工程实践与科技创新 III-D 作业 5

姓名:侯晟元　　学号:518021910604

要求:

1， Create two virtual machines on KVM. Compile and install DPDK library on each virtual machine. 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. Finally evaluate DPDK's performance of L2 forwarding.

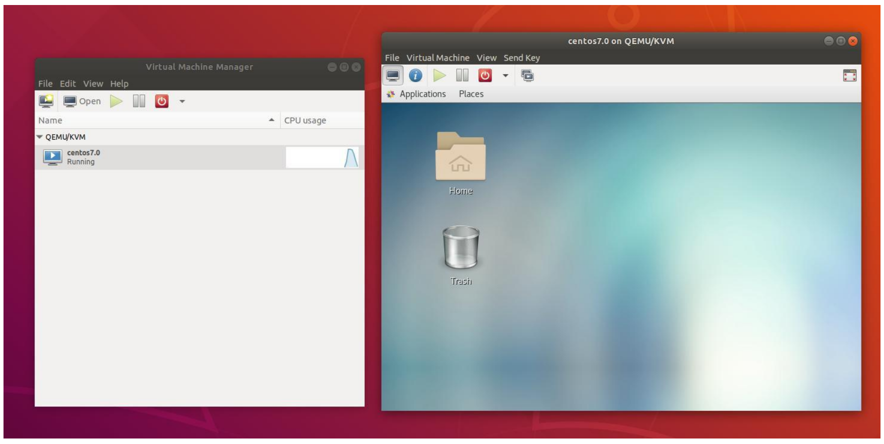
内容简介

数据平面开发套件(DPDK,Data Plane Development Kit)是由 6WIND,Intel 等多家公司开发，主要基于 Linux 系统运行，用于快速数据包处理的函数库与驱动集合，可以极大提高数据处理性能和吞吐量，提高数据平面应用程序的工作效率。

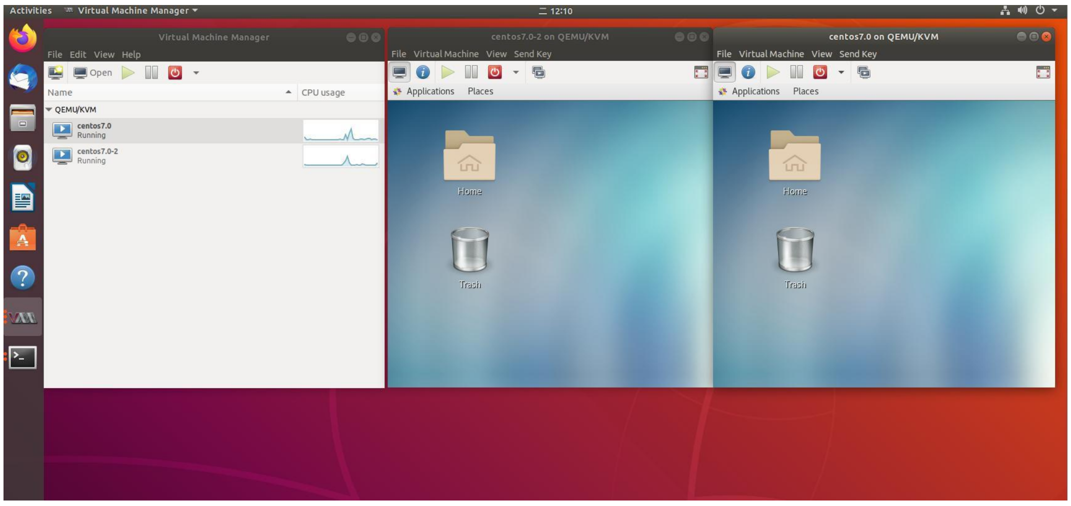
-----百度百科

一.　在 KVM 环境下开启两个虚拟机

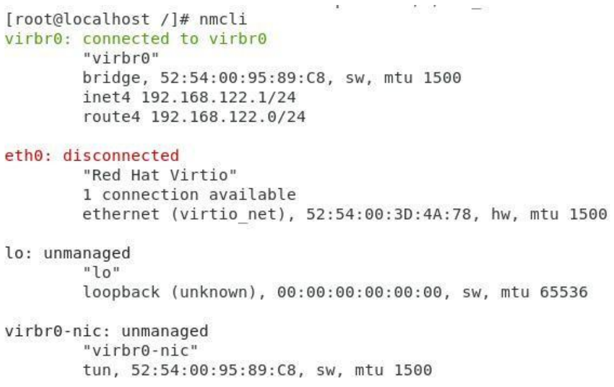
作业二中已经在 QEMU/KVM 环境下创建了一个图形界面 CentOS7.0 虚拟机，截图如下



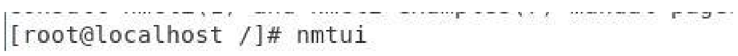
继续创建另一个图形界面 CentOS7.0，内存为 1024M，硬盘为 10G，两个虚拟机截图如下:



以上两张截图可以说明，KVM 中安装两台虚拟机的任务已经实现。Centos7.0on Qemu/KVM 是 VM，Centos7.0-2 on Qemu/KVM 是 VM2。
接下来我们需要确保virt-manager 中的嵌套虚拟机和外网联通，以方便下载安装。同时要确保两个嵌套虚拟机之间可以相互 ping 通，以便可以发送接收包。
使用 nmcli 指令观察两个虚拟机的网卡状况，可以看到虚拟网桥开通，但是本地虚拟网卡 eth0 没有连接，这就导致虚拟机无法连通外网。

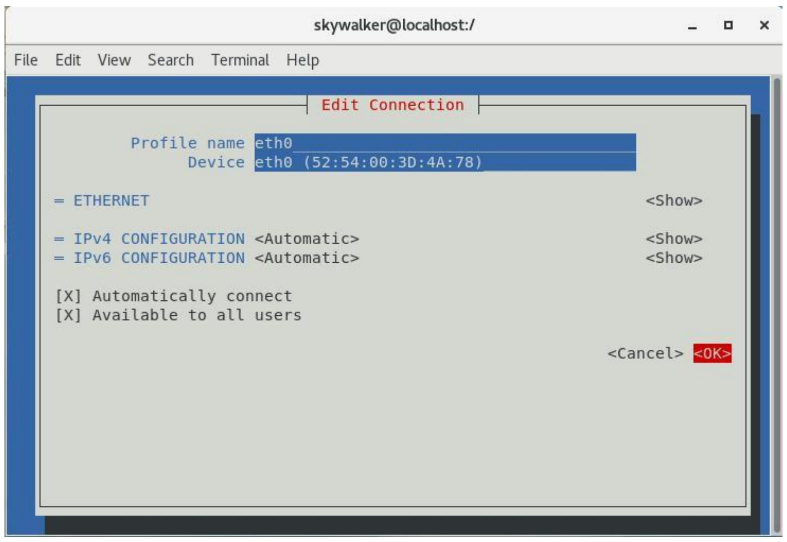


```
[root@localhost /]# nmcli
virbr0: connected to virbr0
        "virbr0"
        bridge, 52:54:00:95:89:C8, sw, mtu 1500
        inet4 192.168.122.1/24
        route4 192.168.122.0/24

eth0: disconnected
        "Red Hat Virtio"
        1 connection available
        ethernet (virtio_net), 52:54:00:3D:4A:78, hw, mtu 1500

lo: unmanaged
        "lo"
        loopback (unknown), 00:00:00:00:00:00, sw, mtu 65536

virbr0-nic: unmanaged
        "virbr0-nic"
        tun, 52:54:00:95:89:C8, sw, mtu 1500
```



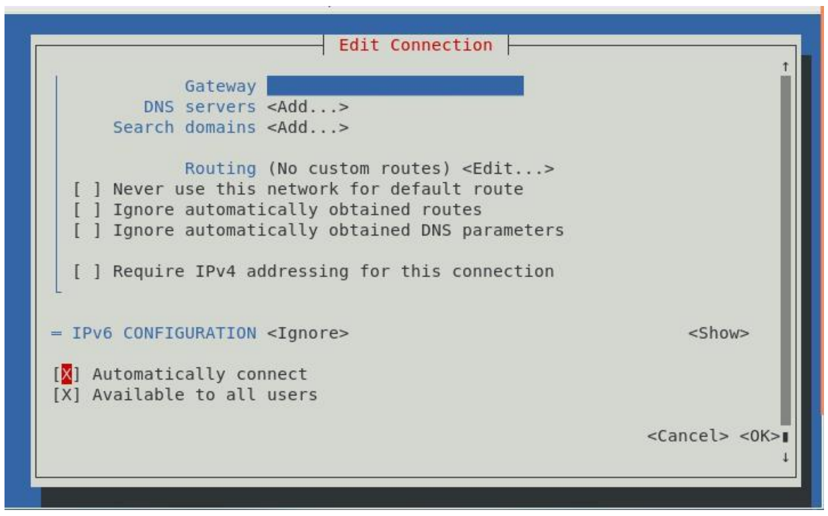
```
[root@localhost /]# nmtui
```

Eth0 选项中编辑设置为自动连接

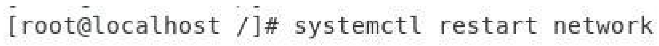


```
skywalker@localhost:/
File  Edit  View  Search  Terminal  Help

                         Edit Connection

         Profile name eth0
               Device eth0 (52:54:00:3D:4A:78)

  = ETHERNET                                          <Show>

  = IPv4 CONFIGURATION <Automatic>                    <Show>
  = IPv6 CONFIGURATION <Automatic>                    <Show>

  [X] Automatically connect
  [X] Available to all users

                                               <Cancel> <OK>
```

vibro 选项编辑中设定为自动连接

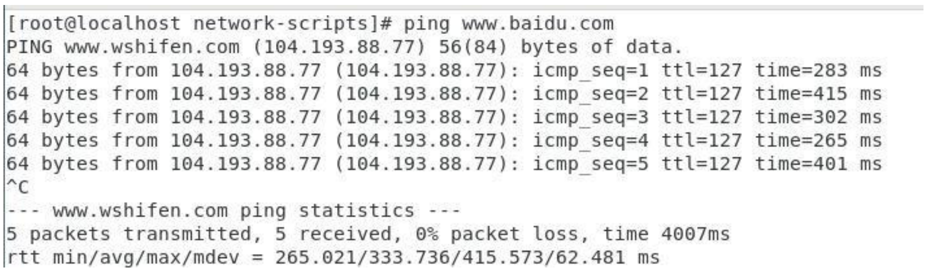


```
                         Edit Connection

            Gateway
        DNS servers <Add...>
     Search domains <Add...>

            Routing (No custom routes) <Edit...>
  [ ] Never use this network for default route
  [ ] Ignore automatically obtained routes
  [ ] Ignore automatically obtained DNS parameters

  [ ] Require IPv4 addressing for this connection

= IPv6 CONFIGURATION <Ignore>                          <Show>

[X] Automatically connect
[X] Available to all users

                                                <Cancel> <OK>
```



```
[root@localhost /]# systemctl restart network
```

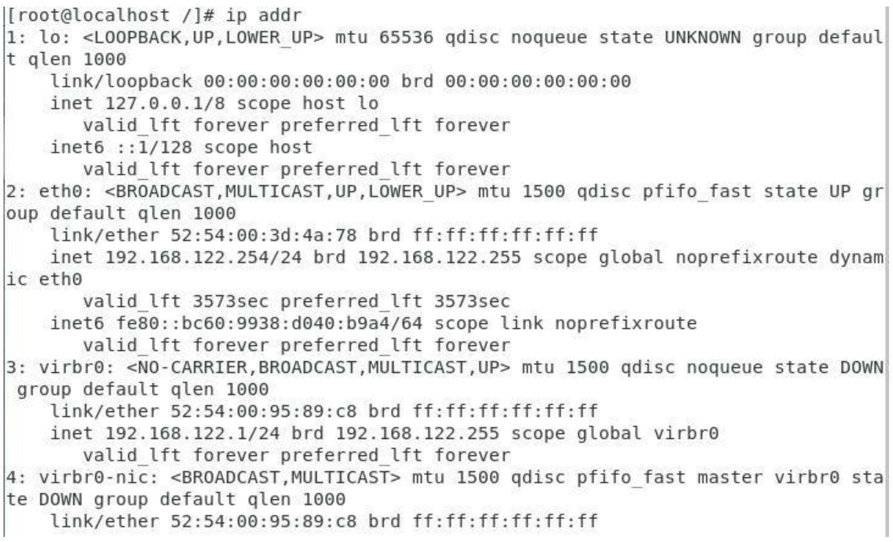
此时可以 ping www.baidu.com 确认虚拟机和外网联通。



```
[root@localhost network-scripts]# ping www.baidu.com
PING www.wshifen.com (104.193.88.77) 56(84) bytes of data.
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=1 ttl=127 time=283 ms
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=2 ttl=127 time=415 ms
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=3 ttl=127 time=302 ms
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=4 ttl=127 time=265 ms
64 bytes from 104.193.88.77 (104.193.88.77): icmp_seq=5 ttl=127 time=401 ms
^C
--- www.wshifen.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 265.021/333.736/415.573/62.481 ms
```

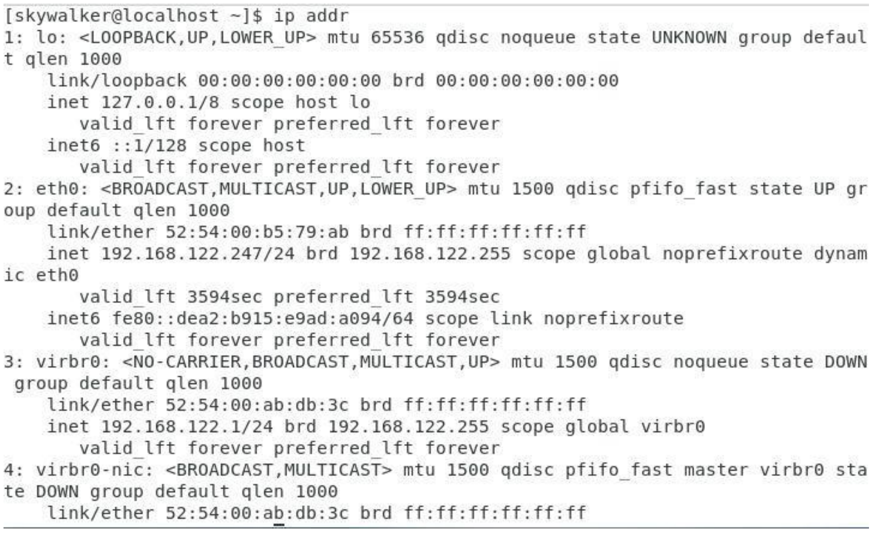
接下来分别获取两台虚拟机的 IP 地址

Centos7.0 on QEMU/KVM(VM)的 IP 地址: 192.168.122.254



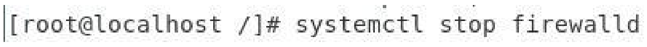
```
[root@localhost /]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP gr
oup default qlen 1000
    link/ether 52:54:00:3d:4a:78 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.254/24 brd 192.168.122.255 scope global noprefixroute dynam
ic eth0
       valid_lft 3573sec preferred_lft 3573sec
    inet6 fe80::bc60:9938:d040:b9a4/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
 group default qlen 1000
    link/ether 52:54:00:95:89:c8 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 sta
te DOWN group default qlen 1000
    link/ether 52:54:00:95:89:c8 brd ff:ff:ff:ff:ff:ff
```

Centos7.0-2 on QEMU/KVM(VM2)的 IP 地址:192.168.122.247



```
[skywalker@localhost ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t 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 pfifo_fast state UP gr
oup default qlen 1000
    link/ether 52:54:00:b5:79:ab brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.247/24 brd 192.168.122.255 scope global noprefixroute dynam
ic eth0
       valid_lft 3594sec preferred_lft 3594sec
    inet6 fe80::dea2:b915:e9ad:a094/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
 group default qlen 1000
    link/ether 52:54:00:ab:db:3c brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
       valid_lft forever preferred_lft forever
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 sta
te DOWN group default qlen 1000
    link/ether 52:54:00:ab:db:3c brd ff:ff:ff:ff:ff:ff
```

关闭两台虚拟机的防火墙

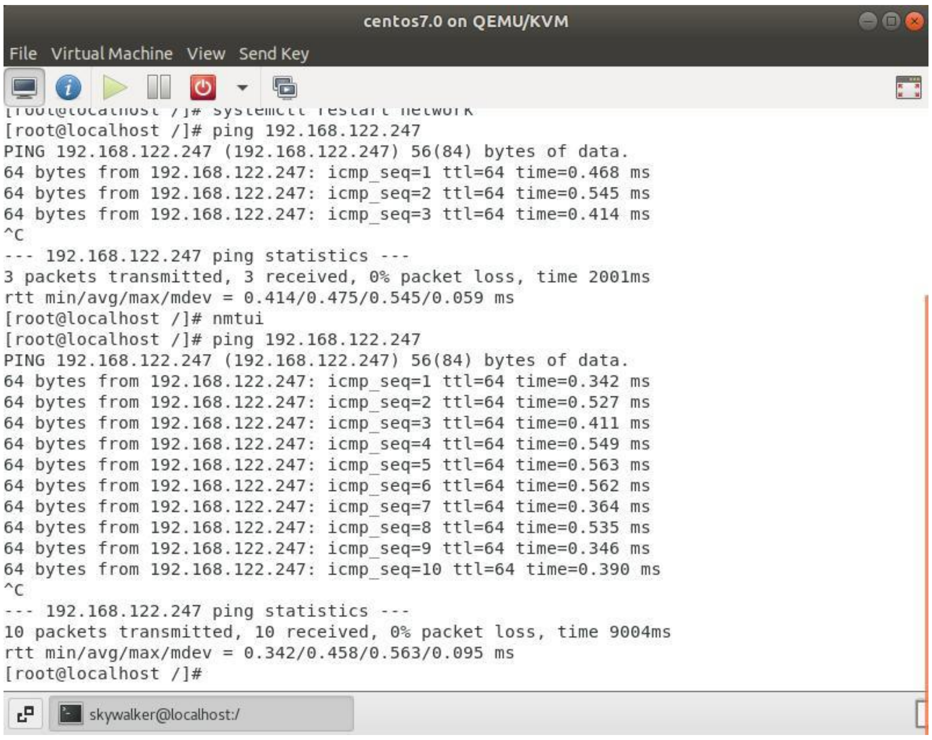


```
[root@localhost /]# systemctl stop firewalld
```

