redhat.

Putting DPDK in production

DPDK users perspective

Franck Baudin, Principal Product Manager, OpenStack NFV
Anita Tragler, Senior Product Manager, Red Hat Enterprise Linux NFV

DPDK summit USA 2016, August 10-11, 2016

Why user stories are important: Timeline & feature prioritization

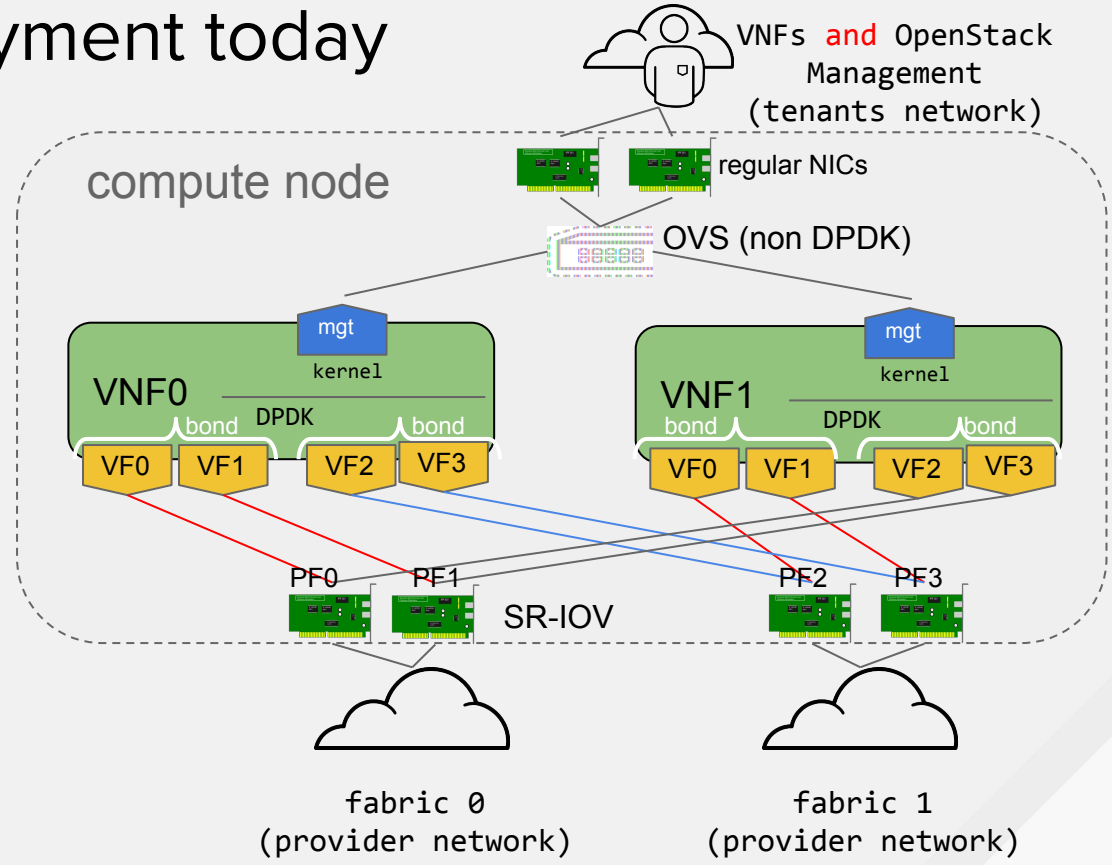
Typical VNF deployment today

Depending on the VNF, it will be connected to one or more fabric network, typically:

- One fabric for control plane
- One fabric for user plane (end user datas: mobile traffic, web, ...)

