NICE Desktop Cloud Visualization

Administrator Guide

NICE Desktop Cloud Visualization: Administrator Guide

Table of Contents

What Is NICE Desktop Cloud Visualization?

NICE Desktop Cloud Visualization is a remote visualization technology that enables users to securely connect to graphic-intensive 3D applications hosted on a remote high-performance server. With NICE DCV, you can make a server's high-performance graphics processing capabilities available to multiple remote users by creating secure client sessions. This enables your users to use resource-intensive applications with relatively low-end client computers by using the server's processor, GPU, I/O capabilities, and memory.

Topics

How NICE DCV Works

In a typical NICE DCV scenario, a graphic-intensive application, such as a 3D modeling or computer-aided design application, is hosted on a high-performance server that provides a high-end GPU, fast I/O capabilities, and large amounts of memory. The **NICE DCV server** software is installed and configured on the server and it is used to create a secure session. You use a **NICE DCV client** to remotely connect to the session and use the application hosted on the server. The server uses its hardware to perform the high-performance processing required by the hosted application. The **NICE DCV server** software compresses the visual output of the hosted application and streams it back to you as an encrypted pixel stream. Your **NICE DCV client** receives the compressed pixel stream, decrypts it, and then outputs it to your local display.

Features of NICE DCV

NICE DCV offers the following features:

- **Enables collaboration** — It provides sessions that support multiple collaborative clients. Sessions are dynamic and clients can connect and disconnect at any time during the session.
- **Supports H.264-based encoding** — It uses H.264-based video compression and encoding to reduce bandwidth consumption.
- **Supports NVIDIA GRID** — It uses the latest NVIDIA Grid SDK technologies, such as NVIDIA H.264 hardware encoding, to improve performance and reduce system load. Requires an NVIDIA GRID compatible GPU.
- **Shares the entire desktop** — It uses the high-performance NICE DCV protocol to share full control of the entire desktop.
- **Supports NVIDIA vGPU technology** — It uses the NVIDIA virtual GPU (vGPU) technology to simplify the deployment of Windows virtual machines and to support GPU sharing. Requires an NVIDIA GRID compatible GPU.
- **Supports lossless quality video compression** - It supports lossless quality video compression when the network and processor conditions allow.

- **Transport images only** — It transports rendered images as pixels instead of geometry and scene information. This provides an additional layer of security as no proprietary customer information is sent over the network.
- **Adapts compression levels** — It automatically adapts the video compression levels based on the network's available bandwidth and latency.
- **Supports smart card remotization** — It provides seamless access to local smart cards using the Personal Computer/Smart Card (PC/SC) interface. Smart cards can be used for encrypting emails, signing documents, and authenticating against remote systems. Requires the native Windows NICE DCV client and a Linux NICE DCV server.
- **Matches display layouts** — It automatically adapts the server's screen resolution and display layout to match the size of the client window.
- **Provides an HTML5 client** - It offers an HTML5 client that can be used with any modern web browser on Windows and Linux.
- **Supports modern Linux desktop environments** —It supports modern Linux desktops, such as Gnome 3 on RHEL 7.

NICE DCV Requirements

For a good user experience with NICE DCV, ensure that the server and client computers meet the following minimum requirements. Keep in mind that your users' experience is largely dependent on the number of pixels streamed from the NICE DCV server to the NICE DCV client.

Contents

NICE DCV Server Requirements

If you are installing the NICE DCV server on an Amazon EC2 instance, we recommend that you use an Amazon EC2 G3 instance type. These instance types offer NVIDIA GPUs that support hardware-based OpenGL and GPU sharing. For more information, see Amazon EC2 G3 Instances. You can install the NICE DCV server on any other instance type, but there might be screen resolution limitations. A third-party driver can be used to bypass this limitation. If you need the third-party driver, request it from NICE Support.

NICE DCV servers must meet the minimum requirements listed in the following table.

	Windows server	Linux server
Operating system	• Windows 7 — Requires NvFBC compatible NVIDIA GPU • Windows Server 2008 R2 — Requires NvFBC compatible NVIDIA GPU • Windows 8.1 • Windows Server 2012 R2 • Windows 10 • Windows Server 2016 **Note** All supported Windows operating systems require .NET Framework	• CentOS 6.7 or later • CentOS 7.3 or later • RHEL 6.7 or later • RHEL 7.3 or later • SUSE Linux Enterprise 12 with SP2 or later • Ubuntu 18.04 **Note** All supported Linux operating systems must support the x86-64 architecture.

	Windows server	Linux server
	4.5 and must support the x86-64 architecture.	
CPU	64-bit processors only	
GPU	(Optional) An NVIDIA GPU that supports NVIFR or NVENC is required for hardware-based OpenGL. Software-based OpenGL is used if the server does not have an NVIDIA GPU.	
		An NVIDIA GPU is required for GPU sharing across virtual sessions.
Network	By default, the NICE DCV server communicates over port 8443. The port is configurable but must be greater than 1024. Ensure that the server allows communication over the required port.	

NICE DCV Client Requirements

The following table lists the minimum system requirements for the NICE DCV clients.

	Native Windows client	Web browser client	Linux client
Software	The Native Windows client is supported on 32-bit and 64-bit versions of the following operating systems: • Windows 7 • Windows 8.1 • Windows 10 The client also requires the following additional software: • .NET Framework 4.6.2 • Microsoft Visual C++ + Redistributable for Visual Studio. For more information and download instructions, see the Microsoft Support website.	The web browser client is supported on the following browsers across all desktop operating systems: • Firefox • Chrome • Edge • Internet Explorer 11 • Safari 11 The web browser client also requires WebGL and asm.js. **Note** The web browser client is not supported on mobile operating systems, such as Android and iOS.	The Linux client is supported on the following modern Linux operating systems: • RHEL 7.x and CentOS 7.x • SUSE Linux Enterprise 12.x • Ubuntu 16.04 and 18.04
Network	The client must be able to connect to the NICE DCV server, and it must be able to communicate over the required port (8443 by default).		

NICE DCV Pricing

There is no additional charge for using the NICE DCV server on an Amazon EC2 instance. You pay the standard rates for the instance and other Amazon EC2 features that you use.

A license is required to install the NICE DCV server on an on-premises or alternative cloud-based server. For more information, see Licensing NICE DCV (p. 17).

Setting Up NICE DCV

To use NICE DCV, install the NICE DCV server software on the server where you intend to host NICE DCV sessions. Ensure that it is properly licensed.

The following topics explain how to install and license the NICE DCV server. The Licensing (p. 17) topic applies to installations on on-premises and alternative cloud-based servers only, as a license is not required to use the NICE DCV server on an Amazon EC2 instance.

Topics

Installing the NICE DCV Server

The following topics explain how to install the NICE DCV server on a Windows and Linux server. Follow these steps if you are installing NICE DCV on an Amazon EC2 instance, or on an on-premises or alternative cloud-based server.

Topics

Installing the NICE DCV Server on Windows

The NICE DCV server can be installed on a Windows host server using an installation wizard. The wizard walks you through a series of steps that let you customize your NICE DCV server installation. Alternatively, you can use the command line to perform an unattended installation, which uses default settings to automate the installation procedure.

> **Note**
> The NICE DCV server installer is signed by an SHA-256 certificate. If you are using a Windows 7 or Windows Server 2008 R2 server, you must install a Microsoft security update to support this certificate type. For more information, see Microsoft Security Advisory 3033929 and Microsoft Windows Support.

Contents

Using the Wizard

Use the NICE DCV server installation wizard for a guided installation.

To install the NICE DCV server on Windows using the wizard

1. Launch and connect to the server on which to install the NICE DCV server.

2. Download the NICE DCV server installer from the NICE website.

   Note
   The NICE DCV server is only available in a 64-bit version and supported on 64-bit Windows operating systems.

3. Run `nice-dcv-server-x64-Release-2017.4-`*`version_number`*`.msi`.
4. On the Welcome screen, choose **Next**.
5. On the End-User License Agreement screen, read the license agreement. If you accept the terms, select the **I accept the terms in the License Agreement** check box and choose **Next**.
6. (Optional) On the **Drivers Selection** screen, select **USB device remotization** and choose **Will be installed on local hard drive**, **Next**. This installs the drivers required to support some specialized USB devices, such as 3D pointing devices and graphic tablets.
7. On the DCV Service Configuration screen:

   a. (Optional) To manually configure your server's firewall to allow communication over the required port, select **No, I will manually configure my firewall later**.

   b. (Optional) To manually start the NICE DCV server after the installation, select **No, I want to start a DCV Service manually**. If you select this option, you can't start a console session automatically after the installation has completed. The next step is skipped.

8. Choose **Next**.
9. On the DCV Session Management Configuration screen, specify the owner for the automatic console session. Alternatively, to prevent the automatic console session from starting after the installation has completed, select **No, I will create the session manually**.

   Note
   Complete this step only if you previously chose to allow the server to start automatically.

10. Choose **Install**.

Unattended Installation

The unattended installation does the following by default:

- Adds a firewall rule to allow communication over port 8443.
- Enables NICE DCV server auto-start.
- Creates an automatic console session.
- Sets the console session owner to the user who performs the installation.

You can override the default actions by appending the following options to the installation command:

- `DISABLE_FIREWALL=1` — Prevents the installer from adding the firewall rule.
- `DISABLE_SERVER_AUTOSTART=1` — Prevents the NICE DCV server from starting automatically after the installation.
- `DISABLE_SERVER_AUTOMATIC_SESSION_CREATION=1` — Prevents the installer from starting the automatic console session.
- `AUTOMATIC_SESSION_OWNER=`*`owner_name`* — Specifies a different owner for the automatic console session.
- `ADDLOCAL=ALL` — Installs the DCV drivers required for USB remotization.

To install the NICE DCV server on Windows using an unattended installation

1. Launch and connect to the server on which to install the NICE DCV server.

2. Download the NICE DCV server installer from the NICE website.

   Note
   The NICE DCV server is only available in a 64-bit version and supported on 64-bit Windows operating systems.

3. Open a command prompt window and navigate to the folder where you downloaded the installer.
4. Execute the unattended installer:

```
C:\> msiexec.exe /i nice-dcv-server-x64-Release-2017.4-version_number.msi /quiet /
norestart /l*v dcv_install_msi.log
```

Installing the NICE DCV Server on Linux

This section explains how into install NICE DCV on a Linux server.

Topics

Prerequisites for Linux Servers

NICE DCV enables clients to access a remote graphical X session on a Linux server, which provides access to the corresponding Linux desktop. NICE DCV supports two different types of Linux desktop streaming: console sessions and virtual sessions.

With console sessions, NICE DCV captures the content of the X server running on the server. NICE DCV console sessions have direct access to the Linux server's GPU. Only one X server session can run at a time, and therefore, only one NICE DCV session can run at at time.

With virtual sessions, NICE DCV starts a special X server, called Xdcv, and runs a desktop environment inside the X server. If the dcv-gl package is installed and licensed, NICE DCV virtual sessions share access to the server's GPUs. Since there is a dedicated X server for each virtual session, several sessions can run at the same time.

This topic explains how to install the prerequisites required to use NICE DCV on a Linux server.

Contents

Install a Desktop Environment and Desktop Manager

To help improve your experience with NICE DCV on a Linux server, you can install a desktop environment and desktop manager.

A desktop environment is a graphical user interface (GUI) that helps you to interact with the Linux operating system. There are several desktop environments, and NICE DCV works with many of them.

A desktop manager is a program that manages the user login screen, and starts and stops the desktop environment sessions and the X server.

To install and configure the desktop environment and desktop manager

1. Install the desktop environment and the desktop manager packages.

   - RHEL 7.x

     Note
     The default desktop environment is Gnome3 and the default desktop manager is GDM.

     ```
     $ sudo yum groupinstall 'Server with GUI'
     ```

   - CentOS 7.x

     Note
     The default desktop environment is Gnome3 and the default desktop manager is GDM.

     ```
     $ yum groupinstall "GNOME Desktop"
     ```

   - RHEL 6.x and CentOS 6.x

     Note
     The default desktop environment is Gnome and the default desktop manager is GDM.

     ```
     $ sudo yum groupinstall\"X Window System" "Desktop" "General Purpose Desktop"
     ```

   - Amazon Linux 2

     Note
     The default desktop environment is Gnome3 and the default desktop manager is GDM.

     ```
     $ sudo yum install gdm gnome-session gnome-classic-session gnome-session-xsession
     ```

     ```
     $ sudo yum install xorg-x11-server-Xorg xorg-x11-fonts-Type1 xorg-x11-drivers
     ```

     ```
     $ sudo yum install gnome-terminal gnu-free-fonts-common gnu-free-mono-fonts gnu-free-
     sans-fonts gnu-free-serif-fonts
     ```

   - Ubuntu 18.x

     Note
     The default desktop environment is Gnome3 and the default desktop manager is GDM3. GDM3 is currently not supported with NICE DCV console sessions. We recommend that you use the LightDM desktop manager if you plan to work with NICE DCV console sessions.

