

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



Top #1 in China Top #3 in the world

Server



HPC



Supercomputer



AI



Switch



Storage

Hardware Products



Enterprise Cloud Solution



Public Cloud Solution



HPC Solution



AI Solution

Solutions

# Running OVS-DPDK without hugepages, busy loop and exclusive cores

Yi Yang @ Inspur

- 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

CPU%	MEM%	TIME+	Command
100.	0.1	3:36.93	/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd
100.	0.1	3:32.53	/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd

```
HugePages_Total:      8
HugePages_Free:       0
HugePages_Rsvd:       0
HugePages_Surp:       0
Hugepagesize:         1048576 kB
```

# Actually OVS DPDK can do so

CPU%	MEM%	TIME+	Command
8.5	13.0	0:07.94	/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd
8.5	13.0	0:03.20	/home/yyang13/ovs-for-demo/vswitchd/ovs-vswitchd

```
HugePages_Total:      8
HugePages_Free:       8
HugePages_Rsvd:       0
HugePages_Surp:       0
Hugepagesize:        1048576 kB
```



# Then how to attain this

- DPDK PMD thread can work in interrupt mode

```
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);
    }
}
```

```
struct rte_eth_conf port_conf = {
    ...
    .intr_conf = {
        .rxq = 1,
    },
    ...
};
```

- DPDK application can run by using normal 4K pages

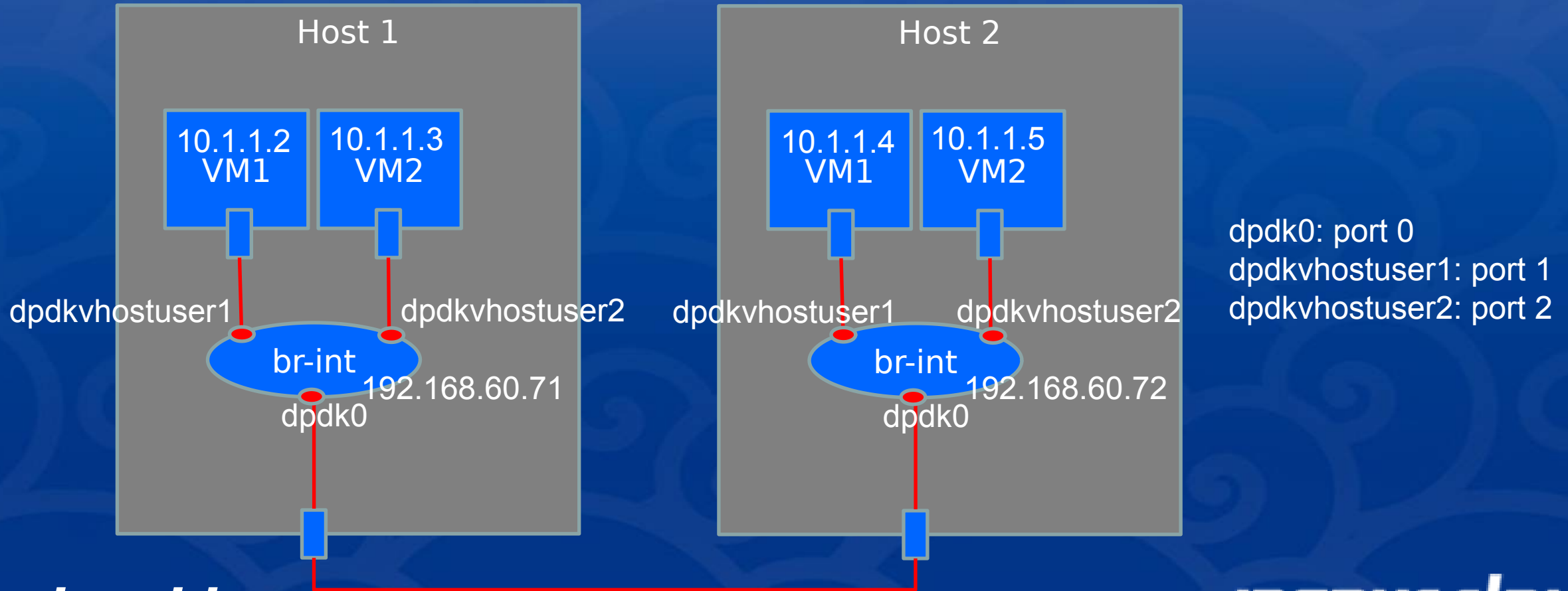
*ovs-vsctl set Open\_vSwitch . other\_config:dpdk-extra="--no-huge -m 1024"*

# Requirements

- ✓ DPDK  $\geq$  18.05 (4K pages support, vhost-user interrupt mode)
- ✓ OVS  $\geq$  2.9 (vdev support)
- ✓ qemu  $\geq$  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 18.0 8)





# cmds for my experiment setup

9

```
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)

10

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
```

- OVS-DPDK pmd thread can't be blocked
  - pmd thread requires netdev\_dpdk\_rxq\_rcv 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

<https://github.com/yyang13/ovs-conf-2018>



# Thank you! Q&A



WeChat



APP

[cloud.inspur.com](http://cloud.inspur.com)

*inspur cloud*