One or two “fat VM” per numa node

- DPDK in the VM, often old/patched
- Bonding for HA in the VM
- SR-IOV to ensure high performance
- VLAN based tenant (no VxLAN)
- No containers in the picture, no vhost-user



OpenStack SR-IOV
Host/VNFs guests ressources partitioning

Typical 18 cores per node dual socket compute node (E5-2599 v3)

Hyperthreaded and non HT cores supported

This looks like RT but is not RT, just partitioning

SR-IOV interfaces bonding handled by the VNF

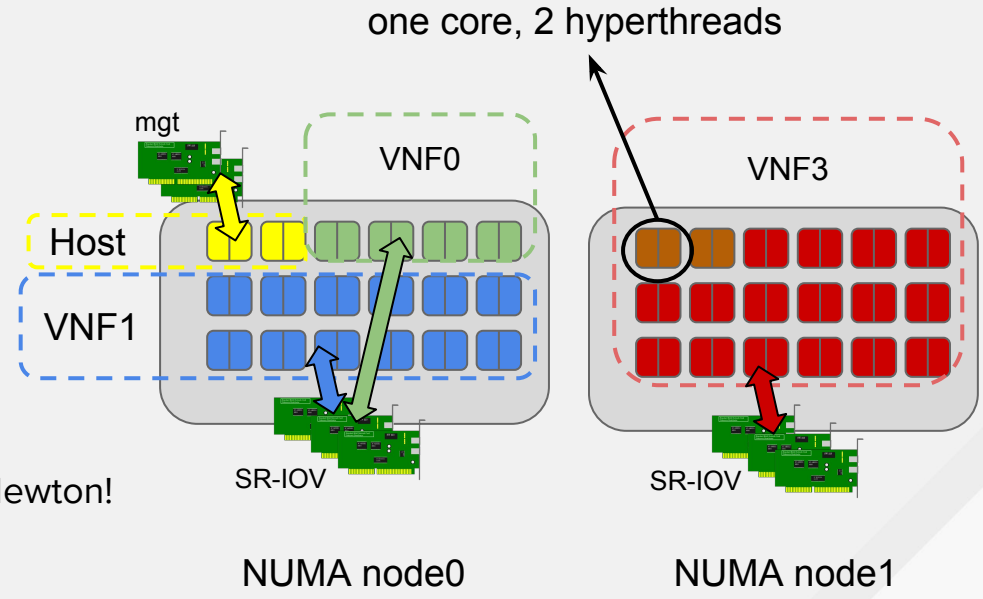
All host IRQs routed on host cores

All VNFx cores dedicated to VNFx

- Isolation from others VNFs
- Isolation from the host

Cross NUMA VNFs not supported by OpenStack Newton!

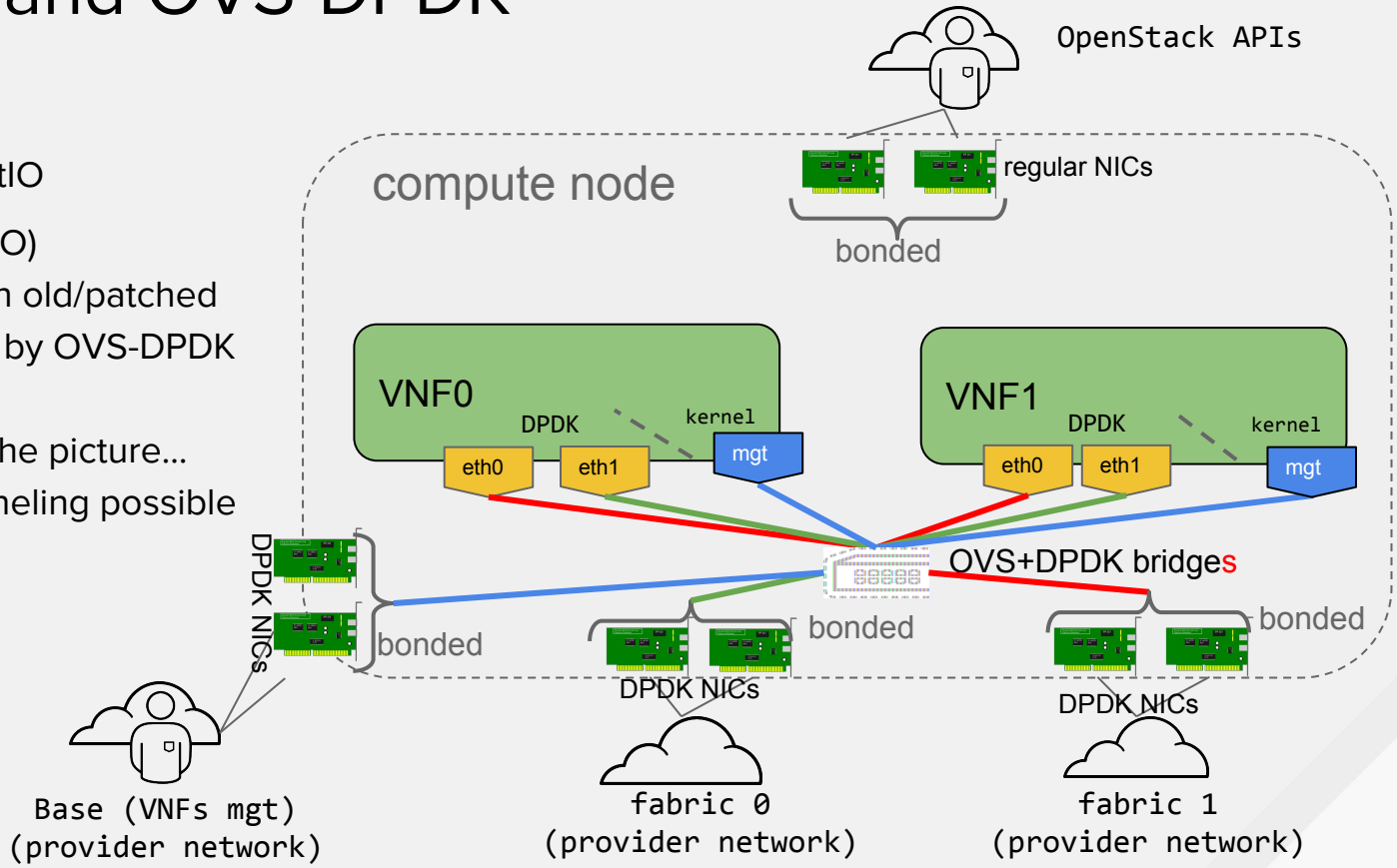
- a VNF has to fit on a single NUMA node
- A VNF has to use the local SR-IOV NICs



