KVM FORUM

# Configuring and Benchmarking Open vSwitch, DPDK and vhost-user

Pei Zhang (张 培)
pezhang@redhat.com
October 26, 2017

redhat.

# Agenda

1. Background
2. Configure Open vSwitch, DPDK and vhost-user
3. Improve network performance
4. Show results

# 1.Background(1/2)

**NFV** stands for **N**etwork **F**unction **V**irtualization, it’s a new network architecture concept.

Softwares

\+ Virtualization

\+ Standard hardwares

**Replace**

Dedicated network appliances

redhat.

# 1.Background(2/2)

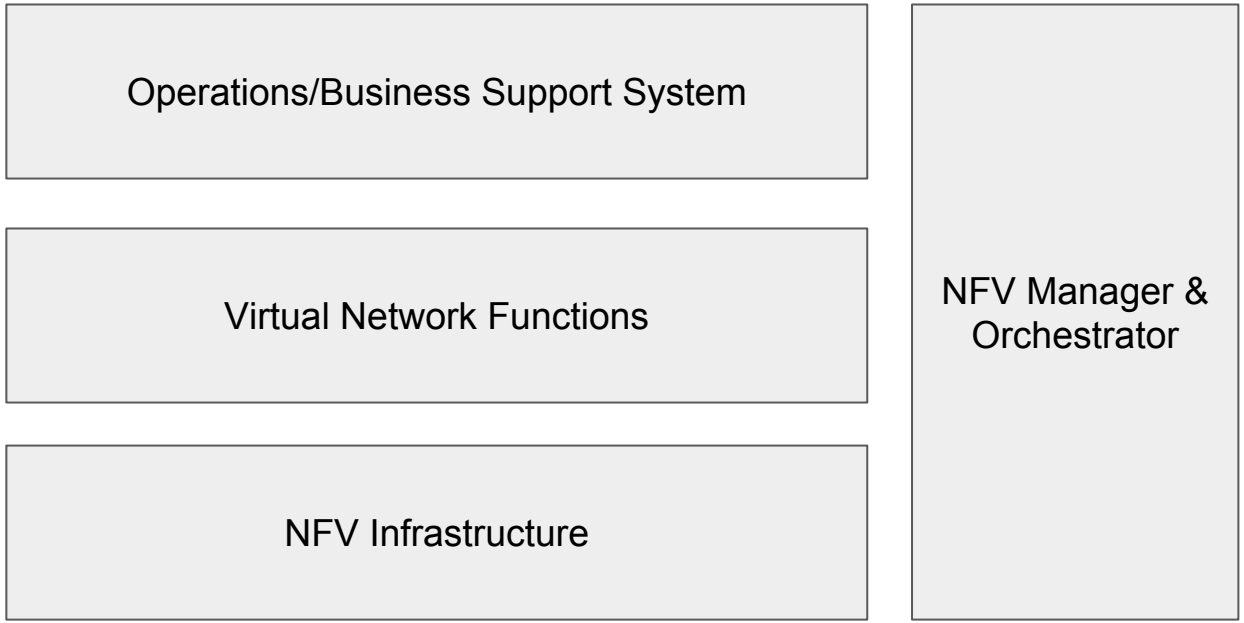



**Fig. ETSI NFV Architecture Framework**

redhat.