在两台虚拟机中分别输入 ping 指令，测试是否连通。

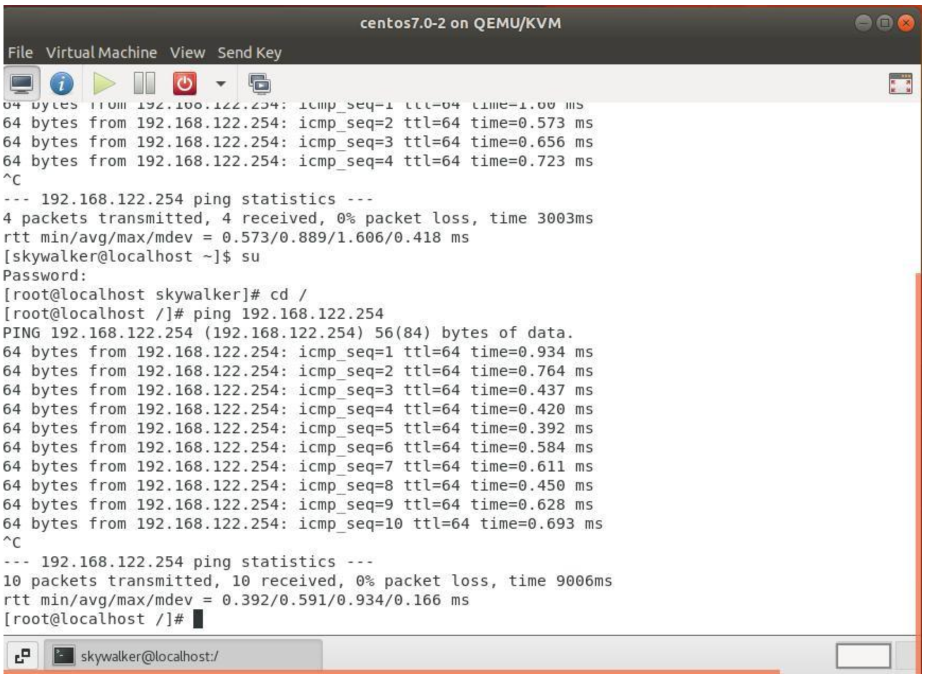
由以下结果可见，两台虚拟机已经成功连通

Centos7.0 on QEMU/KVM(VM) ping VM2 的 IP 地址ping 192.168.122.247



```
[root@localhost /]# systemctl restart network
[root@localhost /]# ping 192.168.122.247
PING 192.168.122.247 (192.168.122.247) 56(84) bytes of data.
64 bytes from 192.168.122.247: icmp_seq=1 ttl=64 time=0.468 ms
64 bytes from 192.168.122.247: icmp_seq=2 ttl=64 time=0.545 ms
64 bytes from 192.168.122.247: icmp_seq=3 ttl=64 time=0.414 ms
^C
--- 192.168.122.247 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.414/0.475/0.545/0.059 ms
[root@localhost /]# nmtui
[root@localhost /]# ping 192.168.122.247
PING 192.168.122.247 (192.168.122.247) 56(84) bytes of data.
64 bytes from 192.168.122.247: icmp_seq=1 ttl=64 time=0.342 ms
64 bytes from 192.168.122.247: icmp_seq=2 ttl=64 time=0.527 ms
64 bytes from 192.168.122.247: icmp_seq=3 ttl=64 time=0.411 ms
64 bytes from 192.168.122.247: icmp_seq=4 ttl=64 time=0.549 ms
64 bytes from 192.168.122.247: icmp_seq=5 ttl=64 time=0.563 ms
64 bytes from 192.168.122.247: icmp_seq=6 ttl=64 time=0.562 ms
64 bytes from 192.168.122.247: icmp_seq=7 ttl=64 time=0.364 ms
64 bytes from 192.168.122.247: icmp_seq=8 ttl=64 time=0.535 ms
64 bytes from 192.168.122.247: icmp_seq=9 ttl=64 time=0.346 ms
64 bytes from 192.168.122.247: icmp_seq=10 ttl=64 time=0.390 ms
^C
--- 192.168.122.247 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9004ms
rtt min/avg/max/mdev = 0.342/0.458/0.563/0.095 ms
[root@localhost /]#
```

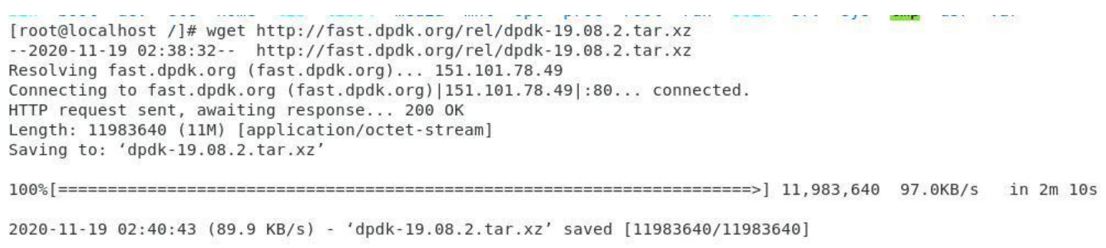
Centos7.0-2 on QEMU/KVM(VM2) ping VM 的 IP 地址
ping 192.168.122.254



```
64 bytes from 192.168.122.254: icmp_seq=1 ttl=64 time=1.60 ms
64 bytes from 192.168.122.254: icmp_seq=2 ttl=64 time=0.573 ms
64 bytes from 192.168.122.254: icmp_seq=3 ttl=64 time=0.656 ms
64 bytes from 192.168.122.254: icmp_seq=4 ttl=64 time=0.723 ms
^C
--- 192.168.122.254 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 0.573/0.889/1.606/0.418 ms
[skywalker@localhost ~]$ su
Password:
[root@localhost skywalker]# cd /
[root@localhost /]# ping 192.168.122.254
PING 192.168.122.254 (192.168.122.254) 56(84) bytes of data.
64 bytes from 192.168.122.254: icmp_seq=1 ttl=64 time=0.934 ms
64 bytes from 192.168.122.254: icmp_seq=2 ttl=64 time=0.764 ms
64 bytes from 192.168.122.254: icmp_seq=3 ttl=64 time=0.437 ms
64 bytes from 192.168.122.254: icmp_seq=4 ttl=64 time=0.420 ms
64 bytes from 192.168.122.254: icmp_seq=5 ttl=64 time=0.392 ms
64 bytes from 192.168.122.254: icmp_seq=6 ttl=64 time=0.584 ms
64 bytes from 192.168.122.254: icmp_seq=7 ttl=64 time=0.611 ms
64 bytes from 192.168.122.254: icmp_seq=8 ttl=64 time=0.450 ms
64 bytes from 192.168.122.254: icmp_seq=9 ttl=64 time=0.628 ms
64 bytes from 192.168.122.254: icmp_seq=10 ttl=64 time=0.693 ms
^C
--- 192.168.122.254 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9006ms
rtt min/avg/max/mdev = 0.392/0.591/0.934/0.166 ms
[root@localhost /]#
```

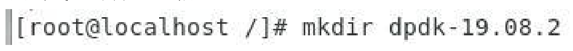
二.在两个QEMU/KVM 虚拟机上源码编译安装DPDK

wget 方式下载 dpdk-19.08.2 版本源代码
wget http://fast.dpdk.org/rel/dodk-19.08.2.tar.xz

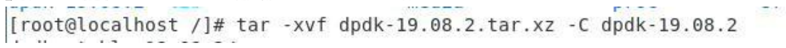


```
[root@localhost /]# wget http://fast.dpdk.org/rel/dpdk-19.08.2.tar.xz
--2020-11-19 02:38:32--  http://fast.dpdk.org/rel/dpdk-19.08.2.tar.xz
Resolving fast.dpdk.org (fast.dpdk.org)... 151.101.78.49
Connecting to fast.dpdk.org (fast.dpdk.org)|151.101.78.49|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11983640 (11M) [application/octet-stream]
Saving to: ‘dpdk-19.08.2.tar.xz’

100%[======================================================================>] 11,983,640  97.0KB/s   in 2m 10s

2020-11-19 02:40:43 (89.9 KB/s) - ‘dpdk-19.08.2.tar.xz’ saved [11983640/11983640]
```

源代码解压缩

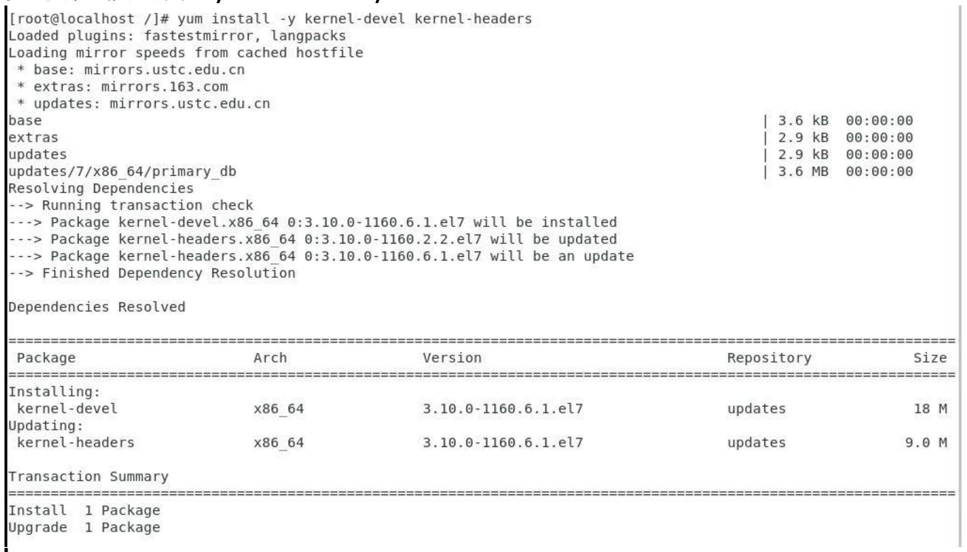


```
[root@localhost /]# mkdir dpdk-19.08.2
```

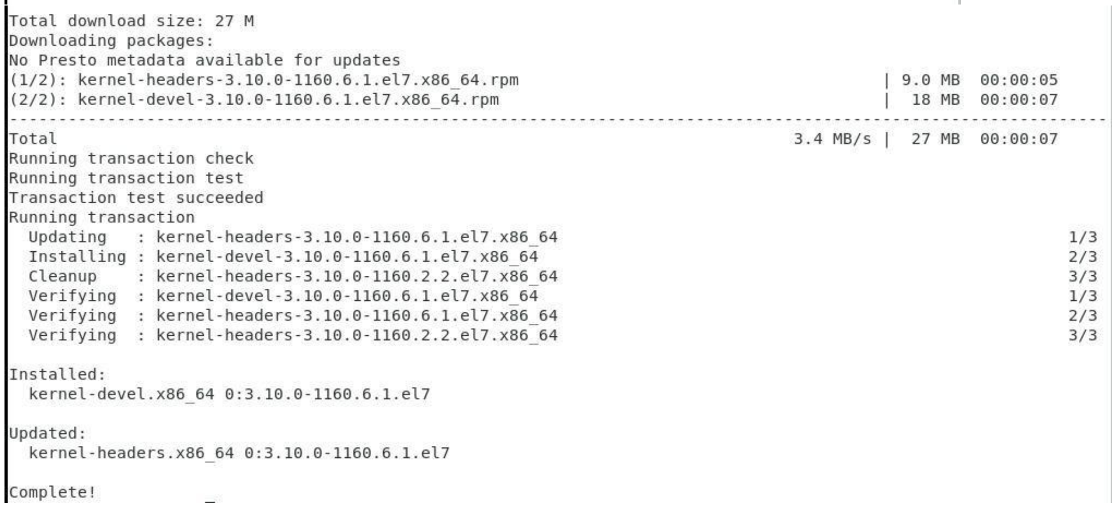


```
[root@localhost /]# tar -xvf dpdk-19.08.2.tar.xz -C dpdk-19.08.2
```

安装依赖的库 yum install -y kernel-devel kernel-headers

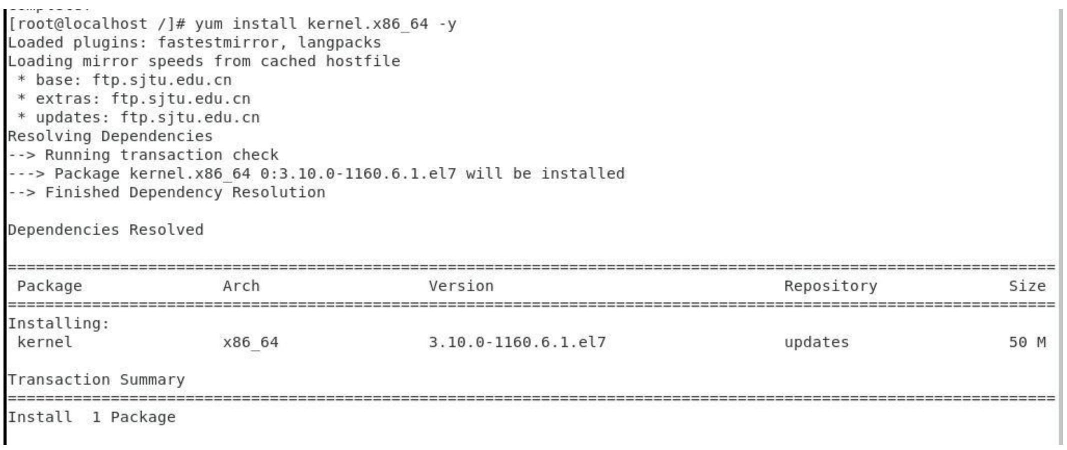
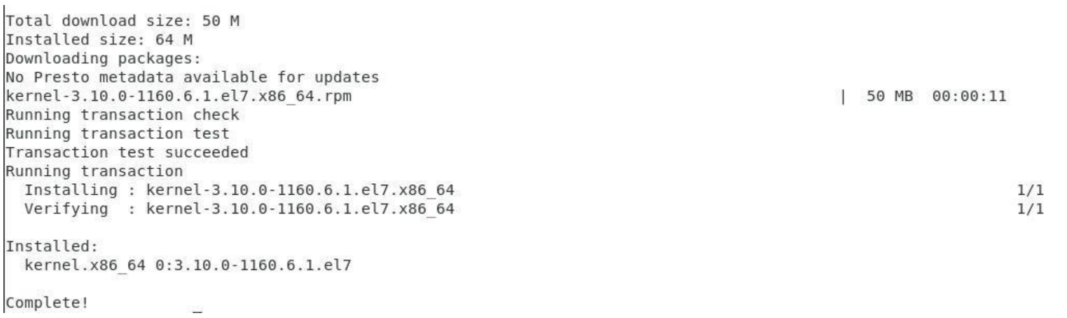


```
[root@localhost /]# yum install -y kernel-devel kernel-headers
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.ustc.edu.cn
 * extras: mirrors.163.com
 * updates: mirrors.ustc.edu.cn
base                                                                                  | 3.6 kB  00:00:00
extras                                                                                | 2.9 kB  00:00:00
updates                                                                               | 2.9 kB  00:00:00
updates/7/x86_64/primary_db                                                           | 3.6 MB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package kernel-devel.x86_64 0:3.10.0-1160.6.1.el7 will be installed
---> Package kernel-headers.x86_64 0:3.10.0-1160.2.2.el7 will be updated
---> Package kernel-headers.x86_64 0:3.10.0-1160.6.1.el7 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

==============================================================================================================
 Package                      Arch                Version                              Repository        Size
==============================================================================================================
Installing:
 kernel-devel                 x86_64              3.10.0-1160.6.1.el7                  updates           18 M
Updating:
 kernel-headers               x86_64              3.10.0-1160.6.1.el7                  updates          9.0 M

Transaction Summary
==============================================================================================================
Install  1 Package
Upgrade  1 Package
```



```
Total download size: 27 M
Downloading packages:
No Presto metadata available for updates
(1/2): kernel-headers-3.10.0-1160.6.1.el7.x86_64.rpm                                 | 9.0 MB  00:00:05
(2/2): kernel-devel-3.10.0-1160.6.1.el7.x86_64.rpm                                   |  18 MB  00:00:07
--------------------------------------------------------------------------------------------------------------
Total                                                                       3.4 MB/s |  27 MB  00:00:07
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating   : kernel-headers-3.10.0-1160.6.1.el7.x86_64                                                  1/3
  Installing : kernel-devel-3.10.0-1160.6.1.el7.x86_64                                                    2/3
  Cleanup    : kernel-headers-3.10.0-1160.2.2.el7.x86_64                                                  3/3
  Verifying  : kernel-devel-3.10.0-1160.6.1.el7.x86_64                                                    1/3
  Verifying  : kernel-headers-3.10.0-1160.6.1.el7.x86_64                                                  2/3
  Verifying  : kernel-headers-3.10.0-1160.2.2.el7.x86_64                                                  3/3

Installed:
  kernel-devel.x86_64 0:3.10.0-1160.6.1.el7

Updated:
  kernel-headers.x86_64 0:3.10.0-1160.6.1.el7

Complete!
```

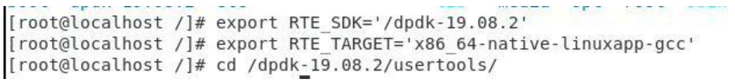
安装 yum install kernel.x86_64 -y


```
[root@localhost /]# yum install kernel.x86_64 -y
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.sjtu.edu.cn
 * extras: ftp.sjtu.edu.cn
 * updates: ftp.sjtu.edu.cn
Resolving Dependencies
--> Running transaction check
---> Package kernel.x86_64 0:3.10.0-1160.6.1.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

==========================================================================================================
 Package               Arch                Version                        Repository              Size
==========================================================================================================
Installing:
 kernel                x86_64              3.10.0-1160.6.1.el7            updates                 50 M

Transaction Summary
==========================================================================================================
Install  1 Package
```

```
Total download size: 50 M
Installed size: 64 M
Downloading packages:
No Presto metadata available for updates
kernel-3.10.0-1160.6.1.el7.x86_64.rpm                                                  |  50 MB  00:00:11
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : kernel-3.10.0-1160.6.1.el7.x86_64                                                      1/1
  Verifying  : kernel-3.10.0-1160.6.1.el7.x86_64                                                      1/1

Installed:
  kernel.x86_64 0:3.10.0-1160.6.1.el7

Complete!
```
配置DPDK的安装环境，设置环境变量(RTE_SDK为DPDK的安装目录，RTE_TARGET为DPDK的目标环境目录)，进入目录下

