

# Nerve 2.3.1 Documentation

© 2019-2021 TTTech Industrial Automation AG. All rights reserved.

## Getting started



This documentation provides assistance for the usage of the Nerve product. It offers guides that can be found in the header of the page, catering to different levels of usage of the Nerve product.

Note that this is the documentation of the latest Nerve version. For previous versions of the product, select [Previous Versions](#) in the footer.

### NOTE

Have the customer profile ready for the following chapters. It has been sent as part of the delivery. If the customer profile has not been part of the delivery, contact a sales representative or TTTech Industrial customer support at [support@tttech-industrial.com](mailto:support@tttech-industrial.com).

Select a Nerve Device from the table below for a step-by-step installation:



MFN 100



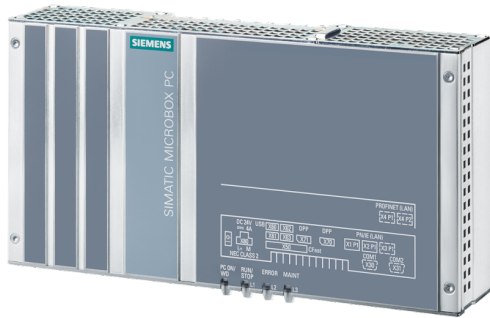
Kontron Kbox A-150-APL



Kontron Kbox A-250



Maxtang AXWL10-8665U



Siemens SIMATIC IPC427E



Siemens SIMATIC IPC127E



Supermicro SuperServer E100-9AP-IA



Supermicro SuperServer 1019D-16C-FHN13TP



Supermicro SuperServer 5029C-T



Vecow SPC-5600-i5-8500



Winmate EACIL20

## Latest documentation updates

### 2021-09-30

Updated documentation to reflect version 2.3.1:

- Added [Release Notes for 2.3.1](#)
- Nerve Data Services: Adapted documentation for new features (Kafka Producer output, incremental data mode and custom JSON format messages for JSON outputs)
- Minor changes and improvements

Added the [Security Recommendations](#) section.

Moved the [Downloads](#) and [Previous Versions](#) sections to the footer.

### 2021-08-03

Completed documentation for version 2.3.0. The latest updates include:

#### Nerve Data Services

Data Services documentation reworked to reflect the implementation of the new GUI and the local Gateway as the single point of configuration.

- [Applying a configuration through the GUI](#)  
Added instructions on how to configure the Gateway with the graphical configuration tool.
- [NerveDB with data buffering](#)  
Added a new example demonstrating the data buffering feature and the new NerveDB.
- [Nerve Data Services Database](#)  
Added a new page for more information on the database menu item.

### 2021-06-16

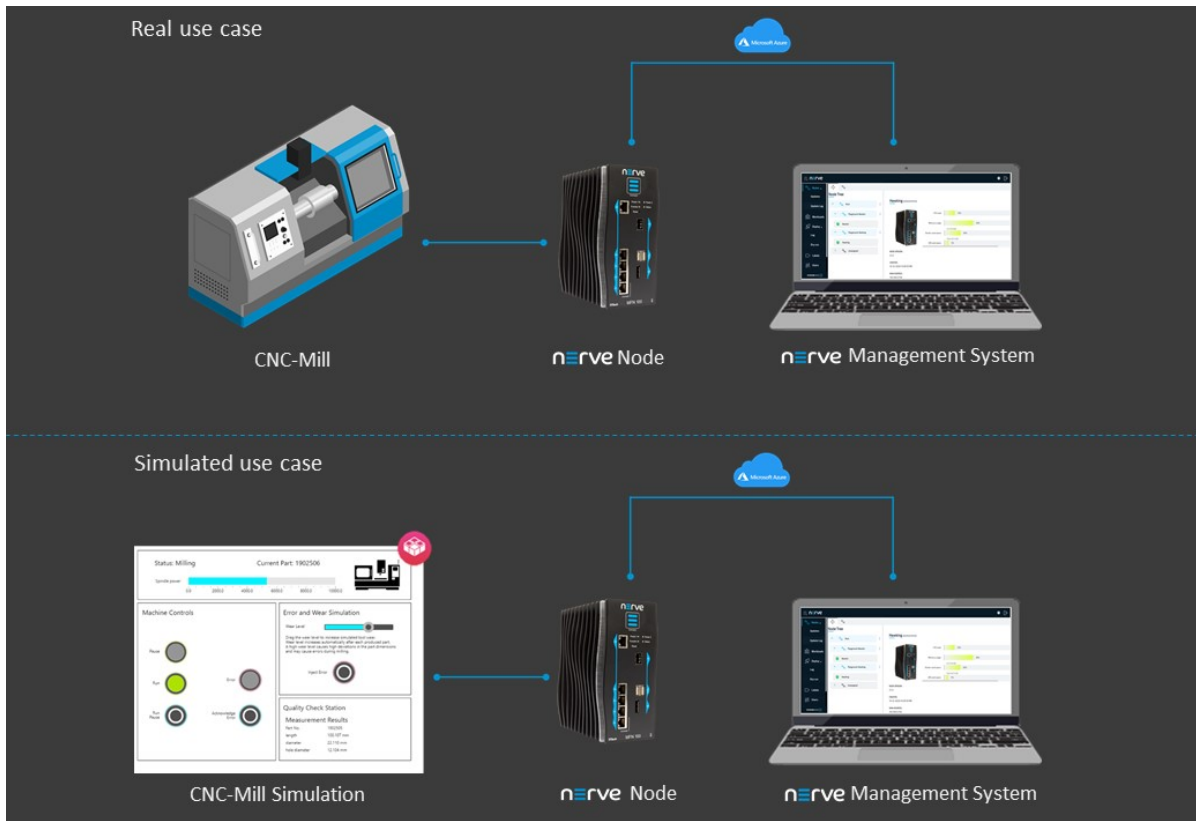
Partial release of version 2.3.0 documentation. New and updated chapters are:

- [Release Notes for 2.3.0](#)
- [Menu structure of the Management System](#)  
Updated table with descriptions of new elements in the Management System.
- [Setting system notifications](#)
- [Usage reports](#)
- [Logging and monitoring](#)
- [Managing nodes in the node tree](#)  
Updated table with descriptions of new elements in the node details view.

- [Workload control](#)  
Updated table with descriptions of new elements in the workload control window.
- [Applying configuration files to a workload](#)
- [Changing resource allocation of a deployed Virtual Machine workload](#)
- [Virtual machine video options](#)
- [Updating a deployed workload](#)
- [Local UI dashboard](#)  
Updated table with descriptions of new elements in the Local UI.
- [Workload management](#)  
Updated table with descriptions of new elements in the workload management window.
- [Updating a deployed workload in the Local UI](#)

## Tutorial: Machine efficiency insight

Nerve is a platform for applications running on machines and accessing machine data. This tutorial shows how to create an application to gain insight into production and machine efficiency. It shows how to use Nerve to measure and visualize overall equipment effectiveness (OEE) of a virtual CNC mill. In a real-life environment a CNC mill would be physically connected to a Nerve Node with Nerve installed. A remote connection is established to the Nerve Node via the Nerve Management System.



Going through this demo will take about 30 minutes and cover how to deploy container workloads, remotely access nodes and configure a Node-RED system to display an OEE and quality control dashboard.

Items needed for this tutorial are:

- a workstation or PC

- login credentials to the Nerve Management System and the fleshsaring server
- the Nerve Connection Manager app
- a flow file for Node-RED

All links, login credentials and the instructions to download files will be sent via emails. In case files or login credentials are missing, contact [trynerve@tttech-industrial.com](mailto:trynerve@tttech-industrial.com).

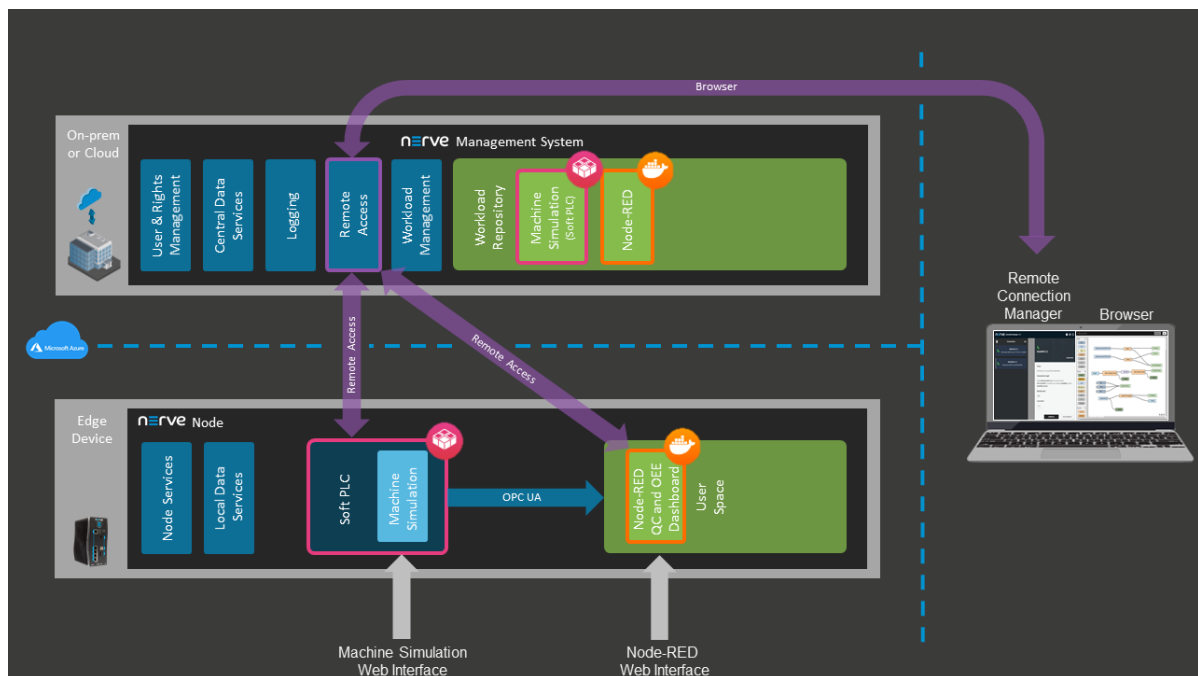
The tutorial environment is set up in the evaluation laboratory at TTTech Industrial in Vienna. The Nerve Device is an MFN 100 running an Intel Atom quad core CPU.

## NOTE

In case the evaluation system is used, be aware that other users may access the same evaluation system, meaning that other nodes might be registered in the Management System. Information and workloads uploaded into the system may be visible to other users.

## Tutorial architecture

In the course of this tutorial a simulation of a CNC machine (machine simulation) is installed on a Nerve Device. This machine simulation creates data which is read by the Node-RED application and visualized on a server running on the Nerve Device. The Node-RED application transforms the data received into an OEE and quality dashboard. The machine simulation, Node-RED and all dashboards are accessed on the Nerve Device directly, using the remote access feature of the Nerve cloud management platform called the Nerve Management System. In Nerve, applications managed by customers are called workloads.



## Machine simulation

The simulation model simulates a machine which continuously creates parts and is subject to wear. The machine provides status information and Quality Control

information through a built-in OPC UA server. The application is programmed in IEC 61131-3 and running in a soft PLC.

## Node-RED

Node-RED is an open source programming tool for wiring together sensor inputs, APIs and online services. In this tutorial Node-RED reads data from the machine simulation through OPC UA, modifies it and displays it on an OEE and quality dashboard. The system is implemented based on predefined graphical modules, which are connected through drag and drop.

Refer to [Node-RED documentation](#) for more information on Node-RED. TTTech provides Node-RED as a supported third party application.

## Initial setup

The Nerve Management System is set up so that the tutorial can be started straightaway. The necessary applications have already been configured and the remote connections are set up. The evaluation system does not limit experimentation to the extent of this tutorial. It is possible to create, deploy and test new workloads. However, the features of Local UI, Data Services and user management are not enabled in the standard evaluation system. Contact [sales@tttech-industrial.com](mailto:sales@tttech-industrial.com) for information on how to obtain the full version.

To explore the Nerve system even further, refer to the [user guide](#).

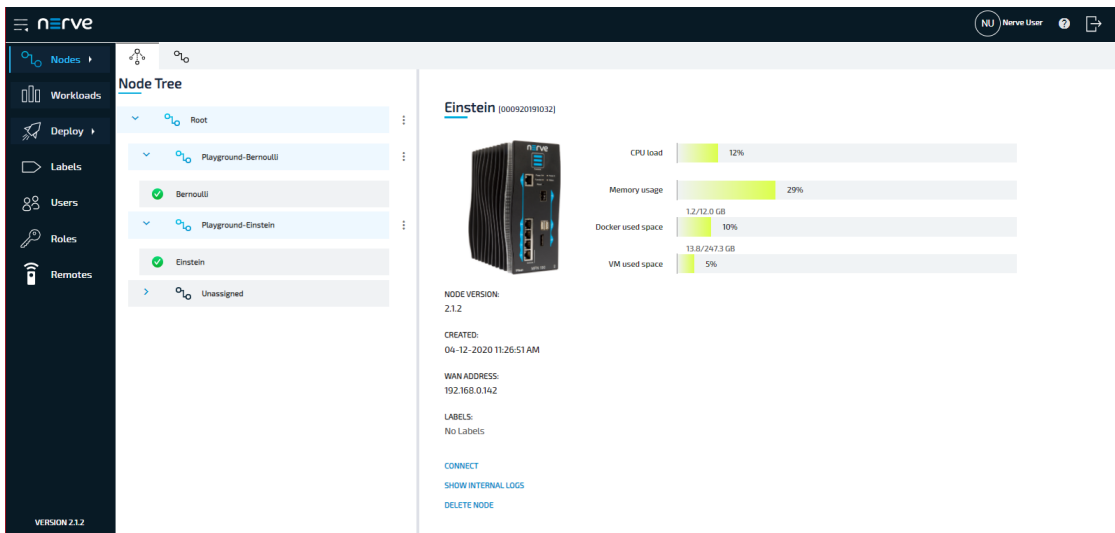
## Viewing nodes in the node tree

The node tree is the first visible page of the Management System after logging in. It presents a means of organization for nodes that are connected to the Management System. Nodes are embedded into elements of the node tree.

1. Log in to the Management System with your credentials.
2. Look for your node in the node tree. It is embedded in its own tree element.

### NOTE

The node and the tree element are named after a scientist. The name has been sent in an email.



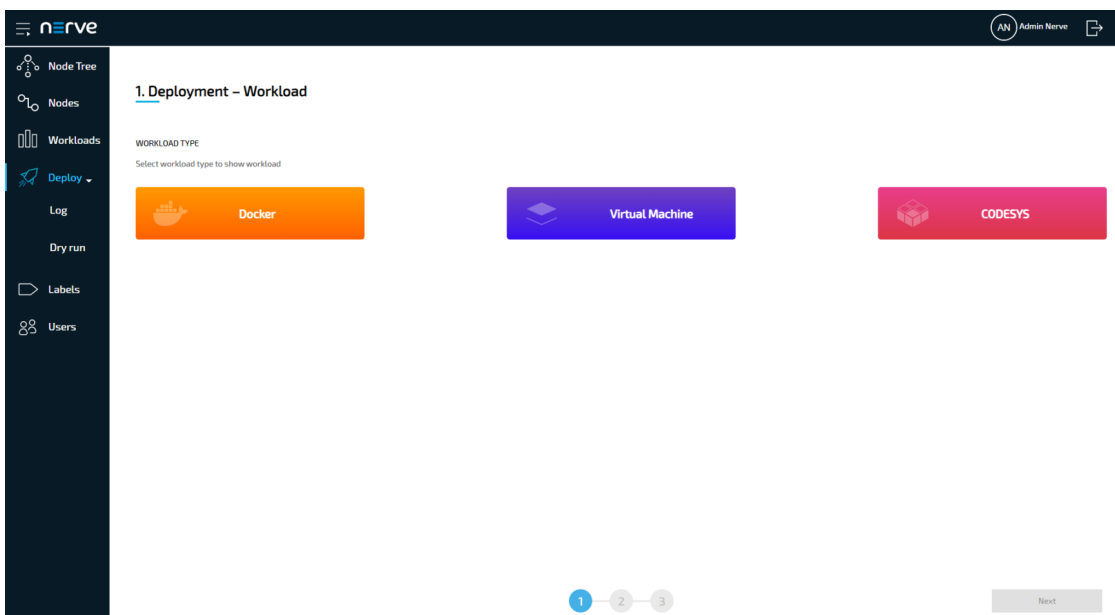
Be aware that other users may access the same evaluation system. Make sure to use the designated node that was mentioned in the instruction emails.

## Step 1: Deploying the machine simulation and Node-RED workloads

Deployment is the process of downloading workloads to Nerve Devices through the Nerve Management System. With Nerve you can deploy and manage Docker containers, virtual machines and CODESYS applications. The instructions below show how to deploy the machine simulation as a CODESYS workload and Node-RED as a Docker workload.

### Machine simulation

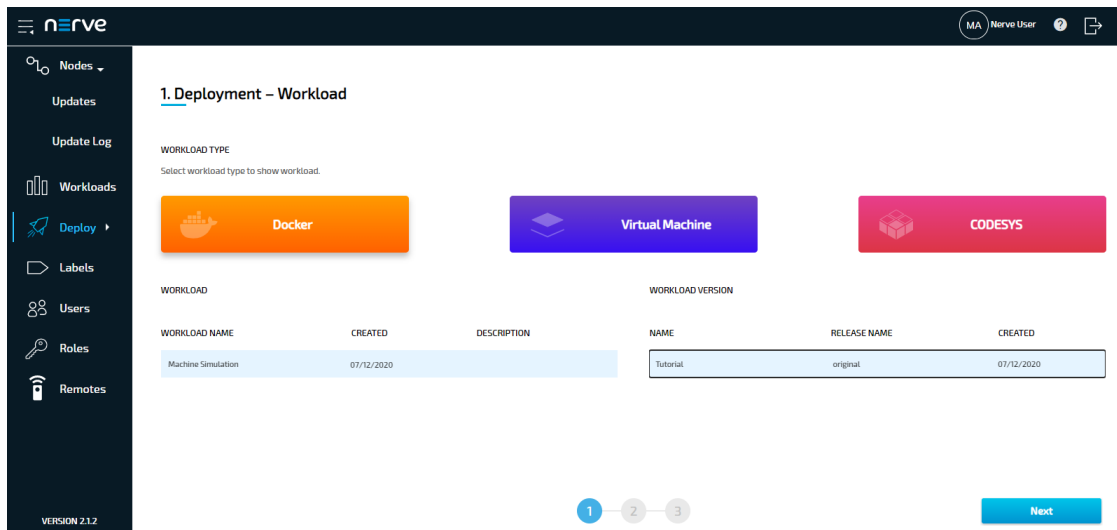
1. Log in to the Management System.
2. Select **Deploy** in the navigation on the left.



3. Select the red CODESYS tab. A list of CODESYS workloads will appear below.



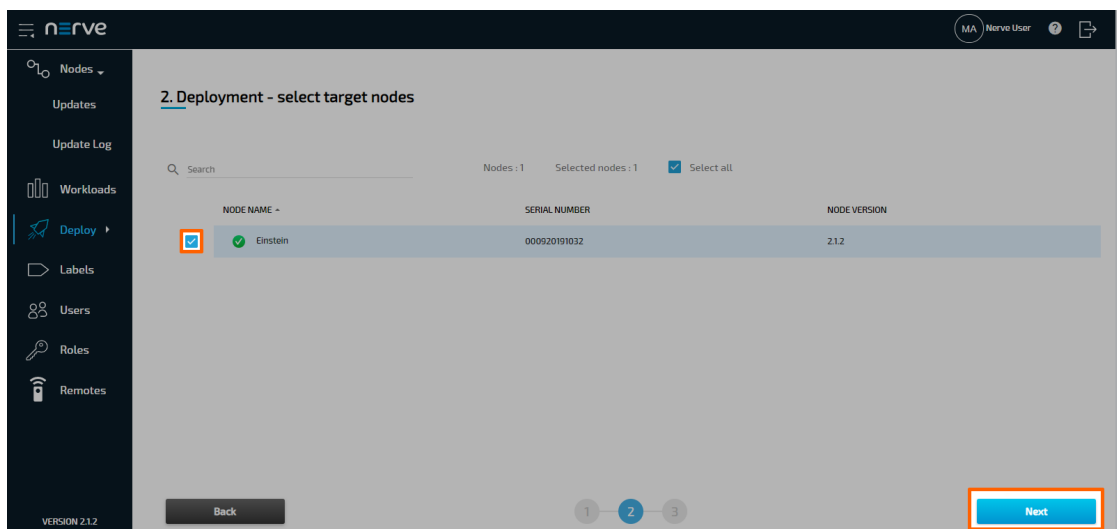
- Select **Machine Simulation** in the list of workloads. A list of version of this workload will appear to the right.



- Select the latest version of the workload on the right.

- Click **Next** in the lower-right corner.
- Tick the checkbox next to your node.

- Select **Next** in the lower-right corner.

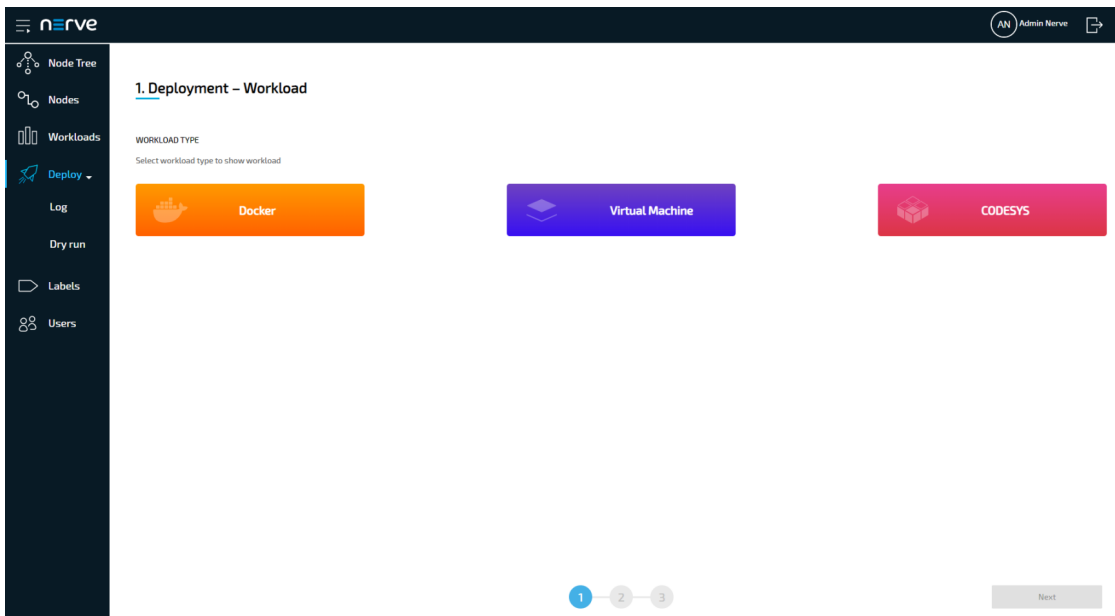


- Select **Deploy** to execute the deployment.  
*Optional: Enter a Deploy name above the Summary of the workload to make this deployment easy to identify. A timestamp is filled in automatically.*

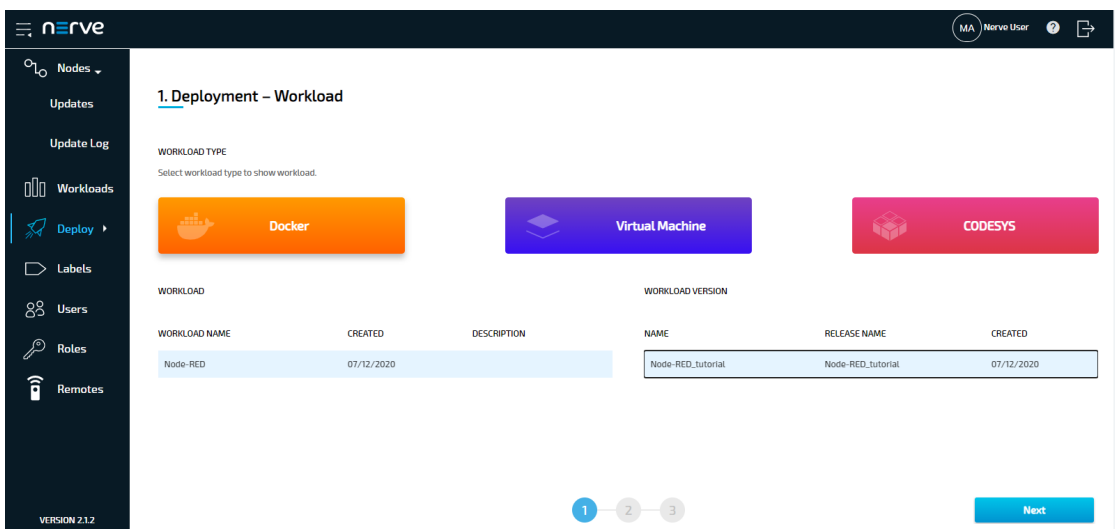
The deployment should now be visible at the top of the deployment log. Click the log entry of the deployment to see a more detailed view.

## Node-RED

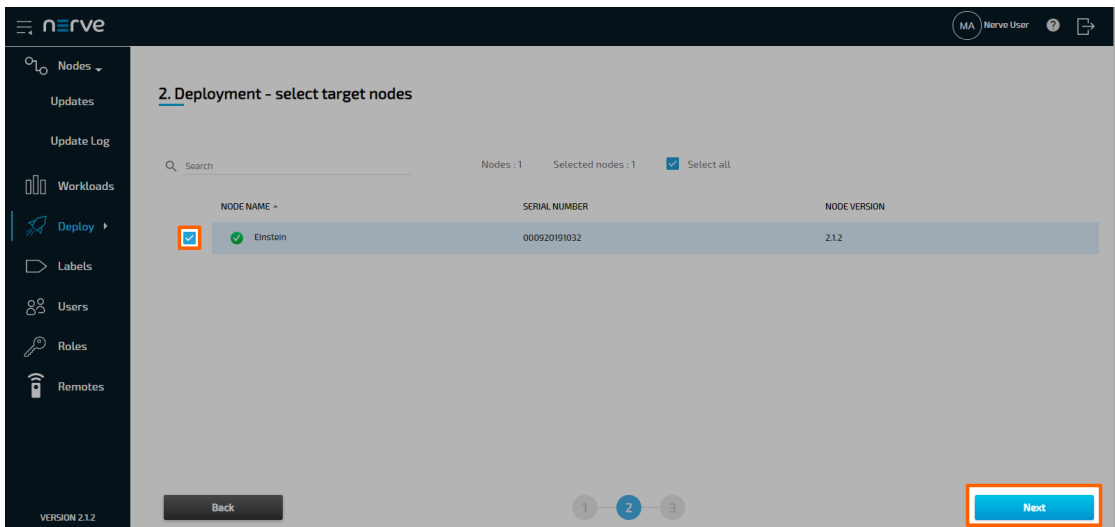
- Select **Deploy** in the navigation on the left.



2. Select the orange Docker tab. A list of Docker workloads will appear below.
3. Select **Node-RED** in the list of workloads. A list of version of this workload will appear to the right.
4. Select the latest version of the workload.



5. Select **Next** in the lower-right corner.
6. Tick the checkbox next to your node.
7. Select **Next** in the lower-right corner.



8. Select **Deploy** to execute the deployment.

*Optional: Enter a Deploy name above the Summary of the workload to make this deployment easy to identify. A timestamp is filled in automatically.*


The deployment should now be visible at the top of the deployment log. Click the log entry of the deployment to see a more detailed view.

To confirm if the workloads have been deployed successfully, select **Nodes** in the navigation on the left. Select your node in the node tree and confirm if two workload tiles are showing underneath the bar graph. The workloads should show the status **STARTED**.

#### NOTE


It can take a number of seconds until the status of workloads switches from **IDLE** to **STARTED** after deployment.


**Einstein** [000920191032]



CPU load 12%  
 Memory usage 33%  
 Docker used space 2.8/12.0 GB 23%  
 VM used space 13.8/247.3 GB 5%

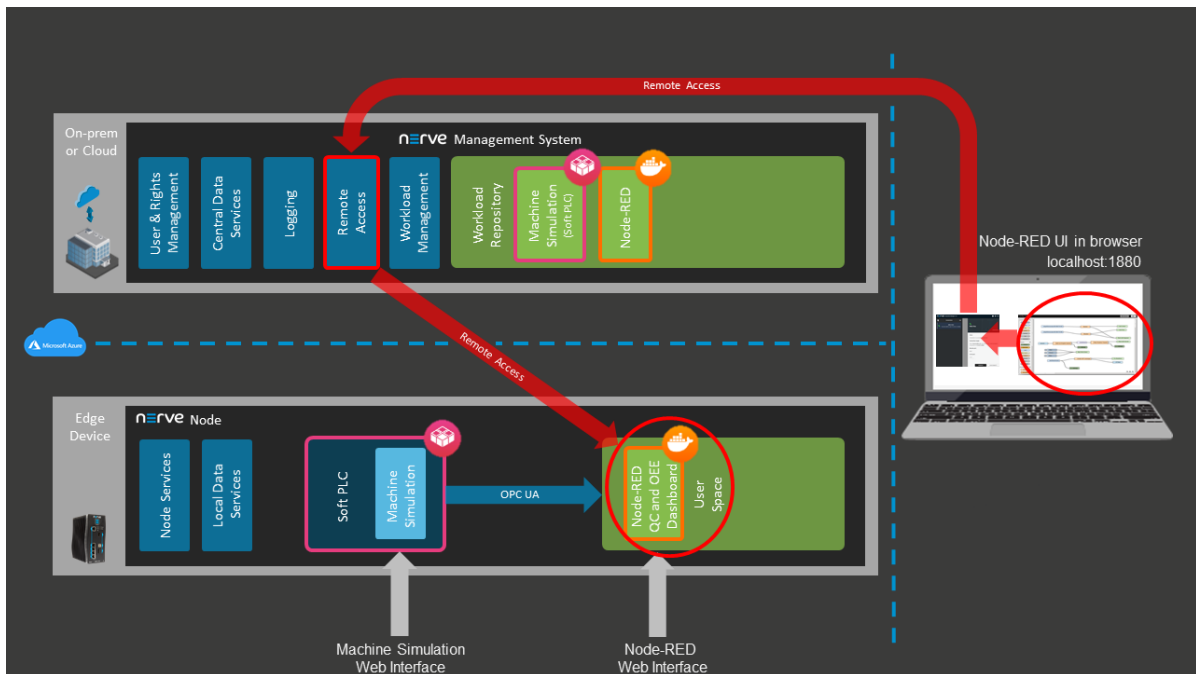
**NODE VERSION:**  
 2.1.2  
**CREATED:**  
 04-12-2020 11:26:51 AM  
**WAN ADDRESS:**  
 192.168.0.142  
**LABELS:**  
 No Labels  
[CONNECT](#)  
[SHOW INTERNAL LOGS](#)  
[DELETE NODE](#)

Machine Simulation  
  
 Status: STARTED

Node-RED  
  
 Status: STARTED

## Step 2: Connecting to the workloads through remote access

To access the deployed workloads, it is required to create a connection between the computer used for this tutorial and the Management System, and from the Management System to the web servers of the workloads on the Nerve Device. This is done through the remote access feature, specifically through the use of remote tunnels. The remote tunnel connection has already been configured in the Management System and is ready for use. The image below illustrates the remote access feature, showing a connection to the deployed Node-RED workload on the Nerve Device.



Before continuing, install the Nerve Connection Manager that was part of the delivery, as it is required for using remote tunnels.

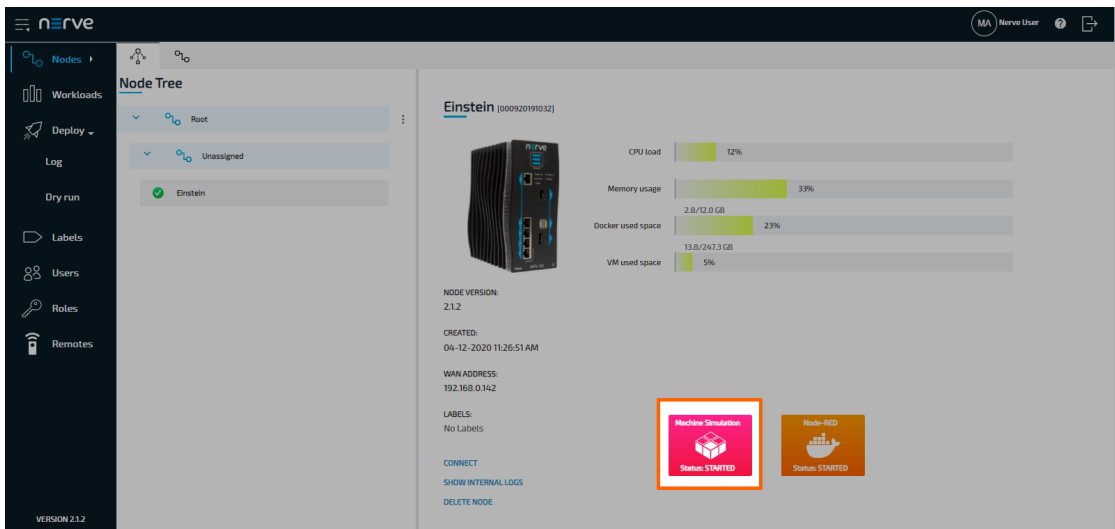
#### NOTE

Local admin rights might be required to successfully install the Nerve Connection Manager.

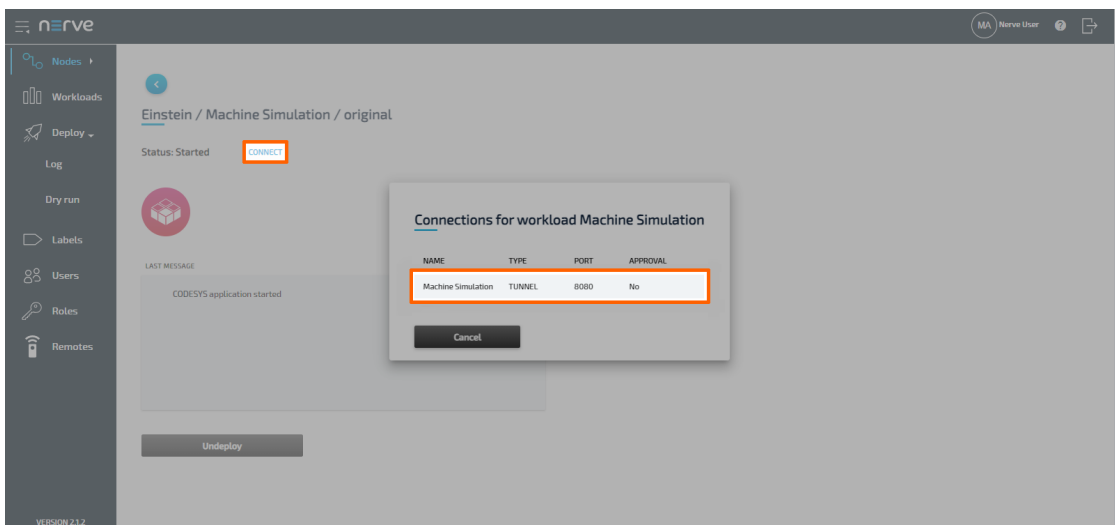
### Connecting to the machine simulation

The machine simulation is accessible at port 8080 on the Nerve Device through a web user interface. The preconfigured remote tunnel will use this port and port 8080 on the local computer to create a connection between the computer and the Nerve Device to access the machine simulation.

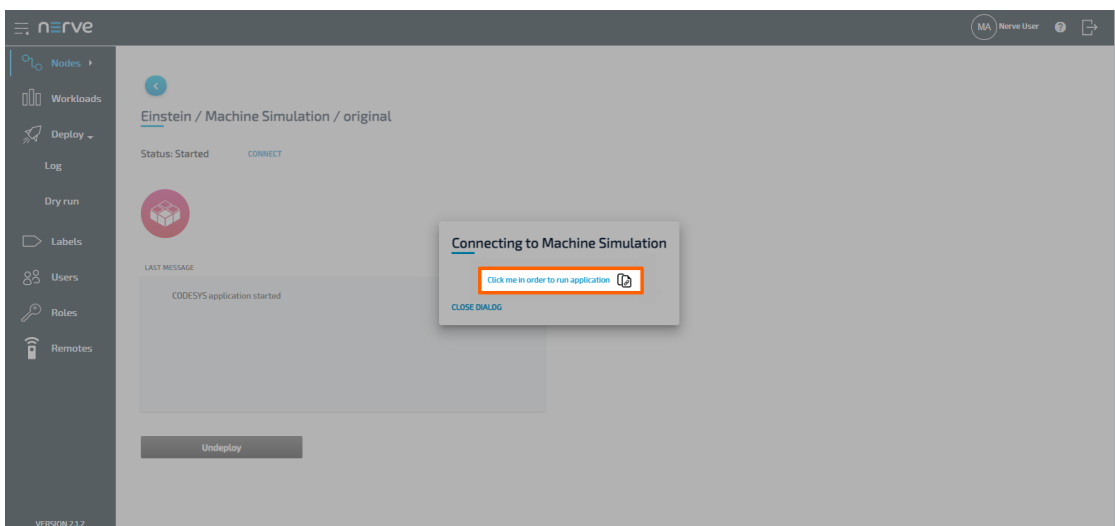
1. Select **Nodes** in the navigation on the left.
2. Select your node in the node tree.
3. Select the Machine Simulation CODESYS workload.



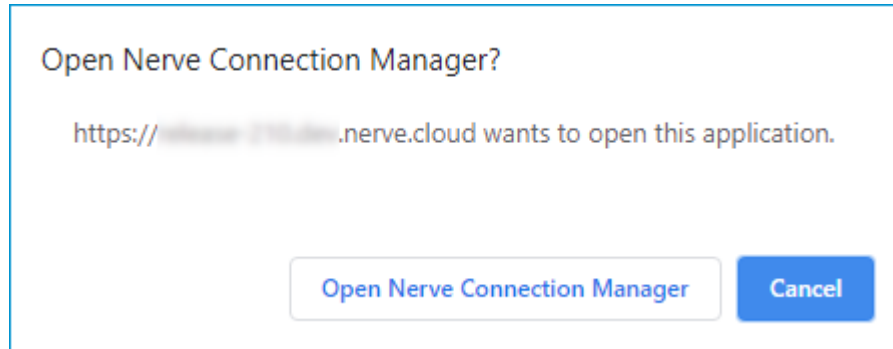
4. Select **CONNECT** next to the workload status. Available connections will appear in a window.
5. Select the **Machine Simulation** remote tunnel from the list.



6. Select **Click me in order to run application** in the new window.



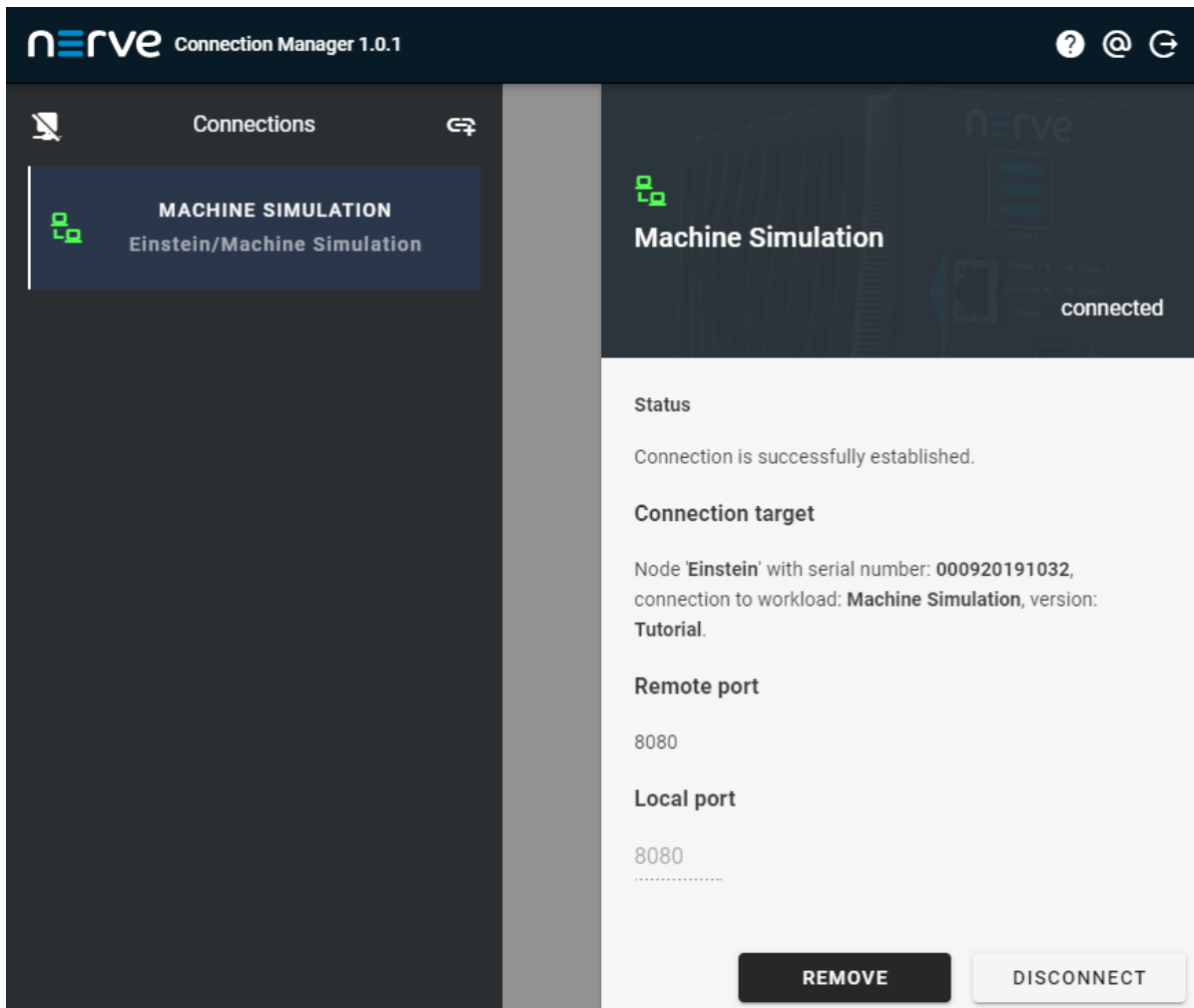
- If the Nerve Connection Manager installed correctly, confirm the browser message
7. that the Nerve Connection Manager shall be opened.  
Depending on the browser that is used, this message will differ. The Nerve Connection Manager will start automatically once the message is confirmed.



**NOTE**

If the Nerve Connection Manager does not start automatically, refer to [Using a remote tunnel to a node or external device](#) in the user guide.

The remote tunnel will be established once the Nerve Connection Manager starts and the remote connection will turn green in the Nerve Connection Manager. The remote tunnel is now ready to be used.



Open a new browser tab and enter localhost:8080 to open the machine simulation dashboard.



## Machine Simulation

The Machine Simulation interface is divided into several sections:

- Status:** Milling, Current Part: 1900648
- Spindle power:** A horizontal bar chart showing power levels from 0.0 to 10000.0.
- Machine Controls:** Includes buttons for Run (green), Pause, Error, Run/Pause, and Acknowledge Error.
- Error and Wear Simulation:** Includes a Wear Level slider and an Inject Error button. Text: "Drag the wear level to increase simulated tool wear. Wear level increases automatically after each produced part. A high wear level causes high deviations in the part dimensions and may cause errors during milling."
- Quality Check Station:** Measurement Results table:

Measurement	Value
Part No.	1900647
length	100.130066
diameter	22.067999
hole diameter	12.139433

## Connecting to Node-RED

Node-RED is accessible at port 1880 on the Nerve Device through a web user interface. The preconfigured remote tunnel will use this port and port 1880 on the local computer to create a connection between the computer and the Nerve Device to access the interface of Node-RED.

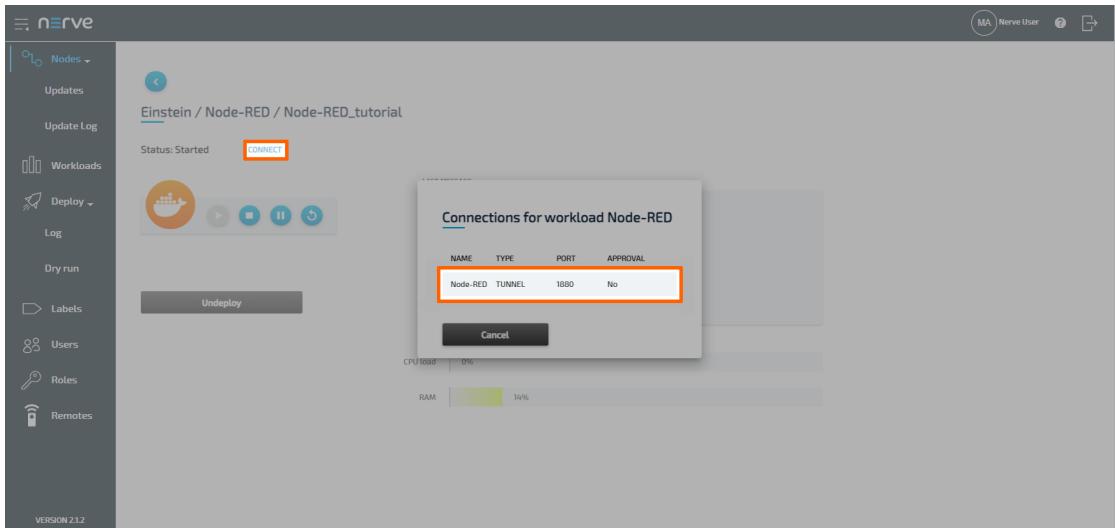
1. Select **Nodes** in the navigation on the left.
2. Select your node in the node tree.
3. Select the Node-RED Docker workload.

The screenshot shows the Nerve web interface with the following details:

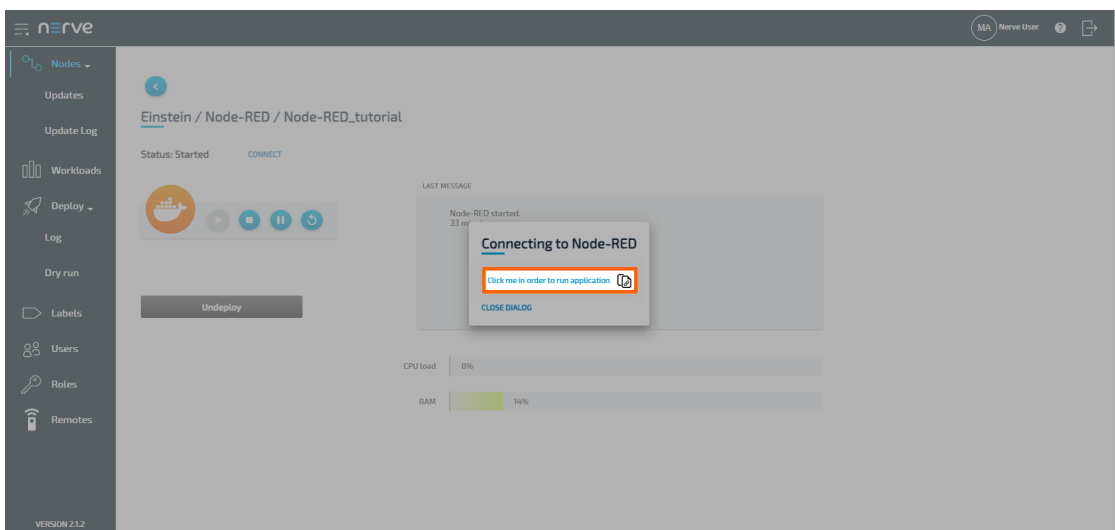
- Navigation:** Nodes, Workloads, Deploy, Log, Dry run, Labels, Users, Roles, Remotes.
- Node Tree:** Root > Unassigned > Einstein (checked).
- Einstein (000920191032) Details:**
  - CPU load: 12%
  - Memory usage: 33%
  - Docker used space: 2.8/12.0 GB (23%)
  - VM used space: 13.8/247.3 GB (5%)
  - NODE VERSION: 2.1.2
  - CREATED: 04-12-2020 11:26:51 AM
  - WAN ADDRESS: 192.168.0.142
  - LABELS: No Labels
  - CONNECT, SHOW INTERNAL LOGS, DELETE NODE buttons are visible.
- Workload Status:** Machine Simulation (Status: STARTED) and Node-RED (Status: STARTED) buttons are shown.

4. Select **CONNECT** next to the workload status. Available connections will appear in a window.

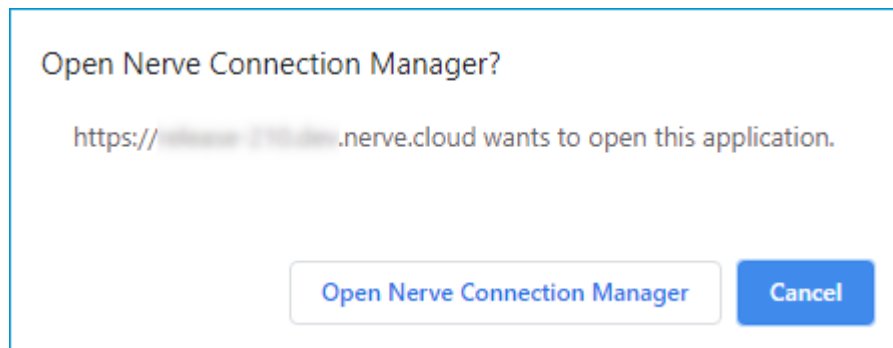
5. Select the **Node-RED** remote tunnel from the list.



6. Select **Click me in order to run application** in the new window.



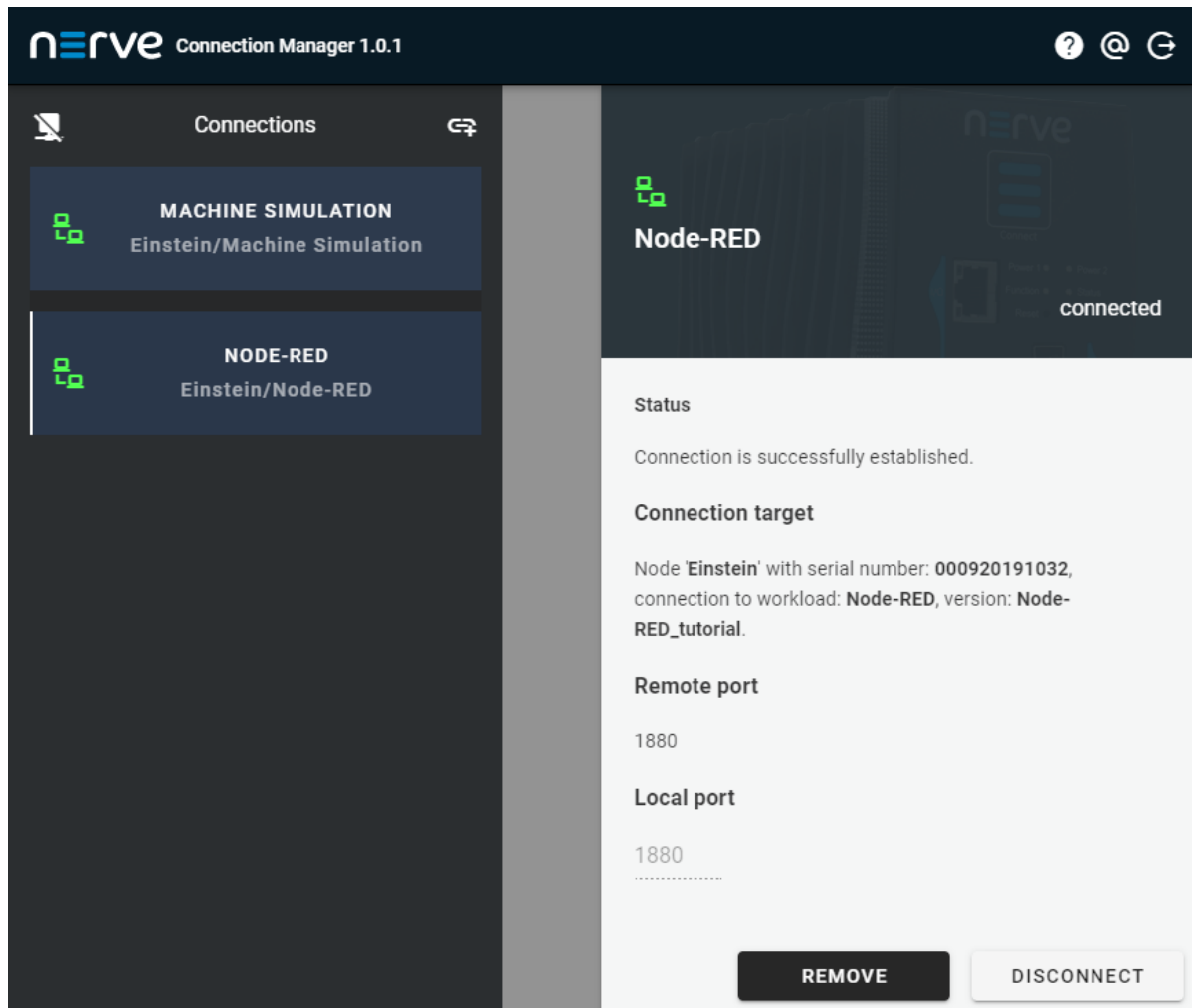
7. If the Nerve Connection Manager installed correctly, confirm the browser message that the Nerve Connection Manager shall be opened. Depending on the browser that is used, this message will differ. The Nerve Connection Manager will start automatically once the message is confirmed.



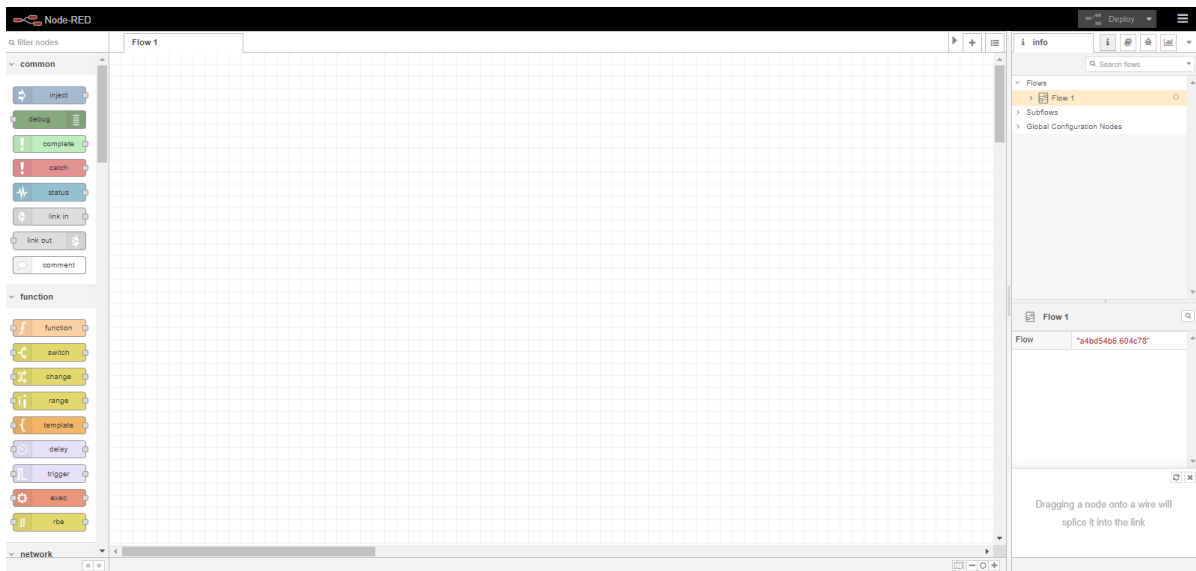
**NOTE**

If the Nerve Connection Manager does not start automatically, refer to [Using a remote tunnel to a workload](#) in the user guide.

The remote tunnel will be established once the Nerve Connection Manager starts and the remote connection will turn green in the Nerve Connection Manager. The remote tunnel is now ready to be used.



Open a new browser tab and enter localhost:1880 to open Node-RED.



### Step 3: Setting up Node-RED

Node-RED is running on the node and the workstation is connected to it through a remote tunnel. Switch to the browser tab that has Node-RED opened at localhost: 1880. The flow is imported in the instructions below.

**NOTE**

Download the JSON file that has been part of the delivery and remember its download location.

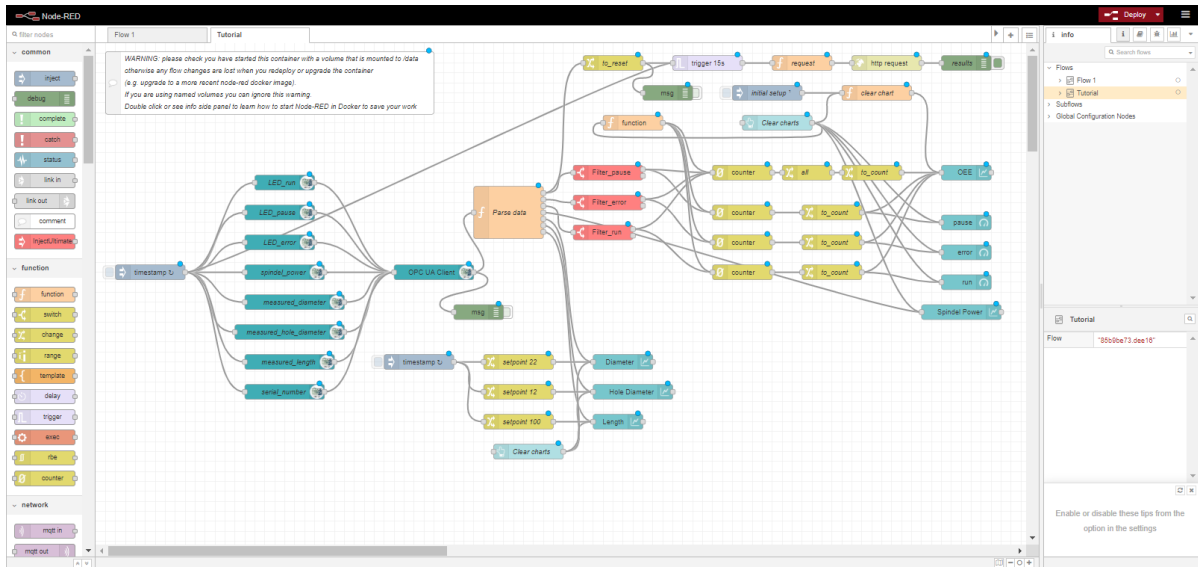
1. Select the burger menu in the top-right corner.
2. Select **Import**.



3. Select **select a file to import**.
4. Look for the JSON file that has been downloaded before.
5. Select **Open**.

6. Select **Import** in the lower-right corner.

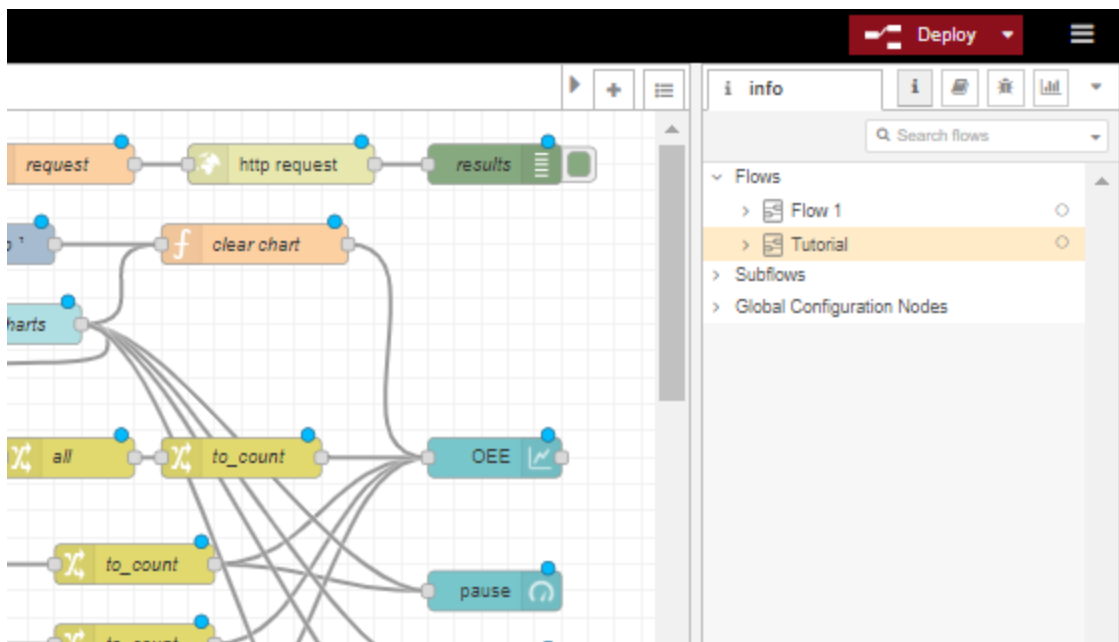
The flow will appear in a new tab inside of Node-RED labelled **Tutorial** in the top-left corner.



## Step 4: Accessing the OEE and quality dashboard

After the Node-RED setup, data generated by the machine simulation can be viewed on the OEE and quality dashboard.

1. In the Node-RED browser tab select **Deploy** in the top-right corner to start the flow. Make sure that the **Tutorial** tab is active.



2. Open a new browser tab.

3. Enter localhost:1880/ui to open the OEE dashboard.

## NOTE

It may take a moment for the dashboard to load.

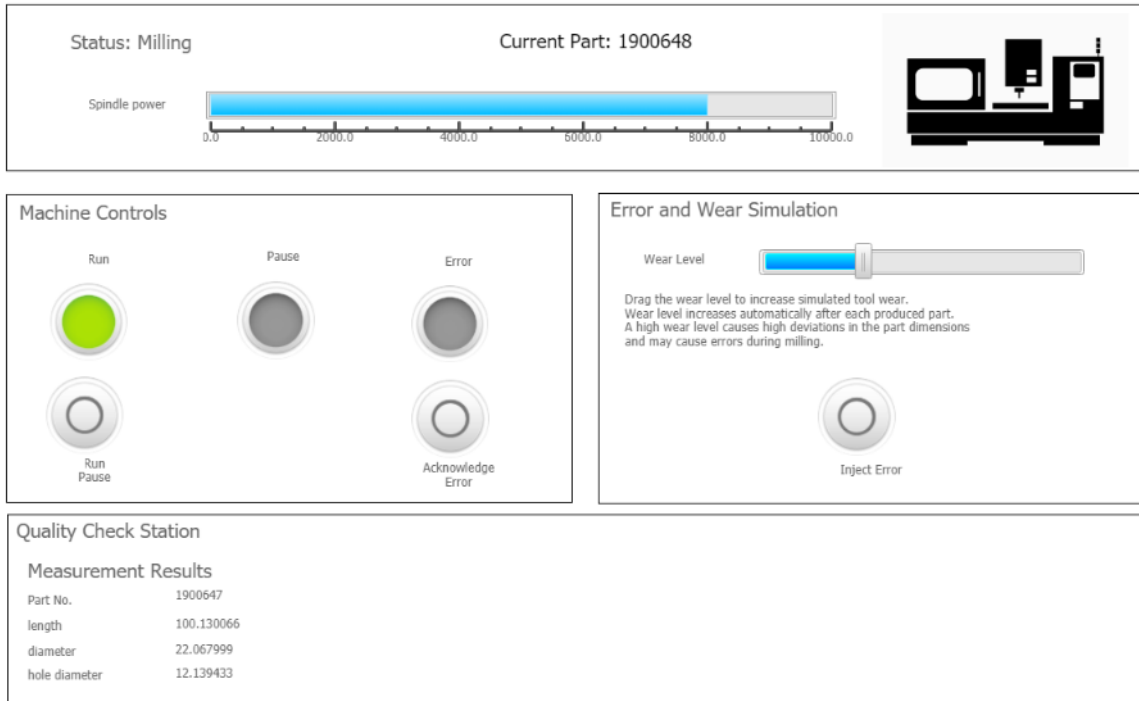


## Experimenting with the machine simulation

The machine simulation runs through a process of part production. The production of a part begins with the machine waiting for a part in order to load it. Once the part is loaded in, the machine starts milling and the spindle power goes up. After the machine is finished milling and the spindle power goes down again, the part is then unloaded and sent to a measuring station. The **Measurement Results** in the **Quality Check Station** are updated as soon as measuring is completed. The operation of the machine simulation and the measurements of the parts are impacted by the adjustable **Wear Level**.

The user interface of the machine simulation is split up into different parts.

## Machine Simulation



Item	Description
------	-------------

This shows the current status of the machine simulation, the number of the current part and the spindle power when the machine is milling. The status of the machine cycles through the following way in regular operation:

**Status**

- **Waiting for Part**
- **Loading**
- **Milling**
- **Unloading**
- **Measuring**

If the machine operation is stopped by an error or a pause, the status will display **Error** or **Pause** and the operation at which the pause or error occurred in parentheses, e.g **Error (waiting for part)**.

Item	Description
<b>Machine Controls</b>	<p>The machine controls consist of three colored lights and two buttons.</p> <ul style="list-style-type: none"> <li>• <b>Run</b> This light is green if the machine is running as intended.</li> <li>• <b>Pause</b> This light is yellow if the machine has been paused by the user.</li> <li>• <b>Error</b> This light is red if the machine is in an error state, either caused by a high wear level or by injecting an error manually.</li> <li>• <b>Run/Pause</b> Use this button to pause and unpause the machine simulation.</li> <li>• <b>Acknowledge Error</b> If the machine is in an error state, this button must be pressed to acknowledge the error and have the machine resume its operation.</li> </ul>
<b>Error and Wear Simulation</b>	<p>The slider here emulates the wear of the system. It increases gradually when the machine is running. The higher the wear level, the higher the probability of an error occurring in the simulation. The slider can be adjusted to preference to simulate different levels of wear. Manually inject an error by pressing the <b>Inject Error</b> button.</p>
<b>Quality Check Station</b>	<p>After the completion of a part, the <b>Measurement Results</b> are updated, showing new values for the <b>Part No.</b>, <b>length</b>, <b>diameter</b> and <b>hole diameter</b> of a part. The higher the wear level, the higher are the deviations of the part measurements.</p>

The data of the machine operation is amassed and displayed in an OEE dashboard by Node-RED.

## Node-RED and the OEE dashboard

The data flow in Node-RED reads from an OPC UA Client and parses the data into the OEE statistics and quality check data. The dashboard consists of three parts:

- **OEE**  
This part of the dashboard shows the amount of the time the machine has spent in **run**, **pause** and **error** states in two different visualizations.
- **Spindle Power**  
This is a graph of how high spindle power is over time.
- **Quality Control**  
**Length**, **Diameter** and **Hole Diameter** are displayed in graphs, showing the measurements of the parts produced. Deviations caused by a high wear level can be analyzed here.





This concludes the tutorial for Nerve. For questions, requests and further evaluation, contact [trynerve@tttech-industrial.com](mailto:trynerve@tttech-industrial.com).

## User Guide

## User Guide

The user guide for Nerve covers the features and configurations a user has with Nerve using the two user interfaces of Nerve: the Local UI and the Management System. It is supported by the device guide that contains device specific information required for working with Nerve and the instructions in the user guide.

If there are any questions about the software installed and the features provided, contact a sales representative or TTech Industrial customer support at [support@tttech-industrial.com](mailto:support@tttech-industrial.com).

## Recommended chapters after node registration

Once a node has been registered following the [Getting started](#) chapter, the following topics might be of interest:

- [Management System overview](#)
- [Adding a user](#)
- [Adding a new role](#)
- [Provisioning a CODESYS workload](#)
- [Provisioning a Docker workload](#)
- [Provisioning a Virtual Machine workload](#)
- [Deploying a workload](#)
- [First steps with CODESYS programming](#)
- [Nerve Data Services introduction](#)
- [Node internal networking](#)

# Hardware specifics

The user guide focuses on how to operate the Nerve software. As such it will not contain any device specific information. Whenever device specific information is required, the user guide will link to the [device guide](#).

Refer to the chapter of the Nerve Device in the device guide in order to setup and install the hardware, and find out device specific information:



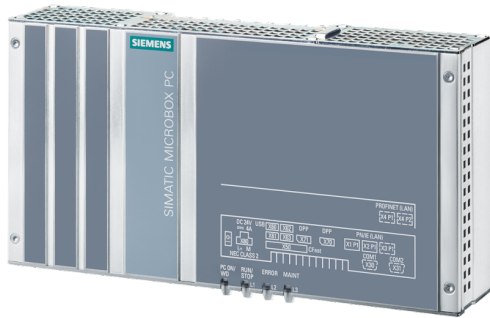
MFN 100

Kontron Kbox A-150-APL



Kontron Kbox A-250

Mxtang AXWL10-8665U



Siemens SIMATIC IPC427E



Siemens SIMATIC IPC127E



Supermicro SuperServer E100-9AP-IA



Supermicro SuperServer 1019D-16C-FHN13TP



Supermicro SuperServer 5029C-T



Vecow SPC-5600-i5-8500



Winmate EACIL20

## License activation

### NOTE

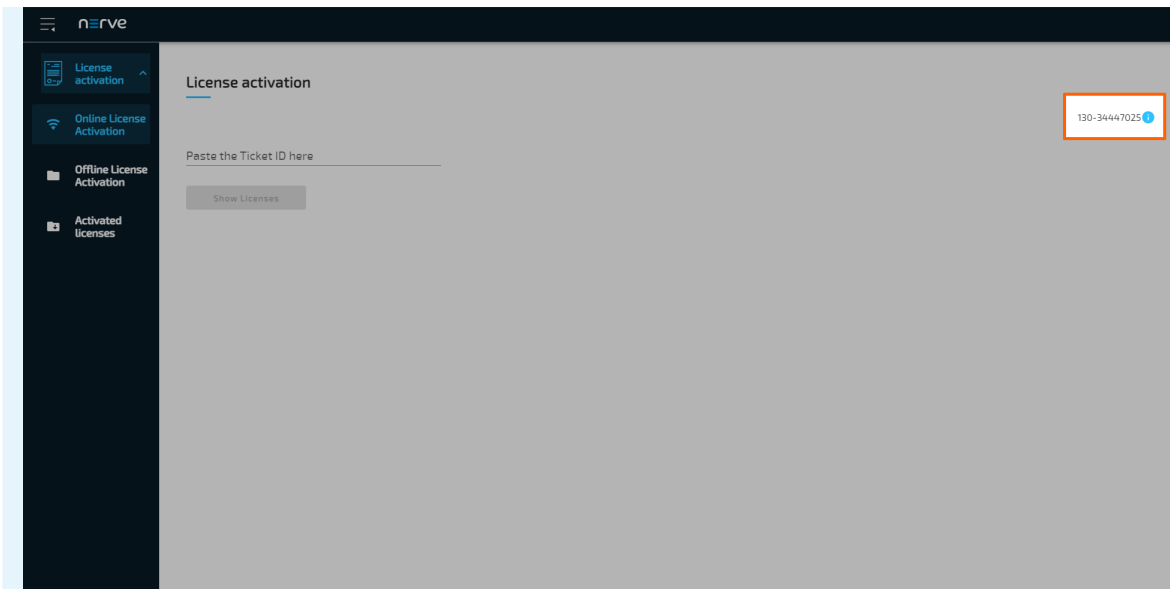
The instructions below are also valid for the activation of trial licenses, as the process is the same. However, note that the CODESYS runtime in the RTVM is running in demo mode in the trial version, meaning that the CODESYS runtime shuts off after 30 minutes and needs to be restarted again. The trial license is valid for 30 days.

The license for using Nerve has to be activated on the node before the product can be used. A ticket ID that is required for the activation of the license has been sent as part of the delivery, along with a link to a web depot. With this ticket ID and the web depot link, the product can be activated in two ways:

- **Online activation**  
The node must have access to the internet.
- **Offline activation**  
The node does not have access to the internet. Note that offline activation requires a workstation with a connection to the internet.

### NOTE

Note down the license serial number on the right side. It is required to reactivate a license that has been used before.



Refer to [Reusing activated licenses](#) below for more information on how to activate the product by reusing previously activated licenses.

Make sure to connect a workstation to the physical port of the Nerve Device associated with host access and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the host access interface with a 255.255.255.0 subnet mask. Refer to the [device guide](#) for more information per Nerve Device.

After connecting a workstation to the Nerve Device, follow the link to reach the UI for activating the license. This link is Nerve Device specific and is listed per Nerve Device in the table below:

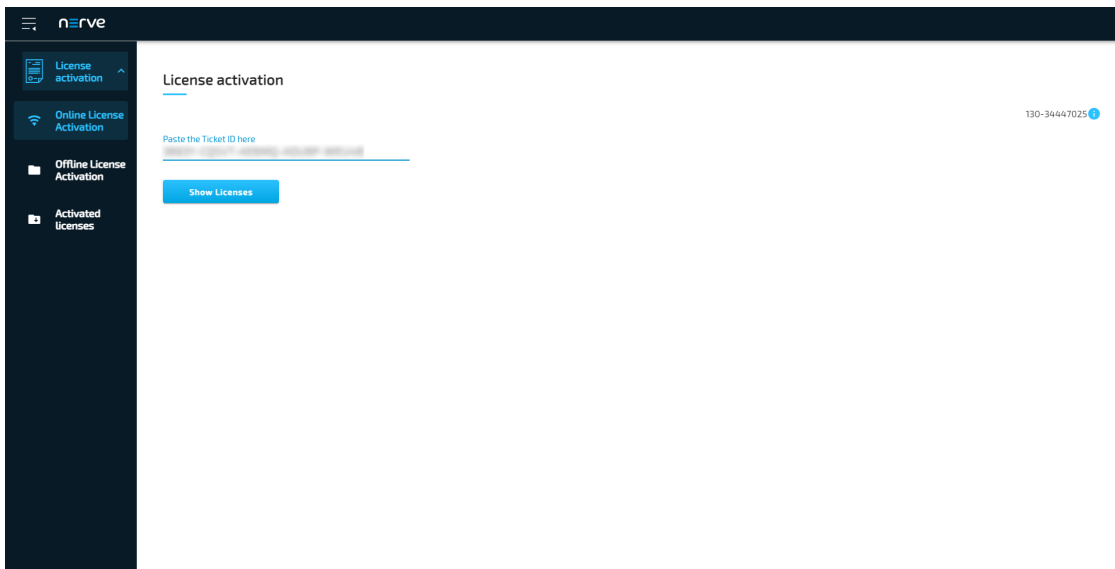
Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	<p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.</p> <p>&lt;wanip&gt;:3333</p>
<b>Kontron KBox A-250</b>	ETH 2	<p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.</p> <p>&lt;wanip&gt;:3333</p>
<b>Maxtang AXWL10</b>	LAN1	<p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.</p> <p>&lt;wanip&gt;:3333</p>

Nerve Device	Physical port	Local UI
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

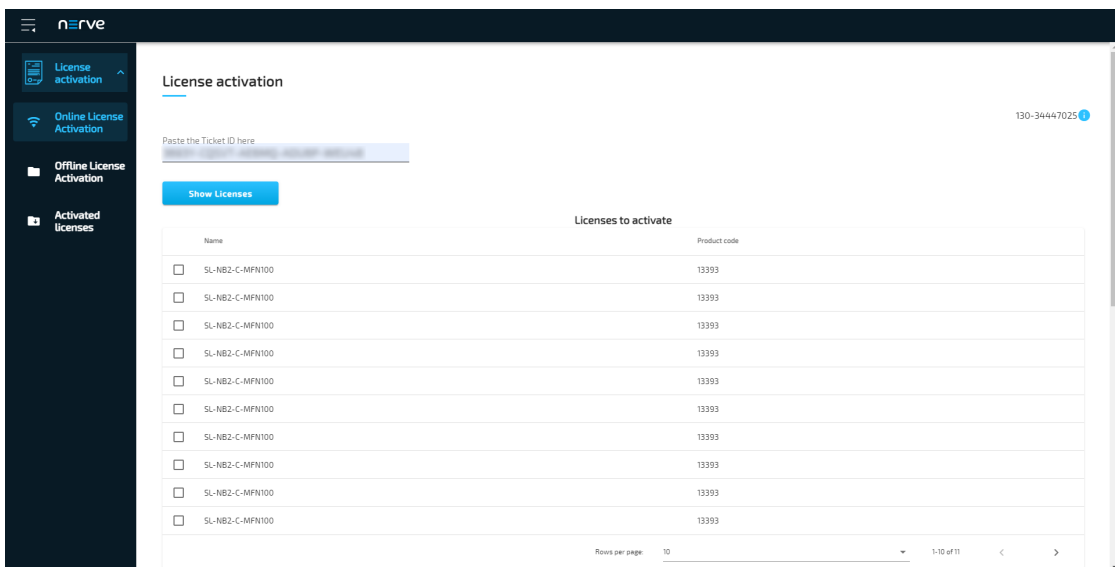
## Online license activation

If the node has internet access, the node will automatically connect to the licensing server. Online license activation is selected in the navigation on the left by default.

1. Enter the ticket ID under **Paste the Ticket ID here**.

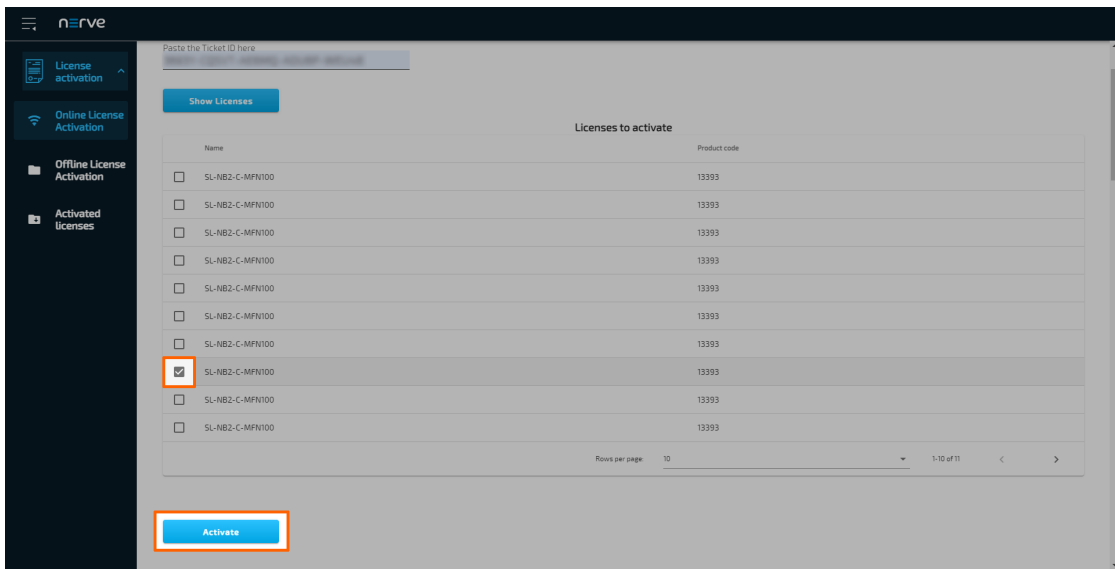


2. Select **Show Licenses**. Available licenses will appear below.

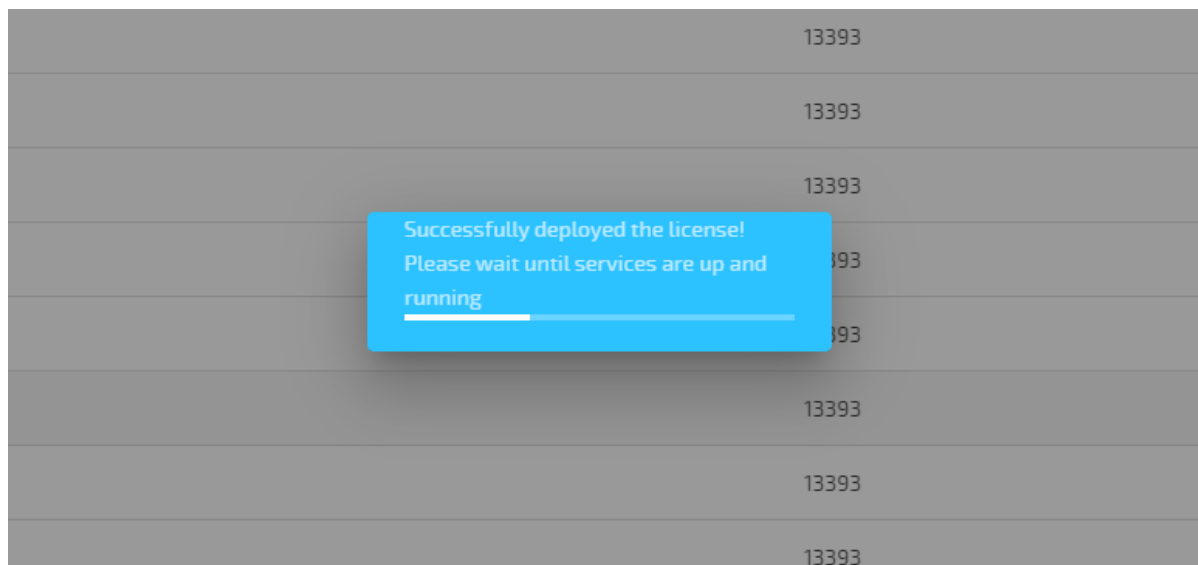


3. Tick the checkbox next to a license. The list is automatically filtered to only show licenses for the Nerve Device currently used.

4. Select **Activate** below the list of licenses.



The system will proceed to activate the license and automatically redirect to the Local UI after a successful activation.

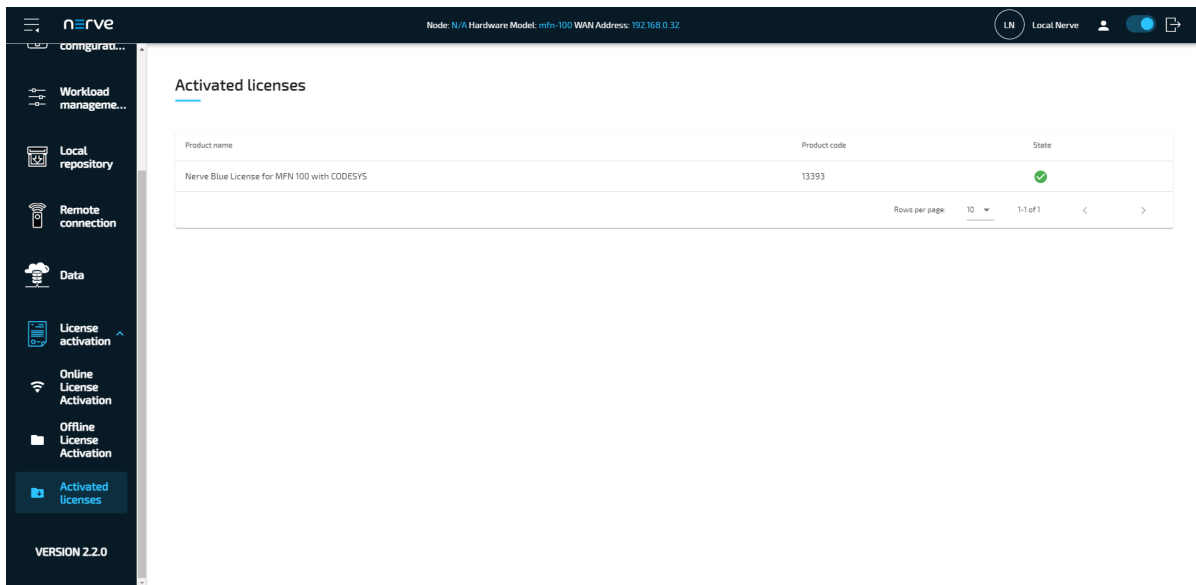


#### NOTE

When using an MFN 100, the device will light up blue once the license has been activated and the necessary services are up and running.

To double-check if the license has been activated properly, expand **License activation > Activated licenses** in the navigation on the left of the Local UI. An activated license is displayed in the table with a green check mark on the right side, signifying a successful activation on the correct Nerve Device.



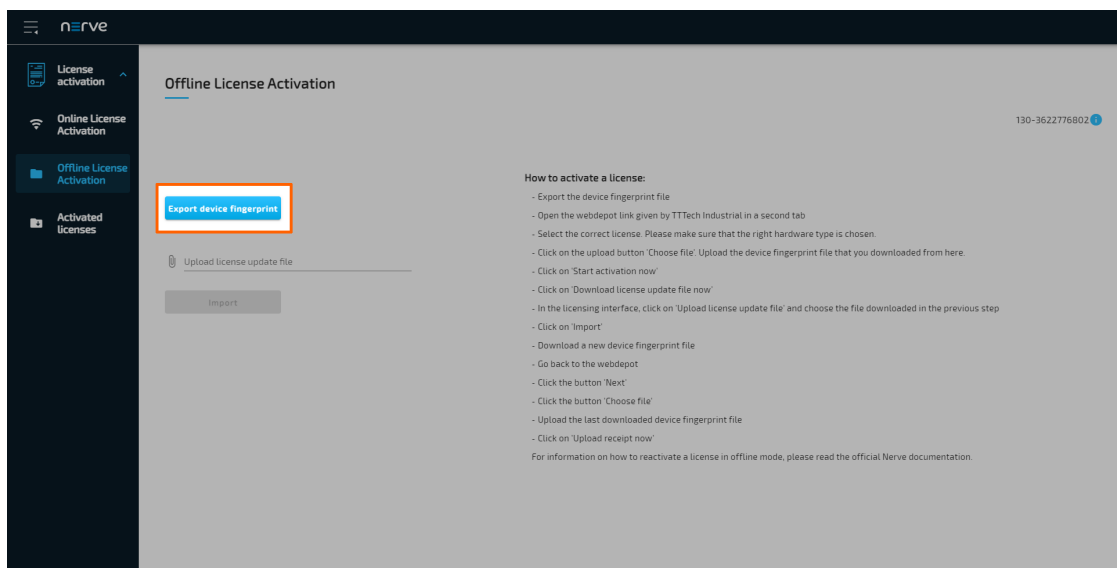


With the license activated, return to the previous device guide page and continue with **Accessing the Local UI and registering the device.**

## Offline license activation

In case of the node not having internet access, the license can be activated with a file-based method. However, note that a workstation with an internet connection is required for connecting to the licensing server in order to upload and download files.

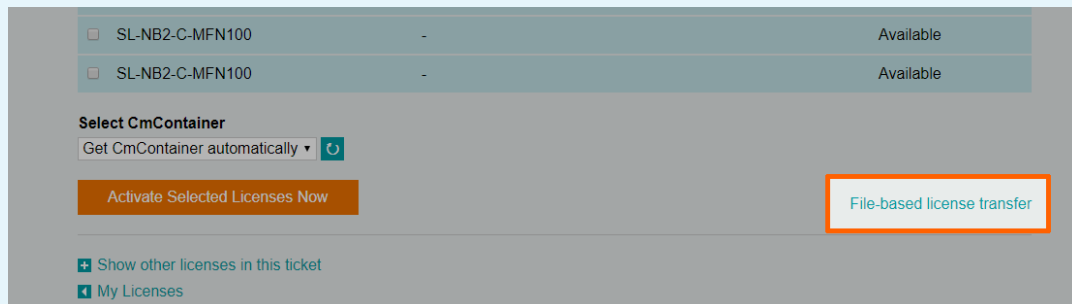
1. Select **License activation > Offline License Activation** in the navigation on the left.
2. Select **Export device fingerprint** to download the license update file.



3. Open a new browser tab.
4. Enter the link to the web depot. The link has been sent as part of the delivery.

### NOTE

If other applications using licenses from Wibu Systems (e.g. CODESYS) are present on the workstation, the web depot will suggest activating licenses in a different way. In order to properly activate Nerve following the instructions here, select **File-based license transfer** at the bottom of the page.



5. Select a license for the appropriate Nerve Device. The license will have the device name in its name.

#### NOTE

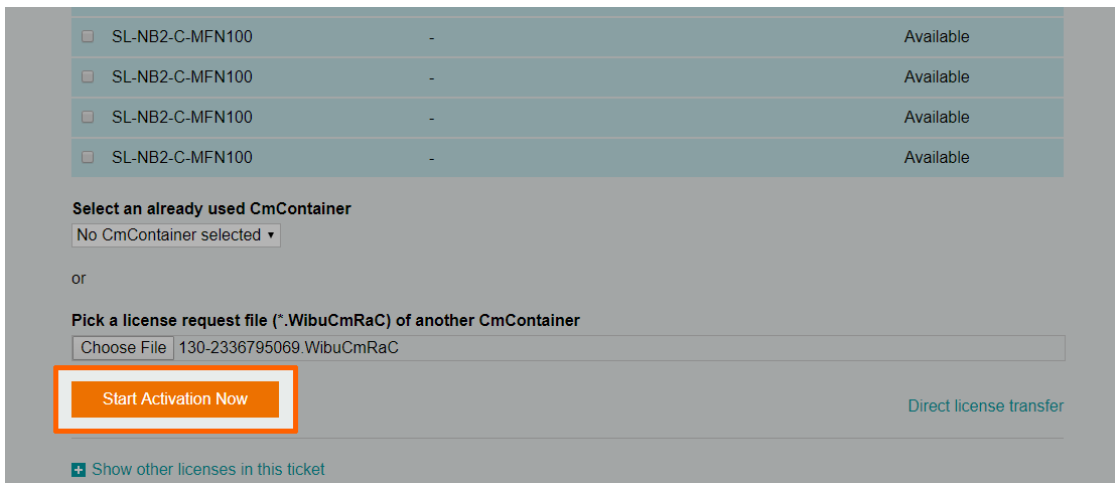
Make sure to always select a single license per device. Selecting multiple licenses when activating one device will waste purchased licenses.

6. Select **Choose File** below the list of licenses under **Pick a license request file (\*.WibuCmRaC) of another CmContainer** to open the file browser.