OpenStack and OVS-DPDK

Same fat VMs ported on virtIO

- Full virtualization (virtIO)
- DPDK in the VM, often old/patched
- Bonding for HA done by OVS-DPDK
- Performances TBC...
- Still no containers in the picture...
- VxLAN and other tunneling possible



OpenStack OVS-DPDK
Host/VNFs guests ressources partitioning

Typical 18 cores per node dual socket compute node (E5-2599 v3)

Hyperthreaded and non HT cores supported

This looks like RT but is not RT, just partitioning

interfaces bonding handled by OVS-DPDK

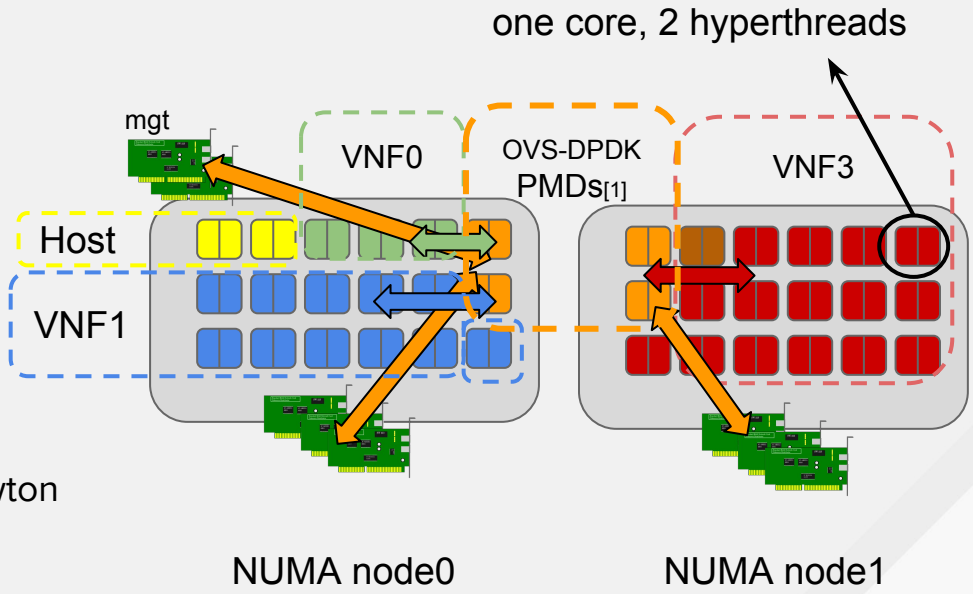
All host IRQs routed on host cores

All VNF(x) cores dedicated to VNF(x)

