Lab 2 — send and receive packets with D...

Handout: March 9, 2021
Deadline: March 24 23:00, 2021 (No extension)

Assignment overview:

In this assignment, you will learn what DPDK is and what it's used for, you will learn how to set up a DPDK development environment and understand the implementation of DPDK modules and programs. In part 1, you are asked to answer some questions about DPDK. In part 2, you need to write a DPDK application to construct UDP packets and send them to NIC using DPDK.

Prerequisite:

In this lab, DPDK must be installed in your virtual machine. VMware and Ubuntu 20.04 are suggested.

The steps for building DPDK development environment are as follows:

1. Get Source Code

```
$ git clone git://dpdk.org/dpdk # Get DPDK
$ git clone http://dpdk.org/git/dpdk-kmods # Get igb_uio
$ cp -r ./dpdk-kmods/linux/igb_uio ./dpdk/kernel/linux/ # Copy dpdk-kmods/linux/igb_uio/ to dpdk/kernel/linux/
```

- add igb_uio in dpdk/kernel/linux/meson.build subdirs as below:

```
subdirs = ['kni', 'igb_uio']
```

- create a file of meson.build in dpdk/kernel/linux/igb_uio/ as below:

```
# SPDX-License-Identifier: BSD-3-Clause
# Copyright(c) 2017 Intel Corporation
mkfile = custom_target('igb_uio_makefile',
        output: 'Makefile',
        command: ['touch', '@OUTPUT@'])
custom_target('igb_uio',
        input: ['igb_uio.c', 'Kbuild'],
        output: 'igb_uio.ko',
        command: ['make', '-C', kernel_dir + '/build',
```

```
                'M=' + meson.current_build_dir(),
                'src=' + meson.current_source_dir(),
                'EXTRA_CFLAGS=-I' + meson.current_source_dir() +
                        '/../../../lib/librte_eal/include',
                'modules'],
        depends: mkfile,
        install: true,
        install_dir: kernel_dir + '/extra/dpdk',
        build_by_default: get_option('enable_kmods'))
```

2. Configure and build DPDK runtime environment

```
$ sudo apt-get install python3
$ sudo apt-get install python3-pip
$ sudo pip3 install meson
$ sudo pip3 install ninja
$ sudo pip3 install pwntools
$ cd dpdk # 进入DPDK文件夹
$ sudo meson -D examples=all build # 使用选项 -D examples 指定编译所有样例程序
$ cd build
$ sudo ninja install
$ mkdir -p /dev/hugepages # 创建 hugetlbfs 挂载点
$ mountpoint -q /dev/hugepages || mount -t hugetlbfs nodev /dev/hugepages # 挂载 hugetlbfs
$ echo 64 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages # 分配大页
```

3. Run the sample program to make sure you have successfully installed DPDK

```
$ cd examples # 进入 DPDK 中的例子
$ sudo ./dpdk-helloworld # 运行
```

For more information about installing DPDK,refer to the guidance (http://doc.dpdk.org/guides/linux_gsg/).

Part 1: Get familiar with DPDK

Take DPDK official website http://www.dpdk.org/ and Chapter 1 of **”深入浅出DPDK”** as references, answer the questions below:

Q1: What’s the purpose of using hugepage?
Q2: Take examples/helloworld as an example, describe the execution flow of DPDK programs?
Q3: Read the codes of examples/skeleton, describe DPDK APIs related to sending and receiving packets.
Q4: Describe the data structure of ’rte_mbuf’.

Part 2: send packets with DPDK

1.Construct UDP packets with DPDK according to the definition of UDP/IP/Ethernet header. Contents of packets can be filled as you wish.

