Service Host Administrator Guide

Version 2.5

Disclaimer

Vizrt provides this publication "as is" without warranty of any kind, either expressed or implied. This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt's policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time. Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

Antivirus Considerations

Vizrt advises customers to use an AV solution that allows for custom exclusions and granular performance tuning to prevent unnecessary interference with our products. If interference is encountered:

- **Real-Time Scanning:** Keep it enabled, but exclude any performance-sensitive operations involving Vizrt-specific folders, files, and processes. For example:
    - C:\Program Files\[Product Name]
    - C:\ProgramData\[Product Name]
    - Any custom directory where [Product Name] stores data, and any specific process related to [Product Name].

- **Risk Acknowledgment:** Excluding certain folders/processes may improve performance, but also create an attack vector.
- **Scan Scheduling:** Run full system scans during off-peak hours.
- **False Positives:** If behavior-based detection flags a false positive, mark that executable as a trusted application.

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Created on

2026/03/12

Contents

1 Introduction

Service Host is a component that abstracts the Windows Service layer. The Service Host operates with Service Host plug-ins that implement the services and functionality, while the Service Host itself manages Windows Service related operations, such as logging, on behalf of all plug-ins.

An example is the **Control Mode** plug-in. By default, it is registered as a Windows service named *ServiceHost.control* upon successful installation. **Control Mode** provides a REST interface and comes with a web-based user interface, and is used to operate the Service Host plug-in instances and installed Viz Engine instances.

The fastest way to get started with Service Host is:

1. Install Service Host, see Install and Uninstall.
2. Launch the web GUI of the Control Mode. After each launch of The Control Mode, a shortcut to the web GUI endpoint is updated under *%ProgramData%/Vizrt/ServiceHost/ServiceHost.control*. Open this shortcut with your favorite browser. A description of this web GUI is found under Control Mode.

1.1 Related Documents

For more information about all of the Vizrt products, visit:

- www.vizrt.com
- Vizrt Documentation Center
- Vizrt Training Center
- Vizrt Forum

1.2 Feedback and Suggestions

We encourage suggestions and feedback about our products and documentation. To give feedback and/or suggestions, please contact your local Vizrt customer support team at www.vizrt.com.

2 Getting Started

Service Host is a component that abstracts the Windows Service layer. The Service Host operates with Service Host plug-ins that implement the services and functionality, while the Service Host itself manages Windows Service related operations, such as logging, on behalf of all plug-ins.

An example is the **Control Mode** plug-in. By default, it is registered as a Windows service named *ServiceHost.control* upon successful installation. **Control Mode** provides a REST interface and comes with a web-based user interface, and is used to operate the Service Host plug-in instances and installed Viz Engine instances.

The fastest way to get started with Service Host is:

1. Install Service Host, see Install and Uninstall.
2. Launch the web GUI of the Control Mode. After each launch of The Control Mode, a shortcut to the web GUI endpoint is updated under *%ProgramData%/vizrt/ServiceHost/ServiceHost.control*. Open this shortcut with your favorite browser. A description of this web GUI is found under Control Mode.

2.1   Security Advisor

After installing the software, we recommend to:

- Only allow authenticated users (**Configure Service Host > Rest Server Config**) or restrict the access to **local access only**.
- Restrict *listen to* to the network interface being used.
- Change the default port (`9000`).
- Verify the log files of Service Host on a regular basis.
- Disable Remote Access for WIBU Codemeter Webadmin and use SSL (**Configuration > Basic > Webadmin**).
- Remove any unneeded services.

> ⚠️ **Note:** Be aware that if more services of, for example, Channel Recorder are created, more network ports are opened.

Service Host is using the following network ports by default:

Service Host		
`9000` (configurable)	TCP	Service Host configuration Web interface
Channel Recorder (Basic Configuration)		
`6180` (configurable)	TCP	Channel Recorder Communication port
`5210` (configurable)	TCP	MVCP protocol port
Other Components		
`427`	TCP, UDP	OpenSLP
`22352`	TCP, UDP	Codemeter

3   Service Host Component

Service Host is a Vizrt component that abstracts the Windows Service layer. The Service Host operates with plug-ins that implement a desired service or functionality. Service Host itself takes care of logging and all the Windows Service stuff.

An example is the Control Mode plug-in, which by default is registered as a Windows service called *ServiceHost.control*. Control Mode provides a REST interface and comes with a web-based user interface. It provides the necessary functionality to manage the Service Host plug-ins and installed Viz Engine instances.

3.1 Install and Uninstall

This page contains information on the following topics:

- Installers
- Installation
    - Command Line Options for Bundle Installer
    - msi Installer
- Upgrade an Existing Installation
- Uninstall
- Control Mode

3.1.1 Installers

The Service Host installer comes in two flavors:

- *ServiceHostBundle-x64*.exe*
- *ServiceHost-x64*.msi*

The main difference is that the bundle installer takes care of upgrading an existing installation and installs the required operating system run-times (if not already present), whereas the *.msi* installer only installs Service Host.
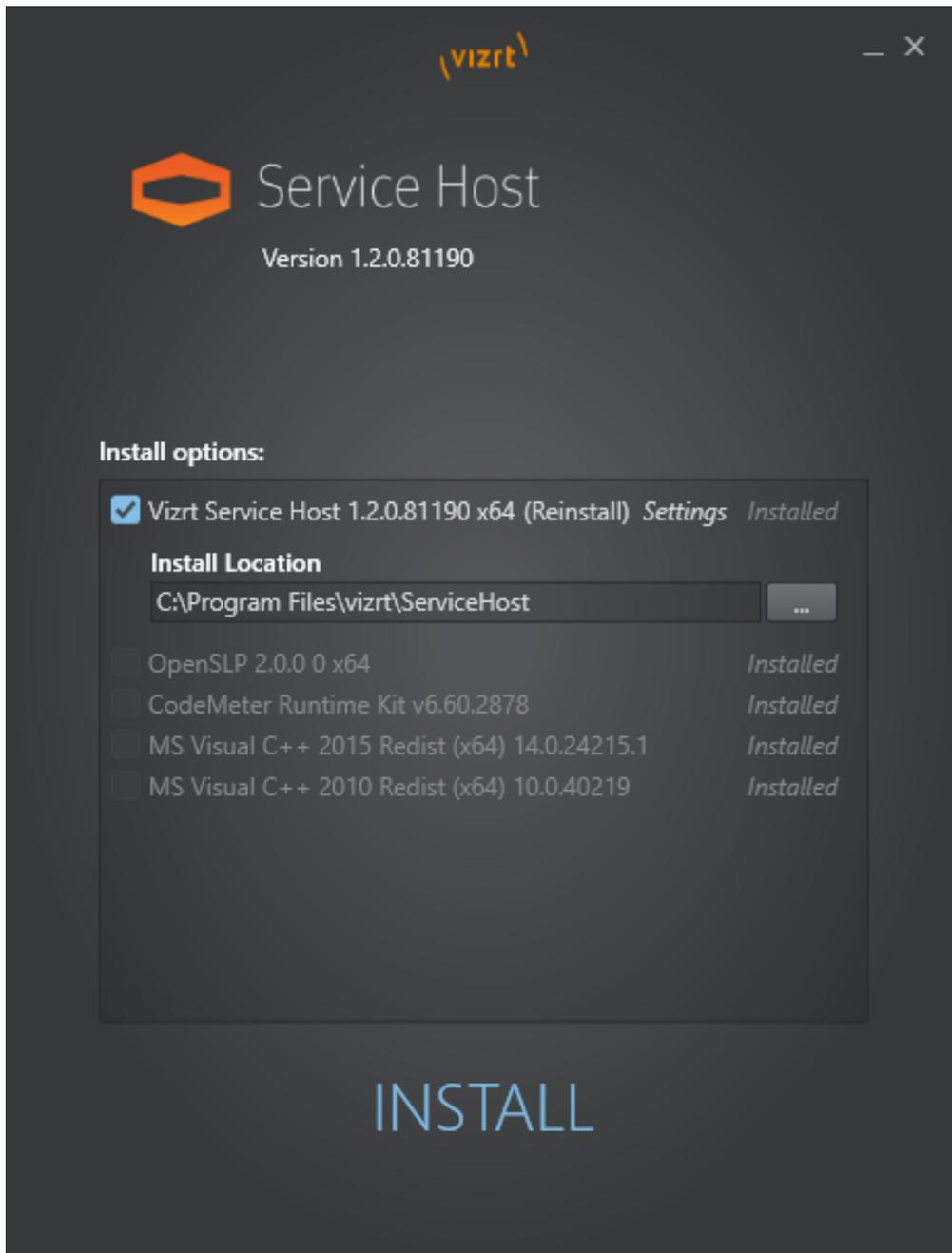
3.1.2   Installation

The bundle installer itself is using the *.msi* installer. The *.msi* and other files can be extracted from the bundle. This is mostly used for silent and unattended installs. Please run the bundle with `-h` command line switch to get all the available options.

Command Line Options for Bundle Installer

```
Z:\my_path> ServiceHostBundle-x64-1.0.0.68324 -h

Z:\my_path>
Running with no parameters will install the product
  -s, --silent              Runs the installer with no user interaction

      --msi                 Extracts all .msi files to a subdirectory
      --dump                Extracts all files to a subdirectory
      --dumpTo=VALUE        Extracts all files to the specified path
  -h, -?, --help            Prints this help
```

msi Installer

```
Z:\my_path> msiexec /package ServiceHost-x64-1.99.0.68094.msi /l*vx! install.log
```

Preconditions

- No Service Host is installed prior to the procedure.

Postconditions

- Service Host is installed at the indicated location. the default value is *%ProgramFiles%/vizrt/ServiceHost*.
- Configuration and log files are located at *%ProgramData%/vizrt/ServiceHost*.
- If the configuration of the Control Mode does not exist, then a default one will be created at *%ProgramData%/vizrt/ServiceHost/ServiceHost.control.xml*.
- The Control Mode is registered as Windows Service and started.
- A shortcut, *ServiceHost.control*, is created in *%ProgramData%/vizrt/ServiceHost* to point to the web GUI of the Control Mode. Use this shortcut to verify the installation.

3.1.3   Upgrade an Existing Installation

Upgrading an existing installation using the *.msi* installer is possible when doing an uninstall of the old package followed by an install of the new package.

Preconditions

- Service Host is installed.
- The *Control Mode* is running.

The uninstallation process writes the current configuration and state into *Reinstall.profile*. This file is then used by the installer of the new package to re-establish the state and configuration.

Postconditions

- The new version of Service Host is installed.
- The existing configuration and state of Service Host services is re-established.

3.1.4   Uninstall

```
Z:\my_path> msiexec /uninstall ServiceHost-x64-1.99.0.68094.msi /l*vx! uninstall.log
```

Preconditions

- Service Host is installed.
- Zero, one or more services may be registered with this Service Host.
- Zero, one or more services may be registered with this Service Host and running.
- The *Control Mode* has to be running. otherwise the uninstall silently fails.

Postconditions

- The configuration and status of services under this Service Host are stored as the config file location in *Reinstall.profile*. They can be used later for further installations.
- Any running services are stopped.
- Any registered services are unregistered.
- The files under *%ProgramFiles%/vizrt/ServiceHost* are removed.

Notes

If the *Control Mode* is unreachable during uninstallation, it is still possible to uninstall. The dialog informs users that the *Control Mode* is unreachable and prompt users to decide to force uninstall, or cancel. If users force uninstall, users have to manually unregister all Service Host services registered as Windows services.
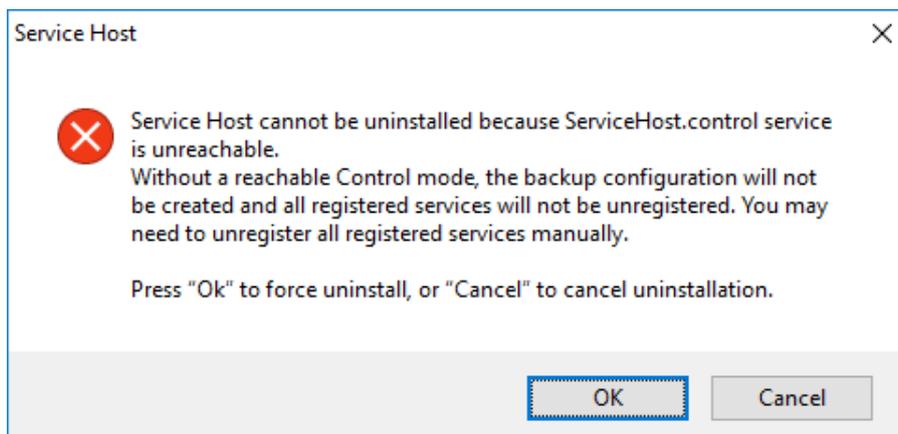
3.1.5   Control Mode

To register/unregister and start/stop Service Host plug-in instances manually, please take a look at Control Mode.

3.2   Control Mode

The *Control Mode* plug-in is registered as a Windows Service when Service Host is installed. This service is the administration point for this Service Host. The shortcut to the *Control Mode* of the current Service Host can be found at *%ProgramData%/vizrt/ServiceHost/ServiceHost.control* . It provides the following functionalities via REST and Command interface.

This page contains information on the following topics:

- Viz Engine
    - Supported Versions
    - Start a Viz Engine Instance
    - Stop a Viz Engine Instance
    - View Viz Engine Configuration
    - View Viz Engine Logs
    - Notes
- Plug-ins
    - Register a Plug-in Instance
    - Configure a Plug-in Instance
    - Plug-in Monitor
    - Start/Stop a Plug-in Instance
    - Unregister a Plug-in Instance
    - Report/Logs
    - Notes
- Service Host (Control Mode)
    - Configuration
    - Debugging of Service Host

On the welcome page, the first thing to do is to log in. The login name is used to log all interactions with the Control Mode. The login name can be viewed on the upper right of the GUI, next to the **Login/Logout** button.
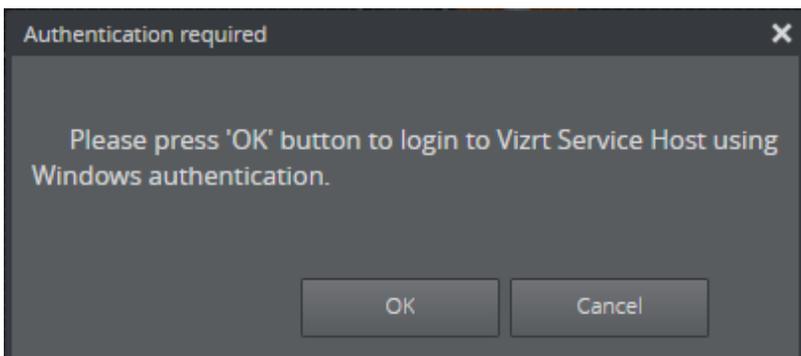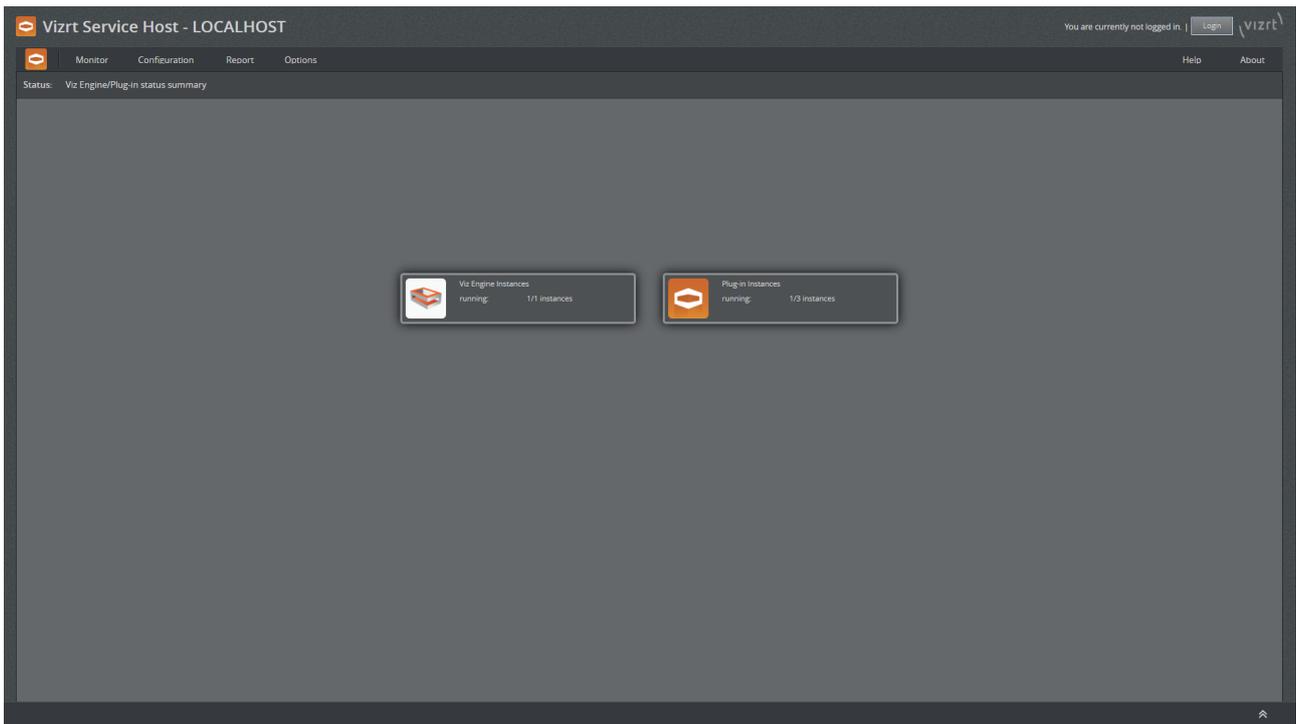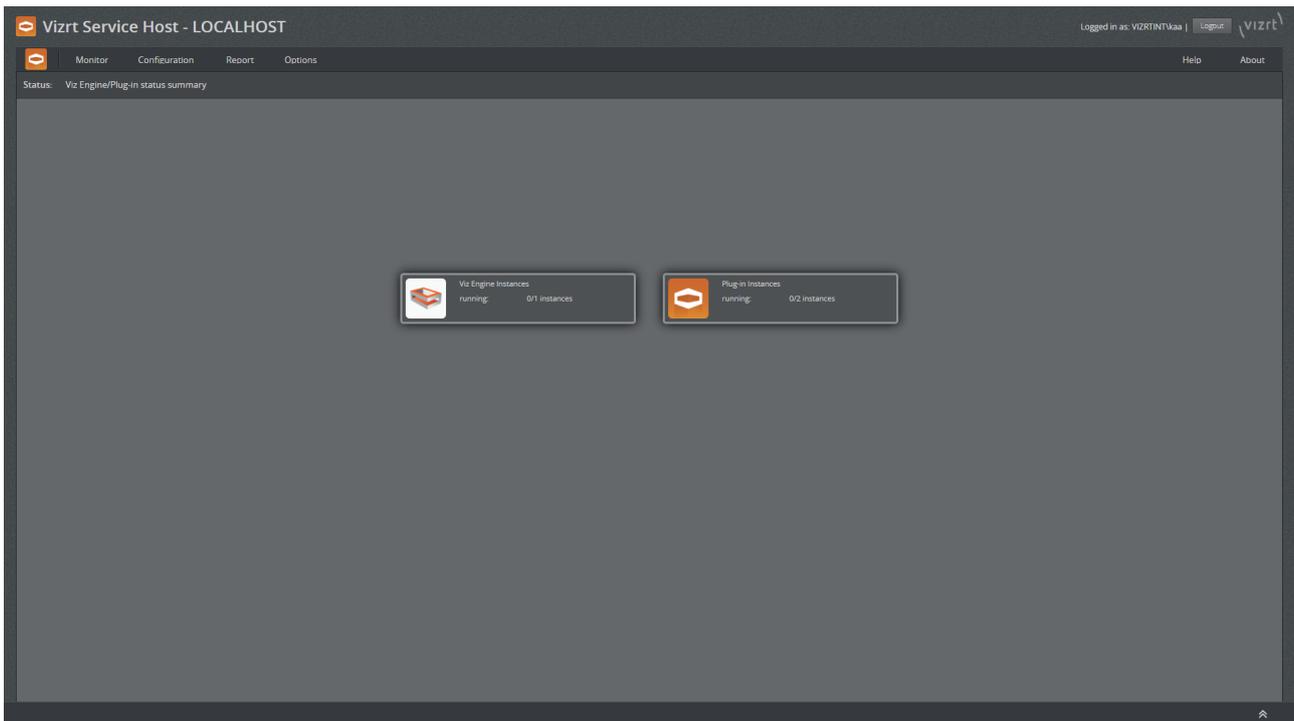
3.2.1   Viz Engine

Supported Versions

	Viz 3.x Support	Viz 4.x Support	Viz 5.x Support
Service Host 2.0 or higher	-	-	✅
Service Host 1.5 or higher	✅	✅	-
Service Host 1.4	-	✅	-
Service Host 1.3 or previous	✅	-	-

The *Control Mode* is aware of Viz Engine instances on the local machine. They can be controlled from here.

Follow the menu **Monitor > Viz Engine.**
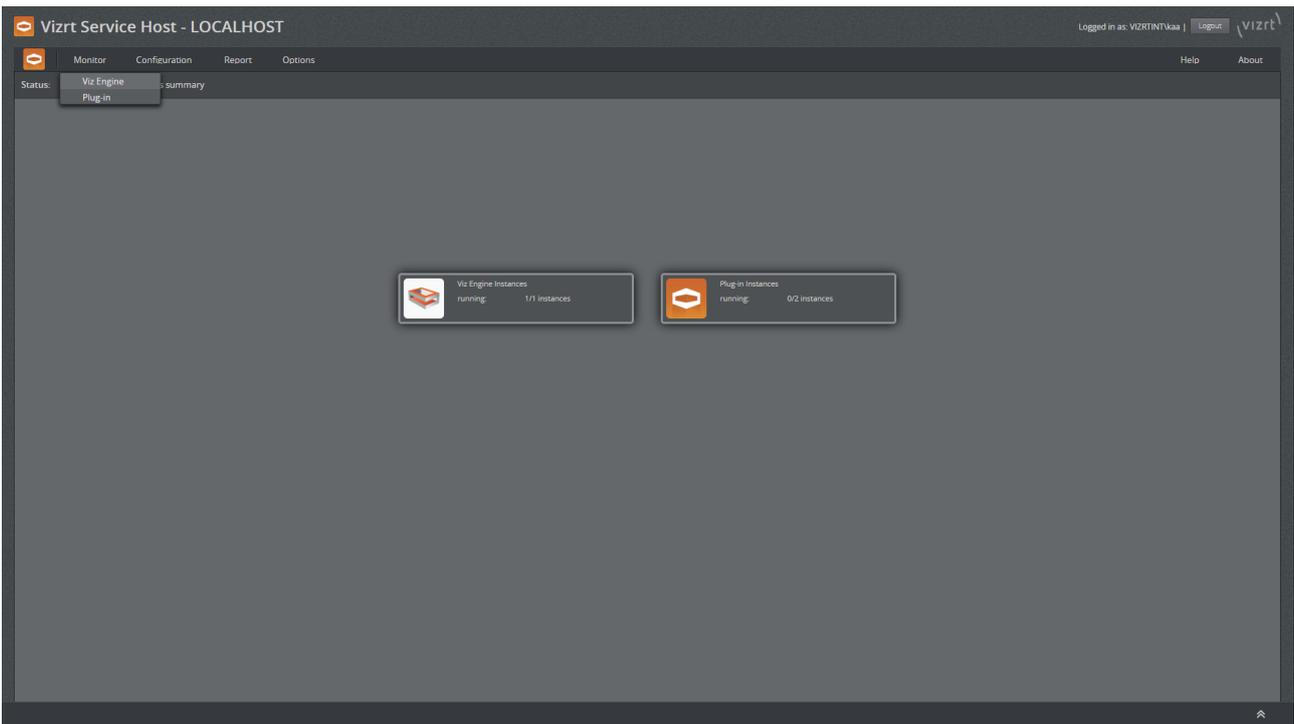
And go to this screen.
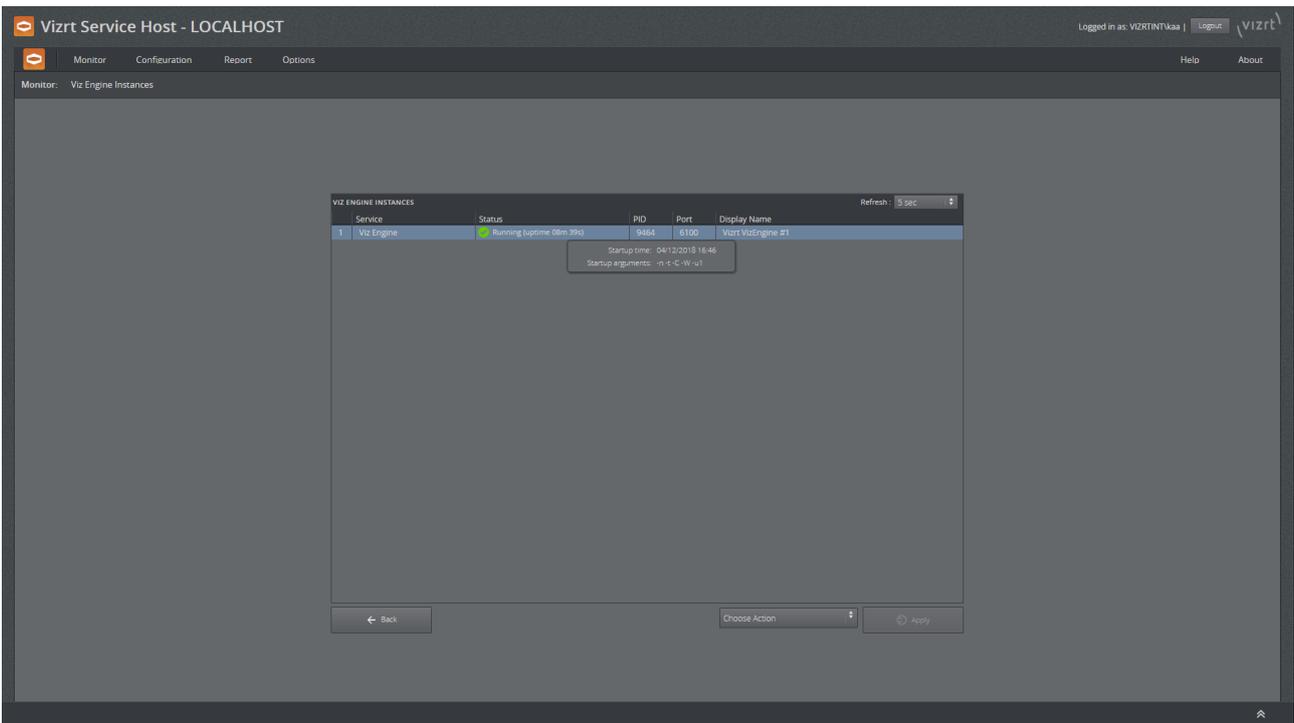
Start a Viz Engine Instance

From the Viz Engine Instances Monitor page, the context menu on an individual Viz Engine instance or use the action bar on the bottom of the screen can be used to start Viz Engine instances.
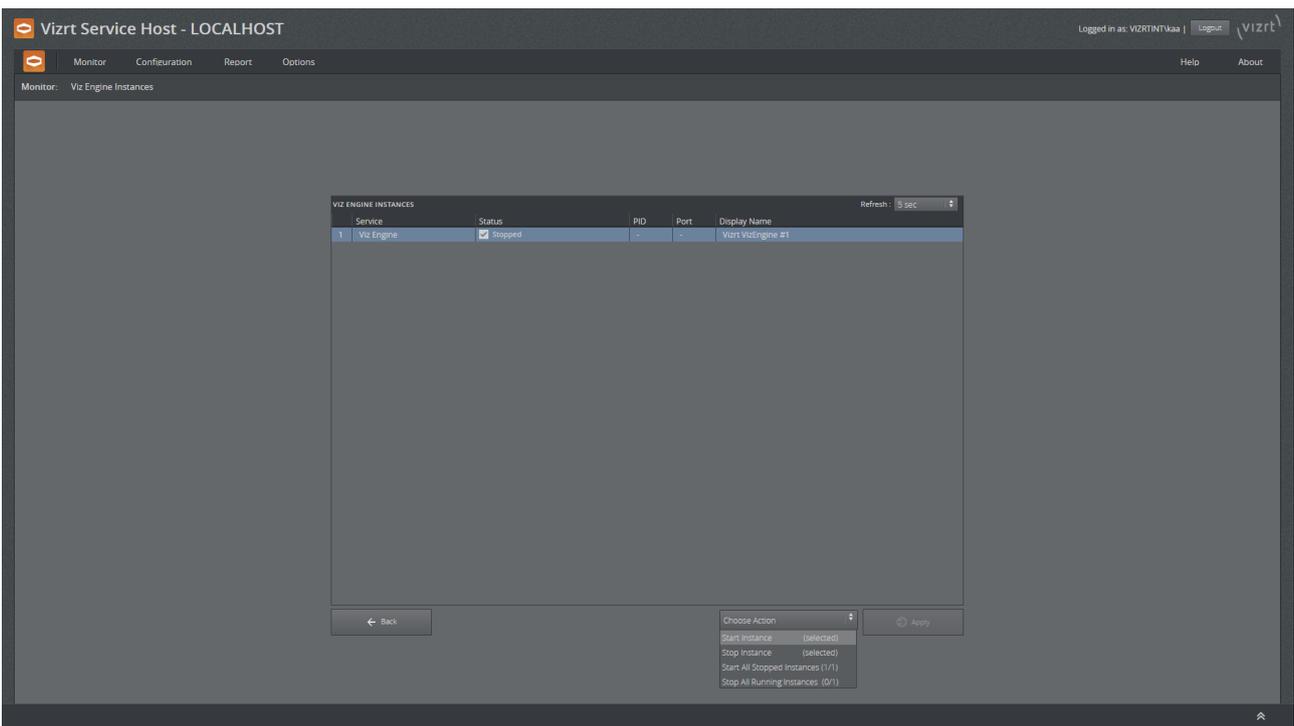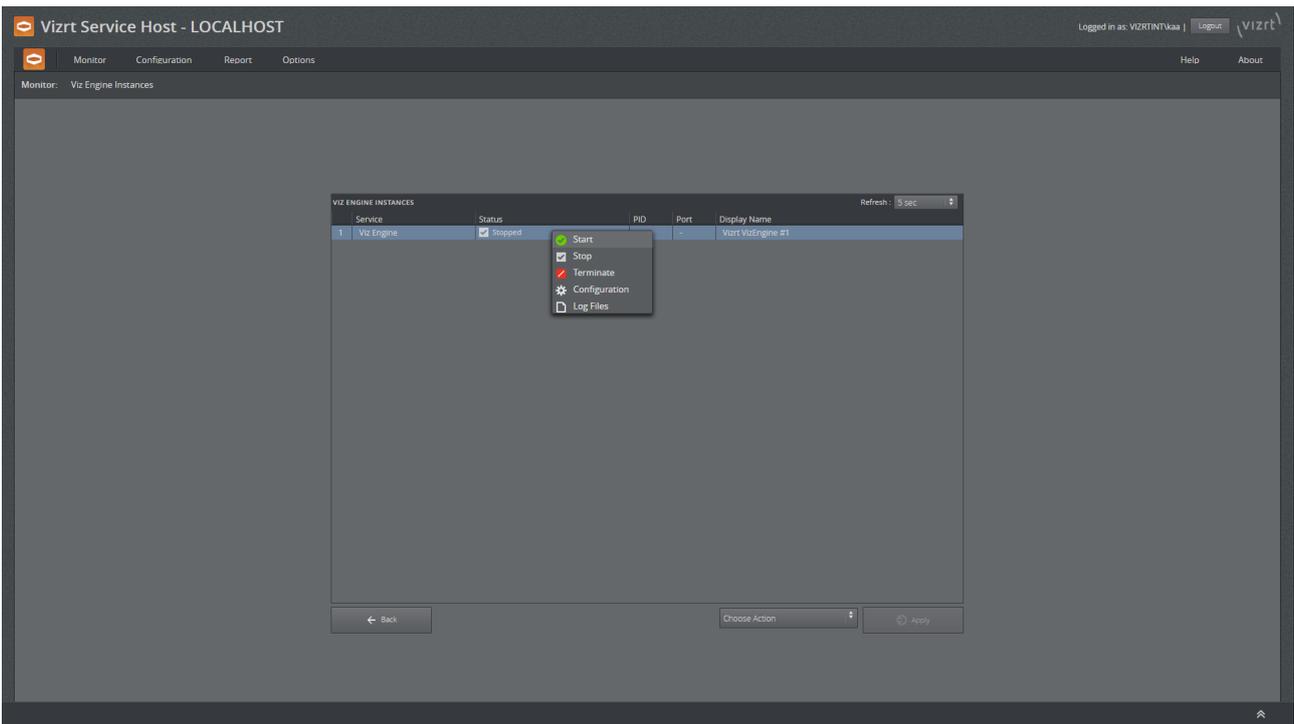
The page is automatically refreshed every five seconds by default.

Preconditions

- Viz Engine needs to be installed at the default location.

Postconditions

- Start: A Viz Engine was launched without GUI and without a console.
- Start: The launch may have failed. Please check the reason in the logs. Context menu Logs or **Menu Report > Logs > Viz Engine**.

Stop a Viz Engine Instance

From the Viz Engine Instances Monitor page, the context menu on an individual Viz Engine instance or use the action bar on the bottom of the screen can be used to stop Viz Engine instances.
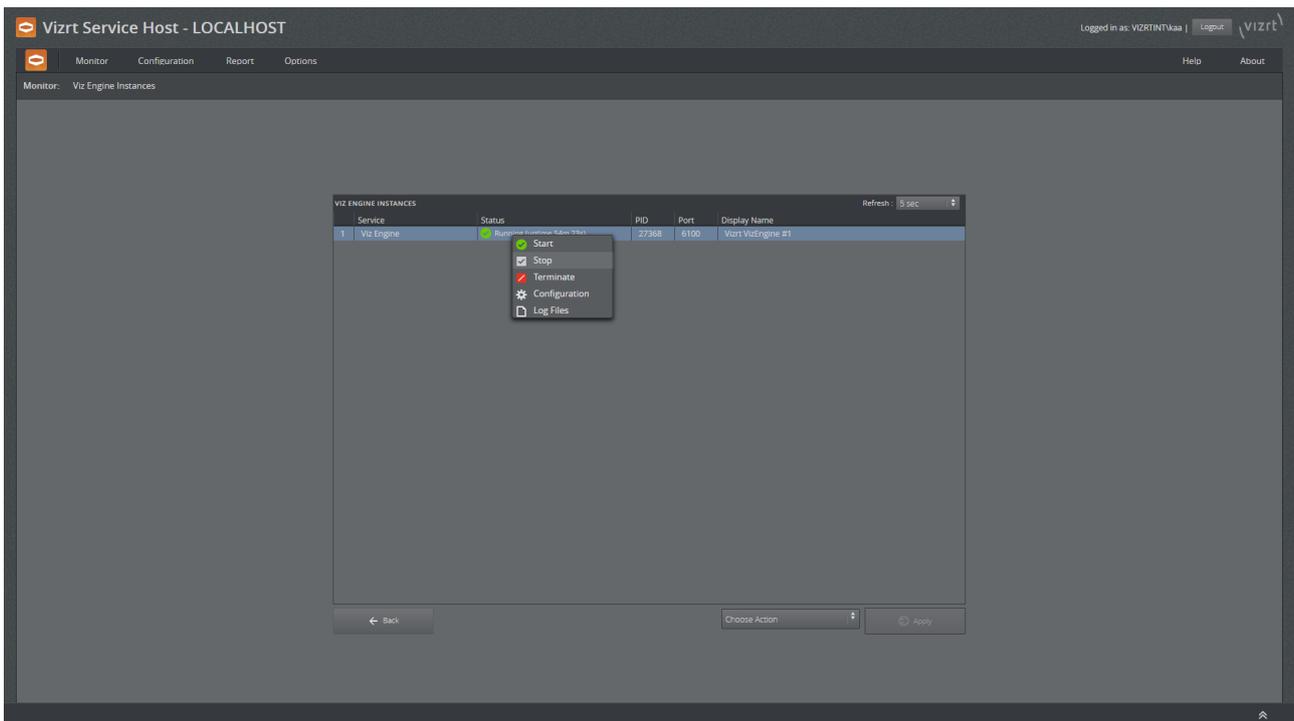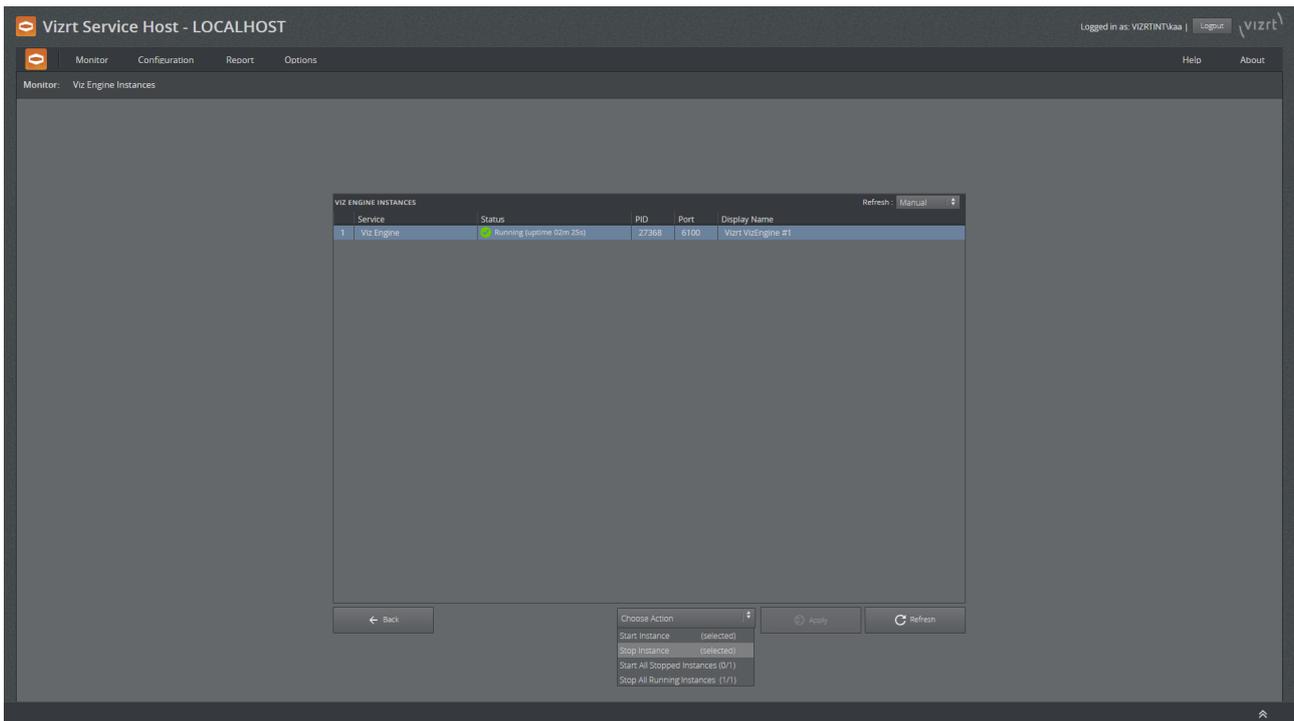
The page is automatically refreshed every five seconds by default.

