Acceleration Needed ?

- Software defined (everything) is a key for SDN & NFV transformation
- Today's software solutions deliver functionality, but lack performance & efficiency
- **Hardware Acceleration** required to gain flexibility, performance and cost efficiency

Software

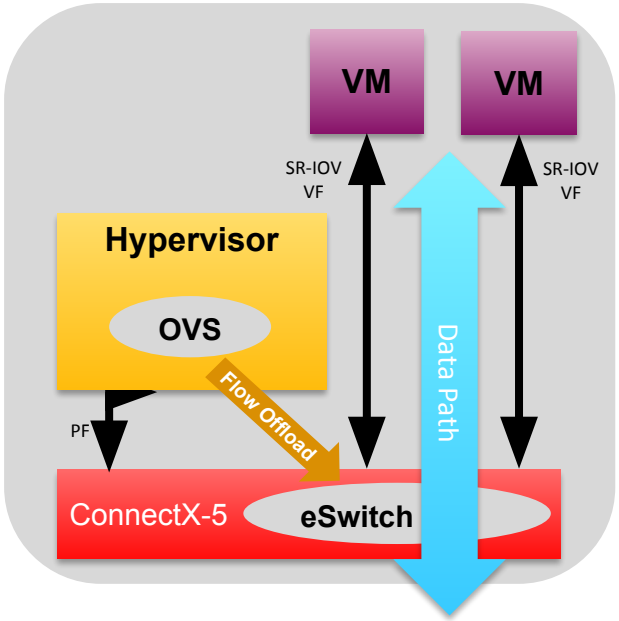
Software + Hardware Acceleration

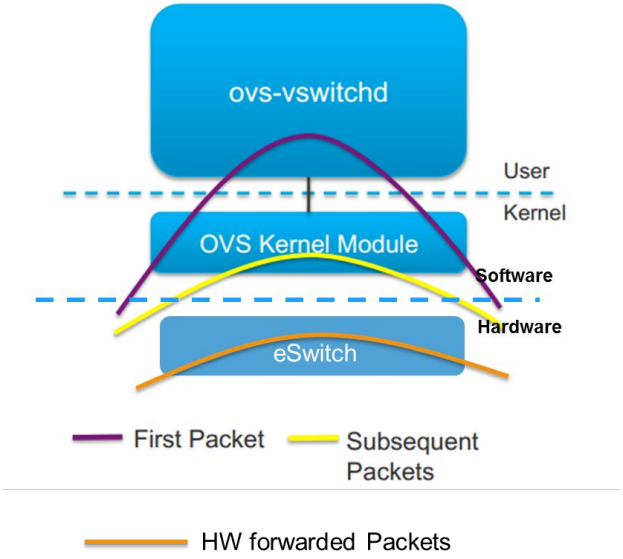
HW Acceleration Reduces Server Footprint by an Order of Magnitude

Full OVS Hardware Offload – NIC Architecture

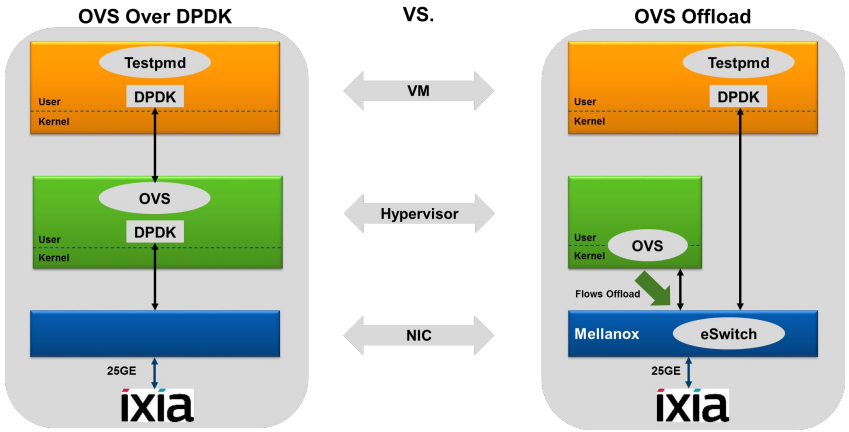
- **Accelerated Switching and Packet Processing (ASAP^2)**
  - Open vSwitch as Standard SDN Control
  - OVS data-plane offload to NIC embedded Switch (eSwitch) – SR-IOV Data Path