- Isolation from others VNFs
- Isolation from the host

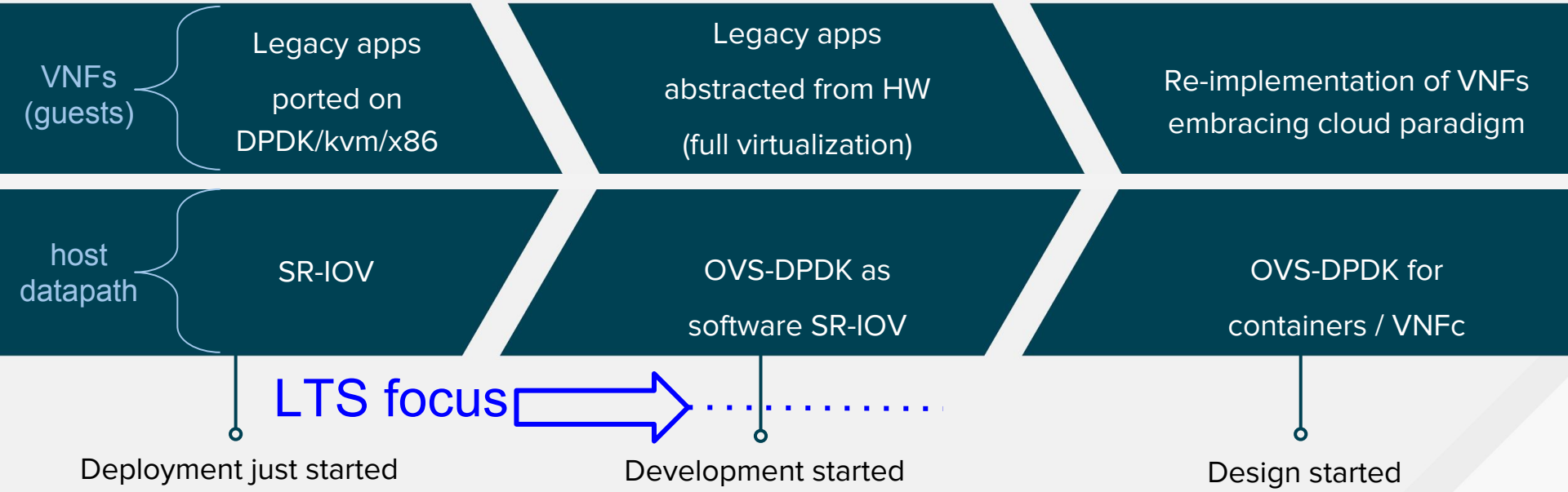
Cross NUMA still not supported by OpenStack Newton

- a VNF has to fit on a single NUMA node
- A VNF has to use the local NICs



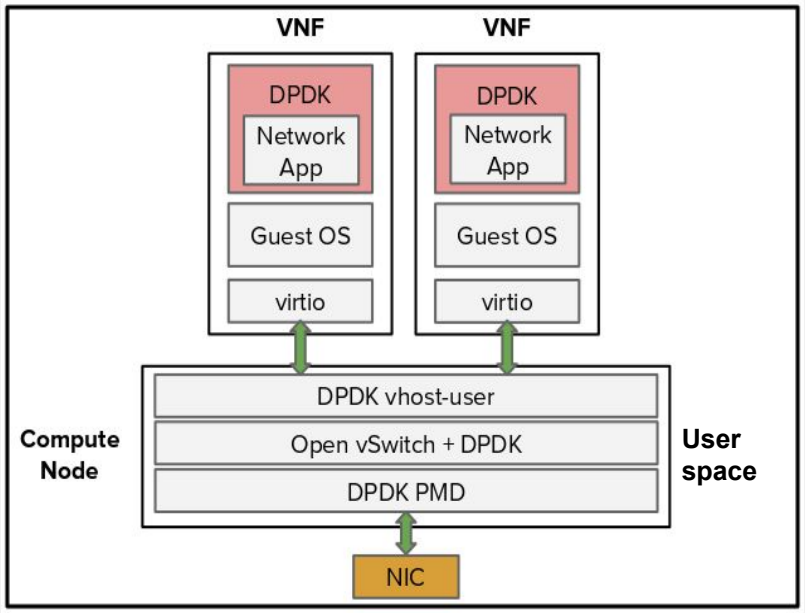
VNFs evolution driving host datapath evolution

