Inspur is a China-based leading total solutions provider for datacenter, cloud computing and big data.

**Top #1 in China**
- Server
- HPC
- Switch

**Top #3 in the world**
- Supercomputer
- Storage
- AI

#### Hardware Products

#### Solutions
- Enterprise Cloud Solution
- Public Cloud Solution
- HPC Solution
- AI Solution
Running OVS-DPDK without hugepages, busy loop and exclusive cores

Yi Yang @ Inspur
Why

- Motivation: use OVS-DPDK to handle all the use cases and avoid burdensome and boring OVS kernel maintenance work
- It’s not easy to maintain OVS-DPDK and OVS kernel in the same code base
- The cycle a new feature is implemented is very long because many depended kernel patches must be backported into OVS
- Backporting kernel patches is very burdensome and boring
- It is more difficult to push some patches into kernel because kernel community has different maintainers from ovs community even if ovs community looks forward to merging them ASAP.
- Saving power, cpu cores, memory resources
But current OVS DPDK only can do so

<table>
<thead>
<tr>
<th>CPU%</th>
<th>MEM%</th>
<th>TIME+</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.1</td>
<td>3:36.93</td>
<td>/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd</td>
</tr>
<tr>
<td>100</td>
<td>0.1</td>
<td>3:32.53</td>
<td>/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd</td>
</tr>
</tbody>
</table>

- HugePages_Total: 8
- HugePages_Free: 0
- HugePages_Rsvd: 0
- HugePages_Surp: 0
- Hugepagesize: 1048576 kB
Actually OVS DPDK can do so

<table>
<thead>
<tr>
<th>CPU%</th>
<th>MEM%</th>
<th>TIME+</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.5</td>
<td>13.0</td>
<td>0:07.94</td>
<td>/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd</td>
</tr>
<tr>
<td>8.5</td>
<td>13.0</td>
<td>0:03.20</td>
<td>/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd</td>
</tr>
</tbody>
</table>

```
HugePages_Total:     8
HugePages_Free:      8
HugePages_Rsvd:      0
HugePages_Rsvmd:     0
Hugepagesize:        1048576 kB
```
Then how to attain this

- DPDK PMD thread can work in interrupt mode

  ```c
  uint32_t data = port_id << CHAR_BIT | queue_id;
  ret = rte_eth_dev_rx_intr_ctl_q(port_id, queue_id,
      RTE_EPOLL_PER_THREAD,
      RTE_INTR_EVENT_ADD,
      (void *)&(uintptr_t)data));

  nb_rx = rte_eth_rx_burst(...);
  if (!nb_rx) {
      rte_eth_dev_rx_intr_enable(port_id, queue_id);
      n = rte_epoll_wait(RTE_EPOLL_PER_THREAD, event, max_event_nr, timeout);
      if (n > 0) {
          rte_eth_dev_rx_intr_disable(port_id, queue_id);
      }
  }
  ```

- DPDK application can run by using normal 4K pages

  `ovs-vsctl set Open_vSwitch . other_config:dpdk-extra="--no-huge -m 1024"`
Requirements

✓ DPDK >= 18.05 (4K pages support, vhost-user interrupt mode)
✓ OVS >= 2.9 (vdev support)
✓ qemu >= 2.5
✓ UIO driver must be vfio-pci
✓ VT-d must be supported and enabled in hardware platform
✓ IOMMU must be enabled (iommu=pt intel_iommu=on)
My experiment setup

- DPDK 18.08
- OVS 2.10.0 (only can support DPDK 17.11.3, need a patch to support DPDK 18.08)
cmds for my experiment setup

```bash
sudo ovs-vsctl set Open_vSwitch . other_config:dpdk-init=true
sudo ovs-vsctl set Open_vSwitch . other_config:dpdk-extra="--no-huge -m 4096"
sudo modprobe vfio-pci

sudo ovs-vsctl add-port br-int dpdk0 -- set Interface dpdk0 type=dpdk
  options:dpdk-devargs=0000:08:00.1

sudo ovs-vsctl add-port br-int dpdkvhostuser1 -- set Interface dpdkvhostuser1 type=dpdk
  options:dpdk-devargs=net_vhost0,iface=/var/run/openvswitch/dpdkvhostuser1,queues=1

sudo ovs-vsctl add-port br-int dpdkvhostuser2 -- set Interface dpdkvhostuser2 type=dpdk
  options:dpdk-devargs=net_vhost1,iface=/var/run/openvswitch/dpdkvhostuser2,queues=1
```
cmds for my experiment setup (cont’d)

VM1:
qemu-system-x86_64 -smp 2 -m 4096 -enable-kvm -chardev socket,id=char0,path=/var/run/openvswitch/dpdkvhostuser1
   -netdev type=vhost-user,id=mynet1,chardev=char0 \
   -device virtio-net-pci,netdev=mynet1,mac=52:54:00:02:d9:00 \
   -net nic,model=virtio \
   -net user,hostfwd=tcp::2222-:22 \
   -numa node,memdev=mem -mem-prealloc \
   -object memory-backend-file,id=mem,size=4096M,mem-path=/home/yyang13/vhost-workspace/tmpfs,share=on \
   -D qemu.log -monitor telnet::5552,server,nowait \
   -vnc :2 \
   -daemonize \
   ubuntu-16.04-server-cloudimg-amd64-disk1.img
Known issues/challenges

- OVS-DPDK pmd thread can’t be blocked
  - pmd thread requires netdev_dpdk_rxq_recv to return immediately for multiple port Rx
  - rte_epoll_wait requires to wait 1 millisecond at least if there isn’t Rx interrupt happening
  - Possible solutions:
    1) change epoll API to support microsecond-level timer
    2) usleep + rte_epoll_wait with 0 timeout, ugly but can work

- DPDK still needs much memory even if we use 4K pages
  - Possible solution: change DPDK code to consume less memory

- Can’t get interrupt from vhost-user
Resources

https://github.com/yyang13/ovs-conf-2018
Thank you! Q&A