(Tips: refer to codes of examples/skeleton and construct UDP packets in the array of rte_mbuf, you may use the function rte_pktmbuf_mtod https://doc.dpdk.org/api/rte__mbuf_8h.html#a2a8b10263496c7b580e9d0c7f2a1f073)

```
/* Ethernet 协议头的定义在 lib/librte_net/rte_ether.h 中： */
/**
 * Ethernet header: Contains the destination address, source address
 * and frame type.
 */
struct rte_ether_hdr {
        struct rte_ether_addr d_addr; /**< Destination address. */
        struct rte_ether_addr s_addr; /**< Source address. */
        uint16_t ether_type;      /**< Frame type. */
} __rte_aligned(2);
/* IP 协议头的定义在 lib/librte_net/rte_ip.h 中： */
/**
 * IPv4 Header
 */
struct rte_ipv4_hdr {
        uint8_t  version_ihl;           /**< version and header length */
        uint8_t  type_of_service;       /**< type of service */
        rte_be16_t total_length;        /**< length of packet */
        rte_be16_t packet_id;           /**< packet ID */
        rte_be16_t fragment_offset;     /**< fragmentation offset */
        uint8_t  time_to_live;          /**< time to live */
        uint8_t  next_proto_id;         /**< protocol ID */
        rte_be16_t hdr_checksum;        /**< header checksum */
        rte_be32_t src_addr;            /**< source address */
        rte_be32_t dst_addr;            /**< destination address */
} __rte_packed;
/* UDP 协议头的定义在 lib/librte_net/rte_udp.h 中： */
/**
 * UDP Header
 */
struct rte_udp_hdr {
        rte_be16_t src_port;    /**< UDP source port. */
        rte_be16_t dst_port;    /**< UDP destination port. */
        rte_be16_t dgram_len;   /**< UDP datagram length */
        rte_be16_t dgram_cksum; /**< UDP datagram checksum */
} __rte_packed;
```

2. Write a DPDK application to construct and send UDP packets. (Refer to examples/skeleton). Before writing any code, you need to enable the virtual NIC in your virtual machine. If you are using VMware, refer to this website to enable a host-only mode virtual NIC: https://blog.csdn.net/ka_ka314/article/details/78936105

After virtual NIC is enabled, use following command to bind it to DPDK:

```
sudo ifconfig ens33 down #停止 ens33 网卡的工作， ens33 为刚激活的 host-only 模式下的虚拟网卡
#（ens33 可替换成其他网卡， 在绑定网卡到 DPDK 前需要使其从活跃状态变为不活跃状态）
sudo modprobe uio
cd ../../kernel/linux/igb_uio
make
sudo insmod igb_uio.ko
cd ../../..
sudo usertools/dpdk-devbind.py --bind=igb_uio ens33
sudo usertools/dpdk-devbind.py -s #验证虚拟网卡是否被成功绑定到 DPDK 上
```

For binding ports to DPDK modules, refer to the contents of chapter 5.4 of http://doc.dpdk.org/guides/linux_gsg/linux_drivers.html

3. To verify the correctness of your program, you need to capture the packets and display the contents of them. Wireshark is the world's foremost and widely-used network protocol analyzer. It can help to capture the packets sent to NIC. Install it in your physical machine (NOT in virtual machine) and test your program. https://www.wireshark.org/