Preconditions

- Viz Engine needs to be installed at the default location.
- The General Communication Port of each Viz Engine instance needs to be configured properly. Otherwise, the control instance is not able to stop the specified Viz Engine instance correctly.

Postconditions

- Stop: The Viz Engine instance was sent the `EXIT` command.

The shutdown of a Viz Engine may take some time. To kill the process, with all consequences, select **Terminate** from the context menu.

View Viz Engine Configuration

The config file of a Viz Engine instance can be viewed either from the Context menu on the Monitor page or via menu **Configuration > Viz Engine.**
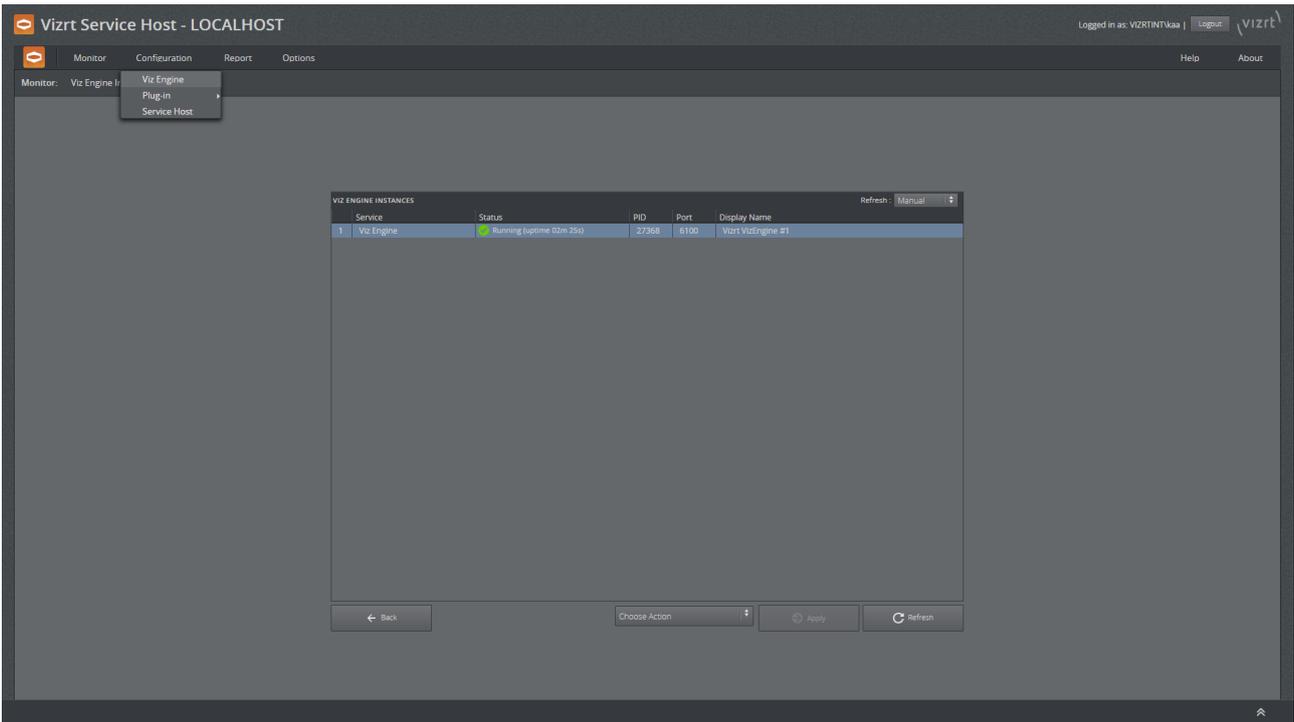
From there, select the Viz Engine instance.
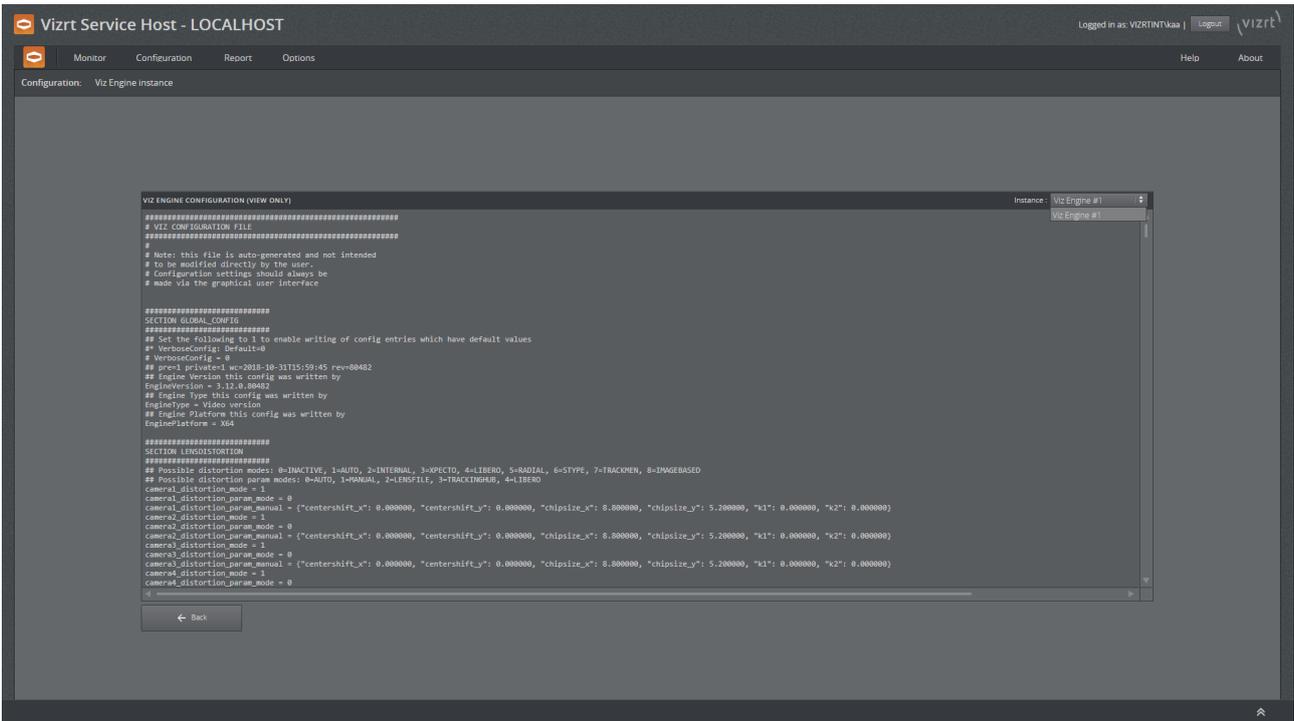
The page is view only.

View Viz Engine Logs

The log file of a Viz Engine instance can be viewed either from the Context menu on the Monitor page or via menu **Report > Logs > Viz Engine**.
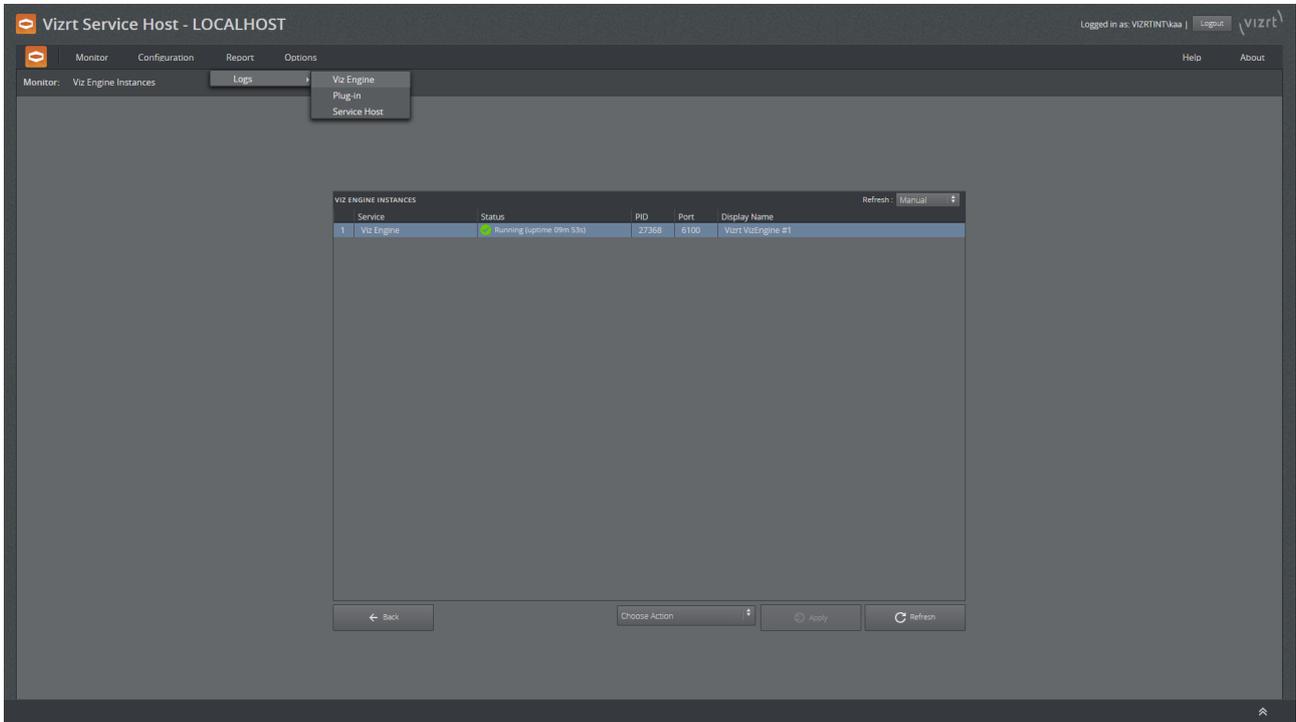
From there, select the Viz Engine instance to filter the available log files and how many lines to display.
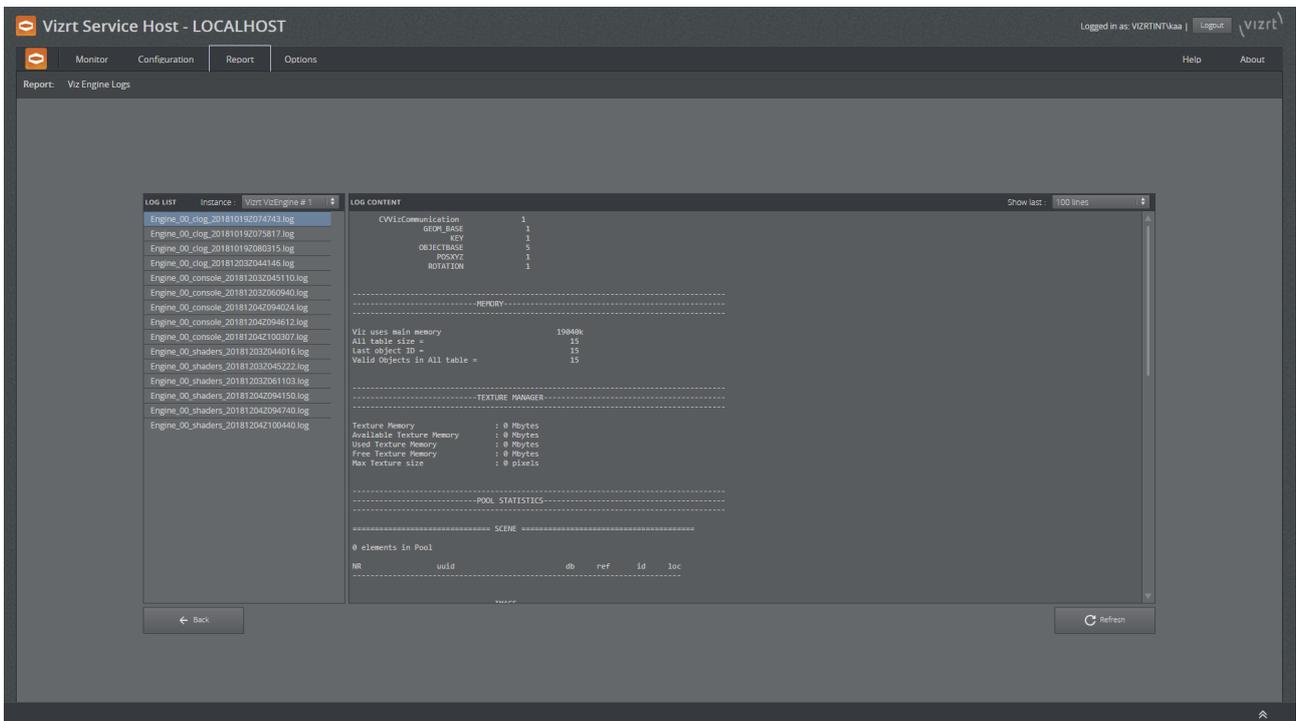
Notes

Viz Engine is not Installed

If Viz Engine is not installed on the machine, Service Host shows *no installation* at the status page.
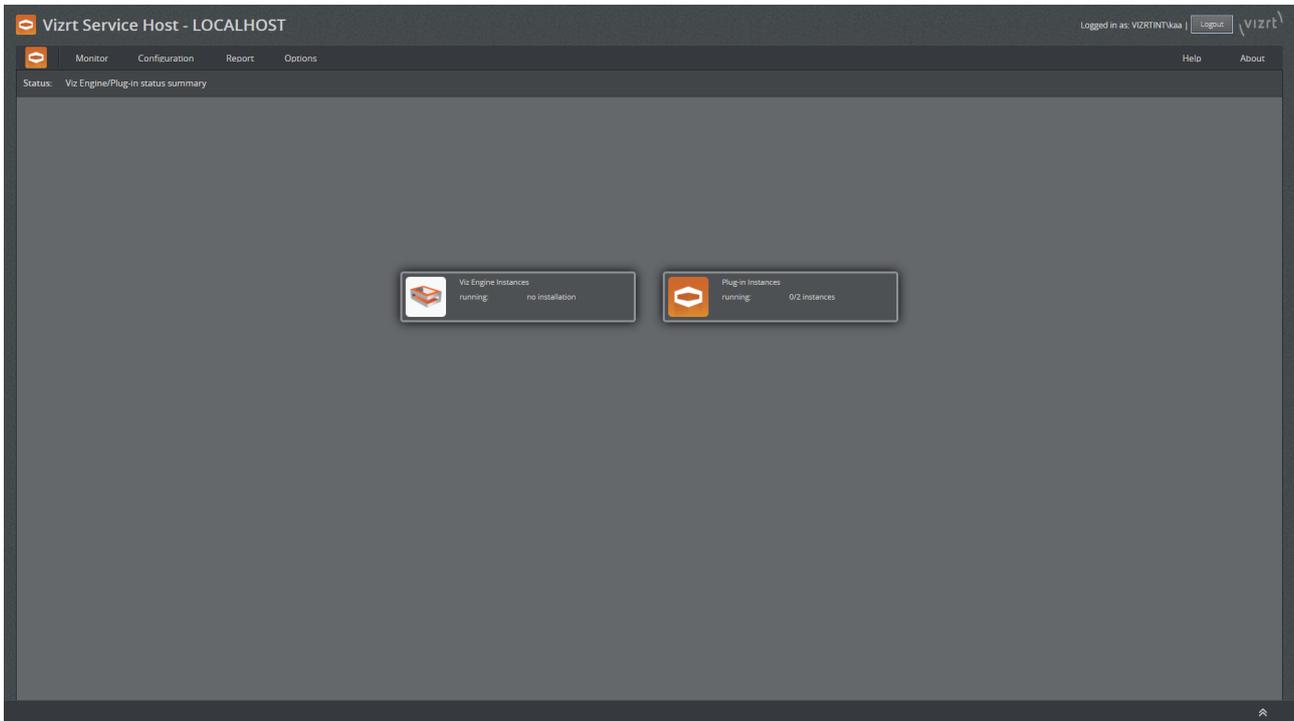
No instances are shown in the Monitor Viz Engine page.
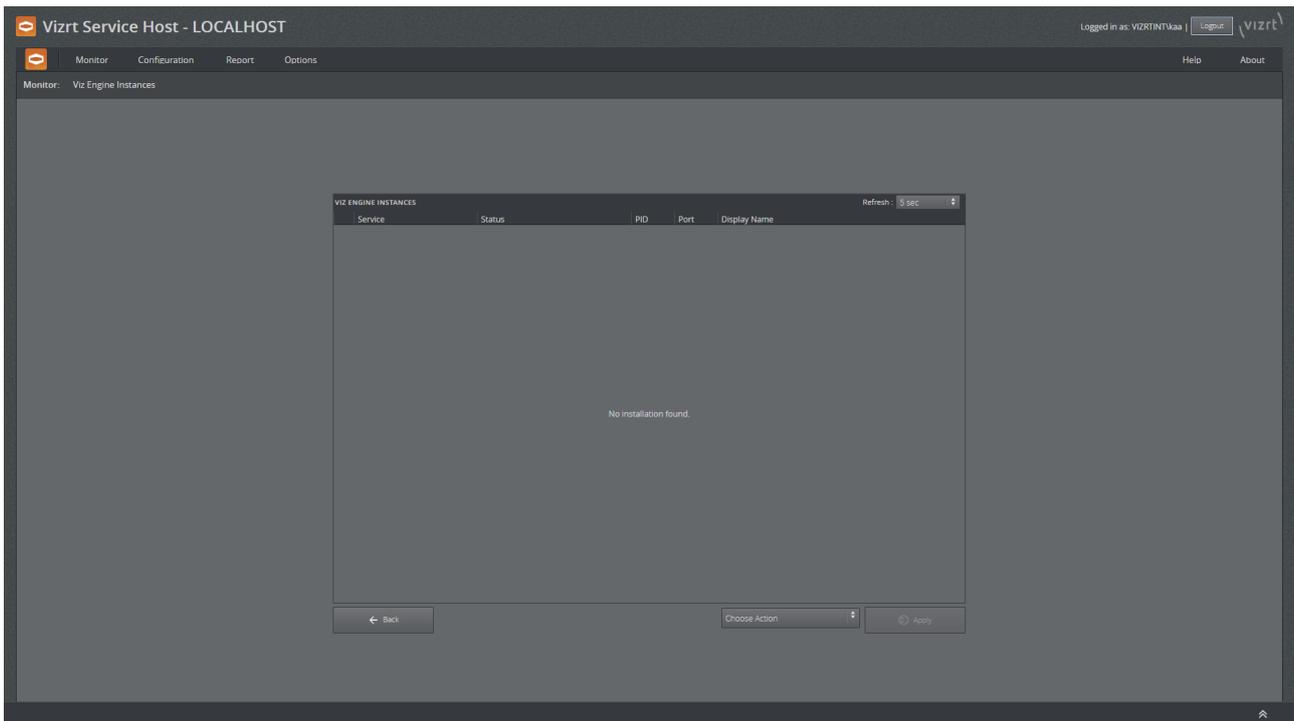
Viz Engine Started then Stopped

On some machines, it is found that after using Service Host (Control Mode) to start Viz Engine, The Viz Engine instance starts, but stops shortly after. This could be related to Session 0 Isolation problem and NVIDIA's DCH Display driver. There are two workarounds for this problem:

- Install NVIDIA's Standard Display driver instead. Perform a *Clean* installation, or
- Configure Service Host(Control Mode) to start Viz Engine on User Session. For more information, please see the Configuration section.

3.2.2   Plug-ins

Register a Plug-in Instance

To register a new instance of a plug-in as a Windows Service, use the menu **Configuration > Plug-in > Register New Instance**.
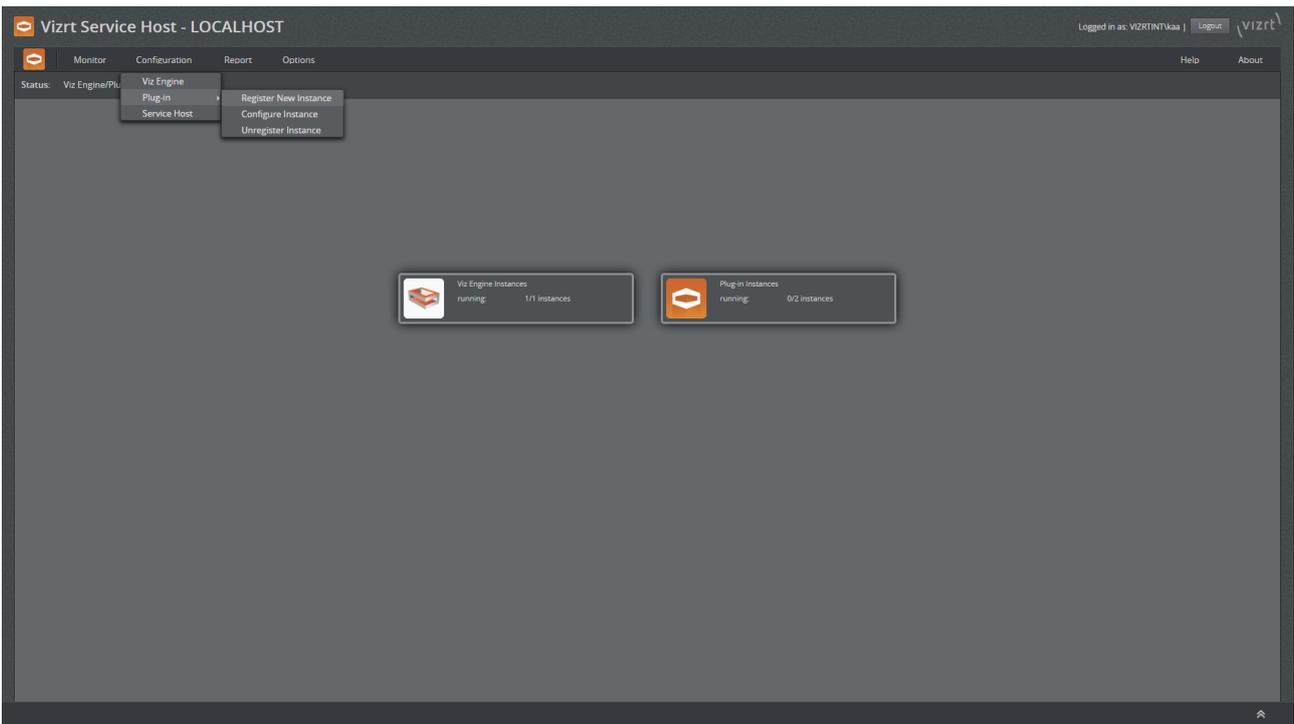
On the following screen, select the plug-in type and give the instance a service and display name (for example, register a Channel Recorder):
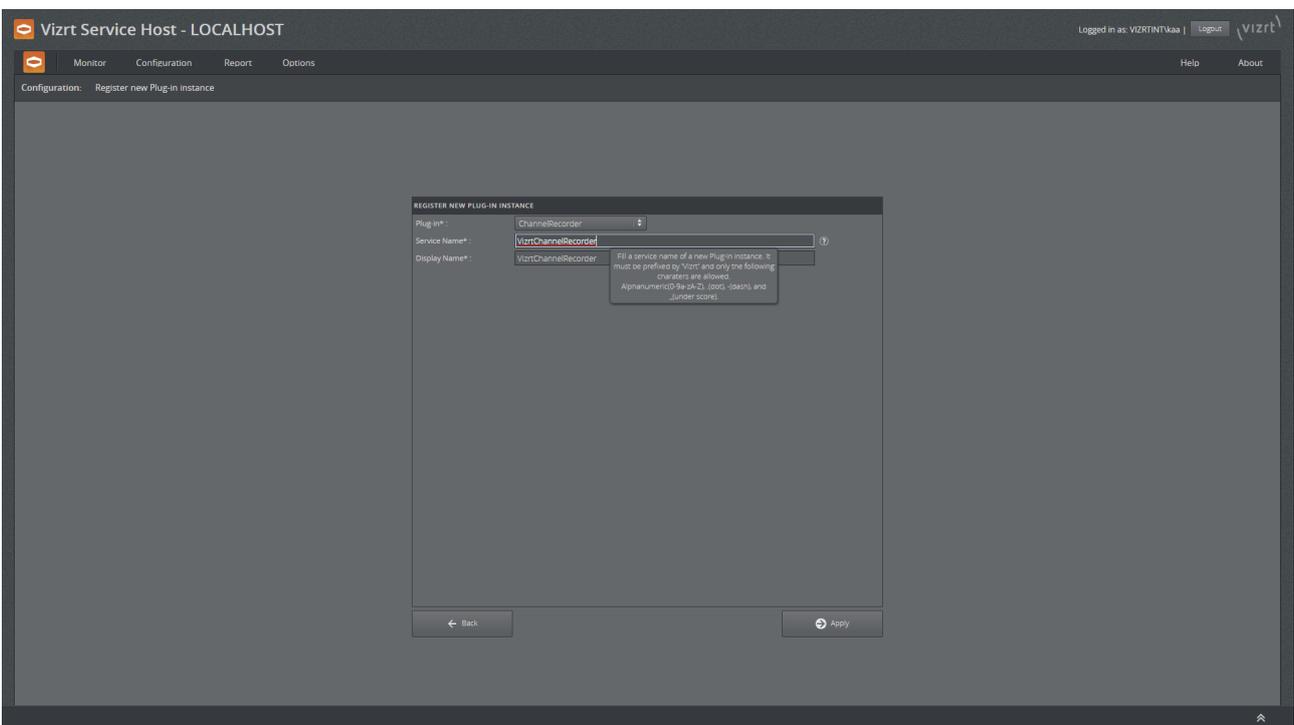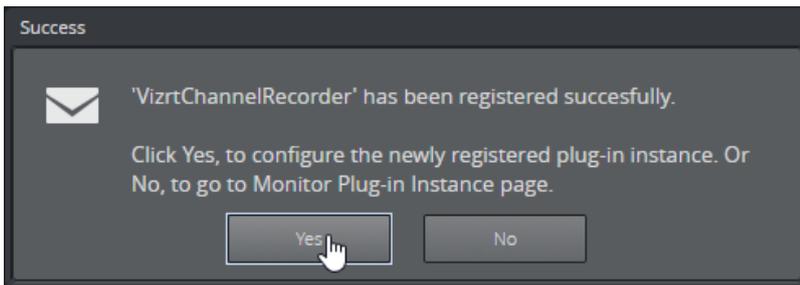
Confirm by clicking **Yes**. The dialog moves directly to the configuration of this plug-in instance.

Configure a Plug-in Instance

The configuration of a plug-in instance can be reached from the Monitor plug-in page via the context menu of a plug-in instance or via the menu **Configuration > Plug-in > Configure Instance**.
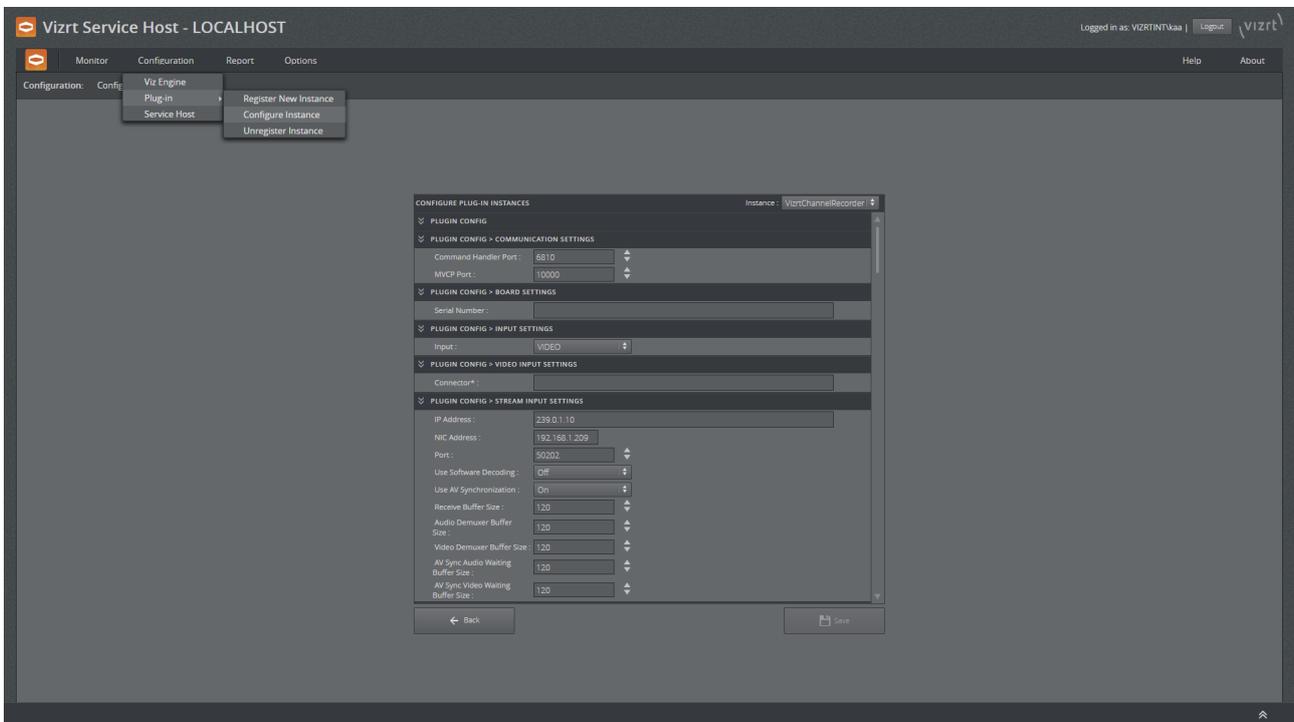
Plug-in Monitor

The plug-in monitor page can be reached via the menu **Monitor > Plug-in** or from the welcome page by clicking on the Plug-in instances icon.
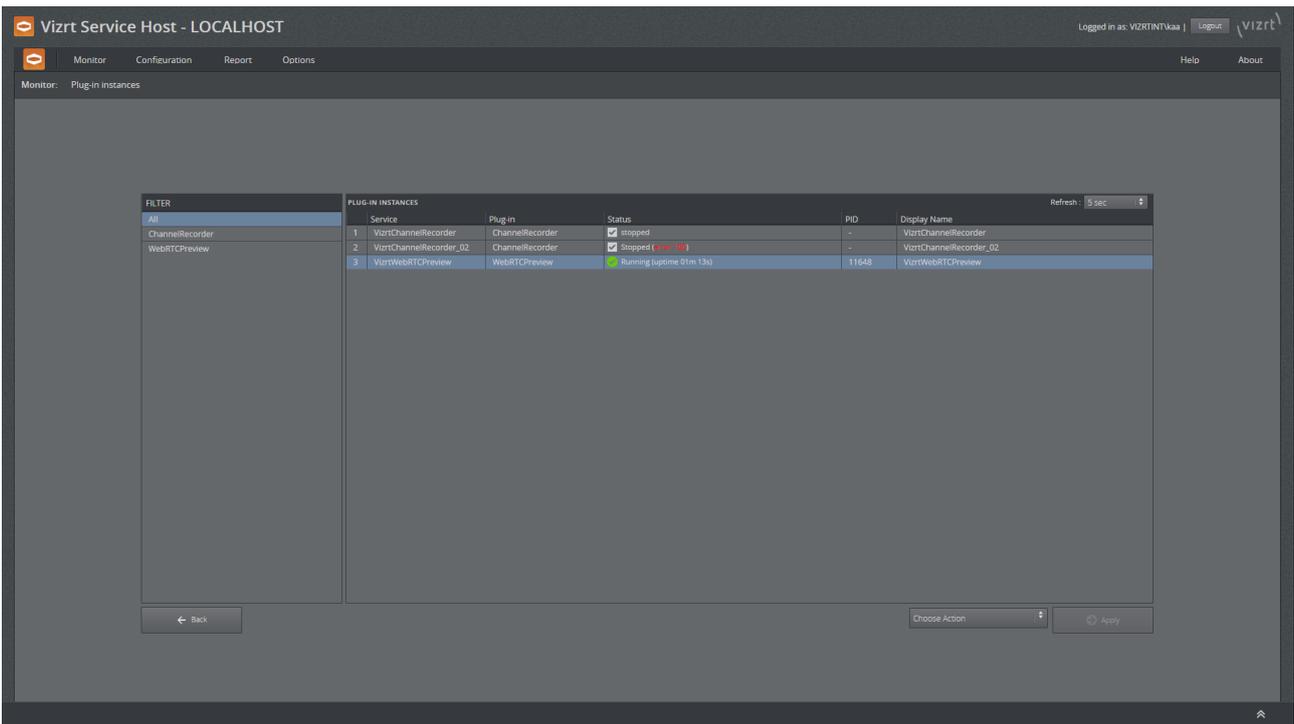
On this page, there is a filter on the left-hand side to select the available plug-in instances. Each discrete entry on the right-hand side has a context menu that allows the plug-in instance to be unregistered.