# 1.Background(2/2)

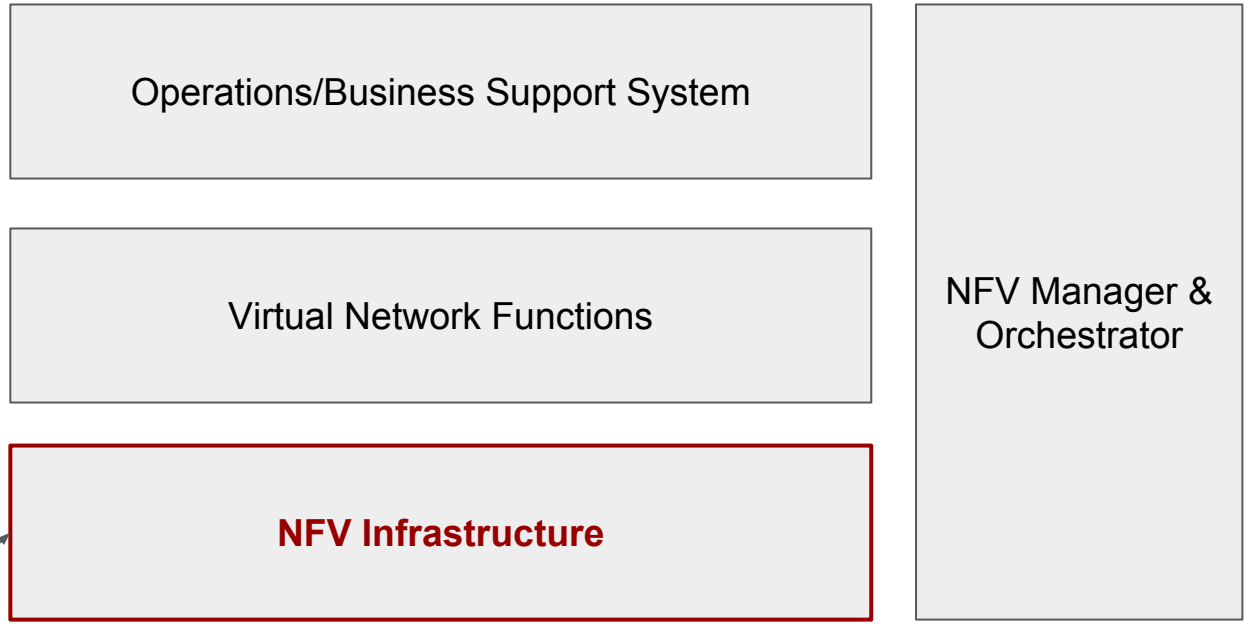



**Fig. ETSI NFV Architecture Framework**

NFVI provides basic environment for network performance.

redhat.

# 2.Configure Open vSwitch, DPDK and vhost-user

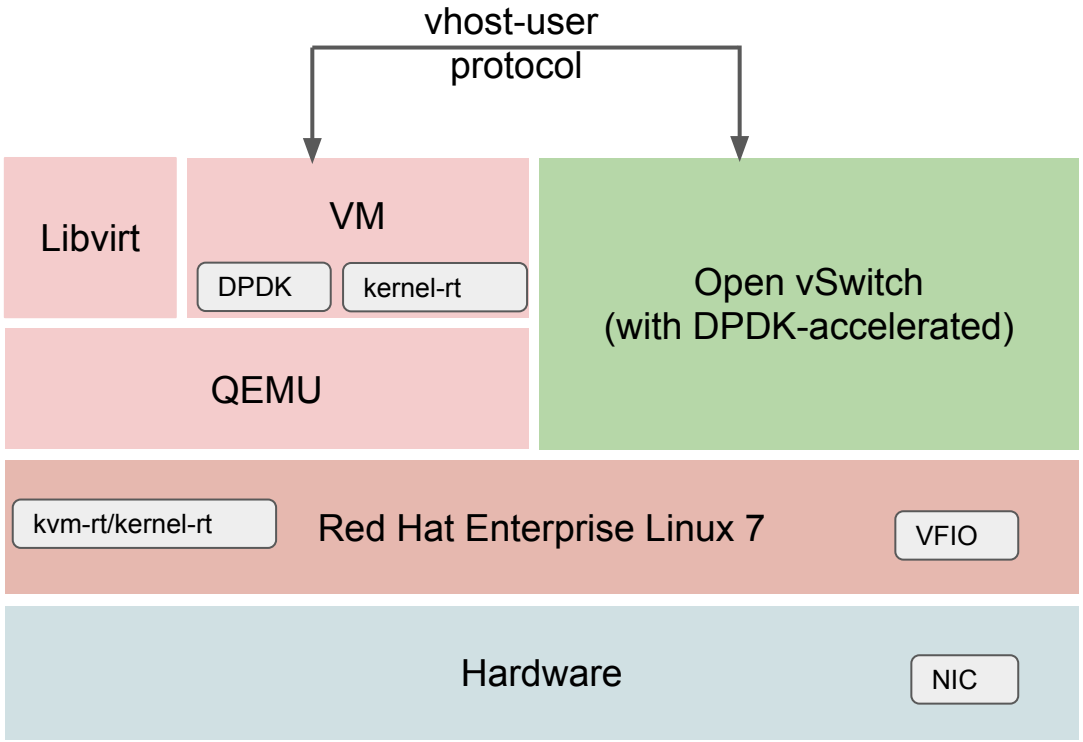



Fig. Topology

redhat.

