Red Hat OpenStack Platform 13

OVS-DPDK End to End Troubleshooting Guide

A guide containing OVS-DPDK end to end troubleshooting procedures

Last Updated: 2021-03-25

Red Hat OpenStack Platform 13 OVS-DPDK End to End Troubleshooting Guide

A guide containing OVS-DPDK end to end troubleshooting procedures

Legal Notice

Abstract

Procedures for OVS-DPDK system administrators to identify and resolve common issues related to packet loss in Red Hat OpenStack Platform 13.

Table of Contents

PREFACE

This document contains procedures for OVS-DPDK system administrators for identifying and resolving common issues related to packet loss in Red Hat OpenStack Platform 13. The procedures documented in this guide supersede the previously published knowledge base articles.

CHAPTER 1. PRELIMINARY CHECKS

This guide assumes that you are familiar with the planning and deployment procedures in the following documents:

- Planning your OVS-DPDK deployment
- Configuring an OVS-DPDK Deployment

CHAPTER 2. VALIDATING AN OVS-DPDK DEPLOYMENT

This chapter describes the validation steps to take following a deployment.

2.1. CONFIRMING OPENSTACK

Use the following commands to confirm OpenStack and OVS-DPDK configuration.

2.1.1. Show the Network Agents

Ensure that the value for **Alive** is **True** and **State** is **UP** for each agent. If there are any issues, view the logs in **/var/log/containers/neutron** and **/var/log/openvswitch/ovs-vswitchd.log** to determine the issue.

```
$ openstack network agent list

+------------------+-----------------+-----------------+------------------+-------+-------+--------------------+
| ID               | Agent Type      | Host            | Availability Zone | Alive | State | Binary             |
+------------------+-----------------+-----------------+------------------+-------+-------+--------------------+
| 19188fa7-50f1-4a | DHCP agent      | control-0.locald | nova             | True  | UP    | neutron-dhcp-
agent |
| b1-a86c-         |                 | omain           |                  |       |       |                    |
| 986724e6e75d     |                 |                 |                  |       |       |                    |
| 6b58175c-a07e-49 | L3 agent        | control-0.locald | nova             | True  | UP    | neutron-l3-agent
|
| 56-a736-dc2a3f27 |                 | omain           |                  |       |       |                    |
| 2a34             |                 |                 |                  |       |       |                    |
| b4bc9e26-959c-   | Metadata agent  | control-0.locald | None             | True  | UP    | neutron-
metadata-  |
| 402a-ab24-b7ccad |                 | omain           |                  |       | agent |                    |
| b8119f           |                 |                 |                  |       |       |                    |
| eb7df511-5e09-46 | Open vSwitch    | control-0.locald | None             | True  | UP    | neutron-
|
| 55-a82d-         | agent           | omain           |                  |       | openvswitch-agent  |
| 8aa52537f730     |                 |                 |                  |       |       |                    |
| fc1a71f0-06af-   | Open vSwitch    | compute-0.locald | None             | True  | UP    | neutron-           |
| 43e3-b48a-       | agent           | omain           |                  |       | openvswitch-agent  |
| f0923bcec843     |                 |                 |                  |       |       |                    |
+------------------+-----------------+-----------------+------------------+-------+-------+--------------------+
```

2.1.2. Show the Hosts in the Compute Service

Ensure that the value for **Status** is **enabled** and **State** is **up** for each host. If there are any issues, see the logs in **/var/log/containers/nova** to determine the issue.

```
$ openstack compute service list

+----+-----------------+----------------------+----------+---------+-------+----------------------------+
| ID | Binary          | Host                 | Zone     | Status  | State | Updated At                 |
+----+-----------------+----------------------+----------+---------+-------+----------------------------+
|  3 | nova-consoleauth | control-0.localdomain | internal | enabled | up    | 2019-02-
06T16:21:52.000000 |
|  4 | nova-scheduler   | control-0.localdomain | internal | enabled | up    | 2019-02-
06T16:21:51.000000 |
```

```
| 5 | nova-conductor   | control-0.localdomain | internal | enabled | up    | 2019-02-
06T16:21:50.000000 |
| 6 | nova-compute     | compute-0.localdomain | dpdk     | enabled | up    | 2019-02-
06T16:21:45.000000 |
+----+------------------+-----------------------+----------+---------+-------+----------------------------+
```

For more information about confirming a Red Hat OpenStack Platform configuration see Validating a containerized overcloud in the *Upgrading Red Hat OpenStack Platform* guide.

2.2. CONFIRMING COMPUTE NODE OVS CONFIGURATION

To verify the configuration and health of network adapters and OpenvSwitch, complete the following the steps:

1. To verify the DPDK network device on the compute node, run the following command. This rpm is found in repo: **rhel-7-server-extras-rpms**.

   ```
   $ yum install dpdk-tools
   ```

2. Show the network devices managed by DPDK and those used for networking.

   ```
   $ dpdk-devbind --status
   ```

   The devices using a DPDK driver are the types **ovs_dpdk_bond** or **ovs_dpdk_port** in the Tripleo compute role templates:

   ```
   Network devices using DPDK-compatible driver
   ============================================
   0000:04:00.1 'Ethernet 10G 2P X520 Adapter 154d' drv=vfio-pci unused=
   0000:05:00.0 'Ethernet 10G 2P X520 Adapter 154d' drv=vfio-pci unused=

   Network devices using kernel driver
   ===================================
   0000:02:00.0 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em1 drv=tg3 unused=vfio-
   pci *Active*
   0000:02:00.1 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em2 drv=tg3 unused=vfio-
   pci
   0000:03:00.0 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em3 drv=tg3 unused=vfio-
   pci
   0000:03:00.1 'NetXtreme BCM5720 Gigabit Ethernet PCIe 165f' if=em4 drv=tg3 unused=vfio-
   pci *Active*
   0000:04:00.0 'Ethernet 10G 2P X520 Adapter 154d' if=p1p1 drv=ixgbe unused=vfio-pci
   0000:05:00.1 'Ethernet 10G 2P X520 Adapter 154d' if=p2p2 drv=ixgbe unused=vfio-pci
   ```

3. Run the following command to confirm that DPDK is enabled:

   ```
   $ sudo  ovs-vsctl get Open_vSwitch . iface_types

   [dpdk, dpdkr, dpdkvhostuser, dpdkvhostuserclient, geneve, gre, internal, lisp, patch, stt,
   system, tap, vxlan]
   ```

4. Run the following command. The results show PCI devices from the DPDK compatible drivers, for example, **0000:04:00.1** and **:05:00.0** as **type: dpdk** with no errors.

```
$ ovs-vsctl show

Bridge "br-link0"
        Controller "tcp:127.0.0.1:6633"
            is_connected: true
        fail_mode: secure
        Port "phy-br-link0"
            Interface "phy-br-link0"
                type: patch
                options: {peer="int-br-link0"}
        Port "dpdkbond0"
            Interface "dpdk1"
                type: dpdk
                options: {dpdk-devargs="0000:04:00.1", n_rxq="2"}
            Interface "dpdk0"
                type: dpdk
                options: {dpdk-devargs="0000:05:00.0", n_rxq="2"}
        Port "br-link0"
            Interface "br-link0"
                type: internal
    ovs_version: "2.9.0"
```

The following output shows an error:

```
Port "dpdkbond0"
            Interface "dpdk1"
                type: dpdk
                options: {dpdk-devargs="0000:04:00.1", n_rxq="2"}
                error: "Error attaching device '0000:04:00.1' to DPDK"
```

5. To show details about interfaces, run the following command:

```
$ sudo ovs-vsctl list interface dpdk1 | egrep "name|mtu|options|status"
```

6. Run the following command. Note that lacp is not enabled.

```
$ ovs-appctl bond/show dpdkbond0

bond_mode: active-backup
bond may use recirculation:
no, Recirc-ID : -1
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
lacp_status: off
lacp_fallback_ab: false
active slave mac: a0:36:9f:e5:da:82(dpdk1)

slave dpdk0: enabled
    may_enable: true

slave dpdk1: enabled
    active slave
    may_enable: true
```

7. Check that all ovs bridges on compute nodes are **netdev** for fast data path (user space) networking

NOTE

Mixing system (kernel) and netdev (user space) datapath types is not supported.

```
$ ovs-vsctl list bridge | grep -e name -e datapath_type

datapath_type       : netdev
name                : br-int
datapath_type       : netdev
name                : "br-link0"
```

8. Run the following command to check for persistent Open vSwitch errors:

```
$ grep ERROR /var/log/openvswitch/ovs-vswitchd.log
```

2.3. CONFIRMING OVS FOR INSTANCE CONFIGURATION

To ensure that vhostuser DMA works, configure instances with OVS-DPDK ports to have dedicated CPUs and huge pages enabled using flavors. For more information, see Step 3 in: Creating a flavor and deploying an instance for OVS-DPDK.

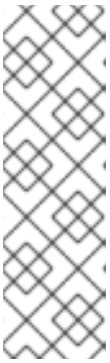
To confirm the instance configuration, complete the following steps:

1. Confirm the instance has pinned CPUs. Dedicated CPUs can be identified with **virsh**:

```
$ sudo virsh vcpupin 2
```

2. Confirm that the emulator threads used for the instance are not running on the same vCPUs assigned to that instance:

```
$ sudo virsh emulatorpin 2
```



NOTE

Beginning with Red Hat OpenStack Platform 12, you can select where the emulator thread will run by flavor. See Configuring emulator threads policy with Red Hat OpenStack Platform 12.

For older versions, you must perform emulator thread pinning manually when the instance is powered on. See About the impact of using virsh emulatorpin in virtual environments with NFV, with and without isolcpus, and about optimal emulator thread pinning.

3. Confirm the instance is using huge pages, which is required for optimal performance.

```
$ sudo virsh numatune 1
```

4. Confirm that the receive queues for the instance are being serviced by a poll mode driver (PMD).

The ports and queues should be equally balanced across the PMDs. Optimally, ports will be serviced by a CPU in the same NUMA node as the network adapter.

```
$ sudo ovs-appctl dpif-netdev/pmd-rxq-show

pmd thread numa_id 0 core_id 2:
   isolated : false
   port: dpdk0              queue-id:  1    pmd usage:  0 %
   port: dpdk1              queue-id:  0    pmd usage:  0 %
   port: vhu94ccc316-ea     queue-id:  0    pmd usage:  0 %
pmd thread numa_id 1 core_id 3:
   isolated : false
pmd thread numa_id 0 core_id 22:
   isolated : false
   port: dpdk0              queue-id:  0    pmd usage:  0 %
   port: dpdk1              queue-id:  1    pmd usage:  0 %
   port: vhu24e6c032-db     queue-id:  0    pmd usage:  0 %
pmd thread numa_id 1 core_id 23:
   isolated : false
```

5. Show statistics for the PMDs. This helps to determine how well receive queues are balanced across PMDs. For more information, see PMD Threads in the Open vSwitch documentation.

NOTE

The **pmd-rxq-rebalance** option was added in OVS 2.9.0. This command performs new PMD queue assignments in order to balance equally across PMDs based on the latest rxq processing cycle information.

The **pmd-stats-show** command shows the full history since the PMDs were running or since the statistics were last cleared. If it is not cleared, it will have incorporated into the stats before the ports were set up and data was flowing. If it is being used to see the load on a datapath (which it typically is) it would then be useless.

It is best to put the system into a steady state, clear the stats, wait a few seconds, and then show the stats. This provides an accurate picture of the datapath.

Use the following command to show statistics for the PMDs:

```
$ sudo ovs-appctl dpif-netdev/pmd-stats-show

pmd thread numa_id 0 core_id 2:
   packets received: 492207
   packet recirculations: 0
   avg. datapath passes per packet: 1.00
   emc hits: 419949
   megaflow hits: 2485
   avg. subtable lookups per megaflow hit: 1.33
   miss with success upcall: 69773
   miss with failed upcall: 0
   avg. packets per output batch: 1.00
   idle cycles: 1867450752126715 (100.00%)
   processing cycles: 5274066849 (0.00%)
   avg cycles per packet: 3794046054.19 (1867456026193564/492207)
   avg processing cycles per packet: 10715.14 (5274066849/492207)
```

```
pmd thread numa_id 1 core_id 3:
    packets received: 0
    packet recirculations: 0
    avg. datapath passes per packet: 0.00
    emc hits: 0
    megaflow hits: 0
    avg. subtable lookups per megaflow hit: 0.00
    miss with success upcall: 0
    miss with failed upcall: 0
    avg. packets per output batch: 0.00
pmd thread numa_id 0 core_id 22:
    packets received: 493258
    packet recirculations: 0
    avg. datapath passes per packet: 1.00
    emc hits: 419755
    megaflow hits: 3223
    avg. subtable lookups per megaflow hit: 1.49
    miss with success upcall: 70279
    miss with failed upcall: 1
    avg. packets per output batch: 1.00
    idle cycles: 1867449561100794 (100.00%)
    processing cycles: 6465180459 (0.00%)
    avg cycles per packet: 3785961963.68 (1867456026281253/493258)
    avg processing cycles per packet: 13107.10 (6465180459/493258)
pmd thread numa_id 1 core_id 23:
    packets received: 0
    packet recirculations: 0
    avg. datapath passes per packet: 0.00
    emc hits: 0
    megaflow hits: 0
    avg. subtable lookups per megaflow hit: 0.00
    miss with success upcall: 0
    miss with failed upcall: 0
    avg. packets per output batch: 0.00
main thread:
    packets received: 16
    packet recirculations: 0
    avg. datapath passes per packet: 1.00
    emc hits: 1
    megaflow hits: 9
    avg. subtable lookups per megaflow hit: 1.00
    miss with success upcall: 6
    miss with failed upcall: 0
    avg. packets per output batch: 1.00
```

6. Reset the PMD statistics. The **pmd-stats-show** command shows the PMD statistics since the last **pmd-stats-clear** command. If there was no previous **pmd-stats-clear** issued, it contains data since the PMD began running.
   If you are examining a system under load, it is useful to clear the PMD statistics and then show them. Otherwise, the statistics can also include data from an earlier time when the system was not under load (before traffic flowing).

   Use the following command to reset the PMD statistics:

   ```
   $ sudo ovs-appctl dpif-netdev/pmd-stats-clear
   ```

2.4. OTHER HELPFUL COMMANDS

Use these commands to perform additional validation checks.

- Find the OVS-DPDK Port & Physical NIC Mapping Configured by os-net-config

```
cat /var/lib/os-net-config/dpdk_mapping.yaml
```

- Find the DPDK port for an instance with the Nova instance $ID

```
sudo  ovs-vsctl find interface external_ids:vm-uuid="$ID" | grep ^name
```

- Find the Nova ID for an instance using a DPDK port

```
sudo  ovs-vsctl get interface vhu24e6c032-db external_ids:vm-uuid
```

- Perform a tcpdump on a dpdk port

```
sudo ovs-tcpdump -i vhu94ccc316-ea
```

NOTE

ovs-tcpdump is from the openvswitch-test RPM located in the **rhel-7-server-openstack-13-devtools-rpms** repo.

NOTE

For performance concerns, **ovs-tcpdump** is not recommended for production environments. For more information, see: How to use ovs-tcpdump on vhost-user interfaces in Red Hat OpenStack Platform?.

2.5. SIMPLE COMPUTE NODE CPU PARTITIONING AND MEMORY CHECKS

Prerequisites

Run this command on a deployed compute node and note how the cpu masks map to TripleO Heat Template values:

```
$ sudo ovs-vsctl get Open_vSwitch . other_config

{dpdk-init="true", dpdk-lcore-mask="300003", dpdk-socket-mem="3072,1024", pmd-cpu-mask="c0000c"}
```

Note the following:

- **dpdk-lcore-mask** maps to **OvsDpdkCoreList** in TripleO Heat Templates.
- **dpdk-socket-mem** maps to **OvsDpdkSocketMemory** in TripleO Heat Templates.
- **pmd-cpu-mask** maps to **OvsPmdCoreList** in TripleO Heat Templates.

To convert these cpu masks to decimal values that can be reconciled back to TripleO Heat Templates and actual system values see: How to convert a hexadecimal CPU mask into a bit mask and identify the masked CPUs?

2.5.1. Detecting CPUs

To detect CPUs for pid 1, use the following command. No PMDs or Nova vCPUs should be running on these cores:

```
$ taskset -c -p 1

pid 1's current affinity list: 0,1,20,21
```

2.5.2. Detecting PMD Threads

To see PMD threads, use the following command. The output should reflect the values of the Tripleo parameter **OvsPmdCoreList**. There should be no overlap with the values of Tripleo parameters **OvsDpdkCoreList** or **HostIsolatedCoreslist**:

```
$ ps -T -o spid,comm -p $(pidof ovs-vswitchd) |grep '\<pmd' |while read spid name; do echo $name
$(taskset -p -c $spid); done

pmd44 pid 679318's current affinity list: 3
pmd45 pid 679319's current affinity list: 23
pmd46 pid 679320's current affinity list: 22
pmd47 pid 679321's current affinity list: 2
```

2.5.3. Detecting NUMA node

For optimal performance ensure that physical network adapters, PMD threads, and pinned CPUs for instances are all on the same NUMA node. For more information, see: CPUs and NUMA nodes.

The following is a simple exercise for examining NUMA assignments.

1. Examine the vhu port for an instance on a compute node:

```
$ sudo virsh domiflist 1

Interface  Type       Source     Model       MAC
-------------------------------------------------------
vhu24e6c032-db vhostuser  -          virtio      fa:16:3e:e3:c4:c2
```

2. Examine the PMD thread that is servicing that port and note the NUMA node:

```
$ sudo ovs-appctl dpif-netdev/pmd-rxq-show

pmd thread numa_id 0 core_id 2:
   isolated : false
   port: vhu24e6c032-db      queue-id:  0    pmd usage:  0 %
   port: vhu94ccc316-ea      queue-id:  0    pmd usage:  0 %
```

3. Find the physical pinned cpus for the instance. For example, the PMD servicing the port for this instance is on cpu 2 and the instance is serviced by cpus 34 and 6.

```
$ sudo virsh dumpxml 1 | grep cpuset

   <vcpupin 1 vcpu='0' cpuset='34'/>
   <emulatorpin cpuset='6'/>
```

4. Examine the cores for each NUMA node. Note that the CPUs servicing the instance (34,6) are on the same NUMA node (0).

```
$ lscpu | grep ^NUMA

NUMA node(s):          2
NUMA node0 CPU(s):     0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38
NUMA node1 CPU(s):     1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,37,39
```

Additionally, network adapters that are not managed by OVS DPDK will have an entry here that indicates what NUMA node they belong to:

```
$ sudo cat /sys/class/net/<device name>/device/numa_node
```

Alternatively, you can see the NUMA node for a network adapter by querying the PCI address, even for those managed by OVS DPDK:

```
$ sudo lspci -v -s 05:00.1 | grep -i numa

Flags: bus master, fast devsel, latency 0, IRQ 203, NUMA node 0
```

These exercises demonstrate that the PMD, instance, and network adapter are all on NUMA 0, which is optimal for performance. For an indication of cross NUMA polling from the openvswitch logs (located in **/var/log/openvswitch**), look for a log entry similar to this:

```
dpif_netdev|WARN|There's no available (non-isolated) pmd thread on numa node 0. Queue 0 on port
'dpdk0' will be assigned to the pmd on core 7 (numa node 1). Expect reduced performance.
```

2.5.4. Detecting Isolated CPUs

Use the following command to show isolated CPUs. The output should be the same as the value of the TripleO parameter **IsolCpusList**.

```
$ cat /etc/tuned/cpu-partitioning-variables.conf | grep -v ^#

isolated_cores=2-19,22-39
```

2.5.5. Detecting CPUs Dedicated to Nova Instances

Use the following command to show the CPUs dedicated to Nova instances. This output should be the same as the value of the parameter isolcpus without poll mode driver (PMD) CPUs:

```
$ grep ^vcpu_pin_set /var/lib/config-data/puppet-generated/nova_libvirt/etc/nova/nova.conf

vcpu_pin_set=4-19,24-39
```

2.5.6. Confirming Huge Pages Configuration

Check for huge pages configuration on the compute node.

```
[root@compute-0 ~]# cat /sys/devices/system/node/node*/meminfo  | grep -i huge
Node 0 AnonHugePages:      4096 kB
Node 0 HugePages_Total:    16
Node 0 HugePages_Free:     11
Node 0 HugePages_Surp:      0
Node 1 AnonHugePages:      8192 kB
Node 1 HugePages_Total:    16
Node 1 HugePages_Free:     15
Node 1 HugePages_Surp:      0
```

If huge pages are not configured or are exhausted, see KernelArgs.

2.6. CAUSES FOR PACKET DROPS

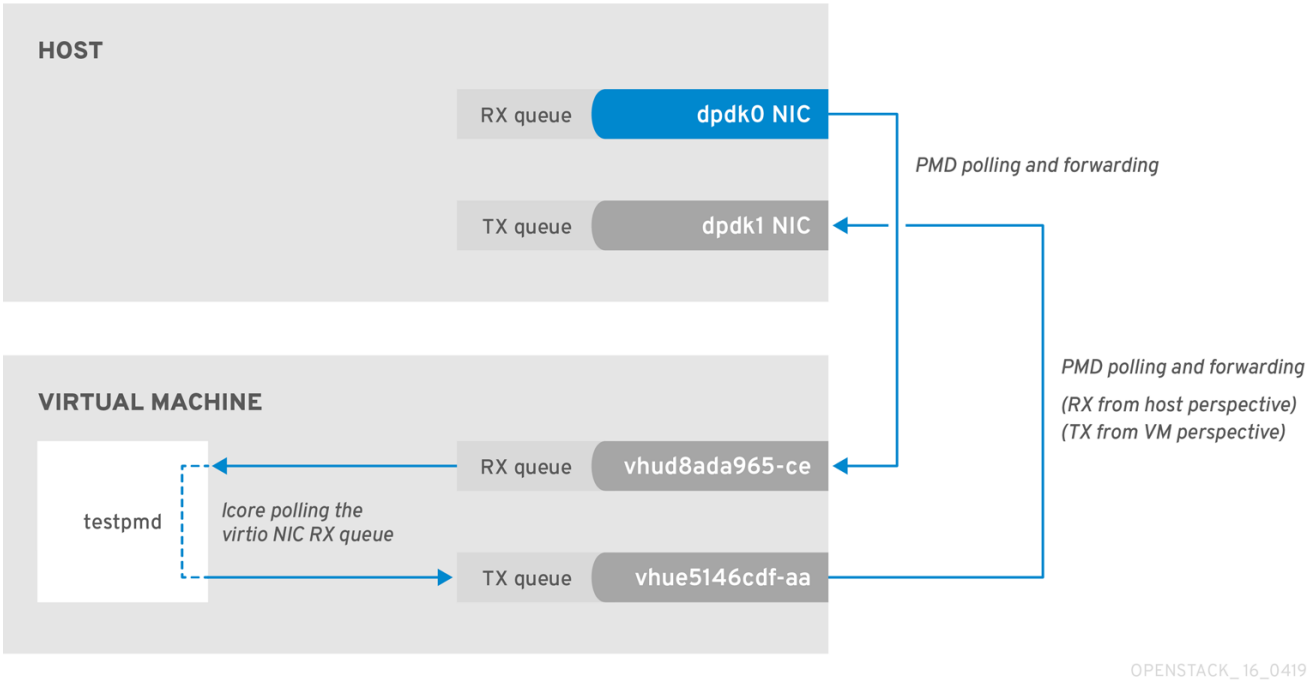
Packets are dropped when a queue is full, usually when the queue is not drained fast enough. The bottleneck is the entity that is supposed to drain the queue when the queue is not draining quickly enough. In most instances, a drop counter is used to track dropped packets. Sometimes a bug in the hardware or software design can cause packets to skip the drop counter.

