EI313 Lab5

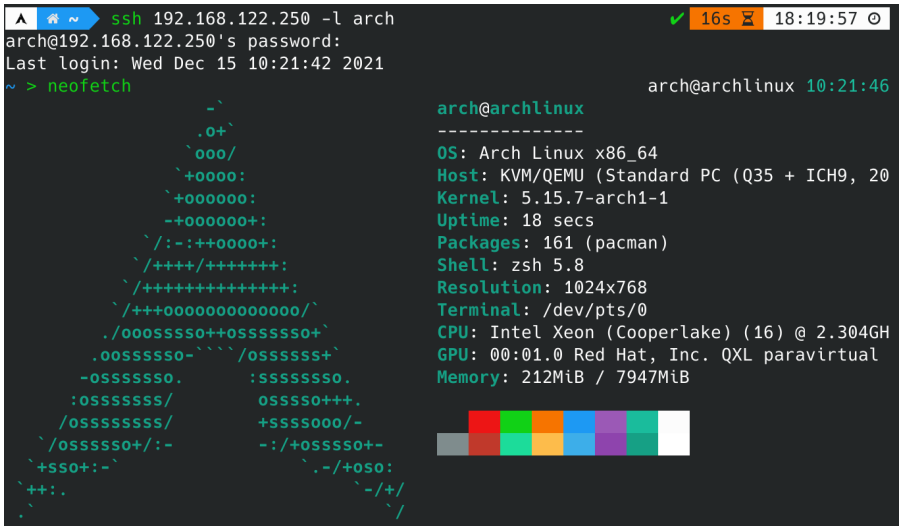
唐亚周 519021910804

DPDK performance test.

1 配置虚拟机

> (1) Create two virtual machines on KVM.

这里我继续使用Lab4中配置的虚拟机，并在宿主机上使用ssh连接虚拟机，从而方便实验的进行。虚拟机信息如下：



2 配置DPDK

> (2) Compile and install DPDK library on each virtual machine.

2.1 编译安装[1]

首先安装依赖的包。

```
sudo pacman -S wget meson gcc python-pyelftools
```

下载最新版DPDK：

进入DPDK所在文件夹，输入以下命令进行编译安装。这里同时编译安装了l2fwd样例程序。

```
meson -Dexamples=l2fwd build
cd build
ninja
sudo ninja install
sudo ldconfig
```

等待安装完成：

```
Build targets in project: 960

DPDK 21.11.0

  User defined options
    examples: l2fwd

Found ninja-1.10.2 at /usr/bin/ninja
~/dpdk-21.11 > cd build                                    4s arch@archlinux 13:36:42
~/dpdk-21.11/build > ninja                                    arch@archlinux 13:36:46
[2750/2750] Linking target app/dpdk-pdump
~/d/build >                                             1m 54s arch@archlinux 13:38:42
```

2.2 配置

2.2.1 Hugepage配置

在lab4中配置过Hugepage，因此不再赘述，具体信息如下：

```
~ > cat /proc/meminfo | grep Huge
AnonHugePages:         0 kB
ShmemHugePages:        0 kB
FileHugePages:         0 kB
HugePages_Total:    1024
HugePages_Free:     1024
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
Hugetlb:         2097152 kB
```

2.2.2 网卡驱动绑定[2]

由于我在尝试使用VFIO作为驱动时，一直绑定失败，并且没能找到解决方案，我选择UIO来重新进行尝试。[3]

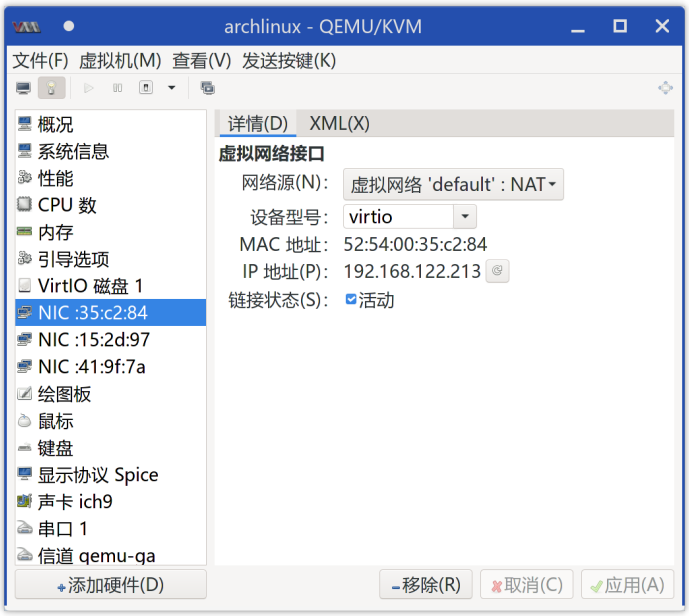
首先编辑grub配置文件，在内核启动参数上加上`intel_iommu=off`以禁用IOMMU。

```
# GRUB boot loader configuration

GRUB_DEFAULT=0
GRUB_TIMEOUT=1
GRUB_DISTRIBUTOR="Arch"
GRUB_CMDLINE_LINUX_DEFAULT="rootflags=compress-force=zstd transparent_hugepage=never intel_iommu=off"
GRUB_CMDLINE_LINUX="net.ifnames=0"
```