# How performance is improved? - DPDK

Data Plane Development Kit(DPDK) is a set of libraries and user space drivers for fast packet processing.

- **polling mode** drivers
- using **hugepage** memory.
- running mostly in **user space**.

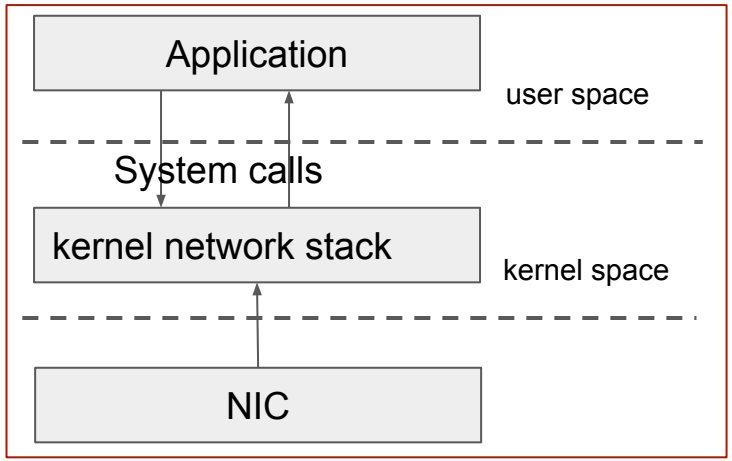



Fig. standard packets flow

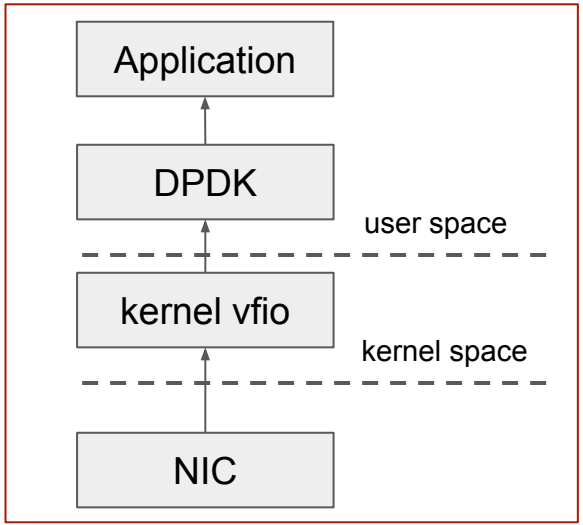
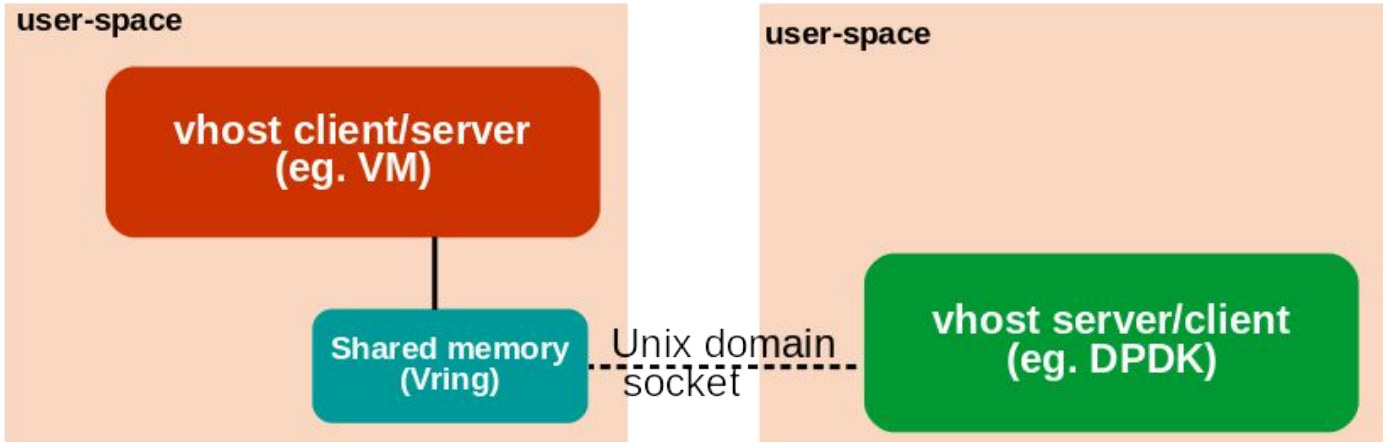



Fig. packets flow with dpdk

redhat.