The Data Plan Development Kit (DPDK) includes the **testpmd** application for forwarding packets. In the scenarios shown in this chapter, **testpmd** is installed on a VM and polls ports with its assigned logical cores (lcores) to forward packets from one port to another. **testpmd** is ordinarily used with a traffic generator to test, in this case, throughput across a physical-virtual-physical (PVP) path.

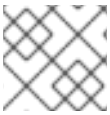
2.6.1. OVS-DPDK Too Slow to Drain Physical NICs

This example shows that a PMD thread is responsible for polling the receive (RX) queue of the physical network adapter (dpdk0). When the PMD thread cannot keep up with the packet volume, or is interrupted, packets might be dropped.

Figure 2.1. Polling the physical adapter RX queue



The following command shows statistics from the dpdk0 interface. If packets are being dropped because ovs-dpdk is not draining the physical adapter fast enough, you will see the value of **rx_dropped** increasing rapidly.



NOTE

There should be no more than one physical CPU core per NUMA node for PMDs.

```
# ovs-vsctl --column statistics list interface dpdk0

statistics          : {mac_local_errors=0, mac_remote_errors=0, "rx_1024_to_1522_packets"=26,
"rx_128_to_255_packets"=243,
"rx_1523_to_max_packets"=0, "rx_1_to_64_packets"=102602, "rx_256_to_511_packets"=6100,
"rx_512_to_1023_packets"=27,
"rx_65_to_127_packets"=16488, rx_broadcast_packets=2751, rx_bytes=7718218, rx_crc_errors=0,
rx_dropped=0, rx_errors=0,
rx_fragmented_errors=0, rx_illegal_byte_errors=0, rx_jabber_errors=0, rx_length_errors=0,
rx_mac_short_dropped=0,
rx_mbuf_allocation_errors=0, rx_oversize_errors=0, rx_packets=125486, rx_undersized_errors=0,
"tx_1024_to_1522_packets"=63,
"tx_128_to_255_packets"=319, "tx_1523_to_max_packets"=0, "tx_1_to_64_packets"=1053,
"tx_256_to_511_packets"=50,
"tx_512_to_1023_packets"=68, "tx_65_to_127_packets"=7732, tx_broadcast_packets=12,
tx_bytes=466813, tx_dropped=0,
tx_errors=0, tx_link_down_dropped=0, tx_multicast_packets=5642, tx_packets=9285}
```

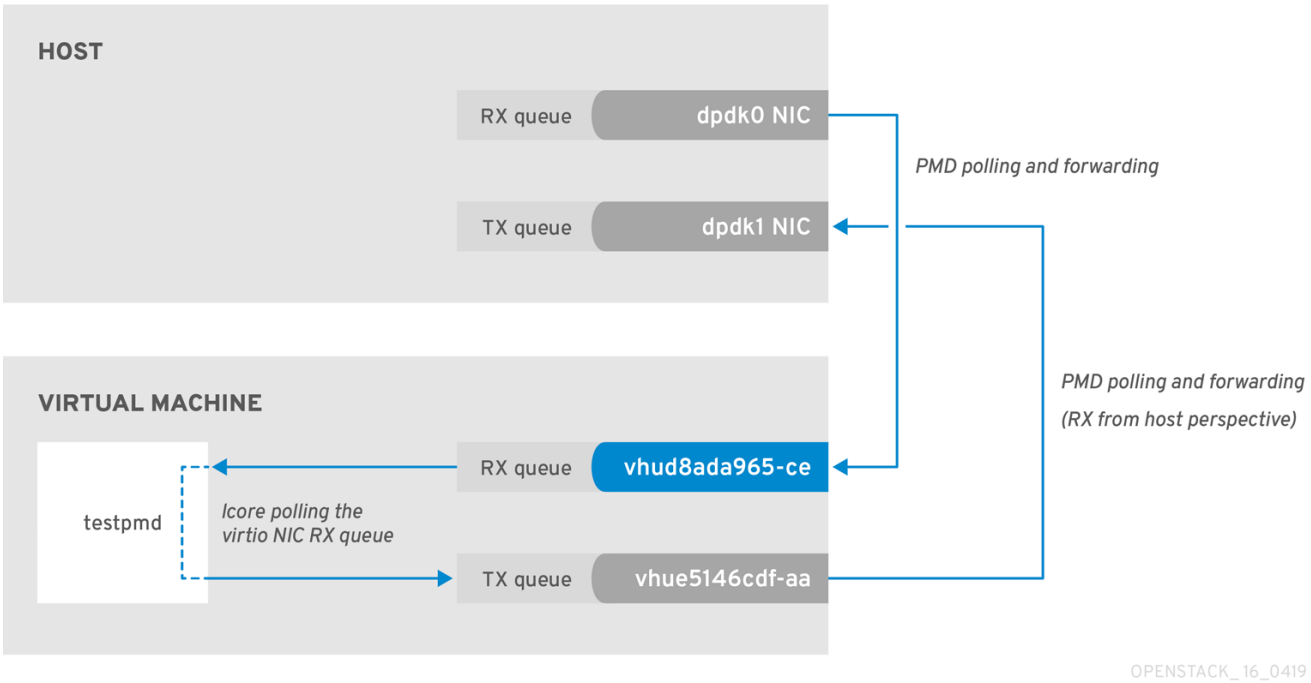
2.6.2. VM Too Slow to Drain vhost-user

This example is similar to the example in Figure 2.1, in that you might experience packet loss if the lcore thread is overwhelmed by the packet volume sent to the instance receive (RX) queue.

For more information, see the following articles:

- About the impact of using virsh emulatorpin in virtual environments with NFV, with and without isolcpus, and about optimal emulator thread pinning
- Change RX queue size and TX queue size of virtio NICs that are connected to OVS DPDK with Red Hat OpenStack Director

Figure 2.2. Polling the virtual adapter RX queue



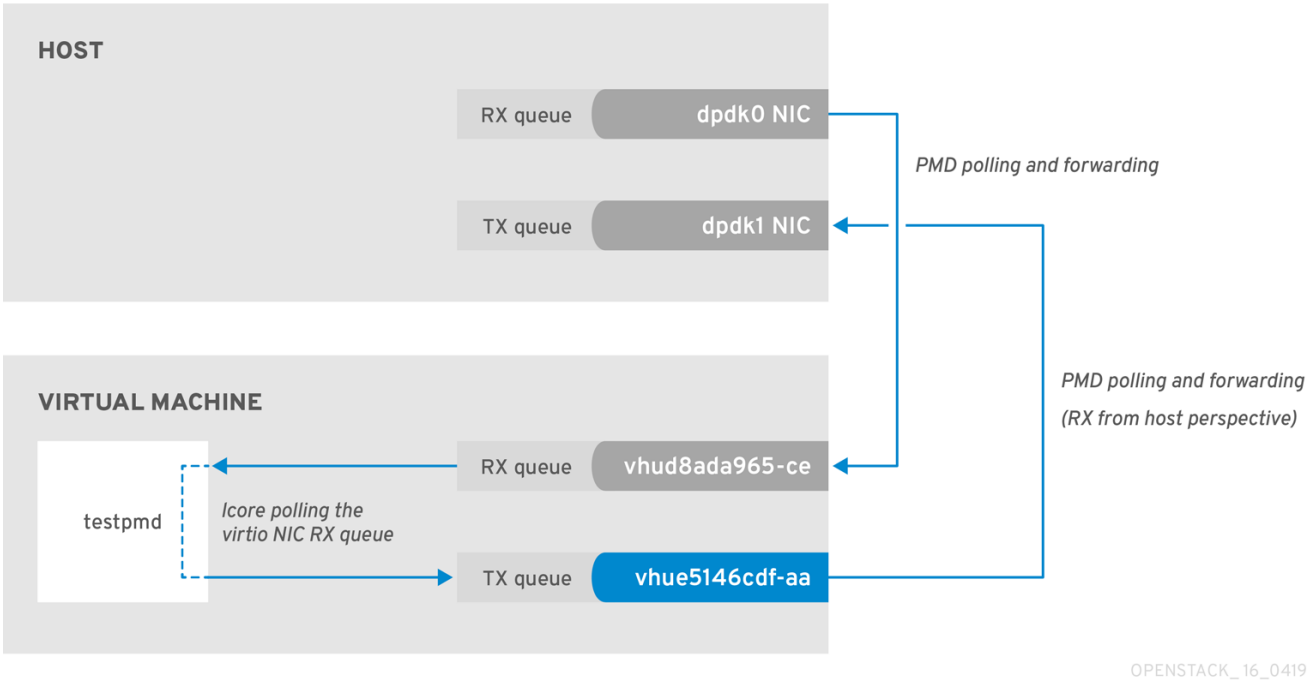
To check if the **tx_dropped** value of the host corresponds to the **rx_dropped** value of the VM, run the following command:

```
ovs-vsctl --column statistics list interface vhud8ada965-ce

statistics          : {"rx_1024_to_1522_packets"=0, "rx_128_to_255_packets"=0,
"rx_1523_to_max_packets"=0,
"rx_1_to_64_packets"=0, "rx_256_to_511_packets"=0, "rx_512_to_1023_packets"=0,
"rx_65_to_127_packets"=0, rx_bytes=0,
rx_dropped=0, rx_errors=0, rx_packets=0, tx_bytes=0, tx_dropped=0, tx_packets=0}
```

2.6.3. OVS-DPDK Too Slow to Drain vhost-user

In this example, a PMD thread is polls the virtio TX, the receive queue from the host perspective. If the PMD thread is overwhelmed by the packet volume, or is interrupted, packets might drop.

Figure 2.3. Polling the virtual adapter TX queue



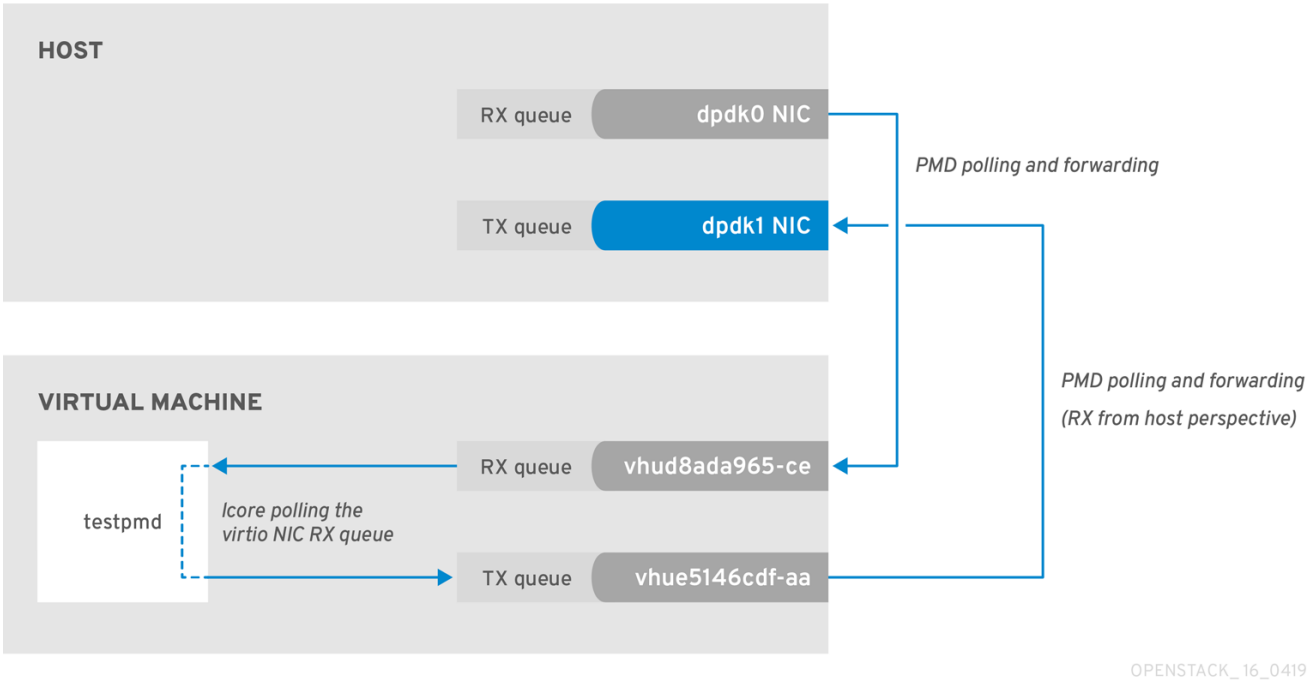
The trace the return path of the packets from the VM and provides values from drop counters on both the host (**tx_dropped**) and VM (**rx_dropped**) sides, run the following command:

```
ovs-vsctl --column statistics list interface vhue5146cdf-aa

statistics          : {"rx_1024_to_1522_packets"=0, "rx_128_to_255_packets"=0,
"rx_1523_to_max_packets"=0,
"rx_1_to_64_packets"=0, "rx_256_to_511_packets"=0, "rx_512_to_1023_packets"=0,
"rx_65_to_127_packets"=0,
rx_bytes=0, rx_dropped=0, rx_errors=0, rx_packets=0, tx_bytes=0, tx_dropped=0, tx_packets=0}
```

2.6.4. Packet Loss on Egress Physical Interface

A slow transfer rate between the PCIe and RAM can result in the physical adapter dropping packets from the TX queue. While this is infrequent, it's important to know how to identify and resolve this issue.

Figure 2.4. Polling the physical adapter TX queue



The following command shows statistics from the dpdk1 interface. If **tx_dropped** is greater than zero and growing rapidly, open a support case with Red Hat.

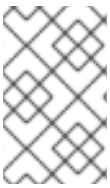
```
ovs-vsctl --column statistics list interface dpdk1

statistics          : {mac_local_errors=0, mac_remote_errors=0, "rx_1024_to_1522_packets"=26,
"rx_128_to_255_packets"=243, "rx_1523_to_max_packets"=0, "rx_1_to_64_packets"=102602,
"rx_256_to_511_packets"=6100,
"rx_512_to_1023_packets"=27, "rx_65_to_127_packets"=16488, rx_broadcast_packets=2751,
rx_bytes=7718218,
rx_crc_errors=0, rx_dropped=0, rx_errors=0, rx_fragmented_errors=0, rx_illegal_byte_errors=0,
rx_jabber_errors=0,
rx_length_errors=0, rx_mac_short_dropped=0, rx_mbuf_allocation_errors=0, rx_oversize_errors=0,
rx_packets=125486,
rx_undersized_errors=0, "tx_1024_to_1522_packets"=63, "tx_128_to_255_packets"=319,
"tx_1523_to_max_packets"=0,
"tx_1_to_64_packets"=1053, "tx_256_to_511_packets"=50, "tx_512_to_1023_packets"=68,
"tx_65_to_127_packets"=7732,
tx_broadcast_packets=12, tx_bytes=466813, tx_dropped=0, tx_errors=0, tx_link_down_dropped=0,
tx_multicast_packets=5642, tx_packets=9285}
```

If you see these types of packet losses, consider reconfiguring the memory channels.

- To calculate memory channels, see: Memory parameters in the Network Functions Virtualization Planning and Conifguration Guide.
- To determine the number of memory channels, see: How to determine the number of memory channels for NeutronDpdkMemoryChannels or OvsDpdkMemoryChannels in Red Hat OpenStack Platform.

CHAPTER 3. NFV COMMAND CHEATSHEET

This chapter contains many of the most commonly used commands for Red Hat OpenStack Platform 13 system observability.



NOTE

Some of the commands below may not be available by default. To install the required tools for a given node, run the following command:
sudo yum install tuna qemu-kvm-tools perf kernel-tools dmidecode

3.1. UNIX SOCKETS

Use these commands to show process ports and UNIX socket domains.

Action	Command
Show all TCP and UDP SOCKETS in all states (LISTEN, ESTABLISHED, CLOSE_WAIT, etc) without hostname lookup	**# lsof -ni**
Show all TCP SOCKETS in all states (LISTEN, ESTABLISHED, CLOSE_WAIT, etc) without hostname lookup	**# lsof -nit**
Show all UDP SOCKETS in all states (LISTEN, ESTABLISHED, CLOSE_WAIT, etc) without hostname lookup	**# lsof -niu**
Show all TCP and UDP SOCKETS in all states (LISTEN, ESTABLISHED, CLOSE_WAIT, etc) without hostname lookup for IPv4	**# lsof -ni4**
Show all TCP and UDP SOCKETS in all states (LISTEN, ESTABLISHED, CLOSE_WAIT, etc) without hostname lookup for IPv6	**# lsof -ni6**
Show all related SOCKETS (LISTEN, ESTABLISHED, CLOSE_WAIT, etc) without hostname lookup for a given port	**# lsof -ni :4789**
Show all SOCKETS in LISTEN state without hostname lookup	**# ss -ln**
Show all SOCKETS in LISTEN state without hostname lookup for IPv4	**# ss -ln4**
Show all SOCKETS in LISTEN state without hostname lookup for IPv6	**# ss -ln6**

3.2. IP

Use these commands to show IP L2 and L3 configs, drivers, PCI busses, and network statistics.

Action	Command
Show all L2 (both physical and virtual) interfaces and their statistics	**# ip -s link show**
Show all L3 interfaces and their statistics	**# ip -s addr show**
Show default (main) IP routing table	**# ip route show**
Show routing rules of a given routing table	**# ip route show table external**
Show all routing tables	**# ip rule show**
Show routing rules for a given destination	**# ip route get 1.1.1.1**
Show all Linux namespaces	**# ip netns show**
Log in into a Linux namespace	**# ip netns exec ns0 bash**
Show detailed network interface counters of a given interface	**# tail /sys/class/net/ens6/statistics/***
Show detailed bonding information of a given bond device	**# cat /proc/net/bonding/bond1**
Show global network interface counter view	**# cat /proc/net/dev**
Show physical connection type (TP, FIBER etc), link speed mode supported and connected for a given network interface	**# ethtool ens6**
Show Linux driver, driver version, firmware, and PCIe BUS ID of a given network interface	**# ethtool -i ens6**
Show default, enabled, and disabled hardware offloads for a given network interface	**# ethtool -k ens6**
Show MQ (multiqueue) configuration for a given network interface	**# ethtool -l ens6**
Change MQ setup for both RX and TX for a given network interface	**# ethtool -L ens6 combined 8**
Change MQ setup only for TX for a given network interface	**# ethtool -L ens6 tx 8**

Action	Command
Show queue size for a given network interface	**# ethtool -g ens6**
Change RX queue size for a given network interface	**# ethtool -G ens6 rx 4096**
Show enhanced network statistics	**# cat /proc/net/softnet_stat**
Show quick important network device info (Interface name, MAC, NUMA, PCIe slot, firmware, kernel driver)	**# biosdevname -d**
Show kernel internal drop counters. For more information, see: Monitoring network data processing.	**# cat /proc/net/softnet_stat**

3.3. OVS

Use these commands to show Open vSwitch related information.