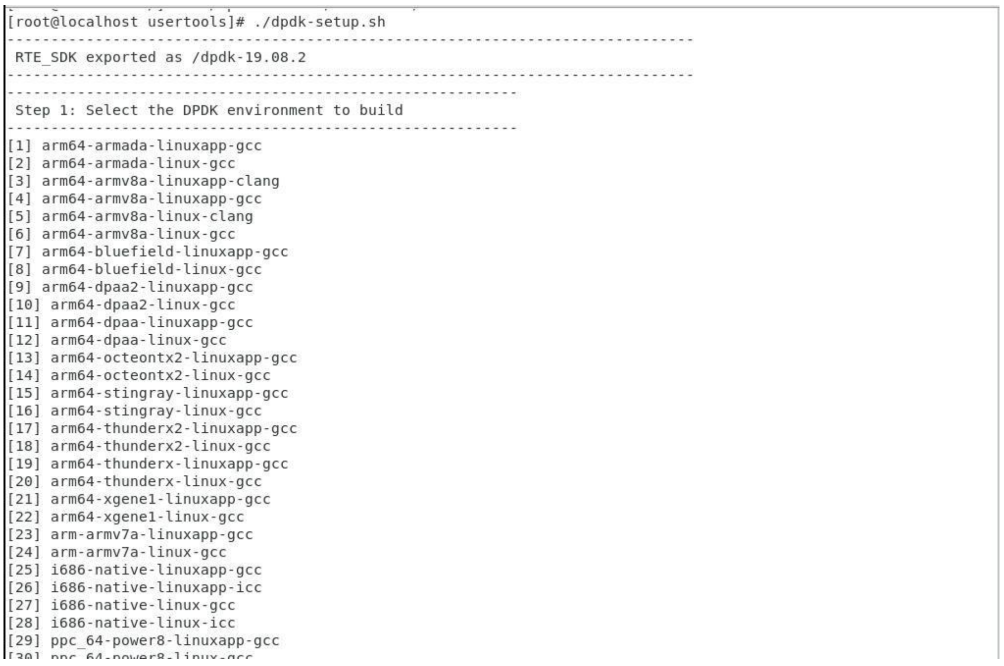
export RTE_SDK='/dpdk-19.08.2'

export RTE_TARGET='x86_64-native-linuxapp-gcc'

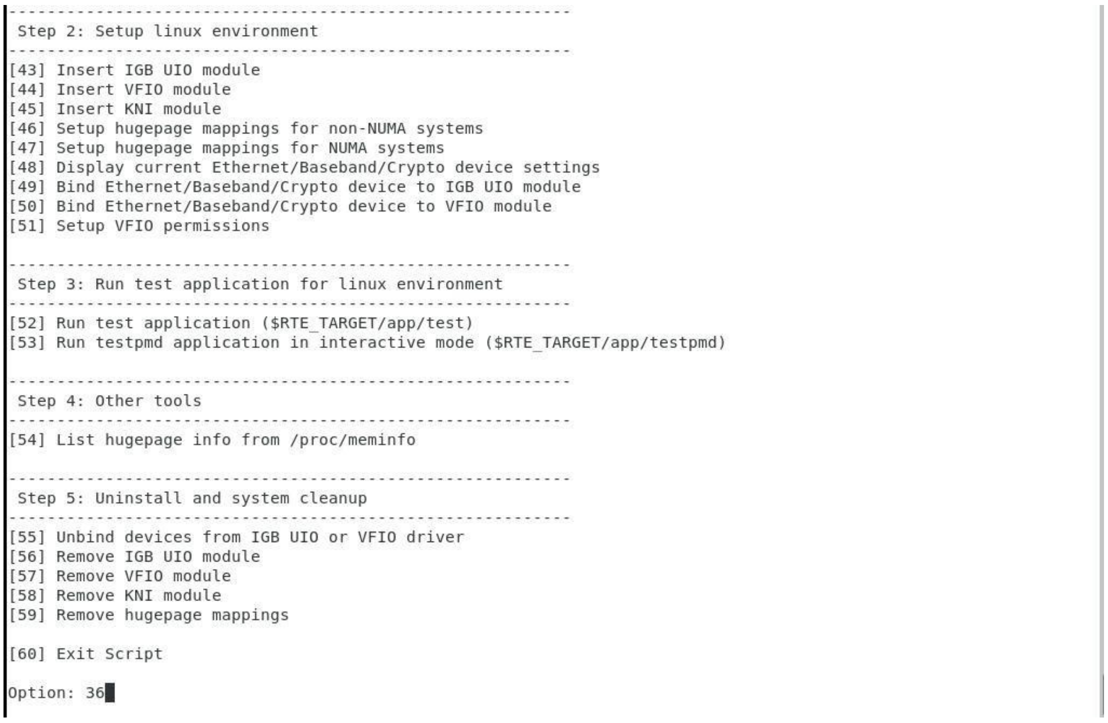
cd /dpdk-19.08.2/usertools/


```
[root@localhost /]# export RTE_SDK='/dpdk-19.08.2'
[root@localhost /]# export RTE_TARGET='x86_64-native-linuxapp-gcc'
[root@localhost /]# cd /dpdk-19.08.2/usertools/
```
编译安装，运行脚本文件

./dpdk-setup.sh

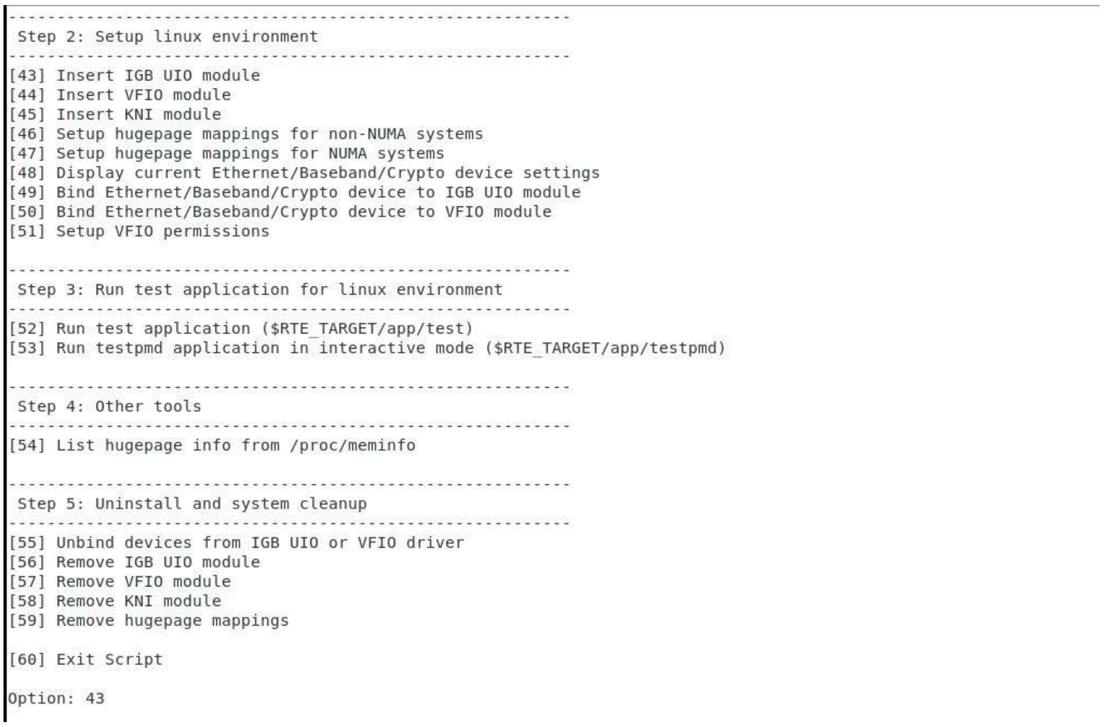

```
[root@localhost usertools]# ./dpdk-setup.sh
------------------------------------------------------------------------------
 RTE_SDK exported as /dpdk-19.08.2
------------------------------------------------------------------------------
----------------------------------------------------------
 Step 1: Select the DPDK environment to build
----------------------------------------------------------
[1] arm64-armada-linuxapp-gcc
[2] arm64-armada-linux-gcc
[3] arm64-armv8a-linuxapp-clang
[4] arm64-armv8a-linuxapp-gcc
[5] arm64-armv8a-linux-clang
[6] arm64-armv8a-linux-gcc
[7] arm64-bluefield-linuxapp-gcc
[8] arm64-bluefield-linux-gcc
[9] arm64-dpaa2-linuxapp-gcc
[10] arm64-dpaa2-linux-gcc
[11] arm64-dpaa-linuxapp-gcc
[12] arm64-dpaa-linux-gcc
[13] arm64-octeontx2-linuxapp-gcc
[14] arm64-octeontx2-linux-gcc
[15] arm64-stingray-linuxapp-gcc
[16] arm64-stingray-linux-gcc
[17] arm64-thunderx2-linuxapp-gcc
[18] arm64-thunderx2-linux-gcc
[19] arm64-thunderx-linuxapp-gcc
[20] arm64-thunderx-linux-gcc
[21] arm64-xgene1-linuxapp-gcc
[22] arm64-xgene1-linux-gcc
[23] arm-armv7a-linuxapp-gcc
[24] arm-armv7a-linux-gcc
[25] i686-native-linuxapp-gcc
[26] i686-native-linuxapp-icc
[27] i686-native-linux-gcc
[28] i686-native-linux-icc
[29] ppc_64-power8-linuxapp-gcc
```

第一步选择 36 配置编译环境为 x86_64-native-linuxapp-gcc



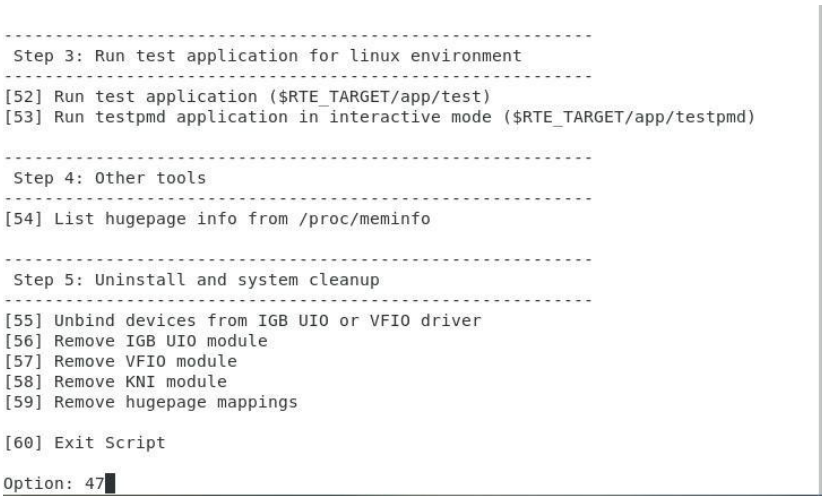
```
----------------------------------------------------------
 Step 2: Setup linux environment
----------------------------------------------------------
[43] Insert IGB UIO module
[44] Insert VFIO module
[45] Insert KNI module
[46] Setup hugepage mappings for non-NUMA systems
[47] Setup hugepage mappings for NUMA systems
[48] Display current Ethernet/Baseband/Crypto device settings
[49] Bind Ethernet/Baseband/Crypto device to IGB UIO module
[50] Bind Ethernet/Baseband/Crypto device to VFIO module
[51] Setup VFIO permissions

----------------------------------------------------------
 Step 3: Run test application for linux environment
----------------------------------------------------------
[52] Run test application ($RTE_TARGET/app/test)
[53] Run testpmd application in interactive mode ($RTE_TARGET/app/testpmd)

----------------------------------------------------------
 Step 4: Other tools
----------------------------------------------------------
[54] List hugepage info from /proc/meminfo

----------------------------------------------------------
 Step 5: Uninstall and system cleanup
----------------------------------------------------------
[55] Unbind devices from IGB UIO or VFIO driver
[56] Remove IGB UIO module
[57] Remove VFIO module
[58] Remove KNI module
[59] Remove hugepage mappings

[60] Exit Script

Option: 36
```

选择 43，即硬件支持的模式 启动 IGB UIO 驱动

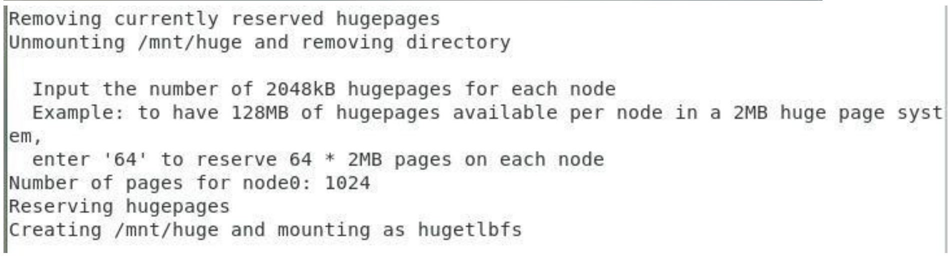


```
----------------------------------------------------------
 Step 2: Setup linux environment
----------------------------------------------------------
[43] Insert IGB UIO module
[44] Insert VFIO module
[45] Insert KNI module
[46] Setup hugepage mappings for non-NUMA systems
[47] Setup hugepage mappings for NUMA systems
[48] Display current Ethernet/Baseband/Crypto device settings
[49] Bind Ethernet/Baseband/Crypto device to IGB UIO module
[50] Bind Ethernet/Baseband/Crypto device to VFIO module
[51] Setup VFIO permissions

----------------------------------------------------------
 Step 3: Run test application for linux environment
----------------------------------------------------------
[52] Run test application ($RTE_TARGET/app/test)
[53] Run testpmd application in interactive mode ($RTE_TARGET/app/testpmd)

----------------------------------------------------------
 Step 4: Other tools
----------------------------------------------------------
[54] List hugepage info from /proc/meminfo

----------------------------------------------------------
 Step 5: Uninstall and system cleanup
----------------------------------------------------------
[55] Unbind devices from IGB UIO or VFIO driver
[56] Remove IGB UIO module
[57] Remove VFIO module
[58] Remove KNI module
[59] Remove hugepage mappings

[60] Exit Script

Option: 43
```

设置大页内存，配置大页的个数，提示输入 512

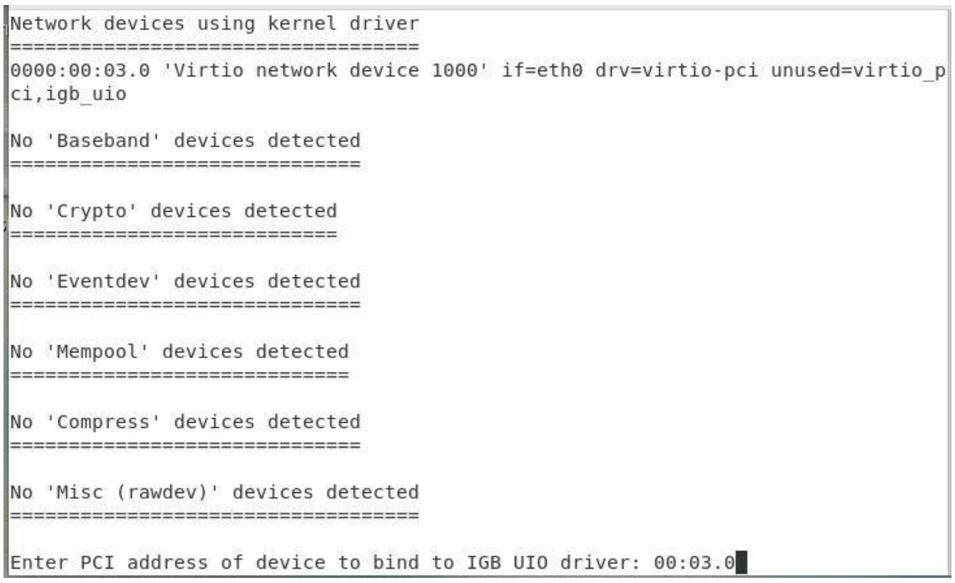


```
----------------------------------------------------------
 Step 3: Run test application for linux environment
----------------------------------------------------------
[52] Run test application ($RTE_TARGET/app/test)
[53] Run testpmd application in interactive mode ($RTE_TARGET/app/testpmd)

----------------------------------------------------------
 Step 4: Other tools
----------------------------------------------------------
[54] List hugepage info from /proc/meminfo

----------------------------------------------------------
 Step 5: Uninstall and system cleanup
----------------------------------------------------------
[55] Unbind devices from IGB UIO or VFIO driver
[56] Remove IGB UIO module
[57] Remove VFIO module
[58] Remove KNI module
[59] Remove hugepage mappings

[60] Exit Script

Option: 47
```



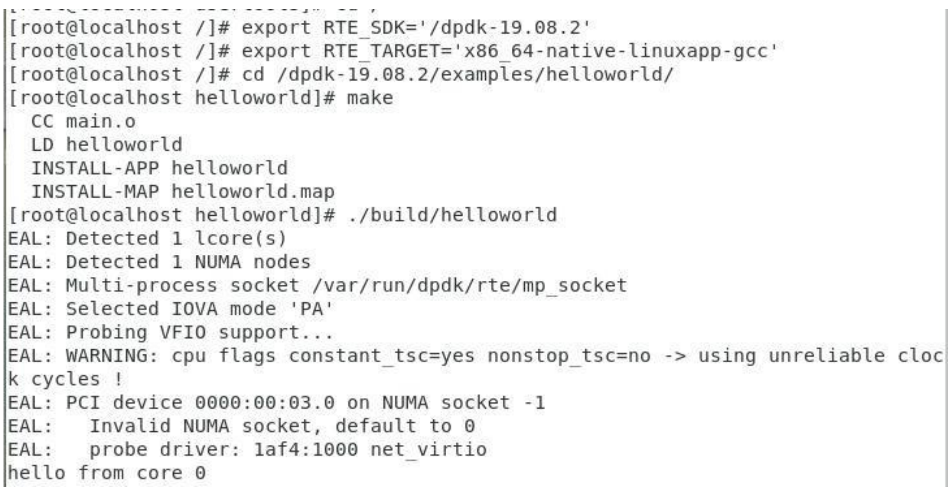
```
Removing currently reserved hugepages
Unmounting /mnt/huge and removing directory

  Input the number of 2048kB hugepages for each node
  Example: to have 128MB of hugepages available per node in a 2MB huge page syst
em,
  enter '64' to reserve 64 * 2MB pages on each node
Number of pages for node0: 1024
Reserving hugepages
Creating /mnt/huge and mounting as hugetlbfs
```