     ```
     $ sudo apt update
     ```

     ```
     $ sudo apt install ubuntu-desktop
     ```

     Install LightDM.

     ```
     $ sudo apt install lightdm
     ```

   - SUSE Linux Enterprise 12.x

Note
The default desktop environment is SLE Classic and the default desktop manager is GDM.

```
$ sudo zypper install -t pattern gnome-basic
```

```
$ sudo sed -i "s/DISPLAYMANAGER=\"\"/DISPLAYMANAGER=\"gdm\"/" /etc/sysconfig/
displaymanager
```

```
$ sudo sed -i "s/DEFAULT_WM=\"\"/DEFAULT_WM=\"sle-classic\"/" /etc/sysconfig/
windowmanager
```

2. Update the software packages to ensure that the Linux server is up to date.

   - RHEL 6.x/7.x, CentOS 6.x/7.x, and Amazon Linux 2

     ```
     $ sudo yum upgrade
     ```

   - Ubuntu 18.x

     ```
     $ sudo apt upgrade
     ```

   - SUSE Linux Enterprise 12.x

     ```
     $ sudo zypper update
     ```

3. Reboot the Linux server.

   ```
   $ sudo reboot
   ```

Configure the X Server

If you intend to use a console session or GPU sharing, you must ensure that your Linux server has a properly configured and running X server.

Note
If you intend to use virtual sessions without GPU sharing, you do not need an X server.

The X server packages are typically installed as dependencies of the desktop environment and the desktop manager. We recommend that you configure the X server to start automatically when your Linux server boots.

To configure and start the X server

1. Configure the X server to start automatically when the Linux server boots.

   - RHEL 6.x and CentOS 6.x

     ```
     $ cat /etc/inittab  | grep "id.*initdefault"
     ```

     If the command returns `id:5:initdefault`, the X server is already configured to start automatically. Continue to the next step.

     If the command returns `id:3:initdefault`, the X server is not configured to start automatically. Run the following command.

```
$ sudo sed -i "s/id:3:initdefault:/id:5:initdefault:/" /etc/inittab
```

- RHEL 7.x, CentOS 7.x, Amazon Linux 2, Ubuntu 18.x, and SUSE Linux Enterprise 12.x

```
$ sudo systemctl get-default
```

If the command returns `graphical.target`, the X server is already configured to start automatically. Continue to the next step.

If the command returns `multi-user.target`, the X server is not configured to start automatically. Run the following command:

```
$ sudo systemctl set-default graphical.target
```

2. Start the X server.

- RHEL 6.x and CentOS 6.x

```
$ sudo init 5
```

- RHEL 7.x, CentOS 7.x, Amazon Linux 2, Ubuntu 18.x, and SUSE Linux Enterprise 12.x

```
$ sudo systemctl isolate graphical.target
```

3. Verify that the X server is running.

```
$ ps aux | grep X | grep -v grep
```

The following shows example output if the X server is running.

```
root      1891  0.0  0.7 277528 30448 tty7     Ssl+ 10:59   0:00 /usr/bin/Xorg :0 -
background none -verbose -auth /run/gdm/auth-for-gdm-wltseN/database -seat seat0 vt7
```

Install the glxinfo Utility

The glxinfo utility provides information about your Linux server's OpenGL configuration. It can be used to determine whether your Linux server is configured to support OpenGL hardware or software rendering, and it provides information about the drivers and supported extensions.

The glxinfo utility is installed as a package dependency of DCV GL. Therefore, if you installed DCV GL the glxinfo utility will already be installed on your Linux server.

To install the glxinfo utility

Run the following command:

- RHEL 6.x/7.x, CentOs 6.x/7.x, and Amazon Linux 2

```
$ sudo yum install glx-utils
```

- Ubuntu 18.x

```
$ sudo apt install mesa-utils
```

- SUSE Linux Enterprise 12.x

```
$ sudo zypper in Mesa-demo-x
```

Verify OpenGL Software Rendering

On non-GPU Linux servers, OpenGL is only supported in software rendering mode using the Mesa drivers. If you're using a non-GPU Linux server, and you intend to use OpenGL, ensure that the Mesa drivers are installed and properly configured on your Linux server.

Note
This applies to non-GPU Linux servers only.

To verify that OpenGL software rendering is available

Run the following command:

```
$ sudo DISPLAY=:0 glxinfo | grep -i "opengl.*version"
```

The following shows example output if OpenGL software rendering is available

```
OpenGL core profile version string: 3.3 (Core Profile) Mesa 17.0.5
OpenGL core profile shading language version string: 3.30
OpenGL version string: 3.0 Mesa 17.0.5
OpenGL shading language version string: 1.30
OpenGL ES profile version string: OpenGL ES 3.0 Mesa 17.0.5
OpenGL ES profile shading language version string: OpenGL ES GLSL ES 3.00
```

Install and Configure NVIDIA Drivers

With Linux servers that have a dedicated NVIDIA GPU, you must ensure that the appropriate NVIDIA drivers are installed and properly configured. For more information about installing the NVIDIA drivers on an Amazon EC2 Linux instance, see Installing the NVIDIA Driver on Linux Servers in the *Amazon EC2 User Guide for Linux Instances*.

Note
This applies to Linux servers with NVIDIA GPUs only.

After you have installed the NVIDIA drivers on your Linux server, you must update the `xorg.conf`.

To generate an updated xorg.conf

Run the following command:

```
nvidia-xconfig --preserve-busid
```

If your Linux server is configured with more than one NVIDIA GPU, include the `--enable-all-gpus` parameter.

Note
Make sure that your server does not have the legacy `/etc/X11/XF86Config` file. If it does, `nvidia-xconfig` updates that configuration file instead of generating the required `/etc/X11/xorg.conf` file. Run the following command to remove the legacy `XF86Config` file:

```
rm -rf /etc/X11/XF86Config*
```

You must also ensure that OpenGL hardware rendering is available on the Linux server.

To verify that OpenGL hardware rendering is available

Run the following command:

```
$ sudo DISPLAY=:0 glxinfo | grep -i "opengl.*version"
```

The following shows example output if OpenGL hardware rendering is available

```
OpenGL core profile version string: 4.4.0 NVIDIA 390.75
OpenGL core profile shading language version string: 4.40 NVIDIA via Cg compiler
OpenGL version string: 4.6.0 NVIDIA 390.75
OpenGL shading language version string: 4.60 NVIDIA
OpenGL ES profile version string: OpenGL ES 3.2 NVIDIA 390.75
OpenGL ES profile shading language version string: OpenGL ES GLSL ES 3.20
```

Install the NICE DCV Server

The NICE DCV server is installed using a series of RPM or .deb packages, depending on your host server's operating system. The packages install all required packages and their dependencies, and perform the necessary server configuration.

Note
You must be logged in as the root user to install the NICE DCV server.

To install the NICE DCV server on a Linux server

1. Launch and connect to the server on which to install the NICE DCV server.
2. The NICE DCV server packages are digitally signed with a secure GPG signature. To allow the package manager to verify the package signature, you must import the NICE GPG key. To do so, open a terminal window and import the NICE GPG key.

   - RHEL, CentOS, SUSE Linux Enterprise

     ```
     $ sudo rpm --import https://s3-eu-west-1.amazonaws.com/nice-dcv-publish/NICE-GPG-KEY
     ```

   - Ubuntu Linux 18.04

     ```
     $ wget https://s3-eu-west-1.amazonaws.com/nice-dcv-publish/NICE-GPG-KEY
     ```

     ```
     $ gpg --import NICE-GPG-KEY
     ```

3. Download the packages from the NICE website. The RPM and deb packages are packaged into a `.tgz` archive. Ensure that you download the correct archive for your operating system.
4. Extract the contents of the `.tgz` archive.

   - RHEL 6.x and CentOS 6.x

     ```
     $ tar -xvzf nice-dcv-2017.4-version-el6.tgz
     ```

   - RHEL 7.x and CentOS 7.x

     ```
     $ tar -xvzf nice-dcv-2017.4-version-el7.tgz
     ```

   - SUSE Linux Enterprise 12.x

     ```
     $ tar -xvzf nice-dcv-2017.4-version-sles12.tgz
     ```

   - Ubuntu 18.04

```
$ tar -xvzf nice-dcv-2017.4-version-ubuntu1804.tgz
```

5. Navigate into the extracted folder.

   - RHEL 6.x and CentOS 6.x

   ```
   $ cd nice-dcv-2017.4-version-el6
   ```

   - RHEL 7.x and CentOS 7.x

   ```
   $ cd nice-dcv-2017.4-version-el7
   ```

   - SUSE Linux Enterprise 12.x

   ```
   $ cd nice-dcv-2017.4-version-sles12
   ```

   - Ubuntu 18.04

   ```
   $ cd nice-dcv-2017.4-version-ubuntu1804
   ```

6. Install the NICE DCV server.

   - RHEL 6.x and CentOS 6.x

   ```
   $ sudo yum install nice-dcv-server-2017.4.version.el6.x86_64.rpm
   ```

   - RHEL 7.x and CentOS 7.x

   ```
   $ sudo yum install nice-dcv-server-2017.4.version.el7.x86_64.rpm
   ```

   - SUSE Linux Enterprise 12.x

   ```
   $ sudo zypper install nice-dcv-server-2017.4.version.sles12.x86_64.rpm
   ```

   - Ubuntu 18.04

   ```
   $ sudo apt install ./nice-dcv-server-2017.4.version-1_amd64.ubuntu1804.deb
   ```

7. (Optional) If you plan to use virtual sessions, install the `nice-xdcv` package.

   - RHEL 6.x and CentOS 6.x

   ```
   $ sudo yum install nice-xdcv-2017.4.version.el6.x86_64.rpm
   ```

   - RHEL 7.x and CentOS 7.x

   ```
   $ sudo yum install nice-xdcv-2017.4.version.el7.x86_64.rpm
   ```

   - SUSE Linux Enterprise 12.x

   ```
   $ sudo zypper install nice-xdcv-2017.4.version.sles12.x86_64.rpm
   ```

   - Ubuntu 18.04

   ```
   $ sudo apt install ./nice-xdcv-2017.4.version-1_amd64.ubuntu1804.deb
   ```

8. (Optional) If you plan to use GPU sharing, install the `nice-dcv-gl` package.

   - RHEL 6.x and CentOS 6.x

     ```
     $ sudo yum install nice-dcv-gl-2017.4.version.el6.x86_64.rpm
     ```

   - RHEL 7.x and CentOS 7.x

     ```
     $ sudo yum install nice-dcv-gl-2017.4.version.el7.x86_64.rpm
     ```

   - SUSE Linux Enterprise 12.x

     ```
     $ sudo zypper install nice-dcv-gl-2017.4.version.sles12.x86_64.rpm
     ```

   - Ubuntu 18.04

     ```
     $ sudo apt install ./nice-dcv-gl-2017.4.version-1_amd64.ubuntu1804.deb
     ```

     Note
     You can optionally install the `nice-dcv-gltest` package. This package includes a simple OpenGL application that can be used to determine whether your virtual sessions are properly configured to use hardware-based OpenGL.

9. (Optional) If you plan to use NICE DCV with NICE EnginFrame, install the `nice-dcv-simple-external-authenticator` package.

   - RHEL 6.x and CentOS 6.x

     ```
     $ sudo yum install nice-dcv-simple-external-
     authenticator-2017.4.version.el6.x86_64.rpm
     ```

   - RHEL 7.x and CentOS 7.x

     ```
     $ sudo yum install nice-dcv-simple-external-
     authenticator-2017.4.version.el7.x86_64.rpm
     ```

   - SUSE Linux Enterprise 12.x

     ```
     $ sudo zypper install nice-dcv-simple-external-
     authenticator-2017.4.version.sles12.x86_64.rpm
     ```

   - Ubuntu 18.04

     ```
     $ sudo apt install ./nice-dcv-simple-external-
     authenticator-2017.4.version-1_amd64.ubuntu1804.deb
     ```

10. (Optional) If you plan to support specialized USB devices using USB remotization, install the DCV USB drivers.

    To install the DCV USB drivers, you must have Dynamic Kernel Module Support (DKMS) installed on your server. Use the following commands to install DKMS.

