DPDK-HTTPDUMP

Config System

1. isolcpus
    - 将指定cpu设置为独占状态
    - 建议仅隔离所需使用的CPU核心，一般不要使用core 0
    - 格式: 1，2，3 或者 1-3
2. hugepages
    - 设置hugepages大小及数量
    - CPU支持的hugepage大小可以通过CPU的flags得知
        - 使用命令cat /proc/cpuinfo | grep flags查看
        - 如果pse存在，则支持2M的hugepage
        - 如果pdpe1gb存在，则支持1G的hugepage
        - 对于64位系统，如果平台支持，建议使用1GB大小的hugepage
    - 最多Hugepage页面数量取决于物理内存大小，占用内存过多会引起性能降低
3. transparent_hugepage
    - 默认开启的此功能可能会引起hugepage被其他程序占用，为提高性能需要将其关闭
4. Example
    - 将cpu 1-26设置为独占
    - 默认开启1G大小Hugepage，默认页面数量32；或者开启64个2M大小Hugepage
    - 关闭transparent_hugepage
    - 重启系统使修改生效

```
#若支持1G大小hugepage
vim /etc/default/grub
    GRUB_CMDLINE_LINUX_DEFAULT="isolcpus=1-26 default_hugepagesz=1G hugepagesz=1G hugepages=32"
update-grub

echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag

reboot
```

```
#若不支持1Gc
vim /etc/default/grub
    GRUB_CMDLINE_LINUX_DEFAULT="isolcpus=1-26"
update-grub

mkdir -p /mnt/huge
mount -t hugetlbfs nodev /mnt/huge
echo 64 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages

echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo never > /sys/kernel/mm/transparent_hugepage/defrag

reboot
```

Config DPDK

```
apt instal make gcc libpcap-dev
wget http://fast.dpdk.org/rel/dpdk-17.05.tar.xz
tar -xf dpdk.tar.xz
./DPDK/usertools/dpdk-setup.sh
```

1. Step 1: Select the DPDK environment to build
    - 根据架构-平台-应用类型-编译器进行选择
    - 对于一般x86_64平台选择[12]即x86_64-native-linuxapp-gcc即可
2. Step 2: Setup linuxapp environment
    - 按顺序进行以下操作：
        1. [15] Insert IGB UIO module
        2. [21] Bind Ethernet/Crypto device to IGB UIO module
            - 根据所列出的网络设备PCI地址进行绑定

Compile

```
vim /etc/profile
    export  RTE_SDK={your_dpdk_path}
    export  RTE_TARGET=x86_64-native-linuxapp-gcc

vim dpdk-httpdump/bypasscap.h
    line 40: char l_flag[] = "-l0-3"; #配置需要使用的逻辑核心

cd dpdk-httpdump && make
```

Usage

```
Usage: %s [-c <per_port_c_cores>] [-w <num_w_cores>]  [-p <portmask>] [-r <rotate_seconds>]
        -c      Number of cores to capture packages on each port
        -w      Number of cores to analysis traffic
        -p      An hexadecimal bit mask of the port to listen on
        -r      Every X seconds to generate new files for writing
Example: ./dpdk-httpdump -c3 -w6 -p0x3 -r600
```

- per_port_c_cores
    - 每个端口上用来捕获数据包的核心数量
    - 一般10Gps网络环境下设置为4即可
- num_w_cores
    - 用来分析数据包的核心数量
    - 不能超过捕获核心的总数，即num_w_cores <= per_port_c_cores * ports
    - 建议设置为per_port_c_cores * ports
- portmask
    - 用来设置监听端口的十六进制掩码
    - 如已绑定2张网卡
        - 需要监听第二张则设置为0x2
        - 需要监听第一张及第二张则设置为0x3
- rotate_seconds
    - 记录文件的刷新间隔
    - 每隔X秒产生一系列新的记录文件
- 如果运行时出现EAL: Error reading from file descriptor XX: Input/output error错误
    - 更改dpdk/lib/librte_eal/linuxapp/igb_uio/igb_uio.c

        ```
        @@ -404,14 +404,11 @@  igbuio_pci_probe(struct pci_dev *dev, const struct pci_device_id
        *id)
        ```

```
        }
        /* fall back to INTX */
    case RTE_INTR_MODE_LEGACY:
-               if (pci_intx_mask_supported(dev)) {
-                       dev_dbg(&dev->dev, "using INTX");
-                       udev->info.irq_flags = IRQF_SHARED;
-                       udev->info.irq = dev->irq;
-                       udev->mode = RTE_INTR_MODE_LEGACY;
-                       break;
-               }
-               dev_notice(&dev->dev, "PCI INTX mask not supported\n");
+               dev_dbg(&dev->dev, "using INTX");
+               udev->info.irq_flags = IRQF_SHARED;
+               udev->info.irq = dev->irq;
+               udev->mode = RTE_INTR_MODE_LEGACY;
+               break;
        /* fall back to no IRQ */
    case RTE_INTR_MODE_NONE:
        udev->mode = RTE_INTR_MODE_NONE;
```

- 重新回到Config DPDK步骤

Clean Up

```
./DPDK/usertools/dpdk-setup.sh
```

1. [27] Unbind devices from IGB UIO or VFIO driver
   - 解绑网卡
   - 需要输入网卡驱动，如ixgbe
2. [28] Remove IGB UIO module