Red Hat OpenStack Platform 10

OVS-DPDK エンドツーエンドトラブルシューティングガイド

トラブルシューティング手順を OVS-DPDK エンドに含むガイド

Last Updated: 2022-08-15

Red Hat OpenStack Platform 10 OVS-DPDK エンドツーエンドトラブルシューティングガイド

トラブルシューティング手順を OVS-DPDK エンドに含むガイド

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

OVS-DPDK システム管理者が、Red Hat OpenStack Platform 10 のパケットロスに関連する一般的な問題を特定して解決する手順

目次

前書き

本書では、Red Hat OpenStack Platform 10 のパケットロスに関連する典型的な問題を特定および解決するための OVS-DPDK システム管理者向けの手順を説明します。このガイドに記載されている手順は、以前に公開されたナレッジベースの記事に優先します。

第1章 事前確認

このガイドは、次のドキュメントの計画とデプロイメントの手順に精通していることを前提としています。

- DPDK で高速化した Open vSwitch (OVS) のネットワーク設定

第2章 OVS-DPDK デプロイメントの検証

この章では、デプロイメント後に実行する検証手順について説明します。

2.1. OPENSTACK の確認

次のコマンドを使用して、OpenStack および OVS-DPDK の設定を確認します。

2.1.1. ネットワークエージェントの表示

各エージェントの **Alive** の値が **True** で、**State** が **UP** であることを確認します。問題がある場合には、**/var/log/neutron** および **/var/log/openvswitch/ovs-vswitchd.log** のログを表示して、問題を確認します。

```
$ 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. Compute Service でホストを表示する

各ホストの **Status** の値が **enabled** になっていて、**State** が **up** になっていることを確認します。問題がある場合には、**/var/log/nova** のログを確認して問題を確認してください。

```
$ 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 |
+----+------------------+-----------------------+----------+---------+-------+----------------------------+
```

2.2. コンピュートノード OVS 設定の確認

ネットワークアダプターおよび OpenvSwitch の設定およびヘルスを確認するには、以下の手順を実施します。

1. コンピュートノード上で DPDK ネットワークデバイスを確認するには、dpdk ツールをインストールします。以下のコマンドを実行します。この rpm はリポジトリーにあります: **rhel-7-server-extras-rpms**。

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

2. DPDK によって管理されているネットワークデバイスとネットワークに使用されているデバイスを表示します。

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

   DPDK ドライバーを使用するデバイスは、Tripleo コンピュートロールテンプレートのタイプ **ovs_dpdk_bond** または **ovs_dpdk_port** です。

   ```
   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. DPDK が有効であることを確認するには、以下のコマンドを実行します。

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

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

4. 以下のコマンドを実行します。結果は、DPDK 互換ドライバーの PCI デバイス (たとえば、**0000:04:00.1** および **:05:00.0**) を **type: dpdk** としてエラーなしで示しています。

```
$ 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"
```

次の出力はエラーを示しています。

```
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"
```

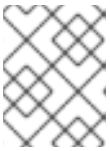
5. インターフェイスの詳細を表示するには、次のコマンドを実行します。

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

6. 以下のコマンドを実行します。lacp が有効になっていないことに注意してください。

```
$ 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. コンピュートノードの ovs ブリッジがすべて、高速データパス（ユーザー空間）ネットワーク用に **netdev** であることを確認します。



注記

システム (カーネル) と netdev (ユーザースペース) のデータパスタイプの混合はサポートされていません。

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

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

8. 次のコマンドを実行して、永続的な Open vSwitch エラーを確認します。

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

2.3. インスタンス設定の OVS の確認

vhostuser DMA が確実に機能するようにするには、OVS-DPDK ポートを使用してインスタンスを設定し、フレーバーを使用して専用の CPU と Huge Page を有効にします。詳しくは、『オーバークラウドの高度なカスタマイズ』の「ステップ 3」および「OVS-DPDK 用インスタンスのデプロイ 」を参照してください。

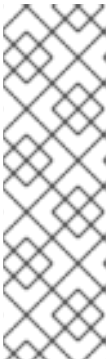
インスタンス設定を確認するには、次の手順を実行します。

1. インスタンスが CPU を固定していることを確認します。専用の CPU は、virsh を使用して特定できます。

```
$ sudo virsh vcpupin 2
```

2. インスタンスに使用されているエミュレータースレッドが、そのインスタンスに割り当てられている同じ vCPU で実行されていないことを確認します。

```
$ sudo virsh emulatorpin 2
```



注記

Red Hat OpenStack Platform 12 以降、フレーバーでエミュレーターピンを設定することができます。Configuring emulator threads policy with Red Hat OpenStack Platform 12 を参照してください。

以前のバージョンでは、インスタンスの電源をオンにする際に、エミュレータースレッドピニングを手動で行う必要があります。About the impact of using virsh emulatorpin in virtual environments with NFV, with and without isolcpus, and about optimal emulator thread pinning を参照してください。

3. インスタンスが Huge Page を使用していることを確認します。これは、最適なパフォーマンスに必要です。

```
$ sudo virsh numatune 1
```

4. インスタンスの受信キューがポーリングモードドライバー (PMD) によって処理されていることを確認します。
ポートとキューは、PMD 間で均等にバランスを取る必要があります。最適なのは、ネットワークアダプターと同じ NUMA ノードにある CPU がポートをサービスすることです。

```
$ 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. PMD の統計を表示します。これは、受信キューが PMD 間でどの程度バランスが取れているかを判断するのに役立ちます。詳細については、Open vSwitch のドキュメントの PMD Threads を参照してください。

注記

pmd-rxq-rebalance オプションが OVS 2.9.0 で追加されました。このコマンドは、最新の rxq 処理サイクル情報に基づいて PMD 間で均等にバランスを取るために、新しい PMD キュー割り当てを実行します。

pmd-stats-show コマンドは、PMD が実行されてから、または統計が最後にクリアされてからの完全な履歴を表示します。クリアされていない場合は、ポートが設定されてデータが流れる前に統計に組み込まれます。データパス (通常はそうです) の負荷を確認するために使用されている場合は、役に立ちません。

システムを定常状態にし、統計をクリアし、数秒待ってから統計を表示するのが最善です。これにより、データパスの正確なイメージが提供されます。

次のコマンドを使用して、PMD の統計を表示します。

```
$ 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. PMD 統計をリセットします。**pmd-stats-show** コマンドは、最後の **pmd-stats-clear** コマンド以降の PMD 統計を表示します。以前に発行された **pmd-stats-clear** がなかった場合は、PMD の実行が開始されてからのデータが含まれています。

負荷がかかっているシステムを調べている場合は、PMD 統計をクリアしてから、それらを表示すると便利です。そうでない場合には、（トラフィックの流れる前）システムがロードしていないとき、統計には以前の時間からのデータが含まれる可能性があります。

次のコマンドを使用して、PMD 統計をリセットします。

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

2.4. その他の役立つコマンド

これらのコマンドを使用して、追加の検証チェックを実行します。

- os-net-config によって設定された OVS-DPDK ポートと物理 NIC マッピングを検索します

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

- Nova インスタンス $ID を持つインスタンスの DPDK ポートを検索します

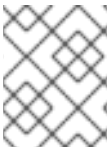
```
sudo  ovs-vsctl find interface external_ids:vm-uuid="$ID" | grep ^name
```

- DPDK ポートを使用してインスタンスの Nova ID を検索します

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

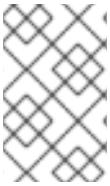
- dpdk ポートで tcpdump を実行します

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



注記

ovs-tcpdump は、**rhel-7-server-openstack-10-devtools-rpms リポジトリーにある openvswitch-** test RPM からのものです。



注記

パフォーマンス上の懸念から、実稼働環境では **ovs-tcpdump** は推奨されません。詳細は、「 How to use ovs-tcpdump on vhost-user interfaces in Red Hat OpenStack Platform? 」を参照してください。

2.5. 単純なコンピュートノードの CPU パーティショニングとメモリーチェック

前提条件

デプロイされたコンピュートノードでこのコマンドを実行し、CPU マスクが TripleO Heat Template 値にどのようにマップされるかに注意する。

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

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

以下の点に注意してください。

- **DPDK-lcore-mask** は、TripleO Heat テンプレートの **HostCpusList** にマッピングします。
- **DPDK-socket-mem** は、TripleO Heat テンプレートの **NeutronDpdkSocketMemory** にマッピングします。
- **PMD-cpu-mask** は、TripleO Heat テンプレートの **NeutronDpdkCoreList** にマッピングされます。
  これらの CPU マスクを 10 進数の値 に変換して、TripleO Heat テンプレートと実際のシステム値に戻すには、「16 進数の CPU マスクをビットマスクに変換し、マスクされた CPU を特定する方法」を参照してください。

2.5.1. CPU の検出

pid 1 の CPU を検出するには、次のコマンドを使用します。これらのコアでは、PMD または Nova vCPU を実行しないでください。

```
$ taskset -c -p 1

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

2.5.2. PMD スレッドの検出

PMD スレッドを表示するには、次のコマンドを使用します。出力には、Tripleo パラメーター **NeutronDpdkCoreList** の値が反映されているはずです。Tripleo パラメーター **HostCpusList** または **HostIsolatedCoreslist** の値は重複しないようにする必要があります。

```
$ 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. NUMA ノードの検出

最適なパフォーマンスを得るには、物理ネットワークアダプター、PMD スレッド、およびインスタンスの固定 CPU がすべて同じ NUMA ノード上にあることを確認してください。詳しくは、「CPUs and NUMA nodes」を参照してください。

以下は、NUMA 割り当てを調べるための簡単な演習です。

1. コンピュートノード上のインスタンスの vhu ポートを調べます。

```
$ sudo virsh domiflist 1

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

2. そのポートにサービスを提供している PMD スレッドを調べて、NUMA ノードに注意してください。

```
$ 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. インスタンスの物理的に固定された CPU を見つけます。たとえば、このインスタンスのポートにサービスを提供する PMD は CPU 2 にあり、インスタンスは CPU 34 および 6 によって提供されます。

```
$ sudo virsh dumpxml 1 | grep cpuset

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

4. 各 NUMA ノードのコアを調べます。インスタンス (34,6) を提供する CPU は同じ NUMA ノード (0) 上にあることに注意してください。

```
$ 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
```

さらに、OVS DPDK によって管理されていないネットワークアダプターには、それらが属する NUMA ノードを示すエントリーがここにあります。

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

または、OVS DPDK によって管理されている場合でも、PCI アドレスを照会することで、ネットワークアダプターの NUMA ノードを確認できます。

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

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

これらの演習は、PMD、インスタンス、およびネットワークアダプターがすべて NUMA 0 上にあり、パフォーマンスに最適であることを示しています。openvswitch ログ (**/var/log/openvswitch** にあります) からのクロス NUMA ポーリングの兆候については、次のようなログエントリーを探してください。

```
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. 検出された CPU の検出

次のコマンドを使用して、分離された CPU を表示します。出力は、TripleO パラメーター **HostIsolatedCoreList** の値と同じである必要があります。

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

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

2.5.5. Nova インスタンス専用 CPU の検出

次のコマンドを使用して、Nova インスタンス専用の CPU を表示します。この出力は、ポーリングモードドライバー (PMD) CPU を使用しないパラメーター isolcpus の値と同じである必要があります。