然后重新生成grub配置，并重启。

```
sudo grub-mkconfig -o /boot/grub/grub.cfg
```

注意，由于用于绑定的网卡必须处于关闭状态，考虑到虚拟机的网络连接需求，我使用virt-manager加入了两张虚拟网卡。



网卡信息如下：

```
~ > ip a                                                    arch@archlinux 07:01:12
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 52:54:00:35:c2:84 brd ff:ff:ff:ff:ff:ff
    altname enp1s0
    inet 192.168.122.213/24 metric 1024 brd 192.168.122.255 scope global dynamic eth0
       valid_lft 3593sec preferred_lft 3593sec
    inet6 fe80::5054:ff:fe35:c284/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 52:54:00:15:2d:97 brd ff:ff:ff:ff:ff:ff
    altname enp7s0
4: eth2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 52:54:00:41:9f:7a brd ff:ff:ff:ff:ff:ff
    altname enp8s0
```

然后加载UIO模块。

```
1 sudo modprobe uio_pci_generic
```

然后绑定网卡。首先查看设备信息：

```
~ > dpdk-devbind.py -s

Network devices using kernel driver
===================================
0000:01:00.0 'Virtio network device 1041' if=eth0 drv=virtio-pci unused=virtio_pci,uio_pci_generic *Active*
0000:07:00.0 'Virtio network device 1041' if=eth1 drv=virtio-pci unused=virtio_pci,uio_pci_generic
0000:08:00.0 'Virtio network device 1041' if=eth2 drv=virtio-pci unused=virtio_pci,uio_pci_generic

No 'Baseband' devices detected
==============================

No 'Crypto' devices detected
============================

No 'DMA' devices detected
=========================

No 'Eventdev' devices detected
==============================

No 'Mempool' devices detected
=============================

No 'Compress' devices detected
==============================

No 'Misc (rawdev)' devices detected
===================================

No 'Regex' devices detected
===========================
```

这里我选择eth1网卡进行绑定。

```
sudo dpdk-devbind.py -b uio_pci_generic 07:00.0
```

绑定完成后再次查看信息，发现绑定成功。

```
~ > sudo dpdk-devbind.py -b uio_pci_generic 07:00.0
~ > dpdk-devbind.py -s

Network devices using DPDK-compatible driver
============================================
0000:07:00.0 'Virtio network device 1041' drv=uio_pci_generic unused=virtio_pci

Network devices using kernel driver
===================================
0000:01:00.0 'Virtio network device 1041' if=eth0 drv=virtio-pci unused=virtio_pci,uio_pci_generic *Active*
0000:08:00.0 'Virtio network device 1041' if=eth2 drv=virtio-pci unused=virtio_pci,uio_pci_generic

No 'Baseband' devices detected
==============================

No 'Crypto' devices detected
============================

No 'DMA' devices detected
=========================

No 'Eventdev' devices detected
==============================

No 'Mempool' devices detected
=============================

No 'Compress' devices detected
==============================

No 'Misc (rawdev)' devices detected
===================================

No 'Regex' devices detected
===========================
```

2.2.3 编译并测试Helloworld样例程序

```
cd ./dpdk-21.11/examples/helloworld
make
cd build
sudo ./helloworld
```