Action	Command
OVS DPDK human readable statistics	See Open vSwitch DPDK Statistics.
Show OVS basic info (version, dpdk enabled, PMD cores, lcore, ODL bridge mapping, balancing, auto-balancing etc)	**# ovs-vsctl list Open_vSwitch**
Show OVS global switching view	**# ovs-vsctl show**
Show OVS all detailed interfaces	**# ovs-vsctl list interface**
Show OVS details for one interface (link speed, MAC, status, stats, etc)	**# ovs-vsctl list interface dpdk0**
Show OVS counters for a given interface	**# ovs-vsctl get interface dpdk0 statistics**
Show OVS all detailed ports	**# ovs-vsctl list port**
Show OVS details for one port (link speed, MAC, status, stats, etc)	**# ovs-vsctl list port vhu3gf0442-00**
Show OVS details for one bridge (datapath type, multicast snooping, stp status etc)	**# ovs-vsctl list bridge br-int**
Show OVS log status	**# ovs-appctl vlog/list**

Action	Command
Change all OVS log to debug	**# ovs-appctl vlog/set dbg**
Change one specific OVS subsystem to debug mode for the file log output	**# ovs-appctl vlog/set file:backtrace:dbg**
Disable all OVS logs	**# ovs-appctl vlog/set off**
Change all OVS subsystems to debug for file log output only	**# ovs-appctl vlog/set file:dbg**
Show all OVS advanced commands	**# ovs-appctl list-commands**
Show all OVS bonds	**# ovs-appctl bond/list**
Show details about a specific OVS bond (status, bond mode, forwarding mode, LACP status, bond members, bond member status, link status)	**# ovs-appctl bond/show bond1**
Show advanced LACP information for members, bond and partner switch	**# ovs-appctl lacp/show**
Show OVS interface counters	**# ovs-appctl dpctl/show -s**
Show OVS interface counters highlighting differences between iterations	**# watch -d -n1 "ovs-appctl dpctl/show -s|grep -A4 -E '(dpdk|dpdkvhostuser)'|grep -v '\-\-'"**
Show OVS mempool info for a given port	**# ovs-appctl netdev-dpdk/get-mempool-info dpdk0**
Show PMD performance statistics	**# ovs-appctl dpif-netdev/pmd-stats-show**
Show PMD performance statistics in a consistent way	**# ovs-appctl dpif-netdev/pmd-stats-clear && sleep 60s && ovs-appctl dpif-netdev/pmd-stats-show**
Show DPDK interface statistics human readable	**# ovs-vsctl get interface dpdk0 statistics|sed -e "s/,/\n/g" -e "s/[\",\{,\},]//g" -e "s/=/ ==⇒ /g"**
Show OVS mapping between ports/queue and PMD threads	**# ovs-appctl dpif-netdev/pmd-rxq-show**
Trigger OVS PMD rebalance (based on PMD cycles utilization)	**# ovs-appctl dpif-netdev/pmd-rxq-rebalance**
Create affinity between an OVS port and a specific PMD (disabling the PMD from any balancing)	**# ovs-vsctl set interface dpdk other_config:pmd-rxq-affinity="0:2,1:4"**

Action	Command
(OVS 2.11+ and FDP18.09) Set PMD balancing based on cycles	**# ovs-vsctl set Open_vSwitch . other_config:pmd-rxq-assign=cycles**
(OVS 2.11+ and FDP18.09) Set PMD balancing in round robin	**# ovs-vsctl set Open_vSwitch . other_config:pmd-rxq-assign=roundrobin**
Set number of OVS-DPDK Physical ports queues	**# ovs-vsctl set interface dpdk options:n_rxq=2**
Set number of OVS-DPDK Physical ports queue sizes	**# ovs-vsctl set Interface dpdk0 options:n_rxq_desc=4096** **# ovs-vsctl set Interface dpdk0 options:n_txq_desc=4096**
Show OVS MAC address table (used for action=normal)	**# ovs-appctl fdb/show br-provider**
Set OVS vSwitch MAC Address table aging time (default 300s)	**# ovs-vsctl set bridge br-provider other_config:mac-aging-time=900**
Set OVS vSwitch MAC Address table size (default 2048s)	**# ovs-vsctl set bridge br-provider other_config:mac-table-size=204800**
Show OVS datapath flows (kernel space)	**# ovs-dpctl dump-flows -m**
Show OVS datapath flows (dpdk)	**# ovs-appctl dpif/dump-flows -m br-provider**
Show mapping between datapath flows port number and port name	**# ovs-dpctl show**
Show OVS OpenFlow rules in a given bridge	**# ovs-ofctl dump-flows br-provider**
Show mapping between OpenFlow flows port number and port name	**# ovs-ofctl show br-provider**
(OVS 2.11+) - Enable auto-rebalance	**# ovs-vsctl set Open_vSwitch . other_config:pmd-auto-lb="true"**
(OVS 2.11+) - Change auto-rebalance interval to a different value (default 1 minute)	**# ovs-vsctl set Open_vSwitch . other_config:pmd-auto-lb-rebalance-intvl="5"**
Detailed OVS internal configs	**# man ovs-vswitchd.conf.db**
To download OVS tcpdump	**# curl -O -L** ovs-tcpdump.in

Action	Command
To perform a packet capture from a DPDK interface	**# ovs-tcpdump.py --db-sock unix:/var/run/openvswitch/db.sock -i <bond/vhu> <tcpdump standard arguments such as -v -nn -e -w <path/to/file>>**
(OVS 2.10+) Detailed PMD performance stats	**# ovs-appctl dpif-netdev/pmd-perf-show**

3.4. IRQ

Use these commands to show Interrupt Request Line (IRQ) software and hardware interrupts.

Action	Command
Show SoftIRQ balancing per CPU executed by the ksoftirqd workers	**# cat /proc/softirqs | less -S**
Show SoftIRQ balancing per CPU executed by the ksoftirqd workers every second	**# watch -n1 -d -t "cat /proc/softirqs"**
Show hardware and software interrupts (NMI, LOC, TLB, RSE, PIN, NPI, PIW) balancing per CPU	**# cat /proc/interrupts | less -S**
Show hardware and software interrupts (NMI, LOC, TLB, RSE, PIN, NPI, PIW) balancing per CPU every second	**# watch -n1 -d -t "cat /proc/interrupts"**
Show Timer interrupts	**# cat /proc/interrupts | grep -E "LOC|CPU" | less -S**
Show Timer interrupts every second	**# watch -n1 -d -t "cat /proc/interrupts | grep -E 'LOC|CPU'"**
Show default IRQ CPU affinity	**# cat /proc/irq/default_smp_affinity**
Show IRQ affinity for a given IRQ (CPUMask)	**# cat /proc/irq/89/smp_affinity**
Show IRQ affinity for a given IRQ (DEC)	**# cat /proc/irq/89/smp_affinity_list**
Set IRQ affinity for a given IRQ (CPUMask)	**# echo -n 1000 > /proc/irq/89/smp_affinity**
Set IRQ affinity for a given IRQ (DEC)	**# echo -n 12 > /proc/irq/89/smp_affinity_list**
Show hardware interrupts CPU affinity	**# tuna --show_irqs**
Set IRQ affinity for a given IRQ (DEC supporting rage, e.g. 0-4 means from 0 to 4)	**# tuna --irqs=<IRQ> --cpus=<CPU> --move**

Action	Command
Show IRQ CPU utilization distribution	**# mpstat -I CPU | less -S**
Show IRQ CPU utilization distribution for a given CPU	**# mpstat -I CPU -P 4 | less -S**
Show SoftIRQ CPU utilization distribution	**# mpstat -I SCPU | less -S**
Show SoftIRQ CPU utilization distribution for a given CPU	**# mpstat -I SCPU -P 4 | less -S**

3.5. PROCESSES

Use these commands to show processes and threads in Linux, Process Scheduler, and CPU Affinity.

Action	Command
Show for a given process name distribution CPU usage and CPU affinity including all process threads	**# pidstat -p $(pidof qemu-kvm) -t**
Show for a given process name distribution CPU usage and CPU affinity including all process threads, every 10 seconds for 30 iterations	**# pidstat -p $(pidof qemu-kvm) -t 10 30**
Show for a given process name page faults and memory utilization including all process threads	**# pidstat -p $(pidof qemu-kvm) -t -r**
Show for a given process name I/O statistics including all process threads	**# pidstat -p $(pidof qemu-kvm) -t -d**
Show for a given process name its PID, all the child PID(s) including the process name, and the CPU Time	**# ps -T -C qemu-kvm**
Show for a given process and all the child PID(s) real-time performance statistics	**# top -H -p $(pidof qemu-kvm)**
Show all system threads with process scheduler type, priority, command, CPU Affinity, and Context Switching information	**# tuna --show_threads**
Set for a given PID RealTime (FIFO) scheduling with highest priority	**# tuna --threads=<PID> --priority=FIFO:99**
Show PMD and CPU threads rescheduling activities	**# watch -n1 -d "grep -E 'pmd|CPU' /proc/sched_debug"**
Browser scheduler internal operation statistics	**# less /proc/sched_debug**

Action	Command
Show comprehensive process statistics and affinity view: 1. Open top and then press "zbEEH". 2. Press "f" and look for "P = Last Used Cpu (SMP)". 3. Select it using "arrow right". 4. Move it up before CPU Usage using "arrow up". 5. De-select it using "arrow left". 6. Enable it using "d". 7. Sort by CPU number using "<".	**# top**
Show all system processes and their CPU affinity	**# ps -eF**
Show all system processes displaying sleeping and running processes and, when sleeping, at which function	**# ps -elfL**
Show CPU Affinity for a given PID	**# taskset --pid $(pidof qemu-kvm)**
Set a CPU Affinity for a given PID	**# taskset --pid --cpu-list 0-9,20-29 $(pidof <Process>)**

3.6. KVM

Use these commands to show Kernel-based Virtual Machine (KVM) related domain statistics.

Action	Command
Show real-time KVM hypervisor statistics (VMExit, VMEntry, vCPU wakeup, context switching, timer, Halt Pool, vIRQ)	**# kvm_stat**
Show deep KVM hypervisor statistics	**# kvm_stat --once**
Show real-time KVM hypervisor statistics for a given guest (VMExit, VMEntry, vCPU wakeup, context switching, timer, Halt Pool, vIRQ)	**# kvm_stat --guest=<VM name>**
Show deep KVM hypervisor statistics for a given guest	**# kvm_stat --once --guest=<VM name>**

Action	Command
Show KVM profiling trap statistics	**# perf kvm stat live**
Show KVM profiling statistics	**# perf kvm top**
Show vCPU Pinning for a given VM	**# virsh vcpupin <Domain name/ID>**
Show QEMU Emulator Thread for a given VM	**# virsh emulatorpin <Domain name/ID>**
Show NUMA Pinning for a given VM	**# virsh numatune <Domain name/ID>**
Show memory statistics for a given VM	**# virsh dommemstat <Domain name/ID>**
Show vCPU statistics for a given VM	**# virsh nodecpustats <Domain name/ID>**
Show all vNIC for a given VM	**# virsh domiflist <Domain name/ID>**
Show vNIC statistics for a given VM (does not work with DPDK VHU)	**# virsh domifstat <Domain name/ID> <vNIC>**
Show all vDisk for a given VM	**# virsh domblklist <Domain name/ID>**
Show vDisk statistics for a given VM	**# virsh domblkstat <Domain name/ID> <vDisk>**
Show all statistics for a given VM	**# virsh domstats <Domain name/ID>**

3.7. CPU

Use these commands to show CPU utilization, process CPU distribution, frequency, and SMI.

Action	Command
Show for a given process name distribution CPU usage and CPU affinity including all process threads	**# pidstat -p $(pidof qemu-kvm) -t**
Show virtual memory, I/O, and CPU statistics	**# vmstat 1**
Show detailed CPU usage aggregated	**# mpstat**
Show detailed CPU usage distribution	**# mpstat -P ALL**
Show detailed CPU usage distribution for a given CPU (it does not support a range)	**# mpstat -P 2,3,4,5**
Show detailed CPU usage distribution for a given CPU every 10 seconds for 30 iteration	**# mpstat -P 2,3,4,5 10 30**

Action	Command
Show hardware limits and frequency policy for a given CPU frequency	**# cpupower -c 24 frequency-info**
Show current CPU frequency information	**# cpupower -c all frequency-info|grep -E "current CPU frequency|analyzing CPU"**
Show frequency and CPU % C-States stats for all CPU(s)	**# cpupower monitor**
Show real-time frequency and CPU % C-States stats for all CPUs highlighting any variation	**# watch -n1 -d "cpupower monitor"**
Show more detailed frequency and CPU % C-States stats for all CPU including SMI (useful for RT)	**# turbostat --interval 1**
Show more detailed frequency and CPU % C-States stats for a given CPU including SMI (useful for RT)	**# turbostat --interval 1 --cpu 4**
Show CPU details and ISA supported	**# lscpu**
Specific for Intel CPU: Display very low-level details about CPU Usage, CPU IPC, CPU Execution in %, L3 and L2 Cache Hit, Miss, Miss per instruction, Temperature, Memory channel usage, and QPI/UPI Usage	**git clone Processor Counter Monitor make ./pcm.x"**

3.8. NUMA

Use these commands to show Non-Uniform Memory Access (NUMA) statistics and process distribution.

Action	Command
Show hardware NUMA topology	**# numactl -H**
Show NUMA statistics	**# numastat -n**
Show meminfo like system-wide memory usage	**# numastat -m**
Show NUMA memory details and balancing for a given process name	**# numastat qemu-kvm**

Action	Command
Show for a given NUMA node specific statistics	**# /sys/devices/system/node/node<NUMA node number>/numastat**
Show in a very clear why NUMA topology with NUMA nodes and PCI devices	**# lstopo --physical**
Generate an graph (svg format) of the physical NUMA topology with related devices	**# lstopo --physical --output-format svg > topology.svg**

3.9. MEMORY

Use these commands to show memory statistics, huge pages, DPC, physical DIMM, and frequency.

Action	Command
Show meminfo like system-wide memory usage	**# numastat -m**
Show virtual memory, I/O, and CPU statistics	**# vmstat 1**
Show global memory information	**# cat /proc/meminfo**
Show the total number of 2MB huge pages for a given NUMA node	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-2048kB/nr_hugepages**
Show the total number of 1GB huge pages for a given NUMA node	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-1048576kB/nr_hugepages**
Show the total free 2MB huge pages for a given NUMA node	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-2048kB/free_hugepages**
Show the total free 1GB huge pages for a given NUMA node	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-1048576kB/free_hugepages**
Allocate 100x 2MB huge pages in real-time to NUMA0 (NUMA node can be changed)	**# echo 100 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages**
Allocate 100x 1GB huge pages in real-time to NUMA0 (NUMA node can be changed)	**# echo 100 > /sys/devices/system/node/node0/hugepages/hugepages-1048576kB/nr_hugepages**
Show real-time SLAB information	**# slabtop**

Action	Command
Show detailed SLAB information	**# cat /proc/slabinfo**
Show total installed memory DIMM	**# dmidecode -t memory | grep Locator**
Show installed memory DIMM Speed	**# dmidecode -t memory | grep Speed**

3.10. PCI

Use these commands to show PCI statistics, PCI details, and PCI driver override.

Action	Command
Show detailed PCI device information in system	**# lspci -vvvnn**
Show PCI tree view	**# lspci -vnnt**
Show PCI device NUMA information	**# lspci -vmm**
Show PCIe max link speed for a given device	**# lspci -s 81:00.0 -vv | grep LnkCap**
Show PCIe link speed status for a given device	**# lspci -s 81:00.0 -vv | grep LnkSta**
Show PCI device and kernel driver	**# driverctl list-devices**
Show PCI device driver override (typical for DPDK and SR-IOV interfaces)	**# driverctl list-overrides**
Set different kernel driver for PCI device (reboot persistent)	**# driverctl set-override 0000:81:00.0 vfio-pci**
Unset overridden kernel driver for PCI device (if device is in use the command will hang)	**# driverctl unset-override 0000:81:00.0**

3.11. TUNED

Use these commands to show tuned profiles, verification, and logs.

Action	Command
Show tuned current enabled profile and description	**# tuned-adm profile_info**
Show tuned available profiles and current enabled profiles	**# tuned-adm list**

Action	Command
Enabled a specific tuned profile	**# tuned-adm profile realtime-virtual-host**
Verify current enabled profile	**# tuned-adm verify**
Tuned's log	**# less /var/log/tuned/tuned.log**

3.12. PROFILING PROCESS

Use these commands to show CPU profiling, process profiling, and KVM profiling.

Section	Action	Command
Process	Profiling on specific PID	**# perf record -F 99 -p PID**
Process	Profiling on specific PID for 30 seconds	**# perf record -F 99 -p PID sleep 30**
Process	Profiling real-time on specific PID	**# perf top -F 99 -p PID**
CPU	Profiling on specific CPU Core list for 30 seconds for any events	**# perf record -F 99 -g -C <CPU Core(s)> – sleep 30s**
CPU	Profiling real-time on specific CPU Core list for any events	**# perf top -F 99 -g -C <CPU Core(s)>**
Context Switching	Profiling on specific CPU Core list for 30 seconds and looking only for Context Switching	**# perf record -F 99 -g -e sched:sched_switch -C <CPU Core(s)> – sleep 30**
KVM	Profiling KVM guest for a given time	**# perf kvm stat record sleep 30s**
Cache	Profiling on specific CPU Core list for 5 seconds looking for the cache efficiency	**# perf stat -C <CPU Core(s)> -B -e cache-references,cache-misses,cycles,instructions,branches,faults,migrations sleep 5**
Report	Analyze perf profiling	**# perf report**
Report	Report perf profiling in stdout	**# perf report --stdio**
Report	Report KVM profiling in stdout	**# perf kvm stat report**

3.13. BLOCK I/O

Use these commands to show storage I/O distribution and I/O profiling.

Action	Command
Show I/O details for all system device	**# iostat**
Show advanced I/O details for all system device	**# iostat -x**
Show advanced I/O details for all system device every 10 seconds for 30 iterations	**# iostat -x 10 30**
Generate advanced I/O profiling for a given block device	**# blktrace -d /dev/sda -w 10 && blkparse -i sda.* -d sda.bin**
Report blktrace profiling	**# btt -i sda.bin**

3.14. REAL TIME

Use these commands to show Real Time tests related, SMI, and latency.

Action	Command
Identify if any SMI are blocking the normal RT kernel execution exercising the defined threshold.	**# hwlatdetect --duration=3600 --threshold=25**

Action	Command
Verify maximum scheduling latency for a given time with a number of additional options: **--duration** Specify a time value for the test run. **--mlockall** Lock current and future memory allocations. **--priority** Set the priority of the first thread. **--nanosleep** Use clock_nanosleep instead of posix interval timers. **--interval** Set the base interval of the thread(s) in microseconds. **--histogram** Dump latency histogram to stdout after the run. **--histfile** Dump the latency histogram to **<path>** instead of **stdout**. **--threads** Set the number of test threads. **--numa** Standard NUMA testing. **--notrace** Suppress tracing.	**# cyclictest --duration=3600 ** **--mlockall ** **--priority=99 ** **--nanosleep ** **--interval=200 ** **--histogram=5000 ** **--histfile=./output ** **--threads ** **--numa ** **--notrace**

3.15. SECURITY

Use these commands to verify speculative executions and the GRUB boot parameter.

Action	Command
Check all current Speculative execution security status	See: Spectre & Meltdown vulnerability/mitigation checker for Linux & BSD.
GRUB parameter to disable all Speculative Execution remediation	**spectre_v2=off spec_store_bypass_disable=off pti=off l1tf=off kvm-intel.vmentry_l1d_flush=never**
Verify CVE-2017-5753 (Spectre variant 1) status	**# cat /sys/devices/system/cpu/vulnerabilities/spectre_v1**
Verify IBPB and Retpoline (CVE-2017-5715 Spectre variant 2) status	**# cat /sys/devices/system/cpu/vulnerabilities/spectre_v2**

Action	Command
Verify KPTI (CVE-2017-5754 Meltdown) status	**# cat /sys/devices/system/cpu/vulnerabilities/meltdown**
Verify Spectre-NG (CVE-2018-3639 Spectre Variant 4) status	**# cat /sys/devices/system/cpu/vulnerabilities/spec_store_bypass**
Verify Foreshadow (CVE-2018-3615 Spectre Varian 5 also known as L1TF) status	**# cat /sys/devices/system/cpu/vulnerabilities/l1tf**
Verify Foreshadow VMEntry L1 cache effect	**# cat /sys/module/kvm_intel/parameters/vmentry_l1d_flush**
Verify SMT status	**# cat /sys/devices/system/cpu/smt/control**

3.16. JUNIPER CONTRAIL VROUTER

Use these commands to show vRouter VIF, MPLS, Nexthost, VRF, VRF's routes, flows, and dump information.

Action	Command
vRouter Kernel space human readable statistics	See: Display Contrail vRouter statistics.
vRouter DPDK human readable statistics	See: Display Contrail vRouter statistics.
To perform a packet capture from a DPDK interface (do not use grep after vifdump)	**# vifdump vif0/234 <tcpdump standard arguments such as -v -nn -e -w <path/to/file>>**
Display all vRouter interfaces and sub-interfaces statistics and details	**# vif --list**
Display vRouter statistics and details for a given interface	**# vif --list --get 234**
Display vRouter packer rate for all interfaces and sub-interfaces	**# vif --list --rate**
Display vRouter packer rate for a given interfaces	**# vif --list --rate --get 234**
Display vRouter packet drop statistics for a given interface	**# vif --list --get 234 --get-drop-stats**

Action	Command
Display vRouter flows	**# flow -l**
Display real-time vRouter flow actions	**# flow -r**
Display vRouter packet statistics for a given VRF (you can find VRF number from vif --list)	**# vrfstats --get 0**
Display vRouter packet statistics for all VRF	**# vrfstats --dump**
Display vRouter routing table for a given VRF (you can find the VRF number from vif --list)	**# rt --dump 0**
Display vRouter IPv4 routing table for a given VRF (you can find the VRF number from vif --list)	**# rt --dump 0 --family inet**
Display vRouter IPv6 routing table for a given VRF (you can find the VRF number from vif --list)	**# rt --dump 0 --family inet6**
Display vRouter forwarding table for a given VRF (you can find the VRF number from vif --list)	**# rt --dump 0 --family bridge**
Display vRouter route target in a given VRF for a given address	**# rt --get 0.0.0.0/0 --vrf 0 --family inet**
Display vRouter drop statistics	**# dropstats**
Display vRouter drop statistics for a given DPDK core	**# dropstats --core 11**
Display vRouter MPLS labels	**# mpls --dump**
Display vRouter nexthop for a given one (can be found from mpls --dump output)	**# nh --get 21**
Display all vRouter nexthops	**# nh --list**
Display all vRouter VXLAN VNID	**# vxlan --dump**
Display vRouter agents (supervisor, xmmp connection, vrouter agent etc) status	**# contrail-status**
Restart vRouter (and all Contrail local compute node components)	**# systemctl restart supervisor-vrouter**

For more information on Juniper Contrail vRouter CLI utilitlies, see the following documentation:

- Juniper Contrail 3.2 Documentation

- Juniper Contrail 4.0 Documentation
- Juniper Contrail 4.1 Documentation
- Juniper Contrail 5.0 Documentation

3.17. CONTAINERS

These are some of the commonly-used Docker and Podman commands for containers.

Action	Docker RHEL7	Podman RHEL8
Display all running containers	**# docker ps**	**# podman ps**
Display all containers (running, stopped etc)	**# docker ps -a**	**# podman ps -a**
Display all containers (running, stopped etc) without output truncated	**# docker ps -a --no-trunc**	**# podman ps -a --no-trunc**
Display all containers (running, stopped etc) json output	**# docker ps --format '{{ json .}}' | jq -C '.' s|# podman ps -a --format json**	**jq -C '.'**
Display container process tree for a given container	**# docker top <container ID>**	**# podman pod top <container ID>**
Display real-time containers resource utilization (CPU, Memory, I/O, Net) - TOP-like	**# docker stats**	**# podman stats**
Display real-time resource utilization for a given container (CPU, Memory, I/O, Net) - TOP-like	**# docker stats <container ID>**	**# podman stats <container ID>**
Log in to a given running container	**# docker exec -it <container ID> /bin/bash**	**# podman exec -it <container ID> /bin/bash**
Log in to a given running container as root user	**# docker exec -u root -it <container ID> /bin/bash**	**# podman exec -u root -it <container ID> /bin/bash**
Display port mapping in a given container	**# docker port <container ID>**	**# podman port <container ID>**
Display all locally stored images with name, ID, and tag	**# docker image ls** **# docker images"**	**# podman image ls** **# podman images"**

Action	Docker RHEL7	Podman RHEL8
Display history for a given image	**# docker history <image id>**	**# podman history <image id>**
Display low-level configuration for a given container	**# docker inspect <container ID>**	**# podman inspect <container ID>**
Display all volumes for a given container	**# docker inspect -f "{{ .Mounts }}" <container ID>**	**# podman inspect -f "{{ .Mounts }}" <container ID>**
Restart all containers with the same pattern	**# docker ps -q --filter "name=swift" | xargs -n1 docker restart**	**# podman ps -q --filter "name=swift" | xargs -n1 docker restart**

For more information on docker or podman, see the following documentation:

- Docker command reference
- Podman command reference

3.18. OPENSTACK

Use these OpenStack commands to show VM compute nodes.

