Linux installation guide

Delivery

The package contains all the resources needed to run the IP virtual card.

Once this package is installed, you will be able to run any application linked to the IP Virtual Card solution.

Prerequisites

> To be able to install the IP Virtual Card, you must first install some prerequisites.

Red Hat Enterprise Linux

```
sudo dnf -y install gcc-c++ make git
```

Ubuntu

```
sudo apt-get -y update
sudo apt-get -y install g++ make git
```

Installation

The installation is automated by the script `install.sh`.

This script must be launched with root privilege :

```
sudo sh install.sh
```

The script will :

- check needed resources availability
- disable NTP
- install and register `ptp4l` as a service
- install the VideoMasterIP libraries
- install and register the VirtualCardService as a service
- install licensing module

The script can also be launched with the following argument :

- `clean` : this will remove everything that has been done during installation
- `update` : this will update the existing libraries and VirtualCardService executable without reinstalling all the needed packages like ‘ptp4l’ etc.

KBDPDK

> The installation of KBDPDK is only required when using VCS in DPDK mode.

Redist and dev packages contain KBDPDK package in their library/external folder. To activate DPDK, the KBDPDK package must be installed separately.

The first thing to do before installing KBDPDK is to check the compatibility with your system. The following script is located in `kbdpdk`

```
 sudo python kbcheck_compat.p
y
```

This script requires python (v2.7 or v3.6 and higher) to be installed on the system.

Prerequisites

> To be able to use KBDPDK, you must first install some prerequisites.

First, run the commands concerning your OS, then run the common part.

Red Hat Enterprise Linux 8

```
# Activate codeready-builder repo for RHEL 8
sudo subscription-manager repos --enable codeready-builder-for-rhel-8-x86_64-rpms
# Install package group "Development Tools
"
sudo dnf groupinstall -y "Development Tools"
# Install other required packages
sudo dnf install -y numactl-devel meson ninja-build python3-pip python3 pkg-config
```

Red Hat Enterprise Linux 9

```
# Activate codeready-builder repo for RHEL 9
sudo subscription-manager repos --enable codeready-builder-for-rhel-9-x86_64-rpms
# Install package group "Development Tools
"
sudo dnf groupinstall -y "Development Tools"
# Install other required packages
sudo dnf install -y numactl-devel meson ninja-build python3-pip python3 pkg-config
```

Ubuntu

```
# Install required packages
sudo apt-get install -y build-essential libnuma-dev meson ninja-build \
 python3-pip python3 pkg-config
```

Common

```
# Install pip packages
sudo pip3 install pyelftools distr
o
```

Mellanox NIC

> This section is intended only for mellanox NIC.

Requirements

- Network card :
  - ConnectX-5 :
    - Firmware: **16.21.1000** and above.
  - ConnectX-6 :
    - Firmware: **20.27.0090** and above.
  - ConnectX-6 Lx
    - Firmware: **20.27.0090** and above.
  - ConnectX-6 Dx
    - Firmware: **20.27.0090** and above.
  - BlueField-2
    - Firmware: **18.25.1010** and above.
- Minimal kernel version :
  - v4.14

Driver install

1. Download the latest version of MLNX_OFED for your linux distribution : https://network.nvidia.com/products/infiniband-drivers/linux/mlnx_ofed/
2. Extract it on the target machine
3. Install the required libraries by installing Mellanox OFED/EN:

```
sudo ./mlnxofedinstall --upstream-libs --dpd
k
```

4. Verify firmware version

```
ibv_devinfo
```

5. Restart driver

```
sudo /etc/init.d/openibd restart
#OR
sudo service openibd restart
```

Enable traffic shaping narrow linear profile

Requirement:

- Hardware : ConnectX-6 Dx and higher
- OFED Version: 5.1-2 and higher
- Firmware Version : 22.28.2006 and higher

Procedure:

1. Download the latest MFT (Mellanox Firmware Tools) tool package : https://network.nvidia.com/products/adapter-software/firmware-tools/

2. Extract it on the target machine
3. Install MFT tools

```
#In mft tool extracted folder
sudo ./install.sh
```

4. Get the pci address of your nics

```
lspci | grep Mellanox
01:00.0 Ethernet controller: Mellanox Technologies MT28841
01:00.1 Ethernet controller: Mellanox Technologies MT28841
```

5. For each NIC, enable firmware functionality

```
sudo mlxconfig -d [PCI_ADDR] s REAL_TIME_CLOCK_ENABLE=
1
sudo mlxconfig -d [PCI_ADDR] s ACCURATE_TX_SCHEDULER=1
#For example for nic 01:00.0
sudo mlxconfig -d 01:00.0 s REAL_TIME_CLOCK_ENABLE=1
sudo mlxconfig -d 01:00.0 s ACCURATE_TX_SCHEDULER=1
```

6. Restart your machine

KBAPI Installation

The installation is automated by the script `install.sh` located in `kbdpdk`.

The vcs service **must** be shut down before this step.

This script must be launched with root privilege :

- **With** Mellanox NIC

  ```
  sudo sh install.sh mlnx
  5
  ```

- **Without** Mellanox NIC

  ```
  sudo sh install.sh
  ```

The script will :

- Compile DPDK
- Install the kbdpdk utility scripts
- Install the kbapi library.

KBAPI Configuration

The kbapi is configured via python scripts. Those scripts are accessible on the system in `/usr/local/sbin`.