```
$ grep ^vcpu_pin_set /etc/nova/nova.conf

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

2.5.6. Huge Page 設定の確認

コンピュートノードで Huge Page の設定を確認してください。

```
[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
```

hugepages が設定されていないか、または使い切られる場合は、「ComputeKernelArgs」を参照してください。

2.6. パケットドロップの原因

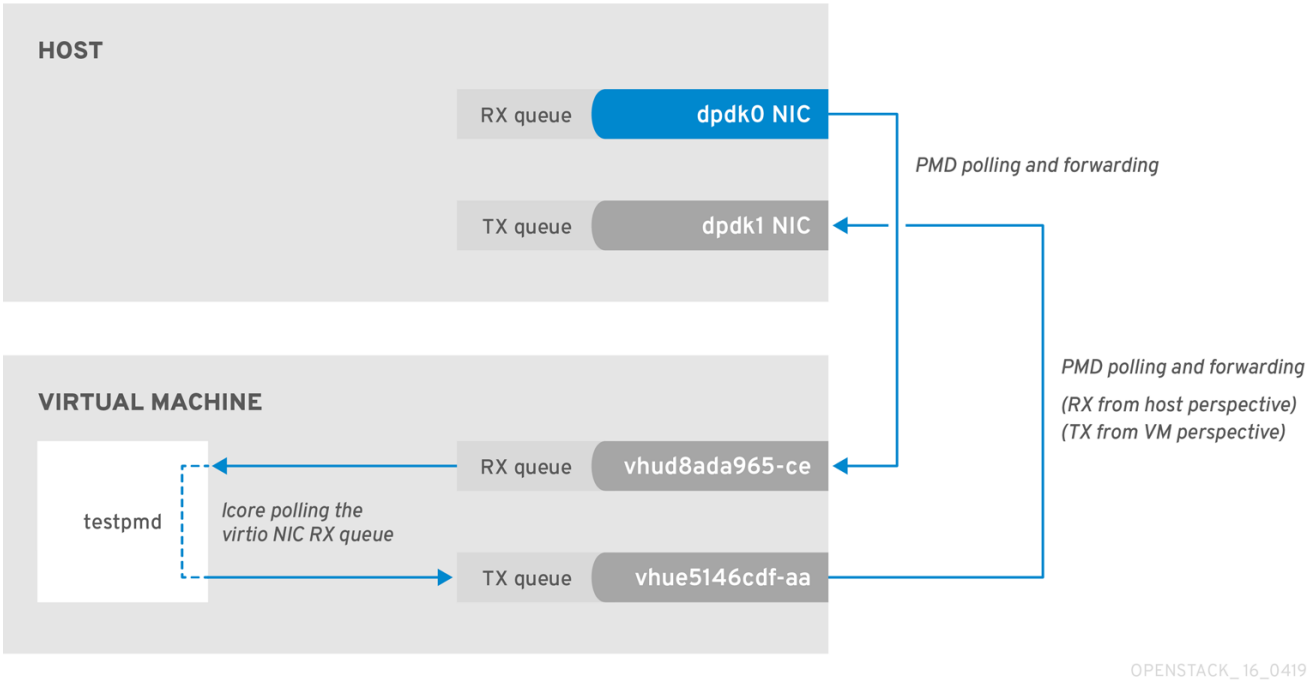
キューがいっぱいになると、通常はキューが十分に速く排出されないときに、パケットがドロップされます。ボトルネックは、キューが十分に速く排出されていないときにキューを排出することになっているエンティティーです。ほとんどの場合、ドロップカウンターは、ドロップされたパケットを追跡するために使用されます。ハードウェアまたはソフトウェアの設計にバグがあると、パケットがドロップカウンターをスキップすることがあります。

データプラン開発キット (DPDK) には、パケットを転送するための **testpmd** アプリケーションが含まれています。本章のシナリオでは、**testpmd** が仮想マシンにインストールされ、論理コア(lcore)を持つポートをポーリングして、あるポートから別のポートにパケットを転送します。**testpmd** は、テストするトラフィックジェネレーターと使用します。この場合は、物理仮想物理(PVP)パス間のスループットをテストします。

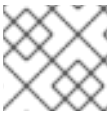
2.6.1. OVS-DPDK が遅すぎて物理 NIC をドレインできない

この例は、PMD スレッドが物理ネットワークアダプター (dpdk0) の受信 (RX) キューのポーリングを担当していることを示しています。PMD スレッドがパケット量に追いつかない場合、または中断された場合、パケットがドロップされる可能性があります。

図2.1物理アダプター RX キューのポーリング



次のコマンドは、dpdk0 インターフェイスからの統計を表示します。ovs-dpdk が物理アダプターを十分な速度で排出していないためにパケットがドロップされている場合は、**rx_dropped** の値が急速に増加していることがわかります。



注記

PMD の NUMA ノードごとに物理 CPU コアは 1 つだけにする必要があります。

```
# 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}
```

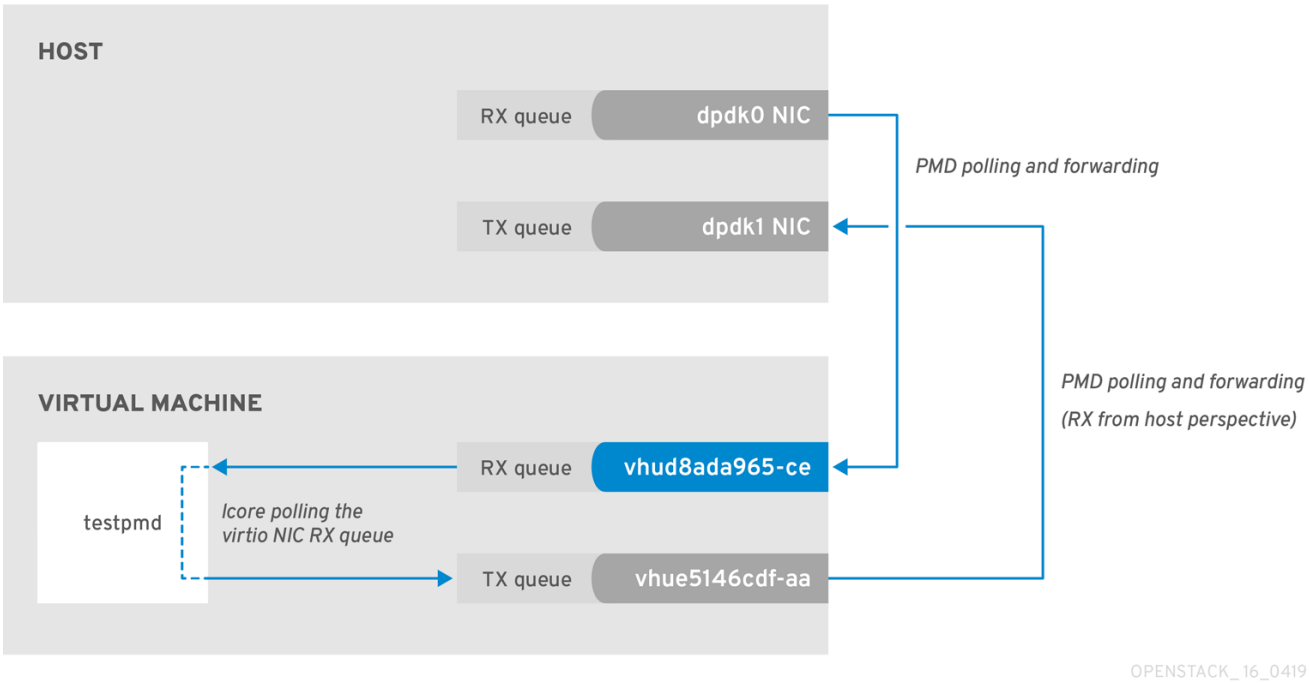
2.6.2. VM が遅すぎて vhost-user をドレインできない

この例は、図 2.1 の例と似ており、インスタンス受信 (RX) キューに送信されたパケットボリュームによって lcore スレッドが圧倒されると、パケット損失が発生する可能性があります。

詳細については、次の記事を参照してください。

- isolcpus を使用する場合と使用しない場合の、NFV を使用する仮想環境で virsh エミュレーターピンを使用することの影響、および最適なエミュレータースレッドの固定について
- Red Hat OpenStack Director を使用して OVS DPDK に接続されている virtio NICs の RX キューサイズと TX キューサイズを変更する

図2.2 仮想アダプター RX キューのポーリング



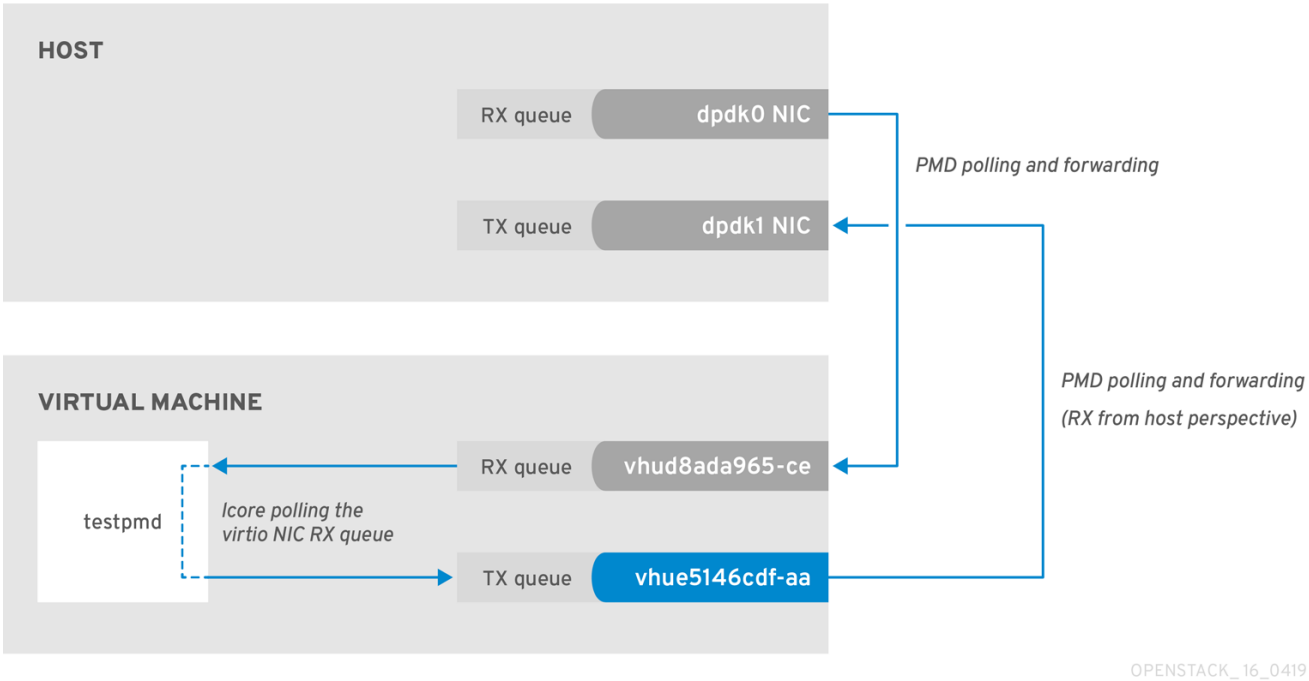
ホストの **tx_dropped** 値が VM の **rx_dropped** 値に対応するかどうかを確認するには、次のコマンドを実行します。

```
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 が遅すぎて vhost-user をドレインできない

この例では、PMD スレッドは、ホストの観点から受信キューである virtio TX をポーリングします。PMD スレッドがパケット量に圧倒されたり、中断されたりすると、パケットがドロップする可能性があります。

図2.3 仮想アダプターの TX キューのポーリング



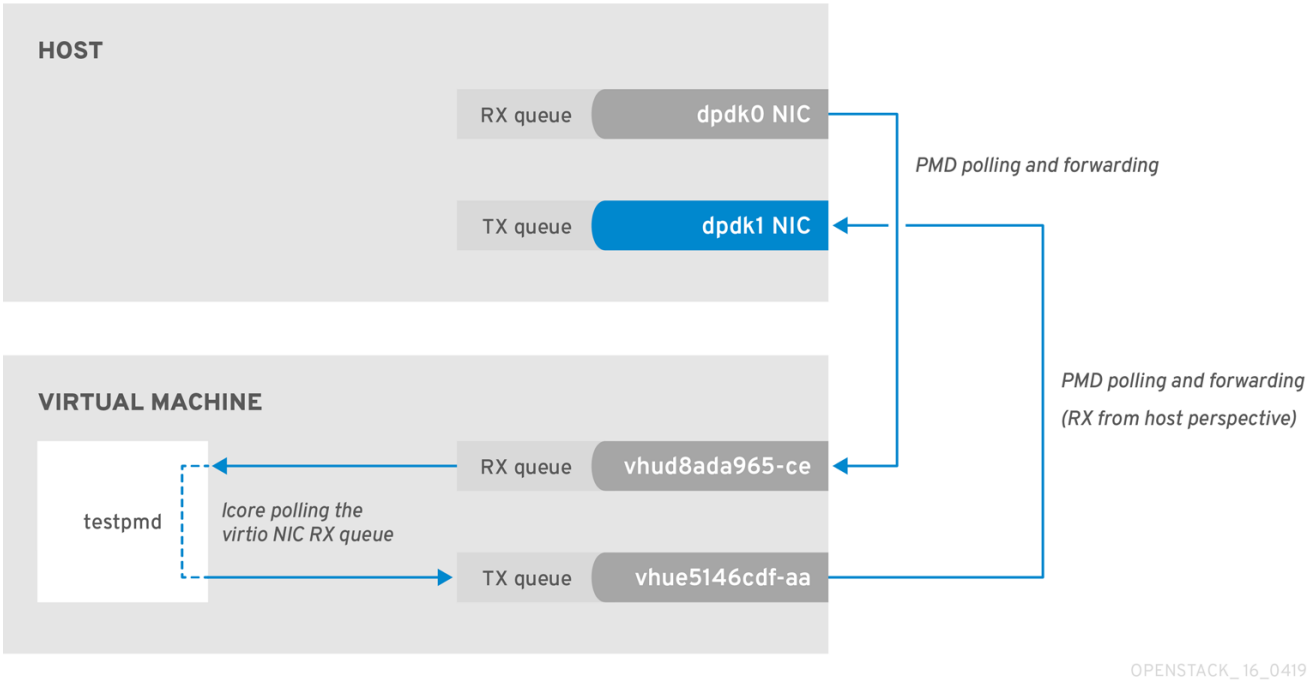
VM からのパケットのリターンパスをトレースし、ホスト (**tx_dropped**) 側と VM(**rx_dropped**) 側の両方のドロップカウンターからの値を提供し、次のコマンドを実行します。