Start/Stop a Plug-in Instance

The context menu provides the capability to start/stop a plug-in instance.
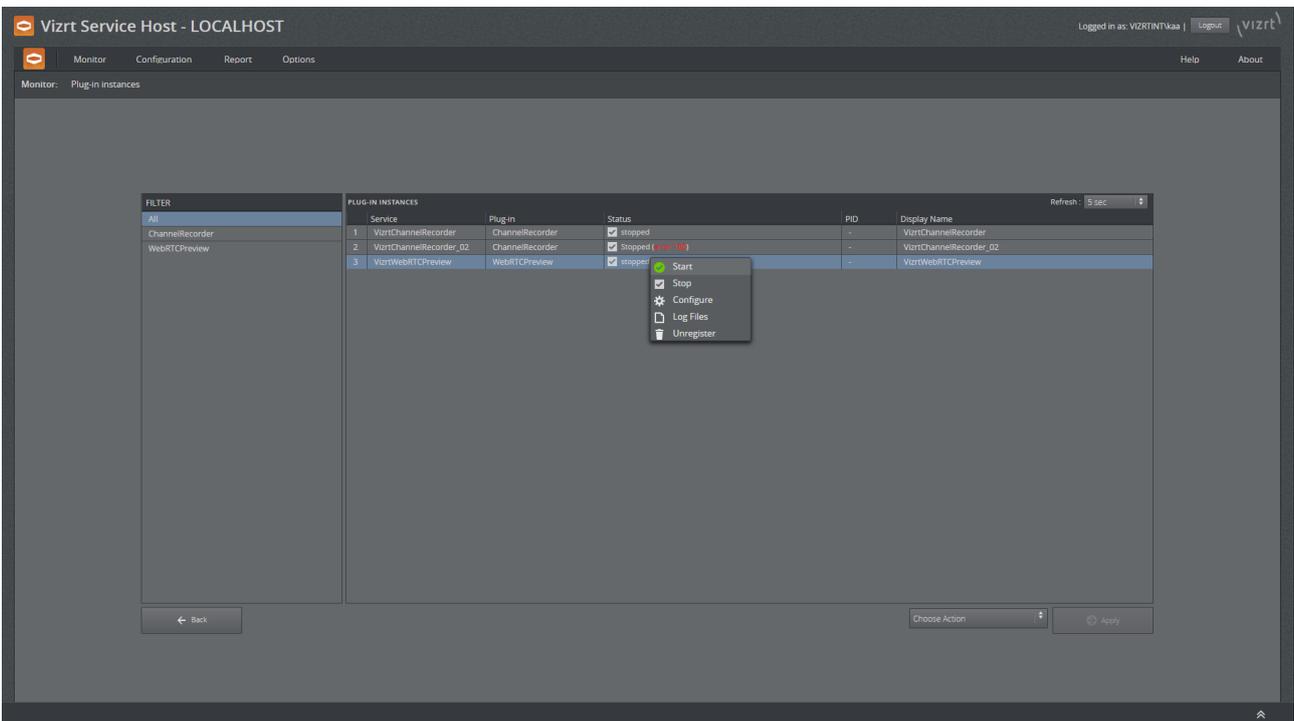
Unregister a Plug-in Instance

The context menu provides the capability to unregister a plug-in instance. Once selected and the following dialog has been answered with Yes, then the plug-in instance is stopped if it was running and unregistered from Windows services.

Any log and configuration files remain in *%ProgramData%* for later use by a subsequent new registration.
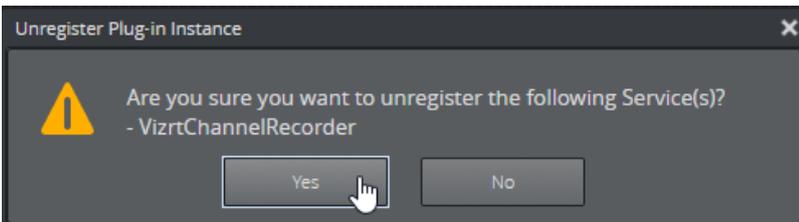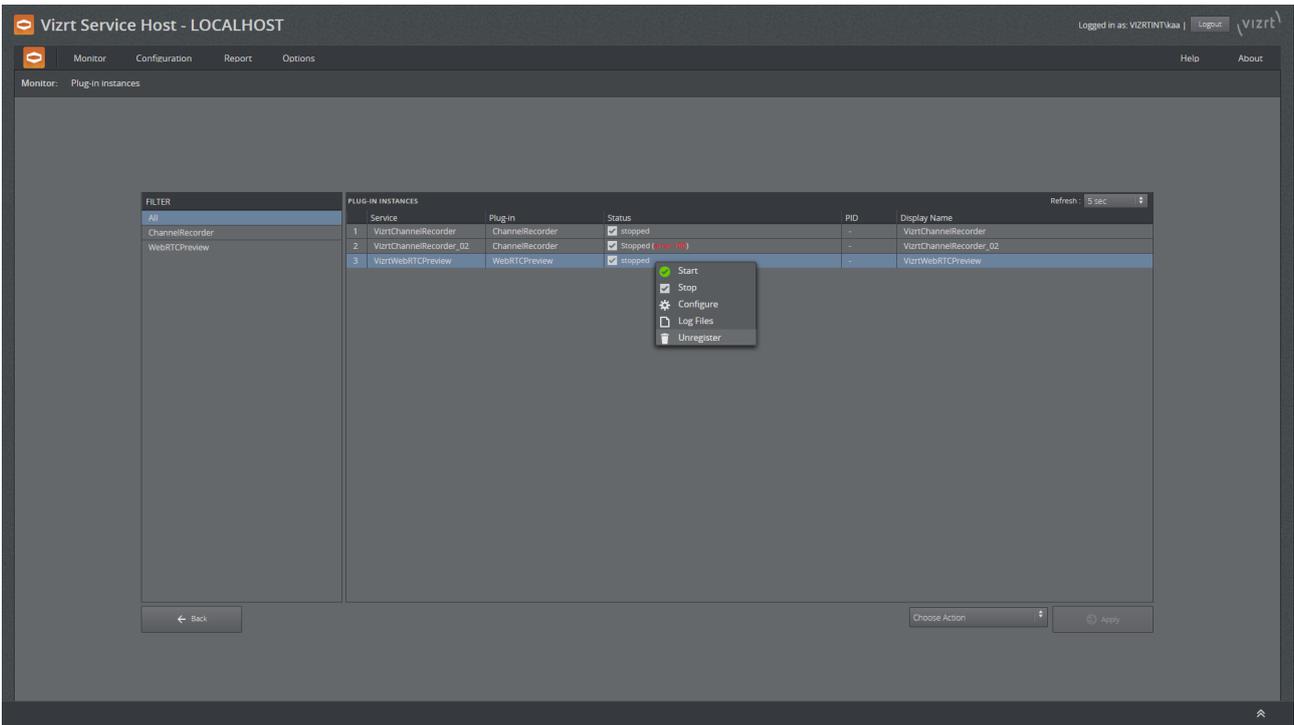
Report/Logs

The logs can be reached either via the context menu on the plug-in monitor page or from the menu **Report > Logs > Plug-in**.
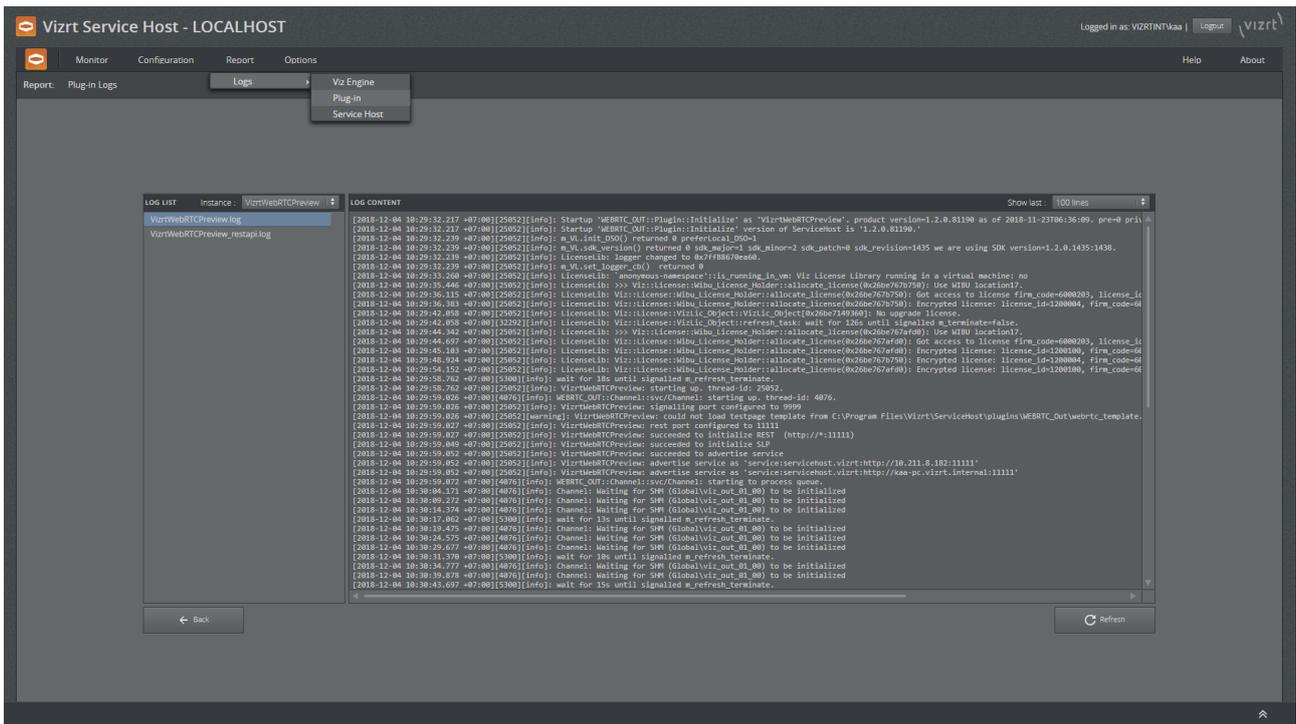
On the left-hand side, there is the instance filter and the content of the selected log file is displayed on the right-hand side of the screen. The number of lines displayed can be selected. It operates like the `tail` command.

Notes

The Configuration File of a Plug-in Instance is Missing

The configuration file of a plug-in instance, located at *%ProgramData%\vizrt\ServiceHost*, contains the plug-in name, which is a most important information. Service Host uses the plug-in name to load the correct DLL at run time to serve functionalities and get some information from the DLL, such as the configuration model. If the configuration file missing, a plug-in instance cannot be started.

At the Plug-in monitor page, for the plug-in instances which their configuration files missing, the *Plug-in* field is empty. To fix the problem, users need to unregister and re-register them.
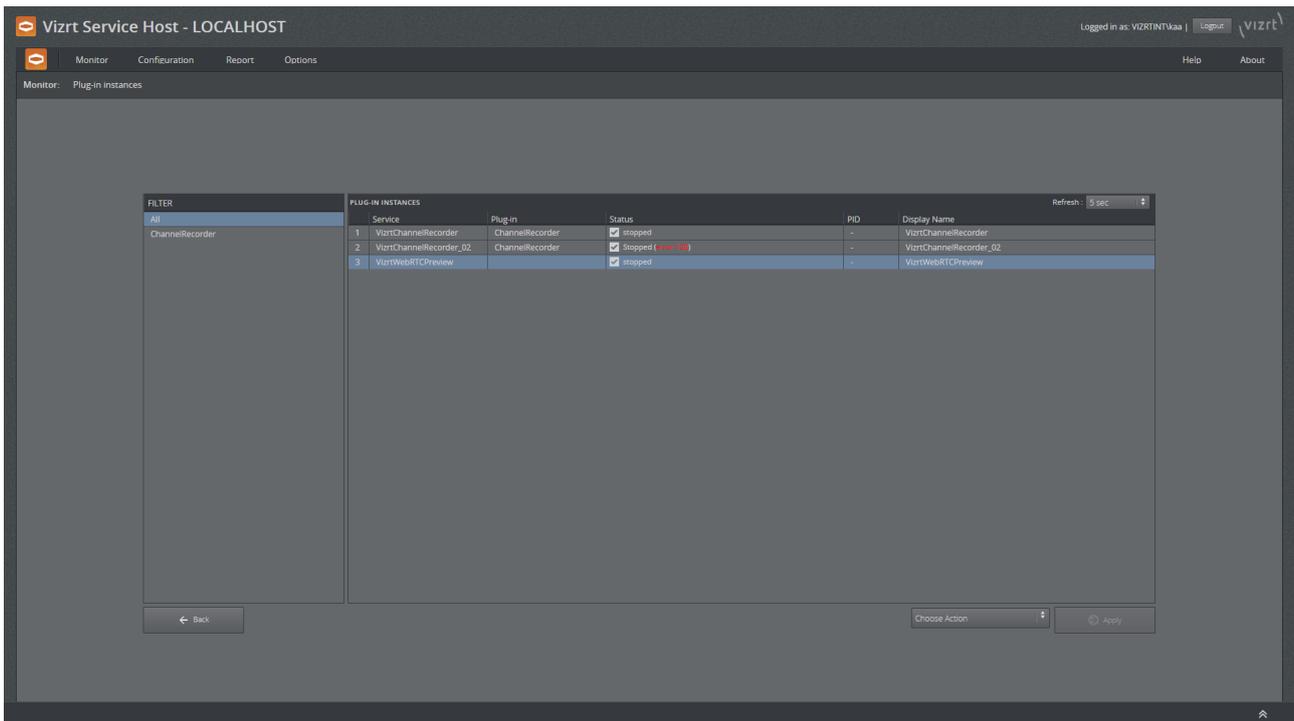
3.2.3   Service Host (Control Mode)

The *Control Mode* can be configured too. For security reasons, the configuration of the *Control Mode* needs to be configured by administrators.

Configuration

The configuration page for the *Control Mode* may be reached via the menu **Configuration > Service Host**.
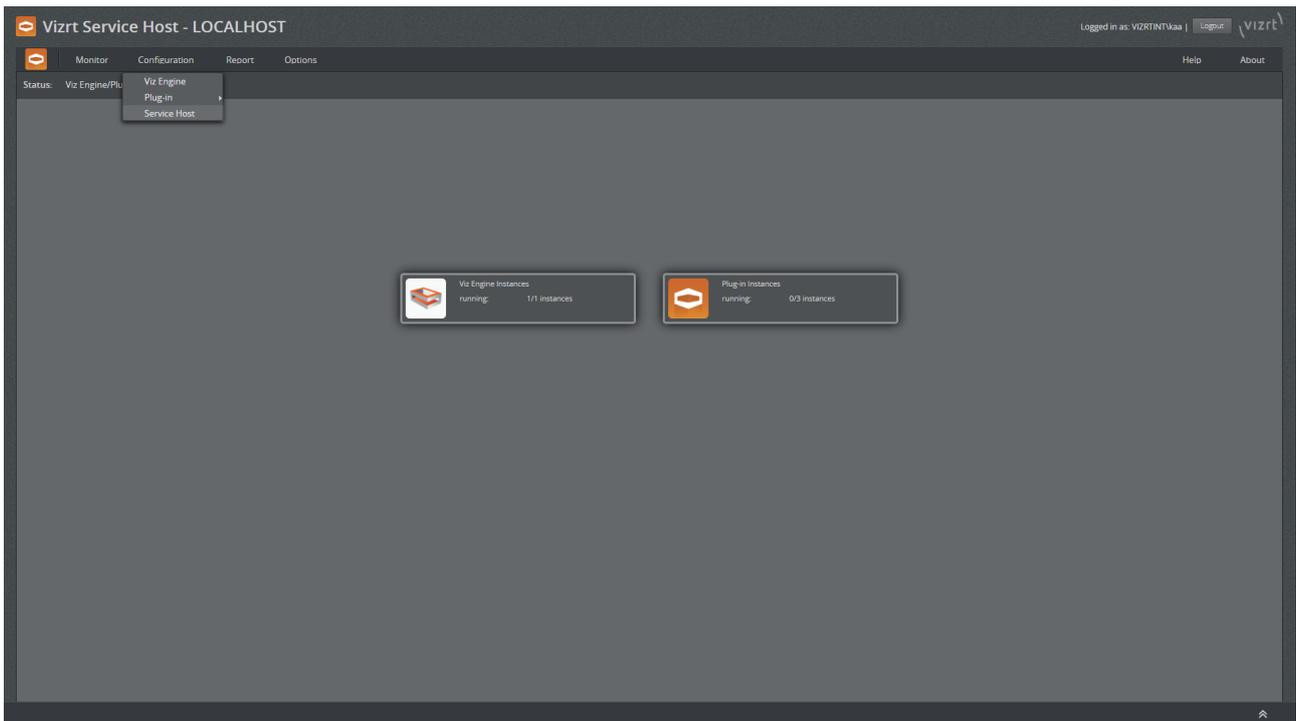
On this page, there are two main sections that can be configured.
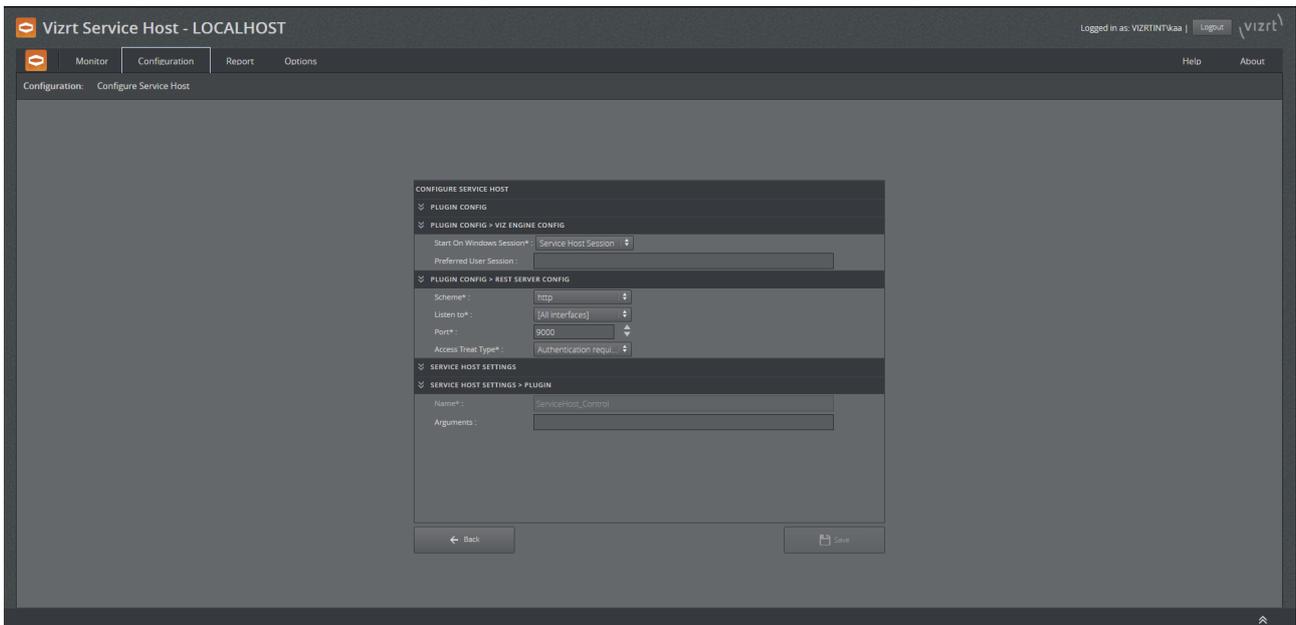
Plug-in Configuration

VIZ ENGINE CONFIG

It is possible to configure a Windows Session on which Viz Engine runs. Service Host(Control Mode) runs as a Windows Service running Session 0, thus a Viz Engine process started by Service Host(Control Mode) runs on Session 0 too. On some environment, Session 0 Isolation problem can be found and it causes Viz Engine to not able

able to access a GPU and terminates. A feasible work around for this might be upgrading NVIDIA Display driver to Standard type or to configure Service Host(Control Mode) to start Viz Engine on a User Session instead.

To configure Service Host(Control Mode) to start Viz Engine on what Windows Session, set **Start On Windows Session** field to one of the following values:

- **Service Host Session (SERVICEHOST_SESSION):** To start Viz Engine on the same Windows Session as Service Host(Control Mode) , this is the default setting.
- **User Session (USER_SESSION):** To start Viz Engine on the same Windows Session as an interactively logged-on user.

> ⚠️ **Note:** If there is no interactively logged-on user, Service Host (Control Mode) is not able to start Viz Engine.

If multiple users interactively log on at the same time, it is possible to configure a preferred username at *Preferred User Session* field.

REST SERVER CONFIG

The REST listening address can be configured. By default, Service Host is listening to all interfaces. This is specified by *[All interfaces]* in the field. For a port number, any valid and unused port number may be used. Later versions may support automatically finding of an available port number.

The Access Treat Type controls who is able to interact with Service Host.

- **Only local access (LOCAL_ONLY):** To allow only users accessing the GUI via the local machine to change the system (such register/unregister new plug-in instance or start/stop Viz Engine instances). Users accessing the GUI from external machine, regardless of whether authenticated, are not allowed to make any changes.
- **Authentication required (AUTHENTICATION_REQUIRED):** Same as above, only local access; except that users accessing externally are allowed as long as they are authenticated. In future versions, this may change to all users are required to authenticate before they can make changes to the system.
- **No authentication (NO_AUTHENTICATION):** No restrictions apply. All users may change the system, both locally and externally. We do no recommend this setting since there is no way of gathering user information in the operating logs.

Service Host Settings

PLUGIN

The Arguments editor allows passing arbitrary startup arguments to the control mode plug-in. In this case, we set the log level to debug mode.

Allowed values for Arguments are:

- -l debug
- -l info
- -l warning
- -l error
- -l off

Report/Logs

The logs produced by the control mode may be accessed via the menu **Report > Logs > Service Host**.
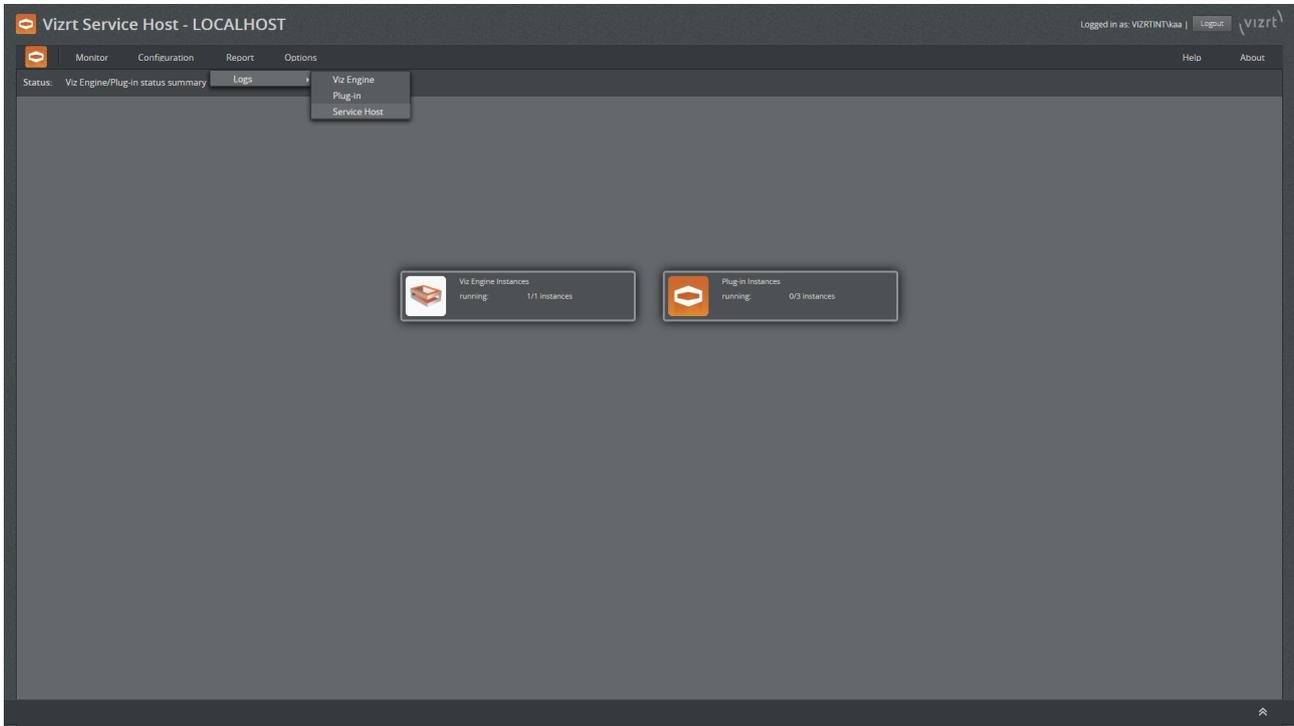
On the left-hand side, select the log file to be displayed. There are two types of logs available.

The Control Mode Log

This log file is named *ServiceHost.control.log*, it does log rotation and may also be reached via the log panel on the bottom of the screen.
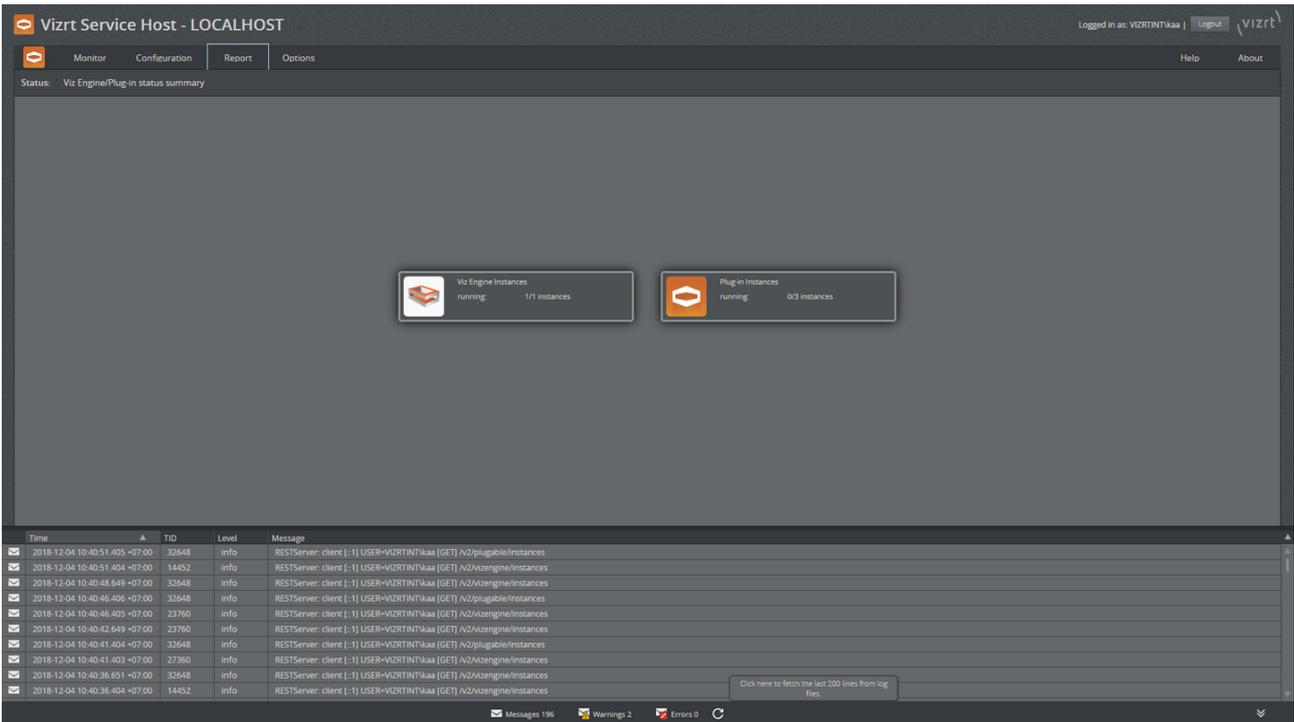
and
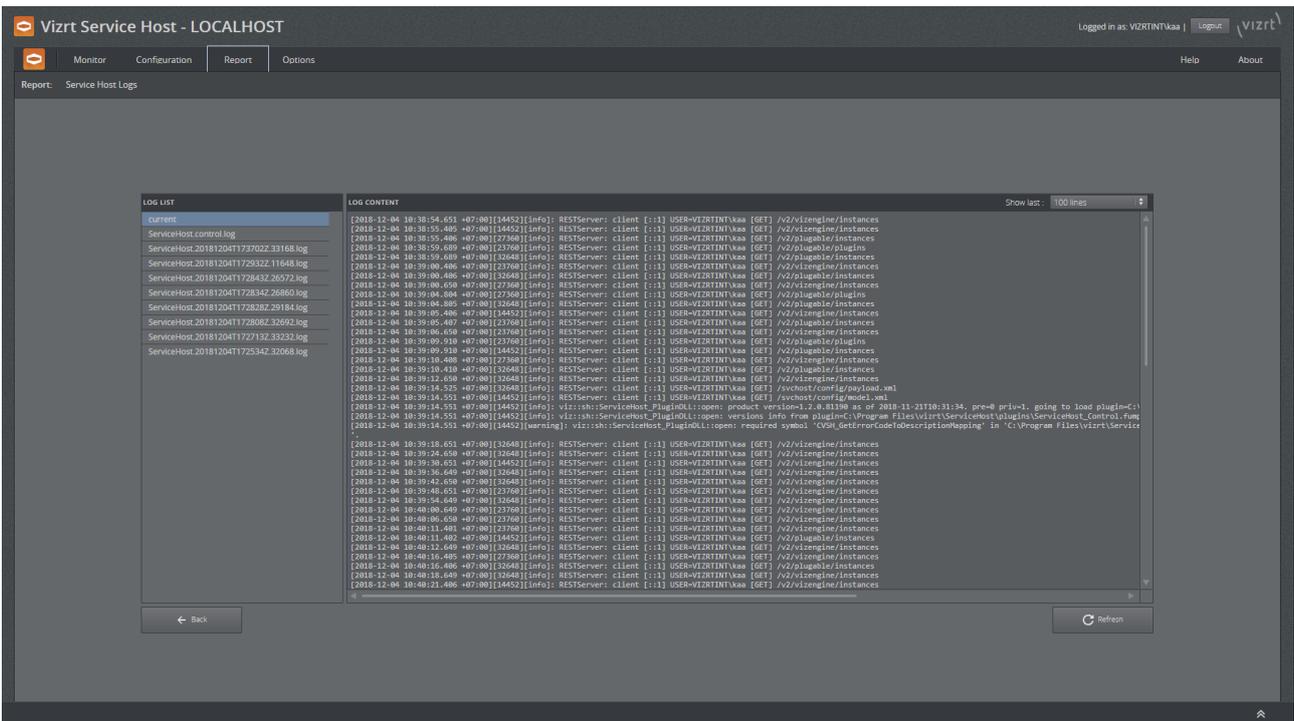
Service Host Startup Logs

Logs are produced during startup and kept as long as this Service Host is running, so possible startup issues can be captured before a plug-in is loaded and initialized. These log files are removed after a successful shutdown. They

are named *ServiceHost.dateandtime.pid.log*, where dateandtime is the date and time the selected timezone, pid is the process ID.

Debugging of Service Host

Manual Register/Unregister and Stop/Start

Service Host is capable of registering and unregistering itself as a Control Mode service. This might be useful in cases not covered by the installer.

Register Control Mode from Command Line

```
Z:\> ServiceHost.exe -c -i
```

Unregister Control Mode from Command Line

```
Z:\> ServiceHost.exe -c -r
```

Service Host in Control Mode can also be started and stopped from the command line.

Start Control Mode from Command Line

```
Z:\> ServiceHost.exe -c -s
```

Stop Control Mode from Command Line

```
Z:\> ServiceHost.exe -c -k
```

Launch Service Host in Foreground with a Console Window

For debugging, Service Host can be launched in foreground and with a console window attached. To list all the possibilities use the command line switch `-h` .

Command line Options for Service Host

```
Z:\>ServiceHost.exe -h Usage: $0 [-v] ... start from within windows services,
optional verbose mode $0 [-v] -N service_name -i ... install $service_name as service
-v ... optional verbose mode $0 [-v] -N service_name -r ... remove $service_name as
service -v ... optional verbose mode $0 [-v] -N service_name -s ... start service
$service_name -v ... optional verbose mode $0 [-v] -N service_name -k ... kill
service $service_name -v ... optional verbose mode $0 [-v] -N service_name -d ... run
$service_name in debug mode with a console window -v ... optional verbose mode $0 [-
v] -c -i ... install control as service, the service name is '{ProductName}.control'
-v ... optional verbose mode $0 [-v] -c -r ... remove control as service, the service
name is '{ProductName}.control' -v ... optional verbose mode $0 [-v] -c -s ... start
control service, the service name is '{ProductName}.control' -v ... optional verbose
mode $0 [-v] -c -k ... kill control service, the service name is
'{ProductName}.control' -v ... optional verbose mode $0 [-v] -c -d ... run control in
debug mode with a console window -v ... optional verbose mode -p "[argument...]" ...
arguments that gets forwarded to the plug-in e.g. -p "-l debug -v"
```

What we are interested in here are the lines with the `-d` option.

To start Service Host in Control Mode with a console window, make sure it is not running as windows service then launch it with `-c -d` options.

```
Z:\>ServiceHost.exe -c -k Z:\>ServiceHost.exe -c -d
```

Optionally, you can add verbose and log parameters to Service Host and the Control Mode plug-in.

```
Z:\>ServiceHost.exe -c -d -v -l debug -p "-v -l debug"
```

the first `-v` `-l` debug are for Service Host executable and the one inside the `-p` option are forwarded to the Control Mode plug-in. In the console mode, the logs are also forwarded to the console window.

4 Channel Recorder

This section details the installation, configuration and operation of Channel Recorder. Channel Recorder is a service that creates video clips, using an SDI or IP stream as source. It can be controlled using an arbitrary set of commands, or by remote applications utilizing the Multiport Video Computer Protocol (MVCP). The incoming stream is captured into a file.

4.1   Installation and Configuration

This section describes the installation requirements and procedure required to use Channel Recorder. It contains information on the following topics:

- Hardware and Software Requirements
- Licensing
- Installing, Registering and Removing Channel Recorder
- Upgrading from 1.0
- NMOS Configuration
- Persistence Configuration
- Example Configuration File
- Integration with Other Services
- Initialization Errors
- Upgrading from 1.1 and Later
- Channel Recorder Configuration

4.1.1   Hardware and Software Requirements

Hardware Requirements

The Viz Engine Administrator Guide provides a detailed description of these machines.

> ⚠ **Note:** To be able to achieve maximum performance using Channel Recorder, **Hyperthreading** should be enabled. This is located in the BIOS.

Channel Recorder requires one of the following video boards:

Video Boards	Recording Capability
Matrox X.mio3	SDI
Matrox DSX LE4	SDI
Matrox X.mio5 IP	SMPTE ST 2110
Matrox X.mio5 X2	SDI
Matrox X.mio5 / 8	SDI
Matrox DSX LE5 / 4	SDI
Matrox DSX LE5 / 8	SDI
Matrox M264	RTP/H.264
DSX.Core [1]	RTP/H.264

[1] From Service Host version 2.1.0 and later, DSX.core has a new license mechanism. The CodeMeter **server search list** must be updated so that the old server license that contained DSX.core licenses is removed from the list. If CodeMeter **server search list** still has the old server license listed, then Channel Recorder still attempts to use the old license implementation.

> ⚠ **Note:** All the above boards should be upgraded to enable recording with different codecs. To check which upgrades the board has available, use the **Matrox Utils** in the task bar and check if the board has 500 or 550 upgrade applied (for ProRes). If the board does not have this upgrade, this must be ordered. Additionally, to receive RTP/H.264 the Matrox M.264 must have the RTP upgrade. Refer to Recording RTP/H.264 Stream for more information.
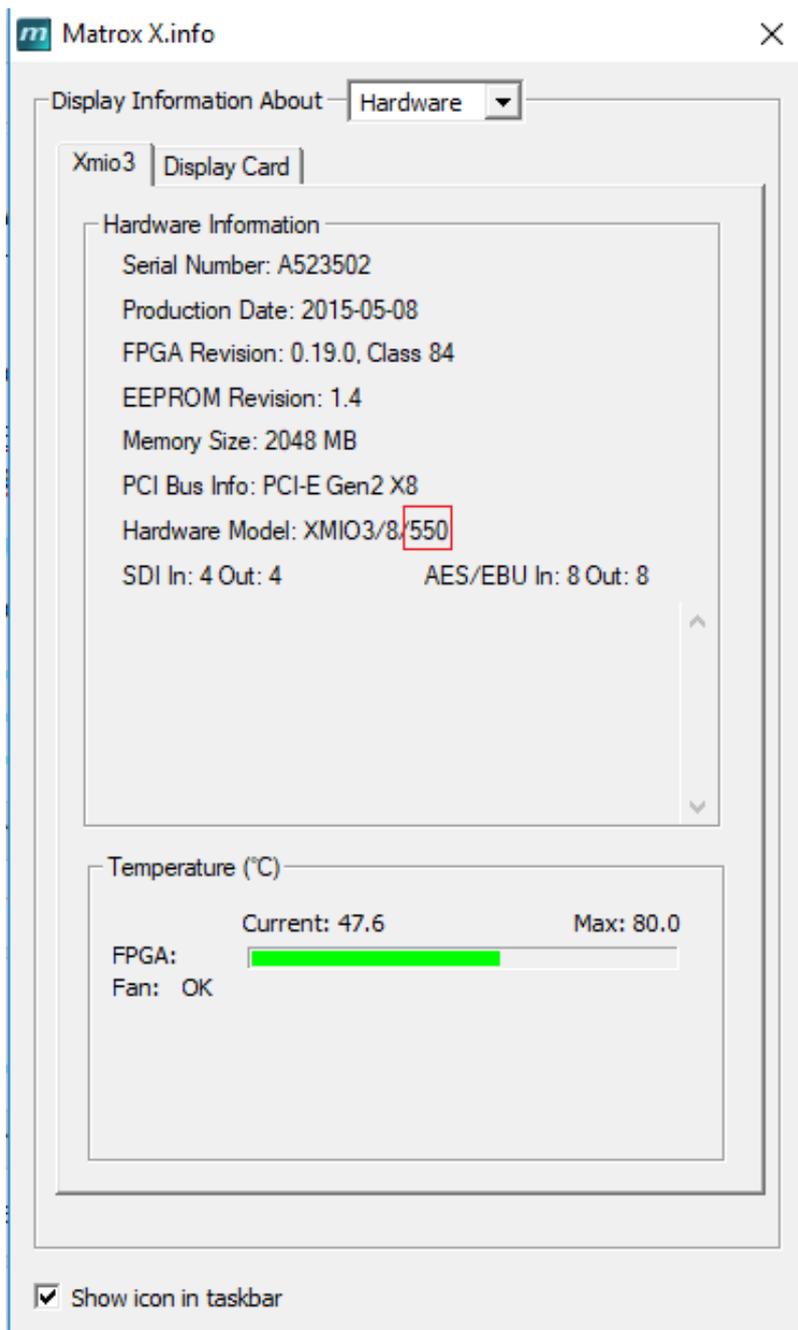
> ⚠️ **Note:** The DSX.core is not technically a physical board and is used to receive and record RTP/H.264 streams. Refer to Recording RTP/H.264 Stream for more information.