选择 49，对 IGB UIO 绑定网卡，绑定网卡前需要先停掉该网卡此处我用的是本地网卡 eth0，因此需要先执行 ifconfig eth0 down
直接输入 00:03.0 对应的就是 eth0 网卡

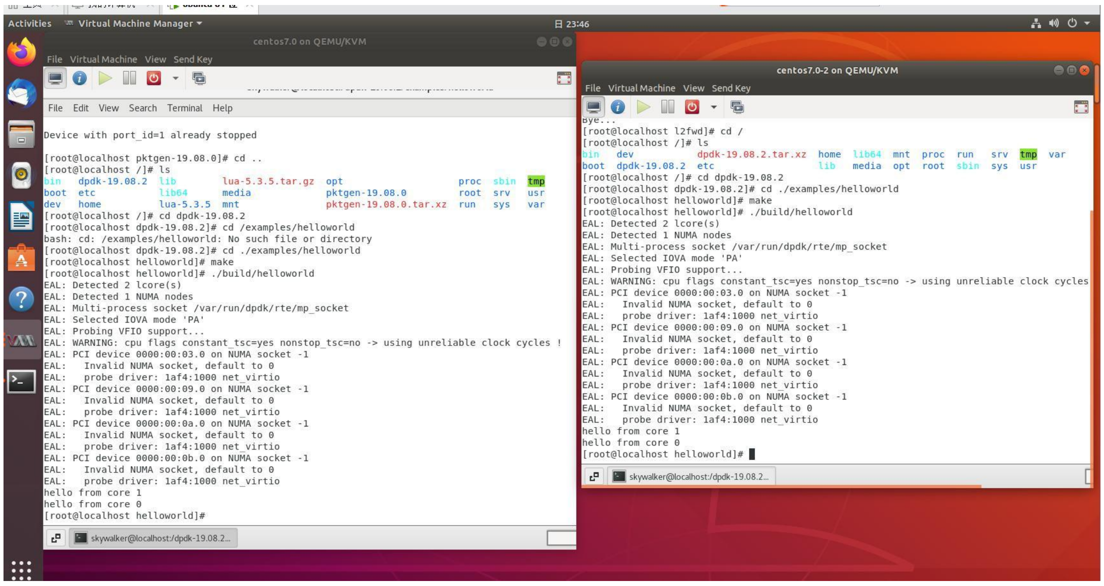


```
Network devices using kernel driver
===================================
0000:00:03.0 'Virtio network device 1000' if=eth0 drv=virtio-pci unused=virtio_p
ci,igb_uio

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

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

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

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

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

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

Enter PCI address of device to bind to IGB UIO driver: 00:03.0
```

到此为止，我们的环境已经安装好，接下来通过 helloworld 程序检验 DPDK 是否安装完成。
首先编译代码 cd /dpdk-19.08.2/examples/helloworld
make
运行可执行文件
./build/helloworld



```
[root@localhost /]# export RTE_SDK='/dpdk-19.08.2'
[root@localhost /]# export RTE_TARGET='x86_64-native-linuxapp-gcc'
[root@localhost /]# cd /dpdk-19.08.2/examples/helloworld/
[root@localhost helloworld]# make
  CC main.o
  LD helloworld
  INSTALL-APP helloworld
  INSTALL-MAP helloworld.map
[root@localhost helloworld]# ./build/helloworld
EAL: Detected 1 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable cloc
k cycles !
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
hello from core 0
```


Activities
Virtual Machine Manager
23:46
centos7.0 on QEMU/KVM
File Virtual Machine View Send Key
File Edit View Search Terminal Help
skywalker@localhost:/dpdk-19.08.2...
centos7.0-2 on QEMU/KVM

```
Device with port_id=1 already stopped

[root@localhost pktgen-19.08.0]# cd ..
[root@localhost /]# ls
bin   dpdk-19.08.2  lib         lua-5.3.5.tar.gz  opt                    proc  sbin  tmp
boot  etc           lib64       media             pktgen-19.08.0         root  srv   usr
dev   home          lua-5.3.5   mnt               pktgen-19.08.0.tar.xz  run   sys   var
[root@localhost /]# cd dpdk-19.08.2
[root@localhost dpdk-19.08.2]# cd /examples/helloworld
bash: cd: /examples/helloworld: No such file or directory
[root@localhost dpdk-19.08.2]# cd ./examples/helloworld
[root@localhost helloworld]# make
[root@localhost helloworld]# ./build/helloworld
EAL: Detected 2 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable clock cycles !
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:09.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0a.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0b.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
hello from core 1
hello from core 0
[root@localhost helloworld]#
```

```
Bye...
[root@localhost l2fwd]# cd /
[root@localhost /]# ls
bin   dev           dpdk-19.08.2.tar.xz  home  lib64  mnt   proc  run   srv  tmp  var
boot  dpdk-19.08.2  etc                  lib   media  opt   root  sbin  sys  usr
[root@localhost /]# cd dpdk-19.08.2
[root@localhost dpdk-19.08.2]# cd ./examples/helloworld
[root@localhost helloworld]# make
[root@localhost helloworld]# ./build/helloworld
EAL: Detected 2 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable clock cycles
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:09.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0a.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0b.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
hello from core 1
hello from core 0
[root@localhost helloworld]#
```

可以看到 helloworld 程序成功运行，正确输出了”hello from core 0”，说明我们 DPDK 的编译安装已经完成。

三. VM2 上编译并运行DPDK 样例程序 l2fwd

接下来进入测试阶段，首先在 VM2 上编译运行 l2fwd

运行指令./build/l2fwd -c 0x1 -n 4 -- -p 0x1 -q 1

L2fwd 参数解析:

./build/l2fwd [EAL options] -- -p PORTMASK [-q NQ -T t]

EAL(DPDK 环境抽象层参数)形式必须为-c COREMASK -n NUM

COREMASK 是一个十六进制的掩码表示分配的逻辑内核数量

NUM 表示一个十进制整数表示每个逻辑内核的内存通道数量

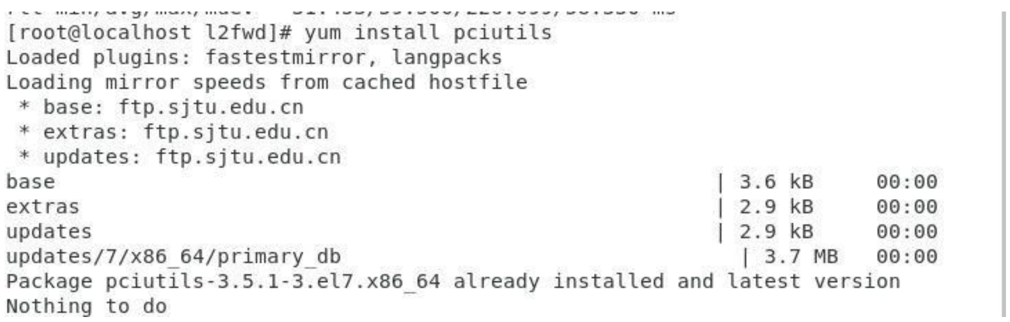
L2fwd 的应用程序参数

-p PORTMASK:一个十六进制位掩码表示分配的端口的数量，例如 0x3 表示分配端口 0 和 1

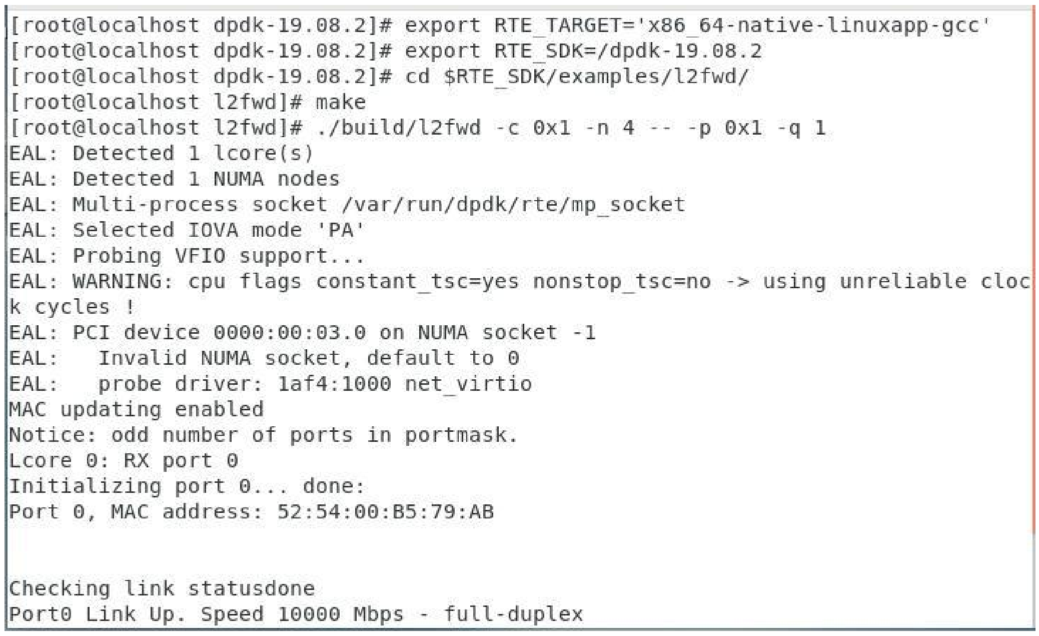
-q NQ NQ 表示分配给每个逻辑内核的收发队列数量

-T t t 表示统计数据打印到屏幕上的时间间隔

测试指令含义即为分配 1 个逻辑内核，每个逻辑内核四个内存通道，1 个收发队列，分配一个端口(端口 0)。

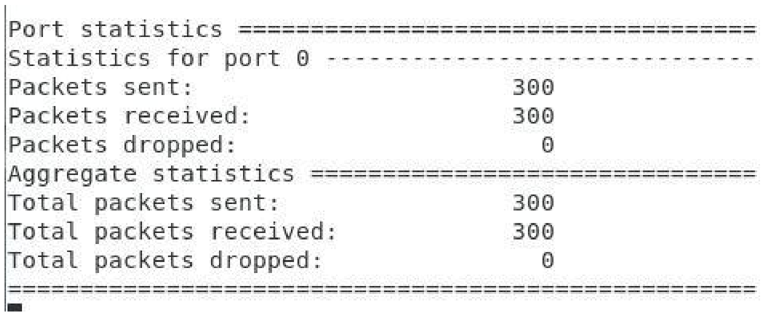


```
[root@localhost l2fwd]# yum install pciutils
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.sjtu.edu.cn
 * extras: ftp.sjtu.edu.cn
 * updates: ftp.sjtu.edu.cn
base                                                    | 3.6 kB     00:00
extras                                                  | 2.9 kB     00:00
updates                                                 | 2.9 kB     00:00
updates/7/x86_64/primary_db                               | 3.7 MB   00:00
Package pciutils-3.5.1-3.el7.x86_64 already installed and latest version
Nothing to do
```



```
[root@localhost dpdk-19.08.2]# export RTE_TARGET='x86_64-native-linuxapp-gcc'
[root@localhost dpdk-19.08.2]# export RTE_SDK=/dpdk-19.08.2
[root@localhost dpdk-19.08.2]# cd $RTE_SDK/examples/l2fwd/
[root@localhost l2fwd]# make
[root@localhost l2fwd]# ./build/l2fwd -c 0x1 -n 4 -- -p 0x1 -q 1
EAL: Detected 1 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable cloc
k cycles !
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
MAC updating enabled
Notice: odd number of ports in portmask.
Lcore 0: RX port 0
Initializing port 0... done:
Port 0, MAC address: 52:54:00:B5:79:AB


Checking link statusdone
Port0 Link Up. Speed 10000 Mbps - full-duplex
```

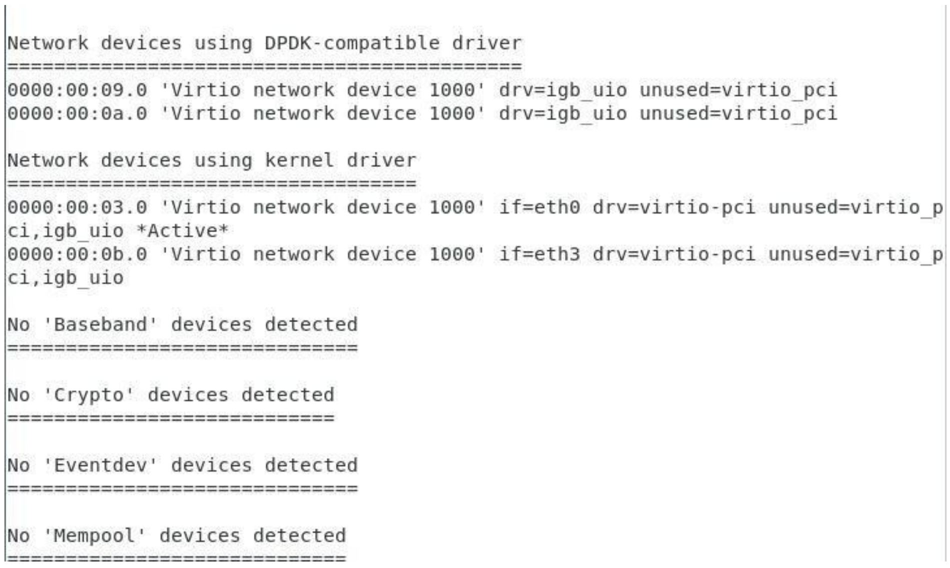
运行结果如下，l2fwd 显示的 VM2 收到 VM 的包数量，转发给 VM 的包数量，丢弃的包数量。



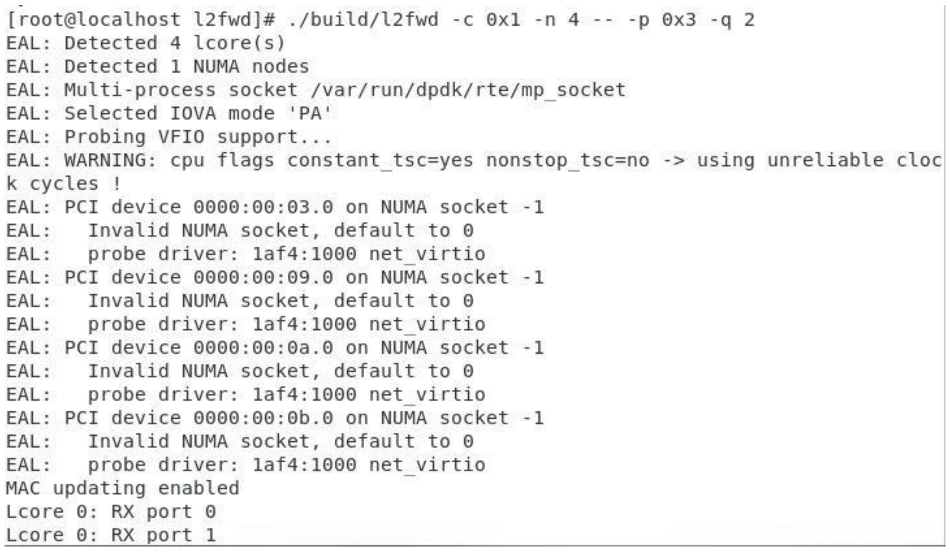
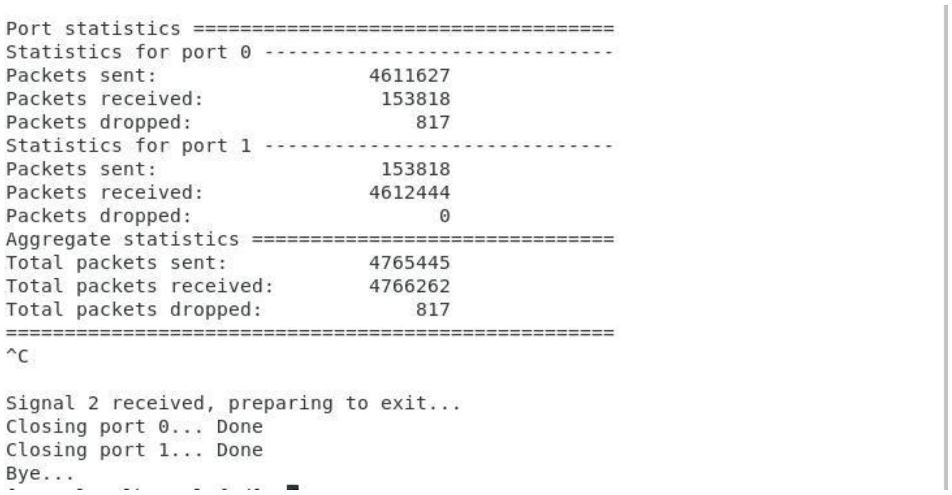
```
Port statistics ====================================
Statistics for port 0 ----------------------------
Packets sent:                     300
Packets received:                 300
Packets dropped:                    0
Aggregate statistics ===============================
Total packets sent:               300
Total packets received:           300
Total packets dropped:              0
====================================================
```

由于绑定的网卡需要断开与原本内核驱动的连接，向 VM 和 VM2 中额外添加三个网卡，同时记录下其 MAC 地址，以供后续 pktgen 使用。同时为了防止后续 pktgen 出现 core dumped，将每台虚拟机的内存调整至 2G，分配两个 vCPU。同时为了防止设置过多 vCPU 影响到性能，将 VMware 宿主虚拟机的核的数量调整到 4，调高内存至 8GB。

将 VM 和 VM2 每台虚拟机 DPDK 都各绑定两个网卡，重新配置环境，加载驱动设置大页内存为 512*2MB，注意大页内存必须要足够大，防止 pktgen 无法启动。