```
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. 出力物理インターフェイスでのパケット損失

PCIe と RAM の間の転送速度が遅いと、物理アダプターが TX キューからパケットをドロップする可能性があります。これはまれですが、この問題を特定して解決する方法を知ることが重要です。

図2.4 物理アダプター TX キューのポーリング



次のコマンドは、dpdk1 インターフェイスからの統計を表示します。**tx_dropped** がゼロより大きく、急速に増加している場合は、Red Hat でサポートケースを開きます。

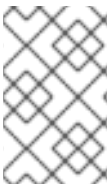
```
ovs-vsctl --column statistics list interface dpdk1

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}
```

これらのタイプのパケット損失が発生した場合は、メモリーチャネルの再設定を検討してください。

- メモリーチャネルを計算するには、『Network Functions Virtualization Planning and Conifguration Guide』の「Memory parameters」を参照してください。
- メモリーチャネルの数を確認するには、「How to determine the memory channel for NeutronDpdkMemoryChannels or OvsDpdkMemoryChannels in Red Hat OpenStack Platform」を参照してください。

第3章 NFV コマンドのチートシート

本章では、Red Hat OpenStack Platform 10 のシステム可観測性で最も一般的に使用されるコマンドの多くについて説明します。



注記

以下のコマンドの一部は、デフォルトでは使用できない場合があります。特定のノードに必要なツールをインストールするには、次のコマンドを実行します。
sudo yum install tuna qemu-kvm-tools perf kernel-tools dmidecode

3.1. UNIX ソケット

これらのコマンドを使用して、プロセスポートと UNIX ソケットドメインを表示します。

アクション	コマンド
ホスト名ルックアップなしで、すべての状態 (LISTEN、ESTABLISHED、CLOSE_WAIT など) のすべての TCP および UDP SOCKETS を表示します	**# lsof -ni**
ホスト名ルックアップなしで、すべての状態 (LISTEN、ESTABLISHED、CLOSE_WAIT など) のすべての TCP ソケットを表示します	**# lsof -nit**
ホスト名ルックアップなしで、すべての状態 (LISTEN、ESTABLISHED、CLOSE_WAIT など) のすべての UDP SOCKETS を表示します	**# lsof -niu**
IPv4 のホスト名ルックアップなしで、すべての状態 (LISTEN、ESTABLISHED、CLOSE_WAIT など) のすべての TCP および UDP SOCKETS を表示します	**# lsof -ni4**
IPv6 のホスト名ルックアップなしで、すべての状態 (LISTEN、ESTABLISHED、CLOSE_WAIT など) のすべての TCP および UDP SOCKETS を表示します	**# lsof -ni6**
特定のポートのホスト名ルックアップなしで、関連するすべてのソケット (LISTEN、ESTABLISHED、CLOSE_WAIT など) を表示します	**# lsof -ni :4789**
ホスト名ルックアップなしで LISTEN 状態のすべての SOCKETS を表示します	**# ss -ln**
IPv4 のホスト名ルックアップなしで LISTEN 状態のすべてのソケットを表示します	**# ss -ln4**
IPv6 のホスト名ルックアップなしで LISTEN 状態のすべての SOCKETS を表示する	**# ss -ln6**

3.2. IP

これらのコマンドを使用して、IP L2 および L3 設定、ドライバー、PCI バス、およびネットワーク統計を表示します。

アクション	コマンド
すべての L2 (物理および仮想の両方) インターフェイスとその統計を表示します	**# ip -s link show**
すべての L3 インターフェイスとその統計を表示します	**# ip -s addr show**
デフォルト (メイン) の IP ルーティングテーブルを表示します	**# ip route show**
特定のルーティングテーブルのルーティングルールを表示します	**# ip route show table external**
すべてのルーティングテーブルを表示します	**# ip rule show**
特定の宛先のルーティングルールを表示します	**# ip route get 1.1.1.1**
すべての Linux 名前空間を表示します	**# ip netns show**
Linux namespace にログインします	**# ip netns exec ns0 bash**
特定のインターフェイスの詳細なネットワークインターフェイスカウンターを表示します	**# tail /sys/class/net/ens6/statistics/***
特定の結合デバイスの詳細な結合情報を表示します	**# cat /proc/net/bonding/bond1**
グローバルネットワークインターフェイスのカウンタービューを表示します	**# cat /proc/net/dev**
特定のネットワークインターフェイスでサポートおよび接続されている物理接続タイプ (TP、FIBER など)、リンク速度モードを表示します	**# ethtool ens6**
特定のネットワークインターフェイスの Linux ドライバー、ドライバーバージョン、ファームウェア、および PCIe BUS ID を表示します	**# ethtool -i ens6**
特定のネットワークインターフェイスのデフォルト、有効、および無効のハードウェアオフロードを表示します	**# ethtool -k ens6**
特定のネットワークインターフェイスの MQ (マルチキュー) 設定を表示します	**# ethtool -l ens6**

アクション	コマンド
特定のネットワークインターフェイスの RX と TX の両方の MQ セットアップを変更します	**# ethtool -L ens6 combined 8**
特定のネットワークインターフェイスの TX に対してのみ MQ 設定を変更します	**# ethtool -L ens6 tx 8**
特定のネットワークインターフェイスのキューサイズを表示します	**# ethtool -g ens6**
特定のネットワークインターフェイスの RX キューサイズを変更します	**# ethtool -G ens6 rx 4096**
拡張ネットワーク統計を表示します	**# cat /proc/net/softnet_stat**
重要なネットワークデバイス情報 (インターフェイス名、MAC、NUMA、PCIe スロット、ファームウェア、カーネルドライバー) をすばやく表示します	**# biosdevname -d**
カーネルの内部ドロップカウンターを表示します。詳細は、「ネットワークデータ処理の監視」を参照してください。	**# cat /proc/net/softnet_stat**

3.3. OVS

これらのコマンドを使用して、Open vSwitch 関連の情報を表示します。

アクション	コマンド
OVS DPDK の人間が読める形式の統計	Open vSwitch DPDK 統計 を参照してください。
OVS の基本情報を表示します (バージョン、dpdk 有効化、PMD コア、lcore、ODL ブリッジマッピング、バランシング、自動バランシングなど)	**# ovs-vsctl list Open_vSwitch**
OVS グローバルスイッチングビューを表示します	**# ovs-vsctl show**
OVS にすべての詳細なインターフェイスを表示します	**# ovs-vsctl list interface**
1 つのインターフェイスの OVS 詳細 (リンク速度、MAC、ステータス、統計など) を表示します	**# ovs-vsctl list interface dpdk0**
特定のインターフェイスの OVS カウンターを表示します	**# ovs-vsctl get interface dpdk0 statistics**
OVS にすべての詳細なポートを表示します	**# ovs-vsctl list port**

アクション	コマンド
1つのポートの OVS の詳細を表示します (リンク速度、MAC、ステータス、統計など)	**# ovs-vsctl list port vhu3gf0442-00**
1つのブリッジの OVS 詳細を表示します (データパスタイプ、マルチキャストスヌーピング、stp ステータスなど)	**# ovs-vsctl list bridge br-int**
OVS ログステータスを表示します	**# ovs-appctl vlog/list**
すべての OVS ログをデバッグに変更します	**# ovs-appctl vlog/set dbg**
1つの特定の OVS サブシステムを、ファイルログ出力のデバッグモードに変更します	**# ovs-appctl vlog/set file:backtrace:dbg**
すべての OVS ログを無効にします	**# ovs-appctl vlog/set off**
すべての OVS サブシステムを、ファイルログ出力のみをデバッグするように変更します	**# ovs-appctl vlog/set file:dbg**
すべての OVS 拡張コマンドを表示します	**# ovs-appctl list-commands**
すべての OVS ボンディングを表示します	**# ovs-appctl bond/list**
特定の OVS ボンディングに関する詳細を表示します (ステータス、ボンディングモード、転送モード、LACP ステータス、ボンディングメンバー、ボンディングメンバーステータス、リンクステータス)	**# ovs-appctl bond/show bond1**
メンバー、ボンディング、パートナースイッチの高度な LACP 情報を表示します	**# ovs-appctl lacp/show**
OVS インターフェイスカウンターを表示します	**# ovs-appctl dpctl/show -s**
反復間の違いを強調する OVS インターフェイスカウンターを表示します	**# watch -d -n1 "ovs-appctl dpctl/show -s|grep -A4 -E '(dpdk|dpdkvhostuser)'|grep -v '\-\-'"**
特定のポートの OVSmempool 情報を表示します	**# ovs-appctl netdev-dpdk/get-mempool-info dpdk0**
PMD パフォーマンス統計を表示します	**# ovs-appctl dpif-netdev/pmd-stats-show**
一貫した方法で PMD パフォーマンス統計を表示します	**# ovs-appctl dpif-netdev/pmd-stats-clear && sleep 60s && ovs-appctl dpif-netdev/pmd-stats-show**
人間が読める形式の DPDK インターフェイス統計を表示します	**# ovs-vsctl get interface dpdk0 statistics|sed -e "s/,/\n/g" -e "s/[\",\{,\},]//g" -e "s/=/ ==⇒ /g"**