For high performances VNFs (DPDK based)



Why user stories are important: consumability

DPDK LTS and ABI stabilization
OVS is dependent on DPDK



DPDK accelerated OVS moving to Deployment?

- [dpdk.org] Upstream ABI changing. May need time to stabilize
- Every Time DPDK changes OVS needs to be upgraded
- OVS has LTS (upstream) and stable ABI- major fixes backported for 2-3 years. Good for deployment :)
- OVS upstream [ovs.org] has DPDK as TechPreview; OVS-DPDK difficult to deploy for long term

OVS-DPDK example

LTS: what is going into production

OVS provides LTS version, **but** sees DPDK update as a new feature: stuck on a DPDK version!

- OVS 2.5 which is the latest LTS, is based DPDK 2.2.0!
- Updating DPDK to 2.2.1, 2.2.2, … would however be acceptable and welcome

Delayed to next LTS

- Drivers fixes, including new PCI ID support, and critical bug fixes, CVEs
- vhost-user performances improvement (it’s a bug, not a feature!)
- Feature maturation (not addition), for instance live migration fixes/improvements

Typical issue of **layered** projects/product: lag behind dependencies version

- LTS requires stable ABIs or backports becomes impossible

All VNFs vendors have the same issue, behind closed doors: re-validate my vEPC on each DPDK?

vSwitches and VNFs end to end performance

PP performance always “marketed”

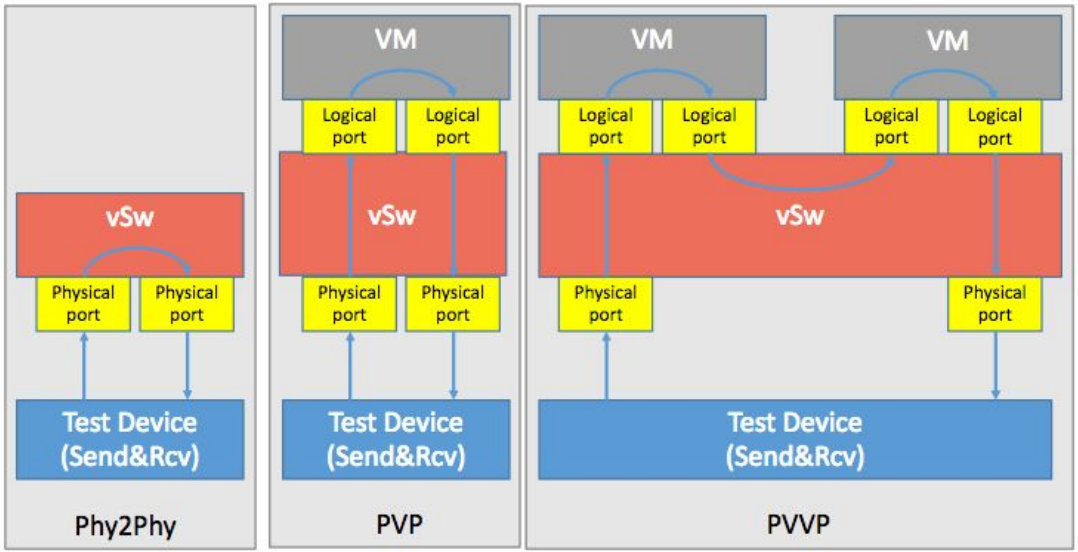
- OVS-DPDK, VPP, …

PVP and PVVP, with zero frame loss

- Today *not* measured with Zero frame loss as zero frame loss performances are quite “low”
- **Baseline numbers from dpdk.org with end to end documentation would gives a baseline regarding vhost-user**

VSPERF LTD Supported Deployment Scenarios

Credit: Maryam Tahhan - Al Morton
Benchmarking Virtual Switches in OPNFV
draft-vsperf-bmwg-vswitch-opnfv-01



redhat.