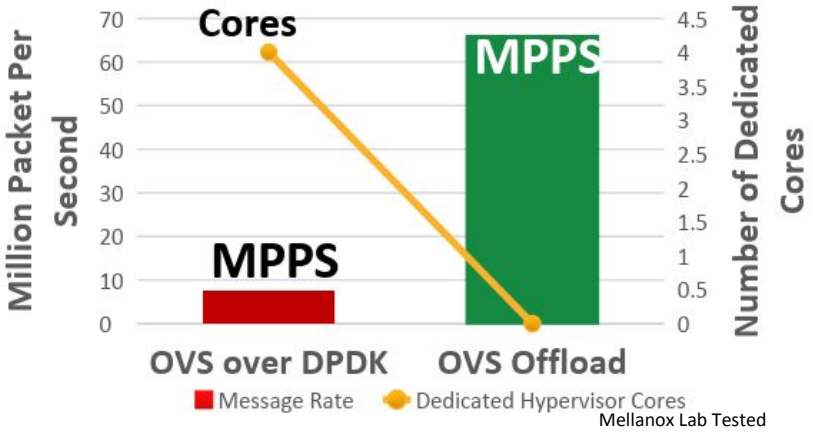
- **Best of Both Worlds:** SDN Programmability and Faster Switching Performance





OVS over DPDK vs. OVS Offload – ConnectX-5





Mellanox Lab Tested

- **Mellanox OVS Offload (ASAP2) Benefits**
  - Highest VXLAN throughput & packet rate
  - 8X-10X better performance than OVS over DPDK
  - Line rate performance at 25/40/50/100Gbps
  - 100% CapEx Savings with Zero CPU Utilization

- **Open Source Enabled – No Vendor Lock-In**
  - Adopted broadly by Linux community & industry
  - Full Community Support (OVS, Linux, Openstack)
  - Ecosystem Support (Nuage/Nokia, Red Hat, etc.)

OpenStack Networking - Datapath Performance with Open vSwitch

Throughput Measured in Packets per second with 64 Byte packet size

No Tuning, default deployment

Up to 50 Kpps

Up to 4 Mpps per socket*

*Lack of NUMA Awareness

OpenStack Blueprint NUMA aware vswitch (Rocky+)

20+ Mpps per core (line rate?)

OVS 2.9, OpenStack Queens

RHOSP 13/RHEL 7.5 (Tech Preview)

Open vSwitch (OVS) Hardware offload (tc-flower)

Kernel 4.8+, OpenStack Pike/Queens, OpenvSwitch 2.8+

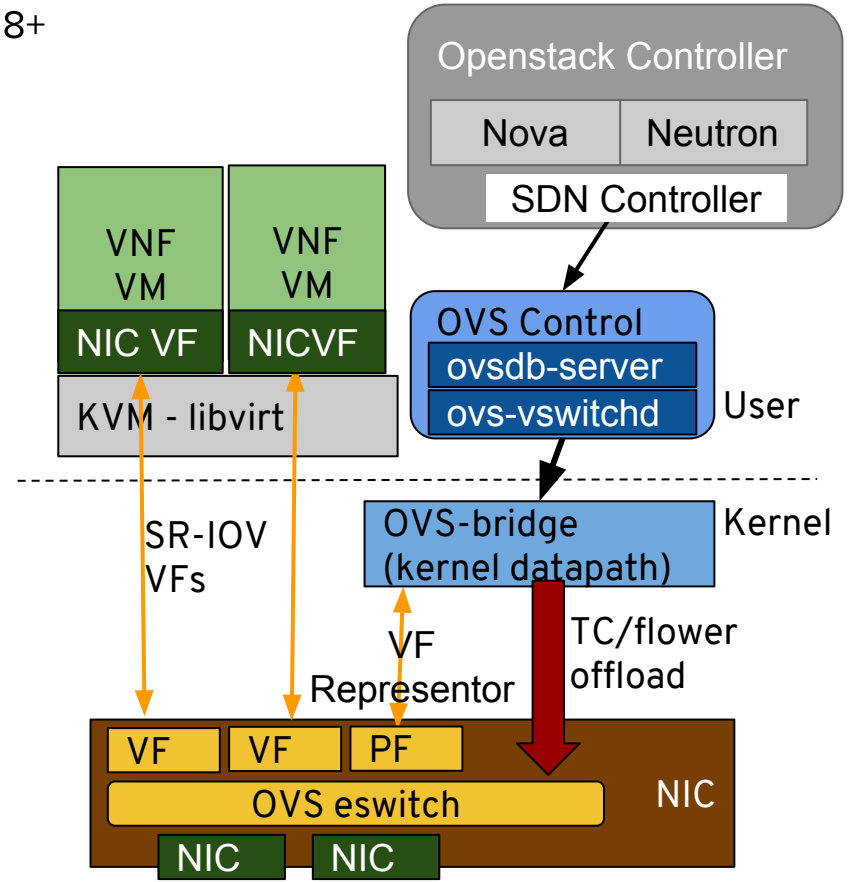
[OpenStack documentation]