    - RHEL and CentOS 6.x/7.x

      DKMS can be installed from the Extra Packages for Enterprise Linux (EPEL) repository. Run the following command to enable the EPEL repository:

```
$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-
latest-6.noarch.rpm
```

After you have enabled the EPEL repository, run the following command to install DKMS:

```
$ sudo yum install dkms
```

- SUSE Linux Enterprise 12.x

  Run the following command to install DKMS:

```
$ sudo zypper install http://download.opensuse.org/repositories/home:/Ximi1970:/
Dkms:/Staging/SLE_12_SP4/noarch/dkms-2.5-11.1.noarch.rpm
```

- Ubuntu 18.04

  DKMS is available in the official Ubuntu repository. Run the following command to install DKMS:

```
$ sudo apt install dkms
```

After you have installed DKMS, run the following command to install the DCV USB drivers:

```
$ sudo dcvusbdriverinstaller
```

Post-Installation Checks

This topic provides some post-installation checks that you should perform after installing NICE DCV to ensure that your NICE DCV server is properly configured.

Contents

Ensure the NICE DCV Server is Reachable

By default, the NICE DCV server is configured to communicate over port 8443. Ensure that the server is reachable over this port. If you have a firewall that prevents access over port 8443, you must change the port over which the NICE DCV server communicates. For more information, see Changing the NICE DCV Server TCP Port (p. 28).

Ensure that the X Server is Accessible

You must ensure that NICE DCV console and virtual sessions can access the X server.

Console Sessions

When the NICE DCV server is installed, a `dcv` user is created. You must ensure that this user can access the X server.

To verify that the `dcv` user can access the X server

Run the following command:

```
$ sudo DISPLAY=:0 XAUTHORITY=$(ps aux | grep "X.*\-auth" | grep -v grep | awk -F"-auth "
 '{print $2}' | awk '{print $1}') xhost | grep "SI:localuser:dcv$"
```

If the command returns `SI:localuser:dcv`, the dcv user can access the X server.

If the command does not return `SI:localuser:dcv`, the dcv user does not have access to the X server. Run the following commands to restart the X server:

- RHEL 7.x, CentOs 7.x, Amazon Linux 2, Ubuntu 18.x, and SUSE Linux Enterprise 12.x

```
$ sudo systemctl isolate multi-user.target
```

```
$ sudo systemctl isolate graphical.target
```

- RHEL 6.x and CentOs 6.x

```
$ sudo init 3
```

```
$ sudo init 5
```

Virtual Sessions

If you installed the DCV GL package, you must ensure that local users can access the X server. This ensures that OpenGL hardware acceleration works correctly with virtual sessions.

To verify that local users can access the X server

Run the following command:

```
$ sudo DISPLAY=:0 XAUTHORITY=$(ps aux | grep "X.*\-auth" | grep -v grep | awk -F"-auth "
 '{print $2}' | awk '{print $1}') xhost | grep "LOCAL:$"
```

If the command returns `LOCAL:`, local users can access the X server.

If the command does not return `LOCAL:`, local users do not have access to the X server. Run the following commands to restart the X server, and to disable and re-enable DCV GL:

- RHEL 7.x, CentOs 7.x, Amazon Linux 2, Ubuntu 18.x, and SUSE Linux Enterprise 12.x

```
$ sudo systemctl isolate multi-user.target
```

```
$ sudo dcvgladmin disable
```

```
$ sudo dcvgladmin enable
```

```
$ sudo systemctl isolate graphical.target
```

- RHEL 6.x and CentOs 6.x

```
$ sudo init 3
```

```
$ sudo dcvgladmin disable
```

```
$ sudo dcvgladmin enable
```

```
$ sudo init 5
```

Verify that DCV GL is Properly Installed

The **dcvgldiag** utility is automatically installed when you install the DCV GL package. You can use this utility to check that the Linux server configuration meets the DCV GL requirements.

To run the dcvgldiag utility

Use the following command:

```
$ sudo dcvgldiag
```

The utility returns a list of warnings and errors, along with the possible solutions.

Licensing NICE DCV

The NICE DCV licensing requirements differ depending on where you are installing and using the NICE DCV server.

NICE DCV on Amazon EC2

You do not need a license server to install and use the NICE DCV server on an EC2 instance. The NICE DCV server automatically detects that it is running on an Amazon EC2 instance and periodically connects to an Amazon S3 bucket to determine whether a valid license is available.

Make sure that your instance:

- Can reach the Amazon S3 endpoint. If it has access to the internet, it connects using the Amazon S3 public endpoint. If your instance does not have access to the internet, configure a gateway endpoint for your VPC with an outbound security group rule or Access Control List (ACL) policy that allows you to reach Amazon S3 via HTTPS. For more information, see Gateway VPC Endpoints in the *Amazon VPC User Guide*. If you experience any issues connecting to the Amazon S3 bucket, see Why can't I connect to an S3 bucket using a gateway VPC endpoint? in the *AWS Knowledge Center*.
- Has permission to access the required Amazon S3 object. Add the following Amazon S3 access policy to the instance's IAM role and replace the *region* placeholder with your region. For more information, see Create IAM Role.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::dcv-license.region/*"
        }
    ]
}
```

If you are installing and using the NICE DCV server on an Amazon EC2 instance, you can skip the rest of this chapter. The rest of this chapter only applies to using the NICE DCV server on an on-premises or alternative cloud-based server.

NICE DCV on On-premises and Other Cloud-based Servers

A license is required to install and use the NICE DCV server on an on-premises or alternative cloud-based server. The following licensing options are available:

- **Automatic demo license**— Automatically installed when you install the NICE DCV server. These licenses are valid for a period of 15 days from the date of installation. After the license expires, you are no longer able to create and host NICE DCV sessions on the server. These licenses are ideal for short-term testing. To test for a longer period, request an extended demo license.

  Note
  The NICE DCV server defaults to the automatic demo license if no other license has been configured.
- **Extended demo license**—An extended demo license is a demo license that is valid for a period longer than 15 days. The period is determined by NICE on a case-by-case basis. Extended demo licenses become invalid when they reach their expiration date, and you are no longer able to create and host NICE DCV sessions on the server. Extended demo licenses must be requested from NICE. They come as a license file that must be installed on the NICE DCV server.
- **Floating license**—A floating license is a full license that you purchase from NICE. Floating licenses do not have an expiration date and can be used for an indefinite period. However, they do limit the number of sessions that you can host on the NICE DCV server concurrently. These licenses come as a license file to be installed on a Reprise License Manager (RLM) server.

For more information about licenses, see the NICE website.

Note
NICE DCV clients do not require a license.

Topics

Installing an Extended Demo License

When you request an extended demo license from NICE, you receive a `license.lic` file that defines the license.

To install the extended demo license

Place the `license.lic` file in the following folder on your server:

- Windows server

```
C:\Program Files\NICE\DCV\Server\license\license.lic
```

- Linux server

```
/usr/share/dcv/license/license.lic
```

Alternatively, to place the `license.lic` in a different folder on the server, you must update the `license-file` configuration parameter so that it specifies the license file's full path.

To update the `license-file` configuration parameter on a Windows server

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/license/** key and select the **license-file** parameter.

   If there is no `license-file` parameter in the registry key, create one:

   a. Open the context (right-click) menu for the **license** key in the left-hand panel and choose **New**, **String Value**.
   b. For **Name**, type `license-file` and press **Enter**.
3. Open the **license-file** parameter. For **Value data**, type the full path to the `license.lic` file.
4. Choose **OK** and close the Windows Registry Editor.

To update the `license-file` configuration parameter on a Linux server

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `license-file` parameter in the `[license]` section, and replace the existing path with the new full path to the `license.lic` file.

   If there is no `license-file` parameter in the `[license]` section, add it manually using the following format:

   ```
   license-file = "/custom-path/license.lic"
   ```
3. Save and close the file.

Installing a Floating License

When you purchase a floating license from NICE, you receive a `license.lic` file that defines the license. To install the license, you must:

1. Modify the license file.
2. Prepare the Reprise License Manager (RLM) server.
3. Configure the NICE DCV server.

Contents

Step 1: Modify the License File

The `license.lic` file that you receive from NICE specifies the following information:

- The RLM server's hostname, `rlmhostid` identifier, and TCP port number
- The ISV port number. This is the port on which the RLM server listens for DCV licenses.
- The NICE DCV products covered by the license, along with the following details for each product:

- The major version covered by the license. For example, `2017` for the 2017 NICE DCV products.
- The expiration date. `Permanent` indicates that the license does not expire.
- The maximum number of concurrent sessions. For example, `10` for 10 concurrent sessions on the server.
- The license checksum.
- The license signature.

The following code block shows the format of the `license.lic` file:

```
HOST RLM_server_hostname RLM_server_identifier RLM_server_port
ISV nice port=port_number
LICENSE product_1 major_version expiration_date concurrent_sessions share=hi _ck=checksum
 sig="signature"
LICENSE product_2 major_version expiration_date concurrent_sessions share=hi _ck=checksum
 sig="signature"
```

Note
The ISV port is optional.

The following code block shows an example of a `license.lic` file with the ISV port omitted. The license file includes licenses for two NICE products - DCV and dcv-gl.

```
HOST My-RLM-server abcdef123456 5053
ISV nice
LICENSE nice dcv 2017 permanent 10 share=hi _ck=456789098a
 sig="abcdefghijklmnopqrstuvwxyz1234567890abcdefghijklmnopqrstuvwxyz1234567890ab"
LICENSE nice dcv-gl 2017 permanent 10 share=hi _ck=123454323x
 sig="1234567890abcdefghijklmnopqrstuvwxyz1234567890abcdefghijklmnopqrstuvwxyz12"
```

To modify the `license.lic` file received from NICE

1. Open the file with your preferred text editor.
2. Add your RLM server's hostname and the TCP port number to the first line in the file, which starts with `HOST`.

   Warning
   The *`RLM_server_identifier`* corresponds to the `rlmhostid` identifier that was used to generate the license. The *`RLM_server_identifier`* cannot be modified.
3. (Optional) Complete the ISV port number in second line in the file, which starts with `ISV`, by adding `port=`*`port_number`*.

   If you do not want to specify an ISV port, omit `port=`*`port_number`*. If you do not specify a port, a random port is used. Using a random port could cause conflicts with your firewall configuration.
4. Save and close the file.

Warning
Modifying any other part of the license corrupts the file's signature and invalidates the license.

Step 2: Prepare the RLM Server

The license file must be installed on an RLM server. Any NICE DCV server that can access the RLM server can use the license.

If you put the RLM server behind a firewall, be aware that the RLM license server listens on two separate TCP ports. One is the main port, is specified in the license file `HOST` line and by default is `5053`. The

other is related to the ISV license, is specifed in the license file `ISV` line and if not specified is randomly assigned: if you want the server to use a fixed port (e.g. for easier firewall configuration) you need to define the ISV port in the license file. See the section called "Step 1: Modify the License File" (p. 19) for more details.

For more information about RLM, see the Reprise Software website.

To prepare the RLM server on Windows

1. On your RLM server, download the RLM License Administration Bundle from the Reprise Software website.
2. Extract the contents of the RLM License Administration Bundle to `C:\RLM`.
3. Copy the `license.lic` file that you received from NICE to `C:\RLM\license\`.
4. Copy the `nice.set` file from your NICE DCV server and place it in the `C:\RLM\` folder on your RLM server.

   The `nice.set` file can be found in the following location on your NICE DCV server:

   - Windows NICE DCV server:

     ```
     C:\Program Files\NICE\DCV\Server\license\
     ```

   - Linux NICE DCV server:

     ```
     /usr/share/dcv/license/
     ```

5. On your RLM server, open a command prompt window and do the following:

   a. Create an `RLM` root folder:

      ```
      C:\> cd C:\RLM
      ```

   b. Install the RLM server as a Windows service. For more information about the RLM startup options, see the RLM License Administration Manual.

      ```
      C:\> rlm.exe -nows -dlog C:\RLM\rlm.log -c C:\RLM\license -install_service -
      service_name dcv-rlm
      ```

6. Start the RLM server:

   ```
   C:\> net start dcv-rlm
   ```

7. Confirm that the RLM server is running and functioning as expected.

   a. Open the `rlm.log` file located in `C:\RLM\` with your preferred text editor and confirm that the following line appears:

      Note
      The contents of the `rlm.log` file might vary slightly depending on the RLM server version.

      ```
      date_time (nice) Server started on license1 (hostid: host_id) for: dcv dcv-gl
      ```

   b. Run the following command:

      ```
      C:\RLM\rlmstat -a -c rlm_server_hostname@5053
      ```

To prepare the RLM server on Linux

1. Log into your RLM server as `root` and download the RLM License Administration Bundle from the Reprise Software website.
2. Create a user group and a new `rlm` user.

   Note
   This can be any valid user or service account. We strongly recommend that this value not be the root account.

   ```
   $ groupadd -r rlm
   ```

   ```
   $ useradd -r -g rlm -d "/opt/nice/rlm" -s /sbin/nologin -c "RLM License Server" rlm
   ```
3. Create the `/opt/nice/rlm` and `/opt/nice/rlm/license` folders needed for the RLM server:

   ```
   $ mkdir -p /opt/nice/rlm/license
   ```
4. Extract the contents of the RLM License Administration Bundle to `/opt/nice/rlm/`, and ensure that the files are owned by the `rlm` user:

   ```
   $ tar xvf x64_l1.admin.tar.gz -C /opt/nice/rlm/ -stripcomponents 1
   ```

   ```
   $ chown -R rlm:rlm /opt/nice/rlm
   ```
5. Copy the `license.lic` file that you received from NICE to `/opt/nice/rlm/license/`.
6. Copy the `nice.set` file from your NICE DCV server and place it in `/opt/nice/rlm` on your RLM server.

   The `nice.set` file can be found in the following location on your NICE DCV server:

   - Windows NICE DCV server:

     ```
     C:\Program Files\NICE\DCV\Server\license\
     ```
   - Linux NICE DCV server:

     ```
     /usr/share/dcv/license/
     ```
7. Start the RLM server:

   ```
   $ service dcv-rlm start
   ```
8. Verify that the RLM server is running and functioning as expected. Open `var/log/rlm.log` with your preferred text editor and confirm that the following line appears:

   Note
   The contents of the `rlm.log` might vary slightly depending on the RLM server version.

   ```
   date_time (nice) Server started on license1 (hostid: host_id) for: dcv dcv-gl
   ```
9. Ensure that the RLM server starts automatically.

   a. Create a file named `dcv-rlm` in the `/opt/nice/rlm/` folder:

```
$ touch /opt/nice/rlm/dcv-rlm
```

b. Open the file using your preferred text editor and add the following script. Save and close the file.