能够输出正常结果，说明安装成功。

```
~/dpdk-2/e/helloworld > make                                    arch@archlinux 07:50:55
ln -sf helloworld-shared build/helloworld
~/dpdk-2/e/helloworld > cd build                                arch@archlinux 07:50:57
~/dpdk-2/e/h/build > sudo ./helloworld                          arch@archlinux 07:51:02
EAL: Detected CPU lcores: 16
EAL: Detected NUMA nodes: 1
EAL: Detected shared linkage of DPDK
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: No available 1048576 kB hugepages reported
EAL: Probe PCI driver: net_virtio (1af4:1041) device: 0000:01:00.0 (socket 0)
eth_virtio_pci_init(): Failed to init PCI device
EAL: Requested device 0000:01:00.0 cannot be used
EAL: Probe PCI driver: net_virtio (1af4:1041) device: 0000:07:00.0 (socket 0)
EAL: Probe PCI driver: net_virtio (1af4:1041) device: 0000:08:00.0 (socket 0)
eth_virtio_pci_init(): Failed to init PCI device
EAL: Requested device 0000:08:00.0 cannot be used
TELEMETRY: No legacy callbacks, legacy socket not created
hello from core 1
hello from core 2
hello from core 3
hello from core 4
hello from core 5
hello from core 6
hello from core 7
hello from core 8
hello from core 9
hello from core 10
hello from core 11
hello from core 12
hello from core 13
hello from core 14
hello from core 15
hello from core 0
```

3 编译安装l2fwd和pktgen-dpdk

> (3) Compile and run DPDK sample application l2fwd on VM2, then compile and run pktgen-dpdk on VM1. pkgen-dpdk will record the size of the packages VM1 sends and the amount of packages received from VM2, while l2fwd just send back the packages it received from VM1.

作业要求在VM1上运行pktgen-dpdk，在VM2上运行l2fwd。这里我选择在一个虚拟机上安装这两个软件，再将其复制得到另一个虚拟机。

3.1 l2fwd[4]

l2fwd在编译安装DPDK时已经一同安装，这里我们测试它的运行。

```
cd ./dpdk-21.11/build/examples
sudo ./dpdk-l2fwd -c 0x1 -n 4 -- -p 0x1 -T 1
```

这里，`--` 前的参数是EAL（环境抽象层）参数，`--` 后的参数是该应用本身的参数。

具体来说，`-c 0x1` 以十六进制掩码的形式表示CPU的核心，这里代表启用0号核心；`-n 4` 意思是设置4个内存通道。`-p 0x1` 以十六进制掩码的形式表示端口号，这里代表配置0号端口；`-T 1` 代表每过1s刷新一次结果。一段时间后：

```
Port statistics ====================================
Statistics for port 0 ------------------------------
Packets sent:                        11
Packets received:                    11
Packets dropped:                      0
Aggregate statistics ===============================
Total packets sent:                  11
Total packets received:              11
Total packets dropped:                0
====================================================
```

3.2 pktgen-dpdk[5]

3.2.1 编译安装

首先下载最新版本pktgen-dpdk源代码并解压。

```
1 wget https://git.dpdk.org/apps/pktgen-dpdk/snapshot/pktgen-dpdk-pktgen-21.11.0.tar.xz
2 tar xf pktgen-dpdk-pktgen-21.11.0.tar.xz
```

然后编译安装，方法与DPDK相同。

```
~/pktgen-dpdk-pktgen-21.11.0 > meson build                          arch@archlinux 08:06:25
The Meson build system
Version: 0.60.3
Source dir: /home/arch/pktgen-dpdk-pktgen-21.11.0
Build dir: /home/arch/pktgen-dpdk-pktgen-21.11.0/build
Build type: native build
Program cat found: YES (/usr/bin/cat)
Project name: pktgen
Project version: 21.11.0
C compiler for the host machine: cc (gcc 11.1.0 "cc (GCC) 11.1.0")
C linker for the host machine: cc ld.bfd 2.36.1
Host machine cpu family: x86_64
Host machine cpu: x86_64
Compiler for C supports arguments -mavx: YES
Compiler for C supports arguments -mavx2: YES
Compiler for C supports arguments -Wno-pedantic: YES
Compiler for C supports arguments -Wno-format-truncation: YES
Found pkg-config: /usr/bin/pkg-config (1.8.0)
Run-time dependency libdpdk found: YES 21.11.0
Library librte_net_bond found: YES
Program python3 found: YES (/usr/bin/python)
Library rte_net_i40e found: YES
Library rte_net_ixgbe found: YES
Library rte_net_ice found: YES
Library rte_bus_vdev found: YES
Run-time dependency threads found: YES
Library numa found: YES
Library pcap found: YES
Library dl found: YES
Library m found: YES
Program sphinx-build found: NO
Program echo found: YES (/usr/bin/echo)
Build targets in project: 9

Found ninja-1.10.2 at /usr/bin/ninja
~/pktgen-dpdk-pktgen-21.11.0 > cd build                             arch@archlinux 08:06:32
~/p/build > ninja                                                   arch@archlinux 08:06:48
[70/70] Linking target app/pktgen
~/p/build > sudo ninja install                                   3s arch@archlinux 08:06:54
[0/1] Installing files.
Installing app/pktgen to /usr/local/bin
~/p/build > sudo ldconfig                                           arch@archlinux 08:06:59
~/p/build >                                                         arch@archlinux 08:07:07
```