Action	Command
Show list of all VMs on their compute nodes sorted by compute nodes	**$ nova list --fields name,OS-EXT-SRV-ATTR:host --sort host**
Show list of all VMs on their compute nodes sorted by vm name	**$ nova list --fields name,OS-EXT-SRV-ATTR:host**

CHAPTER 4. HIGH PACKET LOSS IN THE TX QUEUE OF THE INSTANCE'S TAP INTERFACE

Use this section to troubleshoot packet loss in the TX queue.

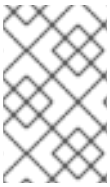
4.1. SYMPTOM

During a test of a virtual network function (VNF) using host-only networking, high packet loss can be observed in the TX queue of the instance's tap interface. The test setup sends packets from one VM on a node to another VM on the same node. The packet loss appears in bursts.

The following example shows a high number of dropped packets in the tap's TX queue.

```
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500034259301 132047795 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
               0        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5481296464 81741449 0       11155280 0       0
    TX errors: aborted  fifo   window heartbeat transns
          0       0       0       0       0
```

4.2. DIAGNOSIS



NOTE

This section examines packet drop on tap (kernel path) interfaces. For packet drops on vhost user interfaces in the user datapath, see https://access.redhat.com/solutions/3381011

TX drops occur because of interference between the instance's vCPU and other processes on the hypervisor. The TX queue of the tap interface is a buffer that can store packets for a short while in case that the instance cannot pick up the packets. This happens if the instance's CPU is prevented from running (or freezes) for a long enough time.

A TUN/TAP device is a virtual device where one end is a kernel network interface, and the other end is a user space file descriptor.

A TUN/TAP interface can run in one of two modes:

- **Tap mode** feeds L2 ethernet frames with L2 header into the device, and expects to receive the same out from user space. This mode is used for VMs.
- **Tun mode** feeds L3 IP packets with L3 header into the device, and expects to receive the same out from user space. This mode is mostly used for VPN clients.

In KVM networking, the user space file descriptor is owned by the **qemu-kvm** process. Frames that are sent into the tap (TX from the hypervisor's perspective) end up as L2 frames inside **qemu-kvm**, which can then feed those frames to the virtual network device in the VM as network packets received into the

virtual network interface (RX from the VM's perspective).

A key concept with TUN/TAP is that the transmit direction from the hypervisor is the receive direction for the virtual machine. This same is true of the opposite direction; receive for the hypervisor is equal to transmit from the virtual machine.

There is no "ring buffer" of packets on a virtio-net device. This means that if the TUN/TAP device's TX queue fills up because the VM is not receiving (either fast enough or at all) then there is nowhere for new packets to go, and the hypervisor sees TX loss on the tap.

If you notice TX loss on a TUN/TAP, increase the tap **txqueuelen** to avoid that, similar to increasing the RX ring buffer to stop receive loss on a physical NIC.

However, this assumes the VM is just "slow" and "bursty" at receive. If the VM is not executing fast enough all the time, or otherwise not receiving at all, tuning the TX queue length won't help. You must find out why the VM is not running or receiving.

If the only need is to improve VM packet handling performance, complete the following steps:

- Enable **virtio-net multiqueue** on the hypervisor.
- Balance those multiple virtual device interrupts on difference cores inside the VM.

This is documented in the libvirt domain spec for KVM and can be done with **virsh edit** on a RHEL KVM hypervisor.

If you cannot configure **virtio-net multiqueue** in Red Hat OpenStack Platform, consider configuring RPS inside the VM to balance receive load across multiple CPU cores with software. For more information, see **scaling.txt** in the kernel-doc package, or see the RPS section in the RHEL product documentation.

4.2.1. Workaround

To alleviate small freezes at the cost of higher latency and other disadvantages, increase the TX queue.

To temporarily increase **txqueuelen**, use the following command:

```
/sbin/ip link set tap<uuid> txqueuelen <new queue length>
```

To permanently increase **txqueulen**, create a udev rule:

```
cat <<'EOF'>/etc/udev/rules.d/71-net-txqueuelen.rules
SUBSYSTEM=="net", ACTION=="add", KERNEL=="tap*", ATTR{tx_queue_len}="10000"
EOF
```

After reloading udev or rebooting the system, new tap interfaces will come up with a queue length of 10000. For example:

```
[root@overcloud-compute-0 ~]# ip link ls | grep tap
29: tap122be807-cd: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 5505
qdisc pfifo_fast master qbr122be807-cd state UNKNOWN mode DEFAULT
group default qlen 10000
```

4.2.2. Diagnostic Steps

Use the following script to view the effects of CPU time being stolen from the hypervisor.

```
[root@ibm-x3550m4-9 ~]# cat generate-tx-drops.sh
#!/bin/bash

trap 'cleanup' INT

cleanup() {
  echo "Cleanup ..."
  if [ "x$HPING_PID" != "x" ]; then
    echo "Killing hping3 with PID $HPING_PID"
    kill $HPING_PID
  fi
  if [ "x$DD_PID" != "x" ]; then
    echo "Killing dd with PID $DD_PID"
    kill $DD_PID
  fi
  exit 0
}

VM_IP=10.0.0.20
VM_TAP=tapc18eb09e-01
VM_INSTANCE_ID=instance-00000012
LAST_CPU=$( lscpu | awk '/^CPU\(s\):/ { print $NF - 1 }' )
# this is a 12 core system, we are sending everything to CPU 11,
# so the taskset mask is 800 so set dd affinity only for last CPU
TASKSET_MASK=800

# pinning vCPU to last pCPU
echo "virsh vcpupin $VM_INSTANCE_ID 0 $LAST_CPU"
virsh vcpupin $VM_INSTANCE_ID 0 $LAST_CPU

# make sure that: nova secgroup-add-rule default udp 1 65535 0.0.0.0/0
# make sure that: nova secgroup-add-rule default tcp 1 65535 0.0.0.0/0
# make sure that: nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
# --fast, --faster or --flood can also be used
echo "hping3 -u -p 5000 $VM_IP --faster > /dev/null "
hping3 -u -p 5000 $VM_IP --faster > /dev/null &
HPING_PID=$!

echo "hping is running, but dd not yet:"
for i in { 1 .. 3 }; do
  date
  echo "ip -s -s link ls dev $VM_TAP"
  ip -s -s link ls dev $VM_TAP
  sleep 5
done

echo "Starting dd and pinning it to the same pCPU as the instance"
echo "dd if=/dev/zero of=/dev/null"
dd if=/dev/zero of=/dev/null &
DD_PID=$!
echo "taskset -p $TASKSET_MASK $DD_PID"
taskset -p $TASKSET_MASK $DD_PID

for i in { 1 .. 3 }; do
```

```
  date
  echo "ip -s -s link ls dev $VM_TAP"
  ip -s -s link ls dev $VM_TAP
  sleep 5
done

cleanup
```

Log in to the instance and start **dd if=/dev/zero of=/dev/null** to generate additional load on its only vCPU. Note that this is for demonstration purposes. You can repeat the same test with and without load from within the VM. TX drop only occurs when another process on the hypervisor is stealing time from the instance's vCPU.

The following example shows an instance before the test:

```
%Cpu(s): 22.3 us, 77.7 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  1884108 total,  1445636 free,    90536 used,   347936 buff/cache
KiB Swap:        0 total,        0 free,        0 used.  1618720 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
30172 root      20   0  107936    620    528 R 99.9  0.0   0:05.89 dd
```

Run the following script and observe the dropped packages in the TX queue. These only occur when the dd process consumes a significant amount of processing time from the instance's CPU.

```
[root@ibm-x3550m4-9 ~]# ./generate-tx-drops.sh
virsh vcpupin instance-00000012 0 11

hping3 -u -p 5000 10.0.0.20 --faster > /dev/null
hping is running, but dd not yet:
Tue Nov 29 12:28:22 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500034259301 132047795 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
               0        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5481296464 81741449 0       11155280 0       0
    TX errors: aborted  fifo   window heartbeat transns
               0        0       0       0       0
Tue Nov 29 12:28:27 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500055729011 132445382 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
               0        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5502766282 82139038 0       11155280 0       0
    TX errors: aborted  fifo   window heartbeat transns
```

```
            0       0       0       0       0
Tue Nov 29 12:28:32 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500077122125 132841551 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5524159396 82535207 0       11155280 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Tue Nov 29 12:28:37 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500098181033 133231531 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5545218358 82925188 0       11155280 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Tue Nov 29 12:28:42 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500119152685 133619793 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5566184804 83313451 0       11155280 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Starting dd and pinning it to the same pCPU as the instance
dd if=/dev/zero of=/dev/null
taskset -p 800 8763
pid 8763's current affinity mask: fff
pid 8763's new affinity mask: 800
Tue Nov 29 12:28:47 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500140267091 134010698 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5587300452 83704477 0       11155280 0       0
```

```
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Tue Nov 29 12:28:52 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500159822749 134372711 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5606853168 84066563 0       11188074 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Tue Nov 29 12:28:57 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500179161241 134730729 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5626179144 84424451 0       11223096 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Tue Nov 29 12:29:02 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500198344463 135085948 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5645365410 84779752 0       11260740 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Tue Nov 29 12:29:07 EST 2016
ip -s -s link ls dev tapc18eb09e-01
69: tapc18eb09e-01: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master qbrc18eb09e-01 state UNKNOWN mode DEFAULT qlen 1000
    link/ether fe:16:3e:a5:17:c0 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    5500217014275 135431570 0       0       0       0
    RX errors: length   crc     frame   fifo    missed
            0       0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    5664031398 85125418 0       11302179 0       0
    TX errors: aborted  fifo   window heartbeat transns
            0       0       0       0       0
Cleanup ...
Killing hping3 with PID 8722
```

```
Killing dd with PID 8763
[root@ibm-x3550m4-9 ~]#
--- 10.0.0.20 hping statistic ---
3919615 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

The following example shows the effects of **dd** on the hypervisor during the test. The **st** label identifies the percentage of time stolen from the hypervisor.

```
%Cpu(s):  7.0 us, 27.5 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi, 20.2 si, 45.4 st
KiB Mem :  1884108 total,  1445484 free,    90676 used,   347948 buff/cache
KiB Swap:        0 total,        0 free,        0 used.  1618568 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
30172 root      20   0  107936    620    528 R 54.3  0.0   1:00.50 dd
```

Note that **ssh** can become sluggish during the second half of the test on the instance, including the possibility of timing out if the test runs too long.

4.3. SOLUTION

While increasing the TX queue helps to mitigate these small freezes, complete isolation with CPU pinning and isolcpus in the kernel parameters is the best solution. Form more information, see Configure CPU pinning with NUMA in OpenStack for further details.

CHAPTER 5. TX DROPS ON INSTANCE VHU INTERFACES WITH OPEN VSWITCH DPDK

Use this procedure to troubleshoot transmit drops on instance vhost-user (VHU) interface.

5.1. SYMPTOM

Packets go from the vswitch to the guest using the virtio transport without passing through the kernel or qemu processes. This is done by exchanging packets with the VHU interface.

The VHU is mostly implemented by DPDK librte_vhost that also offers functions to send or receive batches of packets. The backend of VHU is a virtio ring provided by qemu to exchange packets with the virtual machine. The virtio ring has a special format comprised of descriptors and buffers.

The TX/RX (transmit/receive) statistics are for OpenvSwitch (OVS). This means that transmit statistics relate directly to receive statistics for the VM.

If the VM does not process packets fast enough, the OVS TX queue overflows and drops packets.

5.1.1. Explanation for Packet Drops

A saturated virtio ring causes TX drops on the vhost-user device. The virtio ring is located in the guest's memory and it works like a queue where the vhost-user pushes packets and the VM consumes them. If the VM is not fast enough to consume the packets, the virtio ring runs out of buffers and the vhost-user drops packets.

Use the Perf and Ftrace tools to troubleshoot packet drops.

- Use Perf to count the number of scheduler switches, which could show whether the qemu thread preempted.
- Use Ftrace to show the reason for preemption, as well as how long it took.

Reasons for preemption include:

- Time Interrupt (kernel ticks):
  These add the cost of at least two context switches. The timer interrupt can also run read-copy update (RCU) callbacks which can take an unpredictable amount of time.
- CPU power management and hyperthreading

You can find these tools in the following packages:

- PERF: **perf rpm in rhel-7-server-rpms/7Server/x86_64**. For more information, see About Perf
- FTRACE: **trace-cmd info rhel-7-server-rpms/7Server/x86_64**. For more information, see About Ftrace

5.1.2. Explanation for other drops

Prior to OVS 2.9, vHost user ports were created in **dpdkvhostuser** mode. In this mode, OVS acts as the vhost server, and QEMU acts as the client. When an instance goes down or restarts, the vhost user port on the OVS bridge, still active, drops packets destined for the VM. This increases the **tx_drop_counter**:

In the following example, the VM was stopped with **nova stop <UUID>**:

```
[root@overcloud-compute-0 network-scripts]# ovs-vsctl list interface vhubd172106-73 | grep _state
admin_state         : up
link_state          : down
```

This is similar to what happens when the kernel port is shut down with **ip link set dev <br internal port name>** down and frames are dropped in userspace.

When the VM is up, it connects to the same vhu socket and will start emptying the virtio ring buffer. TX is no longer interrupted and normal network traffic resumes.

5.1.3. Increasing the TX and RX queue lengths for DPDK

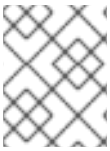
You can change TX and RX queue lengths for DPDK with the following OpenStack director template modifications:

```
NovaComputeExtraConfig:
      nova::compute::libvirt::rx_queue_size: '"1024"'
      nova::compute::libvirt::tx_queue_size: '"1024"'
```

The following example shows validation checks:

```
[root@overcloud-compute-1 ~]# ovs-vsctl get interface vhu9a9b0feb-2e status
{features="0x0000000150208182", mode=client, num_of_vrings="2", numa="0",
socket="/var/lib/vhost_sockets/vhu9a9b0feb-2e", status=connected, "vring_0_size"="1024",
"vring_1_size"="1024"}

[root@overcloud-compute-1 ~]# virsh dumpxml instance-00000017 | grep rx
<driver rx_queue_size='1024' tx_queue_size='1024'/>
<driver rx_queue_size='1024' tx_queue_size='1024'/>
```

Due to kernel limitations, you cannot increase the queue size beyond 1024.



NOTE

If you plan for PXE boot to be available for neutron networks over DPDK, you must verify that the PXE version supports 1024 bytes.

5.2. DIAGNOSIS

You can see TX drops towards the vhost user ports when the guest cannot receive packets. TCP is designed to recover from packet loss, which occurs in normal network conditions. NFVi has strict requirements with less tolerance for packet drops.

Use DPDK-accelerated OVS, as the kernel datapath is too slow for NFVi. Additionally, it is important to deploy DPDK-enabled guests that can match the packet processing speed of the host.

5.3. SOLUTION

Ensure that the vCPUs allocated to the VM are only processing tasks for the guests.

- Check that the cluster was deployed with the heat following template parameters:
    - **IsolcpusList**: Removes CPUs from scheduling

- **NovaVcpuPinSet**: Assigns CPUs for pinning
- **NovaComputeCpuSharedSet**: Allocates CPUs for emulator thread pinning

Example:

```
  parameter_defaults:
    ComputeOvsDpdkParameters:
      KernelArgs: "default_hugepagesz=1GB hugepagesz=1G hugepages=32 iommu=pt
intel_iommu=on isolcpus=2-19,22-39"
      IsolCpusList: "2-19,22-39"
      NovaVcpuPinSet: ['4-19,24-39']
      NovaReservedHostMemory: 4096
      OvsDpdkSocketMemory: "3072,1024"
      OvsDpdkMemoryChannels: "4"
      OvsDpdkCoreList: "0,20,1,21"
      OvsPmdCoreList: "2,22,3,23"
      NovaComputeCpuSharedSet: [0,20,1,21]
```

- Ensure that VMs are deployed with a flavor that takes advantage of pinned CPUs and the emulator pool set.

Example:

```
openstack flavor create --ram <size_mb> --disk <size_gb> -\
-vcpus <vcpus> --property dpdk=true \
--property hw:mem_page_size=1G \
--property hw:cpu_policy=dedicated \
--property hw:emulator_threads_policy=share <flavor>
```

- Ensure that these settings are operating as intended. For more information, see Simple Compute Node CPU Partitioning and Memory Checks for details.

If you allocate completely dedicated CPU resources to the instance and still observe network packet loss, ensure that the instance is properly tuned and DPDK enabled.

CHAPTER 6. INTERPRETING THE OUTPUT OF THE PMD-STATS-SHOW COMMAND IN OPEN VSWITCH WITH DPDK

Use this section to interpret the output of the **pmd-stats-show** command (**ovs-appctl dpif-netdev/pmd-stats-show**) in Open vSwitch (OVS) with DPDK.

6.1. SYMPTOM

The **ovs-appctl dpif-netdev/pmd-stats-show** command provides an inaccurate measurement. This is due to gathered statistics that have been charted since PMD was started.

6.2. DIAGNOSIS

To obtain useful output, put the system into a steady state and reset the statistics that you want to measure:

```
# put system into steady state
ovs-appctl dpif-netdev/pmd-stats-clear
# wait <x> seconds
sleep <x>
ovs-appctl dpif-netdev/pmd-stats-show
```

Here's an example of the output:

```
[root@overcloud-compute-0 ~]# ovs-appctl dpif-netdev/pmd-stats-clear && sleep 10 && ovs-appctl
dpif-netdev/pmd-stats-show |
egrep 'core_id (2|22):' -A9
pmd thread numa_id 0 core_id 22:
    emc hits:17461158
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:4948219259 (25.81%)
    processing cycles:14220835107 (74.19%)
    avg cycles per packet: 1097.81 (19169054366/17461158)
    avg processing cycles per packet: 814.43 (14220835107/17461158)
--
pmd thread numa_id 0 core_id 2:
    emc hits:14874381
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:5460724802 (29.10%)
    processing cycles:13305794333 (70.90%)
    avg cycles per packet: 1261.67 (18766519135/14874381)
    avg processing cycles per packet: 894.54 (13305794333/14874381)
```

Note that **core_id 2** is mainly busy, spending 70% of the time processing and 30% of the time polling.

```
polling cycles:5460724802 (29.10%)
processing cycles:13305794333 (70.90%)
```

In this example, **miss** indicates packets that were not classified in the DPDK datapath ('emc' or 'dp' classifier). Under normal circumstances, they would then be sent to the **ofproto** layer. On rare occasions, due to a flow revalidation lock or if the **ofproto** layer returns an error, the packet is dropped. In this case, the value of **lost** will also be incremented to indicate the loss.

```
emc hits:14874381
megaflow hits:0
avg. subtable lookups per hit:0.00
miss:0
lost:0
```

For more information, see OVS-DPDK Datapath Classifier.

6.3. SOLUTION

This section shows the procedures for viewing traffic flow using the **ovs-appctl** command.

6.3.1. Idle PMD

The following example shows a system where the core_ids serve the PMDs that are pinned to dpdk0, with only management traffic flowing through dpdk0:

```
[root@overcloud-compute-0 ~]# ovs-appctl dpif-netdev/pmd-stats-clear && sleep 10 && ovs-appctl
dpif-netdev/pmd-stats-show |
egrep 'core_id (2|22):' -A9
pmd thread numa_id 0 core_id 22:
    emc hits:0
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:12613298746 (100.00%)
    processing cycles:0 (0.00%)
--
pmd thread numa_id 0 core_id 2:
    emc hits:5
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:12480023709 (100.00%)
    processing cycles:14354 (0.00%)
    avg cycles per packet: 2496007612.60 (12480038063/5)
    avg processing cycles per packet: 2870.80 (14354/5)
```

6.3.2. PMD under load test with packet drop

The following example shows a system where the core_ids serve the PMDs that are pinned to dpdk0, with a load test flowing through dpdk0, causing a high number of RX drops:

```
[root@overcloud-compute-0 ~]# ovs-appctl dpif-netdev/pmd-stats-clear && sleep 10 && ovs-appctl
dpif-netdev/pmd-stats-show |
egrep 'core_id (2|4|22|24):' -A9
pmd thread numa_id 0 core_id 22:
    emc hits:35497952
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:1446658819 (6.61%)
    processing cycles:20453874401 (93.39%)
    avg cycles per packet: 616.95 (21900533220/35497952)
    avg processing cycles per packet: 576.20 (20453874401/35497952)
--
pmd thread numa_id 0 core_id 2:
    emc hits:30183582
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:2
    lost:0
    polling cycles:1497174615 (6.85%)
    processing cycles:20354613261 (93.15%)
    avg cycles per packet: 723.96 (21851787876/30183584)
    avg processing cycles per packet: 674.36 (20354613261/30183584)
```

Where packet drops occur, you can see a high ratio of processing cycles vs polling cycles (more than 90% processing cycles):

```
polling cycles:1497174615 (6.85%)
processing cycles:20354613261 (93.15%)
```

Check the average cycles per packet (CPP) and average processing cycles per packet (PCPP). You can expect a PCPP/CPP ratio of 1 for a fully loaded PMD as there will be no idle cycles counted.

```
avg cycles per packet: 723.96 (21851787876/30183584)
avg processing cycles per packet: 674.36 (20354613261/30183584)
```

6.3.3. PMD under loadtest with 50% of mpps capacity

The following example shows a system where the core_ids serve the PMDs that are pinned to dpdk0, with a load test flowing through dpdk0, sending 6.4 Mpps (around 50% of the maximum capacity) of this dpdk0 interface (around 12.85 Mpps):

```
[root@overcloud-compute-0 ~]# ovs-appctl dpif-netdev/pmd-stats-clear && sleep 10 && ovs-appctl
dpif-netdev/pmd-stats-show |
egrep 'core_id (2|4|22|24):' -A9
pmd thread numa_id 0 core_id 22:
    emc hits:17461158
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:4948219259 (25.81%)
    processing cycles:14220835107 (74.19%)
```

```
    avg cycles per packet: 1097.81 (19169054366/17461158)
    avg processing cycles per packet: 814.43 (14220835107/17461158)
--
pmd thread numa_id 0 core_id 2:
    emc hits:14874381
    megaflow hits:0
    avg. subtable lookups per hit:0.00
    miss:0
    lost:0
    polling cycles:5460724802 (29.10%)
    processing cycles:13305794333 (70.90%)
    avg cycles per packet: 1261.67 (18766519135/14874381)
    avg processing cycles per packet: 894.54 (13305794333/14874381)
```

Where the pps are about half of the maximum for the interface, you can see a lower ratio of processing cycles vs polling cycles (approximately 70% processing cycles):

```
polling cycles:5460724802 (29.10%)
processing cycles:13305794333 (70.90%)
```

6.3.4. Hit vs miss vs lost

The following examples shows the man pages regarding the subject:

```
an ovs-vswitchd
(...)
  DPIF-NETDEV COMMANDS
      These commands are used to expose  internal  information  (mostly  statistics)
      about  the  `dpif-netdev`  userspace datapath. If there is only one datapath
      (as is often the case, unless dpctl/ commands are used), the dp  argument  can
      be omitted.

      dpif-netdev/pmd-stats-show [dp]
            Shows  performance  statistics  for each pmd thread of the datapath dp.
            The special thread ``main'' sums up the statistics  of  every  non  pmd
            thread.   The  sum of ``emc hits'', ``masked hits'' and ``miss'' is the
            number of packets received by the datapath.  Cycles are  counted  using
            the  TSC  or  similar  facilities when available on the platform.  To
            reset these counters use dpif-netdev/pmd-stats-clear. The  duration  of
            one cycle depends on the measuring infrastructure.
(...)