# How performance is improved?  - vhost-user

vhost-user protocol allows qemu shares virtqueues with a user space process on the same host.




redhat.

# How performance is improved?  - Open vSwitch

Open vSwitch(OVS) is designed to be used as a vSwitch within virtualized server environments.

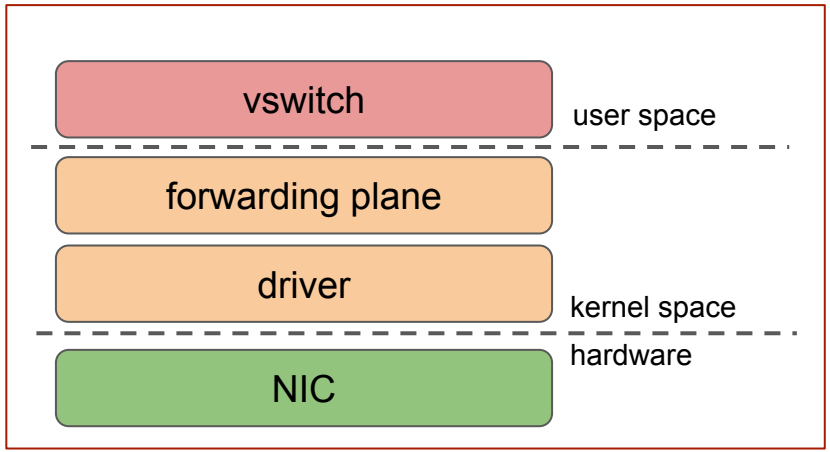



Fig. standard OVS

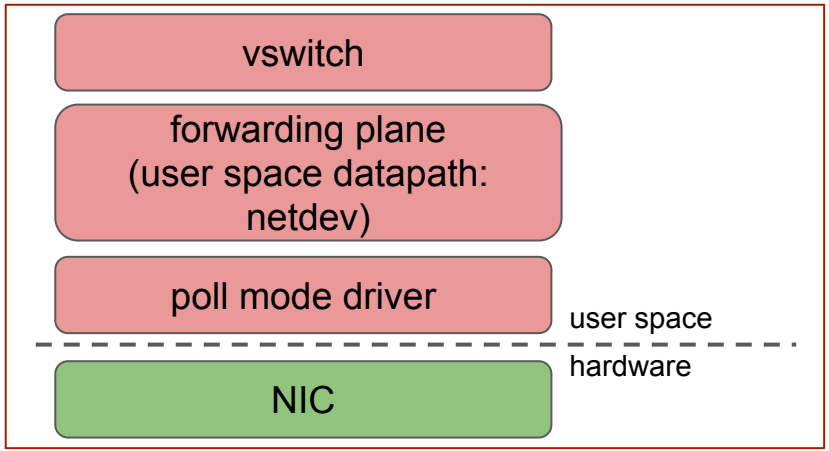



Fig. OVS with DPDK

redhat.

# How performance is improved?  - KVM-RT

Real-Time always keeps low latency, it’s used for latency-sensitive workloads.

Real-Time KVM(RT-KVM) is the extension of KVM, it allows the VM to be real time operating system.

KVM-RT can be used in latency-sensitive VNFs.

# Peculiarities summary

**(1) Handling network packets in user space during whole process.**

**(2) Polling thread.**

**(3) Hugepages.**

**(4) Cores isolation**

- Isolated cores will only be used when explicitly setting.
- Pin vCPUs to individual cores

**(5) Strict NUMA policy**

- Cores and memory used should be same NUMA node with network device.