> ❌ **Important:** This procedure needs to be followed after installing DSX.core:
> 1. Unregister *mvfDsxCore.dll*.
>   a. Click **Start > Run** (or use the Windows command line: **Search > CMD >** (Right click) **Run as Administrator**).

b. Type `REGSVR32 /U "C:\Program Files\Matrox DSX-TopologyUtils\System64\mvfDsxCore.dll"` and press **ENTER**.

2. Shut down *X.info* in the task manager.
3. Delete *mvfDsxCore.dll* from the folder *C:\Program Files\Matrox DSX-TopologyUtils\System64*.
4. Start *X.info*.

> ⚠ **Note:** The M.264 encoder/decoder board is required for CPU intense codecs like XAVC. It is mandatory for UHD recording and strongly recommended for AVC codecs.

To Upgrade Class and Firmware

> ❌ **Important:**
>
> When upgrading from 10.4 to 11.x Matrox driver, follow these mandatory steps:
> - Uninstall old Matrox driver
> - Cold Boot (Turn off computer for at least 20 seconds)
> - Switch Computer on
> - Continue installation of new driver

To upgrade your Matrox board, for example to enable more IO-ports or a codec upgrade for HD recording, you must use the *mvDongleUpdater.exe* update utility with your Vizrt supplied license upgrade file *<filename>.OPT*. This utility is installed with the Matrox driver package and is normally installed to *C:\Program Files\Matrox DSK.utils\drivers\mvDongleUpdater.exe*.

To upgrade the card use this syntax:

```
C:\Program Files\Matrox DSX.utils\drivers\mvDongleUpdater.exe upgrade
-sn="BoardSerialNumber" -f="Path to the upgrade file"
```

The command above must be run from a Windows command line window.

Software Requirements

Channel Recorder with Service Host requires Matrox DSXTopology Utils version 11.2.101.2608.

In most cases, a system with a configured and reliable NTP server is necessary. This is particularly important when working with other components such as Viz Dart or Viz One, in which case they should all use the same NTP server. The NTP should be configured so that the polling interval is low enough to not cause noticeable drifting. A value of one minute polling time should be used as a maximum. In Windows, this can be changed using regedit. The important values are located in:

- **HKEY_LOCAL_MACHINE > SYSTEM > ControlSet001 > services > W32Time > TimeProviders > NtpClient**
  - SpecialPollInterval should be `60`.
- **HKEY_LOCAL_MACHINE > SYSTEM > ControlSet001 > services > W32Time > Config**
  - MaxNegPhaseCorrection should be `0xFFFFFFFF` (default value for domain members).

- MaxPosPhaseCorrection should be `0xFFFFFFFF` (default value for dmain members).
- UpdateInterval should be `100`.

More info is available at https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/windows-time-service-tools-and-settings#Configuration.

See Also

- Viz Engine Administrator Guide

4.1.2   Licensing

This chapter describes management and usage of the licensing system based on CodeMeter from WIBU Systems available in Channel Recorder version 1.3 and later.

Important Pre-installation Information

The WIBU licensing system requires the installation of the CodeMeter Runtime Software 8.40a (6.60a for versions below 1.5.0). This is included as part of the bundle installation of Service Host. When the license is retrieved from a dedicated license server, it must be configured in the Vizrt Licensing Service (see the **Installation** section of the Viz Licensing Administrator Guide) or the CodeMeter WebAdmin.

Please refer to the Viz Licensing Administrator Guide for further detailed information.

> ℹ **Info:** There is an auto discovery if no license server is configured in the server search list of CodeMeter.

> ℹ **Info:** On network disconnect and reconnect, it may happen that a license is checked out twice. In this case, it must be released manually on the CodeMeter service on the license server or the license server can be restarted.

Key Features of the Licensing System

- Dongle less operation on the clients with monitoring and logging capabilities.
- Grace periods for allocated licenses to avoid immediate expiration on short network interruptions.
- Configurable WIBU license container location.

General Workflow of the Licensing System

- Channel Recorder is unable to start up if any requested license is missing during start up.
- If the Service Host Core feature (`SVCHOST_CORE`) has expired or is not accessible during the execution of Channel Recorder, then the user is unable to start any kind of recording.
- If the SDI input feature (`SDI_IN_MAX_BINARY`) has expired or is not accessible during the execution of Channel Recorder, then the recording will be black 100 frames every 400 frames.
- If DNxHD feature (`DNXHD_BINARY`) has expired or is not accessible during the execution of Channel Recorder, then the user is unable to set the config codec to `DNxHD` or container to `DNXHDMXF` .
- If the UHD feature (`4K_ENABLEMENT_BINARY`) has expired or is not accessible during the execution of Channel Recorder, then the user is unable to use the feature.
- If any of the licenses are not accessible or have expired **during a recording**, then the recording will be black 100 frames every 400 frames.

> ⚠ **Note:** Expired means after the grace period has ended.

Feature	Description	WIBU License ID Enabling this Feature
`SVCHOST_CORE`	Service Host Core	1200004, Service Host Core
`SDI_IN_MAX_BINARY`	enable unlimited SDI Inputs	1200200, SDI/IP Video Input Enablement (max), Requires Service Host Core.
`4K_ENABLEMENT_BINARY`	enable 2k-4k Resolution	There are two UHD license options that can be used for ingest. However, the user can only choose one. Both require Service Host Core. 1200400, 4K Video Enablement 1200404, Ingest - 4K Video Enablement
`DNXHD_BINARY`	enable DNxHD Codec	1200603, Viz Engine DNxHD Codec Support, Requires Service Host Core.

Some licenses are the same as for Viz Engine and can be shared with Viz Engine when used on the same machine (1200200, 1200400, 1200603).

There is a demo mode available, however the core still needs to be licensed. The demo mode is useful if there is no SDI input license but the user still wants to test Channel Recorder. In the demo version UHD or DNxHD recording is not available and the recordings will be black 100 frames every 400 frames.

OEM licenses contain the features listed above in a single license, which means only the core license may be sufficient. When using an OEM license, set the other configuration options to *No additional license* for License, 4K Video Enablement and DNxHD Enablement. If they are allocated, an error occurs as those licenses require the Service Host Core.

Basic Setup

These are the steps to set up Channel Recorder licensing:

1. Install Channel Recorder with the bundle installer (CodeMeter Runtime is included in the bundle).
   a. Configure CodeMeter using the CodeMeter WebAdmin (can be opened from the CodeMeter Control Center) or through *http://localhost:22352/*.
2. Configure the license system for the corresponding plugin instance in the Service Host web interface (*http://localhost:9000/*).

> ⚠ **Note:** In the Container and Core selection compatible licenses are searched and shown. This depends on the CodeMeter configuration. If a license with shared mode exclusive is already allocated on this host then it is listed, even if it is not compatible (for example, Viz Engine combination feature). The reason is that the license is locked for any other processes on this host and cannot be queried for license details, thus listed by default.

Please refer to the Viz Licensing Administrator Guide for detailed information regarding installation and operation of WIBU Licensing.

4.1.3   Installing, Registering and Removing Channel Recorder

Channel Recorder runs as a plug-in of Service Host Administrator Guide. Channel Recorder cannot operate without Service Host, for this reason the installation of Channel Recorder consists of the following parts:

1. Running the **Service Host Setup Wizard**.
2. Manually registering one or more Channel Recorder instance(s) using the Service Host web interface.

Before installing Service Host, please make sure to obtain the latest installer from the Vizrt FTP server: ftp:// download.vizrt.com/

- To Install Service Host
- To Register Channel Recorder Instance
- Upgrading or Repairing an Existing Installation
- Removing Channel Recorder
- Starting First Recording

To Install Service Host

Refer to the Service Host Administrator Guide documentation.

To Register Channel Recorder Instance

Refer to the Service Host Administrator Guide documentation. After successfully registering your Channel Recorder instance, the following dialog window appears:
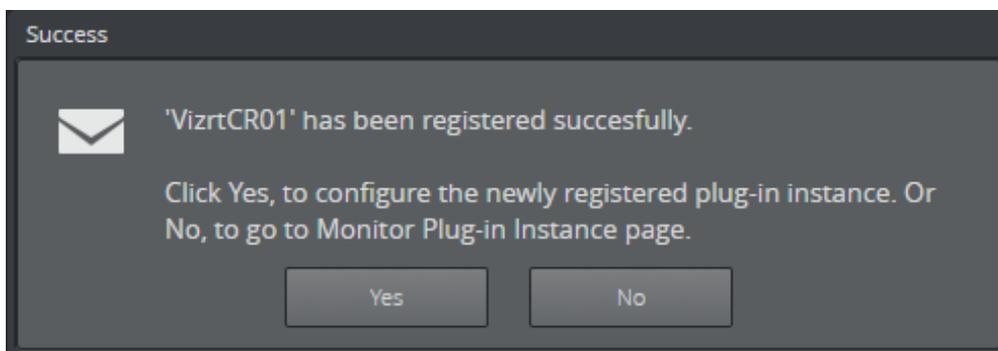
This is to inform you that the registration of the instance is successful. You are prompted to choose between configuring the instance now or later. See Channel Recorder Configuration for further details.

Upgrading or Repairing an Existing Installation

Use the Service Host Setup Wizard to upgrade or repair an existing installation of Channel Recorder. When upgrading or repairing, Service Host remembers the registered instances and which states they were before the upgrade or repair occurs. After a successful upgrade or repair operation, any previously running services stopped by the Setup Wizard is restarted.

Removing Channel Recorder

Use the Service Host web interface to remove Channel Recorder. For more information, refer to the Service Host Administrator Guide documentation.

Starting First Recording

To start recording a Matrox board needs to be installed with the correct Matrox Utils version and Matrox Codec license depending on the user needs. See Hardware and Software Requirements for further details. Before starting a recording, using the different available protocols check the log to find any possible error. See Troubleshooting Channel Recorder for further details.

4.1.4   Upgrading from 1.0

Starting with Channel Recorder version 1.1, Channel Recorder is considered a plug-in of Service Host. This means that the Channel Recorder is managed by Service Host through the web interface that it provides. With Service Host you can:

- Register a new instance.
- Remove an existing instance.
- Start an existing instance.
- Stop an existing instance.
- Provide an **initial configuration** to an instance.

For more information, refer to the Service Host documentation.

Matrox Driver

Please refer to the Release Notes about the required Matrox driver.

Configuration File

For users upgrading from Channel Recorder 1.0, the only required step is the translation of the configuration into the new format. Previously, the configuration was provided via a simple text file located in *%Programdata%/vizrt/ Channel Recorder*.

This file might contain the following:

```
COMMAND_HANDLER PORT SET 6810
MAIN BOARD SELECT A523502
MVCP PORT 10000
CONFIG SET RINGBUFFERSIZE 60
MAIN CLOCKTYPE GENLOCK
MAIN CONNECTOR SELECT A
CONFIG SET FILEEXTENSION OFF
CONFIG SET PRIORITY REALTIME
CONFIG SET CLIP_ROOT D:/
CONFIG SET TIMEOUT 5000
CONFIG SET AUDIOCHANNELS 4
CONFIG SET CODEC XDCamHD
CONFIG SET CONTAINER XDCAMMXF
CONFIG SET TIMECODE TIME_OF_DAY
CONFIG SET TDIRENABLE FALSE
CONFIG SET VBI OFF
```

In Channel Recorder with Service Host 1.5, the configuration is provided via an *.xml* file located in *%Programdata%/ vizrt/ServiceHost*. The configuration file name is the **Service Name** used when registering the Channel Recorder instance. This xml file can be edited directly, however, it is highly recommended that you use the Service Host web interface.

> ⚠️ **Note:** All configuration settings that were in 1.0 are still present with the same parameters and similar names.

You must use the Service Host web interface and configure Channel Recorder appropriately to achieve the same behavior as in the old version. Configuration is described in Channel Recorder Configuration.

> ⚠️ **Note:** Do not forget to restart the instance after any changes to the configuration.

Log File

As with the configuration file, the log file is now located in *%ProgramData%/vizrt/ServiceHost* instead of *%Programdata%/vizrt/Channel Recorder.*

The old file used to configure the logging mechanism is gone. The only option in the new version is to change the logging level. Refer to the Service Host Section in Channel Recorder Configuration.

4.1.5   NMOS Configuration

General Information

NMOS control is enabled on a driver level. No changes are needed in Channel Recorder. NMOS connection requests must be sent out-of-band for X.mio3. For X.mio5, the requests can be either out-of-band or in-band.

- **Out-of-band:** RDS and receiver NIC on the attached machines must be located in a network separated from the essence network, which hosts video, audio and ancillary data flows.
- **In-band:** The RDS can be hosted in the essence network, NMOS requests are sent alongside the essence flows.

> ❌ **Important:** PTP signals are always in-band.

Configuration

- Configure Channel Recorder as usual.
- Open *%ProgramFiles%\Matrox DSX-TopologyUtils\System64\Axxxxxx.json* with any text editor, where *Axxxxxx* is the serial number of your installed Matrox IP card.
- Edit the mandatory entries:
    - *enabled* must be set to `true` to enable NMOS functionality.
    - *host name* and *port* in the section *fallback registration server* must point to the RDS (Registration & Discovery Server) for manual configuration.
    - *use service discovery* must be set to `true` for automatic discovery of the RDS.

    It is recommended to set the Control IP and port to the values shown in the following example:

> ℹ️ **Example:**
> ```
> "enabled": true,
> "local host name": "0.0.0.0",
> "local port": 8080,
> "fallback registration server": {
>     "host name": "10.1.1.240",
>     "port": 5432,
>     "api version": "auto"
> },
> "use service discovery": true,
> "use dns-sd unicast only": false,
> "heartbeat interval": 5,
> "use secure communication": false,
> ```

- Edit all the labels, so that they can easily be identified in control applications or NMOS explorers (optional).

- **Save** the file and restart either the machine or *mvNetworkService* from Windows services.

4.1.6 Persistence Configuration

It is possible to have the configuration and recordings persist through service restart or crash. This feature is controlled by the following configuration:

With persistence configuration enabled:

- During startup, if a backup file does not already exist, one is created containing the parameters of the current configuration.
- A new back up file is created every time there is a change in the **runtime** configuration.
- The changes in the **runtime** configuration can come from commands or NMOS requests.
- If there is a crash or the service is stopped, next time the service is started, **the back up file is used instead of the usual configuration file**.
- The back up file is located in *%ProgramData%\vizrt\ServiceHost*.
- The back up file name contains a timestamp which indicates when it was created (e.g. *VizrtCRTest_20200909152851194.xml*).
- Since the back up file is used as configuration it is not possible to change *some* of the configuration via GUI anymore.
- To disable persistence you can still use the GUI. The GUI can still be used to configure the connector and output.

> ⚠ **Note:** For IP workflow this feature can replace the auto-recovery option found in Viz Engine.

4.1.7 Example Configuration File

You can manually change the configuration, without using the web interface as described in Channel Recorder Configuration.

The configuration file is located in *%ProgramData%\vizrt\ServiceHost*.

The name of the configuration file is the **Service Name** that has been used during the registration of the instance.

> ❌ **Warning:** Manual manipulation of the configuration file is prone to errors. For this reason, you are strongly advised not to do it. Whenever possible, use the web interface provided by Service Host.

```xml
<?xml version="1.0"?>
<payload model="model.xml" xmlns="http://www.vizrt.com/types">
    <field name="plugin-config">
        <field name="2059-settings">
            <field name="best-master-selection">
                <value>1</value>
            </field>
        </field>
        <field name="2110-input-anc-redundancy-settings">
            <field name="filter-list" />
            <field name="filter-type">
                <value>2</value>
            </field>
            <field name="ipv4address">
                <value>239.1.10.14</value>
            </field>
            <field name="packet-interval-threshold">
                <value>819.2</value>
            </field>
            <field name="port">
                <value>50006</value>
            </field>
        </field>
        <field name="2110-input-anc-settings">
            <field name="anc-connector">
                <value>0</value>
            </field>
            <field name="enable">
                <value>1</value>
            </field>
            <field name="filter-list" />
            <field name="filter-type">
                <value>2</value>
            </field>
            <field name="ipv4address">
                <value>239.0.10.14</value>
            </field>
            <field name="packet-interval-threshold">
                <value>819.2</value>
```

```xml
            </field>
            <field name="port">
                <value>50006</value>
            </field>
            <field name="rtp-payload-id">
                <value>100</value>
            </field>
        </field>
        <field name="2110-input-audio-redundancy-settings">
            <field name="filter-list" />
            <field name="filter-type">
                <value>2</value>
            </field>
            <field name="ipv4address">
                <value>239.1.10.12</value>
            </field>
            <field name="packet-interval-threshold">
                <value>102.4</value>
            </field>
            <field name="port">
                <value>50004</value>
            </field>
        </field>
        <field name="2110-input-audio-settings">
            <field name="audio-connector">
                <value>0</value>
            </field>
            <field name="enable">
                <value>1</value>
            </field>
            <field name="filter-list" />
            <field name="filter-type">
                <value>2</value>
            </field>
            <field name="ipv4address">
                <value>239.0.10.12</value>
            </field>
            <field name="packet-duration">
                <value>0</value>
            </field>
            <field name="packet-interval-threshold">
                <value>102.4</value>
            </field>
            <field name="port">
                <value>50004</value>
            </field>
            <field name="rtp-payload-id">
                <value>97</value>
            </field>
            <field name="track-count">
                <value>8</value>
            </field>
            <field name="use-track-count">
```

```xml
                    <value>0</value>
                </field>
        </field>
        <field name="2110-input-general-settings">
            <field name="join-type">
                <value>1</value>
            </field>
            <field name="redundancy">
                <value>1</value>
            </field>
        </field>
        <field name="2110-input-video-redundancy-settings">
            <field name="filter-list" />
            <field name="filter-type">
                <value>2</value>
            </field>
            <field name="ipv4address">
                <value>239.1.10.10</value>
            </field>
            <field name="packet-interval-threshold">
                <value>6.4</value>
            </field>
            <field name="port">
                <value>50002</value>
            </field>
        </field>
        <field name="2110-input-video-settings">
            <field name="enable">
                <value>1</value>
            </field>
            <field name="filter-list" />
            <field name="filter-type">
                <value>2</value>
            </field>
            <field name="ipv4address">
                <value>239.0.10.10</value>
            </field>
            <field name="packet-interval-threshold">
                <value>6.4</value>
            </field>
            <field name="port">
                <value>50002</value>
            </field>
            <field name="rtp-payload-id">
                <value>96</value>
            </field>
            <field name="video-connector">
                <value>0</value>
            </field>
        </field>
        <field name="board-settings">
            <field name="serial-number">
                <value>A649930</value>
```

```xml
            </field>
        </field>
        <field name="capture-settings">
            <field name="alias-node-source">
                <value>INPUT</value>
            </field>
            <field name="clip-root">
                <value>D:\Clips</value>
            </field>
            <field name="colorimetry">
                <value>1</value>
            </field>
            <field name="m264-core-index">
                <value>0</value>
            </field>
            <field name="multi-connector-mode">
                <value>0</value>
            </field>
            <field name="node-audio-channels">
                <value>2</value>
            </field>
            <field name="resolution">
                <value>1080i25</value>
            </field>
            <field name="resolution-detection">
                <value>0</value>
            </field>
            <field name="restarting">
                <value>0</value>
            </field>
            <field name="restarting-delay">
                <value>3000</value>
            </field>
            <field name="sd-aspect-ratio">
                <value>4_3</value>
            </field>
            <field name="uhd">
                <value>0</value>
            </field>
            <field name="v210">
                <value>0</value>
            </field>
            <field name="vbi">
                <value>1</value>
            </field>
        </field>
        <field name="clip-settings">
            <field name="audio-channels">
                <value>2</value>
            </field>
            <field name="bitrate">
                <value>0</value>
            </field>
```

```xml
        <field name="codec">
            <value>XDCam</value>
        </field>
        <field name="container">
            <value>XDCAMMXF</value>
        </field>
        <field name="disk-access-size">
            <value>4MiB</value>
        </field>
        <field name="duration">
            <value>0</value>
        </field>
        <field name="file-extension">
            <value>0</value>
        </field>
        <field name="hqxdcamhd422">
            <value>1</value>
        </field>
        <field name="in" />
        <field name="out" />
        <field name="starttc" />
        <field name="stop-at-end">
            <value>1</value>
        </field>
        <field name="tdir">
            <value>1</value>
        </field>
        <field name="tdir-duration-predefined">
            <value>0</value>
        </field>
        <field name="tdir-interval">
            <value>0.001</value>
        </field>
        <field name="update-mt-enable">
            <value>0</value>
        </field>
        <field name="vbr">
            <value>0</value>
        </field>
    </field>
</field>
<field name="clock-settings">
    <field name="clock-type">
        <value>INPUT</value>
    </field>
</field>
<field name="communication-settings">
    <field name="command-handler-port">
        <value>6810</value>
    </field>
    <field name="mvcp-port">
        <value>52501</value>
    </field>
</field>
```

```
<field name="input-settings">
    <field name="input">
        <value>VideoIn</value>
    </field>
</field>
<field name="license">
    <field name="4K-video-enablement">
        <value>0</value>
    </field>
    <field name="container-serial-and-core-license-string">
        <value>0-0000000-ENG_SVCHOST_CORE-1</value>
    </field>
    <field name="dnxhd-enablement">
        <value>0</value>
    </field>
    <field name="license">
        <value>1200200</value>
    </field>
</field>
<field name="log-settings">
    <field name="timecode-log-interval">
        <value>0</value>
    </field>
</field>
<field name="manager-settings">
    <field name="ring-buffer">
        <value>60</value>
    </field>
    <field name="timeout">
        <value>5000</value>
    </field>
</field>
<field name="output-settings">
    <field name="output">
        <value>None</value>
    </field>
</field>
<field name="persistence-settings">
    <field name="persistence">
        <value>0</value>
    </field>
    <field name="persistence-interval">
        <value>0.0</value>
    </field>
</field>
<field name="process-settings">
    <field name="priority">
        <value>REALTIME</value>
    </field>
</field>
<field name="scheduler-settings">
    <field name="discontinuous-tc">
        <value>0</value>
```

```xml
        </field>
        <field name="inclusive-out">
            <value>0</value>
        </field>
        <field name="timecode-source">
            <value>VITC</value>
        </field>
        <field name="trigger-threshold">
            <value>5</value>
        </field>
        <field name="writers">
            <value>2</value>
        </field>
    </field>
    <field name="sfp-a-2059-settings">
        <field name="announce-receipt-timeout">
            <value>6</value>
        </field>
        <field name="delay-mechanism">
            <value>1</value>
        </field>
        <field name="dhcp">
            <value>1</value>
        </field>
        <field name="filter-list" />
        <field name="filter-type">
            <value>2</value>
        </field>
        <field name="ip-mode">
            <value>1</value>
        </field>
        <field name="join-type">
            <value>1</value>
        </field>
        <field name="master-clock-domain-number">
            <value>127</value>
        </field>
        <field name="type-of-service-dscp">
            <value>0</value>
        </field>
    </field>
    <field name="sfp-a-general-settings">
        <field name="ipv4address">
            <value>10.0.0.2</value>
        </field>
        <field name="ipv4gateway">
            <value>10.0.0.1</value>
        </field>
        <field name="ipv4netmask">
            <value>255.255.255.0</value>
        </field>
    </field>
    <field name="sfp-b-2059-settings">
```

```xml
        <field name="announce-receipt-timeout">
            <value>6</value>
        </field>
        <field name="delay-mechanism">
            <value>1</value>
        </field>
        <field name="dhcp">
            <value>1</value>
        </field>
        <field name="filter-list" />
        <field name="filter-type">
            <value>2</value>
        </field>
        <field name="ip-mode">
            <value>1</value>
        </field>
        <field name="join-type">
            <value>1</value>
        </field>
        <field name="master-clock-domain-number">
            <value>127</value>
        </field>
        <field name="type-of-service-dscp">
            <value>0</value>
        </field>
    </field>
    <field name="sfp-b-general-settings">
        <field name="ipv4address">
            <value>10.0.0.3</value>
        </field>
        <field name="ipv4gateway">
            <value>10.0.0.1</value>
        </field>
        <field name="ipv4netmask">
            <value>255.255.255.0</value>
        </field>
    </field>
    <field name="sfp-c-2059-settings">
        <field name="announce-receipt-timeout">
            <value>6</value>
        </field>
        <field name="delay-mechanism">
            <value>1</value>
        </field>
        <field name="dhcp">
            <value>1</value>
        </field>
        <field name="filter-list" />
        <field name="filter-type">
            <value>2</value>
        </field>
        <field name="ip-mode">
            <value>1</value>
```

```xml
        </field>
        <field name="join-type">
            <value>1</value>
        </field>
        <field name="master-clock-domain-number">
            <value>127</value>
        </field>
        <field name="type-of-service-dscp">
            <value>0</value>
        </field>
    </field>
    <field name="sfp-c-general-settings">
        <field name="ipv4address">
            <value>10.0.0.4</value>
        </field>
        <field name="ipv4gateway">
            <value>10.0.0.1</value>
        </field>
        <field name="ipv4netmask">
            <value>255.255.255.0</value>
        </field>
    </field>
    <field name="sfp-d-2059-settings">
        <field name="announce-receipt-timeout">
            <value>6</value>
        </field>
        <field name="delay-mechanism">
            <value>1</value>
        </field>
        <field name="dhcp">
            <value>1</value>
        </field>
        <field name="filter-list" />
        <field name="filter-type">
            <value>2</value>
        </field>
        <field name="ip-mode">
            <value>1</value>
        </field>
        <field name="join-type">
            <value>1</value>
        </field>
        <field name="master-clock-domain-number">
            <value>127</value>
        </field>
        <field name="type-of-service-dscp">
            <value>0</value>
        </field>
    </field>
    <field name="sfp-d-general-settings">
        <field name="ipv4address">
            <value>10.0.0.5</value>
        </field>
```

```xml
            <field name="ipv4gateway">
                <value>10.0.0.1</value>
            </field>
            <field name="ipv4netmask">
                <value>255.255.255.0</value>
            </field>
        </field>
        <field name="stream-input-settings">
            <field name="ipaddress">
                <value>rtsp://localhost:8554/stream1</value>
            </field>
            <field name="nicaddress">
                <value>0.0.0.0</value>
            </field>
            <field name="port">
                <value>0</value>
            </field>
            <field name="use-av-synchronization">
                <value>1</value>
            </field>
            <field name="use-software-decoding">
                <value>0</value>
            </field>
        </field>
        <field name="test-settings">
            <field name="performance-test-mode">
                <value>NONE</value>
            </field>
        </field>
        <field name="video-input-settings">
            <field name="connector">
                <value>1</value>
            </field>
        </field>
        <field name="video-output-settings">
            <field name="connector">
                <value>1</value>
            </field>
        </field>
    </field>
    <field name="service-host">
        <field name="plugin">
            <field name="plugin-arguments" />
            <field name="plugin-name">
                <value>ChannelRecorder</value>
            </field>
        </field>
    </field>
</payload>
```

4.1.8   Integration with Other Services

Channel Recorder can be integrated with other services.

- Ingest
- Viz Libero
- Viz Virtual Studio (Tracking Hub)
- Viz Dart

Ingest

Ingest is a tool for scheduling media recordings and can utilize Channel Recorder as a target server. In the timeline part of Ingest it's possible to do both scheduled recordings to be started in the future and to start immediate crash recordings. It utilizes the Channel Recorder Control Commands to control the Channel Recorder. In the Ingest Configuration, admins can easily setup new channel recorders to be controlled by Ingest. It expects exclusivity use so no other integration may be used in conjunction with Ingest. In the configuration it's also possible to configure routers and sources and connect these to the channel recorders to reflect the physical connections.

Viz Libero

Channel Recorder is being used by Viz Libero as a real-time recording unit for sport analysis. It is supported in both the SDI and the IP workflow.

Viz Virtual Studio (Tracking Hub)

To embed tracking data into pre-recorded clips, Channel Recorder can be used from within Tracking Hub. For more information please see the Viz Virtual Studio documentation.

Viz Dart

The Viz Dart video acquisition tool can be configured to acquire assets using Channel Recorder. Integration of Channel Recorder with Viz Dart is based on the MLT Video Control Protocol, or MVCP. MVCP is by default activated on port `10000`. To change the port, refer to Channel Recorder Configuration.

When operating Viz Dart with Channel Recorder, the timing between the different components is important. Take the following example:
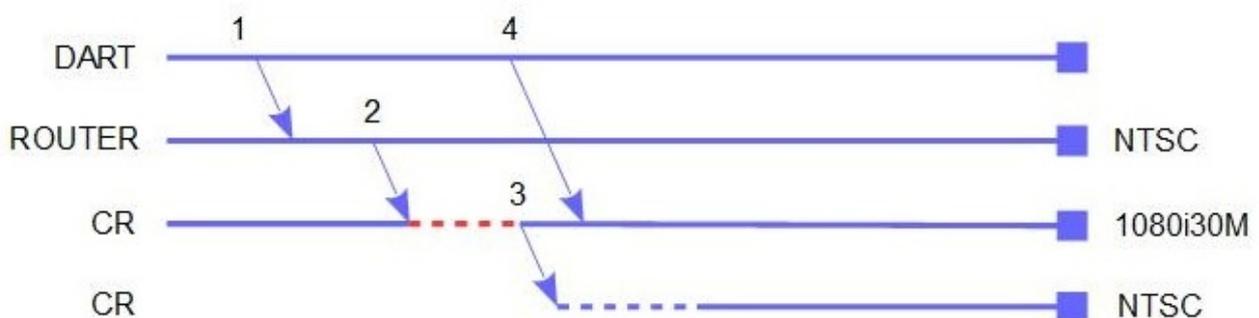
The typical operation when using Viz Dart is as follows:

1. Dart sends a command to the router to route an NTSC signal to Channel Recorder.
2. The router sends the NTSC signal to Channel Recorder.
3. Channel Recorder was configured as 1080i30M, so it triggers a configurable timeout (dotted red line).
    a. By default, this timeout is three seconds duration and can be configured (restarting_delay).
    b. After the timeout, Channel Recorder restarts with NTSC signal resolution.
    c. If the same signal as configured (1080i30M) is received again within those three seconds, the restart is cancelled.
4. During the restart (dotted blue line) it is possible that Dart sends the MVCP commands to schedule a recording. If this happens, Channel Recorder queues the commands until it is ready with the new resolution.

There are other several situations (this is not meant to be an exhaustive list, but rather some examples):
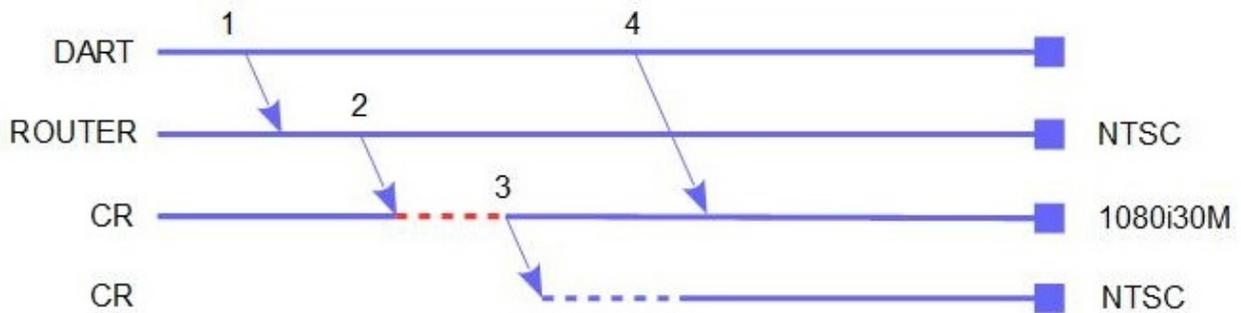
In this case, the MVCP commands are sent after the restart has already finished.
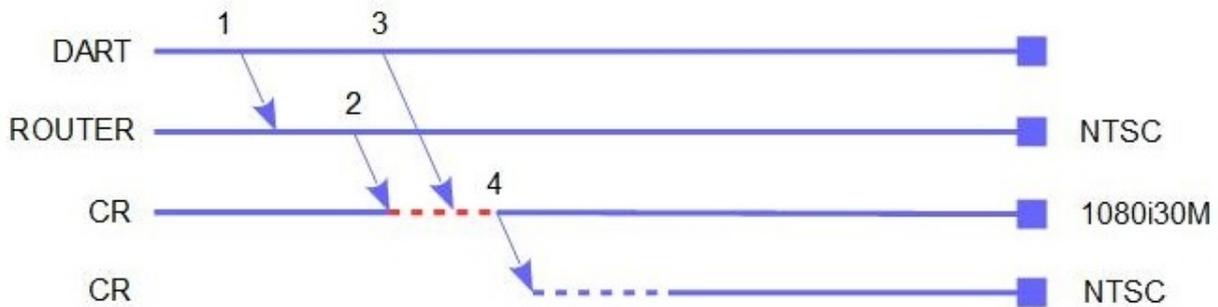
In this case, the MVCP commands are sent before the restart occurred. When the restart finishes Channel Recorder recovers the timeline created before the restart.

Another important thing to consider is when Dart is configured with *trigger_duration_out: 0.* This configuration means that Dart does not send */SEQA SET UX MED vtr.media.input.trigger.duration.out* to Channel Recorder and instead manually stops the recording by sending *STOP UX.*

One of the common issue with this configuration is that Channel Recorder is stopped unexpectedly (Dart reports this as "REC Unexpected DONE or BUSY state"). This is because Channel Recorder has received */SEQA SET UX MED vtr.media.input.trigger.duration.out* before and is using the duration for all other recordings. So recordings that are longer than the specified duration are now going to be failing because Channel Recorder stop them at the configured duration instead of when Dart send the *STOP UX* command. One thing to note is that this duration is persistent among recordings.

Viz Dart also offers some configuration options, such as when the MVCP commands should be sent (*prestarttime*). To see the other options, use the following command on the Viz Dart machine: `/opt/dart/bin/editconf -ey dartsettings`.

For further information on how to configure Viz Dart, or how to operate it on a Video Disk Recorder, refer to the Viz Dart Documentation.

4.1.9   Initialization Errors

The following errors can occur during initialization. For more information regarding the error it is always good to check the log. The ID associated with the error is displayed in the web interface when starting the instance failed.

Configuration

- **100: Connector is invalid.** Please specify a valid connector in the configuration.
- **101: Path for clip is invalid.** Please specify a valid path in the configuration.
- **102: Failed to load configuration.** Please check that the configuration file exists and is not corrupted.

Board

- **200: Installed Matrox DSX.utils version is not compatible.** Please install the correct version. Please check the log for more information.
- **201: Could not use specified connector.** Please check the connector is not already used by another application or instance.
- **202: Matrox DSX.utils is not found.** Please check the log for more information.
- **203: No Matrox board found.**
- **204: Matrox board initialization failed.** Please check the log for more information.
- **205: Matrox board does not have the required codec license.** Please check the log for more information.