Raw

man ovs-dpctl
(...)
      dump-dps
            Prints the name of each configured datapath on a separate line.

      [-s | --statistics] show [dp...]
            Prints  a summary of configured datapaths, including their datapath numbers and a list of
ports connected to each datapath.  (The local port is
            identified as port 0.)  If -s or --statistics is specified, then packet and byte counters are also
printed for each port.
```

> The datapath numbers consists of flow stats and mega flow mask stats.
>
> The "lookups" row displays three stats related to flow lookup triggered by processing incoming packets in the datapath. "hit" displays number
> of packets matches existing flows. "missed" displays the number of packets not matching any existing flow and require user space processing.
> "lost" displays number of packets destined for user space process but subsequently dropped before reaching userspace. The sum of "hit" and
> "miss" equals to the total number of packets datapath processed.
> (...)
>
> Raw
>
> man ovs-vswitchd
> (...)
> dpctl/show [-s | --statistics] [dp...]
> Prints a summary of configured datapaths, including their datapath numbers and a list of ports connected to each datapath. (The local port is identified as
> port 0.) If -s or --statistics is specified, then packet and byte counters are also printed for each port.
>
> The datapath numbers consists of flow stats and mega flow mask stats.
>
> The "lookups" row displays three stats related to flow lookup triggered by processing incoming packets in the datapath. "hit" displays number of packets
> matches existing flows. "missed" displays the number of packets not matching any existing flow and require user space processing. "lost" displays number of
> packets destined for user space process but subsequently dropped before reaching userspace. The sum of "hit" and "miss" equals to the total number of packets
> datapath processed.
> (...)

NOTE

Some of the documentation is referring to the kernel datapath, so when it says **user space processing** it means the packet is not classified in the kernel **sw** caches (equivalents to **emc** & **dpcls**) and sent to the ofproto layer in userspace.

CHAPTER 7. ATTACHING AND DETACHING SR-IOV PORTS IN NOVA

Use the following section to attach and detach SR-IOV ports.

7.1. SYMPTOM

You are unable to attach or detach SR-IOV ports in nova in Red Hat OpenStack Platform 10 and later. Nova logs report **No conversion for VIF type hw_veb yet**.

7.2. DIAGNOSIS

You cannot attach or detach SR-IOV ports to an instance that has already been created. SR-IOV ports need to be attached at instance creation.

7.3. SOLUTION

The following example shows an attempt to attach interfaces after an instance boot:

```
RHEL_INSTANCE_COUNT=1
NETID=$(neutron net-list | grep provider1 | awk '{print $2}')
for i in `seq 1 $RHEL_INSTANCE_COUNT`;do
#  nova floating-ip-create provider1
  portid1=`neutron port-create sriov1 --name sriov1 --binding:vnic-type direct  | awk '$2 == "id" {print $(NF-1)}'`
  portid2=`neutron port-create sriov2 --name sriov2 --binding:vnic-type direct  | awk '$2 == "id" {print $(NF-1)}'`
  openstack server create --flavor m1.small  --image rhel --nic net-id=$NETID --key-name id_rsa sriov_vm${i}
  serverid=`openstack server list | grep sriov_vm${i} | awk '{print $2}'`
  status="NONE"
  while [ "$status" != "ACTIVE" ]; do
    echo "Server $serverid not active ($status)" ; sleep 5 ;
    status=`openstack server show $serverid | grep -i status | awk '{print $4}'`
  done
  nova interface-attach --port-id $portid1 $serverid
  nova interface-attach --port-id $portid2 $serverid
done
```

This fails with the following error:

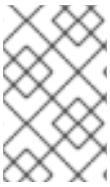
```
ERROR (ClientException): Unexpected API Error. Please report this at http://bugs.launchpad.net/nova/ and attach the Nova API log if possible.
<type 'exceptions.KeyError'> (HTTP 500) (Request-ID: req-36b544f4-91a6-442e-a30d-6148220d1449)
```

The correct method is to spawn an instance directly with SR-IOV ports:

```
RHEL_INSTANCE_COUNT=1
NETID=$(neutron net-list | grep provider1 | awk '{print $2}')
for i in `seq 1 $RHEL_INSTANCE_COUNT`;do
#  nova floating-ip-create provider1
```

```
  portid1=`neutron port-create sriov1 --name sriov1 --binding:vnic-type direct  | awk '$2 == "id" {print
$(NF-1)}'`
  portid2=`neutron port-create sriov2 --name sriov2 --binding:vnic-type direct  | awk '$2 == "id" {print
$(NF-1)}'`
  openstack server create --flavor m1.small  --image rhel --nic net-id=$NETID --nic port-id=$portid1 --
nic port-id=$portid2 --key-name id_rsa   sriov_vm${i}
done
```

CHAPTER 8. CONFIGURE AND TEST LACP BONDING WITH OPEN VSWITCH DPDK



NOTE

OVS bonds with LACP might not be supported depending on the version of Red Hat OpenStack Platform (RHOSP) you are using. Check the product documentation to verify that OVS bonds with LACP are supported.

To use Open vSwitch DPDK to configure and test LACP bonding, complete the following tasks:

1. Configure the switch ports for LACP.
2. Configure Linux kernel bonding for LACP as a baseline.
3. Configure OVS DPDK bonding for LACP.

NOTE

This topic describes switch configuration with a Dell S4048-ON switch. Whereas configuration of RHEL and OVS remains the same, different switch vendors' operating systems will use a different syntax to configure LACP.

8.1. CONFIGURING THE SWITCH PORTS FOR LACP

1. Reset the switch interfaces to their default settings:

```
S4048-ON-sw#config t
S4048-ON-sw(conf)#default int te1/2
S4048-ON-sw(conf)#default int te1/7
```

2. Configure the port-channel and other port settings:

```
S4048-ON-sw(conf)#int range te1/2,te1/7
S4048-ON-sw(conf-if-range-te-1/2,te-1/7)#port-channel-protocol lacp
S4048-ON-sw(conf-if-range-te-1/2,te-1/7-lacp)#
S4048-ON-sw(conf-if-range-te-1/2,te-1/7-lacp)#port-channel 1 mode active
S4048-ON-sw(conf-if-range-te-1/2,te-1/7-lacp)#end
S4048-ON-sw#config t
S4048-ON-sw(conf)#int range te1/2,te1/7
S4048-ON-sw(conf-if-range-te-1/2,te-1/7)# no ip address
S4048-ON-sw(conf-if-range-te-1/2,te-1/7)# mtu 9216
S4048-ON-sw(conf-if-range-te-1/2,te-1/7)# flowcontrol rx on tx off
S4048-ON-sw(conf-if-range-te-1/2,te-1/7)# no shutdown
S4048-ON-sw(conf-if-range-te-1/2,te-1/7)#end
S4048-ON-sw#show run int te1/2
!
interface TenGigabitEthernet 1/2
 no ip address
 mtu 9216
 flowcontrol rx on tx off
!
```

```
port-channel-protocol LACP
 port-channel 1 mode active
no shutdown
```

3. Configure the VLANs:

```
S4048-ON-sw#config t
S4048-ON-sw(conf)#int range vlan901-909
S4048-ON-sw(conf-if-range-vl-901-909)#tagged Port-channel 1
S4048-ON-sw(conf-if-range-vl-901-909)#end
S4048-ON-sw#
```

4. Verify VLAN tagging:

```
S4048-ON-sw#show vlan id 902

Codes: * - Default VLAN, G - GVRP VLANs, R - Remote Port Mirroring VLANs, P - Primary,
C - Community, I - Isolated
       O - Openflow, Vx - Vxlan
Q: U - Untagged, T - Tagged
   x - Dot1x untagged, X - Dot1x tagged
   o - OpenFlow untagged, O - OpenFlow tagged
   G - GVRP tagged, M - Vlan-stack
   i - Internal untagged, I - Internal tagged, v - VLT untagged, V - VLT tagged

    NUM    Status    Description                  Q Ports
    902    Active    Tenant                     T Po1()
                                              T Te 1/1,1/3-1/6,1/8-1/20
```

5. Verify the LACP configuration:

```
S4048-ON-sw#show lacp 1
Port-channel 1 admin up, oper down, mode lacp
LACP Fast Switch-Over Disabled
Actor   System ID:  Priority 32768, Address 1418.7789.9a8a
Partner System ID:  Priority 0, Address 0000.0000.0000
Actor Admin Key 1, Oper Key 1, Partner Oper Key 1, VLT Peer Oper Key 1
LACP LAG 1 is an individual link

LACP LAG 1 is a normal LAG

A - Active LACP, B - Passive LACP, C - Short Timeout, D - Long Timeout
E - Aggregatable Link, F - Individual Link, G - IN_SYNC, H - OUT_OF_SYNC
I - Collection enabled, J - Collection disabled, K - Distribution enabled
L - Distribution disabled, M - Partner Defaulted, N - Partner Non-defaulted,
O - Receiver is in expired state, P - Receiver is not in expired state

Port Te 1/2 is disabled, LACP is disabled and mode is lacp
Port State: Not in Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEHJLMP Key 1 Priority 32768
  Partner is not present

Port Te 1/7 is enabled, LACP is enabled and mode is lacp
Port State: Not in Bundle
```

```
Actor   Admin: State ACEHJLMP Key 1 Priority 32768
        Oper: State ACEHJLMP Key 1 Priority 32768
Partner is not present
```

8.2. CONFIGURING LINUX KERNEL BONDING FOR LACP AS A BASELINE

Configure Linux kernel bonding as a baseline, then verify that the host can form an LACP bond with the switch.

1. Move all interfaces to the kernel space and test with kernel space bonding. In this example, p1p1 maps to bus address **0000:04:00.0** and p1p2 maps to bus address **0000:04:00.1**.

```
[root@baremetal ~]# driverctl unset-override 0000:04:00.0
[root@baremetal ~]# driverctl unset-override 0000:04:00.1
```

2. Load the bonding driver, configure a bond interface (**bond10**) and enslave interfaces **p1p1** and **p1p2**:

```
[root@baremetal ~]# modprobe bonding miimon=100 mode=4 lacp_rate=1
[root@baremetal ~]# ip link add name bond10 type bond
[root@baremetal ~]# ifenslave bond10 p1p1 p1p2
Illegal operation; the specified master interface 'bond10' is not up.
[root@baremetal ~]# ip link set dev bond10 up
[root@baremetal ~]# ifenslave bond10 p1p1 p1p2
```

3. Verify LACP from RHEL:

```
[root@baremetal ~]# cat /proc/net/bonding/bond10
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: IEEE 802.3ad Dynamic link aggregation
Transmit Hash Policy: layer2 (0)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0
802.3ad info
LACP rate: fast
Min links: 0
Aggregator selection policy (ad_select): stable
System priority: 65535
System MAC address: a0:36:9f:e3:dd:c8
Active Aggregator Info:
        Aggregator ID: 1
        Number of ports: 2
        Actor Key: 13
        Partner Key: 1
        Partner Mac Address: 14:18:77:89:9a:8a

Slave Interface: p1p1
MII Status: up
Speed: 10000 Mbps
Duplex: full
```

```
Link Failure Count: 0
Permanent HW addr: a0:36:9f:e3:dd:c8
Slave queue ID: 0
Aggregator ID: 1
Actor Churn State: monitoring
Partner Churn State: monitoring
Actor Churned Count: 0
Partner Churned Count: 0
details actor lacp pdu:
    system priority: 65535
    system mac address: a0:36:9f:e3:dd:c8
    port key: 13
    port priority: 255
        port number: 1
        port state: 63
    details partner lacp pdu:
        system priority: 32768
        system mac address: 14:18:77:89:9a:8a
        oper key: 1
        port priority: 32768
        port number: 203
        port state: 63

    Slave Interface: p1p2
    MII Status: up
    Speed: 10000 Mbps
    Duplex: full
    Link Failure Count: 0
    Permanent HW addr: a0:36:9f:e3:dd:ca
    Slave queue ID: 0
    Aggregator ID: 1
    Actor Churn State: monitoring
    Partner Churn State: monitoring
    Actor Churned Count: 0
    Partner Churned Count: 0
    details actor lacp pdu:
        system priority: 65535
        system mac address: a0:36:9f:e3:dd:c8
        port key: 13
        port priority: 255
        port number: 2
        port state: 63
    details partner lacp pdu:
        system priority: 32768
        system mac address: 14:18:77:89:9a:8a
        oper key: 1
        port priority: 32768
        port number: 208
        port state: 63
```

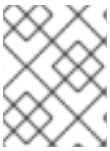
4. Verify LACP from the switch:

```
S4048-ON-sw#show lacp 1
Port-channel 1 admin up, oper up, mode lacp
LACP Fast Switch-Over Disabled
Actor   System ID:  Priority 32768, Address 1418.7789.9a8a
```

```
Partner System ID:  Priority 65535, Address a036.9fe3.ddc8
Actor Admin Key 1, Oper Key 1, Partner Oper Key 13, VLT Peer Oper Key 1
LACP LAG 1 is an aggregatable link
LACP LAG 1 is a normal LAG

A - Active LACP, B - Passive LACP, C - Short Timeout, D - Long Timeout
E - Aggregatable Link, F - Individual Link, G - IN_SYNC, H - OUT_OF_SYNC
I - Collection enabled, J - Collection disabled, K - Distribution enabled
L - Distribution disabled, M - Partner Defaulted, N - Partner Non-defaulted,
O - Receiver is in expired state, P - Receiver is not in expired state

Port Te 1/2 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ACEGIKNP Key 13 Priority 255

Port Te 1/7 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ACEGIKNP Key 13 Priority 255
S4048-ON-sw#
```

5. Remove the bonding configuration:

```
[root@baremetal ~]# ip link del dev bond10
[root@baremetal ~]#
```



NOTE

For information about changing the bonding mode, see: How to change the bonding mode without rebooting the system?

8.3. CONFIGURING OVS DPDK BONDING FOR LACP

The next objective is to configure an LACP bond within OVS DPDK.

8.3.1. Prepare Open vSwitch

1. Ensure that huge pages and other values are configured in RHEL:

```
[root@baremetal bonding]# cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinuz-3.10.0-693.17.1.el7.x86_64 root=UUID=fa414390-f78d-49d4-
a164-54615a32977b ro console=tty0
console=ttyS0,115200n8 crashkernel=auto rhgb quiet default_hugepagesz=1GB
hugepagesz=1G hugepages=32 iommu=pt intel_iommu=on
isolcpus=2,4,6,8,10,12,14,16,18,22,24,26,28,30,32,34,36,38,3,5,7,9,11,13,15,17,19,23,25,27,2
9,31,33,35,37,39 skew_tick=1
nohz=on
nohz_full=2,4,6,8,10,12,14,16,18,22,24,26,28,30,32,34,36,38,3,5,7,9,11,13,15,17,19,23,25,27,
29,31,33,35,37,39
```

```
rcu_nocbs=2,4,6,8,10,12,14,16,18,22,24,26,28,30,32,34,36,38,3,5,7,9,11,13,15,17,19,23,25,27,29,31,33,35,37,39
tuned.non_isolcpus=00300003 intel_pstate=disable nosoftlockup
```

2. Configure OVS for DPDK:

```
[root@baremetal bonding]# ovs-vsctl list Open_vSwitch | grep other
other_config        : {}
[root@baremetal bonding]# ovs-vsctl --no-wait set Open_vSwitch . other_config:dpdk-init="true"
[root@baremetal bonding]# ovs-vsctl --no-wait set Open_vSwitch . other_config:pmd-cpu-mask=0x17c0017c
[root@baremetal bonding]# ovs-vsctl --no-wait set Open_vSwitch . other_config:dpdk-lcore-mask=0x00000001
```

3. Switch interfaces into user space:

```
[root@baremetal bonding]# ethtool -i p1p1 | grep bus
bus-info: 0000:04:00.0
[root@baremetal bonding]# ethtool -i p1p2 | grep bus
bus-info: 0000:04:00.1
[root@baremetal bonding]# driverctl set-override 0000:04:00.0 vfio-pci
[root@baremetal bonding]# driverctl set-override 0000:04:00.1 vfio-pci
```

4. Restart Open vSwitch, **journalctl -u ovs-vswitchd -f &** running in the background:

```
[root@baremetal bonding]# systemctl restart openvswitch
Apr 19 13:02:49 baremetal systemd[1]: Stopping Open vSwitch Forwarding Unit...
Apr 19 13:02:49 baremetal systemd[1]: Stopping Open vSwitch Forwarding Unit...
Apr 19 13:02:49 baremetal ovs-ctl[91399]: Exiting ovs-vswitchd (91202) [  OK  ]
Apr 19 13:02:49 baremetal ovs-ctl[91399]: Exiting ovs-vswitchd (91202) [  OK  ]
Apr 19 13:02:49 baremetal systemd[1]: Starting Open vSwitch Forwarding Unit...
Apr 19 13:02:49 baremetal systemd[1]: Starting Open vSwitch Forwarding Unit...
Apr 19 13:02:49 baremetal ovs-ctl[91483]: Starting ovs-vswitchd EAL: Detected 40 lcore(s)
Apr 19 13:02:49 baremetal ovs-ctl[91483]: Starting ovs-vswitchd EAL: Detected 40 lcore(s)
Apr 19 13:02:49 baremetal ovs-ctl[91483]: EAL: Probing VFIO support...
Apr 19 13:02:49 baremetal ovs-vswitchd[91509]: EAL: Probing VFIO support...
Apr 19 13:02:49 baremetal ovs-ctl[91483]: EAL: VFIO support initialized
Apr 19 13:02:49 baremetal ovs-vswitchd[91509]: EAL: VFIO support initialized
Apr 19 13:02:49 baremetal ovs-ctl[91483]: EAL: Probing VFIO support...
Apr 19 13:02:49 baremetal ovs-vswitchd[91509]: EAL: Probing VFIO support...
Apr 19 13:02:49 baremetal ovs-ctl[91483]: EAL: VFIO support initialized
Apr 19 13:02:49 baremetal ovs-vswitchd[91509]: EAL: VFIO support initialized
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: PCI device 0000:04:00.0 on NUMA socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: PCI device 0000:04:00.0 on NUMA socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:04:00.0 on NUMA socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL:   using IOMMU type 1 (Type 1)
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL:   using IOMMU type 1 (Type 1)
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:04:00.0 on NUMA socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL:   using IOMMU type 1 (Type 1)
```

```
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL:   using IOMMU type 1 (Type 1)
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: Ignore mapping IO port bar(2) addr: 3021
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: Ignore mapping IO port bar(2) addr: 3021
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: Ignore mapping IO port bar(2) addr:
3021
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: Ignore mapping IO port bar(2) addr:
3021
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:04:00.1 on NUMA
socket 0
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:04:00.1 on NUMA
socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: PCI device 0000:04:00.1 on NUMA socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: PCI device 0000:04:00.1 on NUMA socket 0
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: Ignore mapping IO port bar(2) addr: 3001
Apr 19 13:02:59 baremetal ovs-ctl[91483]: EAL: Ignore mapping IO port bar(2) addr: 3001
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: Ignore mapping IO port bar(2) addr:
3001
Apr 19 13:02:59 baremetal ovs-vswitchd[91509]: EAL: Ignore mapping IO port bar(2) addr:
3001
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL: PCI device 0000:05:00.0 on NUMA socket 0
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL: PCI device 0000:05:00.0 on NUMA socket 0
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:05:00.0 on NUMA
socket 0
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:05:00.0 on NUMA
socket 0
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL: PCI device 0000:05:00.1 on NUMA socket 0
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL: PCI device 0000:05:00.1 on NUMA socket 0
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-ctl[91483]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:05:00.1 on NUMA
socket 0
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL: PCI device 0000:05:00.1 on NUMA
socket 0
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-vswitchd[91509]: EAL:   probe driver: 8086:154d net_ixgbe
Apr 19 13:03:00 baremetal ovs-ctl[91483]: [  OK  ]
Apr 19 13:03:00 baremetal ovs-ctl[91483]: [  OK  ]
Apr 19 13:03:00 baremetal ovs-ctl[91483]: Enabling remote OVSDB managers [  OK  ]
Apr 19 13:03:00 baremetal ovs-ctl[91483]: Enabling remote OVSDB managers [  OK  ]
Apr 19 13:03:00 baremetal systemd[1]: Started Open vSwitch Forwarding Unit.
Apr 19 13:03:00 baremetal systemd[1]: Started Open vSwitch Forwarding Unit.
[root@baremetal bonding]#
```

8.3.2. Configure LACP Bond

1. Add the bond:

```
[root@baremetal bonding]# ovs-vsctl add-br ovsbr0  -- set bridge ovsbr0
datapath_type=netdev
[root@baremetal bonding]# ovs-vsctl add-bond ovsbr0 dpdkbond dpdk0 dpdk1
bond_mode=balance-tcp     lacp=active -- set
interface dpdk0 type=dpdk     -- set Interface dpdk1 type=dpdk
```

2. Verify from Open vSwitch:

```
[root@baremetal bonding]# ovs-appctl lacp/show dpdkbond
---- dpdkbond ----
        status: active negotiated
        sys_id: a0:36:9f:e3:dd:c8
        sys_priority: 65534
        aggregation key: 1
        lacp_time: slow

slave: dpdk0: current attached
        port_id: 2
        port_priority: 65535
        may_enable: true

        actor sys_id: a0:36:9f:e3:dd:c8
        actor sys_priority: 65534
        actor port_id: 2
        actor port_priority: 65535
        actor key: 1
        actor state: activity aggregation synchronized collecting distributing

        partner sys_id: 14:18:77:89:9a:8a
        partner sys_priority: 32768
        partner port_id: 203
        partner port_priority: 32768
        partner key: 1
        partner state: activity timeout aggregation synchronized collecting distributing

slave: dpdk1: current attached
        port_id: 1
        port_priority: 65535
        may_enable: true

        actor sys_id: a0:36:9f:e3:dd:c8
        actor sys_priority: 65534
        actor port_id: 1
        actor port_priority: 65535
        actor key: 1
        actor state: activity aggregation synchronized collecting distributing

        partner sys_id: 14:18:77:89:9a:8a
        partner sys_priority: 32768
        partner port_id: 208
        partner port_priority: 32768
        partner key: 1
        partner state: activity timeout aggregation synchronized collecting distributing

[root@baremetal bonding]# ovs-appctl bond/show dpdkbond
---- dpdkbond ----
```

```
bond_mode: balance-tcp
bond may use recirculation: yes, Recirc-ID : 1
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
next rebalance: 6817 ms
lacp_status: negotiated
active slave mac: a0:36:9f:e3:dd:c8(dpdk0)

slave dpdk0: enabled
        active slave
        may_enable: true

slave dpdk1: enabled
        may_enable: true
```

3. Verify from the switch:

```
S4048-ON-sw#show lacp 1
Port-channel 1 admin up, oper up, mode lacp
LACP Fast Switch-Over Disabled
Actor   System ID:  Priority 32768, Address 1418.7789.9a8a
Partner System ID:  Priority 65534, Address a036.9fe3.ddc8
Actor Admin Key 1, Oper Key 1, Partner Oper Key 1, VLT Peer Oper Key 1
LACP LAG 1 is an aggregatable link
LACP LAG 1 is a normal LAG

A - Active LACP, B - Passive LACP, C - Short Timeout, D - Long Timeout
E - Aggregatable Link, F - Individual Link, G - IN_SYNC, H - OUT_OF_SYNC
I - Collection enabled, J - Collection disabled, K - Distribution enabled
L - Distribution disabled, M - Partner Defaulted, N - Partner Non-defaulted,
O - Receiver is in expired state, P - Receiver is not in expired state

Port Te 1/2 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ADEGIKNP Key 1 Priority 65535

Port Te 1/7 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ADEGIKNP Key 1 Priority 65535
S4048-ON-sw#
```

8.3.3. Enabling / Disabling Ports from OVS

You can enable or disable ports with **ovs-ofctl mod-port <bridge> <port> [up|down]**

1. Shut down a port:

```
[root@baremetal bonding]#  ovs-ofctl mod-port ovsbr0 dpdk1 down
```

2. Verify the shutdown:

```
[root@baremetal bonding]# ovs-appctl lacp/show dpdkbond
---- dpdkbond ----
        status: active negotiated
        sys_id: a0:36:9f:e3:dd:c8
        sys_priority: 65534
        aggregation key: 1
        lacp_time: slow

slave: dpdk0: current attached
        port_id: 2
        port_priority: 65535
        may_enable: true

        actor sys_id: a0:36:9f:e3:dd:c8
        actor sys_priority: 65534
        actor port_id: 2
        actor port_priority: 65535
        actor key: 1
        actor state: activity aggregation synchronized collecting distributing

        partner sys_id: 14:18:77:89:9a:8a
        partner sys_priority: 32768
        partner port_id: 203
        partner port_priority: 32768
        partner key: 1
        partner state: activity timeout aggregation synchronized collecting distributing

slave: dpdk1: defaulted detached
        port_id: 1
        port_priority: 65535
        may_enable: false

        actor sys_id: a0:36:9f:e3:dd:c8
        actor sys_priority: 65534
        actor port_id: 1
        actor port_priority: 65535
        actor key: 1
        actor state: activity aggregation defaulted

        partner sys_id: 00:00:00:00:00:00
        partner sys_priority: 0
        partner port_id: 0
        partner port_priority: 0
        partner key: 0
        partner state:
[root@baremetal bonding]# ovs-appctl bond/show dpdkbond
---- dpdkbond ----
bond_mode: balance-tcp
bond may use recirculation: yes, Recirc-ID : 1
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
next rebalance: 3315 ms
lacp_status: negotiated
```

```
active slave mac: a0:36:9f:e3:dd:c8(dpdk0)

slave dpdk0: enabled
        active slave
        may_enable: true

slave dpdk1: disabled
        may_enable: false
```

3. Verify on the switch:

```
S4048-ON-sw#show lacp 1
Port-channel 1 admin up, oper up, mode lacp
LACP Fast Switch-Over Disabled
Actor   System ID:  Priority 32768, Address 1418.7789.9a8a
Partner System ID:  Priority 65534, Address a036.9fe3.ddc8
Actor Admin Key 1, Oper Key 1, Partner Oper Key 1, VLT Peer Oper Key 1
LACP LAG 1 is an aggregatable link
LACP LAG 1 is a normal LAG

A - Active LACP, B - Passive LACP, C - Short Timeout, D - Long Timeout
E - Aggregatable Link, F - Individual Link, G - IN_SYNC, H - OUT_OF_SYNC
I - Collection enabled, J - Collection disabled, K - Distribution enabled
L - Distribution disabled, M - Partner Defaulted, N - Partner Non-defaulted,
O - Receiver is in expired state, P - Receiver is not in expired state

Port Te 1/2 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ADEGIKNP Key 1 Priority 65535

Port Te 1/7 is disabled, LACP is disabled and mode is lacp
Port State: Not in Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEHJLNP Key 1 Priority 32768
  Partner is not present
```

4. Re-enable the port:

```
[root@baremetal bonding]#  ovs-ofctl mod-port ovsbr0 dpdk1 up
```

5. Verify from RHEL:

```
[root@baremetal bonding]# ovs-appctl bond/show dpdkbond
---- dpdkbond ----
bond_mode: balance-tcp
bond may use recirculation: yes, Recirc-ID : 1
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
next rebalance: 7846 ms
lacp_status: negotiated
active slave mac: a0:36:9f:e3:dd:c8(dpdk0)
```

```
slave dpdk0: enabled
        active slave
        may_enable: true

slave dpdk1: enabled
        may_enable: true

[root@baremetal bonding]# ovs-appctl lacp/show dpdkbond
---- dpdkbond ----
        status: active negotiated
        sys_id: a0:36:9f:e3:dd:c8
        sys_priority: 65534
        aggregation key: 1
        lacp_time: slow

slave: dpdk0: current attached
        port_id: 2
        port_priority: 65535
        may_enable: true

        actor sys_id: a0:36:9f:e3:dd:c8
        actor sys_priority: 65534
        actor port_id: 2
        actor port_priority: 65535
        actor key: 1
        actor state: activity aggregation synchronized collecting distributing

        partner sys_id: 14:18:77:89:9a:8a
        partner sys_priority: 32768
        partner port_id: 203
        partner port_priority: 32768
        partner key: 1
        partner state: activity timeout aggregation synchronized collecting distributing

slave: dpdk1: current attached
        port_id: 1
        port_priority: 65535
        may_enable: true

        actor sys_id: a0:36:9f:e3:dd:c8
        actor sys_priority: 65534
        actor port_id: 1
        actor port_priority: 65535
        actor key: 1
        actor state: activity aggregation synchronized collecting distributing

        partner sys_id: 14:18:77:89:9a:8a
        partner sys_priority: 32768
        partner port_id: 208
        partner port_priority: 32768
        partner key: 1
        partner state: activity timeout aggregation synchronized collecting distributing
```

6. Verify from the switch:

```
S4048-ON-sw#show lacp 1
Port-channel 1 admin up, oper up, mode lacp
LACP Fast Switch-Over Disabled
Actor   System ID:  Priority 32768, Address 1418.7789.9a8a
Partner System ID:  Priority 65534, Address a036.9fe3.ddc8
Actor Admin Key 1, Oper Key 1, Partner Oper Key 1, VLT Peer Oper Key 1
LACP LAG 1 is an aggregatable link
LACP LAG 1 is a normal LAG

A - Active LACP, B - Passive LACP, C - Short Timeout, D - Long Timeout
E - Aggregatable Link, F - Individual Link, G - IN_SYNC, H - OUT_OF_SYNC
I - Collection enabled, J - Collection disabled, K - Distribution enabled
L - Distribution disabled, M - Partner Defaulted, N - Partner Non-defaulted,
O - Receiver is in expired state, P - Receiver is not in expired state

Port Te 1/2 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ADEGIKNP Key 1 Priority 65535

Port Te 1/7 is enabled, LACP is enabled and mode is lacp
Port State: Bundle
  Actor   Admin: State ACEHJLMP Key 1 Priority 32768
          Oper: State ACEGIKNP Key 1 Priority 32768
  Partner Admin: State BDFHJLMP Key 0 Priority 0
          Oper: State ADEGIKNP Key 1 Priority 65535
```

CHAPTER 9. DEPLOYING DIFFERENT BOND MODES WITH OVS DPDK

Use this procedure to deploy different bond modes with OVS-DPDK in Red Hat OpenStack Platform.

9.1. SOLUTION

Make the following changes to the **compute.yaml** environment file. Note that this example also sets the MTU value to 2000.

```
(...)
          -
            type: ovs_user_bridge
            name: br-link
            mtu: 2000
            use_dhcp: false
            members:
              -
                type: ovs_dpdk_bond
                name: dpdkbond0
                ovs_options: "bond_mode=balance-slb"
                mtu: 2000
                ovs_extra:
                  - set interface dpdk0 mtu_request=$MTU
                  - set interface dpdk1 mtu_request=$MTU
                members:
                  -
                    type: ovs_dpdk_port
                    name: dpdk0
                    members:
                      -
                        type: interface
                        name: p1p2
                  -
                    type: ovs_dpdk_port
                    name: dpdk1
                    members:
                      -
                        type: interface
                        name: p1p1
(...)
```

Deploy or redeploy the overcloud with the template changes made above. When complete, perform the following steps on an overcloud node.

Verify the **os-net-config** configuration:

```
cat /etc/os-net-config/config.json | python -m json.tool
(...)
      {
          "members": [
              {
                  "members": [
                      {
```

```
                    "members": [
                        {
                            "name": "p1p2",
                            "type": "interface"
                        }
                    ],
                    "name": "dpdk0",
                    "type": "ovs_dpdk_port"
                },
                {
                    "members": [
                        {
                            "name": "p1p1",
                            "type": "interface"
                        }
                    ],
                    "name": "dpdk1",
                    "type": "ovs_dpdk_port"
                }
            ],
            "mtu": 2000,
            "name": "dpdkbond0",
            "ovs_extra": [
                "set interface dpdk0 mtu_request=$MTU",
                "set interface dpdk1 mtu_request=$MTU"
            ],
            "ovs_options": "bond_mode=balance-slb",
            "type": "ovs_dpdk_bond"
        }
    ],
    "mtu": 2000,
    "name": "br-link",
    "type": "ovs_user_bridge",
    "use_dhcp": false
},
(...)
```

Verify the bond:

```
[root@overcloud-compute-0 ~]# ovs-appctl bond/show dpdkbond0
---- dpdkbond0 ----
bond_mode: balance-slb
bond may use recirculation: no, Recirc-ID : -1
bond-hash-basis: 0
updelay: 0 ms
downdelay: 0 ms
next rebalance: 9221 ms
lacp_status: off
active slave mac: a0:36:9f:e5:da:82(dpdk1)

slave dpdk0: enabled
    may_enable: true

slave dpdk1: enabled
    active slave
    may_enable: true
```

-

CHAPTER 10. RECEIVING THE COULD NOT OPEN NETWORK DEVICE DPDK0 (NO SUCH DEVICE) IN OVS-VSCTL SHOW MESSAGE

10.1. SYMPTOM

You receive the **Could not open network device dpdk0 (No such device) in ovs-vsctl show** message.

10.2. DIAGNOSIS

Red Hat supports a subset of the Poll Mode Drivers (PMDs) listed in DPDK Supported Hardware . Red Hat disabled unsupported PMDs in August of 2017.

Upstream PMDs might have security or performance issues. Therefore, a PMD needs to go through significant testing to pass Red Hat's qualification tests.

You can see a list of all enabled PMDs in /usr/share/doc/openvswitch-<version>/README.DPDK-PMDS. This list might contain PMDs not supported by Red Hat. Poll Mode Drivers not listed in **README.DPDK-PMDS** are not supported.

10.3. SOLUTION

The following example shows the supported PMDs for openvswitch-2.6.1:

```
[root@overcloud-compute-0 ~]# cat /usr/share/doc/openvswitch-2.6.1/README.DPDK-PMDS
DPDK drivers included in this package:

E1000
ENIC
I40E
IXGBE
RING
VIRTIO

For more information about the drivers, see
http://dpdk.org/doc/guides-16.11/nics/index.html
```

This example shows the supported PMDs for openvswitch-2.9.0:

```
[root@undercloud-r430 ~]# cat /usr/share/doc/openvswitch-2.9.0/README.DPDK-PMDS
DPDK drivers included in this package:

BNXT
E1000
ENIC
FAILSAFE
I40E
IXGBE
MLX4
MLX4_GLUE
MLX5
MLX5_GLUE
```

NFP
RING
SOFTNIC
VIRTIO

For more information about the drivers, see
http://dpdk.org/doc/guides-17.11/nics/index.html

CHAPTER 11. INSUFFICIENT FREE HOST MEMORY PAGES AVAILABLE TO ALLOCATE GUEST RAM WITH OPEN VSWITCH DPDK

11.1. SYMPTOM

You deploy an instance onto a compute node with sufficient huge pages and other resources, and you see output such as the following example:

```
[stack@undercloud-4 ~]$ nova show 1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc
(...)
| fault                        | {"message": "Exceeded maximum number of retries. Exceeded max
scheduling attempts 3
for instance 1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc. Last exception: internal error: process exited
while connecting to monitor:
2017-11-23T19:53:20.311446Z qemu-kvm: -chardev pty,id=cha", "code": 500, "details": "  File
\"/usr/lib/python2.7/site-packages
/nova/conductor/manager.py\", line 492, in build_instances |
|                              |     filter_properties, instances[0].uuid)
|
|                              |   File \"/usr/lib/python2.7/site-packages/nova/scheduler/utils.py\", line 184, in
populate_retry
|
|                              |     raise exception.MaxRetriesExceeded(reason=msg)
|
|                              | ", "created": "2017-11-23T19:53:22Z"}
(...)
```

And **/var/log/containers/nova/nova-compute.log** on the compute node gives the following ERROR message:

```
2017-11-23 19:53:21.021 153615 ERROR nova.compute.manager [instance: 1b72e7a1-c298-4c92-
8d2c-0a9fe886e9bc]
2017-11-23T19:53:20.477183Z qemu-kvm: -object memory-backend-file,id=ram-
node0,prealloc=yes,mem-path=/dev/hugepages/libvirt
/qemu/7-instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind:
os_mem_prealloc: Insufficient free host memory
pages available to allocate guest RAM
```

Additionally, libvirt creates the following log file:

```
[root@overcloud-compute-1 qemu]# cat instance-00000006.log
2017-11-23 19:53:02.145+0000: starting up libvirt version: 3.2.0, package: 14.el7_4.3 (Red Hat, Inc.
<http://bugzilla.redhat.com/bugzilla>, 2017-08-22-08:54:01, x86-039.build.eng.bos.redhat.com),
qemu version: 2.9.0(qemu-
kvm-rhev-2.9.0-10.el7), hostname: overcloud-compute-1.localdomain
LC_ALL=C PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin QEMU_AUDIO_DRV=none
/usr/libexec/qemu-kvm -name
guest=instance-00000006,debug-threads=on -S -object
secret,id=masterKey0,format=raw,file=/var/lib/libvirt/qemu/domain-
5-instance-00000006/master-key.aes -machine pc-i440fx-rhel7.4.0,accel=kvm,usb=off,dump-guest-
core=off -cpu
SandyBridge,vme=on,hypervisor=on,arat=on,tsc_adjust=on,xsaveopt=on -m 512 -realtime mlock=off
```

```
-smp
1,sockets=1,cores=1,threads=1 -object memory-backend-file,id=ram-node0,prealloc=yes,mem-
path=/dev/hugepages/libvirt/qemu/5
-instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind -numa
node,nodeid=0,cpus=0,memdev=ram-node0 -uuid
1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc -smbios 'type=1,manufacturer=Red
Hat,product=OpenStack
Compute,version=14.0.8-5.el7ost,serial=4f88fcca-0cd3-4e19-8dc4-4436a54daff8,uuid=1b72e7a1-
c298-4c92-8d2c
-0a9fe886e9bc,family=Virtual Machine' -no-user-config -nodefaults -chardev
socket,id=charmonitor,path=/var/lib/libvirt
/qemu/domain-5-instance-00000006/monitor.sock,server,nowait -mon
chardev=charmonitor,id=monitor,mode=control -rtc
base=utc,driftfix=slew -global kvm-pit.lost_tick_policy=delay -no-hpet -no-shutdown -boot strict=on -
device piix3
-usb-uhci,id=usb,bus=pci.0,addr=0x1.0x2 -drive file=/var/lib/nova/instances/1b72e7a1-c298-4c92-
8d2c-0a9fe886e9bc
/disk,format=qcow2,if=none,id=drive-virtio-disk0,cache=none -device virtio-blk-
pci,scsi=off,bus=pci.0,addr=0x4,drive=drive-
virtio-disk0,id=virtio-disk0,bootindex=1 -chardev
socket,id=charnet0,path=/var/run/openvswitch/vhu9758ef15-d2 -netdev vhost-
user,chardev=charnet0,id=hostnet0 -device virtio-net-
pci,netdev=hostnet0,id=net0,mac=fa:16:3e:d6:89:65,bus=pci.0,addr=0x3
-add-fd set=0,fd=29 -chardev file,id=charserial0,path=/dev/fdset/0,append=on -device isa-
serial,chardev=charserial0,id=serial0
-chardev pty,id=charserial1 -device isa-serial,chardev=charserial1,id=serial1 -device usb-
tablet,id=input0,bus=usb.0,port=1
-vnc 172.16.2.8:2 -k en-us -device cirrus-vga,id=video0,bus=pci.0,addr=0x2 -device virtio-balloon-
pci,id=balloon0,bus=pci.0,addr=0x5 -msg timestamp=on
2017-11-23T19:53:03.217386Z qemu-kvm: -chardev pty,id=charserial1: char device redirected to
/dev/pts/3 (label charserial1)
2017-11-23T19:53:03.359799Z qemu-kvm: -object memory-backend-file,id=ram-
node0,prealloc=yes,mem-path=/dev/hugepages/libvirt
/qemu/5-instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind:
os_mem_prealloc: Insufficient free host memory
pages available to allocate guest RAM

2017-11-23 19:53:03.630+0000: shutting down, reason=failed
2017-11-23 19:53:10.052+0000: starting up libvirt version: 3.2.0, package: 14.el7_4.3 (Red Hat, Inc.
<http://bugzilla.redhat.com/bugzilla>, 2017-08-22-08:54:01, x86-039.build.eng.bos.redhat.com),
qemu version: 2.9.0(qemu-
kvm-rhev-2.9.0-10.el7), hostname: overcloud-compute-1.localdomain
LC_ALL=C PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin QEMU_AUDIO_DRV=none
/usr/libexec/qemu-kvm -name
guest=instance-00000006,debug-threads=on -S -object
secret,id=masterKey0,format=raw,file=/var/lib/libvirt/qemu/domain-
6-instance-00000006/master-key.aes -machine pc-i440fx-rhel7.4.0,accel=kvm,usb=off,dump-guest-
core=off -cpu
SandyBridge,vme=on,hypervisor=on,arat=on,tsc_adjust=on,xsaveopt=on -m 512 -realtime mlock=off
-smp
1,sockets=1,cores=1,threads=1 -object memory-backend-file,id=ram-node0,prealloc=yes,mem-
path=/dev/hugepages/libvirt/qemu/6-
instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind -numa
node,nodeid=0,cpus=0,memdev=ram-node0 -uuid
1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc -smbios 'type=1,manufacturer=Red
Hat,product=OpenStack
```

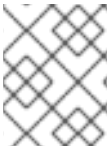
```
Compute,version=14.0.8-5.el7ost,serial=4f88fcca-0cd3-4e19-8dc4-4436a54daff8,uuid=1b72e7a1-
c298-4c92-8d2c-
0a9fe886e9bc,family=Virtual Machine' -no-user-config -nodefaults -chardev
socket,id=charmonitor,path=/var/lib/libvirt
/qemu/domain-6-instance-00000006/monitor.sock,server,nowait -mon
chardev=charmonitor,id=monitor,mode=control -rtc
base=utc,driftfix=slew -global kvm-pit.lost_tick_policy=delay -no-hpet -no-shutdown -boot strict=on -
device piix3
-usb-uhci,id=usb,bus=pci.0,addr=0x1.0x2 -drive file=/var/lib/nova/instances/1b72e7a1-c298-4c92-
8d2c-0a9fe886e9bc
/disk,format=qcow2,if=none,id=drive-virtio-disk0,cache=none -device virtio-blk-
pci,scsi=off,bus=pci.0,addr=0x4,drive=drive-
virtio-disk0,id=virtio-disk0,bootindex=1 -chardev
socket,id=charnet0,path=/var/run/openvswitch/vhu9758ef15-d2 -netdev vhost-
user,chardev=charnet0,id=hostnet0 -device virtio-net-
pci,netdev=hostnet0,id=net0,mac=fa:16:3e:d6:89:65,bus=pci.0,addr=0x3
-add-fd set=0,fd=29 -chardev file,id=charserial0,path=/dev/fdset/0,append=on -device isa-
serial,chardev=charserial0,id=serial0
-chardev pty,id=charserial1 -device isa-serial,chardev=charserial1,id=serial1 -device usb-
tablet,id=input0,bus=usb.0,port=1
-vnc 172.16.2.8:2 -k en-us -device cirrus-vga,id=video0,bus=pci.0,addr=0x2 -device virtio-balloon-
pci,id=balloon0,bus=pci.0,addr=0x5 -msg timestamp=on
2017-11-23T19:53:11.466399Z qemu-kvm: -chardev pty,id=charserial1: char device redirected to
/dev/pts/3 (label charserial1)
2017-11-23T19:53:11.729226Z qemu-kvm: -object memory-backend-file,id=ram-
node0,prealloc=yes,mem-path=/dev/hugepages/libvirt
/qemu/6-instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind:
os_mem_prealloc: Insufficient free host memory
pages available to allocate guest RAM