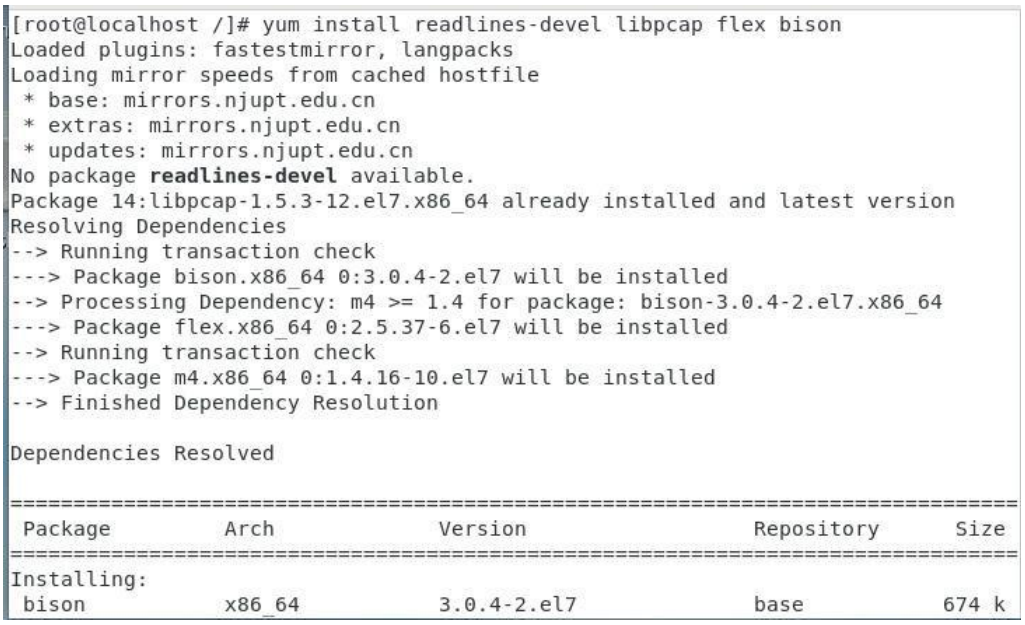

```
Network devices using DPDK-compatible driver
============================================
0000:00:09.0 'Virtio network device 1000' drv=igb_uio unused=virtio_pci
0000:00:0a.0 'Virtio network device 1000' drv=igb_uio unused=virtio_pci

Network devices using kernel driver
===================================
0000:00:03.0 'Virtio network device 1000' if=eth0 drv=virtio-pci unused=virtio_p
ci,igb_uio *Active*
0000:00:0b.0 'Virtio network device 1000' if=eth3 drv=virtio-pci unused=virtio_p
ci,igb_uio

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

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

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

No 'Mempool' devices detected
=============================
```
使用两个端口测试收发性能，并且每一个逻辑内核都有两个收发队列，运行后 10s 显示出第一份结果，可以观察到 l2fwd 收发效率提升明显。

./build/l2fwd -c 0x1 -n 4 -- -p 0x3 -q 2

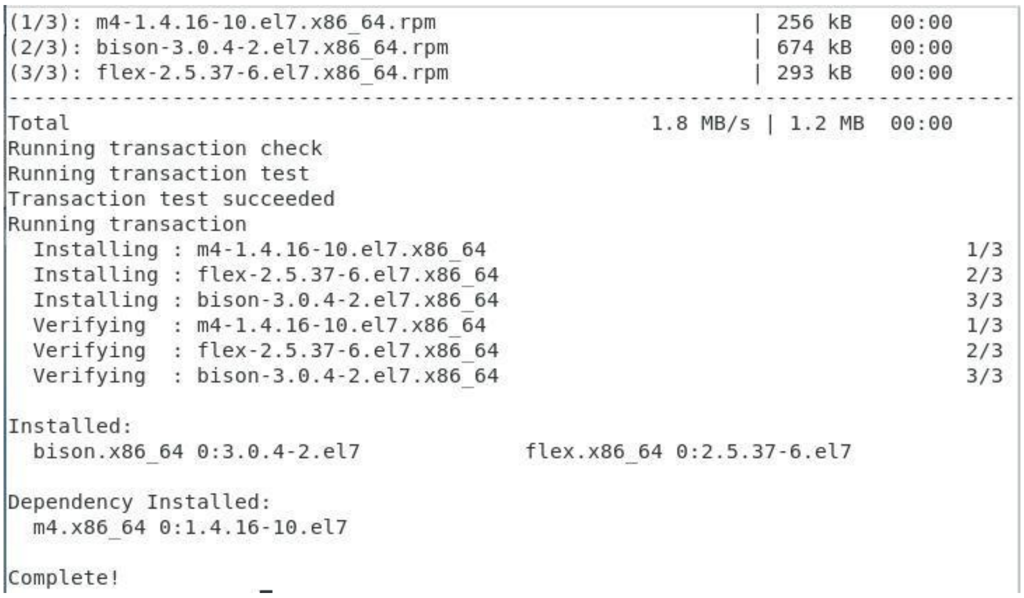

```
[root@localhost l2fwd]# ./build/l2fwd -c 0x1 -n 4 -- -p 0x3 -q 2
EAL: Detected 4 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable cloc
k cycles !
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:09.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0a.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0b.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
MAC updating enabled
Lcore 0: RX port 0
Lcore 0: RX port 1
```

```
Port statistics ====================================
Statistics for port 0 ------------------------------
Packets sent:                  4611627
Packets received:               153818
Packets dropped:                   817
Statistics for port 1 ------------------------------
Packets sent:                   153818
Packets received:              4612444
Packets dropped:                     0
Aggregate statistics ===============================
Total packets sent:            4765445
Total packets received:        4766262
Total packets dropped:             817
====================================================
^C

Signal 2 received, preparing to exit...
Closing port 0... Done
Closing port 1... Done
Bye...
```
四. VM 上编译并运行pktgen-dpdk

首先下载必需的安装依赖包

yum install readlines-devel libpcap flex bison



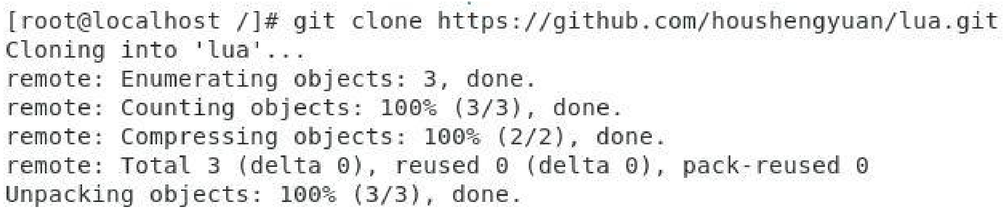
```
[root@localhost /]# yum install readlines-devel libpcap flex bison
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.njupt.edu.cn
 * extras: mirrors.njupt.edu.cn
 * updates: mirrors.njupt.edu.cn
No package readlines-devel available.
Package 14:libpcap-1.5.3-12.el7.x86_64 already installed and latest version
Resolving Dependencies
--> Running transaction check
---> Package bison.x86_64 0:3.0.4-2.el7 will be installed
--> Processing Dependency: m4 >= 1.4 for package: bison-3.0.4-2.el7.x86_64
---> Package flex.x86_64 0:2.5.37-6.el7 will be installed
--> Running transaction check
---> Package m4.x86_64 0:1.4.16-10.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package         Arch              Version                   Repository     Size
================================================================================
Installing:
 bison           x86_64            3.0.4-2.el7               base          674 k
```



```
(1/3): m4-1.4.16-10.el7.x86_64.rpm                           | 256 kB   00:00
(2/3): bison-3.0.4-2.el7.x86_64.rpm                          | 674 kB   00:00
(3/3): flex-2.5.37-6.el7.x86_64.rpm                          | 293 kB   00:00
--------------------------------------------------------------------------------
Total                                                1.8 MB/s | 1.2 MB  00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : m4-1.4.16-10.el7.x86_64                                      1/3
  Installing : flex-2.5.37-6.el7.x86_64                                     2/3
  Installing : bison-3.0.4-2.el7.x86_64                                     3/3
  Verifying  : m4-1.4.16-10.el7.x86_64                                      1/3
  Verifying  : flex-2.5.37-6.el7.x86_64                                     2/3
  Verifying  : bison-3.0.4-2.el7.x86_64                                     3/3

Installed:
  bison.x86_64 0:3.0.4-2.el7             flex.x86_64 0:2.5.37-6.el7

Dependency Installed:
  m4.x86_64 0:1.4.16-10.el7

Complete!
```

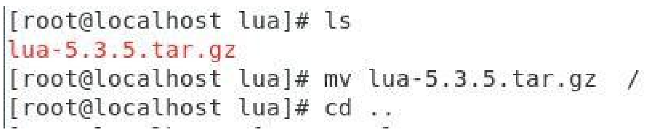
下载 Lua 源代码，Lua 是一种轻量小巧的脚本语言，用标准 C 语言编写并以源代码形式开放， 其设计目的是为了嵌入应用程序中，从而为应用程序提供灵活的扩展和定制功能。因此此处下载编译安装 Lua 应该是为了便于在 pktgen-dpdk 测试阶段进行命令行操作。

git clone https://github.com/houshengyuan/lua.git



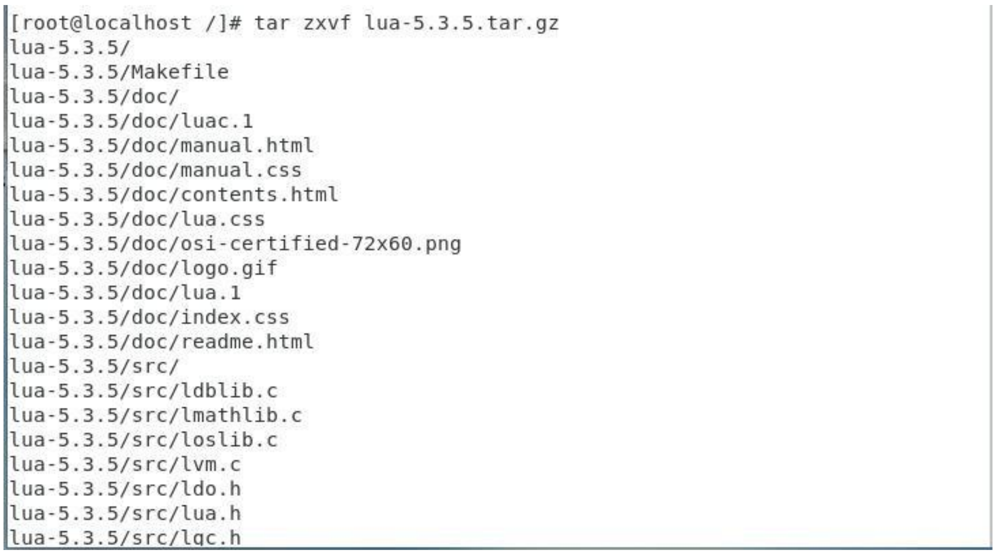
```
[root@localhost /]# git clone https://github.com/houshengyuan/lua.git
Cloning into 'lua'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

解压缩并进入目录



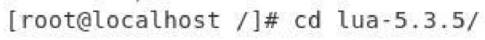
```
[root@localhost lua]# ls
lua-5.3.5.tar.gz
[root@localhost lua]# mv lua-5.3.5.tar.gz  /
[root@localhost lua]# cd ..
```

tar zxvf lua-5.3.5.tar.gz



```
[root@localhost /]# tar zxvf lua-5.3.5.tar.gz
lua-5.3.5/
lua-5.3.5/Makefile
lua-5.3.5/doc/
lua-5.3.5/doc/luac.1
lua-5.3.5/doc/manual.html
lua-5.3.5/doc/manual.css
lua-5.3.5/doc/contents.html
lua-5.3.5/doc/lua.css
lua-5.3.5/doc/osi-certified-72x60.png
lua-5.3.5/doc/logo.gif
lua-5.3.5/doc/lua.1
lua-5.3.5/doc/index.css
lua-5.3.5/doc/readme.html
lua-5.3.5/src/
lua-5.3.5/src/ldblib.c
lua-5.3.5/src/lmathlib.c
lua-5.3.5/src/loslib.c
lua-5.3.5/src/lvm.c
lua-5.3.5/src/ldo.h
lua-5.3.5/src/lua.h
lua-5.3.5/src/lgc.h
```

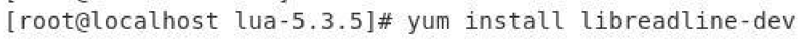
cd lua-5.3.5/



```
[root@localhost /]# cd lua-5.3.5/
```

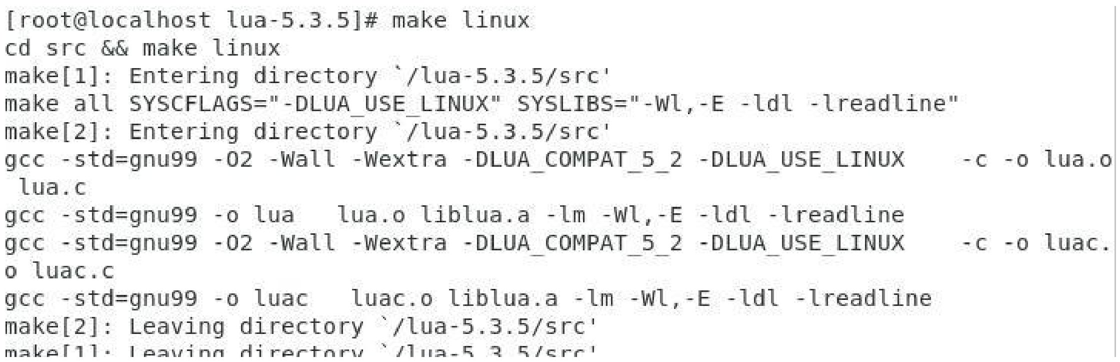
编译源代码(make linux)，根据编译结果补充安装相应的包

yum install libreadline-dev



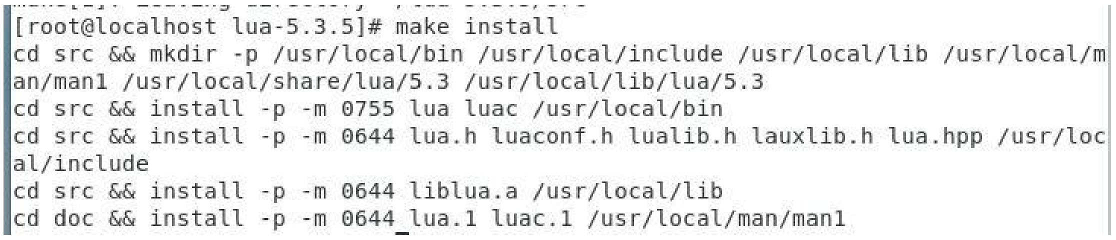
```
[root@localhost lua-5.3.5]# yum install libreadline-dev
```

make linux



```
[root@localhost lua-5.3.5]# make linux
cd src && make linux
make[1]: Entering directory `/lua-5.3.5/src'
make all SYSCFLAGS="-DLUA_USE_LINUX" SYSLIBS="-Wl,-E -ldl -lreadline"
make[2]: Entering directory `/lua-5.3.5/src'
gcc -std=gnu99 -O2 -Wall -Wextra -DLUA_COMPAT_5_2 -DLUA_USE_LINUX    -c -o lua.o
 lua.c
gcc -std=gnu99 -o lua   lua.o liblua.a -lm -Wl,-E -ldl -lreadline
gcc -std=gnu99 -O2 -Wall -Wextra -DLUA_COMPAT_5_2 -DLUA_USE_LINUX    -c -o luac.
o luac.c
gcc -std=gnu99 -o luac   luac.o liblua.a -lm -Wl,-E -ldl -lreadline
make[2]: Leaving directory `/lua-5.3.5/src'
make[1]: Leaving directory `/lua-5.3.5/src'
```

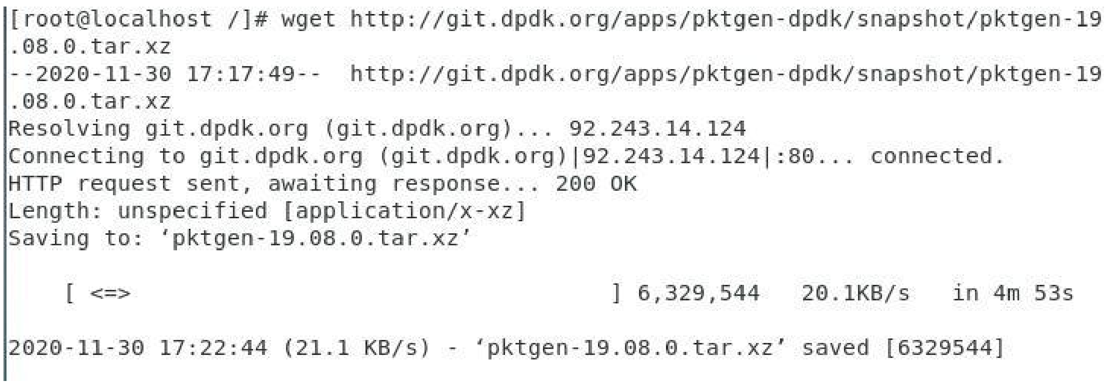
安装 make install



```
[root@localhost lua-5.3.5]# make install
cd src && mkdir -p /usr/local/bin /usr/local/include /usr/local/lib /usr/local/m
an/man1 /usr/local/share/lua/5.3 /usr/local/lib/lua/5.3
cd src && install -p -m 0755 lua luac /usr/local/bin
cd src && install -p -m 0644 lua.h luaconf.h lualib.h lauxlib.h lua.hpp /usr/loc
al/include
cd src && install -p -m 0644 liblua.a /usr/local/lib
cd doc && install -p -m 0644 lua.1 luac.1 /usr/local/man/man1
```

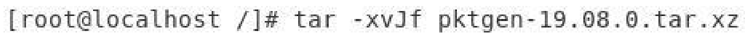
下载 pktgen-dpdk 源代码，注意此处要和 DPDK 源代码的版本接近，不然编译阶段会报出很多错误。

wget http://git.dpdk.org/apps/pktgen-dpdk/snapshot/pktgen-19.08.0.tar.xz


```
[root@localhost /]# wget http://git.dpdk.org/apps/pktgen-dpdk/snapshot/pktgen-19
.08.0.tar.xz
--2020-11-30 17:17:49--  http://git.dpdk.org/apps/pktgen-dpdk/snapshot/pktgen-19
.08.0.tar.xz
Resolving git.dpdk.org (git.dpdk.org)... 92.243.14.124
Connecting to git.dpdk.org (git.dpdk.org)|92.243.14.124|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-xz]
Saving to: ‘pktgen-19.08.0.tar.xz’