License

- **300: Could not instantiate WIBU License.** Please install CodeMeter Control Center.
- **301: Could not allocate one of the requested license.** Please check the log for more information.
- **302: Unknown license requested.** Please check the log for more information.
- **303: Issue found in license configuration.** Please check the log for more information.
- **304: License provider failed due to unknown reason.** Please check the log for more information.

Miscellaneous

- **400: Could not initialize due to unknown error.** Please check the log for more information.
- **500: Failed to initialize spdlog.** Please check the log for more information.

4.1.10   Upgrading from 1.1 and Later

This process should be handled automatically during the upgrade process. However, if the automatic upgrade fails, the user will not be able to configure or start the service. In this case, the user must:

1. Create a new service with a different name than the old one, using the new installation to generate a new default configuration.

⚠ **Note:** Both the new and old configurations are located at *C:\ProgramData\vizrt\ServiceHost*.

2. Manually copy the relevant configuration parameters from the old configuration to the new one using a text or XML editor.
3. The old configuration can be removed and the new configuration can be renamed as the old one. Now the service can be configured and started via the web interface.

Please refer to the Release Notes about the required Matrox driver.

4.1.11 Channel Recorder Configuration

You can configure Channel Recorder at any time after successfully registering an instance. For any configuration changes to take effect, you must restart the instance. Every plug-in of Service Host has its own configuration page. For more information, refer to the Service Host Administrator Guide documentation.

Every plug-in configuration page consist of two main sections:

- Plug-in Configuration Section
- Service Host Section

The Plug-in Configuration Section is unique for each plug-in. From the Service Host Section, you can set specific command arguments to the plug-in. This is useful mainly for debugging purposes.

Plug-in Configuration Section

The Plug-in Configuration Section consists of the following parts:

- **Communication Settings**
  - **Command Handler Port:** Selects the port used to communicate with Channel Recorder via **Viz Send**. The default value is `6810` .
  - **MVCP Port:** Selects the port used to communicate with Channel Recorder using MVCP. Some Vizrt components, such as Ingest, Dart or Capture, use this protocol to communicate with Channel Recorder. The default value is `10000` , while for example, Ingest uses `5250` as default for its first channel and subsequent port numbers for any additional channels.

> ✅ **Tip:** If several instances of Channel Recorder are present in the machine, the ports must be different for each.

- **Board Settings** (changes to these settings only take effect by restarting the Channel Recorder instance).
  - **Serial Number:** Selects the board with the specified serial number. If no serial number is specified, the first detected board is selected. By default, no value is specified.

> ✅ **Tip:** The serial number on Matrox video boards can be found in the hardware tab of **Matrox** X.info, it usually begins with an A.

- **Input Settings**
  - **Input:** Selects the type of input to use. At the moment, only Video input is possible.

- **Video Input Settings**
  - **Connector:** Selects the connector to be used for recording live input. This is a required value. The default value is *A*. Accepted values are `A` to `P` or numerical values `1 to 32`, depending on the input configuration of the Matrox board. For IP 2110 boards to be able to address all available inputs, a numerical value is needed. Channel Recorder does not follow the Matrox input naming convention. This means that the first input corresponds to connector A, the second input corresponds to connector B (whereas the second input could correspond to SDI Input Connector C for Matrox, depending on the board connector configuration), etc.

- **Stream Input Settings**
  - **IP Address:** Indicates the source IP address.
  - **NIC Address:** Indicates which NIC is used for receiving the stream.
  - **Port:** Indicates the source port.
  - **Use Software Decoding:** Chooses between using the hardware or software decoder.
  - **Use AV Synchronization:** Enables audio and video synchronization of both are present. If only video is present this option must be disabled.
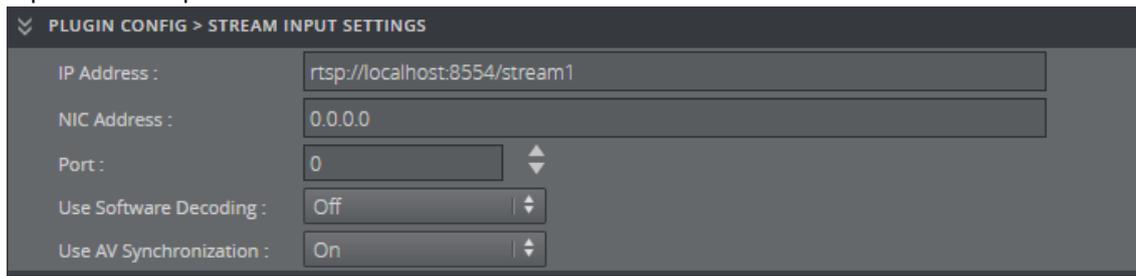
- **Process Settings**
  - **Priority:** Sets the process priority class. The values correspond to the Windows process priority levels. The default value is `Normal`.

- **Capture Settings**
  - **Clip Root:** Sets the default folder for the recordings. The default value is `D:\\cr_clip`.
  - **UHD:** Enables detection of UHDTV signals. When set to `On`, Channel Recorder scans the signal resolution on the four corresponding input connectors. If four 3G signals are detected, they are interpreted as one UHDTV signal. When set to `Off`, the four connectors are treated as separate 3G signals. The default value is `Off`.
  - **V210:** Uses the 10-bit surface format V210. This is needed to record XAVC, it increases performance when for example recording ProRes.
  - **Colorimetry:** Specifies the colorimetry to be used to be able to record in HDR. By default `ITUR_BT_601` is used for SD resolution and `ITUR_BT_709` for HD and UHD resolution.

- **VBI:** Enables VBI recording. To capture closed caption this value must be set to `On` . The default value is `On` .
- **Resolution:** Sets the default resolution. The possible values are: `NTSC` , `PAL` , `720p50` , `720p60M` , `1080i25` , `1080p60` , `1080i30M` , `1080i30` , `1080p50` , `1080p60M` and `1080p60` . The default value is `NTSC` .
- **Multi Connector Mode:** Allows switching between 12G and different quad-link settings for UHD inputs. Only supported for Xmio5 12G SDI.
- **SD Aspect Ratio:** Specifies the aspect ratio for SD resolution.
- **Alias Node Source:** Specifies the source of the alias node which specify from which output or input connector the application records.
- **Restarting Delay:** Specifies the time in milliseconds before the channel is restarted after detecting a change in resolution.
- **Restarting:** Enables or disables restarting after detecting a change in resolution.
- **Resolution Detection:** Enables or disables resolution detection.
- **M264 Core Index:** Chooses which M264 core to use when the hardware is available.
- **Node Audio Channels:** Number of audio channels for Matrox input node. This is relevant for IP workflow. This must match the number of audio channels of the audio output flow.
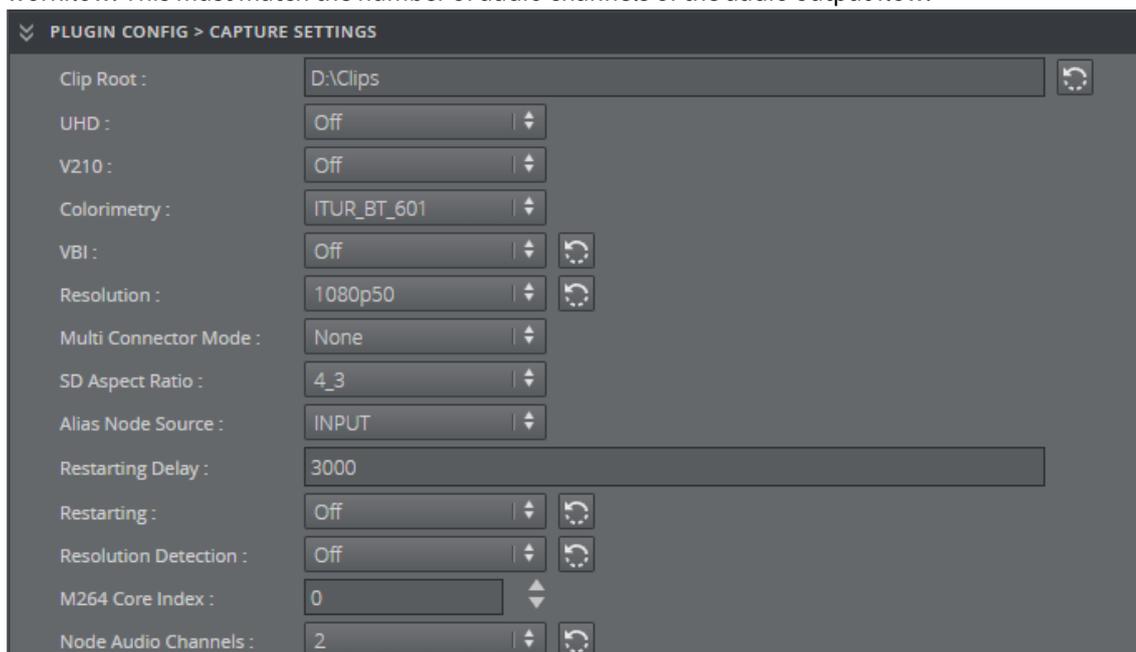
- **Test Settings**
  - **Performance Test Mode:** Selects the mode when a performance test is to be performed. The values are `GENERATE` and `CONSUME` . `GENERATE` generates a RAW file using the input signal. `CONSUME` reads the generated file **as fast as possible** and send it for encoding. While using `CONSUME` mode calling `RECORD PERFORMANCE GET` gives you the maximum frame rate, which gives you how many recording can be run in the machine taking into account **only** the CPU usage. For example, if the value is 280 FPS using as input 720p, the machine can only do five recordings at 720p50.
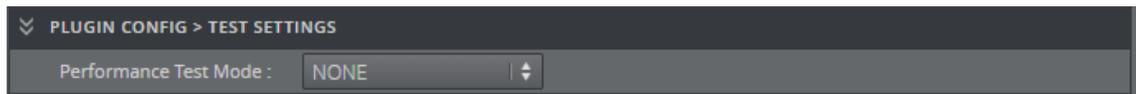
- **Manager Settings**
  - **Ring Buffer:** Sets the size of the capture ringbuffer. The default value is `60`.
  - **Timeout:** Sets the timeout for the capture operation in milliseconds. If the recorder reports timeout errors, increasing the timeout could help.
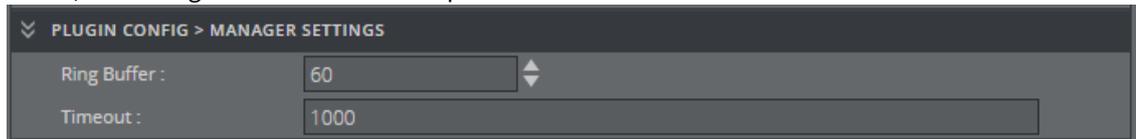
- **Persistence Settings**
  - **Persistence:** Chooses if a backup of the scheduled recording is needed. This is only useful for scheduled recordings. The default value is `Off`.
  - **Persistence Interval:** Defines, in seconds, the interval in which the backup is written to disk. The default value is `0`.
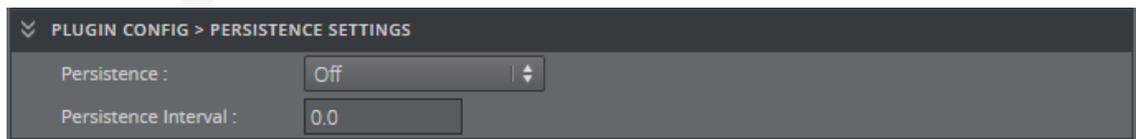
- **Log Settings**
  - **Timecode Log Interval:** Specifies the interval at which the current timecode is logged. The value can either be a number of frames or a timecode-based relative value. The default value is `0`, which means that every full second is logged.

- **Scheduler Settings**
  - **Writers:** Sets the number of writers to initialize. This value is only relevant in loop and scheduled recordings in which consecutive recordings are very close to each other within the timeline (less than six seconds). The default value is `2` and should never be below `2`.
  - **Trigger Threshold:** Determines whether to execute a timed command late if it misses the execution time. If the time is still within the trigger threshold, it is executed late. Outside of this window, the command is ignored until the next time the timecode is received. The value can either be a number of frames or a timecode-based relative value. The default value is `5`.
  - **Timecode Source:** Specifies the timecode source. The default value is `TIME_OF_DAY`.
  - **Inclusive Out:** When enabled the last timecode written to the file is the defined out point of the recording.
  - **Discontinuous TC:** When enabled each field/frame uses the timecode calculated from the system time in case of TIME_OF_DAY and VITC/LTC embedded in the fields/frames if this is chosen as timecode source. This option is useful to support DST (Daylight Saving Time) or when the input's timecode is expected to have gaps and this must be kept in the recorded file. StartTC will not work when this is enabled.
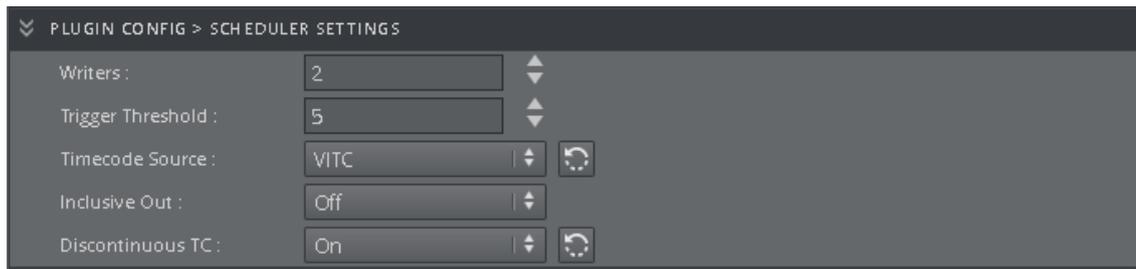
- **Clip Settings** (these are the default values for any future recordings)
  - **File Extensions:** Enables or disables automatically adding a file extension to the file name. If this feature is turned off, the client application has full control over the file name. The default value is `Off`.
  - **Container:** Sets the container type of the recorded file. Possible values are `AVCINTRAMXF`, `AVI`, `DVCPROMXF`, `MOV`, `MXF`, `XAVCMXF`, `XDCAMMXF`. For capturing OP-Atom the value should be `MXF`. The default value is `XDCAMMXF`.
  - **Codec:** Sets the codec type of the recorded file. The possible values are: `DvCam`, `DvCPro`, `Dv50`, `IFrame`, `XDCam`, `AVCIntra50`, `AVCIntra100`, `ProRes`. The default value is `XDCam` which corresponds to XDCAM HD422.
  - **Bitrate:** Sets the bitrate for the video encoding in mbps (megabits per second). Not all codecs allow changes to the bitrate. In such cases, this value is ignored. The default value is `0`.
  - **IN:** Sets the default value for the recording in point. The value must be in timecode format: `00:00:00:00`.
  - **OUT:** Sets the default value for the recording out point. The value must be in timecode format: `00:00:00:00`.
  - **Duration:** Sets the default duration of the recording. The default value is `0`.
  - **StartTC:** Sets the default value for the starting timecode. The value must be in timecode format: `00:00:00:00`.
  - **Audio Channels:** Sets the number of audio channels to record. How many channels are actually recorded depends on the codec and the input signal. The default value is `8`.
  - **Disk Access Size:** Sets the size of data blocks written to the disk in bytes. Postfixes like KiB, Kb, k, etc., are allowed, but must not be separated from the value with a blank space. The default value is `4MiB` (1*4194304 bytes). The minimum value is `32KiB` (32768 bytes).
  - **VBR:** Sets whether the recorded file, when the format supports it, should be in VBR (Variable Bit Rate).
  - **TDIR:** Sets the default behavior of TDIR (Time Delayed Instant Replay). The default value is `On`.
  - **TDIR Interval:** Sets the interval of file header updates in TDIR recordings. The value is in seconds and fractions of seconds, meaning both `11.1` and `11.2` are considered valid values. Minimum allowed value is `10.0`, which is interpreted by Channel Recorder as every frame. The maximum value is `60.0`. The default value is `10.0`.
  - **TDIR Duration Predefined:** Sets the TDIR duration to -1 while the file is growing, when enabled. The default value is `Off`.

- **Update Modification Time:** Updates the modification time of the recorded clip regardless of TDIR setting value. The default value is `On` .
- **Stop At End:** Sets the default value at the end of the recording. For loop and scheduled recordings this value should be `Off` . The default value is `Off` .
- **HQ XDCAMHD422:** High Quality XDCAMHD422. Gives better quality at the expense of higher CPU usage. The default value is `On` .

> ✅ **Tip:** Some formats have specific audio support, so even if the user specifies a higher or lower value, it always record the same number of channels. Please refer to the documentation for each format.

- **Clock Settings**
  - **Clock Type:** Defines what type of clock is used for recording. The possible values are: `GENLOCK` and `INPUT` . The default value is `INPUT` .
  - **Genlock Flywheel:** Uses the flywheel in case the genlock signal is lost if Clock Type is `GENLOCK` .
  - **Flywheel Timeout:** Sets the timeout of the genlock flywheel in seconds. This defines the time until the genlock switches to free run, as well as the maximum time the flywheel can use to resynchronize. The default value is `5.0` .

> ✅ **Tip:** When recording without a signal, `GENLOCK` clock type is recommended due to the drift that `INPUT` clock type can cause. `INPUT` clock type is generally the better choice when a signal is present all the time while recording.

- **Output Settings**
  - **Output:** Specifies the output mode. The possible values are: `VIDEO`, `NONE`. The default value is `NONE`.

> ✅ **Tip:** Only one type of output can be configured at startup using the web interface. However, after startup, it is still possible to configure another output via Viz Send.

- **Video Output Settings**
  - **Connector:** Selects the connector to be used for outputting the recorded frames. Accepted values are `A` to `H`, depending on the output configuration of the Matrox board.

- **2110 Input General Settings**
  - **Redundancy:** Enables redundancy if `true`.
  - **Join Type:** Indicates the type of membership request made when IPv4 address is a multicast address.

- **2110 Input Video Settings**
  - **Enable:** Enables video flow when set to `True`.
  - **Connector:** Specifies the video input connector to be used for recording. If set to 0, the connector defined in Video Input Settings is used instead.
  - **IPv4 Address:** Sets destination IPv4 address.
  - **Port:** Sets destination port.
  - **RTP Payload ID:** Indicates the RTP (Real-time Transfer Protocol) Payload ID to capture. Used when RTP Payload ID is set to `True`.
  - **Packet Interval Threshold:** Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For video, the range is from `6.4` nanoseconds to `419424.0` nanoseconds, in intervals of `6.4` nanoseconds.
  - **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
  - **Filter List:** Lists IPv4 addresses to be included or excluded, separated by a space character.

- **2110 Input Video Redundancy Settings**
  - **IPv4 Address:** Sets destination IPv4 address.
  - **Port:** Sets destination port.
  - **Packet Interval Threshold:** Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For video, the range is from `6.4` nanoseconds to `419424.0` nanoseconds, in intervals of `6.4` nanoseconds.
  - **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
  - **Filter List:** Lists IPv4 addresses to be included or excluded, separated by a space character.

- **2110 Input Audio Settings**
  - **Enable:** Enables audio flow when set to `True`.
  - **Connector:** Specifies the audio input connector to be used for recording. If set to 0, the connector defined in Video Input Settings is used instead.
  - **IPv4 Address:** Sets destination IPv4 address.
  - **Port:** Sets destination port.
  - **RTP Payload ID:** Indicates the RTP (Real-time Transfer Protocol) Payload ID to capture. Used when RTP Payload ID is set to `True`.
  - **Packet Interval Threshold:** Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For audio, the range is from `102.4` nanoseconds to `6710681.6` nanoseconds in intervals of `102.4` nanoseconds.
  - **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
  - **Filter List:** Lists IPv4 addresses to be included or excluded separated by a space character.
  - **Use Track Count:** Uses track count to specify the number of tracks for the incoming IP stream when set to `True`.
  - **Track Count:** Specifies the number of tracks for the incoming IP stream.

- **Packet Duration:** Indicates the duration of the incoming audio packet. The packet size (in samples) can be computed using the sample rate (`48000` samples/sec) and the packet duration.

- **2110 Input Audio Redundancy Settings**
  - **IPv4 Address:** Sets destination IPv4 address.
  - **Port:** Sets destination port.
  - **Packet Interval Threshold:** Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For audio, the range is from `102.4` nanoseconds to `6710681.6` nanoseconds in intervals of `102.4` nanoseconds.
  - **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
  - **Filter List:** Lists IPv4 addresses to be included or excluded separated by a space character.

- **2110 Input ANC Settings**
  - **Enable:** Enables ANC flow when set to `True`.
  - **Connector:** Specifies the anc input connector to be used for recording. If set to 0, the connector defined in Video Input Settings is used instead.
  - **IPv4 Address:** Sets destination IPv4 address.
  - **Port:** Sets destination port.
  - **RTP Payload ID:** Indicates the RTP (Real-time Transfer Protocol) Payload ID to capture. Used when RTP Payload ID is set to `True`.
  - **Packet Interval Threshold:** Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For ancillary data, the range is from `819.2` nanoseconds to `53685452.8` nanoseconds in intervals of `819.2` nanoseconds.

- **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
- **Filter List:** Lists IPv4 addresses to be included or excluded separated by a space character.

- **2110 Input ANC Redundancy Settings**
  - **IPv4 Address:** Sets destination IPv4 address.
  - **Port:** Sets destination port.
  - **Packet Interval Threshold:** Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For audio, the range is from `102.4` nanoseconds to `6710681.6` nanoseconds in intervals of `102.4` nanoseconds.
  - **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
  - **Filter List:** Lists IPv4 addresses to be included or excluded separated by a space character.
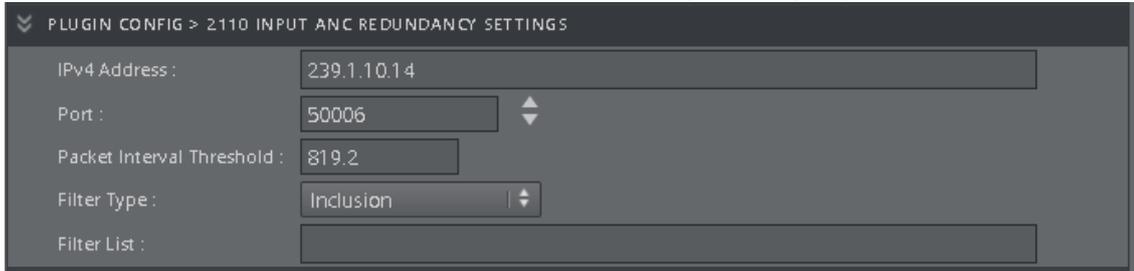
- **SFP A / B / C / D General Settings**
  - **IPv4 Address:** Sets IPv4 address.
  - **IPv4 Gateway:** Sets IPv4 gateway.
  - **IPv4 Netmask:** Sets IPv4 netmask.

- **SFP A / B / C / D 2059 Settings**
  - **Enable DHCP:** Enables DHCP protocol to automatically configure the IPv4 address.
  - **Type of Service DSCP:** Specifies a datagram's priority and requests a route for low-delay, high-throughput, or highly-reliable service.
  - **Delay Mechanism:** Specifies the type of network delay mechanism to use for the time server connection.

- **IP Mode:** Specifies the type of internet protocol mode to use for the time server connection.
- **Master Clock Domain Number:** Specifies the master clock domain number.
- **Announce Receipt Timeout:** Specifies the announce receipt timeout.
- **Join Type:** Indicates the type of membership request made when IPv4 address is a multicast address.
- **Filter Type:** Specifies the type of filtering applied to the source list when IGMPv3 is used.
- **Filter List:** Lists IPv4 addresses to be included or excluded separated by a space character.
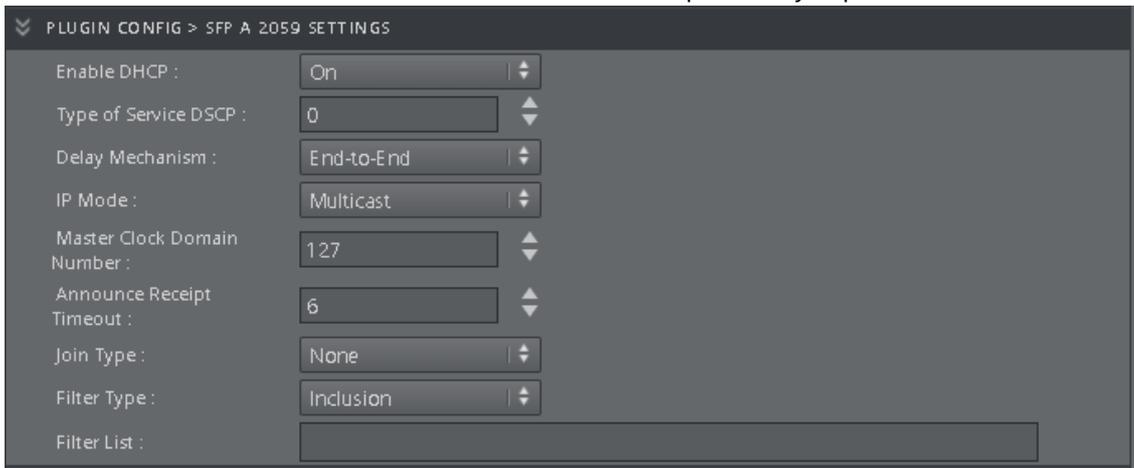
- **2059 Settings**
    - **Best Master Selection:** Specifies the BMCA (Best Master Clock Algorithm) used to select the genlock over IP signal.
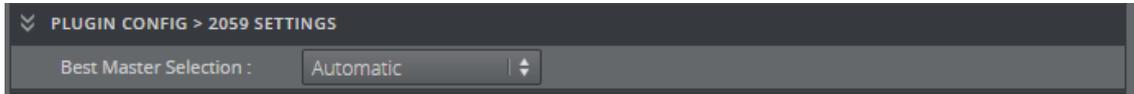
- **License Settings**
    - **Container and Core:** Defines container and core license string.
    - **License:** Defines the license type. If the use does not have SDI IN (max). Demo option can be used, however the recordings will have 100 black frames every 400 frames.
    - **4K Video Enablement:** Enables 4K video license fetching. If none is found, the user is not able to start Channel Recorder. If already recording and the license is not accessible or expired, the user will have 100 black frames every 400 frames.
    - **DNxHD Enablement:** Enables DNxHD license fetching. If none is found, the user is not able to start Channel Recorder. If already recording and the license is not accessible or expired, the user will have 100 black frames every 400 frames.

Service Host Section

Only one setting can be set in this section:

- **Arguments:** Specifies the arguments that are going to be passed to Channel Recorder.

An example of a string that can be used for enabling log level debug is: `-v -l debug -n 10 -m 50`

> ⚠ **Note:** Channel Recorder 1.2 introduces two new parameters: *n* and *m*. *n* specifies the maximum number of log files and *m* the maximum size of each log file in MB. By default, the number of log files is four and the maximum size of each log file is `10` MB.

See Also

- Channel Recorder Control Commands
- Channel Recorder Command Line Options
- Example Configuration File

4.2   Operation and Troubleshooting

You can use the application **Viz Send**, which is included in the **Viz Engine** installation, to communicate with Channel Recorder. Connect the tool to the port specified in the **Channel Recorder** configuration file.

4.2.1   Channel Recorder Initialization

You can start Channel Recorder with the Service Host web interface.

When Channel Recorder is started, the following happens:

1. User specified configuration runs in a specific order:
   a. If no serial number is specified, Channel Recorder selects the first board detected by the system.
   b. If no connector is selected, Channel Recorder fails to execute.
2. Depending on the selected clock type, one of the following can happen:
   a. If GENLOCK is selected, Channel Recorder queries for the genlock signal. Channel Recorder compares the genlock frequency to the signal frequency. The signal frequencies need to be from the same frequency family, for example, `25i` and `50p` frequencies match, but `29.97i` and `50p` do not.
   b. If INPUT is selected, Channel Recorder does not query for the genlock signal.

> ⚠ **Note:** `INPUT` clock type is only available for Matrox X.mio3 and Matrox DSX LE4.

If everything works as expected, the input channel is set up and Channel Recorder starts capturing from it. As long as no clip is recorded, the captured frames are thrown away. When a recording starts, these frames are written to the file. This is necessary to be able to provide:

1. Instant recording
2. Scheduling of recordings

4.2.2   Workflow Example

1. Start the service from the Service Host web interface. For more information, refer to the Service Host documentation.
2. Check the log in the Service Host web interface to make sure that everything has started correctly. You can also check the log file located at *%programdata%vizrt/ServiceHost/[service name]*.
3. You should not see any errors or warning messages. If there is any error or warning messages, please try to amend it and restart the service using the web interface provided by Service Host. For more information, refer to the Service Host documentation. The following is an example of a successful initial run by Channel Recorder:

```
[2018-06-29 08:23:35.031 +02:00][24300][info]: Startup
'ChannelRecorder::ChannelRecorder::Initialize' as 'VizrtCRTest'
[2018-06-29 08:23:35.031 +02:00][24300][info]: Startup
'ChannelRecorder::ChannelRecorder::Initialize' version of ServiceHost is
'1.1.0.77448.'
[2018-06-29 08:23:35.031 +02:00][24300][info]: Startup
'ChannelRecorder::ChannelRecorder::Initialize' version of ChannelRecorder is
'1.1.0.77448.'
```

```
[2018-06-29 08:23:35.046 +02:00][24300][info]: [VizrtCRTest] starting up.
thread-id: 24300
[2018-06-29 08:23:35.046 +02:00][24300][info]: [VizrtCRTest] Applying settings
from configuration file!
[2018-06-29 08:23:35.046 +02:00][24300][info]: [VizrtCRTest] Received: -1
COMMAND_HANDLER PORT SET 6810
[2018-06-29 08:23:35.046 +02:00][24300][info]: [VizrtCRTest] setting
COMMAND_HANDLER PORT=6810
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] OK
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] Received: -1 MVCP
PORT SET 10000
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] setting MVCP
PORT=10000
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] OK
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] Received: -1
CONFIG SET CLIP_ROOT D:/cr_clips
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] setting
CLIP_ROOT=D:/cr_clips
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] OK
[2018-06-29 08:23:35.047 +02:00][24300][info]: [VizrtCRTest] Received: -1 MAIN
CONNECTOR SELECT A
[2018-06-29 08:23:39.303 +02:00][24300][info]:
[VizrtCRTest::CaptureChannelSystemInA] SetInputResolution: Detected input
resolution is 1080i25
[2018-06-29 08:23:39.303 +02:00][24300][info]:
[VizrtCRTest::CaptureChannelSystemInA] DetectInputResolutionSdi: Selected
resolution is 1080i25
[2018-06-29 08:23:40.908 +02:00][24300][info]: [VizrtCRTest] Capture from
channel InA on board XMIO3/8/550 (serial: A523502)
[2018-06-29 08:23:40.909 +02:00][24300][info]: [VizrtCRTest] OK
[2018-06-29 08:23:40.909 +02:00][2916][info]:
[VizrtCRTest::CaptureChannelSystemInA] starting up
[2018-06-29 08:23:40.912 +02:00][5000][info]:
[VizrtCRTest::CaptureChannelSystemInA::ClipOutChannel0] starting up
[2018-06-29 08:23:40.912 +02:00][29676][info]:
[VizrtCRTest::CaptureChannelSystemInA::ClipOutChannel1] starting up
```

4. The most common errors are not specifying a connector, or specifying a connector that is already in use or non-existing.

When Channel Recorder is successfully running, you can control it using Viz Send, or by using Viz Dart through the MVCP protocol. Refer to Channel Recorder Control Commands for available commands.

> ⚠️ **Note:** Telnet can also be used as a way to control Channel Recorder through the MVCP protocol.

4.2.3 Change of Input Resolution

Starting with version 1.1, the Channel Recorder handles input resolution changes during run-time. When Channel Recorder detects a change in the input resolution, it resets the channel to use the new resolution, without the need to restart the service.

4.2.4   Recording with No Input Signal

Starting with version 1.1, the Channel Recorder can record without an input signal attached. When a signal is attached during recording, Channel Recorder seamlessly uses the signal. This means that the recorder file has a black signal, plus the new connected signal. This requires that the configured resolution matches the connected signal resolution, and that you select the **INPUT Clock Type** during initial configuration.

4.2.5   Drop Frame Timecode

The user should be mindful when using commands to operate Channel Recorder than involves inputting timecode in drop frame resolutions. Such commands require the input of a valid timecode. However, when a timecode is not recognized as valid, Channel Recorder attempts to fix it by rounding down the timecode to a a valid one. For example:

```
RECORD SET schedule.mxf IN=15:00:00:00 OUT=16:00:00:00
```

Both timecodes are not valid when operating in drop frame resolution. The command and timecode automatically change to:

```
RECORD SET schedule.mxf IN=15:00:00:04 OUT=16:00:00:04
```

for progressive resolution. For interlace resolution, the timecode automatically changes to:

```
RECORD SET schedule.mxf IN=15:00:00:02 OUT=16:00:00:02
```

See Also

- Channel Recorder Control Commands
- Channel Recorder Command Line Options
- Channel Recorder Configuration

4.2.6   Troubleshooting Channel Recorder

This page contains some common troubleshooting tips:

- Enabling Debug Logging
- Channel Recorder and Viz Engine
- The Service Reports Timeout Errors / Failed to Get Frame / Uninitialized Pointer
- Recordings Stop Unexpectedly when Using Viz Dart
- The Service Reports Output Dropped a Buffer, Cancelled Late
- Service Stops without a Dump File
- Recording with Unsupported Codec / Container / Resolution
- Configuring via Service Host Web Interface is Slow
- Channel Recorder Hangs when Changing Genlock Framerate Family
- Channel Recorder Does not Record to a Network Disk

Enabling Debug Logging

To enable debug logging, go to the Service Host web interface and to the configuration page of the instance of Channel Recorder that you wish to enable the debug logging. In the configuration page go to:
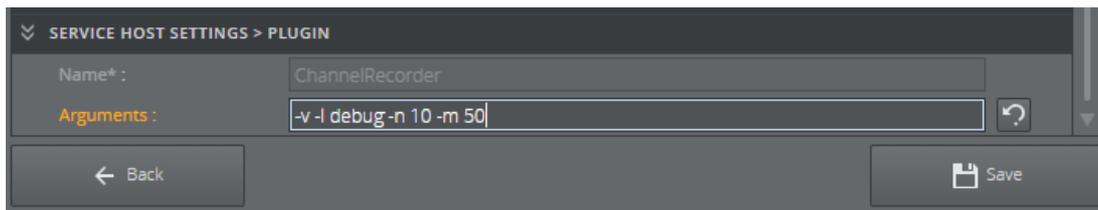
Add the following arguments: `-v -l debug -n 10 -m 50`

Where *n* specifies the maximum number of log files and *m* the maximum size of each log file in MB. By default, the number of log files is four and the maximum size of each log file is `10` MB.

Channel Recorder and Viz Engine

As both Channel Recorder and Viz Engine use Matrox topology, make sure Viz Engine starts before Channel Recorder if you need to use a shared input. If Viz Engine starts after Channel Recorder, the acquisition of the selected input connector fails.

The Service Reports Timeout Errors / Failed to Get Frame / Uninitialized Pointer

If Channel Recorder reports timeout errors, try increasing the timeout value by sending the command `CONFIG SET TIMEOUT [timeout]`, or by configuring the timeout value using the web interface provided by Service Host.

Channel Recorder reports the following:

```
[2018-09-26 09:55:39.056 +10:00][9408][error]:
[Vizrt_Channnel04::CaptureChannelSystemInD] GetAVVField: videoManager
Vizrt_Channnel04::CaptureChannelSystemInD::m_pJSMVideo: Error: 0x89998048
facility=0x999 error_code=0x8048 Failed because there was a time out.
[2018-09-26 09:55:39.056 +10:00][9408][error]:
[Vizrt_Channnel04::CaptureChannelSystemInD] Failed to get frame
[2018-09-26 09:55:39.056 +10:00][9408][error]:
[Vizrt_Channnel04::CaptureChannelSystemInD] GetNodeContentFields: Error: 0x89998045
facility=0x999 error_code=0x8045 One of the method parameter is a uninitialized
pointer.
```

This is usually related to the performance of the HDD and/or RAID.

These errors also appear when Genlock is configured to a different framerate family than the resolution (for example, if input resolution is 720p50, but Genlock is running at 60M). When this happens, the errors flood and Channel Recorder might hang and must be killed using the task manager.

Recordings Stop Unexpectedly when Using Viz Dart

The issue is usually related to the signal being unstable and Channel Recorder restarting due to change in resolution. Check in the Channel Recorder logs for the *Status of connector* string. If there are more than two signals present in the logs, then the signal is unstable.

> ⚠ **Note:** Channel Recorder requires a stable signal to function properly with Viz Dart.