2017-11-23 19:53:12.159+0000: shutting down, reason=failed
2017-11-23 19:53:19.370+0000: starting up libvirt version: 3.2.0, package: 14.el7_4.3 (Red Hat, Inc.
<http://bugzilla.redhat.com/bugzilla>, 2017-08-22-08:54:01, x86-039.build.eng.bos.redhat.com),
qemu version: 2.9.0(qemu-
kvm-rhev-2.9.0-10.el7), hostname: overcloud-compute-1.localdomain
LC_ALL=C PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin QEMU_AUDIO_DRV=none
/usr/libexec/qemu-kvm -name
guest=instance-00000006,debug-threads=on -S -object
secret,id=masterKey0,format=raw,file=/var/lib/libvirt/qemu/domain-
7-instance-00000006/master-key.aes -machine pc-i440fx-rhel7.4.0,accel=kvm,usb=off,dump-guest-
core=off -cpu
SandyBridge,vme=on,hypervisor=on,arat=on,tsc_adjust=on,xsaveopt=on -m 512 -realtime mlock=off
-smp
1,sockets=1,cores=1,threads=1 -object memory-backend-file,id=ram-node0,prealloc=yes,mem-
path=/dev/hugepages/libvirt/qemu/7-
instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind -numa
node,nodeid=0,cpus=0,memdev=ram-node0 -uuid
1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc -smbios 'type=1,manufacturer=Red
Hat,product=OpenStack
Compute,version=14.0.8-5.el7ost,serial=4f88fcca-0cd3-4e19-8dc4-4436a54daff8,uuid=1b72e7a1-
c298-4c92-8d2c
-0a9fe886e9bc,family=Virtual Machine' -no-user-config -nodefaults -chardev
socket,id=charmonitor,path=/var/lib/libvirt
/qemu/domain-7-instance-00000006/monitor.sock,server,nowait -mon
chardev=charmonitor,id=monitor,mode=control -rtc
base=utc,driftfix=slew -global kvm-pit.lost_tick_policy=delay -no-hpet -no-shutdown -boot strict=on -
```

```
device piix3-
usb-uhci,id=usb,bus=pci.0,addr=0x1.0x2 -drive file=/var/lib/nova/instances/1b72e7a1-c298-4c92-
8d2c-0a9fe886e9bc
/disk,format=qcow2,if=none,id=drive-virtio-disk0,cache=none -device virtio-blk-
pci,scsi=off,bus=pci.0,addr=0x4,drive=drive-
virtio-disk0,id=virtio-disk0,bootindex=1 -chardev
socket,id=charnet0,path=/var/run/openvswitch/vhu9758ef15-d2 -netdev vhost-
user,chardev=charnet0,id=hostnet0 -device virtio-net-
pci,netdev=hostnet0,id=net0,mac=fa:16:3e:d6:89:65,bus=pci.0,addr=0x3
-add-fd set=0,fd=29 -chardev file,id=charserial0,path=/dev/fdset/0,append=on -device isa-
serial,chardev=charserial0,id=serial0
-chardev pty,id=charserial1 -device isa-serial,chardev=charserial1,id=serial1 -device usb-
tablet,id=input0,bus=usb.0,port=1
-vnc 172.16.2.8:2 -k en-us -device cirrus-vga,id=video0,bus=pci.0,addr=0x2 -device virtio-balloon-
pci,id=balloon0,bus=pci.0,addr=0x5 -msg timestamp=on
2017-11-23T19:53:20.311446Z qemu-kvm: -chardev pty,id=charserial1: char device redirected to
/dev/pts/3 (label charserial1)
2017-11-23T19:53:20.477183Z qemu-kvm: -object memory-backend-file,id=ram-
node0,prealloc=yes,mem-path=/dev/hugepages/libvirt
/qemu/7-instance-00000006,share=yes,size=536870912,host-nodes=0,policy=bind:
os_mem_prealloc: Insufficient free host memory
pages available to allocate guest RAM

2017-11-23 19:53:20.724+0000: shutting down, reason=failed
```

11.2. DIAGNOSIS

Without additional settings, nova does not know that a certain amount of huge page memory is used by other processes. By default, nova assumes that all huge page memory is available for instances. Nova will first fill up NUMA node 0 if it believes that there are still free pCPUs and free hugepage memory on this NUMA node. This issue can occur due to the following causes:

- The requested pCPUs still fit into NUMA 0
- The combined memory of all existing instances plus the memory of the instance to be spawned still fit into NUMA node 0
- Another process such as OVS holds a certain amount of hugepage memory on NUMA node 0



NOTE

Ensure you allocate a flavor RAM amount equal to the huge page multiplier, to avoid an **[Errno 12] Cannot allocate memory** error.

11.2.1. Diagnostic Steps

1. Check **meminfo**. The following show a hypervisor with 2MB hugepages and 512 free hugepages per NUMA node:

```
[root@overcloud-compute-1 ~]# cat /sys/devices/system/node/node*/meminfo  | grep -i huge
Node 0 AnonHugePages:      2048 kB
Node 0 HugePages_Total:  1024
Node 0 HugePages_Free:    512
Node 0 HugePages_Surp:      0
Node 1 AnonHugePages:      2048 kB
```

```
Node 1 HugePages_Total:  1024
Node 1 HugePages_Free:    512
Node 1 HugePages_Surp:      0
```

2. Check the NUMA architecture:

```
[root@overcloud-compute-1 nova]# lscpu  | grep -i NUMA
NUMA node(s):          2
NUMA node0 CPU(s):     0-3
NUMA node1 CPU(s):     4-7
```

3. Check the huge pages reserved by OVS. In the following output, OVS reserves 512MB of huge pages per NUMA node:

```
[root@overcloud-compute-1 virt]# ovs-vsctl list Open_vSwitch | grep mem
other_config        : {dpdk-init="true", dpdk-lcore-mask="3", dpdk-socket-mem="512,512",
pmd-cpu-mask="1e"}
```

4. Deploy instances with the following flavor (1 vCPU and 512 MB or memory):

```
[stack@undercloud-4 ~]$ nova flavor-show m1.tiny
+----------------------------+-----------------------------------------------------------------+
| Property                   | Value                                                           |
+----------------------------+-----------------------------------------------------------------+
| OS-FLV-DISABLED:disabled   | False                                                           |
| OS-FLV-EXT-DATA:ephemeral  | 0                                                               |
| disk                       | 8                                                               |
| extra_specs                | {"hw:cpu_policy": "dedicated", "hw:mem_page_size": "large"}     |
| id                         | 49debbdb-c12e-4435-97ef-f575990b352f                            |
| name                       | m1.tiny                                                         |
| os-flavor-access:is_public | True                                                            |
| ram                        | 512                                                             |
| rxtx_factor                | 1.0                                                             |
| swap                       |                                                                 |
| vcpus                      | 1                                                               |
+----------------------------+-----------------------------------------------------------------+
```

The new instance will boot and will use memory from NUMA 1:

```
[stack@undercloud-4 ~]$ nova list | grep d98772d1-119e-48fa-b1d9-8a68411cba0b
| d98772d1-119e-48fa-b1d9-8a68411cba0b | cirros-test0 | ACTIVE | -          | Running     |
provider1=2000:10::f816:3eff:fe8d:a6ef, 10.0.0.102 |
```

```
[root@overcloud-compute-1 nova]# cat /sys/devices/system/node/node*/meminfo  | grep -i
huge
Node 0 AnonHugePages:      2048 kB
Node 0 HugePages_Total:  1024
Node 0 HugePages_Free:      0
Node 0 HugePages_Surp:      0
Node 1 AnonHugePages:      2048 kB
Node 1 HugePages_Total:  1024
Node 1 HugePages_Free:    256
Node 1 HugePages_Surp:      0
```

```
nova boot --nic net-id=$NETID --image cirros --flavor m1.tiny --key-name id_rsa cirros-test0
```

This instance fails to boot:

```
[stack@undercloud-4 ~]$ nova list
+--------------------------------------+-------------+--------+------------+-------------+------------------
----------------------+
| ID                                   | Name        | Status | Task State | Power State | Networks
|
+--------------------------------------+-------------+--------+------------+-------------+------------------
----------------------+
| 1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc | cirros-test0 | ERROR  | -          | NOSTATE     |
|
| a44c43ca-49ad-43c5-b8a1-543ed8ab80ad | cirros-test0 | ACTIVE | -          | Running     |
provider1=2000:10::f816:3eff:fe0f:565b, 10.0.0.105 |
| e21ba401-6161-45e6-8a04-6c45cef4aa3e | cirros-test0 | ACTIVE | -          | Running     |
provider1=2000:10::f816:3eff:fe69:18bd, 10.0.0.111 |
+--------------------------------------+-------------+--------+------------+-------------+------------------
----------------------+
```

5. From the compute node, check that free huge pages on NUMA Node 0 are exhausted. There is, however, enough space on NUMA node 1:

```
[root@overcloud-compute-1 qemu]# cat /sys/devices/system/node/node*/meminfo  | grep -i
huge
Node 0 AnonHugePages:      2048 kB
Node 0 HugePages_Total:  1024
Node 0 HugePages_Free:      0
Node 0 HugePages_Surp:      0
Node 1 AnonHugePages:      2048 kB
Node 1 HugePages_Total:  1024
Node 1 HugePages_Free:    512
Node 1 HugePages_Surp:      0
```

6. The information in **/var/log/containers/nova/nova-compute.log** reveals that the instance CPU is pinned to NUMA node 0:

```
<name>instance-00000006</name>
  <uuid>1b72e7a1-c298-4c92-8d2c-0a9fe886e9bc</uuid>
  <metadata>
    <nova:instance xmlns:nova="http://openstack.org/xmlns/libvirt/nova/1.0">
      <nova:package version="14.0.8-5.el7ost"/>
      <nova:name>cirros-test0</nova:name>
      <nova:creationTime>2017-11-23 19:53:00</nova:creationTime>
      <nova:flavor name="m1.tiny">
        <nova:memory>512</nova:memory>
        <nova:disk>8</nova:disk>
        <nova:swap>0</nova:swap>
        <nova:ephemeral>0</nova:ephemeral>
        <nova:vcpus>1</nova:vcpus>
      </nova:flavor>
      <nova:owner>
        <nova:user uuid="5d1785ee87294a6fad5e2bdddd91cc20">admin</nova:user>
        <nova:project uuid="8c307c08d2234b339c504bfdd896c13e">admin</nova:project>
      </nova:owner>
```

```
    <nova:root type="image" uuid="6350211f-5a11-4e02-a21a-cb1c0d543214"/>
  </nova:instance>
</metadata>
<memory unit='KiB'>524288</memory>
<currentMemory unit='KiB'>524288</currentMemory>
<memoryBacking>
  <hugepages>
    <page size='2048' unit='KiB' nodeset='0'/>
  </hugepages>
</memoryBacking>
<vcpu placement='static'>1</vcpu>
<cputune>
  <shares>1024</shares>
  <vcpupin vcpu='0' cpuset='2'/>
  <emulatorpin cpuset='2'/>
</cputune>
<numatune>
  <memory mode='strict' nodeset='0'/>
  <memnode cellid='0' mode='strict' nodeset='0'/>
</numatune>
```

In the **numatune** section, nodeset="0" indicates that memory will be claimed from NUMA 0.

11.3. SOLUTION

Administrators can input the amount of huge page memory not used by instances into nova.

```
[root@overcloud-compute-1 virt]# grep reserved_huge /var/lib/config-data/puppet-
generated/nova_libvirt/etc/nova/nova.conf -B1
[DEFAULT]
reserved_huge_pages=node:0,size:2048,count:512
reserved_huge_pages=node:1,size:2048,count:512
```

The size parameter is the huge page size in KiB. The count parameter is the number of huge pages that are used by OVS per NUMA node. For example, for 4096 of socket memory used by Open vSwitch, use the following values:

```
[DEFAULT]
reserved_huge_pages=node:0,size:1GB,count:4
reserved_huge_pages=node:1,size:1GB,count:4
```

See How to set reserved_huge_pages in /etc/nova/nova.conf in Red Hat OpenStack Platform 10 for details about how to implement this with OpenStack director.

```
reserved_huge_pages = None

(Unknown) Number of huge/large memory pages to reserved per NUMA host cell.

Possible values:

    A list of valid key=value which reflect NUMA node ID, page size (Default unit is KiB) and number of
pages to be reserved.

    reserved_huge_pages = node:0,size:2048,count:64 reserved_huge_pages =
```

```
node:1,size:1GB,count:1
```

In this example we are reserving on NUMA node 0 64 pages of 2MiB and on NUMA node 1 1 page of 1GiB.

<<<<<<< HEAD With debug enabled in **/var/lib/config-data/puppet-generated/nova_libvirt/etc/nova/nova.conf**, you should see the following in the logs after a restart of the **openstack-nova-compute** docker container:

With debug enabled in **/etc/nova/nova.conf**, you should see the following information in the logs after a restart of **openstack-nova-compute**: >>>>>>> cc5a5fd7d... Complete Chapter 11 edits from peer review

```
[root@overcloud-compute-1 virt]# docker restart nova_compute
(...)
[root@overcloud-compute-1 virt]# grep reserved_huge_pages /var/log/containers/nova/nova-compute.log | tail -n1
2017-12-19 17:56:40.727 26691 DEBUG oslo_service.service [req-e681e97d-7d99-4ba8-bee7-5f7a3f655b21 - - - - -]
reserved_huge_pages            = [{'node': '0', 'count': '512', 'size': '2048'}, {'node': '1', 'count': '512', 'size': '2048'}] log_opt_values /usr/lib/python2.7/site-packages/oslo_config/cfg.py:2622
[root@overcloud-compute-1 virt]#
```

= Troubleshoot OVS DPDK PMD CPU Usage with perf and Collect and Send the Troubleshooting Data

1. Prerequisites Use the steps in this section to install troubleshooting tools.
2. Install **perf** on the compute node:

```
yum install perf -y
```

3. Install Open vSwitch debug RPMs:

```
subscription-manager repos --enable=rhel-7-server-openstack-13-debug-rpms
```

4. Install sysstat (needed for the **pidstat** command):

```
yum install sysstat -y
```

== Diagnosis Use the steps in this section to troubleshoot and collect data.

=== PMD Threads

1. Determine the location of PMD threads:

```
IFS=$'\n' ; for I in $(ps -T -p `pidof ovs-vswitchd` | grep pmd);do PID=`echo $I | awk '{print $2}'`; PMD=`echo $I | awk
'{print $NF}'` ; PCPU=`taskset -c -p $PID | awk '{print $NF}'` ; echo "$PMD with PID $PID in on pCPU $PCPU"; done
```

For example:

```
[root@overcloud-compute-1 ~]# IFS=$'\n' ; for I in $(ps -T -p `pidof ovs-vswitchd` | grep pmd);do PID=`echo $I | awk
'{print $2}'`; PMD=`echo $I | awk '{print $NF}'` ; PCPU=`taskset -c -p $PID | awk '{print $NF}'`
```

```
; echo "$PMD with PID $PID in on pCPU
$PCPU"; done
pmd545 with PID 412314 in on pCPU 2
pmd555 with PID 412315 in on pCPU 4
pmd550 with PID 412316 in on pCPU 6
pmd551 with PID 412317 in on pCPU 8
pmd553 with PID 412318 in on pCPU 22
pmd554 with PID 412319 in on pCPU 24
pmd549 with PID 412320 in on pCPU 26
pmd556 with PID 412321 in on pCPU 28
pmd546 with PID 412322 in on pCPU 3
pmd548 with PID 412323 in on pCPU 5
pmd547 with PID 412324 in on pCPU 23
pmd552 with PID 412325 in on pCPU 25
```

2. While reproducing the issue, run perf record and perf report and save the output.

   - Create the script **gather_perf_data_a.sh**:

```
cat<<'EOF'>>gather_perf_data_a.sh
#!/bin/bash -x
IFS=$'\n' ;
dir_name=/tmp/perf_record_a
mkdir ${dir_name}
rm -f ${dir_name}/*

for l in $(ps -T -p `pidof ovs-vswitchd` | grep pmd);do PID=`echo $l | awk '{print $2}'`;
PMD=`echo $l | awk '{print $NF}'` ; PCPU=`taskset -c -p $PID | awk '{print $NF}'` ; echo
"$PMD with PID $PID in on pCPU $PCPU"; done > ${dir_name}/pmds.txt

for l in $(ps -T -p `pidof ovs-vswitchd` | grep pmd);do
  PID=`echo $l | awk '{print $2}'`;
  PMD=`echo $l | awk '{print $NF}'` ;
  PCPU=`taskset -c -p $PID | awk '{print $NF}'` ;
  echo "$PMD with PID $PID in on pCPU $PCPU";
  date
  perf record -C $PCPU -g -o perf_record_-g_$PCPU sleep 60 &
done

sleep 80

for l in $(ps -T -p `pidof ovs-vswitchd` | grep pmd);do
  PID=`echo $l | awk '{print $2}'`;
  PMD=`echo $l | awk '{print $NF}'` ;
  PCPU=`taskset -c -p $PID | awk '{print $NF}'` ;
  echo "$PMD with PID $PID in on pCPU $PCPU";
  date
  perf record -C $PCPU -o perf_record_$PCPU sleep 60 &
done

sleep 80

for f in perf_record_-g_*;do
  perf report -g -i $f | cat > ${dir_name}/perf_report_$f.txt ;
  rm -f $f
done
```

```
for f in perf_record_*;do
  perf report -i $f | cat > ${dir_name}/perf_report_$f.txt ;
  rm -f $f
done

archive_name="${dir_name}_`hostname`_`date '+%F_%H%m%S'`.tar.gz"
tar -czf $archive_name ${dir_name}
echo "Archived all data in archive ${archive_name}"
EOF
```

- Run the script:

```
chmod +x gather_perf_data_a.sh
./gather_perf_data_a.sh
```

The report can be read using **perf report -i ${archive_name}**. If this is for a case that was opened with Red Hat support, attach the resulting tar archive to the case.

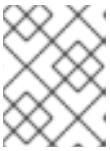
=== Additional Data

1. Create the script **gather_perf_data_b.sh** to collect additional data:

```
cat<<'EOF'>>gather_perf_data_b.sh
#!/bin/bash -x
dir_name=/tmp/perf_record_b
mkdir ${dir_name}
rm -f ${dir_name}/*

date > ${dir_name}/pidstat1.txt
pidstat -u -t -p `pidof ovs-vswitchd`,`pidof ovsdb-server` 5 12 >> ${dir_name}/pidstat1.txt &
perf record -p `pidof ovs-vswitchd` -g --call-graph dwarf sleep 60

sleep 20

date > ${dir_name}/pidstat2.txt
pidstat -u -t -p `pidof ovs-vswitchd`,`pidof ovsdb-server` 1 60 >> ${dir_name}/pidstat2.txt

mv perf.data perf.data_openvswitch

perf script -F tid -i perf.data_openvswitch | sort -u | grep -o '[0-9]*' | xargs -n1 -I{} perf report -i
perf.data_openvswitch --no-children --percentage relative --stdio --tid {} -g none >
${dir_name}/perf_reports.txt
perf script -F tid -i perf.data_openvswitch | sort -u | grep -o '[0-9]*' | xargs -n1 -I{} perf report -i
perf.data_openvswitch --no-children --percentage relative --stdio --tid {}  >
${dir_name}/perf_reports_callgraph.txt

rm -f perf.data_openvswitch

archive_name="${dir_name}_`hostname`_`date '+%F_%H%m%S'`.tar.gz"
tar -czf $archive_name ${dir_name}
echo "Archived all data in archive ${archive_name}"
EOF
```

2. Execute the script:

```
chmod +x gather_perf_data_b.sh
./gather_perf_data_b.sh
```



NOTE

Make sure that there is sufficient disk space. The 'perf.data' file can take up several Gigabytes of disk space.

If this is for a Red Hat support ticket, attach the resulting tar archive to the case.

=== Open vSwitch Logs

1. Provide all Open vSwitch (OVS) logs. Ensure that **/var** has sufficient disk space. Use **df -h** to determine free disk space on /var and **du -sh /var/log/openvswitch** to determine the total size of OVS logs.

```
tar -cvzf /var/openvswitch_`hostname`_`date +"%F_%H%M%S"`.tar.gz /var/log/openvswitch
```

2. Attach the resulting file, for example, **/var/openvswitch_overcloud-compute-0_2018-02-27_153713.tar.gz**, to the support case for analysis.

3. Generate and provide an sosreport. Ensure that **/var** has sufficient disk space. Use **df -h** to determine free disk space on **/var**.

```
sosreport --batch --all-logs
```

= Using virsh emulatorpin in virtual environments with NFV

Use this procedure to determine the impact of using virsh emulatorpin in Red Hat OpenStack Platform with NFV.

== Symptom

You experience packet loss in Red Hat OpenStack Platform {vernum} NFV environment, and have not configured emulator thread pinning.

== Solution

Use this section to investigate and configure emulator thread pinning.

=== qemu-kvm Emulator Threads

Emulator threads handle interrupt requests and non-blocking processes for virtual machine hardware emulation. Threads not running vCPUs are **qemu-kvm** emulator threads. See the following example.

```
[root@overcloud-compute-0 ~]# ps -Tp `pgrep -f instance-00000009`
   PID    SPID TTY          TIME CMD
364936  364936 ?        00:00:02 qemu-kvm
364936  364946 ?        00:00:00 qemu-kvm
364936  364952 ?        00:00:52 CPU 0/KVM
364936  364953 ?        00:00:26 CPU 1/KVM
364936  364954 ?        00:00:30 CPU 2/KVM
364936  364956 ?        00:00:00 vnc_worker
```

Due to the Linux CFS (completely fair scheduler), emulator threads normally move periodically from one pCPU to another, within the defined in libvirt's emulator pin set.

In NFV contexts, you might experience issues if you configure emulator threads when using the **isolcpus** parameter, because this kernel configuration disables the CFS scheduling on those CPUs. If you are not using the **isolcpus parameter**, you can experience packet loss when the emulator threads interrupt CPUs that are processing packets.

Examples of emulator threads include:

- qemu-kvm threads
- vnc_worker threads
- vhost-<qemu-kvm PID> kernel threads (When virtio-net is used (kernel networking on the hypervisor)

=== Default Behavior for Emulator Thread Pinning

By default, nova will configure an emulator thread pin set which spans the pCPUs assigned to all vCPUs. If you are not using the **isolcpus** parameter, then emulator threads can be scheduled on any pCPU, and will periodically move from one pCPU to another.