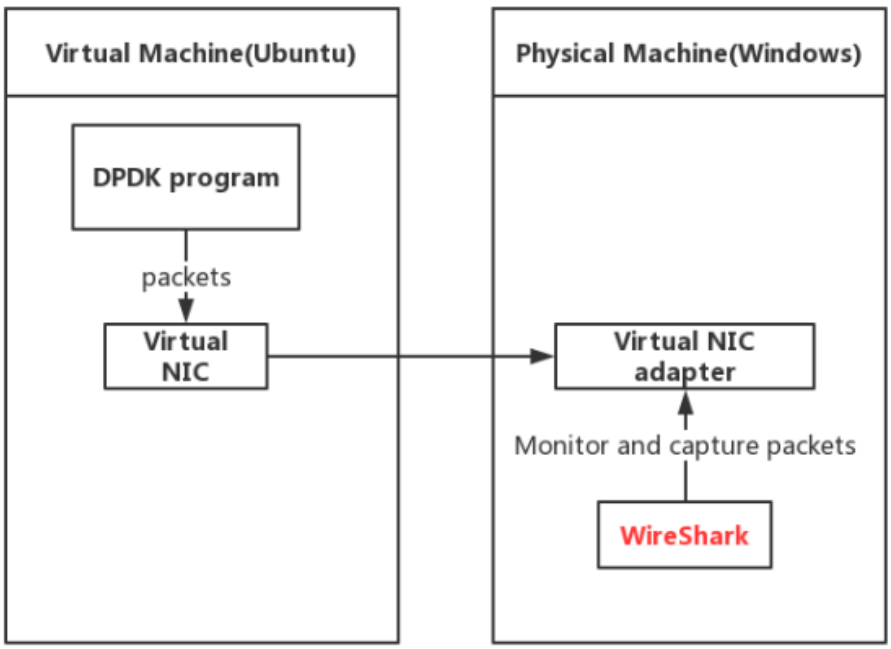


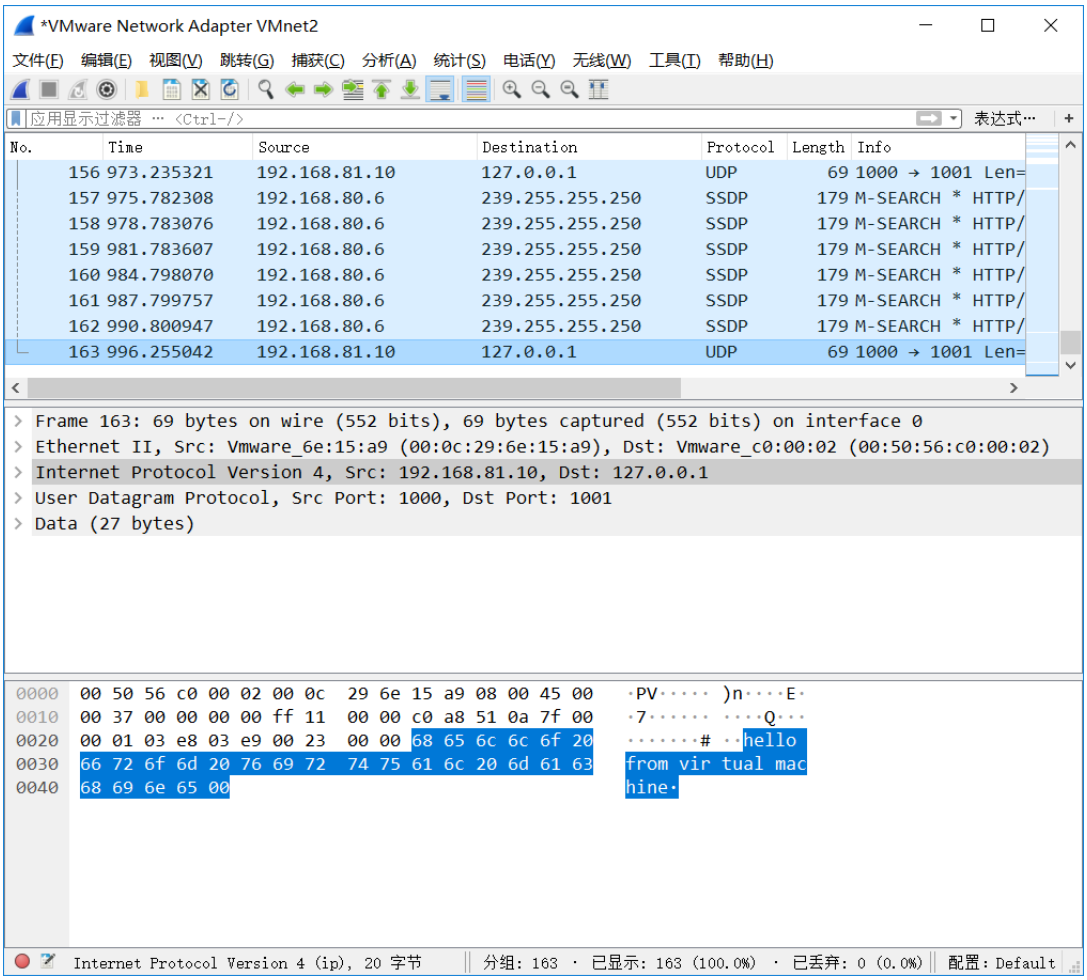
Diagram used to show how to test your program with Wireshark

Tips: If your enabled virtual NIC is in host-only mode, you need to capture packets from the same virtual NIC in your physical machine using wireshark. If your enabled virtual NIC is in bridgeconnection mode, you need to capture packets from the NIC your virtual NIC bridged to in your physical machine using wireshark. NAT mode is not advised.

Handin procedure:

You need to submit the source code of the DPDK application, and a README file including the answers to the questions in part 1 and a description of how you verify the correctness of the application with the screenshot of Wireshark in part 2.

The screenshot is supposed to look like this:



Send your report and source code as a gzipped tar file, named as {Your studentID}.tar.gz, to CANVAS

Grading:

- 40% for your document totally. 20% for the answers to the questions in part 1, and another 20% for a description of how you verify the correctness of the application with the screenshot of Wireshark in part 2.
- 60% for the code of DPDK program. Code will be checked manually and results will be verified by wireshark.

Reference:

1. DPDK official website: http://www.dpdk.org/
2. Helloworld 例程 http://doc.dpdk.org/guides/sample_app_ug/hello_world.html
3. 深入浅出DPDK https://book.douban.com/subject/26798275/
4. DPDK APIs http://doc.dpdk.org/api/