An example of a good output after searching the logs is the following:

```
[2018-11-22 21:43:51.157 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
[2018-11-22 21:43:51.173 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
[2018-11-22 21:43:51.189 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
[2018-11-22 21:43:51.204 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=false, Flywheel=false, Resolution=0iInvalid
[2018-11-22 21:43:51.220 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=false, Resolution=720p60M
[2018-11-22 21:43:51.235 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
[2018-11-22 23:29:37.863 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
[2018-11-22 23:29:37.879 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
[2018-11-22 23:29:37.894 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=false, Flywheel=false, Resolution=0iInvalid
[2018-11-22 23:29:37.910 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=false, Resolution=720p60M
[2018-11-22 23:29:37.926 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
```

```
[2018-11-22 23:29:37.941 -05:00][1628][info]: [SdiInStatusCallback] Status of
connector InC: Signal=true, Flywheel=true, Resolution=720p60M
```

The Service Reports Output Dropped a Buffer, Cancelled Late

If Channel Recorder reports output dropped a buffer errors, try increasing the timeout value by sending the command `CONFIG SET TIMEOUT [timeout]` or configuring the timeout value using the web interface provided by Service Host. Increasing the ringbuffer size by sending the command `CONFIG SET RINGBUFFERSIZE [size]` or configuring the ringbuffer size using the web interface provided by Service Host, might also help.

Channel Recorder reports the following:

```
[2018-12-02 19:55:05.804 -02:00][1892][error]: ACE
Vizrt_CR1[1892]:CV_ClipOutChannel::WriteField_i GetBuffers() failed: Reason=Error:
0xa306f003 facility=0x306 error_code=0xf003 Output dropped a buffer, cancelled late.
[2018-12-02 19:55:05.804 -02:00][1892][error]: ACE Vizrt_CR1[1892]CV_ClipOutChannel::
Last Message repeated 100 times.
```

This is usually related to the performance of the CPU. Please make sure that CPU usage while recording does not exceed 80%. Do not forget to enable Hyperthreading if this is disabled in the BIOS.

Service Stops without a Dump File

If there is a change in Windows time (changing the timezone) while Service Host is running, it stops functioning and must be restarted. This is a known issue and will be fixed in future release. To avoid this problem, make sure that Windows time is synchronized to an NTP server. Daylight Savings Time is not affected by this and does not stop Service Host.

Recording with Unsupported Codec / Container / Resolution

If a combination of Codec / Container / Resolution is not supported Channel Recorder will not start the recording.

Channel Recorder reports the following:

```
[2018-09-26 09:20:45.255 +01:00][6004][error]:
[VizrtCRTest::CaptureChannelSystemInA::Scheduler] Record: could not start recording
due to unsupported codec / container / resolution combination
```

Configuring via Service Host Web Interface is Slow

This is a known issue and happens due to having the license server located in a location that is far from where the Channel Recorder machine is located. This is not only affecting Service Host but also Viz Engine. This used to also affects a Channel Recorder instance start up however in Service Host 2.0.0 this was improved by not contacting the server during start up and using the configured value directly. The license configuration was usually in the form 130-4098016501-ENG_SVCHOST_CORE but in 2.0.0 this has been modified to 130-4098016501-ENG_SVCHOST_CORE-**X**, X is the location of the license and can be local (0) or network (1). This configuration

should be filled automatically during Channel Recorder configuration in the Web Interface. Old license string will still be supported.

Channel Recorder Hangs when Changing Genlock Framerate Family

There is a known issue in which Channel Recorder hangs when the input GENLOCK is changed to a different framerate family (for example, p50 to p60M). The hang can still happen even after a full service restart. The messages shown are:

```
[11:21:58.231235 +01:00] [info][pid=8804][tid=12044]: [_sd_id:
servicehost@27566.3.4.0]
[VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI IN
A::ClipOutChannel1] starting up
[11:21:59.251413 +01:00] [error][pid=8804][tid=1332]: [_sd_id:
servicehost@27566.3.4.0]
[VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI IN A]
LL_WaitForWriteCompletion: Error: 0x89998048 facility=0x999 error_code=0x8048 Failed
because there was a time out.
[11:22:00.255798 +01:00] [error][pid=8804][tid=1332]: [_sd_id:
servicehost@27566.3.4.0]
[VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI IN A]
LL_WaitForWriteCompletion: Error: 0x89998048 facility=0x999 error_code=0x8048 Failed
because there was a time out.
[11:22:01.259014 +01:00] [error][pid=8804][tid=1332]: [_sd_id:
servicehost@27566.3.4.0]
[VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI IN A]
LL_WaitForWriteCompletion: Error: 0x89998048 facility=0x999 error_code=0x8048 Failed
because there was a time out.
[11:22:01.259145 +01:00] [error][pid=8804][tid=1332]: [_sd_id:
servicehost@27566.3.4.0]
[VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI IN A]
OnAvCompletion(VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI
IN A::m_pJSMVBI): curr=60 inUse=60 max=255 Error: 0xe306f000 facility=0x306
error_code=0xf000 Sync: Buffer has been dropped, cancelled
[11:22:01.259183 +01:00] [error][pid=8804][tid=1332]: [_sd_id:
servicehost@27566.3.4.0]
[VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI IN A]
OnAvCompletion(VizrtChannelRecorderGenlockFamilySwitching::CaptureChannelSystem SDI
IN A::m_pJSMAudio): curr=62 inUse=62 max=255 Error: 0xe306f000 facility=0x306
error_code=0xf000 Sync: Buffer has been dropped, cancelled
```

The only way to fix this is to restart the service several times, until the last log line shown is not an error. Usually the service must be restarted once.

Channel Recorder Does not Record to a Network Disk

This is usually caused by a permission issue. However, there is a known issue in **Windows 11** where the setting *Priority* must be set to *Normal* for Channel Recorder to be able to record to a network disk.

4.2.7 Logging

The log file for Channel Recorder is located at *%ProgramData%/vizrt/ServiceHost*. The log file name reflects the **Service Name** the Channel Recorder instance is registered as in Service Host. For troubleshooting, you can enable a higher level of logging for more information.

Enabling Debug Logging

To enable debug logging, go to the Service Host web interface and to the configuration page of the instance of Channel Recorder that you wish to enable the debug logging. In the configuration page go to:
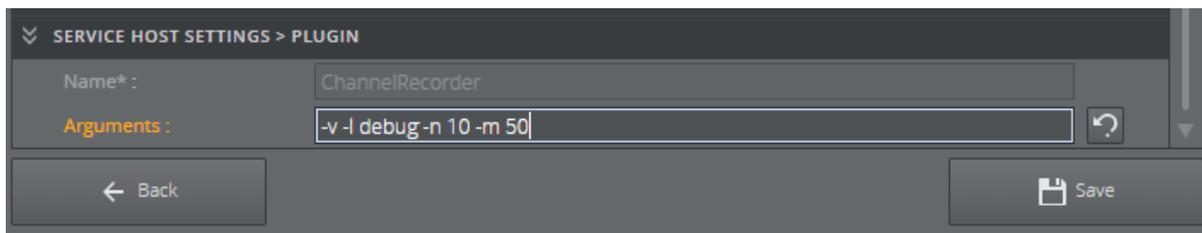
Add the following arguments: `-v -l debug -n 10 -m 50` , where *n* specifies the maximum number of log files and *m* the maximum size of each log file in MB. By default, the number of log files is four and the maximum size of each log file is `10` MB.

See Also

- The Service Host Section in Channel Recorder Configuration.

4.2.8   Crash Recording

Channel Recorder can be used to instantly record any input signal into a clip file. The delay between command and first field written to the file is usually less than four fields.

An example of how to invoke crash recording with MVCP command would be the following:

```
UADD CS_ENCODER1 * SHAR Capture
LOAD U1 "MVCP_Crash" IN CRTE NOEX
CUER U1
REC U1
USTA U1
STOP U1
UNLD U1
UCLS U1
BYE
```

An example of how to invoke crash recording with regular Channel Recorder Commands:

```
RECORD CLIP Crash.mxf
RECORD START
RECORD STOP
```

Both examples create an instant recording (also called a crash recording) using the values in the configuration file.

4.2.9   Loop Recording

This mode of recording only works with **Timecode Source** time of day. Loop recordings can also be split into various chunks. These can either be specified via timecode format or with fields being written.

An example on how to run a loop recording using regular Channel Recorder commands:

```
RECORD LOOP CHUNK=00:01:00:00
RECORD LOOP PREFIX=Loopy
RECORD LOOP POSTFIX=End
RECORD LOOP START
```

This generates chunks, each consisting of one minute, until you stop the recording.

Possible options for loop recordings are:

- **LENGTH:** Determines the length of the loop in frames or timecode. At least this amount of frames will reside on the disk. The minimum length value is `00:02:00:00` or the equivalent number of fields / frames.
- **CHUNK:** Determines the size of one chunk in frames or timecode. The chunk size influences the loop length. The minimum chunk value is `00:00:10:00` or the equivalent number of fields / frames.
- **DISKSIZE:** Determines the size of the loop, defined by the size of the clips on the disk. The oldest clips are deleted until the size of the clips drops below the specified value.
- **DISKFREE:** Determines the size of the loop, defined by the space left on the disk. If it drops below the value, the oldest clips are deleted until at least the specified amount is free again.
- **NAME:** Name of the file that accepts a subset of strftime.
- **PREFIX:** Adds a prefix to the generated filename.
- **POSTFIX:** Adds postfix to the generated filename.

DISKSIZE and **DISKFREE** are defined in bytes. It is possible to add the following postfix which will multiply the base values by different factors:

Postfix	Factor
kib	1024
kb	1000
k	1024
mib	1024*1024
mb	1000000
m	1024*1024
gib	1024 * 1024 * 1024

Postfix	Factor
gb	1000000000
g	1024 * 1024 * 1024

Understanding Chunk and Length Parameters

An example on how `LENGTH` and `CHUNK` parameters are related:

```
RECORD LOOP CHUNK=00:00:30:00
RECORD LOOP LENGTH=00:04:00:00
RECORD LOOP PREFIX=Loopy
RECORD LOOP START
```

This generates a total of nine files after four minutes. One of the files is always zero bytes, this is the next file the Channel Recorder writes to. Another file has a changing size, this is the file Channel Recorder is currently writing to. The other seven files have already been written. When Channel Recorder reaches four minutes it starts removing the first file it has written.

Chunk File Naming

When performing a loop recording the chunks have by default the following name format: {}{:02}{:02}{:02}{:02}_{}{}. Each {} translates to the following in the same order:

- Prefix specified with the `RECORD LOOP PREFIX` command.
- Two digit hour specifying when the chunk started recording.
- Two digit minute specifying when the chunk started recording.
- Two digit seconds specifying when the chunk started recording.
- Two digit frame / field specifying when the chunk started recording.
- Postfix specified with the `RECORD LOOP POSTFIX` command.
- Numerical value that increases for each chunk.

An example chunk name would be *Loopy_13593509_0_End.mxf.* For this example prefix would be *Loopy_* and postfix would be *_End*.

More Flexible Chunk File Naming

It is also possible to be more flexible with the naming by using the standard format **strftime**. **strftime** is a function to format date and time into a readable form.

An example would be `RECORD LOOP NAME=%F_recording_%n.mxf` . This generates something like the following: *2001-08-23_recording_0, 2001-08-23_recording_1, etc.*

The key specified %n is mandatory as to prevent collision between chunks. If this is not specified, one is provided at the end of the filename before the extension. So for example `RECORD LOOP NAME=%F_recording.mxf` generates the same as the above. Even though %n was not specified.

Not all specifiers in *strftime* are supported. This is because Windows file name does not allow some of the characters returned by **strftime**. The following are excluded because it would result in an invalid name for Windows:

Specifier	Description	Result
%c	Date and time representation	Thu Aug 23 14:55:02 2001
%D	Short MM/DD/YY date, equivalent to %m/%d/%y	08/23/01
%r	12-hour clock time	02:55:02 pm
%R	24-hour HH:MM time, equivalent to %H:%M	14:55
%t	Horizontal-tab character ('\t')	
%T	ISO 8601 time format (HH:MM:SS), equivalent to %H:%M:%S	14:55:02
%x	Date representation	08/23/01
%X	Time representation	14:55:02

Ability to Protect Chunks of Ongoing Loop Recordings

The ability to protect chunks of on-going loop recordings by specifying time-spans that are not overwritten by the ongoing recordings if the recording reaches the size/length limit and starts to overwrite the oldest content of the current recording.

- `RECORD LOOP PROTECT IN=hh:mm:ss:ff OUT=hh:mm:ss:ff` (can be called multiple times if multiple protections are needed)
- `RECORD LOOP UNPROTECT` (assuming no specific files / time-span to unprotect, this removes all active protection)

Alternative with files:

- `RECORD LOOP PROTECT file3.mxf file4.mxf` (can also be called multiple times)
- `RECORD LOOP UNPROTECT` (assuming no specific files / time-span to unprotect, this will remove all active protection)

It is not possible to protect all files. The minimum number of unprotected files is the number of writers. This is to prevent a lock down and the recorder not being able to record anything.

- `RECORD LOOP IN=00:00:00:00 OUT=23:59:59:59 PROTECT`

This attempts to protect all the files.

4.2.10   Scheduled Recording

Channel Recorder has its own scheduler to plan recordings in the future.

The command to record an input signal starting at 12:00 and running for one hour would be the following:

```
RECORD SET scheduled.mxf IN=12:00:00:00 OUT=13:00:00:00
```

If a scheduled recording is being interrupted by a loop or crash recording, the following logic applies:

- A recording is being issued while a scheduled recording is running: **The scheduled recording is interrupted and stopped.**
- During loop recording someone decides to schedule a recording in the future: **The recording is scheduled and run if/when the loop recording stops before the specified time, otherwise it fails.**

4.2.11   Trace Generation

From version 2.4.0, it is possible to generate traces with Channel Recorder. The generated trace is located in *C:\ProgramData\vizrt\ServiceHost\trace*.

Supported commands are:

- `TRACE START`
- `TRACE STOP`
- `TRACE RECORD_FRAMES`
- `TRACE RECORD_DURATION` (in milliseconds, like Viz Engine)

Trace generation can help identify bottlenecks and, when used in conjunction with logs, serves as a valuable tool for troubleshooting.

4.3 References and Specifications

This section details references and specifications for the Channel Recorder.

This section contains the following topics:

- Channel Recorder Command Line Options
- Data Types
- Files and Directories
- Supported MVCP Commands
- Clock Behavior
- Recording Modes and States
- Channel Recorder Control Commands
- Supported Codecs

4.3.1 Channel Recorder Command Line Options

It is possible to reroute some commands to Channel Recorder during initialization. Refer to the Service Host Section in Channel Recorder Configuration. The only use for this is to activate a higher level of logging.

4.3.2   Data Types

Booleans

Booleans are a data type with only two possible values; true or false. The Channel Recorder accepts `YES` , `NO` , `TRUE` , `FALSE` , `ON` , or `OFF` as Boolean values, where:

- `YES` , `TRUE` and `ON` are positive values switching a flag `On` , and
- `NO` , `FALSE` and `OFF` are negative values switching a flag `Off` .

Timecodes

Timecodes in Channel Recorder have the form `<hh>:<mm>:<ss>:<ff>,` where

- `<hh>` represents the hour in 24 hour format,
- `<mm>` is minutes,
- `<ss>` is seconds, and
- `<ff>` refers to the frame number, according to the frame rate.

> **Example:** A zero timecode looks like this: `00:00:00:00` .

4.3.3   Files and Directories

The Channel Recorder uses various files and directories while running.

Description	Type	Location
Installation Directory	Directory	*<viz install folder>\ServiceHost*
ProgramData Directory	Directory	*%ProgramData%\vizrt\ServiceHost*
Log	File	*<instance_name>.log*
	Path	*%ProgramData%\vizrt\ServiceHost\<instance_name>.log*
Configuration	File	*<instance_name>.conf*
	Path	*%ProgramData% \vizrt\ServiceHost\<instance_name>.conf*

4.3.4 Supported MVCP Commands

Channel Recorder supports the following sets of Multiport Video Computer Protocol (MVCP) commands:

Supported Global MVCP Commands

Command	Function
`BYE`	Closes the current connection.
`MON`	Initiates event monitoring mode.
`PLS`	Returns the supported media ports.
`UADD`	Creates a new unit.
`ULS`	Returns the existing VST units.

Supported Unit MVCP Commands

Command	Function
`CCST`	Gets the current status of the VST Clip Cache (the numbers displayed are bytes used/available).
`CUER`	Cues recording of the unit's currently loaded clip.
`LOAD`	Loads a clip into a unit.
`SET`	Sets controls for the unit: **vtr.media.clip.format:** Allowed values: `[movie/stream/mxf]` **vtr.media.video.input.compression.type** `[mpg2]` **vtr.media.mpeg.bit_rate:** Sets the recording bitrate. **vtr.media.mpeg.imx:** Enables IMX recording. **vtr.media.audio.input.channels:** Sets the number of audio channels to record. **vtr.media.audio.input.sample.width:** Sets audio bits per sample. **vtr.media.video.standard:** The command is understood, however, as the video standard is determined by the input signal, it does not get changed. **vtr.media.input.trigger.duration.out:** Sets the duration of the recording. **vtr.media.clip.start.mode:** Allowed values: `[time-of-day][vitc][ltc]`

Command	Function
STOP	Stops playback or recording.
UCLS	Closes a unit.
UNLD	Unloads the unit's currently loaded clip.
USTA	Returns the status of a unit.

Example MVCP Commands Sent by Viz Dart

```
UADD CS_ENCODER1 * SHAR dart
DISC global U1
UERR U1
SET U1 MED vtr.media.clip.format "movie/stream/mxf"
SET U1 MED vtr.media.video.input.compression.type "mpeg2-ibp-422p@hl"
SET U1 MED vtr.media.audio.input.channels 4
SET U1 MED vtr.media.audio.input.sample.width 16
LOAD U1 LUTAS_-_GLORY_62_-_SPORTV.mxf IN CRTE NOEX
SET U1 MED vtr.media.clip.start.mode "time-of-day"
SET U1 MED vtr.media.input.trigger.duration.out "04:00:00:00"
CUER U1
@14:50:00:00 REC U1
USTA U1
UCLS U1
UNLD U1
```

See Also

- The Video Server Toolkit Developer's Guide on ftp.sgi.com (alternate link).

4.3.5   Clock Behavior

The clock is a very important component of Channel Recorder. If no clock is present Channel Recorder might fail in performing scheduled, loop or frame accurate recordings. There are four sources of resolution: input, genlock, configuration and internal.

For clock type `GENLOCK`, the priority is given in the following order:

- Input
- Genlock
- Configuration
- Internal

For clock type `INPUT`, the priority is given in the following order:

- Input
- Configuration

Important points to remember are:

- The resolution from the configuration is only used when there is no input or genlock signal detected.
- Internal resolution is only relevant when the following is true:
    - There is another application using Matrox board.
    - The application in question is using a different frame rate family.
- Clock type `INPUT` is the simplest and should be used unless there is a compelling reason not to use it.

> ⚠️ **Note:** For IP boards, `2059` is used by default.

Recording Accuracy

Recording accuracy in this context means the expected duration of the recorded clip in relation to the requested recording done via commands. The following points are important to take into account to understand how Channel Recorder works in relation to this topic:

- There are two possible clocks that can be configured for Channel Recorder: INPUT and GENLOCK.
  - The clock defines the rate at which the frames are received by Channel Recorder.
  - INPUT should never be used if accuracy is important and a GENLOCK signal is available.
- There are three possible timecode sources: TIME_OF_DAY, VITC, and LTC.
  - VITC and LTC do not get affected by recording accuracy since the timecode can be found contained in the frames that Channel Recorder received to start or stop a recording.
  - TIME_OF_DAY is calculated from the system time (which is why it is important to have Windows NTP correctly configured to avoid sync issues).

Channel Recorder deals with recording in TIME_OF_DAY as follows:

- During first start up, Channel Recorder gets the current system time, convert it to timecode and use this as reference to start and stop recording.
- Channel Recorder still attempts to get the system time in every frame fetch, however this value is not used until a certain threshold is surpassed.
- Channel Recorder increases the timecode it gets during start up by one frame and compare it to the current system time. If a threshold is exceeded, it uses the system time (instead of the previous TC increased by 1). This is called a *resync*.
- There are two thresholds:
  - During idle the threshold is **two frames** in both directions.
  - During crash and scheduled recordings, the threshold is **25 frames** in both directions plus a configurable threshold: *trigger-threshold*. The bigger threshold during recording is to allow for back-to-back recording in a less than ideal set up in which small drifting is expected.
  - During loop recording there is no threshold and a resync never happens.
- A *resync* usually means that the end recording has more or less frames than expected depending on the direction of the drift.

Regarding the TC that is contained within the frames. The TC is **continuous** (unless the option discontinuous-tc is enabled). The TC of each frame starts at the defined *START_TC* which can be configured for each recording. This means that even if there is a jump due to resync, the recorded frames have continuous TC. The purpose of the resync is to start and stop at the correct local time.

Summary of Resync Behavior

CR timecode is derived from the Windows system time (when using TIME_OF_DAY) and is initialized at the beginning of starting CR. Every time Matrox gives CR a frame the *CR timecode* is increased by one frame. So, if CR is configured with 720p50. This means that every 50 frames, one second should have passed. If this is not the case then there is a drift. CR guarantees (best effort) that it records for the amount of time it is instructed to do: A recording for one hour means that it records for one hour (of system time); however, if the amount of frames that it receives within that hour is less/more than 180000 then… that is what the duration of the recordings are.

When CR is not recording the correction happens when +2/-2 drifting from the system time is detected. This is to ensure that the scheduled recordings happen when the user instructed it to happen as precise as possible even in the presence of drift in the input clock / genlock.

When CR is recording the correction happens when +25+trigger_threshold/-25-trigger_threshold (configurable) drifting from the system time is detected. This is to ensure that b2b recording works even if the current recording is sync to the future, overlapping the next one. The current one is stopped and the next one is triggered immediately due to being in the *trigger_threshold* range. When b2b you should not lose any frames, however the frames could be in the next file instead of the previous one or vice versa.

Daylight Saving Time

Since CR uses the system local time. Daylight Saving Time (DST) is provided automatically. If during a recording DST occurs the recording is one hour longer or shorter. Timecode is not affected since it is continuous. It is possible to reflect the change of DST in the timecode by using the configuration discontinuous-tc. Discontinuous TC puts the current system time converted to TC to the frames, so if a jump appears such as during DST, the jump is reflected in the clip TC.

4.3.6　Recording Modes and States

> ⚠️ **Note:** The following information is only relevant when using the native recording features of Channel Recorder.

The first time Channel Recorder runs, it has the *schedule* state. Channel Recorder supports three modes of recordings: *schedule*, *loop* and *crash*. All three modes share the same timeline and writers, thus some kind of logic is needed to avoid conflicting each other operations. The following logic applies:

Crash and *loop* have the same level of priority:

- When a *crash* recording is running a *loop* recording can not be started.
- When a *loop* recording is running a *crash* recording can not be started.

Source Recording Mode	Target Recording Mode	Action	Output Mode
Crash	Loop	Not possible	Crash
Crash	Schedule	Deferred until stopped	Crash
Crash	Crash	Stop the active recording and start another recording	Crash
Loop	Crash	Not possible	Loop
Loop	Schedule	Deferred until stopped	Loop
Loop	Loop	Stop the active recording and start another recording	Loop
Schedule	Loop	Suspend existing entries and timeline	Loop
Schedule	Crash	Suspend existing entries and timeline	Crash
Schedule	Schedule	Attempt to add to the timeline	Schedule

4.3.7 Channel Recorder Control Commands

The application **Viz Send**, included in **Viz Engine** installation, can be used to communicate with Channel Recorder. Connect the tool to the port specified in the configuration file. The following commands are implemented in the service:

- ABOUT
- TRACE
- MAIN
- OUTPUT
- RECORD
- CONFIG
- CONFIG SET

ABOUT

Command	Description
`ABOUT GET`	Prints license information of all libraries used in this software.

TRACE

Command	Description
`TRACE START`	Starts the trace.
`TRACE STOP`	Stops the trace.
`TRACE RECORD_FRAMES`	Record in frames.
`TRACE RECORD_DURATION`	The duration is in milliseconds.

MAIN

Some of the following commands must be called during initialization. To configure these commands, use the configuration web interface and the restart the service. Refer to Channel Recorder Configuration for more information.

Command	Description
`MAIN VERSION`	Returns Channel Recorder version.

Command	Description
`GET VERSION`	Returns Channel Recorder version. Same as `MAIN VERSION`.
`MAIN HOSTNAME`	Returns system hostname.
`MAIN SVCNAME`	Returns Service Host version.
`MAIN BOARD LIST`	Lists all available boards.
`MAIN BOARD SELECT` [serial number]	Selects a board. The board is identified by the serial number. If the board with the given number is not found, the first board is selected.
`MAIN BOARD GET`	Returns the selected board.
`MAIN CONNECTOR SELECT`	Selects a connector. Connectors are labeled consecutively from A to P or numerically from 1 to 32. Numerical value is needed for addressing all inputs in a 2110 capable board. This command can also be used to restart the channel without using the web interface.
`MAIN CONNECTOR GET`	Returns the selected connector.

Command	Description
`MAIN LICENSES`	Returns the state of the licenses defined at configuration. An example output is: ℹ️ **Example:** "Service Host Core" ENG_SVCHOST_CORE 1200004 300 7 1 VL_GRACE_LICENSED VL_SHARED 0 1540987200 0 0 license-server.vizrt.internal "SDI/IP Video Input Enablement (max)" ENG_ING_SDIIP_MAX 1200200 300 21 1 VL_GRACE_LICENSED VL_SHARED 0 1540987200 0 0 license-server.vizrt.internal "4K Video Enablement" ENG_ING_4KVIDEO 1200400 300 15 1 VL_GRACE_LICENSED VL_SHARED 0 1540987200 0 0 license-server.vizrt.internal "Viz Engine DNxHD Codec Support" ENG_PLROY_DNXHD 1200603 300 18 1 VL_GRACE_LICENSED VL_SHARED 0 1540987200 0 0 license-server.vizrt.internal Each line is one license and each row separated by space is: • Long name of the license in quotes. • Short name of the license. • The ID of the license. Available licenses are: • Service Host Core (VL_ENG_SVCHOST_CORE, 1200004) • SDI IN MAX (VL_ENG_ING_SDIIP_MAX, 1200200) • DNxHD Ingest (VL_ENG_PLROY_DNXHD, 1200603) • UHD for Playback and Ingest (VL_ENG_ING_4KVIDEO, 1200400) • UHD for Ingest (VL_ING_4KVIDEO, 1200404) • How many licenses of this type are available in the current container. • How many licenses of this type are currently in use in the current container. • How many licenses are in use by Channel Recorder. • Grace state of the license. Possible values are `VL_GRACE_LICENSED`, `VL_GRACE_UNLICENSED` or `VL_GRACE_GRACE`. • How is the license allocated. Possible values are `VIZLIC_USER_LIMIT`, `VIZLIC_NO_USER_LIMIT`, `VIZLIC_EXCLUSIVE` or `VIZLIC_SHARED`. The licenses used by Channel Recorder always has `VIZLIC_SHARED` as value. • Date when the license starts. In Unix time. • Date when the license expires. In Unix time.

Command	Description
	• Start date of the currently valid support contract of a customer (only filled at application licenses) In Unix time. • End date of the currently valid support contract of a customer (only filled at application licenses). In Unix time. • The host name of the current license server (contains the local computer name in case of a local license).

OUTPUT

One output target can be initialized during startup. During execution more than one output can be enabled. For instance you can have VideoOut A enabled. Please refer to Channel Recorder Configuration to initialize an output during startup.

Command	Description
`OUTPUT GET`	Lists the active targets.
`OUTPUT LIST`	Lists available targets.
`OUTPUT START` [target]	Starts an output handler for the specified [target]. Several output handlers can be started in parallel, but only one is allowed for each target. Valid targets are: • `VideoOut` : This target requires one additional parameter: `[connector]` . The `[connector]` parameter specifies the video output connector of the Matrox board. > ℹ️ **Example:** `OUTPUT START VideoOut A`
`OUTPUT STOP` [target]	Stops the output handler for the specified target.

RECORD

Crash Recording / One-time Scheduling

These are specific commands used only for crash and one-time scheduling of recording. These command are available from version 1.0. For scheduling, it is recommended to use `RECORD SET` introduced from version 1.1.

Command	Description
`RECORD CLIP` [clip name]	Sets the clip name and initializes the recorder. This command does not start recording (see `RECORD START`).
`RECORD START` [duration] [start time] [end mode]	Starts or continues recording. The `[duration]` , `[start time]` , and `[end mode]` parameters are optional. However, the parameters are interdependent as follows: • If the parameter `[start time]` is provided, `[duration]` is also required. • If the parameter `[end mode]` is provided, `[duration]` is also required. The format for both is **Timecode** (see Data Types). • `[duration]` and `[start time]` can be zero timecode, which is then ignored. • `[end mode]` can either be `STOP` , which finalizes and closes the clip, or `PAUSE` (default behavior), where the clip stays open and can be used for further recording.
`RECORD PAUSE`	Pauses recording.
`RECORD STOP` [end time]	Stops recording and flushes the recorder. A new clip needs to be set afterwards (see `RECORD CLIP`). The end time is optional and specifies the timecode when the recording should end.

> ⓘ **Example:**
> **Crash recording:**
> `RECORD CLIP Crash.mxf`
> `RECORD START`
> `RECORD STOP`
> **One-time scheduling recording:**
> `RECORD CLIP Crash.mxf`
> `RECORD START 600 15:00:00:00`

Loop Recording

These are specific commands used only for loop recording. It allows setup, start or stop loop recording. These are introduced from version 1.1.

Command	Description
`RECORD LOOP` [key=value] … [key=value] [operation]	Starts, stops or configures the settings for loop recording. Any number of properties can be applied in the form of key-value-pairs. If no property is supplied, and the entry does not yet exist, it is added with the default values. Available keys are: • `PREFIX` : Prefixes to the filename. It is appended with the timestamp of the start time. • `POSTFIX` : Postfixes to the filename. Is appended after the timestamp. • `LENGTH` : Determines the length of the loop in frames or timecode. At least this amount of frames reside on the disk. The minimum length value is `00:02:00:00` or the equivalent number of fields / frames. • `CHUNK` : Sets the size of one chunk in frames or timecode. The chunk size influences the loop length. The minimum chunk value is `00:00:30:00` or the equivalent number of fields / frames. • `DISKSIZE` : Sets the size of the loop is defined by the size of the clips on the disk. The oldest clips are deleted until the disk size drops below the specified value. • `DISKFREE` : Sets the size of the loop is defined by the space left on the disk. If it drops below the value, the oldest clips are deleted until at least the specified amount is free again. `[operation]` can be `START` or `STOP` . When no `[operation]` is specified then only the settings are set for the specified key-value-pairs. It is possible to specify `[key=value]` `[operation]` at the same time but only when `[operation]` is `START` . ⓘ **Example:** `RECORD LOOP LENGTH=00:05:00:00 CHUNK=00:00:30:00 PREFIX=Loopy START` `RECORD LOOP STOP`

Protected Loop Recording

Starting with version 2.0, it is possible to protect a range of files from being overwritten.

Command	Description
`RECORD LOOP NAME=test%n.mxf CHUNK=00:00:10:00 LENGTH=00:01:00:00 START`	Starts recording in one second chunks.

Command	Description
`RECORD LOOP IN=00:00:00:00 OUT=00:00:20:00 PROTECT`	Protects the first two files (0-10 sec) and (10-20 sec) from being overwritten.
`RECORD LOOP IN=00:00:00:00 OUT=00:00:20:00 UNPROTECT`	The unprotected files are the first one being deleted (*file0.mxf* and then *file1.mxf*). This is determined by the internal timecode.

> ℹ **Information:** It is not possible to protect all files. The minimum number of unprotected files is the number of writers. This is to prevent a lock down and the recorder not being able to record anything.

Scheduled Recording

These are specific commands used only for scheduled recording. It allows setup, start or stop scheduled recording.

Command	Description
`RECORD SET` [name] [key=value] … [key=value]	Adds a new clip to the timeline, or changes a property of an entry defined by `[name]`. Any number of properties can be applied in the form of key-value-pairs. If no property is supplied, and the entry does not yet exist, it is added with the default values. Available keys are: • `FILE_EXTENSION` : Enables or disables automatically adding a file extension to the file name. If this feature is turned off, the client application has full control over the file name. • `CONTAINER` : Sets the container type of the recorded file. With the container type, a default codec is also set. By setting the container type, a valid recording can be started. • `CODEC` : Sets the codec type of the recorded file. By setting the codec type, default values for bitrate and audio are also set. • `BITRATE` : Sets the bitrate for the video encoding. The value can either be applied as bits per second or as Megabits per second. Not all codecs allow changes to the bitrate. In such cases, `BITRATE` is ignored. • `IN` : Sets the default value for the recording in point. The value must be in timecode format: `00:00:00:00`. • `OUT` : Sets the default value for the recording out point. The value must be in timecode format: `00:00:00:00`. • `DURATION` : Sets the default value for the recording duration. The value can be in frames or in timecode format: `00:00:00:00`. • `STARTTC` : Sets the default value for the starting timecode. The value must be in timecode format: `00:00:00:00`. • `AUDIOCHANNELS` : Sets the number of audio channels to record. How many channels are actually recorded depends on the codec and the input signal. • `DISKACCESSSIZE` : Sets the size of data blocks written to the disk in byte. Postfixes like KiB, Kb, k, etc., are allowed, but must not be separated from the value with a blank space. The default value is 4MiB (4194304 bytes). The minimum value is `32KiB` (32768 bytes). KiB and k multiplies the value by 1024. kb multiplies the value by 1000. The same works with _m_ for mega and _g_ for giga. • `VBR` : Sets whether the recorded file, when the format supports it, should be in VBR (Variable Bit Rate). • `TDIR` Sets the default behavior of TDIR (Time Delayed Instant Replay).

Command	Description
	• `TDIRINTERVAL` : Sets the interval of file header updates in TDIR (Time Delayed Instant Replay) recordings. The value is in seconds and fractions of seconds, meaning both 11.1 and 11.2 are considered valid values. Minimum allowed value is `10s` , which is interpreted by Channel Recorder as every frame. The maximum value is `60.0s` . • `TDIRDURATIONPREDEFINED` : Sets the TDIR duration to -1 while the file is growing, when enabled. The initial value is `Off` . • `UPDATEMT` : Updates the modification time of the recorded clip regardless of TDIR value. The initial value is `On` . • `STOPATEND` : Sets the default behavior at the end of the recording. For more information regarding these settings possible values and default values check the section `CONFIG SET` below.
`RECORD REMOVE` [name] ... [name]	Removes the entry `[name]` . Multiple `[name]` arguments can be provided.
`RECORD GET` [name] | [key] ... [name] | [key]	Lists all entries or shows the properties of an entry. If no parameter is applied, a list of all scheduled clips is returned. Any number of `[name]` and `[key]` can be applied. The values of all keys are returned for all values. If no `[key]` is provided, all values of the applied keys are returned and vice versa. This means that if you call `RECORD GET` with clip names only, the command returns all properties of these clips. When called with properties only, it returns this property for all clips. A special `[key]` is `TIMELINE` , which returns all entries in the timeline.

Common Record Commands

Some of these commands are only used for loop and crash recording, while others are used for all three modes.

Command	Description
`RECORD DURATION`	If no duration parameter is applied to the command, it **returns the timecode relative to the start timecode**. If a duration parameter is applied, the duration of an ongoing recording is changed. The duration parameter can be specified as either a number of frames or a timecode (see Data Types). In the case of loop and schedule recording **the value used to set** using this command is ignored.

Command	Description
`RECORD RESOLUTION GET`	Returns the resolution the Channel Recorder is running at.
`RECORD STATE`	Returns the state the Record Channel service is in. By default state is idle. State is only invalid when the clip was configured incorrectly (for example, container / codec combination is wrong). ℹ **State flow:** Idle -- Start() --> Recording -- Stop() --> Paused ^ || `-- Flush() ---------- +--------------------´
`RECORD STARTTC` [starttc]	Defines which start timecode to use for the recording. If the command is not called before the recording starts, the current timecode is written to the clip. If no value is given, the command returns the current value. This is only used for loop and schedule recording.

CONFIG

Command	Description
`CONFIG GET`	Gets the current settings.
`CONFIG LIST` [argument]	Lists all available variables. The optional `[argument]` flag provides information specific to the applied argument. Known arguments are: • `CONTAINER` : Displays all known container types. • `CODEC` : Displays all known codec types. • `DUMP` : Displays the current dump file setting. • `PRIORITY` : Displays the current process priority setting.
`CONFIG SET` [variable] [value]	Sets the `[variable]` to `[value]` . To get all available variables, call `CONFIG LIST` . Refer also to variables and values for `CONFIG SET below` .