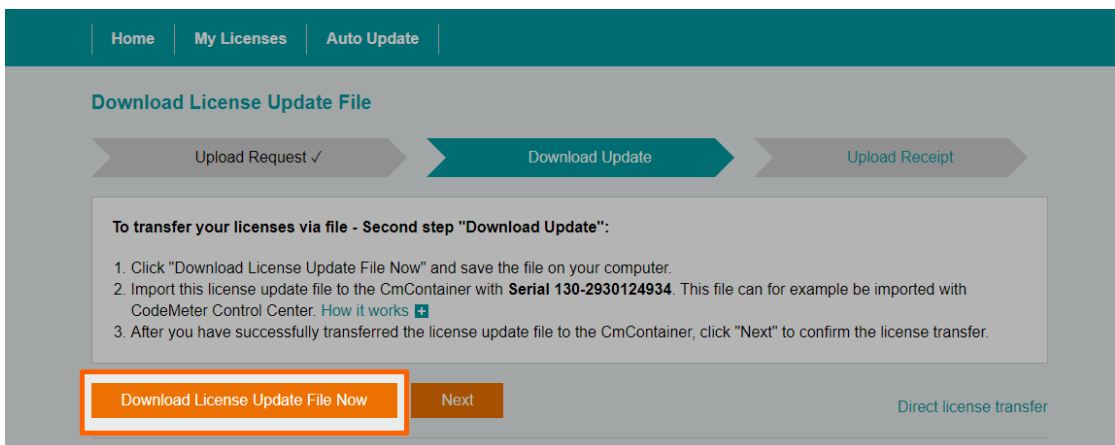


7. Upload the device fingerprint file that was download before.

8. Select **Start Activation Now**.

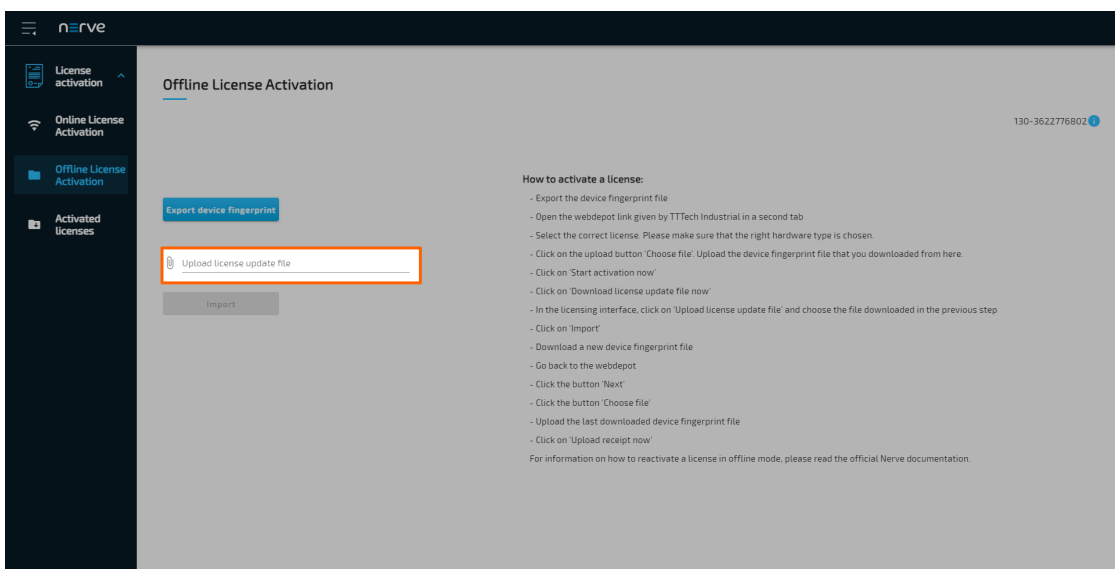


9. Select **Download License Update File now** in the next window to download the license update file named <licensenumbe>.WibuCmRaU.



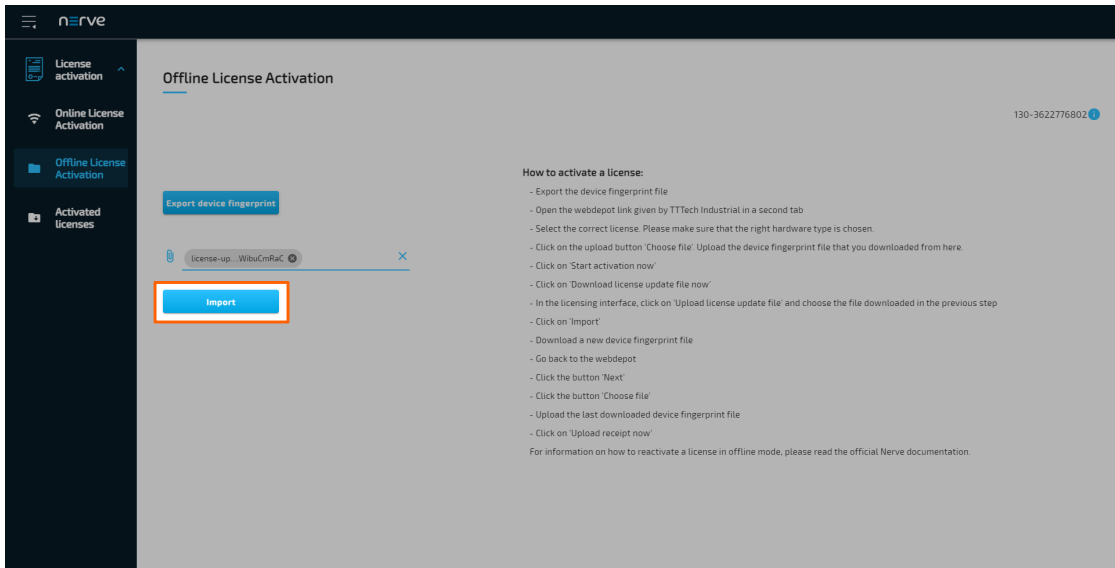
10. Switch back to the Local UI.

11. Select the **Upload license update file** field to open the file browser.

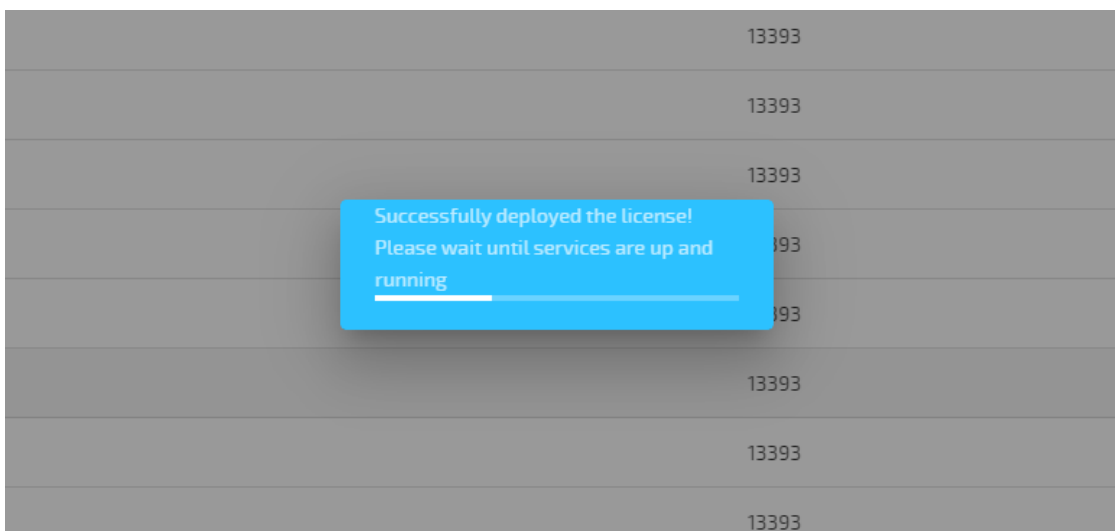


12. Navigate to where the <licensenumbe>.WibuCmRaU file is saved and select it.

13. Select **Import** to import the license.



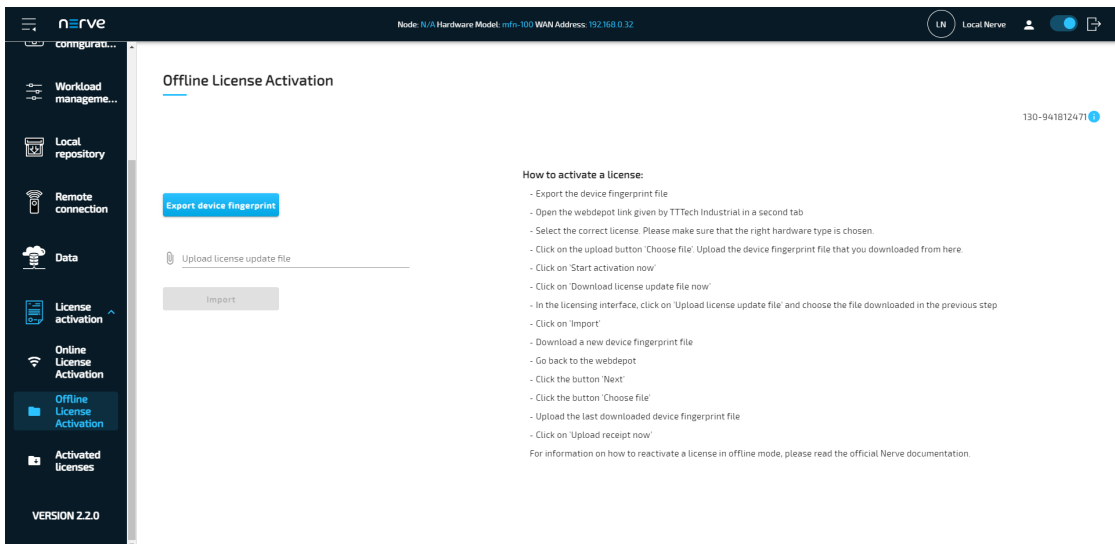
The system will proceed to activate the license and automatically redirect to the Local UI after a successful activation.



#### NOTE

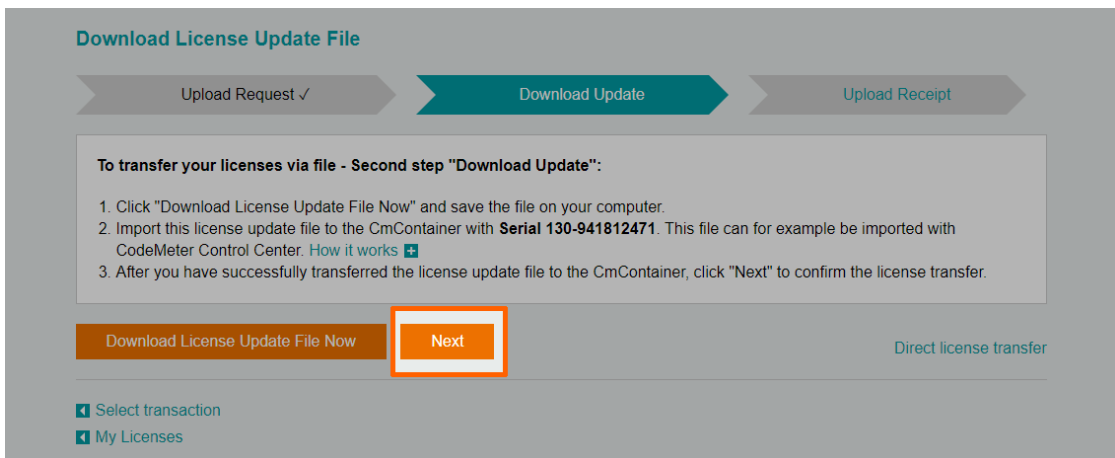
When using an MFN 100, the device will light up blue once the license has been activated and the necessary services are up and running.

14. Log in to the Local UI with the credentials from the customer profile.
15. Select **License activation > Offline License Activation** in the navigation on the left.
16. Select **Export device fingerprint** to download the updated device fingerprint.

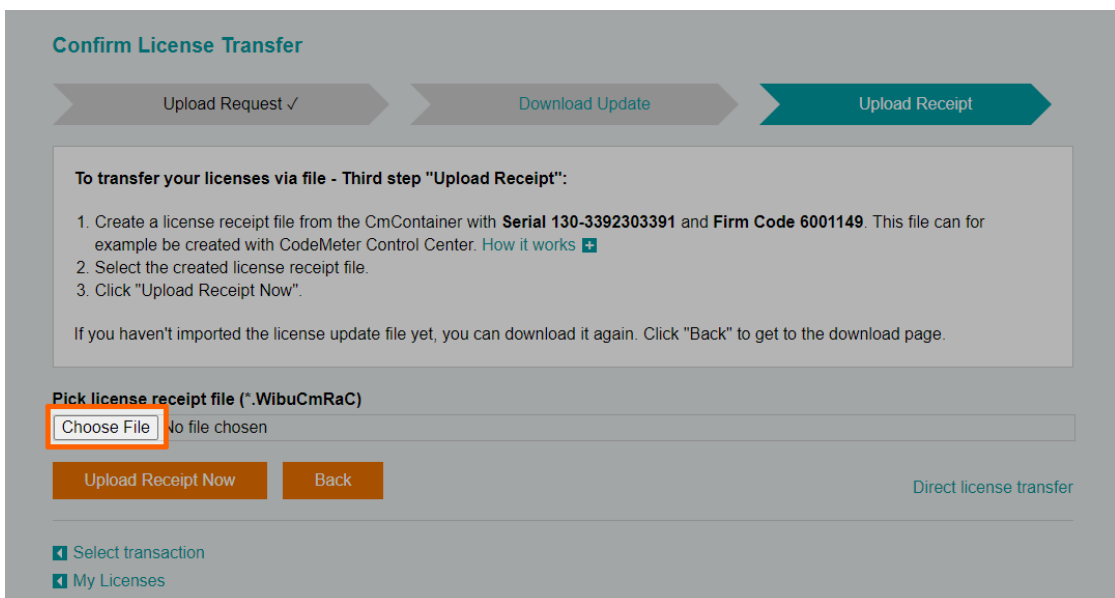


17. Switch back to the web depot.

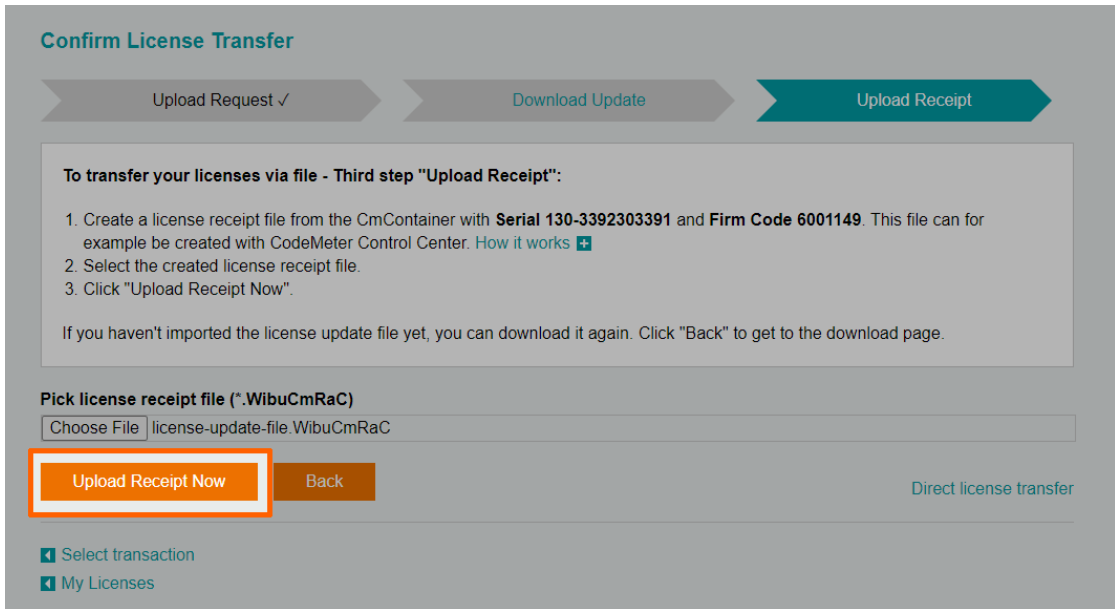
18. Select **Next**.



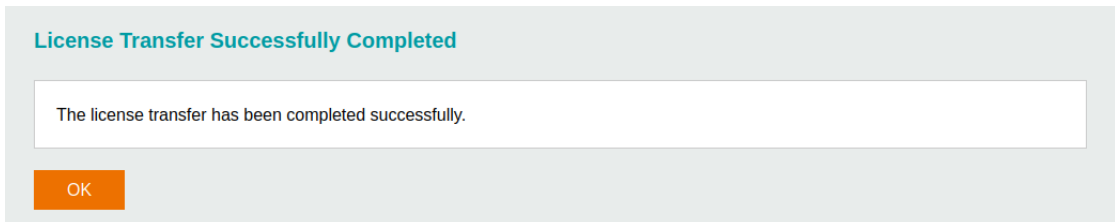
19. Select **Choose file**.



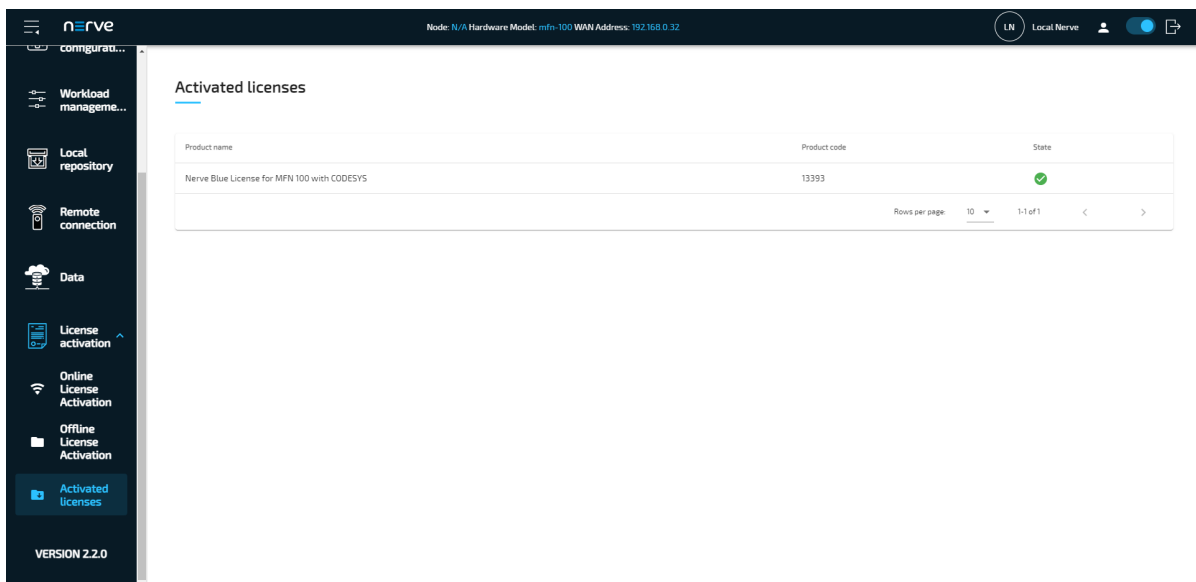
- Add the device fingerprint that was downloaded a few steps before.
- 20.
  21. Select **Upload receipt now**.



The web depot confirms a successful activation.



To double-check if the license has been activated properly, expand **License activation > Activated licenses** in the navigation on the left of the Local UI. An activated license is displayed in the table with a green check mark on the right side, signifying a successful activation on the correct Nerve Device.



With the license activated, return to the previous device guide page and continue with **Accessing the Local UI and registering the device.**

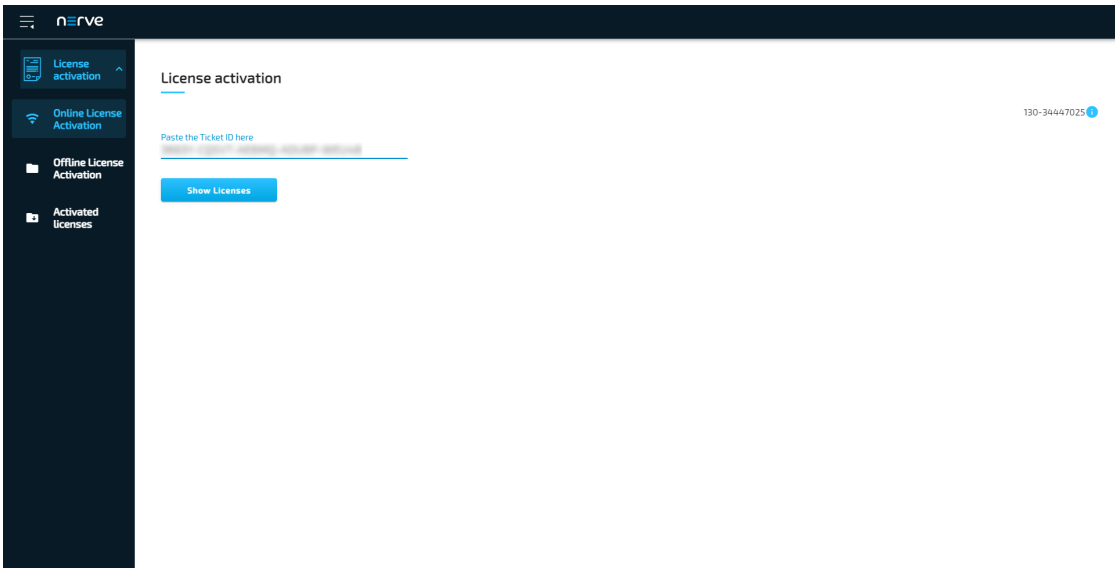
## Reusing activated licenses

Previously activated licenses can be reactivated. This is useful in cases when Nerve is reinstalled in order to wipe the disk or disk failure. However, licenses can only be reused for the same Nerve Device. In addition, the license serial number is required from the time the license was activated first. Without the license serial number from the previous installation, a new license has to be used.

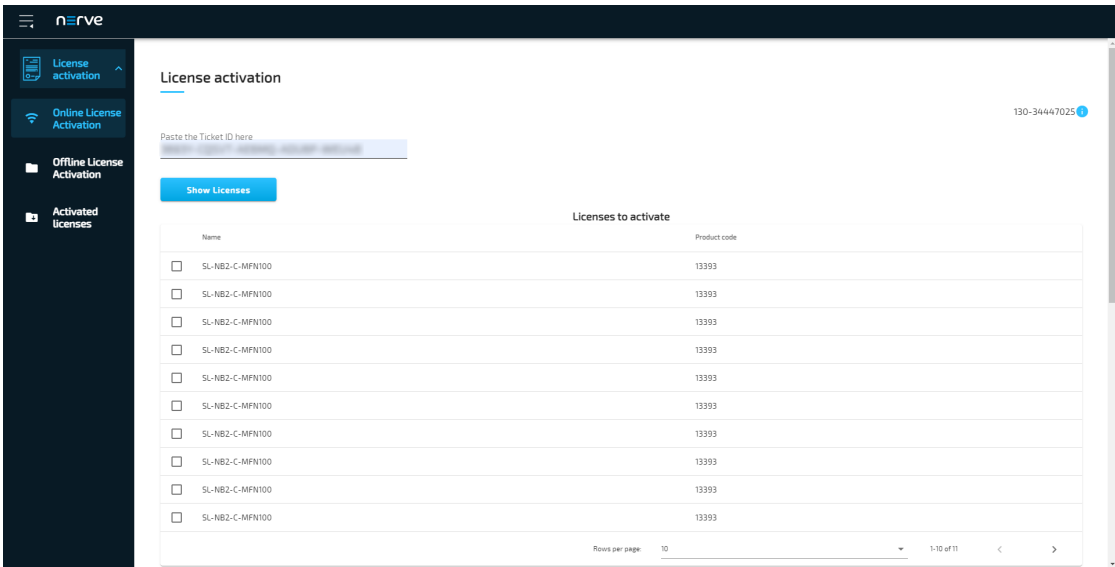
### Online reactivation

If the node has internet access, the node will automatically connect to the licensing server. The process is virtually identical to regular online license activation. Online license activation is selected in the navigation on the left by default.

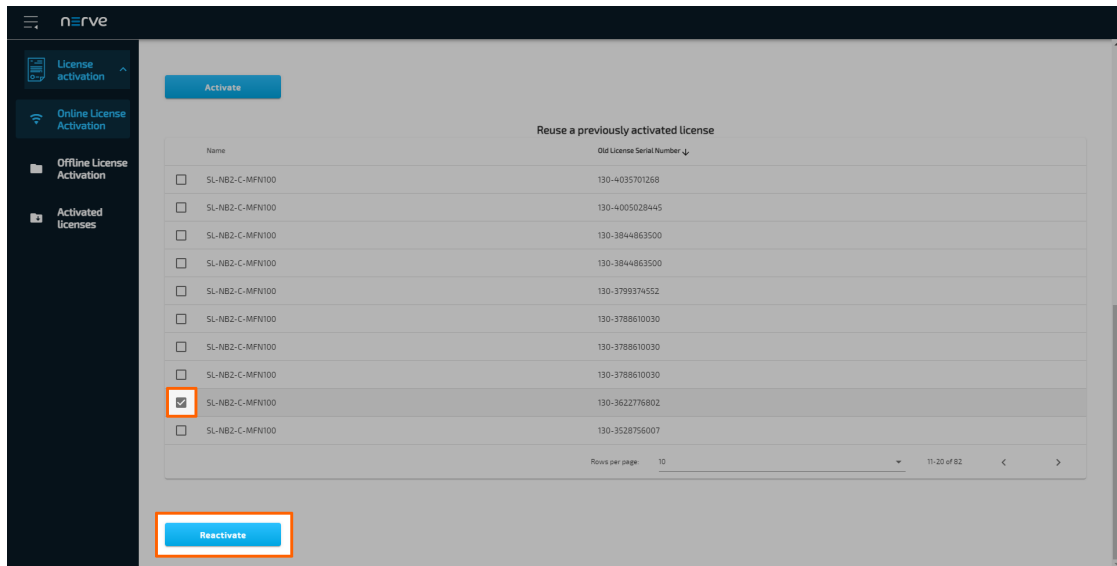
1. Enter the ticket ID under **Paste the Ticket ID here.**



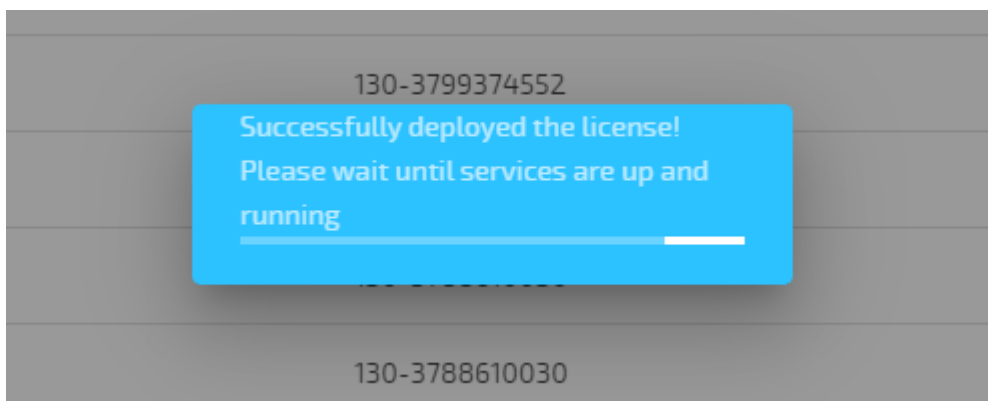
2. Select **Show Licenses.** Available licenses will appear below.



3. Scroll down to reach **Reuse a previously activated license**.
4. Tick the checkbox next to the appropriate license. Look for the license that matches the license serial number of the previous installation in the **Old License Serial Number** column.
5. Select **Reactivate** below the list of licenses.



The system will proceed to activate the license and automatically redirect to the Local UI after a successful activation.

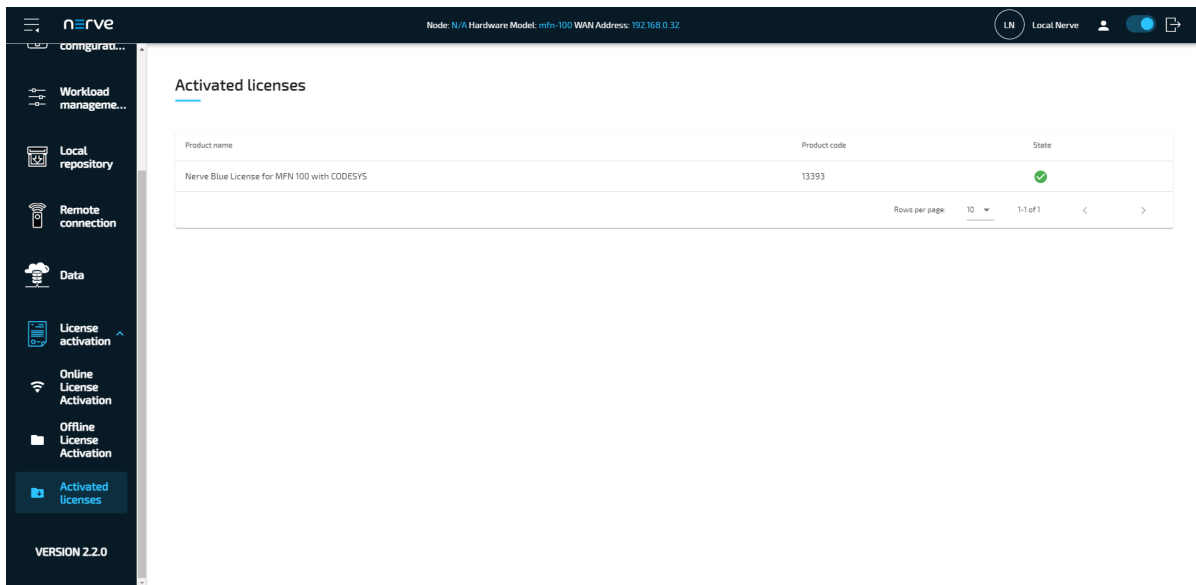


#### NOTE

When using an MFN 100, the device will light up blue once the license has been activated and the necessary services are up and running.

To double-check if the license has been activated properly, expand **License activation > Activated licenses** in the navigation on the left of the Local UI. An activated license is displayed in the table with a green check mark on the right side, signifying a successful activation on the correct Nerve Device.

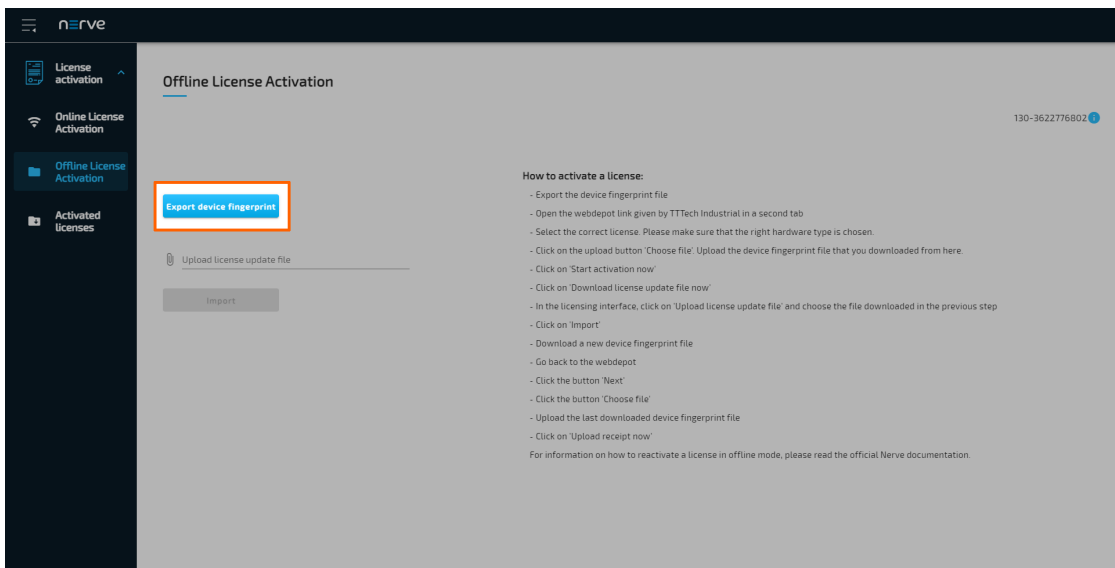




## Offline reactivation

In case of the node not having internet access, the license can be reactivated with a file-based method. However, note that a workstation with an internet connection is required for connecting to the licensing server in order to upload and download files.

1. Select **License activation > Offline License Activation** in the navigation on the left.
2. Select **Export device fingerprint** to download the license update file.



3. Open a new browser tab.
4. Enter the link to the web depot. The link has been sent as part of the delivery.
5. Select **My Licenses**.

WIBU SYSTEMS English

Home My Licenses Auto Update

### Available Licenses

Upload Request Download Update Upload Receipt

**To activate your licenses via file transfer - First step "Upload Request":**

If you have activated licenses from this ticket already, you can transfer additional licenses into the same CmContainer(s). If you want to use another CmContainer, you need a license request file of this new CmContainer.

1. Select an already used CmContainer or create a license request file with **Firm Code 6001149** for the CmContainer where you want to transfer the licenses to. This file can for example be created with CodeMeter Control Center. [How it works](#)
2. Select the licenses you want to activate.
3. Select the created license request file.
4. Click "Continue".

Name	Activated On	CmContainer	Status
<input checked="" type="checkbox"/> SL-NB2-C-K150	-		Available
<input type="checkbox"/> SL-NB2-C-K150	-		Available
<input type="checkbox"/> SL-NB2-C-K150	-		Available
<input type="checkbox"/> SL-NB2-C-K150	-		Available

6. Select **Restore licenses** at the bottom of the page.

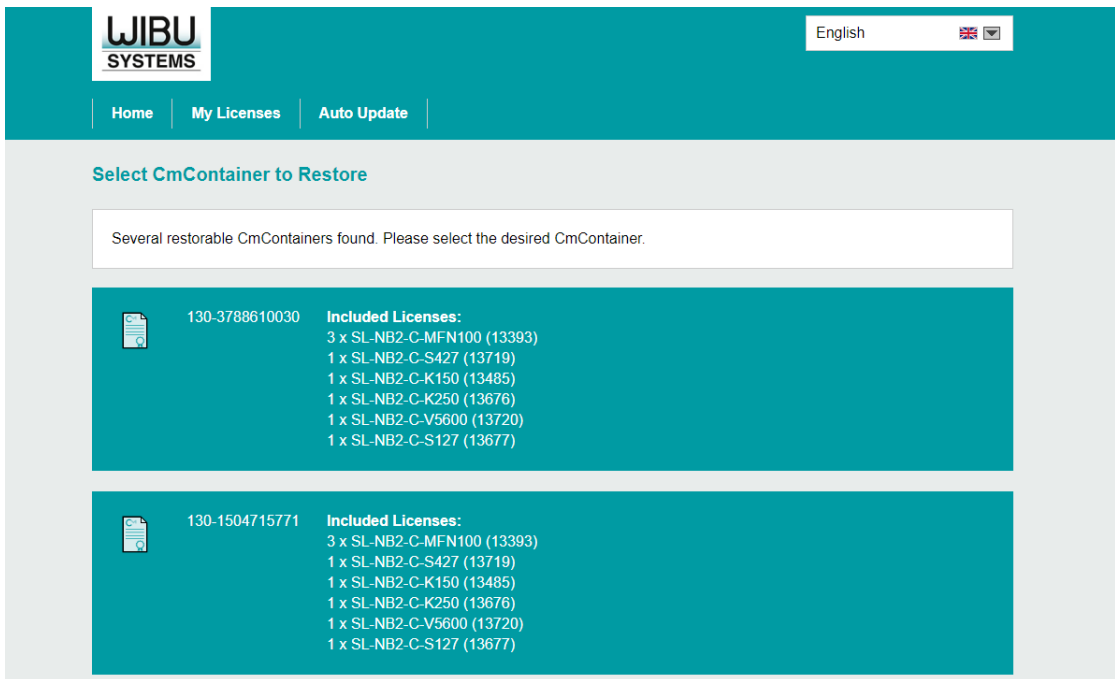
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	-		Available
SL-NB2-C-MFN100	2020-12-18 08:47:53	130-1966863304	Activated
SL-NB2-C-MFN100	-		Available

Activate Licenses **Restore Licenses** Continue License Transfer

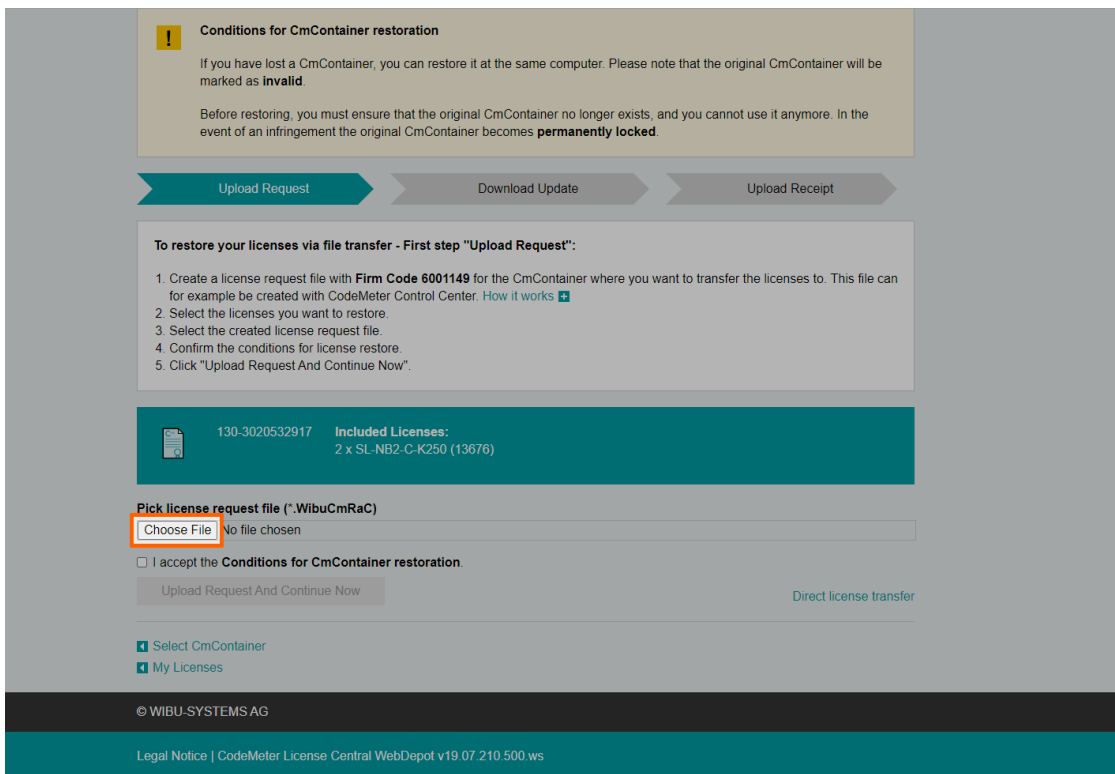
© WIBU-SYSTEMS AG

Legal Notice | CodeMeter License Central WebDepot v19.07.210.500.ws

7. Select the appropriate license. Look for the license that matches the license serial number of the previous installation.



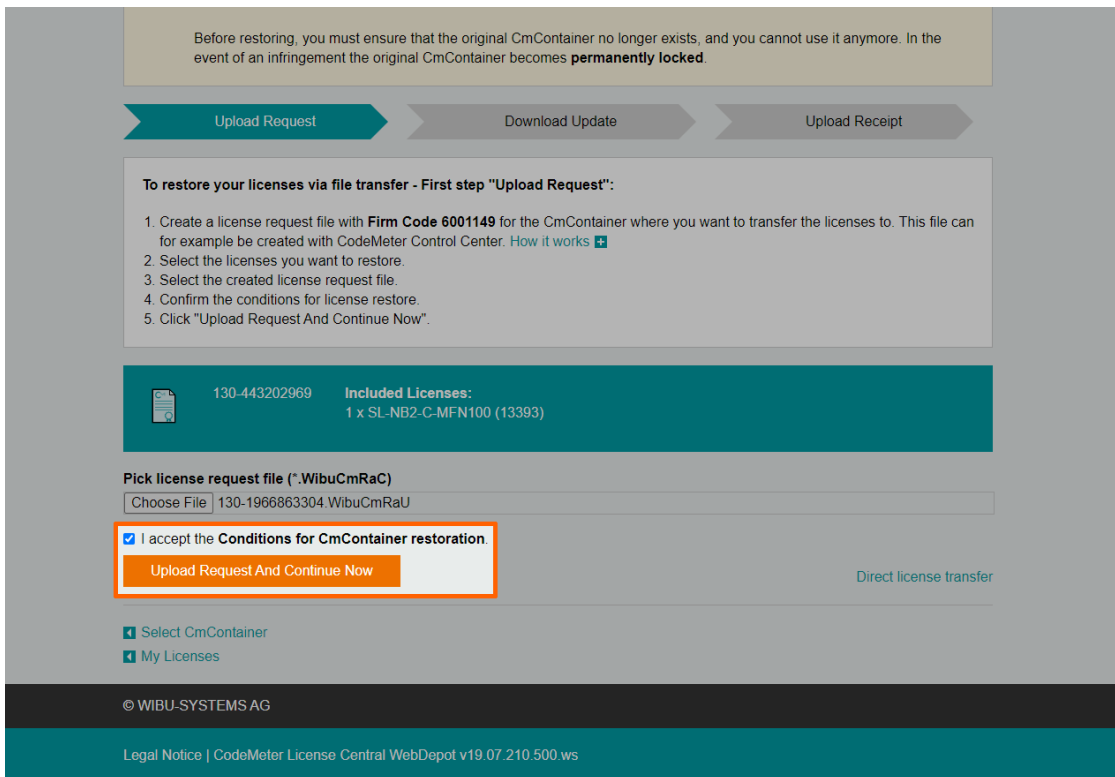
8. Select **Choose file** under **Pick license request file (\*.WibuCmRaC)**.



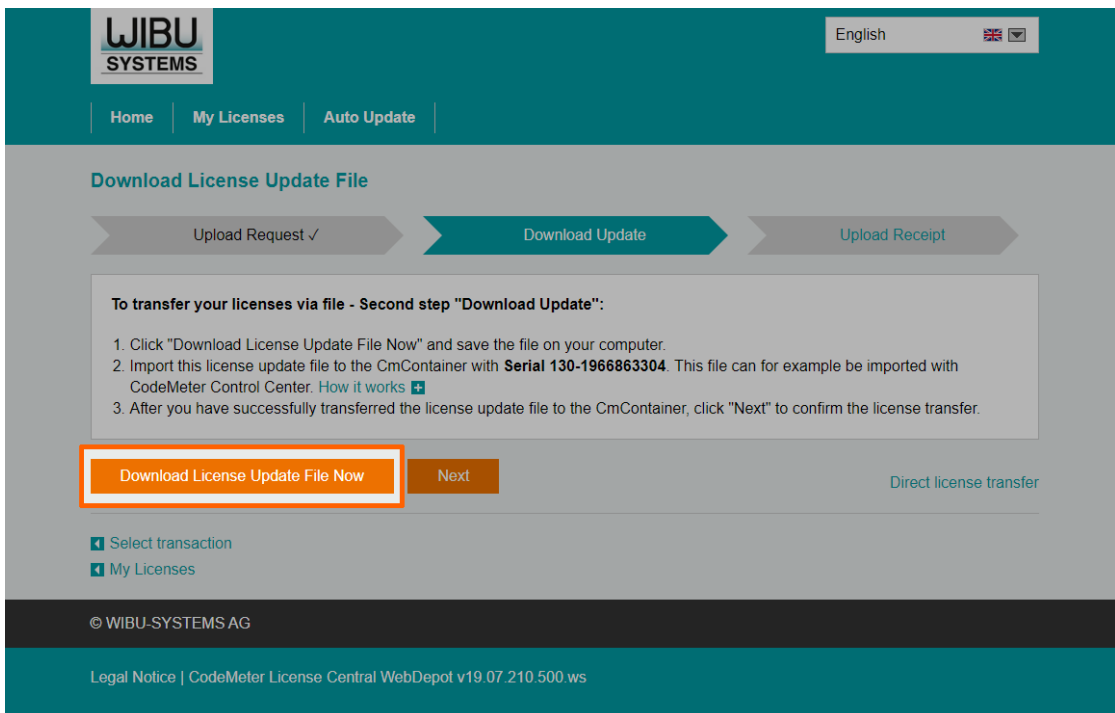
9. Navigate to where the device fingerprint file is saved and select it.

10. Tick the checkbox next to **I accept the Conditions for CmContainer restoration**.

11. Select **Upload Request And Continue Now**.

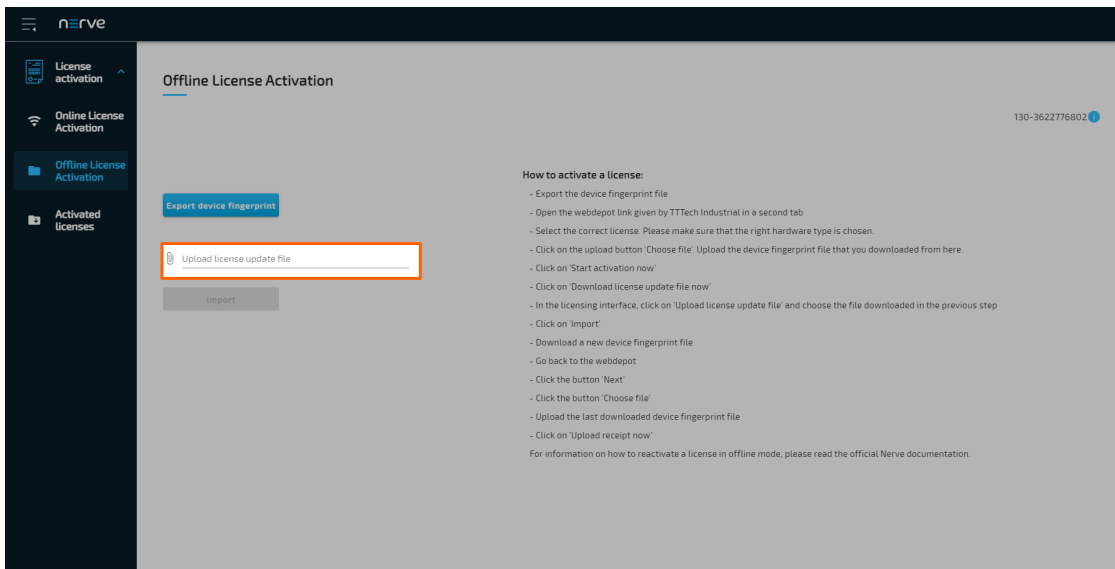


12. Select **Download License Update File now** in the next window to download the license update file named <licensenum>.WibuCmRaU.



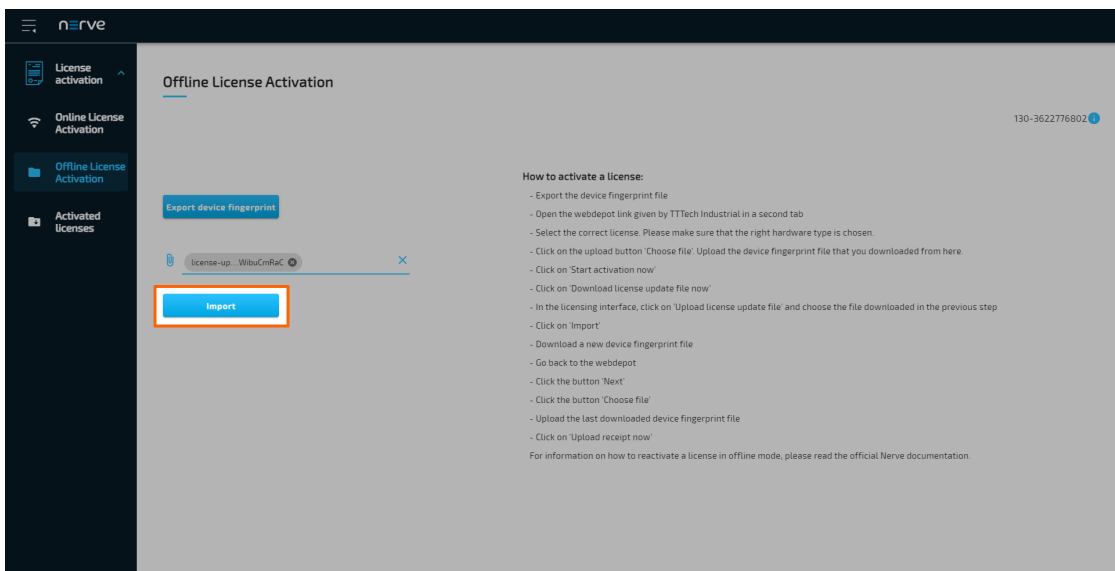
13. Switch back to the Local UI.

14. Select the **Upload license update file** field to open the file browser.

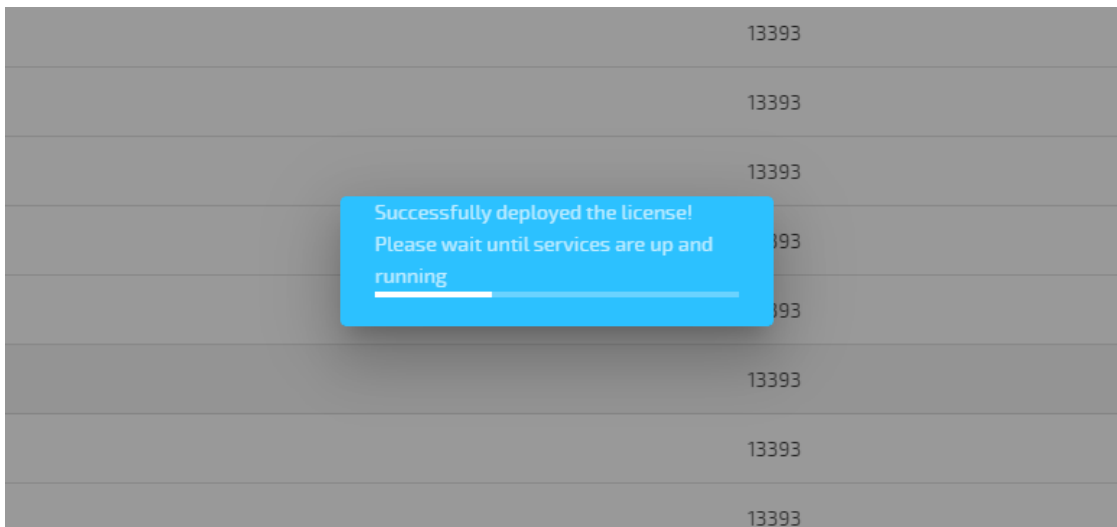


15. Navigate to where the <licensenum>.WibuCmRaU file is saved and select it.

16. Select **Import** to import the license.



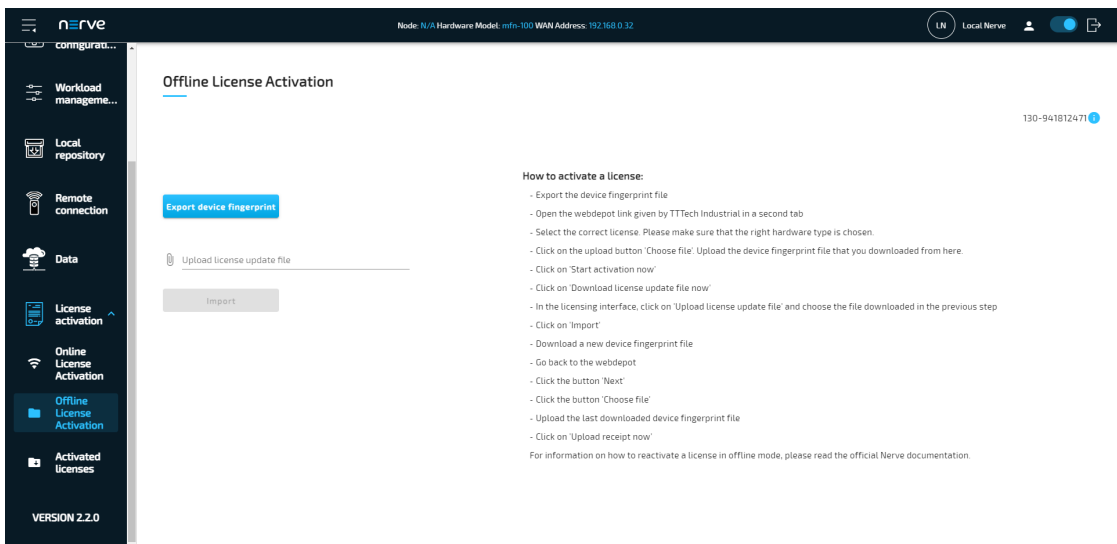
The system will proceed to activate the license and automatically redirect to the Local UI after a successful activation.



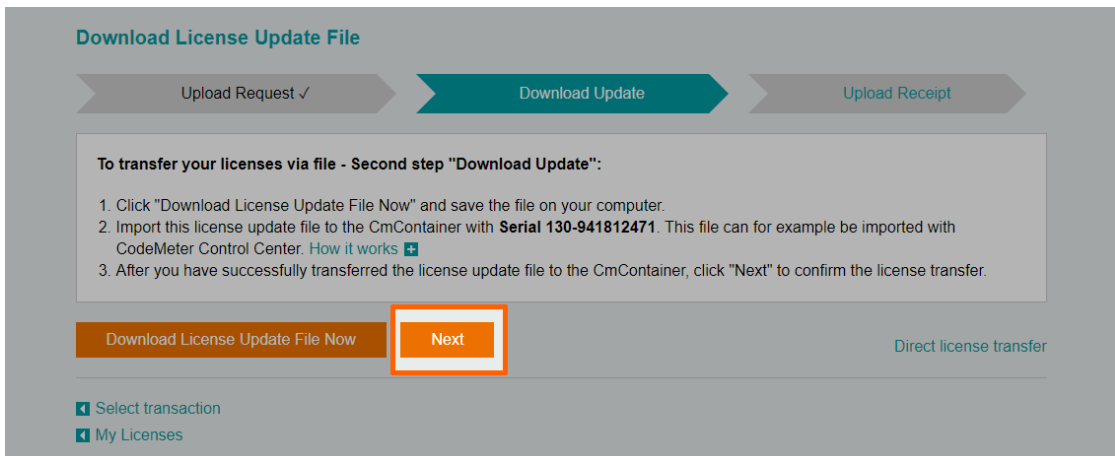
## NOTE

When using an MFN 100, the device will light up blue once the license has been activated and the necessary services are up and running.

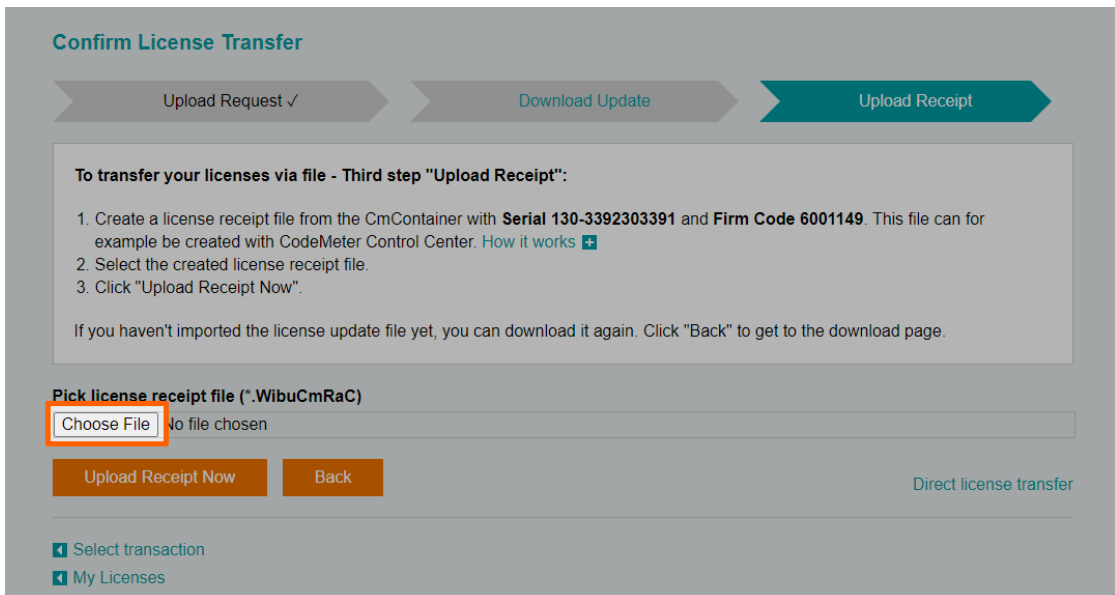
17. Log in to the Local UI with the credentials from the customer profile.
18. Select **License activation > Offline License Activation** in the navigation on the left.
19. Select **Export device fingerprint** to download the updated device fingerprint.



20. Switch back to the web depot.
21. Select **Next**.

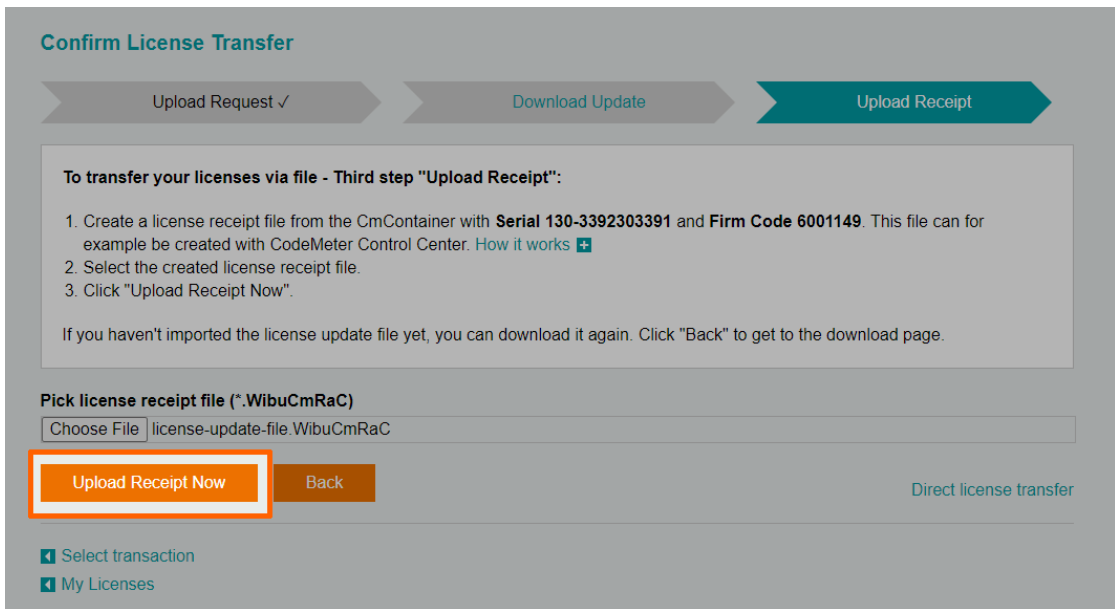


22. Select **Choose file**.

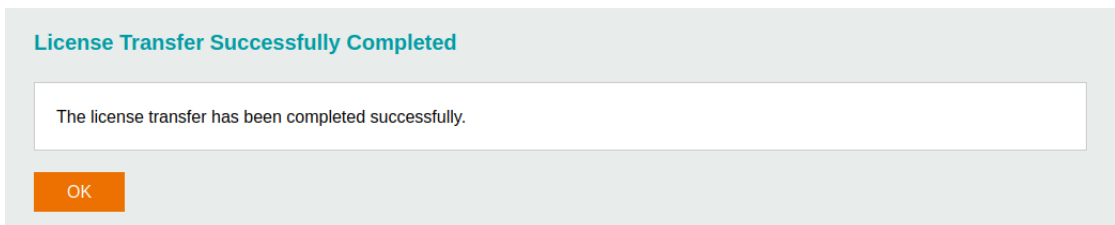


23. Add the device fingerprint that was downloaded a few steps before.

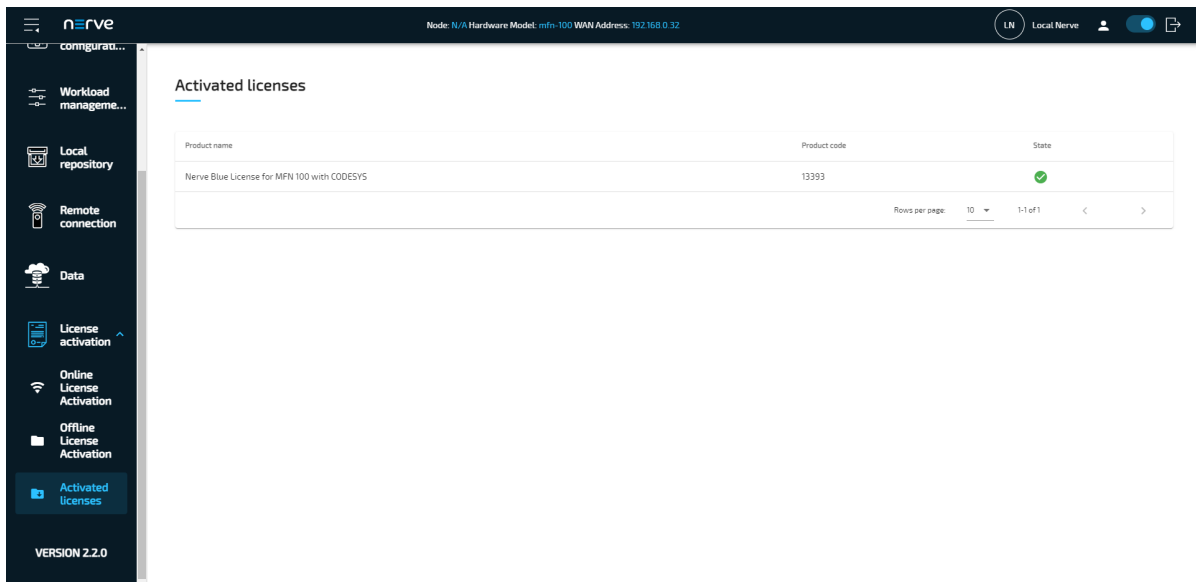
24. Select **Upload receipt now**.



The web depot confirms a successful activation.



To double-check if the license has been activated properly, expand **License activation > Activated licenses** in the navigation on the left of the Local UI. An activated license is displayed in the table with a green check mark on the right side, signifying a successful activation on the correct Nerve Device.



With the license activated, return to the previous device guide page and continue with **Accessing the Local UI and registering the device.**



# Local UI

The Local UI is provided by a web server that is running locally on the Nerve Device. Compared to the Management System, the Local UI covers features that only concern the node itself.

The Local UI gives access to the following features:

- Network configuration
- Node registration and password management
- Workload management
- Local workload deployment
- Local workload repository
- Management of remote connections

## Connecting to the Local UI

Connecting to the Local UI depends on the Nerve Device. Refer to the table below on how to reach the Local UI for each Nerve Device. Make sure to connect a workstation to the physical port of the Nerve Device associated with host access and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the host access interface with a 255.255.255.0 subnet mask.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.
<b>Maxtang AXWL10</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.

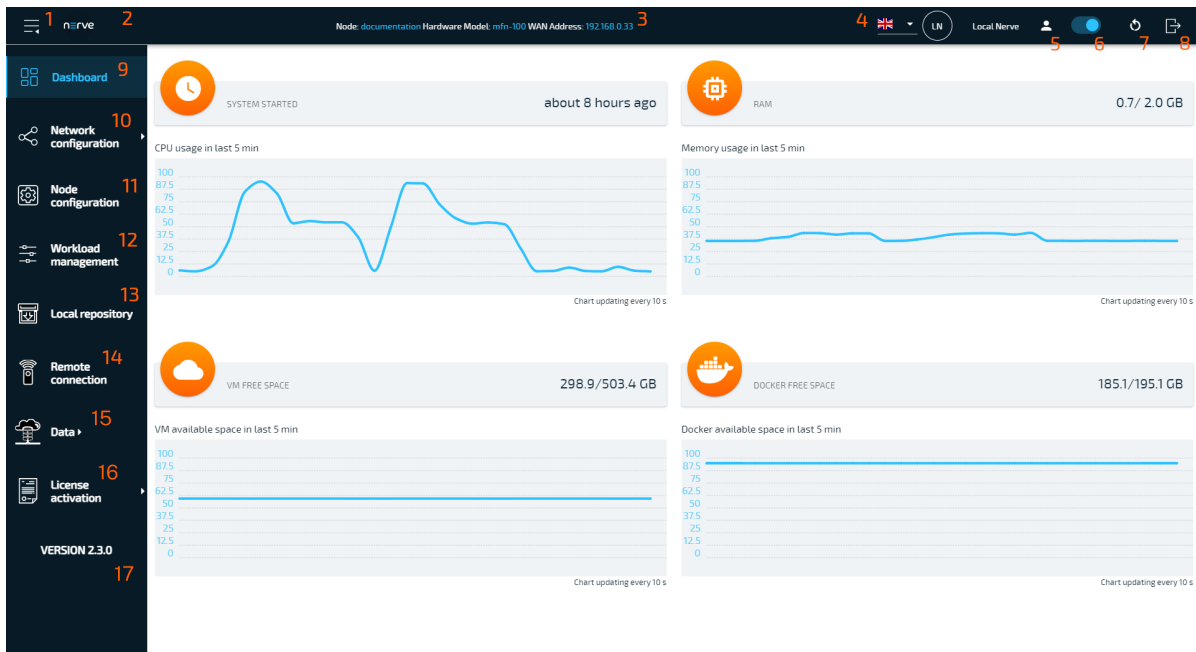
Nerve Device	Physical port	Local UI
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>  <wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>  <wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

Once a connection is established, log in with the credentials for the Local UI to reach the Local UI dashboard.



## Local UI dashboard

The dashboard of the Local UI is the default screen after the log in. Usage statistics of the Nerve Device are displayed in the window with more options in the menu on the left side.

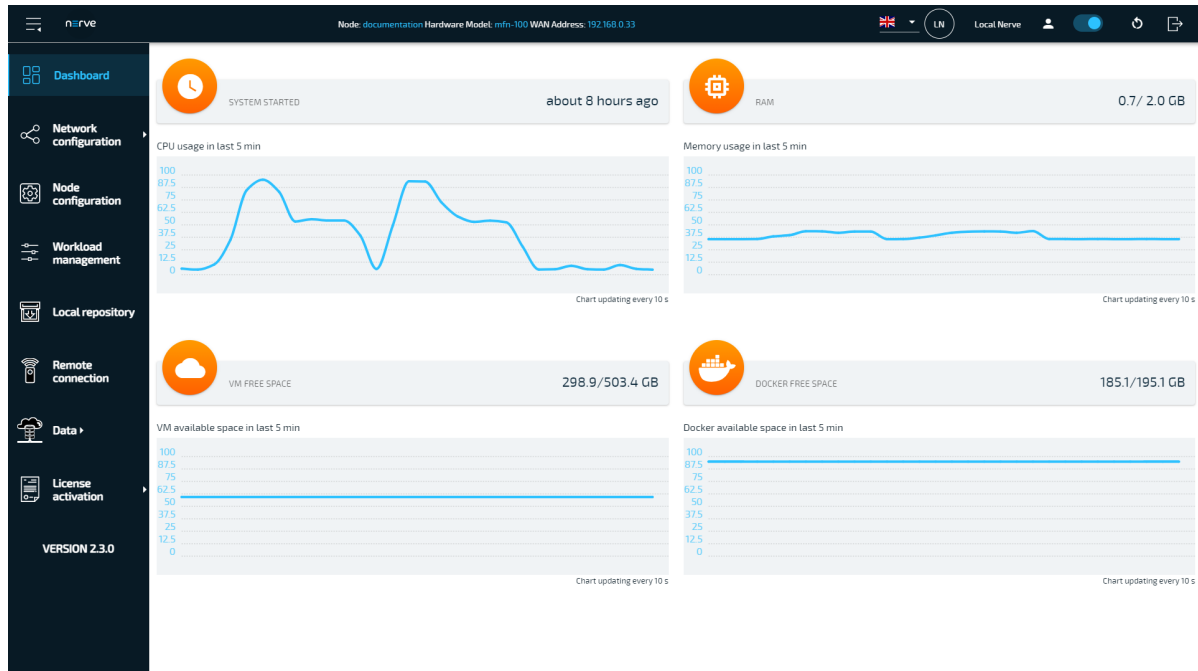


Term	Description
<b>Burger menu (1)</b>	Expand and collapse the left-hand menu by clicking here.
<b>Nerve logo (2)</b>	Click here to return to the dashboard and reload the page.
<b>Node details (3)</b>	Details of the node are displayed here, showing the name of the node in the Management System, the hardware model and the WAN address of the node.
<b>Language selection (4)</b>	Click here to select the interface language. Available languages are English and Korean.
<b>Change password (5)</b>	Clicking here leads to an area where the password to the Local UI can be changed.
<b>Connect button (6)</b>	Toggle the slider here to disconnect the node from the network. Blue indicates an active connection to the network. If the button is white, the node is offline.
<b>Reboot Node (7)</b>	Clicking here will reboot the node after a confirmation dialog. Select <b>YES</b> to initiate the reboot. All running workloads will be stopped. After the reboot, the workloads will return to their previous state. Running workloads will be started. Stopped workloads will stay stopped.
<b>Log out (8)</b>	Click here to log out of the Local UI.
<b>Dashboard (9)</b>	Select this to display the dashboard containing the system metrics — graphs showing available resources of the Nerve Device and their usage over time.
<b>Network configuration (10)</b>	This menu allows to configure the Ethernet ports of the Nerve Device, as well as proxy settings for the node.
<b>Node configuration (11)</b>	Configure data such as the Management System URL that the node will connect to or the serial number of the node here. The information required to register a node in the Management System is configured in this menu. Also, passwords can be changed here.

Term	Description
<b>Workload management (12)</b>	Select this menu for control options for deployed workloads and manual deployment of workloads.
<b>Local Repository (13)</b>	Find settings for the configuration of a local workload repository here.
<b>Remote connection (14)</b>	Manage incoming and active remote connections here.
<b>Data (15)</b>	Access the instance of the Nerve Data Services in the Local UI here. Refer to <a href="#">Nerve Data Services</a> for more information.
<b>License activation (16)</b>	This is the UI that is used for activating licenses. Use this menu to check currently activated licenses.
<b>Node version (17)</b>	This is the currently installed version of the node.

## System metrics

The graphs in the Local UI dashboard show available resources of the Nerve Device and their usage over time. The y-axis displays percentages and the x-axis is updated every 10 seconds, showing a time span of 5 minutes. The percentages displayed are always in relation to the maximum of the available resource:

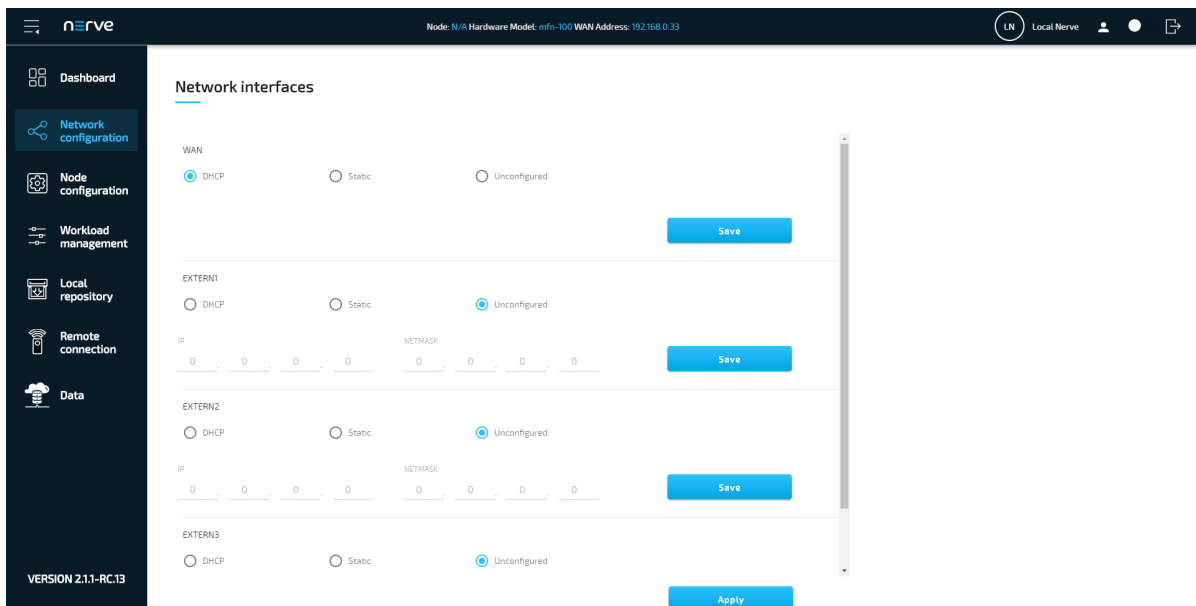


Item	Description
<b>SYSTEM STARTED</b>	This shows how long the Nerve Device has been running. If the device is restarted, this value is reset.
<b>CPU usage in last 5 min</b>	The graph here shows the percentage of processing power that is being used. This includes CPUs that have been assigned to VMs and Docker containers.

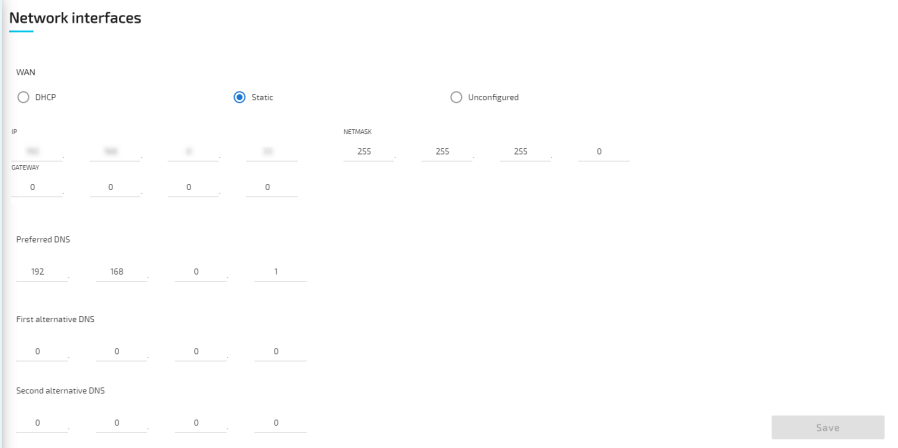
Item	Description
<b>RAM</b>	This shows how much memory is used (left value) and how much memory is available in total (right value). Example: 0.3/1.9 GB
<b>Memory usage in last 5 min</b>	Note that the total amount of memory the Nerve Device offers is not shown here. This is the memory that is available for the host. Similar to CPU usage, the graph shows the percentage of memory used. This includes memory that has been assigned to VMs or Docker containers.
<b>VM FREE SPACE</b>	Virtual machines have their dedicated virtual partition (Logical Volume Manager). The values show how much of this partition is used (left value) and how much is available in total (right value). Example: 86.5/238.5 GB
<b>VM free space in last 5 min</b>	This graph shows the percentage of space that is being used by the Logical Volume Manager.
<b>DOCKER FREE SPACE</b>	Similar to VM free space, Docker containers have their dedicated virtual partition. The values show how much of this partition is used (left value) and how much is available in total (right value). Example: 2.9/40.3 GB
<b>Docker free space in last 5 min</b>	This graph shows the percentage of space for Docker containers that is being used.

## Local network configuration

From the Local UI, the Ethernet ports of the Nerve Device can be configured. Select **Network configuration** in the navigation on the left to reach this menu. The example below is of the MFN 100. The page is specific to the Nerve Device. The number and names of interfaces may differ.



Item	Description
<b>DHCP</b>	The IP address of the port will be assigned by the DHCP server. If an IP address has been assigned, it will be displayed here.

Item	Description
<b>Static</b>	<p>By selecting <b>Static</b>, the IP address of the port needs to be manually defined. Enter the IP address and subnet mask under <b>IP</b> and <b>NETMASK</b> to set a static IP address. For the WAN interface, the <b>GATEWAY</b>, the preferred <b>DNS</b> as well as two alternative DNS can also be set.</p> 

**Unconfigured** If **Unconfigured** is checked, the port is disabled for the host but can still be used for virtual machines with bridged interfaces.

## NOTE

For more information on networking and interfaces refer to the [networking chapter](#).

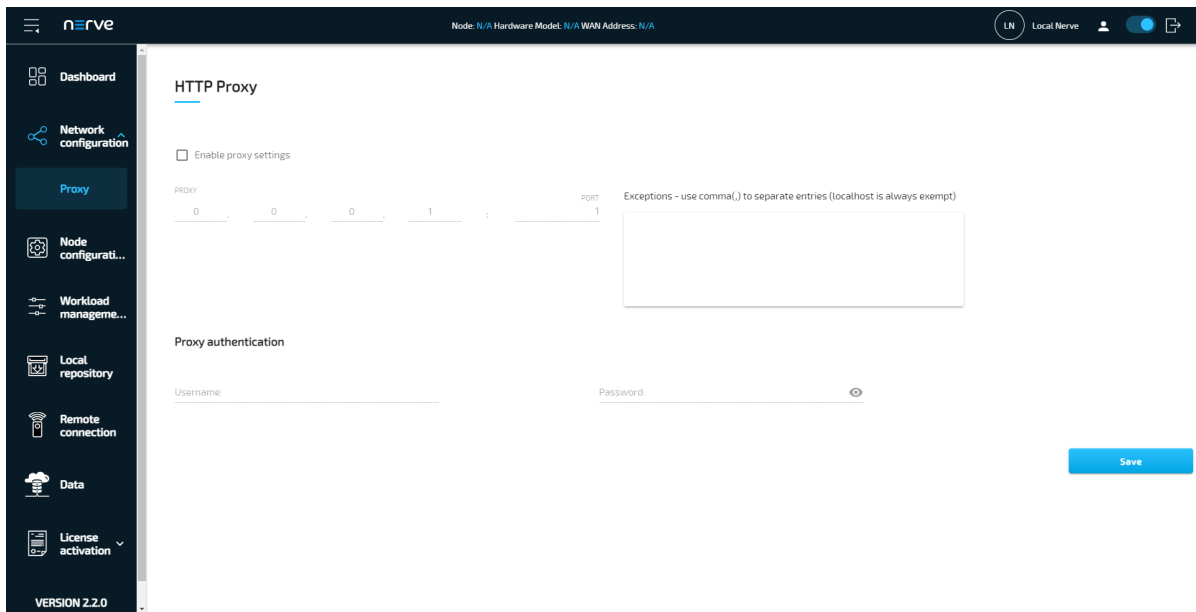
## Proxy settings

Network configuration settings can be extended with the possibility to specify proxy settings. In case a proxy server is present in the company network, these settings can be used to enable communication between the node and the Management System. The proxy settings define an HTTP/S proxy for the node with the possibility to specify credentials if they are requested, as well as exceptions. Note that the proxy settings affect the internal communication of the node towards the Management System, and the connections covered by the proxy settings are only those related to:

- the management of workloads on the node
- logs sent from the node to the Management System
- data transmitted from the Data Services Gateway
- the Debian Advanced Package Tool (APT)

Potential connections generated by workloads on the node do not use the node proxy settings.

Expand **Network configuration** > **Proxy** to reach the proxy settings.



Item	Description
<b>Enable proxy settings</b>	Tick the checkbox here to enable proxy settings.
<b>PROXY</b>	Enter the IP address of the proxy server here.
<b>PORT</b>	Enter the number of the port that is open for communication at the proxy server.
<b>Exceptions - use comma(,) to separate entries (localhost is always exempt)</b>	List IP addresses or hostnames of exceptions separated by a comma here. The node will directly communicate with the exceptions without using the proxy server. An example is a local workload repository that can be defined in the Local UI. Refer to <a href="#">Setting a local repository</a> for more information.
<b>Proxy authentication</b>	Enter the <b>Username</b> and <b>Password</b> required to access the proxy server. This is optional. If the proxy server does not require login credentials, the fields can be left empty.

Select **Save** to apply the configuration. To undo the proxy settings, uncheck **Enable proxy settings** and select **Save**.

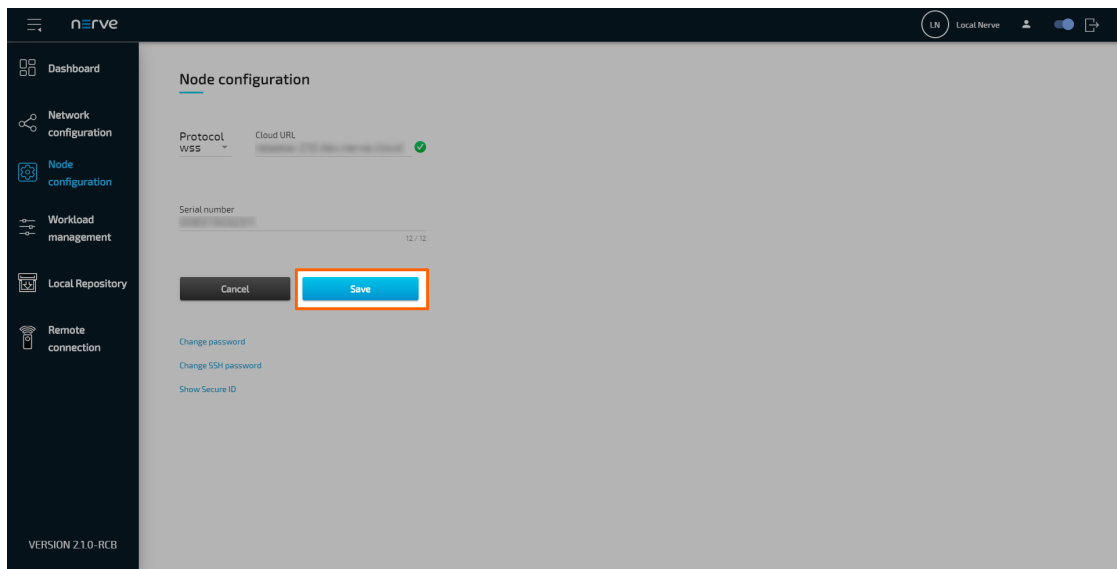
## Node configuration

The first part of registering a node in the Management System is performed in the Local UI. Node details and information required for registering a node in the Management System are configured under **Node configuration**.

1. Select **Node configuration** in the navigation on the left.
2. Enter the following information:

	Select <b>wss</b> or <b>ssl</b> from the drop-down menu: <ul style="list-style-type: none"> <li>◦ <b>wss</b> Selecting this will use the WebSocket Secure protocol for the registration of the node, meaning that port 443 will be used for communication between the node and the Management System.</li> <li>◦ <b>ssl</b> Selecting this will use the Secure Sockets Layer protocol for the registration of the node, meaning that port 8883 will be used for communication between the node and the Management System. Note that port 8883 of the company firewall has to be open when using the SSL protocol.</li> </ul>
<b>Protocol</b>	
<b>Cloud URL</b>	Enter the URL of the Management System without the protocol, e.g example.nerve.cloud.
<b>Serial number</b>	Enter a serial number with a minimum of 12 characters. Note that this serial number can be freely defined. It is required for node registration in the Management System and serves as a means of identification. Entering the serial number that is printed on the Nerve Device is not required but recommended.

3. Click **Save**.



With the serial number saved in the node configuration, the secure ID has been generated and can be displayed by selecting **Show Secure ID**. This secure ID is required when adding the node in the Management System. Refer to [Adding a node](#) on how to add nodes to the Management System.

## Changing the password for host access

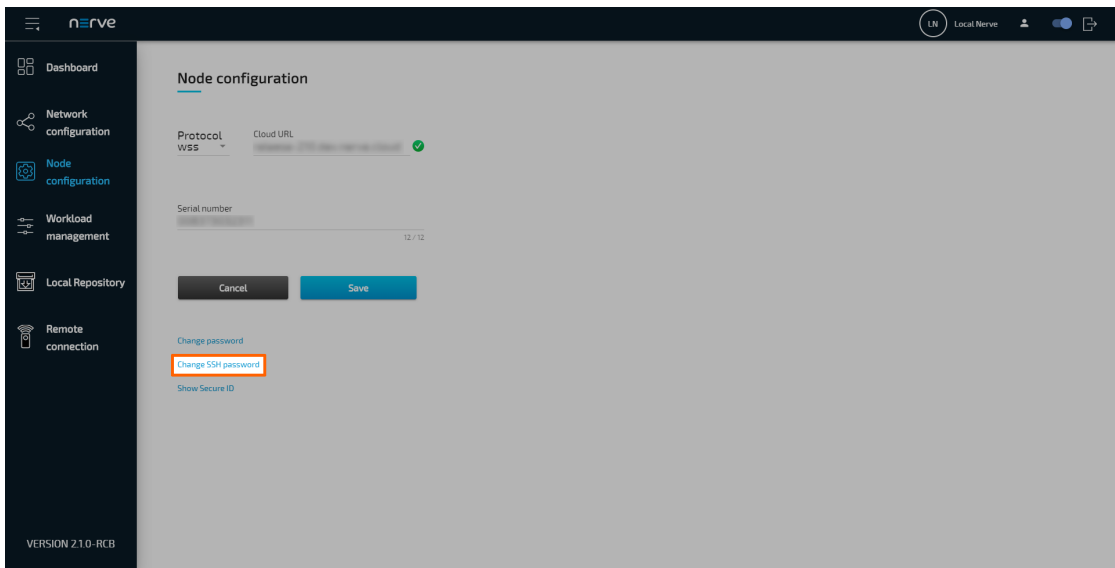
The password for host access via SSH can be changed. The default password for host access is found in the customer profile. Changing the host access password makes it persist through version updates. If the default password is not changed, updating the node to a newer version will change the host access password to a new default.

### NOTE



Access to the Linux host system of Nerve is provided in order to enable advanced use cases. Using host access requires expert Linux knowledge as system internal changes can be performed. Note that changes may impact the Nerve system.

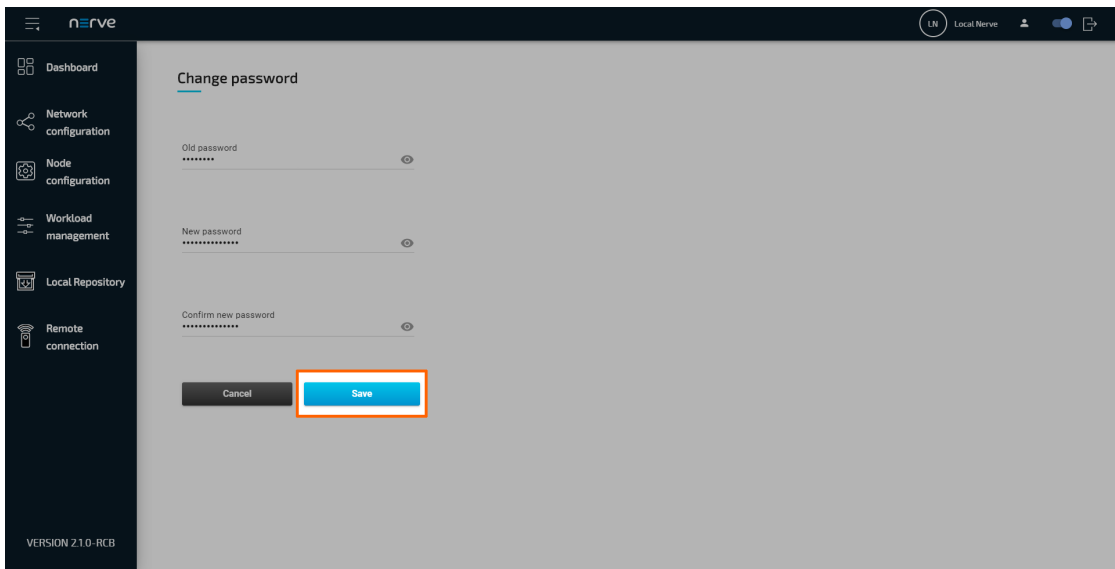
1. Select **Node configuration** in the navigation on the left.
2. Select **Change SSH password**



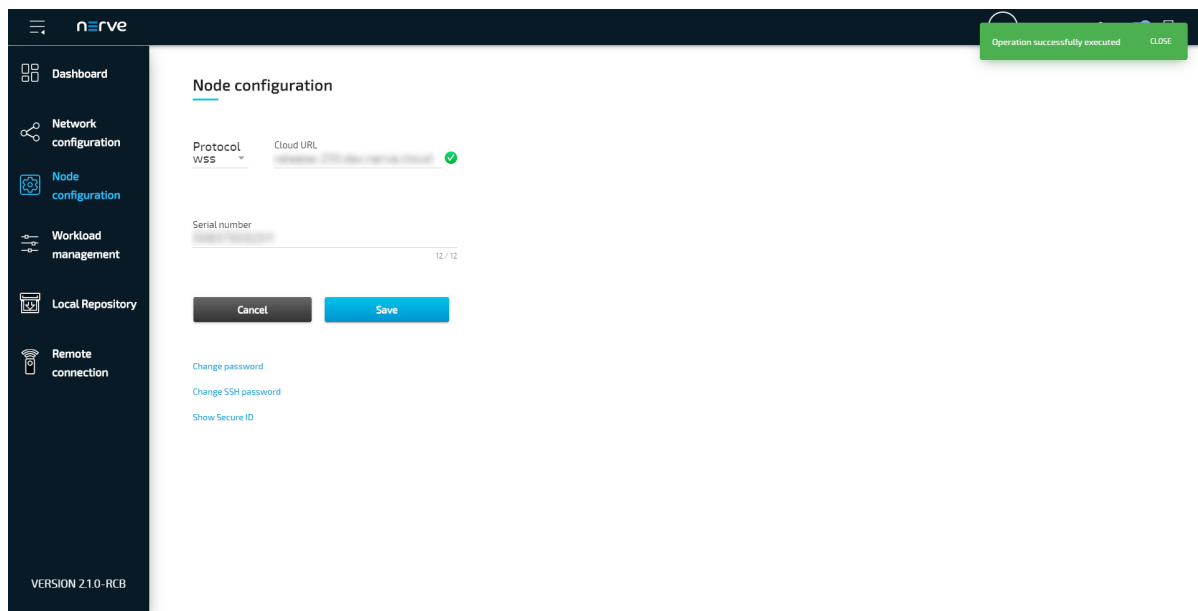
3. Enter the following information:

Item	Description
<b>Old password</b>	Enter the old password for host access.
<b>New password</b>	Enter the new password here. The new password must be 8 characters or longer and it can only consist of alphanumeric characters.
<b>Confirm new password</b>	Enter the new password again. Both passwords must match in order to proceed.