3.2.2 测试运行

```
Cd ./pktgen-dpdk-pktgen-21.11.0/build/app
sudo ./pktgen -c 0x3 -n 4 -- -m "[1].0"
```

这里和l2fwd类似，`--` 前的参数是EAL（环境抽象层）参数，`--` 后的参数是该应用本身的参数。

具体来说，`-c 0x3` 以十六进制掩码的形式表示CPU的核心，这里代表启用0号和1号核心；`-n 4`意思是设置4个内存通道。`-m "[1].0"` 表示1号核心处理0号端口的收发包。特别要注意的是，pktgen-dpdk会使用一个核心作为initial lcore（上述命令中是0号核心），因此至少要启用两个CPU核心，并且处理端口的核心不能为initial lcore，否则报错如下：

```
~/p/build/app > sudo ./pktgen -c 0x3 -n 4 -- -m "[0].0"              arch@archlinux 10:13:30



Copyright(c) <2010-2021>, Intel Corporation. All rights reserved. Powered by DPDK
EAL: Detected CPU lcores: 16
EAL: Detected NUMA nodes: 1
EAL: Detected shared linkage of DPDK
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: No available 1048576 kB hugepages reported
EAL: Probe PCI driver: net_virtio (1af4:1041) device: 0000:01:00.0 (socket 0)
eth_virtio_pci_init(): Failed to init PCI device
EAL: Requested device 0000:01:00.0 cannot be used
EAL: Probe PCI driver: net_virtio (1af4:1041) device: 0000:07:00.0 (socket 0)
EAL: Probe PCI driver: net_virtio (1af4:1041) device: 0000:08:00.0 (socket 0)
eth_virtio_pci_init(): Failed to init PCI device
EAL: Requested device 0000:08:00.0 cannot be used
TELEMETRY: No legacy callbacks, legacy socket not created
*** Error can not use initial lcore for a port
    The initial lcore is 0
```

启动pktgen-dpdk如下，说明安装成功。

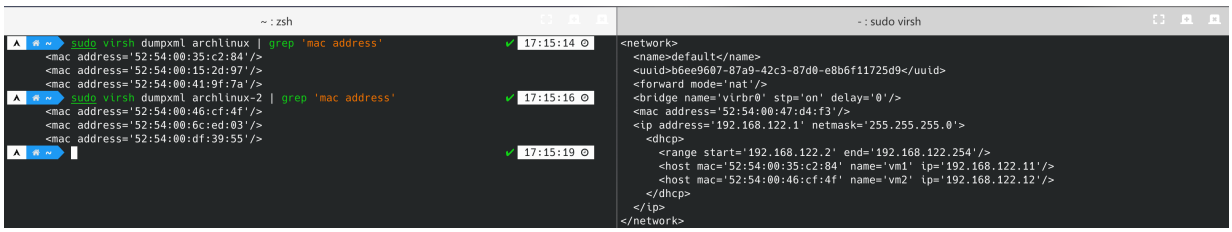
```
| Ports 0-0 of 1   <Main Page>  Copyright(c) <2010-2021>, Intel Corporation
  Flags:Port        : -------Sngl        :0
Link State          :          <--Down-->     ---Total Rate---
Pkts/s Rx           :                   0                    0
       Tx           :                   0                    0
MBits/s Rx/Tx       :                 0/0                  0/0
Pkts/s Rx Max       :                  76                   76
       Tx Max       :                   0                    0
Broadcast           :                   0
Multicast           :                   5
Sizes 64            :                   0
      65-127        :                  12
      128-255       :                  20
      256-511       :                   5
      512-1023      :                   0
      1024-1518     :                   0
Runts/Jumbos        :               207/0
ARP/ICMP Pkts       :                 0/0
Errors Rx/Tx        :                 0/0
Total Rx Pkts       :                 242
      Tx Pkts       :                   0
      Rx/Tx MBs     :                 0/0
TCP Flags           :               .A....
TCP Seq/Ack         :  305419896/305419920
Pattern Type        :                abcd...
Tx Count/% Rate     :        Forever /100%
Pkt Size/Tx Burst   :           64 /  128
TTL/Port Src/Dest   :       64/ 1234/ 5678
Pkt Type:VLAN ID    :      IPv4 / TCP:0001
802.1p CoS/DSCP/IPP :           0/  0/  0
VxLAN Flg/Grp/vid   :     0000/    0/    0
IP  Destination     :          192.168.1.1
    Source          :       192.168.0.1/24
MAC Destination     :    00:00:00:00:00:00
    Source          :    52:54:00:15:2d:97
PCI Vendor/Addr     :    1af4:1041/07:00.0
-- Pktgen 21.11.0 (DPDK 21.11.0)  Powered by DPDK  (pid:1580) -----------------




** Version: DPDK 21.11.0, Command Line Interface without timers
Pktgen:/>
```

