Windows 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

Target platform

Hardware :

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

OS :

- Windows Server 2019, Windows Server 2022, Windows 10 and Windows 11
- NIC driver must be properly installed and up to date

Software

In order to intall IP Virtual Card, Python (v3.8 and higher) is required.

Installation

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

This script must be launched with administrator rights :

```
python install.py
```

The script will :

- install and register DELTA PTP as a service

- install and register the VirtualCardService as a service
- install VideoMasterIP libraries
- install licensing module
- create a rule in the firewall to allow VirtualCardService network communication.

The script can also be launched with the following arguments :

- The installation mode (those options are mutually exclusive):
  - `--install` : this will install the package with all the needed dependencies. This is the default option.
  - `--clean` : this will remove everything that has been done during installation
- Optional argument:
  - `--dpdk` : this will install the KBDPDK package to use VCS in DPDK mode. See KBDPDK section for more details.

KBDPDK

The KBDPDK is installed when installing IPVC if the installation script is called with the option `-k, --kbdpdk`.

Supported NIC

At the moment, only **Intel** NIC's are supported by the KBDPDK integration.

In partical, extended tests have been done with Intel XXV710 and Intel E810 NIC's.

NIC binding

In order to use a NIC with the KBDPDK, the NIC needs to be bound to DPDK.

This is done by installing the netuio driver on the desired NIC's.

To do so, follow the following step:

- Open the Device Manager (devmgmt.msc);
- In the "Network Adapter" section, right-click on the desired NIC;
- Select "Update Driver", then "Browse my computer for drivers" and "Let me pick from a list...";
- on the next screen, click "Have disk" and browse to the following folder : `C:\Program Files\DELTACAST\VCS\X.X.X\netuio`, where `X.X.X` is the IPVC package version;
- In the "Model" box, a driver should be present. Click "Next";
- If the operation was successfull, you should see a new device the "Windows UIO" section of the device manager.

If you wish to use the bound NIC with the official drivers, right-click on it an select "Uninstall Device".
Once this is done, the NIC will disappear from the "Windows UIO" section and reappear in the "Network adapter" section

Note using the Intel E810

To use Intel E810 NIC’s, some additional steps must be performed:

- Download the last driver pack from Intel
- Extract the archive and browse to `DDP_Profiles\810_Series`;
- Extract `ice-X.X.XX.X.zip`;
- Copy the file ice-X.X.XX.X.pkg in the VCS installation folder and rename it `ice.pkg`;
- Update the NIC firmware by running `NVMUpdatePackage\E810\E810_NVMUpdatePackage_vX_XX_Windows\E810_NVMUpdatePackage_vX_XX_Windows.exe`.

VCS configuration

The VCS configuration is done through the `ipvc_configure.py` script. This script requires Python (v3.8 and higher) to be installed on the system. It has to be run using administrator rights.

It is located in `C:\Program Files\DELTACAST\VCS`

The script works in 2 different modes : interactive and silent.

Interactive Mode

In interactive mode, the script will ask you to enter the configuration parameters one by one. After entering all the parameters, the script will ask you if you want to save the configuration in a file. Then it will ask you if you want to apply the configuration to the system. By default, the configuration file is saved in the current directory and is named `ipvc_config.cfg`.

```
python ipvc_configure.py
```

Silent Mode

In silent mode, the script will use the configuration file passed as an argument. Using the following command, the script will apply the configuration to the system:

```
python ipvc_configure.py --config [CONFIG_FILE]
```

Performance considerations

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.

Conductor configuration

The CPU core associated to a conductor must not be used by any process.

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

Licensing

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

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.exe activate ####-####-####-####-####-####-####-#### [--select-custom-mac ############]`

To add a license in offline mode , use the following command:
`.\dlmcli.exe 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.exe 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`.

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.