4. Select **Save** to set the new password.



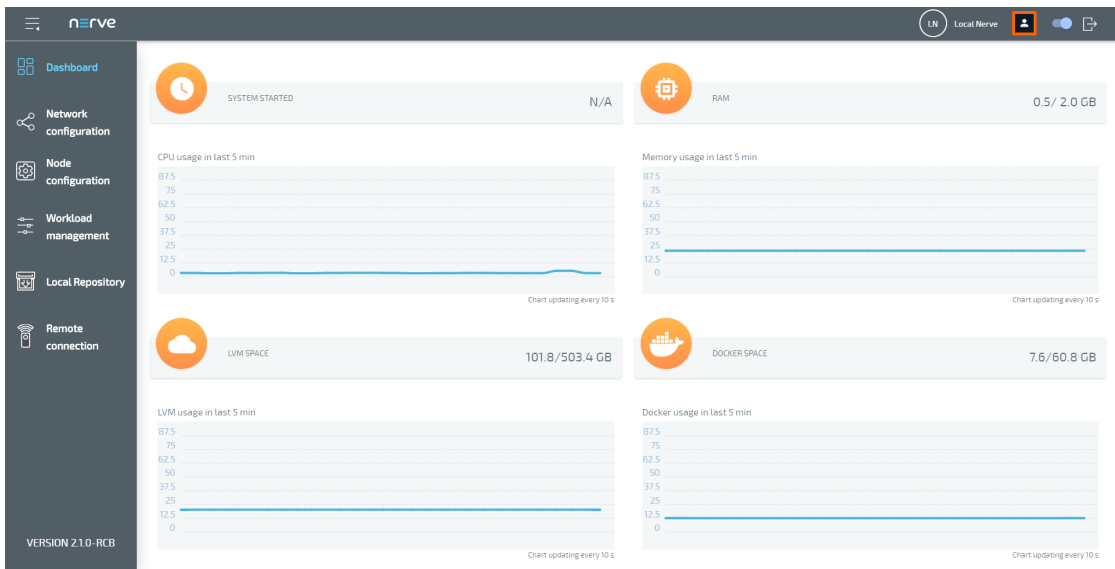
If the process was successful, the Local UI will display the dashboard with a green notice confirming the change in the upper-right corner.



## Changing the password for the Local UI

The password to the Local UI can be changed. The default password for Local UI access can be found in the customer profile. Changing the password to the Local UI makes it persist through version updates. If the default password is not changed, updating the node to a newer version will change the Local UI password to a new default.

1. Select the user icon (**Change password**) in the upper-right.



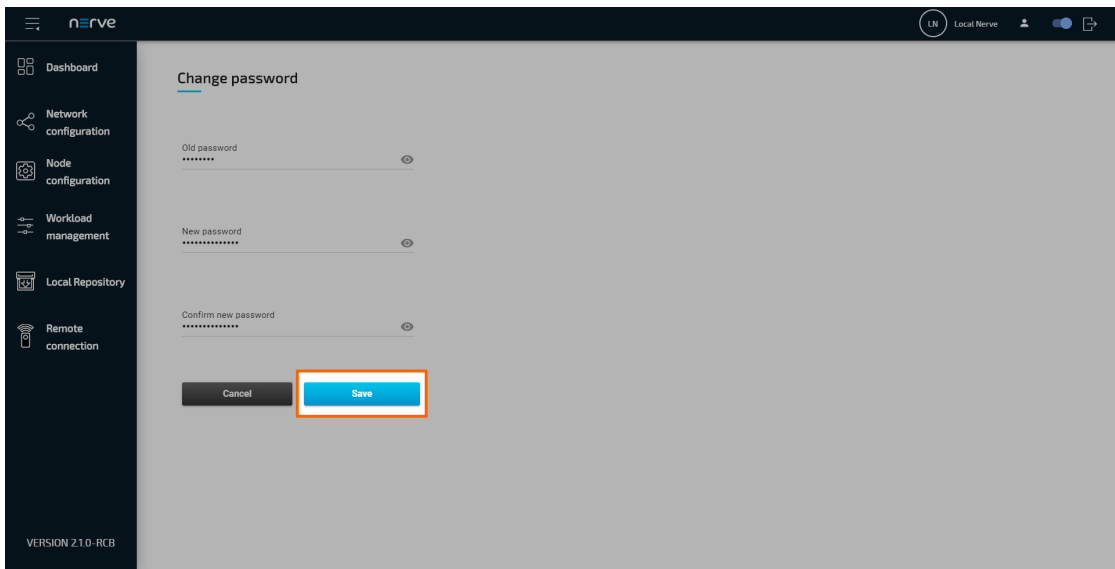
## NOTE

Alternatively, it is also possible to change the password in the Node configuration menu. Select **Node configuration** in the navigation on the left and click **Change password** to reach the password form.

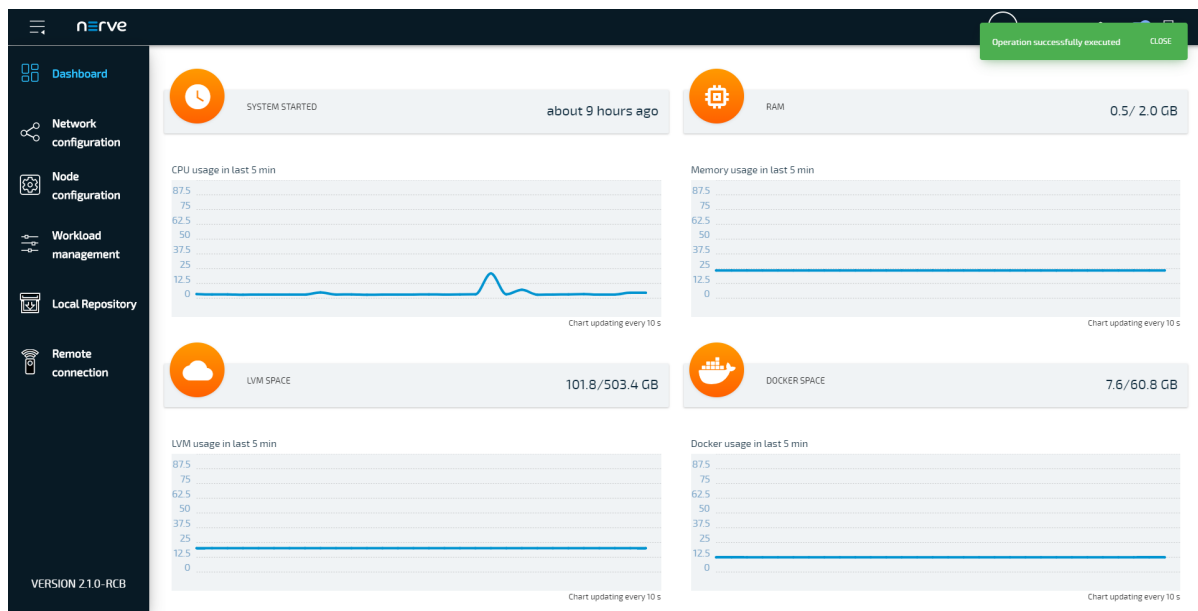
2. Enter the following information:

Item	Description
<b>Old password</b>	Enter the old password to the Local UI.
<b>New password</b>	Enter the new password here. The new password must be 8 characters or longer and it can only consist of alphanumeric characters.
<b>Confirm new password</b>	Enter the new password again. Both passwords must match in order to proceed.

3. Select **Save** to set the new password.

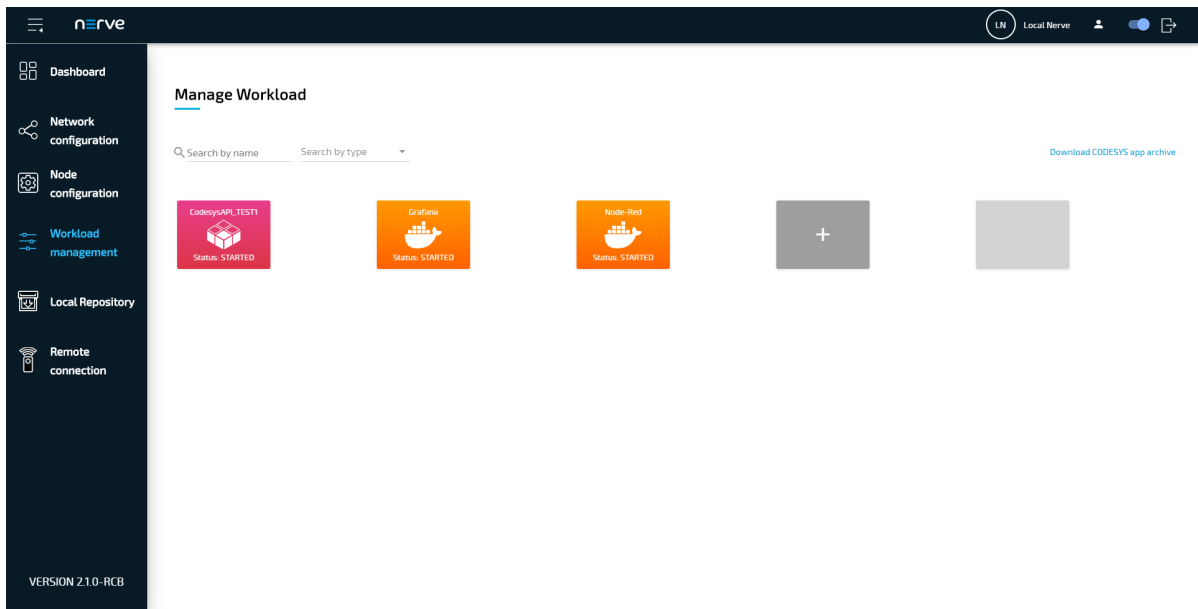


If the process was successful, the Local UI will display the dashboard with a green notice confirming the change in the upper-right corner.

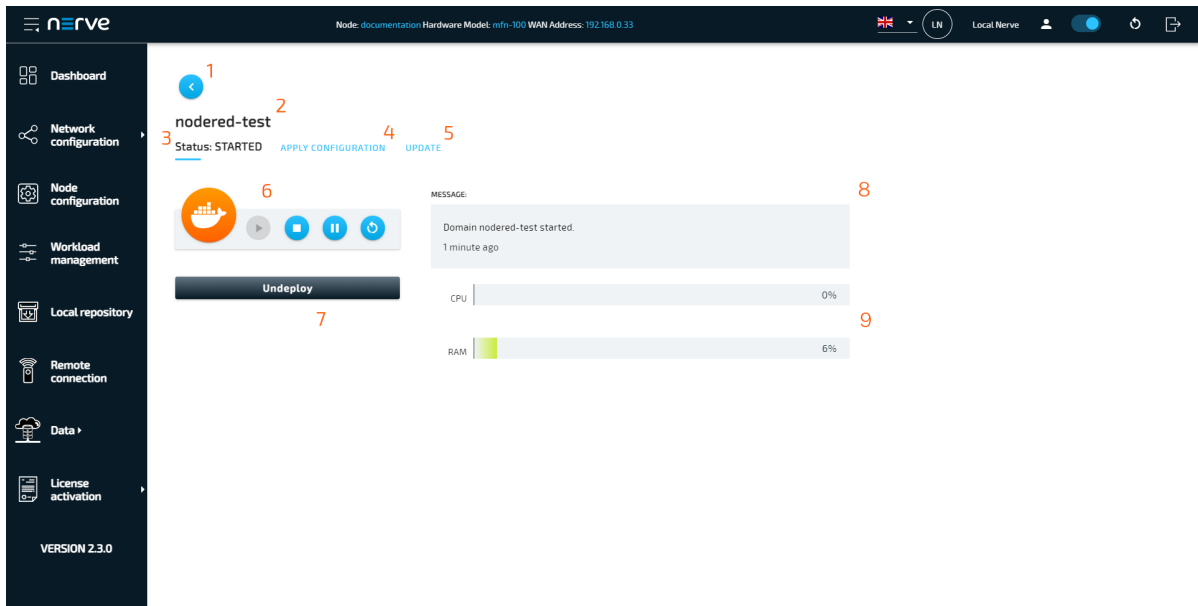


## Workload management

Deployed workloads can be controlled in the Local UI. Select **Workload management** in the navigation menu on the left-hand side for an overview of all deployed workloads. Then, select a workload to reach the interface for controlling workloads running on the Nerve Device. If there is a large number of workloads deployed, filter the list by name or by type. Enter a search query under **Search by name** to filter the list by name. Select **All**, **VM**, **Codesys** and **Docker** from the drop-down menu to filter workloads by type.



Note that CODESYS workloads can only be controlled in the Local UI, as operation of a CODESYS workload may have an impact on machine operation and therefore should not be controlled remotely. Virtual Machine workloads and Docker workloads can also be controlled in the Management System. Refer to the [workload control section](#) in the node tree chapter on how to control workloads in the Management System.



Item	Description
<b>Back button (1)</b>	Click here to return to overview of workloads.
<b>Workload name (2)</b>	This is the name of the workload. The name of the workload version is not displayed.

Item	Description
<p><b>Workload status (3)</b></p>	<p>The current status of the workload is displayed here. The possible statuses are the following:</p> <ul style="list-style-type: none"> <li>• <b>Idle</b> This is the initial state of the workload before it is started.</li> <li>• <b>Creating</b> This is a transitional state of the workload when it is being created on the node.</li> <li>• <b>Starting</b> This is a transitional state when the workload is being started.</li> <li>• <b>Restarting</b> This is a transitional state when the workload is being restarted.</li> <li>• <b>Started</b> The workload is running and operating.</li> <li>• <b>Suspending</b> This is a transitional state when the workload is being suspended.</li> <li>• <b>Suspended</b> The workload has been paused.</li> <li>• <b>Resuming</b> This is a transitional state when the workload is being resumed from the suspended state.</li> <li>• <b>Stopping</b> This is a transitional state when the workload is being stopped.</li> <li>• <b>Stopped</b> The workload has been stopped.</li> <li>• <b>Removing</b> This is a transitional state when the workload is being undeployed.</li> <li>• <b>Error</b> An unknown error has occurred.</li> </ul>
<p><b>APPLY CONFIGURATION (4)</b></p>	<p>This element only applies to Docker workloads with configuration storage defined. Selecting this allows the upload of configuration files. The configuration files have to be archived and uploaded in a ZIP file. For more information, refer to <a href="#">Applying configuration files to a workload</a>.</p>
<p><b>UPDATE (5)</b></p>	<p>This element applies to CODESYS and Docker workloads. Select this to upload a workload version update as a TAR file. Note that the workload update must be exported from the Management System first. Refer to <a href="#">Updating a deployed workload in the Local UI</a> below for more information.</p>

Item	Description
<b>Control panel (6)</b>	<p>There are five control options for workloads here:</p> <ul style="list-style-type: none"> <li>• <b>Play</b> If the workload is in a stopped state, clicking <b>Play</b> will start the workload.</li> <li>• <b>Stop</b> If the workload is running, clicking <b>Stop</b> will stop the workload.</li> <li>• <b>Suspend</b> Clicking <b>Suspend</b> will pause the workload. It can be continued by clicking <b>Play</b>.</li> <li>• <b>Restart</b> This will restart the workload.</li> </ul> <p>Note that the control panel for CODESYS workloads only offers <b>Play</b> and <b>Stop</b>. Also, stopping a CODESYS workload will reset it to its initial values unless the CODESYS application has been written using the retain variables library. Refer to <a href="#">Enabling retain variables</a> for more information.</p>
<b>Undeploy (7)</b>	<p>Clicking here removes the workload from the node. To deploy the workload again, it has to be deployed through the <a href="#">Local UI</a> or the <a href="#">Management System</a>.</p>

Item	Description
<p><b>Message window (8)</b></p>	<p>The message window displays the latest message the workload has sent out including how much time has passed since it was sent out. The type of message that is displayed here depends on the workload. Here is a list of messages that are valid for VMs and Docker containers:</p> <ul style="list-style-type: none"> <li>• "Domain creating."</li> <li>• "ERROR during creating! &lt;errormessage&gt;"</li> <li>• "Domain starting."</li> <li>• "ERROR during starting! &lt;errormessage&gt;"</li> <li>• "Domain &lt;domainname&gt; started."</li> <li>• "Domain stopping."</li> <li>• "ERROR during stopping! &lt;errormessage&gt;"</li> <li>• "Domain &lt;domainname&gt; stopped."</li> <li>• "Domain suspending."</li> <li>• "ERROR during suspending! &lt;errormessage&gt;"</li> <li>• "Domain &lt;domainname&gt; suspended."</li> <li>• "Domain resuming."</li> <li>• "ERROR during resuming! &lt;errormessage&gt;"</li> <li>• "Domain restarting."</li> <li>• "ERROR during restarting."</li> <li>• "Domain removing!!!"</li> <li>• "ERROR during removing."</li> <li>• "ERROR!!! Domain stopping."</li> </ul> <p>In the messages above, &lt;domainname&gt; is a placeholder for the name of the VM or Docker. In case of Docker containers, &lt;errormessage&gt; signifies a message that is generated by the Docker container if an error occurs.</p> <p>For VMs, there is an additional set of messages:</p> <ul style="list-style-type: none"> <li>• "Failed to connect to hypervisor."</li> <li>• "Failed to create domain."</li> <li>• "Domain &lt;domainname&gt; created."</li> <li>• "Cannot start &lt;domainname&gt; domain because it may already be running!"</li> <li>• "Failed to resume &lt;domainname&gt; domain!"</li> <li>• &lt;errormessage&gt;</li> <li>• "Failed to start domain &lt;domainname&gt;. "</li> <li>• &lt;errormessage&gt;</li> </ul> <p>In this case, &lt;errormessage&gt; is a message that is fetched from the libvirt library.</p> <p>CODESYS workloads have the following set of messages:</p> <ul style="list-style-type: none"> <li>• "Preparing files for installation"</li> <li>• "Starting CODESYS application"</li> <li>• "CODESYS application started"</li> <li>• "Stopping CODESYS application"</li> <li>• "CODESYS application stoppped"</li> <li>• "Removing CODESYS application file"</li> <li>• "An unexpected error has occurred. &lt;errormessage&gt;"</li> </ul> <p>Here, &lt;errormessage&gt; is a message that is sent between the node and CODESYS.</p>



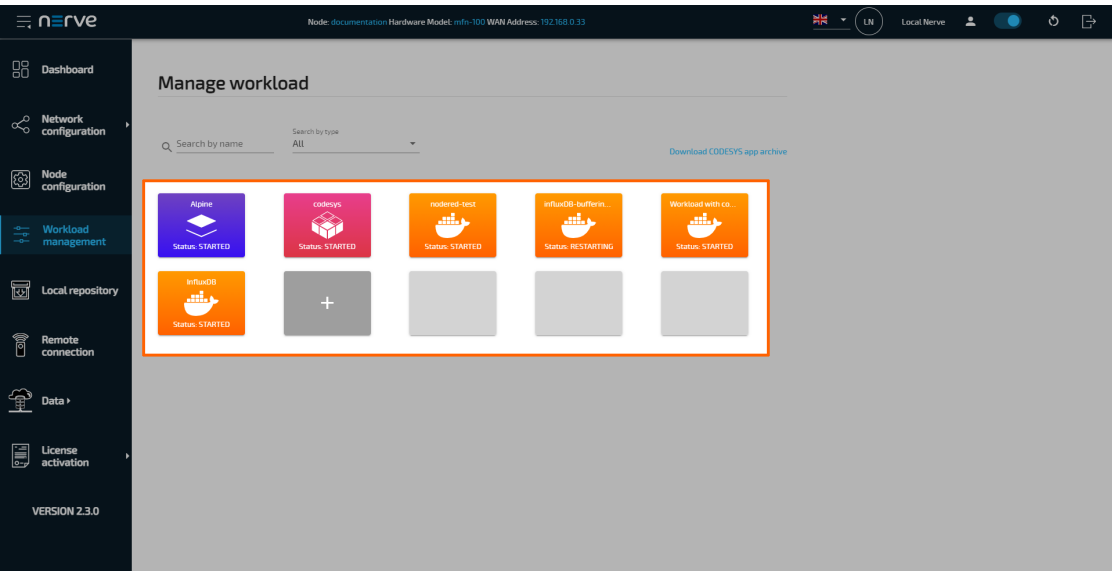
Item	Description
<p><b>Usage statistics (9)</b></p>	<p>Virtual Machine workloads and Docker workloads have their assigned resources they can use. The use of these resources is displayed with bar graphs:</p> <ul style="list-style-type: none"> <li> <p><b>CPU (VM and Docker)</b>            The percentage here shows the usage of CPU resources in relation to the assigned CPUs.            Example: A VM is assigned one CPU core out of four and the core is at 75 % usage capacity. The graph will be at 75 %.</p> </li> <li> <p><b>RAM (Docker only)</b>            Similar to the CPU usage statistic, the percentage here shows the usage of system memory resources in relation to the assigned memory. If the assigned memory is at a 100 % usage capacity, the graph will be at 100 %.            If no memory has been assigned, the graph will show the percentage of used memory in relation to the total available memory of the host.</p> </li> </ul>

All changes performed in the Local UI are reflected in the [Management System](#).

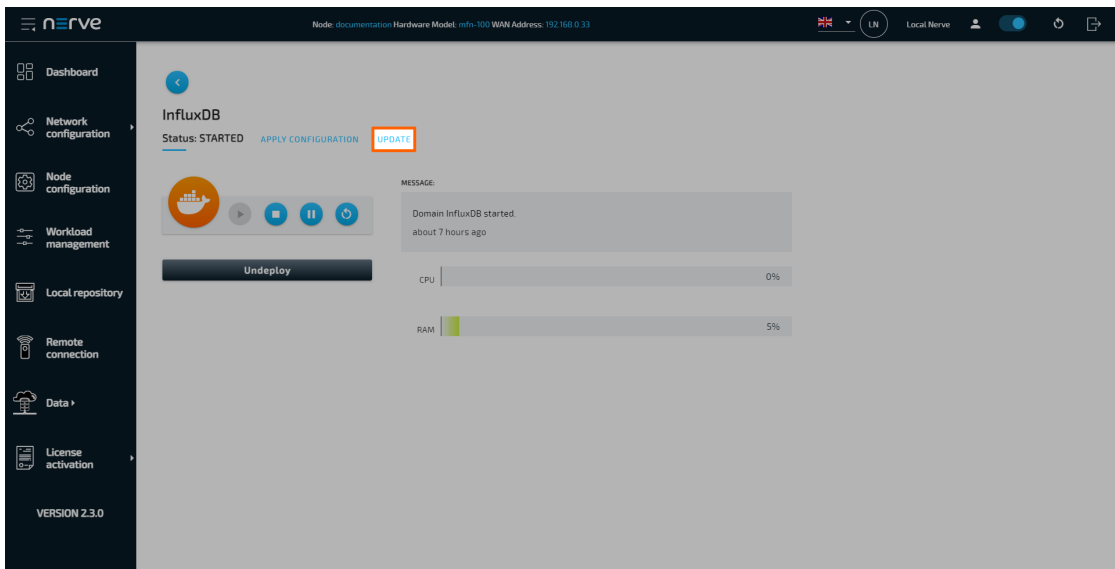
### Updating a deployed workload in the Local UI

Deployed CODESYS and Docker workloads can be updated through the workload control screen. Updating the workload is a quick way to deploy a new version of the same workload. However, note that the workload update needs to be exported first when updating a workload through the Local UI. Refer to [Exporting a workload](#) for more information on how to export a workload in the Management System.

1. Select **Workload management** in the navigation on the left.
2. Select the tile of an updatable workload.



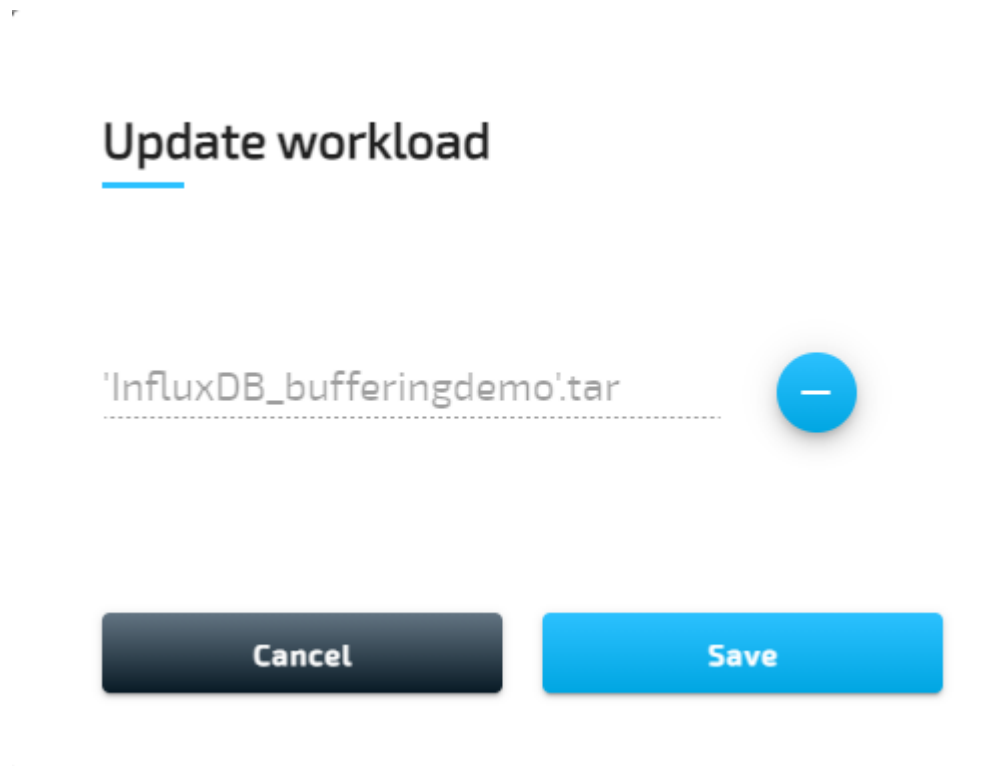
3. Select **UPDATE**.



## NOTE

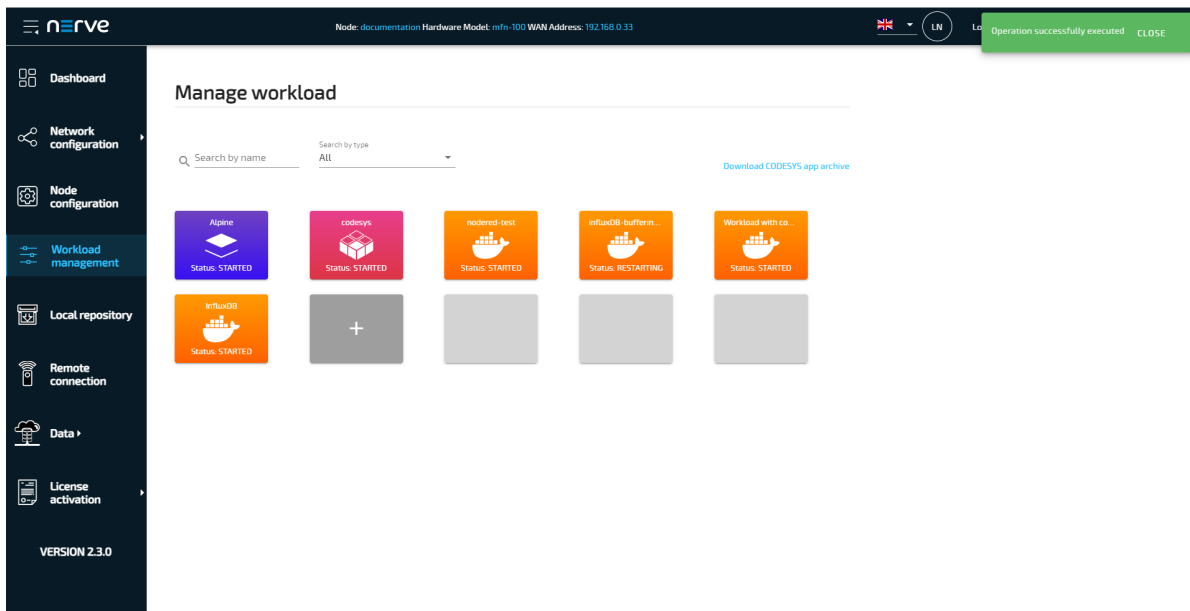
The notification bubble next to **UPDATE** indicates the number of available versions of the workload.

4. Select the plus icon to open the file browser.
5. Select the exported workload version update in TAR format.
6. Select **Open** to add the exported workload update.



7. Select **Save**.

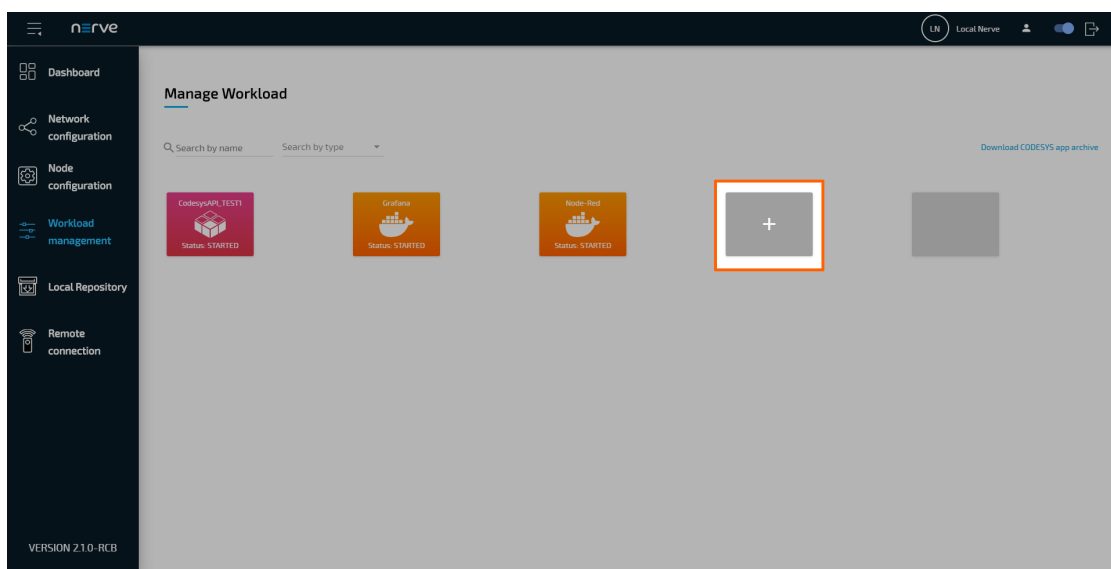
This starts the update of the workload version. Once the update is completed, the Local UI will return to the workload management screen and display a green success message in the upper-right corner.



## Local workload deployment

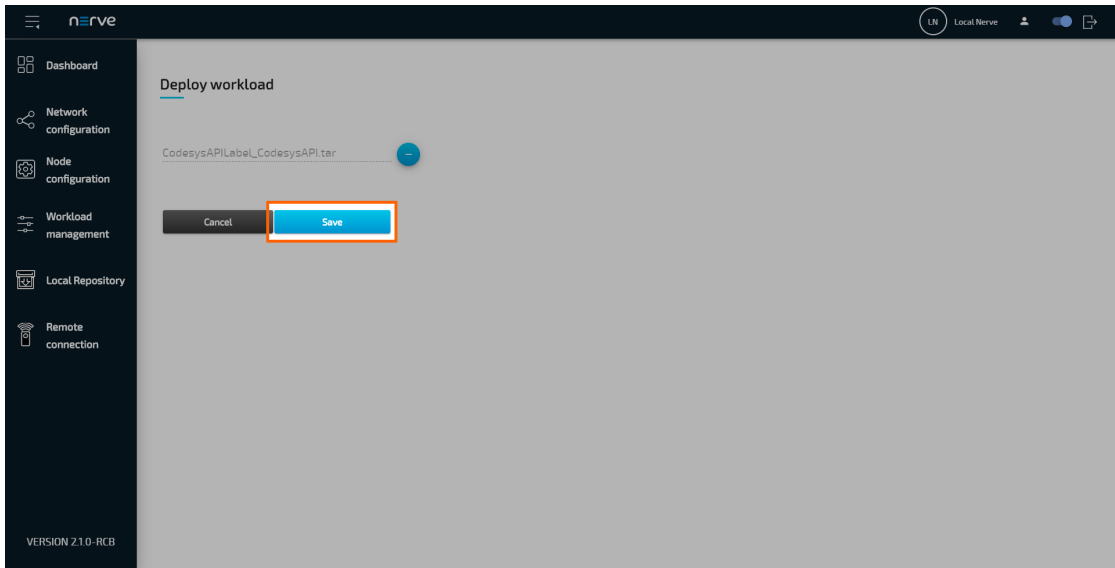
Workloads that have been provisioned in the Management System can be exported and then deployed directly through the Local UI. Refer to the workloads chapter on how to [export a workload](#) before following the instructions below.

1. Export a workload in the Management System.
2. Switch to the Local UI.
3. Select **Workload management** in the navigation on the left side.
4. Click the plus tile in the middle.



5. Select the plus symbol to open the file browser.
6. Add the TAR file containing the exported workload.

7. Select **Save** to deploy the workload.



## Setting a local repository

If desired, the required files for deploying workloads can be stored in a local repository that does not require internet connection. In doing so, the workload image files for deployment are taken from a user defined repository instead of the workload repository in the Management System.

Refer to [Exporting a Workload](#) for information on how to obtain the workload images. Also note that every workload needs to be provisioned in the Management System once before the image files can be transferred to a local repository.

A web server that services static files is required. Popular web servers like Nginx or Apache HTTP Server can be used, as well as a Network-attached storage (NAS) device. Recommended are:

- the [NodeJS http-server](#) for Linux
- the built-in Internet Information Services (IIS) for Windows

The IIS for Windows needs to be activated first in newer versions of Windows like Windows 8 and Windows 10.

1. Open the **Control Panel**
2. Navigate to **Control Panel > Programs > Programs and Features**.
3. Select **Turn Windows features on or off**
4. Tick the checkbox next to **Internet Information Services**.  
Note that ticking the checkbox will only enable the components required to publish a web site. Expand the folder to select other needed features.
5. Select **OK**.

With this the IIS can be used on a Windows machine. Take the following steps to enable the hosting of the workload images. The steps below are not carried out in detail as they are only serving as a guideline.

1. Remove all files from the document root. The default location of the document root is C:\inetpub\wwwroot.
2. Copy the TAR files of the workloads that have been [exported from the Management System](#) to the document root.
3. Enable [directory listing](#).
4. Test the web server by opening <http://localhost> in a web browser.

5. Test if a computer can be reached from another computer in the network.

#### NOTE

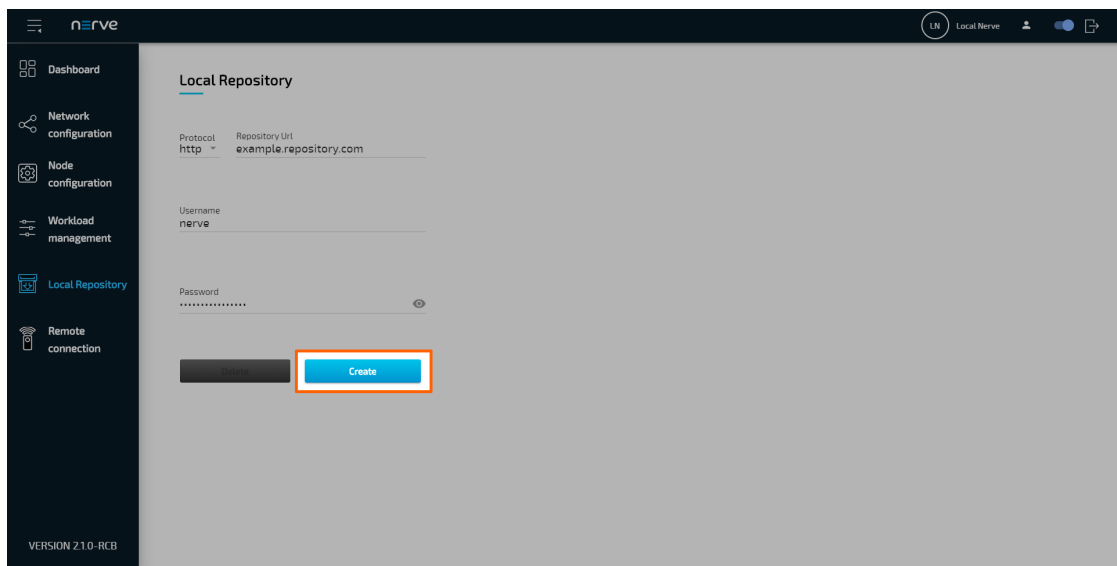
Note that all served files can be accessed from all computers connected to the same network.

For more information, refer to the [IIS documentation](#). Follow the instructions below to enable a local repository for a node:

1. Select **Local Repository** in the navigation on the left.
2. Enter the following information:

<b>Protocol</b>	Select the protocol for communication with the local repository: <b>http</b> or <b>https</b> .
<b>Repository URL</b>	Enter the URL through which the web server hosting the workload images can be accessed.
<b>Username</b>	If login credentials have been defined for the web server, enter the username for accessing the web server hosting the workload images.
<b>Password</b>	If login credentials have been defined for the web server, enter the password for accessing the web server hosting the workload images.

3. Select **Create** to finish the setup.



If a local repository is defined in the Local UI, the Management System will look for the workload images in the local repository first when a workload is deployed to this node. If the workload image is not present in the local repository, the workload repository in the Management System will be used instead. To revert back to using the repository of the Management System, select **Local Repository** in the navigation on the left and select **Delete**.

## Managing remote connections

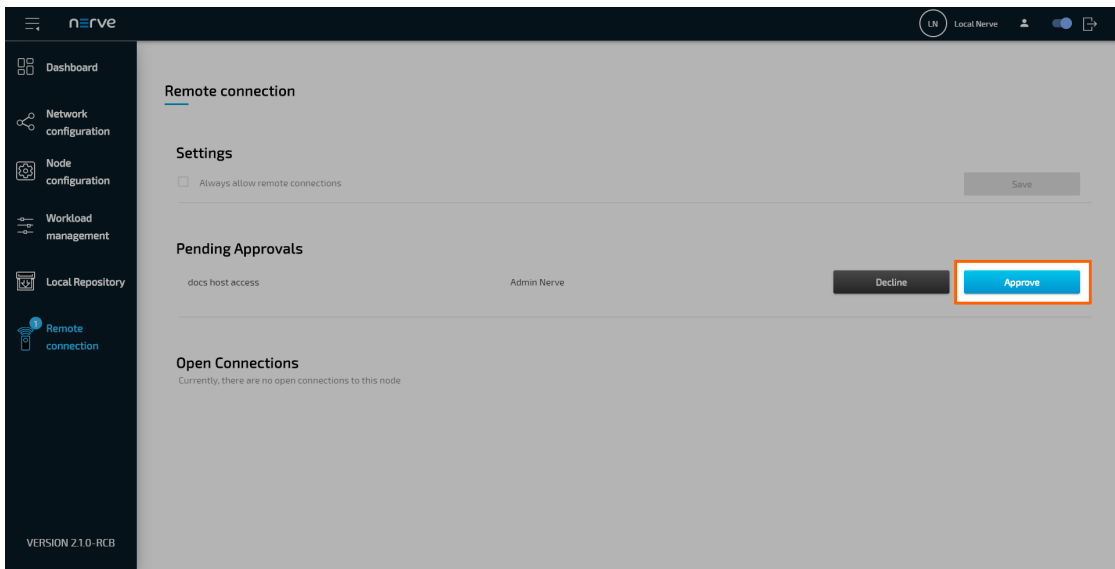
The behavior of nodes for remote connections is defined when a remote connection is configured in the Management System. Refer to [Remote connections](#) for more information.

	Tick the checkbox next to <b>Always allow remote connections</b> to automatically approve all remote connection requests.
<b>Settings</b>	This setting is only valid when <b>Local acknowledgment</b> is set to <b>Yes</b> in the remote connection configuration in the Management System.
<b>Pending Approvals</b>	Incoming connection requests are displayed here. Select <b>Approve</b> to accept the remote connection. Clicking <b>Cancel</b> denies the incoming request.  This setting is only valid when <b>Local acknowledgment</b> is set to <b>Yes</b> in the remote connection configuration in the Management System.
<b>Open Connections</b>	Connection requests that have been approved and are currently open are listed here. To the left, the name of the remote connection is shown. This name is also shown under <b>Remotes</b> in the Management System. In the middle, the user that established the remote connection is displayed. To the right, there is a button to terminate the remote connection.

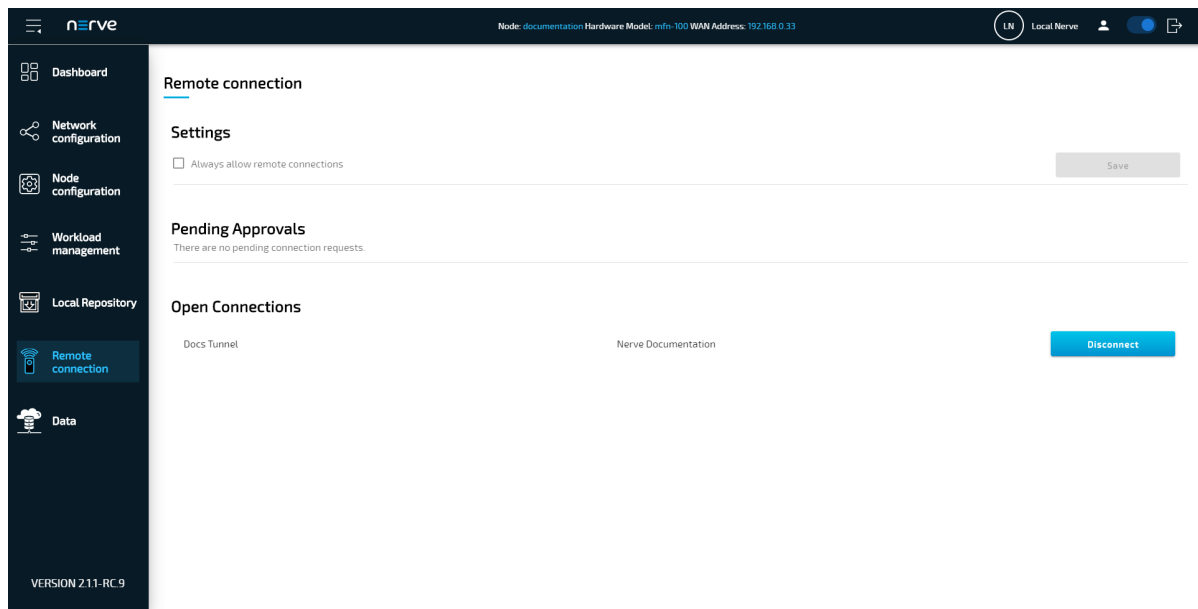
### Approving a remote connection

Local approval for remote connections can be configured in the Management System. Set **Local acknowledgment** to **Yes** when configuring a remote connection. When set to **Yes**, every remote connection has to be approved in the Local UI before it can be established. If approval for an incoming remote connection is pending, a notification bubble appears next to **Remote connections** in the navigation on the left.

1. Select **Remote connections** in the navigation on the left.
2. Search for incoming remote connections under **Pending Approval**.
3. Click **Approve** on the right for the remote connection that shall be established.



Once approved, the open connection will be displayed under **Open Connections** below. Shown are the name of the remote connection and the user that is currently using the remote connection. If the same remote connection is used by multiple users, multiple entries of the same remote connection are shown with different users.



## Nerve Management System

## Nerve Management System

The Nerve Management System is a web-based service that permits management of Nerve nodes that are registered. It can be used to:

- Monitor nodes
- Deploy and control workloads on a node
- Manage workloads

The Management System communicates through pop-up notifications in the upper-right corner. Success messages are displayed in green boxes. Red boxes are used for error messages, while light blue boxes are used for general notifications.

#### NOTE

Google Chrome or Firefox Version 63 or later are recommended for the usage of the Management System.

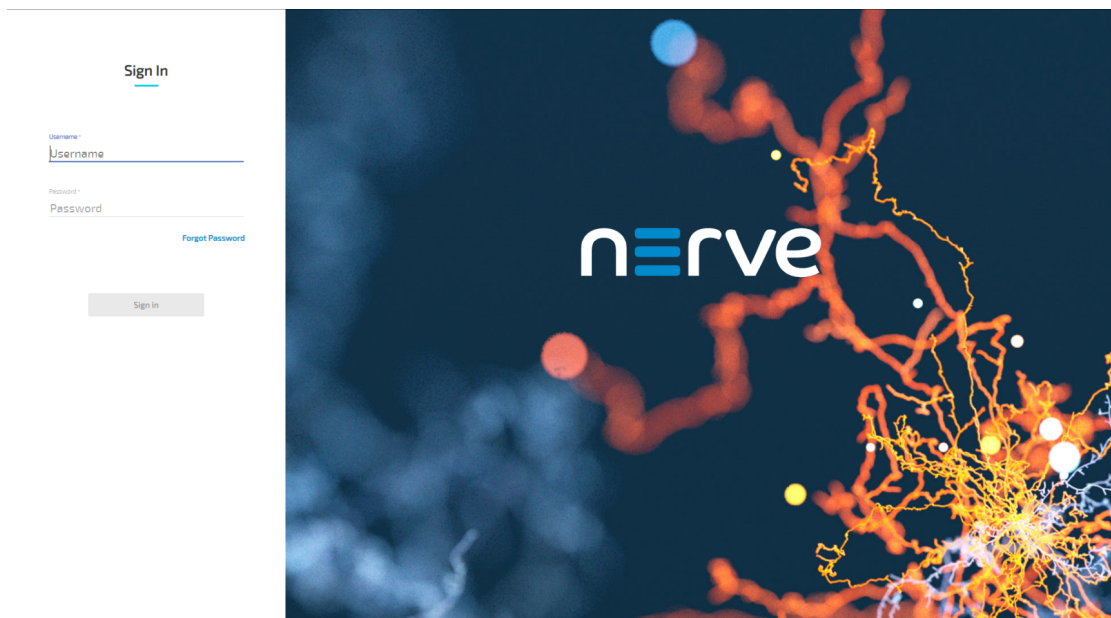
## Logging in to the Management System

The Management System is either hosted by TTTech Industrial or installed on premise. The URL of the Management System changes accordingly. Find the URL of the Management System in the customer profile if it is hosted by TTTech Industrial.

1. Go to the URL of the Management System.
2. Log in with the credentials for the Management System.

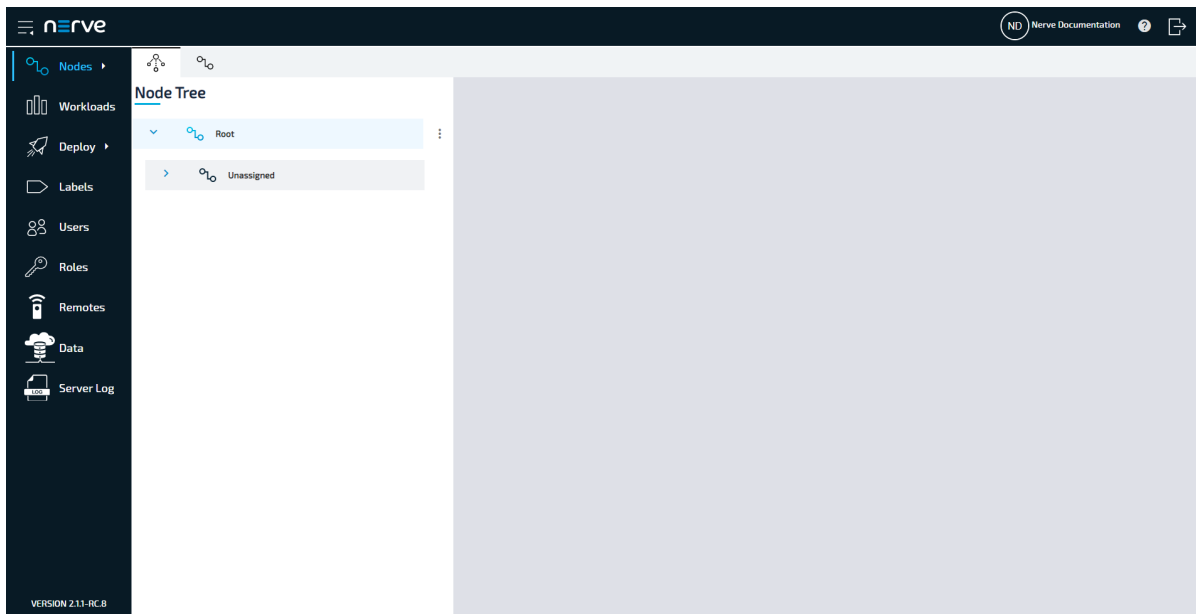
#### NOTE

The initial login credentials of the Management System can be found in the customer profile. If a customer profile has not been part of the delivery, contact a sales representative or TTTech Industrial customer support at [support@tttech-industrial.com](mailto:support@tttech-industrial.com).



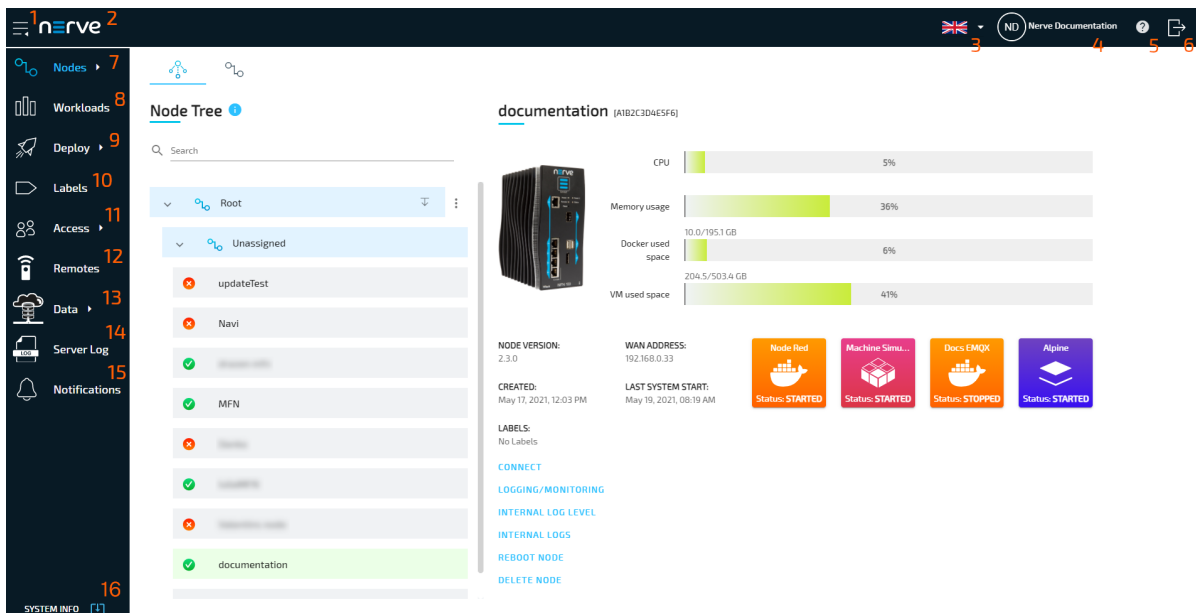
The node tree is the landing page of the Management System. Note that after 30 minutes of inactivity, users are logged out automatically.





## Menu structure of the Management System

As a landing page the Management System shows the node tree. Navigate the site by using the menu bar on the left side.



Item	Description
<b>Burger Menu (1)</b>	Clicking here will expand and collapse the menu bar on the left side. The expanded version of the menu adds names to the menu icons.
<b>Nerve logo (2)</b>	Return to the node tree by clicking the Nerve logo.
<b>Language selection (3)</b>	Click here to select the interface language. Available languages are English and Korean.
<b>User display (4)</b>	Access the user details of the active user from here.

Item	Description
<b>Documentation link (5)</b>	Click the question mark to open the Nerve documentation in a new browser tab.
<b>Log out button (6)</b>	Clicking this icon will log out the active user from the Management System.
<b>Nodes (7)</b>	<p>The nodes menu has two sub-menus in the navigation on the left and two tabs in the default view.</p> <p><b>Tabs</b></p> <ul style="list-style-type: none"> <li>• <b>Node Tree Tab</b> This is the default view of the Management System. It displays all registered nodes in a node tree. Add, delete and move tree elements freely here. It mainly serves an organizational purpose and does not impact the functionality of the nodes. Nodes and workloads can be managed in the node details view, which is reached through the node tree.</li> <li>• <b>Node List Tab</b> Selecting the nodes tab displays a list of all available nodes that have been registered in the Management System. Add and remove nodes, as well as edit their details here.</li> </ul> <p><b>Sub menus</b></p> <ul style="list-style-type: none"> <li>◦ <b>Updates</b> Available updates for nodes can be found here.</li> <li>◦ <b>Update Log</b> Past node updates can be viewed here. When an update is performed, this view also shows the current progress of the update.</li> </ul>
<b>Workloads (8)</b>	<p>All workloads that have been provisioned in the Management System are listed here. Workloads can be added, deleted, disabled and edited through this menu. New versions of workloads can also be added here.</p> <p>Workloads that have been provisioned previously can be deployed using this menu. However, there are two sub menus available here:</p>
<b>Deploy (9)</b>	<ul style="list-style-type: none"> <li>• <b>Log</b> Find a list of all workloads that have been deployed or are currently being deployed. Also, view details of all deployments and delete log entries from the list.</li> <li>• <b>Dry Run</b> A dry run is a simulation of a workload deployment. It allows to test out if the deployment of a workload could be successful. However, note that a successful dry run is not a guarantee for successful deployment.</li> </ul>
<b>Labels (10)</b>	This is a list of all labels that have been defined in the Management System. Add, delete, edit and merge labels here.

Item	Description
<b>Access (11)</b>	<p>Find settings for user, permission and access management in this menu.</p> <ul style="list-style-type: none"> <li>• <b>Users</b> This is the user management menu. It lists all registered users and allows to edit profiles and add new users.</li> <li>• <b>Roles</b> Manage user roles and and permissions here.</li> <li>• <b>LDAP</b> Synchronize the Nerve access management with an existing LDAP server here.</li> </ul>
<b>Remotes (12)</b>	Currently active remote connections are displayed here. Refer to <a href="#">Remote Connections</a> for more information.
<b>Data (13)</b>	Access the instance of the Nerve Data Services in the Management System here. This feature is disabled by default and must first be activated by an admin user. Refer to <a href="#">Nerve Data Services</a> for more information.
<b>Server Logs (14)</b>	Look at internal server logs here. These internal server logs are aimed at Nerve service technicians in case of error and failure. Data is stored with Elasticsearch and the logs are visualized with the Kibana application.
<b>Notifications (15)</b>	Set user-defined system notifications here. A notification can be set to display either before or after logging in using this menu.
<b>SYSTEM INFO (16)</b>	<p>Clicking here leads to two menus: <b>Available Updates</b> and <b>Usage Reports</b>.</p> <ul style="list-style-type: none"> <li>• <b>Available Updates</b> This is the first visible menu after selecting <b>SYSTEM INFO</b>. It displays information about the current version of the Management System and available updates of the Management System.</li> <li>• <b>Usage Reports</b> Find usage information in regards to the Management System that is required for proper billing here. Refer to <a href="#">Usage reports</a> below for more information.</li> </ul>

#### NOTE

The available features of the Management System depend on the user role. Refer to [Roles and Permissions](#) for more information.

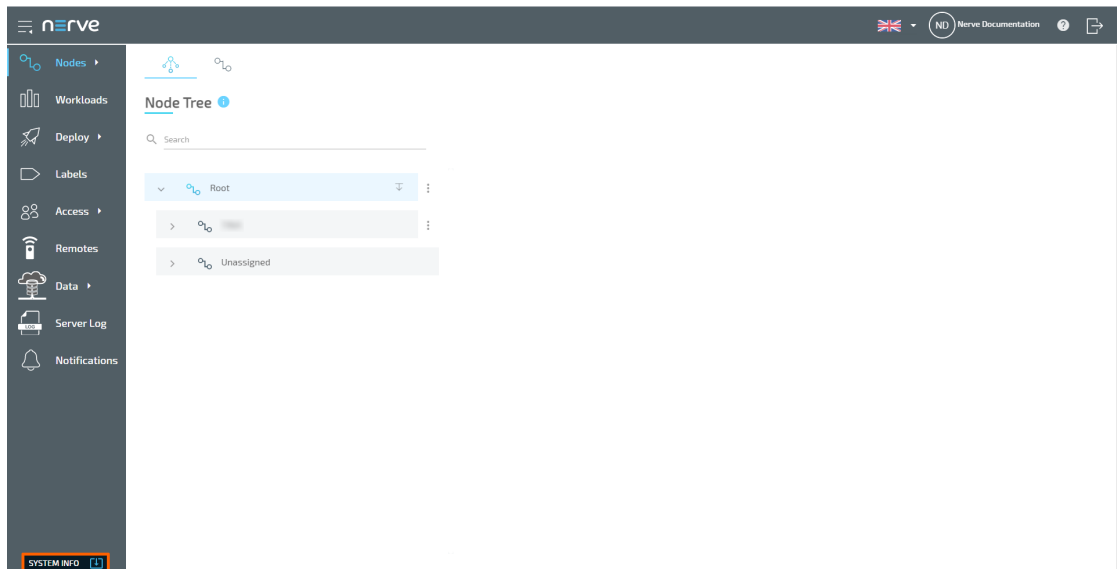
## Updating the Management System

A download icon next to **SYSTEM INFO** appears in the lower-left corner of the Management System if an update to the Management System is available. A backup of the current Management System is made when an update is performed and all workloads are stopped. Reverting to the previous version is possible.

## NOTE

- Do not attempt to update the Management System from 2.3.0 to 2.3.1. This needs to be done manually by a Nerve Service technician. Contact TTTech Industrial customer support at [support@tttech-industrial.com](mailto:support@tttech-industrial.com).
- Note that reverting the Management System to version 2.2.X is not possible after an update to version 2.3.0.

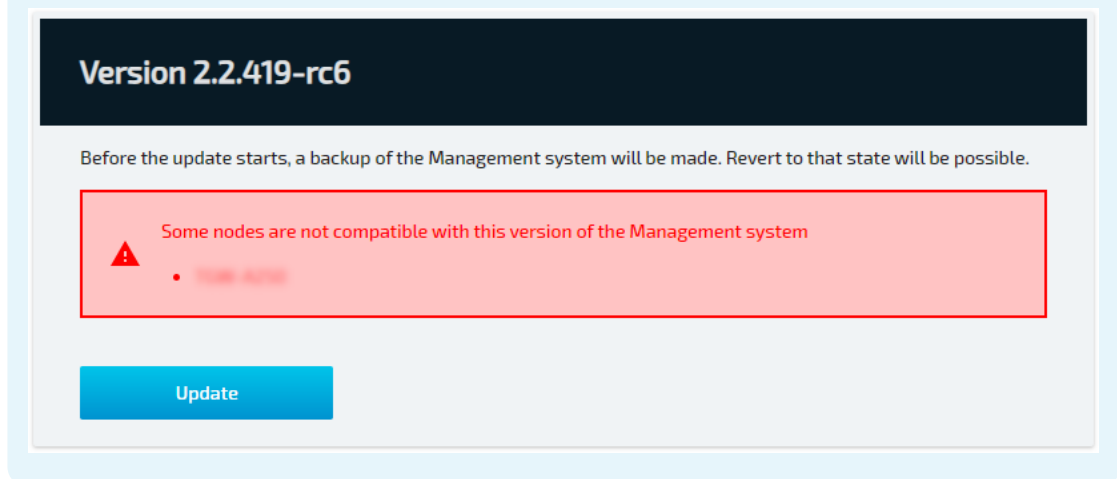
1. Select **SYSTEM INFO** in the lower-left corner.



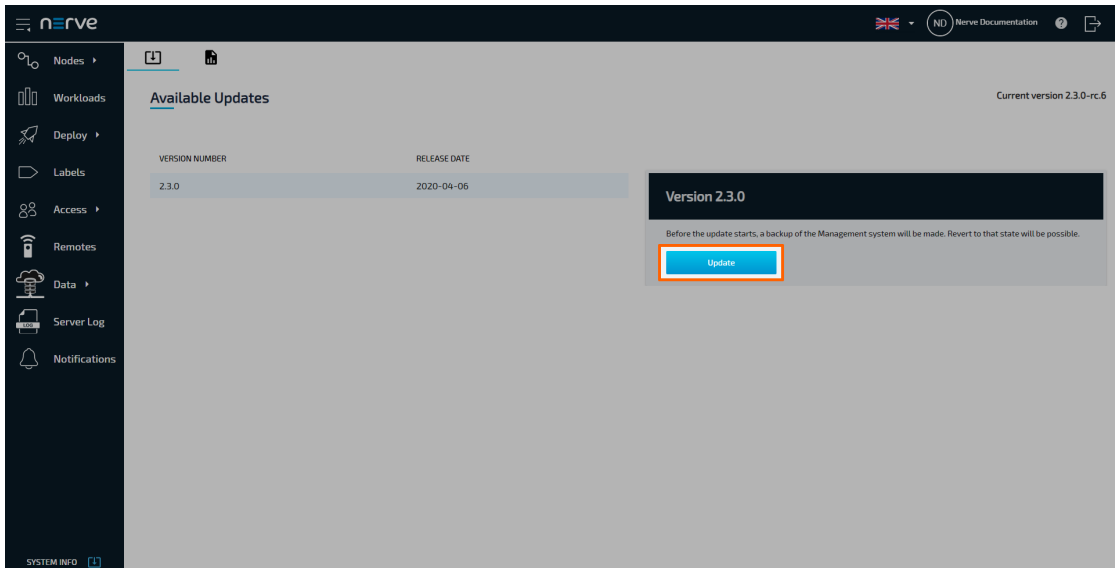
2. Select the new version of the Management System from the list. The current version of the Management System is displayed on the right.

## NOTE

Registered nodes that will be incompatible after the update are shown on the right. These nodes cannot be used with the updated Management System if they are not updated to a newer node version. Refer to [Updating a Node](#) for more information.



3. Select **Update** on the right.



4. Select **YES** in the pop-up window.

Note that updates of the Management System have to be performed in order. Skipping a version is not permitted.

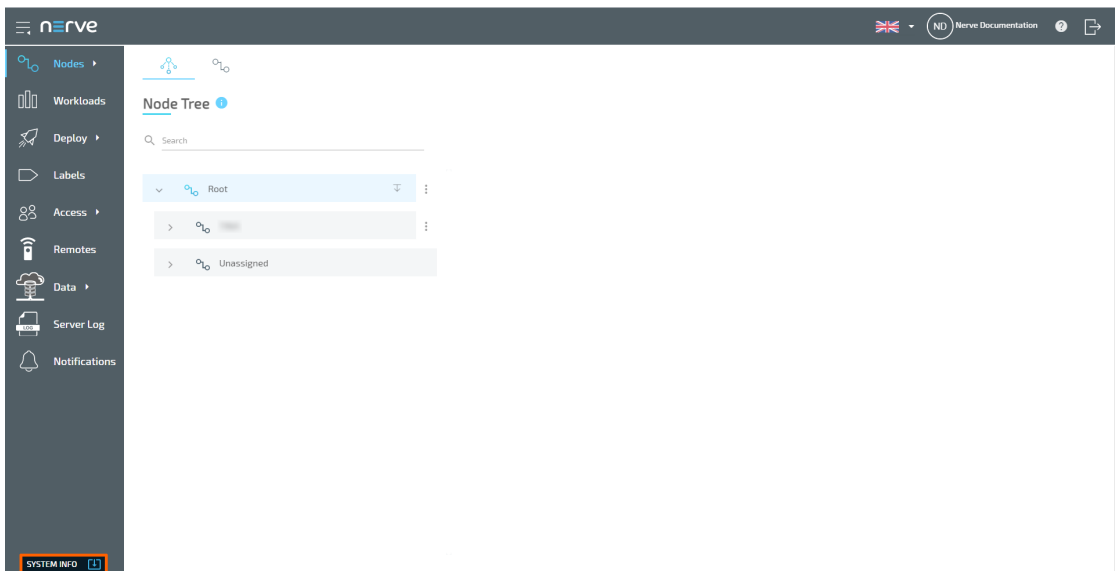
## Reverting the Management System to the previous version

The previous version of the Management System will be marked with an icon in the list of available updates. Note that any new data created after the update will be lost when reverting to the previous version of the Management System. Downgrading the Management System to an older version is not possible.

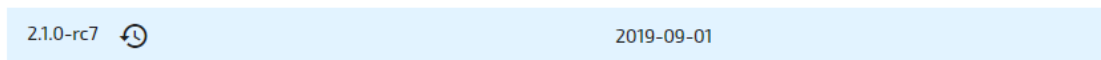
### NOTE

Note that reverting the Management System to version 2.2.X is not possible after an update to version 2.3.0.

1. Select **SYSTEM INFO** in the lower-left corner.



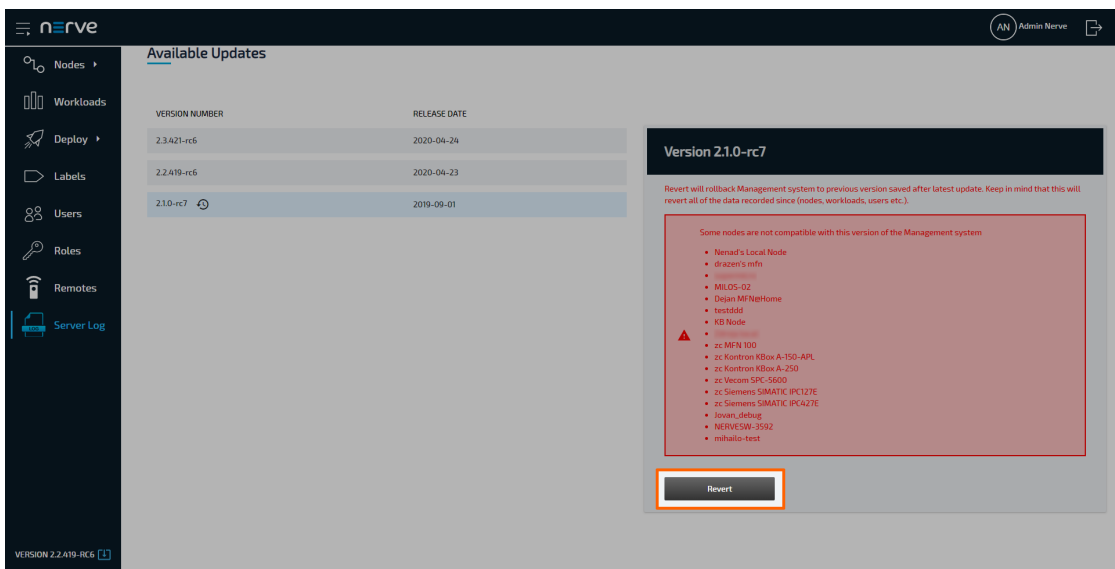
2. Select the previous version of the Management System that is marked with an icon.



### NOTE

Registered nodes that will be incompatible after the update are shown on the right. These nodes cannot be used with the reverted Management System.

3. Click **Revert** on the right.

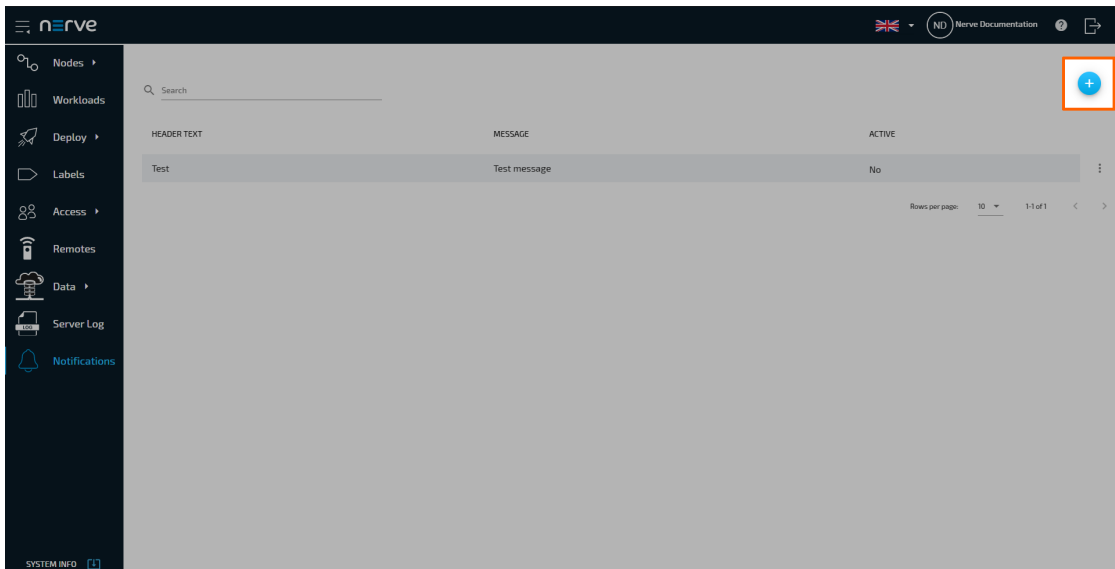


4. Select **YES** in the pop-up window.

## Setting system notifications

User-defined system notifications can be set for the Management System. Notifications are displayed as a pop-up, consisting of header and body text. Multiple notifications can be defined. However, only one message at a time can be active. Notifications are displayed for every user either before or after logging in to the Management System. No notifications are set by default.

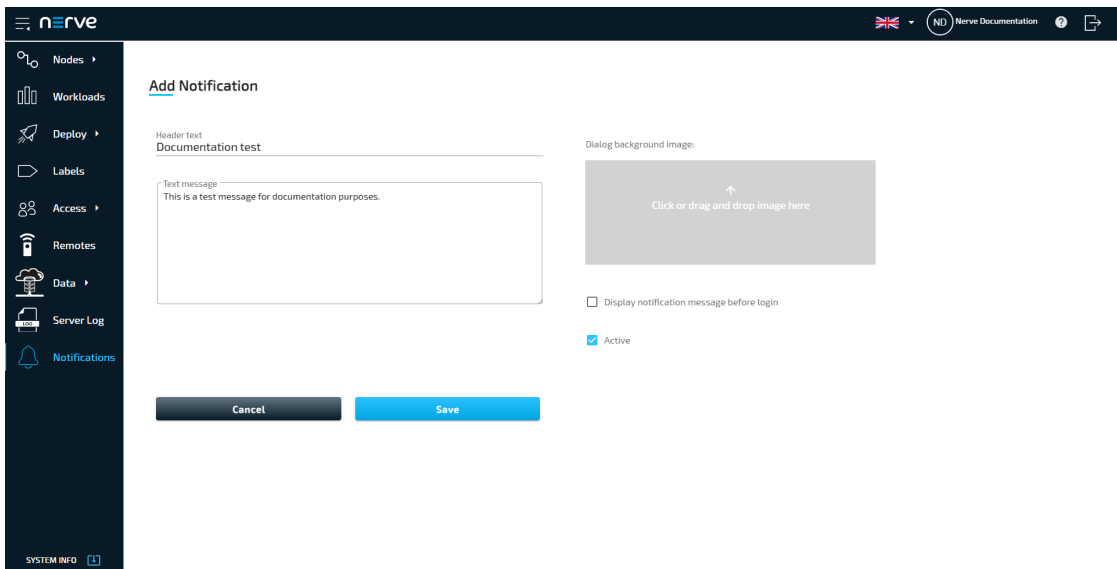
1. Select **Notifications** in the navigation on the left.
2. Select the plus symbol in the upper-right.



3. Enter the following information:

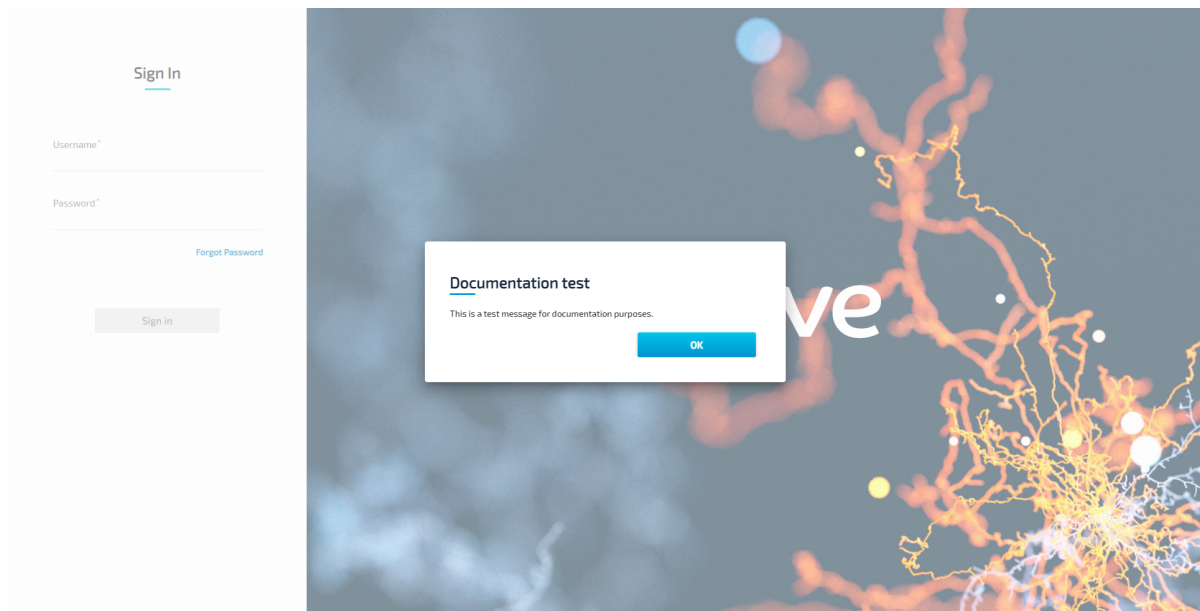
Setting	Description
<b>Header text</b>	Enter the header text of the notification pop-up. This text has a limit of 60 characters.
<b>Text message</b>	Enter the body text of the notification pop-up.
<b>Digital background image</b>	Add a background image to the notification pop-up. Supported formats are JPEG, PNG, JPG, JFIF, PJPEG and PJP. Note that the image will be stretched to fit the entire pop-up, including header and body.
<b>Display notification message before login</b>	Tick the checkbox here to have the notification displayed before logging in. If this checkbox is not ticked, the notification is displayed after logging in.
<b>Active</b>	Tick the checkbox here to set the current notification as active. Doing so deactivates all other notifications. Untick the checkbox to deactivate the notification. Note that this also deactivates notifications in general until another notification is set to active.

4. Select **Save** to save the notification settings.



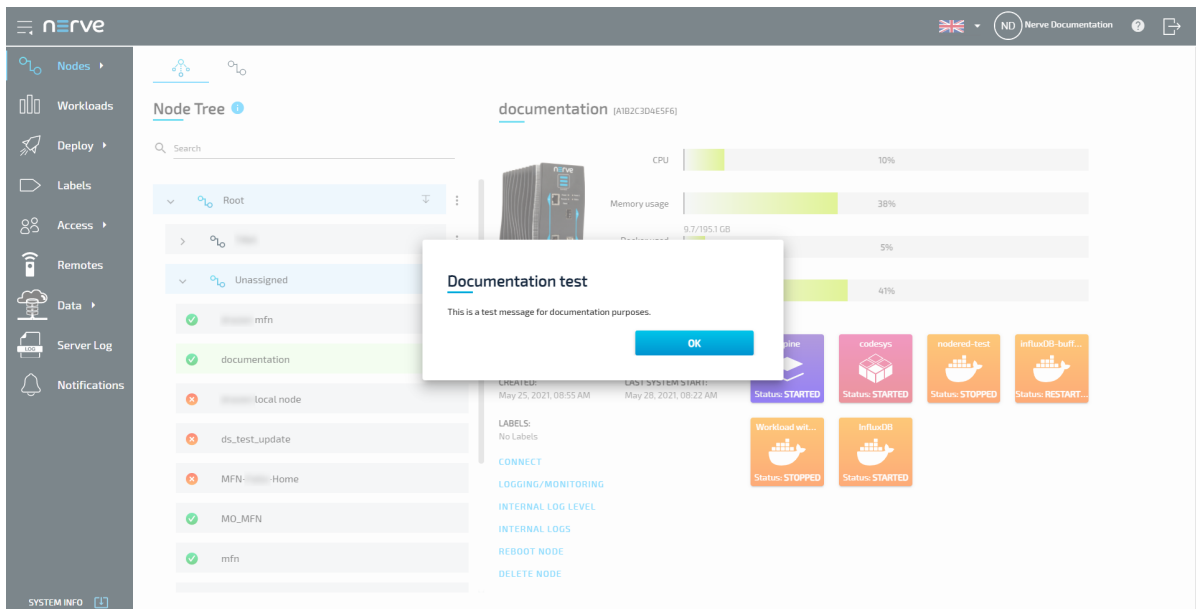
The notification now appears in the list of notifications. The list is split into three columns: **HEADER TEXT**, **MESSAGE** and **ACTIVE**. Select **DELETE** from the ellipsis menu to the right of a notification to delete a notification. Refer to the screenshots below for examples of notifications in use:

### Before logging in



### After logging in

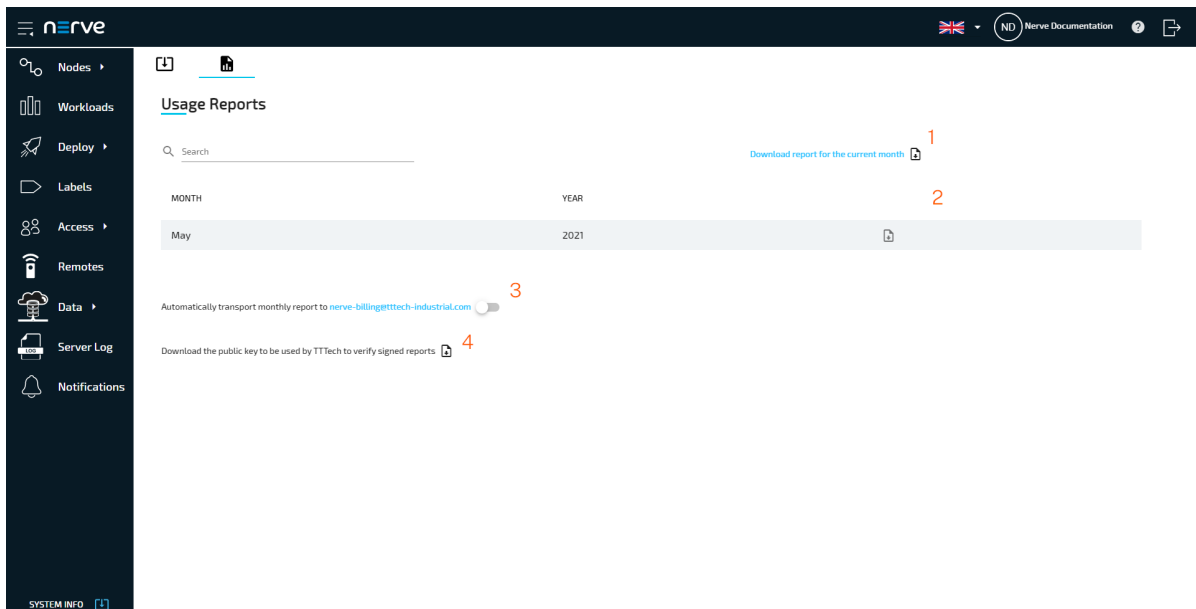




## Usage reports

Usage reports contain information about the usage of the Management System that is required for proper billing by TTTech Industrial. They need to be delivered monthly according to the contractual agreement. This can be done manually or automatically. Usage reports are compiled for each month and digitally signed for authenticity.

Select **SYSTEM INFO** in the lower-left corner and then select the usage reports tab to reach the usage reports menu.



Item	Description
<p><b>Current report download (1)</b></p>	<p>Download a temporary report of the current month. It shows the following information:</p> <ul style="list-style-type: none"> <li>• <b>Date</b> This is the year and month at the time of the usage report export.</li> <li>• <b>Created</b> This is the date the usage report was created.</li> <li>• <b>Management System instance</b> This is the URL of the Management System.</li> <li>• <b>Number of connected nodes</b> This is the number of registered nodes which have had a connection to the Management System instance within the reference month.</li> <li>• <b>Number of updated nodes</b> This is the number of registered nodes in this Management System instance that have been updated to a newer version within the reference month.</li> <li>• <b>Updated nodes</b> This shows details of the node updates that have been performed. The information collected is the serial number, the previous version of the node, the version the node has been updated to and the time of the update.</li> </ul> <p>Note that this report serves as information for the user.</p>
<p><b>List of past months (2)</b></p>	<p>This is a list of the months the Management System has been in use. A new entry is created after every month including a finished usage report that can be downloaded. Select the download symbol on the right side of every entry to download the report of the listed month. The content of the reports are the following:</p> <ul style="list-style-type: none"> <li>• <b>Date</b> This is the year and month the usage report refers to.</li> <li>• <b>Created</b> This is the date the usage report was created.</li> <li>• <b>Management System instance</b> This is the URL of the Management System.</li> <li>• <b>Number of connected nodes</b> This is the total number of registered nodes which have had a connection to the Management System instance within the reference month.</li> <li>• <b>Number of updated nodes</b> This is the total number of registered nodes in this Management System instance that have been updated to a newer version within the reference month.</li> <li>• <b>Updated nodes</b> This shows details of the node updates that have been performed. The information collected is the serial number, the previous version of the node, the version the node has been updated to and the time of the update.</li> </ul> <p>The reports are digitally signed to ensure authenticity. They are also stored indefinitely without any data retention policy.</p>
<p><b>Automatic report transfer (3)</b></p>	<p>Toggle this slider to automatically transfer the usage report each month. If the automatic transfer is not activated, the usage reports need to be sent manually.</p>

Item	Description
<b>Public key download (4)</b>	Ignore this if the Management System is hosted by TTTech Industrial. Download the public key that is required to verify the signature of the usage reports. Note that this key needs to be sent to TTTech Industrial once if the Management System is not hosted by TTTech Industrial or if usage reports are sent manually.

## Nodes

The nodes menu has two sub-menus in the navigation on the left and two tabs in the default view.

The node tree tab is the default view of the Management System. It displays all registered nodes in a node tree. It mainly serves an organizational purpose and does not impact the functionality of the nodes. Nodes and workloads can be managed in the node details view, which is reached through the node tree.

Selecting the node list tab displays a list of all available nodes that have been registered in the Management System. Add, remove, and edit nodes here.

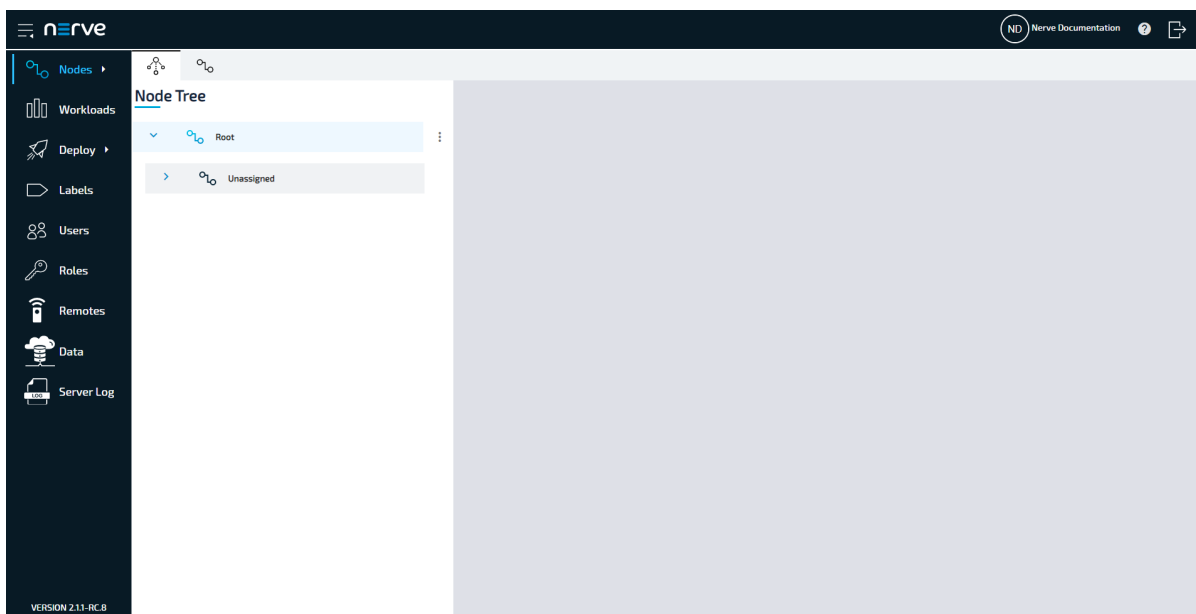
Available updates for nodes can be found in the **Updates** sub-menu. Performed node updates can be viewed in the **Update Log**. When an update is performed, the update log also shows the current progress of the update.

## Node tree

The node tree is the landing page of the Management System. It presents an overview of all nodes that are connected to the Management System, embedded into tree view elements. Being mainly a means of organization, it has no impact on the functionality of the nodes and their workloads. Select **Nodes** in the navigation on the left. Then select the node tree tab



on the right.



There is only one element under the root after the initial setup: **Root > Unassigned**. All nodes that are registered in the Management System are placed in the **Unassigned** element by default. From there they can be moved to new elements that have to be created first.

## Creating a new element in the node tree

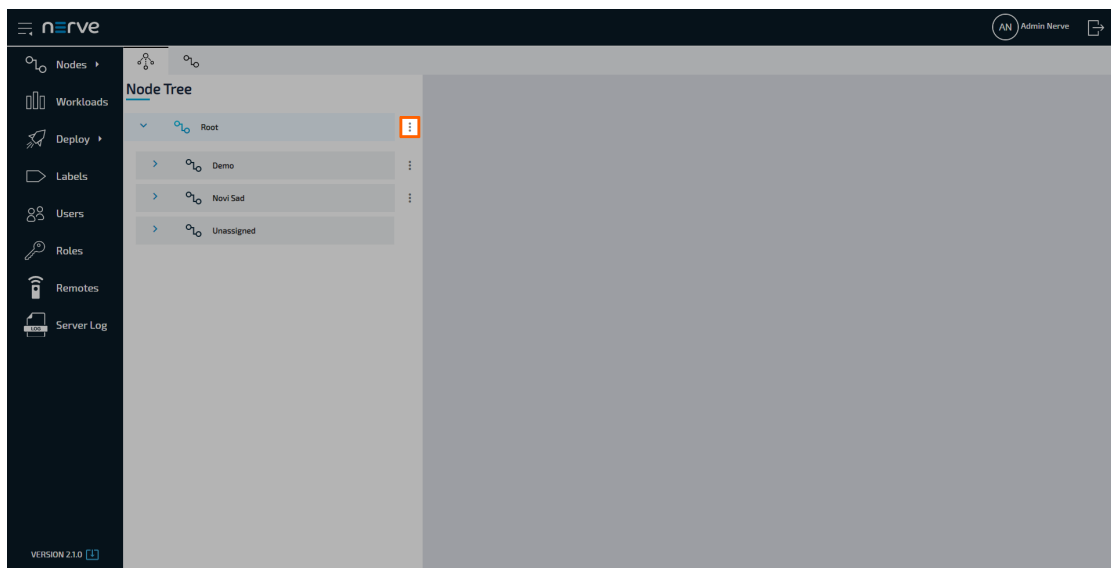
Before moving a node out of the **Unassigned** element, create a new element in the node tree. Elements in the node tree exclusively serve an organizational purpose.

1. Select **Nodes** in the navigation on the left.
2. Select the node tree tab

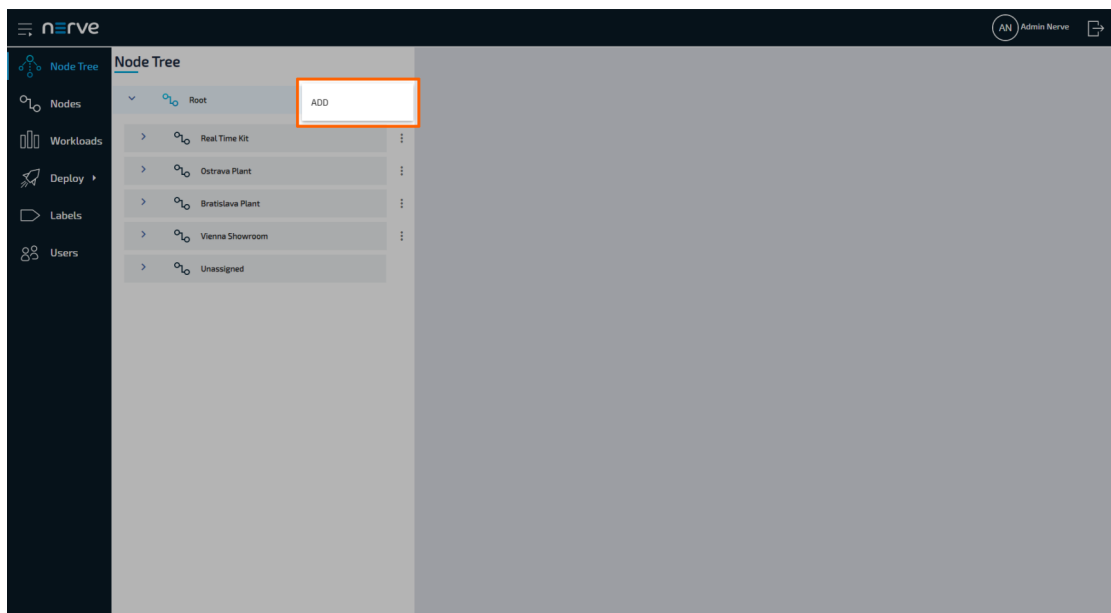


on the right.