アクション	コマンド
ポート/キューと PMD スレッド間の OVS マッピングを表示します	**# ovs-appctl dpif-netdev/pmd-rxq-show**
OVS PMD リバランスをトリガーします (PMD サイクルの使用率に基づく)	**# ovs-appctl dpif-netdev/pmd-rxq-rebalance**
OVS ポートと特定の PMD の間にアフィニティーを作成します (PMD をバランシングから無効にします)	**# ovs-vsctl set interface dpdk other_config:pmd-rxq-affinity="0:2,1:4"**
(OVS 2.11+ および FDP18.09) サイクルに基づいて PMD バランシングを設定します	**# ovs-vsctl set Open_vSwitch . other_config:pmd-rxq-assign=cycles**
(OVS 2.11+ および FDP18.09) ラウンドロビンで PMD バランシングを設定します	**# ovs-vsctl set Open_vSwitch . other_config:pmd-rxq-assign=roundrobin**
OVS-DPDK 物理ポートキューの数を設定します	**# ovs-vsctl set interface dpdk options:n_rxq=2**
OVS-DPDK 物理ポートのキューサイズの数を設定します	**# ovs-vsctl set Interface dpdk0 options:n_rxq_desc=4096** **# ovs-vsctl set Interface dpdk0 options:n_txq_desc=4096**
OVS MAC アドレステーブルを表示します (action=normal に使用)	**# ovs-appctl fdb/show br-provider**
OVS vSwitch MAC アドレステーブルのエージングタイムを設定します (デフォルトは 300 秒)	**# ovs-vsctl set bridge br-provider other_config:mac-aging-time=900**
OVS vSwitch MAC アドレステーブルサイズの設定します (デフォルトは 2048 秒)	**# ovs-vsctl set bridge br-provider other_config:mac-table-size=204800**
OVS データパスフロー (カーネルスペース) を表示します	**# ovs-dpctl dump-flows -m**
OVS データパスフローを表示します (dpdk)	**# ovs-appctl dpif/dump-flows -m br-provider**
データパスフローのポート番号とポート名の間のマッピングを表示します	**# ovs-dpctl show**
特定のブリッジで OVSOpenFlow ルールを表示します	**# ovs-ofctl dump-flows br-provider**
OpenFlow フローのポート番号とポート名の間のマッピングを表示します	**# ovs-ofctl show br-provider**

アクション	コマンド
(OVS 2.11+) - 自動リバランスを有効にします	# ovs-vsctl set Open_vSwitch . other_config:pmd-auto-lb="true"
(OVS 2.11+) - 自動リバランス間隔を別の値に変更します (デフォルトは 1 分)	# ovs-vsctl set Open_vSwitch . other_config:pmd-auto-lb-rebalance-intvl="5"
詳細な OVS 内部設定	# man ovs-vswitchd.conf.db
OVStcpdump をダウンロードするには	# curl -O -L ovs-tcpdump.in
DPDK インターフェイスからパケットキャプチャを実行するには	# 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+) 詳細な PMD パフォーマンス統計	# ovs-appctl dpif-netdev/pmd-perf-show

3.4. IRQ

これらのコマンドを使用して、割り込み要求ライン (IRQ) ソフトウェアおよびハードウェア割り込みを表示します。

アクション	コマンド
ksoftirqd ワーカーによって実行された CPU ごとの SoftIRQ バランシングを表示します	# cat /proc/softirqs | less -S
ksoftirqd ワーカーによって毎秒実行される CPU ごとの SoftIRQ バランシングを表示します	# watch -n1 -d -t "cat /proc/softirqs"
CPU ごとのハードウェアおよびソフトウェア割り込み (NMI、LOC、TLB、RSE、PIN、NPI、PIW) のバランシングを表示します	# cat /proc/interrupts | less -S
1 秒ごとに CPU ごとにバランスをとるハードウェアおよびソフトウェア割り込み (NMI、LOC、TLB、RSE、PIN、NPI、PIW) を表示します	# watch -n1 -d -t "cat /proc/interrupts"
タイマー割り込みを表示します	# cat /proc/interrupts | grep -E "LOC|CPU" | less -S
タイマー割り込みを毎秒表示します	# watch -n1 -d -t "cat /proc/interrupts | grep -E 'LOC|CPU'"
デフォルトの IRQ CPU アフィニティーを表示します	# cat /proc/irq/default_smp_affinity

アクション	コマンド
特定の IRQ に対する IRQ アフィニティーを表示します (CPUMask)	**# cat /proc/irq/89/smp_affinity**
特定の IRQ (DEC) に対する IRQ アフィニティーを表示します	**# cat /proc/irq/89/smp_affinity_list**
特定の IRQ の IRQ アフィニティーを設定します (CPUMask)	**# echo -n 1000 > /proc/irq/89/smp_affinity**
特定の IRQ(DEC) の IRQ アフィニティーを設定します	**# echo -n 12 > /proc/irq/89/smp_affinity_list**
ハードウェア割り込みの CPU アフィニティーを表示します	**# tuna --show_irqs**
特定の IRQ の IRQ アフィニティーを設定します (DEC のサポート範囲たとえば 0-4 は 0 から 4 を意味します)	**# tuna --irqs=<IRQ> --cpus=<CPU> --move**
IRQ CPU 使用率分布を表示します	**# mpstat -I CPU | less -S**
特定の CPU の IRQ CPU 使用率分布を表示します	**# mpstat -I CPU -P 4 | less -S**
SoftIRQCPU 使用率分布を表示します	**# mpstat -I SCPU | less -S**
特定の CPU の SoftIRQ CPU 使用率分布を表示します	**# mpstat -I SCPU -P 4 | less -S**

3.5. プロセス

これらのコマンドを使用して、Linux、プロセススケジューラー、および CPU アフィニティーのプロセスとスレッドを表示します。

アクション	コマンド
特定のプロセス名の分布について、すべてのプロセススレッドを含む CPU 使用率と CPU アフィニティーを表示します	**# pidstat -p $(pidof qemu-kvm) -t**
特定のプロセス名の分布について、すべてのプロセススレッドを含む CPU 使用率と CPU アフィニティーを、30 回の反復で 10 秒ごとに表示します	**# pidstat -p $(pidof qemu-kvm) -t 10 30**

アクション	コマンド
特定のプロセス名のページフォールトと、すべてのプロセススレッドを含むメモリー使用率を表示します	**# pidstat -p $(pidof qemu-kvm) -t -r**
すべてのプロセススレッドを含む、特定のプロセス名の I/O 統計を表示します	**# pidstat -p $(pidof qemu-kvm) -t -d**
特定のプロセス名について、その PID、プロセス名を含むすべての子 PID、および CPU 時間を表示します	**# ps -T -C qemu-kvm**
特定のプロセスとすべての子 PID のリアルタイムパフォーマンス統計を表示します	**# top -H -p $(pidof qemu-kvm)**
プロセススケジューラーのタイプ、優先度、コマンド、CPU アフィニティー、およびコンテキストスイッチング情報を含むすべてのシステムスレッドを表示します	**# tuna --show_threads**
優先度が最も高い特定の PID Real Time (FIFO) スケジューリングに設定します	**# tuna --threads=<PID> --priority=FIFO:99**
PMD および CPU スレッドの再スケジュールアクティビティーを表示します	**# watch -n1 -d "grep -E 'pmd|CPU' /proc/sched_debug"**
ブラウザースケジューラーの内部動作統計	**# less /proc/sched_debug**
包括的なプロセス統計とアフィニティービューを表示します 1. トップを開き、zbEEH を押します。 2. f を押して、P = Last Used Cpu (SMP) を探します。 3. 右矢印で選択してください。 4. 上矢印を使用して、CPU 使用率の前に上に移動します。 5. 左矢印を使用して選択を解除します。 6. d を使用して有効にします。 7. < を使用して CPU 番号でソートします。	**# top**
すべてのシステムプロセスとそれらの CPU アフィニティーを表示します	**# ps -eF**

アクション	コマンド
すべてのシステムプロセスを表示し、スリープおよび実行中のプロセスと、スリープの場合、どの機能で実行されているかを表示します	**# ps -elfL**
特定の PID の CPU アフィニティーを表示します	**# taskset --pid $(pidof qemu-kvm)**
特定の PID の CPU アフィニティーを設定します	**# taskset --pid --cpu-list 0-9,20-29 $(pidof <Process>)**

3.6. KVM

これらのコマンドを使用して、カーネルベースの仮想マシン (KVM) 関連のドメイン統計を表示します。

アクション	コマンド
リアルタイムの KVM ハイパーバイザー統計を表示します (VMExit、VMEntry、vCPU ウェイクアップ、コンテキストスイッチング、タイマー、停止プール、vIRQ)	**# kvm_stat**
深い KVM ハイパーバイザー統計を表示します	**# kvm_stat --once**
特定のゲストのリアルタイム KVM ハイパーバイザー統計を表示します (VMExit、VMEntry、vCPU ウェイクアップ、コンテキストスイッチング、タイマー、停止プール、vIRQ)	**# kvm_stat --guest=<VM name>**
特定のゲストの詳細な KVM ハイパーバイザー統計を表示します	**# kvm_stat --once --guest=<VM name>**
KVM プロファイリングトラップ統計を表示します	**# perf kvm stat live**
KVM プロファイリング統計を表示します	**# perf kvm top**
特定の VM の vCPU ピンを表示します	**# virsh vcpupin <Domain name/ID>**
特定の VM の QEMU エミュレータースレッドを表示します	**# virsh emulatorpin <Domain name/ID>**
特定の VM の NUMA ピンを表示します	**# virsh numatune <Domain name/ID>**
特定の VM のメモリー統計を表示します	**# virsh dommemstat <Domain name/ID>**
特定の VM の vCPU 統計を表示します	**# virsh nodecpustats <Domain name/ID>**

アクション	コマンド
特定の VM のすべての vNIC を表示します	**# virsh domiflist <Domain name/ID>**
特定の VM の vNIC 統計を表示します (DPDK VHU では機能しません)	**# virsh domifstat <Domain name/ID> <vNIC>**
特定の VM のすべての vDisk を表示します	**# virsh domblklist <Domain name/ID>**
特定の VM の vDisk 統計を表示します	**# virsh domblkstat <Domain name/ID> <vDisk>**
特定の VM のすべての統計を表示します	**# virsh domstats <Domain name/ID>**

3.7. CPU

これらのコマンドを使用して、CPU 使用率、プロセス CPU 分散、頻度、および SMI を表示します。

アクション	コマンド
特定のプロセス名の分布について、すべてのプロセススレッドを含む CPU 使用率と CPU アフィニティーを表示します	**# pidstat -p $(pidof qemu-kvm) -t**
仮想メモリー、I/O、および CPU の統計を表示します	**# vmstat 1**
集計された詳細な CPU 使用率を表示します	**# mpstat**
詳細な CPU 使用率の分布を表示します	**# mpstat -P ALL**
特定の CPU の詳細な CPU 使用率分布を表示します (範囲をサポートしていません)	**# mpstat -P 2,3,4,5**
30 回の反復で 10 秒ごとに特定の CPU の詳細な CPU 使用率分布を表示します	**# mpstat -P 2,3,4,5 10 30**
特定の CPU 周波数のハードウェア制限と周波数ポリシーを表示します	**# cpupower -c 24 frequency-info**
現在の CPU 周波数情報を表示します	**# cpupower -c all frequency-info|grep -E "current CPU frequency|analyzing CPU"**
すべての CPU の頻度と CPU % C-States 統計を表示します	**# cpupower monitor**
すべての CPU のリアルタイムの頻度と CPU % C-States 統計を表示し、変動を強調表示します	**# watch -n1 -d "cpupower monitor"**

アクション	コマンド
SMI を含むすべての CPU のより詳細な頻度と CPU % C-States 統計を表示します (RT に役立ちます)	**# turbostat --interval 1**
SMI を含む特定の CPU のより詳細な頻度と CPU % C-States 統計を表示します (RT に役立ちます)	**# turbostat --interval 1 --cpu 4**
CPU の詳細とサポートされている ISA を表示する	**# lscpu**
Intel CPU に固有: CPU 使用率、CPU IPC、CPU 実行率 (%)、L3 および L2 キャッシュヒット、ミス、命令あたりのミス、温度、メモリーチャネル使用率、および QPI/UPI 使用率に関する非常に低レベルの詳細を表示します。	**git clone Processor Counter Monitor make ./pcm.x"**