    [ <=>                                  ] 6,329,544   20.1KB/s   in 4m 53s

2020-11-30 17:22:44 (21.1 KB/s) - ‘pktgen-19.08.0.tar.xz’ saved [6329544]
```

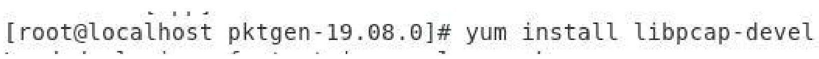
解压源代码

tar -xvJf pktgen-19.08.0.tar.xz

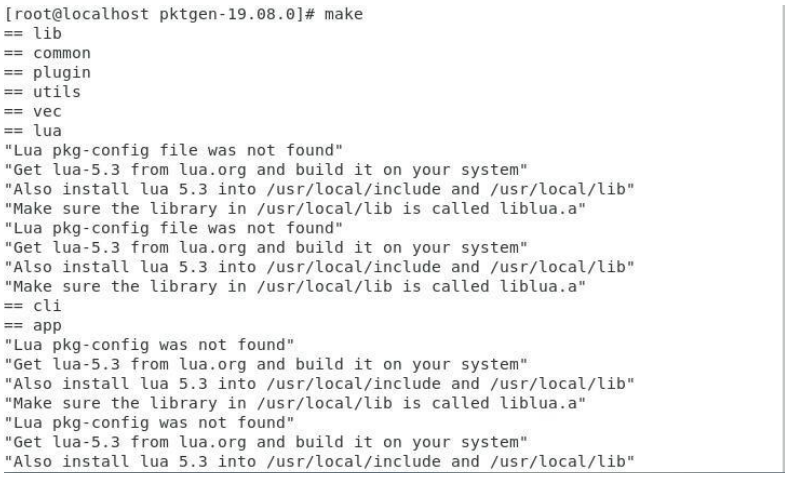

```
[root@localhost /]# tar -xvJf pktgen-19.08.0.tar.xz
```

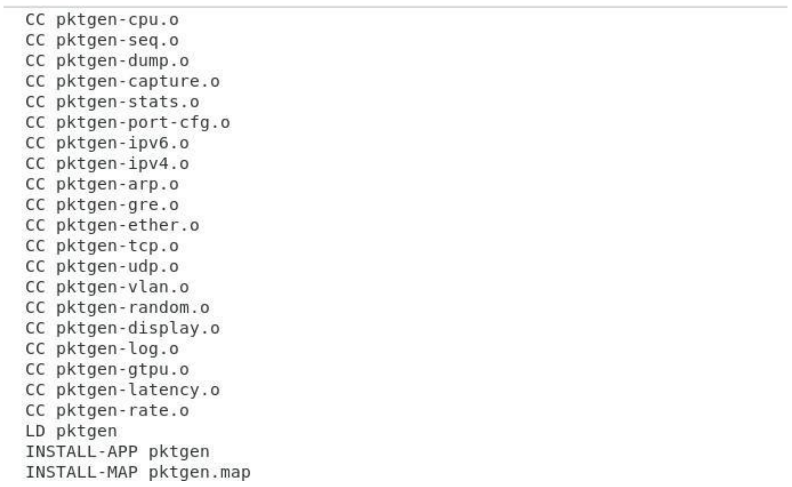
安装相关依赖包

yum install libpcap-devel


```
[root@localhost pktgen-19.08.0]# yum install libpcap-devel
```

编译(make)

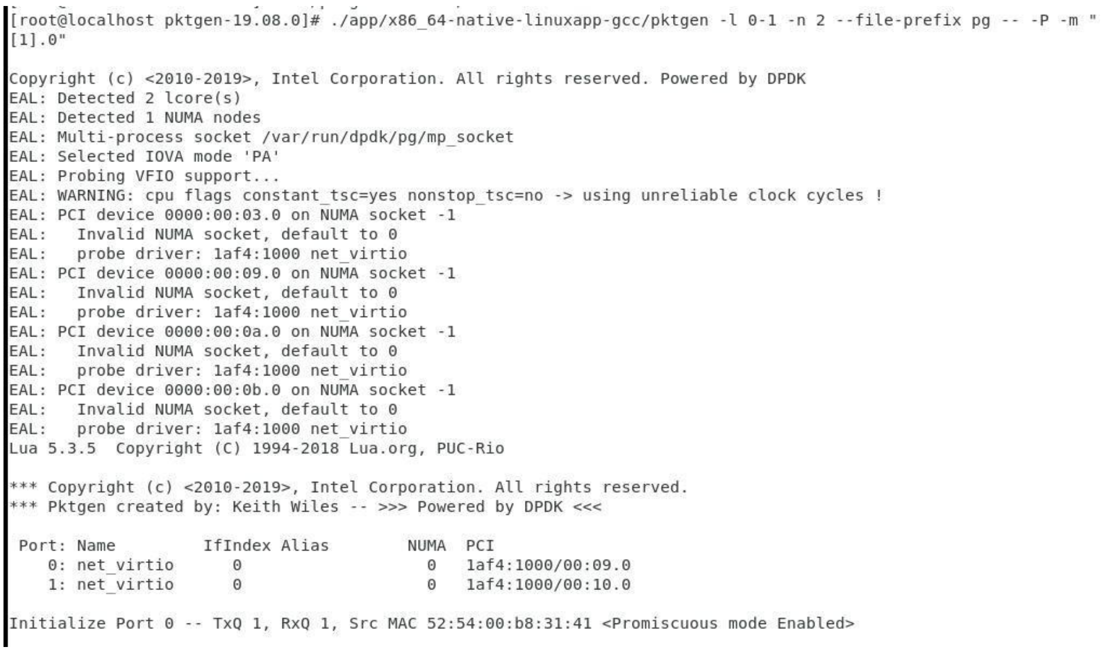

```
[root@localhost pktgen-19.08.0]# make
== lib
== common
== plugin
== utils
== vec
== lua
"Lua pkg-config file was not found"
"Get lua-5.3 from lua.org and build it on your system"
"Also install lua 5.3 into /usr/local/include and /usr/local/lib"
"Make sure the library in /usr/local/lib is called liblua.a"
"Lua pkg-config file was not found"
"Get lua-5.3 from lua.org and build it on your system"
"Also install lua 5.3 into /usr/local/include and /usr/local/lib"
"Make sure the library in /usr/local/lib is called liblua.a"
== cli
== app
"Lua pkg-config was not found"
"Get lua-5.3 from lua.org and build it on your system"
"Also install lua 5.3 into /usr/local/include and /usr/local/lib"
"Make sure the library in /usr/local/lib is called liblua.a"
"Lua pkg-config was not found"
"Get lua-5.3 from lua.org and build it on your system"
"Also install lua 5.3 into /usr/local/include and /usr/local/lib"
```


```
  CC pktgen-cpu.o
  CC pktgen-seq.o
  CC pktgen-dump.o
  CC pktgen-capture.o
  CC pktgen-stats.o
  CC pktgen-port-cfg.o
  CC pktgen-ipv6.o
  CC pktgen-ipv4.o
  CC pktgen-arp.o
  CC pktgen-gre.o
  CC pktgen-ether.o
  CC pktgen-tcp.o
  CC pktgen-udp.o
  CC pktgen-vlan.o
  CC pktgen-random.o
  CC pktgen-display.o
  CC pktgen-log.o
  CC pktgen-gtpu.o
  CC pktgen-latency.o
  CC pktgen-rate.o
  LD pktgen
  INSTALL-APP pktgen
  INSTALL-MAP pktgen.map
```

VM 一侧的状况

./app/x86_64-native-linuxapp-gcc/pktgen -l 0-1 -n 2 --file-prefix pg -- -P -m “[1].0”
EAL 命令行选项
-l 0-1 指定使用核 0 和核 1(pktgen 要求必须要使用两种核)
-n 2 意味着每个处理器使用两个内存通道
--file-prefix 为 DPDK 进程设置一个不同的共享文件前缀，命名为 pg，此处只有一个文件前缀名，因此仅仅设置了一个独立的 DPDK 进程。
pktgen 命令行选项
-P 开始混杂模式
-m DPDK 网卡和逻辑核之间的矩阵映射，[1].0 表示逻辑核 1 处理端口 0 的收发包(rx/tx)。

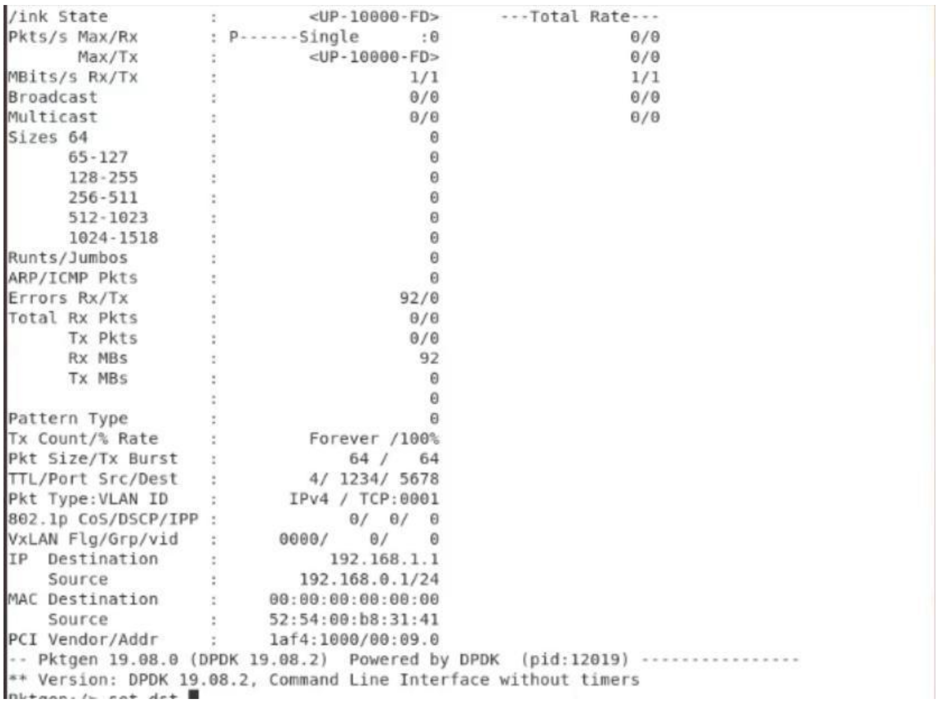
Pktgen 的命令行含义
start portnum　端口 portnum 开始发包，默认情况下一直进行收包工作
stop portnum　端口 portnum 停止发包
quit 退出 pktgen 命令行界面
set portnum src|dst mac 设定端口 portnum 发送的源/目的 MAC 地址

运行 pktgen 命令行(VM 端口 0 开始往 VM2 的端口 1 发包)
>set 0 dst mac　52:54:00:ff:df:2a
>start 0

```
[root@localhost pktgen-19.08.0]# ./app/x86_64-native-linuxapp-gcc/pktgen -l 0-1 -n 2 --file-prefix pg -- -P -m "
[1].0"

Copyright (c) <2010-2019>, Intel Corporation. All rights reserved. Powered by DPDK
EAL: Detected 2 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/pg/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable clock cycles !
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:09.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0a.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0b.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
Lua 5.3.5  Copyright (C) 1994-2018 Lua.org, PUC-Rio

*** Copyright (c) <2010-2019>, Intel Corporation. All rights reserved.
*** Pktgen created by: Keith Wiles -- >>> Powered by DPDK <<<

 Port: Name         IfIndex Alias        NUMA  PCI
    0: net_virtio      0                   0   1af4:1000/00:09.0
    1: net_virtio      0                   0   1af4:1000/00:10.0

Initialize Port 0 -- TxQ 1, RxQ 1, Src MAC 52:54:00:b8:31:41 <Promiscuous mode Enabled>
```
以上运行结果说明 pktgen 安装运行成功

五. 衡量DPDK 中 L2 forwarding 的性能

我们通过 pktgen 中 Rx,Tx 性能指标和 l2fwd 一端的发包收包丢包量来衡量 L2 forwarding 的性能。
（一）VM2 开启 l2fwd 之前 pktgen 的性能指标参数
下图为启动pktgen 之前状况，由于 pktgen 未被启动，且不知道目标发送地址(MAC 地址为 00:00:00:00:00:00)，因此仅仅向外部目标端口周期性发送 1 个数据包，且

由于不知道目标地址，发送均为失败。

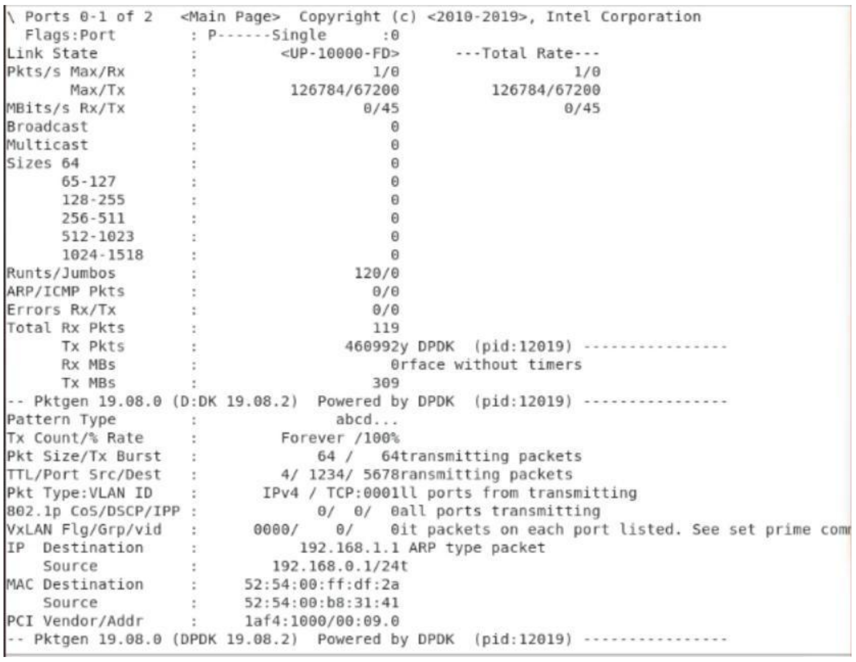


```
/ink State          :         <UP-10000-FD>       ---Total Rate---
Pkts/s Max/Rx       : P------Single      :0                   0/0
       Max/Tx       :         <UP-10000-FD>                   0/0
MBits/s Rx/Tx       :                   1/1                   1/1
Broadcast           :                   0/0                   0/0
Multicast           :                   0/0                   0/0
Sizes 64            :                     0
      65-127        :                     0
      128-255       :                     0
      256-511       :                     0
      512-1023      :                     0
      1024-1518     :                     0
Runts/Jumbos        :                     0
ARP/ICMP Pkts       :                     0
Errors Rx/Tx        :                  92/0
Total Rx Pkts       :                   0/0
      Tx Pkts       :                   0/0
      Rx MBs        :                    92
      Tx MBs        :                     0
                    :                     0
Pattern Type        :                     0
Tx Count/% Rate     :         Forever /100%
Pkt Size/Tx Burst   :             64 /   64
TTL/Port Src/Dest   :        4/ 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:b8:31:41
PCI Vendor/Addr     :     1af4:1000/00:09.0
-- Pktgen 19.08.0 (DPDK 19.08.2)  Powered by DPDK  (pid:12019) ----------------
** Version: DPDK 19.08.2, Command Line Interface without timers
Pktgen:/> set dst
```

（二）运行 pktgen 命令行(VM 端口 0 开始往 VM2 的端口 1 发包)

```
>set 0 dst mac  52:54:00:ff:df:2a
>start 0
```

为 pktgen 端口 0 指定发送目标 MAC 地址后，开启对应发送端口，这时便可以看到 pktgen 数据已经可以正常发送(Error 为 0)，数据和包的发送速率迅速增长(数据发送速率由 0 增长到 45Mb/s，包的发送速率由 0 增长到 67200Pkts/s)，但由于目标地址的 l2fwd 未运行，不能转发回收到的数据包，因此接受速率为 0，总量依然没有变化。

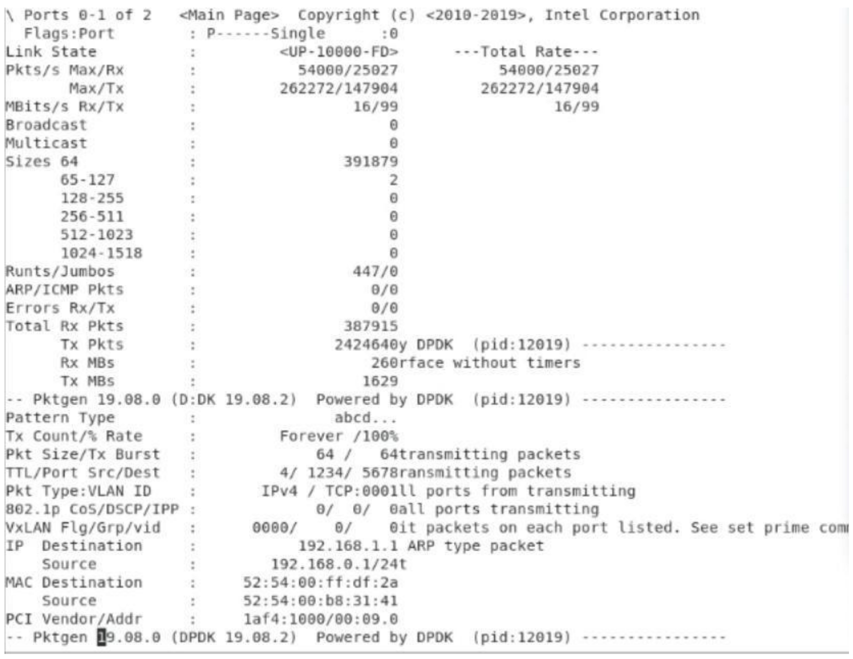