4 测试

> (4) Evaluate DPDK's performance of L2 forwarding.

4.1 克隆并配置两个虚拟机[6]

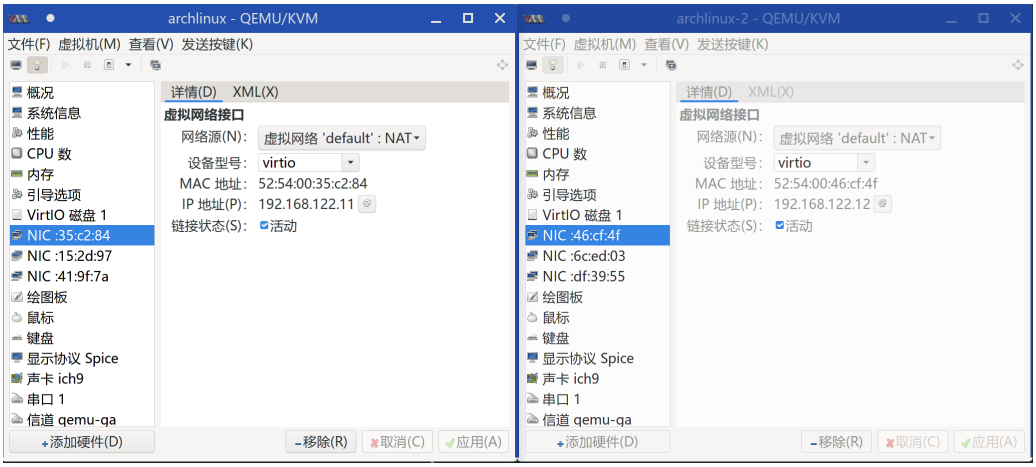
首先复制虚拟机并配置IP地址如下:



然后重新创建 `default` 网络。

```
sudo virsh net-destroy default
sudo virsh  net-start default
```

然后开启两个虚拟机，发现IP地址配置成功。



4.2 测试

4.2.1 测试过程

在VM1上启用pktgen-dpdk后，输入以下命令进行发包。第一行设置了端口0对应的目的地MAC地址（为VM2中绑定到DPDK上的网卡的MAC地址）。第二行开始发包。