```
#! /bin/sh
# chkconfig: 35 99 01
# description: The Reprise License Manager daemon.
# processname: dcv-rlm

### BEGIN INIT INFO
# Provides: dcv-rlm
# Required-Start: $local_fs $remote_fs $syslog
# Required-Stop: $local_fs $remote_fs $syslog
# Default-Start: 3 4 5
# Default-Stop: 0 1 2 6
# Short-Description: The Reprise License Manager daemon.
# Description: A service that runs the Reprise License Manager daemon.
### END INIT INFO

# user used to run the daemon
RLM_USER="rlm"

# root of rlm installation
RLM_ROOT="/opt/nice/rlm"

# license directory (license files should have .lic extension)
RLM_LICENSE_DIR="/opt/nice/rlm/license"

# log file
RLM_LOG_FILE="/var/log/rlm.log"

_getpid() {
 pidof -o $$ -o $PPID -o %PPID -x "$1"
}

start() {
 echo -n "Starting rlm: "
 touch ${RLM_LOG_FILE}
 chown "${RLM_USER}" ${RLM_LOG_FILE}
 su -p -s /bin/sh "${RLM_USER}" -c "${RLM_ROOT}/rlm -c ${RLM_LICENSE_DIR} \
     -nows -dlog +${RLM_LOG_FILE} &"
 if [ $? -ne 0 ]; then
  echo "FAILED"
  return 1
 fi
 echo "OK"
}

stop() {
 echo -n "Stopping rlm: "
 pid=`_getpid ${RLM_ROOT}/rlm`
 if [ -n "$pid" ]; then
  kill $pid >/dev/null 2>&1
  sleep 3
  if [ -d "/proc/$pid" ] ; then
   echo "FAILED"
   return 1
  fi
 fi
 echo "OK"
}
```

```
status() {
 pid=`_getpid ${RLM_ROOT}/rlm`
 if [ -z "$pid" ]; then
  echo "rlm is stopped"
  return 3
 fi
 echo "rlm (pid $pid) is running..."
 return 0
}

restart() {
 stop
 start
}

case "$1" in
 start)
  start
  ;;
 stop)
  stop
  ;;
 status)
  status
  ;;
 restart)
  restart
  ;;
 *)
 echo $"Usage: $0 {start|stop|status|restart}"
 exit 1
esac

exit $?

# ex:ts=4:et:
```

c. Make the script executable, copy it to `/etc/init.d/`, and then add it to the `chkconfig` utility:

```
chmod +x /opt/nice/rlm/dcv-rlm
```

```
cp -a /opt/nice/rlm/dcv-rlm /etc/init.d/
```

```
chkconfig --add dcv-rlm
```

Step 3: Configure the NICE DCV Server

Configure your NICE DCV server to use your RLM server. To do this, you must configure the `license-file` configuration parameter on your NICE DCV server.

To configure the `license-file` configuration parameter on a Windows server

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/license/** key and select the **license-file** parameter.

   If there is no `license-file` parameter in the registry key, you must create one:

a. Open the context (right-click) menu for the **license** key in the left-hand panel and choose **New**, **String Value**.
b. For **Name**, type `license-file` and press **Enter**.

3. Open the **license-file** parameter. For **Value data**, type RLM server port and hostname in the `5053@`*`RLM_server_hostname`* format.

   Note
   You can use the RLM server IP address instead of its hostname.

4. Choose **OK** and close the Windows Registry Editor.

To configure the `license-file` configuration parameter on a Linux server

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `license-file` parameter in the `[license]` section, and replace the existing path with the RLM server's port and hostname in the `5053@`*`RLM_server_hostname`* format.

   If there is no `license-file` parameter in the `[license]` section, add it manually using the following format:

   ```
   license-file = "5053@RLM_server_hostname"
   ```

   Note
   You can use the RLM server IP address instead of its hostname.

3. Save and close the file.

Managing the NICE DCV Server

The NICE DCV server runs as an operating system service. You must be logged in as the administrator (Windows) or root (Linux) to start, stop, or configure the NICE DCV server.

Topics

Starting the NICE DCV Server

The NICE DCV server must be running in order to host sessions.

By default, the NICE DCV server is configured to start automatically when the server it is hosted on starts up. However, if you chose to disable automatic startup when you installed the NICE DCV server, you must start the server manually using the following procedures.

Contents

Starting the NICE DCV Server on Windows

Use the following procedure to manually start the NICE DCV server using the Microsoft Management Console.

To start the NICE DCV server on Windows

1. Open the Microsoft Management Console.
2. In the right-hand pane, open **DCV Server**.
3. Choose **Start**.

> **Note**
> If the server is already running, the **Start** button is disabled.

Use the following procedure to configure the NICE DCV server to start automatically using the Microsoft Management Console.

To configure the NICE DCV server to start automatically on Windows

1. Open the Microsoft Management Console.
2. In the right-hand pane, open **DCV Server**.
3. For **Startup service**, choose **Automatic**.

Starting the NICE DCV Server on Linux

Use the following procedure to manually start the NICE DCV server using the command line.

To start the NICE DCV server on Linux

Use the following commands:

- RHEL 6.x and CentOS 6.x

```
$ sudo service dcvserver start
```

- RHEL 7.x, CentOS 7.x, SUSE Linux Enterprise 12, and Ubuntu 18.x

```
$ sudo systemctl start dcvserver
```

Use the following procedure to configure the NICE DCV server to start automatically using the command line.

To configure the NICE DCV server to start automatically on Linux

Use the following commands:

- RHEL 6.x and CentOS 6.x

```
$ sudo chkconfig —add dcvserver
```

- RHEL 7.x, CentOS 7.x, SUSE Linux Enterprise 12, and Ubuntu 18.x

```
$ sudo systemctl enable dcvserver
```

Stopping the NICE DCV Server

You can stop the NICE DCV server at any time. Stopping the server terminates all active NICE DCV sessions. You can't start new sessions until the server is restarted.

Contents

Stopping the NICE DCV Server on Windows

Use the following procedure to stop the NICE DCV server using the Microsoft Management Console.

To stop the NICE DCV server on Windows

1. Open the Microsoft Management Console.
2. In the right-hand pane, open **DCV Server**.
3. Choose **Stop**.

Note
If the server is already stopped, the **Stop** button is disabled.

Use the following procedure to disable automatic NICE DCV server startup using the Microsoft Management Console.

To prevent the NICE DCV server from starting automatically on Windows

1. Open the Microsoft Management Console.
2. In the right-hand pane, open **DCV Server**.
3. For **Startup service**, choose **Manual**.

Stopping the NICE DCV Server on Linux

Use the following procedure to stop the NICE DCV server using the command line.

To stop the NICE DCV server on Linux

Use the following commands:

- RHEL 6.x and CentOS 6.x

```
$ sudo service dcvserver stop
```

- RHEL 7.x, CentOS 7.x, and SUSE Linux Enterprise 12

```
$ sudo systemctl stop dcvserver
```

Use the following procedure to disable automatic NICE DCV server startup using the command line.

To prevent the NICE DCV server from starting automatically on Linux

Use the following commands:

- RHEL 6.x and CentOS 6.x

```
$ sudo chkconfig —del dcvserver
```

- RHEL 7.x, CentOS 7.x, and SUSE Linux Enterprise 12

```
$ sudo systemctl disable dcvserver
```

Changing the NICE DCV Server TCP Port

By default, the NICE DCV server is configured to communicate over port `8443`. You can specify a custom TCP port after you have installed the NICE DCV server. The port must be greater than 1024.

To allow NICE DCV clients to access your NICE DCV server over the standard HTTPS port (443), we recommend that you use a web proxy or load balancer as a frontend gateway to redirect client connections to the server.

Ensure that you communicate any port changes to your clients, as they need the port number to connect to sessions.

Changing the NICE DCV Server TCP Port on Windows

To change the port used by the NICE DCV server, you must configure the `web-port` parameter using the Windows Registry Editor.

To change the server's TCP port on Windows

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/connectivity/** key and select the **web-port** parameter.

   If there is no `web-port` parameter in the registry key, create one:

   a. In the left-hand pane, open the context (right-click) menu for the **connectivity** key and choose **New**, **DWORD (32-bit) value**.

   b. For **Name**, type `web-port` and press **Enter**.
3. Open the **web-port** parameter. For **Value data**, type the new TCP port number.

   Note
   The TCP port number must be greater than 1024.
4. Choose **OK** and close the Windows Registry Editor.
5. Stop (p. 27) and restart (p. 26) the NICE DCV server.

Changing the NICE DCV Server TCP Port on Linux

To change the port used by the NICE DCV server, you must configure the `web-port` parameter in the `dcv.conf` file.

To change the server's TCP port on Linux

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `web-port` parameter in the `[connectivity]` section, and replace the existing TCP port number with the new TCP port number.

   If there is no `web-port` parameter in the `[connectivity]` section, add it manually using the following format:

   ```
   [connectivity]
   web-port=port_number
   ```

   Note
   The TCP port number must be greater than 1024.
3. Save and close the file.
4. Stop (p. 27) and restart (p. 26) the NICE DCV server.

Disconnecting Idle Clients

NICE DCV can be configured to disconnect idle clients who have not sent any keyboard or pointer input to the NICE DCV server for a specified period. By default, the NICE DCV server disconnects NICE DCV clients that have been idle for a period of 60 minutes.

You can use the following procedures to specify a custom idle timeout period.

Changing the Idle Timeout Period on Windows

To change the NICE DCV server's idle timeout period, you must configure the `idle-timeout` parameter using the Windows Registry Editor.

To change the idle timeout period on Windows

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/connectivity/** key and select the **idle-timeout** parameter.

   If there is no `idle-timeout` parameter in the registry key, create one:

   a. In the left-hand pane, open the context (right-click) menu for the **connectivity** key and choose **New**, **DWORD (32-bit) value**.

   b. For **Name**, type `idle-timeout` and press **Enter**.
3. Open the **idle-timeout** parameter. For **Value data**, enter a value for the idle timeout period (in minutes).

   Note
   To avoid disconnecting idle clients, enter `0` for the `idle-timeout` parameter.
4. Choose **OK** and close the Windows Registry Editor.

Changing the Idle Timeout Period on Linux

To change the NICE DCV server's idle timeout period, you must configure the `idle-timeout` parameter in the `dcv.conf` file.

To change the idle timeout period on Linux

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `idle-timeout` parameter in the `[connectivity]` section, and replace the existing timeout period with the new timeout period (in minutes).

   If there is no `idle-timeout` parameter in the `[connectivity]` section, add it manually using the following format:

   ```
   [connectivity]
   idle-timeout=timeout_in_minutes
   ```

   Note
   To avoid disconnecting idle clients, enter `0` for the `idle-timeout` parameter.
3. Save and close the file.

Enabling GPU Sharing on a Linux NICE DCV Server

GPU sharing enables you to share one or more physical GPUs between multiple NICE DCV virtual sessions. For more information about sessions, see Managing NICE DCV Sessions (p. 41). Using GPU sharing enables you to use a single NICE DCV server and host multiple virtual sessions that share the server's physical GPU resources.

Note
GPU sharing is only supported on Linux NICE DCV servers.

Prerequisites

Before you begin, complete the following prerequisites:

- Install the NICE DCV server on a Linux server.
- Install the NICE DCV `dcv-gl` and `nice-xdcv` packages on the server.
- Ensure that the server has at least one supported NVIDIA GPU.
- Install the NVIDIA GPU driver on the server. Official NVIDIA drivers are required. The open-source NVIDIA drivers are not supported.
- Ensure that the NVIDIA GPU driver supports hardware-accelerated OpenGL.
- Install an X Server, and configure the `Device` and `Screen` sections in the `xorg.conf` file.

  Note
  You can use the `nvidia-xconfig` NVIDIA utility to automatically create an `xorg.conf` file and configure it for all the available NVIDIA GPUs.
- Ensure that the X Server is running.
- (Optional) Verify the NICE DCV server configuration by running the `dcvgldiag` tool. For more information, see Post-Installation Checks (p. 15).

  You can also install the `nice-dcv-gltest` package and run the `dcvgltest` test application to check whether the server is properly configured for GPU sharing.

To enable GPU sharing, you must specify the list of GPUs to be used by the virtual sessions. If you do not specify the GPUs, only the GPU used by the standard X Server, with the display name `:0.0`, is used.

You must specify the GPUs in the `gl-displays` parameter in the `dcv.conf` file after you complete the prerequisites described earlier in this topic.

To enable GPU sharing on a Linux NICE DCV server

1. Navigate to `/etc/dcv/` and open the `dcv.conf` file with your preferred text editor.
2. Add the `[display/linux]` section and the `gl-displays` parameter, then specify the available GPUs in the following format:

   ```
   [display/linux]
   gl-displays = [':xserver_port.screen_number_1',':xserver_port.screen_number_2', ...]
   ```

   Where *`xserver_port`* is the server and *`screen_number`* is the number associated with the screen that is associated with the GPU. *`screen_number`* starts from `0`.

   The following example shows the `gl-displays` parameter for two GPUs running on the default X Server session:

   ```
   [display/linux]
   gl-displays = [':0.0',':0.1']
   ```

3. Save and close the file.
4. Stop (p. 27) and restart (p. 26) the NICE DCV server.

Changing the TLS Certificate

NICE DCV automatically generates a self-signed certificate that is used to secure traffic between the NICE DCV client and NICE DCV server. This certificate is used by default if no other certificate is installed on your NICE DCV server. The default certificate includes two files, the certificate itself (`dcv.pem`), and a key (`dcv.key`).

You can replace the default NICE DCV certificate and its key with your own certificate and key.

To change the NICE DCV server's TLS certificate

- Windows NICE DCV server

  Place the certificate and its key in the following location on your Windows NICE DCV server:

```
C:\Windows\System32\config\systemprofile\AppData\Local\NICE\dcv\
```

- Linux NICE DCV server

  Place the certificate and its key in the following location on your Linux NICE DCV server:

```
/etc/dcv/
```

  Grant ownership of both files to the `dcv` user, and change their permissions to 600 (only the owner can read or write to them).

```
$  sudo chown dcv certificate_filename.pem key_filename.key
```

```
$  sudo chmod 600 certificate_filename.pem key_filename.key
```

Enabling USB Remotization

NICE DCV enables clients to use specialized USB devices, such as 3D pointing devices or graphic tablets. The devices are physically connected to their computer to interact with an application running on a NICE DCV server.

> **Note**
> USB remotization is only supported with the Windows client. It is not supported with the portable Windows client or the web browser client. Additional configuration might be required on the NICE DCV client. For more information, see Using USB Remotization in the *NICE DCV User Guide*.

The NICE DCV server uses a *whitelist* to determine which USB devices clients are allowed to use. Some commonly used USB devices are added to the whitelist by default. This enables clients to connect these USB devices to their computer and use them on the server without any additional configuration.

However, some specialized devices might not be whitelisted by default. These devices must be manually added to the whitelist on the NICE DCV server before they can be used by the client. After they have been added to the whitelist, they appear in the Windows client **Settings** menu.

Whitelisting USB devices on a Windows NICE DCV Server

To add a USB device to the whitelist, you must obtain the USB device's filter string from the client and add it to the `usb-devices.conf` file.

To add a USB device to the whitelist on a Windows NICE DCV server

1. Ensure that you have installed the latest version of the NICE DCV server, and that you opted to install the USB remotization drivers. For more information, see Installing the NICE DCV Server on Windows (p. 5).
2. Install the USB device's hardware drivers on the NICE DCV server.
3. Request the filter string from the client. For more information, see Using USB Remotization in the *NICE DCV User Guide*.
4. Open `C:\Program Files\NICE\DCV\Server\conf\usb-devices.conf` using your preferred text editor and add the filter string to a new line at the bottom of the file.
5. Save and close the file.
6. Stop and restart the NICE DCV server.

Whitelisting USB devices on a Linux NICE DCV Server

To add a USB device to the whitelist, you must obtain the USB device's filter string from the client and add it to the `usb-devices.conf` file.

Adding USB devices to the whitelist on a Linux NICE DCV server

1. Ensure that you have installed the latest version of the NICE DCV server and the DCV USB driver. For more information, see Installing the NICE DCV Server on Linux (p. 7).
2. Install the USB device's hardware drivers on the NICE DCV server.
3. Request the filter string from the client. For more information, see Using USB Remotization in the *NICE DCV User Guide*.
4. Open `/etc/dcv/usb-devices.conf` using your preferred text editor and add the filter string to a new line at the bottom of the file.
5. Save and close the file.
6. Stop and restart the NICE DCV server.

Configuring Smart Card Caching

The smart card caching feature enables the NICE DCV server to cache smart card values. When this feature is enabled, the NICE DCV server caches the results of recent calls to the client's smart card. Future calls are retrieved directly from the server's cache, instead of from the client. This helps to reduce the amount of traffic that is transferred between the client and the server, and improves performance. It is especially useful if the client has a slow internet connection.

> **Note**
> Smart card functionality is only supported with Linux NICE DCV servers.

Smart card caching is disabled by default. Clients can manually enable smart card caching for each application they run by setting the `DCV_PCSC_ENABLE_CACHE` environment variable. For more information, see Using a Smart Card in the *NICE DCV User Guide*. Alternatively, you can configure the

NICE DCV server to permanently enable or disable smart card caching, regardless of the value specified for the `DCV_PCSC_ENABLE_CACHE` environment variable.

To permanently enable or disable smart card caching

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `enable-cache` parameter in the `[smartcard]` section. To permanently enable smart card caching, enter `'always-on'` . To permanently disable smart card caching, enter `'always-off'`.

   If there is no `enable-cache` parameter in the `[smartcard]` section, add it manually using the following format:

   ```
   [smartcard]
   enable-cache='always-on'|'always-off'
   ```
3. Save and close the file.
4. Stop (p. 27) and restart (p. 26) the NICE DCV server.

Enabling Session Storage

Session storage is a folder on the NICE DCV server that clients can access when they are connected to a specific NICE DCV session. When you enable session storage for a session, clients can download files from, and upload files to the specified folder. This feature enables clients to share files while connected to a session.

Enabling Session Storage on a Windows NICE DCV Server

To enable session storage, you must create the folder to use for session storage and then configure the `storage-root` parameter using the Windows Registry Editor.

To enable session storage on Windows

1. Create the folder to use for session storage. For example, `c:\session-storage`.
2. Configure the `storage-root` parameter.

   a. Open the Windows Registry Editor.

   b. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/filestorage/** key and select the **storage-root** parameter.

      If there is no `storage-root` parameter in the registry key, create one:

      i. In the left-hand pane, open the context (right-click) menu for the **file-storage** key and choose **New**, **String**.

      ii. For **Name**, type `storage-root` and press **Enter**.

   c. Open the **storage-root** parameter. For **Value data**, type the full path to the folder created in Step 1.

   d. Choose **OK** and close the Windows Registry Editor.

   e. Stop (p. 27) and restart (p. 26) the NICE DCV server.
3. Start the session and specify the `--storage-root` option. For more information, see Starting NICE DCV Sessions (p. 42).

Enabling Session Storage on Linux

To enable session storage, you must create the folder to use for session storage and then configure the `storage-root` parameter in the `dcv.conf` file.

To enable session storage on Linux

1. Create the folder to use for session storage. For example, `/opt/session-storage/`.
2. Configure the `storage-root` parameter.
   a. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
   b. Locate the `storage-root` parameter in the `[filestorage]` section, and replace the existing path with the full path to the folder created in Step 1.

      If there is no `storage-root` parameter in the `[filestorage]` section, add it manually using the following format:

      ```
      [filestorage]
      storage-root="/opt/session-storage/"
      ```
3. Save and close the file.
4. Stop (p. 27) and restart (p. 26) the NICE DCV server.
5. Start the session and specify the `--storage-root` option. For more information, see Starting NICE DCV Sessions (p. 42).

Configuring NICE DCV Authentication

By default, clients are required to authenticate against the server on which NICE DCV is hosted before connecting to a NICE DCV session. If the client fails to authenticate, they are prevented from connecting to the session. Client authentication requirements can be disabled to allow clients to connect to a session without authenticating against the server.

NICE DCV supports the following authentication methods:

- `system` — This is the default authentication method. Client authentication is delegated to the underlying operating system. For Windows NICE DCV servers, authentication is delegated to WinLogon. For Linux NICE DCV servers, authentication is delegated to PAM. Clients provide their system credentials when connecting to a NICE DCV session. Ensure that your clients have the credentials for the appropriate user accounts on the NICE DCV server.
- `none` — No client authentication is required when connecting to a NICE DCV session. The NICE DCV server automatically grants access to all clients attempting to connect to a session.

Make sure that your clients are aware of the authentication method used by the NICE DCV server, and that they have the information required to connect to the session.

Configuring Authentication on Windows

To change the NICE DCV server's authentication method, you must configure the `authentication` parameter using the Windows Registry Editor.

To change the authentication method on Windows

1. Open the Windows Registry Editor.

2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/security/** key and select the **authentication** parameter.

   If there is no `authentication` parameter in the registry key, create one:

   a. In the left-hand pane, open the context (right-click) menu for the **authentication** key and choose **New**, **string value**.
   b. For **Name**, type `authentication` and press **Enter**.
3. Open the **authentication** parameter. For **Value data**, enter either `system` or `none`.
4. Choose **OK** and close the Windows Registry Editor.

Windows Credentials Provider

Windows Credentials Provider enables users to bypass the Windows login if they successfully authenticate against the DCV server.

Windows Credentials Provider is only supported if the DCV `authentication` parameter is set to `system`. If the DCV `authentication` parameter is set to `none`, users are required to manually log into Windows after they have been automatically authenticated against the DCV server.

Windows Credentials Provider is enabled by default when you install the NICE DCV server.

To disable Windows Credentials Provider

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\Credential Providers\{8A2C93D0-D55F-4045-99D7-B27F5E263407}** key.
3. Choose **Edit**, **New**, **DWORD Value**.
4. For the name, type **Disabled**.
5. Open the value, for **Value data**, type `1` and choose **OK**.
6. Close the Windows Registry Editor.

To re-enable Windows Credentials Provider

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\Credential Providers\{8A2C93D0-D55F-4045-99D7-B27F5E263407}** key.
3. Open the **Disabled** value, for **Value data**, type `0` and choose **OK**.
4. Close the Windows Registry Editor.

Configuring Authentication on Linux

To change the NICE DCV server's authentication method, you must configure the `authentication` parameter in the `dcv.conf` file.

To change the authentication method on Linux

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `authentication` parameter in the `[security]` section, and replace the existing value with either `system` or `none`.

   If there is no `authentication` parameter in the `[security]` section, add it manually using the following format:

```
[security]
authentication=method
```

3. Save and close the file.

Configuring NICE DCV Authorization

Authorization is used to grant or deny NICE DCV clients permissions to specific NICE DCV features. By default, the NICE DCV server grants the session owner full access to all features. However, you can specify custom permissions for your users using a custom permissions file.

NICE DCV Features

The following features can be referenced in the permissions file:

- `display` — Receive visual data from the NICE DCV server.
- `clipboard-copy` — Copy data from the NICE DCV server to the client clipboard.
- `clipboard-paste` — Paste data from the client clipboard to the NICE DCV server.
- `file-download` — Download files from the session storage.
- `file-upload` — Upload files to the session storage.
- `mouse` — Input from the client pointer to the NICE DCV server.
- `keyboard` — Input from the client keyboard to the NICE DCV server.
- `keyboard-sas` — Use secure attention sequence (Control + Alt + Del). Requires the `keyboard` feature. Supported on DCV 2017.3 and later.
- `touch` — Use native touch events. Supported on DCV 2017.3 and later. Not supported with Linux NICE DCV Servers.
- `pointer` — View NICE DCV server mouse position events and pointer shapes. Supported on DCV 2017.3 and later.
- `audio-out` — Play back NICE DCV server audio on the client.
- `audio-in` — Insert audio from the client to the NICE DCV server.
- `printer` — Print PDFs or XPS files from the NICE DCV server to the client.
- `smartcard` — Read the smart card from the client.

Permissions File

The permissions file defines the NICE DCV features to which users have access when they connect to a session. You can create a custom permissions file that defines what each user is allowed to do on a NICE DCV session. If you do not specify a customer permissions file, the NICE DCV server applies the default user permissions. Those permissions grant the session owner full access to all features, and deny access to all other users. For more information, see Creating a Permissions File (p. 37).

After you have created your custom permissions file, you can reference it when starting a new session using the `--permissions-file` option with the `dcv create-session` command. For more information about starting sessions, see Starting NICE DCV Sessions (p. 42).

Creating a Permissions File

You can create a custom permissions file using your preferred text editor. You might need to do the following when creating a custom permissions file.

Contents

Import an Existing Permissions File

The `imports` section is typically the first section of the permissions file. This section lets you reference and include existing permissions files. This enables you to incorporate previously defined NICE DCV permissions into your permissions file.

A permissions file can include multiple imports. An imported permissions file might import other permissions files.

To import an existing permissions file into your permissions file

- Use the `#import` statement and specify the location of the file with an absolute or a relative path..
  - Windows NICE DCV server:

```
#import ..\file_path\file
```

  - Linux NICE DCV server:

```
#import ../file_path/file
```

Example

The following statement imports a permissions file named `dcv-permissions.file` that is located in the NICE DCV installation folder on a Windows NICE DCV server using an absolute path:

```
#import c:\Program Files\NICE\DCV\dcv-permissions.file
```

Create Groups

The `[groups]` section of the permissions file lets you define user groups for users that have similar use cases, or permissions requirements. Groups can be assigned specific permissions. Permissions assigned to a group apply to all of the users that are included in the group.

To create groups in your permissions file, you must first add the groups section heading to the file.