3.8. NUMA

これらのコマンドを使用して、Non-Uniform Memory Access (NUMA) 統計とプロセス分散を表示します。

アクション	コマンド
ハードウェア NUMA トポロジーを表示します	**# numactl -H**
NUMA 統計を表示します	**# numastat -n**
システム全体のメモリー使用量のように meminfo を表示します	**# numastat -m**
特定のプロセス名の NUMA メモリーの詳細とバランシングを表示します	**# numastat qemu-kvm**
特定の NUMA ノード固有の統計を表示します	**# /sys/devices/system/node/node<NUMA node number>/numastat**
NUMA ノードと PCI デバイスを使用した NUMA トポロジーの理由を非常に明確に示します	**# lstopo --physical**
関連するデバイスを使用して、物理 NUMA トポロジーのグラフ (svg 形式) を生成します	**# lstopo --physical --output-format svg > topology.svg**

3.9. メモリー

これらのコマンドを使用して、メモリー統計、Huge Page、DPC、物理 DIMM、および頻度を表示します。

アクション	コマンド
システム全体のメモリー使用量のように meminfo を表示します	**# numastat -m**
仮想メモリー、I/O、および CPU の統計を表示します	**# vmstat 1**
グローバルメモリー情報を表示します	**# cat /proc/meminfo**
特定の NUMA ノードの 2MB の Huge Page の総数を表示します	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-2048kB/nr_hugepages**
特定の NUMA ノードの 1GB の Huge Page の総数を表示します	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-1048576kB/nr_hugepages**
特定の NUMA ノードの 2MB の空いている Huge Page の合計を表示します	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-2048kB/free_hugepages**
特定の NUMA ノードの 1GB の空いている Huge Page の合計を表示します	**# /sys/devices/system/node/node<NUMA node number>/hugepages/hugepages-1048576kB/free_hugepages**
100x 2MB の Huge Page をリアルタイムで NUMA0 に割り当てます (NUMA ノードは変更できます)	**# echo 100 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages**
100x 1GB の Huge Page をリアルタイムで NUMA0 に割り当てます (NUMA ノードは変更できます)	**# echo 100 > /sys/devices/system/node/node0/hugepages/hugepages-1048576kB/nr_hugepages**
リアルタイムの SLAB 情報を表示します	**# slabtop**
詳細な SLAB 情報を表示します	**# cat /proc/slabinfo**
インストールされているメモリー DIMM の合計を表示します	**# dmidecode -t memory | grep Locator**
インストールされているメモリー DIMM の速度を表示します	**# dmidecode -t memory | grep Speed**

3.10. PCI

これらのコマンドを使用して、PCI 統計、PCI 詳細、および PCI ドライバーオーバーライドを表示します。

アクション	コマンド
システムに詳細な PCI デバイス情報を表示します	# lspci -vvvnn
PCI ツリービューを表示します	# lspci -vnnt
PCI デバイスの NUMA 情報を表示します	# lspci -vmm
特定のデバイスの PCIe 最大リンク速度を表示します	# lspci -s 81:00.0 -vv | grep LnkCap
特定のデバイスの PCIe リンク速度ステータスを表示します	# lspci -s 81:00.0 -vv | grep LnkSta
PCI デバイスとカーネルドライバーを表示します	# driverctl list-devices
PCI デバイスドライバーのオーバーライドを表示します (DPDK および SR-IOV インターフェイスで一般的)	# driverctl list-overrides
PCI デバイスに別のカーネルドライバーを設定します (永続的に再起動します)	# driverctl set-override 0000:81:00.0 vfio-pci
PCI デバイスのオーバーライドされたカーネルドライバーの設定を解除します (デバイスが使用中の場合、コマンドはハングします)	# driverctl unset-override 0000:81:00.0

3.11. TUNED

これらのコマンドを使用して、調整されたプロファイル、検証、およびログを表示します。

アクション	コマンド
調整された現在有効なプロファイルと説明を表示します	# tuned-adm profile_info
調整された利用可能なプロファイルと現在有効なプロファイルを表示します	# tuned-adm list
特定の調整されたプロファイルを有効にしました	# tuned-adm profile realtime-virtual-host
現在有効なプロファイルを確認します	# tuned-adm verify
調整されたログ	# less /var/log/tuned/tuned.log

3.12. プロファイリングプロセス

これらのコマンドを使用して、CPU プロファイリング、プロセスプロファイリング、および KVM プロファイリングを表示します。

セクション	アクション	コマンド
Process	特定の PID のプロファイリング	**# perf record -F 99 -p PID**
Process	30 秒間の特定の PID のプロファイリング	**# perf record -F 99 -p PID sleep 30**
Process	特定の PID でのリアルタイムのプロファイリング	**# perf top -F 99 -p PID**
CPU	特定の CPU コアリストでのイベントの 30 秒間のプロファイリング	**# perf record -F 99 -g -C <CPU Core(s)> – sleep 30s**
CPU	イベントの特定の CPU コアリストでのリアルタイムのプロファイリング	**# perf top -F 99 -g -C <CPU Core(s)>**
コンテキストスイッチング	特定の CPU コアリストを 30 秒間プロファイリングし、コンテキストスイッチングのみを検索します	**# perf record -F 99 -g -e sched:sched_switch -C <CPU Core(s)> – sleep 30**
KVM	特定の時間の KVM ゲストのプロファイリング	**# perf kvm stat record sleep 30s**
Cache	キャッシュ効率を探すための 5 秒間の特定の CPU コアリストのプロファイリング	**# perf stat -C <CPU Core(s)> -B -e cache-references,cache-misses,cycles,instructions,branches,faults,migrations sleep 5**
レポート	パフォーマンスプロファイリングの分析	**# perf report**
レポート	stdout でのパフォーマンスプロファイリングの報告	**# perf report --stdio**
レポート	stdout での KVM プロファイリングの報告	**# perf kvm stat report**

3.13. ブロック I/O

これらのコマンドを使用して、ストレージ I/O 分散と I/O プロファイリングを表示します。

アクション	コマンド
すべてのシステムデバイスの I/O の詳細を表示します	**# iostat**
すべてのシステムデバイスの高度な I/O の詳細を表示します	**# iostat -x**

アクション	コマンド
30 回の反復で 10 秒ごとにすべてのシステムデバイスの高度な I/O の詳細を表示します	**# iostat -x 10 30**
特定のブロックデバイスの高度な I/O プロファイリングを生成します	**# blktrace -d /dev/sda -w 10 && blkparse -i sda.* -d sda.bin**
blktrace プロファイリングを報告します	**# btt -i sda.bin**

3.14. リアルタイム

これらのコマンドを使用して、関連するリアルタイムテスト、SMI、および遅延を表示します。

アクション	コマンド
定義されたしきい値を実行している通常の RT カーネル実行をブロックしている SMI があるかどうかを識別します。	**# hwlatdetect --duration=3600 --threshold=25**

アクション	コマンド
いくつかの追加オプションを使用して、特定の時間の最大スケジューリング待ち時間を確認します。 **--duration** テスト実行の時間値を指定します。 **--mlockall** 現在および将来のメモリー割り当てをロックします。 **--priority** 最初のスレッドの優先度を設定します。 **--nanosleep** posix インターバルタイマーの代わりに clock_nanosleep を使用します。 **--interval** スレッドのベース間隔をマイクロ秒単位で設定します。 **--histogram** 実行後、レイテンシーヒストグラムを stdout にダンプします。 **--histfile** レイテンシーヒストグラムを **stdout** ではなく **<path>** にダンプします。 **--threads** テストスレッドの数を設定します。 **--numa** 標準の NUMA テスト。 **--notrace** トレースを抑制します。	**# cyclictest --duration=3600 ** **--mlockall ** **--priority=99 ** **--nanosleep ** **--interval=200 ** **--histogram=5000 ** **--histfile=./output ** **--threads ** **--numa ** **--notrace**

3.15. セキュリティー

これらのコマンドを使用して、投機的な実行と GRUB ブートパラメーターを確認します。

アクション	コマンド
現在の投機的実行のセキュリティーステータスをすべて確認します	Linux および BSD の場合は、Spectre & Meltdown vulnerability/mitigation checker を参照してください。
すべての投機的実行の修復を無効にする GRUB パラメーター	**spectre_v2=off spec_store_bypass_disable=off pti=off l1tf=off kvm-intel.vmentry_l1d_flush=never**
CVE-2017-5753(Spectre バリアント 1) のステータスを確認します	**# cat /sys/devices/system/cpu/vulnerabilities/spectre_v1**

アクション	コマンド
IBPB と Retpoline (CVE-2017-5715 Spectre variant 2) のステータスを確認します	**# cat /sys/devices/system/cpu/vulnerabilities/spectre_v2**
KPTI (CVE-2017-5754 Meltdown) ステータスを確認します	**# cat /sys/devices/system/cpu/vulnerabilities/meltdown**
Spectre-NG (CVE-2018-3639 Spectre Variant 4) ステータスを確認します	**# cat /sys/devices/system/cpu/vulnerabilities/spec_store_bypass**
Foreshadow (CVE-2018-3615 Spectre Varian 5、L1TF とも呼ばれます) のステータスを確認します	**# cat /sys/devices/system/cpu/vulnerabilities/l1tf**
Foreshadow VMEntry L1 キャッシュ効果を確認します	**# cat /sys/module/kvm_intel/parameters/vmentry_l1d_flush**
SMT ステータスを確認します	**# cat /sys/devices/system/cpu/smt/control**

3.16. JUNIPER CONTRAIL VROUTER

これらのコマンドを使用して、vRouter VIF、MPLS、Nexthost、VRF、VRF のルート、フロー、およびダンプ情報を表示します。

アクション	コマンド
vRouter カーネルスペースの人間が読める形式の統計	参照： Contrail vRouter 統計の表示
vRouter DPDK の人間が読める形式の統計	参照： Contrail vRouter 統計の表示
DPDK インターフェイスからパケットキャプチャを実行するには (vifdump の後に grep を使用しないでください)	**# vifdump vif0/234 <tcpdump standard arguments such as -v -nn -e -w <path/to/file>>**
すべての vRouter インターフェイスとサブインターフェイスの統計と詳細を表示します	**# vif --list**
特定のインターフェイスの vRouter 統計と詳細を表示します	**# vif --list --get 234**
すべてのインターフェイスとサブインターフェイスの vRouter パッカーレートを表示します	**# vif --list --rate**

アクション	コマンド
特定のインターフェイスの vRouter パッカーレートを表示します	**# vif --list --rate --get 234**
特定のインターフェイスの vRouter パケットドロップ統計を表示します	**# vif --list --get 234 --get-drop-stats**
vRouter フローを表示します	**# flow -l**
リアルタイムの vRouter フローアクションを表示する	**# flow -r**
特定の VRF の vRouter パケット統計を表示します (vif --list から VRF 番号を見つけることができます)	**# vrfstats --get 0**
すべての VRF の vRouter パケット統計を表示します	**# vrfstats --dump**
特定の VRF の vRouter ルーティングテーブルを表示します (VRF 番号は vif -- list から確認できます)	**# rt --dump 0**
特定の VRF の vRouterIPv4 ルーティングテーブルを表示します (VRF 番号は vif -- list から確認できます)	**# rt --dump 0 --family inet**
特定の VRF の vRouterIPv6 ルーティングテーブルを表示します (VRF 番号は vif -- list から確認できます)	**# rt --dump 0 --family inet6**
特定の VRF の vRouter 転送テーブルを表示します (VRF 番号は vif -- list から確認できます)	**# rt --dump 0 --family bridge**
特定のアドレスの特定の VRF で vRouter ルートターゲットを表示します	**# rt --get 0.0.0.0/0 --vrf 0 --family inet**
vRouter ドロップ統計を表示します	**# dropstats**
特定の DPDK コアの vRouter ドロップ統計を表示します	**# dropstats --core 11**
vRouter MPLS ラベルを表示します	**# mpls --dump**
特定の vRouter ネクストホップを表示します (mpls --dump output から見つけることができます)	**# nh --get 21**
すべての vRouter ネクストホップを表示します	**# nh --list**
すべての vRouter VXLAN VNID を表示します	**# vxlan --dump**