# How to config? - vhost-user socket

```
<cpu mode='host-passthrough' check='none'>
   <feature policy='require' name='tsc-deadline'/>
   <numa>
      <cell id='0' cpus='0-3' memory='8388608' unit='KiB' memAccess='shared'/>
   </numa>
</cpu>
```

```
<interface type='vhostuser'>
   <mac address='88:66:da:5f:dd:02'/>
   <source type='unix' path='/tmp/vhostuser0.sock' mode='server'/>
   <model type='virtio'/>
   <driver name='vhost'/>
   <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
```

# ovs-vsctl add-port ovsbr0 vhost-user0 -- set Interface vhost-user0 type=**dpdkvhostuserclient** options:**vhost-server-path=/tmp/vhostuser0.sock**

# How to config?  - hugepage

```
# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-...  default_hugepagesz=1G

# lscpu
Flags: ... pdpe1g ...
```

```
<memoryBacking>
        <hugepages>
        <page size='1048576' unit='KiB' nodeset='0'/>
        </hugepages>
        <locked/>
  </memoryBacking>
```

# How to config?  - isolate cores

**In normal kernel environment:**

Install package: tuned-profiles-cpu-partitioning

Kernel line:
# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-...  skew_tick=1 nohz=on nohz_full=1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,30,28,26,24,22,20,18,16 rcu_nocbs=1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,30,28,26,24,22,20,18,16 tuned.non_isolcpus=00005555 intel_pstate=disable nosoftlockup

**In real time environment:**

Install package: tuned-profiles-nfv-host/guest

# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-...  skew_tick=1 isolcpus=1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,30,28,26,24,22,20,18,16 nohz=on nohz_full=1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,30,28,26,24,22,20,18,16 rcu_nocbs=1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,30,28,26,24,22,20,18,16 intel_pstate=disable nosoftlockup

# How to config? - NUMA policy

```
# hwloc-ls
Machine (64GB total)
  NUMANode L#0 (P#0 32GB)
        …
  NUMANode L#1 (P#1 32GB)
        ...
        PCIBridge
        PCI 8086:1528
        Net L#7 "p1p1"
        PCI 8086:1528
        Net L#8 "p1p2"
```

Intel X540-AT2 10G Card

```
<vcpu placement='static'>4</vcpu>
<cputune>
      <vcpupin vcpu='0' cpuset='31'/>
      <vcpupin vcpu='1' cpuset='29'/>
      <vcpupin vcpu='2' cpuset='27'/>
      <vcpupin vcpu='3' cpuset='25'/>
      <emulatorpin cpuset='18,20'/>
</cputune>
<numatune>
      <memory mode='strict' nodeset='1'/>
</numatune>
```

# ovs-vsctl set Open_vSwitch . other_config:**pmd-cpu-mask=0xAA** (cores 1,3,5,7)

# How to config? - kvm-rt

```
<cputune>
      <vcpupin vcpu='0' cpuset='30'/>
      <vcpupin vcpu='1' cpuset='31'/>
      <emulatorpin cpuset='2,4,6,8,10'/>
      <vcpusched vcpus='0' scheduler='fifo' priority='1'/>
      <vcpusched vcpus='1' scheduler='fifo' priority='1'/>
</cputune>
```

- Install kernel-rt/kernel-rt-kvm
- Use tuned-profiles-nfv/tuned-profiles-realtime
- Set fifo:1 priority