```
[groups]
```

You can then create your groups below the section heading. To create a new group, provide the group name, and then specify the group members in a comma-separated list. Group members can be individual users, other groups, and operating system user groups.

```
group_name=member_1, member_2, member_3
```

To add a user to a group

Specify the user name.

Note
You can prefix the user name with `user:`.

Note
Windows domain user names can include a domain name.

```
group_name=user_1, user:user_2, domain_name\user_3
```

To add an existing group to a group

Specify the group name prefixed with `group:`:

```
group_name=group:group_1, group:group_2
```

To add an operating system user group to a group

Specify the group's name prefixed with `osgroup:`:

```
group_name=osgroup:os_group_1, osgroup:os_group2
```

Example

The following example adds the groups section heading and creates a group named `my-group` that includes individual users named `john` and `jane`, an existing group named `observers`, and an operating system user group named `guests`:

```
[groups]
my-group=john, user:jane, group:observers, osgroup:guests
```

Create Aliases

The `[aliases]` section of the permissions file lets you create sets of NICE DCV features. After an alias has been defined, you can grant or deny groups or individual users permissions to use it. Granting or denying permissions to an alias grants or denies permissions to all of the features that are included in it.

To create aliases in your permissions file, you must first add the aliases section heading to the file.

```
[aliases]
```

You can then create your aliases below the section heading. To create a new alias, provide the alias name, and then specify the alias members in a comma-separated list. Alias members can be individual NICE DCV features or other aliases.

```
alias_name=member_1, member_2, member_3
```

Example

The following example adds the aliases section heading and creates an alias named `file-management` that includes the `file-upload` and `file-download` features, and an existing alias named `clipboard-management`.

```
[aliases]
file-management=file-upload, file-download, clipboard-management
```

Add Permissions

The `[permissions]` section of the permissions file lets you control user and group access to specific features or aliases.

To add permissions to your permissions file, first add the permissions section heading to the file.

```
[permissions]
```

You can then add your permissions below the section heading. To add a permission, specify the actor that it governs, the rule to be applied, and the features that it applies to.

```
actor rule features
```

The **actor** can be a user, a group, or an operating system group. Groups must be prefixed with `group:`, and operating system groups must be prefixed with `osgroup:`. NICE DCV includes a built-in `%owner%` reference that can be used to refer to the session owner, and a built-in `%any%` reference that can be used to refer to any user.

The following **rules** can be used in permissions statements:

- `allow` — Grants access to the feature.
- `disallow` — Denies access to the feature, but can be overridden by subsequent permissions.
- `deny` — Denies access to the feature and cannot be overridden by subsequent permissions.

The **features** can include individual NICE DCV features, aliases, or a combination of both. The list of features must be separated by a space. NICE DCV includes a built-in `builtin` alias that includes all of the NICE DCV features.

Example

The following example adds the permissions section heading and adds four permissions. The first permission grants a user named `john` access to the `display`, `file-upload`, and `file-download` features. The second permission denies the `observers` group access to the `audio-in` and `audio-out` features, and an alias named `clipboard-management`. The third permission grants the `guests` operating system group access to the `clipboard-management` and `file-management` aliases. The fourth permission grants the session owner access to all features.

```
[permissions]
john allow display file-upload file-download
group:observers deny audio-in audio-out clipboard-management
osgroup:guests allow clipboard-management file-management
%owner% allow builtin
```

Managing NICE DCV Sessions

You must create a NICE DCV session on your NICE DCV server that your clients can connect to. Clients can only connect to a NICE DCV server if there is an active session.

Topics

Introduction to NICE DCV sessions

Every NICE DCV session has the following attributes:

- **ID** — Used to uniquely identify the session on the NICE DCV server.
- **Owner** — The NICE DCV user who created the session. By default, only the owner can connect to the session.

NICE DCV clients need this information to connect to the session.

NICE DCV offers two types of sessions:

Console Sessions

Console sessions are supported on Windows and Linux NICE DCV servers.

Only one console session can be hosted on the NICE DCV server at a time. Console sessions are created and managed by the Administrator on Windows NICE DCV servers, and the root user on Linux NICE DCV servers.

> **Note**
> You can't run console and virtual sessions on the same NICE DCV server at the same time.

Virtual Sessions

Virtual sessions are supported on Linux NICE DCV servers only.

A NICE DCV server can host multiple virtual sessions simultaneously. Virtual sessions are created and managed by NICE DCV users. NICE DCV users can only manage sessions that they have created. The root user can manage all virtual sessions that are currently running on the NICE DCV server.

To use virtual sessions, ensure that you have properly installed and configured an X server. A new virtual X server instance is created for each session. Each session uses the display provided by its virtual server. This enables you to host multiple virtual sessions on a single NICE DCV server. To share hardware-based OpenGL across multiple virtual sessions, you must connect the virtual X server instance to the GPU by configuring the `dcv-gl.conf` file.

> **Note**
> You can't run console and virtual sessions on the same NICE DCV server at the same time.

Using the Command Line Tool to Manage NICE DCV sessions

The NICE DCV server ships with a command line tool that can be used to start, stop, and view NICE DCV sessions.

To use the command line tool on a Windows NICE DCV server, navigate to the folder in which the `dcv.exe` file is located, `C:\Program Files\NICE\DCV\Server\bin\` by default, and open a command prompt window.

On Linux NICE DCV servers, the command line tool is automatically configured in the `$PATH` environment variable. This enables you to use the command line tool from any folder. Open a terminal window and type the command to execute.

Starting NICE DCV Sessions

By default, a console session is automatically created on Windows NICE DCV servers after installation. The default console session is owned by `Administrator`, and has a default session ID of `console`. If you chose to prevent the automatic console session when installing the NICE DCV server, you need to create one manually. You can also enable or disable the automatic console session at any time after installing the NICE DCV server.

> **Note**
> Linux NICE DCV servers do not get a default console after installation.

If you are using a floating license on an on-premises or alternative cloud-based server and you exceed the maximum number of concurrent sessions supported by your license, you could get a `no licenses` error. If you get this error, stop an unused session to release the license and try again.

The NICE DCV server must be running to start a session. For more information, see Starting the NICE DCV Server (p. 26).

Contents

Manually Starting Console and Virtual Sessions

You can start a NICE DCV session at any time. You can only run one console session at a time. If you are using a Linux NICE DCV server, you can run multiple virtual sessions simultaneously.

To create a console or virtual session on a Windows or Linux NICE DCV server

Use the `dcv create-session` command and specify the session type and a unique session ID.

The following options can be used with the `dcv create-session` command:

--type=console|virtual

This option is supported on Linux NICE DCV servers only. It specifies the type of session to be created, and can be either `console` or `virtual`.

--user

This option is supported with virtual sessions on Linux NICE DCV sessions only. This value is the user to be used to create the session. Only the root user can impersonate other users.

--owner

Specifies the session owner. Defaults to the currently logged in user if omitted.

--permissions-file

Specifies a path to a custom permissions file. Defaults to the server defaults if omitted.

--storage-root

Specifies the path to the folder used for session storage.

--gl

This option is supported with virtual sessions on Linux NICE DCV sessions only. It overrides the default `dcv-gl` state, and can be either `on` or `off`.

--max-concurrent-clients

Specifies the maximum number of NICE DCV clients that are allowed to connect to the session. Defaults to unlimited connections if omitted.

--init

This option is supported with virtual sessions on Linux NICE DCV servers only. It specifies the path to a custom `init` script. The script can be used to start a specific desktop environment and launch specific applications automatically when the session starts. The script must be executable. Defaults to a script that starts the default desktop environment if omitted.

Examples

Example 1 - Console session

The following command creates a `console` session owned by `dcv-user` with a unique session ID `my-session`, and specifies a permissions file named `perm-file.txt`:

- Windows NICE DCV server

```
C:\> dcv create-session --owner dcv-user --permissions-file perm-file.txt my-session
```

- Linux NICE DCV server

```
$ sudo dcv create-session --type=console --owner dcv-user --permissions-file perm-
file.txt my-session
```

Example 2 - Virtual Session (Linux NICE DCV Servers only)

The following command creates a `virtual` session using the `root` user to impersonate the intended session owner, `dcv-user`. The session is owned by `dcv-user` even though it is created by the root user.

```
$ sudo dcv create-session --owner dcv-user --user dcv-user my-session
```

Example 3 - Virtual Session (Linux NICE DCV Servers only)

The following command creates a `virtual` session owned by the user who creates it:

```
$ dcv create-session my-session
```

Enabling Automatic Console Sessions

Enabling an automatic console session ensures that a console session is automatically created each time that the NICE DCV server starts. The automatic console session is owned by the NICE DCV user specified by the `owner` configuration parameter, and its session ID is always `console`.

Other parameters affecting automatic console sessions are `max-concurrent-clients`, `permissions-file`, and `storage-root`. For more information about these parameters, see `session-management/automatic-console-session` Parameters (p. 52).

Note
NICE DCV does not support automatic virtual sessions.

To enable an automatic console session on a Windows NICE DCV server

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/session-management** key.
3. Create a `create-session` parameter:
   a. Open the context (right-click) menu for the **session-management** key in the left-hand panel and choose **New**, **DWORD (32-bit) Value**.
   b. For **Name**, type `create-session` and press **Enter**.
   c. Open the **create-session** parameter. For **Value data**, type `1`, and choose **OK**.
4. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/session-management/automatic-console-session** key.
5. Create an `owner` parameter:
   a. Open the context (right-click) menu for the **automatic-console-session** key in the left-hand panel and choose **New**, **String Value**.
   b. For **Name**, type `owner` and press **Enter**.
   c. Open the **owner** parameter. For **Value data**, type the session owner's name and choose **OK**.
6. Choose **OK** and close the Windows Registry Editor.
7. Stop (p. 27) and restart (p. 26) the NICE DCV server.

To enable an automatic console session on a Linux NICE DCV server

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Add the `create-session` and `owner` parameters to the `[session-management/automatic-console-session]` section using the following format:

```
[session-management]
create-session = true

[session-management/automatic-console-session]
owner="session_owner"
```

3. Save and close the file.
4. Stop (p. 27) and restart (p. 26) the NICE DCV server.

Stopping NICE DCV Sessions

A console session can only be stopped by the administrator on Windows NICE DCV servers, and the root user on Linux NICE DCV servers. A virtual session on a Linux NICE DCV server can only be stopped by the root user or the NICE DCV user who created it.

To stop a console or virtual session on a Windows or Linux NICE DCV servers

Use the `dcv close-session` command and specify the unique session ID:

```
dcv close-session session_id
```

For example, the following command stops a session with the unique ID of `my-session`:

```
dcv close-session my-session
```

Viewing NICE DCV Sessions

The administrator on a Windows NICE DCV server or the root user on a Linux NICE DCV server can view all active sessions running on the server. NICE DCV users can only view sessions that they have created.

To view the active console or virtual sessions on a Windows or Linux NICE DCV server

Use the `dcv list-sessions` command:

```
dcv list-sessions
```

The command returns a list of active sessions in the following format:

```
Session: session_id (owner: session_owner)
```

Troubleshooting NICE DCV

This chapter explains how to identify and troubleshoot problems that you might have with NICE DCV.

Topics

If you need additional help, you can log a support ticket on the NICE Support Portal.

Using the Log Files

The NICE DCV log files can be used to identify and troubleshoot problems with your NICE DCV server. The NICE DCV log files can be found in the following location on your NICE DCV server:

- Windows server

```
C:\ProgramData\NICE\dcv\log\
```

- Linux server

```
/var/log/dcv/
```

The NICE DCV server enables you to configure the verbosity level of the log files. The following verbosity levels are available:

- `error` — Provides the least detail. Includes errors only.
- `warning` — Includes errors and warnings.
- `info` — The default verbosity level. Includes errors, warnings, and information messages.
- `debug` — Provides the most detail. Provides detailed information that is useful for debugging issues.

Changing Log File Verbosity on Windows

To configure the log file verbosity, you must configure the `level` parameter using the Windows Registry Editor.

To change the log file verbosity on Windows

1. Open the Windows Registry Editor.
2. Navigate to the **HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/log/** key.
3. Open the **level** parameter by double-clicking. For **Value data**, type either `error`, `warning`, `info`, or `debug`, depending on the required verbosity level.
4. Choose **OK** and close the Windows Registry Editor.

Changing Log File Verbosity on Linux

To configure the log file verbosity, you must configure the `level` parameter in the `dcv.conf` file.

To change the log file verbosity on Linux

1. Navigate to `/etc/dcv/` and open the `dcv.conf` with your preferred text editor.
2. Locate the `level` parameter in the `[log]` section, and replace the existing verbosity level with either `error`, `warning`, `info`, or `debug`.