Provide public and reproducible performances

DPDK counterpart of OPNFV VSperf and FD.io CSIT

OPNFV VSperf project OVS-DPDK and later VPP performances

What would help: DPDK providing vhost performance measurement and methodology

Call to action:

1. Add vhost support to testpmd (*testpmd as a vSwitch*) and provide pvp and pvvp performances, ideally add them in a **public** CI: it's hard to tell today if a vSwitch just better configures DPDK or if it is really better than others as vhost-user performance. Front-end and back-end cycles/packet is publicly unknown.
2. Standardize on methodology and tools: topologies, traffic generator, method to partition DPDK and non DPDK cores (CentOS tuned profiles? scripts?) for both guests and host
3. Can DPDK community contribute to VSperf or FD.io CSIT project to add a pure testpmd performance test for pvp and pvvp topology?

Distribution integration (RHEL/CentOS/Fedora)

driverctl [OPTIONS] COMMAND [DEVICE [DRIVER]]

driverctl may be used to manipulate and inspect the system device driver choices.

Devices are normally assigned to their sole designated kernel driver by default. However in some situations it may be desirable to override that default, for example to try an older driver to work around a regression in a driver or to try an experimental alternative driver. Another common use-case is pass-through drivers and driver stubs to allow userspace to drive the device, such as in case of virtualization.

driverctl integrates with **udev** to support overriding driver selection for both cold- and hotplugged devices from the moment of discovery, but can also change already assigned drivers, assuming they are not in use by the system. The driver overrides created by **driverctl** are persistent across system reboots by default.

Monitoring & debug

“Telco Grade”

Monitor DPDK applications (Example of usage: OpenStack Monitoring, OPNFV Doctor)

- Keep-alive
- Extended statistics (why packets are lost/dropped?)

Tcpdump capabilities

Sampling capabilities (sflow)

Real time network analyzer: skydive, end to end analyzing at scale

- leverage all features above but Keep-alive, see the online demo

SFQM Overview

- Develop the utilities and libraries in DPDK to support:
  - Measuring Telco Traffic and Performance KPIs. Including:
    - Packet Delay Variation (by enabling TX and RX time stamping).
    - Packet loss (by exposing extended NIC stats).
  - Performance + status Monitoring of the DPDK interfaces (by exposing extended NIC stats + collectd Plugin).
  - Detecting and reporting violations that can be consumed by VNFs and higher level fault management systems (through DPDK Keep Alive).

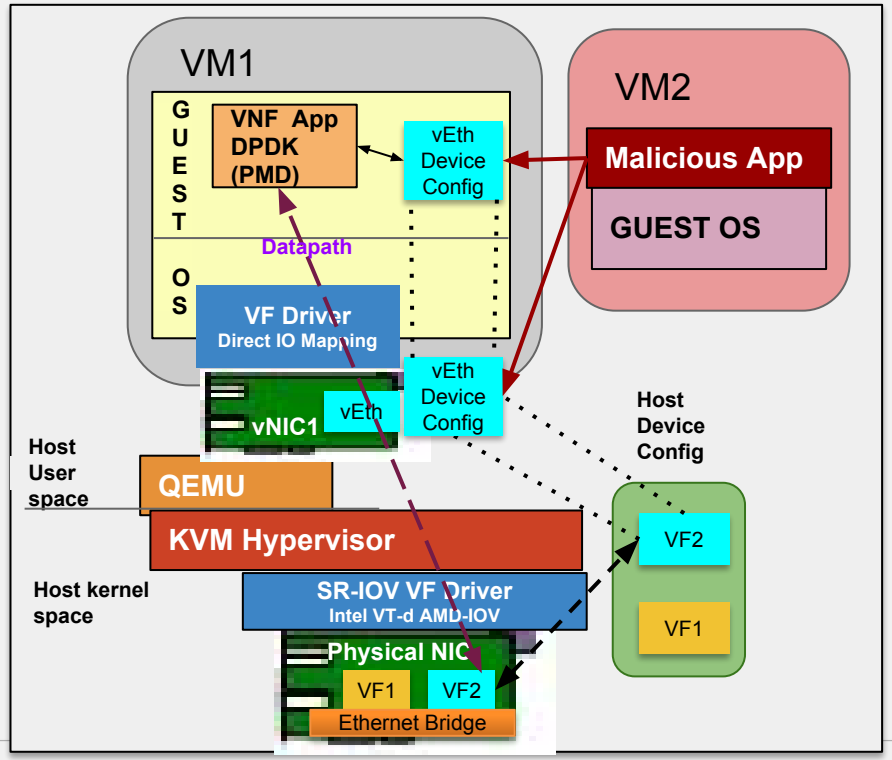
The ability to measure and enforce Telco KPIs in the data-plane will be mandatory for any Telco grade NFVI implementation.