To perform the configuration run the following script:

```
 sudo kbconfig_setup.p
y
```

This script is an interactive script that helps you to configure the kbapi:

- Configure the NIC used by the kbapi
- Configure the hugepages
- Configure the cpu core isolation

A reboot is necessary after executing this script and applying the configuration.

After reboot run the script that allows you to check that the kbapi is properly configured:

```
sudo kbcheck_config.py
```

KBAPI Clean

This script cleans the kbapi configuration on your system:

```
sudo kbconfig_cleanup.p
y
```

VCS configuration

Packag installation copies two vcs config files:

- vcsconfig.xml that contains recommended parameters for socket mode
- vcsconfig_dpdk.xml that contains recommended parameters for DPDK mode

Note that vcs reads its configuration from vcsconfig.xml file. So you need to modify vcsconfig.xml according to the vcsconfig_dpdk.xml content.

Target platform

Hardware :

- CPU speed : minimum 2.1GHz
- CPU architecture : 64 bits
- NIC bandwidth : minimum 10Gb/s

OS :

- Ubuntu 20.04 64 bits
- NIC driver must be properly installed and up to date

Performance considerations

Socket

Redist package installs a script that is launched at each startup and that improves the socket performance.
You can find that script here : `/opt/deltacast/videomasterip/vcs/network_sysctl.sh`

Explanations :
`sysctl -w net.core.rmem_max=33554432`
This sets the max OS receive buffer size for all types of connections.

`sysctl -w net.core.wmem_max=33554432`
This sets the max OS send buffer size for all types of connections.

`sysctl -w net.core.rmem_default=65536`
This sets the default OS receive buffer size for all types of connections.

`sysctl -w net.core.wmem_default=65536`
This sets the default OS send buffer size for all types of connections.

`sysctl -w net.ipv4.route.flush=1`
This will ensure that immediately subsequent connections use these values.

BIOS

According to our observations, C-states, P-states or any energy-saving parameters must be disabled in the BIOS parameter.

This ensures that the computer is running at its peak performances.

Not following the recommendations can lead to unstable or non-compliant streams.

CPU governor

We strongly recommand to disable `ondemand` CPU scaling daemon to ensure best performances:
`sudo systemctl disable ondemand`

To avoid reboot, you can manually change the actual scaling gorvernor:
`echo performance | sudo tee /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor`

Conductor configuration

The CPU core associated to a conductor must not be used by any process.
To avoid kernel interruption on this core, it can be isolated at boot via the option `isolcpus` or with the KBAPI configuration (in case of DPDK mode).

If the hyper-threading is activated, the same guideline must be applied to the associated logical core.

Route

In order to improve the performances during a TX stream emitting toward a multicast address, it is advised to set a general multicast route associated with the used NIC.

This can be done thanks to the following command :
`sudo route add -net 224.0.0.0/4 dev xxxx`,
where ‘xxxx’ is the name of the NIC.

Without this route, the packet interval time (PIT) will not be stable and this can cause non-compliance to the ST2110-21 packet pacing standard (overflow in the VRX bucket).

Licensing

The IP Virtual Card solution is secured by a license manager called **dlmcli**.

To identify the platform, dlmcli take several parameters into account. One of those parameters is one of the NIC MAC. To avoid license issue in case of network configuration changes, we recommend to force dlmcli to use a specific NIC MAC that should never be removed or used by IP Virtual Card (mainboard NIC by example). For that, use the following argument while you add the first license: `--select-custom-mac ############`

If a custom mac is not provided with the first license entry, dlmcli will warn you and list all the available NIC mac.

To add a license in online mode , use the following command: `dlmcli activate ####-####-####-####-####-####-####-#### [--select-custom-mac ############]`

To add a license in offline mode , use the following command: `dlmcli activate --offline requestfile.bin ####-####-####-####-####-####-####-#### [--select-custom-mac ############]`
Provide the processed requestfile.bin to Deltacast. In return, Deltacast will provide you a response file. Then use the following command: `dlmcli process responsefile.bin`

** To update the licensing informations ** in VCS, without having to restart it, call the `VMIP_RefreshLicensing()` functions.
You can also compile and run the `sample_refresh_licensing`.

Synchronize multiple NICs

When using multiple NICs, if **traffic shaping narrow linear** is enabled, You **MUST** synchronize all the NICs on the interface synchronized with PTP4L.

To synchronize the NICs, a python utility is available in the VCS folder:

This script must be run with admin rights.

The script creates all the necessary phc2sys services.

```
/opt/deltacast/videomasterip/vcs/config_nic_sync.py
USAGE:
config_nic_sync.py [ptp_domain] [main_interface] [follower_interface] ... [follower_interface] # Create and loac


config_nic_sync.py clean    # Clean previous configuratio
n
config_nic_sync.py start    # Start all sync service
s
config_nic_sync.py restart  # Restart all sync service
s
config_nic_sync.py stop     # Stop all sync service
s
```

Example of setup:

```
/opt/deltacast/videomasterip/vcs/config_nic_sync.py 127 eno1 eno2 eno3 eno4
```

Virtual machine support

Configuration

The virtual machine support is only available with network cards configured in PCI passthrough. Socket and DPDK mode are both available in virtual machine.

PTP

Firewall

In order to make PTP work, you have to be sure that the ptp packets aren’t blocked by any kind of firewall installed on your machine. If it is the case, you must add a firewall rule to allow those packets or PTP won’t work.