アクション	コマンド
vRouter エージェント (スーパーバイザー、xmmp 接続、vrouter エージェントなど) のステータスを表示します	**# contrail-status**
vRouter (およびすべての Contrail ローカルコンピュートノードコンポーネント) を再起動します	**# systemctl restart supervisor-vrouter**

3.17. OPENSTACK

これらの OpenStack コマンドを使用して、VM コンピュートノードを表示します。

アクション	コマンド
コンピュートノードでソートされたコンピュートノード上のすべての VM のリストを表示します	**$ nova list --fields name,OS-EXT-SRV-ATTR:host --sort host**
コンピュートノード上のすべての VM のリストを VM 名で並べ替えて表示します	**$ nova list --fields name,OS-EXT-SRV-ATTR:host**

第4章 インスタンスの TAP インターフェイスの TX キューでの高いパケット損失

本セクションでは、TX キューでパケットロスのトラブルシューティングを行います。

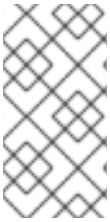
4.1. 現象

ホストオンリーネットワークを使用した仮想ネットワーク機能 (VNF) のテスト中に、インスタンスのタップインターフェイスの TX キューで高いパケット損失が見られる場合があります。テストセットアップは、ノード上の 1 つの VM から同じノード上の別の VM にパケットを送信します。パケット損失はバーストで表示されます。

次の例は、タップの TX キューにドロップされたパケットの数が多いことを示しています。

```
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. 診断



注記

このセクションでは、タップ (カーネルパス) インターフェイスでのパケットドロップについて説明します。ユーザーデータパスの vhost ユーザーインターフェイスでのパケットドロップについては、https://access.redhat.com/solutions/3381011 を参照してください。

TX ドロップは、インスタンスの vCPU とハイパーバイザー上の他のプロセス間の干渉が原因で発生します。タップインターフェイスの TX キューは、インスタンスがパケットを取得できない場合に備えて、パケットを短時間保存できるバッファーです。これは、インスタンスの CPU が十分な時間実行されない (またはフリーズする) 場合に発生します。

TUN/TAP デバイスは、一方の端がカーネルネットワークインターフェイスで、もう一方の端がユーザー空間ファイル記述子である仮想デバイスです。

TUN/TAP インターフェイスは、次の 2 つのモードのいずれかで実行できます。

- **Tap mode** は、L2 ヘッダー付きの L2 イーサネットフレームをデバイスにフィードし、ユーザー空間から同じものを受信することを期待します。このモードは VM に使用されます。
- **Tun mode** は、L3 ヘッダー付きの L3 IP パケットをデバイスにフィードし、ユーザー空間から同じものを受信することを期待します。このモードは主に VPN クライアントに使用されます。

KVM ネットワーキングでは、ユーザー空間ファイル記述子は **qemu-kvm** プロセスによって所有されま

す。タップに送信されたフレーム (ハイパーバイザーの観点からは TX) は、最終的に **qemu-kvm** 内の L2 フレームになり、仮想ネットワークインターフェイスに受信されたネットワークパケットとして VM 内の仮想ネットワークデバイスにそれらのフレームをフィードできます (RX VM の観点)。

TUN/TAP の重要な概念は、ハイパーバイザーからの送信方向が仮想マシンの受信方向であるということです。これは反対方向にも当てはまります。ハイパーバイザーの受信は、仮想マシンからの送信と同じです。

virtio-net デバイスにはパケットのリングバッファーはありません。これは、VM が (十分に高速で、またはまったく) 受信していないために TUN/TAP デバイスの TX キューがいっぱいになった場合、新しいパケットを送信する場所がなく、ハイパーバイザーが tap で TX 損失を確認することを意味します。

TUN/TAP で TX 損失に気付いた場合は、それを回避するために tap **txqueuelen** を増やします。これは、物理 NIC での受信損失を停止するために RX リングバッファーを増やすのと同様です。

ただし、これは、VM が受信時に低速でバーストしていることを前提としています。VM が常に十分な速度で実行されていない場合、またはまったく受信していない場合は、TX キューの長さを調整しても効果がありません。VM が実行または受信されていない理由を確認する必要があります。

仮想マシンのパケット処理のパフォーマンスを向上させる必要がある場合は、以下の手順を実行します。

- ハイパーバイザーで **virtio-net マルチキュー** を有効にします。
- 複数の仮想デバイス割り込みを、仮想マシン内の差異コアに分散します。

これは、KVM の libvirt ドメイン仕様に記載されています。RHEL KVM ハイパーバイザーで **virsh edit** を使用して実行できます。

Red Hat OpenStack Platform で **virtio-net のマルチ** キューを設定することができない場合は、仮想マシン内で RPS を設定して、複数の CPU コア間で受信の負荷とソフトウェアのバランスを取ることを検討してください。詳細は、kernel-doc パッケージの **scaling.txt** か、RHEL 製品ドキュメントの RPS セクションを参照してください。

4.2.1. 回避策

レイテンシーの増加やその他の欠点を犠牲にして小さなフリーズを軽減するには、TX キューを増やします。

txqueuelen を一時的に増やすには、次のコマンドを使用します。

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

txqueulen を永続的に増やすには、udev ルールを作成します。

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

udev を再読み込みするか、システムを再起動すると、新しい tap インターフェースはキューの長さ 10000 とともに起動します。以下に例を示します。

```
[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. 診断手順

次のスクリプトを使用して、ハイパーバイザーから奪われた CPU 時間の影響を確認します。

```
[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
```

インスタンスにログインし、**dd if=/dev/zero of=/dev/null** を開始して、唯一の vCPU に追加の負荷をかけます。これはデモンストレーション用であることに注意してください。VM 内からのロードの有無にかかわらず、同じテストを繰り返すことができます。TX ドロップは、ハイパーバイザー上の別のプロセスがインスタンスの vCPU から時間を奪っている場合にのみ発生します。

次の例は、テスト前のインスタンスを示しています。

```
%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
```

次のスクリプトを実行し、TX キューにドロップされたパッケージを確認します。これらは、dd プロセスがインスタンスの CPU から大量の処理時間を消費する場合にのみ発生します。

```
[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
```

次の例は、テスト中のハイパーバイザーに対する **dd** の影響を示しています。**st** ラベルは、ハイパーバイザーから奪われた時間の割合を示します。

```
%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
```

インスタンスでのテストの後半では、テストの実行時間が長すぎる場合にタイムアウトになる可能性を含め、**ssh** が遅くなる可能性があることに注意してください。

4.3. 解決策

TX キューを増やすと、これらの小さなフリーズを軽減できますが、カーネルパラメーターで CPU 固定と isolcpus を使用して完全に分離することが最善の解決策です。詳細は、Configure CPU pinning with NUMA in OpenStack を参照してください。

第5章 OPENVSWITCHDPDK を使用したインスタンス OPEN VSWITCH インターフェイスでの TX ドロップ

この手順を使用して、インスタンス vhost-user (VHU) インターフェイスでの送信ドロップのトラブルシューティングを行います。

5.1. 現象

パケットは、カーネルまたは qemu プロセスを通過せずに、virtio トランスポートを使用して vswitch からゲストに送信されます。これは、VHU インターフェイスとパケットを交換することによって行われます。

VHU は、ほとんどの場合、パケットのバッチを送受信する機能も提供する DPDK librte_vhost によって実装されます。VHU のバックエンドは、仮想マシンとパケットを交換するために qemu によって提供される virtio リングです。virtio リングには、記述子とバッファーで設定される特別な形式があります。

TX/RX (送信/受信) 統計は OpenvSwitch (OVS) 用です。これは、送信統計が VM の受信統計に直接関連していることを意味します。

VM がパケットを十分に高速に処理しない場合、OVS TX キューはオーバーフローし、パケットをドロップします。

5.1.1. パケットドロップの説明

飽和した virtio リングにより、vhost-user デバイスで TX ドロップが発生します。virtio リングはゲストのメモリーにあり、vhost-user がパケットをプッシュして VM がパケットを消費するキューのように機能します。VM がパケットを消費するのに十分な速度でない場合、virtio リングはバッファーを使い果たし、vhost-user はパケットをドロップします。

Perf および Ftrace ツールを使用して、パケットドロップのトラブルシューティングを行います。

- Perf を使用して、スケジューラースイッチの数をカウントします。これにより、qemu スレッドがプリエンプトされたかどうかを確認できます。
- Ftrace を使用して、プリエンプションの理由と所要時間を表示します。

プリエンプションの理由は次のとおりです。

- 時間割り込み (カーネルティック):
  これらは、少なくとも 2 つのコンテキストスイッチのコストを追加します。タイマー割り込みは、予測できない時間がかかる可能性のあるリードコピー更新 (RCU) コールバックを実行することもできます。
- CPU パワー管理とハイパースレッディング

これらのツールは、次のパッケージに含まれています。

- PERF: **perf rpm in rhel-7-server-rpms/7Server/x86_64**.詳細は、About Perf を参照してください。
- FTRACE: **trace-cmd info rhel-7-server-rpms/7Server/x86_64**.詳細は、About Ftrace を参照してください。

5.1.2. 他のドロップの説明

OVS 2.9 より前は、vHost ユーザーポートは **dpdkvhostuser** モードで作成されていました。このモードでは、OVS が vhost サーバーとして機能し、QEMU がクライアントとして機能します。インスタンスがダウンまたは再起動すると、OVS ブリッジの vhost ユーザーポートはアクティブなままで、VM 宛てのパケットをドロップします。これにより、**tx_drop_counter** が増加します。

次の例では、VM は **nova stop <UUID>** で停止されました。

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

これは、**ip link set dev <br internal port name>** がダウンした状態でカーネルポートがシャットダウンされ、フレームがユーザースペースにドロップされた場合に発生することと似ています。

VM が起動すると、同じ vhu ソケットに接続し、virtio リングバッファーを空にし始めます。TX が中断されることはなくなり、通常のネットワークトラフィックが再開されます。

5.1.3. DPDK の TX および RX キューの長さを増やす

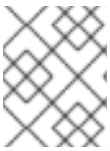
以下の OpenStackDirector テンプレートの変更により、DPDK の TX および RX キューの長さを変更できます。

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

次の例は、検証チェックを示しています。

```
[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'/>
```

カーネルの制限により、キューサイズを 1024 を超えて増やすことはできません。



注記

DPDK を介した neutron ネットワークで PXE ブートを使用できるようにする場合は、PXE バージョンが 1024 バイトをサポートしていることを確認する必要があります。

5.2. 診断

ゲストがパケットを受信できない場合、TX が vhost ユーザーポートに向かって低下することがわかります。TCP は、通常のネットワーク状態で発生するパケット損失から回復するように設計されています。NFVi には厳格な要件があり、パケットドロップに対する許容度は低くなっています。

カーネルデータパスは NFVi には遅すぎるため、DPDK で高速化された OVS を使用します。さらに、ホストのパケット処理速度に一致する DPDK 対応のゲストをデプロイすることが重要です。

5.3. 解決策

VM に割り当てられた vCPU がゲストのタスクのみを処理していることを確認します。

- クラスターが次のテンプレートパラメーターを使用してデプロイされたことを確認します。
  - **IsolCpusList: CPU** をスケジューリングから削除する
  - **NovaVcpuPinSet**: ピニング用の CPU の割り当て
  - **NovaComputeCpuSharedSet**: エミュレータースレッドピニング用の CPU の割り当て

例:

```
  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]
```

- 固定された CPU とエミュレータープールセットを利用するフレーバーで VM がデプロイされていることを確認します。

例:

```
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>
```

- これらの設定が意図したとおりに動作していることを確認してください。詳細は、Simple Compute Node CPU Partitioning and Memory Checks を参照してください。

完全に専用の CPU リソースをインスタンスに割り当てても、ネットワークパケット損失が発生する場合は、インスタンスが適切に調整され、DPDK が有効になっていることを確認してください。