Best of both worlds

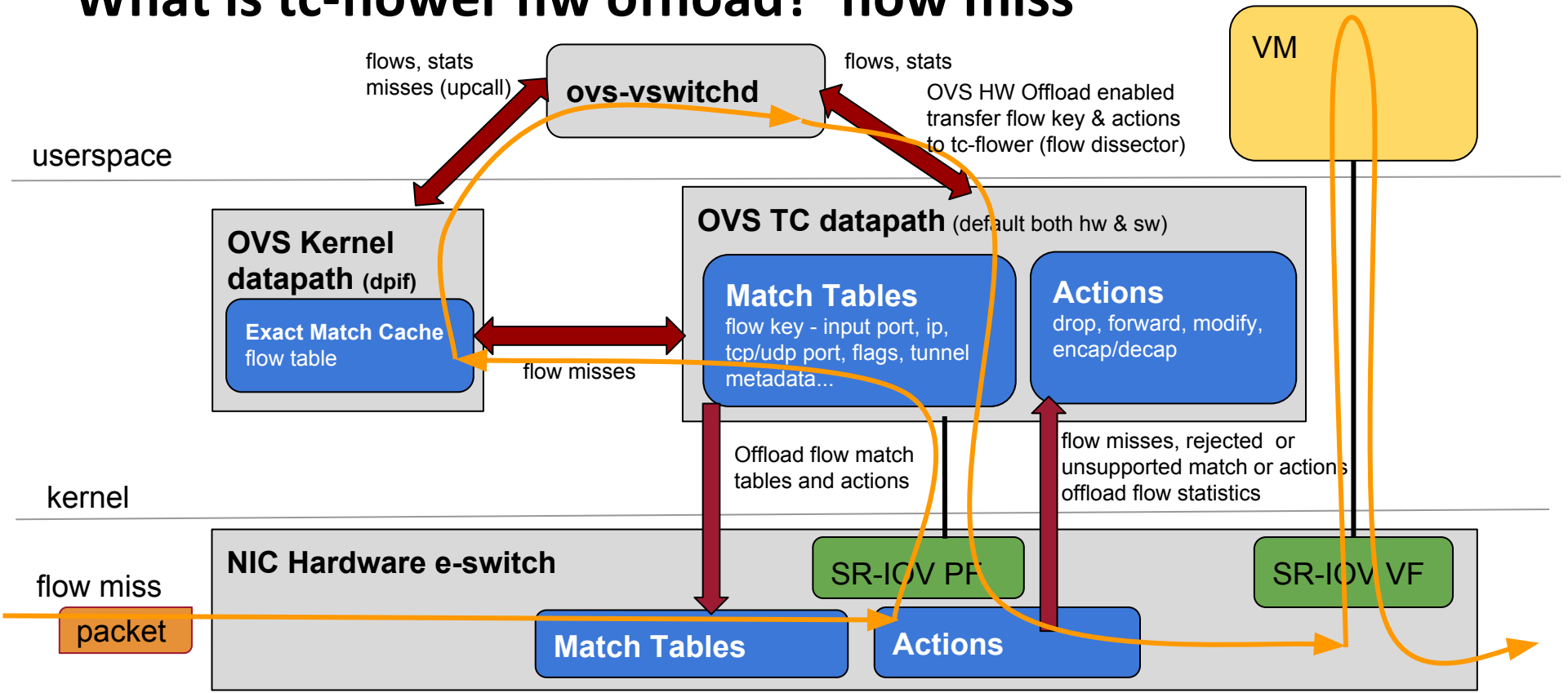
- High performance with SR-IOV
- Fallback to OVS (kernel) networking

