Learnings documented: HPDOS project

Mellanox Bluefield DPU, DPDK, mTCP

Rinku Shah
rinku@cse.iitb.ac.in

Project PI

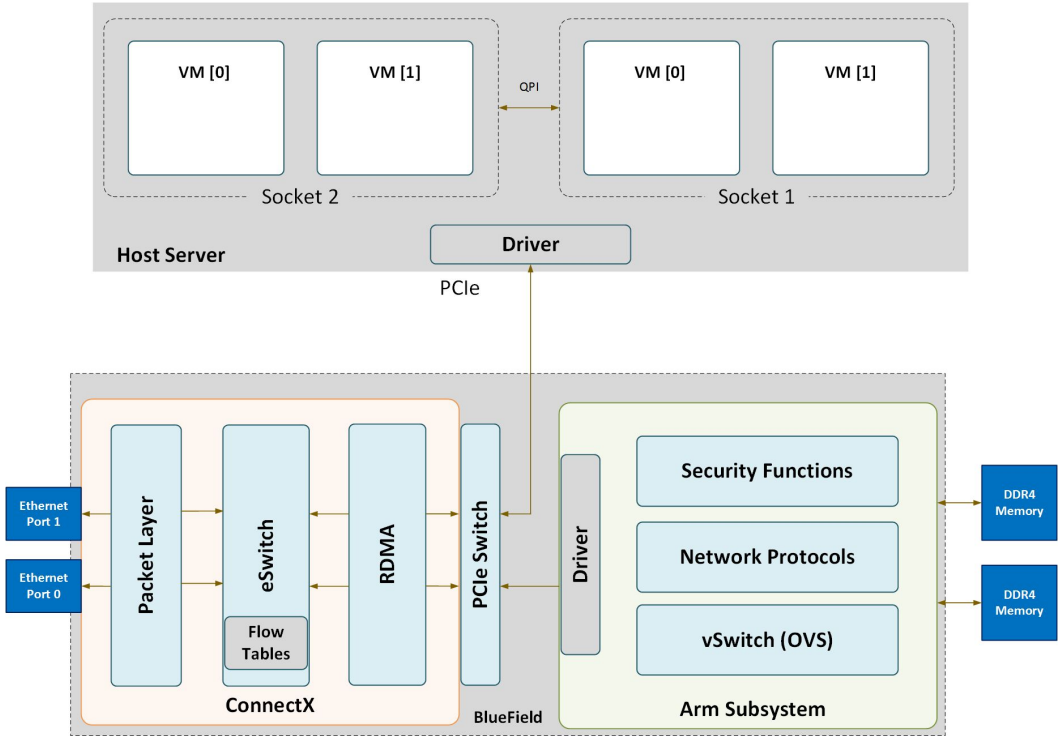
1. Prof. Umesh Bellur
2. Prof. Purushottam Kulkarni

May 26, 2021

Outline

- **Mellanox Bluefield DPU basics**
  - Architecture
  - Modes of operation
  - Configuration
    - DPU communication
    - Configure Bluefield DPU in *SEPARATED* mode
    - Example physical setup and configuration
- **Configure DPDK for Bluefield DPU**
  - Run example *testpmd* application
- **Configure mTCP for Bluefield DPU**
  - Run example *HTTP-based KV server* application
- **Configure SR-IOV Virtual Functions (VFs)**
- **Configuration for Bluefield's Embedded Function mode (OPTIONAL)**
  - Configure Bluefield DPU in *EMBEDDED FUNCTION* mode
  - Configure OpenVswitch (OVS)
  - Configure mediated devices

Mellanox Bluefield DPU: Functional diagram



Reference: https://docs.mellanox.com/display/BlueFieldSWv36011699/Functional+Diagram

Video link: https://drive.google.com/file/d/1WJgQ7PQQy-HT3t0zUYk24jWq4XbPTtkZ/view?usp=sharing

Configuration of SmartNIC communication with host & outside world

On the host, do the following (as root user)

1. Assign IP address to the rshim interface on the host side

   $ ifconfig tmfifo_net0 192.168.100.1 netmask 255.255.255.0 up

   SmartNIC's rshim interface IP: 192.168.100.2; Use this IP to ssh into smartNIC

2. Add an iptable rule; Host acts as a proxy between the smartNIC and Internet

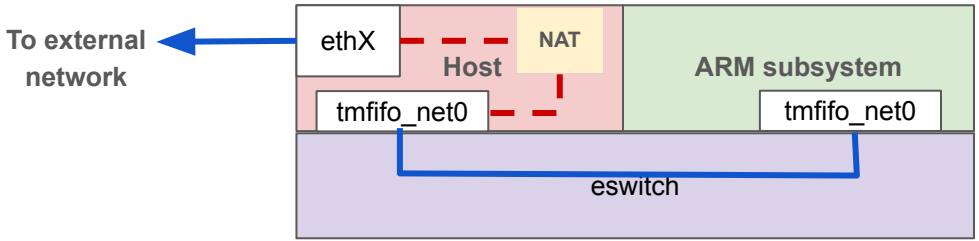
   $ iptables -t nat -A POSTROUTING -o <iface-name> -j MASQUERADE

   iface-name: host interface that is used to send packets to/fro the Internet

3. Enable IP forwarding on the host

   $ echo 1 > /proc/sys/net/ipv4/ip_forward

Tip: Write a script that includes these commands and run it after every system reboot



Mellanox Bluefield DPU: Modes of operation

- **Embedded function (ECPF) ownership**
  - Embedded Arm system
    - controls NIC resources
    - controls data path
  - Default mode
- **Separated host mode**
  - Symmetric model
- **Restricted mode**
  - extension of the ECPF ownership
  - additional restrictions on the host side

This is the **MODE WE USE**

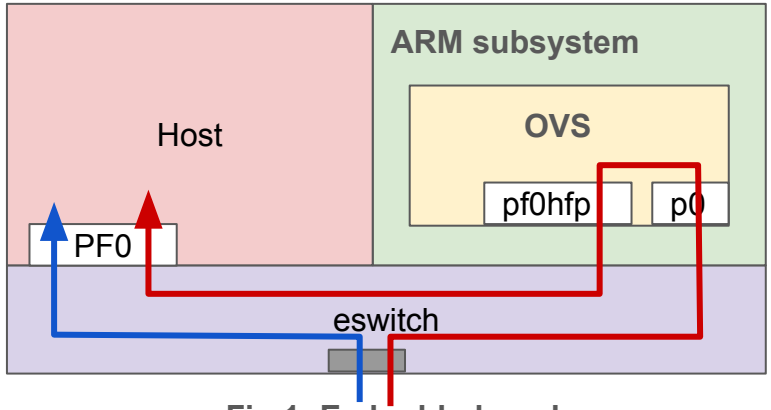


Fig.1: Embedded mode

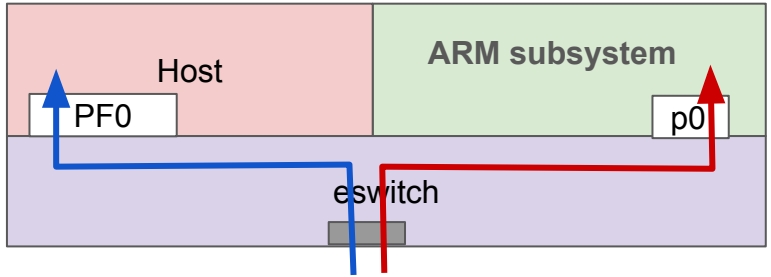
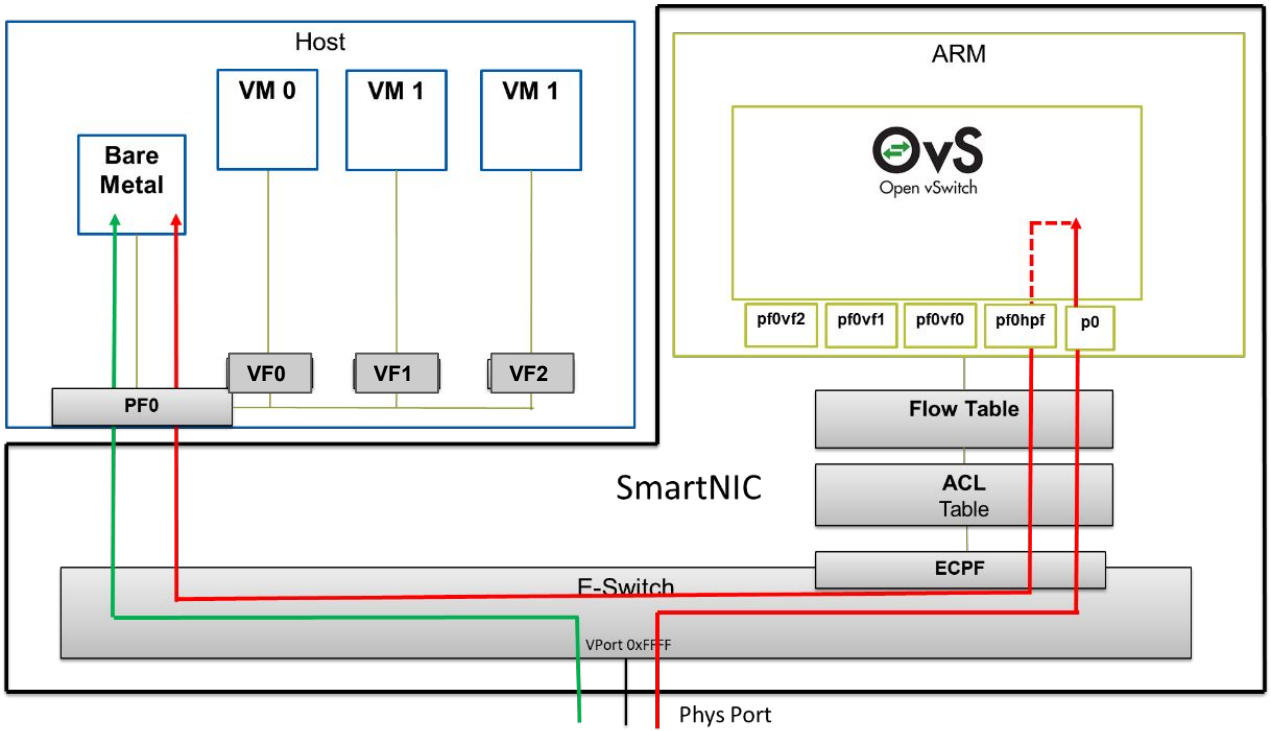


Fig.2: Separated mode

Reference: https://docs.mellanox.com/display/BlueFieldSWv36011699/Modes+of+Operation#ModesofOperation-SeparatedHost

Video link: https://drive.google.com/file/d/1wGDnyth2Yd_wpbB3RJv_Ni2Q23MKBuhE/view?usp=sharing

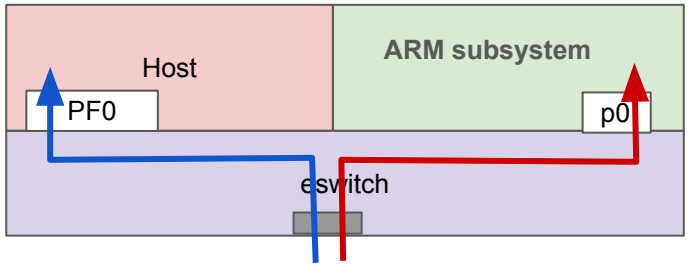
Mellanox Bluefield DPU: Kernel representors model



- Uplink representors
  p<port_number>
- PF representors
  pf<port_number>hpf
- VF representors
  pf<port_number>vf<function_number>

Reference: https://docs.mellanox.com/display/BlueFieldSWv36011699/Kernel+Representors+Model

Configure Bluefield DPU in Separated mode



On the host, do the following (**requires root privileges**)

1. Start MST (Mellanox Software Tools) driver set service:
   $ mst start
2. Enable separated host mode
   $ mlxconfig -d /dev/mst/mt41682_pciconf0 s INTERNAL_CPU_MODEL=0
   INTERNAL_CPU_MODEL: 0: Separated mode; 1: Embedded Function mode
3. Restart the HOST
   $ reboot
4. Verify the configuration
   $ mst start
   $ mlxconfig -d /dev/mst/mt41682_pciconf0 q | grep -i model
   You should confirm that the INTERNAL_CPU_MODEL value is set to 0

On the ARM side, do the following

1. Remove OVS bridges configuration from the Arm-side (for all OVS bridges)
   $ ovs-vsctl del-br <bridge_name>

Physical setup and configuration

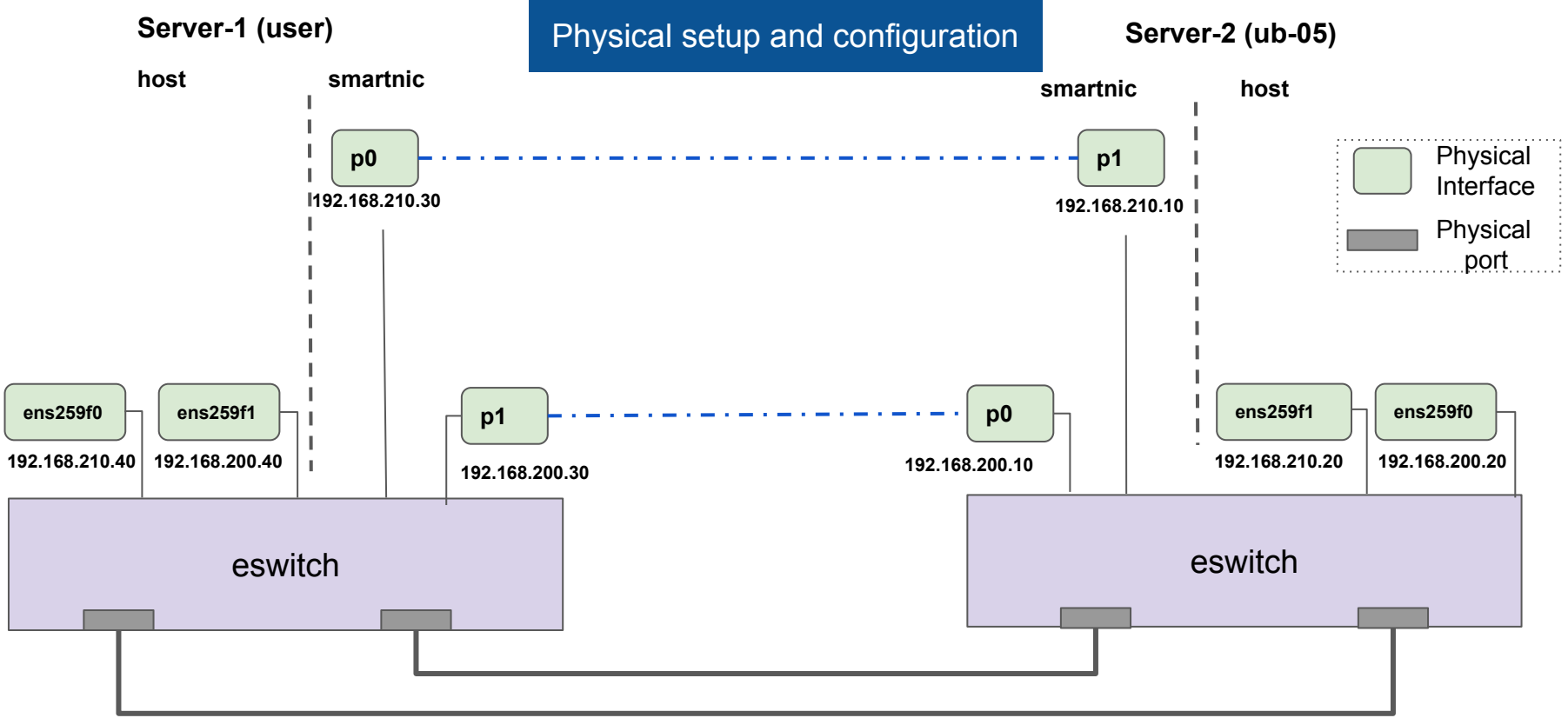


Fig. SmartNIC configuration: SEPARATED mode

Video link: https://drive.google.com/file/d/1ITMtASNNkmBZ65K038IIzjC43Sw1yPld/view?usp=sharing

Host and SmartNIC (DPU) network configuration

- **On “ub-05” smartNIC:** Configure p0 and p1

  $ sudo ifconfig p0 192.168.200.10/24 up

  $ sudo ifconfig p1 192.168.210.10/24 up

- **On “ub-05” host:** Configure ens259f0 and ens259f1

  $ sudo ifconfig ens259f0 192.168.200.20/24 up

  $ sudo ifconfig ens259f1 192.168.210.20/24 up

- **On “user” smartNIC:** Configure p0 and p1

  $ sudo ifconfig p0 192.168.210.30/24 up

  $ sudo ifconfig p1 192.168.200.30/24 up

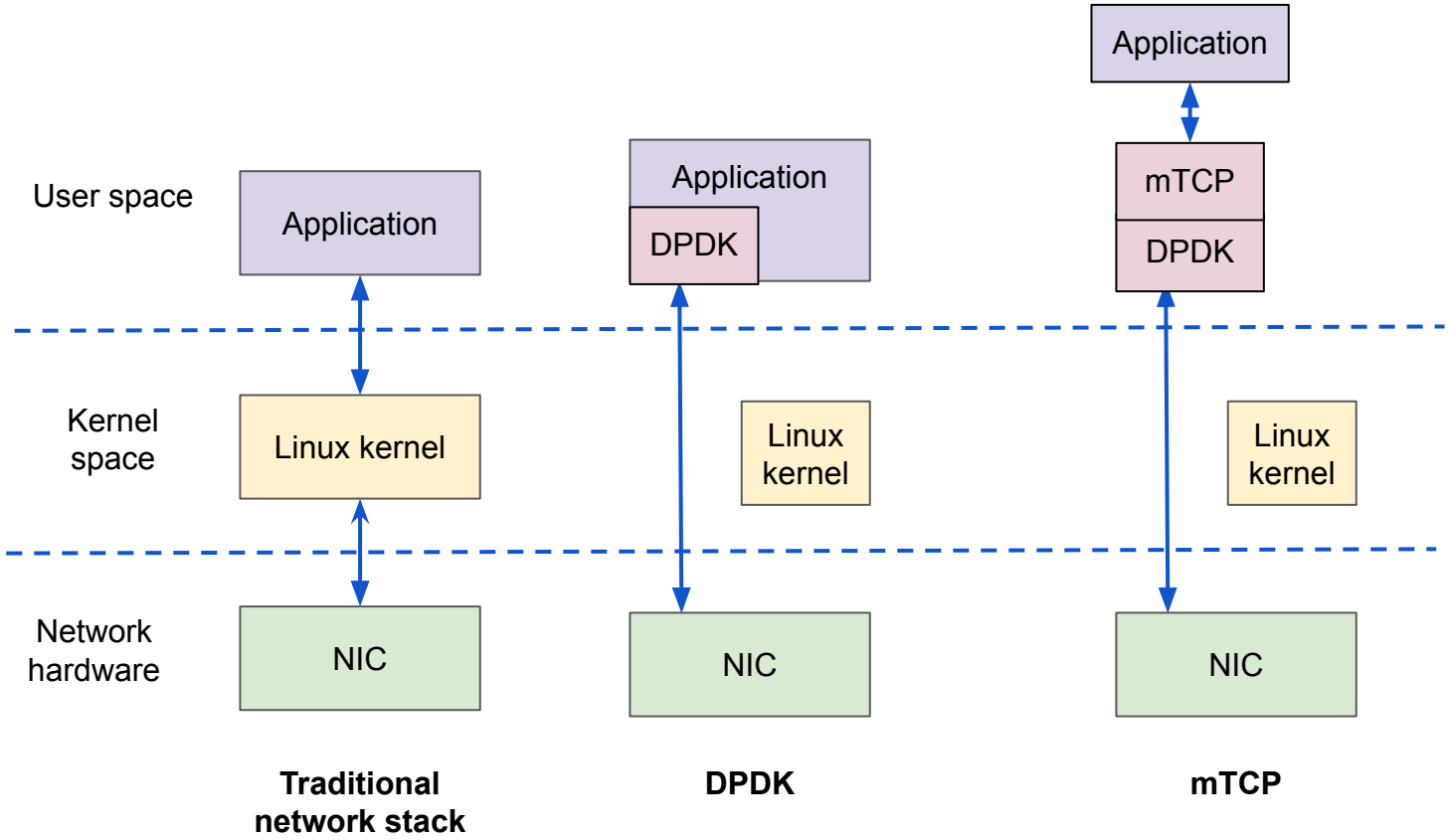
- **On “user” host:** Configure ens259f0 and ens259f1

  $ sudo ifconfig ens259f0 192.168.210.40/24 up

  $ sudo ifconfig ens259f1 192.168.200.40/24 up

Note: The above configuration is as per the figure on slide 8

Traditional network stack vs. DPDK vs. mTCP



Network stack content prepared by Myself and some IITB students*

1. Theory on traditional networking, DPDK, netmap, and mTCP

   https://www.youtube.com/watch?v=MpjlWt7fvrw&list=PLp9oIlA12dfZi09j7LZ71iglR6LLhQh-m&index=5&t=1027s

2. Jump start to run DPDK applications. Follow current slides for installation/configuration; use the video to understand concepts and write + run DPDK applications

   https://www.youtube.com/watch?v=VJ8CVN3oXMw&list=PLp9oIlA12dfZi09j7LZ71iglR6LLhQh-m&index=4&t=936s

* Priyanka Naik, Diptyroop Maji, and Nilesh Unhale

Configure Data Plane Development Kit (DPDK) for setup with Mellanox Bluefield DPU

1. Download the required DPDK compressed version & decompress it.
   a. https://core.dpdk.org/download/
   b. It is good to choose a stable (LTS) version
2. For Mellanox Bluefield NICs, check if the flag for Mellanox driver installation is set *(for DPDK version < 20.11.8 only)*

   <path-to-dpdk-folder>$ vim config/common_linux
   a. *Check if the Poll Mode Driver (PMD) for your Mellanox NIC is set; in our case it is MLX5*
   b. *Add "CONFIG_RTE_LIBRTE_MLX5_PMD=y", if it does not exist*
3. DPDK configuration alternatives
   a. **Alternative 1:** For newer DPDK versions (>= version 18)

      <path-to-dpdk-folder>$ meson build

      <path-to-dpdk-folder>$ cd build

      <path-to-dpdk-folder>$ ninja

      <path-to-dpdk-folder>$ ninja install
   b. **Alternative 2:** For older DPDK versions & ***for our mTCP setup***

      Continued on next slide

Reference: https://community.mellanox.com/s/article/Configuring-DPDK-and-Running-testpmd-on-BlueField-2

Configure Data Plane Development Kit (DPDK) for setup with Mellanox Bluefield DPU (contd.)

Alternative 2: For older DPDK versions & ***for our mTCP setup (use dpdk-19.11.8 for mTCP)***

a. Run:

<path-to-dpdk-folder>/usertools$ sudo ./dpdk-setup.sh

b. Choose

i. **7**: arm64-bluefield-linuxapp-gcc (NIC installation)

ii. **41**: x86_64-native-linuxapp-gcc (Host installation)

Corresponding folder is created at DPDK root folder

For both alternatives:

a. Mellanox driver, mlx5 supports kernel and poll mode

i. No need to unbind the interface from kernel

b. Allocate huge pages for DPDK operation

$ sudo sysctl -w vm.nr_hugepages=4096

c. Test huge page allocation

$ cat /proc/meminfo | grep -i huge

NOTE: *In case of numa.h not found error*

$ sudo apt install libnuma-dev

```
ubuntu@linux:~/dpdk-stable-19.11.8/usertools$ sudo ./dpdk-setup.sh
------------------------------------------------------------------------------
 RTE_SDK exported as /home/ubuntu/dpdk-stable-19.11.8
------------------------------------------------------------------------------
----------------------------------------------------------
 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-dpaa-linuxapp-gcc
[10] arm64-dpaa-linux-gcc
[11] arm64-emag-linuxapp-gcc
[12] arm64-emag-linux-gcc
[13] arm64-graviton2-linuxapp-gcc
[14] arm64-graviton2-linux-gcc
[15] arm64-n1sdp-linuxapp-gcc
[16] arm64-n1sdp-linux-gcc
[17] arm64-octeontx2-linuxapp-gcc
[18] arm64-octeontx2-linux-gcc
[19] arm64-stingray-linuxapp-gcc
[20] arm64-stingray-linux-gcc
[21] arm64-thunderx2-linuxapp-gcc
[22] arm64-thunderx2-linux-gcc
[23] arm64-thunderx-linuxapp-gcc
[24] arm64-thunderx-linux-gcc
[25] arm64-xgene1-linuxapp-gcc
[26] arm64-xgene1-linux-gcc
[27] arm-armv7a-linuxapp-gcc
[28] arm-armv7a-linux-gcc
[29] graviton2
[30] i686-native-linuxapp-gcc
```

Run DPDK application to test network bandwidth (*testpmd*)

1. Identify PCI address of the NIC port which you want to bind to DPDK

   $ lspci

   03:00.0 Ethernet controller: Mellanox Technologies MT416842 BlueField integrated ConnectX-5 network controller => p0

   03:00.1 Ethernet controller: Mellanox Technologies MT416842 BlueField integrated ConnectX-5 network controller => p1

2. Run testpmd on the two smartNICs connected directly using a cable

   <path-to-dpdk-folder>/arm64-bluefield-linuxapp-gcc/build/app/test-pmd$ sudo ./testpmd -w 03:00.1 -- --nb-cores=2 --txq=2 -i

   Parameters explained : w: whitelist device; nb-cores: #forwarding cores; txq: #transmit queues (requires RSS); i: interactive mode; a: automatic traffic start

   You can check for more configuration parameters: testpmd> help

3. In the interactive mode, some configurations before testing network bandwidth

“ub-05” server smartNIC	**“user” server smartNIC**
testpmd> set fwd txonly *testpmd> set eth-peer 0 0c:42:a1:df:ac:41* *testpmd> set eth-peer 1 0c:42:a1:df:ac:40* *testpmd> start* *testpmd> show port stats all //repeat*	*testpmd> set fwd rxonly* *testpmd> set eth-peer 0 0c:42:a1:df:ac:49* *testpmd> set eth-peer 1 0c:42:a1:df:ac:48* *testpmd> start* *testpmd> show port stats all //repeat*

Sample DPDK's testpmd output for: testpmd> show port stats all

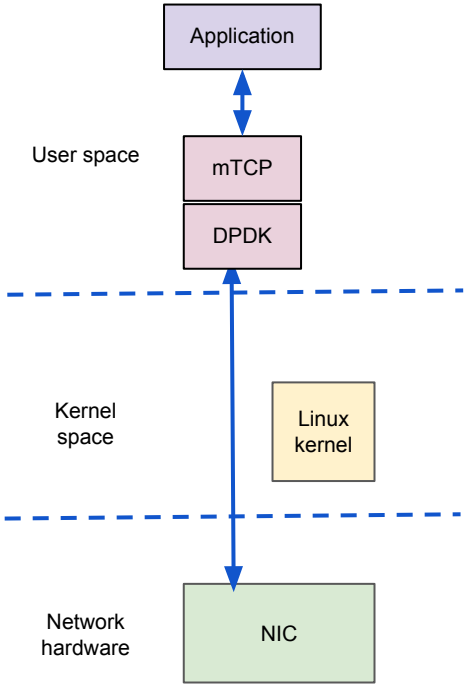
testpmd> show port stats all

```
######################## NIC statistics for port 0  ########################
RX-packets: 29104464   RX-missed: 0          RX-bytes:  2124625872
RX-errors: 0
RX-nombuf:  0
TX-packets: 710763473  TX-errors: 0          TX-bytes:  45752099065

Throughput (since last show)
Rx-pps:            0          Rx-bps:            0
Tx-pps:      6070638          Tx-bps:   3108170088
############################################################################

######################## NIC statistics for port 1  ########################
RX-packets: 29248350   RX-missed: 0          RX-bytes:  2135129550
RX-errors: 0
RX-nombuf:  0
TX-packets: 706903758  TX-errors: 0          TX-bytes:  45503782430

Throughput (since last show)
Rx-pps:            0          Rx-bps:            0
Tx-pps:      6037813          Tx-bps:   3091364136
############################################################################
```

Note: I have implemented DPDK-based key-value server over UDP, and it is available at https://git.cse.iitb.ac.in/synerg/hpdos/tree/metadata-lsmtree/code/lsm-tree-impl/smartnic-app/metadata-server-read-offload/dpdk-server/kv-cache

Multicore-TCP (mTCP) for setup with Mellanox Bluefield DPU

- mTCP runs over DPDK or netmap platform
- mTCP git code is compatible upto DPDK-18 version
- Mellanox Bluefield driver (mlx5) support starts from DPDK-19.11.8
- I have modified mTCP code for mlx5
  - compatible with DPDK-19.11.8 (details on next slide)
- Download the mTCP code from “devel” branch
  - *$ wget https://github.com/mtcp-stack/mtcp/archive/refs/heads/devel.zip*
  - *$ unzip devel.zip*



Video link: https://drive.google.com/file/d/1J1JhjtvtvuFrxeF_z6rXvonki5YdcKUz/view?usp=sharing

Code changes for mTCP for Mellanox Bluefield DPU based setup

Note: The working version of mtcp-devel with all code modifications is available in the git folder.
Follow these steps only if you are starting with a fresh mtcp-devel download

Errors / Issues	Code changes
• core.c: In function ‘mtcp_create_context’: core.c:1332:4: error: ‘lcore_config’ undeclared (first use in this function) 1332 | lcore_config[master].ret = 0;	**Comment the lines as shown below:** ***.../mtcp-devel$*** *vim mtcp/src/core.c* ... if (master == whichCoreID(cpu)) { **//lcore_config[master].ret = 0;** **//lcore_config[master].state = FINISHED;** ...
• /usr/include/aarch64-linux-gnu/bits/string_fortified.h:106:10: error: ‘__builtin_strncpy’ output may be truncated copying 1023 bytes from a string of length 1023 [-Werror=stringop-truncation] 106	return __builtin___strncpy_chk (__dest, __src, __len, __bos (__dest));	**Replace the line as shown below:** ***.../mtcp-devel$*** *vim mtcp/src/config.c* ... **- strncpy(optstr, line, MAX_OPTLINE_LEN - 1);** **+ memcpy(optstr, line, MAX_OPTLINE_LEN - 1);**
• Runtime error when the DPDK tries to bind to the Mellanox NIC port	**Comment the complete function and add return statement** *.../mtcp-devel$ vim mtcp/src/io_module.c* **Comment code for “probe_all_rte_devices” ; add return -1**

Compile and Configure mTCP

1. Download and configure ***DPDK-19.11.8*** using ***Alternative-2***, as shown in ***slide 13***
2. Setup the MTCP-DPDK environment; this involves integration of DPDK with mTCP
   a. *$ export RTE_SDK=<absolute-path-to-dpdk-stable-19.11.8>*
   b. *$ export RTE_TARGET=arm64-bluefield-linuxapp-gcc //For Bluefield smartNIC* ***OR***
      $ export RTE_TARGET=x86_64-native-linuxapp-gcc //For Host
   c. Configure mTCP with our DPDK version
      .../mtcp-devel$ ./configure --with-dpdk-lib=$RTE_SDK/$RTE_TARGET
      CFLAGS="-DMAX_CPUS=<num_host/nic_cpus>"
      // Use --disable-hwcsum when working in virtualized environment
      // If there an error: "configure: error: Could not find gmp.h"; $ sudo apt install libgmp-dev
   d. Integrate mTCP with our DPDK version
      <path-to-mtcp-devel>$ sudo ./setup_mtcp_dpdk_env.sh <absolute-path-to-dpdk-stable-19.11.8>
3. Compile mTCP
      .../mtcp-devel$ make -j
4. To revert back mTCP NIC changes
      <path-to-mtcp-devel>$ ./setup_linux_env.sh <path to $RTE_SDK>

Reference: https://github.com/mtcp-stack/mtcp

Run mTCP application: HTTP client-server: Configuration-I

1. Configure static ARP entries

   .../mtcp-devel/apps/example/config$ vim arp.conf

   //Add ARP entries for remote IPs (see example in Table below)

"user" HOST	"ub-05" NIC
ARP_ENTRY 6 192.168.200.10/32 0c:42:a1:df:ac:48 192.168.210.10/32 0c:42:a1:df:ac:49 192.168.200.20/32 0c:42:a1:df:ac:42 192.168.210.20/32 0c:42:a1:df:ac:43 192.168.200.30/32 0c:42:a1:df:ac:41 192.168.210.30/32 0c:42:a1:df:ac:40	ARP_ENTRY 6 192.168.210.30/32 0c:42:a1:df:ac:40 192.168.200.30/32 0c:42:a1:df:ac:41 192.168.200.20/32 0c:42:a1:df:ac:42 192.168.210.20/32 0c:42:a1:df:ac:43 192.168.210.40/32 0c:42:a1:df:ac:3a 192.168.200.40/32 0c:42:a1:df:ac:3b

2. Configure static route entries

   .../mtcp-devel/apps/example/config$ vim route.conf

   //Add route entries for remote IPs (see example in Table below)

"user" HOST	"ub-05" NIC
ROUTES 2 192.168.210.0/24 ens259f0 192.168.200.0/24 ens259f1	ROUTES 2 192.168.200.0/24 p0 192.168.210.0/24 p1

Run mTCP application: HTTP client-server: Configuration-II

1. Configure HTTP server configuration

   .../mtcp-devel/apps/example$ vim epserver.conf

   - **num_cores =** 4 //set the number of cores that will run mTCP
   - **core_mask =** 0F // bit-mask that specifies which cores to use; 0F=00001111; use cores 0-3
   - **num_tx_desc =** 512 //set TX descriptor ring size
   - **num_rx_desc =** 128 //set RX descriptor ring size
   - **port =** p0 //set the port that should be mapped for mTCP
   - **stat_print =** p0 //set the ports for which stats should be printed
   - *You can configure other parameters too for optimized performance*

2. Configure HTTP client configuration

   ..../mtcp-devel/apps/example$ vim epwget.conf

   Configure the client-side mTCP for the parameters listed in (1)

Reference: https://github.com/mtcp-stack/mtcp/tree/master/apps/example

Run mTCP application: HTTP client-server

1. Configure the RTE vars

   $ export RTE_SDK=<absolute-path-to-dpdk-stable-19.11.8>

   $ export RTE_TARGET=<target>

2. For mTCP based HTTP server
   a. Create a directory where files that are requested for transfer are stored
   b. Run the HTTP server

   .../mtcp-devel/apps/example$ sudo ./epserver -p <absolute-path-to-www-dir> -f epserver.conf [-N <num-cores>]

   Example:

   ~/mtcp-devel/apps/example$ sudo ./epserver -p /home/ubuntu/www -f epserver.conf -N 2

3. For mTCP based HTTP client

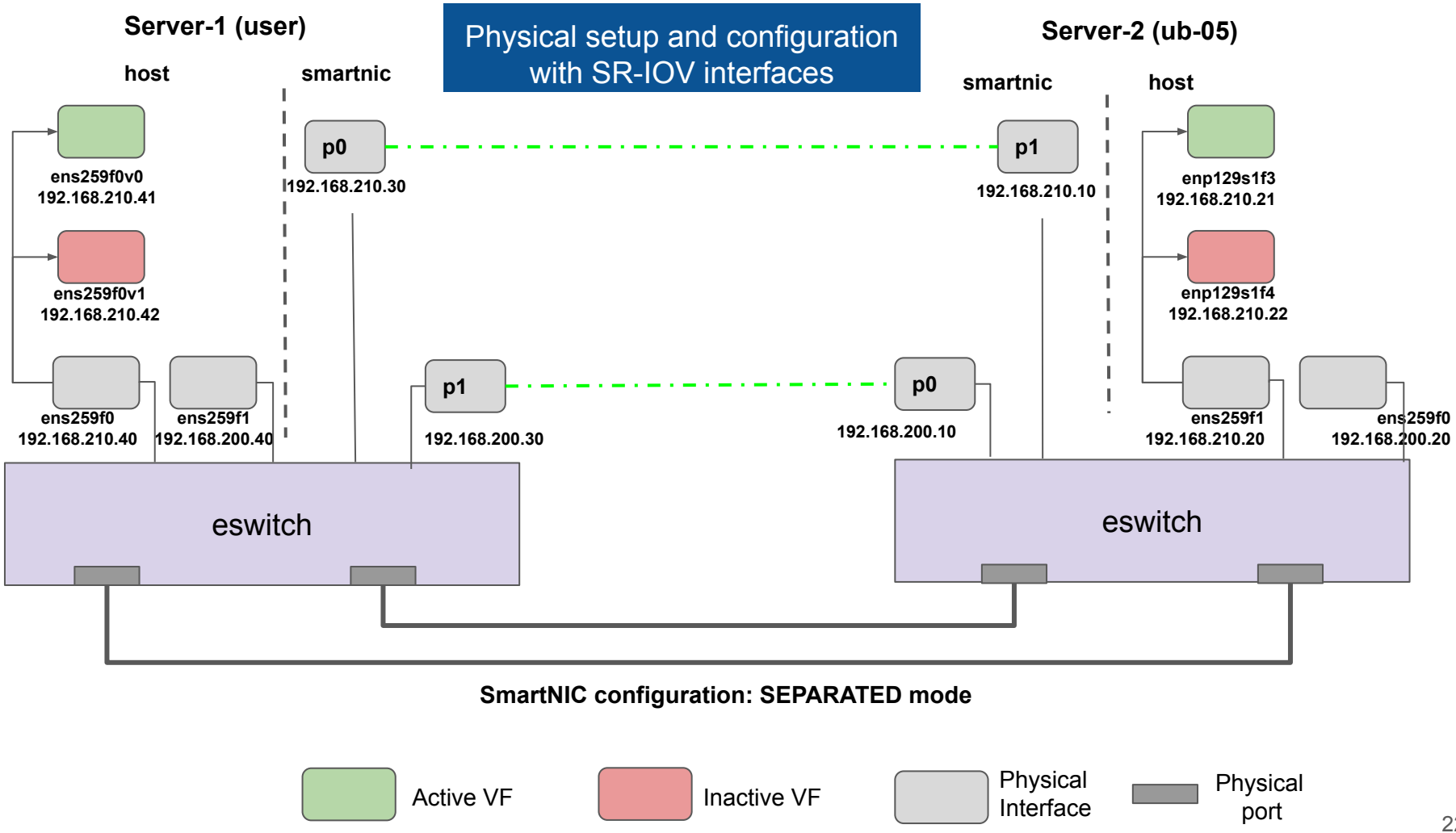
   .../mtcp-devel/apps/example$ sudo ./epwget <server-IP>/<file-name> <num_requests> [-N #cores] [-c concurrency] -f epwget.conf

   Example:

   ~/mtcp-devel/apps/example$ sudo ./epwget 192.168.220.35/example.txt 10000000 -N 8 -c 10000 -f epwget.conf

Reference: https://github.com/mtcp-stack/mtcp/tree/master/apps/example

Physical setup and configuration with SR-IOV interfaces



Configure SR-IOV Virtual Functions (VFs)

Run the commands as “root” user

VF configuration: “ub-05” host	VF configuration: “user” host
• *$ echo 2 > /sys/class/net/enp259f0/device/sriov_numvfs* • *$ echo 2 > /sys/class/net/enp259f1/device/sriov_numvfs* • *$ ip link //check the VF’s iface id* • *$ ifconfig ens129s1f3 192.168.210.21/24 up* • *$ ifconfig ens129s1f4 192.168.210.22/24 up* • *$ ifconfig //verify the configuration*	• *$ echo 2 > /sys/class/net/ens259f0/device/sriov_numvfs* • *$ echo 2 > /sys/class/net/ens259f1/device/sriov_numvfs* • *$ ip link //check the VF’s iface id* • *$ ifconfig ens259f0v0 192.168.210.41/24 up* • *$ ifconfig ens259f0v1 192.168.210.42/24 up* • *$ ifconfig //verify the configuration*
Ping between the hosts to test the configuration For example, at ub-05 $ ping 192.168.210.41	

The same commands can be used to configure the VFs on the smartNIC

Reference:
https://docs.mellanox.com/pages/viewpage.action?pageId=47035976#OVSOffloadUsingASAP%C2%B2Direct-SettingUpSR-IOV

Configure Bluefield DPU in Embedded mode

On the host, do the following (**as root user**)

1. Start MST (Mellanox Software Tools) driver set service:

   $ mst start

2. Enable separated host mode

   $ mlxconfig -d /dev/mst/mt41682_pciconf0 s INTERNAL_CPU_MODEL=1

   INTERNAL_CPU_MODEL: 0: Separated mode; 1: Embedded Function mode

3. Restart the HOST

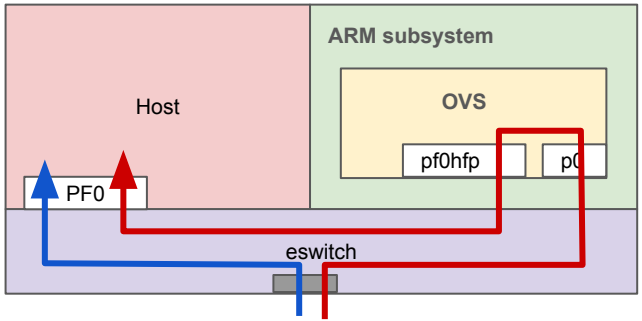
   $ reboot

4. Verify the configuration

   $ mst start

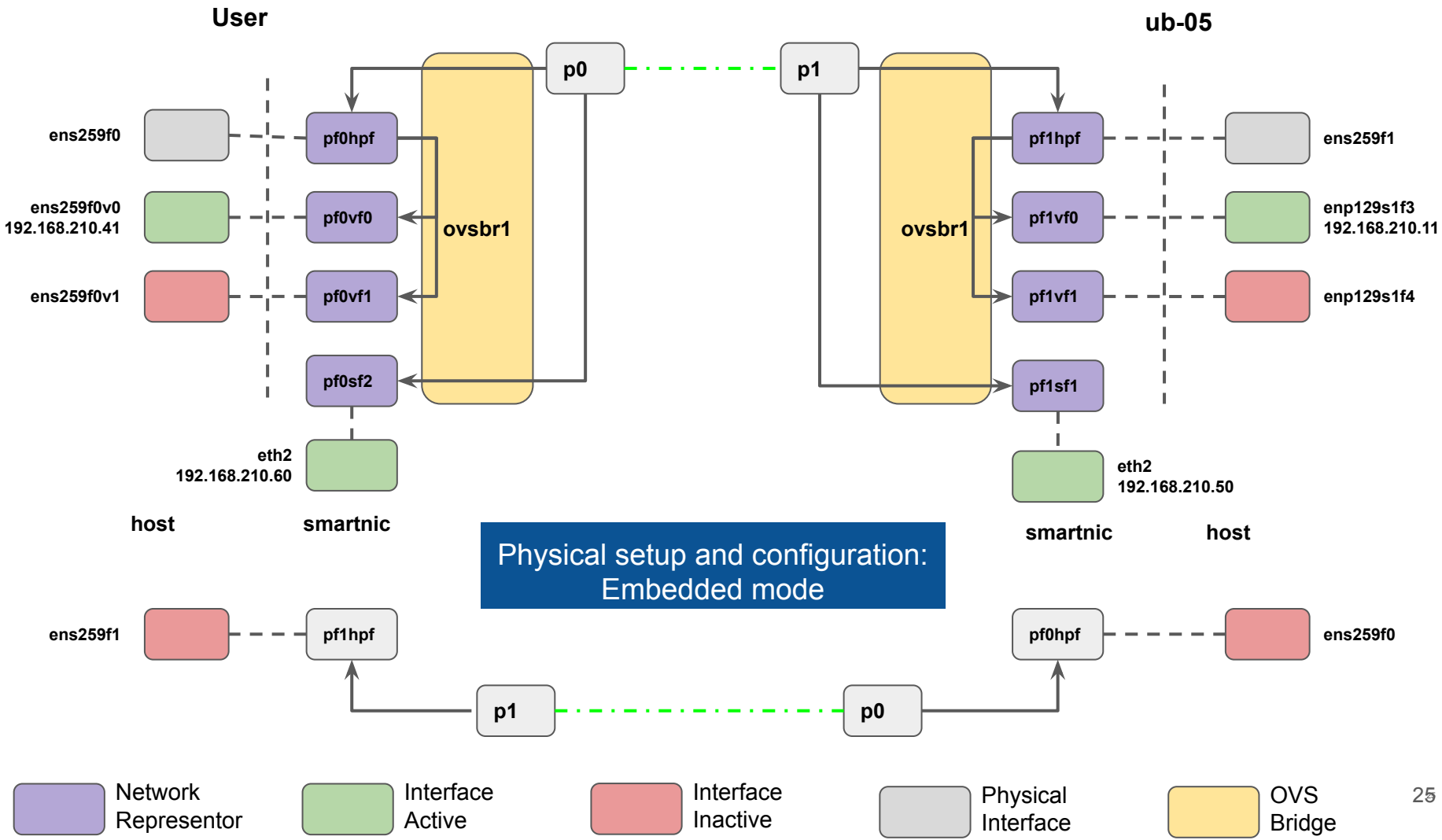
   $ mlxconfig -d /dev/mst/mt41682_pciconf0 q | grep -i model

   Confirm that INTERNAL_CPU_MODEL value is set to 1



On the ARM side, do the following (details in slides 26--29)

1. Add OVS bridge
2. Create and configure VFs (shown in slide 22)
3. Create and configure mediated devices


User
ub-05
p0
p1
ens259f0
pf0hpf
ens259f0v0
192.168.210.41
pf0vf0
ovsbr1
ens259f0v1
pf0vf1
pf0sf2
eth2
192.168.210.60
host
smartnic
pf1hpf
ens259f1
pf1vf0
enp129s1f3
192.168.210.11
ovsbr1
pf1vf1
enp129s1f4
pf1sf1
eth2
192.168.210.50
smartnic
host
Physical setup and configuration:
Embedded mode
ens259f1
pf1hpf
pf0hpf
ens259f0
p1
p0
Network Representor
Interface Active
Interface Inactive
Physical Interface
OVS Bridge

Configure OpenVswitch (OVS) **(applies to Embedded function mode only)**

- OVS runs at the smartNIC in the Embedded Function mode
- All packets arrive at the OVS; can be processed, dropped, or forwarded to host
- Look at sample OVS configuration for 2 SRIOV VFs below (requires root privileges)
  - *$ service openvswitch start*
  - *$ ovs-vsctl add-br armbr1*
  - *$ ovs-vsctl add-port armbr1 p0*
  - *$ ovs-vsctl add-port armbr1 pf0hpf*
  - *$ ovs-vsctl add-port armbr1 pf0vf0*
  - *$ ovs-vsctl add-port armbr1 pf0vf1*
  - *$ ovs-vsctl show*
- Enable OVS hardware offloading
  - # ovs-vsctl set Open_vSwitch . Other_config:hw-offload=true
  - # systemctl restart openvswitch
- Ping from NIC to the host VF to test connectivity

Reference: https://docs.mellanox.com/display/BlueFieldSWv20110841/Virtual%20Switch%20on%20BlueField%20SmartNIC

Configure Mediated devices I **(applies to Embedded function mode only)**

Mediated devices

- non-SRIOV acceleration devices created on the Bluefield system
- supports NIC and RDMA
- offer the same level of ASAP2 offloads as SR-IOV VFs

Create and configure mediated devices (requires root privileges)

1. Identify the PCI device ID. Use “lspci” as discussed in slide 9
2. Decrease the number of maximum mediated devices[mdev] --- Optional; done to avoid overheads

   $ echo 4 > /sys/bus/pci/devices/0000:03:00.0/mdev_supported_types/mlx5_core-local/max_mdevs
3. Mediated devices are uniquely identified using UUID; generate one & use it for further configuration

   $ uuidgen
4. Create mdev device using the UUID generated in step 3

   $ echo <UUID> > /sys/bus/pci/devices/0000\:03\:00.1/mdev_supported_types/mlx5_core-local/create
5. *Continued ...*

Reference: https://docs.mellanox.com/display/OFEDv512371/Mediated+Devices

Configure Mediated devices II

Create and configure mediated devices (requires root privileges) --- contd.

1. By default, the mdev device is binded to the vfio_mdev driver . To create netdevice and access RDMA we must unbind this device from the driver

   $ echo <UUID> > /sys/bus/mdev/drivers/vfio_mdev/unbind

2. Configure MAC address for the new mediated device

   $ echo <mac_addr> > /sys/bus/mdev/devices/<UUID>/devlink-compat-config/mac_addr

3. Like Bluefield SRIOV devices, mediated devices are created in pairs. One end is the mdev device on the NIC, the other end can be queried as follows

   $ cat /sys/bus/mdev/devices/<UUID>/devlink-compat-config/netdev

4. Bind the mediated device to mlx5_core driver

   $ echo <UUID> >/sys/bus/mdev/drivers/mlx5_core/bind

5. After binding mediated device to mlx5_core driver, its respective netdevice and/or RDMA device is also created. To inspect the netdevice and RDMA device, RUN

   $ ls /sys/bus/mdev/devices/<UUID>/net/

   $ ls /sys/bus/mdev/devices/<UUID>/infiniband/

6. *Continued ...*

Reference: https://docs.mellanox.com/display/OFEDv512371/Mediated+Devices

Configure Mediated devices **III**

Create and configure mediated devices (requires root privileges) --- contd.

1. Add the created mdev's PF end to OVS bridge (run as non-sudo user)

   $ sudo ovs-vsctl add-port ovsbr1 pf1sf1

2. Assign an IP address to the eth side of the interface

   $ sudo ifconfig <mdev-eth-iface-name> <ip-addr>/<mask> up

3. Pings between mdev and other devices on host/NIC should start working.

Important links

- Github code link:

  https://git.cse.iitb.ac.in/synerg/hpdos/tree/metadata-lsmtree
- Running slide doc link

  https://docs.google.com/presentation/d/1NOxpgLvCdQnCP35Ia-PdAUEgfxyiBkhS5-TimaQIERg/edit?usp=sharing
- Traditional network stack + DPDK + netmap + mTCP
  - https://www.youtube.com/watch?v=MpjlWt7fvrw&list=PLp9oIlA12dfZi09j7LZ71iglR6LLhQh-m&index=5&t=1027s
  - https://www.youtube.com/watch?v=VJ8CVN3oXMw&list=PLp9oIlA12dfZi09j7LZ71iglR6LLhQh-m&index=4&t=936s
- Documented video links:
  - https://drive.google.com/file/d/1WJgQ7PQQy-HT3t0zUYk24jWq4XbPTtkZ/view?usp=sharing
  - https://drive.google.com/file/d/1wGDnyth2Yd_wpbB3RJv_Ni2Q23MKBuhE/view?usp=sharing
  - https://drive.google.com/file/d/1ITMtASNNkmBZ65K038IIzjC43Sw1yPld/view?usp=sharing
  - https://drive.google.com/file/d/1J1JhjtvtvuFrxeF_z6rXvonki5YdcKUz/view?usp=sharing