CONFIG SET

Variable and values for `CONFIG SET` command. The `CONFIG SET` command is used to set the configuration globally, this means that all the recording mode uses these settings by default unless specified otherwise. Only

schedule recording can specify different settings for each schedule by specifying them via `RECORD SET` command.

IPIN

Variable	Value
`IPIN IPGENERAL REDUNDANCY` [bool]	Enables redundancy flows.
`IPIN IPGENERAL JOINTYPE` [type]	Indicates the type of membership request made when IPv4 address is a multicast address. Valid values for `[type]` are: • `0 None` • `1 IGMPv2` • `2 IGMPv3` The default value is None.
`IPIN IPVIDEO` [bool]	Enables video flow if `True` .
`IPIN IPVIDEO ADDRESS` [address]	Sets destination IPv4 address. The default value is `239.0.10.10` .
`IPIN IPVIDEO PORT` [port]	Sets destination port. The default value is `50002` .
`IPIN IPVIDEO RTPPAYLOADID` [id]	Indicates the RTP (Real-time Transfer Protocol) Payload ID to capture. Used when RTP Payload ID is set to `True` . The default value is `96` .
`IPIN IPVIDEO PACKETINTERVALTHRESH OLD` [threshold]	Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For video, the range is from `6.4` nanoseconds to `419424.0` nanoseconds in intervals of `6.4` nanoseconds. The default value is `6.4` .
`IPIN IPVIDEO FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: • `1 Inclusion` • `2 Exclusion` `The default value is 1.`

Variable	Value
`IPIN IPVIDEO IGMPV3 FILTERLIST` [list]	Lists IPv4 addresses to be included or excluded separated by a space character.
`IPIN IPVIDEO_S ADDRESS` [address]	Destination IPv4 address. The default value is 239.1.10.10.
`IPIN IPVIDEO_S PORT` [port]	Destination port. The default value is `50002`.
`IPIN IPVIDEO_S PACKETINTERVALTHRESH OLD` [threshold]	Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For video, the range is from `6.4` nanoseconds to `419424.0` nanoseconds in intervals of `6.4` nanoseconds. The default value is `6.4`.
`IPIN IPVIDEO_S IGMPV3 FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: • `1 Inclusion` • `2 Exclusion` `The default value is 1.`
`IPIN IPVIDEO_S IGMPV3 FILTERLIST` [list]	Lists IPv4 addresses to be included or excluded separated by a space character.
`IPIN IPAUDIO` [bool]	Enables audio flow if `True`.
`IPIN IPAUDIO ADDRESS` [address]	Destination IPv4 address. The default value is `239.0.10.12`.
`IPIN IPAUDIO PORT` [port]	Destination port. The default value is `50004`.
`IPIN IPAUDIO RTPPAYLOADID` [id]	Indicates the RTP (Real-time Transfer Protocol) Payload ID to capture. Used when RTP Payload ID is set to `True`. The default value is `97`.

Variable	Value
`IPIN IPAUDIO PACKETINTERVALTHRESHOLD` [threshold]	Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For audio, the range is from `102.4` nanoseconds to `6710681.6` nanoseconds in intervals of `102.4` nanoseconds. The default value is `102.4`.
`IPIN IPAUDIO FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: • `1 Inclusion` • `2 Exclusion` `The default value is 1.`
`IPIN IPAUDIO FILTERLIST` [list]	Lists IPv4 addresses to be included or excluded separated by a space character.
`IPIN IPAUDIO USETRACKCOUNT` [bool]	Uses track count to specify the number of tracks for the incoming IP stream if `True`. The default value is `False`.
`IPIN IPAUDIO TRACKCOUNT` [count]	Specifies the number of tracks for the incoming IP stream. Valid values for `[count]` are: • `1 Mono` • `2 Stereo` • `4 4` • `8 8` • `16 16` • `32 32` • `64 64` `The default value is 8.`

Variable	Value
`IPIN IPAUDIO PACKETDURATION` [duration]	Indicates the duration of the incoming audio packet. The packet size (in samples) can be computed using the sample rate (48000 samples/sec) and the packet duration. Valid values for `[duration]` are: `0 125us` `1 250us` `2 333us` `3 1ms` `4 4ms` `The default value is 3.`
`IPIN IPAUDIO_S ADDRESS` [address]	Destination IPv4 address. The default value is `239.1.10.12` .
`IPIN IPAUDIO_S PORT` [port]	Destination port. The default value is `50004` .
`IPIN IPAUDIO_S PACKETINTERVALTHRESH OLD` [threshold]	Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For audio, the range is from `102.4` nanoseconds to `6710681.6` nanoseconds in intervals of `102.4` nanoseconds. The default value is `102.4` .
`IPIN IPAUDIO_S IGMPV3 FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: `1 Inclusion` `2 Exclusion` `The default value is 1.`
`IPIN IPAUDIO_S IGMPV3 FILTERLIST` [list]	Lists IPv4 addresses to be included or excluded separated by a space character.
`IPIN IPANC` [bool]	Enables ANC flow if `True` .
`IPIN IPANC ADDRESS` [address]	Destination IPv4 address. The default value is `239.0.10.14` .

Variable	Value
`IPIN IPANC PORT` [port]	Destination port. The default value is `50006` .
`IPIN IPANC RTPPAYLOADID` [id]	Indicates the RTP (Real-time Transfer Protocol) Payload ID to capture. Used when RTP Payload ID is set to true. The default value is `100` .
`IPIN IPANC PACKETINTERVALTHRESH OLD` [threshold]	Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For ancillary data, the range is from `819.2` nanoseconds to `53685452.8` nanoseconds in intervals of `819.2` nanoseconds. The default value is `819.2` .
`IPIN IPANC FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: • `1 Inclusion` • `2 Exclusion` `The default value is 1.`
`IPIN IPANC FILTERLIST` [list of ips]	List of IPv4 addresses to be included or excluded separated by a space character.
`IPIN IPANC_S ADDRESS` [address]	Destination IPv4 address. The default value is `239.1.10.14` .
`IPIN IPANC_S PORT` [port]	Destination port. The default value is `50006` .
`IPIN IPANC_S PACKETINTERVALTHRESH OLD` [threshold]	Indicates the threshold for generating the time interval between the IP packets alarm on the main IP stream. For ancillary data, the range is from `819.2` nanoseconds to `53685452.8` nanoseconds in intervals of `819.2` nanoseconds. The default value is `819.2` .

Variable	Value
`IPIN IPANC_S IGMPV3 FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: • `1 Inclusion` • `2 Exclusion` `The default value is 1.`
`IPIN IPANC_S IGMPV3 FILTERLIST` [list]	List of IPv4 addresses to be included or excluded separated by a space character.

SFP

`[sfp]` can be A, B, C or D and it refers to the SFP cage.

> ⚠️ **Note:** These parameters are board-wide, meaning that any change done affects any other instances of any application that is using the board. The values for these parameters should be the same for every instance of Channel Recorder that is using the board.

Variable	Value
`SFP` [sfp] `ADDRESS` [ip]	Sets IPv4 address. The default value is `10.0.0.3`.
`SFP` [sfp] `GATEWAY` [ip]	Sets IPv4 gateway. The default value is `10.0.0.1`.
`SFP` [sfp] `NETMASK` [ip]	Sets IPv4 netmask. The default value is `255.255.255.0`.
`SFP` [sfp] `DHCP` [bool]	Enable DHCP protocol.
`SFP` [sfp] `TOSDSCP` [number]	Specifies a datagram's priority and request a route for low-delay, high-throughput, or highly-reliable service. The default value is `0`.
`SFP` [sfp] `DELAYMECHANISM` [mechanism]	Specifies the type of network delay mechanism to use for the time server connection. Valid values for `[mechanism]` are: • `1 End-to-End` • `2 Peer-to-Peer` The default value is `End-to-End`.

Variable	Value
`SFP` [sfp] `IPMODE` [mode]	Specifies the type of internet protocol mode to use for the time server connection. Valid values for `[mode]` are: • `1 Multicast` • `2 Unicast` • `3 Hybrid` `The default value is Multicast.`
`SFP` [sfp] `MASTERCLOCKDOMAINNUMBER` [number]	Specifies the master clock domain number. The default value is `127`.
`SFP` [sfp] `ANNOUNCERECEIPTTIME` [number]	Specifies the announce receipt time.
`SFP` [sfp] `JOINTYPE` [type]	Indicates the type of membership request made when IPv4 address is a multicast address. Valid values for `[type]` are: • `0 None` • `1 IGMPv2` • `2 IGMPv3` The default value is `None`.
`SFP` [sfp] `IGMPV3 FILTERTYPE` [type]	Specifies the type of filtering applied to the source list when IGMPv3 is used. Valid values for `[type]` are: • `1 Inclusion` • `2 Exclusion` `The default value is 1.`
`SFP` [sfp] `IGMPV3 FILTERLIST` [list of ips]	List of IPv4 addresses to be included or excluded separated by a space character.

2059

Variable	Value
`2059 BESTMASTERSELEC TION`	Specifies the BMCA (Best Master Cock Algorithm) used to select the genlock over IP signal.

PROCESS

Variable	Value
`PRIORITY` [priority]	Sets the process priority class. The values correspond to the Windows process priority. Valid values for `[priority]` are: • `IDLE` • `BELOW_NORMAL` • `NORMAL` • `ABOVE_NORMAL` • `HIGH` • `REALTIME` The default value is `REALTIME`.

CAPTURE

All the configuration below requires a channel restart. This can be achieved by restarting the service or by calling MAIN CONNECTOR SELECT while Channel Recorder is running.

Variable	Value
`CLIP_ROOT` [path]	Sets the root folder for the captured files. If the filename in command `RECORD CLIP` does not contain a absolute path name, the `CLIP_ROOT` is prepended.
`UHD` [bool]	Enables detection of UHDTV signals. When set to `On`, the Channel Recorder scans the signal resolution on the four corresponding input connectors. If four 3G signals are detected, they are interpreted as one UHDTV signal. When set to `OFF`, the four connectors are treated as separate 3G signals. This flag must be set before selecting a connector using `MAIN CONNECTOR SELECT`. The default value is `Off`.

Variable	Value
`V210` [bool]	Uses the 10-bit surface format V210. This is needed to record XAVC. It also increases performance when for example recording ProRes. This surface format is not supported on the Matrox X.mio2+. This flag must be set before selecting a connector using `MAIN CONNECTOR SELECT`. The default value is `Off`.
`COLORIMETRY`	Specifies the colorimetry to be used to be able to record in HDR. By default, ITUR_BT_601 is used for SD resolution and ITUR_BT_709 for HD and UHD resolution.
`VBI` [bool]	Turns off VBI recording. The default value is `On`. This flag must be set before selecting a connector using `MAIN CONNECTOR SELECT`.
`RESOLUTION` [resolution]	Sets the default resolution. Format of `[resolution]` is: `WxHs@F, WxHs@FM, Hs@F, Hs@FM, NTSC, PAL` With `W` = Width, `H` = Height, `s` = Scanmode, `F` = Framerate, and `M` = Drop Frame Flag. Possible values for s are: • `i` or `I` : Interlaced • `p` or `P` : Progressive • `psf` or `PSF` : Progressive segmented `M` is optional, and as an alternative, you can use framerate with two decimal points (e.g. `30M` = 29.97). Examples: `NTSC`, `PAL`, `1280x720p@50`, `1920X1080P@50`, `1280x720p@60M`, `1280x720p@59.97`, `720p50`, `720P60M`, `1080P50`. This flag must be set before selecting a connector using `MAIN CONNECTOR SELECT`. The default value is `NTSC`.
`MULTICONNECTORMODE` [mode]	Specifies if an UHD signal is single-link or quad-link. For quad-link inputs, this also specifies how the signal is split. Only used with X.mio5/DSX.LE5 SDI. Valid values are: • `NONE` • `2SI` • `SQUARE_DIVISION`

Variable	Value
`SDASPECTRATIO` [type]	Specifies the aspect ratio for SD resolution. Available sources are: • `4_3` • `16_9`
`ALIASNODESOURCE` [type]	Specifies the input source of an alias node. Available sources are: • `INPUT` • `OUTPUT`
`RESTARTINGDELAY` [delay]	Specifies the time in milliseconds before the channel is restarted after detecting a change in resolution.
`RESTARTING` [bool]	Enables or disables restarting after detecting a change in resolution.
`RESOLUTIONDETECTION` [bool]	Enables or disables resolution detection.
`ALIASNODESOURCE` [source]	Specifies the source of the alias node which specify from which output or input connector the application records.

TEST

Variable	Value
`PERFORMANCETESTMODE` [mode]	Sets the number of audio channels to record. How many channels are actually recorded depends on the codec and the input signal.

MANAGER

Variable	Value
`RINGBUFFERSIZE` [size]	Sets the size of the capture ring buffer. The default value is `60`.

Variable	Value
`TIMEOUT` [timecode]	Sets the timeout for the capture operation in milliseconds. If the recorder reports timeout errors, increasing the timeout could help. Refer to Data Types for information on time code formatting.

PERSISTENCE

Variable	Value
`PERSISTENCE` [bool]	Turns on persistence for operational data like the config and the timeline. Network changes done by NMOS persist through service restarts. The default value is `Off`.
`PERSISTENCEINTERVAL` [seconds]	Sets the interval, at which the data is written to the disk. The value is in seconds and fractions of seconds, meaning both `0.1` and `1.0` are considered valid values. A value of `0` means that every change is written. The default value is `0`.

SCHEDULER

Variable	Value
`WRITERSCOUNT` [number]	Sets the number of writers to initialize. More writers cause more memory usage. This flag must be set before selecting a connector.
`TRIGGERTHRESHOLD` [frames|timecode]	If a timed command missed the execution time, but is still within the trigger threshold, it is executed late. Outside of this window it is just ignored until the next time the timecode is received. The value can either be a number of frames or a timecode based relative value. The default value is `5` frames.
`TIMECODESOURCE` [source]	Specifies the timecode source. Available sources are: • `VITC` • `LTC` • `TIME_OF_DAY`
`INCLUSIVEOUT` [bool]	When enabled the last timecode written to the file is the defined out point of the recording. The default value is `NO` / `FALSE` / `OFF`.

Variable	Value
`DISCONTINUOUSTC` [bool]	When enabled, each field/frame uses the timecode calculated from the system time in case of TIME_OF_DAY and ATC/VITC/ATC/LTC embedded in the fields/frames if this is chosen as timecode source. This option is useful to support DST (Daylight Saving Time) or when the input's timecode is expected to have gaps and this must be kept in the recorded file. StartTC will not work when this is enabled. The default value is `NO` / `FALSE` / `OFF` .

LOG

Variable	Value
`TCLOGINTERVAL` [frames|timecode]	Specifies the interval at which the current timecode is logged. The value can either be a number of frames or a timecode-based relative value. The default value is `0` , which means that every full second is logged.

CLIP

For the following parameters to take effect Channel Recorder does not need to be restarted, however, a new recording must be issued.

Variable	Value
`FILE_EXTENSION` [bool]	Enables or disables automatically adding a file extension to the file name. If this feature is turned off, the client application has full control over the file name. The default value is `On` . Please refer to Data Types for details on accepted boolean values.

Variable	Value
`CONTAINER` [container type]	Sets the container type of the recorded file. With the container type, a default codec is also set. By setting the container type, a valid recording can be started. Available container types are: • `AVCINTRAMXF` • `AVI` • `DVCPROMXF` • `MOV` • `MXF` • `XAVCMXF` • `XDCAMMXF` • `DNxHDMXF` ⚠ **Note:** To utilize the TDIR capabilities of the **ProRes** codec with a **.mov** container in Viz Engine, you must use the *.Ref*-file ⚠ **Note:** When container is XDCAMMXF, the default codec is XDCam which is HD422. The default bitrate for XDCam, XDCamEX and XDCamHD are `50`, `35` and `35` or `25` respectively. XDCamEX outputs 1920x1080 clip at 35 Mbps and XDCamHD outputs 1440x1080 clip at 35 Mbps (VBR) or 25 Mbps (CBR) depending on the bitrate chosen. Refer to the Supported Codecs for further details.

Variable	Value
`CODEC` [codec type]	Sets the codec type of the recorded file. By setting the codec type, default values for bitrate and audio are also set. These values can be changed (Refer to the Supported Codecs for further details). Not all codec types are available with all container types. Available codec types are: • `DvCam` • `DvCPro` • `Dv50` • `IFrame` • `XDCam` • `XDCamEX` • `XDCamHD` • `AVCIntra50` • `AVCIntra100` • `ProRes` • `XAVC` • `DNxHD` ⚠ **Note:** To utilize the TDIR capabilities of the **ProRes** codec with a **.mov** container in Viz Engine, you must use the *.Ref*-file. ⚠ **Note:** When container is XDCAMMXF, the default codec is XDCam which is HD422. The default bitrate for XDCam, XDCamEX and XDCamHD are `50`, `35` and `35` respectively. XDCamEX outputs 1920x1080 clip at 35 Mbps and XDCamHD outputs 1440x1080 clip at 35 Mbps (VBR) or 25 Mbps (CBR) depending on the bitrate chosen. Refer to the Supported Codecs for further details.
`BITRATE` [rate]	Sets the bitrate for the video encoding. The value can either be applied as bits per second or as Megabits per second. Not all codecs allow changes to the bitrate. In such cases, `BITRATE` is ignored.
`IN` [timecode]	Sets the default value for the recording start time. The initial value is empty.
`OUT` [timecode]	Sets the default value for the recording stop time. The initial value is empty.

Variable	Value
`DURATION` [frames|timecode]	Sets the default value for the recording duration. The initial value is `0` . When applying a duration to the `RECORD START` command, the default value is ignored but not changed. Refer to Data Types for information on time code formatting.
`STARTTC` [timecode]	Sets the default value for the starting timecode. The value must be in timecode format: `00:00:00:00` .
`AUDIOCHANNELS` [channels]	Sets the number of audio channels to record. How many channels are actually recorded depends on the codec and the input signal.
`DISKACCESSSIZE` [size]	Sets the size of data blocks written to the disk in byte. Postfixes like KiB, Kb, k, etc., are allowed, but must not be separated from the value with a blank space (see example). • The default value is `4 MiB` (4194304 bytes). • The minimum value is `32 KiB` (32768 bytes). • `KiB` and `k` multiplies the value by `1024.` `kb` multiplies the value by `1000` . • The same works with `m` for mega and `g` for giga. ⓘ **Example:** `CONFIG SET DISKACCESSSIZE 1024KiB`
`VBR` [bool]	Activates variable bit rates for codecs which support it (currently XAVC only). The default setting is `Off` .
`TDIR` [bool]	Sets the default behavior of TDIR (Time Delayed Instant Replay). The default setting is `On` .
`TDIRINTERVAL` [time]	Sets the interval of file header updates in TDIR recordings. The value is in seconds and fractions of seconds, meaning both `0.1` and `1.0` are considered valid values. Minimum allowed value is `0.001` , which is interpreted by Channel Recorder as every frame. A typical value would be `10.0` , the default value is `3.0` . ❌ **Warning:** To secure proper operation with Viz Engine, this value must **not** exceed `10.0` .

Variable	Value
`TDIRDURATIONPREDEFINED` [bool]	Sets the TDIR duration to -1 while the file is growing if enabled. The initial value is `Off` .
`UPDATEMT` [bool]	Updates the modification time of the recorded clip regardless of TDIR value. The initial value is `On` .
`STOPATEND` [bool]	Sets the default behavior at the end of the recording.

CLOCK

Variable	Value
`CLOCKTYPE` [type]	Defines the type of clock used. Valid values for `[type]` are: • `INPUT` • `GENLOCK` ⚠ **Note:** `INPUT` clock should be use if automatic detection of input is a requirement.
`GENLOCKFLYWHEEL` [bool]	Uses the flywheel in case of genlock loss, if genlock is used. The default value is `Off` .
`FLYWHEELTIMEOUT` [timeout]	Sets the timeout for the genlock flywheel in seconds. This defines the time until the genlock switches to free run, as well as the maximum time the flywheel can use to resynchronize. The initial value is `5.0` .

COMMUNICATION

Command	Description
`COMMAND_HANDLER PORT SET` [port]	Sets the port of the command interface. A port can only be set once.
`COMMAND_HANDLER DUMP`	Prints this list of available commands.

Command	Description
`MVCP PORT` [port]	Sets the port for the MVCP communication. This command is required to initialize MVCP.

EXIT

Command	Description
`exit, EXIT`	Stops all channels, cleans up the hardware and stops the service.

4.3.8 Supported Codecs

The Channel Recorder supports the following containers and codecs.

> ⚠️ **Note:** If a combination of CODEC / CONTAINER / RESOLUTION is not supported, Channel Recorder floods the logs with errors. If this happens stop the recording using `RECORD STOP`. Please make sure that the mentioned combination are supported by consulting the tables below.

> ⚠️ **Note:** Only the containers AVCINTRAMXF, DVCPROMXF, XDCAMMXF, XAVCMXF support capturing of closed captions. The closed captions must be part of SMPTE 334M CDP. The captured close caption is found in the recorded clip in the wrapper according to SMPTE 436M and in the actual MXF essence.

> ⚠️ **Note:** The examples below are not exhaustive. Some parameters such as the number of audio channels have other options.

- PAL Codecs
  - MPEG-IFrame
  - DVCAM
  - DVCPRO
  - XDCAM
- NTSC Codecs
  - MPEG IFrame
  - DVCAM
  - DVCPRO
  - XDCAM
- 720p50 and 720p59.94 (60M) Codecs
  - MPEG-IFrame
  - DVCPRO HD
  - XDCAM HD 422
  - XDCAM EX HQ
  - AVC-Intra
  - ProRes
  - DNxHD
  - XAVC
- 1080i25 and 1080i29.97 (30M) Codecs
  - MPEG-IFrame
  - DVCPRO HD
  - XDCAM HD 422
  - XDCAM HD LP
  - XDCAM HD SP
  - XDCAM HD HQ
  - XDCAM EX HQ
  - AVC-Intra
  - ProRes

- DNxHD
- XAVC
- 1080p50 and 1080p59.94 (60M) Codecs
  - AVC-Intra
  - MPEG-IFrame
  - ProRes
  - DNxHD
  - XAVC
- 2160p50 and 2160p59.94 (60M) Codecs
  - XAVC

PAL Codecs

MPEG-IFrame

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	MPEG2-IFrame422 720x576	Alpha: No Bitrate: 10-50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC IFrame` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` RECORD CLIP clip `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

DVCAM

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.avi	DV/DVCAM 4:2:0
	
	720x576
		Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio
			
			PCM:
			• 2 ch: 24 in 32-bit; 48kHz
			• 4 ch: 24 in 32-bit; 48kHz
			• 8 ch: 24 in 32-bit; 48kHz
			• 16 ch: 24 in 32-bit; 48kHz
				`CONFIG SET CODEC DvCam`
				`CONFIG SET CONTAINER AVI`
				`CONFIG SET AUDIOCHANNELS 8`
				`RECORD CLIP clip`
				`RECORD START`
					Specifying this codec / container combination is not supported through MVCP.
					
					If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCAM 4:2:0 720x576	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCam` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*
.mxf Panasonic P2 (OP-Atom)	DVCAM 4:2:0 720x576	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC DvCam` `CONFIG SET CONTAINER MXF` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

> **ℹ Information:** Video bitrate is 25, but the encoded bitrate is 28.75 (15% more) which includes other data.

DVCPRO

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DVCPRO 4:1:1 720x576	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DVCPRO 50 4:2:2 720x576	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCPRO 4:1:1 720x576	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCPRO 50 4:2:2 720x576	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DVCPRO 4:1:1 720x576	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER DVCPROMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "dvcpro" SET U1 MED vtr.media.mpeg.bit_rate "25000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DVCPRO 50 4:2:2 720x576	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER DVCPROMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "dvcpro" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DVCPRO 4:1:1 720x576	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG` `SET` `CODEC` `DvCPro` `CONFIG` `SET` `CONTAIN` `ER MXF` RECORD CLIP clip `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*
.mxf Panasonic P2 (OP-Atom)	DVCPRO 50 4:2:2 720x576	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG` `SET` `CODEC` `Dv50` `CONFIG` `SET` `CONTAIN` `ER MXF` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

> **ⓘ Information:** Video bitrate is 25, but the encoded bitrate is 28.75 (15% more) which includes other data. The same applies for 50 bitrate. The encoded bitrate is 57.6 (15% more).

XDCAM

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf XDCAM (OP1a)	D10 (IMX50) 4:2:2 720x608	Alpha: No Bitrate: 50 Bit depth: 10 VBI: required Timecode: required TDIR: Yes CC: Yes	AES3: • 4ch: 24 in 32-bit; 48kHz • 8ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XDCam` `CONFIG SET CONTAINER XDCAMMXF` CONFIG SET BITRATE 50 `CONFIG SET AUDIOCHANNELS 4` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "mpeg2" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "4" CUER U1 REC U1
Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf XDCAM (OP1a)	D10 (IMX40) 4:2:2 720x608	Alpha: No Bitrate: 40 Bit depth: 10 VBI: required Timecode: required TDIR: Yes CC: Yes	AES3: • 4ch: 24 in 32-bit; 48kHz • 8ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XDCam` `CONFIG SET CONTAINER XDCAMMXF` `CONFIG SET BITRATE 40` `CONFIG SET AUDIOCHANNELS 4` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "mpeg2" SET U1 MED vtr.media.mpeg.bit_rate "40000000" SET U1 MED vtr.media.audio.input.channels "4" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf XDCAM (OP1a)	D10 (IMX30) 4:2:2 720x608	Alpha: No Bitrate: 40 Bit depth: 10 VBI: required Timecode: required TDIR: Yes CC: Yes	AES3:
			
			• 4ch: 24 in 32-bit; 48kHz
			• 8ch: 24 in 32-bit; 48kHz
				`CONFIG SET`
				`CODEC XDCam`
				`CONFIG SET`
				`CONTAINER`
				`XDCAMMXF`
				`CONFIG SET`
				`BITRATE 40`
				`CONFIG SET`
				`AUDIOCHANNELS`
				`4`
				`RECORD CLIP`
				`clip`
				`RECORD START`
					UADD CS_ENCODER1 * SHAR dart
					
					SET U1 MED vtr.media.video.input.compression.type "mpeg2"
					
					SET U1 MED vtr.media.mpeg.bit_rate "30000000"
					
					SET U1 MED vtr.media.audio.input.channels "4"
					
					CUER U1
					
					REC U1

NTSC Codecs

MPEG IFrame

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	MPEG2-IFrame422 720x480	Alpha: No Bitrate: 10-50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC IFrame` `CONFIG SET CONTAINER AVI` `CONFIG SET BITRATE 50` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

DVCAM

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DV/DVCAM 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCam` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCAM 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCam` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*
.mxf Panasonic P2 (OP-Atom)	DVCAM 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: N/A CC: No	No Audio	`CONFIG SET CODEC DvCam` `CONFIG SET CONTAINER MXF` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

> ℹ️ **Information:** Video bitrate is 25, but the encoded bitrate is 28.75 (15% more) which includes other data.

DVCPRO

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DVCPRO 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DVCPRO 50 4:2:2 720x480	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCPRO 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVC PRO 50 4:2:2 720x480	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DVCPRO 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER DVCPROMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "dvcpro" SET U1 MED vtr.media.mpeg.bit_rate "25000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DVCPRO 50 4:2:2 720x480	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER DVCPROMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "dvcpro" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DVCPRO 4:1:1 720x480	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER MXF` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*
.mxf Panasonic P2 (OP-Atom)	DVCPRO 50 4:2:2 720x480	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC Dv50` `CONFIG SET CONTAINER MXF` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type*

> **ℹ Information:** Video bitrate is 25, but the encoded bitrate is 28.75 (15% more) which includes other data. The same applies for 50 bitrate. The encoded bitrate is 57.6 (15% more).

XDCAM

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf XDCAM (OP1a)	D10 (IMX50) 4:2:2 720x512	Alpha: No Bitrate: 50 Bit depth: 10 VBI: Required Timecode: Required TDIR: Yes CC: Yes	AES3: • 4 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XDCam` `CONFIG SET CONTAINER XDCAMMXF` `CONFIG SET BITRATE 50` `CONFIG SET AUDIOCHANNELS 4` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "mpeg2" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "4" CUER U1 REC U1
Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf XDCAM (OP1a)	D10 (IMX40) 4:2:2 720x512	Alpha: No Bitrate: 40 Bit depth: 10 VBI: Required Timecode: Required TDIR: Yes CC: Yes	AES3: • 4 ch: 24 in 32-bit; 48kHz • 8ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XDCam` `CONFIG SET CONTAINER XDCAMMXF` `CONFIG SET BITRATE 40` `CONFIG SET AUDIOCHANNELS 4` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "mpeg2" SET U1 MED vtr.media.mpeg.bit_rate "40000000" SET U1 MED vtr.media.audio.input.channels "4" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf XDCAM (OP1a)	D10 (IMX30) 4:2:2 720x512	Alpha: No Bitrate: 30 Bit depth: 10 VBI: Required Timecode: Required TDIR: Yes CC: Yes	AES3: • 4 ch: 24 in 32-bit; 48kHz • 8ch: 24 in 32-bit; 48kHz	`CONFIG SET` `CODEC XDCam` `CONFIG SET` `CONTAINER` `XDCAMMXF` `CONFIG SET` `BITRATE 30` `CONFIG SET` `AUDIOCHANNELS` `4` `RECORD CLIP` `clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "mpeg2" SET U1 MED vtr.media.mpeg.bit_rate "30000000" SET U1 MED vtr.media.audio.input.channels "4" CUER U1 REC U1

720p50 and 720p59.94 (60M) Codecs

MPEG-IFrame

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	MPEG 2-IFrame422 1280x720	Alpha: No Bitrate: 50-300 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC IFrame` `CONFIG SET CONTAINER AVI` `CONFIG SET BITRATE 50` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

DVCPRO HD

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DVCPRO HD 1280x720	Alpha: No Bitrate: 100-115 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER AVI` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCPRO HD 960x720	Alpha: No Bitrate: 100-115 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DVCPRO HD 960x720	Alpha: No Bitrate: 100-115 Bit depth: 8 VBI: N/A Time code: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER DVCPROMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "dvcpro" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf
 	
Panasonic P2 (OP-Atom)
	DVCPRO HD
	
	960x720
		Alpha: No
		Bitrate: 100-115
		Bit depth: 8
		VBI: N/A
		Timecode: N/A
		TDIR: Yes
		CC: No
			No Audio	`CONFIG`
				`SET`
				`CODEC`
				`DvCPro`
				`CONFIG`
				`SET`
				`CONTAIN`
				`ER MXF`
				`RECORD`
				`CLIP`
				`clip`
				`RECORD`
				`START`
					Specifying this codec / container combination is not supported through MVCP.
					
					If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XDCAM HD 422

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XDCAM HD 422 1280x720	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET` `CODEC XDCam` `CONFIG SET` `CONTAINER` `XDCAMMXF` `CONFIG SET` `AUDIOCHANNELS` `8` `RECORD CLIP` `clip` `RECORD START`	UADD CS_ENCODER1 * SHARdart SET U1 MED vtr.media.video.input.compression.type "mpeg2-ibp-422p@hl" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

XDCAM EX HQ

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XDCAM EX HQ 1280x720	Alpha: No Bitrate: 35 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	No Audio PCM: • 2 ch: 16 in 16-bit; 48kHz • 4 ch: 16 in 16-bit; 48kHz • 8 ch: 16 in 16-bit; 48kHz • 16 ch: 16 in 16-bit; 48kHz	`CONFIG` `SET CODEC` `XDCamEX` `CONFIG` `SET` `CONTAINER` `XDCAMMXF` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

AVC-Intra

Conta iner	Code c	Feat ures	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	AVCIn tra Class 50 960x 720	Alph a: No Bitrat e: 50 Bit dept h: 8 VBI: N/A Time code: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET` `CODEC` `AVCIntra50` `CONFIG SET` `CONTAINER` `AVCINTRAMX F` `CONFIG SET` `BITRATE 50` `CONFIG SET` `AUDIOCHANN ELS 8` `RECORD` `CLIP clip` `RECORD` `START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "avc-intra" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Conta iner	Code c	Feat ures	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	AVCIn tra Class 100 1280 x720	Alph a: No Bitrat e: 100 Bit dept h: 8 VBI: N/A Time code: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC AVCIntra10 0` `CONFIG SET CONTAINER AVCINTRAMX F` `CONFIG SET BITRATE 100` `CONFIG SET AUDIOCHANN ELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "avc-intra" SET U1 MED vtr.media.mpeg.bit_rate "100000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1
.mxf Panaso nic P2 (OP-Atom)	AVCIn tra Class 50 960x 720	Alph a: No Bitrat e: 50 Bit dept h: 8 VBI: N/A Time code: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC AVCIntra50` `CONFIG SET CONTAINER AVCINTRAMX F` `CONFIG SET BITRATE 50` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf Panasonic P2 (OP-Atom)	AVCIntra Class 100
	
	1280 x720
		Alpha: No Bitrate: 100 Bit depth: 8 VBI: N/A Time code: N/A TDIR: Yes CC:No	No Audio	`CONFIG SET CODEC AVCIntra100`
				`CONFIG SET CONTAINER AVCINTRAMXF`
				`CONFIG SET BITRATE 100`
				`RECORD CLIP clip`
				`RECORD START`
					Specifying this codec / container combination is not supported through MVCP.
					
					If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

ProRes

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	I-Frame 4:2:2 1280x720	Alpha: No Bitrate: 100 (LT), 147, 220 (HQ) Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes, with *.ref* file CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC ProRes` `CONFIG SET CONTAINER MOV` `CONFIG SET BITRATE 100` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

DNxHD

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DNxHD *(50)* 120, 175 DNxHD *(59.94)* 145, 220 1280x720	Alpha: No Bitrate (50): 120,175 Bitrate (59.94): 145,220 Bit depth: 8 VBI: N/A Timecode: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER DNXHDMXF` `CONFIG SET BITRATE 120` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DNxHD *(50)* 175x DNxHD *(59.94)* 220x 1280x720	Alpha: No Bitrate (50): 175x Bitrate (59.94): 220x Bit depth: 10 VBI: N/A Timecode: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER DNXHDMXF` `CONFIG SET BITRATE 175` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DNxHD *(50)* 120, 175 DNxHD *(59.94)* 145, 220 1280 x720	Alpha: No Bitrate (50): 120,175 Bitrate (59.94): 145,220 Bit depth: 8 VBI: N/A Timecode: N/A CC: No	No Audio	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 120` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DNxHD *(50)* 175x DNxHD *(59.94)* 220x 1280x720	Alpha: No Bitrate (50): 175x Bitrate (59.94): 220x Bit depth: 10 VBI: N/A Timecode: N/A CC: No	No Audio	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 175` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XAVC

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP 1a)	XAVC	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG` `SET` `CODEC` `XAVC` `CONFIG` `SET` `CONTAINER` `XAVCMXF` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP 1a)	XAVC Long GOP	Alpha: No Bitrate: 50 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XAVCLong GOP` `CONFIG SET CONTAINER XAVCMXF` `CONFIG SET BITRATE 50` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

> ⚠️ **Note:** The configuration **v210** must be enabled in the configuration section for recording XAVC.

1080i25 and 1080i29.97 (30M) Codecs

MPEG-IFrame

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	MPEG2-IFrame422 1920x1080	Alpha: No Bitrate: 50-300 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: 2 ch: 24 in 32-bit; 48kHz 4 ch: 24 in 32-bit; 48kHz 8 ch: 24 in 32-bit; 48kHz 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC IFrame` `CONFIG SET CONTAINER AVI` `CONFIG SET BITRATE 50` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

DVCPRO HD

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	DVCPRO HD *(25)* 1920x 1080 DVCPRO HD *(29.97)* 1260x 1080	Alpha: No Bitrate: 100 Bit depth: 8 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAIN ER AVI` `CONFIG SET AUDIOCH ANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	DVCPRO HD *(25)* 1920x 1080 DVCPRO HD *(29.97)* 1260x 1080	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DvCPro` `CONFIG SET CONTAINER MOV` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf
Panasonic P2
(OP-Atom)
	DVCPRO HD *(25)* 1920x1080
	DVCPRO HD *(29.97)* 1260x1080
		Alpha: No
		Bitrate: 100
		Bit depth: 10
		VBI: N/A
		Timecode: N/A
		TDIR: Yes
		CC: No
			No Audio	`CONFIG SET CODEC DvCPro`
				`CONFIG SET CONTAINER MXF`
				`RECORD CLIP clip`
				`RECORD START`
					Specifying this codec / container combination is not supported through MVCP.
					