```
\ Ports 0-1 of 2   <Main Page>  Copyright (c) <2010-2019>, Intel Corporation
  Flags:Port        : P------Single      :0
Link State          :         <UP-10000-FD>      ---Total Rate---
Pkts/s Max/Rx       :                   1/0                   1/0
       Max/Tx       :          126784/67200          126784/67200
MBits/s Rx/Tx       :                  0/45                  0/45
Broadcast           :                     0
Multicast           :                     0
Sizes 64            :                     0
      65-127        :                     0
      128-255       :                     0
      256-511       :                     0
      512-1023      :                     0
      1024-1518     :                     0
Runts/Jumbos        :                 120/0
ARP/ICMP Pkts       :                   0/0
Errors Rx/Tx        :                   0/0
Total Rx Pkts       :                   119
      Tx Pkts       :                460992y DPDK  (pid:12019) ----------------
      Rx MBs        :                     0rface without timers
      Tx MBs        :                   309
-- Pktgen 19.08.0 (D:DK 19.08.2)  Powered by DPDK  (pid:12019) ----------------
Pattern Type        :               abcd...
Tx Count/% Rate     :         Forever /100%
Pkt Size/Tx Burst   :             64 /   64transmitting packets
TTL/Port Src/Dest   :        4/ 1234/ 5678ransmitting packets
Pkt Type:VLAN ID    :       IPv4 / TCP:0001ll ports from transmitting
802.1p CoS/DSCP/IPP :             0/  0/  0all ports transmitting
VxLAN Flg/Grp/vid   :      0000/    0/    0it packets on each port listed. See set prime com
IP  Destination     :           192.168.1.1 ARP type packet
    Source          :        192.168.0.1/24t
MAC Destination     :     52:54:00:ff:df:2a
    Source          :     52:54:00:b8:31:41
PCI Vendor/Addr     :     1af4:1000/00:09.0
-- Pktgen 19.08.0 (DPDK 19.08.2)  Powered by DPDK  (pid:12019) ----------------
```

（三）打开 l2fwd 后 VM 一侧的状况截图

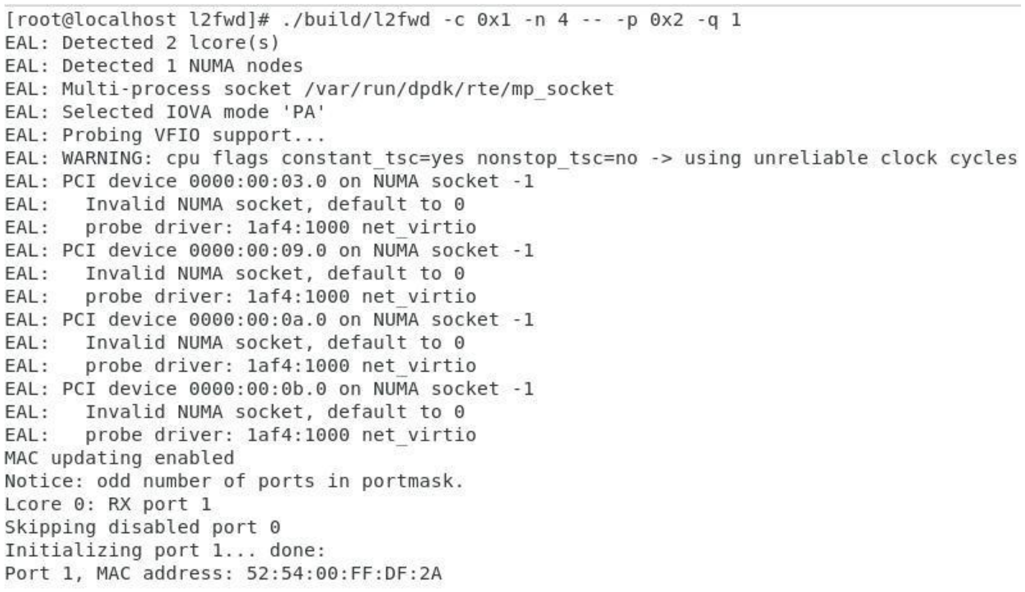
打开 l2fwd 后，l2fwd 一端能够将收到的数据包进行回转操作，此时 pktgen 一端

就可以探测到大量的接收数据包，接收数据速率从 0 变为 16Mb/s，接收数据包的速率从 0 变为 25027Pkts/s。

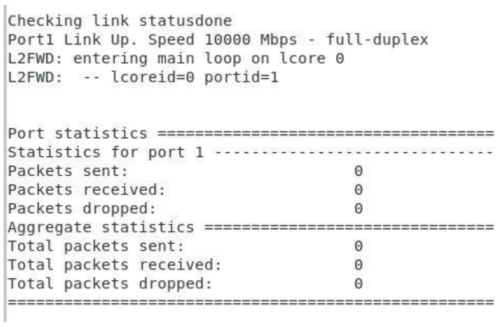


```
\ Ports 0-1 of 2   <Main Page>  Copyright (c) <2010-2019>, Intel Corporation
  Flags:Port        : P------Single      :0
Link State          :         <UP-10000-FD>      ---Total Rate---
Pkts/s Max/Rx       :           54000/25027         54000/25027
       Max/Tx       :         262272/147904       262272/147904
MBits/s Rx/Tx       :                 16/99               16/99
Broadcast           :                     0
Multicast           :                     0
Sizes 64            :                391879
      65-127        :                     2
      128-255       :                     0
      256-511       :                     0
      512-1023      :                     0
      1024-1518     :                     0
Runts/Jumbos        :                 447/0
ARP/ICMP Pkts       :                   0/0
Errors Rx/Tx        :                   0/0
Total Rx Pkts       :                387915
      Tx Pkts       :               2424640y DPDK  (pid:12019) ----------------
      Rx MBs        :                   260rface without timers
      Tx MBs        :                  1629
-- Pktgen 19.08.0 (D:DK 19.08.2)  Powered by DPDK  (pid:12019) ----------------
Pattern Type        :               abcd...
Tx Count/% Rate     :          Forever /100%
Pkt Size/Tx Burst   :              64 /   64transmitting packets
TTL/Port Src/Dest   :          4/ 1234/ 5678ransmitting packets
Pkt Type:VLAN ID    :        IPv4 / TCP:0001ll ports from transmitting
802.1p CoS/DSCP/IPP :              0/  0/  0all ports transmitting
VxLAN Flg/Grp/vid   :       0000/    0/    0it packets on each port listed. See set prime com
IP  Destination     :           192.168.1.1 ARP type packet
    Source          :         192.168.0.1/24t
MAC Destination     :      52:54:00:ff:df:2a
    Source          :      52:54:00:b8:31:41
PCI Vendor/Addr     :      1af4:1000/00:09.0
-- Pktgen 19.08.0 (DPDK 19.08.2)  Powered by DPDK  (pid:12019) ----------------
```

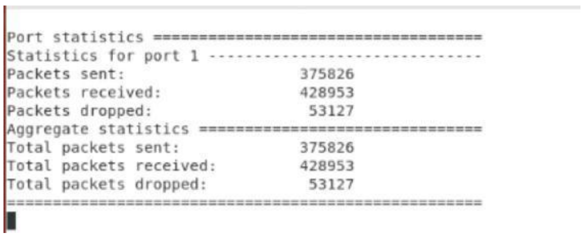
VM2 一侧的状况截图
采用一个逻辑内核，每个内核采用四个内存通道，使用端口 1 接受，1 个收发队列
可以看到端口号 1 的 MAC 地址
packets sent 为 VM2 的转发回去的包
packets received 为 VM2 接受到 VM 发过来的包
packets dropped 为 VM2 丢弃未转发的包



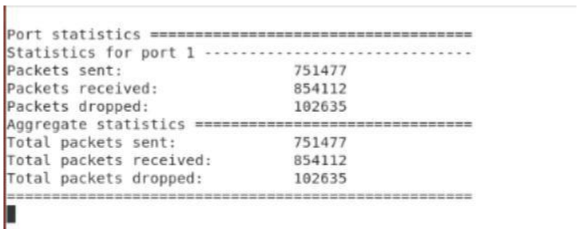
```
[root@localhost l2fwd]# ./build/l2fwd -c 0x1 -n 4 -- -p 0x2 -q 1
EAL: Detected 2 lcore(s)
EAL: Detected 1 NUMA nodes
EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
EAL: Selected IOVA mode 'PA'
EAL: Probing VFIO support...
EAL: WARNING: cpu flags constant_tsc=yes nonstop_tsc=no -> using unreliable clock cycles
EAL: PCI device 0000:00:03.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:09.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0a.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
EAL: PCI device 0000:00:0b.0 on NUMA socket -1
EAL:   Invalid NUMA socket, default to 0
EAL:   probe driver: 1af4:1000 net_virtio
MAC updating enabled
Notice: odd number of ports in portmask.
Lcore 0: RX port 1
Skipping disabled port 0
Initializing port 1... done:
Port 1, MAC address: 52:54:00:FF:DF:2A
```



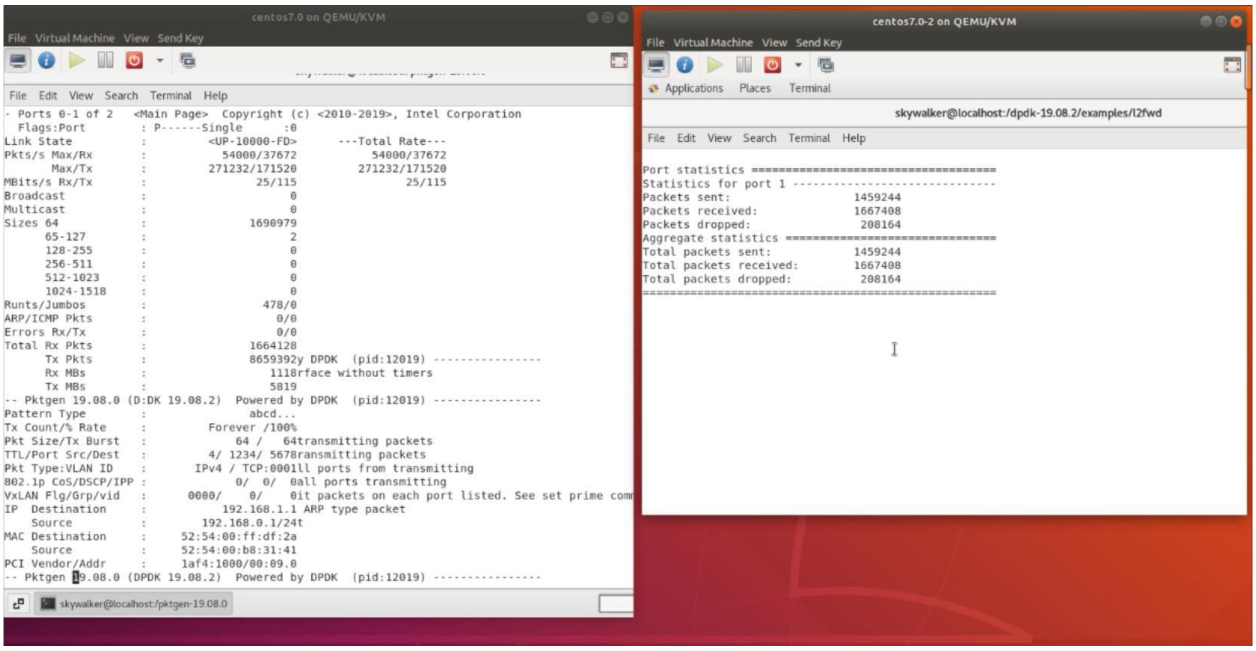
```
Checking link statusdone
Port1 Link Up. Speed 10000 Mbps - full-duplex
L2FWD: entering main loop on lcore 0
L2FWD:  -- lcoreid=0 portid=1


Port statistics ====================================
Statistics for port 1 ------------------------------
Packets sent:                        0
Packets received:                    0
Packets dropped:                     0
Aggregate statistics ===============================
Total packets sent:                  0
Total packets received:              0
Total packets dropped:               0
====================================================
```



```
Port statistics ====================================
Statistics for port 1 ------------------------------
Packets sent:                   375826
Packets received:               428953
Packets dropped:                 53127
Aggregate statistics ===============================
Total packets sent:             375826
Total packets received:         428953
Total packets dropped:           53127
====================================================
```



```
Port statistics ====================================
Statistics for port 1 ------------------------------
Packets sent:                   751477
Packets received:               854112
Packets dropped:                102635
Aggregate statistics ===============================
Total packets sent:             751477
Total packets received:         854112
Total packets dropped:          102635
====================================================
```

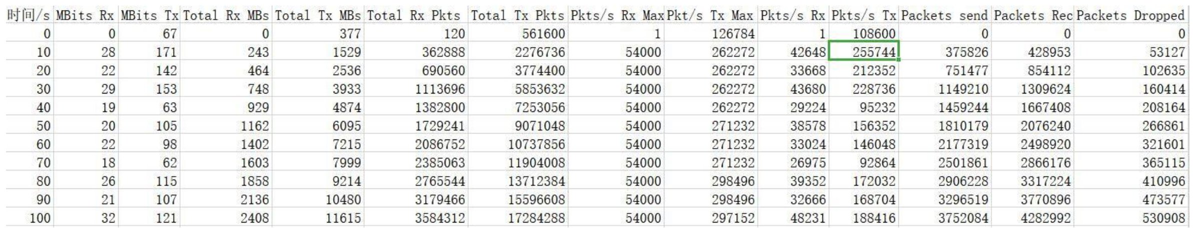


centos7.0 on QEMU/KVM
File Virtual Machine View Send Key
File Edit View Search Terminal Help
skywalker@localhost:/pktgen-19.08.0
centos7.0-2 on QEMU/KVM
Applications Places Terminal
skywalker@localhost:/dpdk-19.08.2/examples/l2fwd