第6章 DPDK を使用した OPEN VSWITCH での PMD-STATS-SHOW コマンドの出力の解釈

このセクションを使用して、DPDK を使用した Open vSwitch (OVS) での **pmd-stats-show** コマンド (**ovs-appctl dpif-netdev/pmd-stats-show**) の出力を解釈します。

6.1. 現象

ovs-appctl dpif-netdev/pmd-stats-show コマンドは、不正確な測定値を提供します。これは、PMD が開始されてからグラフ化され、収集された統計によるものです。

6.2. 診断

有用な出力を取得するには、システムを定常状態にして、測定する統計をリセットします。

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

出力の例を次に示します。

```
[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)
```

core_id 2 は主にビジーであり、処理時間の 70% とポーリング時間の 30% を費やしていることに注意してください。

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

この例では、**miss** は、DPDK データパス (emc または dp 分類子) で分類されなかったパケットを示します。通常の状況では、それらは **ofproto** レイヤーに送信されます。まれに、フローの再検証ロックが原因で、または **ofproto** レイヤーがエラーを返した場合、パケットはドロップされます。この場合、**lost** の値も増分されて損失を示します。

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

詳細については、OVS-DPDK Datapath Classifier を参照してください。

6.3. 解決策

このセクションでは、**ovs-appctl** コマンドを使用してトラフィックフローを表示する手順を示します。

6.3.1. アイドル PMD

次の例は、core_ids が dpdk0 に固定されている PMD にサービスを提供し、管理トラフィックのみが dpdk0 を流れるシステムを示しています。

```
[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. パケットドロップを伴う負荷テスト中の PMD

次の例は、core_ids が dpdk0 に固定されている PMD にサービスを提供し、負荷テストが dpdk0 を通過して、多数の RX ドロップを引き起こすシステムを示しています。

```
[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)
```

パケットドロップが発生する場合、処理サイクルとポーリングサイクルの比率が高いことがわかります (90% 以上の処理サイクル)。

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

パケットあたりの平均サイクル (CPP) とパケットあたりの平均処理サイクル (PCPP) を確認します。アイドルサイクルはカウントされないため、完全にロードされた PMD の PCPP/CPP 比は 1 と予想できます。

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

6.3.3. mpps 容量の 50% で負荷テスト中の PMD

次の例は、core_ids が dpdk0 に固定された PMD にサービスを提供し、負荷テストが dpdk0 を通過して、この dpdk0 インターフェイス (約 12.85 Mpps) の 6.4 Mpps (最大容量の約 50%) を送信するシステムを示しています。

```
[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)
```

pps がインターフェイスの最大値の約半分である場合、処理サイクルとポーリングサイクルの比率が低くなります (約 70% の処理サイクル)。

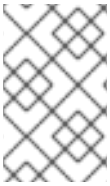
```
polling cycles:5460724802 (29.10%)
processing cycles:13305794333 (70.90%)
```

6.3.4. ヒット vs ミス vs ロスト

次の例は、対象に関する man ページを示しています。

```
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.
> (...)



注記

一部のドキュメントはカーネルデータパスを参照しているため、**user space processing** とは、パケットがカーネル **sw** キャッシュ (**emc** および **dpcls** と同等) に分類されず、ユーザースペースの ofproto レイヤーに送信されないことを意味します。

第7章 NOVA での SR-IOV ポートの接続と取り外し

次のセクションを使用して、SR-IOV ポートを接続および切断します。

7.1. 現象

Red Hat OpenStack Platform 10 以降の nova で SR-IOV ポートを接続または切断することはできません。Nova ログは、**No conversion for VIF type hw_veb yet** を報告します。

7.2. 診断

すでに作成されているインスタンスに SR-IOV ポートをアタッチまたはデタッチすることはできません。SR-IOV ポートは、インスタンスの作成時に接続する必要があります。

7.3. 解決策

次の例は、インスタンスの起動後にインターフェイスを接続する試みを示しています。

```
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
```

これは次のエラーで失敗します。

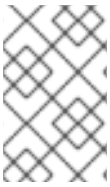
```
ERROR (ClientException): Unexpected API Error. Please report this at
http://bugs.launchpad.net/nova/ and attach the Nova API log if possible.
<type 'exceptions.KeyError'> (HTTP 500) (Request-ID: req-36b544f4-91a6-442e-a30d-6148220d1449)
```

正しい方法は、SR-IOV ポートを使用してインスタンスを直接生成することです。

```
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
```

第8章 OPEN VSWITCH を使用した LACP ボンディングの設定とテスト



注記

使用している Red Hat OpenStack Platform (RHOSP) のバージョンによっては、LACP との OVS 結合がサポートされない場合があります。製品ドキュメントをチェックして、LACP との OVS 結合がサポートされていることを確認します。

Open vSwitch DPDK を使用して LACP ボンディングを設定およびテストするには、次のタスクを実行します。

1. LACP のスイッチポートを設定します。
2. LACP の Linux カーネルボンディングをベースラインとして設定します。
3. LACP の OVS DPDK ボンディングを設定します。

注記

このトピックでは、Dell S4048-ON スイッチを使用したスイッチ設定について説明します。RHEL と OVS の設定は同じままですが、異なるスイッチベンダーのオペレーティングシステムは、異なる構文を使用して LACP を設定します。

8.1. LACP のスイッチポートの設定

1. スイッチインターフェイスをデフォルト設定にリセットします。

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

2. ポートチャネルおよびその他のポート設定を設定します。

```
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. VLAN を設定します。

```
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. VLAN タギングを確認します。

```
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. LACP 設定を確認します。

```
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. LACP の LINUX カーネルボンディングをベースラインとして設定する

Linux カーネルボンディングをベースラインとして設定し、ホストがスイッチと LACP ボンディングを形成できることを確認します。

1. すべてのインターフェイスをカーネルスペースに移動し、カーネルスペースボンディングでテストします。この例では、p1p1 はバスアドレス **0000:04:00.0** にマップされ、p1p2 はバスアドレス **0000:04:00.1** にマップされます。

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

2. ボンディングドライバーをロードし、ボンディングインターフェイス (**bond10**) を設定し、インターフェイス **p1p1** と **p1p2** をスレーブ化します。

```
[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. RHEL から LACP を確認します。

```
[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
```

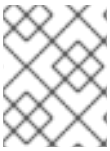
4. スイッチから LACP を確認します。

```
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. ボンディング設定を削除します。

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



注記

ボンディングモードの変更に関する情報は、「How to change the bonding mode without reboot the system?」を参照してください。

8.3. LACP 用の OVS DPDK ボンディングの設定

次の目的は、OVS DPDK 内で LACP ボンディングを設定することです。

8.3.1. Open vSwitch を準備する

1. Huge Page やその他の値が RHEL で設定されていることを確認します。

```
[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. DPDK 用に OVS を設定します。

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

3. インターフェイスをユーザースペースに切り替えます。

```
[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. Open vSwitch、**journalctl -u ovs-vswitchd -f &** を再起動し、バックグラウンドで実行します。

```
[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. LACP ボンディングを設定する

1. ボンディングを追加します。

```
[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. Open vSwitch から確認します:

```
[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. スイッチから確認します。

```
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. OVS からのポートの有効化/無効化

ovs-ofctl mod-port <bridge> <port> [up|down] を使用して、ポートを有効または無効にできます。

1. ポートをシャットダウンします。

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

2. シャットダウンを確認します。

```
[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. スイッチで確認します。

```
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. ポートを再度有効にします。

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

5. RHEL から確認します。

```
[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. スイッチから確認します。

```
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
```

第9章 OVS DPDK を使用したさまざまなボンディングモードのデプロイ

この手順を使用して、Red Hat OpenStack Platform で OVS-DPDK を使用してさまざまなボンディングモードをデプロイします。

9.1. 解決策

compute.yaml 環境ファイルに次の変更を加えます。この例では、MTU 値も 2000 に設定されていることに注意してください。

```
(...)
          -
            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
(...)
```

上記で行ったテンプレートの変更を使用して、オーバークラウドをデプロイまたは再デプロイします。完了したら、オーバークラウドノードで次の手順を実行します。

os-net-config 設定を確認します。

```
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
},
(...)
```

ボンディングを確認します。

```
[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
```

第10章 COULD NOT OPEN NETWORK DEVICE DPDK0 (NO SUCH DEVICE) IN OVS-VSCTL SHOW メッセージを受け取ります。

10.1. 現象

Could not open network device dpdk0 (No such device) in ovs-vsctl show メッセージを受け取ります。

10.2. 診断

Red Hat は、DPDK でサポートされているハードウェア にリストされているポーリングモードドライバー (PMD) のサブセットをサポートしています。Red Hat は、2017 年 8 月にサポートされていない PMD を無効にしました。

アップストリーム PMD には、セキュリティーまたはパフォーマンスの問題がある可能性があります。したがって、PMD は、Red Hat の認定テストに合格するために、重要なテストを通過する必要があります。

有効になっているすべての PMD のリストは、/usr/share/doc/openvswitch-<version>/README.DPDK-PMDS にあります。このリストには、Red Hat でサポートされていない PMD が含まれている可能性があります。**README.DPDK-PMDS** にリストされていないポーリングモードドライバーはサポートされていません。

10.3. 解決策

次の例は、openvswitch-2.6.1 でサポートされている PMD を示しています。

```
[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
```

この例は、openvswitch-2.9.0 でサポートされている PMD を示しています。

```
[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

第11章 OPEN VSWITCH でゲスト RAM を割り当てるために使用できる空きホストメモリーページが不十分です

11.1. 現象

インスタンスをデプロイして、インスタンスに十分な物理 CPU が搭載されているコンピュートノードにスケジューリングする場合、nova はインスタンスメモリー用に十分な空きヒュージページを持っている場合、nova は以下を返します。

```
[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"}
(...)
```

また、コンピュートノード上の **/var/log/nova/nova-compute.log** で以下の ERROR メッセージが表示されます。

```
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
```

さらに、libvirt は次のログファイルを作成します。

```
[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. 診断

追加の設定がないと、nova は特定の量の Huge Page メモリーが他のプロセスによって使用されていることを認識しません。デフォルトでは、nova はすべての Huge Page メモリーがインスタンスで使用可能であると想定しています。Nova は、この NUMA ノードにまだ空き pCPU と空き hugepage メモリーがあると判断した場合、最初に NUMA ノード 0 をいっぱいにします。この問題は、次の原因で発生する可能性があります。

- 要求された pCPU はまだ NUMA0 に適合します
- 既存のすべてのインスタンスの結合されたメモリーと、生成されるインスタンスのメモリーは、NUMA ノード 0 に引き続き適合します。
- OVS などの別のプロセスは、NUMA ノード 0 に一定量の hugepage メモリーを保持します

11.2.1. 診断手順

1. **meminfo** を確認してください。以下は、NUMA ノードごとに 2MB の hugepages と 512 の空き hugepages を持つハイパーバイザーを示しています。

```
[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. NUMA アーキテクチャーを確認します。

```
[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. OVS によって予約されている Huge Page を確認してください。次の出力では、OVS は NUMA ノードごとに 512MB の Huge Page を予約しています。

```
[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. 次のフレーバー (1 つの vCPU と 512 MB またはメモリー) でインスタンスをデプロイします。

```
[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                                                               |
+----------------------------+-----------------------------------------------------------------+
```

新しいインスタンスが起動し、NUMA 1 のメモリーを使用します。

```
[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
```

このインスタンスは起動に失敗します:

-

```
[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. コンピュートノードから、NUMA ノード 0 の空きの Huge Page が使い果たされていることを確認します。ただし、NUMA ノード 1 には十分なスペースがあります。

```
[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. **/var/log/nova/nova-compute.log** の情報により、インスタンスの CPU が NUMA ノード 0 に固定されていることが分かります。

```
<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>
```

numatune セクションで、nodeset=0 は、メモリーが NUMA0 から要求されることを示します。

11.3. 解決策

管理者は、インスタンスで使用されていない Huge Page メモリーの量を nova に入力できます。

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

サイズパラメーターは、KiB の Huge Page サイズです。count パラメーターは、NUMA ノードごとに OVS によって使用される Huge Page の数です。たとえば、Open vSwitch で使用される 4096 のソケットメモリーの場合、次の値を使用します。