```
virsh dumpxml instance-0000001d
(...)
  <vcpu placement='static'>4</vcpu>
  <cputune>
    <shares>4096</shares>
    <vcpupin vcpu='0' cpuset='34'/>
    <vcpupin vcpu='1' cpuset='14'/>
    <vcpupin vcpu='2' cpuset='10'/>
    <vcpupin vcpu='3' cpuset='30'/>
    <emulatorpin cpuset='10,14,30,34'/>
  </cputune>
(...)
```

```
[root@overcloud-compute-0 ~]# virsh dumpxml instance-00000009
(...)
        <nova:vcpus>3</nova:vcpus>
  <vcpu placement='static'>3</vcpu>
    <vcpupin vcpu='0' cpuset='1'/>
    <vcpupin vcpu='1' cpuset='2'/>
    <vcpupin vcpu='2' cpuset='3'/>
(...)
<emulatorpin cpuset='1-3'/>
(...)
```

Therefore any of these CPUs can be preempted by qemu's emulator threads, risking packet drops.

For details on the current progress of new features for emulator thread pinning, see Bug 1468004 and OpenStack Change 510897

At the time of this writing, the draft specified the following thread policies:

```
Valid THREAD-POLICY values are:
```

```
- ``share``: (default) The emulator threads float across the pCPUs
  associated to the guest. To place a workload's emulator threads on
  a set of isolated physical CPUs, set ``share``` and
  ``[compute]/cpu_shared_set`` configuration option to the set of
  host CPUs that should be used for best-effort CPU resources.

- ``isolate``: The emulator threads are isolated on a single pCPU.
```

=== About the Impact of isolcpus on Emulator Thread Scheduling

When isolcpus is used, CFS scheduler is disabled and all emulator threads will run on the first available, lowest indexed pCPU. As a consequence, without intervention or further configuration, one vCPU of the instance runs a high risk for resource contention with the emulator threads.

Further details can be found at Kernel.org Bugzilla – Bug 116701.

Use the following algorithm to determine which vCPU the emulator threads are using:

```
PCPU=MIN([EMULATORPINSET])
VCPU=REVERSE_CPUSET(PCPU)

REVERSE_CPUSET :=  SELECT pcpu from `virsh dumpxml <instance name> | grep
"cpuset=$PCPU"`
```

For example, in this instance, all emulator threads and children inherit affinity 1-3 from the default emulator pin set:

```
[root@overcloud-compute-0 ~]# taskset -a -c -p `pgrep -f instance-00000009`
pid 364936's current affinity list: 1-3
pid 364946's current affinity list: 1-3
pid 364952's current affinity list: 1
pid 364953's current affinity list: 2
pid 364954's current affinity list: 3
pid 364956's current affinity list: 1-3
[root@overcloud-compute-0 ~]# ps -Tp `pgrep -f instance-00000009`
   PID    SPID TTY          TIME CMD
 364936  364936 ?       00:00:02 qemu-kvm
 364936  364946 ?       00:00:00 qemu-kvm
 364936  364952 ?       00:00:51 CPU 0/KVM
 364936  364953 ?       00:00:26 CPU 1/KVM
 364936  364954 ?       00:00:30 CPU 2/KVM
 364936  364956 ?       00:00:00 vnc_worker
[root@overcloud-compute-0 ~]# pgrep -f vhost- | xargs -I {} taskset -a -c -p {}
pid 364948's current affinity list: 1-3
pid 364949's current affinity list: 1-3
pid 364950's current affinity list: 1-3
[root@overcloud-compute-0 ~]#
```

In combination with isolcpus, all emulator threads and the vhost-* threads execute on pCPU1 and are never rescheduled:

```
cat /proc/sched_debug | sed '/^cpu#/,/^runnable/{//!d}' | grep vhost -C3
(...)
cpu#1, 2099.998 MHz
runnable tasks:
```

```
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
     watchdog/1    11        -2.995579    410285     0         0.000000      5025.887998         0.000000 0 /
     migration/1    12         0.000000        79     0         0.000000         3.375060         0.000000 0 /
     ksoftirqd/1    13   5172444.259776        54   120         0.000000         0.570500         0.000000 0 /
     kworker/1:0    14   5188475.472257       370   120         0.000000        14.707114         0.000000 0 /
    kworker/1:0H    15      8360.049510        10   100         0.000000         0.150151         0.000000 0 /
     kworker/1:1  2707   5045807.055876     16370   120         0.000000       793.611916         0.000000
0 /
    kworker/1:1H  2763   5187682.987749     11755   100         0.000000       191.949725
0.000000 0 /
       qemu-kvm 364936      3419.522791     50276   120         0.000000      2476.880384
0.000000 0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/emulator
       qemu-kvm 364946      1270.815296       102   120         0.000000        23.204111         0.000000
0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/emulator
      CPU 0/KVM 364952     52703.660314     53709   120         0.000000     52715.105472
0.000000 0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/vcpu0
     vnc_worker 364956       123.609634         1   120         0.000000         0.016849         0.000000 0
/machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/emulator
    vhost-364936 364948      3410.527677      1039   120         0.000000        84.254772         0.000000
0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/emulator
    vhost-364936 364949      3407.341502        55   120         0.000000         2.894394         0.000000 0
/machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/emulator
    vhost-364936 364950      3410.395220       174   120         0.000000        10.969077         0.000000
0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/emulator

cpu#2, 2099.998 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
     watchdog/2    16        -5.995418    410285     0         0.000000      5197.571153         0.000000 0 /
     migration/2    17         0.000000        79     0         0.000000         3.384688         0.000000 0 /
     ksoftirqd/2    18        -7.031102         3   120         0.000000         0.019079         0.000000 0 /
     kworker/2:0    19         0.119413        39   120         0.000000         0.588589         0.000000 0 /
    kworker/2:0H    20        -1.047613         8   100         0.000000         0.086272         0.000000 0 /
     kworker/2:1  2734   1475469.236026     11322   120         0.000000       241.388582         0.000000
0 /
      CPU 1/KVM 364953     27258.370583     33294   120         0.000000     27269.017017
0.000000 0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/vcpu1

cpu#3, 2099.998 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
     watchdog/3    21        -5.996592    410285     0         0.000000      4970.777439         0.000000 0 /
     migration/3    22         0.000000        79     0         0.000000         3.886799         0.000000 0 /
     ksoftirqd/3    23        -7.035295         3   120         0.000000         0.014677         0.000000 0 /
     kworker/3:0    24        17.758583        38   120         0.000000         0.637152         0.000000 0 /
    kworker/3:0H    25        -1.047727         8   100         0.000000         0.077141         0.000000 0 /
     kworker/3:1 362530    154177.523420        83   120         0.000000         6.544285         0.000000 0
/
      CPU 2/KVM 364954     32456.061889     25966   120         0.000000     32466.719084
0.000000 0 /machine.slice/machine-qemu\x2d6\x2dinstance\x2d00000009.scope/vcpu2
```

=== Optimal Location of Emulator Threads

This section provides descriptions for placing emulator threads on the following networks:

- DPDK networking within the instance and netdev datapath in Open vSwitch
- DPDK networking within the instance, system datapath in Open vSwitch and kernel space networking on the hypervisor
- Kernel networking within the instance and netdev datapath in Open vSwitch

==== Optimal Placement of Emulator Threads with DPDK Networking Within the Instance and netdev datapath in Open vSwitch

If DPDK runs within the instance, packet processing is done entirely in the user space. Do not schedule PMDs to run on vCPU0, as this should remain for the OS and interrupt handling. Because PMD CPUs within the instance run an active loop and need 100% of the CPU, they should not be preempted. Packet loss can occur if one of these vCPUs is preempted. Thus, the emulatorpin cpuset needs to be configured in such a way that it does not overlap with the physical CPUs that handle the virtual CPUs numbered 1 and above.

With DPDK networking within the instance, the optimal location for emulator threads is either the pCPU that is handling vCPU 0 or a dedicated physical CPU that is not handling any virtual CPUs at all.

If OVS-DPDK is used on the hypervisor and DPDK within the instance, place the emulator thread on vCPU 0.

==== Optimal Placement of Emulator Threads with DPDK Networking Within the Instance and System datapath in Open vSwitch

If kernel space networking is used on the hypervisor, then packet processing on the hypervisor is executed within the kernel.

With DPDK networking within the instance, the optimal location for emulator threads is either the pCPU that is handling vCPU 0, or a dedicated physical CPU that is not handling any virtual CPUs.

Note that in this scenario, packet processing for the vNIC queues is executed within **vhost-<qemu-kvm PID>** kernel threads of the hypervisor. Under high traffic, these kernel threads can generate a significant CPU load. The optimal location of the emulator threads needs to be determined on a case-by-case basis.

```
[root@overcloud-compute-0 ~]# ps aux | grep vhost-
root     364948  0.0  0.0      0     0 ?        S    20:32   0:00 [vhost-364936]
root     364949  0.0  0.0      0     0 ?        S    20:32   0:00 [vhost-364936]
root     364950  0.0  0.0      0     0 ?        S    20:32   0:00 [vhost-364936]
```

==== Optimal Placement of Emulator Threads with Kernel Networking within the Instance and netdev datapath in Open vSwitch

With kernel networking within the instance, there are two options:

- Optimize the interrupt distribution, for example, softirqs within the instance. In such a case, you do not have to allocate an additional pCPU for emulator threads and can assign the emulator threads to a pCPU that is not handling any network interrupts.
- Use a dedicated pCPU on the same NUMA node for emulator threads.

Due to the complexity of the first option, the second option is recommended.

== Diagnosis

=== The Demonstration Environment

The demonstration environment runs one instance: **instance-0000001d**. Its associated qemu-kvm thread has the following PID:

```
[root@overcloud-compute-0 ~]# pidof qemu-kvm
73517
```

=== How Emulatorpin works

By default, a Red Hat OpenStack Platform deployment uses the following settings:

```
virsh dumpxml instance-0000001d
(...)
  <vcpu placement='static'>4</vcpu>
  <cputune>
    <shares>4096</shares>
    <vcpupin vcpu='0' cpuset='34'/>
    <vcpupin vcpu='1' cpuset='14'/>
    <vcpupin vcpu='2' cpuset='10'/>
    <vcpupin vcpu='3' cpuset='30'/>
    <emulatorpin cpuset='10,14,30,34'/>
  </cputune>
(...)
```

This leads to an unpredictable allocation of the emulator threads, such as qemu-kvm, vnc_worker, and so on:

```
[root@overcloud-compute-0 ~]# ps -T -p 73517
   PID    SPID TTY          TIME CMD
  73517   73517 ?        00:00:00 qemu-kvm
  73517   73527 ?        00:00:00 qemu-kvm
  73517   73535 ?        00:00:06 CPU 0/KVM
  73517   73536 ?        00:00:02 CPU 1/KVM
  73517   73537 ?        00:00:03 CPU 2/KVM
  73517   73538 ?        00:00:02 CPU 3/KVM
  73517   73540 ?        00:00:00 vnc_worker
[root@overcloud-compute-0 ~]# taskset -apc 73517
pid 73517's current affinity list: 10,14,30,34
pid 73527's current affinity list: 10,14,30,34
pid 73535's current affinity list: 34
pid 73536's current affinity list: 14
pid 73537's current affinity list: 10
pid 73538's current affinity list: 30
pid 73540's current affinity list: 10,14,30,34
```

```
[root@overcloud-compute-0 ~]# virsh vcpupin instance-0000001d | awk '$NF~/[0-9]+/ {print $NF}' |
sort -n | while read CPU; do sed '/cpu#/,/runnable task/{//!d}' /proc/sched_debug | sed -n
"/^cpu#${CPU},/,/^$/p" ; done
cpu#10, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
```

```
    migration/10    64         0.000000       107     0         0.000000        90.232791         0.000000 0 /
    ksoftirqd/10    65       -13.045337         3   120         0.000000         0.004679         0.000000 0 /
    kworker/10:0    66       -12.892617        40   120         0.000000         0.157359         0.000000 0 /
   kworker/10:0H    67        -9.320550         8   100         0.000000         0.015065         0.000000 0 /
    kworker/10:1 17996      9695.675528        23   120         0.000000         0.222805         0.000000 0 /
       qemu-kvm 73517      1994.534332     27105   120         0.000000       886.203254         0.000000
0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
       qemu-kvm 73527       722.347466        84   120         0.000000        18.236155         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
      CPU 2/KVM 73537      3356.749162     18051   120         0.000000      3370.045619
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu2
     vnc_worker 73540       354.007735         1   120         0.000000         0.047002         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
        worker 74584      1970.499537         5   120         0.000000         0.130143         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
        worker 74585      1970.492700         4   120         0.000000         0.071887         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
        worker 74586      1982.467246         3   120         0.000000         0.033604         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
        worker 74587      1994.520768         1   120         0.000000         0.076039         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
        worker 74588      2006.500153         1   120         0.000000         0.004878         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator

cpu#14, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
-------------------------------------------------------------------------------------------------------------
    migration/14    88         0.000000       107     0         0.000000        90.107596         0.000000 0 /
    ksoftirqd/14    89       -13.045376         3   120         0.000000         0.004782         0.000000 0 /
    kworker/14:0    90       -12.921990        40   120         0.000000         0.128166         0.000000 0 /
   kworker/14:0H    91        -9.321186         8   100         0.000000         0.016870         0.000000 0 /
    kworker/14:1 17999      6247.571171         5   120         0.000000         0.028576         0.000000 0 /
      CPU 1/KVM 73536      2274.381281      6679   120         0.000000      2287.691654         0.000000
0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu1

cpu#30, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
-------------------------------------------------------------------------------------------------------------
    migration/30   180         0.000000       107     0         0.000000        89.206960         0.000000 0 /
    ksoftirqd/30   181       -13.045892         3   120         0.000000         0.003828         0.000000 0 /
    kworker/30:0   182       -12.929272        40   120         0.000000         0.120754         0.000000 0 /
   kworker/30:0H   183        -9.321056         8   100         0.000000         0.018042         0.000000 0 /
    kworker/30:1 18012      6234.935501         5   120         0.000000         0.026505         0.000000 0 /
      CPU 3/KVM 73538      2474.183301     12595   120         0.000000      2487.479666
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu3

cpu#34, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
-------------------------------------------------------------------------------------------------------------
    migration/34   204         0.000000       107     0         0.000000        89.067908         0.000000 0 /
    ksoftirqd/34   205       -13.046824         3   120         0.000000         0.002884         0.000000 0 /
    kworker/34:0   206       -12.922407        40   120         0.000000         0.127423         0.000000 0 /
   kworker/34:0H   207        -9.320822         8   100         0.000000         0.017381         0.000000 0 /
```

```
    kworker/34:1 18016     10788.797590         7   120         0.000000         0.042631         0.000000 0 /
      CPU 0/KVM 73535      5969.227225     14233   120         0.000000      5983.425363
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu0
```

The emulator threads can be moved by using virsh emulatorpin:

```
virsh emulatorpin instance-0000001d 34
```

Note that the affinity for all non-CPU threads changes.

```
[root@overcloud-compute-0 ~]# ps -T -p 73517
   PID    SPID TTY          TIME CMD
 73517   73517 ?        00:00:00 qemu-kvm
 73517   73527 ?        00:00:00 qemu-kvm
 73517   73535 ?        00:00:06 CPU 0/KVM
 73517   73536 ?        00:00:02 CPU 1/KVM
 73517   73537 ?        00:00:03 CPU 2/KVM
 73517   73538 ?        00:00:02 CPU 3/KVM
 73517   73540 ?        00:00:00 vnc_worker
[root@overcloud-compute-0 ~]# taskset -apc 73517
pid 73517's current affinity list: 34
pid 73527's current affinity list: 34
pid 73535's current affinity list: 34
pid 73536's current affinity list: 14
pid 73537's current affinity list: 10
pid 73538's current affinity list: 30
pid 73540's current affinity list: 34
```

Note the number of switches in the historic data in **/proc/sched_debug**. In the following example, PID 73517 already moved to cpu#34. The other emulator workers did not run since the last output, and therefore still show on cpu#10:

```
[root@overcloud-compute-0 ~]# virsh vcpupin instance-0000001d | awk '$NF~/[0-9]+/ {print $NF}' |
sort -n | while read CPU; do sed '/cpu#/,/runnable task/{//!d}' /proc/sched_debug | sed -n
"/^cpu#${CPU},/,/^$/p" ; done
cpu#10, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
    migration/10    64         0.000000       107     0         0.000000        90.232791         0.000000 0 /
    ksoftirqd/10    65       -13.045337         3   120         0.000000         0.004679         0.000000 0 /
    kworker/10:0    66       -12.892617        40   120         0.000000         0.157359         0.000000 0 /
   kworker/10:0H    67        -9.320550         8   100         0.000000         0.015065         0.000000 0 /
    kworker/10:1 17996      9747.429082        26   120         0.000000         0.255547         0.000000 0 /
       qemu-kvm 73527       722.347466        84   120         0.000000        18.236155         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
      CPU 2/KVM 73537      3424.520709     21610   120         0.000000      3437.817166
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu2
     vnc_worker 73540       354.007735         1   120         0.000000         0.047002         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator

cpu#14, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
```

```
    migration/14    88         0.000000       107     0         0.000000        90.107596         0.000000 0 /
    ksoftirqd/14    89       -13.045376         3   120         0.000000         0.004782         0.000000 0 /
    kworker/14:0    90       -12.921990        40   120         0.000000         0.128166         0.000000 0 /
   kworker/14:0H    91        -9.321186         8   100         0.000000         0.016870         0.000000 0 /
    kworker/14:1 17999      6247.571171         5   120         0.000000         0.028576         0.000000 0 /
       CPU 1/KVM 73536      2283.094453      7028   120         0.000000      2296.404826         0.000000
0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu1

cpu#30, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
    migration/30   180         0.000000       107     0         0.000000        89.206960         0.000000 0 /
    ksoftirqd/30   181       -13.045892         3   120         0.000000         0.003828         0.000000 0 /
    kworker/30:0   182       -12.929272        40   120         0.000000         0.120754         0.000000 0 /
   kworker/30:0H   183        -9.321056         8   100         0.000000         0.018042         0.000000 0 /
    kworker/30:1 18012      6234.935501         5   120         0.000000         0.026505         0.000000 0 /
       CPU 3/KVM 73538      2521.828931     14047   120         0.000000      2535.125296
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu3

cpu#34, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
    migration/34   204         0.000000       107     0         0.000000        89.067908         0.000000 0 /
    ksoftirqd/34   205       -13.046824         3   120         0.000000         0.002884         0.000000 0 /
    kworker/34:0   206       -12.922407        40   120         0.000000         0.127423         0.000000 0 /
   kworker/34:0H   207        -9.320822         8   100         0.000000         0.017381         0.000000 0 /
    kworker/34:1 18016     10788.797590         7   120         0.000000         0.042631         0.000000 0 /
        qemu-kvm 73517         2.613794     27706   120         0.000000       941.839262         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
       CPU 0/KVM 73535      5994.533905     15169   120         0.000000      6008.732043
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu0
```

Note how thread 73517 moves to cpu#34. If you now interact with a VNC session, you can see that /proc/sched_debug shows the vnc_worker threads on cpu#34 as well.

```
[root@overcloud-compute-0 ~]# virsh vcpupin instance-0000001d | awk '$NF~/[0-9]+/ {print $NF}' |
sort -n | while read CPU; do sed '/cpu#/,/runnable task/{//!d}' /proc/sched_debug | sed -n
"/^cpu#${CPU},/,/^$/p" ; done
cpu#10, 2197.477 MHz
runnable tasks:
            task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
----------------------------------------------------------------------------------------------------------
    migration/10    64         0.000000       107     0         0.000000        90.232791         0.000000 0 /
    ksoftirqd/10    65       -13.045337         3   120         0.000000         0.004679         0.000000 0 /
    kworker/10:0    66       -12.892617        40   120         0.000000         0.157359         0.000000 0 /
   kworker/10:0H    67        -9.320550         8   100         0.000000         0.015065         0.000000 0 /
    kworker/10:1 17996      9963.300958        27   120         0.000000         0.273007         0.000000 0 /
        qemu-kvm 73527       722.347466        84   120         0.000000        18.236155         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
       CPU 2/KVM 73537      3563.793234     26162   120         0.000000      3577.089691
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu2

cpu#14, 2197.477 MHz
runnable tasks:
```

```
          task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
-----------------------------------------------------------------------------------------------------------
    migration/14    88         0.000000       107     0         0.000000        90.107596         0.000000 0 /
    ksoftirqd/14    89       -13.045376         3   120         0.000000         0.004782         0.000000 0 /
    kworker/14:0    90       -12.921990        40   120         0.000000         0.128166         0.000000 0 /
   kworker/14:0H    91        -9.321186         8   100         0.000000         0.016870         0.000000 0 /
    kworker/14:1 17999      6247.571171         5   120         0.000000         0.028576         0.000000 0 /
       CPU 1/KVM 73536      2367.789075      9648   120         0.000000      2381.099448         0.000000
0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu1

cpu#30, 2197.477 MHz
runnable tasks:
          task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
-----------------------------------------------------------------------------------------------------------
    migration/30   180         0.000000       107     0         0.000000        89.206960         0.000000 0 /
    ksoftirqd/30   181       -13.045892         3   120         0.000000         0.003828         0.000000 0 /
    kworker/30:0   182       -12.929272        40   120         0.000000         0.120754         0.000000 0 /
   kworker/30:0H   183        -9.321056         8   100         0.000000         0.018042         0.000000 0 /
    kworker/30:1 18012      6234.935501         5   120         0.000000         0.026505         0.000000 0 /
       CPU 3/KVM 73538      2789.628278     24788   120         0.000000      2802.924643
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu3

cpu#34, 2197.477 MHz
runnable tasks:
          task   PID         tree-key  switches  prio     wait-time             sum-exec        sum-sleep
-----------------------------------------------------------------------------------------------------------
    migration/34   204         0.000000       107     0         0.000000        89.067908         0.000000 0 /
    ksoftirqd/34   205       -13.046824         3   120         0.000000         0.002884         0.000000 0 /
    kworker/34:0   206       -12.922407        40   120         0.000000         0.127423         0.000000 0 /
   kworker/34:0H   207        -9.320822         8   100         0.000000         0.017381         0.000000 0 /
    kworker/34:1 18016     11315.391422        25   120         0.000000         0.196078         0.000000 0 /
        qemu-kvm 73517       471.930276     30975   120         0.000000      1295.543576         0.000000
0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
       CPU 0/KVM 73535      6160.062172     19201   120         0.000000      6174.260310
0.000000 0 /machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/vcpu0
      vnc_worker 73540       459.653524        38   120         0.000000         7.535037         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
          worker 78703       449.098251         2   120         0.000000         0.120313         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
          worker 78704       449.131175         3   120         0.000000         0.066961         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
          worker 78705       461.100994         4   120         0.000000         0.022897         0.000000 0
/machine.slice/machine-qemu\x2d1\x2dinstance\x2d0000001d.scope/emulator
```