Credit: https://www.openstack.org/assets/presentation-media/OpenStack-Summit-Austin-4.pdf

Why user stories are important: How secure is your DPDK application?

Do you trust your neighbors?

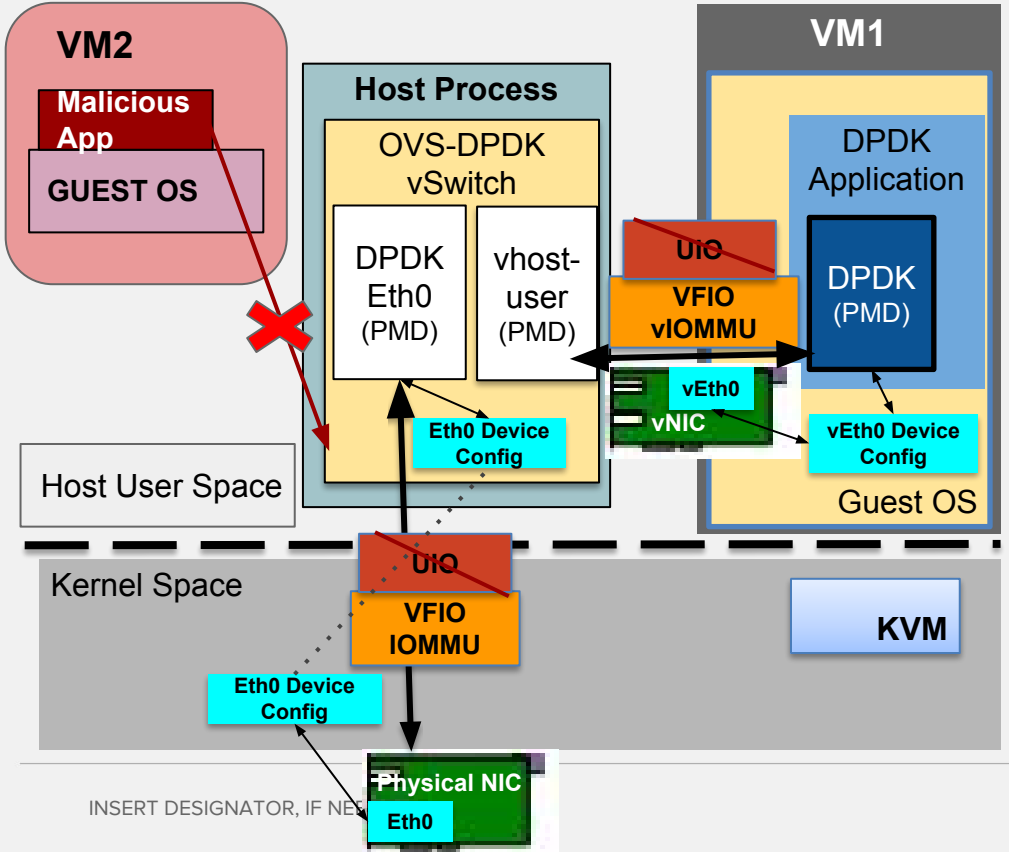
DPDK Guest with SR-IOV



- DPDK needs direct device (PCI/NIC) access in userspace; bypassing the kernel
- DPDK PMD is a userspace driver that enables **DPDK Guest** to talk **directly to vNIC** via **direct IO mapping** using **Direct Memory Access(DMA)**;
- vNIC **device config space(kernel)** gets mapped to **application address space (user space)**
- DMA in the guest opens **guest kernel main memory** to erroneous devices (vNICs) and other **malicious VMs** and **userland processes**
- SR-IOV allows VM (userspace) to directly access(DMA) NIC (kernel) address space
- DMA in host opens host main memory to userland processes

How secure is your host?