					If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

.mxf (OP1a)	DVCPRO HD *(25)* 1920x1080
	DVCPRO HD *(29.97)* 1260x1080
		Alpha: No
		Bitrate: 100
		Bit depth: 10
		VBI: N/A
		Timecode: Yes
		TDIR: Yes
		CC: Yes
			PCM:
			
			• 4 ch: 24 in 32-bit; 48kHz
				`CONFIG SET CODEC DvCPro`
				`CONFIG SET CONTAINER DVCPROMXF`
				`RECORD CLIP clip`
				`RECORD START`
					UADD CS_ENCODER1 * SHAR dart
					SET U1 MED vtr.media.video.input.compression.type "dvcpro"
					SET U1 MED vtr.media.mpeg.bit_rate "100000000"
					SET U1 MED vtr.media.audio.input.channels "8"
					CUER U1
					REC U1

> ℹ **Information:** Video bitrate is 100, but the encoded bitrate is 115 (15% more) which includes other data.

XDCAM HD 422

Contai ner	Codec	Feature s	Audio	VizSend Example	MVCP Example

.mxf (OP1a)	XDCAM HD 422
	
	1920x1 080
		Alpha: No
		Bitrate: 50
		Bit depth: 8
		VBI: N/ A
		Timeco de: Yes
		TDIR: Yes
		CC: Yes
			No Audio
			PCM:
			
			• 2 ch: 24 in 32- bit; 48kHz
			• 4 ch: 24 in 32- bit; 48kHz
			• 8 ch: 24 in 32- bit; 48kHz
			• 16 ch: 24 in 32- bit; 48kHz
				`CONFIG SET`
				`CODEC XDCam`
				`CONFIG SET`
				`CONTAINER`
				`XDCAMMXF`
				`CONFIG SET`
				`AUDIOCHANNELS`
				`8`
				`RECORD CLIP`
				`clip`
				`RECORD START`
					UADD CS_ENCODER1 * SHAR dart
					SET U1 MED vtr.media.video.input.compre ssion.type "mpeg2- ibp-422p@hl"
					SET U1 MED vtr.media.audio.input.channe ls "8"
					CUER U1
					REC U1

XDCAM HD LP

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XDCAM HD LP 1440x1080	Alpha: No Bitrate: 17.5 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	No Audio PCM: • 2 ch: 16 in 16-bit; 48kHz • 4 ch: 16 in 16-bit; 48kHz • 8 ch: 16 in 16-bit; 48kHz • 16 ch: 16 in 16-bit; 48kHz	`CONFIG SET CODEC XDCamHD` `CONFIG SET CONTAINER XDCAMMXF` `CONFIG SET BITRATE 18` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XDCAM HD SP

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XDCAM HD SP 1440x1080	Alpha: No Bitrate: 25 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	No Audio PCM: • 2 ch: 16 in 16-bit; 48kHz • 4 ch: 16 in 16-bit; 48kHz • 8 ch: 16 in 16-bit; 48kHz • 16 ch: 16 in 16-bit; 48kHz	`CONFIG SET CODEC XDCamHD` `CONFIG SET CONTAINER XDCAMMXF` `CONFIG SET BITRATE 25` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XDCAM HD HQ

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XDCAM HD HQ 1440x1080	Alpha: No Bitrate: 35 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	No Audio PCM: 2 ch: 16 in 16-bit; 48kHz 4 ch: 16 in 16-bit; 48kHz 8 ch: 16 in 16-bit; 48kHz 16 ch: 16 in 16-bit; 48kHz	`CONFIG SET CODEC XDCamHD` `CONFIG SET CONTAINER XDCAMMXF` `CONFIG SET BITRATE 35` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XDCAM EX HQ

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XDCAM EX HQ 1920x1080	Alpha: No Bitrate: 35 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	No Audio PCM: • 2 ch: 16 in 16-bit; 48kHz • 4 ch: 16 in 16-bit; 48kHz • 8 ch: 16 in 16-bit; 48kHz • 16 ch: 16 in 16-bit; 48kHz	`CONFIG` `SET CODEC` `XDCamEX` `CONFIG` `SET` `CONTAINER` `XDCAMMXF` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

AVC-Intra

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	AVCIntra Class 50 1440x1080	Alpha: No Bitrate: 50 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC AVCIntra50` `CONFIG SET CONTAINER AVCINTRAMXF` `CONFIG SET BITRATE 50` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "avc-intra" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf (OP1a)	AVCIntra Class 100 1920x1080	Alpha: No Bitrate: 100 Bit depth: 8 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM:
			• 2 ch: 24 in 32-bit; 48kHz
			• 4 ch: 24 in 32-bit; 48kHz
			• 8 ch: 24 in 32-bit; 48kHz
			• 16 ch: 24 in 32-bit; 48kHz
				`CONFIG SET CODEC AVCIntra100`
				`CONFIG SET CONTAINER AVCINTRAMXF`
				`CONFIG SET BITRATE 100`
				`CONFIG SET AUDIOCHANNELS 8`
				`RECORD CLIP clip`
				`RECORD START`
					UADD CS_ENCODER1 * SHAR dart
					SET U1 MED vtr.media.video.input.compression.type "avc-intra"
					SET U1 MED vtr.media.mpeg.bit_rate "100000000"
					SET U1 MED vtr.media.audio.input.channels "8"
					CUER U1
					REC U1

.mxf Panasonic P2 (OP-Atom)	AVCIntra Class 50 1440x1080	Alpha: No Bitrate: 50 Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC AVCIntra50`
				`CONFIG SET CONTAINER MXF`
				`CONFIG SET BITRATE 50`
				`RECORD CLIP clip`
				`RECORD START`
					Specifying this codec / container combination is not supported through MVCP.
					
					If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	AVCIntra Class 100 1920x1080	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC AVCIntra100` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 100` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

ProRes

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	I-Frame 4:2:2 1920x1080	Alpha: No Bitrate: 100 (LT), 147, 220 (HQ) Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes, with *.ref* file CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC ProRes` `CONFIG SET CONTAINER MOV` `CONFIG SET BITRATE 100` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

DNxHD

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DNxHD *(30)* 120, 185 DNxHD *(29.97)* 145, 220 1920x1080	Alpha: No Bitrate (50): 120,185 Bitrate (59.94): 145,220 Bit depth: 8 VBI: N/A Timecode: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER DNXHDMXF` `CONFIG SET BITRATE 120` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DNxHD *(30)* 185x DNxHD *(29.97)* 220x 1920x1080	Alpha: No Bitrate (50): 185x Bitrate (59.94): 220x Bit depth: 10 VBI: N/A Timecode: N/A CC: No	No Audio PCM: 2 ch: 24 in 32-bit; 48kHz 4 ch: 24 in 32-bit; 48kHz 8 ch: 24 in 32-bit; 48kHz 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER DNXHDMXF` `CONFIG SET BITRATE 185` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DNxHD *(30)* 120, 185 DNxHD *(29.97)* 145, 220 1920 x1080	Alpha: No Bitrate (50): 120,185 Bitrate (59.94): 145,220 Bit depth: 8 VBI: N/A Timecode: N/A CC: No	No Audio	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 120` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DNxHD *(30)* 185x DNxHD *(29.97)* 220x 1920x1080	Alpha: No Bitrate (50): 185x Bitrate (59.94): 220x Bit depth: 10 VBI: N/A Timecode: N/A CC: No	No Audio	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 185` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XAVC

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP 1a)	XAVC Intra	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG` `SET` `CODEC` `XAVC` `CONFIG` `SET` `CONTAINER` `XAVCMXF` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP 1a)	XAVC Long GOP	Alpha: No Bitrate: 25, 35, 50 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XAVCLong GOP` `CONFIG SET CONTAINE R XAVCMXF` `CONFIG SET BITRATE 35` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

> ⚠️ **Note:** The configuration **v210** must be enabled in the configuration section for recording XAVC.

1080p50 and 1080p59.94 (60M) Codecs

AVC-Intra

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	AVCIntra Class 50 1440x1080	Alpha: No Bitrate: 50 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC AVCIntra50` `CONFIG SET CONTAINER AVCINTRAMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "avc-intra" SET U1 MED vtr.media.mpeg.bit_rate "50000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	AVCIntra Class 100 1920x1080	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC AVCIntra100` `CONFIG SET CONTAINER AVCINTRAMXF` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	UADD CS_ENCODER1 * SHAR dart SET U1 MED vtr.media.video.input.compression.type "avc-intra" SET U1 MED vtr.media.mpeg.bit_rate "100000000" SET U1 MED vtr.media.audio.input.channels "8" CUER U1 REC U1
.mxf Panasonic P2 (OP-Atom)	AVCIntra Class 50 1440x1080	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC AVCIntra50` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 50` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf
Panasonic P2
(OP-Atom)
	AVCIntra Class 100
	1920x1080
		Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: N/A TDIR: Yes CC: No	No Audio	`CONFIG SET CODEC AVCIntra100`
				`CONFIG SET CONTAINER MXF`
				`CONFIG SET BITRATE 100`
				`RECORD CLIP clip`
				`RECORD START`
					Specifying this codec / container combination is not supported through MVCP.
					
					If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

MPEG-IFrame

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.avi	MPEG 2-IFrame422 1920x1080	Alpha: No Bitrate: 50-300 Bit depth: 8 VBI: N/A Time code: N/A TDIR: Yes CC: No	No Audio PCM: 2 ch: 24 in 32-bit; 48kHz 4 ch: 24 in 32-bit; 48kHz 8 ch: 24 in 32-bit; 48kHz 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC IFrame` `CONFIG SET CONTAINER AVI` `CONFIG SET BITRATE 50` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

ProRes

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mov	I-Frame 4:2:2 1920x1080	Alpha: No Bitrate: 100 (LT), 147, 220 (HQ) Bit depth: 8, 10 VBI: N/A Timecode: N/A TDIR: Yes, with *.ref* file CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG` `SET` `CODEC` `ProRes` `CONFIG` `SET` `CONTAINER MOV` `CONFIG` `SET` `AUDIOCHANNELS 8` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

DNxHD

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DNxHD *(50)* 240, 365 DNxHD *(59.94)* 290, 440 1920x1080	Alpha: No Bitrate (50): 240, 365 Bitrate (59.94): 290, 440 Bit depth: 8 VBI: N/A Timecode: N/A CC: No	No Audio PCM: 2 ch: 24 in 32-bit; 48kHz 4 ch: 24 in 32-bit; 48kHz 8 ch: 24 in 32-bit; 48kHz 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER DNXHDMXF` `CONFIG SET BITRATE 240` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	DNxHD *(50)* 365x DNxHD *(59.94)* 440x 1920x1080	Alpha: No Bitrate (50): 365 x Bitrate (59.94): 440x Bit depth: 10 VBI: N/A Timecode: N/A CC: No	No Audio PCM: • 2 ch: 24 in 32-bit; 48kHz • 4 ch: 24 in 32-bit; 48kHz • 8 ch: 24 in 32-bit; 48kHz • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER DNXHDMXF` `CONFIG SET BITRATE 365` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example

.mxf Panasonic P2 (OP-Atom)	DNxHD *(50)* 240, 365
	 DNxHD *(59.94)* 290, 440
	 1920x1080
		Alpha: No
		 Bitrate (50): 240, 365
		 Bitrate (59.94): 290, 440
		 Bit depth: 8
		 VBI: N/A
		 Timecode: N/A
		 CC: No
			No Audio	`CONFIG SET CODEC DNxHD`
				 `CONFIG SET CONTAINER MXF`
				 `CONFIG SET BITRATE 240`
				 `CONFIG SET AUDIOCHANNELS 8`
				 `RECORD CLIP clip`
				 `RECORD START`
					Specifying this codec / container combination is not supported through MVCP.
					
					 If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf Panasonic P2 (OP-Atom)	DNxHD *(50)* 365x DNxHD *(59.94)* 440x 1920x1080	Alpha: No Bitrate (50): 365 x Bitrate (59.94): 440x Bit depth: 10 VBI: N/A Timecode: N/A CC: No	No Audio	`CONFIG SET CODEC DNxHD` `CONFIG SET CONTAINER MXF` `CONFIG SET BITRATE 365` `CONFIG SET AUDIOCHANNELS 8` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

XAVC

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP 1a)	XAVC Intra	Alpha: No Bitrate: 100 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG` `SET` `CODEC` `XAVC` `CONFIG` `SET` `CONTAINER` `XAVCMXF` `RECORD` `CLIP` `clip` `RECORD` `START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XAVC Long GOP	Alpha: No Bitrate: 25, 35, 50 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XAVCLongGOP` `CONFIG SET CONTAINER XAVCMXF` `CONFIG SET BITRATE 35` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

⚠️ **Note:** The configuration **v210** must be enabled in the configuration section for recording XAVC.

2160p50 and 2160p59.94 (60M) Codecs

XAVC

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP1a)	XAVC Intra	Alpha: No Bitrate: 300, 480, (VBR/CBR) Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	`CONFIG SET CODEC XAVC` `CONFIG SET CONTAINER XAVCMXF` `CONFIG SET BITRATE 300` `RECORD CLIP clip` `RECORD START`	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

Container	Codec	Features	Audio	VizSend Example	MVCP Example
.mxf (OP 1a)	XAVC Long GOP	Alpha: No Bitrate: 200 Bit depth: 10 VBI: N/A Timecode: Yes TDIR: Yes CC: Yes	PCM: • 16 ch: 24 in 32-bit; 48kHz	``` CONFIG SET CODEC XAVCLong GOP CONFIG SET CONTAINER XAVCMXF CONFIG SET BITRATE 200 RECORD CLIP clip RECORD START ```	Specifying this codec / container combination is not supported through MVCP. If this codec / container combination is required then the web interface can be used to configure Channel Recorder. When sending MVCP commands, however, the codec / container must **not** be specified via *vtr.media.video.input.compression.type.*

⚠️ **Note:** A M264 board is recommended for recording UHD.

⚠️ **Note:** The configuration **uhd** and **v210** must be enabled in the configuration section for recording UHD.

4.4 Specific Use Cases

This section details how to configure Channel Recorder to help the users achieve specific use cases.

This section contains the following use cases:

- Recording Discontinuous Timecode
- Recording HDR
- Recording UHD
- Recording Viz Engine
- Recording RTP H264 Stream

4.4.1 Recording Discontinuous Timecode

Starting with version 1.5.0, Channel Recorder can record discontinuous TC. Sometimes it is desirable to have timecodes that are discontinuous, such as when recording a replay from any kind of sports event. This is possible by configuring Channel Recorder as follows:

- **discontinuous-tc:** Should be configured to `true` or `1`.
- **timecode-source:** Should be configured as VITC or LTC.

Not all combination of codecs / containers support discontinuous TC. The following formats have been tested and confirmed to support discontinuous TC:

- XAVCMXF / XAVC
- DNXHDMXF / DNxHD
- DVCPROMXF/ DvCPro
- AVCINTRAMXF / AVCIntra100 or AVCIntra50

This feature can also be used to support DST (Daylight Saving Time) so that the timecode reflects the change of time instead of having a continuous TC.

In this case, **timecode-source** should be configured as time-of-day.

4.4.2   Recording HDR

From version 1.2, Channel Recorder supports the recording of HDR. To be able to record in HDR, the input resolution must be in HD. The possible values for colorimetry are:

- ITUR_BT_601
- ITUR_BT_709
- ITUR_BT_2020
- ITUR_BT_2100_PQ
- ITUR_BT_2100_HLG
- ITUR_BT_2100_SLOG3
- LinearLight

By default, HD resolution records in ITUR_BT_709 and SD resolution records in ITUR_BT_601.

The colorimetry can be changed in the **Capture Settings** of the configuration.
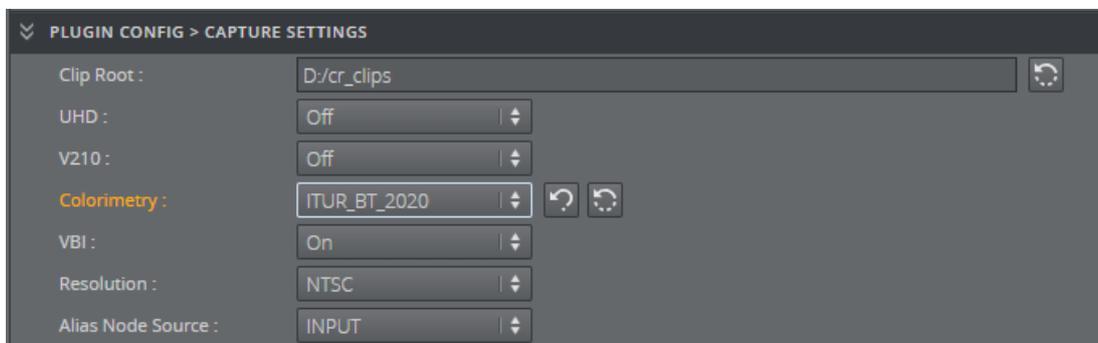
When one of the possible HDR colorimetry options is chosen, the recording is captured in ten bits per sample instead of eight bits per sample.

4.4.3   Recording UHD

From version 1.2, Channel Recorder supports the recording of UHD in 2SI mode as well as HDR. To be able to record in 2SI mode, the board must be of the variant X2 or upgraded with that capability. Please check that the installed board is X2 by accessing the **Matrox X.info** utility.

These are the steps to be able to record UHD in 2SI with HDR support:

1. If not already done, configure the board into 2SI mode via the command prompt by calling in the Matrox driver directory.

> ℹ️ *mvConnectorConfig.exe -2SI=on -sn=XXXXXX*, where **XXXXXX** is the serial number of the board.

2. If not already done, configure the board connectors to be 4in8out (8in4out is not a supported configuration).

> ℹ️ *mvConnectorConfig.exe -4in8out -sn=XXXXXX*, where **XXXXXX** is the serial number of the board.

3. In the Channel Recorder configuration, set the following values to the parameters:

> ℹ️ **UHD:** true
> **V210:** true
> **Resolution:** 2160p50 or 2160p60M
> **Container:** XAVCMXF
> **Codec:** XAVC
> **Colorimetry:** Any HDR values

4.4.4   Recording Viz Engine

From version 1.2, Channel Recorder supports recording input and output signals from Viz Engine. This feature is controlled by the parameter *AliasNodeSource* in the configuration.


The possible values are:

- **INPUT:** Attempts to record directly from the configured Viz Engine input connector. This is a way to **share** the same connector both in Viz Engine and Channel Recorder.
- **OUTPUT:** Attempts to record directly from the configured Viz Engine output connector.

The execution order of the applications is very important. To capture from Viz Engine, this must be executed before Channel Recorder.

Other things to take into account:

- The write-to-read delay of the node to be read must be larger than `0` .
- For this, Viz Engine must be configured with `Matrox.InputHost = 1` .

When Viz Engine uses a 2110 board, it is possible that the video, audio, and ANC connectors do not share the same numerical value. To be able to record from the same inputs, Channel Recorder can be configured using different numerical values for each video, audio, and ANC stream. This is done by using the *Connector* configuration located in each stream section. When this connector, for each stream, is configured, then the main video input connector option is ignored. Example of the connector for the video stream in the GUI:
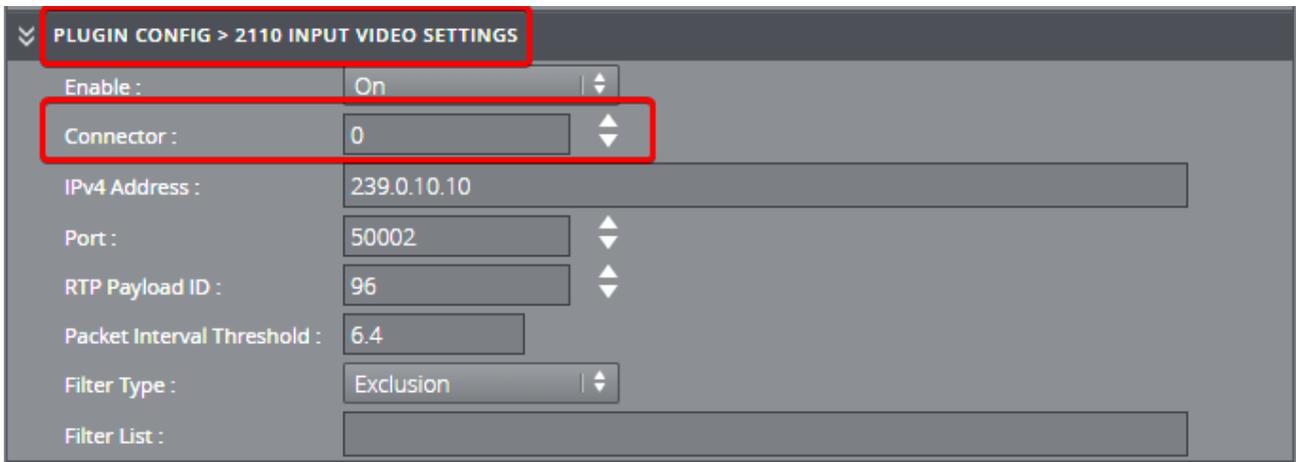
4.4.5   Recording RTP H264 Stream

To record RTP/H264 a DSX.Core or a M264 video board is needed.

DSX.Core

DSX.Core is a cloud based software license from Matrox. It allows a machine in which the **Matrox DSX-TopologyUtils** is installed to acquire a license and to receive RTP/H264 streams along with decoding them for recording. There are two components in a DSX.Core workflow:

- A server machine where the dongle from Matrox with the corresponding licenses is installed.
- A client machine that points to the server to be able to acquire the license.

Steps for Installing the Server Machine

1. Obtain the dongle and the licenses file from Matrox.
2. Install *DSX-TopologyUtils.exe* application.
3. Choose **Matrox DSX Core server** during installation.
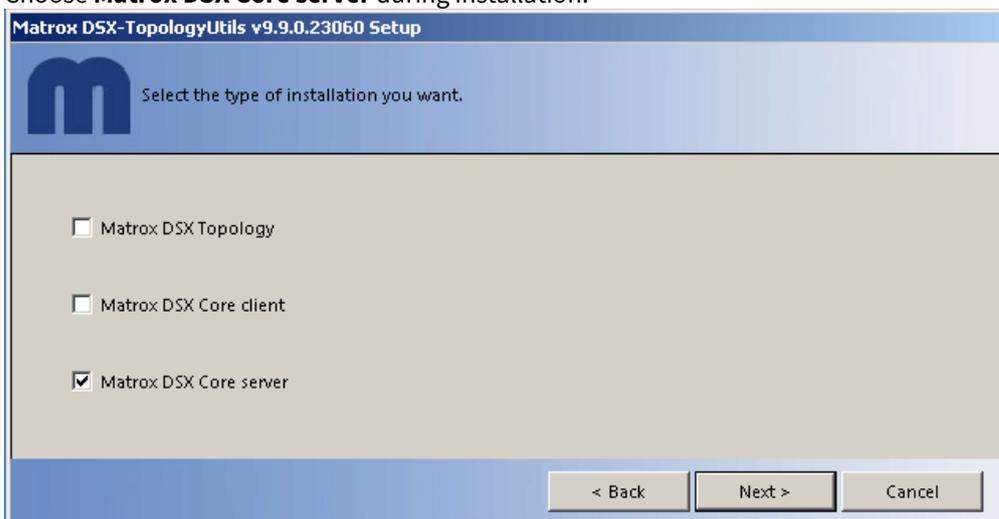
4. After installation, the CodeMeter Control Center is installed and located in the taskbar.
5. Import the licenses provided by Matrox using the application.
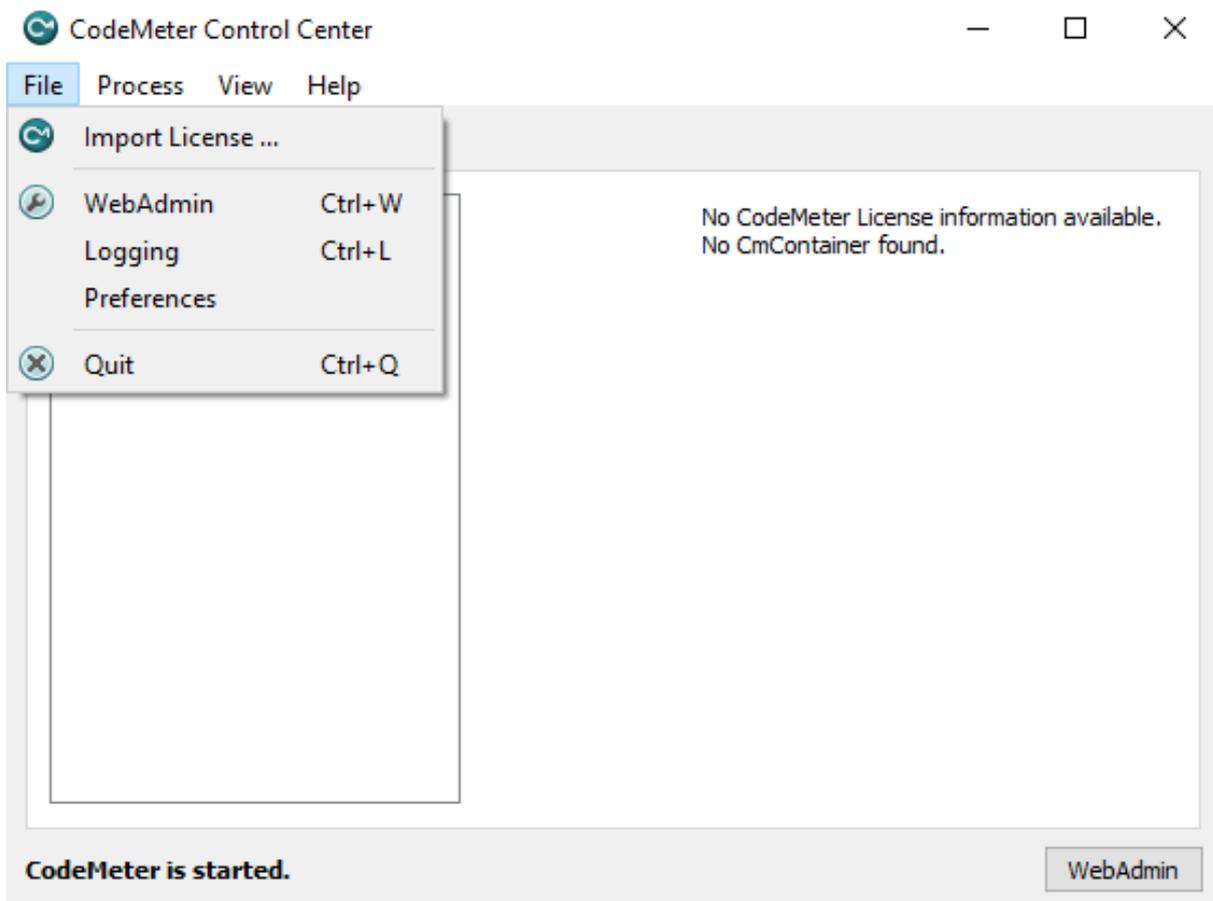
For more information regarding the server installation of DSX.Core please refer to the Matrox DSX.Core documentation.

Steps for Installing the Client Machine

1. Install *DSX-TopologyUtils.exe* application which usually ships with *DSX.utils.exe.*

> ⊗ **Warning:** *DSX.utils.exe* must be uninstalled first before installing *DSX-TopologyUtils.exe*!

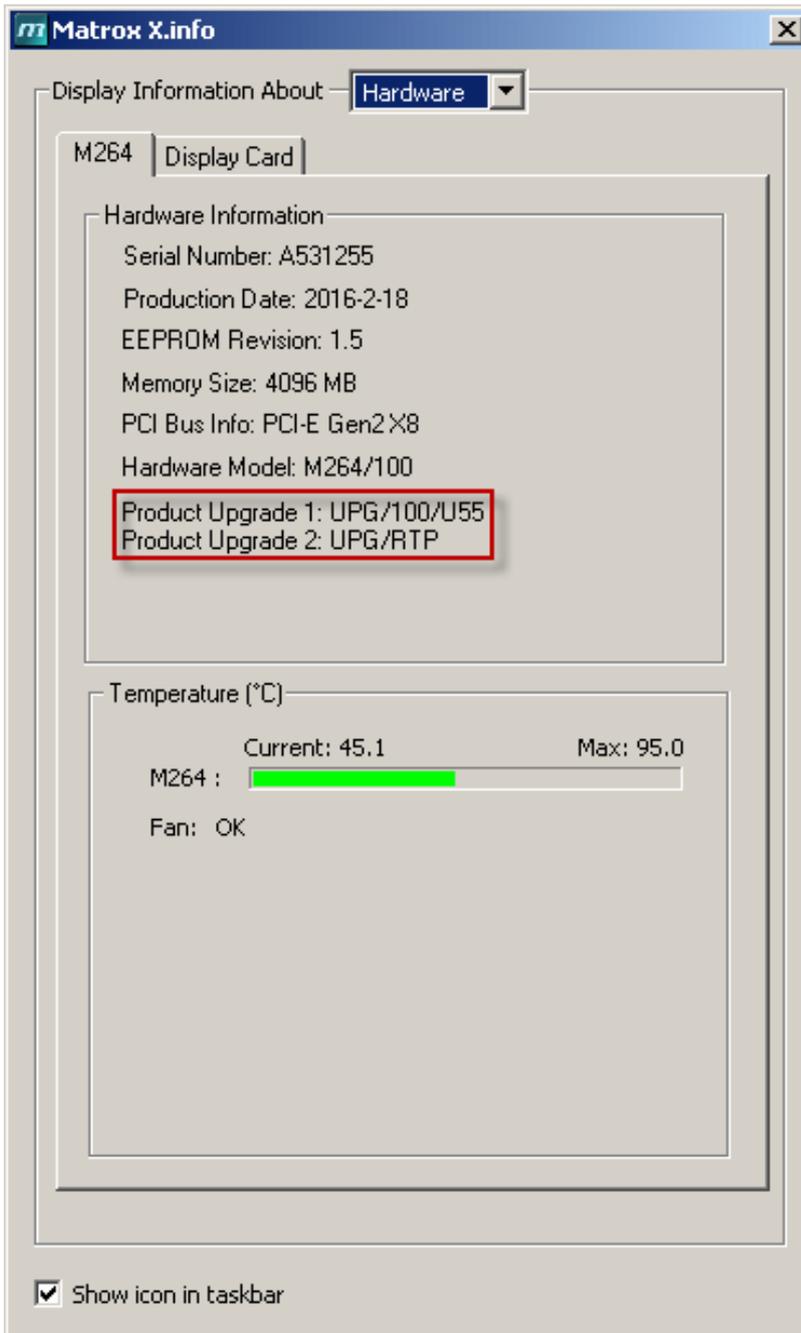2. Choose **Matrox DSX Core client** during installation.
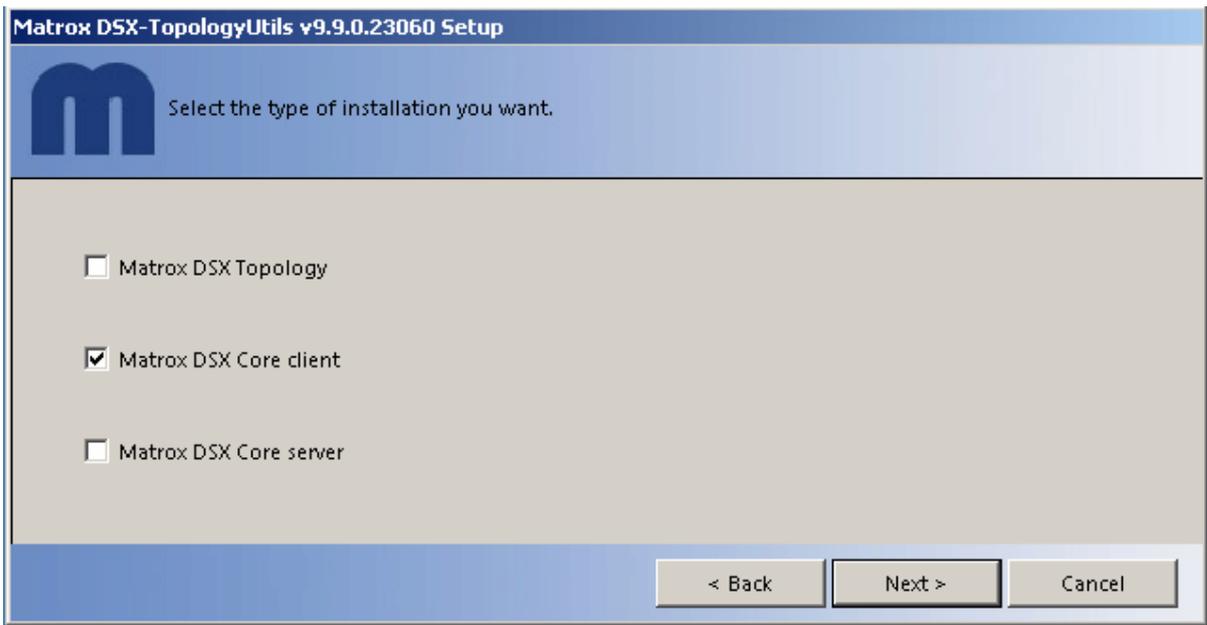
3. The DSX core licenses need to be acquired by Channel Recorder. This can be either a HD or a UHD license.
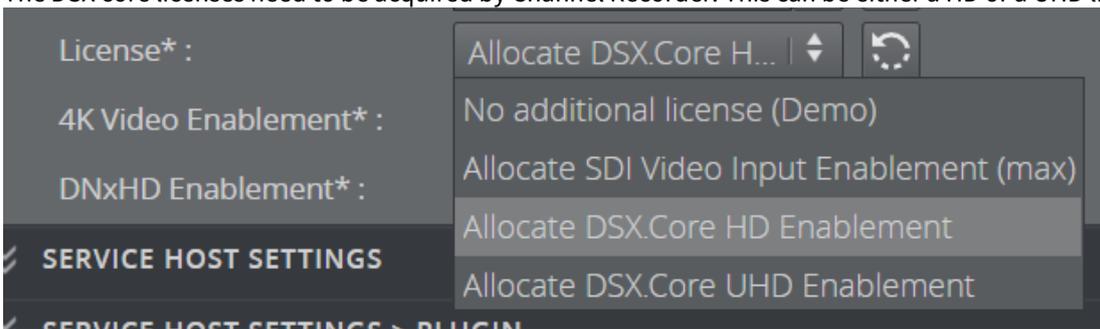
M264

For recording, the board needs the upgrades UPG/100/U55 and UPG/RTP.

Configuring Channel Recorder

When using M264 with another Matrox board, the board must be chosen in the configuration by specifying its serial number.

The most important parameters for receiving RTP/H264 streams are the IP Address, NIC Address and Port. The connector setting still needs to be specified. Any value without space will do.

Additionally the following needs to be taken into account:

- Disable any firewall installed on the system or specifically allow each and every Channel Recorder service to be able to receive incoming RTP/H264 packets.
- The multicast address and port must be already available before starting the service, otherwise Channel Recorder fails to start.
- If no audio is present in the stream, the parameter *Use AV Synchronization* must be disabled.
- The bitrate of the incoming stream must be below 20 Mbps.
- The IP Encapsulation Mode of the incoming stream must be RTP. UDP is not supported.
- Only AAC audio with two channels is supported.
- Only H264 video is supported.
- VBI is not supported.
- Open GOP is not supported for M264 / HW decoding.

If everything is OK, the following appears in the log:

> ℹ️ [2018-10-05 06:13:41.967 +02:00][19500][info]: [ChannelRecorder_TSDemuxer]
> GetElementaryStreamH264VideoDescription: Detected H264 video description: VBR: false, H264 Level: 40,
> H264 Profile: 100, Resolution: 1920x1080, Average Bitrate: 0

Additionally, if debugging is enabled, the following appears in the log:

> ℹ️ [2018-10-05 06:14:14.462 +02:00][22116][debug]: [VizrtCRTest04::CaptureChannelSystemTest]
> TIME_OF_DAY = 06:14:14:00

If none of those logs messages appear then something went wrong. It is usually related to the nature of the incoming stream: Bitrate, GOP configuration, encapsulation, etc.

Performance

When using M264 this is the maximum number of inputs:

Resolutions	4:2:2 10-bit Intra frame	4:2:2 10-bit Long GOP	4:2:0 8-bit Long GOP
3840x2160p50	1	1	2
1920x1080p50	3	5	8
1920x1080i25	6	10	16
1280x720p50	6	10	16
PAL/NTSC	-	-	100