Features

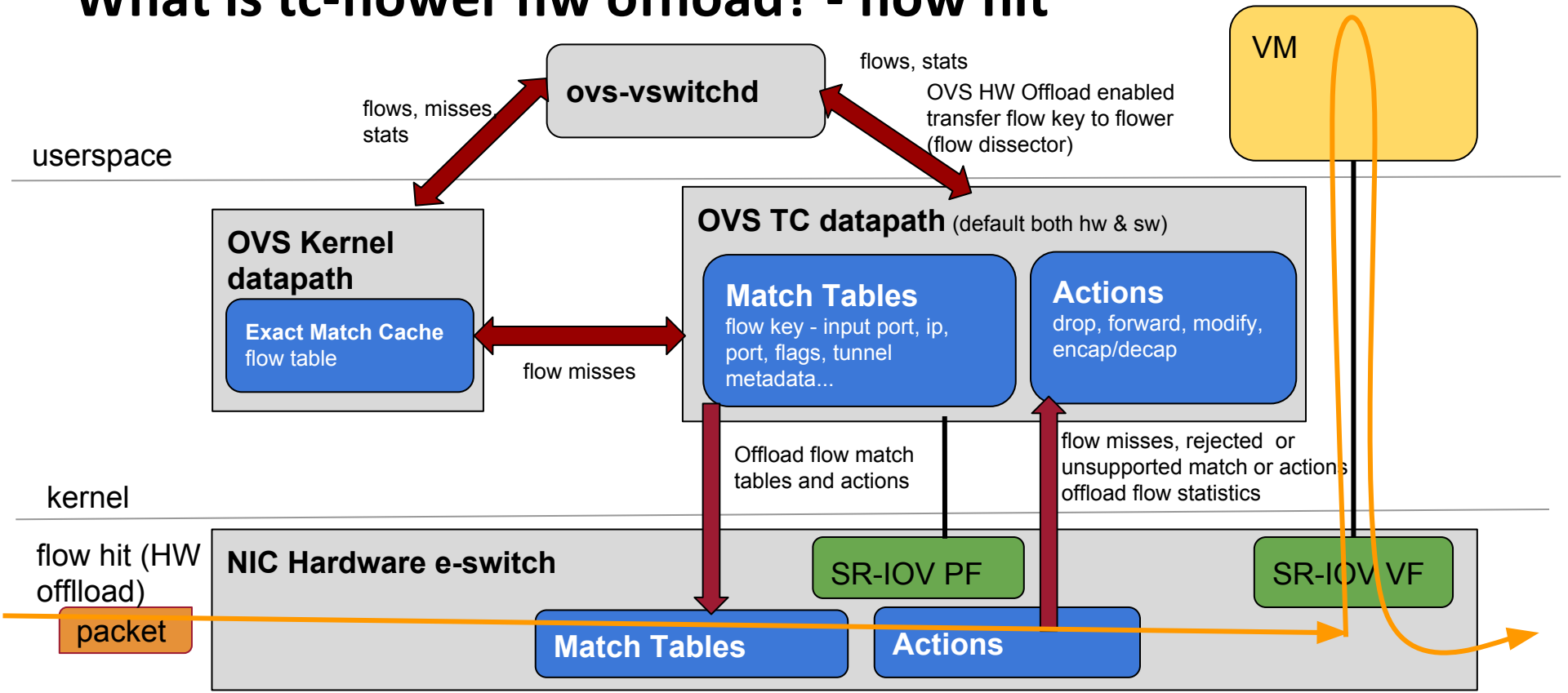
- Basic offload - tunneling, firewall (stateless)
- Match - input port, 5 Tuple -L2 MAC, IP src/dst, TCP/UDP port, TCP flags, tunnel metadata
- Action - Drop, Forward, Encap VLAN, VXLAN, set VLAN priority, IP ToS/DSCP & TTL, ARP
- Future - Bonding, IPv6 ND, more set actions - MPLS, GRE, Geneve; Stateful Firewalls with Connection tracking