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

OpenStack Director でこれを実装する方法の詳細は、How to set reserved_huge_pages in /etc/nova/nova.conf in Red Hat OpenStack Platform 10 を参照してください。

このオプションは、Red Hat OpenStack Platform 10: OpenStack nova.conf - 設定オプションに記載されています。

Red Hat OpenStack Platform 11 では、OpenStack nova.conf の設定オプションについて記載しています。

```
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.
```

/etc/nova/nova.conf でデバッグを有効にすると、 **openstack-nova-compute** を再起動した後、ログに次の情報が表示されます。

```
[root@overcloud-compute-1 virt]# systemctl restart openstack-nova-compute
(...)
[root@overcloud-compute-1 virt]# grep reserved_huge_pages /var/log/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]#
```

第12章 PERF を使用して OVS DPDK PMD CPU 使用率のトラブルシューティングを行い、トラブルシューティングデータを収集して送信します

1. 前提条件、このセクションの手順を使用して、トラブルシューティングツールをインストールします。
2. コンピュートノードに **perf** をインストールします。

```
yum install perf -y
```

3. Open vSwitch デバッグ RPM をインストールします。

```
subscription-manager repos --enable=rhel-7-server-openstack-10-debug-rpms
```

4. sysstat をインストールします (**pidstat** コマンドに必要):

```
yum install sysstat -y
```

12.1. 診断

このセクションの手順を使用して、データのトラブルシューティングと収集を行います。

12.1.1. PMD スレッド

1. PMD スレッドの場所を決定します。

```
IFS=$'\n' ; 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
```

以下に例を示します。

```
[root@overcloud-compute-1 ~]# IFS=$'\n' ; 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
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. 問題を再現しながら、perf レコードと perf レポートを実行し、出力を保存します。

- スクリプト **gather_perf_data_a.sh** を作成します。

```
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
```

- スクリプトを実行します。

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

レポートは、**perf report -i ${archive_name}** を使用して読み取ることができます。これが Red Hat サポートで開かれたケースの場合は、結果の tar アーカイブをケースに添付します。

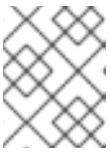
12.1.2. 追加データ

1. スクリプト **gather_perf_data_b.sh** を作成して、追加のデータを収集します。

```
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. スクリプトを実行します。

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



注記

十分なディスク容量があることを確認してください。perf.data ファイルは、数ギガバイトのディスク領域を占める可能性があります。

これが Red Hat サポートチケットの場合は、結果の tar アーカイブをケースに添付します。

12.1.3. Open vSwitch ログ

1. すべての Open vSwitch (OVS) ログを提供します。**/var** に十分なディスク容量があることを確認してください。**df -h** を使用して/var の空きディスク容量を決定し、**du -sh /var/log/openvswitch** を使用して OVS ログの合計サイズを決定します。

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

2. 結果のファイル (例: **/var/openvswitch_overcloud-compute-0_2018-02-27_153713.tar.gz**) を分析用のサポートケースに添付します。

3. sosreport を生成して提供します。**/var** に十分なディスク容量があることを確認してください。**df -h** を使用して、**/var** の空きディスク容量を決定します。

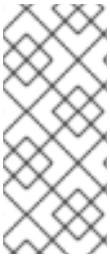
```
sosreport --batch --all-logs
```

第13章 NFV を使用する仮想環境での VIRSH EMULATORPIN の使用

この手順を使用して、NFV を使用する Red Hat OpenStack Platform で virsh emulatorpin を使用した場合の影響を判断します。

13.1. 現象

Red Hat OpenStack Platform {vernum} NFV 環境でパケット損失が発生し、エミュレータースレッドの固定が設定されていません。



注記

Red Hat OpenStack Platform 10 では、エミュレータースレッドの固定に対するサポート例外が必要です。ただし、Red Hat では、ほぼすべての NFV ケースにおいて、エミュレータースレッドを固定することを強く推奨します。デフォルトのエミュレータースレッド設定を変更して、パフォーマンスを大幅に向上させることができます。Red Hat サポートでチケットを作成し、必要に応じてサポート例外をリクエストします。

13.2. 解決策

このセクションを使用して、エミュレーターのスレッドの固定を調査および設定します。

13.2.1. qemu-kvm エミュレータースレッド

エミュレータースレッドは、仮想マシンのハードウェアエミュレーションの割り込み要求およびノンブロッキングプロセスを処理します。vCPU を実行していないスレッドは、**qemu-kvm** エミュレータースレッドです。以下の例を参照してください。

```
[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
```

Linux CFS (完全に公平なスケジューラー) により、エミュレータースレッドは通常、libvirt のエミュレーターピンセットで定義されている範囲内で、ある pCPU から別の pCPU に定期的に移動します。

NFV コンテキストでは、**isolcpus** パラメーターを使用するときにエミュレータースレッドを設定すると、問題が発生する可能性があります。これは、このカーネル設定により、これらの CPU での CFS スケジューリングが無効になるためです。**isolcpus parameter** を使用していない場合、エミュレータースレッドがパケットを処理している CPU に割り込むと、パケット損失が発生する可能性があります。

エミュレータースレッドの例は次のとおりです。

- qemu-kvm スレッド
- vnc_worker スレッド

- vhost-<qemu-kvm PID> カーネルスレッド (virtio-net が使用されている場合 (ハイパーバイザー上のカーネルネットワーク))

13.2.2. エミュレータースレッドの固定化に関するデフォルトの動作

デフォルトでは、nova はすべての vCPU に割り当てられた pCPU にまたがるエミュレータースレッドピンセットを設定します。**isolcpus** パラメーターを使用していない場合、エミュレータースレッドは任意の pCPU でスケジュールでき、ある pCPU から別の pCPU に定期的に移動します。

```
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'/>
(...)
```

したがって、これらの CPU はいずれも、qemu のエミュレータースレッドによってプリエンプションが実行され、パケットドロップのリスクがあります。

13.2.3. OpenStack nova(OpenStack Platform 10)におけるエミュレータースレッドの現在の実装

Red Hat OpenStack Platforms 10 では、エミュレータースレッドを固定するための公式にサポートされている方法はありません。以下の例のように、**virsh エミュレーターピン(...)--live を使用して、エミュレーター** スレッドを pCPU セットに移動できます。

```
# to pin emulator threads of instance instance-0000001d to CPU 34
virsh emulatorpin instance-0000001d 34 -live
# to pin emulator threads of instance instance-0000001d to CPUs 32,34
virsh emulatorpin instance-0000001d 32,34 --live
```

これらの変更は、インスタンスのランタイムのみに対して最後に変更されます。永続的な変更には、cron ジョブ、bash スクリプト、Ansible タスクなどの外部メカニズムが必要です。これは、サポート例外の件名である必要があります。

13.2.4. エミュレータースレッドのスケジューリングに対する isolcpus の影響

isolcpus を使用すると、CFS スケジューラーが無効になり、すべてのエミュレータースレッドが最初に使用可能な最もインデックスの少ない pCPU で実行されます。結果として、介入や追加の設定を行わないと、インスタンスの 1 つの vCPU が、エミュレータースレッドとのリソース競合のリスクが高くなります。

詳細は、Kernel.org Bugzilla – Bug 116701 を参照してください。

次のアルゴリズムを使用して、エミュレータースレッドが使用している vCPU を判別します。

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

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

たとえば、この場合、すべてのエミュレータースレッドと子は、デフォルトのエミュレーターピンセットからアフィニティー 1〜3 を継承します。

```
[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 ~]#
```

isolcpus と組み合わせると、すべてのエミュレータースレッドと vhost- *スレッドが pCPU1 で実行され、再スケジュールされることはありません。

```
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
```

13.2.5. エミュレータースレッドの最適な位置

このセクションでは、エミュレータースレッドを次のネットワークに配置する方法について説明します。

- インスタンス内の DPDK ネットワーキングと Open vSwitch の netdev データパス
- インスタンス内の DPDK ネットワーキング、Open vSwitch のシステムデータパス、ハイパーバイザーのカーネルスペースネットワーキング
- Open vSwitch のインスタンス内カーネルネットワーキングと netdev データパス

13.2.5.1. Open vSwitch のインスタンスと netdev データパス内の DPDK ネットワーキングによるエミュレータースレッドの最適な配置

DPDK がインスタンス内で実行される場合、パケット処理は完全にユーザー空間で実行されます。PMD を vCPU0 で実行するようにスケジュールしないでください。これは、OS と割り込み処理のために残しておく必要があります。インスタンス内の PMD CPU はアクティブループを実行し、CPU の 100% を必要とするため、プリエンプトしないでください。これらの vCPU のいずれかがプリエンプトされると、パケット損失が発生する可能性があります。したがって、emulatorpin cpuset は、1 以上の番号が付けられた仮想 CPU を処理する物理 CPU とオーバーラップしないように設定する必要があります。

インスタンス内の DPDK ネットワーキングでは、エミュレータースレッドの最適な場所は、vCPU 0 を処理する pCPU か、仮想 CPU をまったく処理しない専用の物理 CPU のいずれかです。

OVS-DPDK がハイパーバイザーとインスタンス内の DPDK で使用されている場合は、エミュレータースレッドを vCPU0 に配置します。

13.2.5.2. Open vSwitch における DPDK ネットワーキングを用いたエミュレータースレッドのインスタンスおよびシステムデータパス内への最適配置

ハイパーバイザーでカーネルスペースネットワーキングが使用されている場合、ハイパーバイザーでのパケット処理はカーネル内で実行されます。

インスタンス内の DPDK ネットワーキングでは、エミュレータースレッドの最適な場所は、vCPU 0 を処理する pCPU か、仮想 CPU を処理しない専用の物理 CPU のいずれかです。

このシナリオでは、vNIC キューのパケット処理は、ハイパーバイザーの **vhost-<qemu-kvm PID>** カーネルスレッド内で実行されることに注意してください。トラフィックが多い場合、これらのカーネルスレッドはかなりの CPU 負荷を発生させる可能性があります。エミュレータースレッドの最適な場所は、ケースバイケースで決定する必要があります。

```
[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]
```

13.2.5.3. Open vSwitch のインスタンスと netdev データパス内でのカーネルネットワーキングを利用したエミュレータースレッドの最適な配置

インスタンス内のカーネルネットワークでは、2 つのオプションがあります。

- インスタンス内の softirqs など、割り込み分散を最適化します。このような場合、エミュレータースレッドに追加の pCPU を割り当てる必要はなく、ネットワーク割り込みを処理していない pCPU にエミュレータースレッドを割り当てることができます。
- エミュレータースレッド用に同じ NUMA ノードで専用の pCPU を使用します。

最初のオプションは複雑であるため、2 番目のオプションをお勧めします。

13.3. 診断

13.3.1. デモンストレーション環境

デモンストレーション環境は、**instance-0000001d** という 1 つのインスタンスを実行します。関連する qemu-kvm スレッドの PID は次のとおりです。

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

13.3.2. Emulatorpin のしくみ

デフォルトでは、Red Hat OpenStack Platform デプロイメントは以下の設定を使用します。

```
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>
(...)
```

これにより、qemu-kvm、vnc_worker などのエミュレータースレッドの割り当てが予測できなくなります。

```
[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
```

エミュレータースレッドは、virsh emulatorpin を使用して移動できます。

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

CPU 以外のすべてのスレッドのアフィニティーが変更されることに注意してください。

```
[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
```

/proc/sched_debug の履歴データ内のスイッチの数に注意してください。次の例では、PID 73517 はすでに cpu#34 に移動しています。他のエミュレーターワーカーは最後の出力以降実行されなかったため、cpu#10 に引き続き表示されます。

```
[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
```

スレッド 73517 が CPU#34 に移動する方法に注意してください。ここで VNC セッションを操作すると、/proc/sched_debug が cpu#34 の vnc_worker スレッドも表示していることがわかります。

```
[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
```