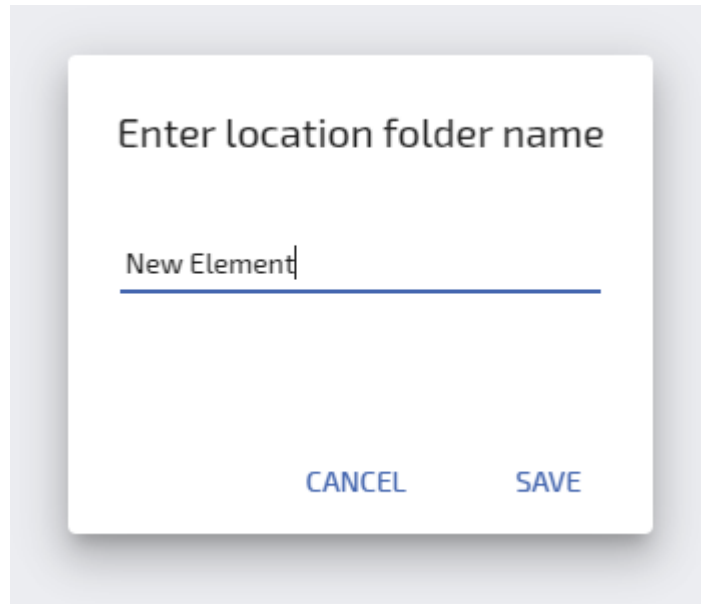
3. Select the ellipsis menu to the right of **Root** in the node tree.



4. Click **Add** in the overlay that popped up.



5. Enter the name of the new element under **Location name**.



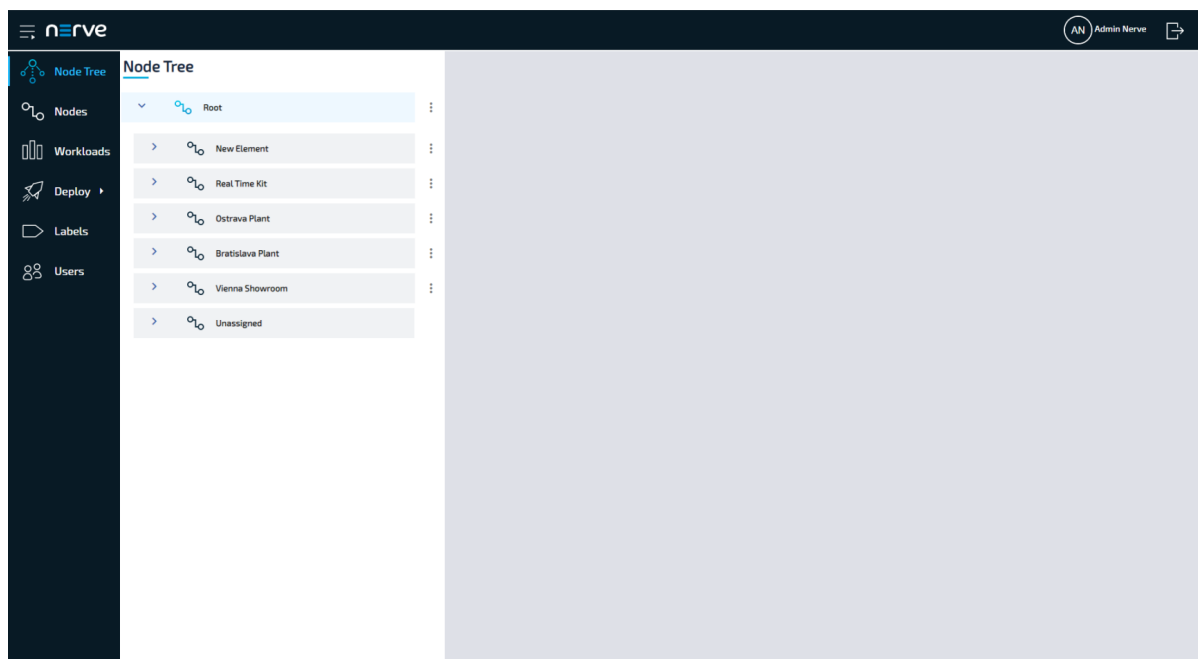
A modal dialog box with a white background and a light gray border. The title is "Enter location folder name". Below the title is a text input field containing "New Element" with a blue underline. At the bottom of the dialog are two buttons: "CANCEL" and "SAVE", both in blue text.

6. Click **Save**.
7. Select **APPLY CHANGES (n)** in the upper-right corner of the node tree.

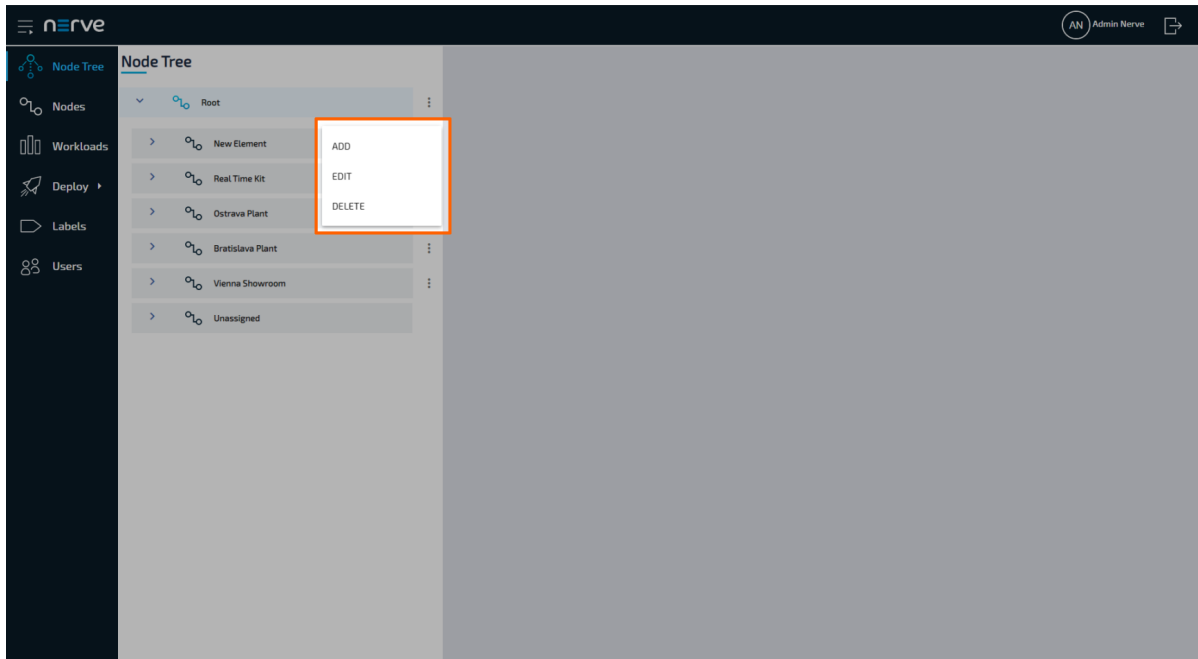
#### NOTE

(n) is a placeholder for the number of changes made to the node tree. If three changes have been performed, (3) will be displayed in the button above the node tree.

The new element now appears under the **Root** element.



Create more elements and freely modify the node tree. To the right of every created element, there is an ellipsis menu that opens up an overlay. Add additional elements, edit the names of elements or delete the elements here.



#### NOTE

- The order of the tree elements can be modified easily. Drag and drop an element to its new position.
- When a tree element is deleted, all of the nodes inside the element will be moved to **Unassigned**.

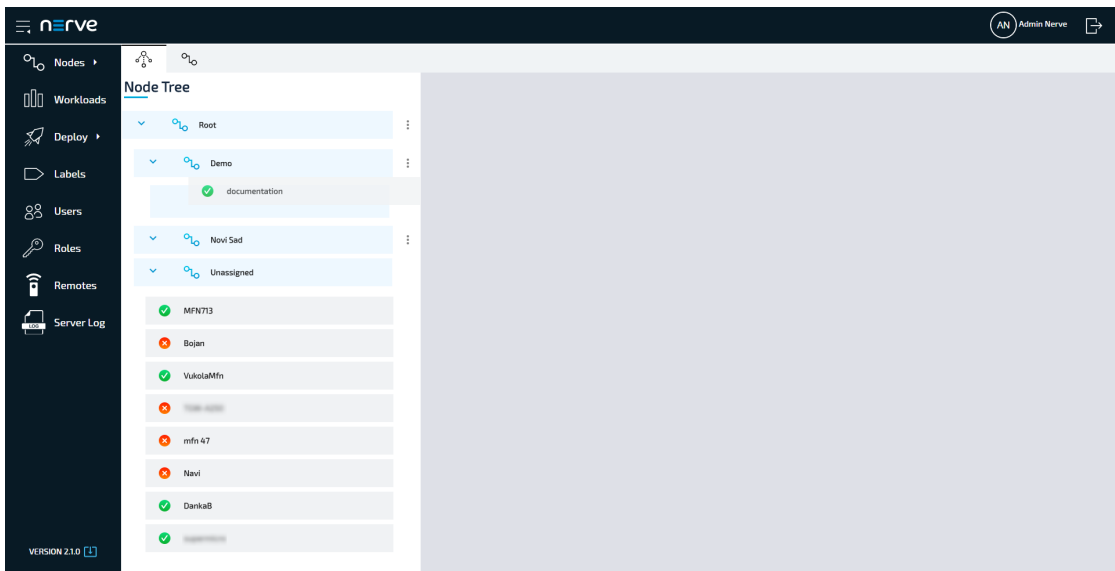
## Moving a node from one tree element to another

Moving nodes in the node tree is very straightforward and intuitive and possible by drag and drop. Make sure that a new tree element is created before attempting to move a node.

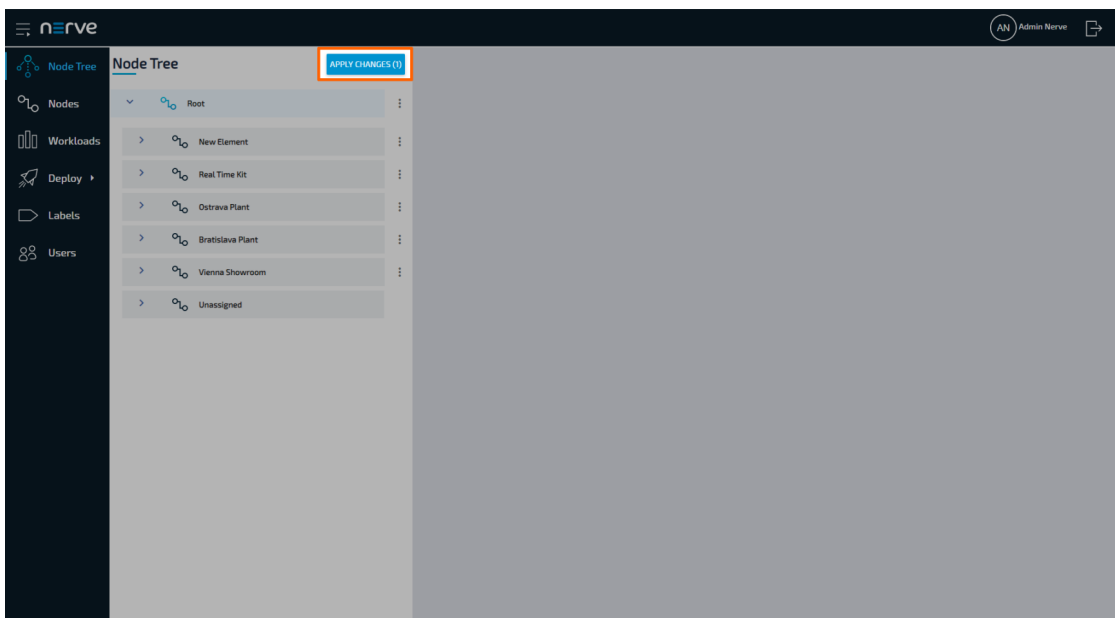
1. Select **Nodes** in the navigation on the left.
2. Select the node tree tab



3. Expand the tree element of the node that will be moved. The default element is **Root > Unassigned**.
4. Choose the node to move.
5. Drag and drop the node to the newly created element. Elements expand automatically once as they are hovered over.



6. Select **APPLY CHANGES (n)** in the upper-right corner of the node tree.



#### NOTE

(n) is a placeholder for the number of changes made to the node tree. If three changes have been performed, (3) will be displayed in the button above the node tree.

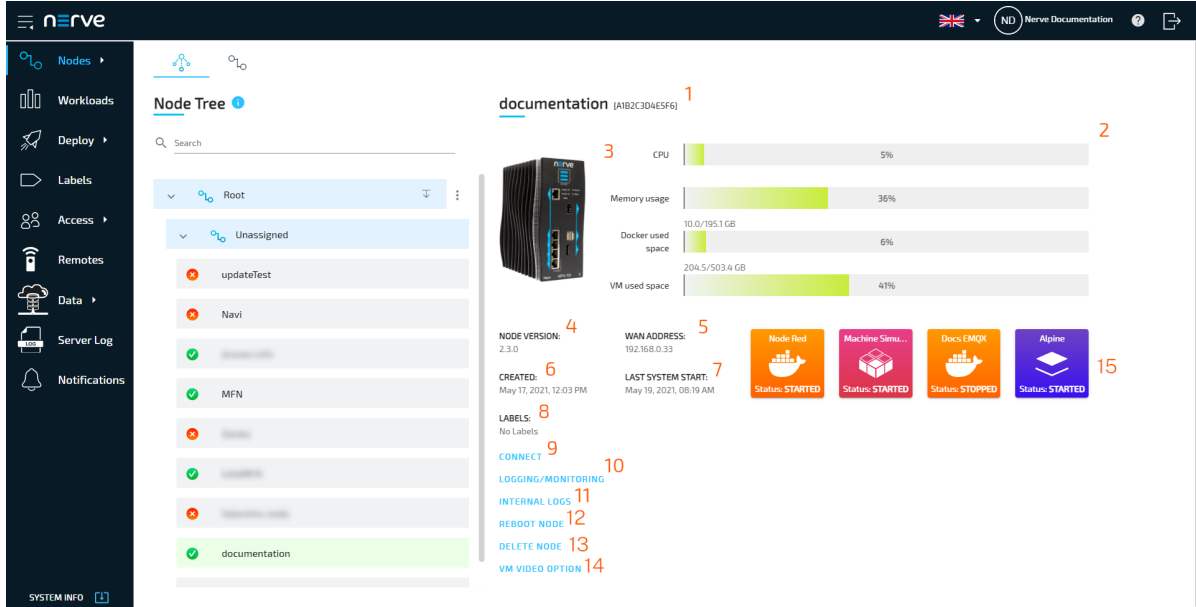
The node has now been moved to the new element.

#### NOTE

A node cannot be moved back manually to **Unassigned** once it has been moved to another element.

## Managing nodes in the node tree

Once nodes are registered in the Management System, view their details next to the node tree. To view the details of a node, select the node name or symbol.



Item	Description
<b>Node name and serial (1)</b>	Here the name and serial number of the node are shown. The serial number is next to the name in brackets.
<b>System metrics (2)</b>	<p>The system metrics that are available in the Local UI dashboard are also visible here:</p> <ul style="list-style-type: none"> <li>• <b>CPU</b> The percentage here shows how much processing power is being used in total at the moment. This includes CPUs that have been assigned to VMs and Docker containers as well.</li> <li>• <b>Memory</b> Similar to CPU usage, the percentage of memory used in total at the moment is shown here. This includes memory that has been assigned to VMs or Docker containers.</li> <li>• <b>Docker used space</b> Two things are shown in this graph: The percentage shows how much of the available space for Docker containers is already used. The value shows the amount of space that is free. Docker containers have their dedicated virtual partition.</li> <li>• <b>VM used space</b> Similar to Docker used space, two things are shown in this graph as well: The percentage shows how much of the available space for virtual machines is already used. The value shows the amount of free space.</li> </ul> <p>Note that deployed Virtual Machine workloads share a logical volume group with the Nerve Base System. Therefore the percentages and values displayed in this graph are in relation to the total amount of space available of the logical volume group.</p>

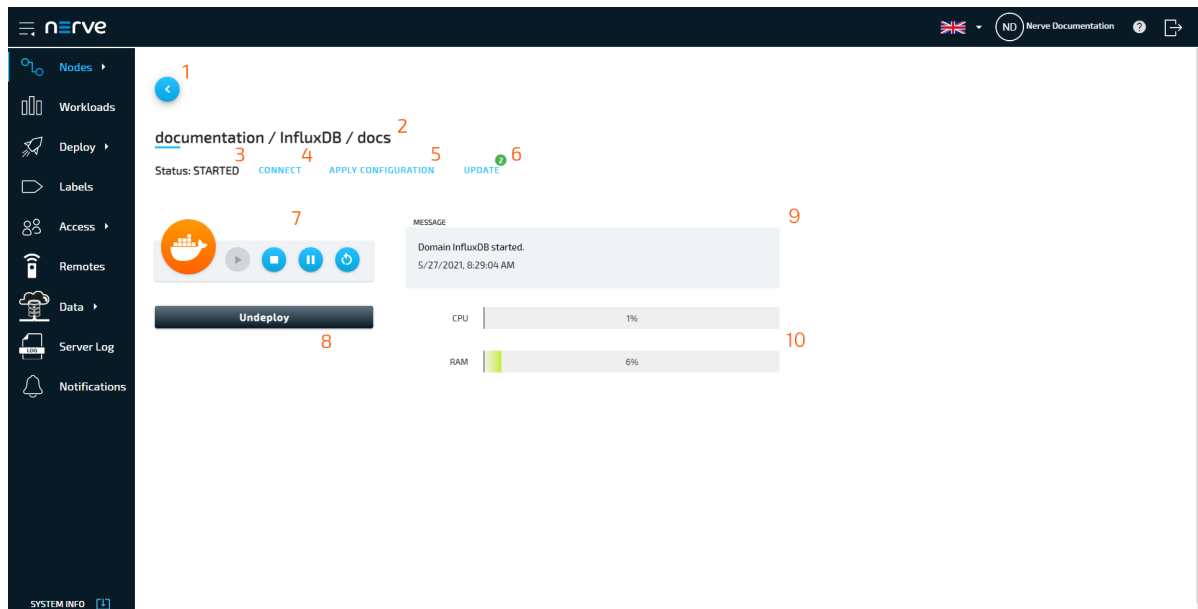


Item	Description
<b>Device image (3)</b>	An image of the hardware model is displayed here according to the device type that was selected when the node was added.
<b>NODE VERSION (4)</b>	The version of the node depending on the product version of Nerve.
<b>WAN ADDRESS (5)</b>	This is the network address of the node that has been assigned by the DHCP server.
<b>CREATED (6)</b>	This is the date when the node was added to the Management System.
<b>LAST SYSTEM START (7)</b>	This is the timestamp of the last node restart.
<b>LABELS (8)</b>	Labels that are assigned to this node are listed here. Labels can be set in the Management System. Refer to <a href="#">Labels</a> for more information.
<b>CONNECT (9)</b>	Clicking here opens an overlay, through which a remote connection to the node can be established. Refer to <a href="#">Remote Connections</a> for more information.
<b>LOGGING/ MONITORING (10)</b>	<p>Select this to open an overlay that allows enabling of logging and monitoring, as well as viewing dashboards. Enable the following settings by toggling the button for each setting:</p> <ul style="list-style-type: none"> <li>• <b>SYSTEM MONITORING</b> Toggle this slider to enable monitoring of the node's resource utilization as a whole.</li> <li>• <b>DOCKER WORKLOAD MONITORING</b> Toggle this slider to enable gathering of metadata about installed Docker containers, as well as monitoring of their resource utilization.</li> <li>• <b>DOCKER WORKLOAD LOGGING</b> Toggle this slider to enable the sending of application logs from Docker workloads on the node to the Management System.</li> </ul> <p>To open the dashboards, select the clickable links next to the sliders. For more detailed information, refer to <a href="#">Logging and monitoring</a>.</p>
<b>INTERNAL LOGS (11)</b>	Clicking here will open a new window and show the system logs of the node. The internal node logs are aimed at Nerve service technicians in case of error and failure. Data is stored with Elasticsearch and the logs are visualized with the Kibana application. The amount of logs can be modified through the log level settings by Nerve service technicians. Contact customer support for more information.
<b>REBOOT NODE (12)</b>	Clicking here will reboot the node after a confirmation dialog. Select <b>Yes</b> to initiate the reboot. All running workloads will be stopped. After the reboot, the workloads will return to their previous state. Running workloads will be started. Stopped workloads will stay stopped.
<b>DELETE NODE (13)</b>	Clicking here removes the node from the Management System. The node needs to be registered again after it has been removed.

Item	Description
<b>VM VIDEO OPTIONS (14)</b>	Note that this feature is only available on request. Configure the video output of Nerve Devices to directly output the interface of deployed virtual machines. Doing so allows operating virtual machines directly at the Nerve Device. Refer to <a href="#">Virtual machine video options</a> below for more information.
<b>Workloads overview (15)</b>	Find workloads that have been deployed to the Nerve Device displayed in tiles here. Selecting these tiles leads to a control area in which the workload can be controlled. If there are no tiles, no workloads have been deployed to the Nerve Device. Refer to <a href="#">Workload control</a> below for more information.

## Workload control

All workloads that have been deployed to the node are displayed in tiles below the node details in the node tree. Clicking these tiles allows control of the respective workload. The control options are minimally different for each workload type. Note that CODESYS workloads can only be controlled from the Local UI.



Item	Description
<b>Back button (1)</b>	Click here to return to the node tree.
<b>Device and workload name (2)</b>	The names of the device, the workload and the release name are displayed here as <b>&lt;devicename&gt; / &lt;workloadname&gt; / &lt;releasename&gt;</b> . The name of the workload version is not displayed.

Item	Description
<p><b>Workload status (3)</b></p>	<p>The current status of the workload is displayed here. The possible statuses are the following:</p> <ul style="list-style-type: none"> <li>• <b>Idle</b> This is the initial state of the workload before it is started.</li> <li>• <b>Creating</b> This is a transitional state of the workload when it is being created on the node.</li> <li>• <b>Starting</b> This is a transitional state when the workload is being started.</li> <li>• <b>Restarting</b> This is a transitional state when the workload is being restarted.</li> <li>• <b>Started</b> The workload is running and operating.</li> <li>• <b>Suspending</b> This is a transitional state when the workload is being suspended.</li> <li>• <b>Suspended</b> The workload has been paused.</li> <li>• <b>Resuming</b> This is a transitional state when the workload is being resumed from the suspended state.</li> <li>• <b>Stopping</b> This is a transitional state when the workload is being stopped.</li> <li>• <b>Stopped</b> The workload has been stopped.</li> <li>• <b>Removing</b> This is a transitional state when the workload is being undeployed.</li> <li>• <b>Error</b> An unknown error has occurred.</li> </ul>
<p><b>CONNECT (4)</b></p>	<p>Clicking here opens an overlay, through which a remote connection to the workload can be established. Refer to <a href="#">Remote Connections</a> for more information.</p>
<p><b>APPLY CONFIGURATION (5)</b></p>	<p>This element only applies to Docker workloads with configuration storage defined. Selecting this allows the upload of configuration files. The configuration files have to be archived and uploaded in a ZIP file. For more information, refer to <a href="#">Applying configuration files to a workload</a> below.</p>
<p><b>UPDATE (6)</b></p>	<p>This element applies to CODESYS and Docker workloads. Select this to open an overlay displaying all available versions of this workload. Note that the notification bubble displays the total number of available versions of this workload. Refer to <a href="#">Updating a deployed workload</a> for more information.</p>

Item	Description
<b>Control panel (7)</b>	<p>The following control options are available:</p> <ul style="list-style-type: none"> <li>• <b>Play</b> If the workload is in a stopped state, clicking <b>Play</b> will start the workload.</li> <li>• <b>Stop</b> If the workload is running, clicking <b>Stop</b> will stop the workload.</li> <li>• <b>Suspend</b> Clicking <b>Suspend</b> will pause the workload. It can be continued by clicking <b>Play</b>.</li> <li>• <b>Restart</b> This will restart the workload.</li> </ul>
<b>Undeploy (8)</b>	<p>Selecting this removes the workload from the node. The tile in the node detail screen disappears. The workload can be deployed again from the deployment menu.</p>

**Item****Description**

The message window displays the latest message the workload has sent out including a time stamp. The type of message that is displayed here depends on the workload.

**Messages for VMs and Docker containers:**

- "Domain creating."
- "ERROR during creating! <errormessage>"
- "Domain starting."
- "ERROR during starting! <errormessage>"
- "Domain <domainname> started."
- "Domain stopping."
- "ERROR during stopping! <errormessage>"
- "Domain <domainname> stopped."
- "Domain suspending."
- "ERROR during suspending! <errormessage>"
- "Domain <domainname> suspended."
- "Domain resuming."
- "ERROR during resuming! <errormessage>"
- "Domain restarting."
- "ERROR during restarting."
- "Domain removing!!!"
- "ERROR during removing."
- "ERROR!!! Domain stopping."

In the messages above, <domainname> is a placeholder for the name of the VM or Docker. In case of Docker containers, <errormessage> signifies a message that is generated by the Docker container if an error occurs.

**Message window  
(9)****Additional set of messages for VMs only:**

- "Failed to connect to hypervisor."
- "Failed to create domain."
- "Domain <domainname> created."
- "Cannot start <domainname> domain because it may already be running!"
- "Failed to resume <domainname> domain!"  
<errormessage>
- "Failed to start domain <domainname>." "  
<errormessage>

In this case, <errormessage> is a message that is fetched from the libvirt library.

**Messages from CODESYS workloads:**

- "Preparing files for installation"
- "Starting CODESYS application"
- "CODESYS application started"
- "Stopping CODESYS application"
- "CODESYS application stopped"
- "Removing CODESYS application file"
- "An unexpected error has occurred. <errormessage>"

Here, <errormessage> is a message that is sent between the node and CODESYS.

Item	Description
<b>Usage statistics (10)</b>	<p>Virtual Machine workloads and Docker workloads have their assigned resources they can use. The use of these resources is displayed with bar graphs:</p> <ul style="list-style-type: none"> <li> <b>CPU (VM and Docker)</b>            The percentage here shows the usage of CPU resources in relation to the assigned CPUs.            Example: A VM is assigned one CPU core out of four and the core is at 75 % usage capacity. The graph will be at 75 %.         </li> <li> <b>RAM (Docker only)</b>            Similar to the CPU usage statistic, the percentage here shows the usage of system memory resources in relation to the assigned memory. If the assigned memory is at a 100 % usage capacity, the graph will be at 100 %.            If no memory has been assigned, the graph will show the percentage of used memory in relation to the total available memory of the host.         </li> </ul>

#### NOTE

Since CODESYS workloads can only be controlled through the Local UI, the workload control screen does not offer any control options. It offers a message window, the option to undeploy the workload and the **CONNECT** button for establishing remote connections.

### Applying configuration files to a workload

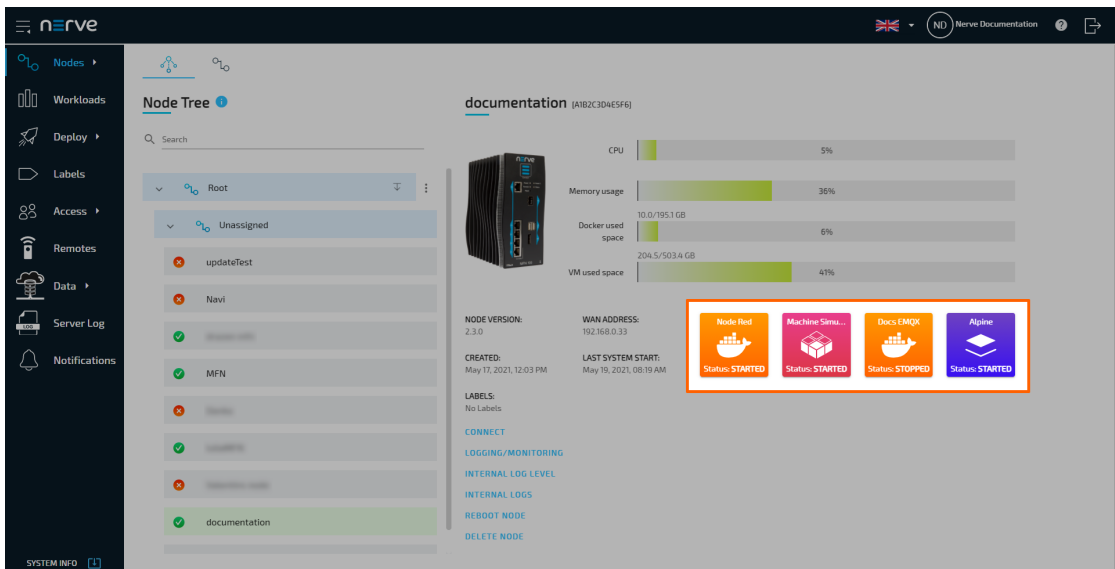
Configuration files can be applied to deployed Docker workloads through the workload control screen. In this case, configuration files are files that the application in the Docker workload needs to perform a specific task. The nature of these files is completely dependent on the application. Therefore, these configuration files need to be prepared by the workload creator beforehand. They also need to be archived as a ZIP file so that they can be applied to a Docker workload using the Management System. Configuration files can be applied to the Docker workload while it is running, stopped or suspended.

Also, note that the workload needs to be properly configured before it is deployed in order to apply a configuration. At least one Docker volume for persistent storage needs to be defined, as well as marking the first volume as configuration storage. For more information, refer to [Settings for Docker workloads](#).

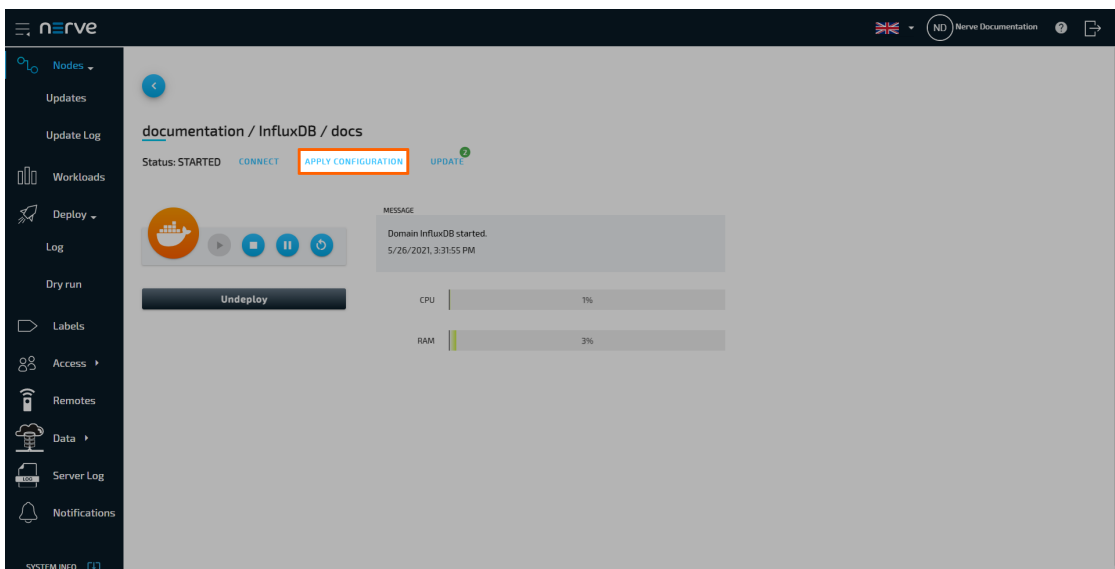
#### NOTE

Note that this functionality is also available in the Local UI.

1. Select **Nodes** in the navigation on the left.
2. Select the node tree symbol.
3. Select a node with a deployed Docker workload from the node tree.
4. Select the tile of the Docker workload in the node details view on the right.



5. Select **APPLY CONFIGURATION**.



6. Select the plus icon to open the file browser.

7. Select the ZIP file containing the configuration files.

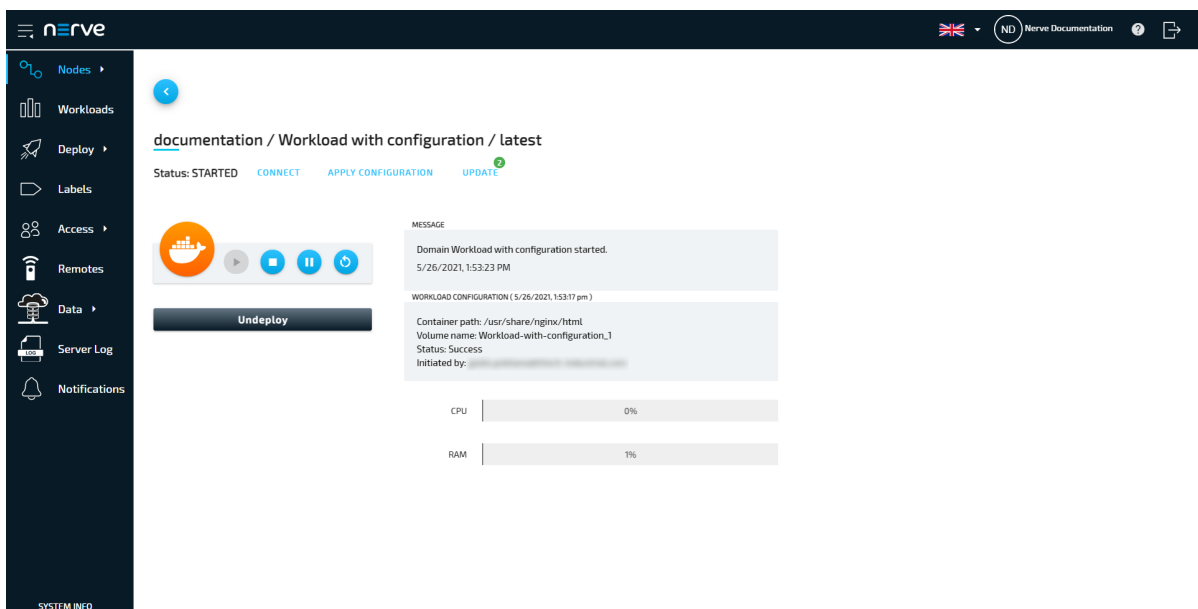
8. Select **Open** to add the ZIP file.



9. Select **Apply**.

A success message will pop up in the upper-right corner once the ZIP file has been applied. Also, a new message window labelled **WORKLOAD CONFIGURATION** is added. It includes the following information:

Item	Description
<b>Timestamp</b>	A timestamp in the format M/DD/YYYY, h:mm:ss am/pm is added next to <b>WORKLOAD CONFIGURATION</b> in the header of the workload configuration message window. This indicates the most recent time configuration files have been applied.
<b>Container path</b>	This is the path of the Docker volume that has been defined in the workload version settings before the deployment of the workload.
<b>Volume name</b>	This is the name of the Docker volume that has been defined in the workload version settings before the deployment of the workload.
<b>Status</b>	This indicates whether the files containing in the ZIP file have been successfully transferred to the Docker container.
<b>Initiated by</b>	Here the user is listed that applied the configuration files.

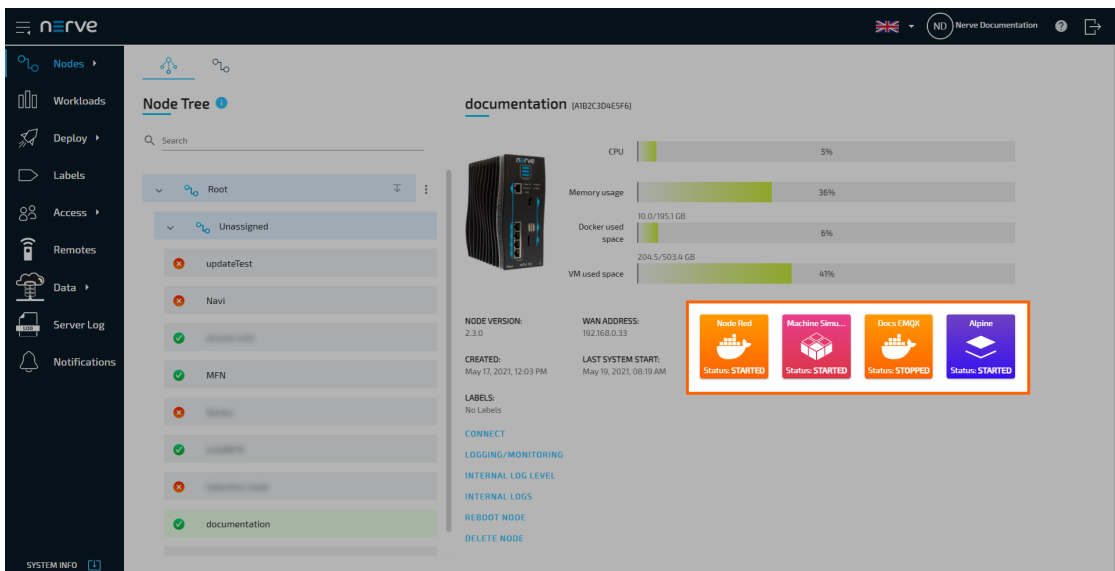




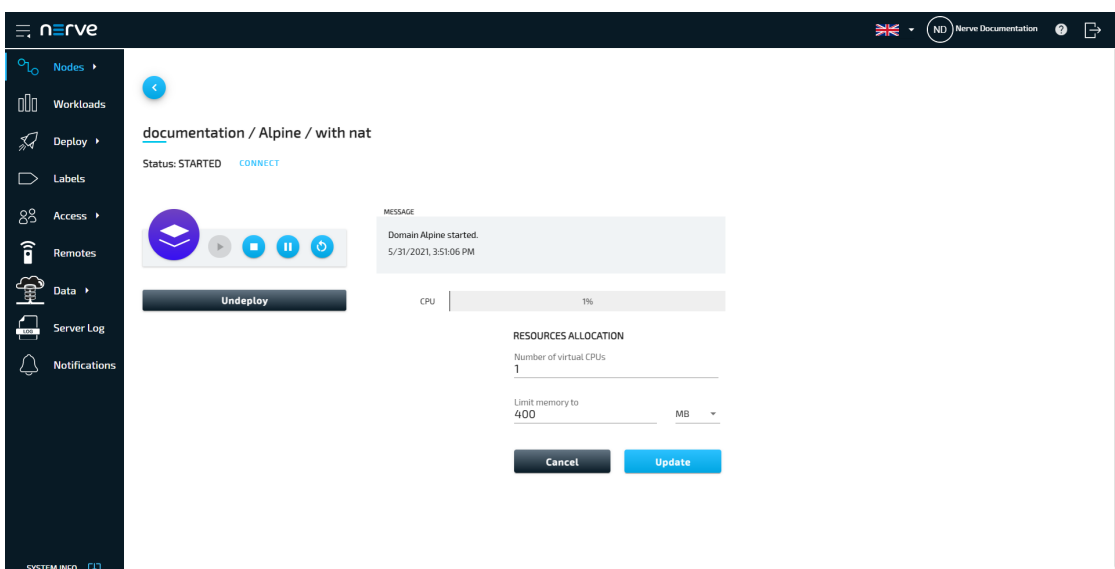
## Changing resource allocation of a deployed Virtual Machine workload

The allocation of resources for Virtual Machine workloads can be modified while the workload is deployed and running on a node. This allows for balancing of resources when several Virtual Machine workloads are running on one node. Note that the workload automatically restarts in order to apply the changes.

1. Select **Nodes** in the navigation on the left.
2. Select the node tree symbol.
3. Select a node with a deployed Virtual Machine workload from the node tree.
4. Select the tile of the Virtual Machine workload in the node details view on the right.



5. Edit the values of **Number of virtual CPUs** and **Limit memory to** to the desired values.



6. Select **Update**.
7. Select **Yes** in the pop-up that appeared.

## Update resource allocation

To update resource allocation the VM must be restarted. Do you want to proceed?

Cancel

Yes

A green notification pops up in the upper-right and the Virtual Machine workload is automatically restarted to apply the changes.

### NOTE

This functionality is also available in the Local UI. Log in to the Local UI and select **Workload Management** in the navigation on the left. Select a Virtual Machine workload and follow the steps above.

## Virtual machine video options

### NOTE

The following feature is not enabled by default. The Nerve Device needs to be configured by service technicians first. Contact a sales representative or TTTech Industrial customer support to request this feature to be enabled. Note that there are two options: regular and grid display. Make sure to state the display mode when requesting the feature to be enabled.

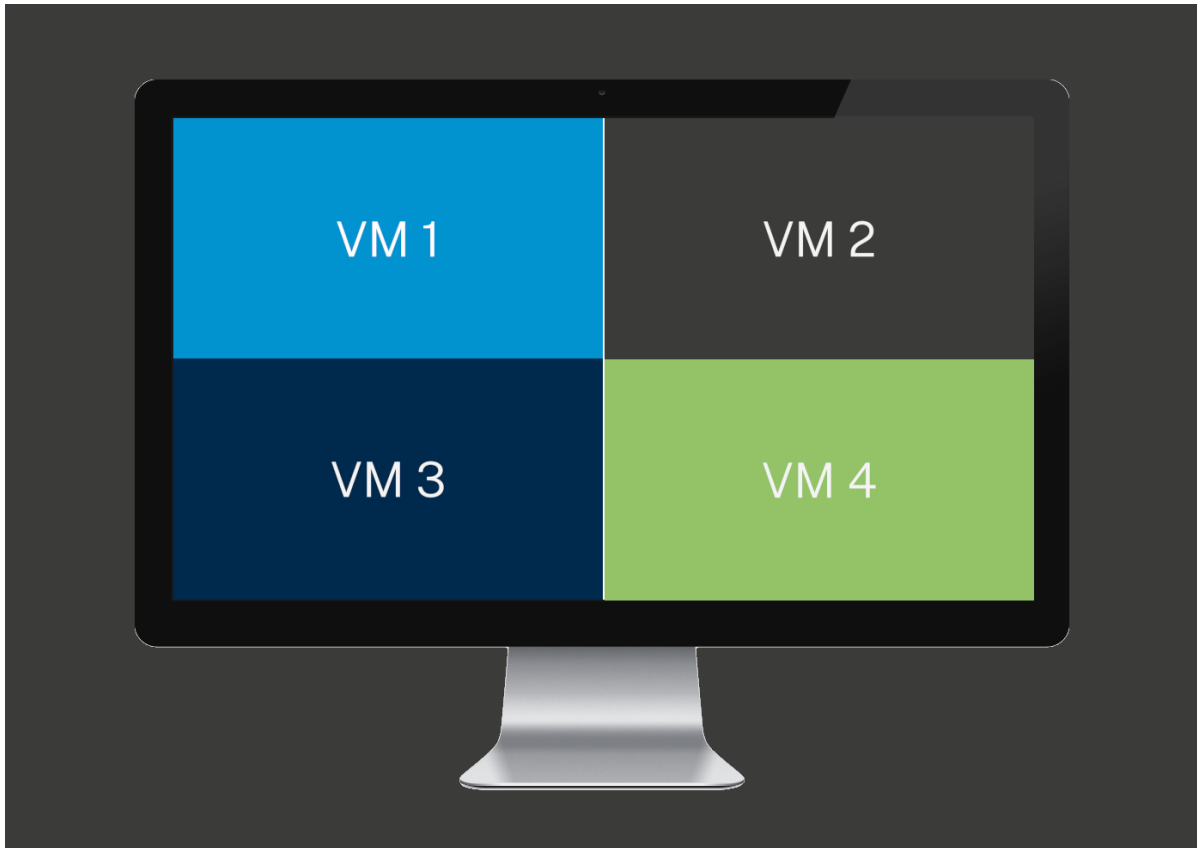
The video output of Nerve Devices can be configured to directly output the interface of deployed virtual machines. This means that by connecting one or more monitors, a keyboard and a mouse, virtual machines can be operated directly at the Nerve Device. There are two display modes that can be chosen: regular and grid display. Note that only one of the two modes can be active. Refer to the images below for more information on possible VM display scenarios:

### One monitor, multiple virtual machines, one at a time



Request regular display mode to use the feature this way. Press F8+Esc to switch between VMs. Determine the order of the VMs by changing the display order settings in the VM video options menu in the Management System.

### One monitor, multiple virtual machines at once



Request grid display mode to use the feature this way. The maximum number of virtual machines that can be displayed is four. Determine the screen position of each VM by changing the display order settings in the VM video options menu in the Management System. All virtual machines can be operated by seamlessly moving the mouse pointer.

### **Two monitors, two virtual machines**



Request regular display mode to use the feature this way. This scenario is only possible with Nerve Devices that have more than one video output. Determine the position of each VM by changing the display order settings in the VM video options menu in the Management System. Both virtual machines can be operated by seamlessly moving the mouse pointer.

### **Two monitors, multiple virtual machines**

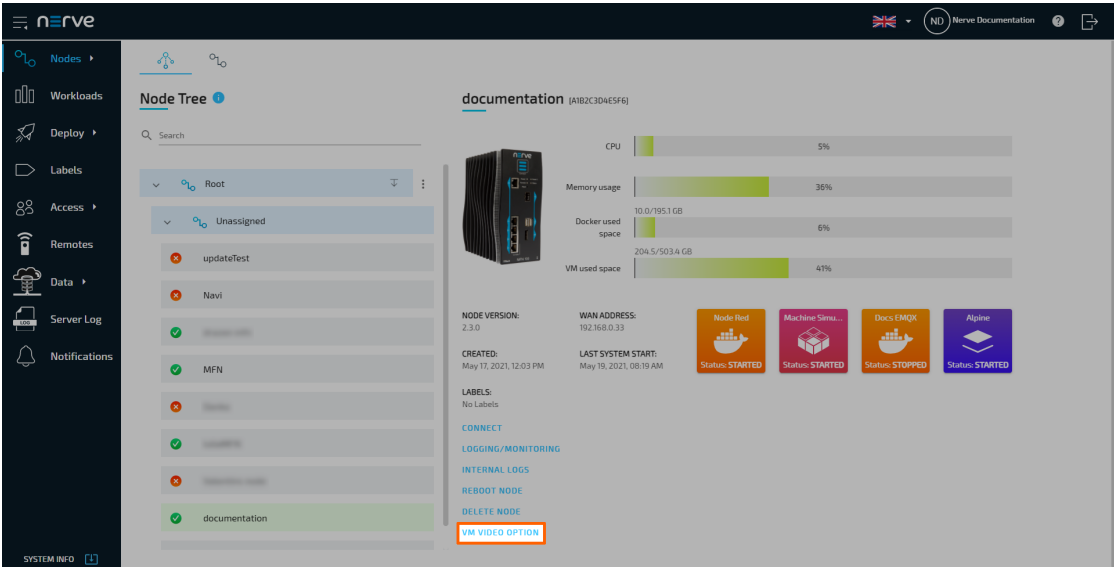


Request grid display mode to use the feature this way. This scenario is only possible with Nerve Devices that have more than one video output. Determine the screen position of each VM by changing the display order settings in the VM video options menu in the Management System. All virtual machines can be operated by seamlessly moving the mouse pointer.

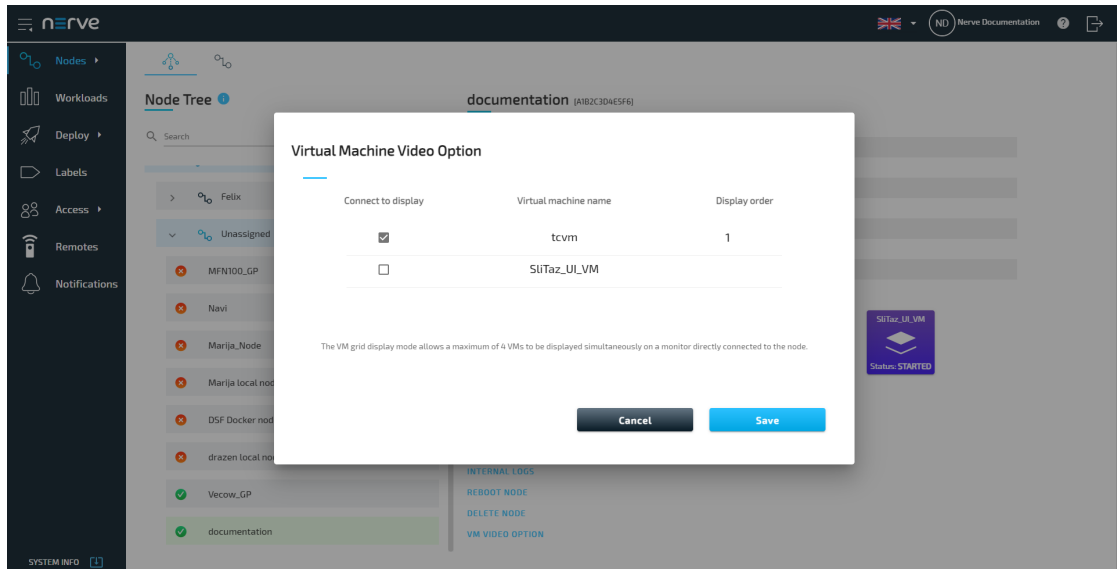
**NOTE**

Connect a monitor to the Nerve Device before the Nerve Device is powered on. This is to avoid having to reboot the node once the VM video options are activated.

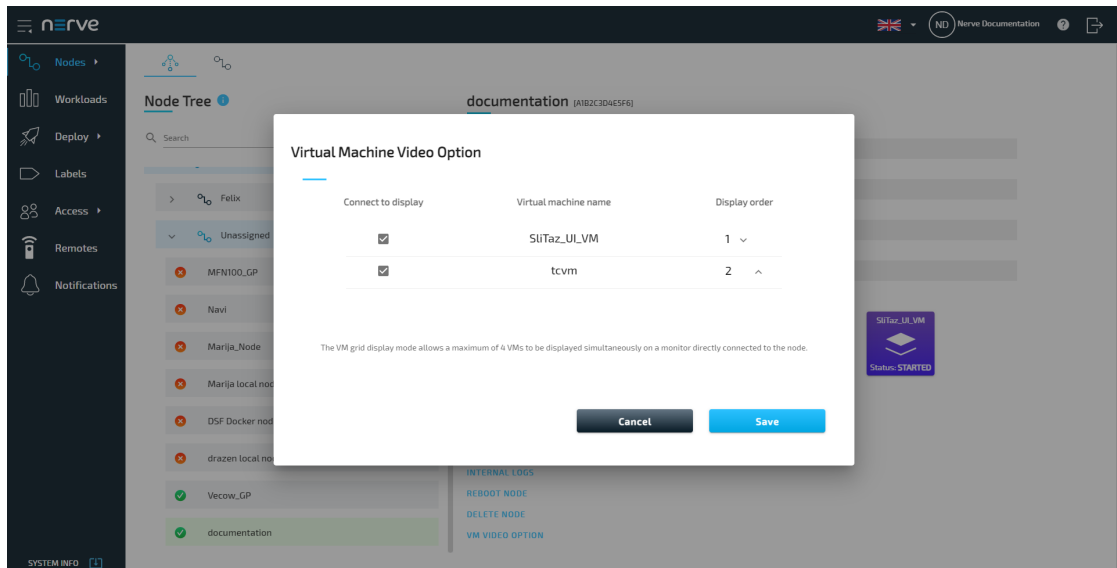
1. Make sure the user has the required permission to view the VM video options. The permission is **VM video output configuration**. Refer to [Editing a role](#) for more information on changing the permissions of a role.
2. Select **Nodes** in the navigation on the left.
3. Select the node that has VM video options enabled from the node tree.
4. Select **VM VIDEO OPTION** in the node details view.



5. Tick the checkbox next to the VMs that shall have video output at the Nerve Device.



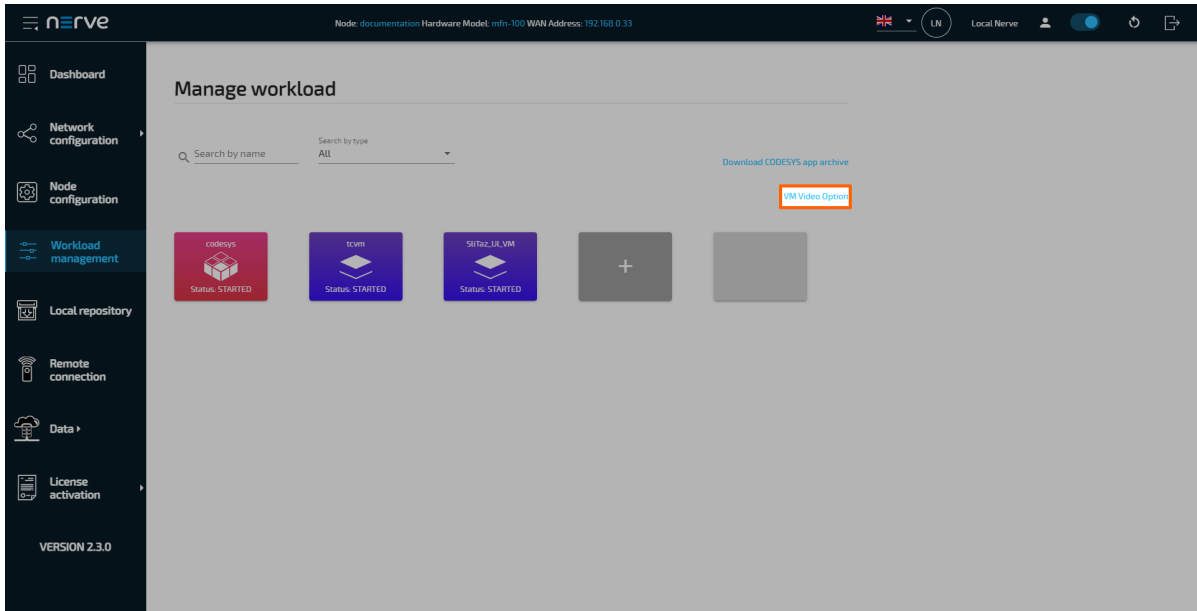
6. Use the arrows under **Display order** to define in which order the VMs shall be displayed.



7. Select **Save**.

The selected virtual machine will now be displayed at the video output of the Nerve Device. However, the VM display resolution might need to be adjusted inside of the VM to make the VM fit its portion of the screen. Note that the node needs to be restarted for the video output to be displayed if the node was already running when the monitor was connected. Also note that having video options enabled for a given virtual machine, makes VNC remote connections to that virtual machine not possible.

This feature is also available in the Local UI. Select **Workload management** in the navigation on the left and then select **VM Video Option**.



## Node list

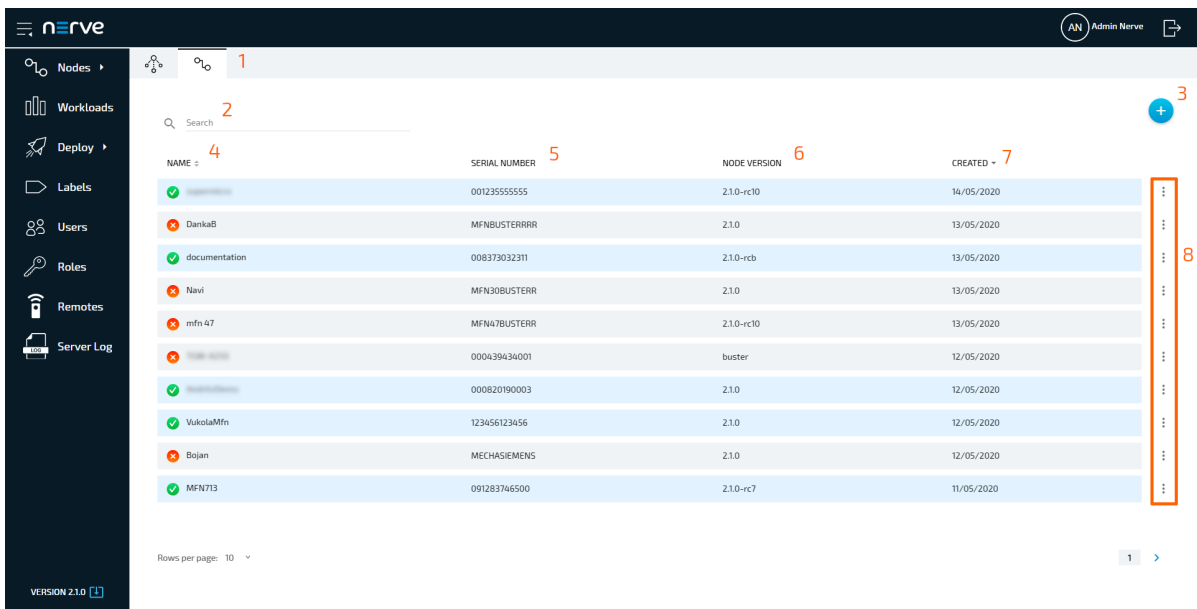
The topics covered in this chapter are mainly means of organization and have no impact on the functionality of the nodes and their workloads. Select **Nodes** in the navigation on the left. Then select the nodes tab



on the right to display the list of registered nodes.

### NOTE

Operational functions of the nodes are located in the node details view in the [node tree](#).



Item	Description
<b>Tab selection (1)</b>	Switch between the node tree and the node list by selecting the appropriate tab.
<b>Search bar (2)</b>	Use the search bar to filter nodes by name.
<b>Add new node (3)</b>	Click here to manually add a new node.
<b>NAME (4)</b>	This is the name of the node. If a node is online or offline can be seen to the left of the name. A green check mark indicates an online node while a red cross shows an offline node. The sorting of the list can also be switched from alphabetical to reverse alphabetical by clicking <b>NAME</b> , as well as being sorted by creation date by clicking <b>CREATED</b> .
<b>SERIAL NUMBER (5)</b>	This is the serial number of the node that was defined during node configuration.
<b>NODE VERSION (6)</b>	This is the version of the node reflecting the version of the Nerve product.
<b>CREATED (7)</b>	This is the date the node was registered in the Management System in the format DD/MM/YYYY. The list can be sorted by creation date when clicking <b>CREATED</b> .
<b>Ellipsis menu (8)</b>	Clicking here opens an overlay that allows deleting nodes.

## Adding a node

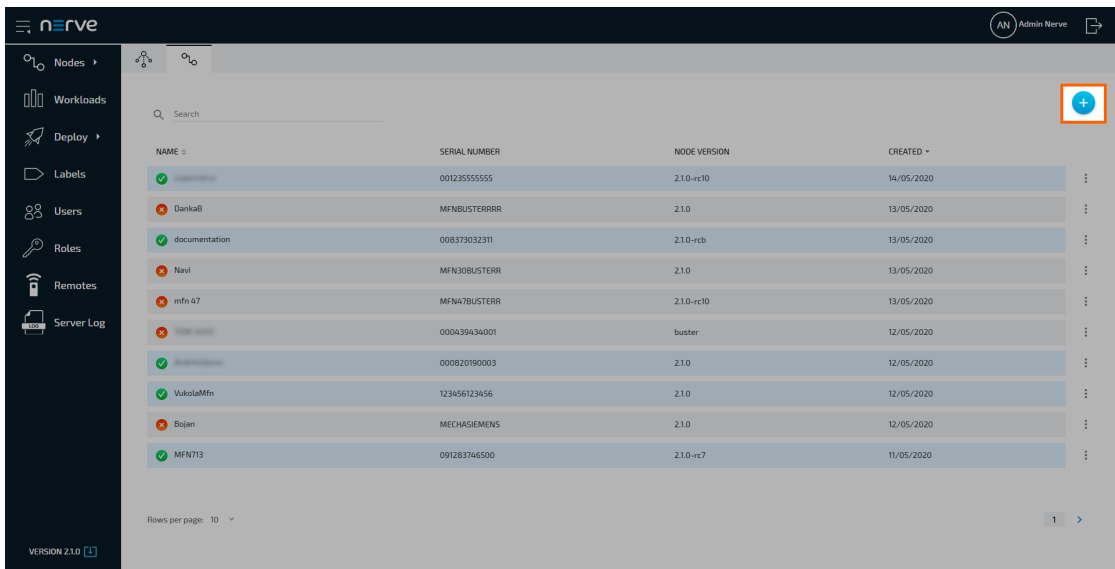
Nodes can only be added to the Management System if they have been configured in the Local UI, as the serial number and secure ID of the node are required. Refer to [Node configuration](#) before continuing.

1. Select **Nodes** in the menu on the left side.
2. Select the nodes tab



on the right to display the list of registered nodes.

3. Select the plus symbol (**Add new node**) in the upper-right corner.

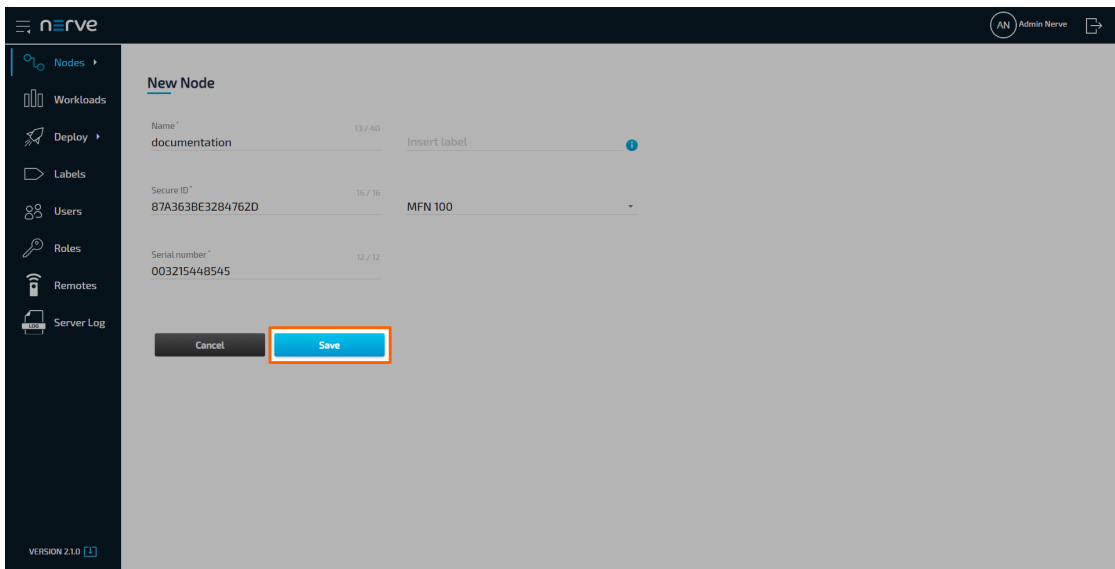


4. Enter the following information:

Item	Description
<b>Name</b>	Enter the name for the node that will make it easy to identify.
<b>Secure ID</b>	Enter the secure ID that is generated when the node is configured in the Local UI. Refer to <a href="#">Node configuration</a> for more information.
<b>Serial no.</b>	Enter the serial number of the Nerve Device that was defined during node configuration.
<b>Insert Label</b>	This field is optional. Add labels to the node for easier identification and workload deployment. Note that labels have to be created first before they can be assigned to nodes. Refer to <a href="#">Labels</a> for more information.
<b>Nerve Device name</b>	Select the Nerve Device, on which the node is hosted from the drop-down menu. The name is set to <b>MFN 100</b> by default. A picture of the selected Nerve Device will be displayed in the node details. Refer to the <a href="#">device guide</a> for more information on qualified Nerve Devices.

5. Click **Save** to save the changes.





The node now appears in the node list and in **Root > Unassigned** in the node tree and can be worked with. Refer to the [user guide](#) for information on how to continue.

## Editing the details of nodes

After registering a node, edit its details in the **Nodes** menu. Use the search bar at the top to search for nodes if a large number of nodes is registered.

1. Select **Nodes** in the left-hand menu.
2. Select the nodes tab



on the right to display the list of registered nodes.

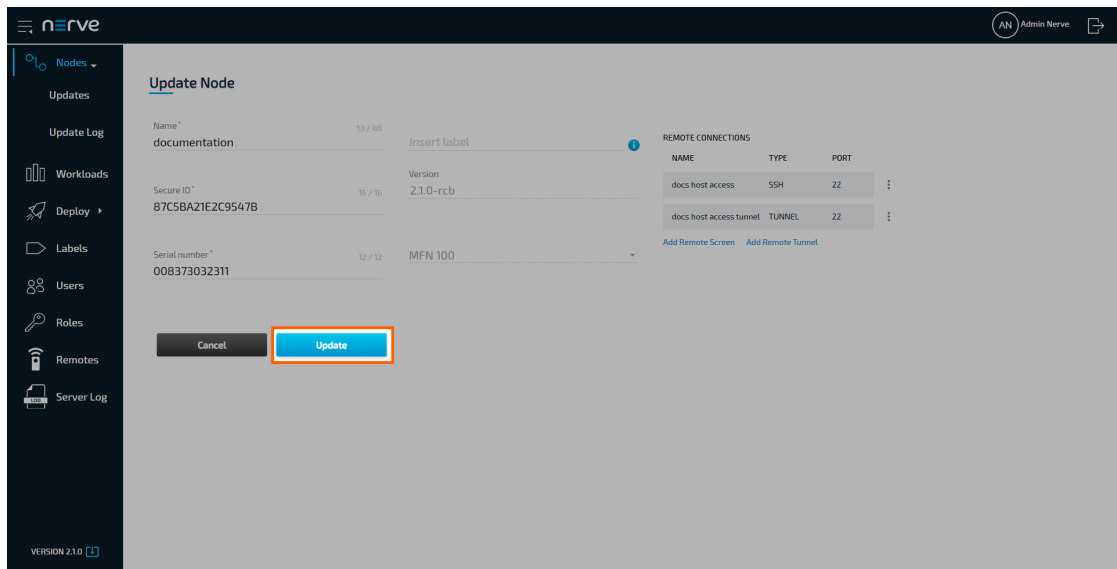
3. Click a node from the list.
4. Edit the details of the node:

Item	Description
<b>Name</b>	Enter the name of the node.
<b>Secure ID</b>	This is the ID generated during the node registration process. Do not edit this ID without a valid reason.
<b>Serial number</b>	This is the serial number of the node. Do not edit the serial number without a valid reason.
<b>Insert label</b>	This field is optional. Add labels to the node for easier identification and workload deployment. Refer to <a href="#">Labels</a> for more information.
<b>Version</b>	This is the node version. This field cannot be edited.
<b>Device type</b>	This is the device the node is hosted on. This field cannot be edited.
<b>REMOTE CONNECTIONS</b>	This is a list of remote connections to the node stating the <b>NAME</b> , <b>TYPE</b> , and <b>PORT</b> of available remote connections. Add or delete remote connections here. Refer to <a href="#">Remote Connections</a> for more information.

## NOTE

Changing the **Secure ID** or **Serial no.** of a node will break the connection between the node and the Management System.

5. Click **Update** to save the changes.



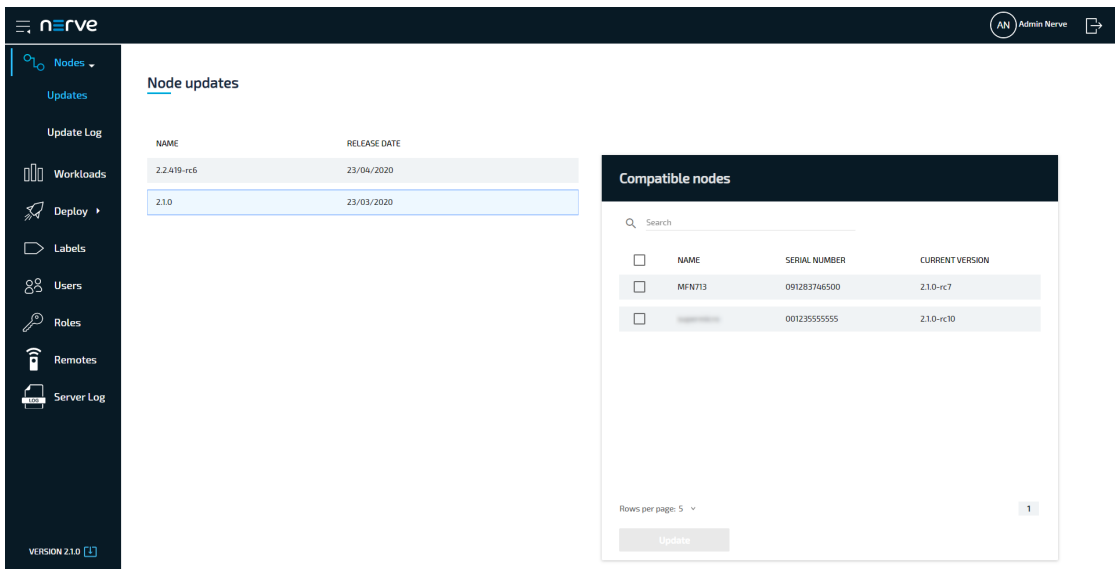
## Updating a node to a new version

Nodes are updated through the Management System where currently available updates are listed. Note that nodes can only be updated in order, meaning that node versions cannot be skipped. When an update is started, a snapshot of the user data is made and running workloads are stopped. Reverting to the previous version is possible. Workloads remain stopped after an update or revert.

## NOTE

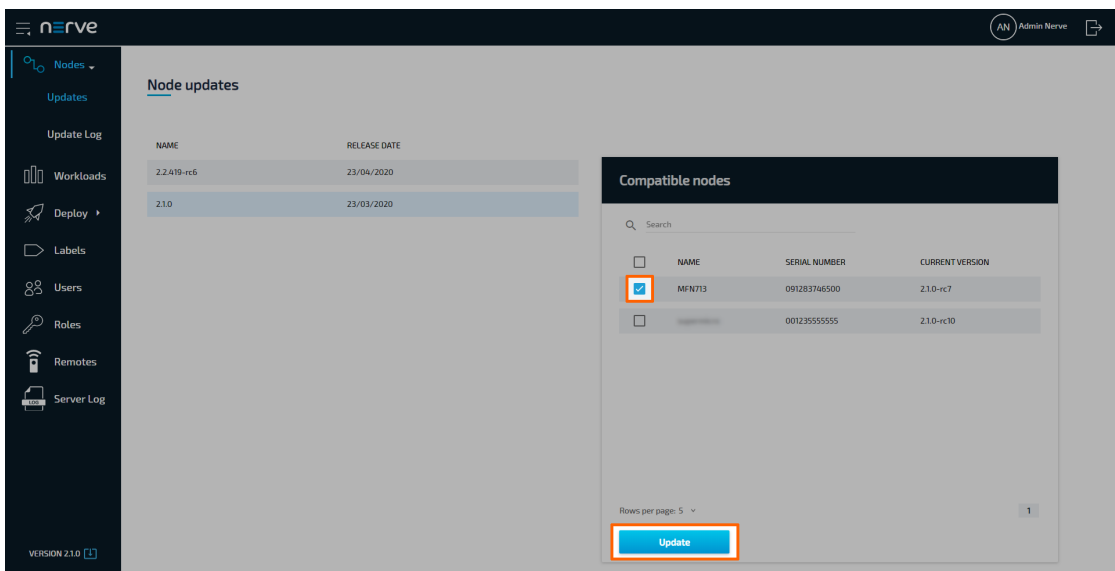
- By updating a node, the passwords to the Local UI and host access are reset if they have not been changed by the user. Local UI and host access passwords set by the user persist through updates.
- Note that the update process described here is valid for nodes from version 2.1.0 and later. Nodes below version 2.1.0 need to be updated manually. Refer to [Updating Nerve from version 2.0 to 2.1](#) for more information.

1. Expand **Nodes > Updates** in the navigation on the left.
2. Select a node version from the list. Available nodes for the update to the selected version will appear on the right.



3. Tick the checkboxes next to the nodes that will be updated.

4. Select **Update**.



5. Select **YES** in the pop-up window.

The Management System will display the log screen where the update is shown as in progress. The progress bar here shows the progress of the entire update campaign if more than one node has been selected before. Click the progress bar to see the progress of the update to each node.

## Removing a node

Note that generally there is no need in removing a node. Only remove a node in case of technical difficulties or by request of customer support. To remove a node, select **DELETE** in the ellipsis menu on the right side of a node in the node list.

NODE VERSION	
-	DELETE
-	⋮
-	⋮
-	⋮

## Workloads

In order to work with CODESYS applications, virtual machines or Docker containers on nodes, workloads need to be provisioned in the Management System. Here, provisioning is the creation of a workload and its storage in the workload repository of the Management System so that it can be deployed to nodes. This requires configuration of the workload and files that need to be uploaded to the Management System. After that, the workload can be deployed to nodes.

There are three types of workloads that can be provisioned: [CODESYS workloads](#), [Virtual Machine workloads](#) and [Docker workloads](#). The process of provisioning each workload is described in their respective chapters.

Select **Workloads** in the left-hand menu to find a list of all workloads that have been provisioned.

The screenshot shows the Nerve Management System interface. The sidebar menu on the left has 'Workloads' selected. The main content area displays a table of workloads with the following columns: NAME, TYPE, and CREATED. The table contains 10 rows of workload data. A vertical ellipsis menu is visible on the right side of the table, and a 'DELETE' button is shown in the top right corner of the table area.

NAME	TYPE	CREATED
TestAPIVM11	vm	18/05/2020
PrometheusAPILabel7	docker	15/05/2020
PrometheusAPILabel6	docker	15/05/2020
gbf	docker	15/05/2020
CodesysAPILabel	codesys	15/05/2020
RemoteViewLight	docker	15/05/2020
PrometheusAPILabel1	docker	15/05/2020
asdasd	docker	15/05/2020
Alpine	vm	15/05/2020
PrometheusAPILabel	docker	15/05/2020

Item	Description
<b>Search bar (1)</b>	Use the search bar to filter workloads in the list by name. Select the cross symbol next to the search bar to reset the search field.
<b>Workload Type (2)</b>	This is a drop-down menu that allows filtering the list below by workload type. The available options are <b>VM</b> , <b>Docker</b> , <b>CODESYS</b> and <b>All</b> .
<b>Show disabled (3)</b>	Disabled workloads are not shown in the list of workloads. Ticking this checkbox shows them again. However, note that this does not enable the workloads again.
<b>Add new workload (4)</b>	Select the plus symbol to provision a new workload.
<b>NAME (5)</b>	This is the name of the workload that has been defined in the provisioning process. The list can be sorted in alphabetical and reverse alphabetical order by clicking <b>NAME</b> .
<b>TYPE (6)</b>	The type of workload is displayed here: <b>codesys</b> , <b>vm</b> or <b>docker</b> . The list can be sorted by workload type when clicking <b>TYPE</b> .
<b>CREATED (7)</b>	This is the date when the workload was provisioned in the format DD/MM/YYYY. The list can be sorted by creation date when clicking <b>CREATED</b> .
<b>Ellipsis menu (8)</b>	Clicking here opens an overlay that gives two options: <b>DELETE</b> and <b>DISABLE</b> .

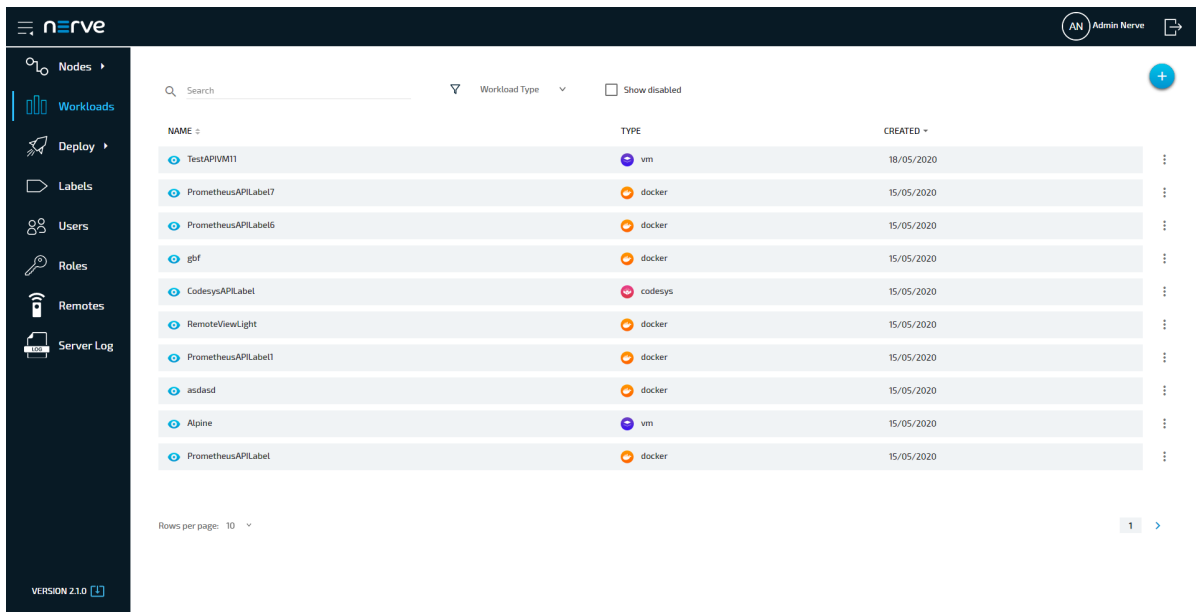
## Provisioning a workload

Provisioning a workload is the creation of a workload in the workload repository of the Management System. Workloads that have been provisioned are ready to be deployed to nodes.

Select the plus symbol in the upper-right corner of the workloads list to start provisioning a workload. The provisioning process of each workload type is covered separately in the following chapters.

- [Provisioning a CODESYS workload](#)
- [Provisioning a Virtual Machine workload](#)
- [Provisioning a Docker workload](#)

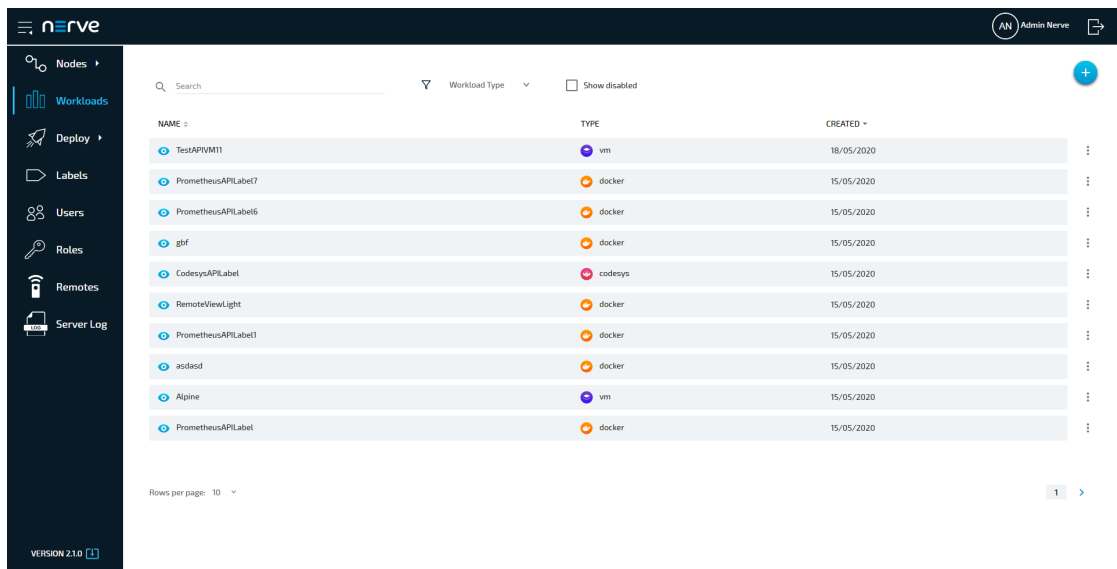
After provisioning a workload, it will appear in the list of workloads.



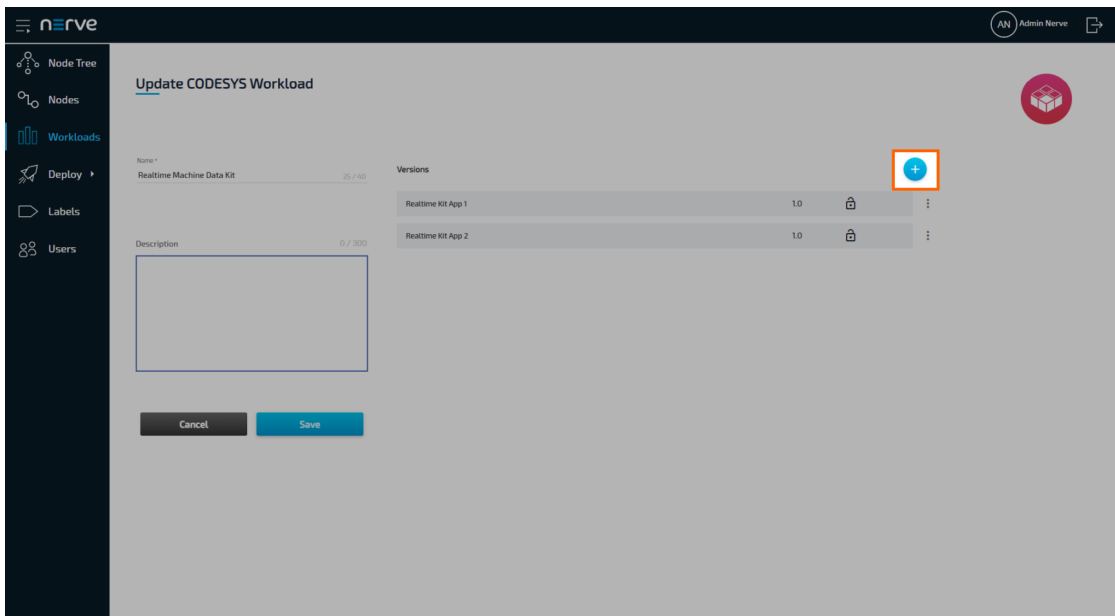
## Adding a new workload version

Add new versions to a provisioned workload to accommodate different use cases.

1. Select **Workloads** from the menu on the left side.
2. Select the workload to which a new version will be added.

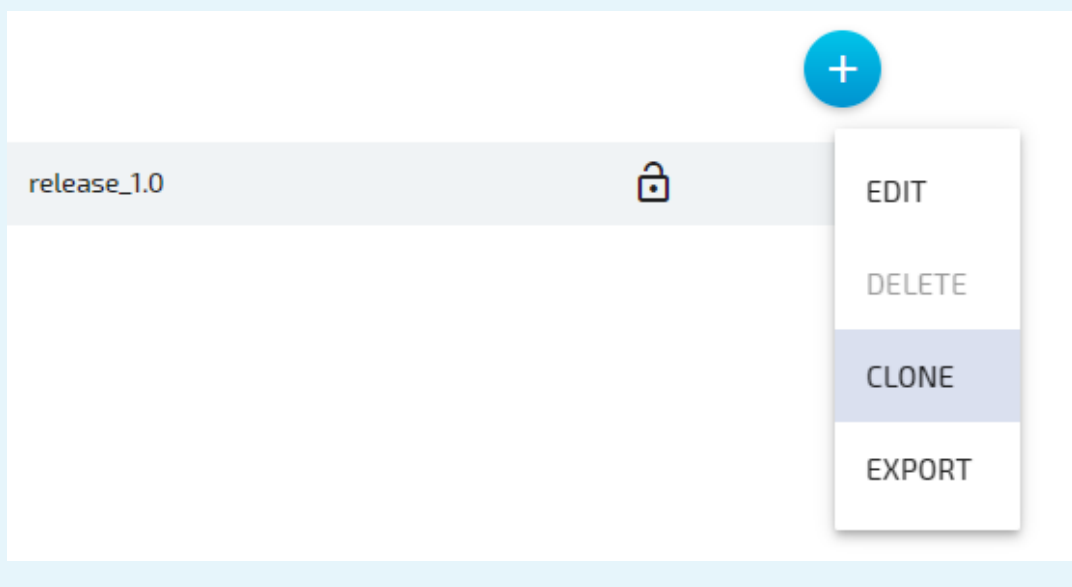


3. Click the plus symbol in the upper-right.



#### NOTE

To add a new version that is a slight modification of an existing one, click the ellipsis menu next to a workload and select **CLONE** from the overlay.



4. Configure the new workload version. Refer to the provisioning chapters of each workload type linked [above](#) for more information on configuration settings.

#### NOTE

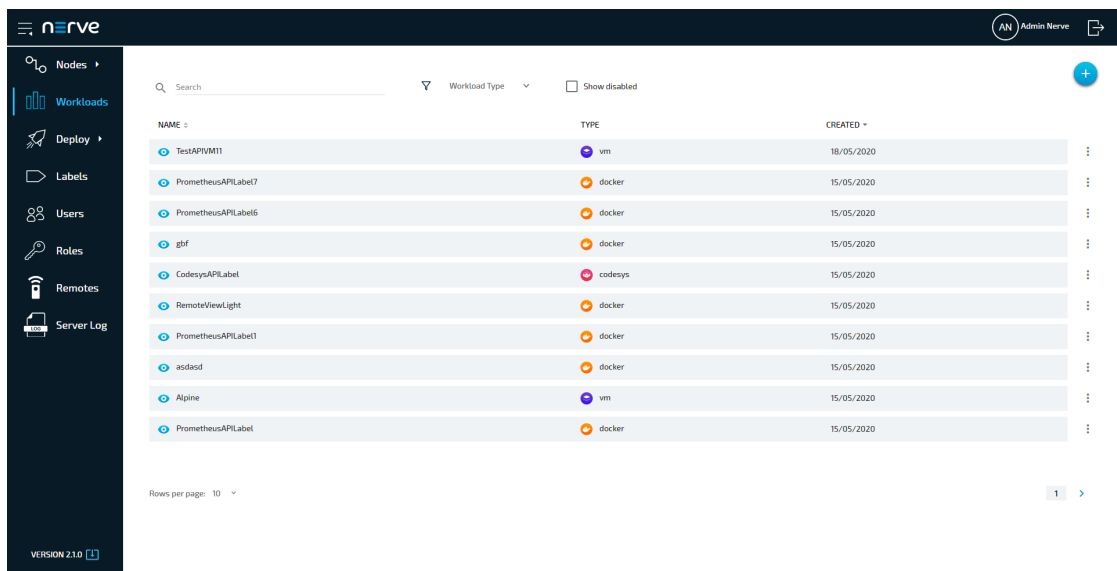
The fields of the new version will already have information filled in. The system enters the settings of the latest version automatically. If **CLONE** next to a workload version was used, the information filled in will be from that version instead.

5. Click **Update** to save the new version of the workload.

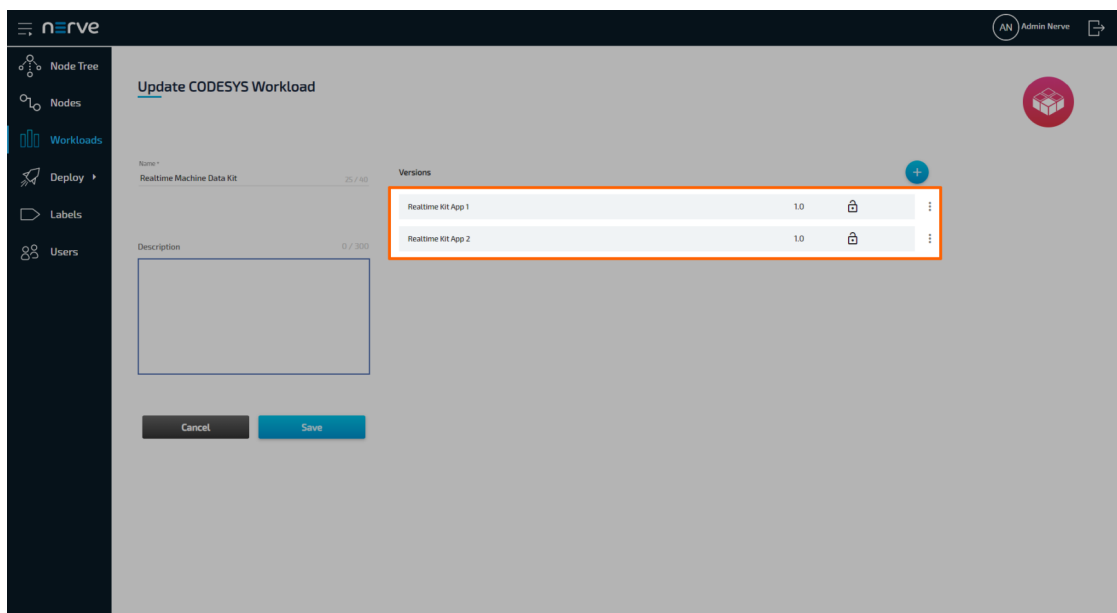
## Editing a workload

General information of a workload and configuration settings of each version can be edited starting from the workload list. General information of a workload is valid for all workload versions.