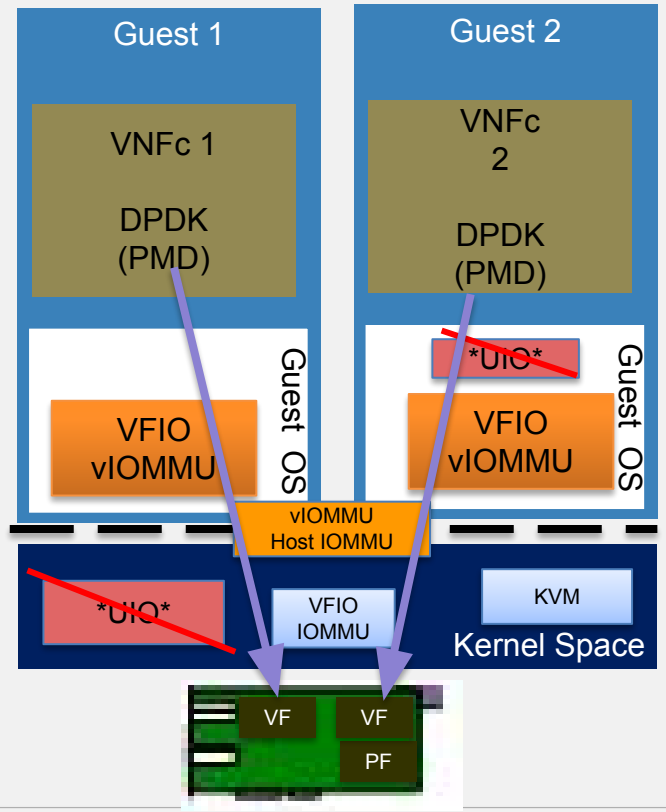
DPDK Guest with DPDK accelerated vSwitch (OVS-DPDK)



VFIO IOMMU driver for Host

- KVM allows PCI device DMA with special pass-through drivers
- *uio* drivers - not designed for DMA devices; no protection against userspace corrupting host kernel memory
  - uio_pci_generic - only legacy interrupts; no support for MSI/MSI-x; does not work with SR-IOV VFs
  - igb_uio - not available upstream
- VFIO provides safe, non-privileged, IOMMU protected DMA access; supported on Intel VT-d and AMD-IOV
- IOMMU enforces safe ownership of devices with access permissions and per device DMA mappings

How do I protect my DPDK guest application?



VFIO vIOMMU driver for DPDK Guest [WIP]

- VFIO IOMMU not supported for Guest
- VFIO with emulated IOMMU [vIOMMU] for Guest VM is WIP upstream (6 -12 months)
- Use Host IOMMU to emulate IOMMU for VM
- Offload IO remapping to HW (Intel VT-d, AMD-IOV)

Interim solution - VFIO NO-IOMMU

- One pass-through driver VFIO with unsafe option
- Provides better security than *UIO* - protects MSIX BAR
- DMA is still possible but without memory protection;
- Use when IOMMU is not available - public cloud (Amazon EC2) OR
- IOMMU not enabled in chipset BIOS for performance reasons

Why user stories are important: Summary

DPDK Consumers: VNFs and vSwitches

Consumability

- Ease performances measurement and reproducibility
- LifeCycle (LTS) and stable ABIs

Security:

- VFIO on the host, documentation should be prescriptive
- VFIO on the guest, in progress

Telco Grade: feedback expected via OVS-DPDK and VPP projects at least

- Bonding (HA)
- Graceful restart of a DPDK vSwitch without disconnecting VMs

Best (open) source of user stories: OPNFV, as it implement end to end examples for NFV

- OVS-DPDK, VPP, OpenStack, OpenDaylight, VSperf, Doctor, Moongen, ...

redhat.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos