README.md - Grip

Using 40G Mellanox Infiniband single- or dual-port NICs

Table of Contents

ToC created by gh-md-toc

Operating environment

- Ubuntu Bionic 18.04.3
- Vanilla upstream kernel 5.4.3 from kernel.org (no additional patches applied)

Should I use Infiniband mode or Ethernet mode?

Advantages of Infiniband

- Lower latency
- Can use for low-latency, high-bandwidth storage links, using iSCSI
- Can additionally use RDMA (remote DMA) for things like NFS
- If you do not have a VPI NIC, you probably cannot put the port in Ethernet mode anyway

Disadvantages of Infiniband

- Requires additional packages to be installed - opensm, rdma-core
- I have not explored using both ports of a Dual-Port 40GBit/sec ConnectX-3 VPI Mellanox NIC card in Infiniband mode - though it is possible

Advantages of Ethernet mode

- Does not require installing the `opensm` and `rdma-core` packages and running the `opensm` server
- Installation and configuration steps are simpler and more familiar
- Renaming the interface names is easier / more familiar to experienced Linux users

Using both ports of a dual-port VPI NIC - what do I get?

WITHOUT using bonding, you can ONLY have two independent network links - e.g. MachineA <---> MachineB and MachineA <---> MachineC simultaneously.

Using bonding, you can ONLY get **balanced round-robin mode** - this means:

- You do **NOT** get double the bandwidth on the bonded link
- You get RESILIENCE from:
  - Failure of one network cable / transceiver
  - A single network cable being disconnected on one or both ends
  - A single port failure on any one machine
  - Failure of a single port on both machines at either end of a single network cable

Quick start - Ethernet mode

Folliwng steps need to be done on **BOTH**machines

Package installation

`apt install mstflint infiniband-diags` You do not need `rdma-core` or `opensm`

Put your ports in Ethernet mode

```
sudo mstconfig query
```

Output will look like:

```
Device #1:
----------

Device type:    ConnectX3
PCI device:     /sys/bus/pci/devices/0000:05:00.0/config

Configurations:                             Next Boot
         SRIOV_EN                           True(1)
         NUM_OF_VFS                         16
         LINK_TYPE_P1                       ETH(2)
         LINK_TYPE_P2                       VPI(3)
         LOG_BAR_SIZE                       5
         BOOT_PKEY_P1                       0
         BOOT_PKEY_P2                       0
         BOOT_OPTION_ROM_EN_P1              True(1)
         BOOT_VLAN_EN_P1                    False(0)
         BOOT_RETRY_CNT_P1                  0
         LEGACY_BOOT_PROTOCOL_P1            PXE(1)
         BOOT_VLAN_P1                       0
         BOOT_OPTION_ROM_EN_P2              True(1)
         BOOT_VLAN_EN_P2                    False(0)
         BOOT_RETRY_CNT_P2                  0
         LEGACY_BOOT_PROTOCOL_P2            PXE(1)
         BOOT_VLAN_P2                       0
         IP_VER_P1                          IPv4(0)
         IP_VER_P2                          IPv4(0)
         CQ_TIMESTAMP                       True(1)
```

`PCI device:` /sys/bus/pci/devices/**`0000:05:00.0`**/config : Device name to use with `mstconfig` is in **bold**```

`Device type: ConnectX3` : ConnectX3 says it is a PCI-Express 3.x capable card

`LINK_TYPE_P1 VPI(3)` : Says Port 1 is in VPI (Auto) mode **`LINK_TYPE_P2 ETH(2)`** : Says Port 2 is in Ethernet mode

On a VPI-capable card, port type can be any of:

- 1 : Infiniband
- 2 : Ethernet
- 3 : VPI (Auto)

Put Port1 in Ethernet mode

```
mstconfig -d 0000:05:00.0 set LINK_TYPE_P1=2
```

Put Port2 in Ethernet mode

```
mstconfig -d 0000:05:00.0 set LINK_TYPE_P2=2
```

Reboot - mstconfig port type settings will only take effect after a reboot.

Set persistent interface names

After reboot, (even without connecting the cables), you should see two new interfaces listed by `ifconfig` command. For example:

```
ifconfig
```

Output should look like:

```
ens5p1: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 00:02:c9:3e:ca:b0  txqueuelen 10000  (Ethernet)
        RX packets 2142  bytes 248089 (248.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6121  bytes 5923956 (5.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens5: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 00:02:c9:3e:ca:b1  txqueuelen 10000  (Ethernet)
        RX packets 2142  bytes 248089 (248.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6121  bytes 5923956 (5.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The `ens5` and `ens5p1` interface names may be different

Edit `/etc/udev/rules.d/70-persistent-net.rules` and add the following lines

```
# Mellanox ConnectX-3 HP 649281-B21 IB FDR/EN 10/40Gb 2P 544QSFP Adapter 656089-001
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:02:c9:3e:ca:b1", ATTR{dev_id}=="0x0", ATTR{type}=="1", NAME="
```

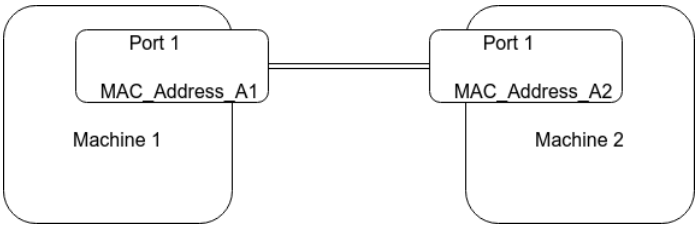
```
# Mellanox ConnectX-3 HP 649281-B21 IB FDR/EN 10/40Gb 2P 544QSFP Adapter 656089-001
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:02:c9:3e:ca:b0", ATTR{dev_id}=="0x0", ATTR{type}=="1", NAME="
```

IMPORTANT: Change the **`ATTR{address}=="00:02:c9:3e:ca:b1"`** and **`ATTR{address}=="00:02:c9:3e:ca:b0"`** to reflect your actual MAC addresses. Leave the names as **`eth40a`** and **`eth40b`**.

If you have only one port, you will only add one (uncommented) line (`eth40a`)

Reboot again to let the persistent names take effect.

Using a pair of SINGLE port Mellanox VPI 40Gbit NICs



If you have a single-port 40GBit/sec NIC or have only a single cable, you can only use one port on each end (at least at a time). So in such cases, bonding does not make sense.

Setup interface

Edit `/etc/network/interfaces/eth40a` to contain:

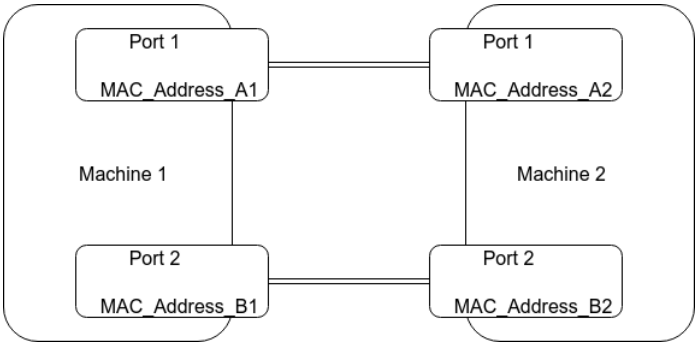
```
allow-hotplug eth40a
iface eth40g inet static
    hwaddress 00:02:c9:3e:ca:b0
    address 10.30.0.2
    netmask 255.255.255.0
    broadcast 10.30.0.255
```

Replace **10.30.0.2** with static IP address of MachineA and MachineB respectively. Replace **netmask 255.255.255.0** and **broadcast 10.30.0.255** based on your IP addresses

Bring up interface

```
ifdown eth40a; ifup eth40a
```

Using both ports of a pair of Mellanox VPI 40Gbit NIC



Enable bonding

- Add `bonding` to `/etc/modules` - load `bonding` module on reboot
- `modprobe bonding` - until next reboot

Setup interfaces

Edit `/etc/network/interfaces/bond0` to contain:

```
allow-hotplug eth40a
iface eth40a inet manual
    bond-mode active-backup
    bond-master bond0
    pre-up ifconfig eth40a txqueuelen 10000 2>/dev/null || true

allow-hotplug eth40b
iface eth40b inet manual
    bond-mode active-backup
```

```
    bond-master bond0
    pre-up ifconfig eth40b txqueuelen 10000 2>/dev/null || true

allow-hotplug bond0
iface bond0 inet static
    bond-mode balance-rr
    use_carrier 1
    bond-slaves eth40a eth40b
    bond-miimon 100
    bond-downdelay 200
    bond-updelay 200
        address 10.30.0.2
        netmask 255.255.0.0
        broadcast 10.30.255.255
    pre-down ifconfig eth40a down 2>/dev/null || true
    pre-down ifconfig eth40b down 2>/dev/null || true
    pre-up ifconfig eth40a up 2>/dev/null || true
    pre-up ifconfig eth40b up 2>/dev/null || true
```

Note that IP address, netmask and broadcast address are **ONLY** associated with the `bond0` address.

Bring up interfaces

```
ifdown bond0 2</dev/null
ifdown eth40a 2>/dev/null
ifdown eth40b 2>/dev/null
ifup bond0 &
ifup eth40a; ifup eth40b
```

Check bonding status

```
cat /proc/net/bonding/bond0
```

Output should look like:

```
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: load balancing (round-robin)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 200
Down Delay (ms): 200
Peer Notification Delay (ms): 0

Slave Interface: eth40b
MII Status: up
Speed: 40000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 00:02:c9:3e:ca:b0
Slave queue ID: 0
```

Quick Start - Infiniband mode

Do the following steps on **BOTH** machines connected by 40Gbit Infiniband cable.

Package installation

```
apt install rdma-core opensm ibutils infiniband-diags
```

Setup interface

Create new file under `/etc/network/interfaces.d` - e.g. named `ib0_4g` (the filename is not important) containing:

```
allow-hotplug ib0
iface ib0 inet static
    address 10.30.0.1
    netmask 255.255.255.0
    broadcast 10.30.0.255
```

Change the contents of the file to reflect your network (IP address, netmask, broadcast address). Do **NOT** change the interface name (ib0).

Connect cable between computers

Do this **AFTER** performing above steps on **BOTH** computers.

Reboot

Your interface (ib0) should be seen, with the right IP address, netmask etc.

If not, use the command `ifconfig -a` to see whether interface `ib0` was detected at all.

Troubleshooting steps

- Use `ibstat` to see status of interface(s)
- `cat /sys/class/net/ib0/mode` - should show `connected` or `datagram`
- Check whether `opensm` service is running: `systemctl status opensm`. Output should resemble the following:

```
● opensm.service - LSB: Start opensm subnet manager.
   Loaded: loaded (/etc/init.d/opensm; generated)
   Active: active (running) since Mon 2019-12-30 00:07:32 PST; 1 day 10h ago
     Docs: man:systemd-sysv-generator(8)
  Process: 3060 ExecStart=/etc/init.d/opensm start (code=exited, status=0/SUCCESS)
    Tasks: 39 (limit: 23347)
   CGroup: /system.slice/opensm.service
           └─3110 /usr/sbin/opensm -g 0x0002c903003f02f2 -f /var/log/opensm.0x0002c903003f02f2.log

Dec 30 00:07:32 fili opensm[3060]: Starting opensm on 0x0002c903003f02f2:
Dec 30 00:07:32 fili systemd[1]: Started LSB: Start opensm subnet manager..
Dec 30 00:07:32 fili OpenSM[3110]: /var/log/opensm.0x0002c903003f02f2.log log file opened
Dec 30 00:07:32 fili OpenSM[3107]: /var/log/opensm.0x0202c9fffe3f02f0.log log file opened
Dec 30 00:07:32 fili OpenSM[3107]: OpenSM 3.3.20
Dec 30 00:07:32 fili OpenSM[3110]: OpenSM 3.3.20
Dec 30 00:07:32 fili OpenSM[3107]: Entering DISCOVERING state
Dec 30 00:07:32 fili OpenSM[3110]: Entering DISCOVERING state
Dec 30 00:07:32 fili OpenSM[3107]: Exiting SM
Dec 30 00:07:32 fili OpenSM[3110]: Entering STANDBY state
```

Checking performance

- On both machines, A and B connected by 40Gbit Infiniband cable, having IP addresses IP_A and IP_B on interface `ib0`respectively, install iperf3: `sudo apt install iperf3`
- On machine A start iperf3 in **server** mode: `iperf3 -B IP_A -i 3 -s` - replace IP_A with IP address of interface `ib0` on Machine A (sudo / root **not** required)
- On machine B start iperf3 in **client** mode: `iperf3 -B IP_B -i 3 -t 15 -s IP_A` - replace IP_A with IP address of interface `ib0` on Machine A and replace IP_B with IP address of interface `ib0` on machine B (sudo / root **not** required)

Output on machine B should look like:

```
Connecting to host 10.30.0.1, port 5201
[  4] local 10.30.0.2 port 51913 connected to 10.30.0.1 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-3.00   sec  8.24 GBytes  23.6 Gbits/sec    0   1.81 MBytes
[  4]   3.00-6.00   sec  8.50 GBytes  24.3 Gbits/sec    0   2.81 MBytes
[  4]   6.00-9.00   sec  9.58 GBytes  27.4 Gbits/sec    0   2.81 MBytes
[  4]   9.00-12.00  sec  8.26 GBytes  23.6 Gbits/sec    0   2.81 MBytes
[  4]  12.00-15.00  sec  8.26 GBytes  23.6 Gbits/sec    0   2.81 MBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-15.00  sec  42.8 GBytes  24.5 Gbits/sec    0             sender
[  4]   0.00-15.00  sec  42.8 GBytes  24.5 Gbits/sec                  receiver

iperf Done.
```

Troubleshooting and improving performance

Identify your card

```
lspci | grep Mellanox
```

Output will look like

```
81:00.0 Network controller: Mellanox Technologies MT27500 Family [ConnectX-3]
```

- **`81:00.0`** is *domain-bus-device* number
- **MT27500** is Mellanox part number
- **ConnectX-3** indicates card **supports** PCI-Express 3 - actual PCI-Express version also depends on PCI-Express capabilities of your motherboard

Get additional details on your card

Use *domain-bus-device* number obtained above

```
lspci -v -s 81:00.0
```

Output will look like:

```
81:00.0 Network controller: Mellanox Technologies MT27500 Family [ConnectX-3]
        Subsystem: Hewlett-Packard Company InfiniBand FDR/EN 10/40Gb Dual Port 544QSFP Adapter
        Physical Slot: 5
        Flags: bus master, fast devsel, latency 0, IRQ 47, NUMA node 1
        Memory at ec100000 (64-bit, non-prefetchable) [size=1M]
```

```
        Memory at 3800fe000000 (64-bit, prefetchable) [size=32M]
        Expansion ROM at ec000000 [disabled] [size=1M]
        Capabilities: <access denied>
        Kernel driver in use: mlx4_core
        Kernel modules: mlx4_core
```

Subsystem: Hewlett-Packard Company InfiniBand FDR/EN 10/40Gb Dual Port 544QSFP Adapter gives further OEM details

Get current PCI-Express version and width used

Use *domain-bus-device* number obtained above

```
sudo lspci -vv -s 81:00.0 | grep LnkSta:
```

Output will look like:

```
LnkSta: Speed 8GT/s, Width x8, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
```

- **Speed 8GT/s** : 8 GT/s indicates PCI-Express version 3.x is being currently used for that slot
- **Width x8** : indicates logical width is x8 (8 lanes)

PCI-Express speeds and maximum possible bandwidth for network link

PCI-E version	Per lane GT/sec	Per lane MBytes/sec
1.x	2.5 GT/sec	250 MBytes/sec
2.x	5 GT/sec	500 MBytes/sec
3.x	8 GT/sec	985 MBytes/sec
4.x	16 GT/sec	1.97 GBytes/sec

PCI-E Version	Per lane GT/sec	Physical	Logical	Bandwidth MBytes/sec
2.x	5 GT/sec	x8	x1	250 MBytes/sec
2.x	5 GT/sec	x8	x4	1 GBytes/sec
2.x	5 GT/sec	x8	x8	2 GBytes/sec
2.x	5 GT/sec	x16	x1	250 MBytes/sec
2.x	5 GT/sec	x16	x4	1 GBytes/sec
2.x	5 GT/sec	x16	x8	2 GBytes/sec
3.x	8 GT/sec	x8	x1	985 MBytes/sec
3.x	8 GT/sec	x8	x4	3.94 GBytes/sec
3.x	8 GT/sec	x8	x8	7.88 GBytes/sec
3.x	8 GT/sec	x16	x1	985 MBytes/sec
3.x	8 GT/sec	x16	x4	3.94 GBytes/sec
3.x	8 GT/sec	x16	x8	7.88 GBytes/sec

Notes:

- Physical width will never be **smaller** than physical width of PCI-Express device (x8 in this case)
- Logical width will never be **larger** than physical width
- Logical width will never be **larger** than actual width of PCI-Express device lane width (x8 in this case)

Limiting factors for maximum bandwidth of network link

- PCI-Express version
- Logical slot width - may depend on configurable settings in the BIOS for your motherboard
- Maximum bandwidth for network link will be **LESSER** of maximum bandwidth for each of the connected machines as explored above

sysctl settings for TCP/IP stack

Put `etc/sysctl.d/60-infiniband.conf` under `/etc/sysctl.d` and **reboot**

Contents of `etc/sysctl.d/60-infiniband.conf`

```
# For Mellanox MT27500 ConnextX-3 (HP InfiniBand FDR/EN 10/40Gb Dual Port 544QSFP)
# Settings from https://furneaux.ca/wiki/IPoIB#Kernel_Tuning
# Originally settings from Mellanox:
#   https://community.mellanox.com/s/article/linux-sysctl-tuning
net.ipv4.tcp_timestamps=0
net.ipv4.tcp_sack=1
net.core.netdev_max_backlog=250000
net.core.rmem_max=4194304
net.core.wmem_max=4194304
net.core.rmem_default=4194304
net.core.wmem_default=4194304
net.core.optmem_max=4194304
net.ipv4.tcp_low_latency=1
net.ipv4.tcp_adv_win_scale=1
```

```
net.ipv4.tcp_rmem=4096 87380 4194304
net.ipv4.tcp_wmem=4096 65536 4194304
```

Interface connected state and MTU

Only applicable to using Infiniband mode

- Put `etc/systemd/system/setup_ib0.service` under `etc/systemd/system/`
- Run `systemctl enable setup_ib0.service` and **reboot**

Contents of `etc/systemd/system/setup_ib0.service`

```
[Unit]
Description=Setup ib0
After=sys-subsystem-net-devices-ib0.device

[Service]
Type=oneshot
ExecStart=/bin/echo connected > /sys/class/net/ib0/mode
ExecStart=/sbin/ifconfig ib0 mtu 65520
```

Run iperf3 with larger number of threads (software bottleneck in iperf3)

- On machine A start iperf3 in **server** mode: `iperf3 -B IP_A -i 3 -P2 -s` - replace IP_A with IP address of interface `ib0` on Machine A (sudo / root **not** required)
- On machine B start iperf3 in **client** mode: `iperf3 -B IP_B -i 3 -t 15 -P2 -s IP_A` - replace IP_A with IP address of interface `ib0` on Machine A and replace IP_B with IP address of interface `ib0` on machine B (sudo / root **not** required)

On each side, try `-P2`, `-P4` and `-P8` to see what extracts the maximum bandwidth from the link. For me I got the maximum with `-P4`.

Run multiple instances of iperf3

On each machine start iperf3 (in server and client modes respectively) adding the **`-p <port_num>`** option to choose a port different from the iperf3 default **5201**