1. Select **Workloads** from the menu on the left side.
2. Select the workload to edit.



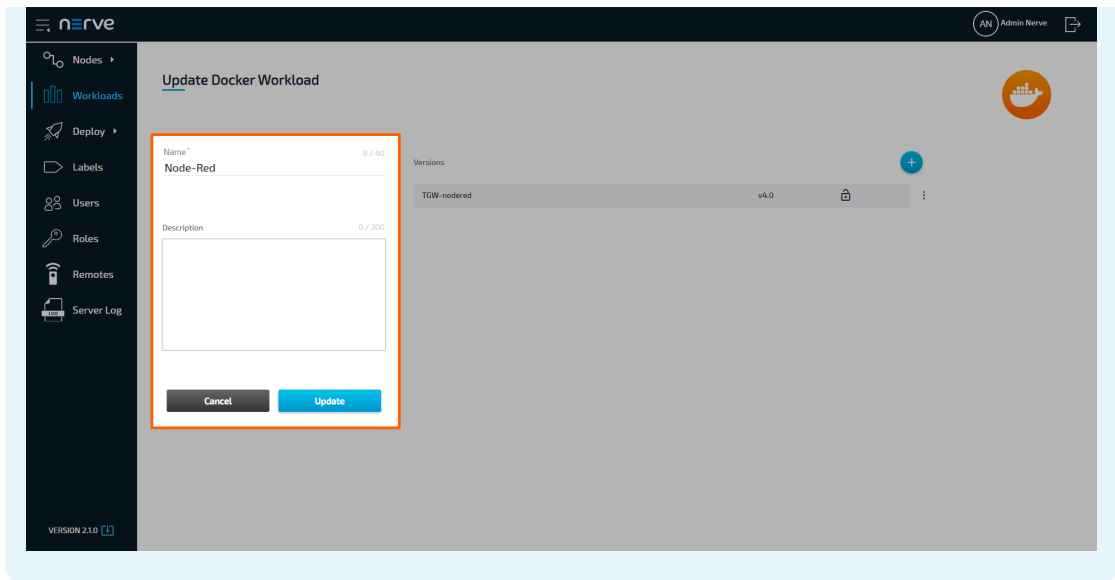
3. Select the workload version to edit from the list on the right.



### NOTE

Editing the **Name** and **Description** of the workload can be done on the left side before selecting a version. Perform the desired changes and click **Update**.





4. Perform changes to the workload version.

#### NOTE

The settings of a workload depend on the workload type. See the version settings for each workload in the provisioning chapters: [CODESYS workloads](#), [Virtual Machine workloads](#) and [Docker workloads](#).

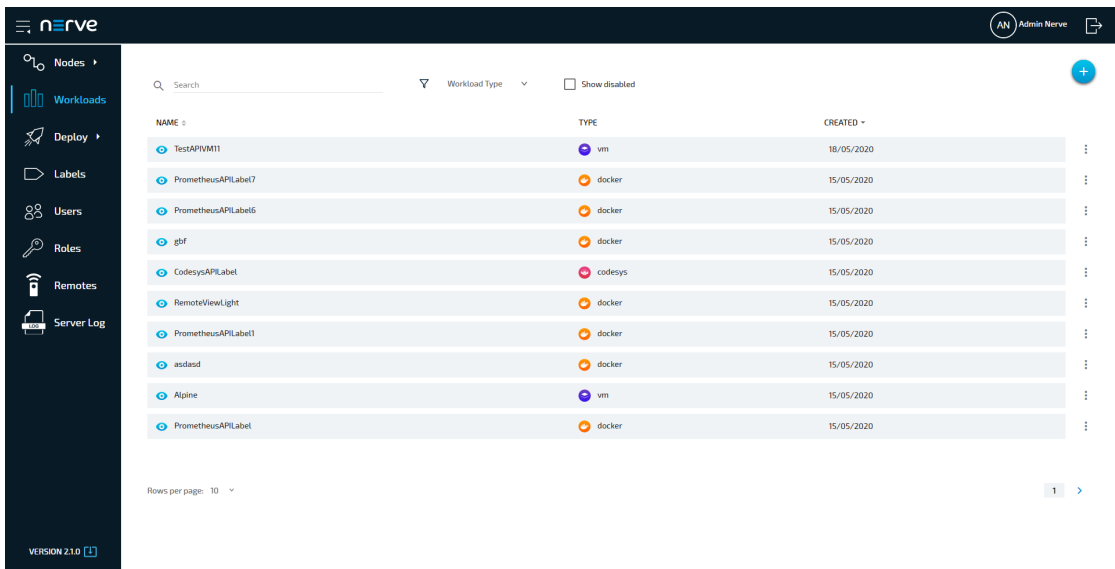
5. Click **Update** to save the changes.

Note that a workload needs to be undeployed first before an updated version can be deployed. However, this does not apply to remote connections. Once a remote connection is configured, it is available immediately for the workload on all nodes that it was deployed to.

## Disabling a workload

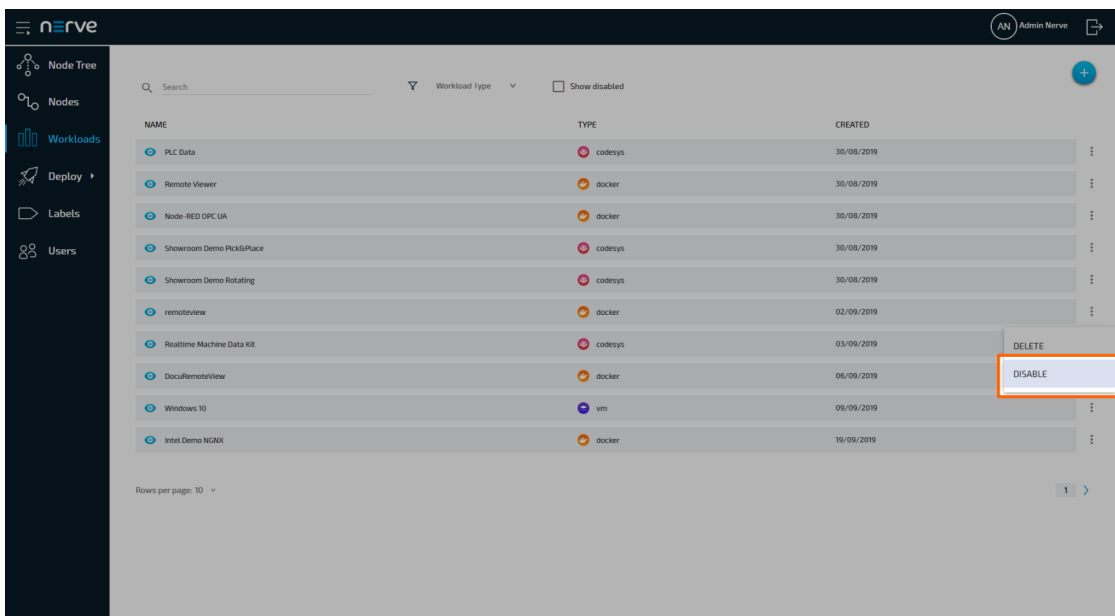
A workload can be disabled to make it hidden and not selectable. This will hide the workload in the workload list and deployment process but it will not be deleted from the Management System. This also means that the workload cannot be deployed. Workloads that have been deployed to nodes before are not affected.

1. Select **Workloads** from the menu on the left side.
2. Choose the workload to disable.



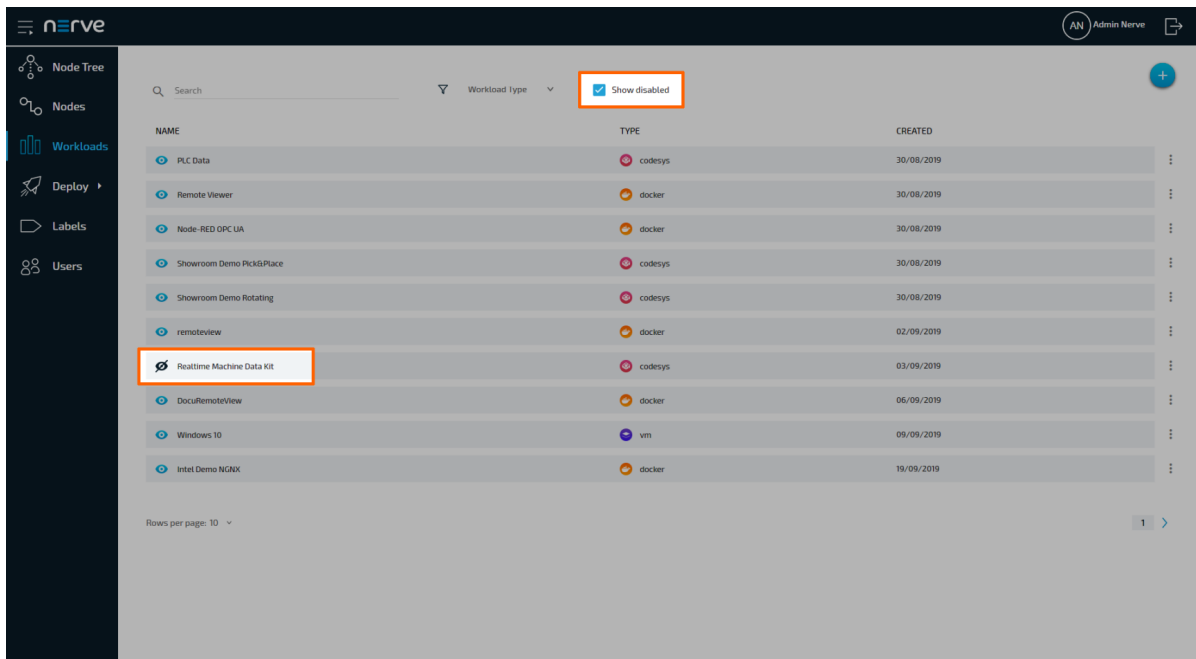
3. Click the ellipsis menu to the right of the workload.

4. Select **DISABLE** from the overlay that appeared.



5. Click **OK** in the new window.

The workload is now disabled and hidden in the list. To show disabled workloads again, tick the checkbox next to **Show Disabled** in the list of workloads. All disabled workloads are marked by an icon resembling a crossed out eye.

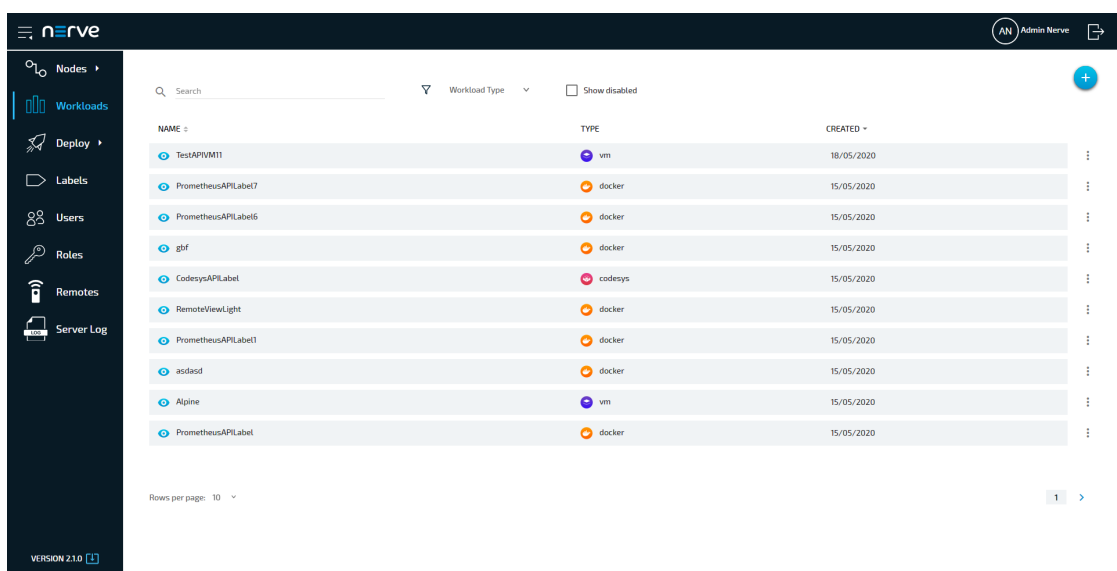


Follow the steps above to enable the workload again. However, select **ENABLE** in the overlay.

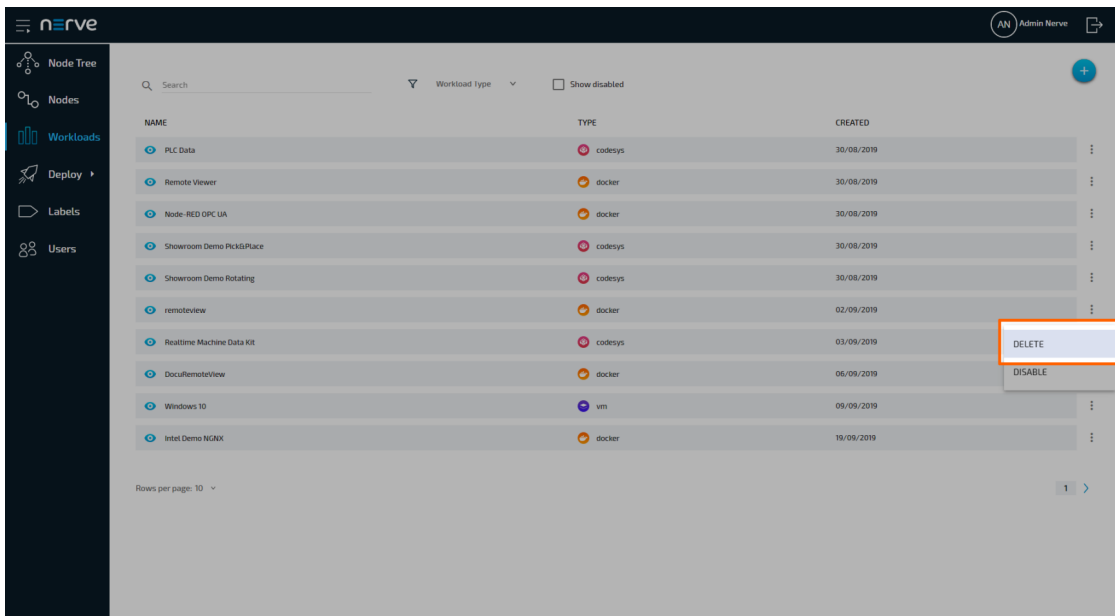
## Deleting a workload

The instructions below cover the deletion of a workload from the repository in the Management System. Note that deleting a workload from the Management System will not automatically remove the workload from nodes. To remove a workload from a node, undeploy the workload in the node tree.

1. Select **Workloads** from the menu on the left side.
2. Choose the workload to delete.



3. Click the ellipsis menu to the right of the workload.
4. Select **DELETE** in the overlay that appeared.



5. Click **OK** in the new window to confirm the deletion.

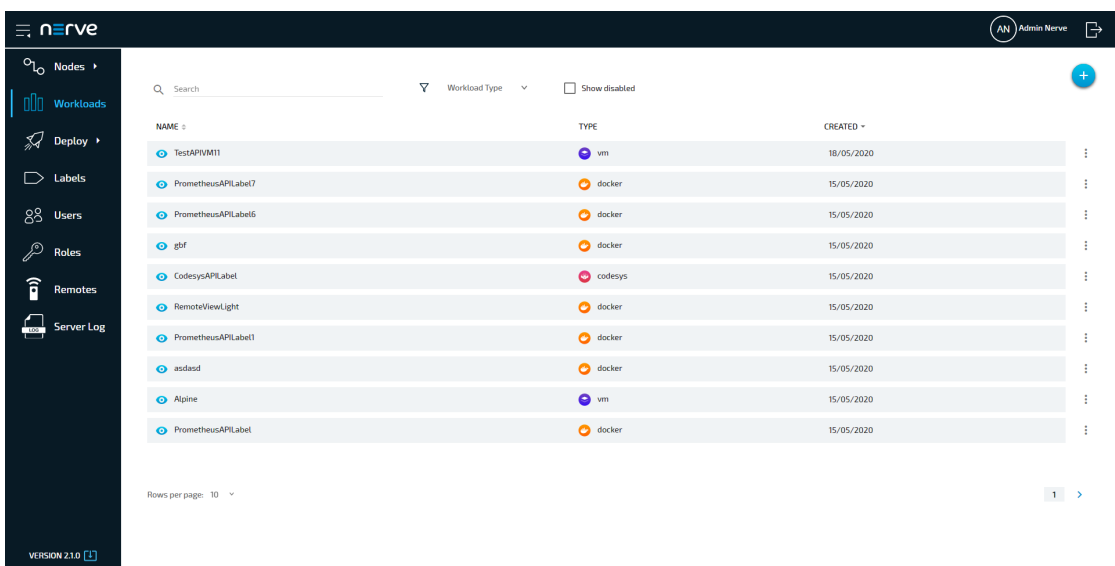
## NOTE

Deleting a workload will automatically delete all versions of the workload as well.

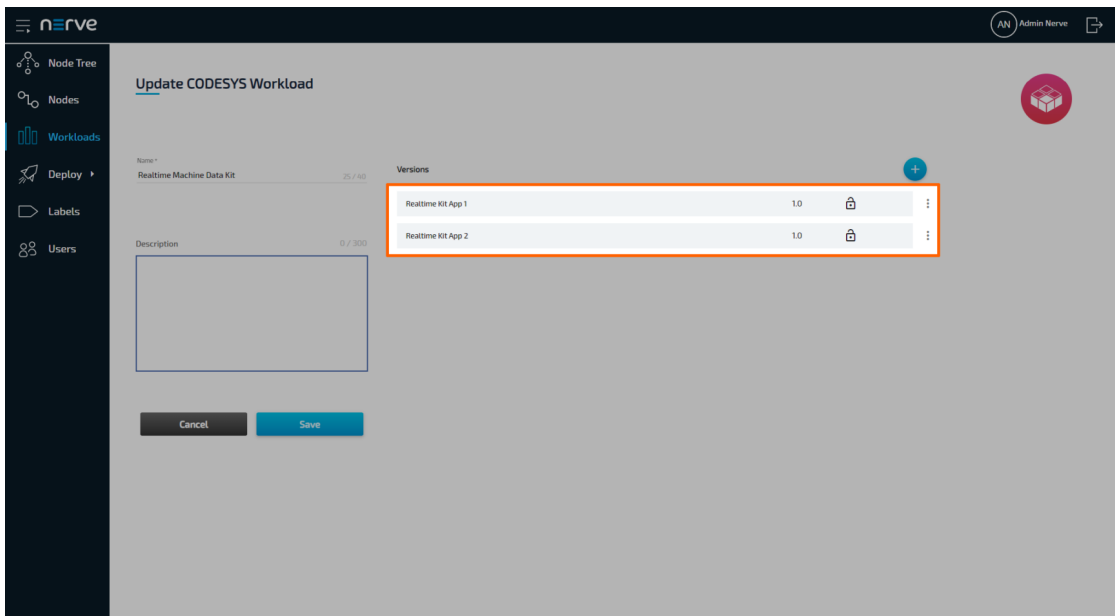
## Deleting a workload version

Workload versions can be deleted separately in the workload details. Note that deleting a workload version is not possible if only one version of the workload exists. Delete the workload to delete the only workload version.

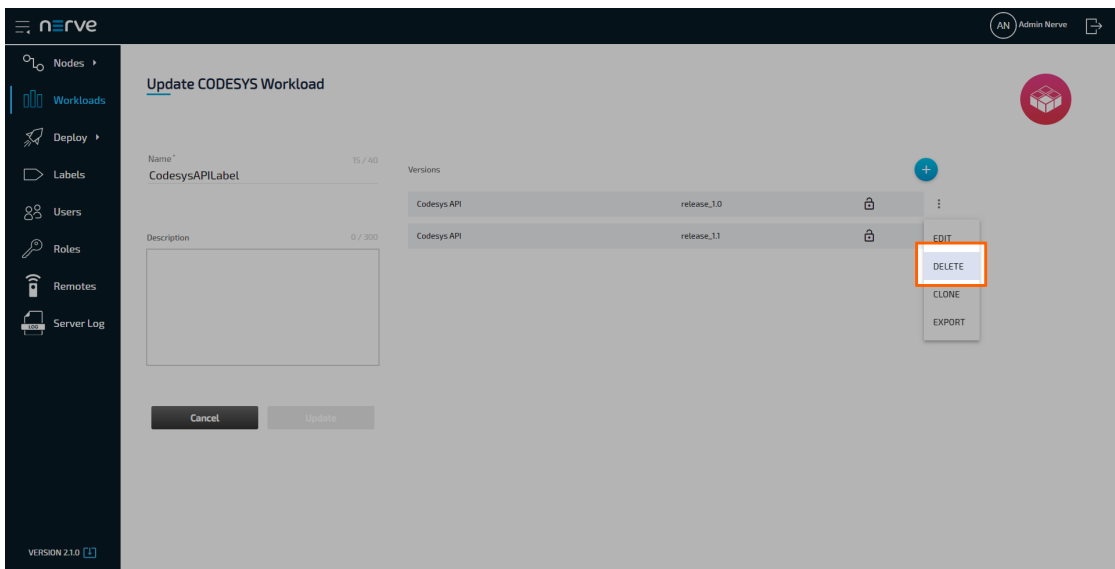
1. Select **Workloads** from the menu on the left side.
2. Select the workload of which a version will be deleted.



3. Choose the workload version to delete.



4. Click the ellipsis menu to the right of the workload version.
5. Select **DELETE** in the overlay that appeared.

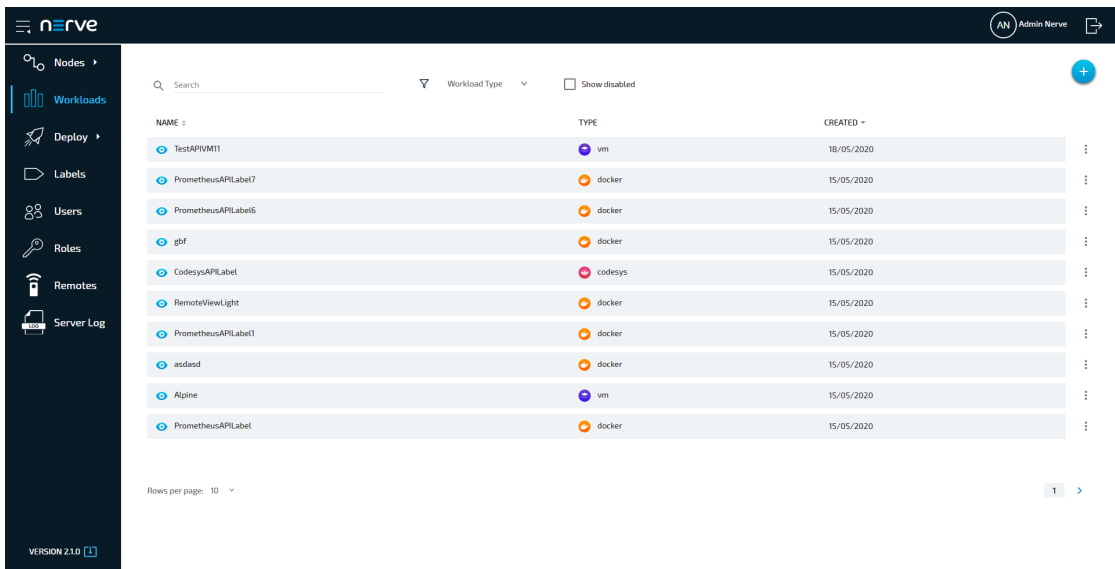


6. Click **OK** in the new window to confirm the deletion.

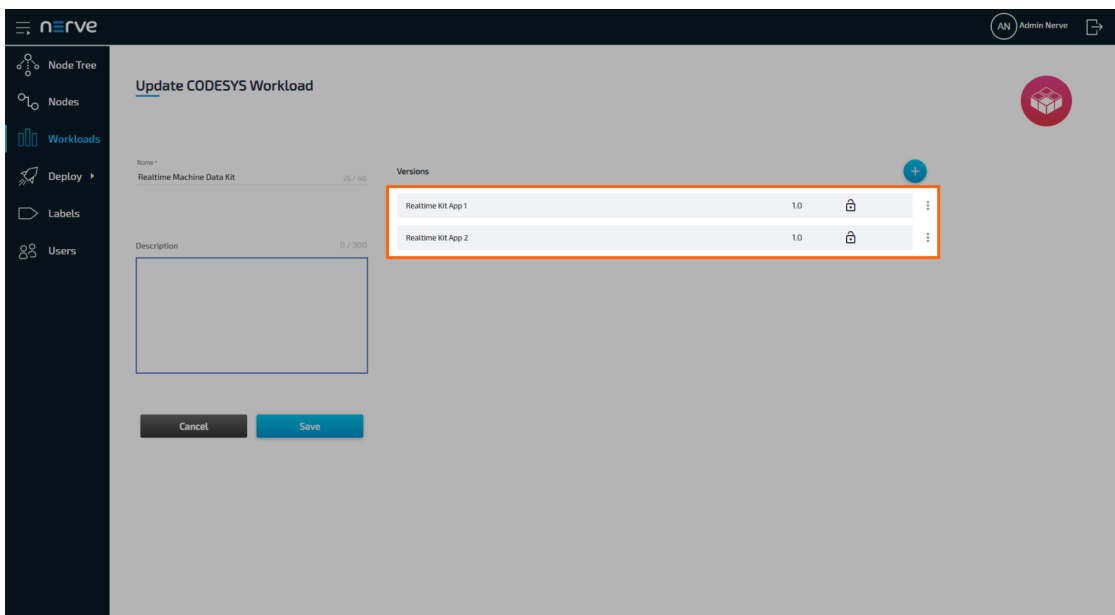
## Exporting a workload

Workloads can be manually deployed to nodes. To do that, a workload version must be exported. For more information on Refer to [Local Workload Deployment](#) for more information on local workload deployment.

1. Select **Workloads** from the menu on the left side.
2. Select the workload of which a version will be exported.

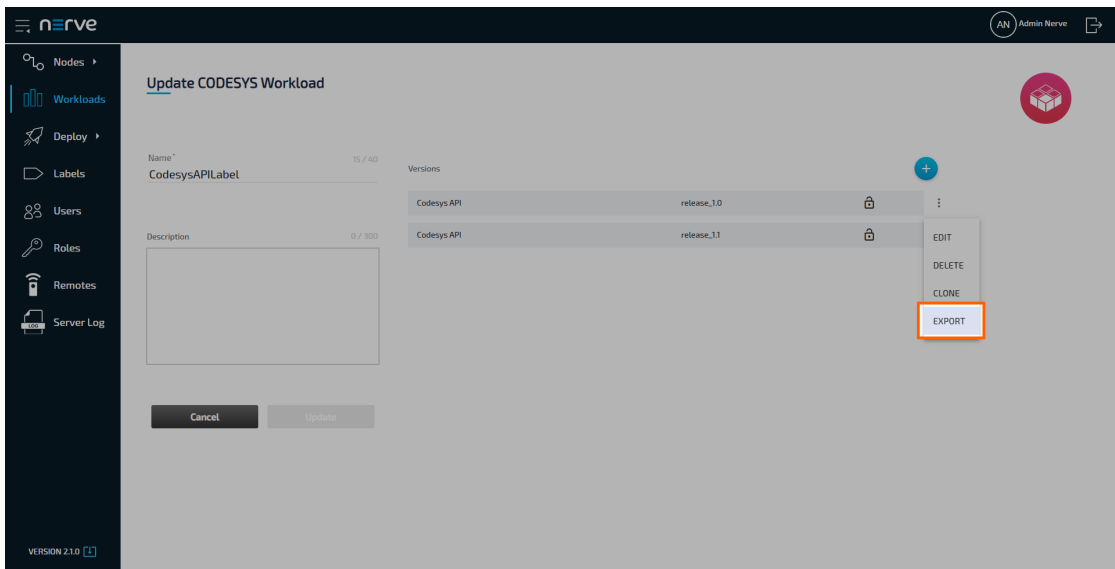


3. Choose the workload version to export.



4. Click the ellipsis menu to the right of the workload version.

5. Select **EXPORT** in the overlay that appeared.



The download of the exported workload version is started automatically. The workload version is compressed into a TAR file.

#### NOTE

- Be careful if a workload export produces a TAR.GZ file. Using a TAR.GZ file for local workload deployment, will break the deployment process. Transform the TAR.GZ file into a TAR file before locally importing a workload.
- If the workload export fails with a 403 error code, delete the cookies from the web browser and try again.

## Controlling a workload

Workloads can be controlled from the node details view in the node tree. Refer to the [node tree chapter](#) for more information.

## Updating a deployed workload

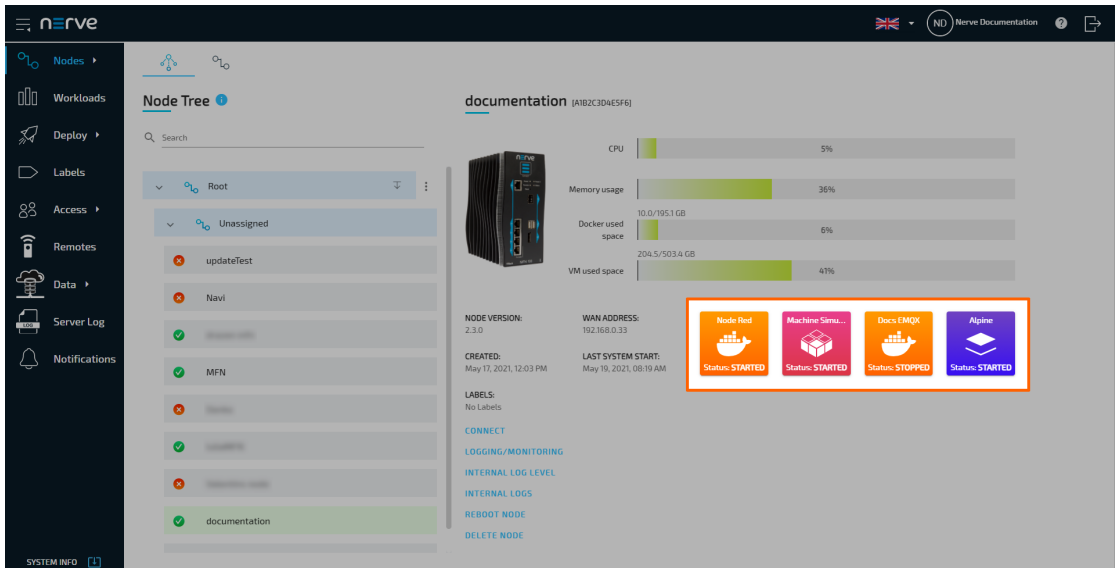
Deployed CODESYS and Docker workloads can be updated through the workload control screen. Updating the workload is a quick way to deploy a new version of the same workload. Also, note that a Docker volume for persistent storage will be removed if the workload version update does not have a volume defined and a workload update is performed.

#### NOTE

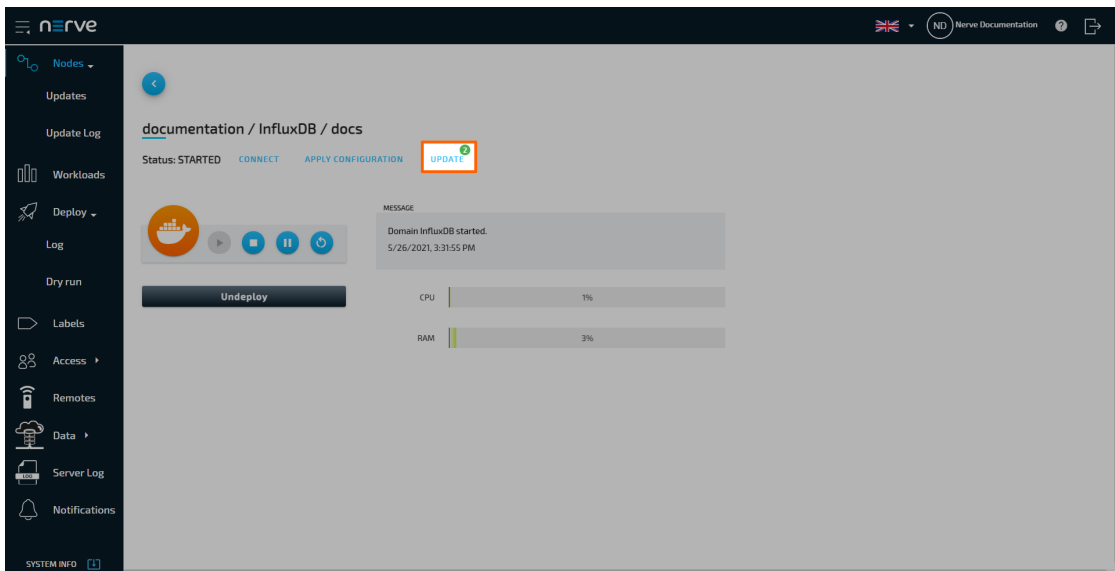
Note that this functionality is also available in the Local UI. However, note that the workload update needs to be exported first. Refer to [Updating a deployed workload in the Local UI](#) for more information.

1. Select **Nodes** in the navigation on the left.
2. Select a node in the node tree that has an updatable workload deployed.

3. Select the tile of the updatable workload in the node details view.



4. Select **UPDATE**.

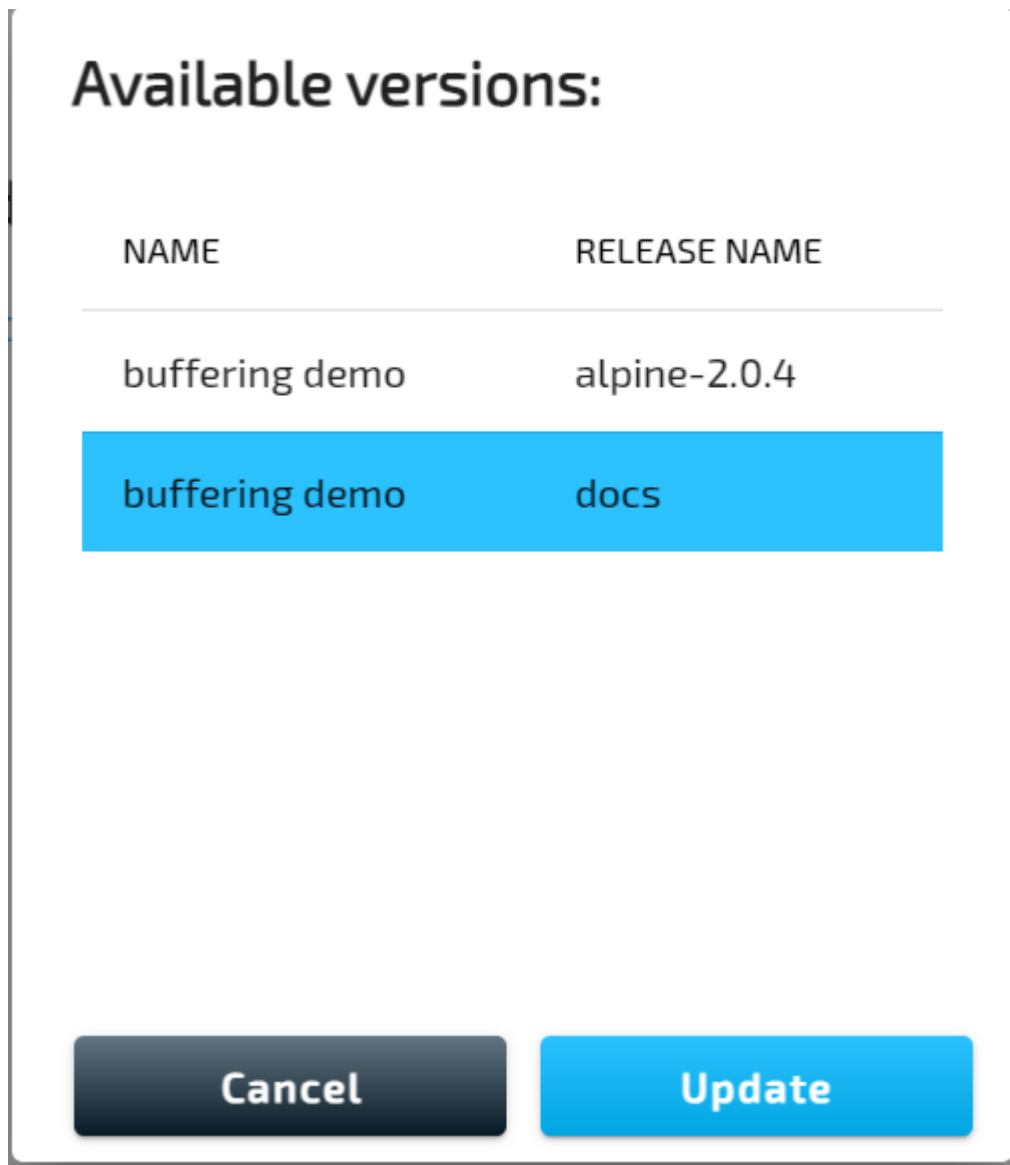


**NOTE**

The notification bubble next to **UPDATE** indicates the number of available versions of the workload.

5. Select a version from the list of available versions.





6. Select **Update**

This starts the deployment of the workload version update. The Management System will continue to the log next. The update is at the top of the list. A timestamp is the name that identifies the deployment in the log.

DEPLOYMENT NAME	ACTION	PROGRESS	STARTED	FINISHED
5/28/2021,9:41:30AM	Deploy	0.00% <span>In progress</span>	28/05/2021 09:41	In progress
5/28/2021,9:35:19AM	Deploy	100.00% <span>Complete</span>	28/05/2021 09:35	28/05/2021 09:35
27/05/2021,12:14:41	Deploy	100.00% <span>Failed</span>	27/05/2021 13:14	27/05/2021 13:15
5/27/2021,1:01:04PM	Deploy	100.00% <span>Failed</span>	27/05/2021 13:01	27/05/2021 13:07
27/05/2021,11:56:40	Deploy	100.00% <span>Complete</span>	27/05/2021 12:56	27/05/2021 12:57
5/26/2021,3:31:06PM	Deploy	100.00% <span>Complete</span>	26/05/2021 15:31	26/05/2021 15:31
5/26/2021,11:37:54AM	Deploy	100.00% <span>Complete</span>	26/05/2021 11:38	26/05/2021 11:39
5/26/2021,11:29:08AM	Deploy	100.00% <span>Complete</span>	26/05/2021 11:29	26/05/2021 11:29
5/26/2021,11:09:57AM	Deploy	100.00% <span>Complete</span>	26/05/2021 11:10	26/05/2021 11:12
5/26/2021,10:54:11AM	Deploy	0.00% <span>In progress</span>	26/05/2021 10:54	26/05/2021 11:00

## Provisioning a CODESYS workload

Before a CODESYS workload can be provisioned, a CODESYS application has to be loaded into the CODESYS runtime first. Refer to [First steps with CODESYS](#) first before continuing.

Once a CODESYS application has been loaded into the CODESYS runtime, the following steps have to be taken before the workload can be provisioned:

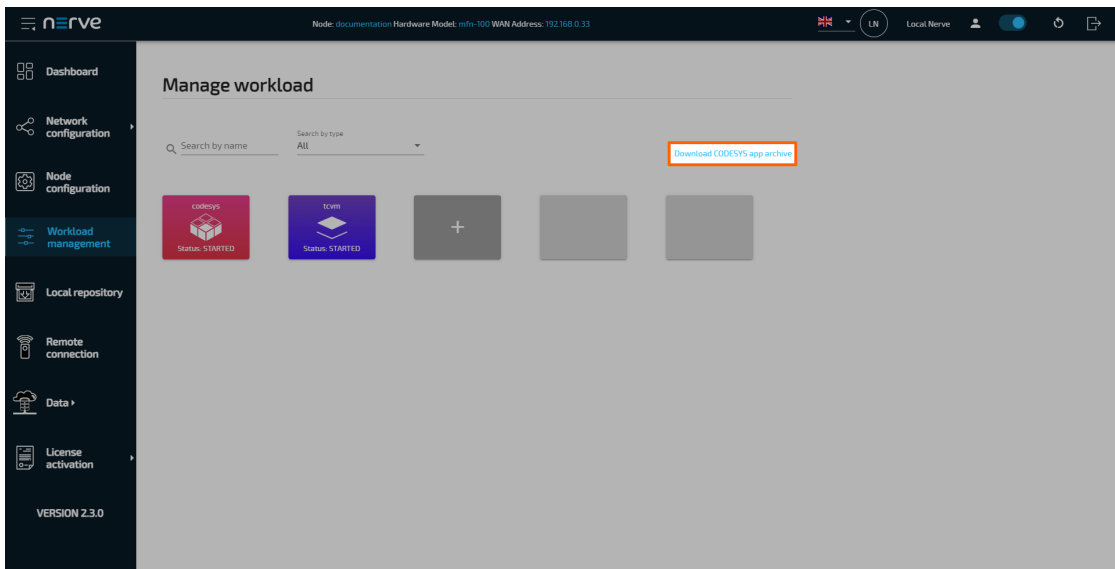
1. Creating the ZIP file of the CODESYS application
2. Transferring the ZIP file to a local workstation

Also the workstation needs to be connected to the physical port of the Nerve Device associated with host access, and the network adapter IP address of the workstation needs to be configured in the correct range. This information is device specific. Refer to the [device guide](#) for information on the Nerve Device.

## Creating the ZIP file on the Nerve Device

First, the CODESYS project needs to be zipped on the Nerve Device before it can be copied from the CODESYS runtime. This is done through the Local UI.

1. Follow the link to the Local UI of the used Nerve Device. Refer to the [device guide](#) for more information.
2. Select **Workload management** in the navigation on the left.
3. Click **Download CODESYS app archive**.



4. Select **YES** in the pop-up. Note that the CODESYS application will be stopped.

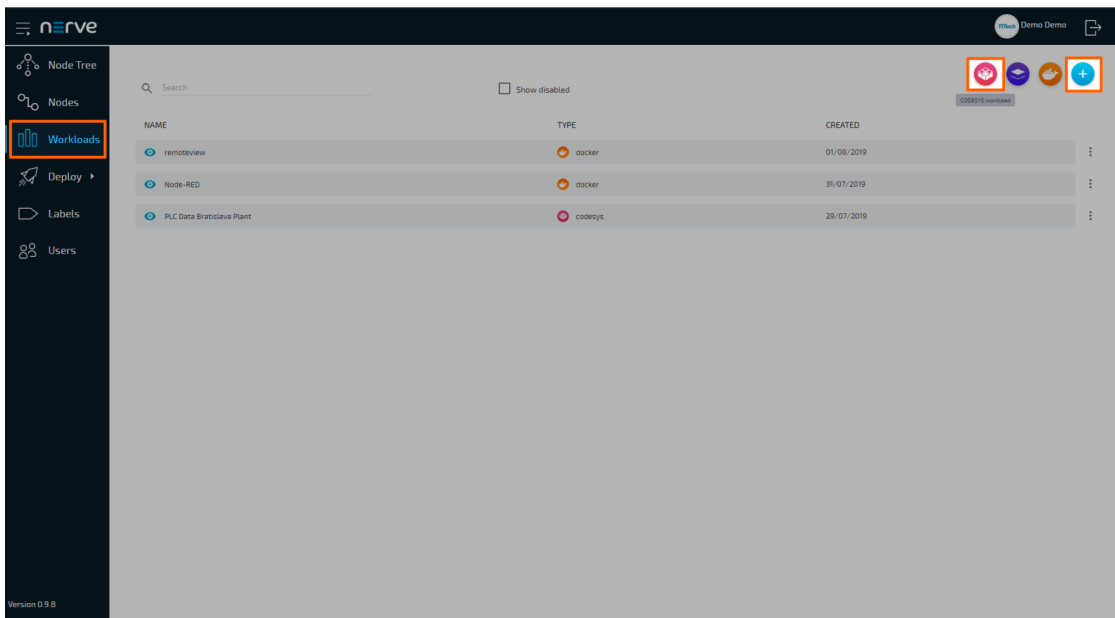
The ZIP file is automatically downloaded to the workstation and a CODESYS workload can now be provisioned in the Management System.

## Provisioning a CODESYS workload

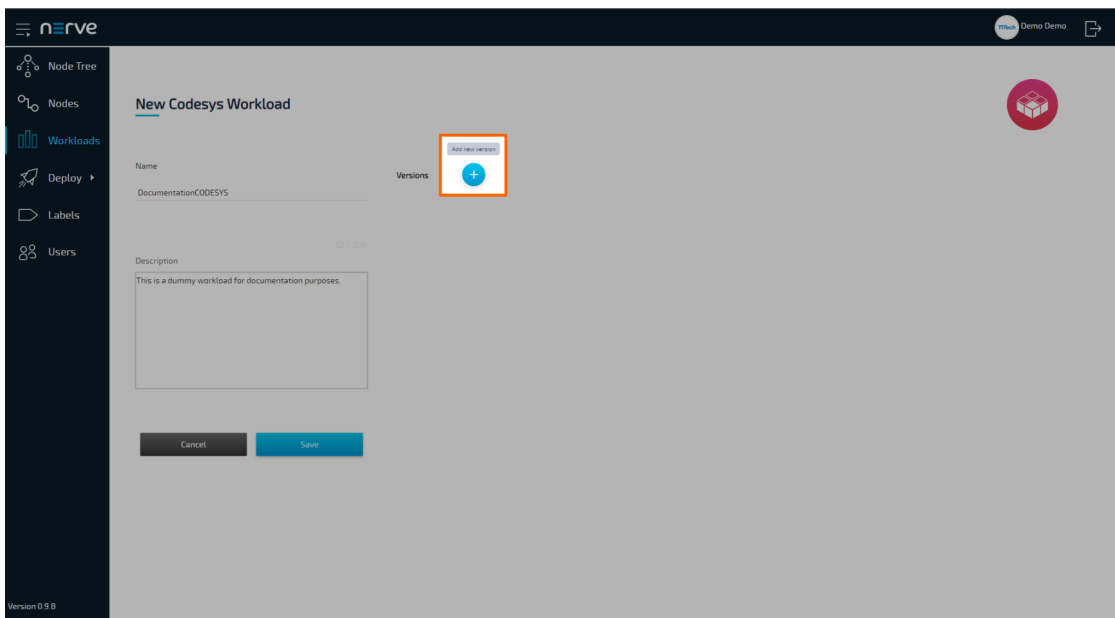
The following instructions cover the basic requirements for provisioning a CODESYS workload. Optional settings will be left out. Extended options are addressed in the last section of this chapter.

There are two further types of workloads that can be provisioned: [Virtual Machine workloads](#) and [Docker workloads](#). The process for each workload is highlighted in its respective chapter.

1. Log in to the Management System.
2. Select **Workloads** in the left-hand menu.
3. Select the plus symbol in the upper-right corner.
4. Select the CODESYS symbol (**CODESYS workload**) on the left of the three symbols that expanded.

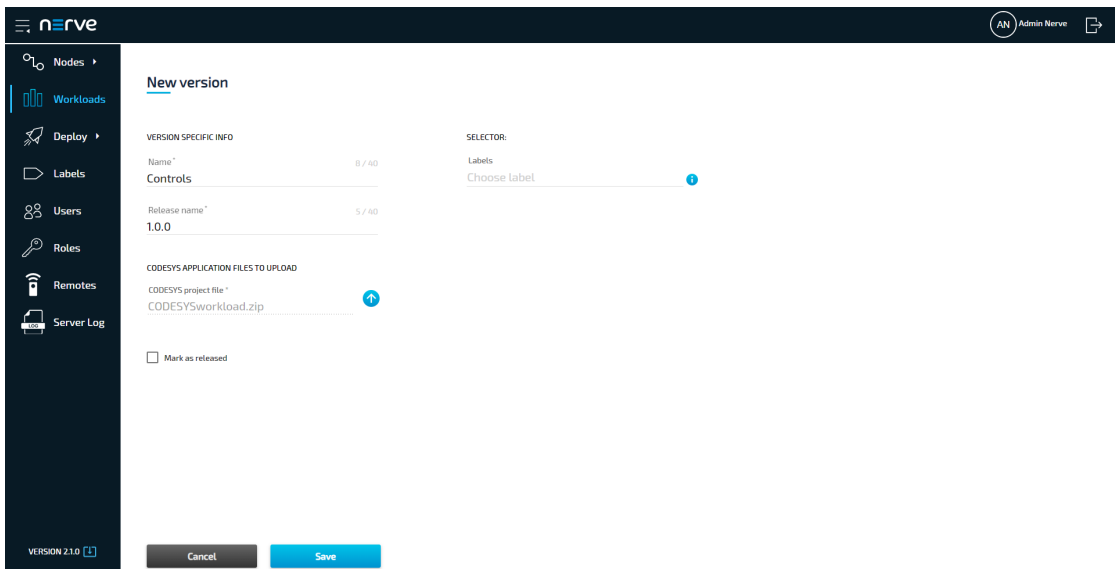


5. In the new window, enter a name for the workload.
6. Select the plus symbol next to **Versions** to add a new version of the workload.



7. In the new window, enter the following information:

Item	Description
<b>Name</b>	Enter a <b>Name</b> for the version of this workload.
<b>Release name</b>	Enter a <b>Release name</b> for the version of this workload.
<b>CODESYS project file</b>	Click the <b>upward arrow</b> symbol to add the CODESYS application ZIP file. This is the ZIP file that was created above.



8. Click **Save**.

The workload has now been provisioned and is ready to be deployed in the **Deploy** menu.

## Settings for CODESYS workloads

In the instructions above, optional settings have been left out. Below is an overview of all options with an explanation to each option.

Setting	Description
<b>VERSION SPECIFIC INFO</b>	<p><b>Name</b> A name for the workload version. Choose a precise name to make the workload version unambiguous.</p> <p><b>Release name</b> A release name for the workload version. This could be a version number. Example: 1.0.1</p>
<b>CODESYS APPLICATION FILES TO UPLOAD</b>	Upload the <b>CODESYS project file</b> here. This is a ZIP file that has to be generated from a CODESYS project running in the CODESYS runtime. Upload it here by clicking the <b>upward arrow</b> symbol to open the file browser.
<b>Mark as released</b>	Tick this checkbox to mark this workload as released. Once marked as released, the workload cannot be edited anymore.
<b>SELECTOR</b>	<p><b>Labels</b> If labels have been defined and assigned to nodes, add them as selectors to the workload. When deploying a workload, the list of nodes will be filtered automatically to the specified label.</p>

## Provisioning a Virtual Machine workload

Before a Virtual Machine workload can be provisioned, it is required to set up the virtual machine. Virtual Machine workloads can be created on any PC but an environment to

effectively create a Virtual Machine workload is not always readily available. Therefore, it is described how the Nerve Device can be utilized to create a Virtual Machine workload. What needs to be done is:

1. Creating a virtual machine on the node
2. Installing an operating system on the virtual machine
3. Obtaining the virtual machine IMG and XML files
4. Provisioning the Virtual Machine workload in the Management System

In this version, Nerve does not provide a GUI based method for installing an OS on a virtual machine and obtaining the virtual machine IMG and XML configuration files. Therefore, this chapter focuses on the manual process. Three tools are required for the instructions below, assuming Windows is used on the workstation:

- an X Server application like [Xming](#)
- an SSH client like [PuTTY](#)
- a file transfer client like [WinSCP](#)

Also the workstation needs to be connected to the physical port of the Nerve Device associated with host access, and the network adapter IP address of the workstation needs to be configured in the correct range. This information is device specific. Refer to the [device guide](#) for information on the Nerve Device.

The instructions below are split up into multiple parts to make them easier to follow. The subsections of the instructions are connected and every subheading is a requirement for the next paragraph.

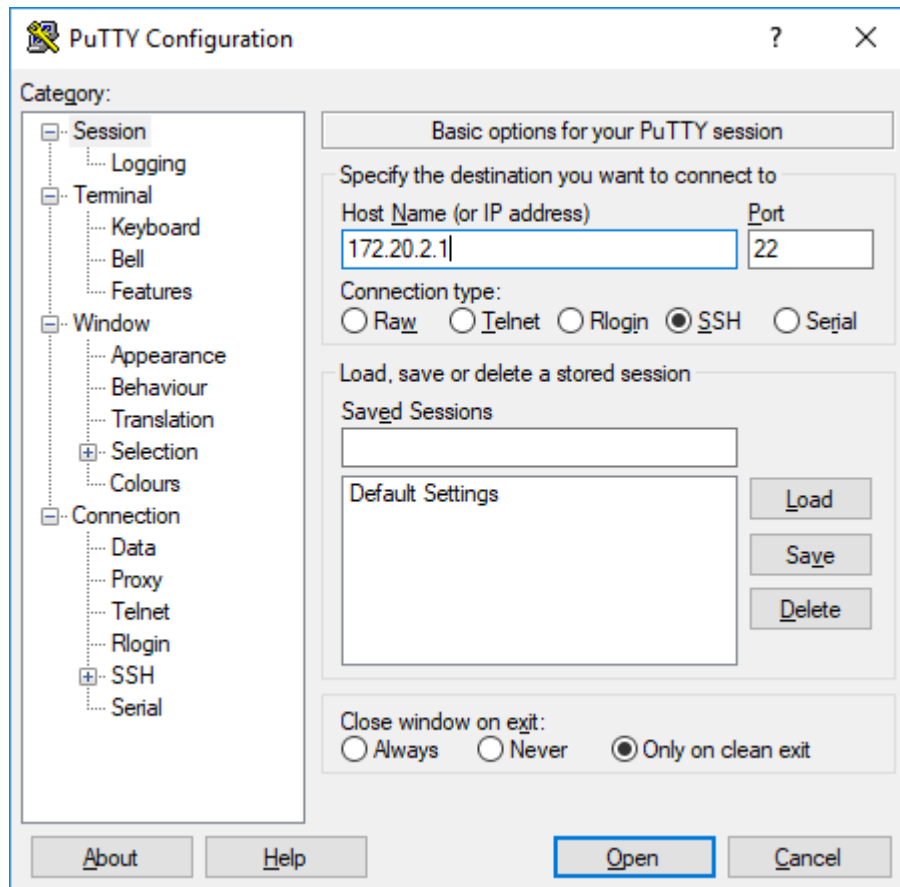
#### NOTE

The virtual machine generated in this chapter is a fresh installation and will be generated on the Nerve Device directly.

## Creating a logical volume

Logical volumes are created using the `lvcreate` command, which takes a number of command line arguments. Firstly, the `-L` flag is used to specify the size of the volume. Secondly, the `-n` flag is used to specify a name for the logical volume. `<image_size>` and `<volume_name>` are used as placeholders in the instructions below. The volume group is already predefined with `nerve` as its name.

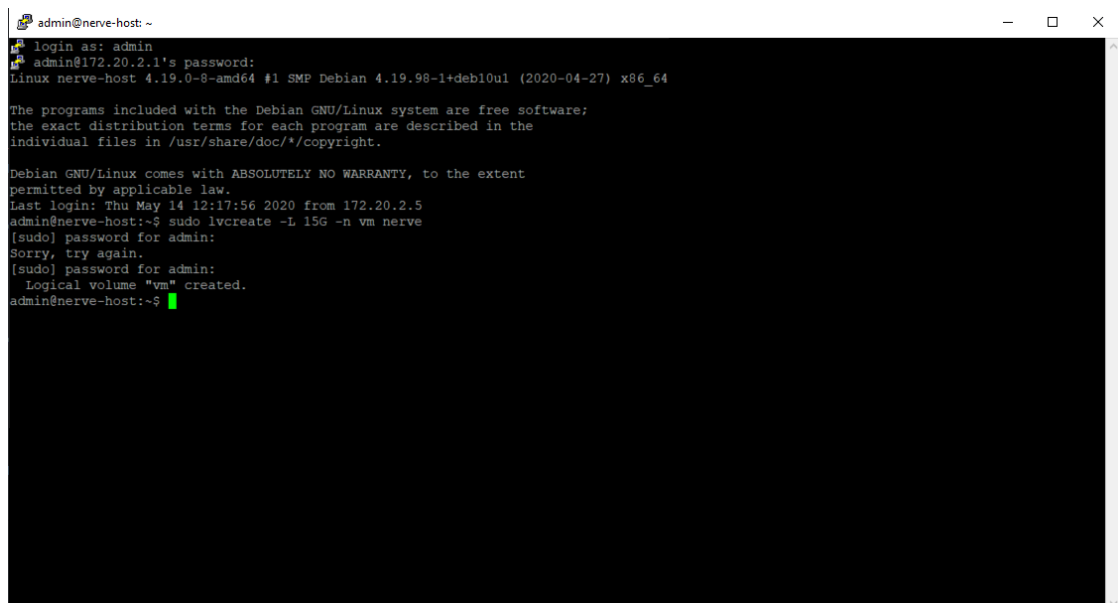
1. Open an SSH client like PuTTY.
2. Enter the IP address for host access to the Nerve Device under **Host Name (or IP address)** to log in to the host of the Nerve Device.



3. Log in with the credentials for host access to the Nerve Device.

4. Enter the following command:

```
sudo lvcreate -L <image_size> -n <volume_name> nerve
```

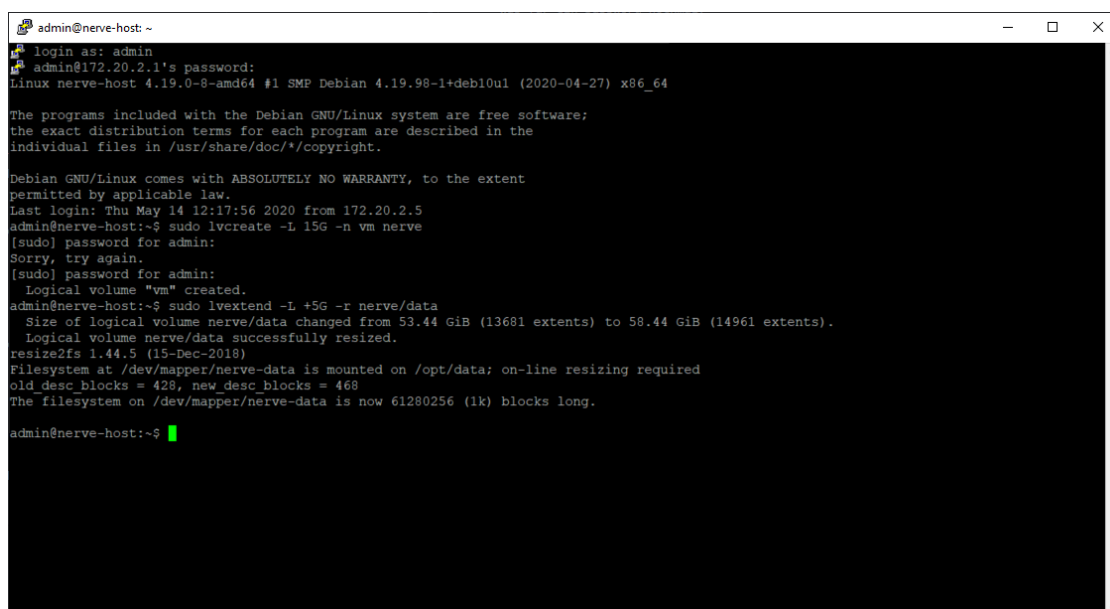


## Increasing storage in /opt/data and creating a filesystem

The ISO image of the OS needs to be copied to the device from which the virtual machine will be installed. In order to do that, the size of /opt/data needs to be increased. A path to store the ISO file will also be created. <size> is a placeholder for the amount of storage that is added, while nerve/data is the name of the volume group and the logical volume.

1. Enter the following command to increase storage:

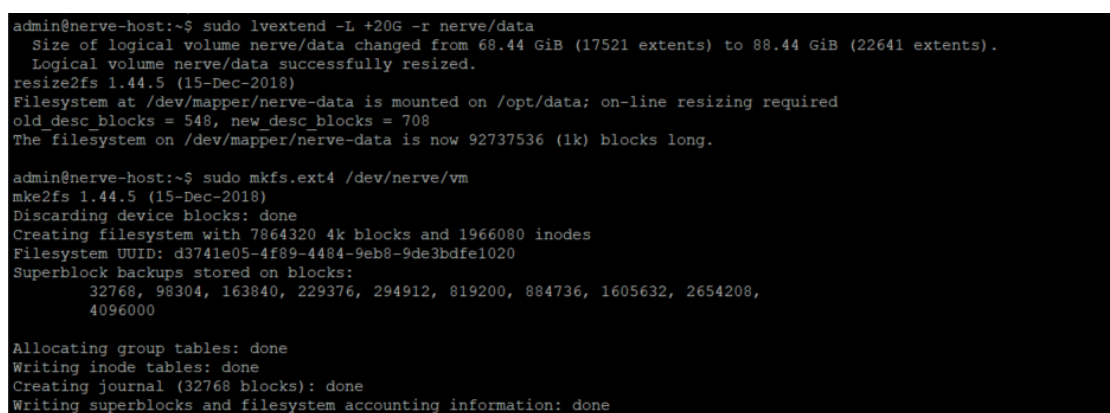
```
sudo lvextend -L +<size> -r nerve/data
```



```
admin@nerve-host:~  
login as: admin  
admin@172.20.2.1's password:  
Linux nerve-host 4.19.0-8-amd64 #1 SMP Debian 4.19.98-1+deb10u1 (2020-04-27) x86_64  
  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Thu May 14 12:17:56 2020 from 172.20.2.5  
admin@nerve-host:~$ sudo lvcreate -L 15G -n vm nerve  
[sudo] password for admin:  
Sorry, try again.  
[sudo] password for admin:  
Logical volume "vm" created.  
admin@nerve-host:~$ sudo lvextend -L +5G -r nerve/data  
Size of logical volume nerve/data changed from 53.44 GiB (13681 extents) to 58.44 GiB (14961 extents).  
Logical volume nerve/data successfully resized.  
resize2fs 1.44.5 (15-Dec-2018)  
Filesystem at /dev/mapper/nerve-data is mounted on /opt/data; on-line resizing required  
old_desc_blocks = 428, new_desc_blocks = 468  
The filesystem on /dev/mapper/nerve-data is now 61280256 (1k) blocks long.  
admin@nerve-host:~$
```

2. Enter the following command to create a filesystem:

```
sudo mkfs.ext4 /dev/nerve/<volume_name>
```



```
admin@nerve-host:~$ sudo lvextend -L +20G -r nerve/data  
Size of logical volume nerve/data changed from 68.44 GiB (17521 extents) to 88.44 GiB (22641 extents).  
Logical volume nerve/data successfully resized.  
resize2fs 1.44.5 (15-Dec-2018)  
Filesystem at /dev/mapper/nerve-data is mounted on /opt/data; on-line resizing required  
old_desc_blocks = 548, new_desc_blocks = 708  
The filesystem on /dev/mapper/nerve-data is now 92737536 (1k) blocks long.  
  
admin@nerve-host:~$ sudo mkfs.ext4 /dev/nerve/vm  
mke2fs 1.44.5 (15-Dec-2018)  
Discarding device blocks: done  
Creating filesystem with 7864320 4k blocks and 1966080 inodes  
Filesystem UUID: d3741e05-4f89-4484-9eb8-9de3bdf1020  
Superblock backups stored on blocks:  
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,  
4096000  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (32768 blocks): done  
Writing superblocks and filesystem accounting information: done
```



## Creating a directory for the ISO file and mounting the filesystem

Create a directory in /opt/data in order to be able to generate the IMG file and XML file that are required for the provisioning of the Virtual machine in the Management System

1. Enter the following command to create a directory for the ISO file:

```
sudo mkdir /opt/data/<directory>
```

2. Enter the following command to mount the filesystem:

```
sudo mount /dev/nerve/<volume_name> /opt/data/<directory>
```

## Granting permission for the newly created directory

Take ownership of the new directory in order to be able to work with it, e.g. copying the ISO file into the directory.

Enter the following command to take ownership of the created directory:

```
sudo chown admin:admin /opt/data/<directory>
```

```
admin@nerve-host:~$ sudo mkdir /opt/data/vm
admin@nerve-host:~$ sudo mount /dev/nerve/vm /opt/data/vm
admin@nerve-host:~$ sudo chown admin:admin /opt/data/v
var/ vm/
admin@nerve-host:~$ sudo chown admin:admin /opt/data/vm/
admin@nerve-host:~$
```

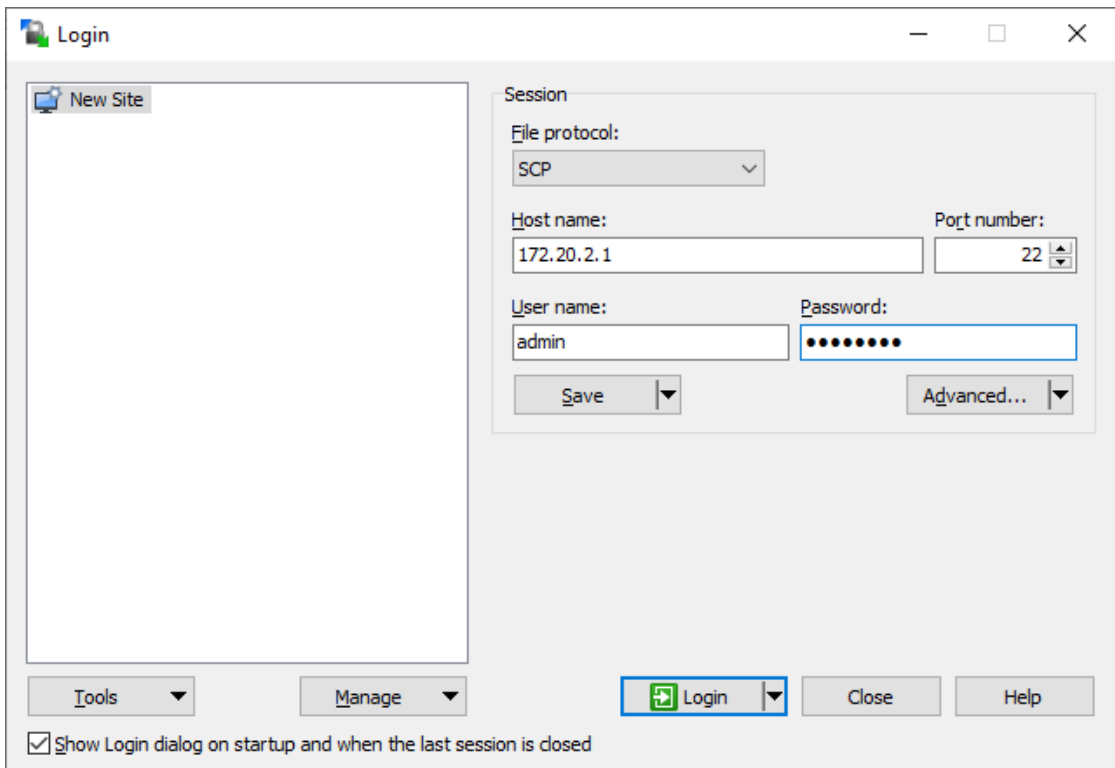
## Copying the ISO file to the Nerve Device

With the directory created and ownership established, copy the ISO file of the OS to the Nerve Device for the installation on the virtual machine.

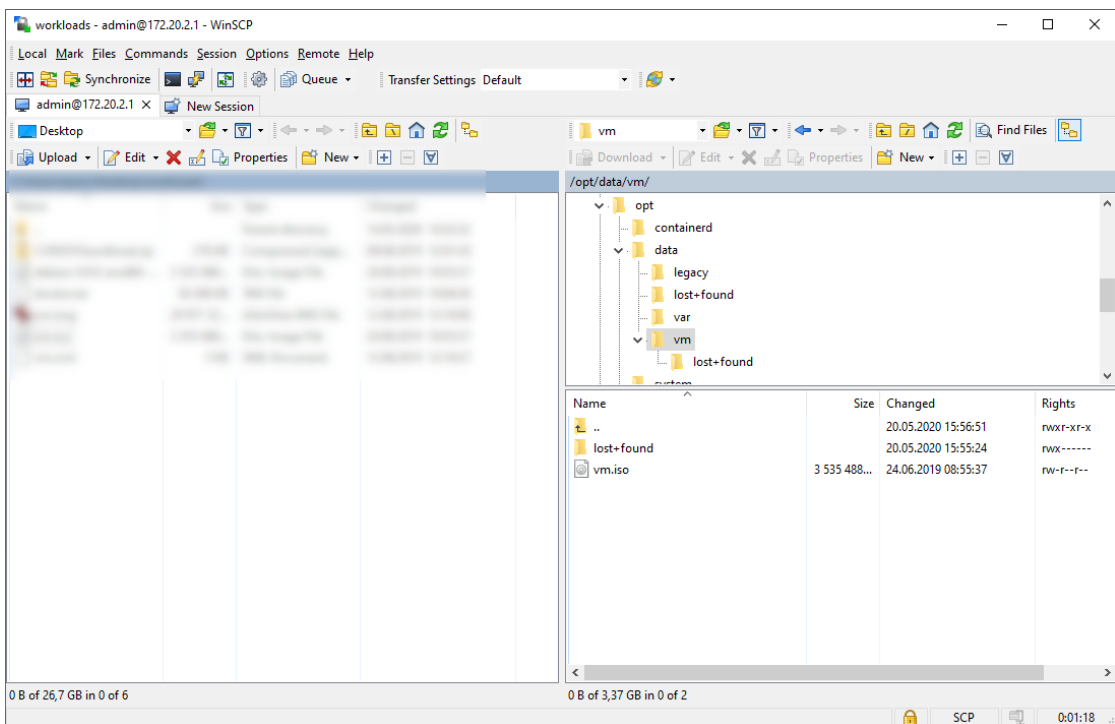
### NOTE

The instructions below are hardware specific. The MFN 100 is used as an example in the screenshots. Refer to the [device guide](#) for specific information on the Nerve Device.

1. Open a file transfer client like WinSCP.
2. Enter the IP address for host access to the Nerve Device under **Host Name**.
3. Enter the credentials for host access to the Nerve Device below under **User name** and **Password**.



4. Navigate to the **/opt/data/<directory>** directory on the right side of the WinSCP window. It is located in the **root** directory.

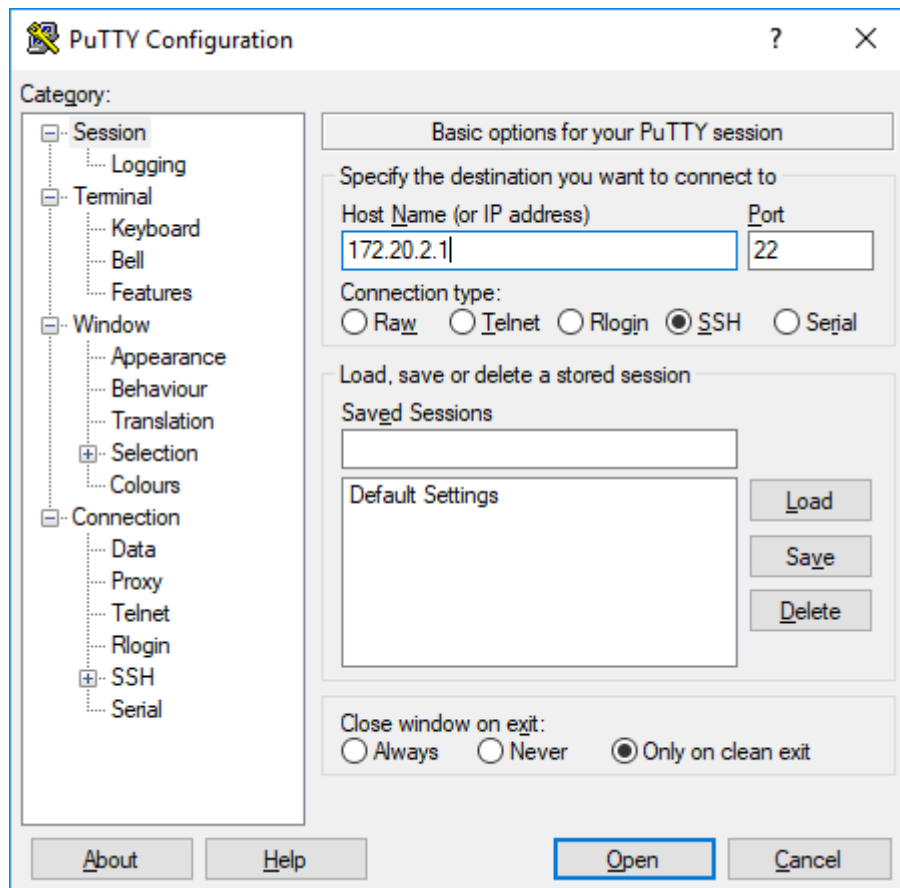


5. Copy the ISO file of the OS that is to be installed on the virtual machine to the directory on the Nerve Device.

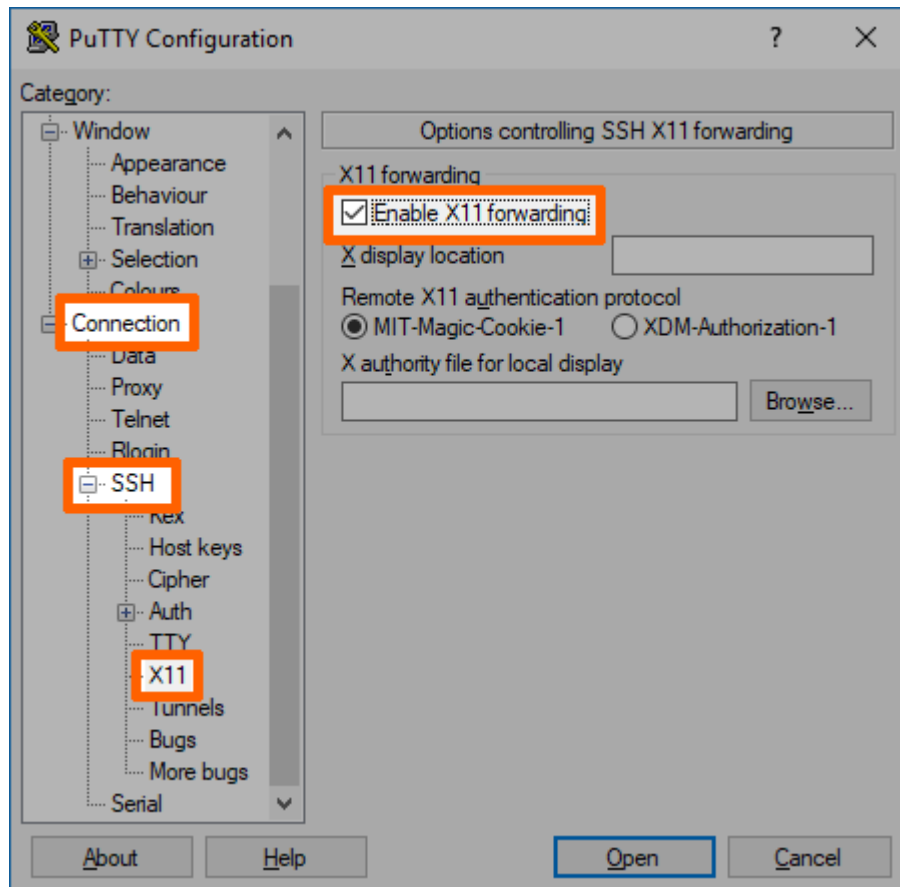
## Creating a virtual machine on a node

Using the Virtual Machine Manager is recommended to create a virtual machine and install the OS from the ISO file. Note that the virtual machine in this chapter is a fresh installation and will be generated on the Nerve Device directly.

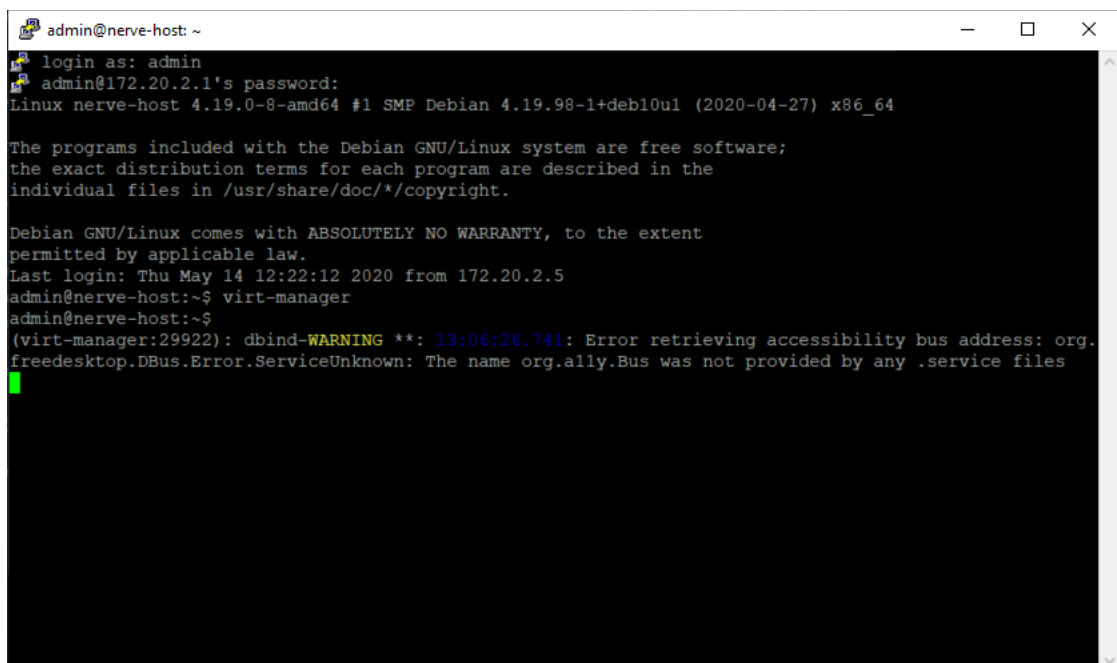
1. Run Xming or an alternative.
2. Open an SSH client like PuTTY.
3. Enter the IP address for host access to the Nerve Device under **Host Name (or IP address)** to log in to the host of the Nerve Device.



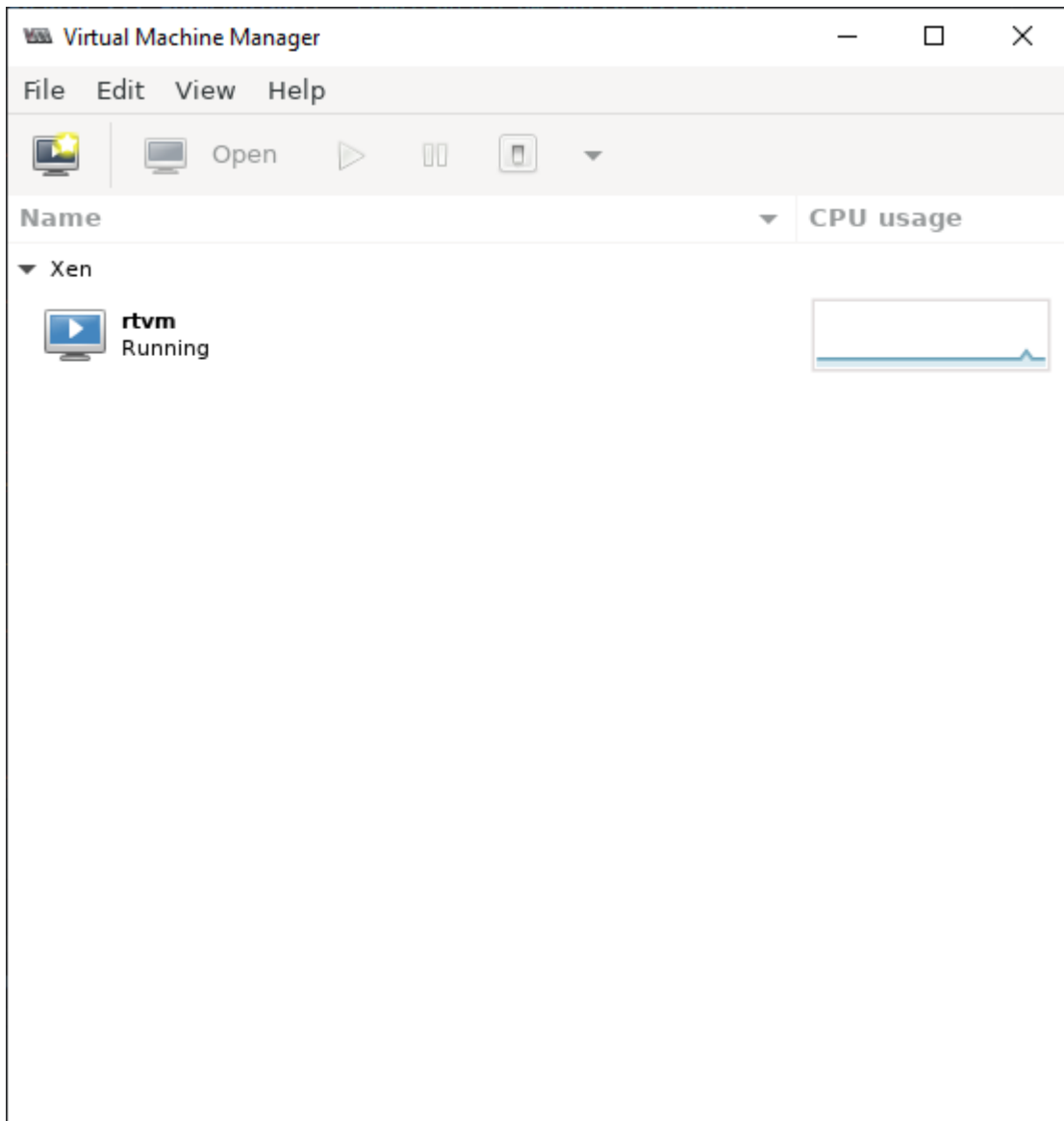
4. Expand **Connection > SSH > X11** on the left side.
5. Tick the checkbox next to **Enable X11 forwarding**.



6. Click **Open**.
7. Log in with the credentials for host access to the Nerve Device.
8. Enter `virt-manager`.



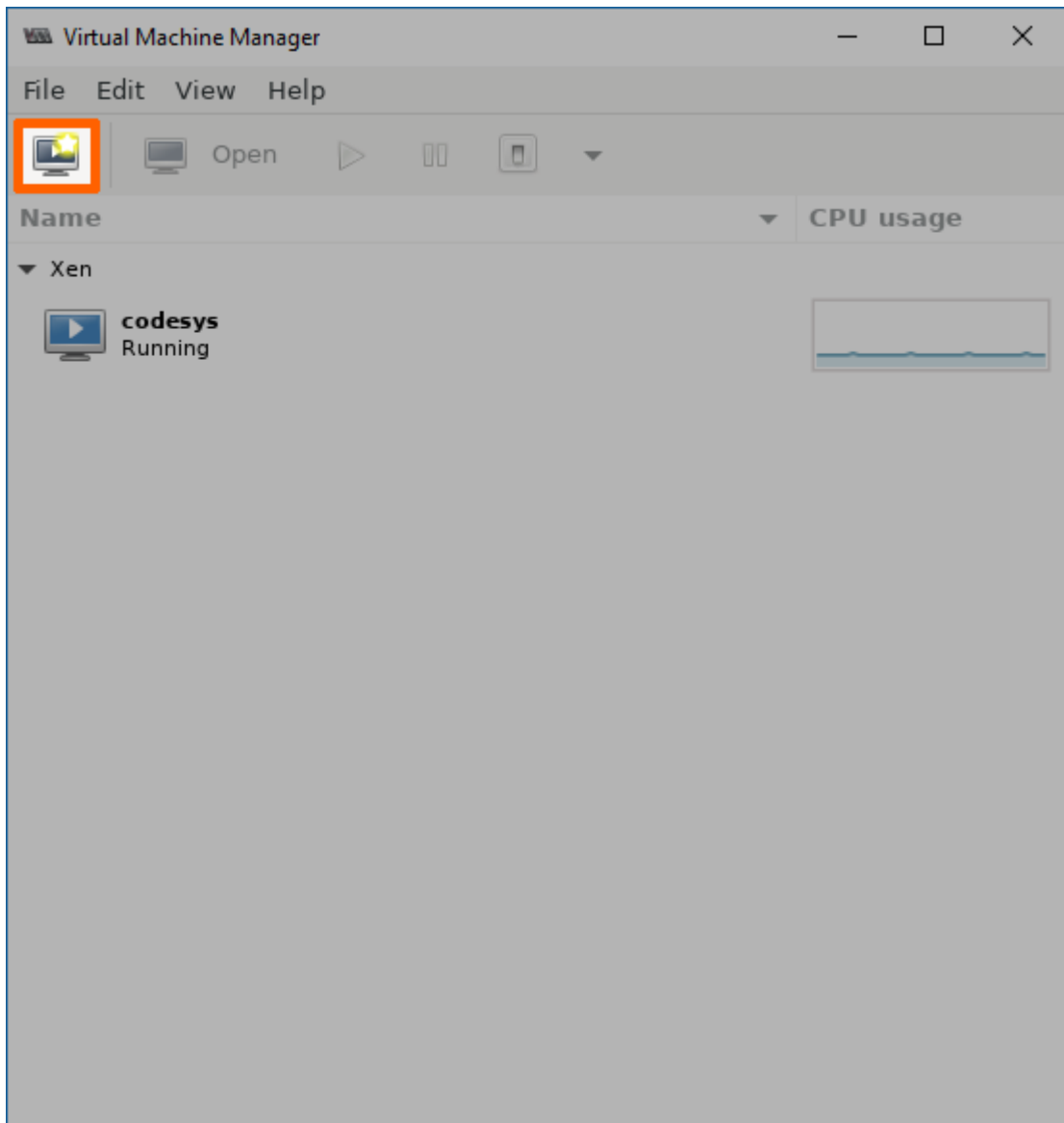
The interface of the Virtual Machine Manager will open.



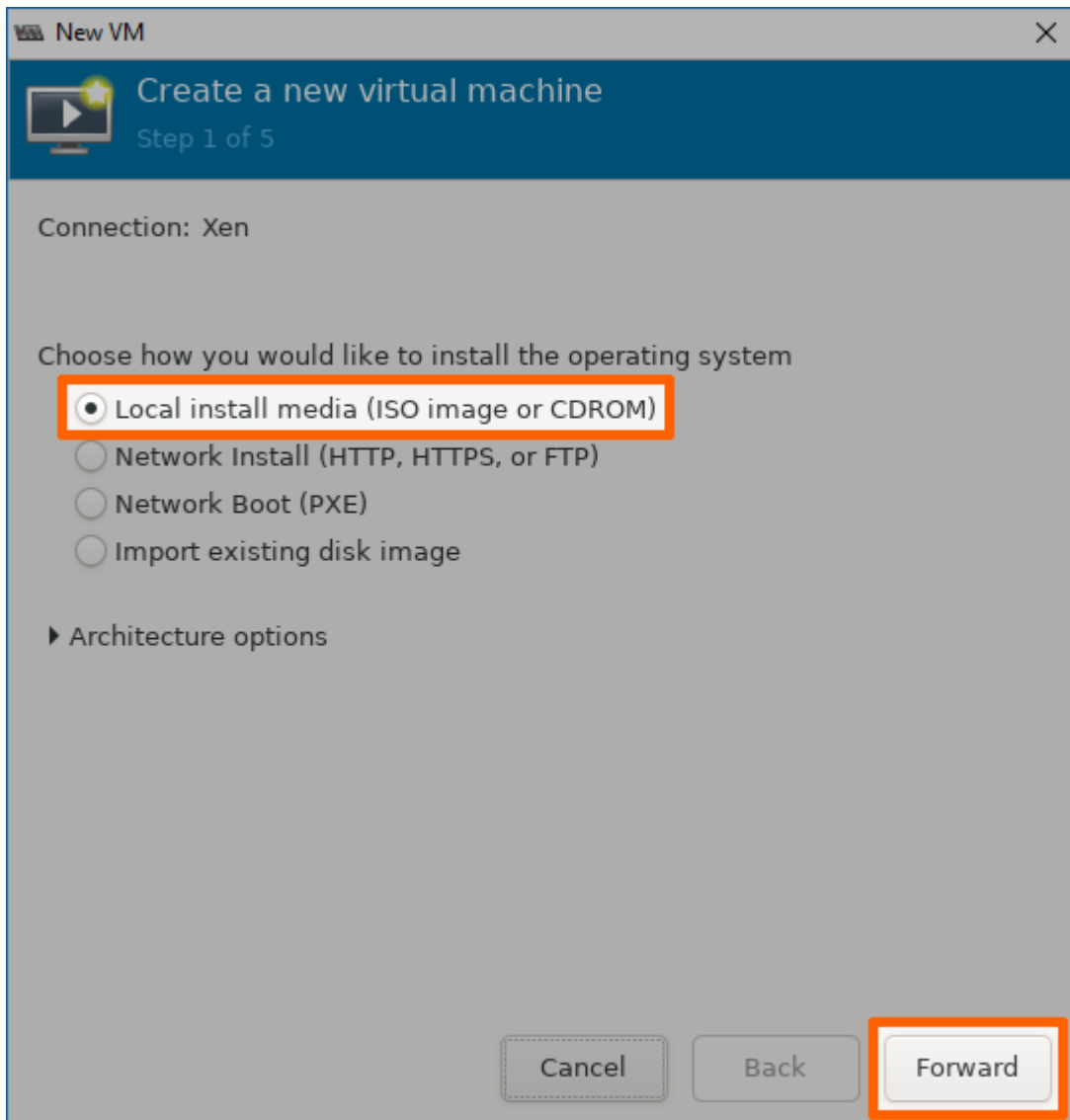
## Inserting the installation file (ISO)

The creation of the virtual machine can now be initiated with the installation of the OS following right after. Note that the Virtual Machine Manager requires the virtual insertion of the ISO file in the beginning while resources for the virtual machine are defined later.