```
- Ports 0-1 of 2   <Main Page>  Copyright (c) <2010-2019>, Intel Corporation
  Flags:Port         : P------Single      :0
Link State           :         <UP-10000-FD>      ---Total Rate---
Pkts/s Max/Rx        :           54000/37672          54000/37672
       Max/Tx        :         271232/171520        271232/171520
MBits/s Rx/Tx        :                25/115               25/115
Broadcast            :                     0
Multicast            :                     0
Sizes 64             :               1690979
      65-127         :                     2
      128-255        :                     0
      256-511        :                     0
      512-1023       :                     0
      1024-1518      :                     0
Runts/Jumbos         :                 478/0
ARP/ICMP Pkts        :                   0/0
Errors Rx/Tx         :                   0/0
Total Rx Pkts        :               1664128
      Tx Pkts        :               8659392y DPDK  (pid:12019) ---------------
      Rx MBs         :                  1118rface without timers
      Tx MBs         :                  5819
-- Pktgen 19.08.0 (D:DK 19.08.2)  Powered by DPDK  (pid:12019) ---------------
Pattern Type         :               abcd...
Tx Count/% Rate      :         Forever /100%
Pkt Size/Tx Burst    :             64 /   64transmitting packets
TTL/Port Src/Dest    :         4/ 1234/ 5678ransmitting packets
Pkt Type:VLAN ID     :       IPv4 / TCP:0001ll ports from transmitting
802.1p CoS/DSCP/IPP  :             0/  0/  0all ports transmitting
VxLAN Flg/Grp/vid    :      0000/    0/    0it packets on each port listed. See set prime comm
IP  Destination      :            192.168.1.1 ARP type packet
    Source           :          192.168.0.1/24t
MAC Destination      :      52:54:00:ff:df:2a
    Source           :      52:54:00:b8:31:41
PCI Vendor/Addr      :      1af4:1000/00:09.0
-- Pktgen 19.08.0 (DPDK 19.08.2)  Powered by DPDK  (pid:12019) ---------------
```

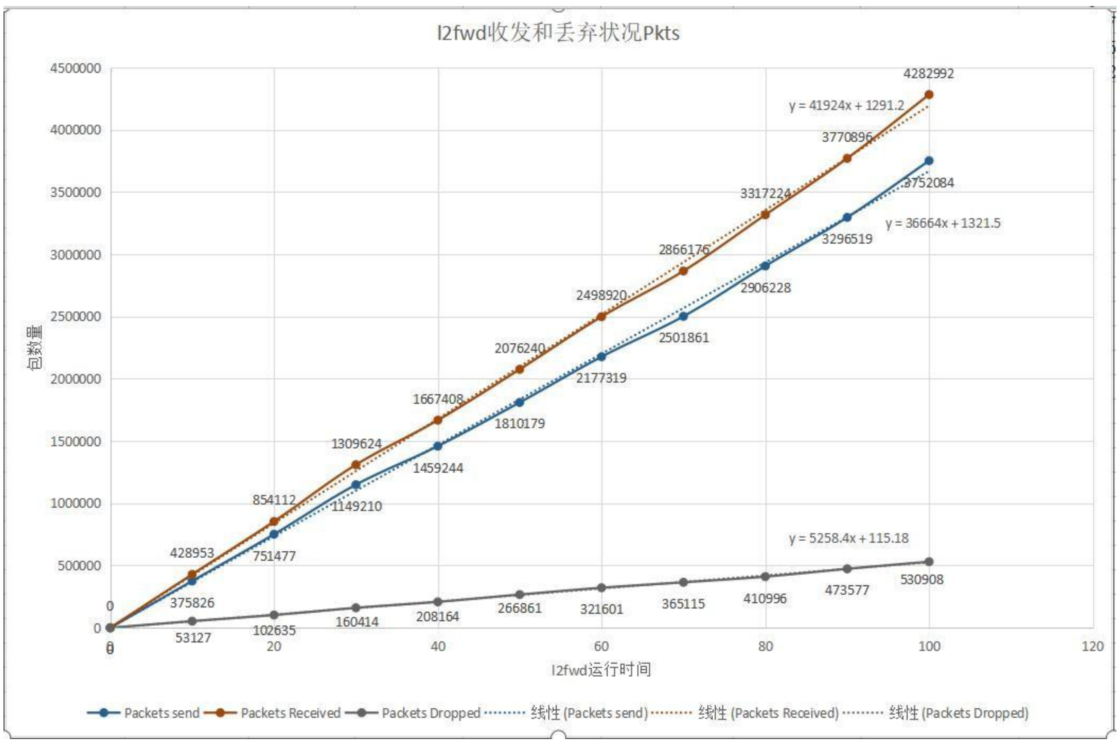
```
Port statistics ====================================
Statistics for port 1 ------------------------------
Packets sent:                  1459244
Packets received:              1667408
Packets dropped:                208164
Aggregate statistics ===============================
Total packets sent:            1459244
Total packets received:        1667408
Total packets dropped:          208164
====================================================
```

由以上实验结果可以看到，当开启 VM2 的 l2fwd 时，在 VM 一侧肉眼可以观察到大量的接收包出现，而发送包一旦端口开启就会大量涌现。
实验统计数据如下，



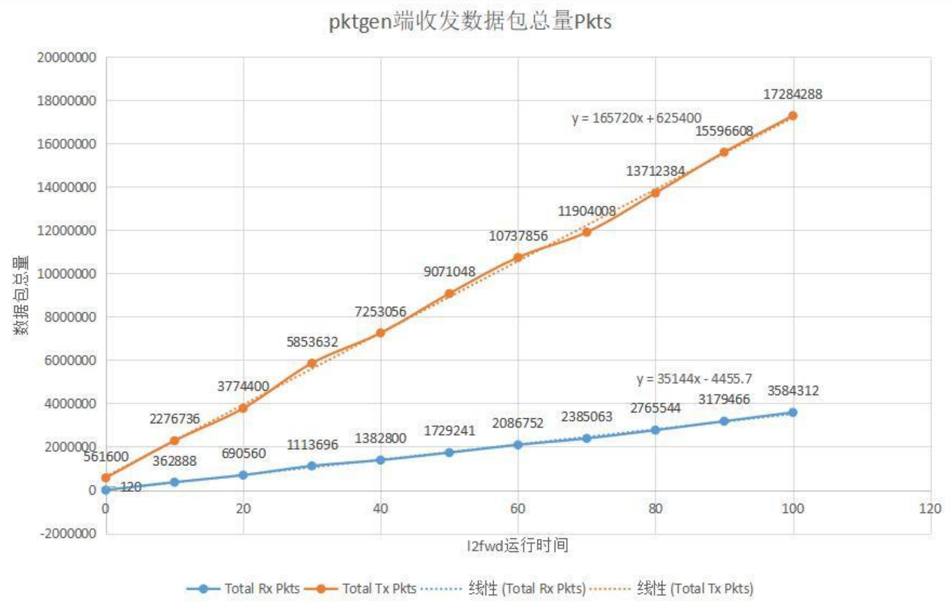
时间/s	MBits Rx	MBits Tx	Total Rx MBs	Total Tx MBs	Total Rx Pkts	Total Tx Pkts	Pkts/s Rx Max	Pkt/s Tx Max	Pkts/s Rx	Pkts/s Tx	Packets send	Packets Rec	Packets Dropped
0	0	67	0	377	120	561600	1	126784	1	108600	0	0	0
10	28	171	243	1529	362888	2276736	54000	262272	42648	255744	375826	428953	53127
20	22	142	464	2536	690560	3774400	54000	262272	33668	212352	751477	854112	102635
30	29	153	748	3933	1113696	5853632	54000	262272	43680	228736	1149210	1309624	160414
40	19	63	929	4874	1382800	7253056	54000	262272	29224	95232	1459244	1667408	208164
50	20	105	1162	6095	1729241	9071048	54000	271232	38578	156352	1810179	2076240	266861
60	22	98	1402	7215	2086752	10737856	54000	271232	33024	146048	2177319	2498920	321601
70	18	62	1603	7999	2385063	11904008	54000	271232	26975	92864	2501861	2866176	365115
80	26	115	1858	9214	2765544	13712384	54000	298496	39352	172032	2906228	3317224	410996
90	21	107	2136	10480	3179466	15596608	54000	298496	32666	168704	3296519	3770896	473577
100	32	121	2408	11615	3584312	17284288	54000	297152	48231	188416	3752084	4282992	530908

(1) l2fwd 数据包收发和丢弃状况



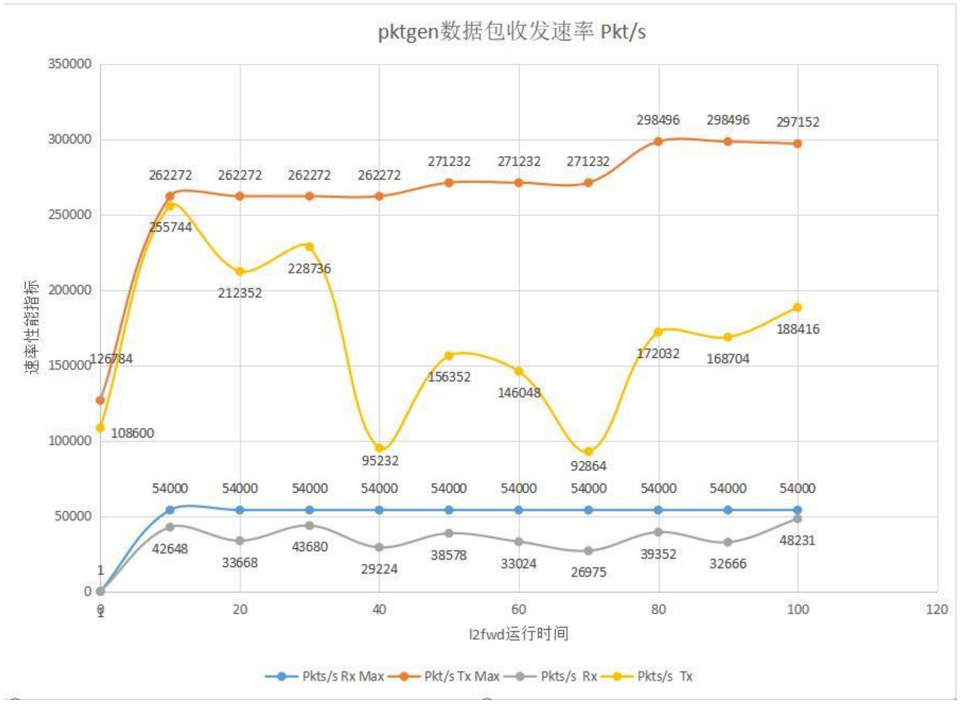
由上图可以发现，l2fwd 每 10s 能够接收大约 42000 个数据包，其中成功转发的大概有 36000 个，被丢弃的大概有 5000 个，数据包丢弃率为 12.2%左右。经过后期了解得到，<u>此处丢弃率之所以不为0是因为网卡配置原因使得l2fwd存在自收发机制，当将网卡从virtio换为e1000，并且设置两个端口，一个接受一个转发后丢包率变为0。由于时间原因，此处不再将virtio换为e1000，采用一个端口进行。</u>

(2)pktgen 端收发数据包总量情况和收发速率



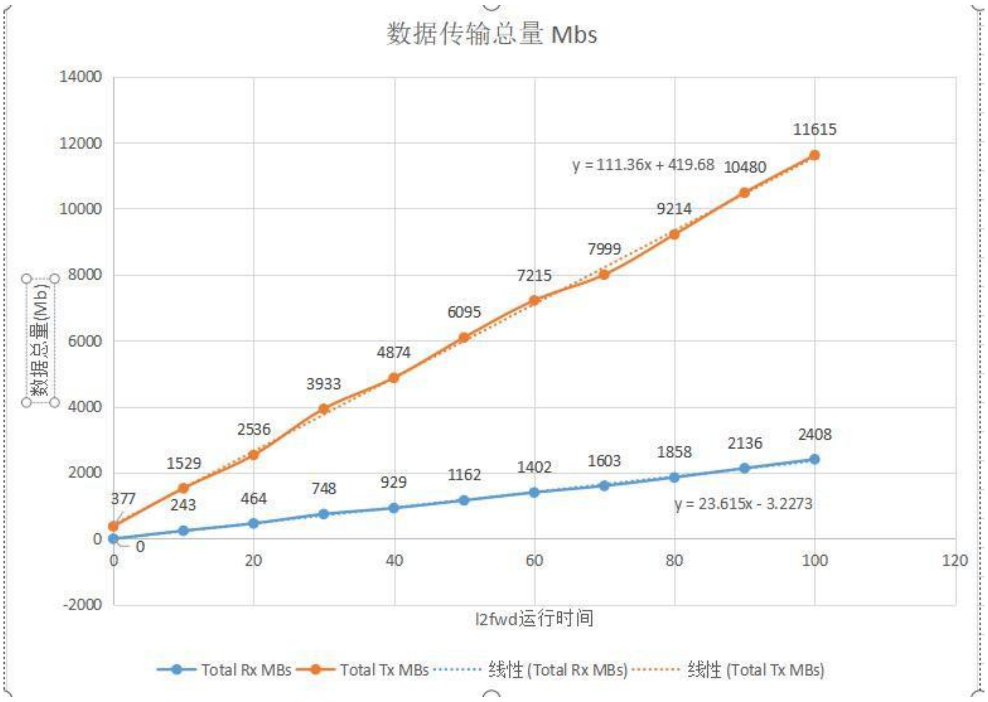
对比 pktgen 一端数据包的收发总量可以发现，pktgen 每 10s 发送出去的数据包数量大概是 165000，是 l2fwd 每 10s 接收数据包数量的 3.5 倍，<u>这可能是因为发送出去的数据包触发了局域网的拥塞控制或出现了传输差错而被丢弃。网卡接收数据包时，需要维护一个缓冲区队列(此处为收发队列)，来缓存可能存在的突发数据，为了实现数据链路层的拥塞控制，收发队列缓冲区往往有一个接收上限，超过上限时后续进入的数据包将被丢弃。由于本次实验只用了一个端口共一个收发队列，因此丢包率较高。</u>观察

Pktgen 对于 l2fwd 转发回的数据包，VM一端每 10s 可以接收近 35500 个数据包，和 l2fwd 数据包的转发速率(36000)相似，说明 pktgen 接收能力至少不差于l2fwd。

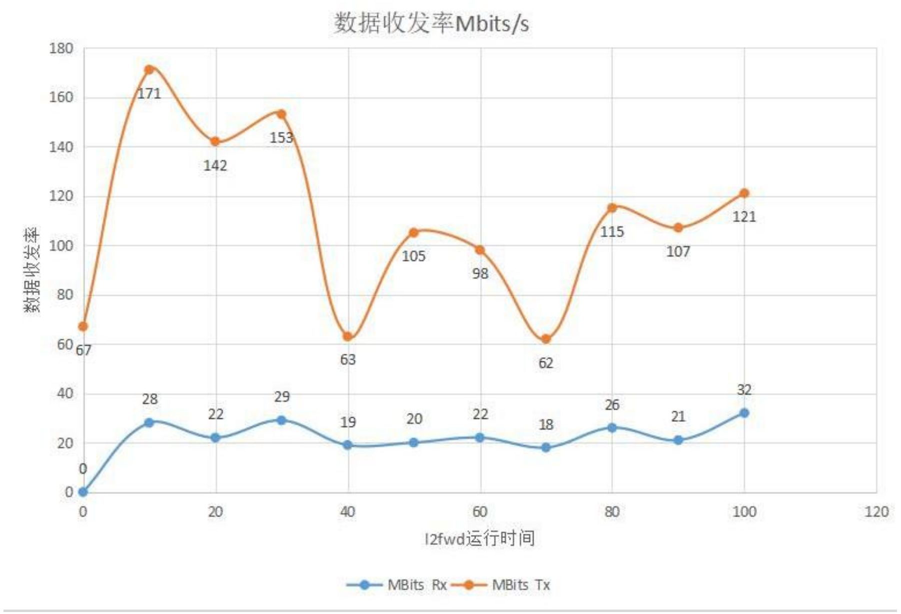


观察实验结果可以发现，数据包的平均接收速率和 l2fwd 的转发速率基本上匹配吻合，都在 35000pkts/s 左右，但 l2fwd 的转发速率存在着瓶颈，这可能是由于其转发端口的数量限制和没有配置多队列导致的。

(3)pktgen 数据收发总量变化和收发速率

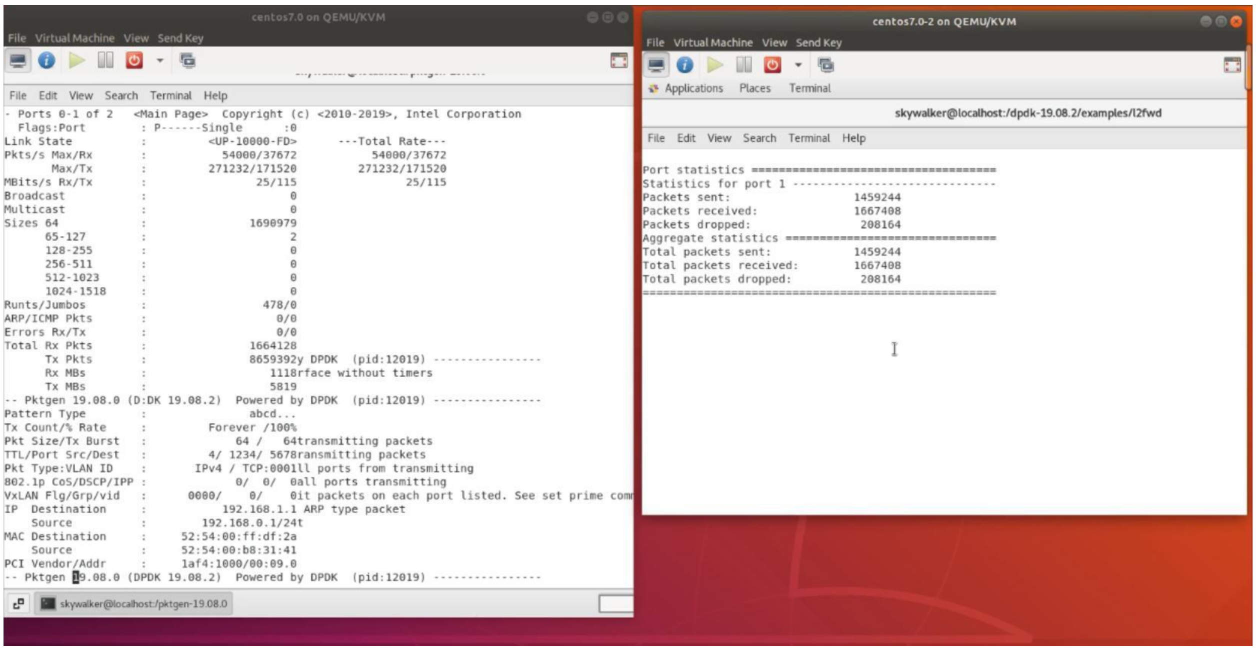


观察实验结果可以发现，pktgen 发送数据的速率约为 111Mb/s，而接受速率只有大约 23.5Mb/s，前者是后者大约 5 倍，而 pktgen 发送率是 l2fwd 数据包接受率的 3.5 倍，再算上 l2fwd 的丢弃率 12%，推算出 l2fwd 转发的数据包数量应该是 pktgen 发送数据包数量的四分之一左右，接近于数据内容大小的比例五分之一。



和之前对数据包的分析一样，虽然 pktgen 发送数据的速率有较大幅度的波动，但由于 l2fwd 收发队列数量不足，转发速率受到限制，加上数据链路层的拥塞控制机制所起到的作用，回转的接收速率出现了一定的瓶颈，在瓶颈下小幅波动。

(4) pktgen 发送数据包大小


centos7.0 on QEMU/KVM
File Virtual Machine View Send Key
File Edit View Search Terminal Help
skywalker@localhost:/pktgen-19.08.0
centos7.0-2 on QEMU/KVM
Applications Places Terminal
skywalker@localhost:/dpdk-19.08.2/examples/l2fwd

```
- Ports 0-1 of 2   <Main Page>  Copyright (c) <2010-2019>, Intel Corporation
  Flags:Port        : P------Single      :0
Link State          :        <UP-10000-FD>      ---Total Rate---
Pkts/s Max/Rx       :          54000/37672        54000/37672
       Max/Tx       :        271232/171520      271232/171520
MBits/s Rx/Tx       :                25/115             25/115
Broadcast           :                     0
Multicast           :                     0
Sizes 64            :               1690979
      65-127        :                     2
      128-255       :                     0
      256-511       :                     0
      512-1023      :                     0
      1024-1518     :                     0
Runts/Jumbos        :                 478/0
ARP/ICMP Pkts       :                   0/0
Errors Rx/Tx        :                   0/0
Total Rx Pkts       :               1664128
      Tx Pkts       :               8659392y DPDK  (pid:12019) ---------------
      Rx MBs        :                  1118rface without timers
      Tx MBs        :                  5819
-- Pktgen 19.08.0 (D:DK 19.08.2)  Powered by DPDK  (pid:12019) ---------------
Pattern Type        :               abcd...
Tx Count/% Rate     :         Forever /100%
Pkt Size/Tx Burst   :              64 /   64transmitting packets
TTL/Port Src/Dest   :         4/ 1234/ 5678ransmitting packets
Pkt Type:VLAN ID    :       IPv4 / TCP:0001ll ports from transmitting
802.1p CoS/DSCP/IPP :              0/  0/  0all ports transmitting
VxLAN Flg/Grp/vid   :      0000/    0/    0it packets on each port listed. See set prime com
IP  Destination     :            192.168.1.1 ARP type packet
    Source          :         192.168.0.1/24t
MAC Destination     :      52:54:00:ff:df:2a
    Source          :      52:54:00:b8:31:41
PCI Vendor/Addr     :      1af4:1000/00:09.0
-- Pktgen 19.08.0 (DPDK 19.08.2)  Powered by DPDK  (pid:12019) ---------------
```

```
Port statistics ====================================
Statistics for port 1 ------------------------------
Packets sent:                    1459244
Packets received:                1667408
Packets dropped:                  208164
Aggregate statistics ===============================
Total packets sent:              1459244
Total packets received:          1667408
Total packets dropped:            208164
====================================================
```

由如上实验中间过程截图可知，pktgen 发出包大小绝大部分为 64，由此可以推测出 pktgen 更倾向发送偏小的数据包，这可能是由于数据包长度小的帧在传输中出错的概率更低，在数据链路层二层回转中，数据包长度越小越容易存活转发效率越高。

经过后续调整包发送大小的实验发现，丢包率会随着包大小增大而降低，这可能是因为大的数据包不容易受到网络波动的影响。

六. 实验感悟

本学期最后一次作业的核心集中在二层转发，在这个过程中我从计算机网络

课程学到的大量知识得到了进一步的巩固和深化，尤其是网络层和数据链路层，例如不同网络设施(网卡，网桥，网关)的作用和配置方法，ICMP 协议中 ping 指令的深入理解以及数据链路层常见功能的实现，还有 DPDK 中的性能指标。得益于实验的趣味性，我开始逐渐对网络的知识内容更感兴趣，呈现在我面前的不再是一句句枯燥的指令，而是背后复杂系统结构的相互联系，作用与影响。一学期过去了，我已经较为熟练的掌握了交大云的使用和云计算的基本原理，以及 qemu 存储虚拟化的常见技术，收获还是蛮大。感谢助教不厌其烦的回答我的问题，以及老师对我们的细心提醒和悉心指导！祝老师和助教新年快乐！