# Testing topology

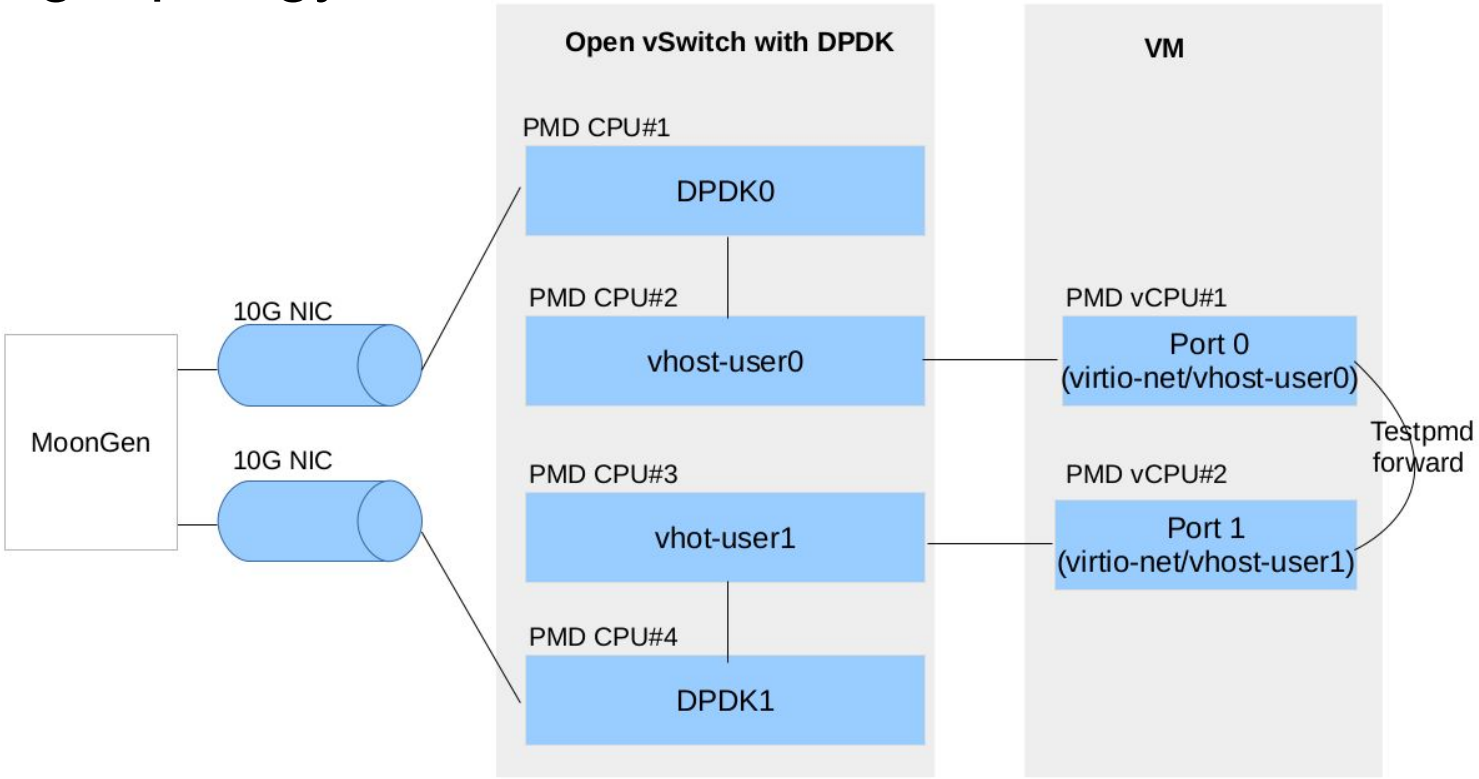



Fig. vhost-user single queue

Note:Using individual core for each port.
(6 cores in this example )

redhat.

# 3.Improve network performance

(1) Using multiple queues to improve throughput

(2) Using tuned-cpu-partitioning to get 0-loss packets and lower L2 network latency