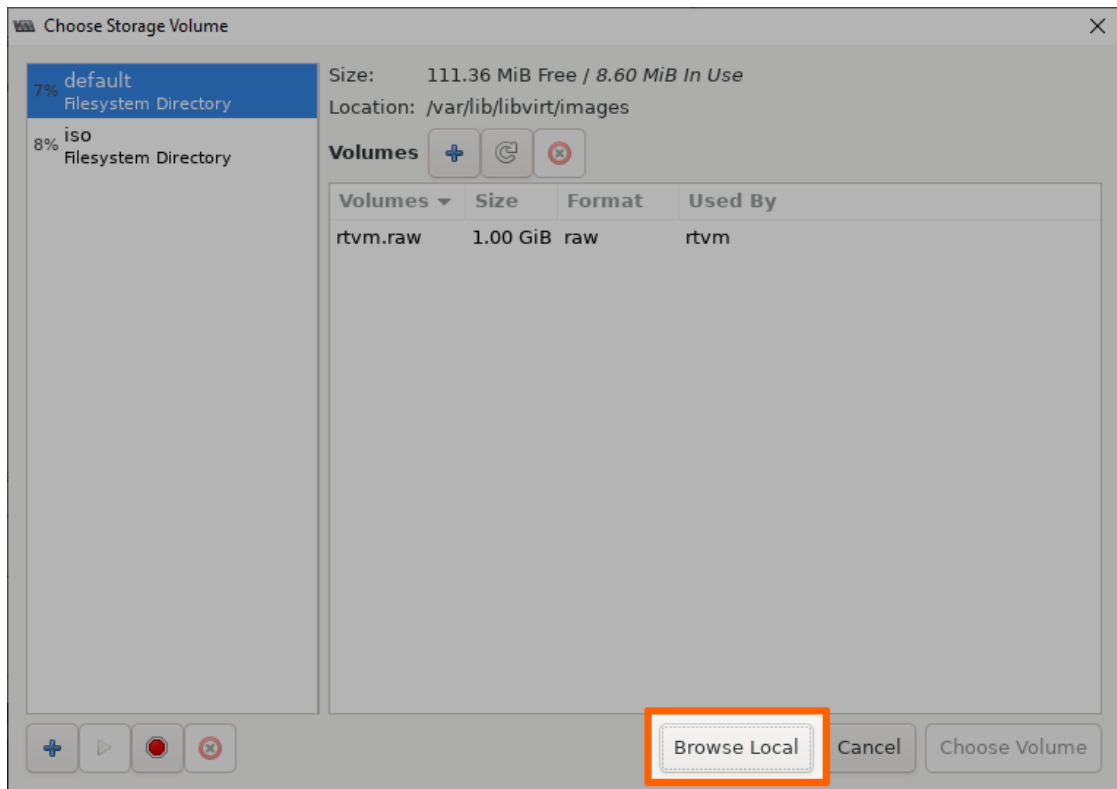
1. Select **File > New Virtual Machine** or click the symbol.



2. Select **Local install media (ISO image or CDROM)**.
3. Click **Forward**.

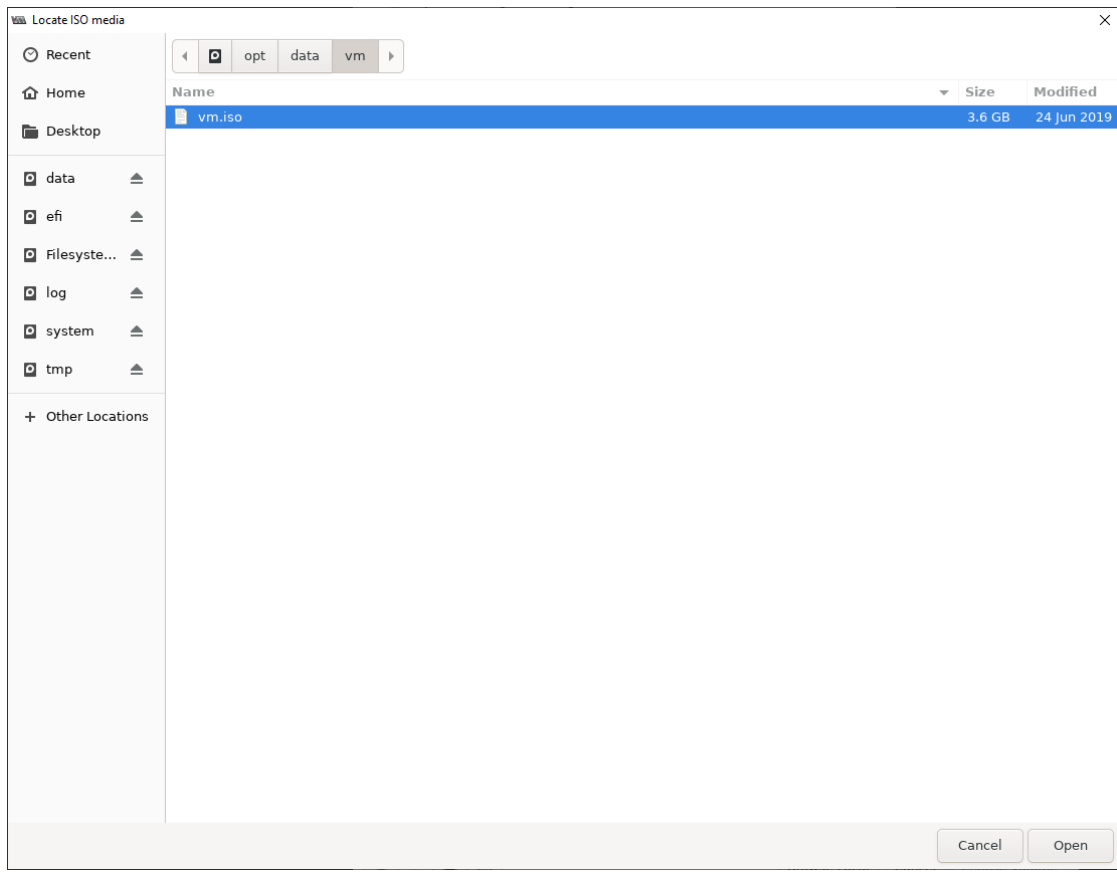


4. Click **Browse...** in the next window.
5. In the new window select **Browse Local**.



6. Navigate to **/opt/data/** by clicking the left arrow next to **admin**. The **opt** directory is located in the root directory.
7. Double-click the directory containing the ISO file.
8. Select the ISO file of the OS that was copied before.



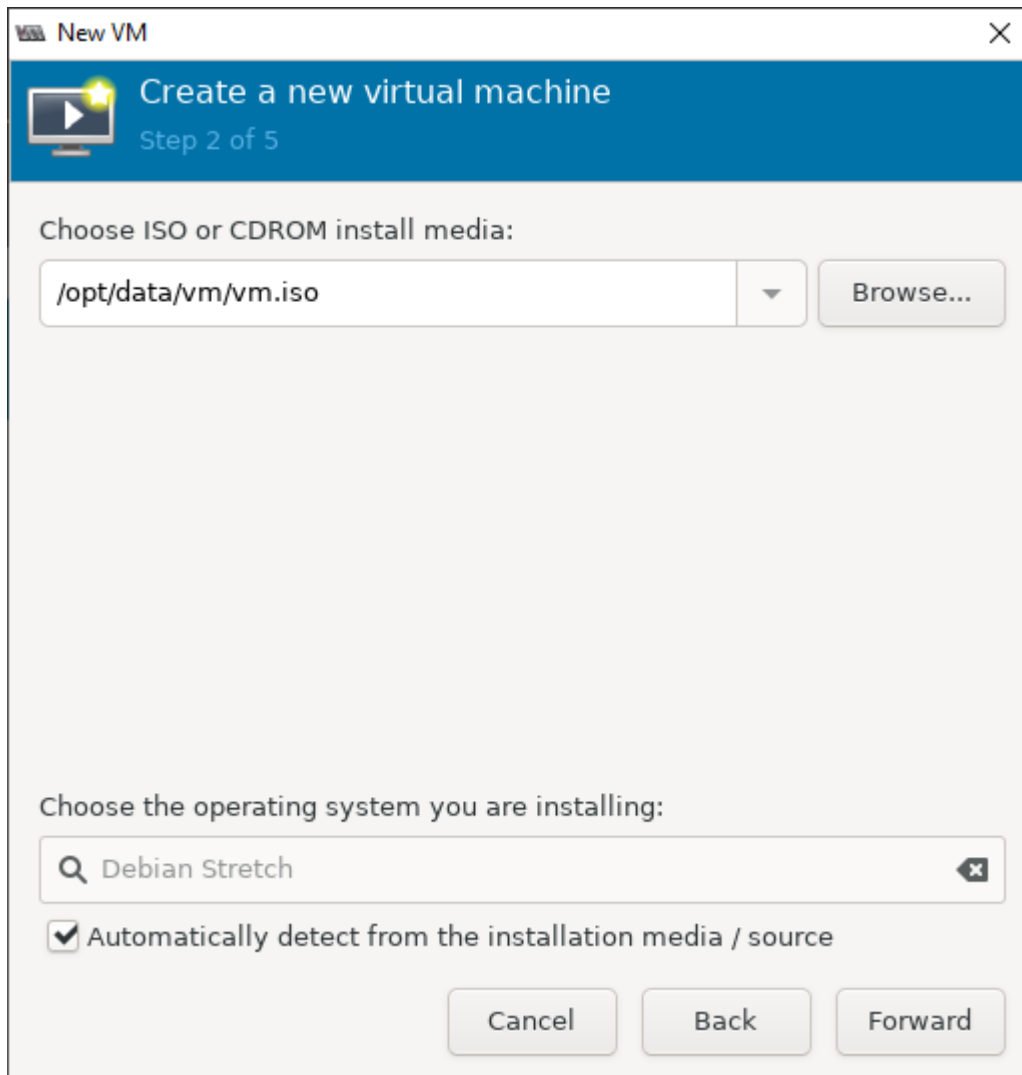


9. Click **Open**.

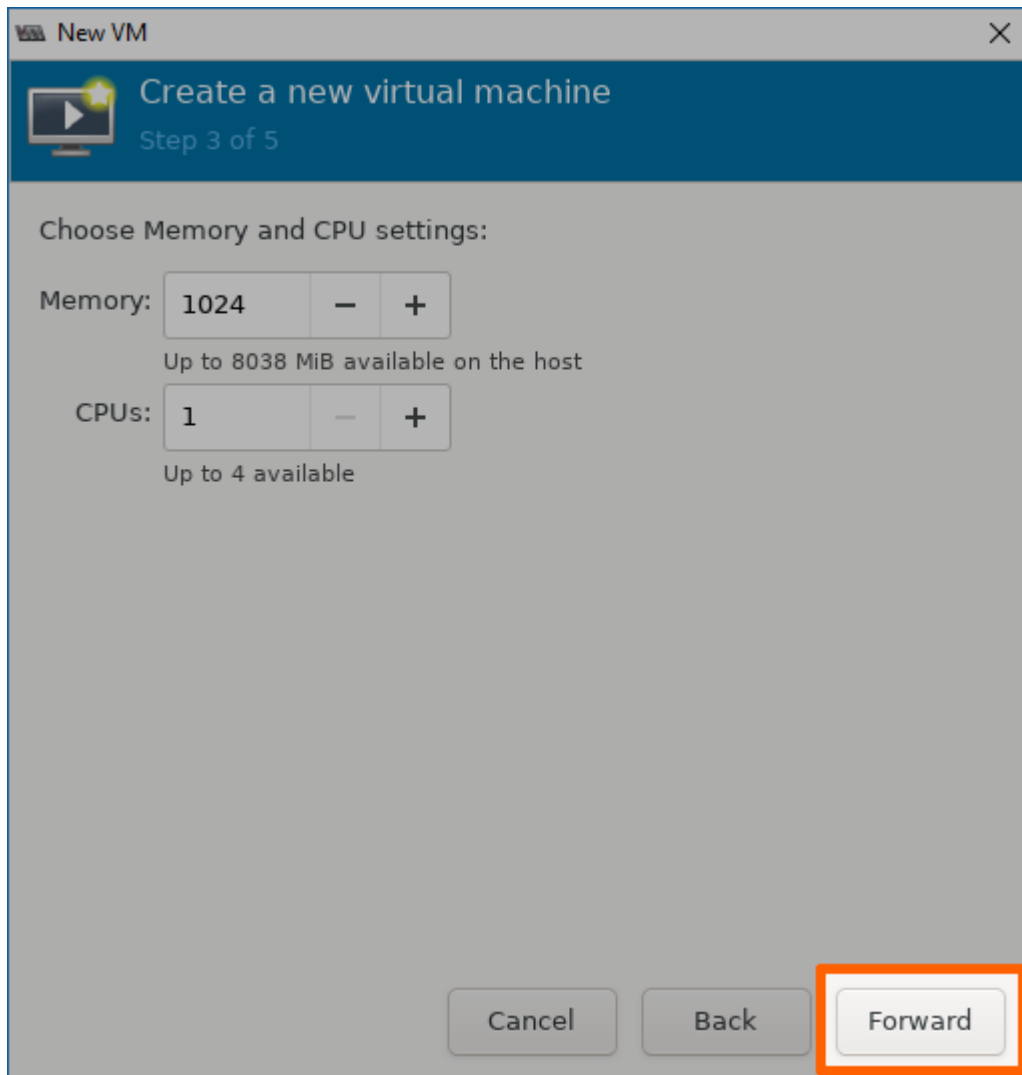
## Defining resources for the virtual machine

Next, the amount of memory and the number of CPUs need to be defined, and the logical volume assigned to the virtual machine needs to be selected.

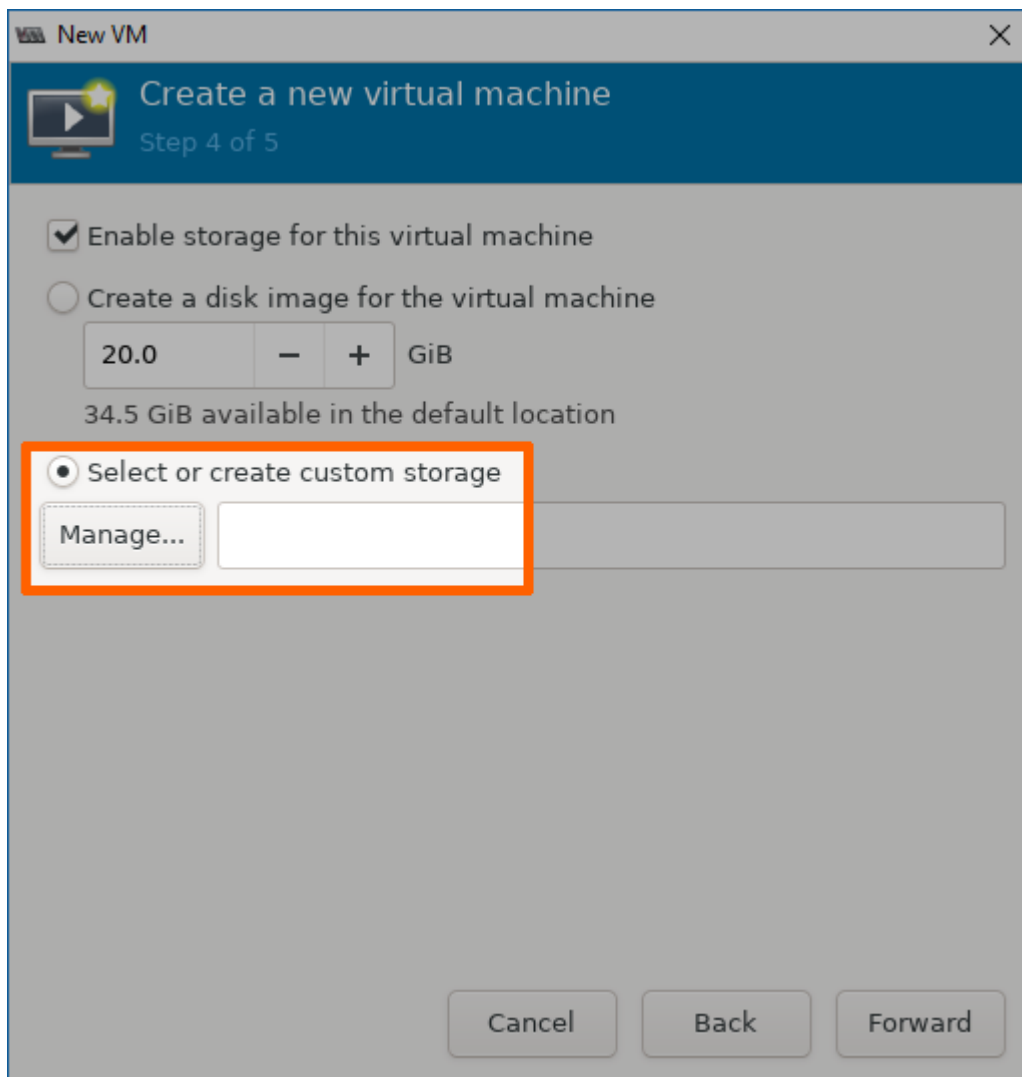
1. Click **Forward**.



2. Define how much memory and how many CPUs to assign to this virtual machine.
3. Select **Forward**.



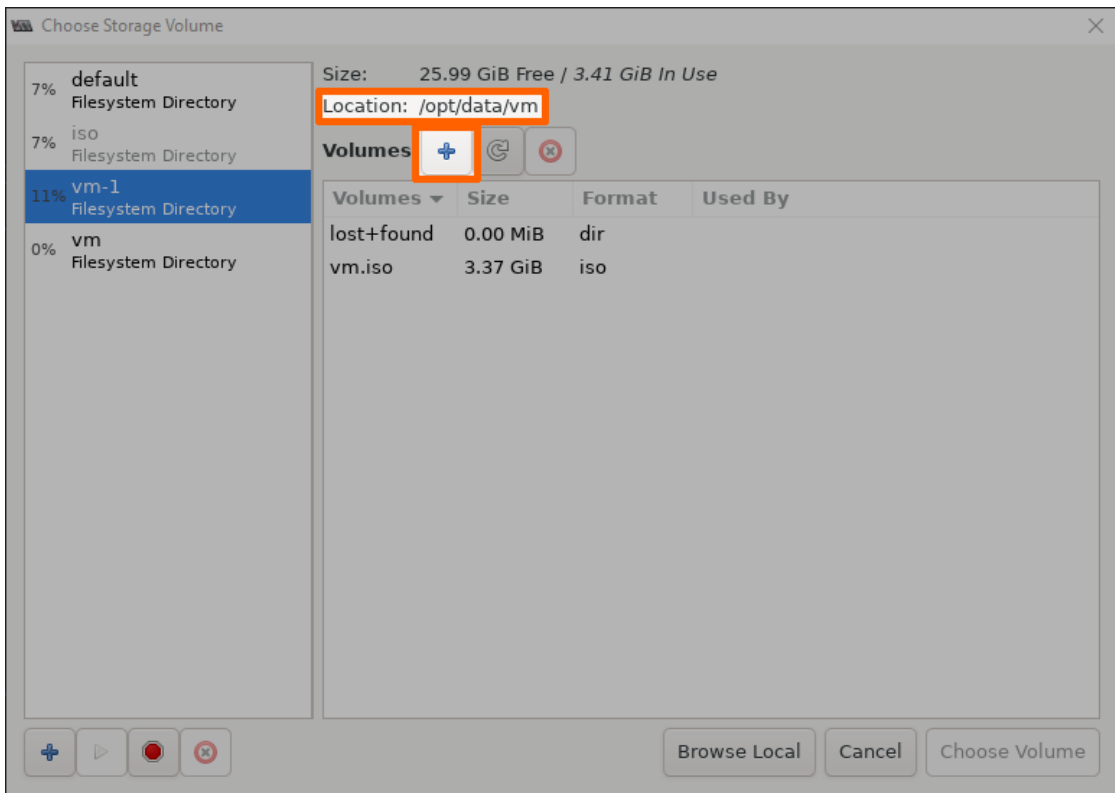
4. Click **Select or create custom storage**.
5. Select **Manage...**



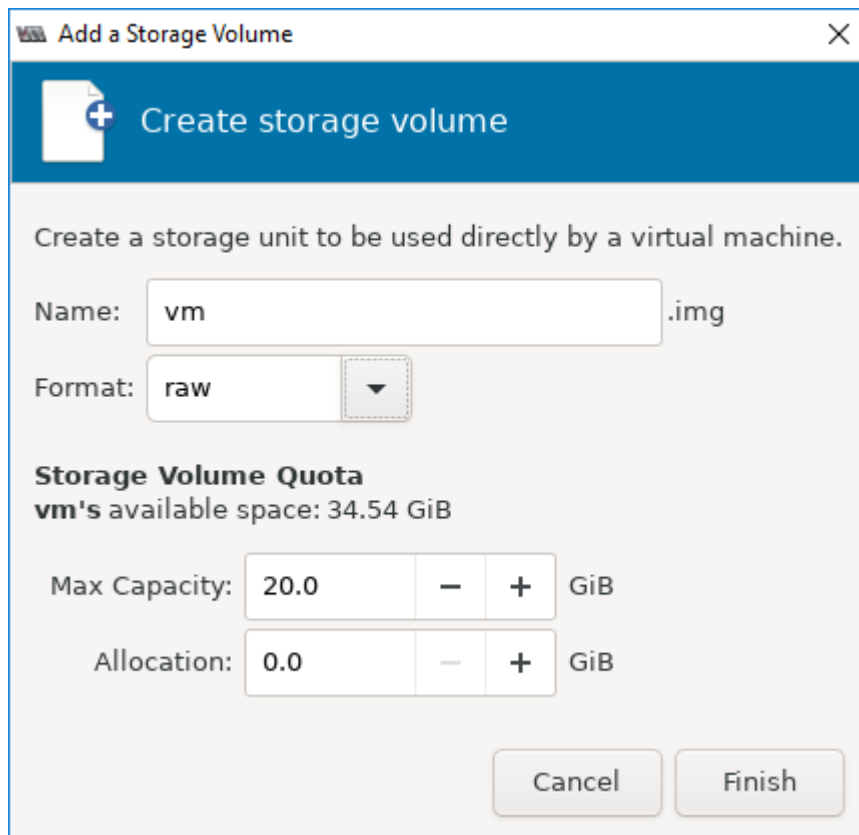
## Installing the operating system

Now the virtual machine will be initiated and the installation of the OS will be started.

1. Select the pool that was created in the command line on the left side in the **Choose Storage Volume** window. Make sure it is the volume with the location **opt/data/**.
2. Select the plus symbol in the middle of the screen.



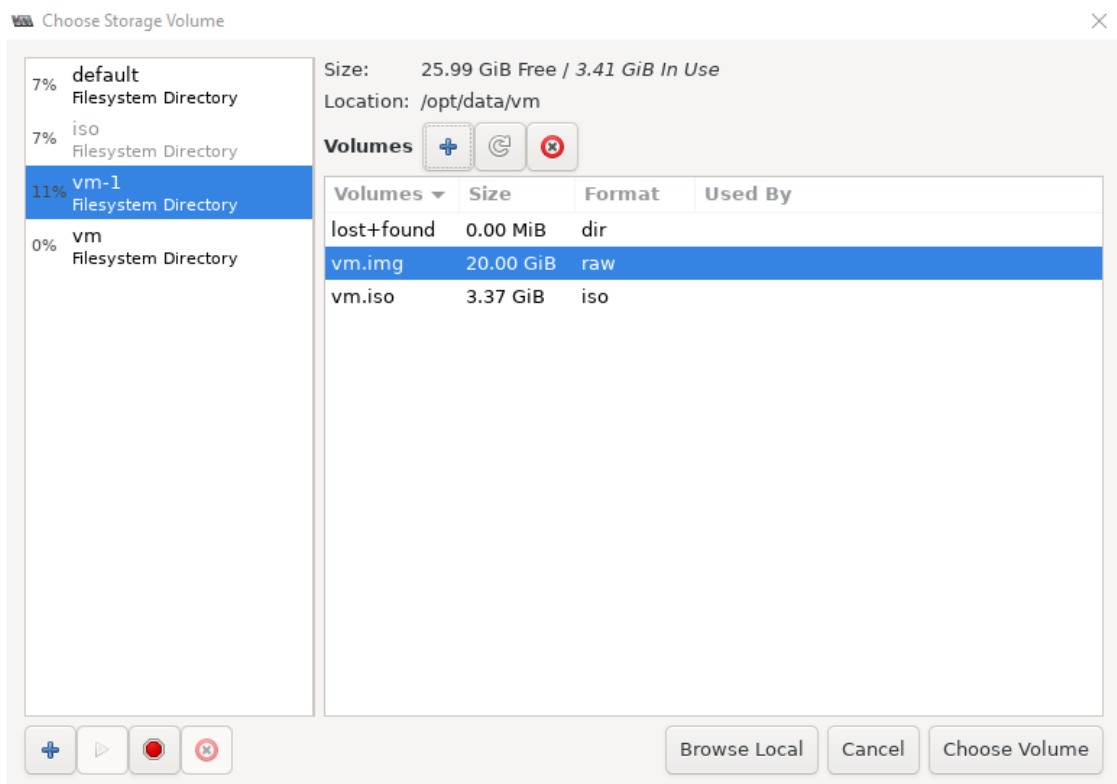
3. Enter a name for the IMG file.



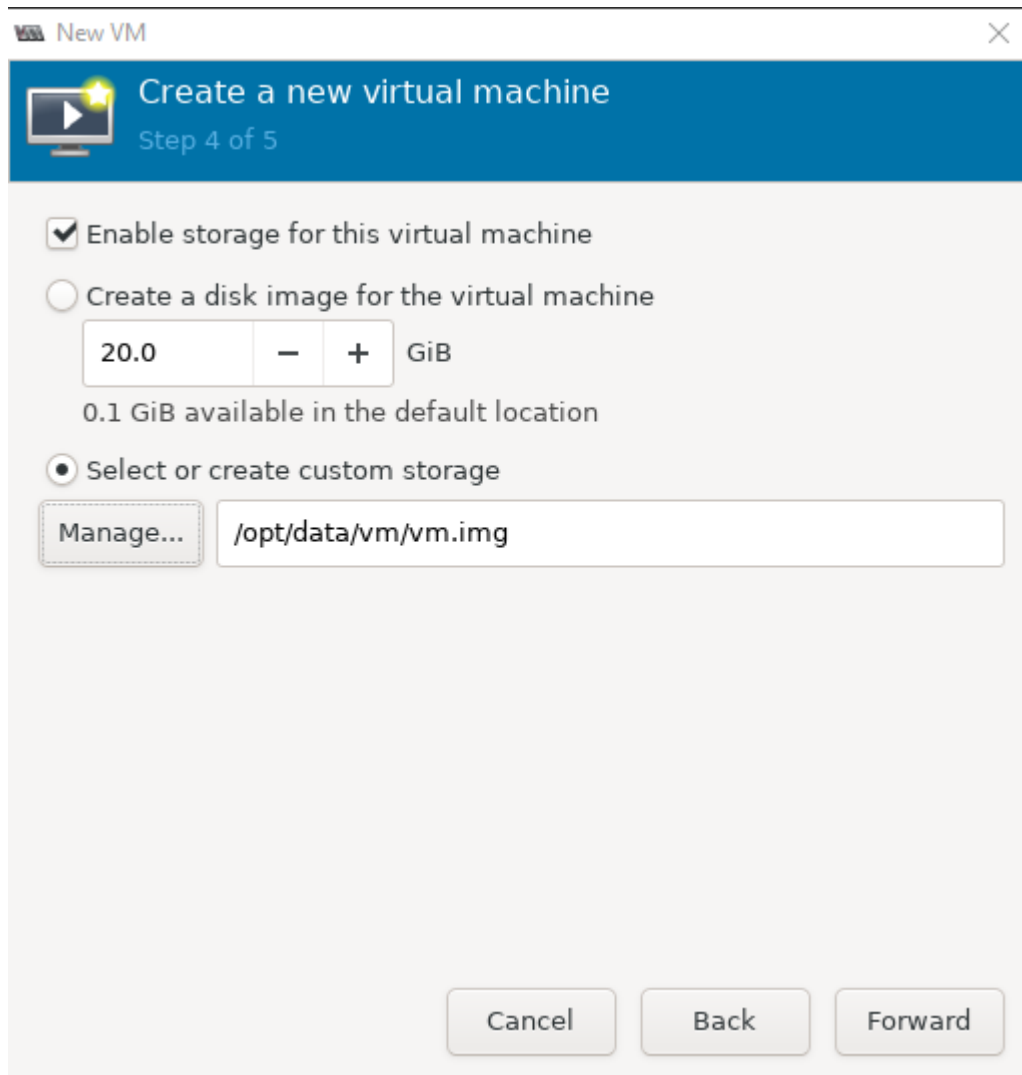
**NOTE**

Make sure that the value entered for **Max Capacity** is a factor of 512 Bytes. Any value that is not a factor of 512 B will cause an error when the virtual machine workload is deployed. In other words, do not enter any decimal value except .0 or .5.

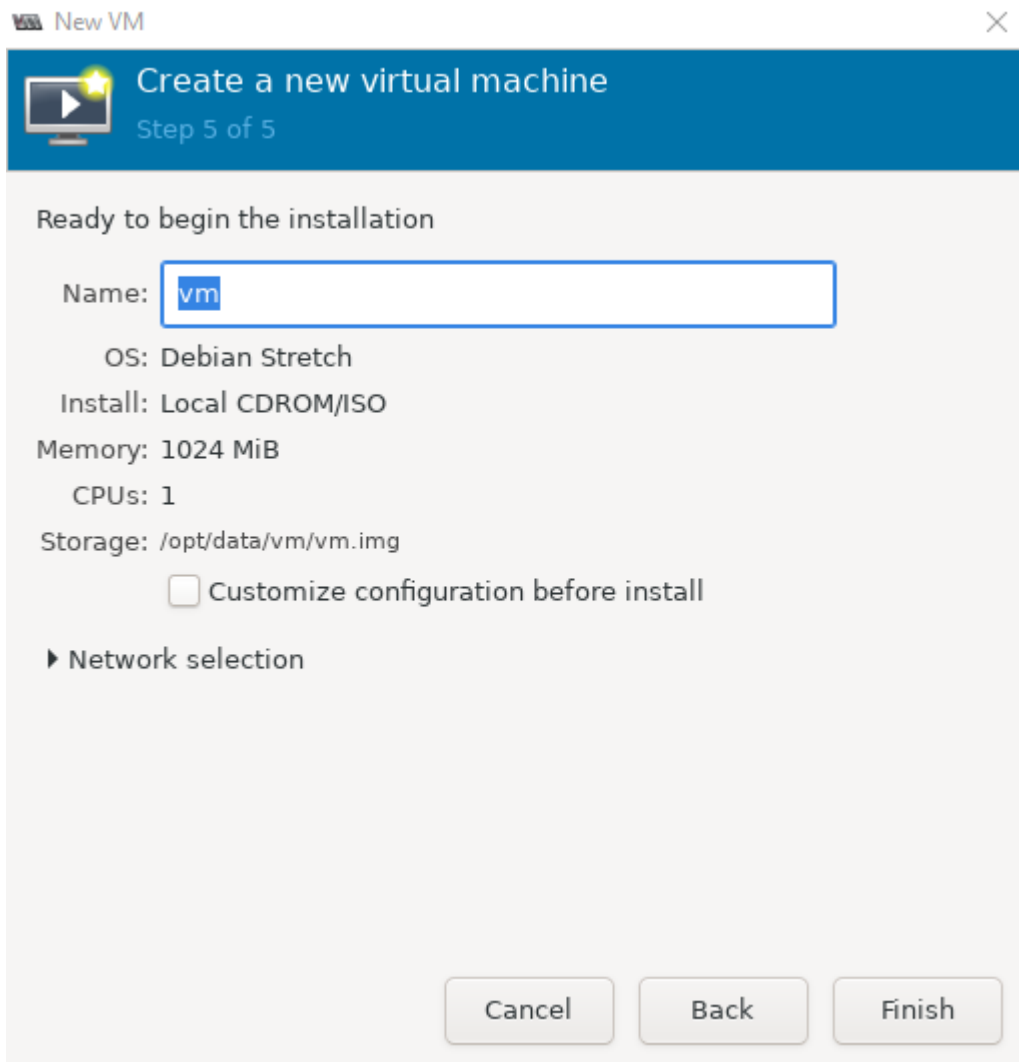
4. Select **Finish**.
5. Select the IMG file from the list in the middle.
6. Click **Choose Volume**.



7. Click **Forward** to initialize the installation of the OS on the virtual machine.

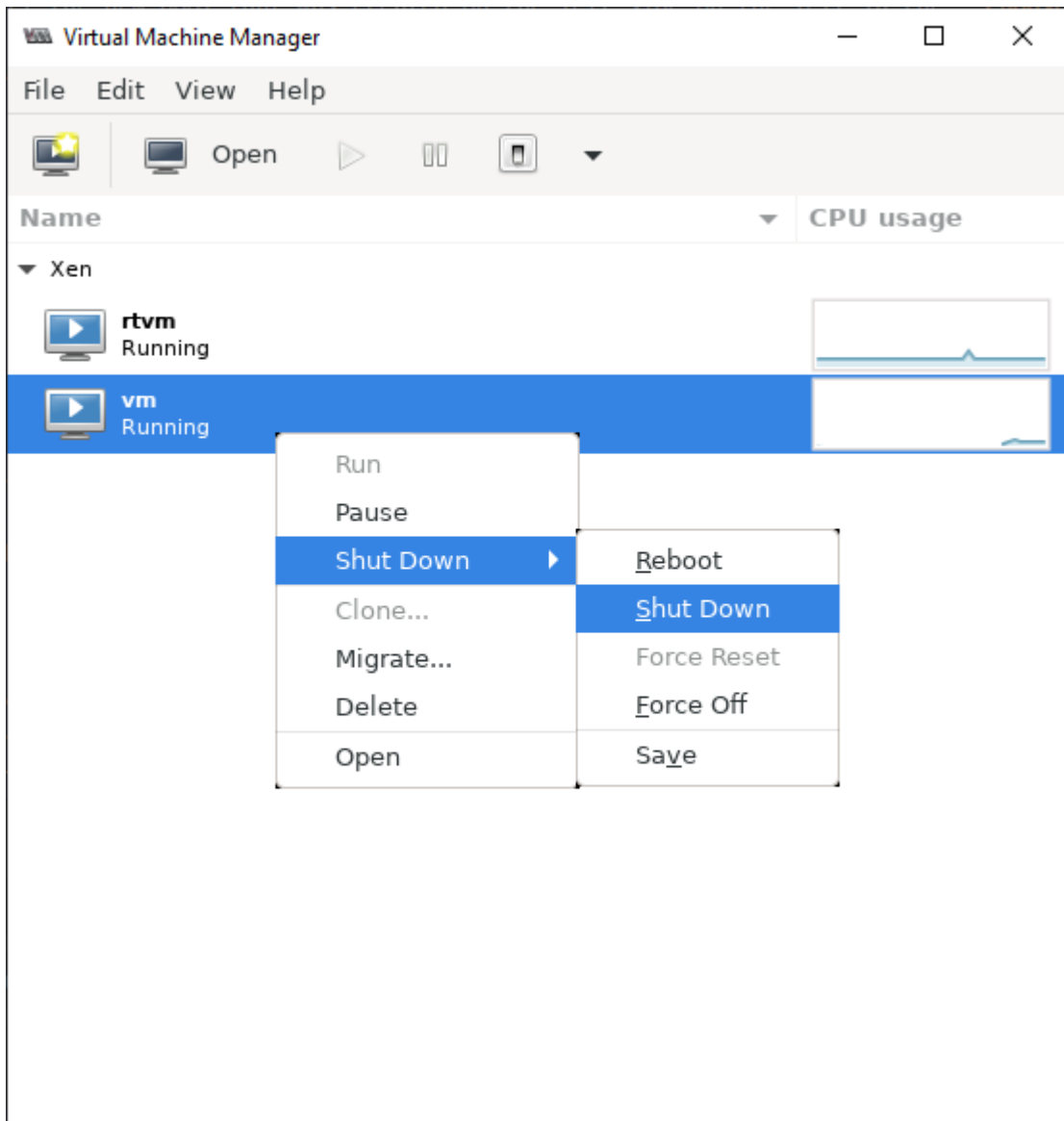


8. Enter a name for the virtual machine.
9. Click **Finish**. The virtual machine will be initiated and the installation of the OS will be started.



10. Complete the installation of the OS. Follow the instructions provided by the vendor.
11. After the installation is completed, right-click the virtual machine in the main Virtual Machine Manager window.
12. Select **Shut Down > Shut Down** to shut down the VM.

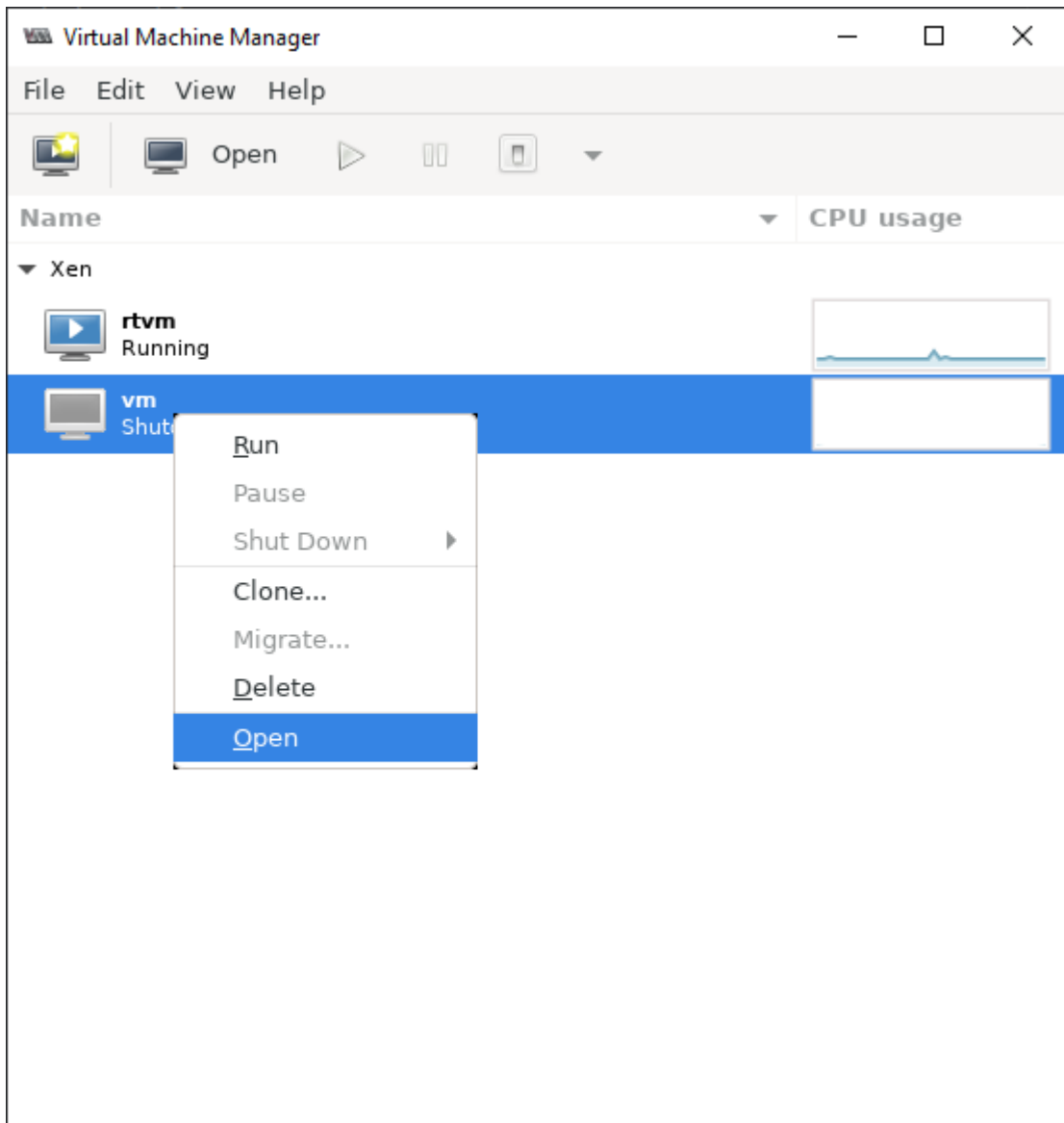




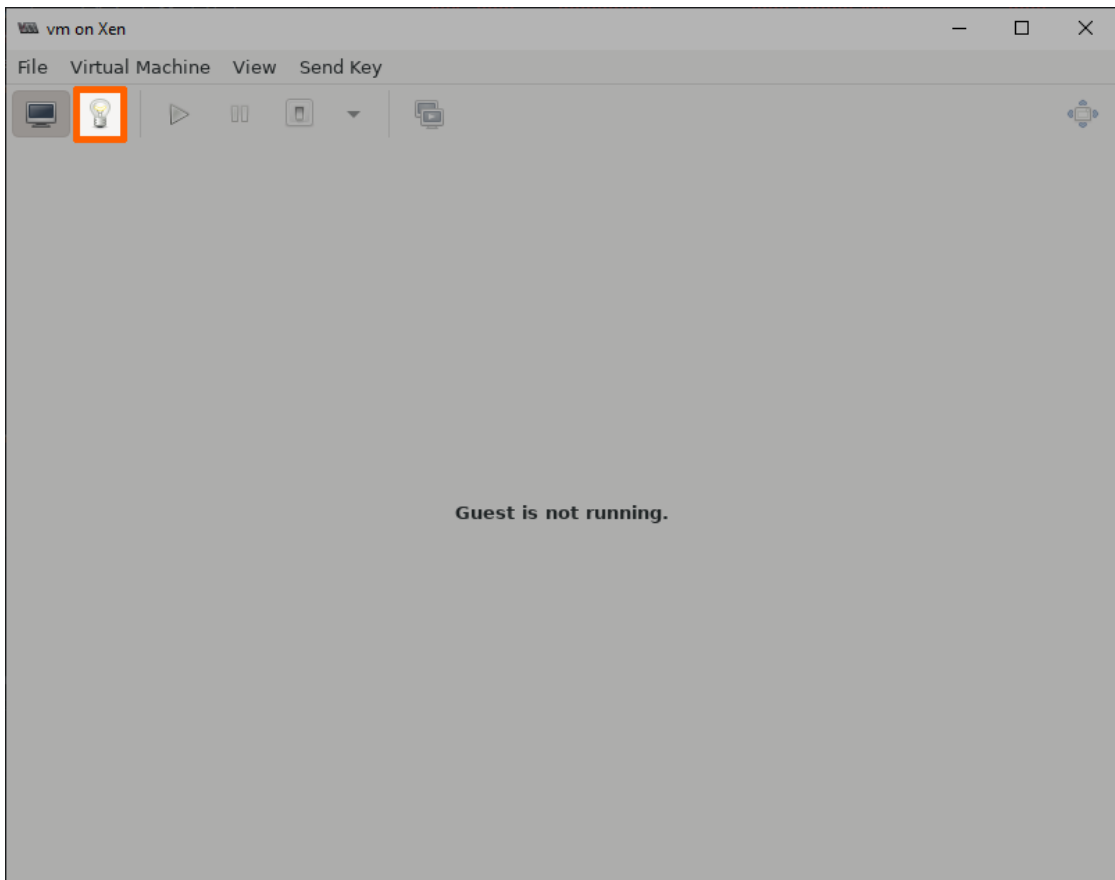
## Defining network interfaces

In order for the VM to communicate with the Nerve network, it needs network interfaces. While these are defined when provisioning the VM workload in the Management System, the VM itself needs to have "network cards" installed that represent these interfaces. This means that the purpose of the VM and its interfaces, the number of interfaces and the Nerve networks that will be connected to them need to be known before attempting the following instructions. Note that this step also requires a user with knowledge about the VM, as it requires configuration inside of the VM.

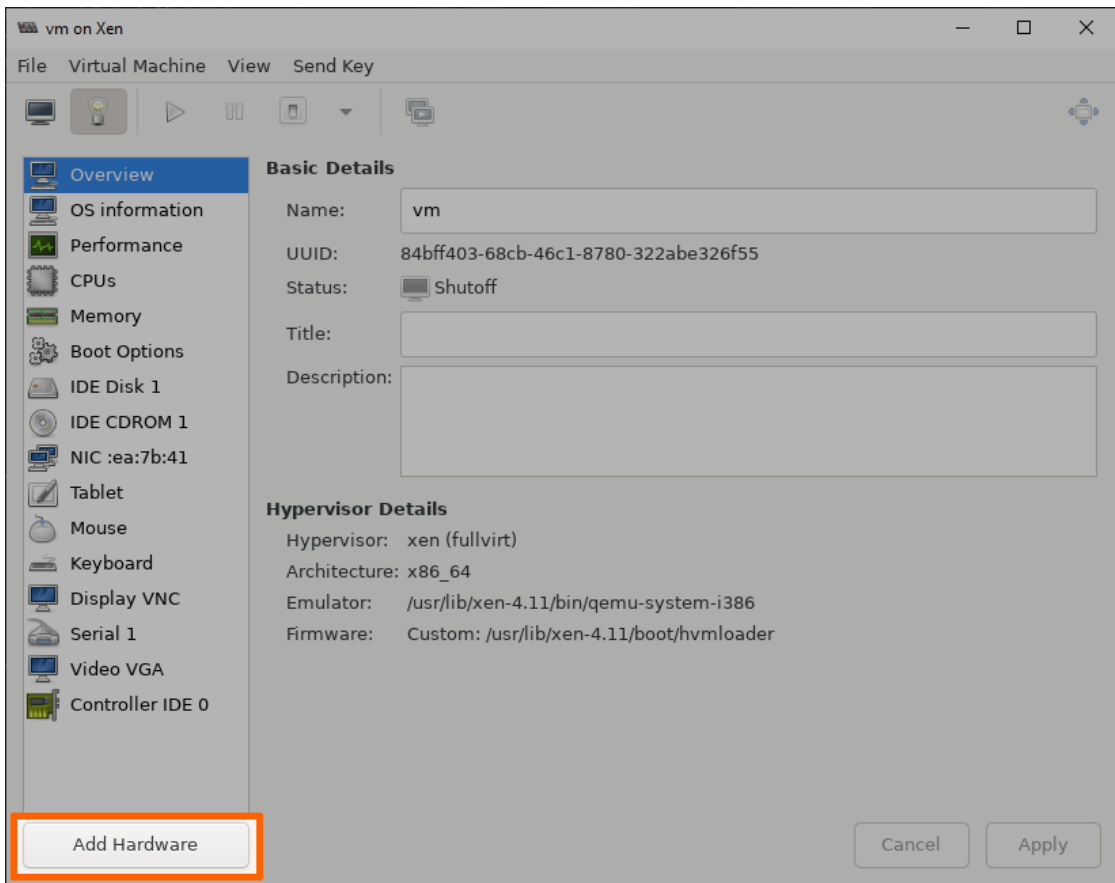
1. Right-click the virtual machine in the main Virtual Machine Manager window.
2. Select **Open** to open the control window.



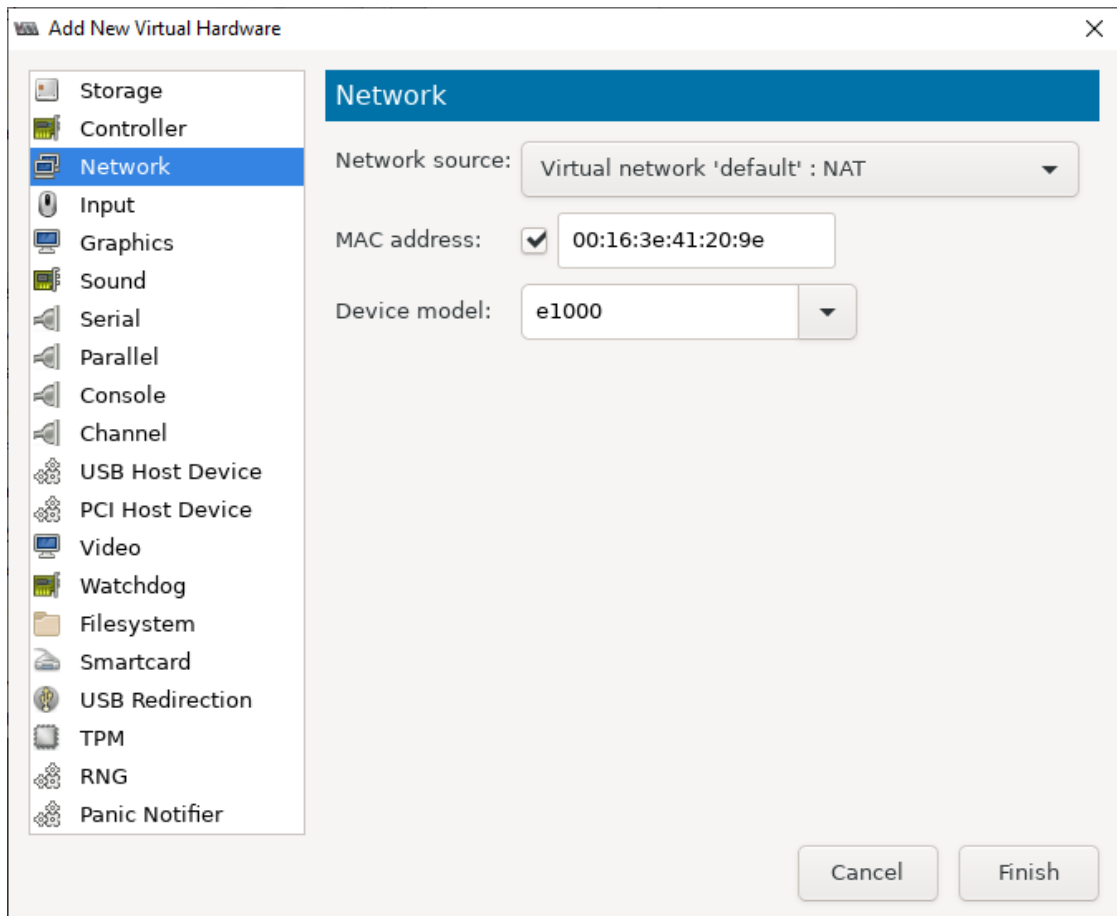
3. Select the light bulb symbol in the toolbar to display hardware details.



4. Select **Add Hardware** in the lower-left.



5. Select **Network** in the list.



6. Configure the **Network source** the following way depending on the desired interface. Refer to [Networks for Virtual Machine workloads](#) for more information on the interfaces.

Interface	Action
<b>mgmt</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu</li> <li>2. Enter mgmt in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>wan</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter wan in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>extern1</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter extern1 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>extern2</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter extern2 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>extern3</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter extern3 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>

Interface	Action
<b>default</b>	<ol style="list-style-type: none"> <li>1. Select <b>Virtual network 'default' : NAT</b> from the drop-down menu.</li> <li>2. Select <b>Finish</b> to add the network card.</li> </ol>
<b>extern1-nat</b>	<ol style="list-style-type: none"> <li>1. Select <b>Virtual network 'extern1-nat' : NAT to extern1</b> from the drop-down menu.</li> <li>2. Select <b>Finish</b> to add the network card.</li> </ol>
<b>extern2-nat</b>	<ol style="list-style-type: none"> <li>1. Select <b>Virtual network 'extern2-nat' : NAT to extern2</b> from the drop-down menu.</li> <li>2. Select <b>Finish</b> to add the network card.</li> </ol>
<b>extern3-nat</b>	<ol style="list-style-type: none"> <li>1. Select <b>Virtual network 'extern3-nat' : NAT to extern3</b> from the drop-down menu.</li> <li>2. Select <b>Finish</b> to add the network card.</li> </ol>
<b>rtvm</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter br-rtvm in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>isolated1</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu</li> <li>2. Enter br-isolated1 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>isolated2</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter br-isolated2 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>isolated3</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter br-isolated3 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>isolated4</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter br-isolated4 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>
<b>isolated5</b>	<ol style="list-style-type: none"> <li>1. Select <b>Specify shared device name</b> from the drop-down menu .</li> <li>2. Enter br-isolated5 in the <b>Bridge name:</b> field.</li> <li>3. Select <b>Finish</b> to add the network card.</li> </ol>

#### NOTE

The device model can be ignored, as the Management System will set the required parameters once the VM is provisioned as a workload.

7. Repeat steps 4 to 6 until all desired interfaces have been defined.

#### NOTE

Note that one virtual network card is defined by default when creating the virtual machine. The **Network source** is set to **Virtual network 'default' :**

**NAT** so that the VM can have internet access through the **default** network. Remove or modify this network card if internet access for the VM is not desired.

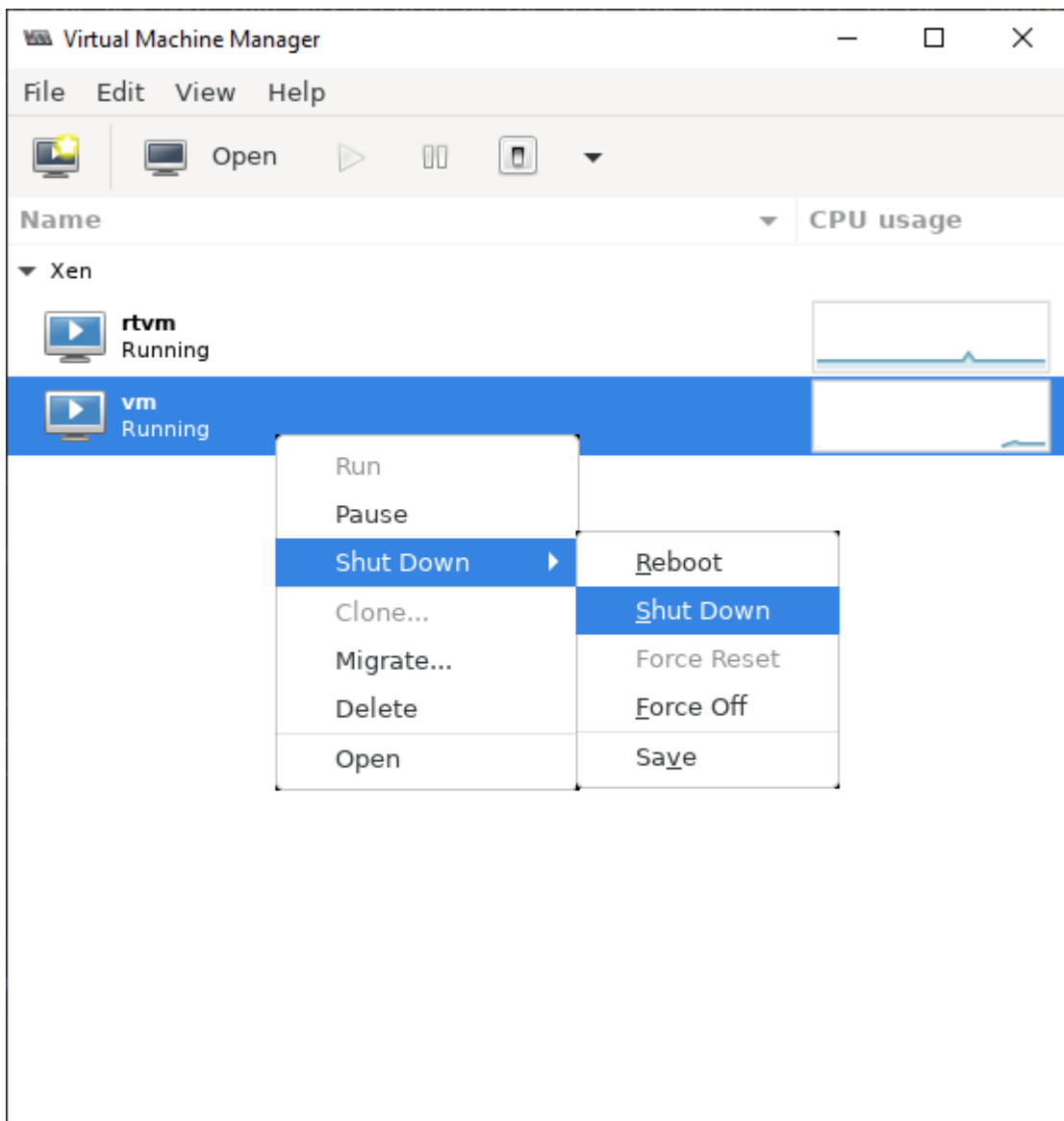
8. Select the play symbol to launch the virtual machine.

The virtual machine is being launched because further configuration steps need to be taken inside of the VM. The IP addresses of the network cards that were defined above need to be configured (static or dynamic). This step requires a user with knowledge about the VM, as it requires configuration inside of the VM. Refer to [Networks for Virtual Machine workloads](#) for more information on the interfaces and their IP ranges.

## Copying the IMG file to a local workstation

With the generation of the VM, the IMG file of the VM has also been generated on the Nerve Device. Copy the IMG file to the local workstation.

1. Right-click the virtual machine in the main Virtual Machine Manager window.
2. Select **Shut Down > Shut Down** to shut down the VM.



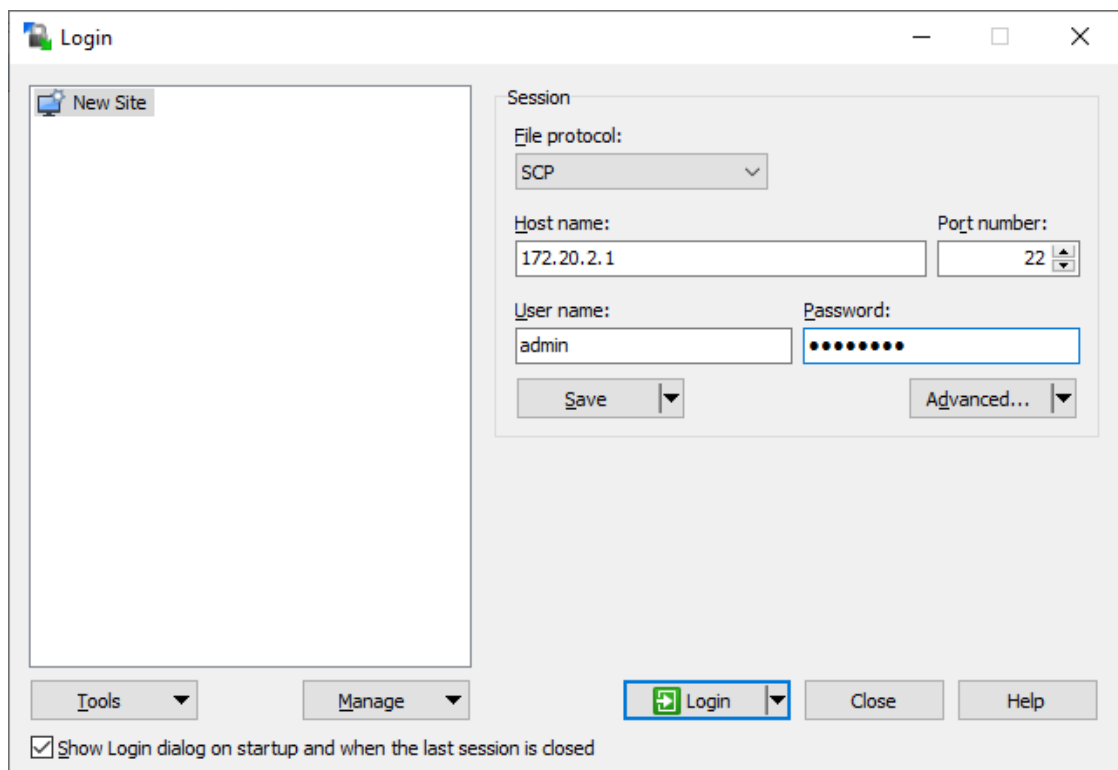
- Switch to the SSH client window.
- - 
  3. Switch to the SSH client window.
  4. Enter the following command:

```
sudo chmod o+r /opt/data/<directory>/<vmname>.img
```

#### NOTE

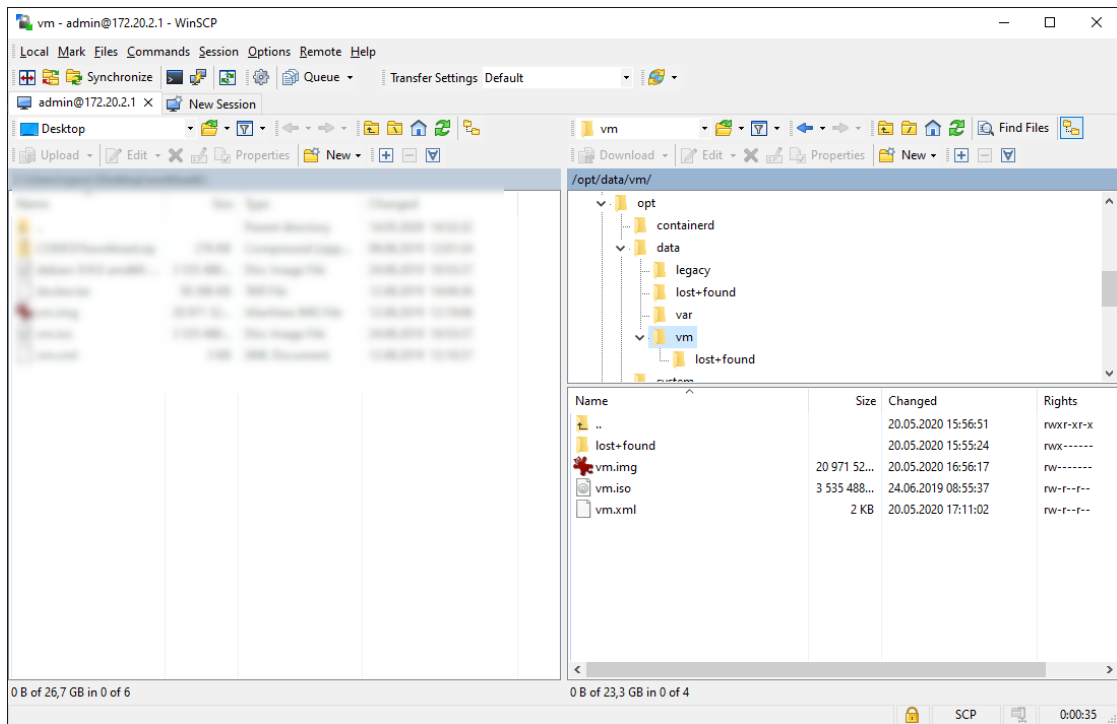
The IMG file is called <vmname>.img in the command example above. Replace the placeholder name of the image in the command with the actual name of the IMG file.

- 
- 
- 
- 
5. Open a file transfer client like WinSCP.
6. Enter the IP address for host access to the Nerve Device under **Host Name**.
7. Enter the credentials for host access to the Nerve Device below under **User name** and **Password**.



- 
- 
- 
- 
- 
- 
- 
8. Navigate to /opt/data/<directory> on the right side of the window. The **opt** directory is located in the **root** directory.
9. Copy the <vmname>.img file to the local workstation.





The virtual machine has now been generated on the Nerve Device and the IMG file of the virtual machine is now on the local workstation.

#### NOTE

Do not deploy the virtual machine from the process above to the same Nerve Device through the Management System. The virtual machine will be present twice. The deployment of the virtual machine from the process above should be done to different nodes.

## Obtaining the XML file

When the IMG file was generated on the Nerve Device, an XML file for the IMG file was generated as well. It also has to be obtained manually. However, before it can be copied to the workstation. It has to be transferred out of the **etc** directory.

1. Switch to the SSH client window.
2. Navigate to /opt/data/<directory> by entering the following command:

```
cd ../../opt/data/<directory>
```

3. Enter the following command to dump the XML file into the directory:

```
virsh dumpxml <volume_name> > <filename>.xml
```

```
/dev/mapper/nerve-vm 30G 3.5G 25G 13% /opt/data/vm
admin@nerve-host:~$ cd ../../opt/data/vm
admin@nerve-host:/opt/data/vm$ virsh dumpxml vm > vm.xml
admin@nerve-host:/opt/data/vm$ ls
lost+found vm.img vm.iso vm.xml
```

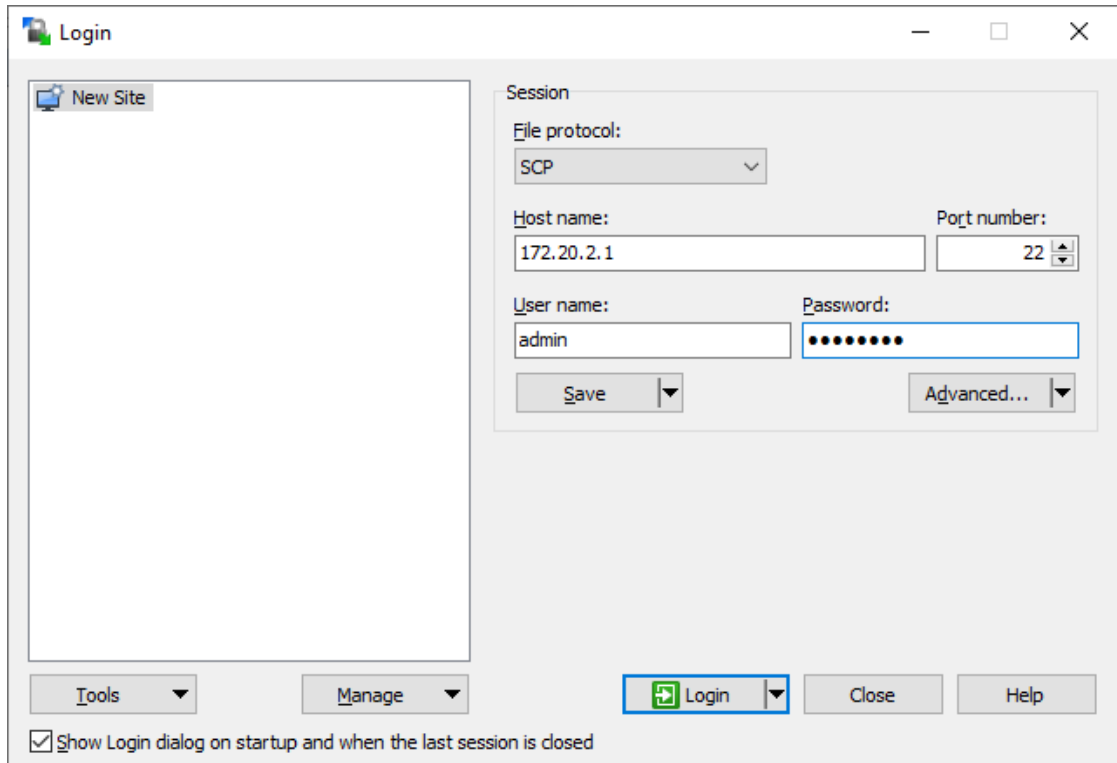
4. Enter the following command:

```
sudo chmod o+r /opt/data/<directory>/<filename>.xml
```

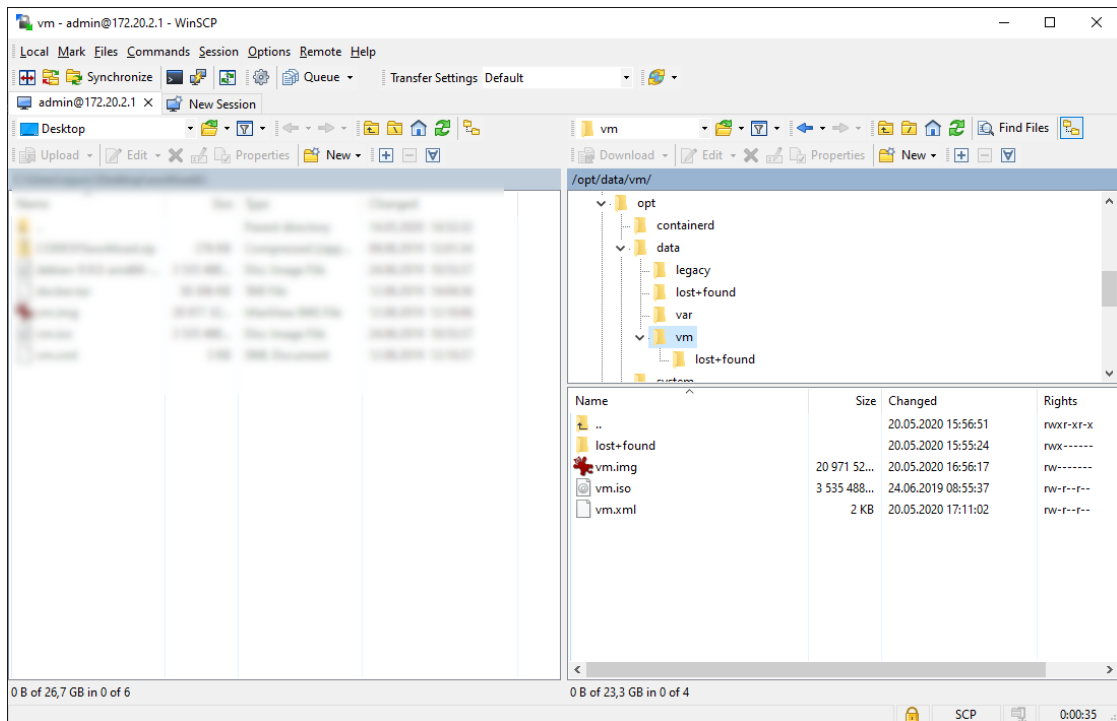
#### NOTE

Replace the placeholder name of the XML file in the command with the actual name of the XML file. The XML file is called `vm.xml` in the following screenshots below.

5. Open a file transfer client like WinSCP.
6. Enter the IP address for host access to the Nerve Device under **Host Name**.
7. Enter the credentials for host access to the Nerve Device below under **User name** and **Password**.



8. Navigate to `/opt/data/<directory>` on the right side of the window. The **opt** directory is located in the **root** directory.
9. Copy the `<filename>.xml` file to the local workstation.



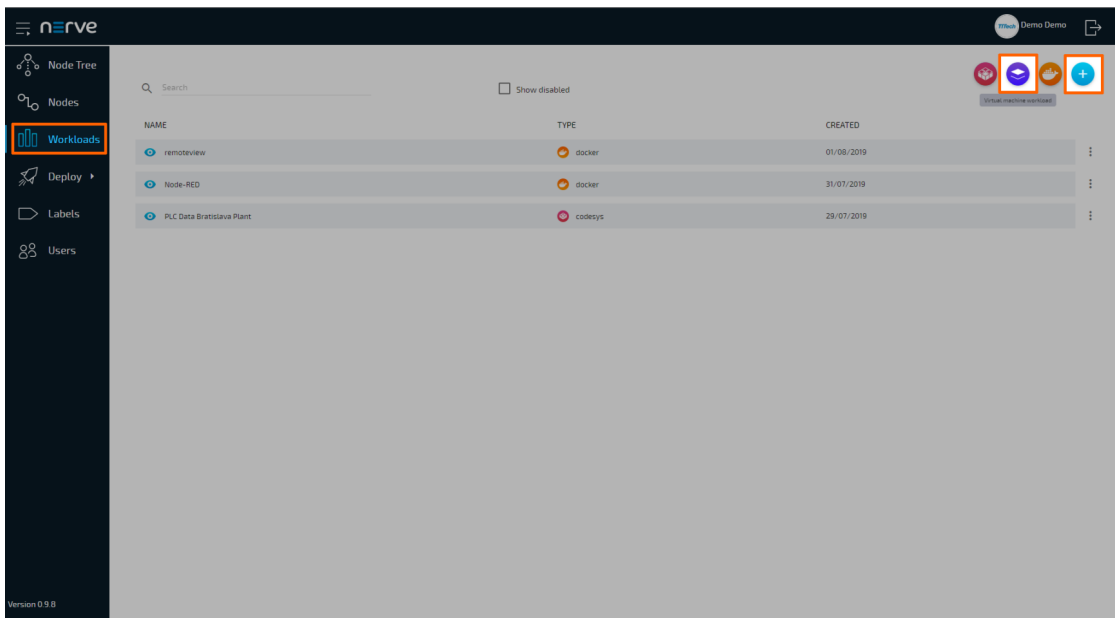
With this all the necessary files to provision a Virtual Machine workload for this virtual machine are ready.

## Provisioning a Virtual Machine workload

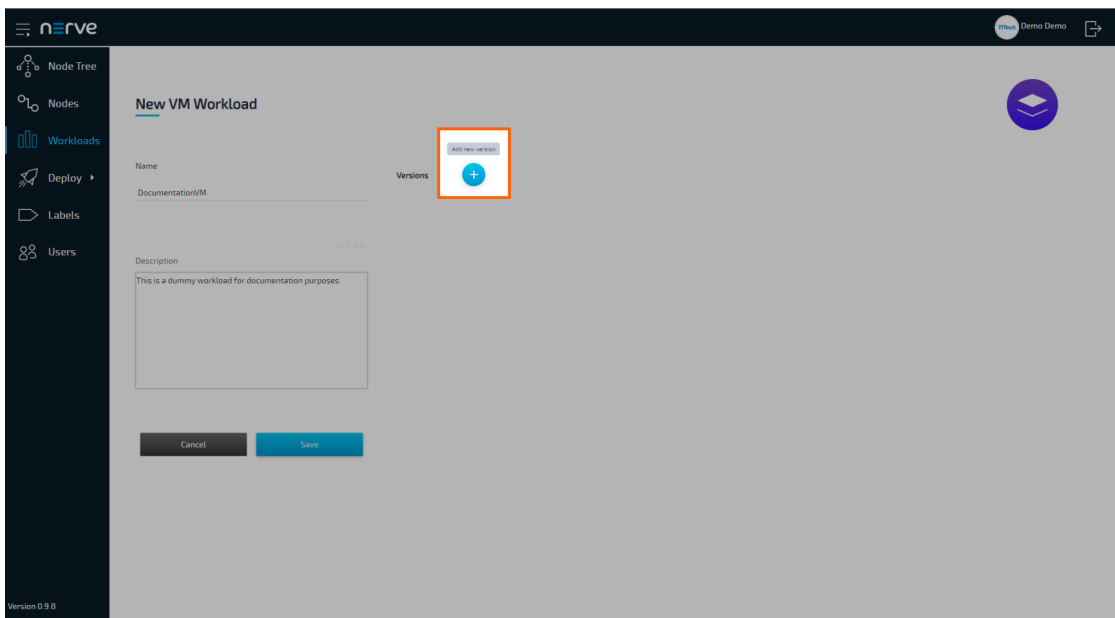
The following instructions cover the basic requirements for provisioning a Virtual Machine workload. Optional settings will be left out. Extended options are addressed in the last section of this chapter.

There are two further types of workloads that can be provisioned: [CODESYS workloads](#) and [Docker workloads](#). The process for each workload is highlighted in its respective chapter.

1. Log in to the Management System.
2. Select **Workloads** in the left-hand menu.
3. Select the plus symbol in the upper-right corner.
4. Select the virtual machine symbol (**Virtual Machine workload**) in the middle of the three symbols that expanded.



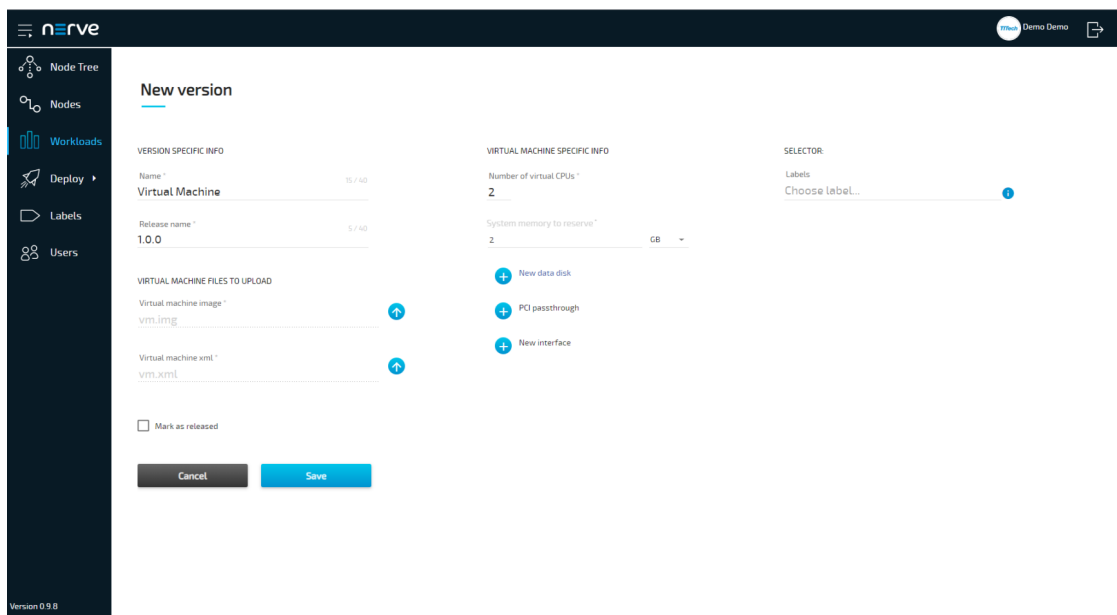
5. In the new window, enter a name for the workload.
6. Select the plus symbol next to **Versions** to add a new version of the workload.



7. In the next window, enter the following information:

Item	Description
<b>Name</b>	In the new window, enter a <b>Name</b> for the version of this workload.
<b>Release name</b>	Enter a <b>Release name</b> for the version of this workload.
<b>Virtual machine image</b>	Click the <b>upward arrow</b> symbol here to add the virtual machine image. The image has to be an IMG or RAW file.

Item	Description
<b>Virtual machine xml</b>	Click the <b>upward arrow</b> symbol here to add the virtual machine XML file.
<b>Number of virtual CPUs</b>	Enter the number of virtual CPUs to use for this virtual machine. This setting can be changed again after the deployment of the workload.
<b>Limit memory to</b>	Assign how much system memory the workload is allowed to use. This setting can be changed again after the deployment of the workload.
<b>New interface</b>	Select the plus symbol to add a new interface. Remember to add the same number of interfaces in the same order as they were added as network cards in <a href="#">Defining network interfaces</a> above.



8. Select **Save** in the lower-left corner.

The workload has now been provisioned and is ready to be deployed in the **Deploy** menu.

#### NOTE

While some settings are not required to provision a Virtual Machine workload in the Management System, additional settings will have to be filled in for the workload to perform as desired. Depending on the virtual machine that will be deployed, new interfaces might have to be defined. Keep this in mind and make sure to learn the details about the virtual machine.

## Settings for Virtual Machine workloads

In the instructions above, all optional settings have been left out. Below is an overview of all the options with an explanation to each option.

Setting	Description
<b>VERSION SPECIFIC INFO</b>	<p><b>Name</b> A name for the workload version. This could be a reminder for a certain configuration. Example: "Unlimited" as a name for a virtual machine that has unlimited access to CPU resources.</p> <p><b>Release name</b> A release name for the workload version. This could be a version number. Example: 1.0.1</p>
<b>VIRTUAL MACHINE FILES TO UPLOAD</b>	<p>Two files need to be added here:</p> <p><b>Virtual machine image</b> Upload the virtual machine image with the file extension RAW or IMG here. Do this by clicking the <b>upward arrow</b> symbol and selecting the file in the file browser. This is the first file generated in the process before.</p> <p><b>Virtual machine xml</b> Upload the virtual machine XML file here. Do this by clicking the <b>upward arrow</b> symbol and selecting the file in the file browser. This is the second file generated in the process before.</p> <p>Note that the settings defined under <b>Virtual machine specific info</b> are going to overwrite parts of this XML file.</p>

Setting	Description
<b>Virtual machine specific info</b>	<p><b>Number of virtual CPUs</b>            Define the number of virtual CPUs to assign to this virtual machine. The CPUs are then reserved exclusively for the Virtual Machine workload and cannot be used by other processes. This setting is mandatory and the workload cannot be provisioned if it is left blank. Note that this setting can be changed again after the deployment of the workload without having to undeploy and redeploy the workload. Refer to <a href="#">Changing resource allocation of a deployed Virtual Machine workload</a> for more information.</p>
	<p><b>Limit memory to</b>            Assign how much system memory the workload is allowed to use. The memory assigned here will be reserved exclusively for this Virtual Machine workload and will not be available for any other processes. This setting is mandatory and the workload cannot be provisioned if it is left blank. Note that this setting can be changed again after the deployment of the workload without having to undeploy and redeploy the workload. Refer to <a href="#">Changing resource allocation of a deployed Virtual Machine workload</a> for more information.</p>
	<p><b>New data disk</b>            Click the plus symbol to add a new data disk for the virtual machine. This data disk functions like an extra hard drive for data separate from the virtual machine. Enter a <b>Data disk name</b> and define the <b>Disk size</b>.</p>
	<p><b>PCI passthrough</b>            Click the plus symbol to add a PCI passthrough to the virtual machine. Enter the PCI address of the interface to pass through to be directly used by the virtual machine. Note that the PCIe address is specific to a certain hardware. When using this option, the installation targets to nodes with this specific hardware should be limited by using selectors. Refer to the <a href="#">labels chapter</a> for more information on selectors.</p>
	<p><b>New interface</b>            Click the plus symbol to add a new interface. Choose between a bridged interface and a NAT-interface. For NAT interfaces <a href="#">port mappings</a> for TCP and UDP can be defined. Also, remember to add the same number of interfaces in the same order as they were added as network cards in <a href="#">Defining network interfaces</a> above.</p> <p>The names of the interfaces here have to match the names of the pre-defined network interfaces. Also, make sure to not use reserved ports for the workload. Refer to the <a href="#">networking chapter</a> for more detailed information.</p>
<b>SELECTOR</b>	<p><b>Labels</b>            If labels have been defined and assigned to nodes, add them as selectors to the workload. When deploying a workload, the list of nodes will be filtered automatically to the specified label.</p>
<b>Mark as released</b>	<p>Tick this checkbox to mark this workload as released. Once marked as released, the workload cannot be edited anymore.</p>

# Provisioning a Docker workload

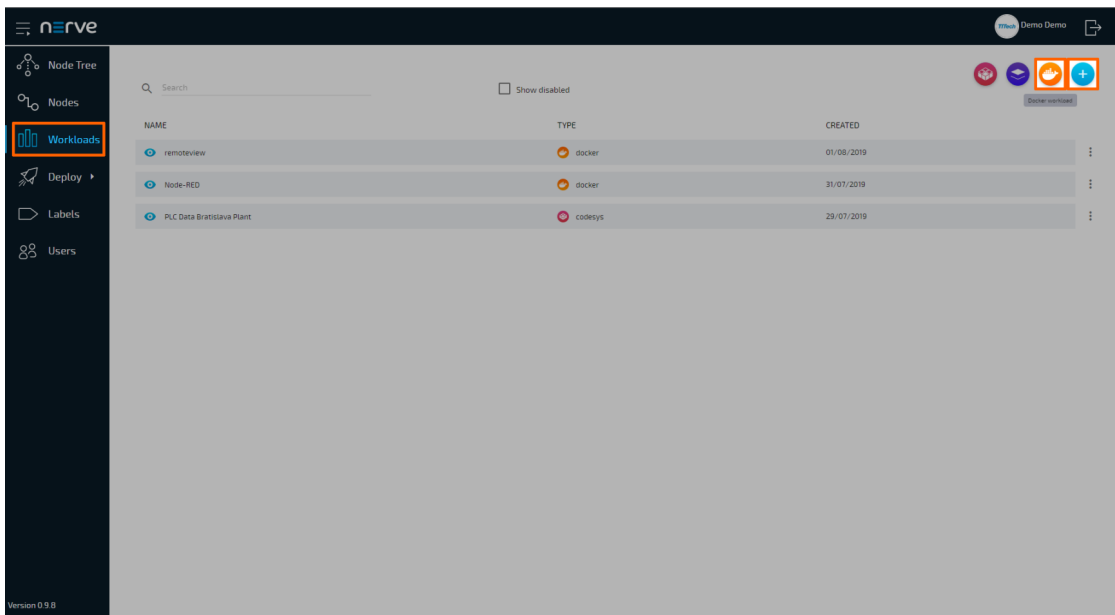
The following instructions cover the basic requirements for provisioning a Docker workload. Optional settings will be left out. Extended options are addressed in the last section of this chapter.

There are two further types of workloads that can be provisioned: [CODESYS workloads](#) and [Virtual Machine workloads](#). The process for each workload is highlighted in its respective chapter.

## NOTE

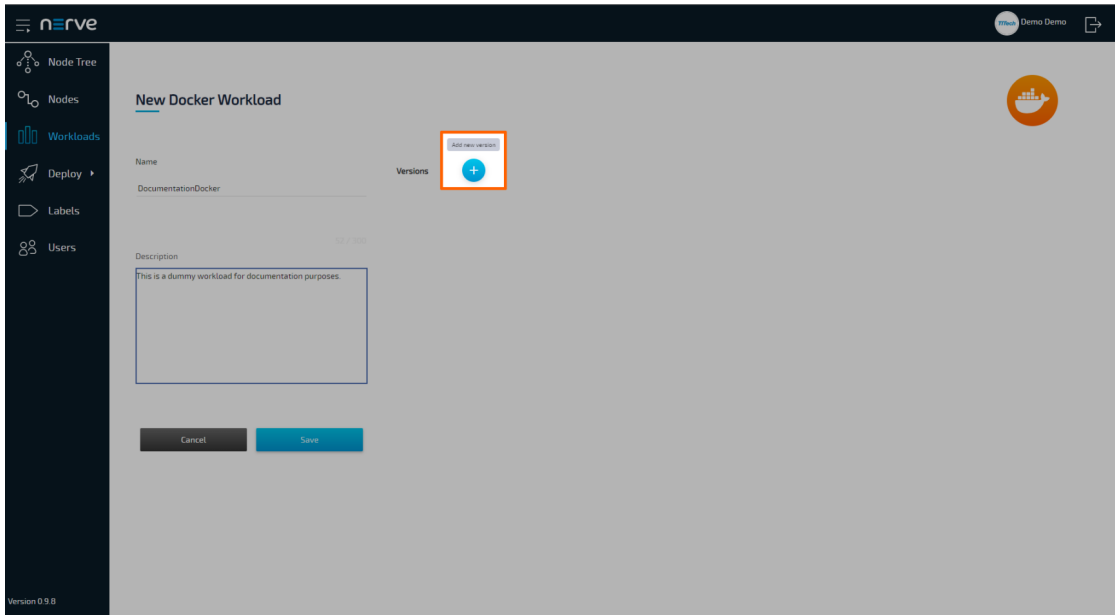
A Docker image is required for the following instructions. Refer to the [Docker documentation](#) for help on creating a Docker image. Note that Docker images in TAR.GZ format are not supported.

1. Log in to the Management System.
2. Select **Workloads** in the left-hand menu.
3. Select the plus symbol in the upper-right corner.
4. Select the Docker symbol (**Docker workload**) on the right of the three symbols that expanded.



5. In the new window, enter a name for the workload.
6. Select the plus symbol next to **Versions** to add a new version of the workload.





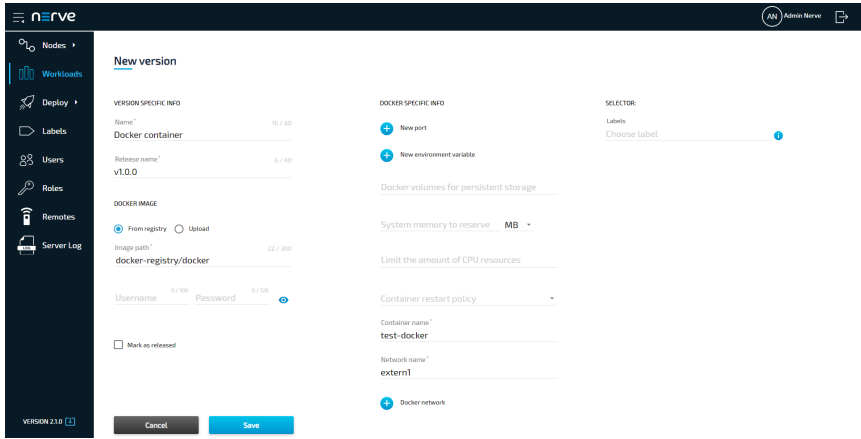
7. In the next window, enter the following information:

Item	Description
<b>Name</b>	Enter a <b>Name</b> for the version of this workload.
<b>Release name</b>	Enter a <b>Release name</b> for the version of this workload.

Item	Description
------	-------------

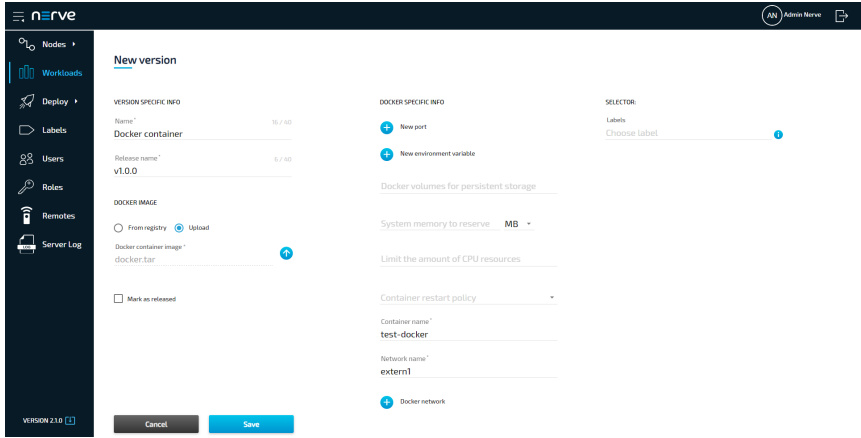
Select **From registry** or **Upload**. Note that Docker images in TAR.GZ format are not supported.

**From registry:**  
Enter the Docker registry to the Docker container image under **Image path**.



**DOCKER IMAGE**

**Upload:**  
Click the **upward arrow** symbol to open the file browser and upload the Docker container image.



Do not upload a Docker image in TAR.GZ format. This will produce a Docker workload file in the wrong format when the workload is exported.

**Container name** Enter a name for the Docker container. This will be the name of the Docker container on the node.

**Network name** Enter the network interface name through which the Docker container can be reached. Refer to [Node internal networking](#) for more information.

Note that in this version the **bridge** network is automatically assigned to a Docker workload when any Docker network (**mgmt**, **wan** or **extern1** to **extern3**) is assigned.

8. Select **Save** in the lower-left corner.

The workload has now been provisioned and is ready to be deployed in the **Deploy** menu.

#### NOTE

While certain settings are not required to provision a Docker workload in the Management System, additional settings have to be filled in for the workload to perform as desired. Depending on the Docker container that will be deployed, ports need to be defined and environment variables need to be configured. Keep this in mind and make sure to learn the details about the Docker container. Refer to the description of Docker workload settings below for more information.

## Settings for Docker workloads

In the instructions above, all optional settings have been left out. Below is an overview of all the options with an explanation to each option.