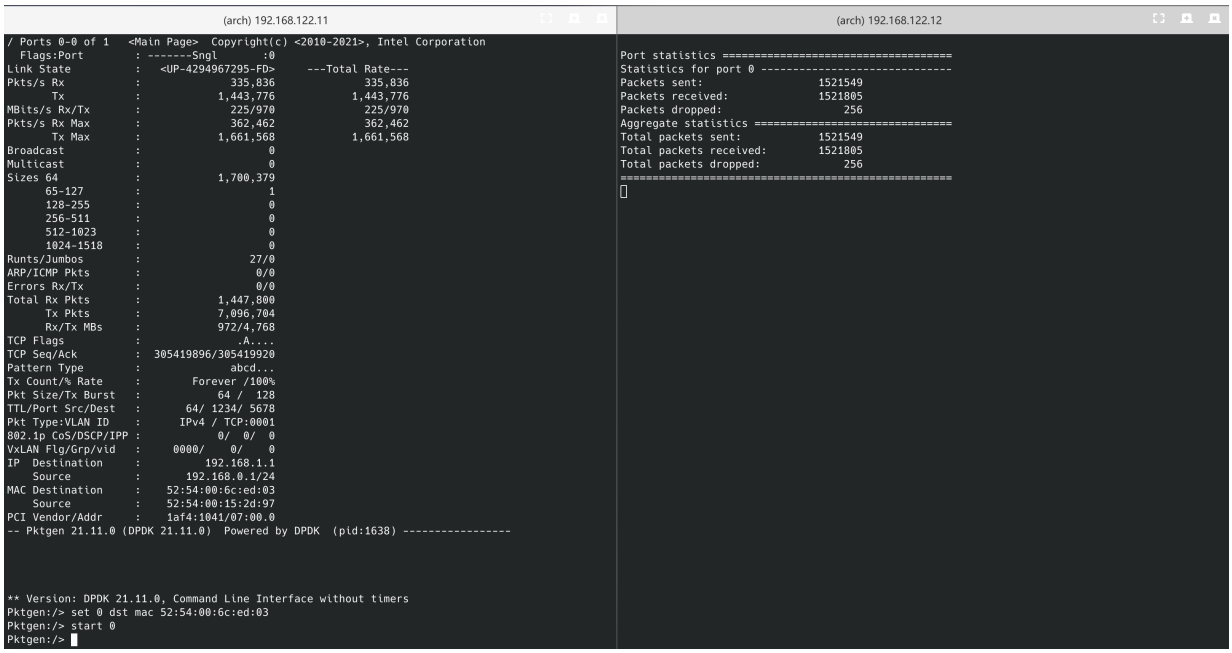
```
set 0 dst mac 52:54:00:6c:ed:03
start 0
```

运行效果如下。可以看到发送速率大约为2.5MPkts/s，但接收速率基本上为0。

(arch) 192.168.122.11

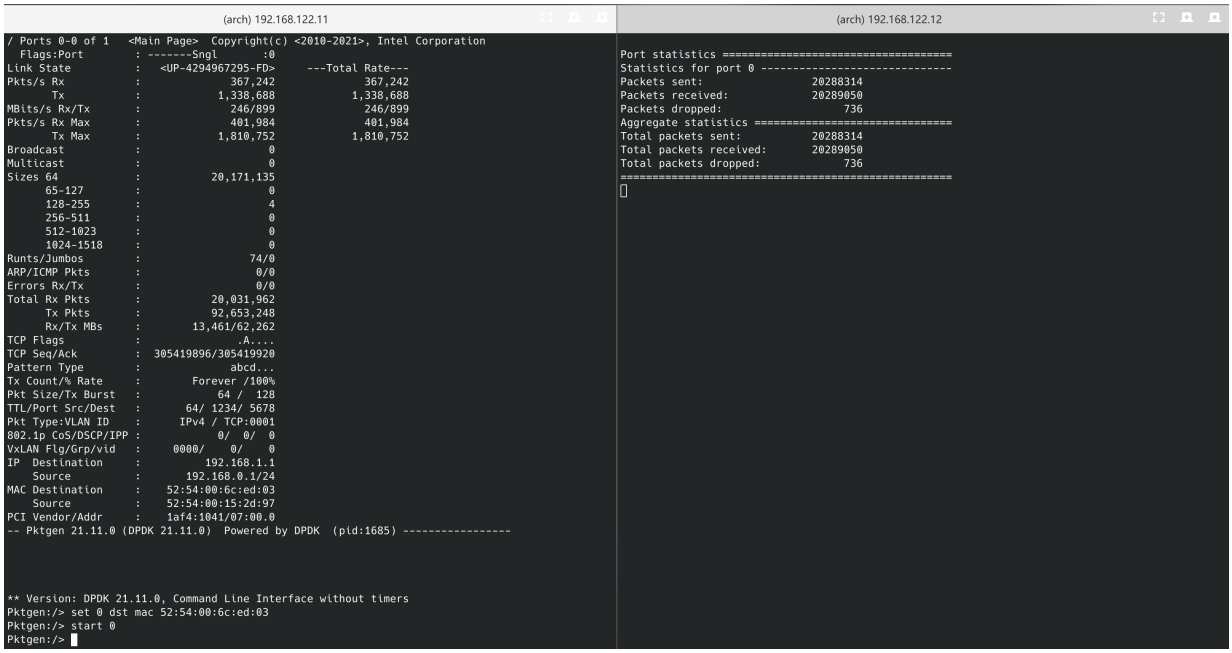
```
- Ports 0-0 of 1   <Main Page>  Copyright(c) <2010-2021>, Intel Corporation
  Flags:Port        : -------Sngl       :0
Link State          :   <UP-4294967295-FD>     ---Total Rate---
Pkts/s Rx           :                    0                    0
       Tx           :            2,440,512            2,440,512
MBits/s Rx/Tx       :              0/1,640              0/1,640
Pkts/s Rx Max       :                    3                    3
       Tx Max       :            2,547,712            2,547,712
Broadcast           :                    0
Multicast           :                    0
Sizes 64            :                    0
      65-127        :                    0
      128-255       :                    0
      256-511       :                    0
      512-1023      :                    0
      1024-1518     :                    0
Runts/Jumbos        :                 16/0
ARP/ICMP Pkts       :                  0/0
Errors Rx/Tx        :                  0/0
Total Rx Pkts       :                   16
      Tx Pkts       :           25,142,080
      Rx/Tx MBs     :              0/16,895
TCP Flags           :                .A....
TCP Seq/Ack         :  305419896/305419920
Pattern Type        :             abcd...
Tx Count/% Rate     :        Forever /100%
Pkt Size/Tx Burst   :            64 /  128
TTL/Port Src/Dest   :       64/ 1234/ 5678
Pkt Type:VLAN ID    :      IPv4 / TCP:0001
802.1p CoS/DSCP/IPP :           0/  0/  0
VxLAN Flg/Grp/vid   :     0000/    0/    0
IP  Destination     :          192.168.1.1
    Source          :       192.168.0.1/24
MAC Destination     :    52:54:00:6c:ed:03
    Source          :    52:54:00:15:2d:97
PCI Vendor/Addr     :    1af4:1041/07:00.0
-- Pktgen 21.11.0 (DPDK 21.11.0)  Powered by DPDK  (pid:1660) -----------------




** Version: DPDK 21.11.0, Command Line Interface without timers
Pktgen:/> set 0 dst mac 52:54:00:6c:ed:03
Pktgen:/> start 0
Pktgen:/>
```