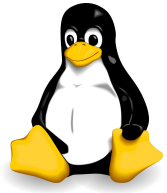
What is tc-flower hw offload? flow miss



What is tc-flower hw offload? - flow hit



Communities & Integration



[TC-flower] [kernel 4.8+]- flow based traffic classifier

- `# tc qdisc add dev eth0 ingress`
- `# tc filter add dev eth0 protocol ip parent ffff: flower skip_sw ip_proto tcp dst_port 80 action drop`

Host kernel requires NIC vendor **SR-IOV PF driver**,

- NIC vendor **tc-flower offload** capability
- `# ethtool -K ens1p1 hw-tc-offload on`

Open vSwitch (OVS) 2.8+ - tc-flower offload

- `ovs-vsctl set Open_vSwitch . other_config:hw-offload=true`

Guest kernel requires vendor **SR-IOV VF driver**

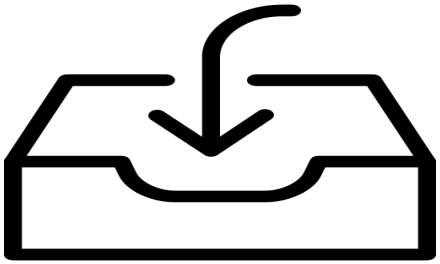
OpenStack Pike/Queens TripleO blueprint

- Nova SR-IOV PCIPassThrough with VF representor to OVS
- `# openstack port create --network NorthSouthData --vnic-type=direct --binding-profile '{"capabilities": ["switchdev"]}' sr-iov_port1`
- assign port to guest instance vm1
- `# openstack server create --flavor m1.small --image rhel75 --nic port-id=sr-iov_port1 vm1`

What if I told you - Everything is upstream and in-box

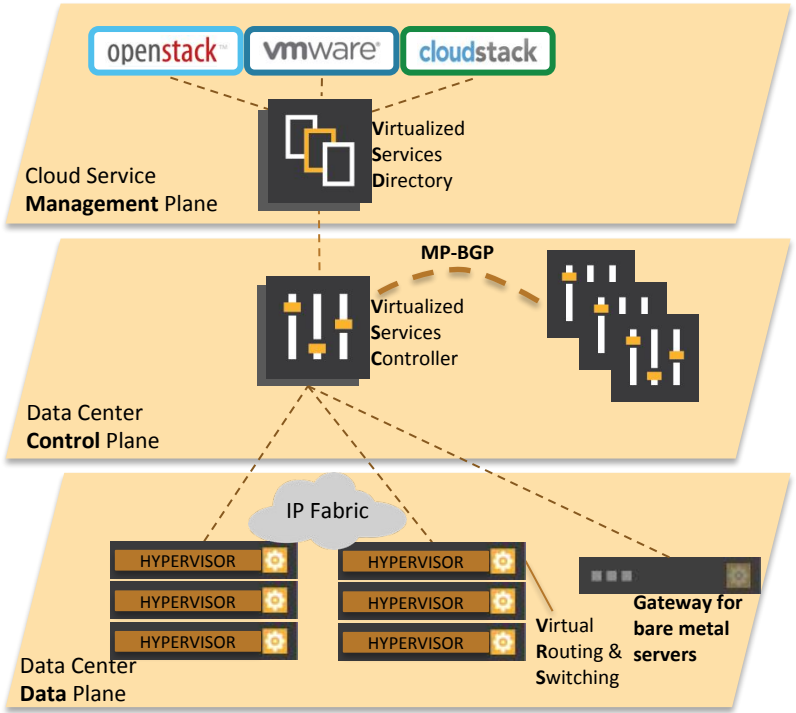
No Kernel Patches

- ✔ NIC drivers
- ✔ Kernel TC-flower
- ✔ OVS HW offload
- ✔ OpenStack VIF, Nova/Neutron
- ✔ OpenStack TripleO