```
[log]
level="verbosity_level"
```

3. Save and close the file.

Common Issues

This topic lists some common issues.

Topics

X Forwarding

By default, NICE DCV 2017 prevents the use of X forwarding due to inherent security risks. NICE DCV inherits this behavior from the newer versions of the Xorg server. The NICE DCV server implements the following inherited mitigations to minimize the security risks:

- By default, the NICE DCV server prevents X connections from the network. The NICE DCV server is configured to start with `-nolisten tcp` command line option. However, it is possible to change the default behavior to enable remote X connections to the X server. For more information about the workaround, see Enable Remote X Connections to the X Server (p. 47).
- By default, the X server disables GLX indirect contexts. Due to conflicts with DCV-GL, there is currently no workaround to enable GLX indirect contexts

For more information about the security risks and the mitigations, see the X.Org Security Advisory.

Enable Remote X Connections to the X Server

By default, NICE DCV is configured to start with `-nolisten tcp` command line option to reduce exposure to the security risks. However, it is possible to change the default behavior to enable X forwarding.

To enable X forwarding

Open `/etc/dcv/dcv.conf` using your preferred text editor and add the following to the end of the file:

- To enable X forwarding over IPv4 and IPv6

```
[session-management]
virtual-session-xdcv-args="-listen tcp"
```

- To enable X forwarding over IPv4 only

```
[session-management]
virtual-session-xdcv-args="-listen inet -nolisten inet6"
```

Note
This does not affect existing sessions. X forwarding is only enabled for new sessions started after enabling the X forwarding.

To test the X forwarding

1. Connect the NICE DCV session.
2. Confirm that the NICE DCV server is listening on a port in the range between 6000-6063.

```
$ netstat -punta | grep 600
```

3. Retrieve the NICE DCV session display number.

```
$ dcv describe-session session_name | grep display
```

4. SSH into the remote server on which the application is hosted.

```
$ ssh user@remote_server_ip
```

5. From the remote server, export the display environment variable to point to the X server of the NICE DCV session.

```
$ export DISPLAY=dcv_server_ip:display_number
```

6. From the remote server, run an application to test the X forwarding functionality. For example:

```
xterm
```

The test application, in this case xterm, should appear in NICE DCV server's desktop environment.

NICE DCV Server Parameter Reference

The following table lists the parameters that can be configured to customize the NICE DCV server.

Note
The **Reload context** column in each table indicates when the parameter is reloaded. Possible contexts include:

- `server`—The parameter is loaded once when the server is started. If the parameter value is updated, the new value is loaded when the server is restarted.
- `session`—The parameter is loaded when the session is created. If the parameter value is updated, the new value is loaded for subsequent sessions.
- `connection`—The parameter is loaded when a new client connection is established. If the parameter value is updated, the new value is used for subsequent client connections.
- `custom`—The conditions under which the parameter loads is unique to this parameter. See the parameter description for more information.

Topics

`connectivity` Parameters

The following table describes the configuration parameters in the `[connectivity]` section of the `/etc/dcv/dcv.conf` file for Linux NICE DCV servers, and the `connectivity` registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
web-port	DWORD (32-bit)	server	8443	**TCP port for the web client** — Specifies the TCP port on which the DCV server listens for client

Parameter	Type (Windows only)	Reload context	Default value	Description
				connections. The port number must be between 1024 and 65535.
web-url-path	String	server	'/'	**URL path for the embedded web server** — Specifies the URL path for the embedded web server, must start with '/'. For example, setting it to /test/foo means that the web server is reachable at https://host:port/test/foo.
web-root	String	server	''	**Document root for the embedded web server** — Specifies the document root for the embedded web server.
ws-keepalive-interval	DWORD (32-bit)	server	10	**Websocket keepalive interval** — Specifies the interval (in seconds) after which to send a keepalive message. If set to 0, the keepalive message is disabled.
idle-timeout	DWORD (32-bit)	custom	60	**Idle timeout** — Specifies the number of minutes to wait before disconnecting idle clients. Specify 0 to never disconnect idle clients. This parameter value is read every 5 seconds.
idle-timeout-warning	DWORD (32-bit)	custom	350	**Idle timeout warning** — Specifies the number of seconds relative to idle-timeout to wait before warning idle clients about idle timeout disconnection. Specify 0 to never warn idle clients.

`session-management` Parameters

The following table describes the configuration parameters in the `[session-management]` section of the `/etc/dcv/dcv.conf` file for Linux NICE DCV servers, and the `session-management` registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
create-session	DWORD (32-bit)	server	false	**Create a console session at server startup** — Specifies

Parameter	Type (Windows only)	Reload context	Default value	Description
				whether to automatically create a console session (with ID "console") at server startup.
max-concurrent-clients	DWORD (32-bit)	session	-1	**Maximum number of concurrent clients per session** — Specifies the maximum number of concurrent clients per session. If set to -1, no limit is enforced. To set the limit only for the automatic session, use 'max-concurrent-clients' of section 'session-management/automatic-console-session'.
enable-gl-in-virtual-sessions	String	session	'default-on'	**Whether to employ dcv-gl feature** — Specifies whether to use the dcv-gl feature (a license is required). Allowed values: 'always-on', 'always-off', 'default-on', 'default-off'.
virtual-session-font-path	String	session	''	**Whether to add special font paths** — Specifies the path of special fonts. Some applications require a special font to be passed to the X server.
virtual-session-default-layout	String	session	[]	**Default layout for virtual sessions** — If this is set, Xdcv is configured to create the specified layout at startup. Each monitor can be configured with resolution (w,h) and position (x,y). All specified monitors are enabled. Default layout example value: [{'w':<800>, 'h':<600>, 'x':<0>, 'y': <0>}, {'w':<1024>, 'h':<768>, 'x':<800>, 'y':<0>}] For this setting, the maximum number of monitors (specified in the virtual-session-monitors setting) has more priority than the number of elements in the array. For example, if five monitors have been set, but the maximum number of monitors is four, only the first four monitors are created. If this key is set, the number of enabled monitors (specified in the virtual-session-monitors setting) is ignored.

Parameter	Type (Windows only)	Reload context	Default value	Description
virtual-session-xdcv-args	String	session	''	**Additional arguments to pass to Xdcv** — Specifies any additional arguments to be passed to Xdcv.

session-management/defaults Parameters

The following table describes the configuration parameters in the `[session-management/defaults]` section of the `/etc/dcv/dcv.conf` file for Linux NICE DCV servers, and the `session-management/defaults` registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
permissions-file	String	session	''	**Default permissions included in all sessions** — Specifies the path to the permissions file to be automatically merged with the permissions selected by the user for each session. If empty, use the 'default.perm' file, which is located in /etc/dcv/ for Linux, or in the DCV installation folder (for example, 'C:\Program Files\NICE\DCV\Server\conf') for Windows.

session-management/automatic-console-session Parameters

The following table describes the configuration parameters in the `[session-management/automatic-console-session]` section of the `/etc/dcv/dcv.conf` file for Linux NICE DCV servers, and the `session-management/automatic-console-session` registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
max-concurrent-clients	DWORD (32-bit)	server	-1	**Maximum number of concurrent clients per session** — Specifies the maximum number of concurrent clients allowed per session. If set to -1, no limit is enforced.

Parameter	Type (Windows only)	Reload context	Default value	Description
permissions-file	String	server	''	**Permissions file for the automatic "console" session** — Specifies the path to the permissions file to be used to check user access to DCV features. If empty, only the owner has full access to the session.
owner	String	server	''	**Owner of the automatically created "console" session** — Specifies the username of the "console" session owner. If empty, the owner is the user who started the DCV server. This setting is applied only to the "console" session automatically created at server startup when the create-session setting is set to true.
storage-root	String	server	''	**Path to file storage root folder** — Specifies the full path to the folder to be used for console session storage. If the storage-root is empty or the folder does not exist, file storage is disabled.

security Parameters

The following table describes the configuration parameters in the [security] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the security registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
authentication	String	server	'system'	**Authentication method** — Specifies the client authentication method used by the DCV server. Use 'system' to delegate client authentication to the underlying operating system. Use 'none' to disable client authentication and grant access to all clients.
authentication-threshold	DWORD (32-bit)	server	3	**Authentication threshold** — Specifies how many times each client can fail authentication

Parameter	Type (Windows only)	Reload context	Default value	Description
				before the connection is closed by the server. To allow unlimited authentication attempts, use 0.
passwd-file	String	server	''	**Password file** — Specifies the password file to be used to check user credentials (only with dcv authentication mode). If empty, use the default file in ${XDG_CONFIG_HOME}/NICE/dcv/passwd for Linux, or %CSIDL_LOCAL_APPDATA%\NICE\dcv\passwd for Windows.
pam-service-name	String	server	'dcv'	**PAM service name** — Specifies the name of the PAM configuration file used by DCV. The default PAM service name is 'dcv' and corresponds with the /etc/pam.d/dcv configuration file. This parameter is only used if the 'system' authentication method is used.
enable-gssapi	DWORD (32-bit)	server	false	**Enable GSSAPI SASL mechanism** — Enables or disables GSSAPI SASL mechanism, that allows DCV authentication with kerberos.
server-fqdn	String	server	''	**Server FQDN** — Server fully qualified domain name. Empty means gethostname().
user-realm	String	server	''	**Server user realm** — Specifies a user realm for the server.
ca-file	String	server	''	**CA file** — Specifies the file containing the certificate authorities (CAs) trusted by the DCV server. If empty, use the default trust store provided by the system.
auth-token-verifier	String	server	''	**The endpoint of the authentication token verifier** — Specifies the endpoint (URL) of the authentication token verifier used by the DCV server. If empty, the internal authentication token verifier is used.

Parameter	Type (Windows only)	Reload context	Default value	Description
no-tls-strict	DWORD (32-bit)	server	false	**Disable strict certificate validation** — Enables or disables strict certificate validation when connecting to an external authentication token verifier. Strict certificate validation must be disabled if the authentication token verifier uses a self-signed certificate.
allowed-http-host-regex	String	server	'^.+$'	**Allowed host regular expression** — Specifies a regular expression pattern representing the host names that this DCV server can serve. If the Host header of an incoming HTTP request does not match this pattern, the request itself fails with a 403 Forbidden status code. This is a security measure to prevent HTTP Host header attacks. The pattern must be a valid Javascript-like regular expression. Letters in the pattern match both uppercase and lowercase letters. Example: '^(www\.)?example\.com$']
allowed-ws-origin-regex	String	server	'^https://.+$'	**Allowed origins** — Specifies a regular expression pattern representing the origins that this DCV server accepts. When establishing a WebSocket connection, the Origin header field in the client's handshake indicates the origin of the script establishing the connection. If the Origin header of an incoming HTTP request does not match this pattern, the request itself fails with a 403 Forbidden status code. This is a security measure to prevent cross-site WebSocket hijacking (CSWSH) attacks. The pattern must be a valid Javascript-like regular expression. Letters in the pattern match both uppercase and lowercase letters. The Origin header has the form: <scheme> "://" <host> [":" <port>]. Example: '^https://(www\.)?example\.com(:443)?$'

Parameter	Type (Windows only)	Reload context	Default value	Description
max-connections-per-user	DWORD (32-bit)	server	10	**Maximum number of user's connections** — Specifies the maximum number of allowed concurrent connections per user. Exceeding connections are rejected.
connection-estab-timeout	DWORD (32-bit)	server	5	**Connection establishment timeout** — Specifies the amount of time (in seconds) allowed for the connection procedure to be completed before timing out. If the procedure takes more, then the connection is closed. If set to 0, the connection establishment does not time out.
connection-setup-timeout	DWORD (32-bit)	server	5	**Channel connection setup timeout** — Specifies the amount of time (in seconds) allowed for the channel connection setup procedure to be completed before timing out. If the procedure takes more, then the channel is closed. If set to 0, the channel connection setup does not time out.
auth-connection-setup-timeout	DWORD (32-bit)	server	120	**Authentication channel connection setup timeout** — Specifies the amount of time (in seconds) allowed for the authentication channel connection setup procedure to be completed before timing out. If the procedure takes more, then the channel is closed. If set to 0, the authentication channel connection setup timeout is disabled.

Parameter	Type (Windows only)	Reload context	Default value	Description
ciphers	String	server	'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-SHA384'	**Cipher list used on the TLS connections** — Specifies the cipher list used on TLS connections. The cipher list must be separated using the character ":" and must be supported by openssl and the clients.
os-auto-lock	DWORD (32-bit)	custom	true	**Whether to lock the OS session when last client connection ends** — If enabled, the OS session is locked when the last client connection is closed. This setting is read every time the session is going to be closed.

`license` Parameters

The following table describes the configuration parameters in the `[license]` section of the `/etc/dcv/dcv.conf` file for Linux NICE DCV servers, and the `license` registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
license-file	String	session	''	**License** — Specifies a demo license file or RLM server port and hostname. If you are using a floating license on an RLM server, use this parameter to specify the RLM server's port and hostname in the port@hostname format. If you are using an extended demo license, and you have not placed the license.lic file in the default location, use this parameter to specify the full path of license.lic license file. If the default file does not exist, a demo license is used. This value is read from the configuration and updated every time a new session is created.

input Parameters

The following table describes the configuration parameters in the [input] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the input registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
enable-relative-mouse	DWORD (32-bit)	session	true	**Whether to allow relative mouse movements** — Whether to allow relative mouse movements.
enable-autorepeat	DWORD (32-bit)	session	true	**Whether to allow autorepeat on Linux** — Specifies whether to allow autorepeat for a single key. Excludes key combinations and key modifiers.
enable-touch	DWORD (32-bit)	session	true	**Whether to allow touch input** — Specifies whether touch is enabled.

display Parameters

The following table describes the configuration parameters in the [display] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the display registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
max-compressor-threads	DWORD (32-bit)	session	4	**Max compressor threads** — Specifies the maximum number of compressor threads.
grabber-target-fps	DWORD (32-bit)	session	0	**Target frames per second of frame grabber** — The upper limit to grab frames per second. A value of 0 defaults to target-fps.
enable-qu	DWORD (32-bit)	session	true	**Whether to send quality updates** — Specifies whether to send quality updates.
enable-client-resize	DWORD (32-bit)	session	true	**Whether to allow clients to set the display layout** — Specifies whether clients are allowed to set the display layout.
max-head-resolution	String	session	(4096, 2160)	**Max head resolution** — The maximum resolution of a

Parameter	Type (Windows only)	Reload context	Default value	Description
				display head. A display head is equivalent to a host monitor.
min-head-resolution	String	session	(640, 480)	**Min head resolution** — The minimum resolution of a display head. A display head is equivalent to a host monitor.
max-num-heads	DWORD (32-bit)	session	4	**Max number of heads** — Specifies the maximum number of heads. A display head is equivalent to a host monitor.
console-session-default-layout	String	session	[]	**Default screen resolution and position for console sessions** — Specifies the default screen resolution and position for console sessions. If this is set, DCV sets the requested layout at startup. Each monitor can be configured with resolution (w,h) and position (x,y). All specified monitors are enabled. Default layout example value: [{'w':<800>, 'h':<600>, 'x':<0>, 'y': <0>}, {'w':<1024>, 'h':<768>, 'x':<800>,'y':<0>}]
use-grabber-dirty-region	DWORD (32-bit)	session	true	**Whether to use dirty regions** — Specifies whether to use dirty screen regions. If enabled, the grabber tries to compute new frames out of the dirty regions from the screen.
cuda-devices	String	connection	[]	**CUDA devices used for stream encoding** — Specifies the list of local CUDA devices which DCV uses to distribute encoding and CUDA workloads. Each device is identified by a number that can be retrieved from the nvidia-smi command. For example, cuda-devices=['0', '2'] indicates that DCV uses two GPUs, with IDs 0 and 2. This setting is similar to the CUDA_VISIBLE_DEVICES environment variable, but it only applies to DCV.

display/linux Parameters

The following table describes the configuration parameters in the [display/linux] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the display/linux registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
gl-displays	String	connection	[':0.0']	**3D accelerated X displays** — Specifies the list of local 3D accelerated X displays and screens used by DCV for OpenGL rendering in virtual sessions. If this value is missing, you can't run OpenGL applications in virtual sessions. This setting is ignored for console sessions.
h264-encoder-displays	String	connection	[]	**H.264 encoder X displays** — Specifies the list of local X displays and screens that support accelerated H.264 encoding. If empty, DCV uses the same display selected for OpenGL rendering. This setting is useful only in cases when some of the GPUs installed on the system do not provide acceleration for H.264 encoding using one of the supported technologies.

log Parameters

The following table describes the configuration parameters in the [log] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the log registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
directory	String	server	''	**Log output directory** — Specifies the destination to which logs are saved. If not specified it defaults to "C:\ProgramData\NICE\DCV\log\" on Windows and to "/var/log/dcv/" on Linux.
level	String	custom	'info'	**Log level** — Specifies the log file verbosity level. The verbosity

Parameter	Type (Windows only)	Reload context	Default value	Description
				levels (in order of the amount of detail they provide) are: 'error', 'warning', 'info', and 'debug'. This value is automatically reload when it is changed on the configuration, so that it can be changed without restarting the server.
rotate	DWORD (32-bit)	server	10	**Number of log file rotations** — Specifies the number of times that log files are rotated before being removed. If the value is 0, old versions are removed rather than rotated.
transfer-audit	String	server	'none'	**Transfer direction to audit** — Specifies which transfer direction to audit. If this parameter is enabled, a new CSV file logs transfers between the server and clients. The allowed values are: 'none', 'server-to-client', 'client-to-server', and 'all'. If this value is missing or equal to 'none', transfer audits are disabled and no file is created.

windows Parameters

The following table describes the configuration parameters in the `[windows]` section of the `/etc/dcv/dcv.conf` file for Linux NICE DCV servers, and the `windows` registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
disable-display-sleep	DWORD (32-bit)	session	true	**Prevent display from entering power-saving mode** — Specifies whether to prevent the display from entering power-saving mode.
printer	String	session	'DCV printer'	**Printer to be set as default** — Specifies the name of the virtual DCV printer. Defaults to 'DCV printer'.

clipboard Parameters

The following table describes the configuration parameters in the [clipboard] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the clipboard registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
max-payload-size	DWORD (32-bit)	session	20971520	**Maximum size of clipboard's data** — Specifies the maximum size (in bytes) of clipboard data that can be transferred from server to clients. If this value is missing, the default limit of 20 MB is enforced.
max-text-len	DWORD (32-bit)	session	-1	**Maximum number of characters of clipboard's text** — Specifies the maximum number of characters of clipboard text that can be transferred from server to clients. If this value is missing or set to -1, no limit is enforced.
max-image-area	DWORD (32-bit)	session	-1	**Maximum area of clipboard's image** — Specifies the maximum area (number of pixels) of clipboard images that can be transferred from server to clients. If this value is missing or set to -1, no limit is enforced.

smartcard Parameters

The following table describes the configuration parameters in the [smartcard] section of the /etc/dcv/dcv.conf file for Linux NICE DCV servers, and the smartcard registry key for Windows NICE DCV servers.

Parameter	Type (Windows only)	Reload context	Default value	Description
enable-cache	String	custom	'default-off'	**Whether to enable smartcard caching messages.** — Enables or disables smart card caching. When enabled, the NICE DCV server caches the last value received from the client's smart card. Future calls are retrieved directly from the server's cache,

Parameter	Type (Windows only)	Reload context	Default value	Description
				instead of from client. This helps to reduce the amount of traffic that is transferred between the client and the server, and improves performance. Allowed values include 'always-on', 'always-off', 'default-on', and 'default-off'. This value is read from the configuration every time a client smartcard application is started.

Modifying Configuration Parameters

This section describes how to modify the configuration parameters for your NICE DCV server. For more information about the registry keys for Windows servers, sections for Linux servers, parameter names, types, and valid values, see the NICE DCV Server Parameter Reference (p. 49).

Topics

Windows NICE DCV Servers

For Windows NICE DCV servers, modify the configuration parameters using the Windows Registry Editor, PowerShell, or the command line.

To modify a configuration parameter using the Windows Registry Editor

1. Open the Windows Registry Editor.
2. Navigate to the following registry path:

```
HKEY_USERS/S-1-5-18/Software/GSettings/com/nicesoftware/dcv/
```

3. Select the registry key in which the parameter exists. If the registry key does not exist, create it using the exact key name described in the NICE DCV Server Parameter Reference (p. 49).
4. Open (double-click) the parameter. If the parameter does not exist, add it using the type and name described in the NICE DCV Server Parameter Reference (p. 49).

To modify a configuration parameter using the PowerShell

1. Run PowerShell as the administrator.
2. Add the registry key using the key name described in the NICE DCV Server Parameter Reference (p. 49).

```
PS C:\>  New-Item -Path "Microsoft.PowerShell.Core\Registry::\HKEY_USERS
\S-1-5-18\Software\GSettings\com\nicesoftware\dcv\" -Name registry_key -Force
```

3. Create the parameter in the registry key using the type and name described in the NICE DCV Server Parameter Reference (p. 49).

```
PS C:\>  New-ItemProperty -Path "Microsoft.PowerShell.Core\Registry::\HKEY_USERS
\S-1-5-18\Software\GSettings\com\nicesoftware\dcv\registry_key" -Name parameter_name -
PropertyType parameter_type -Value parameter_value -Force
```

To modify a configuration using the command line

1. Run the command line as the administrator.
2. Create the registry key and add the parameter using the key name, and parameter type and name described in the NICE DCV Server Parameter Reference (p. 49).

```
C:\>  reg.exe ADD "HKEY_USERS\S-1-5-18\Software\GSettings\com\nicesoftware\dcv
\registry_key" /v parameter_name /t parameter_type /d parameter_value /f
```

Linux NICE DCV servers

For Linux NICE DCV servers, the configuration parameters can be modified using a text editor or a command line tool, such as **crudini**.

To modify a configuration parameter using a text editor

1. Open `/etc/dcv/dcv.conf` using your preferred text editor.
2. Locate the appropriate section in the file. If the section does not exist, add it using the section name described in the NICE DCV Server Parameter Reference (p. 49).

```
[section]
```

3. Locate the parameter in the section and modify the value. If the parameter does not exist in the section, add it using the parameter name described in the NICE DCV Server Parameter Reference (p. 49).

```
parameter_name="parameter_value"
```

4. Save and close the file.

To modify a configuration parameter using crudini

Create the section and add the parameter using the section and parameter names described in the NICE DCV Server Parameter Reference (p. 49).

```
$  sudo crudini --set /etc/dcv/dcv.conf section_name parameter_name 'parameter_value'
```

Document History for NICE DCV

The following table describes the documentation for this release of NICE Desktop Cloud Visualization.

- **API version: latest**
- **Latest documentation update:** October 08, 2018

Change	Description	Date
Smart card caching	The NICE DCV server can now cache smart card data received from the client to help improve performance. For more information, see Configuring Smart Card Caching (p. 33).	October 08, 2018
Linux client	NICE DCV offers Linux clients for RHEL 7, CentOS 7, SLES 12, and Ubuntu 16.04/18.04. For more information, see Linux Client in the *NICE DCV User Guide*.	August 29, 2018
Updated Parameter Reference	The Parameter Reference has been updated. For more information, see NICE DCV Server Parameter Reference (p. 49).	August 07, 2018
USB remotization	NICE DCV enables clients to use specialized USB devices, such as 3D pointing devices or graphic tablets. For more information, see Enabling USB Remotization (p. 32).	August 07, 2018
Initial release of NICE DCV	First publication of this content.	June 05, 2018