(3) Using KVM-RT to get lower cyclictest latency

# Higher throughput - multiple queues(1/2)

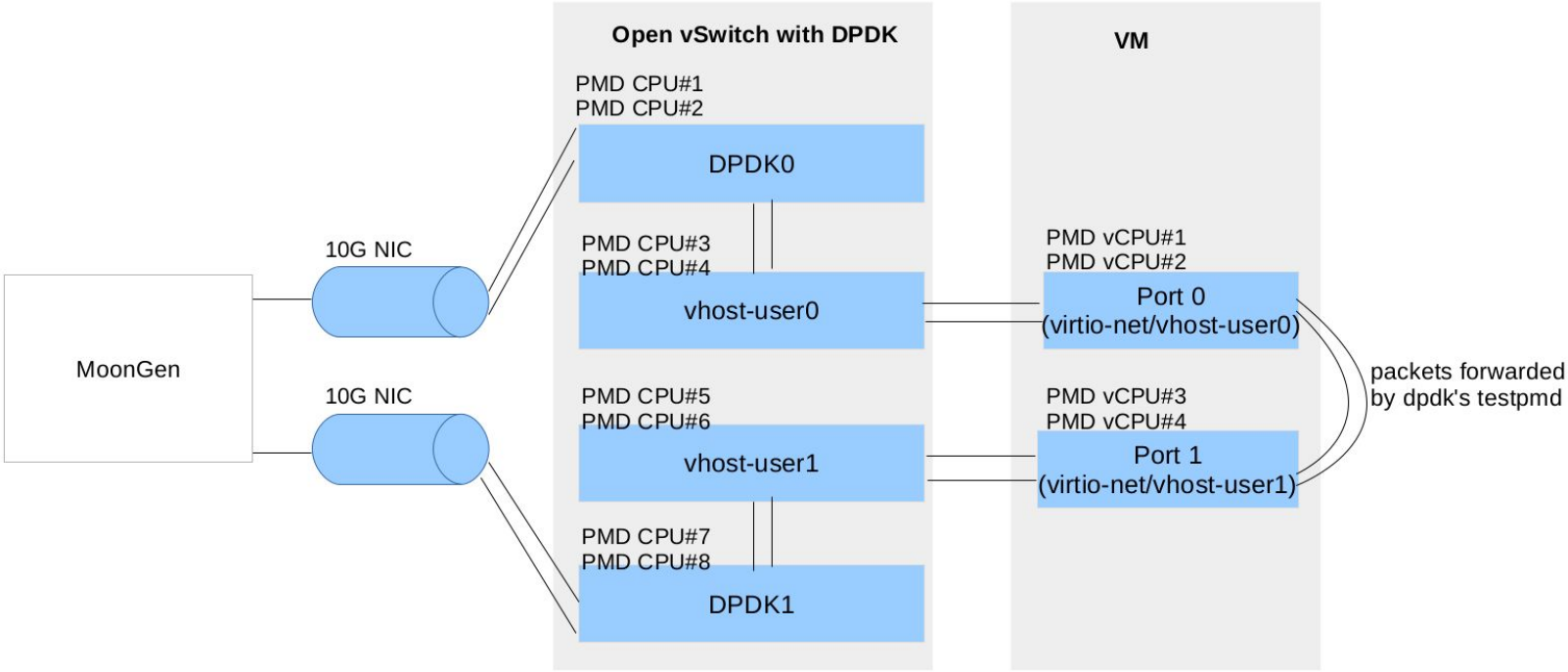



Fig. vhost-user 2 queues

## Cores Number = Ports * Queues

Note:Using individual core for each port each queue. (12 cores in this example)

redhat.

# Higher throughput - multiple queues(2/2)

**Open vSwitch for 2 queues:**

ovs-vsctl set Open_vSwitch . other_config:pmd-cpu-mask=**0xAAAA(1,3,5,7,9,11,13,15)**
ovs-vsctl set Interface dpdk0 options:**n_rxq=2**
ovs-vsctl set Interface dpdk1 options:**n_rxq=2**

**VM for 2 queues:**

```
<interface type='vhostuser'>
  <mac address='88:66:da:5f:dd:02'/>
  <source type='unix' path='/var/run/openvswitch/vhost-user0' mode='client'/>
  <model type='virtio'/>
  <driver name='vhost' queues='2'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
```

redhat.

# 4.Show Results

## (1) Multiple queues has better throughput

**Single queue: 13.02Mpps (43.75% of line rate 14.88Mpps)**

**Two queues: 21.13Mpps (71% of line rate 14.88Mpps)(Better)**

Testing Environment:

- Platform: Red Hat Enterprise Linux 7
- Traffic Generator: MoonGen
- Acceptable Loss: 0.002%
- Frame Size: 64Byte
- Bidirectional: Yes
- Validation run time: 30s
- CPU: Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz
- NIC: 10-Gigabit X540-AT2

# (2) tuned-cpu-partitioning has better 0-loss throughput and L2 network latency

**Throughput:**

- **no cpu-partitioning throughput: 21.13 (0.000718% loss)**
- **cpu-partitioning throughput: 21.31(0 loss)(Better)**

**L2 network latency**

- **no tuned-cpu-partitioning latency: 1242.073us**
- **cpu-partitioning latency: 37us(Better)**

Testing Environment:

- Platform: Red Hat Enterprise Linux 7
- Traffic Generator: MoonGen
- Running time: 12 hours
- Frame Size: 64Byte
- Bidirectional: Yes
- CPU: Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz
- NIC: 10-Gigabit X540-AT2

redhat.

## (3) kvm-rt has better cyclictest latency results

**non-rt:  max cyclictest latency: 00616us**

**kvm-rt:  max cyclictest latency: 00018us(Better)**

Testing Environment:

- Platform: Red Hat Enterprise Linux 7
- Testing method: cyclictest

redhat.

# Q&A

KVM FORUM

# Thanks!

redhat.