RHOSP 13, RHEL 7.5 (Tech Preview)

Nuage Networks Virtualized Services Platform (VSP)



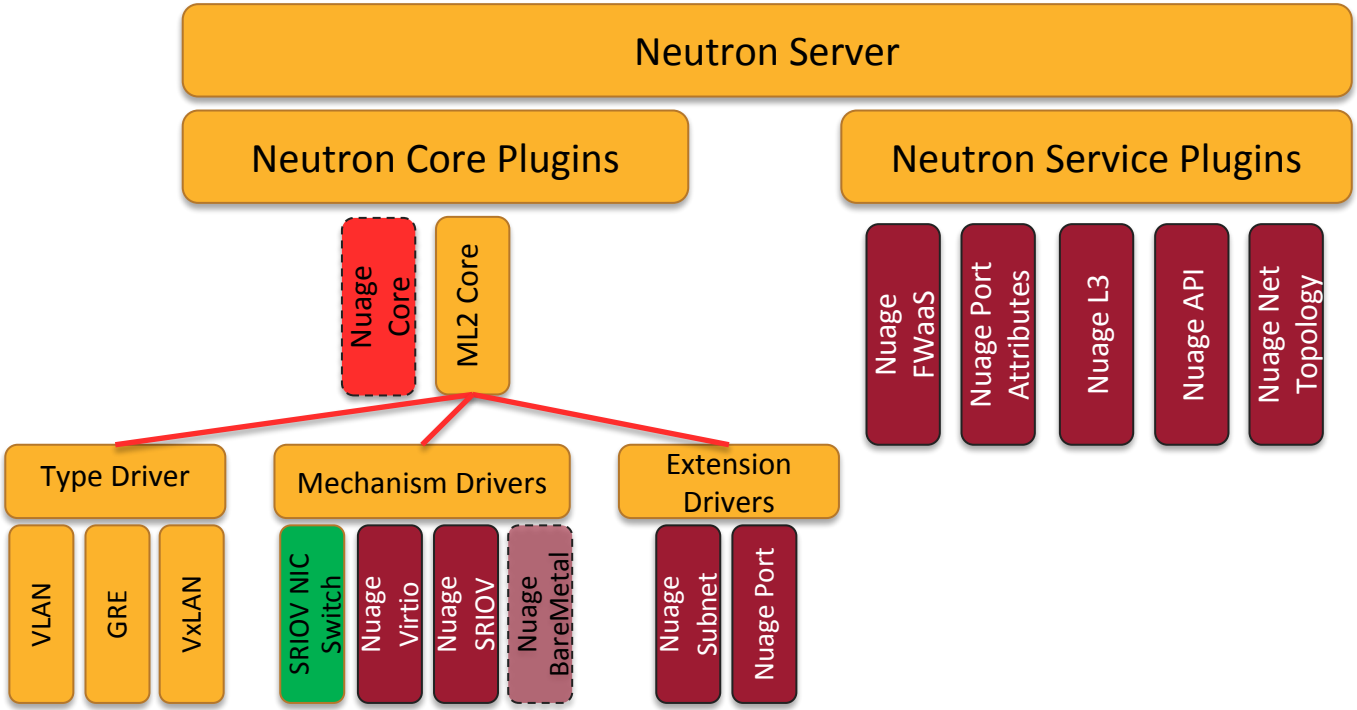
Nuage Networks
Virtualized Services Platform (VSP)

Virtualized Services Directory (VSD)
- Network Policy Engine – abstracts complexity
- Service templates and analytics

Virtualized Services Controller (VSC)
- SDN Controller, programs the network
- Rich routing feature set

Virtual Routing & Switching (VRS-TC)
- Distributed switch / router – L2-4 rules
- Integration of bare metal assets

Evolution of Nuage VSP Openstack Integration



Drivers for change

1. Work together with SR-IOV ports
2. Implement OS-Managed in ML2 context
3. Implement fabric automation for SR-IOV port
4. Implement fabric automation for e.g. Ironic

SmartNIC Demo

Technology Overview

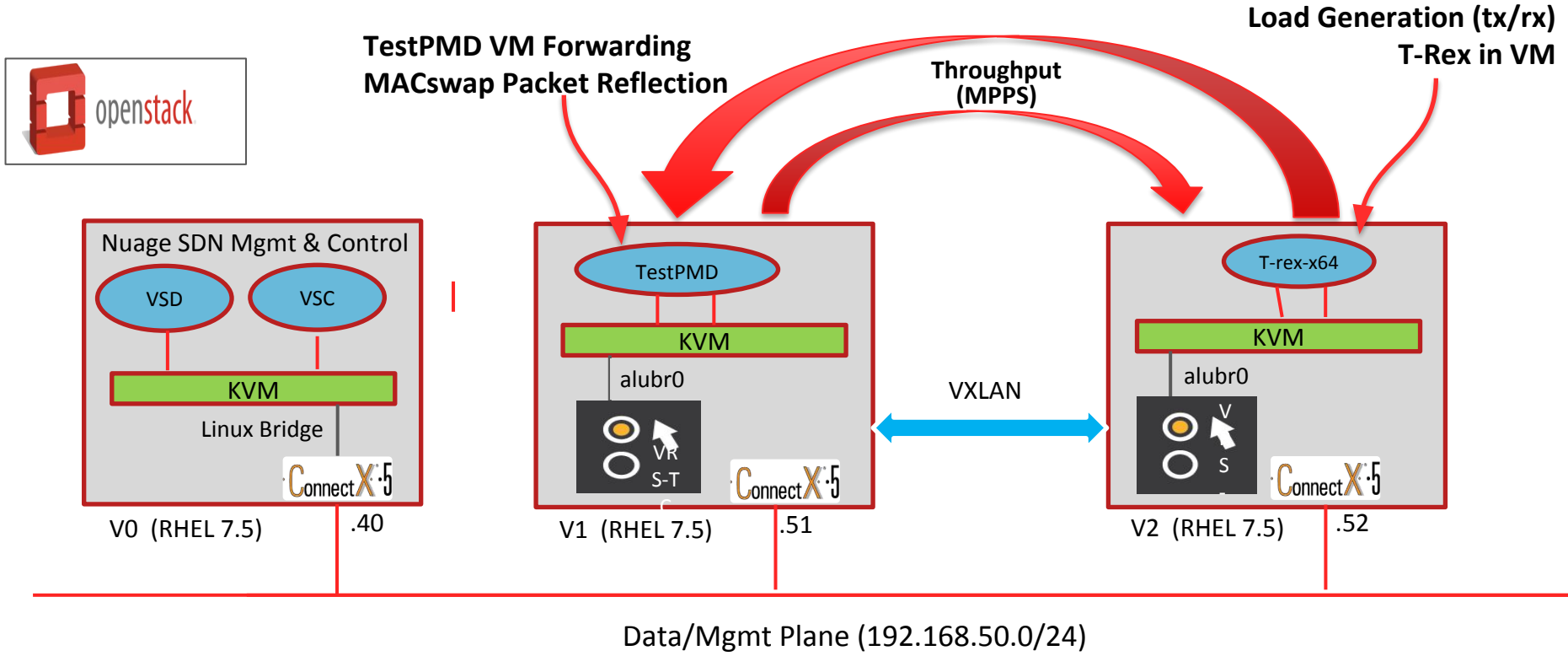
- Nuage Virtualized Service Platform or VSP (SDN) Components
  - Virtualized Services Directory or VSD—management, automation, REST APIs
  - Virtualized Services Controller or VSC—scalable SDN controller
  - Virtualized Router and Switch with TC Flower VRS-TC (pre-release)
- Test Configuration see diagram
  - 2x VRS-TC Intel x64 Compute Nodes
  - ConnectX-5 NICs with ASAP^2 Direct OVS offloading capability
  - Mellanox SN2100 100Gbps switching fabric

DEMO—VXLAN Packet Tunneling:

- SRIOV vs historical VIRTIO
- Zero CPU networking with OVS offloading

Packet Throughput Test Setup

DPDK Applications generate and forward packets over Nuage manage VXLAN tunnel



Demo Script

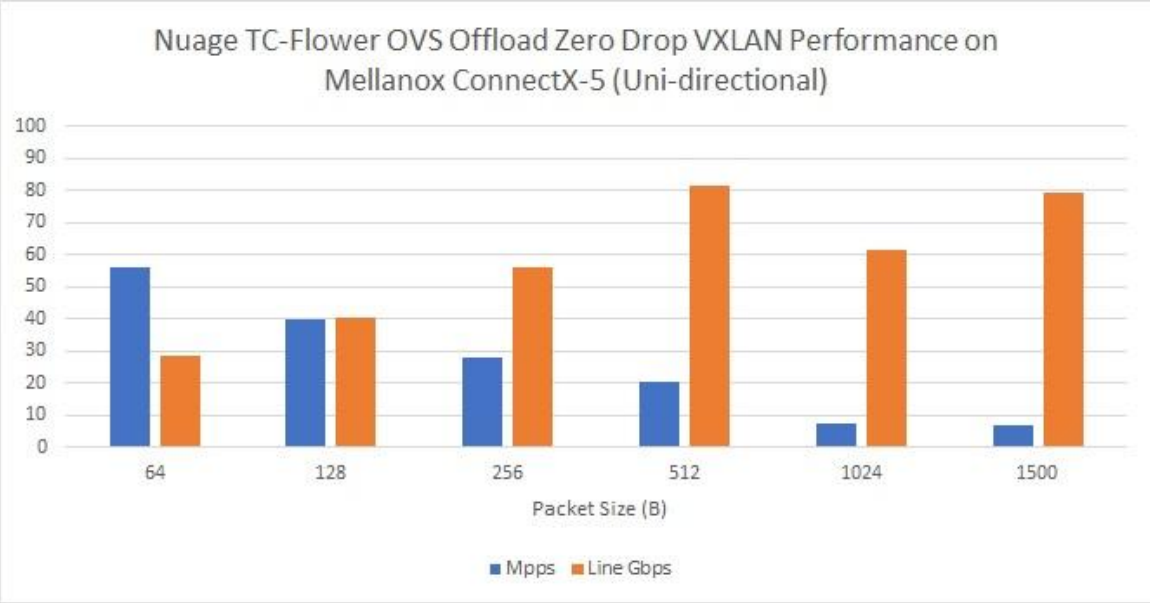
1. Test Setup Slide
2. VSD Configuration
3. Instantiate L3 Template
4. Login to v1 and v2 show we're running RHEL 7.5
5. Configure OVS and NIC on both systems
6. Launch pg1 VM
7. Launch trex VM
8. See vports created on VSD
9. Verify overlay connectivity
10. Start testpmd with SRIOV interface on v1
11. Start t-rex-x64 with SRIOV interface on v2
12. Start trex-console
13. Start T-rex TUI
14. Start load gen
15. Examine testPMD stats
16. Examine trex tui
17. Dump flows ovs-dpctl
18. Dump NIC flows
19. Repeat 6-19 using virtio

Nuage VSP on ConnectX-5 Offload Performance

OVS Flows are programmed via tc-flower interface

Results:

- Zero CPU usage for VXLAN tunnels
- Uni-directional measurements (multiple by 2 for bi-directional)
- No packet loss in forwarding app
- T-Rex and TestPMD run in VMs
- 2 active tunneling flows



System Specs:

- E5-2667 v3 @ 3.20GHz
- Mellanox ConnectX-5 NIC (100Gbps)
- RHEL 7.5 Host and Guest
- Mellanox SN2100 Fabric Switch

OVS Offload Availability Status

Who's Ready to Test Drive at Warp Speed?

Open Source Components:

- **Kernel** code is upstream: **Kernel 4.8+**
- **OVS** code is upstream: **OVS 2.8+**
- **OpenStack Release: Queens**

Commercial Products:

- **Mellanox: ConnectX-4 and ConnectX-5**
- **Red Hat: RHEL 7.5 and RHOSP 13** (Tech-Preview)
- **Nuage Networks:** VSP 6.0 (Target)