在VM2上开启l2fwd后，由于l2fwd会把收到的包转发回来，VM1上观察到接收速率为0.3MPkts/s左右，发送速率降低到1.5MPkts/s左右：



4.2.2 统计分析

这里我统计l2fwd和pktgen-dpdk在运行了一分钟之后的结果并进行分析。截图如下：



1. VM2上，l2fwd在一分钟内接收到20289050个包，成功转发出去20288314个，丢包73个，丢包率约为 3.6×10^{-6}。
2. VM1上总共发出了92653248个包，但VM2上只接受到20289050个包，说明在传输过程中丢包率较高。可能的原因是，由于发送数据包的速率太快，超出了缓冲区队列的大小，引起丢包。VM2上总共发出了20288314个包，VM1上接受到20031962个包，说明VM2向VM1发送数据包的过程中，丢包率较低。
3. VM1上观察到接收速率为0.3MPkts/s左右，发送速率降低到1.5MPkts/s左右。又因为VM1向VM2发送数据包的过程中丢包率较高，VM2向VM1发送数据包的过程中丢包率较低，说明性能瓶颈很有可能在于VM2上的l2fwd。

5 总结

在本次实验中，我通过测试DPDK的性能，对计算机网络中数据链路层的知识有了更深刻的认识。

6 致谢

感谢我的同学刘梓睿和杨钦崴在本次实验中对我的帮助。

---

1. https://doc.dpdk.org/guides/linux_gsg/build_dpdk.html ↩

2. https://doc.dpdk.org/guides/linux_gsg/linux_drivers.html ↩

3. https://github.com/openvswitch/ovs-issues/issues/191 ↩

4. https://doc.dpdk.org/guides/sample_app_ug/l2_forward_real_virtual.html ↩

5. https://github.com/pktgen/Pktgen-DPDK/blob/dev/INSTALL.md ↩

6. https://huataihuang.gitbooks.io/cloud-atlas/content/virtual/kvm/startup/in_action/kvm_libvirt_static_ip_for_dhcp_and_port_forwarding.html ↩