Setting	Description
<b>VERSION SPECIFIC INFO</b>	<b>Name</b> A name for the workload version. This could be a reminder for a certain configuration. Example: "Unlimited" as a name for a Node-RED version that has unlimited access to CPU resources.
	<b>Release name</b> A release name for the workload version. This could be a version number. Example: 1.0.1
<b>DOCKER IMAGE</b>	Select between two options here and either use a Docker registry URL to link to an online repository or upload the Docker container image from the workstation. Note that Docker images in TAR.GZ format are not supported.
	<b>From registry</b> Specify a URL pointing to the Docker container image under <b>Image path</b> . Note the differences between public Docker Hub registries and private registries. Private registries require the full URL to be specified, as well as a username and password if they require authentication. Public Docker Hub registries can be specified in their short form. Examples: <ul style="list-style-type: none"><li>• Public Docker Hub registry nodered/node-red-docker</li><li>• Private registry with authentication and a tag at the end auth.docker.test.host.cloud/workload:v1.3</li></ul>
	<b>Upload</b> Upload the Docker container image from the workstation. Obtain a Docker image for the upload by executing a command on the workstation where the Docker image for provisioning is located. Enter <code>docker save &lt;image_name&gt; -o &lt;filename&gt;</code> . Then in the Management System click the <b>upward arrow</b> symbol to open the file browser and upload the resulting <filename> Docker image. Also, do not upload Docker images in TAR.GZ format. This will produce a Docker workload file in the wrong format when the workload is exported.
<b>Mark as released</b>	Tick this checkbox to mark this workload as released. Once marked as released, the workload cannot be edited anymore.

Setting	Description
<b>Restart on configuration update?</b>	Tick this checkbox to trigger an automatic restart of the Docker workload when configuration files are applied.
<b>Mark first volume as a configuration storage</b>	Tick this checkbox to allow the application of configuration files. Note that at least one <b>Docker volume for persistent storage</b> must be defined for this option to be available.

### **New port**

Click the plus symbol to define **Host port**, **Container port**, and **Protocol**.

- **Protocol**

Choose **TCP** or **UDP** here.

- **Host port**

This is the port through which the Docker workload will be reachable on the host. Make sure to not use reserved ports in the Nerve system for the workload. Refer to the [networking chapter](#) for more information.

- **Container port**

This is the internal port of the Docker container. Note that every Docker container has a default port. Entering a value that is different than the default port will use the entered port instead of the standard port of the Docker container.

### **New environment variable**

Click the plus symbol to add an environment variable (**Env. variable**) and its **Variable value**. Make sure to define the appropriate variables and values as they depend on the Docker container that will be deployed.

### **Docker volumes for persistent storage**

Persistent storage for a Docker workload can be defined using named volumes. Enter a path in the following format to define a Docker volume for persistent storage: <volumename>:<containerpath>

- <volumename>

Define a name through which the persistent Docker volume can be accessed on the host. This name can be any string. Note that the Management System fills in a volume name suggestion by default using the workload name.

- <containerpath>

Define a path inside the Docker container for the storage.

The volume for persistent storage is not erased when the Docker workload is restarted or undeployed. The data also persists through a node version update. However, note that a volume will be removed if the next version of the workload does not have a volume defined and a workload update is performed. Docker volumes for persistent storage can be used for any workload by using the same volume name again. Docker workload storage can be found in /opt/data/var/lib/docker/volumes by default. Multiple volumes can also be defined separated by , i.e. <volumename1>:<containerpath1>,<volumename2>:<containerpath2>.

### **Limit memory to**

Assign how much system memory this workload is allowed to use. The memory assigned here is an upper limit that the Docker workload can use and is not exclusively reserved for the Docker workload. Other processes can use these resources as well.

### **CPU resource in percentage**

Specify here the percentage of CPU resources the workload is allowed to use. If this field is left blank, the workload is allowed to use all available resources.

### **Container restart policy**

Choose the container restart policy here to determine when the Docker container can be restarted.

- **no**

The container does not restart automatically.

- **on-failure**

The container restarts when it exits due to an error.

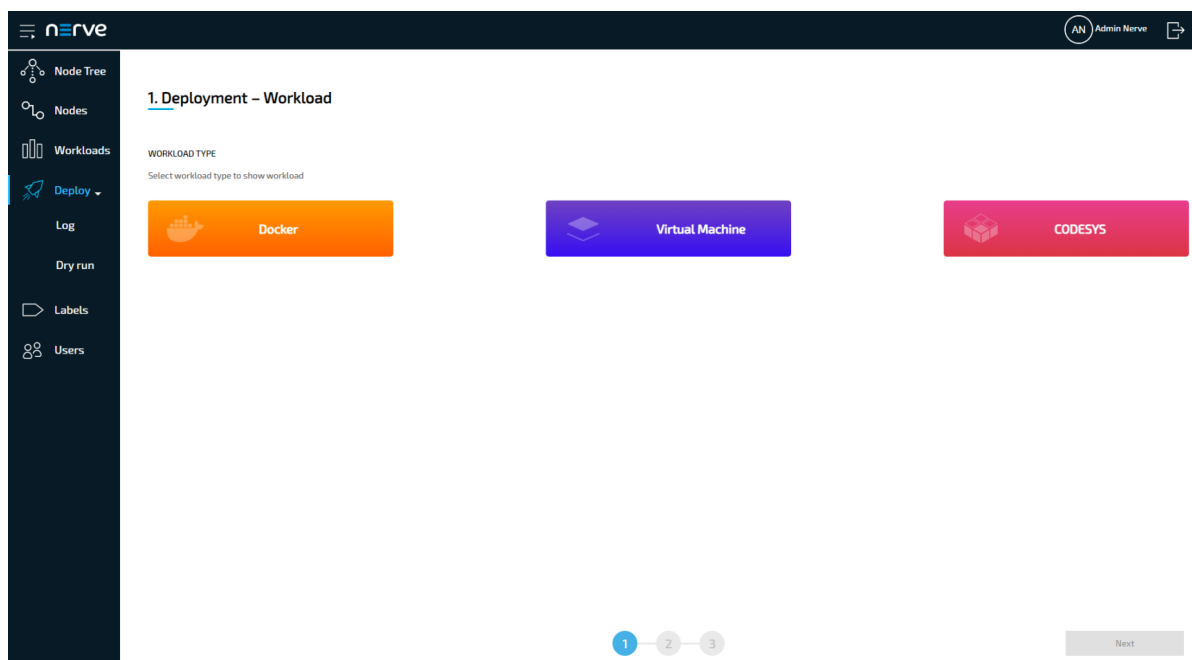
- **always**

The container restarts every time it stops. However, manually stopping the container is the exception. If a container is manually stopped, it is only restarted when the Docker daemon restarts or the container is restarted manually.

Setting	Description
<b>SELECTOR</b>	<b>Labels</b> If labels have been defined and assigned to nodes, add them as selectors to the workload. When deploying a workload, the list of nodes will be filtered automatically to the specified label.

## Deploy menu

Workloads that have been provisioned in the Management System are ready to be deployed to nodes through the **Deploy** menu. It expands into three menus:



Item	Description
<b>Deploy</b>	This is the landing page of the deployment menu. Deploy workloads to nodes from here.
<b>Log</b>	This is the history of deployments and dry runs.
<b>Dry run</b>	Structurally the same as the deployment process for workloads, simulate the deployment of a workload from this menu.

## Log

The log is the history of deployments and dry runs. This includes:

- deployments in progress
- dry runs in progress
- failed deployments
- failed dry runs
- successful deployments
- successful dry runs

It is displayed in reverse chronological order and can be filtered according to search criteria. It also offers some control functionality for active and failed deployments.

DEPLOYMENT NAME	ACTION	PROGRESS	STARTED	FINISHED
nextdeployment	Deploy	100.00% Complete	04/09/2019 10:05	04/09/2019 10:05
201909041001	Deploy	100.00% Complete	04/09/2019 10:01	04/09/2019 10:01
mfarem3	Deploy	100.00% Complete	03/09/2019 15:58	03/09/2019 15:58
mfarem2	Deploy	100.00% Complete	03/09/2019 15:57	03/09/2019 15:57
mfarem1	Deploy	100.00% Complete	03/09/2019 15:57	03/09/2019 15:57
remotepick	Deploy	100.00% Complete	03/09/2019 15:26	03/09/2019 15:26
remoterotating	Deploy	100.00% Complete	03/09/2019 15:25	03/09/2019 15:25
remotertkit	Deploy	100.00% Complete	03/09/2019 15:25	03/09/2019 15:25
201909031358	Deploy	100.00% Complete	03/09/2019 13:58	03/09/2019 13:59
201909031357	Deploy	100.00% Complete	03/09/2019 13:57	03/09/2019 13:57

Item	Description
<b>Search by name (1)</b>	Enter text here to filter the list by deployment name.
<b>Deployment Type (2)</b>	Select an option from the drop-down menu to filter the list for <b>Deploys</b> or <b>Dry runs</b> .
<b>Workload Type (3)</b>	Select an option from the drop-down menu to filter the list for a specific workload type: <b>VM</b> , <b>Docker</b> or <b>CODESYS</b> .
<b>DEPLOYMENT NAME (4)</b>	This is the name of the deployment with the workload type displayed as a symbol. The default is the time of deployment unless a deployment name has been entered during the deployment process. Note that this is not the name of the workload.
<b>ACTION (5)</b>	Here the deployment type is displayed: <b>Deploy</b> or <b>Dry run</b> .
<b>PROGRESS (6)</b>	<p>The progress bar is an indicator for both progress and status. Depending on the status of the deployment it changes its color:</p> <ul style="list-style-type: none"> <li>• <b>Green</b> If a workload was deployed successfully, the bar will be green at a 100%.</li> <li>• <b>Blue</b> If a workload is currently being deployed, the bar will be blue, fill up gradually and display the progress of the deployment in percent.</li> <li>• <b>Red</b> If the deployment of a workload has failed, the bar will be red at a 100%.</li> </ul>
<b>STARTED (7)</b>	This is the date and time the workload deployment was started. The date format is DD/MM/YYYY.

Item	Description
<b>FINISHED (8)</b>	This is the date and time the workload deployment was completed. The date format is DD/MM/YYYY. This field will display <b>In progress</b> if a workload is in progress of being deployed.
<b>Ellipsis menu (9)</b>	Select the ellipsis menu to trigger an overlay with the <b>DELETE</b> option. Selecting <b>DELETE</b> will remove the entry from the log. When a workload is in progress, this entry is grayed out.

## Deployment details

Clicking an entry in the log will show the details of the deployment.

The screenshot shows the 'Admin Nerve' interface. On the left is a navigation sidebar with options: Nodes, Workloads, Deploy, Labels, Users, Roles, Remotes, and Server Log. The main content area is titled 'Details of deployment Docs Docker'. It features a search bar (3) and a filter section (4) with checkboxes for 'Successful', 'In progress', 'Failed', and 'Canceled'. Below this is a summary table (5) with the following data:

Workload name: Grafana	Workload version: Grafana	Operation start time: 14/05/2020 17:36:12	Operation finish time: 14/05/2020 17:36:24
Release name: graf	Type: docker	Status: Completed	Progress: 100.00%

Below the summary table is an 'Operation task list' (6) table:

SERIAL NUMBER	STATUS	PROGRESS	RETRY COUNTER/MAX	STARTED	FINISHED
00837032311	Success	<div style="width: 100%;"></div>	1/3	14/05/2020 17:36:12	14/05/2020 17:36:24

At the bottom left, it says 'VERSION 2.1.0'. At the bottom right, there is a 'Rows per page: 10' dropdown and a page indicator '1'.

Item	Description
<b>Back button (1)</b>	Click here to return to the log.
<b>Header (2)</b>	The header states the name of the deployment in the format <b>Details of deployment &lt;deploymentname&gt;</b> .
<b>Search (3)</b>	Enter text here to filter the <b>Operation task list</b> by device name. The search function can be combined with the status checkboxes to the right of the search bar.



Item	Description
<b>Status checkboxes (4)</b>	<p>The checkboxes to the right of the search bar filter the <b>Operation task list</b> by status:</p> <ul style="list-style-type: none"> <li>• <b>Success</b> Deployments that were completed successfully are shown in the list if this checkbox is ticked.</li> <li>• <b>In progress</b> Deployments that are currently in progress are shown in the list if this checkbox is ticked.</li> <li>• <b>Failed</b> Deployments that could not be completed are shown if this checkbox is ticked.</li> <li>• <b>Canceled</b> Deployments that have been aborted by the user are shown in the list if this checkbox is ticked. However, it is not possible to cancel deployments in this version.</li> </ul> <p>If a checkbox is ticked, tasks with the corresponding status will be displayed in the operation task list. All checkboxes are ticked by default.</p>

This is information about the deployment.

**Deployment information (5)**

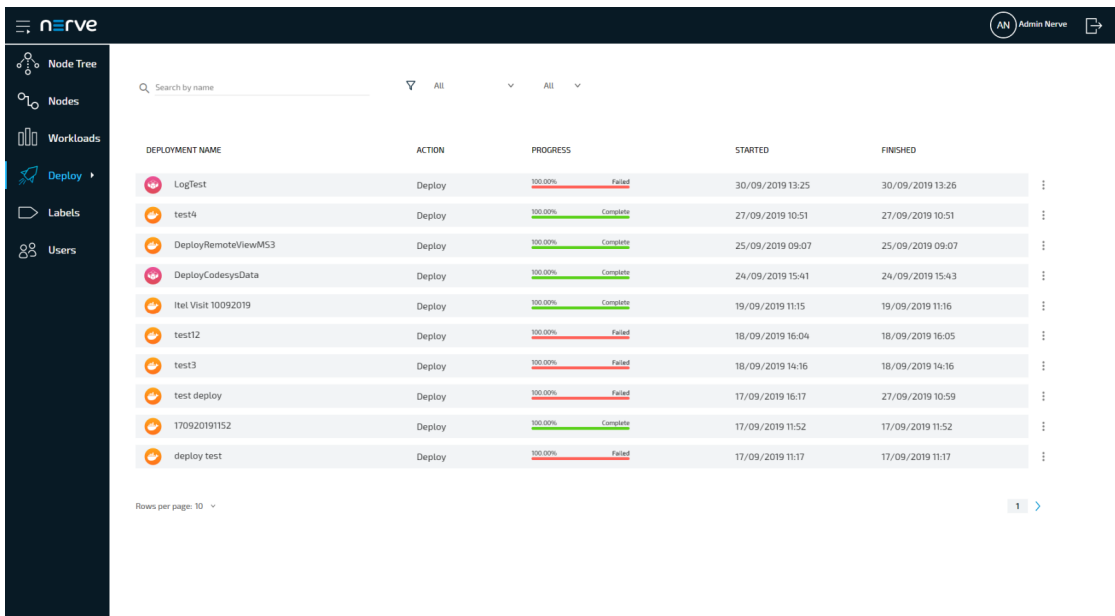
- **Workload name**  
The name of the workload that has been defined in the provisioning process.
- **Workload version**  
The name of the workload version.
- **Operation start time**  
This is the date and time the workload deployment was started. The date format is DD/MM/YYYY.
- **Operation finish time**  
This is the date and time the workload deployment finished. The date format is DD/MM/YYYY.
- **Release name**  
This is the release name of the workload version.
- **Type**  
This is the type of the workload that has been deployed: **codesys, vm** or **docker**.
- **Status**  
This is the status of the deployment. Possible statuses are **Created, In progress, Completed, Error, Canceled**.
- **Progress**  
The progress of the deployment in percent.

Item	Description
<b>Operation task list (6)</b>	<p>The operation task list displays details for single deployments that are part of the deployment campaign. The list displays information in six columns:</p> <ul style="list-style-type: none"> <li>• <b>SERIAL NUMBER</b> This is the serial number of the node that is the target of the deployment.</li> <li>• <b>STATUS</b> This is the status of the deployment. The information in this column here corresponds with the checkboxes to the right of the search bar: <b>Success, In progress, Failed</b> and <b>Canceled</b></li> <li>• <b>PROGRESS</b> This is the progress bar. It displays a different color and gradually fills up according to the progress and status of the deployment. <ul style="list-style-type: none"> <li>◦ <b>Success</b> If a workload was deployed successfully, the bar will be green at a 100%.</li> <li>◦ <b>In progress</b> If a workload is currently being deployed, the bar will be blue and display the progress of the deployment in percent.</li> <li>◦ <b>Failed</b> If the deployment of a workload has failed, the bar will be red at a 100%.</li> </ul> </li> <li>• <b>RETRY COUNTER/MAX</b> In case of failure, the Management System will attempt the deployment of a workload up to three times automatically. The left number of the counter shows the number of the current attempt. The right number is the maximum number of attempts.</li> <li>• <b>STARTED</b> This is the date and time the workload deployment was started. The date format is DD/MM/YYYY.</li> <li>• <b>FINISHED</b> This is the date and time the workload deployment was finished. The date format is DD/MM/YYYY.</li> </ul>

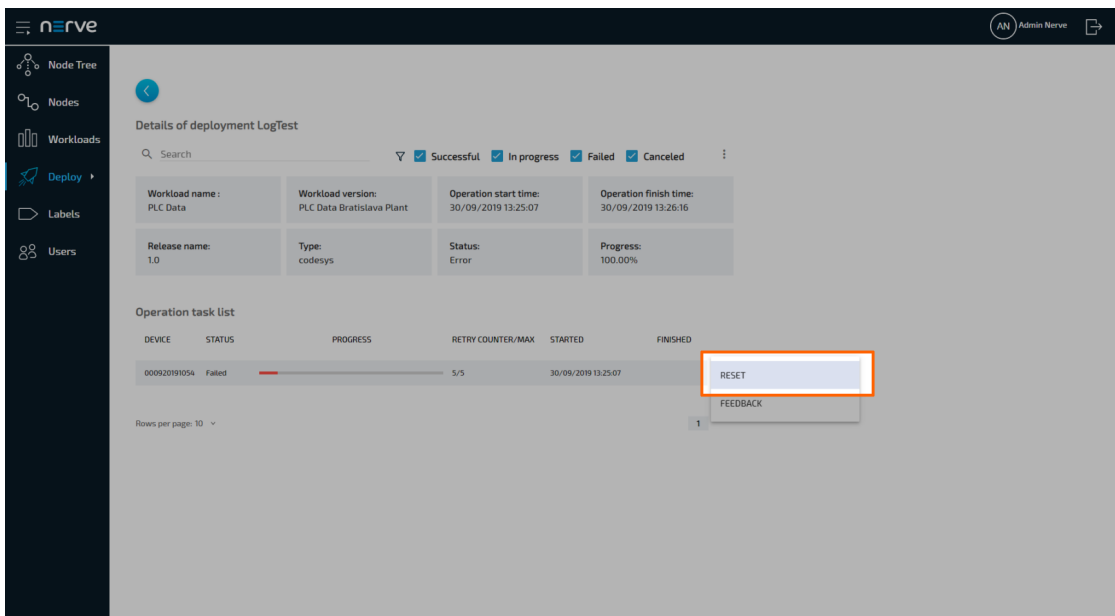
## Restarting a failed deployment

In case a deployment fails, the Management System will attempt the deployment of a workload up to three times automatically. After that, the deployment can be restarted manually through the ellipsis menu in the operation task list.

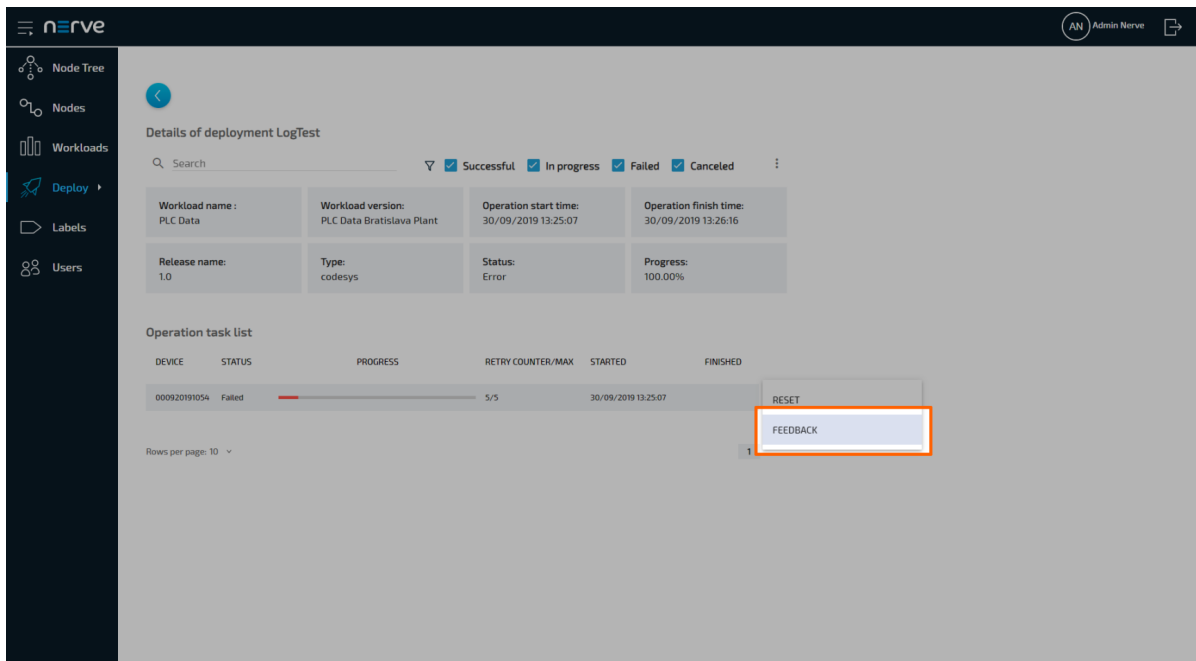
1. Select **Deploy > Log** from the menu on the left.
2. Select the failed deployment from the log.



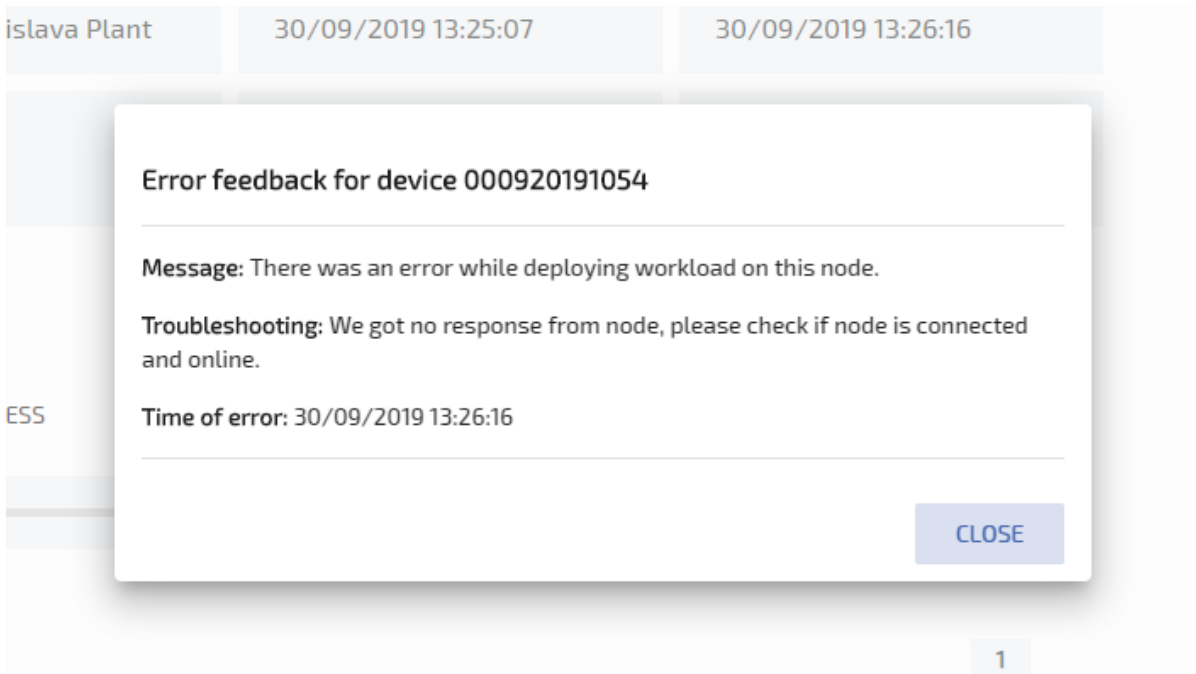
3. Choose the failed deployment from the operation task list.
4. Select the ellipsis menu to the right of the deployment entry.
5. Select **RESET** in the overlay that appeared.



The deployment is restarted immediately. To see the error information of the deployment, select **FEEDBACK** in the overlay of the ellipsis menu instead.

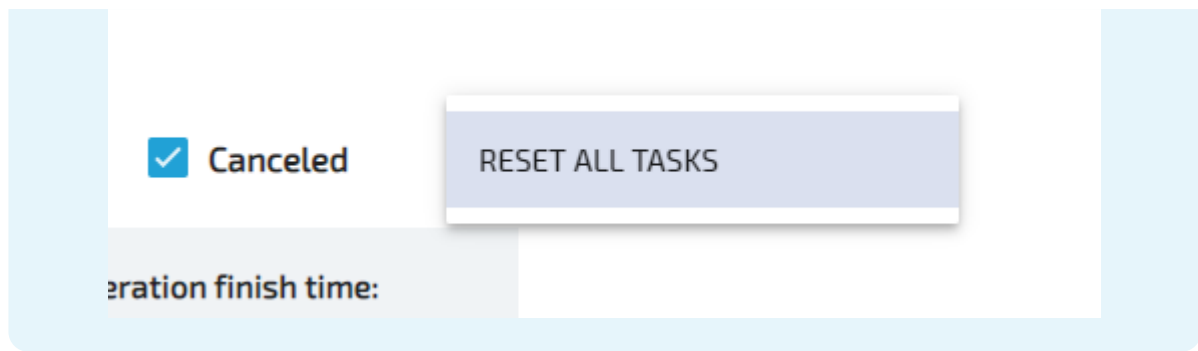


This opens a window giving information on the type of error in a message, a short troubleshooting hint and the time the error occurred. Note that troubleshooting hints are not available for every error case.



**NOTE**

To restart all deployments at once, click the ellipsis menu next to the status checkboxes and select **RESET ALL TASKS** from the overlay that appears.



## Deploying workloads and dry runs

Deployment of workloads and dry runs are covered in a separate chapter: [Deploying a Workload](#).

## Deploying a workload

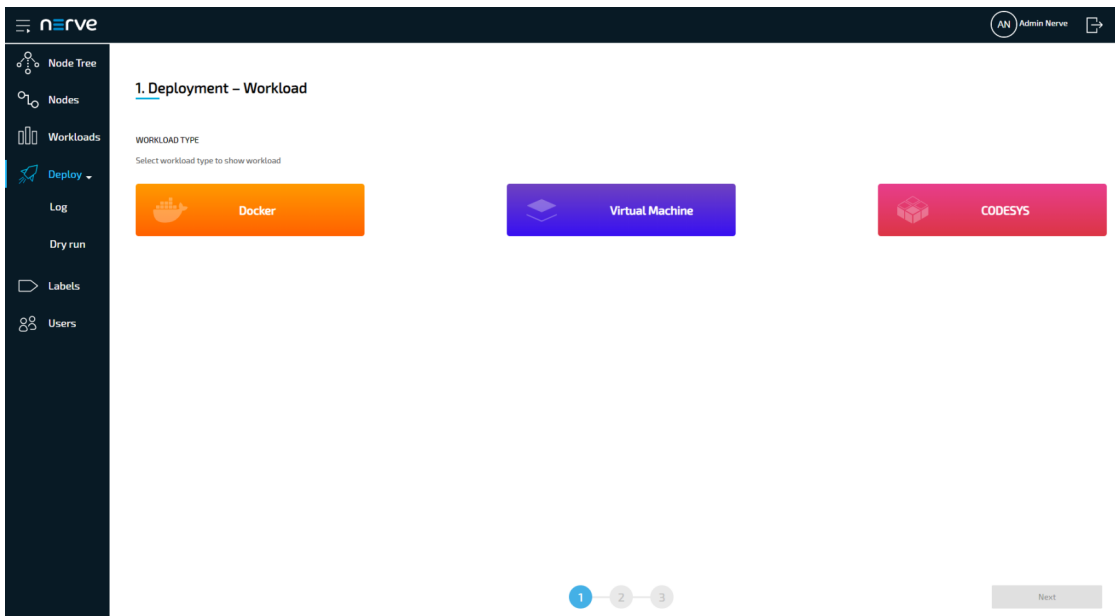
This chapter covers the deployment process of the available workloads: CODESYS workloads, Virtual Machine workloads and Docker workloads. The process of deploying workloads is identical for all three types of workloads. Therefore, the instructions below contain no specific information.

### NOTE

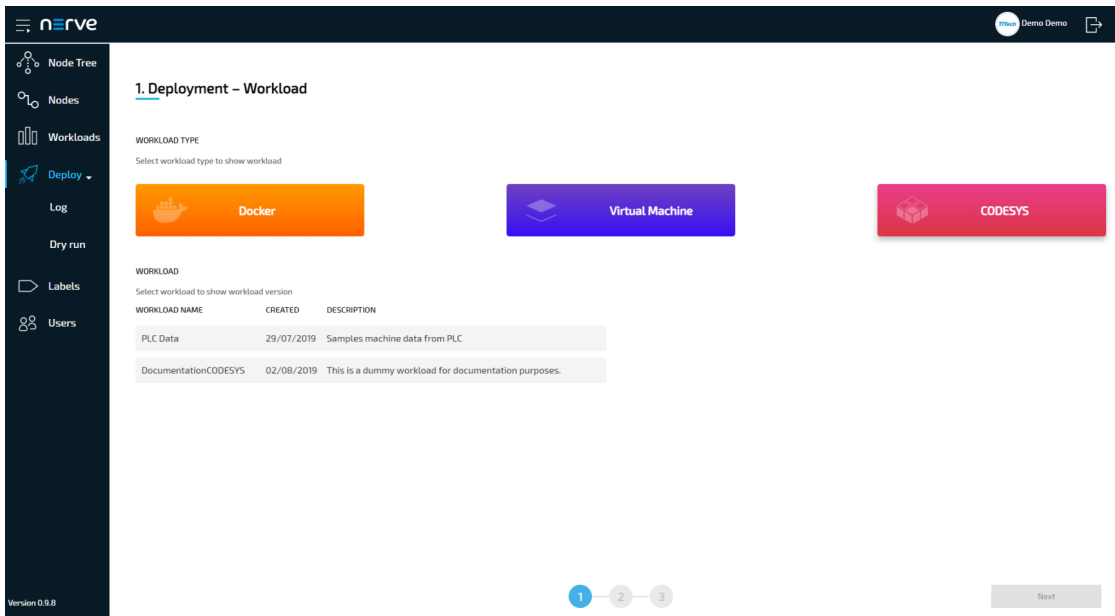
To test a deployment first, select **Deploy > Dry run** in the left-hand menu and follow the steps below starting from step 3.

However, note that a successful dry run does not guarantee a successful deployment as it is only a simulation.

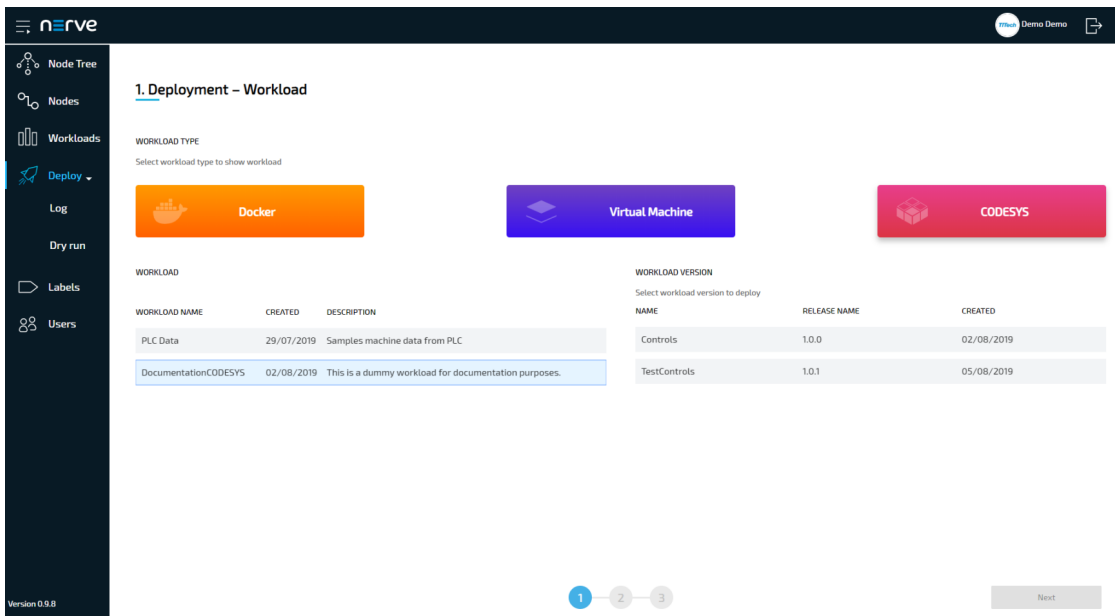
1. Select **Deploy** in the left-hand menu.



2. Select one of the three icons for workload types. A list of corresponding workloads will appear below.

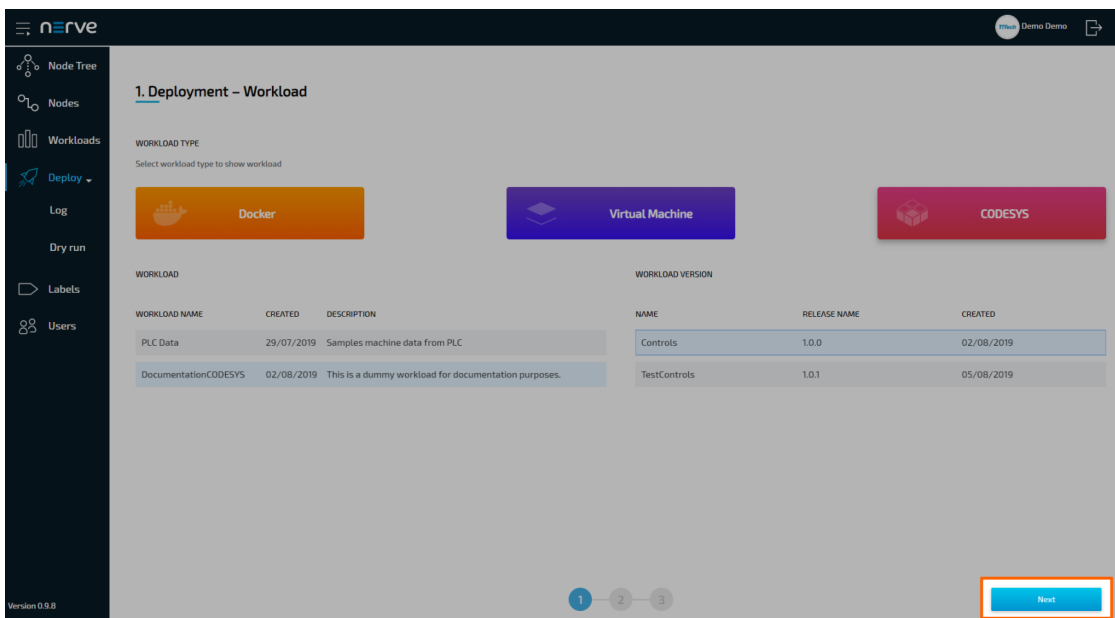


3. Select a workload from the list. A list of versions of this workload will appear to the right.



4. Select the version of the workload to deploy.

5. Click **Next** in the bottom-right corner.

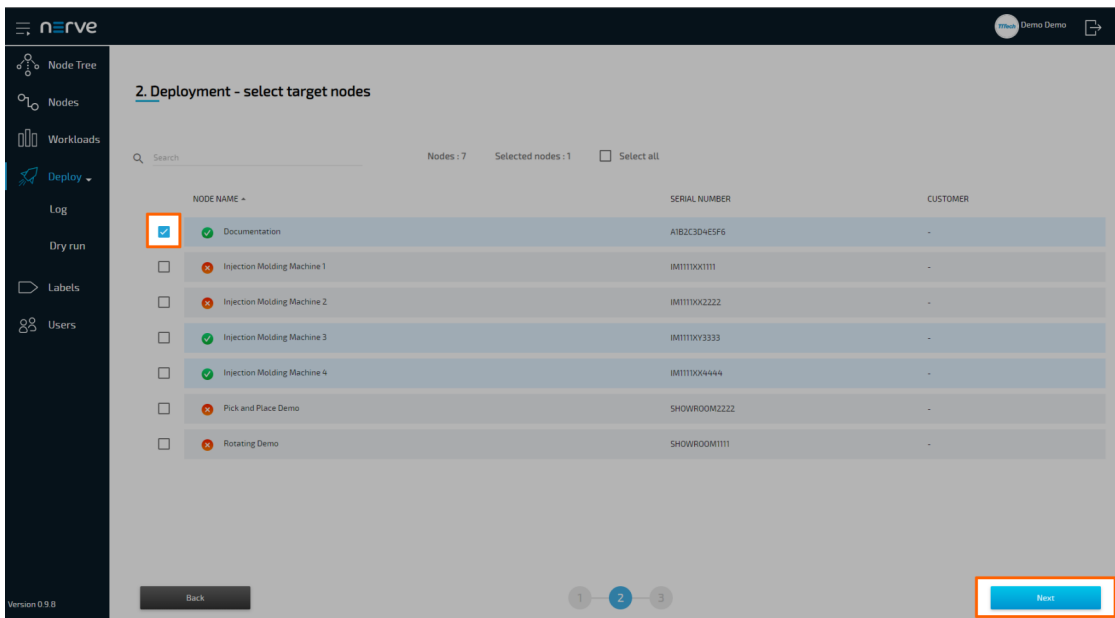


6. In the next window, select one or more nodes from the list for deployment by ticking the checkboxes on the left.

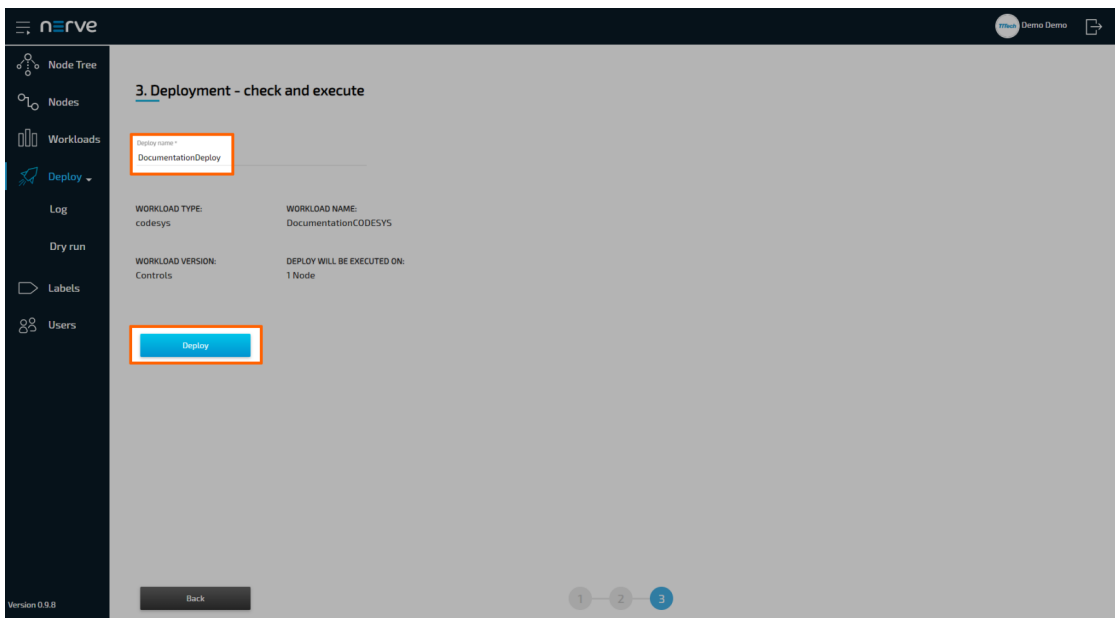
#### NOTE

This list of nodes might not include all nodes that are registered in the Management System. It is automatically filtered depending on the labels the workload has assigned.

7. Select **Next** in the lower-right corner.



8. Select **Deploy** to execute the deployment.  
*Optional: Enter a **Deploy name** above the **Summary** of the workload to make this deployment easy to identify. A timestamp is filled in automatically.*



The Management System will continue to the log next. The current deployment is at the top of the list. The **Deploy name** chosen before is the name that identifies the deployment in the log.



The screenshot shows the Nerve interface with a deployment log table. The table has columns for DEPLOYMENT NAME, ACTION, PROGRESS, STARTED, and FINISHED. The progress column includes a percentage and a status (In progress, Complete, Failed). The table shows several deployment entries with their respective dates and times.

DEPLOYMENT NAME	ACTION	PROGRESS	STARTED	FINISHED
5/28/2021,9:41:30AM	Deploy	0.00% In progress	28/05/2021 09:41	In progress
5/28/2021,9:35:19AM	Deploy	100.00% Complete	28/05/2021 09:35	28/05/2021 09:35
27/05/2021,12:14:41	Deploy	100.00% Failed	27/05/2021 13:14	27/05/2021 13:15
5/27/2021,1:01:04PM	Deploy	100.00% Failed	27/05/2021 13:01	27/05/2021 13:07
27/05/2021,11:56:40	Deploy	100.00% Complete	27/05/2021 12:56	27/05/2021 12:57
5/26/2021,3:31:06PM	Deploy	100.00% Complete	26/05/2021 15:31	26/05/2021 15:31
5/26/2021,11:37:54AM	Deploy	100.00% Complete	26/05/2021 11:38	26/05/2021 11:39
5/26/2021,11:29:08AM	Deploy	100.00% Complete	26/05/2021 11:29	26/05/2021 11:29
5/26/2021,11:09:57AM	Deploy	100.00% Complete	26/05/2021 11:10	26/05/2021 11:12
5/26/2021,10:54:11AM	Deploy	0.00% In progress	26/05/2021 10:54	26/05/2021 11:00

The progress of the current deployment is displayed here. Select the log entry of the deployment to see a more detailed view.

The screenshot shows the Nerve interface displaying details for a deployment named 'DocumentationDeploy'. It includes a search bar, filter buttons for Successful, In progress, Failed, and Canceled, and a table of operation task list.

Details of deployment DocumentationDeploy

Workload name:	Workload version:	Time of operations start:	Time of operations finish:
DocumentationCODESYS	Controls	12/08/2019 14:53:47	12/08/2019 14:53:50
Release name:	Type:	Status:	Progress:
1.0.0	codesys	Completed	100.00%

Operation task list

DEVICE	STATUS	PROGRESS	RETRY COUNTER/MAX	TIME OF START	TIME OF FINISH
A1B2C3D4E5F6	Success	<div style="width: 100%;"></div>	1/3	12/08/2019 14:53:47	12/08/2019 14:53:50

The workload has been deployed and can be controlled in the node tree. Select **Nodes** in the navigation on the left and select the node tree tab



on the right. Select the node with the deployed workload to find the tile of the deployed workload.

Reach the workload control area by clicking the tile of each workload. All workloads are started as soon as they are deployed.

## NOTE

CODESYS applications can only be controlled through the Local UI.

## Removing logical volumes after unsuccessful VM deployment

If a virtual machine workload fails in the download phase of the deployment process, it is possible that the logical volumes created for the workload stay behind. This does not negatively impact the system. However, the disk space reserved for the virtual machine workload will stay reserved unless it is removed. Follow the instructions below to find out the workload ID of the failed deployment and how to remove the logical volume.

### Finding out the workload ID

Every workload has a unique ID that is used for naming the logical volume when it is created during the deployment process. This workload ID can be found in a JSON file when a workload is exported from the Management System.

1. Access the Management System.
2. Export the virtual machine workload the deployment of which has failed by following [Exporting a workload](#). The workload is downloaded as a TAR file.
3. Navigate to the folder where the TAR file is located.
4. Extract the TAR file.
5. Open the JSON file with a text editor. The TAR file only contains one JSON file.
6. Look for "workloadId" in the file. It is the second entry in the file.

```
1605188607608_workload.json - Notepad
File Edit Format View Help
{"name": "Debian10", "workloadId": "5f4e717a165a6600f7bdc377", "type": "vm", "url": "https://release-212-2.dev.nerve.cloud/nerve_workload/storage/",
"releaseName": "Debian10", "object": "691", "memory": "1024MB", "no_of_vCPUs": "2", "templateSpecific": [], "id": "5f4e717a165a6600f7bdc378", "releaseNam
", "\${hashKey}": "\object:691\""}
Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

7. Note down the ID. It is required to identify the logical volume that needs to be removed.

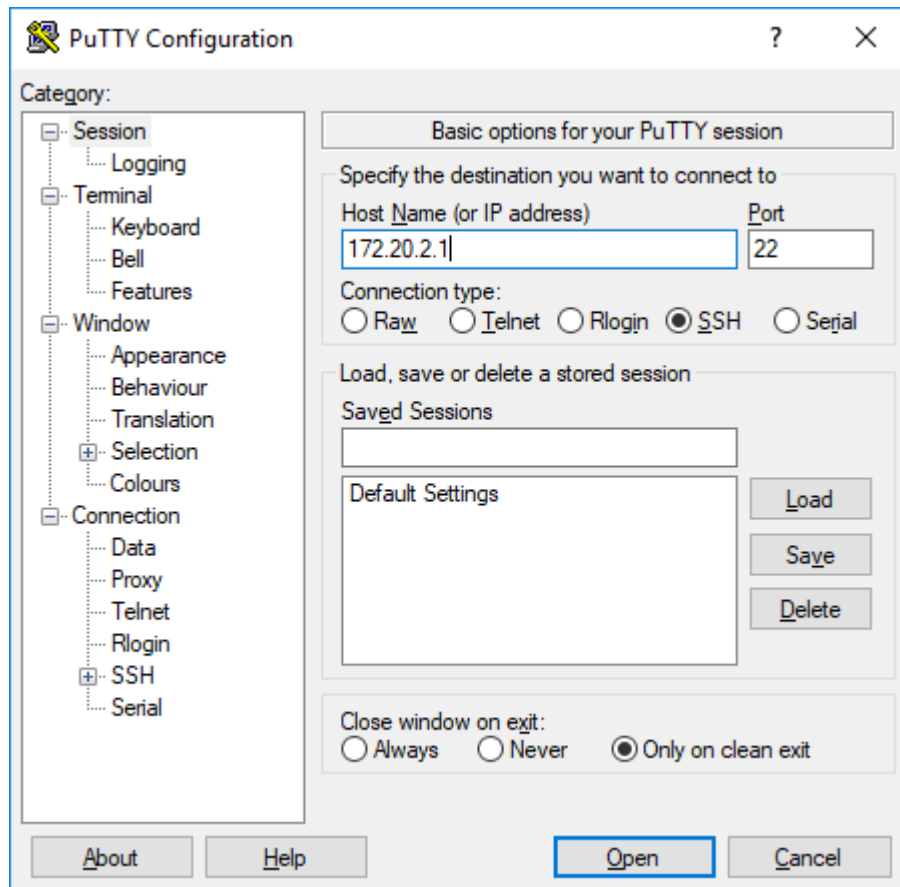
## Removing the logical volumes

With the workload ID of the unsuccessfully deployed workload, the logical volumes can be correctly identified and removed. For this, the workstation needs to be connected to the physical port of the Nerve Device associated with host access, and the network adapter IP address of the workstation needs to be configured in the correct range. This information is device specific. Refer to the [device guide](#) for information on the Nerve Device.

### NOTE

The following instructions require access to the Linux host system of Nerve. Using host access requires expert Linux knowledge as system internal changes can be performed. Note that changes may impact the Nerve system.

1. Open an SSH client like PuTTY.
2. Enter the IP address for host access to the Nerve Device under **Host Name (or IP address)** to log in to the host of the Nerve Device.



3. Log in with the credentials for host access to the Nerve Device.
4. Enter `lsblk` to display a list of volumes.
5. Find the logical volume containing the workloadId from the JSON file.

```
admin@nerve-host:~$ lsblk
NAME                                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                                  8:0    0  477G  0 disk
├─sda1                               8:1    0    2M  0 part
├─sda2                               8:2    0  128M  0 part /boot/efi
├─sda3                               8:3    0    3G   0 part /ro
├─sda4                               8:4    0    3G   0 part
└─sda5                               8:5    0 468.8G 0 part
   └─nerve-log                       254:0    0  256M  0 lvm  /var/log
      └─nerve-rtvm                    254:1    0  512M  0 lvm
         └─nerve-data                  254:2    0  23.5G 0 lvm  /opt/data
            └─nerve-system              254:3    0  512M  0 lvm  /opt/system
               └─nerve-overlay          254:4    0  128M  0 lvm  /rw
                  └─nerve-5f4e/1/a165a6600f/bdc3/7--1605791394292--vmachine 254:5    0    2G   0 lvm
```

6. Enter the following command to remove the logical volumes:

```
sudo lvremove /dev/nerve/<volume-name> -y
```

#### NOTE

Note that the steps above can also be performed through a remote connection. Refer to [Configuring a remote tunnel to a node](#) for more information.

# Remote connections

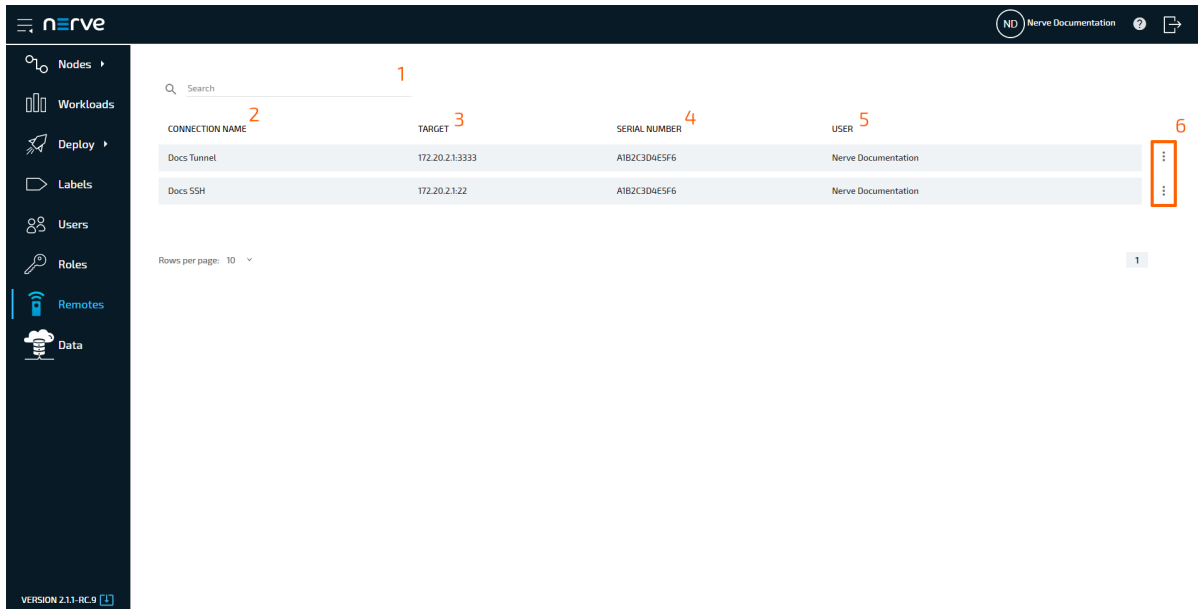
Remote connections are a fully integrated feature of Nerve. They are available in two flavors: remote screens and remote tunnels.

- Remote screens are connections that are established between the Management System and a target. They are visualized by the Management System in a new browser tab and support SSH, RDP and VNC protocols.
- Remote tunnels are connections that are established from the local workstation to a target, similar to a VPN connection. They allow access to services and servers on the target from the user's local workstation. Remote tunnels are managed and established in the Nerve Connection Manager application and the Management System. The locally opened connection endpoint can then be used in a web browser, with SSH clients, or with remote desktop applications, depending on the target.

The targets of these remote connections can be nodes, workloads or external devices, which can be accessed from the node through the network.

Remote connections to workloads can be defined in existing workloads. Note that a workload does not have to be deployed again if a remote connection has been added. Defining a remote connection to a workload adds the remote connection to the workload across the Management System, meaning that it will also be available if the workload has already been deployed to nodes.

Select **Remotes** in the navigation on the left to view a list of currently established remote connections.

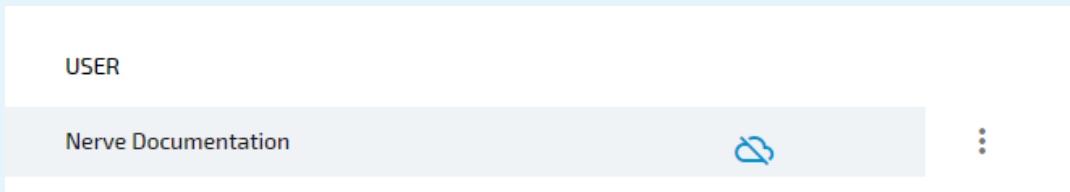


Item	Description
<b>Search bar (1)</b>	Use the search bar to filter remote connections by name or serial number.
<b>CONNECTION NAME (2)</b>	This is the name of the remote connection that is defined when the remote connection is configured.
<b>TARGET (3)</b>	This is the target of the remote connection. Note that this is not the name of the node. The hostname that was defined in the remote connection configuration is displayed here.

Item	Description
<b>SERIAL NUMBER (4)</b>	This is the serial number of the node to which the remote connection was established. In case of remote connections to workloads, the serial number of the node will be displayed to which the workload was deployed. For remote connections to external devices, the serial number of the node that the external device is connected to will be displayed.
<b>USER (5)</b>	This shows which user is using the established remote connection. If the same remote connection is used by two users, the remote connection will be listed again with a different user in the <b>User</b> column.
<b>Ellipsis menu (6)</b>	Clicking here opens an overlay that allows terminating connections.

#### NOTE

- Note that the list of active remote connections is not updated in real-time. Refresh the page to see changes.
- If a node goes offline while a remote connection is established, a cloud symbol will be displayed next to the remote connection to indicate that the connection to the node is interrupted.



## Remote screens

A remote screen is established from the Management System to the target. It is opened in a new tab in the used web browser as soon as the remote connection is established.

Below are instructions on how to create SSH, VNC and RDP connections to nodes and workloads in the Management System.

#### NOTE

If the target of the remote connection is the host of the Nerve system, use the IP address of the host: 172.20.2.1. Using localhost is not supported.

## Configuring an SSH connection to a node

An SSH connection to a node can be used for accessing the host operating system of the node or an external device connected to the node that is reachable through an SSH connection.

1. Select **Nodes** from the navigation on the left.
2. Select the nodes tab



on the right to display the list of registered nodes.

3. Select a node from the list to which a remote connection will be established.

NAME	SERIAL NUMBER	NODE VERSION	CREATED
documentation	001235555555	2.1.0-rc10	14/05/2020
DankalB	MFNBUSTERRRR	2.1.0	13/05/2020
documentation	008373032311	2.1.0-rcb	13/05/2020
Navi	MFN30BUSTERR	2.1.0	13/05/2020
mfn 47	MFN47BUSTERR	2.1.0-rc10	13/05/2020
buster	000439434001	buster	12/05/2020
documentation	000820190003	2.1.0	12/05/2020
VukoMfn	123456723456	2.1.0	12/05/2020
Bojan	MECHASIEMENS	2.1.0	12/05/2020
MFN713	091282746500	2.1.0-rc7	11/05/2020

4. Click **Add Remote Screen** under **REMOTE CONNECTIONS** on the right side.

**Update Node**

Name: documentation (13 / 40)

Secure ID: 87C5BA21E2C9547B (16 / 16)

Serial number: 008373032311 (12 / 12) MFN 100

Version: 2.1.0-rcb

**REMOTE CONNECTIONS**

- [Add Remote Screen](#)
- [Add Remote Tunnel](#)

5. Enter a name for the remote connection in the new window.

### NOTE

Make sure to use a unique name for every connection on a node to avoid confusion.

6. Select **SSH** from the drop-down menu under **Connection type**.
7. Enter the port used for SSH connection. The default port 22 is automatically filled in.
8. Enter the remaining information if applicable:

## NERVE PARAMETERS

### Number of connections

Enter the maximum number of simultaneous connections. The default value is 1.

### Local acknowledgment

Select **Yes** or **No** from the drop-down menu.

Selecting **Yes** will require approval of the remote connection in the Local UI before the connection can be established. If **No** is selected, the settings in the Local UI do not apply.

Refer to [Approving a remote connection](#) for information on how to approve remote connections in the Local UI.

## NETWORK PARAMETERS

### Hostname

Enter the IP address or the hostname of the target here.

### Autoretry

Set the number of retries if the remote connection fails. The default value is 1.

The display settings offer configuration options that affect visualization.

### Swap red blue

If colors appear to not be displayed correctly, select **true** from the drop-down menu. This can occur when using VNC servers. Select **false** otherwise.

### Cursor

This setting determines if the cursor is rendered locally or remotely. Enter **local** for a local cursor or enter **remote** for a remote cursor. If set to **remote**, the mouse pointer will be rendered remotely, and the local position of the mouse pointer will be indicated by a small dot. A remote mouse cursor will have added input lag compared to a local cursor. However, a remote cursor might be necessary if the server does not support sending the cursor image to the client.

## DISPLAY SETTINGS

### Read only

Select **true** or **false** from the drop-down menu. If set to **true**, no input will be accepted on the connection. Select **false** to allow input.



## AUTHENTICATION

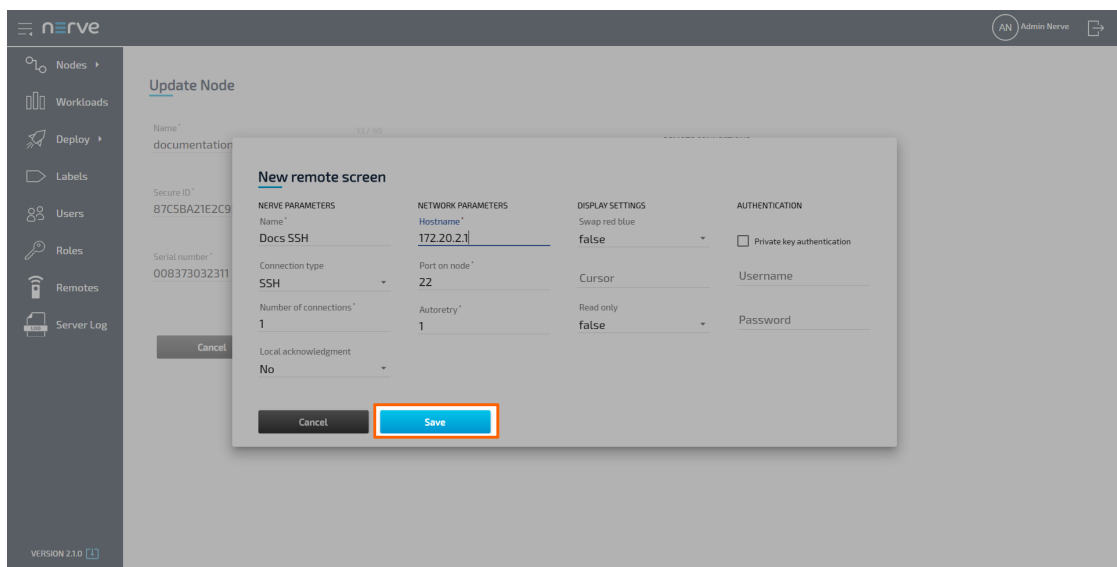
Enter **Username** and **Password** or tick the checkbox next to **Private key authentication** to use a private SSH key.

Note that ticking the checkbox changes the interface. Enter the username and choose one of the methods to add the private SSH key:

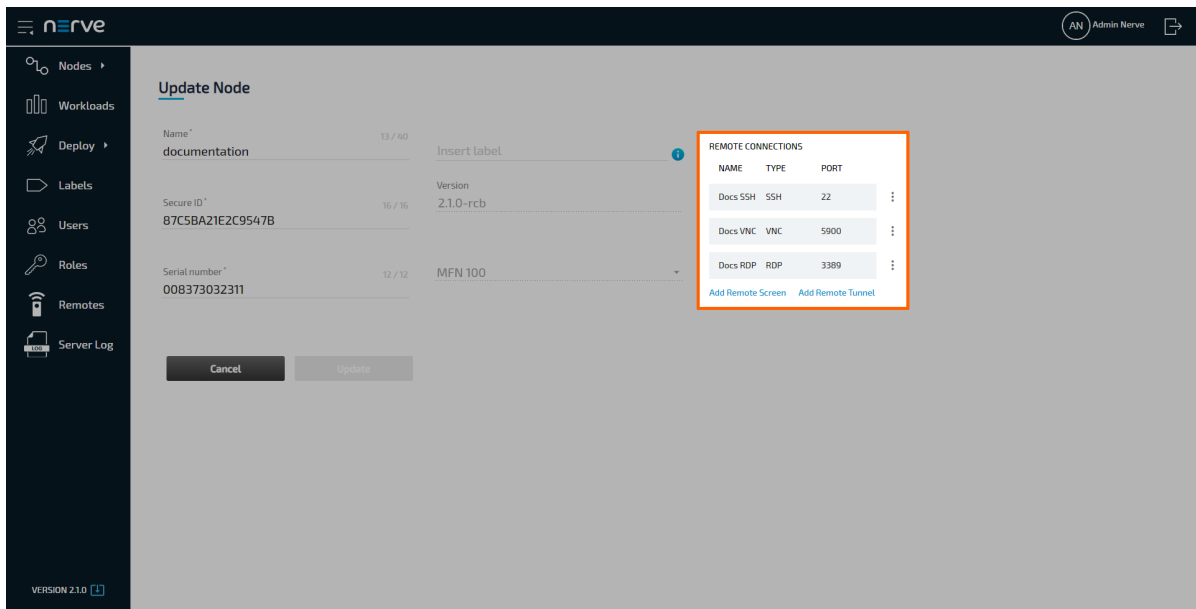
- Click **Choose File** to open the local file browser and select the private SSH key file.
- Drag and drop the private SSH key file into the dotted line box saying **Drop Private Key Here**.
- Copy the private SSH key and paste it into the empty input field.

Note that entering wrong login credentials will cause an error when the remote screen is established. If an error occurs, close the browser tab. Check the login credentials and re-establish the remote screen.

9. Select **Save** to add the remote connection.



The connection is now displayed under **REMOTE CONNECTIONS** on the right side, showing the **NAME**, **TYPE** and **PORT** of the remote connection.



## Configuring a VNC connection to a node

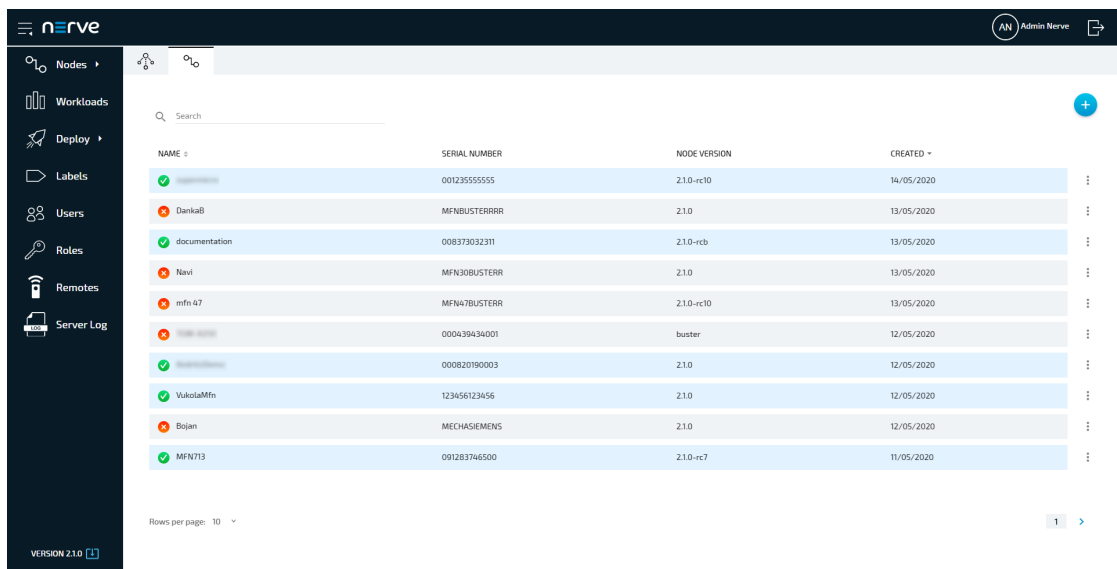
A VNC connection to a node can be used to connect to a Linux environment on an external device, which is connected to the node or to the same network that the node is connected to.

1. Select **Nodes** from the navigation on the left.
2. Select the nodes tab

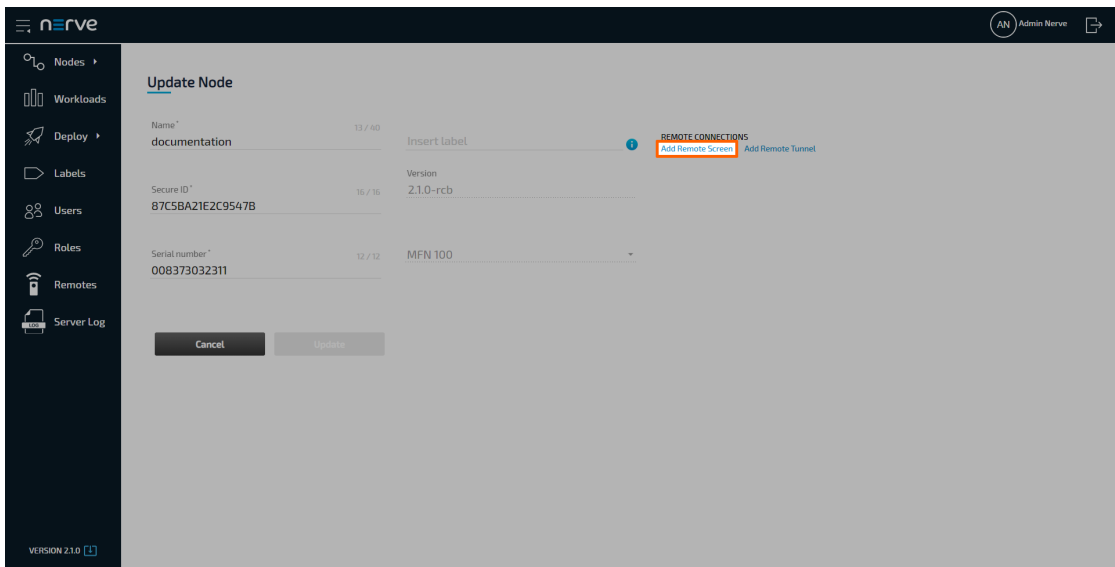


on the right to display the list of registered nodes.

3. Select a node from the list to which a remote connection will be established.



4. Click **Add Remote Screen** under **REMOTE CONNECTIONS** on the right side.



5. Enter a name for the remote connection in the new window.

#### NOTE

Make sure to use a unique name for every connection on a node to avoid confusion.

6. Select **VNC** from the drop-down menu under **Connection type**.
7. Enter the port used for VNC connection. The default port 5900 is automatically filled in.
8. Enter the password that was set for VNC connections at the target.

#### NOTE

Entering wrong login credentials will cause an error when the remote screen is established. If an error occurs, close the browser tab. Check the login credentials and re-establish the remote screen.

9. Enter the remaining information if applicable:

---

**NERVE  
PARAMETERS**

**Number of connections**

Enter the maximum number of simultaneous connections. The default value is 1.

**Local acknowledgment**

Select **Yes** or **No** from the drop-down menu.

Selecting **Yes** will require approval of the remote connection in the Local UI before the connection can be established. If **No** is selected, the settings in the Local UI do not apply.

Refer to [Approving a remote connection](#) for information on how to approve remote connections in the Local UI.

**NETWORK  
PARAMETERS**

**Hostname**

Enter the IP address or the hostname of the target here.

**Autoretry**

Set the number of retries if the remote connection fails. The default value is 1.

The display settings offer configuration options that affect visualization.

**Swap red blue**

If colors appear to not be displayed correctly, select **true** from the drop-down menu. This can occur when using VNC servers. Select **false** otherwise.

**DISPLAY  
SETTINGS**

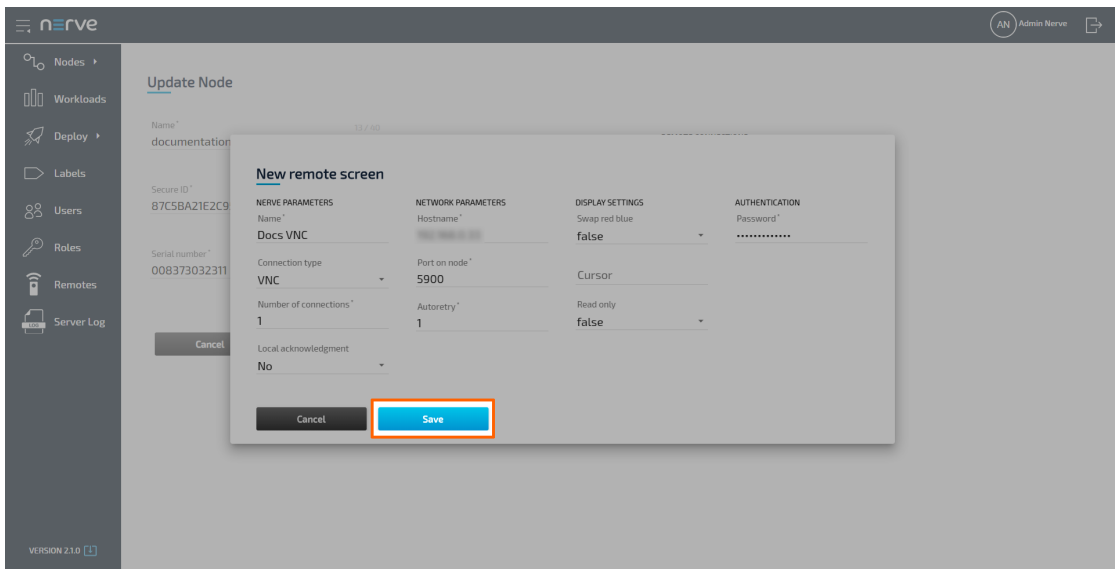
**Cursor**

This setting determines if the cursor is rendered locally or remotely. Enter `local` for a local cursor or enter `remote` for a remote cursor. If set to `remote`, the mouse pointer will be rendered remotely, and the local position of the mouse pointer will be indicated by a small dot. A remote mouse cursor will have added input lag compared to a local cursor. However, a remote cursor might be necessary if the server does not support sending the cursor image to the client.

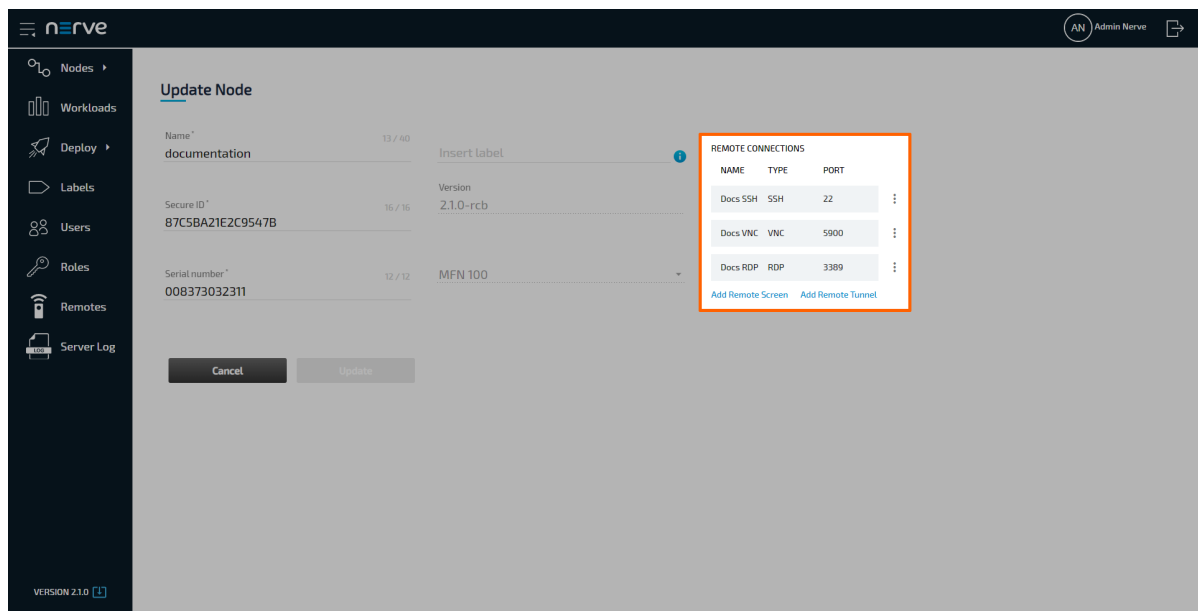
**Read only**

Select **true** or **false** from the drop-down menu. If set to **true**, no input will be accepted on the connection. Select **false** to allow input.

10. Select **Save** to add the remote connection.



The connection is now displayed under **REMOTE CONNECTIONS** on the right side, showing the **NAME**, **TYPE** and **PORT** of the remote connection.



## Configuring an RDP connection to a node

An RDP connection to a node can be used to connect to a Windows environment on an external device, which is connected to the node or to the same network that the node is connected to.

1. Select **Nodes** from the navigation on the left.
2. Select the nodes tab



on the right to display the list of registered nodes.

3. Select a node from the list to which a remote connection will be established.

NAME	SERIAL NUMBER	NODE VERSION	CREATED
documentation	001235555555	2.1.0-rc10	14/05/2020
DankaB	MFNBUSTERRRR	2.1.0	13/05/2020
documentation	008373032311	2.1.0-rcb	13/05/2020
Navi	MFN30BUSTERR	2.1.0	13/05/2020
mfn 47	MFN47BUSTERR	2.1.0-rc10	13/05/2020
buster	000439434001	buster	12/05/2020
documentation	000820190003	2.1.0	12/05/2020
VukolaMfn	123456123456	2.1.0	12/05/2020
Bojan	MECHASIEMENS	2.1.0	12/05/2020
MFN713	091283746500	2.1.0-rc7	11/05/2020

4. Click **Add Remote Screen** under **REMOTE CONNECTIONS** on the right side.

**Update Node**

Name: documentation

Secure ID: 87C5BA21E2C9547B

Serial number: 008373032311

Version: 2.1.0-rcb

Serial number: MFN 100

Buttons: Cancel, Update

REMOTE CONNECTIONS: Add Remote Screen, Add Remote Tunnel

5. Enter a name for the remote connection in the new window.

**NOTE**

Make sure to use a unique name for every connection on a node to avoid confusion.

6. Select **RDP** from the drop-down menu under **Connection type**.

7. Enter the port used for RDP connection. The default port 3389 is automatically filled in.

8. Enter the remaining information if applicable:

---

**NERVE  
PARAMETERS**

**Number of connections**

Enter the maximum number of simultaneous connections. The default value is 1.

**Local acknowledgment**

Select **Yes** or **No** from the drop-down menu.

Selecting **Yes** will require approval of the remote connection in the Local UI before the connection can be established. If **No** is selected, the settings in the Local UI do not apply.

Refer to [Approving a remote connection](#) for information on how to approve remote connections in the Local UI.

## NETWORK PARAMETERS

### **Hostname**

Enter the IP address or the hostname of the target here.

### **Autoretry**

Set the number of retries if the remote connection fails. The default value is 1.

### **Security mode**

This mode dictates how data will be encrypted and what type of authentication will be performed, if any. Select an option from the drop-down menu. Possible values are:

- **ANY**  
This is the default if the field is left blank. Automatically select the security mode based on the security protocols supported by both the client and the server.
- **NLA (Network Level Authentication)**  
This mode uses TLS encryption and requires the username and password to be given in advance. Unlike RDP mode, the authentication step is performed before the remote desktop session actually starts, avoiding the need for the Windows server to allocate significant resources for users that may not be authorized.
- **RDP encryption**  
This is the standard RDP encryption. It is generally only used for older Windows servers or in cases where a standard Windows login screen is desired. Newer versions of Windows have this mode disabled by default and will only accept NLA unless explicitly configured otherwise.
- **TLS encryption**  
Select this for RDP authentication and encryption implemented via TLS (Transport Layer Security). The TLS security mode is primarily used in load balanced configurations where the initial RDP server may redirect the connection to a different RDP server.

### **Ignore Server Certificate**

If checked, the certificate returned by the server will be ignored, even if that certificate cannot be validated. This is useful if the server and the connection to the server is universally trusted, and if the server's certificate cannot be validated (for example, if it is self-signed).



## DISPLAY SETTINGS

The display settings offer configuration options that affect visualization.

### Swap red blue

If colors appear to not be displayed correctly, select **true** from the drop-down menu. This can occur when using VNC servers. Select **false** otherwise.

### Cursor

This setting determines if the cursor is rendered locally or remotely. Enter `local` for a local cursor or enter `remote` for a remote cursor. If set to `remote`, the mouse pointer will be rendered remotely, and the local position of the mouse pointer will be indicated by a small dot. A remote mouse cursor will have added input lag compared to a local cursor. However, a remote cursor might be necessary if the server does not support sending the cursor image to the client.

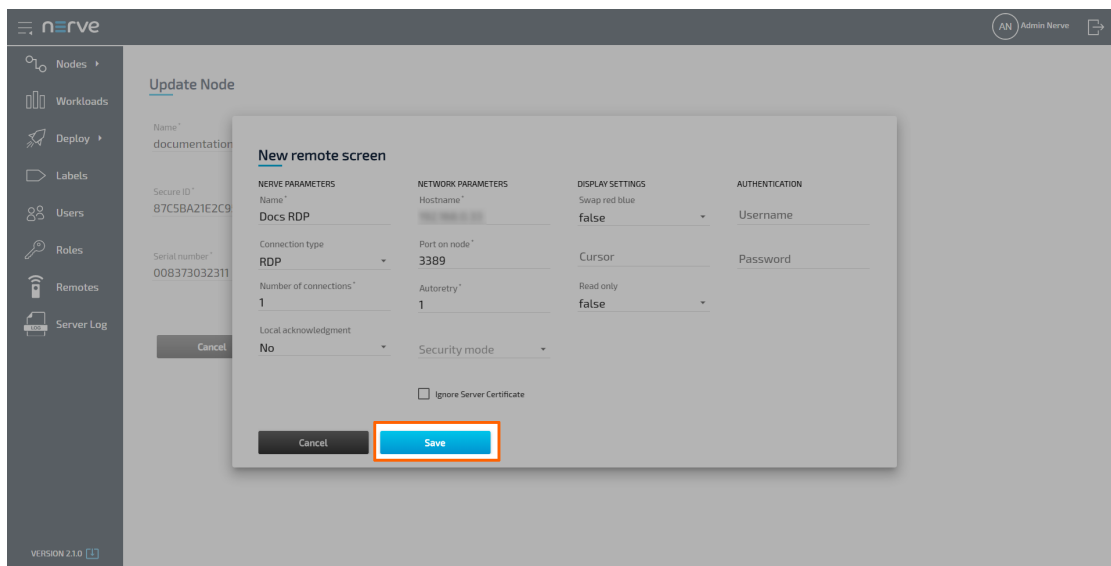
### Read only

Select **true** or **false** from the drop-down menu. If set to **true**, no input will be accepted on the connection. Select **false** to allow input.

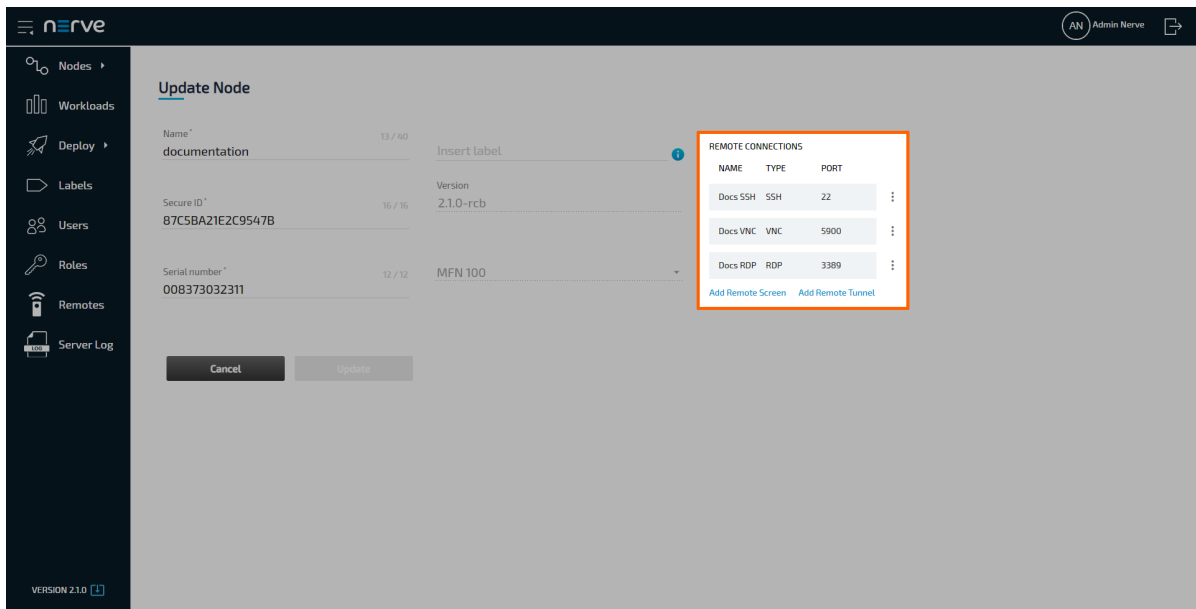
## AUTHENTICATION

Enter **Username** and **Password** for Windows login. Note that entering wrong login credentials will cause an error when the remote screen is established. If an error occurs, close the browser tab. Check the login credentials and re-establish the remote screen.

9. Select **Save** to add the remote connection.



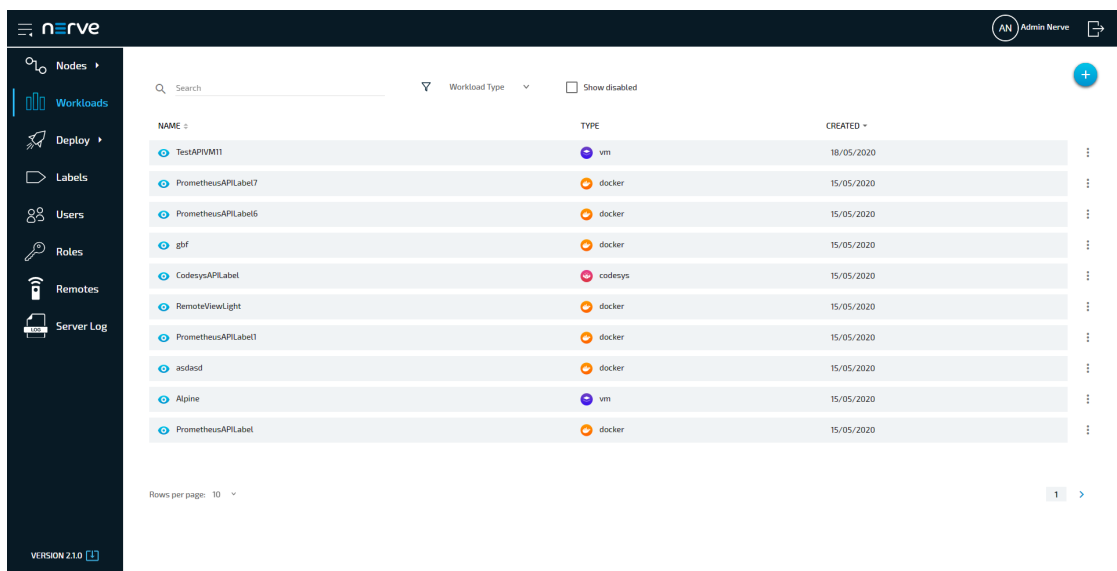
The connection is now displayed under **REMOTE CONNECTIONS** on the right side, showing the **NAME**, **TYPE** and **PORT** of the remote connection.



## Configuring a remote screen to a workload

A remote screen to a workload can be configured, regardless of a workload being deployed or not. Configuring a remote screen for a workload will immediately add the remote screen to the workload on all nodes that it has been deployed to. Note that remote screens to CODESYS workloads cannot be established.

1. Select **Workloads** in the navigation on the left.
2. Select a workload from the list.



3. Select the workload version to which the remote connection will be added.

### NOTE

Note that the configured remote connection will only be available for the version that was selected.

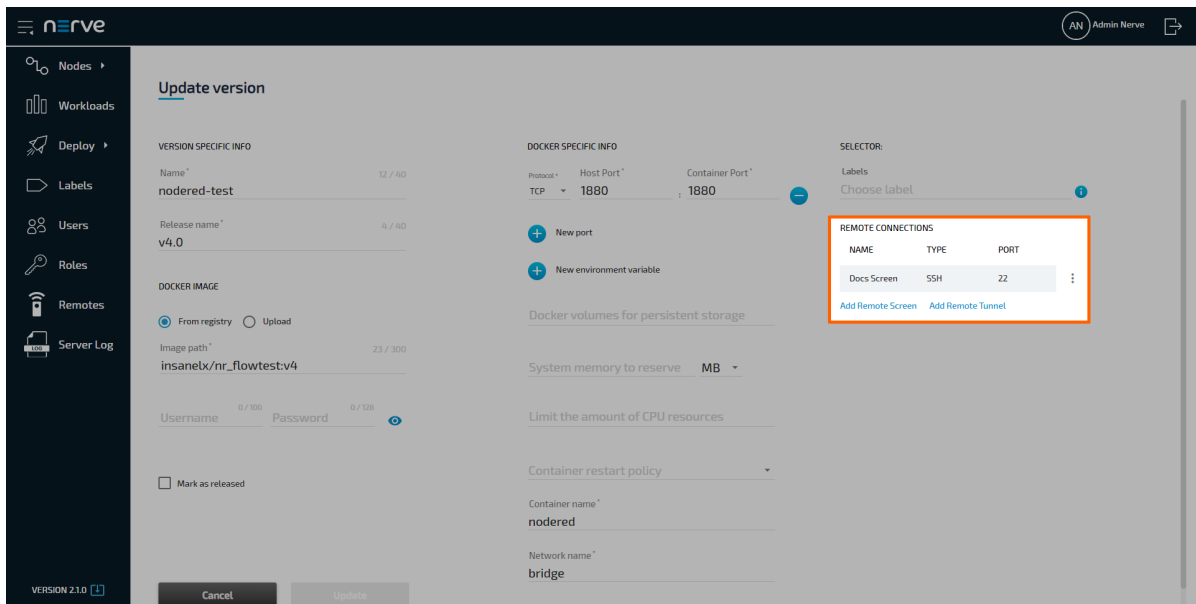
- Click **Add Remote Screen** under **REMOTE CONNECTIONS** on the right side.
- Follow steps 4 to 9 in the instructions above for [SSH](#), [VNC](#), or [RDP](#) connections.

### NOTE

Note that adding the hostname is not required when configuring a remote screen to a Docker workload. The system automatically detects the hostname when the workload is deployed.

In case of Virtual Machine workloads, the hostname entry is not displayed for VNC connections. For SSH and RDP connections, enter the IP address or hostname under **VM hostname / IP**.

The connection is saved and now displayed under **REMOTE CONNECTIONS** on the right side, showing the **NAME**, **TYPE** and **PORT** of the remote connection.



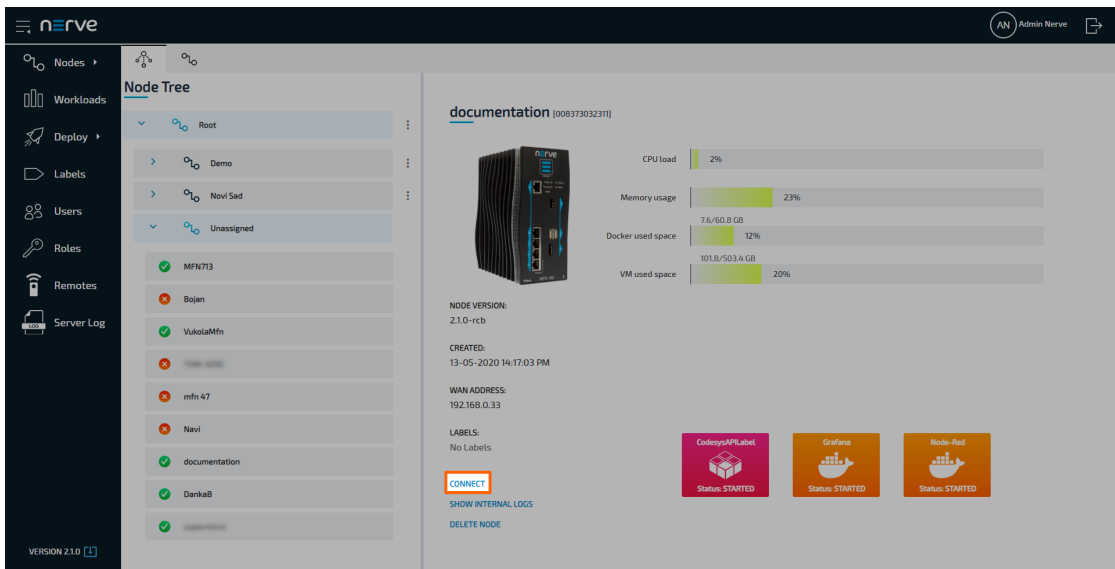
## Using a remote screen to a node or external device

Established remote screens are listed under **Remotes** in the navigation on the left until they are terminated.

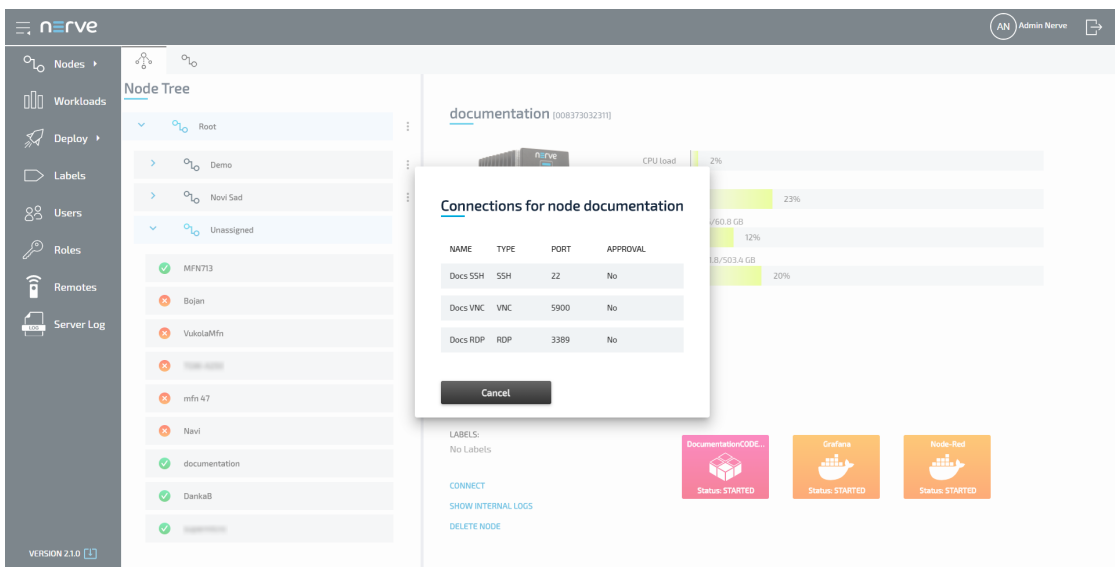
- Select **Nodes** in the navigation on the left.
- Select the node tree tab



- on the right to display registered nodes in the node tree.
- Select a node with a remote screen from the node tree.
- Click **CONNECT** in the node details on the right.



5. Select the remote connection from the list in the new window.



The remote screen will be opened and displayed in a new browser tab after a few seconds if **Local acknowledgement** has been set to **No**. If set to **Yes**, the remote connection has to be approved in the Local UI. Refer to [Approving a remote connection](#) for more information.

#### NOTE

Make sure not to exceed the defined number of connections of the same remote screen. This causes an error and the connection has to be terminated and established again. If there is a connection error, close the tab, terminate and re-establish the connection.

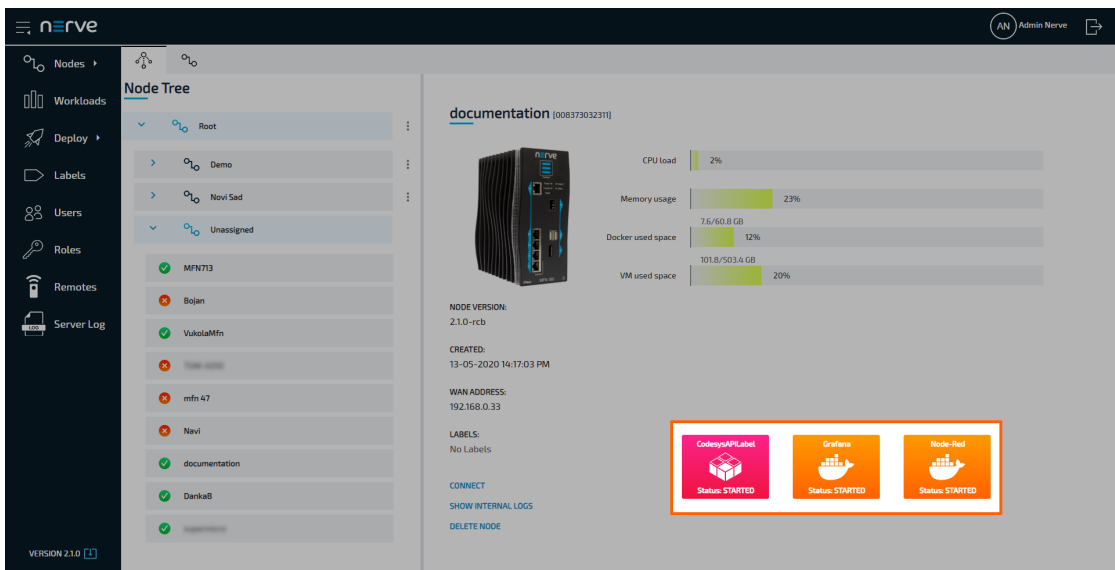
## Using a remote screen to a workload

Established remote screens are listed under **Remotes** in the navigation on the left until they are terminated.

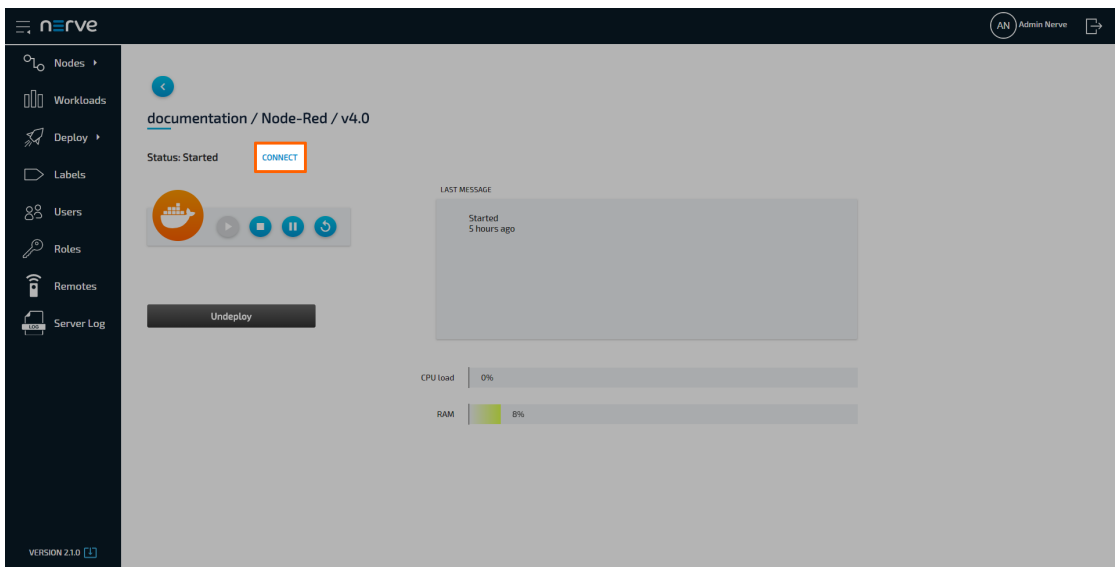
1. Select **Nodes** in the navigation on the left.
2. Select the node tree tab



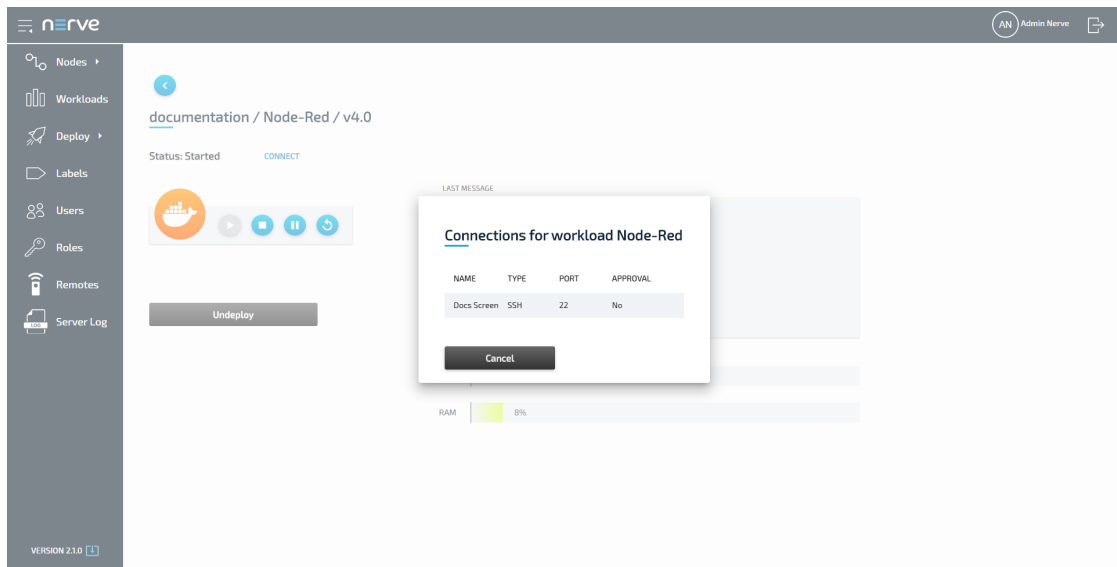
- on the right to display registered nodes in the node tree.
3. Select the node that has a deployed workload with a remote connection.
4. Select the workload.



5. Click **CONNECT** next to the workload status.



6. Select the remote connection from the list in the new window.



The remote screen will be opened and displayed in a new browser tab after a few seconds if **Local acknowledgement** has been set to **No**. If set to **Yes**, the remote connection has to be approved in the Local UI. Refer to [Approving a remote connection](#) for more information.

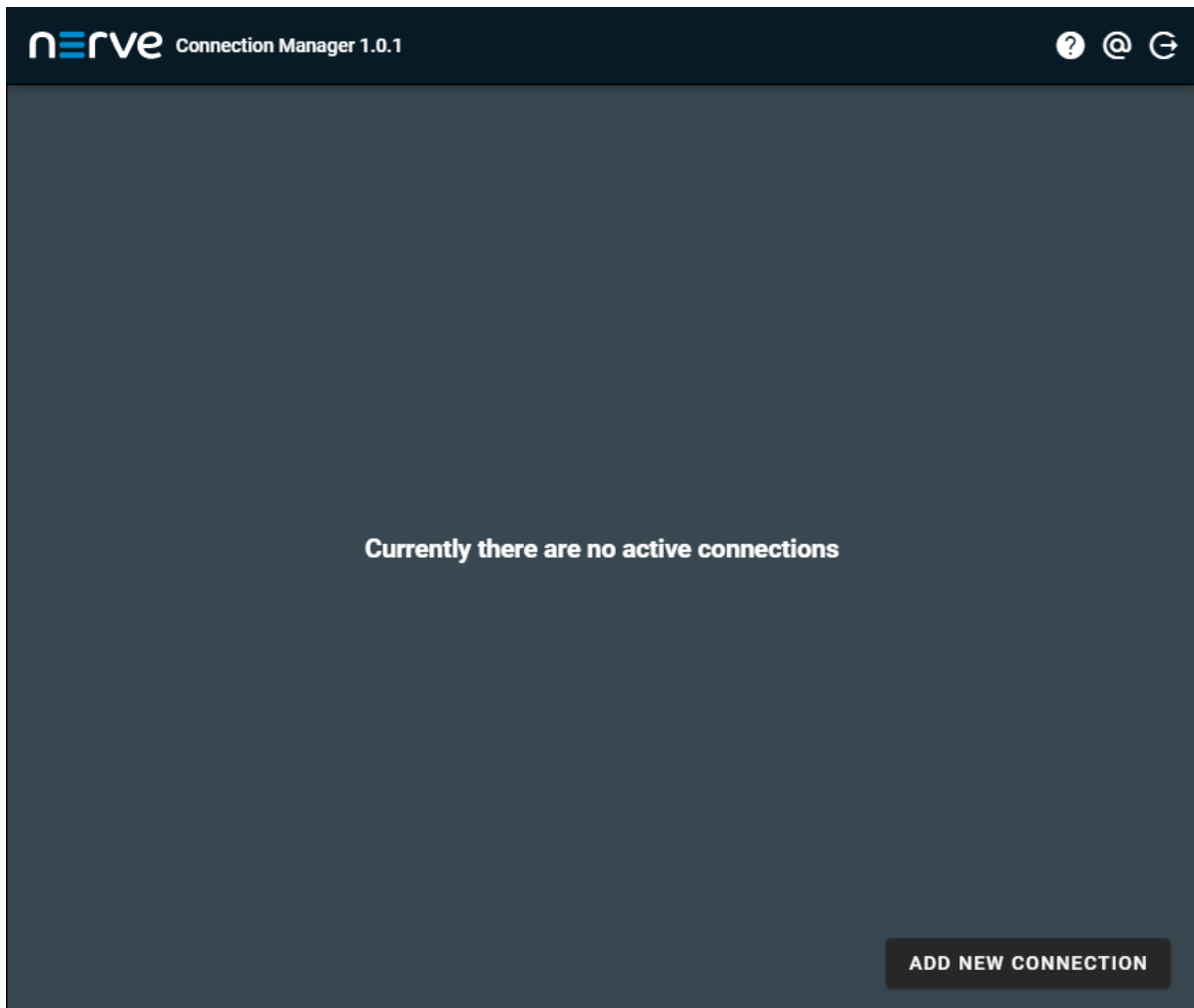
#### NOTE

Make sure not to exceed the defined number of connections of the same remote screen. This causes an error and the connection has to be terminated and established again. If there is a connection error, close the tab, terminate and re-establish the connection.

## Remote tunnels

The Nerve Connection Manager is an application that is installed locally on the workstation. It is required for establishing and using remote connections from the local workstation. Download the Nerve Connection Manager from the [Nerve Software Center](#) first.

The Nerve Connection Manager installation file is an executable file. Open the installation file and follow the installation process. The filename of the installation file is Nerve Connection Manager Setup <version>.exe on Windows or Nerve Connection Manager Setup <version>.deb on Linux.



Once installed, the Nerve Connection Manager will be associated with `nerverm://` links that are generated in the Management System. Clicking such a link will automatically open the Nerve Connection Manager.

**NOTE**

If the target of the remote connection is the host of the Nerve system, use the IP address of the host: `172.20.2.1`. Using `localhost` is not supported.

**Compatibility of the Nerve Connection Manager**

Make sure that the correct version of the Nerve Connection Manager is installed according to the version of the Management System that is used:

Management System	Nerve Connection Manager
v2.2.X and higher	v1.0.2
v2.1.X	v1.0.1
v2.1.0	v1.0.0

## Configuring a remote tunnel to a node

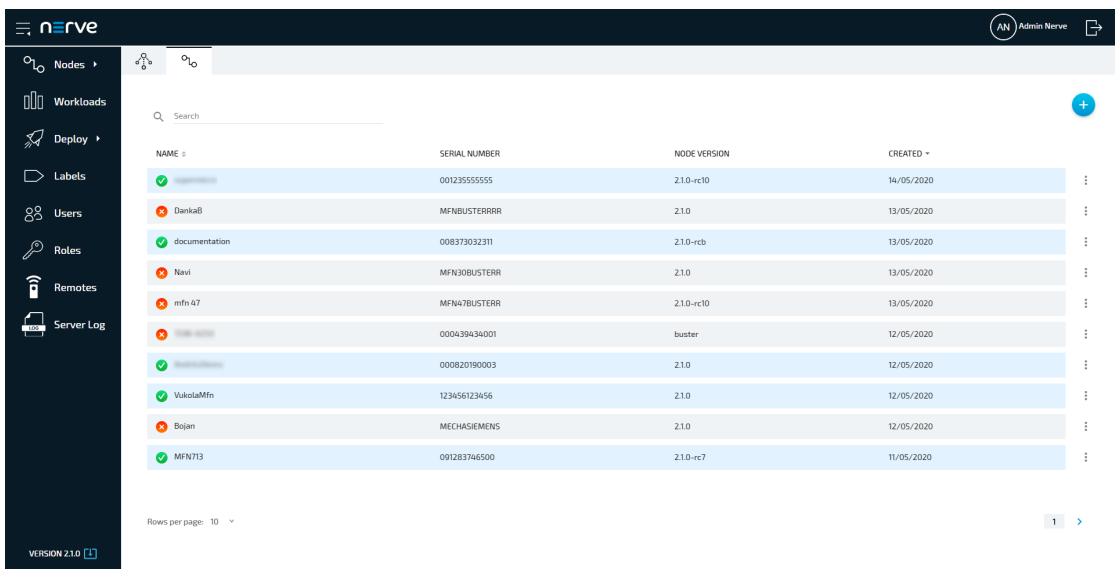
Depending on the target, a remote tunnel to a node can be used in a web browser, with SSH clients, or with remote desktop applications, for example.

1. Select **Nodes** in the navigation on the left.
2. Select the nodes tab

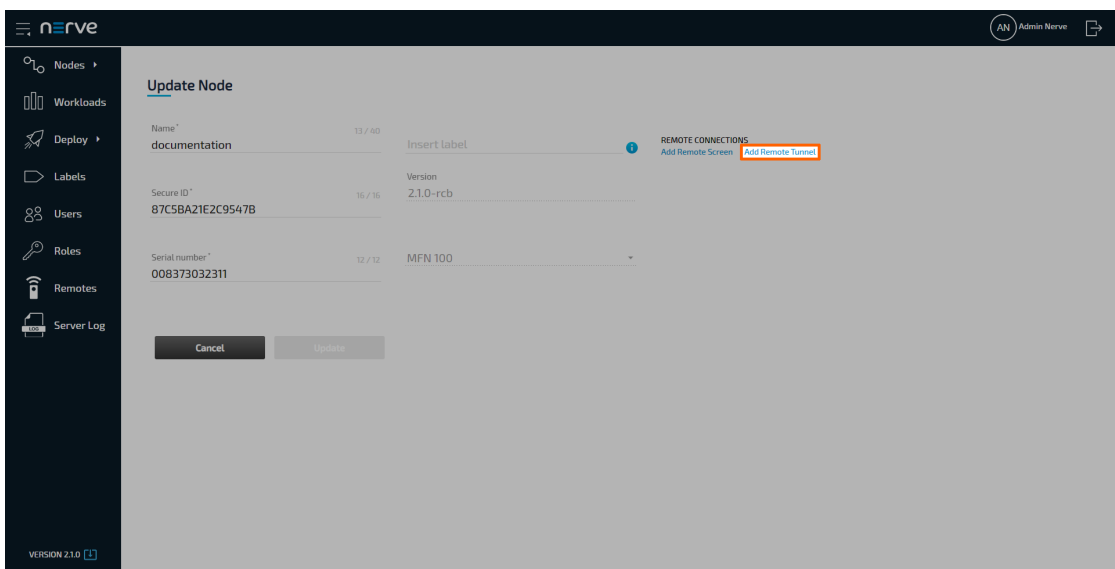


on the right to display the list of registered nodes.

3. Select a node from the list.



4. Select **Add Remote Tunnel** under **REMOTE CONNECTIONS** on the right side.



5. Enter the following information:



## NERVE PARAMETERS

### Name

Enter a name for the remote connection. Make sure to use a unique name for every connection on a node to avoid confusion.

### Local acknowledgment

Select **Yes** or **No** from the drop-down menu.

Selecting **Yes** will require approval of the remote connection in the Local UI before the connection can be established. If **No** is selected, the settings in the Local UI do not apply.

Refer to [Approving a remote connection](#) for information on how to approve remote connections in the Local UI.

## NETWORK PARAMETERS

### Hostname

Enter the IP address or the hostname of the target here.

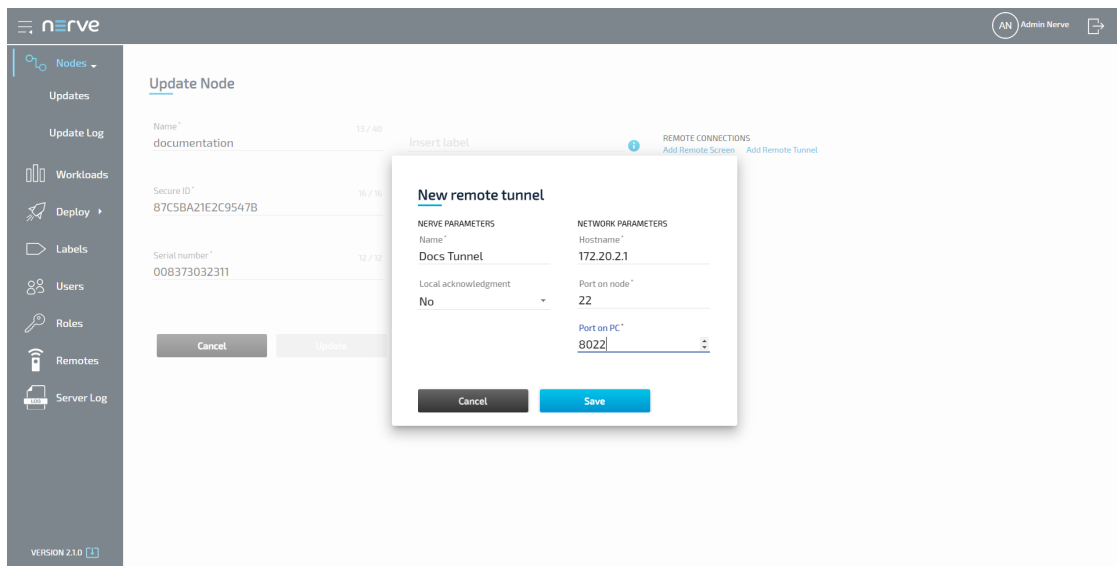
### Port on node

Enter the port the target listens on.

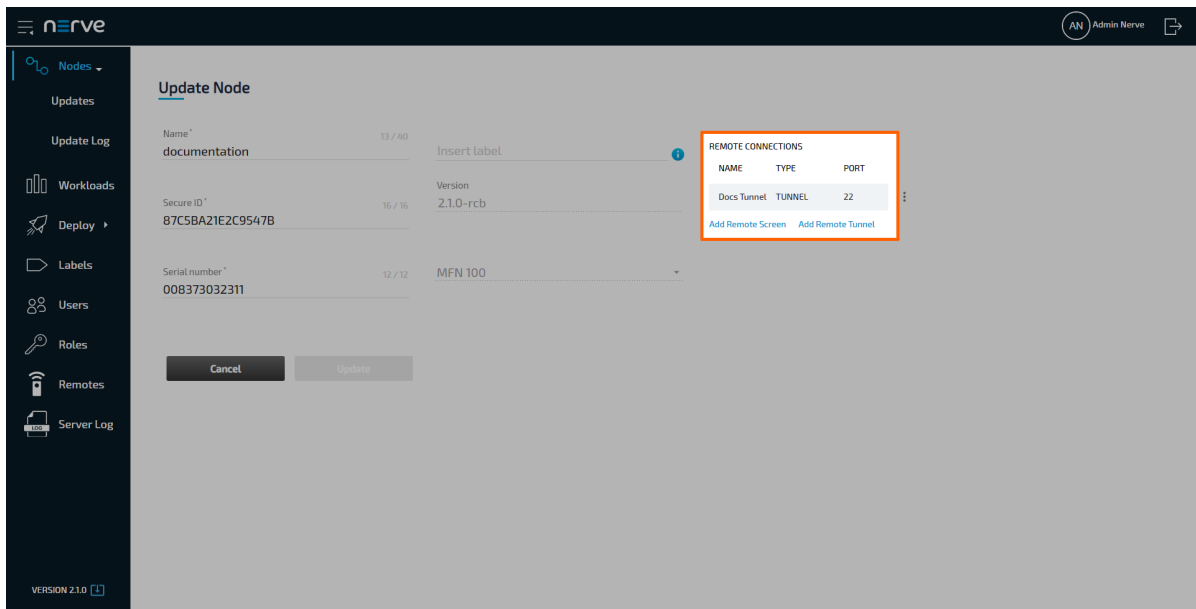
### Port on PC

Enter the port that will be used for communication on the local workstation. The port entered here serves as a default port that can be changed in the Nerve Connection Manager in case it is already in use. Note that some systems might restrict usage of ports under 1024. This is true for Linux systems especially. Enter port numbers higher than 1024 to avoid possible port conflicts.

6. Select **Save** to save the remote connection configuration.



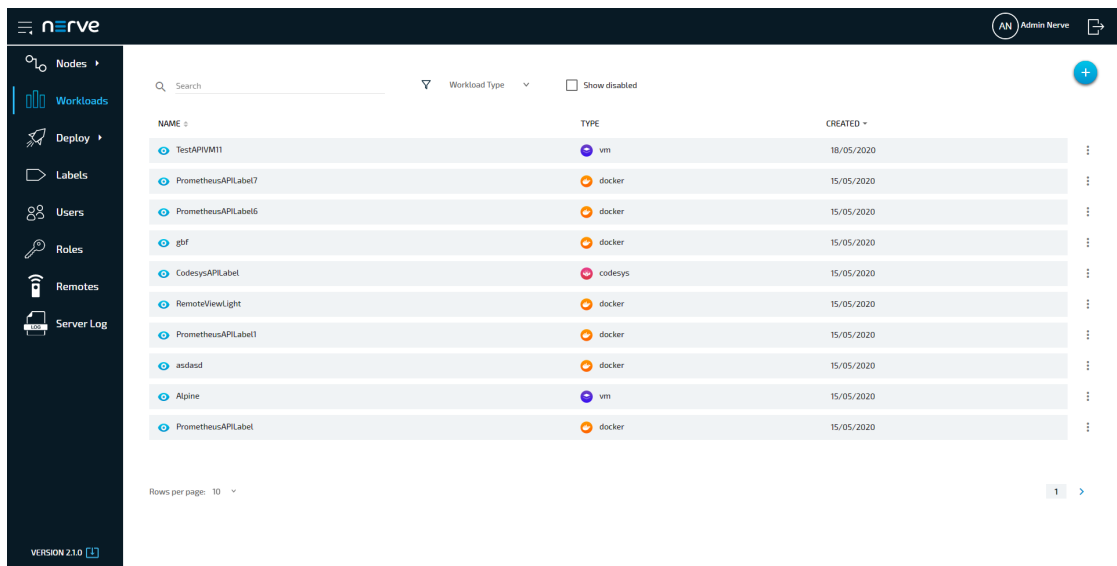
The connection is saved and now displayed under **REMOTE CONNECTIONS** on the right side, showing the **NAME**, **TYPE** and **PORT** of the remote connection.



## Configuring a remote tunnel to a workload

A remote tunnel to a workload can be configured, regardless of a workload being deployed or not. Configuring a remote tunnel for a workload will immediately add the remote tunnel to the workload on all nodes that it has been deployed to. Depending on the target, a remote tunnel to a workload can be used in a web browser, with SSH clients, or with remote desktop applications, for example.

1. Select **Workloads** in the navigation on the left.
2. Select a workload from the list.

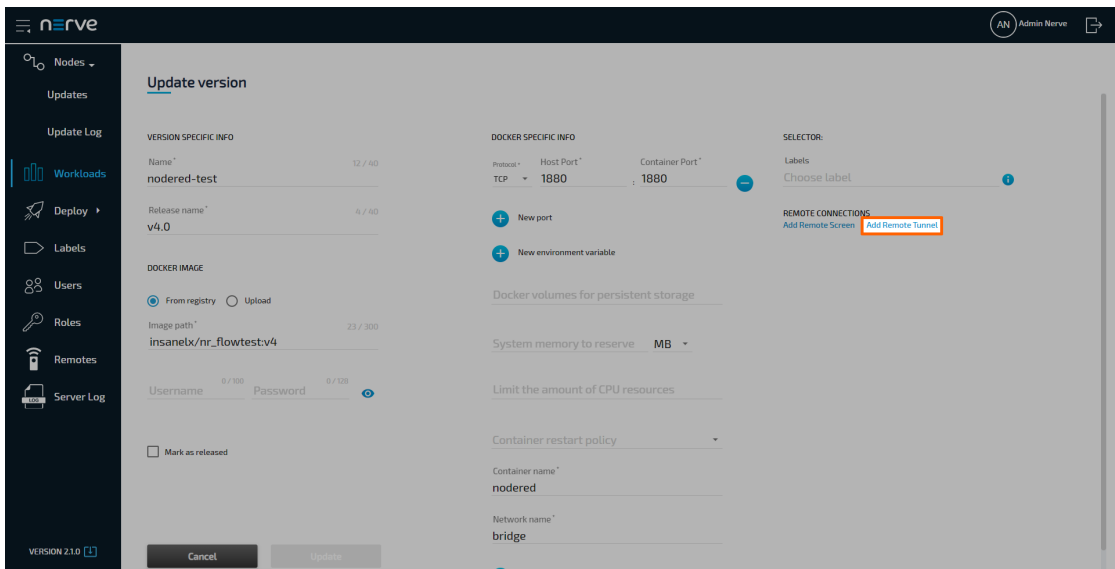


3. Select the workload version to which a remote connection will be established.

## NOTE

Note that the configured remote connection will only be available for the version that was selected.

4. Select **Add Remote Tunnel** under **REMOTE CONNECTIONS** on the right side.



5. Enter the following information:

### NERVE PARAMETERS

#### Name

Enter a name for the remote connection. Make sure to use a unique name for every connection on a node to avoid confusion.

#### Local acknowledgment

Select **Yes** or **No** from the drop-down menu.

Selecting **Yes** will require approval of the remote connection in the Local UI before the connection can be established. If **No** is selected, the settings in the Local UI do not apply.

Refer to [Approving a remote connection](#) for information on how to approve remote connections in the Local UI.

### NETWORK PARAMETERS

#### Port on node

Enter the port the target listens on.

#### Port on PC

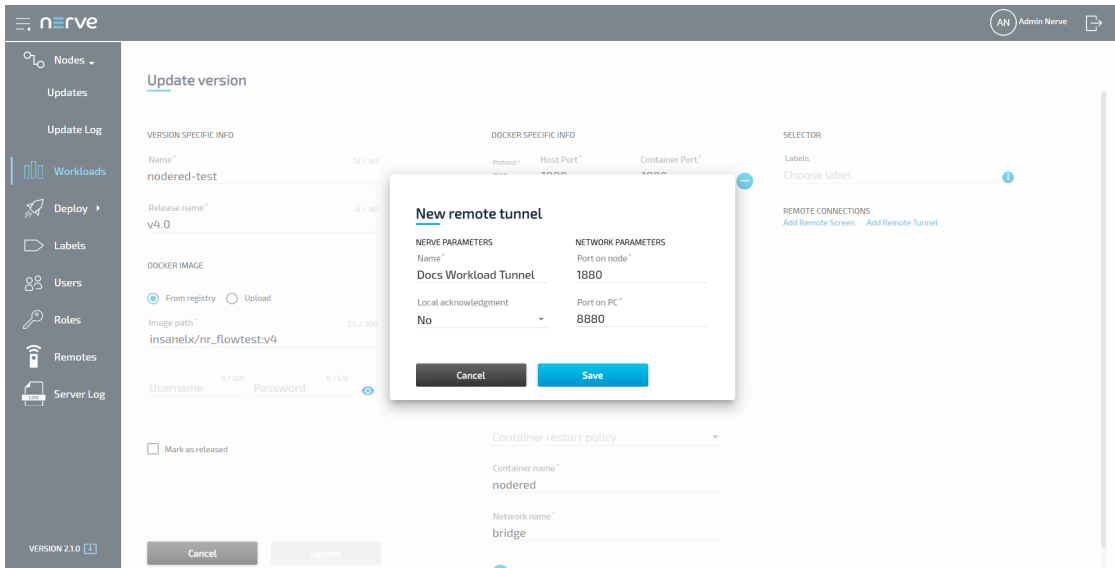
Enter the port that will be used for communication on the local workstation. The port entered here serves as a default port that can be changed in the Nerve Connection Manager in case it is already in use. Note that some systems might restrict usage of ports under 1024. This is true for Linux systems especially. Enter port numbers higher than 1024 to avoid possible port conflicts.

## NOTE

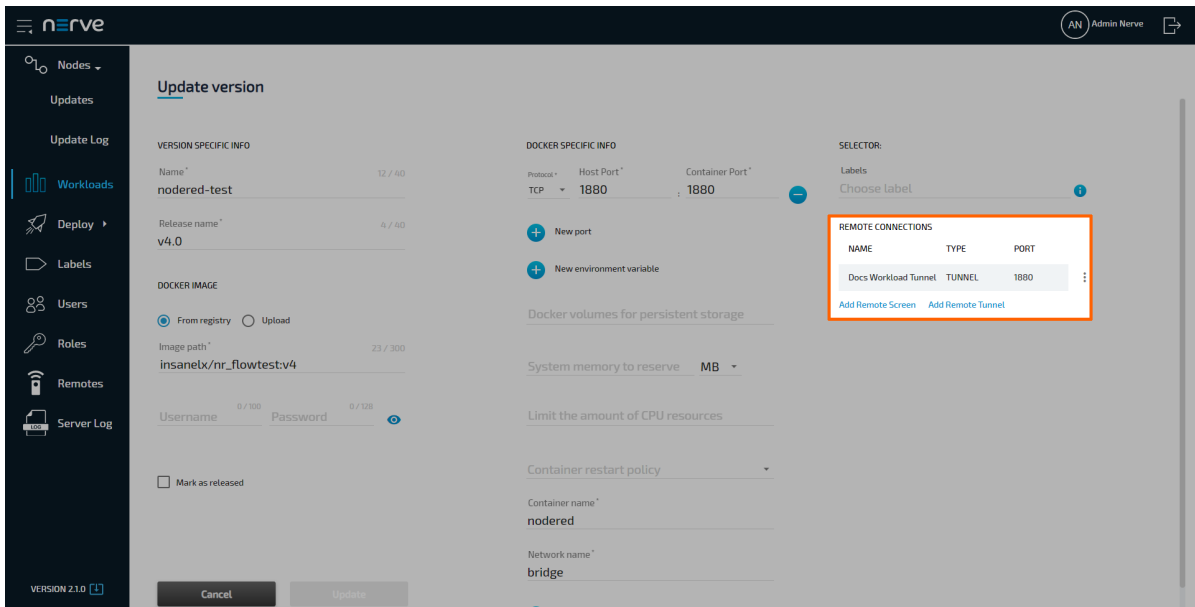
Note that adding the hostname is not required when configuring a remote tunnel to a Docker workload. The system automatically detects the hostname when the workload is deployed.

For CODESYS workloads, the **Hostname** and **Port on node** fields are filled in by the default. They contain the IP address and default port of the CODESYS runtime.

### 6. Select **Save** to save the remote connection configuration.



The connection is saved and now displayed under **REMOTE CONNECTIONS** on the right side, showing the **NAME**, **TYPE** and **PORT** of the remote connection.



## Using a remote tunnel to a node or external device

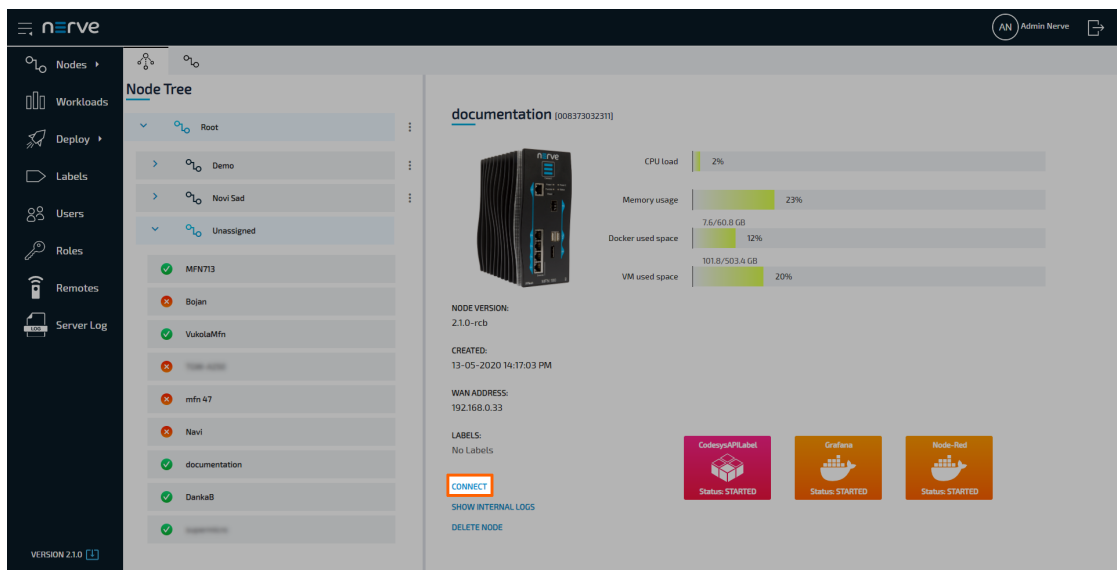
Note that the Nerve Connection Manager is required to use a remote tunnel. Download the Nerve Connection Manager from the [Nerve Software Center](#) and install it first.

Established remote tunnels are listed under **Remotes** in the navigation on the left until they are terminated.

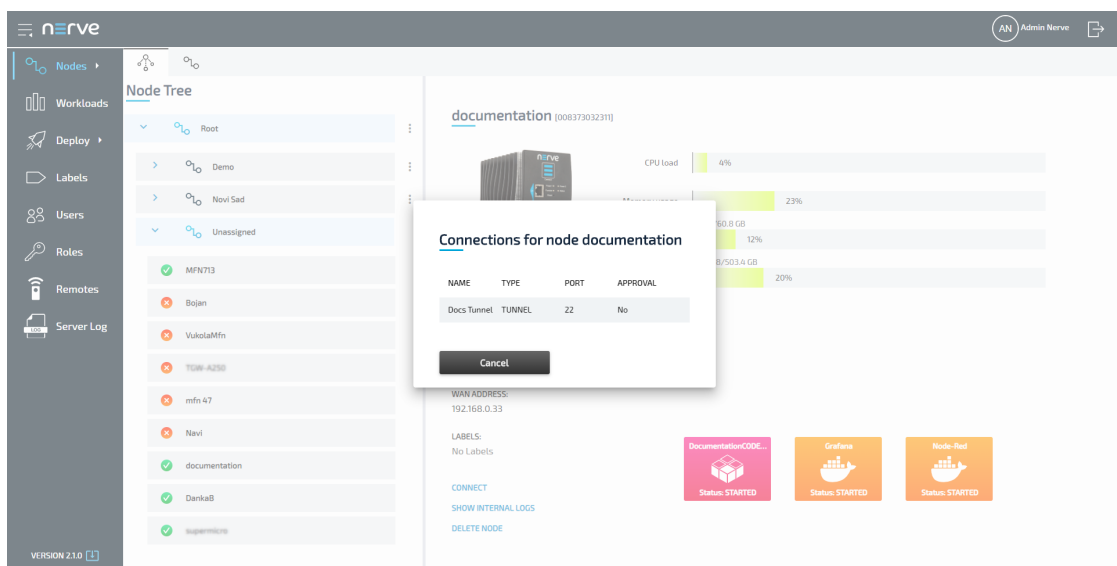
1. Select **Nodes** in the navigation on the left.
2. Select the node tree tab



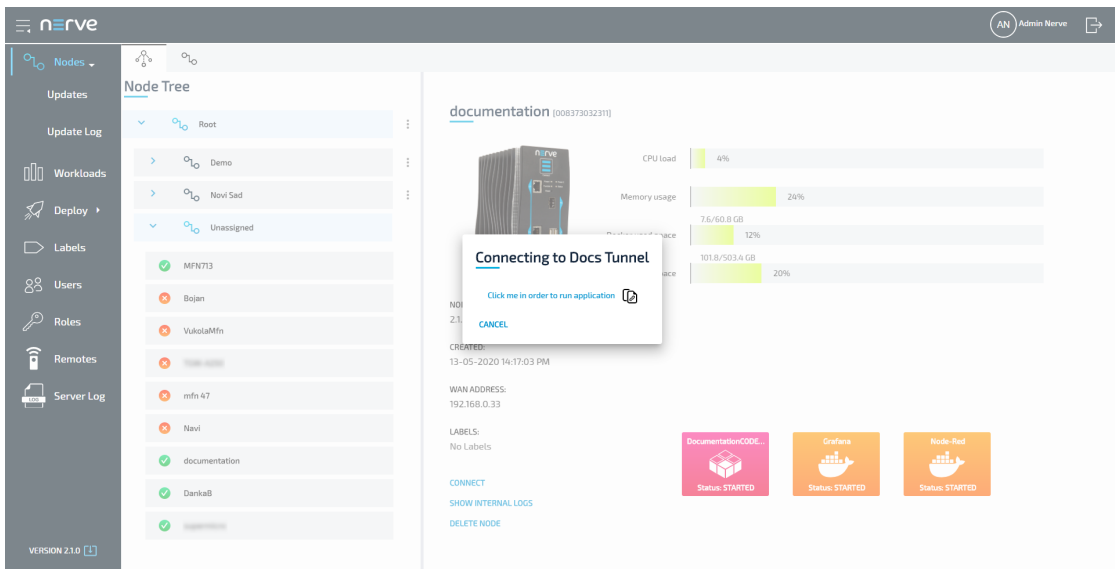
3. Select a node with a remote tunnel from the node tree.
4. Click **CONNECT** in the node details on the right.



5. Select the remote connection from the list in the new window. Note that remote tunnels have the type **TUNNEL**.



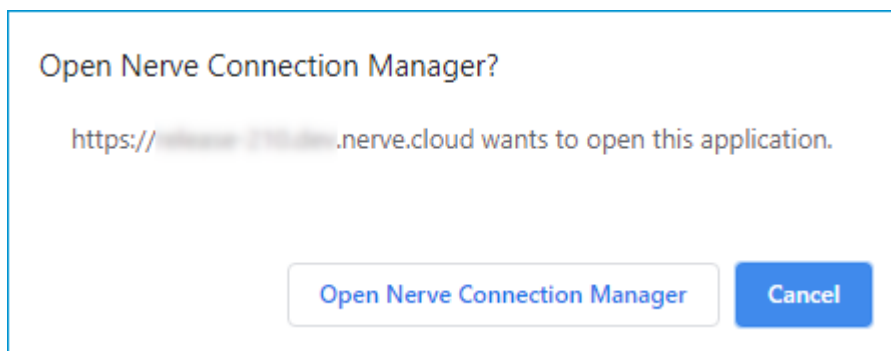
6. Select **Click me in order to run application** in the new window.



## NOTE

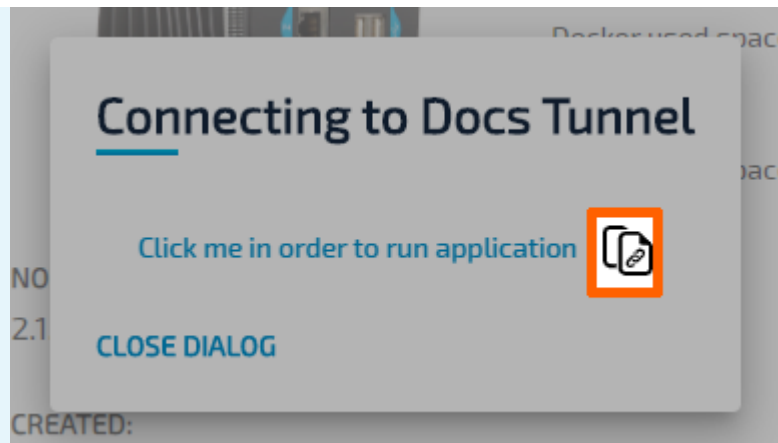
If **Local acknowledgment** is set to **Yes**, the Management System will wait for approval until the remote connection has been locally approved before displaying the window above. Refer to [Approving a remote connection](#) for more information.

- If the Nerve Connection Manager installed correctly, confirm the browser message that the Nerve Connection Manager shall be opened. Depending on the browser that is used, this message will differ. The Nerve Connection Manager will start automatically once the message is confirmed.

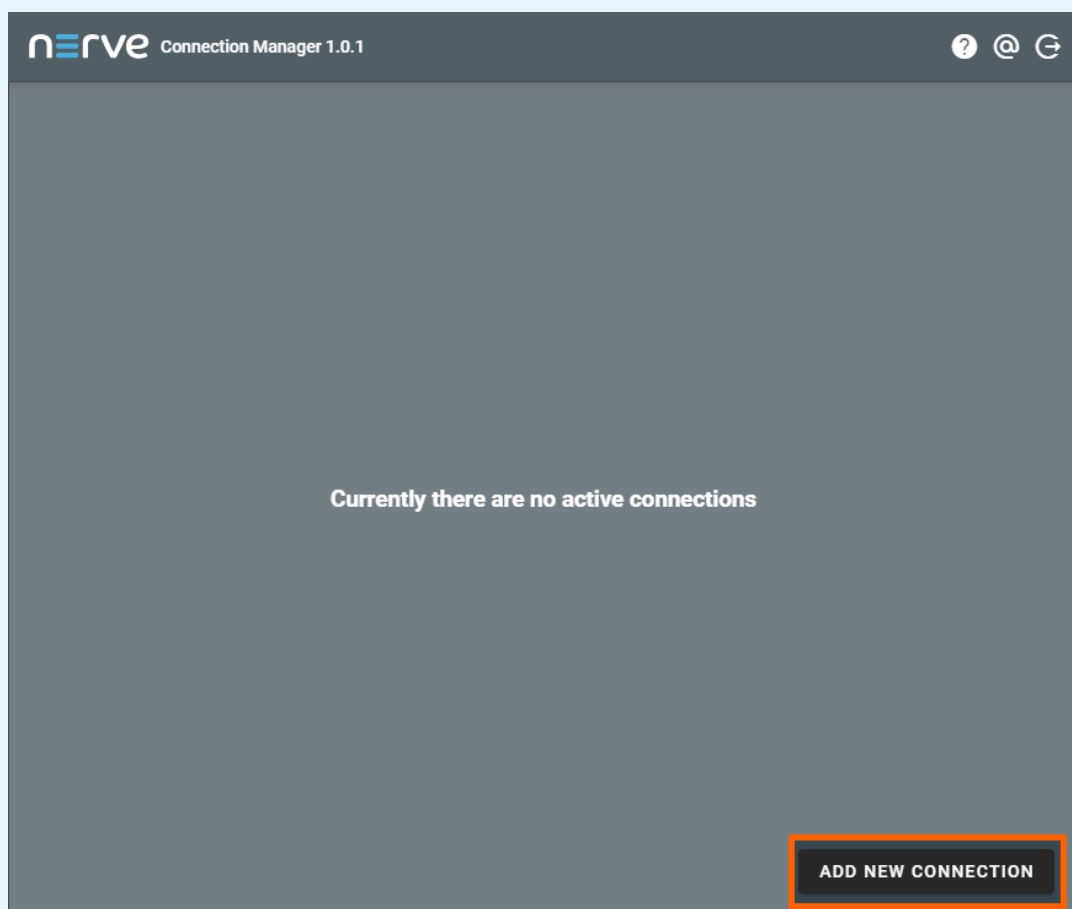


## NOTE

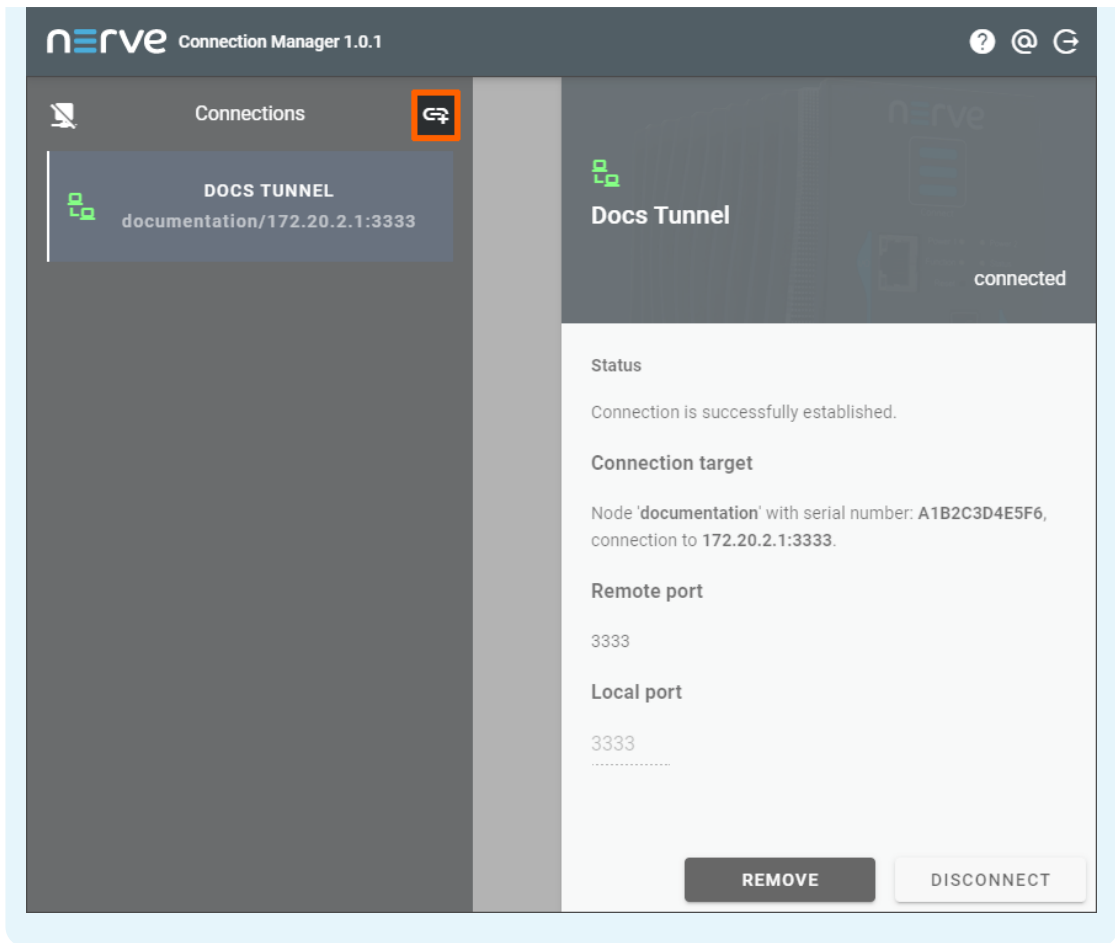
If the Nerve Connection Manager does not start automatically, select the copy to clipboard symbol next to **Click me in order to run application** in the Management System. This copies the remote connection URL.



Start the Nerve Connection Manager manually and add the new connection by clicking **ADD NEW CONNECTION** in the lower right and pasting the URL.

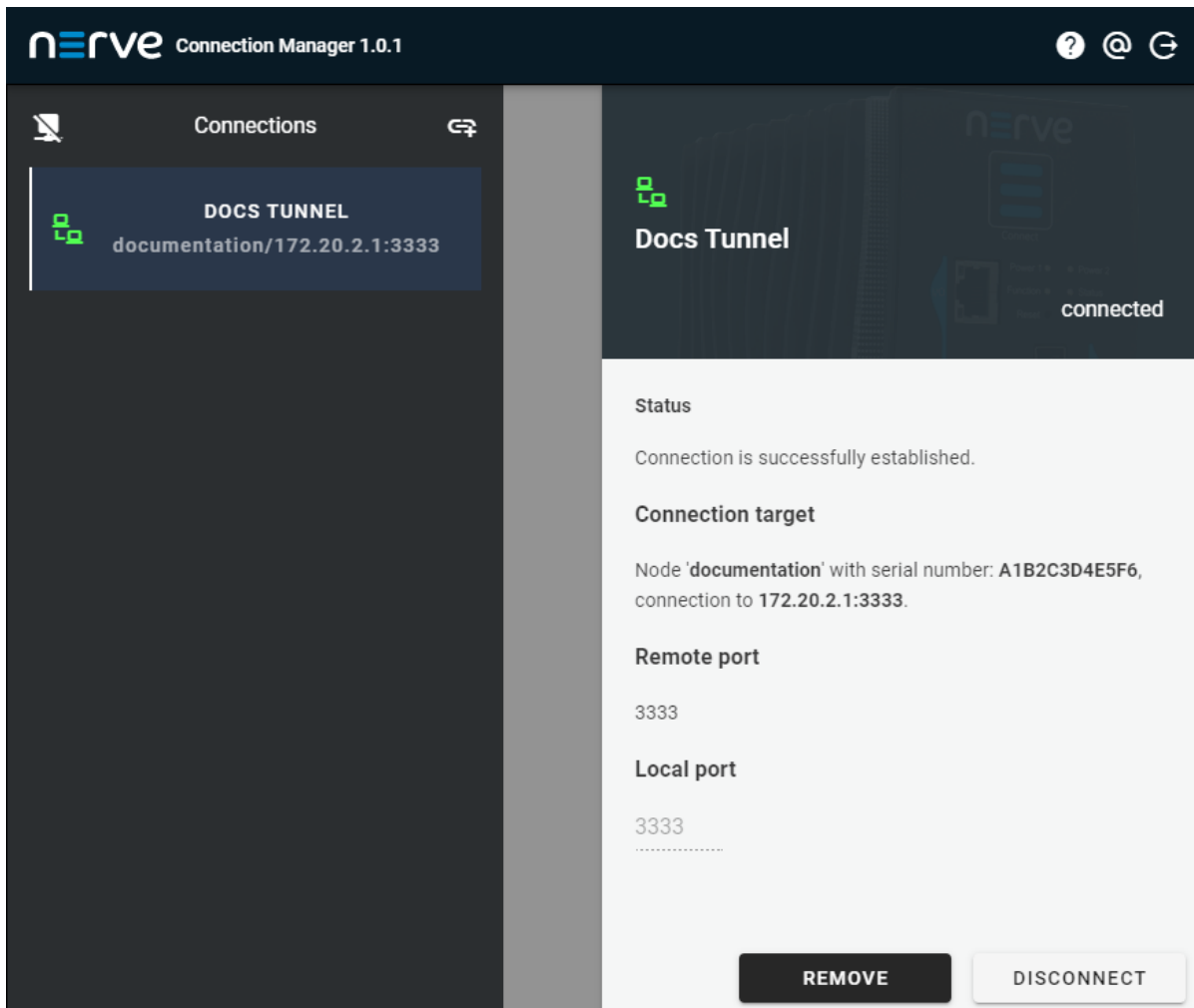


If an established connection already exists in the Nerve Connection Manager, select the **Add new connection** symbol next to **Connections** on the left side of the window.



The remote connection will be established once the Nerve Connection Manager starts.



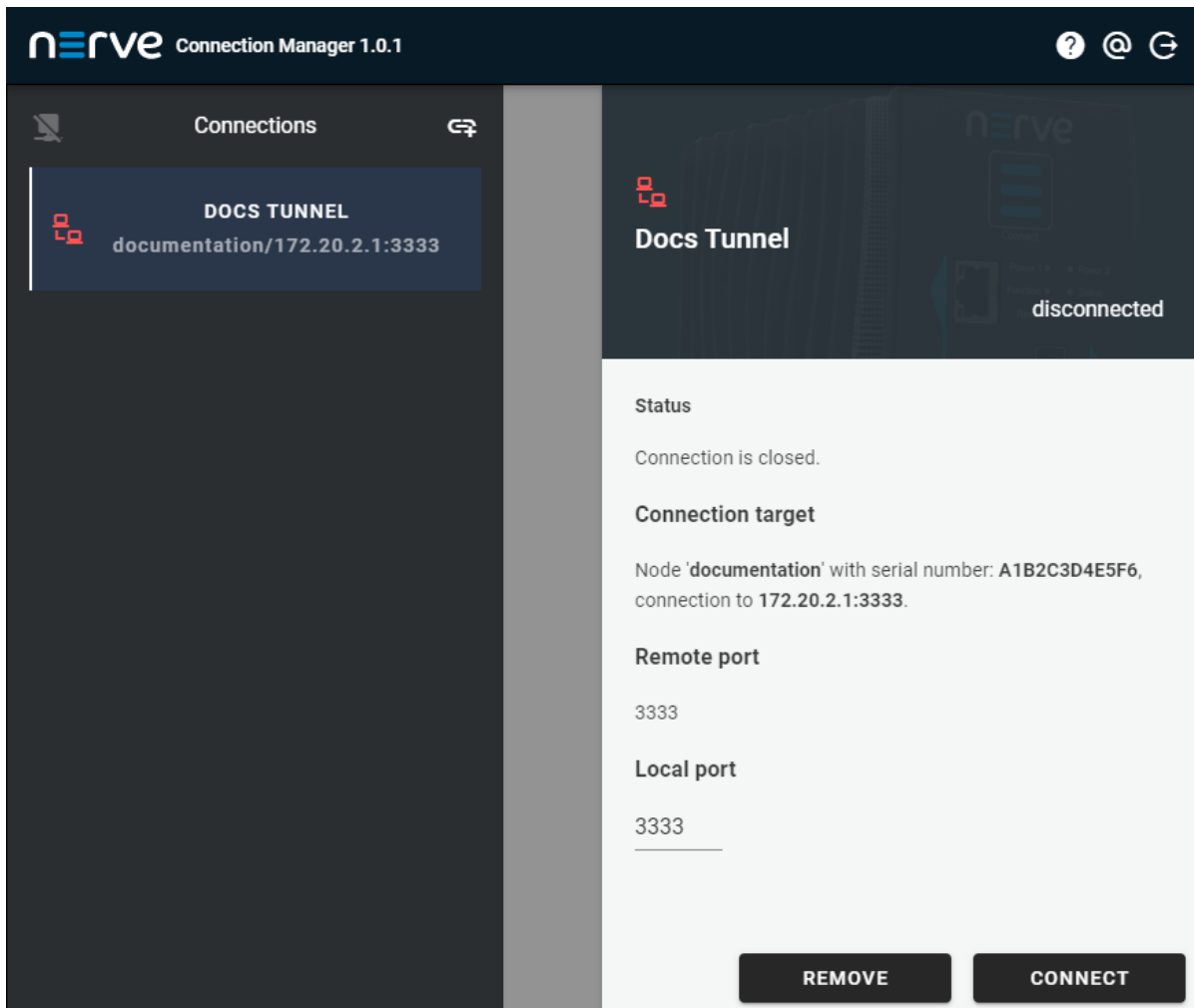


Data about the establish remote tunnel is displayed on the right half of the Nerve Connection Manager window, showing the **Status**, **Connection target**, **Remote port** and **Local port** with a summary on the left side under the remote tunnel name.

#### NOTE

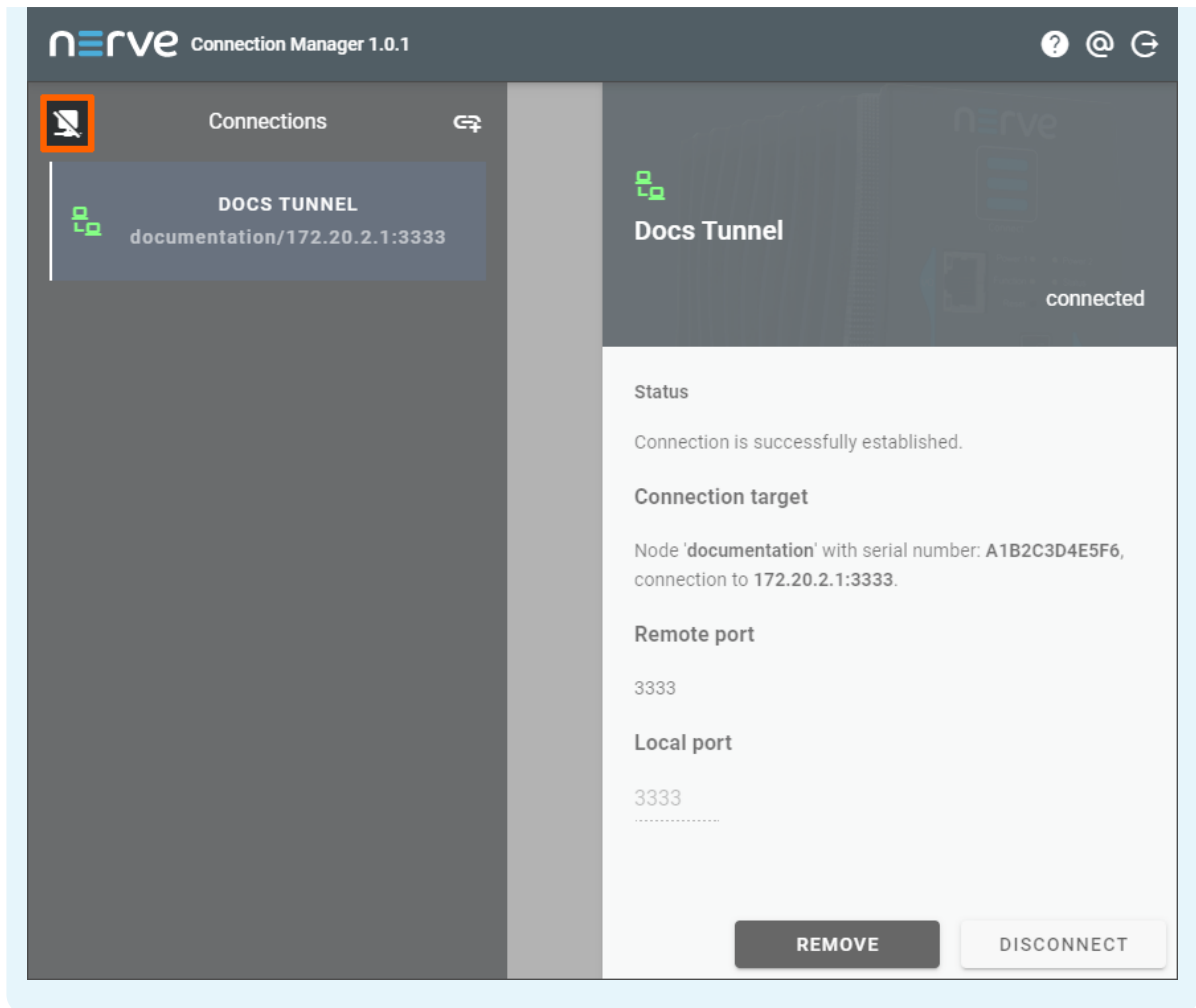
If the local port on the local workstation is already in use or occupied by the system, the Nerve Connection Manager will not establish a connection. **Local port** will be marked on the right. Enter a different port in this field that is not used on the workstation in order to establish the remote tunnel.

The connection can now be used from the local workstation by using `localhost:<localport>` through PuTTY in order to establish an SSH connection or in a web browser. Disconnect by clicking **DISCONNECT** in the lower right corner.



#### NOTE

Alternatively, all remote connections can be disconnected at once by clicking the **Disconnect all** symbol on the left side next to **Connections**.



Note that disconnecting does not terminate the connection. The connection will stay established until it is terminated in the Nerve Connection Manager, the Local UI or the Management System.

## Using a remote tunnel to a workload

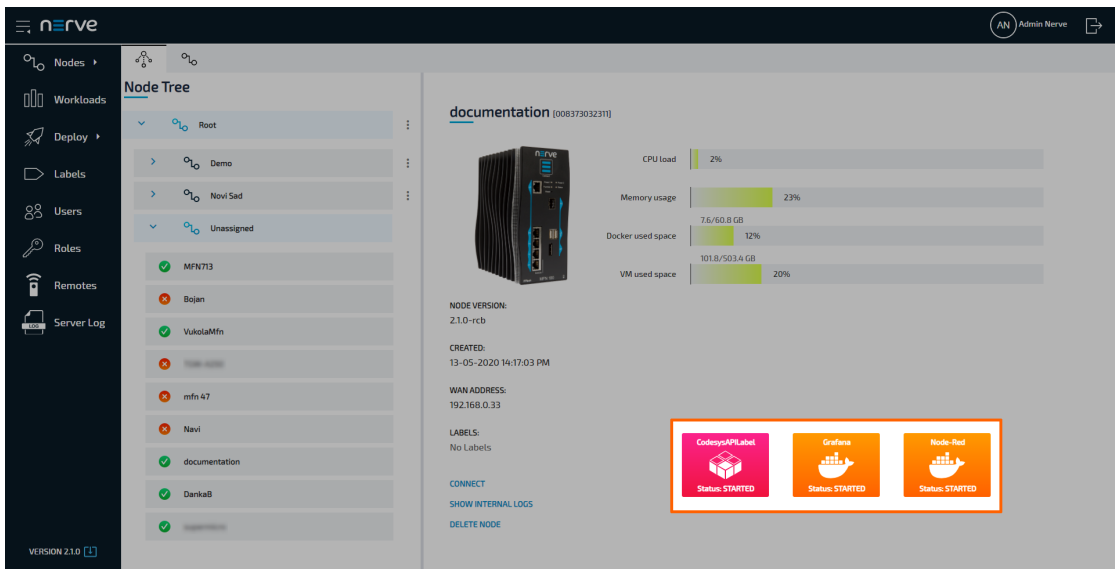
Note that the Nerve Connection Manager is required to use a remote tunnel. Download the Nerve Connection Manager from the [Nerve Software Center](#) and install it first.

Established remote tunnels are listed under **Remotes** in the navigation on the left until they are terminated.

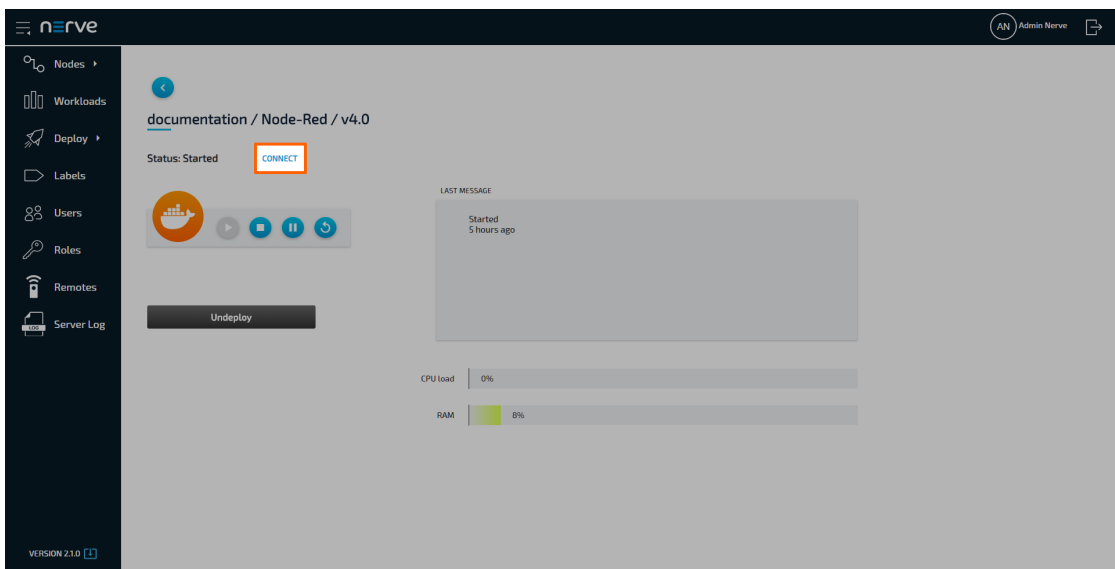
1. Select **Nodes** in the navigation on the left.
2. Select the node tree tab



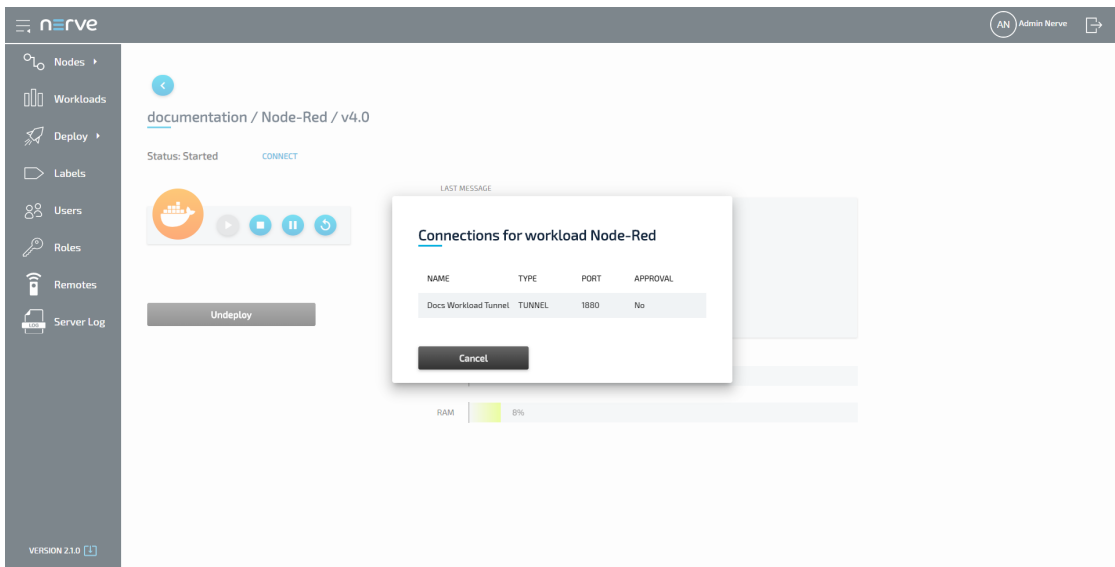
- on the right to display registered nodes in the node tree.
3. Select the node that has a deployed workload with a remote connection.
4. Select the workload.



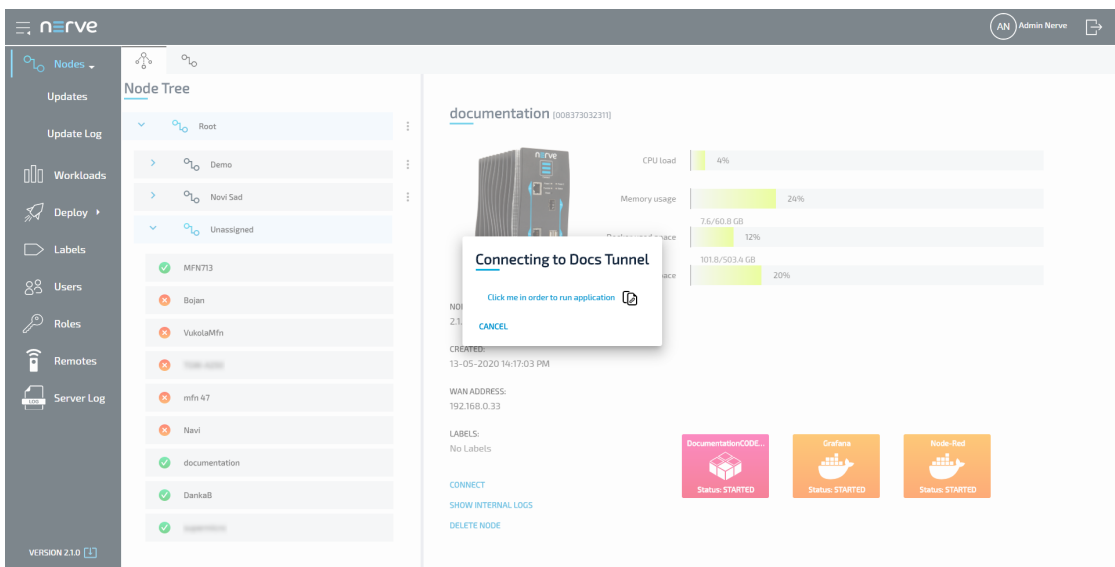
5. Click **CONNECT** next to the workload status.



6. Select the remote connection from the list in the new window. Note that remote tunnels have the type **TUNNEL**.



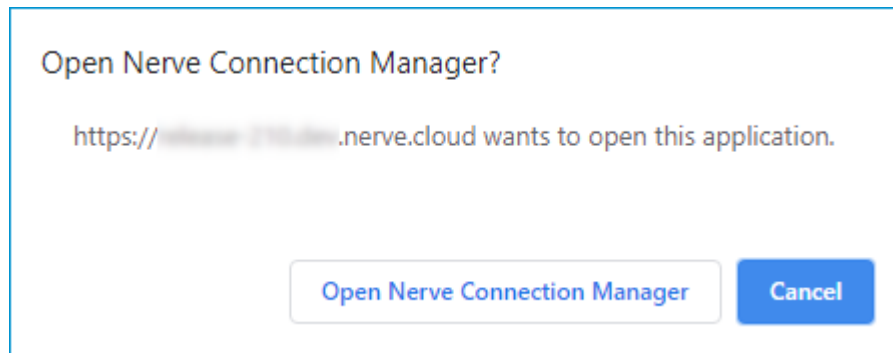
7. Select **Click me in order to run application** in the new window.



#### NOTE

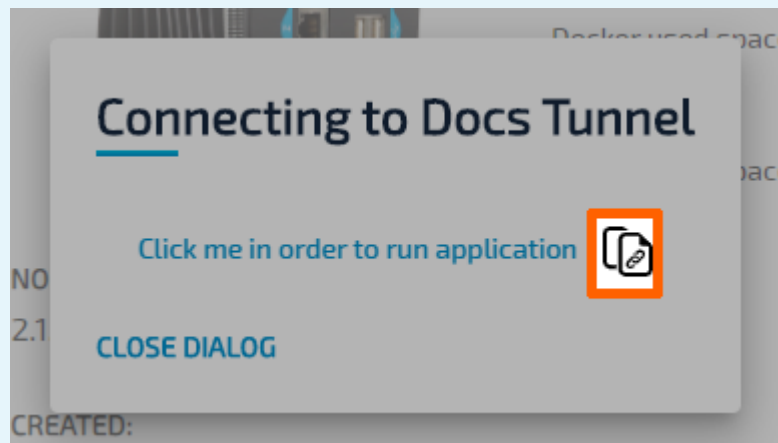
If **Local acknowledgment** is set to **Yes**, the Management System will wait for approval until the remote connection has been locally approved before displaying the window above. Refer to [Approving a remote connection](#) for more information.

8. If the Nerve Connection Manager installed correctly, confirm the browser message that the Nerve Connection Manager shall be opened. Depending on the browser that is used, this message will differ. The Nerve Connection Manager will start automatically once the message is confirmed.

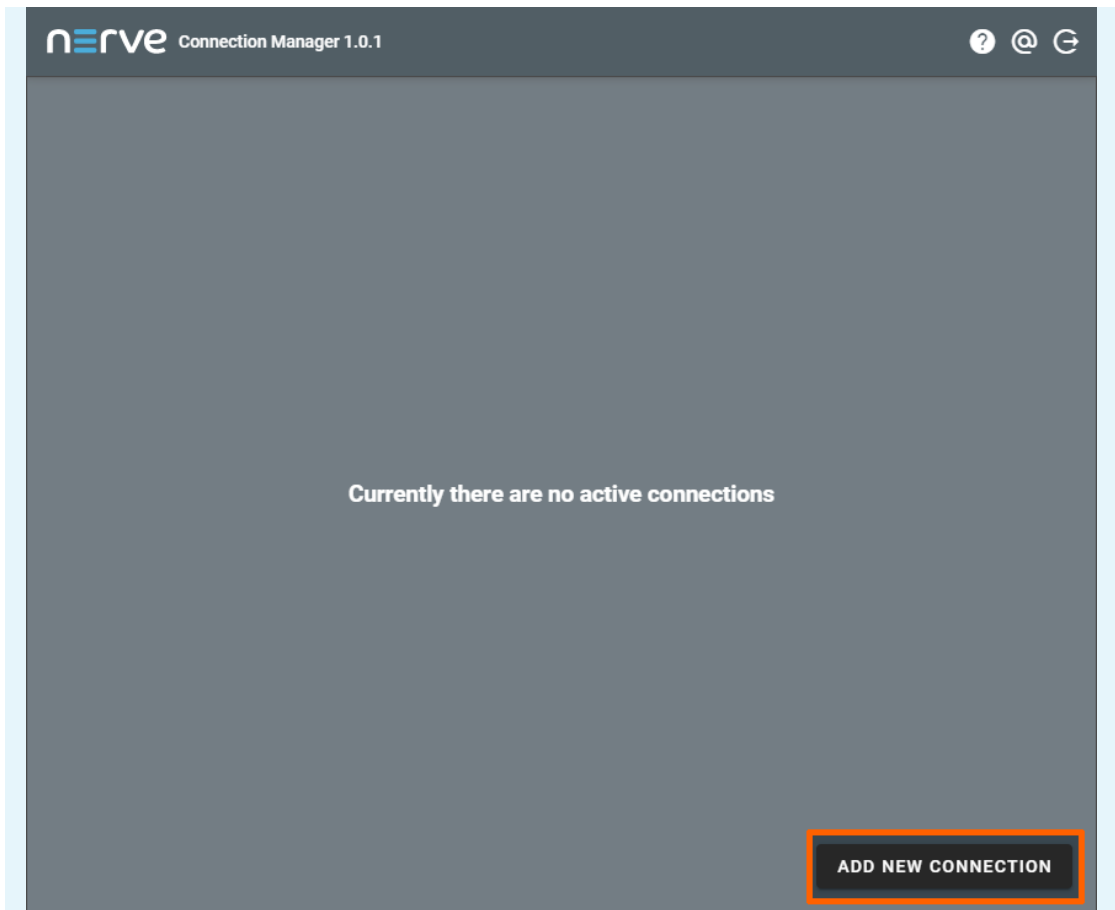


## NOTE

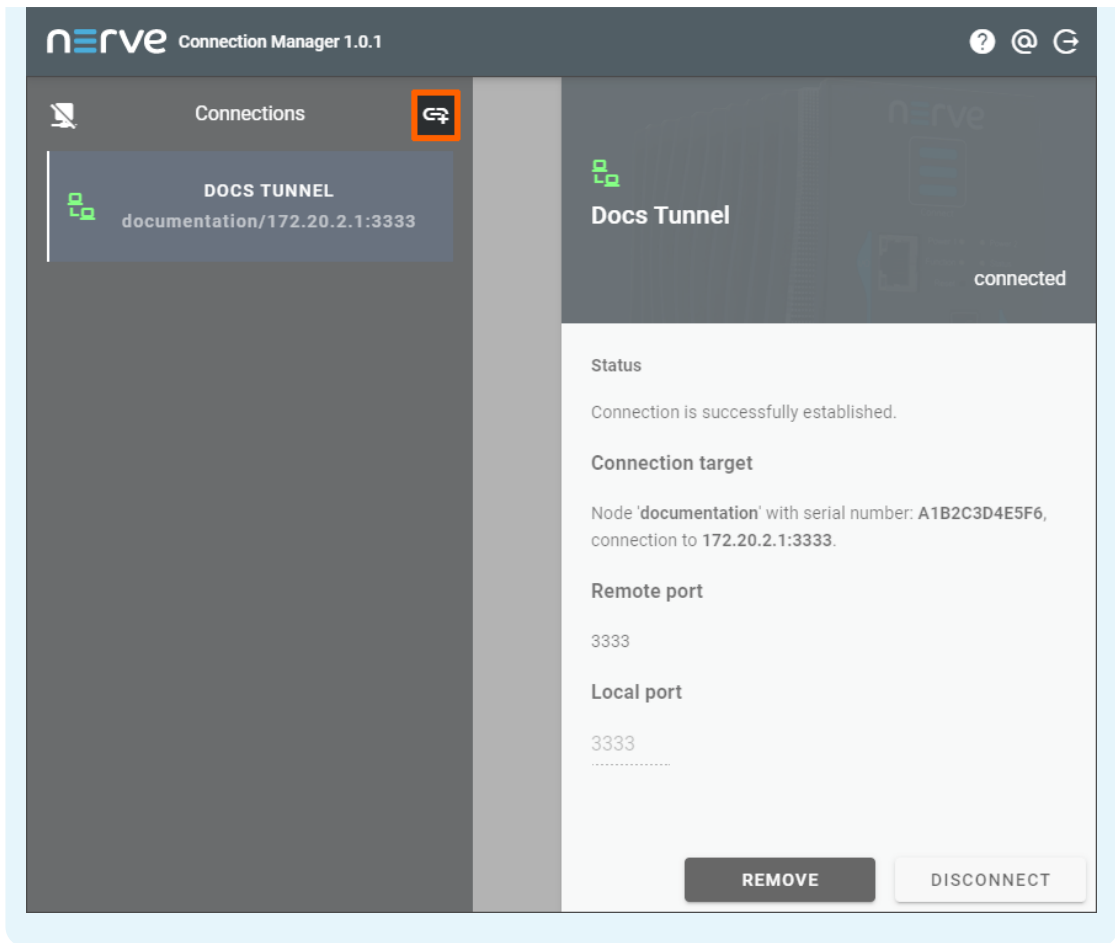
If the Nerve Connection Manager does not start automatically, select the copy to clipboard symbol next to **Click me in order to run application** in the Management System. This copies the remote connection URL.



Start the Nerve Connection Manager manually and add the new connection by clicking **ADD NEW CONNECTION** in the lower right and pasting the URL.

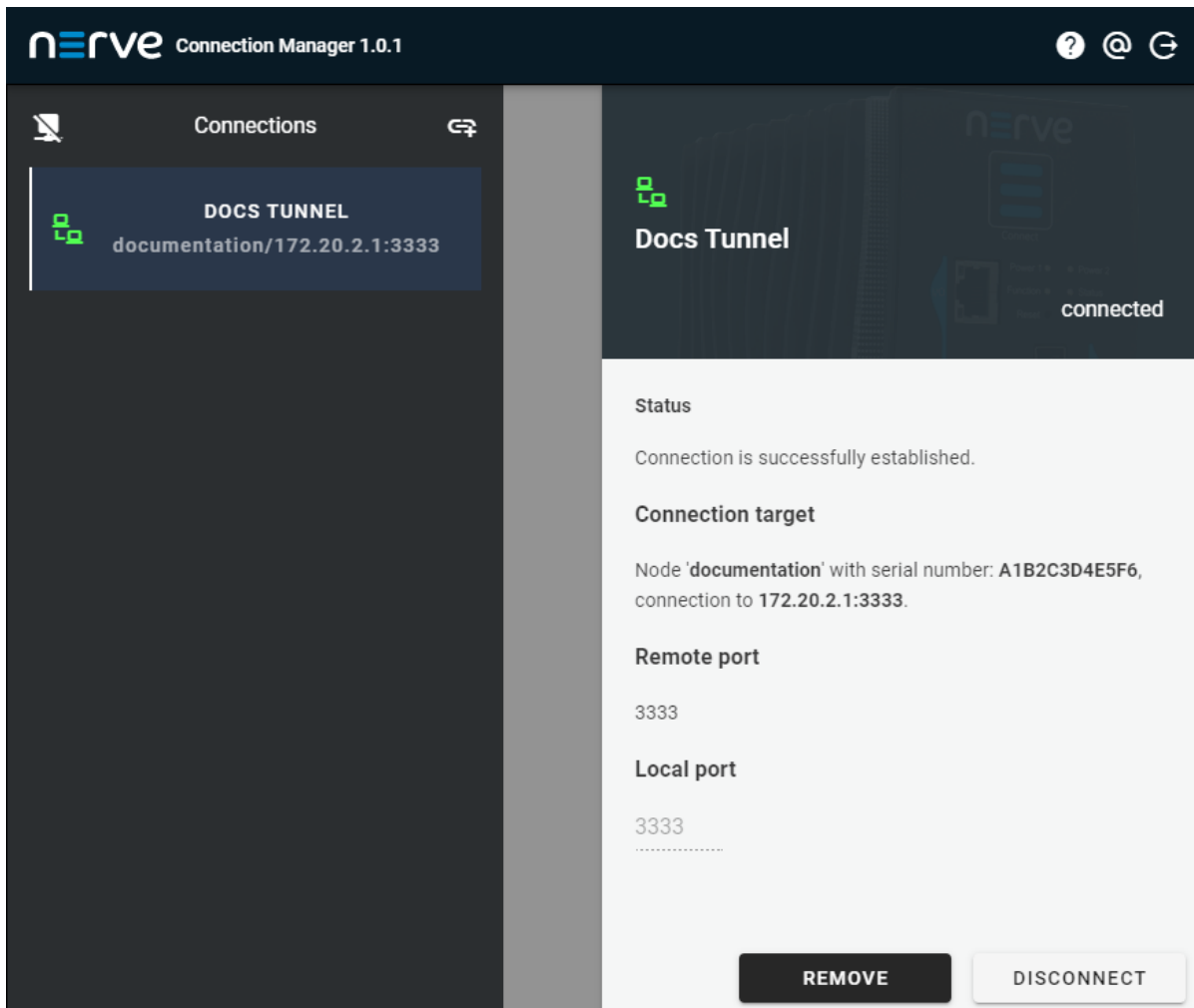


If an established connection already exists in the Nerve Connection Manager, select the **Add new connection** symbol next to **Connections** on the left side of the window.



The remote connection will be established once the Nerve Connection Manager starts.



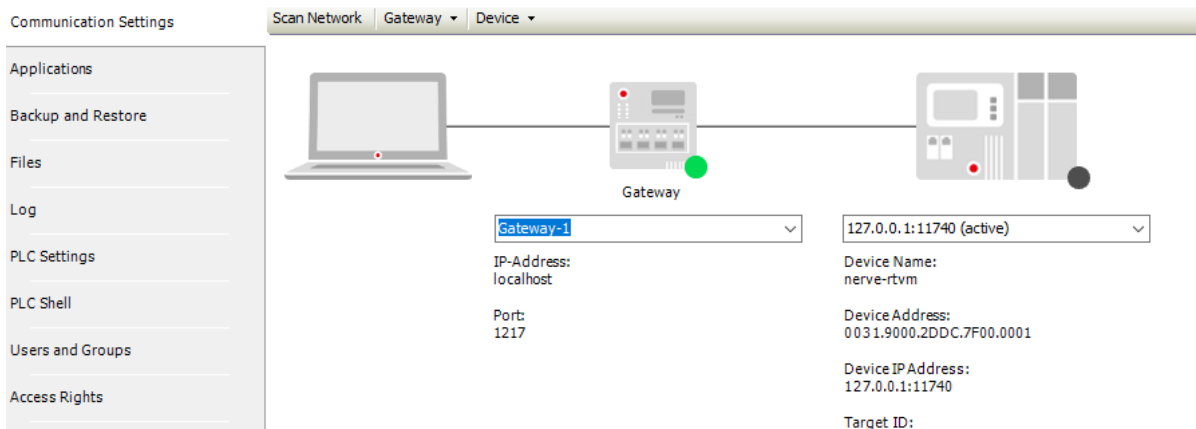


Data about the establish remote tunnel is displayed on the right half of the Nerve Connection Manager window, showing the **Status**, **Connection target**, **Remote port** and **Local port** with a summary on the left side under the remote tunnel name.

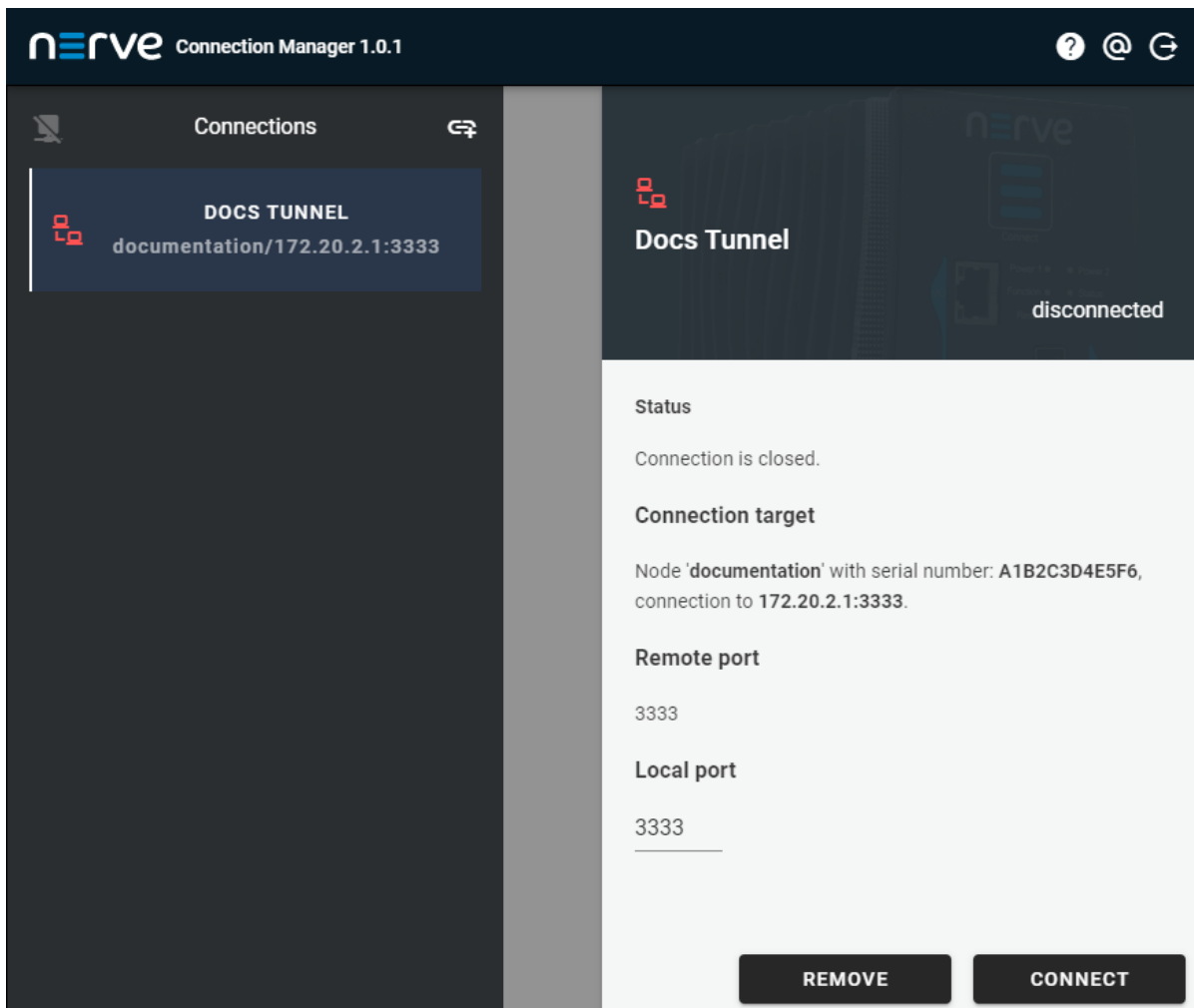
#### NOTE

If the local port on the local workstation is already in use or occupied by the system, the Nerve Connection Manager will not establish a connection. **Local port** will be marked on the right. Enter a different port in this field that is not used on the workstation in order to establish the remote tunnel.

The connection can now be used from the local workstation by using `localhost:<localport>` through PuTTY in order to establish an SSH connection or in a web browser. The screenshot below shows how to connect to a node through a remote tunnel using the CODESYS Development System. In an open project, double-click **Device (Nerve\_MFN\_100)** in the tree view on the left. Go to **Communication Settings** in the middle of the window and enter `127.0.0.1:<portonpc>` in the text box under the device on the right. Replace `<portonpc>` with the port number that was defined under **Port on PC** in the Management System.

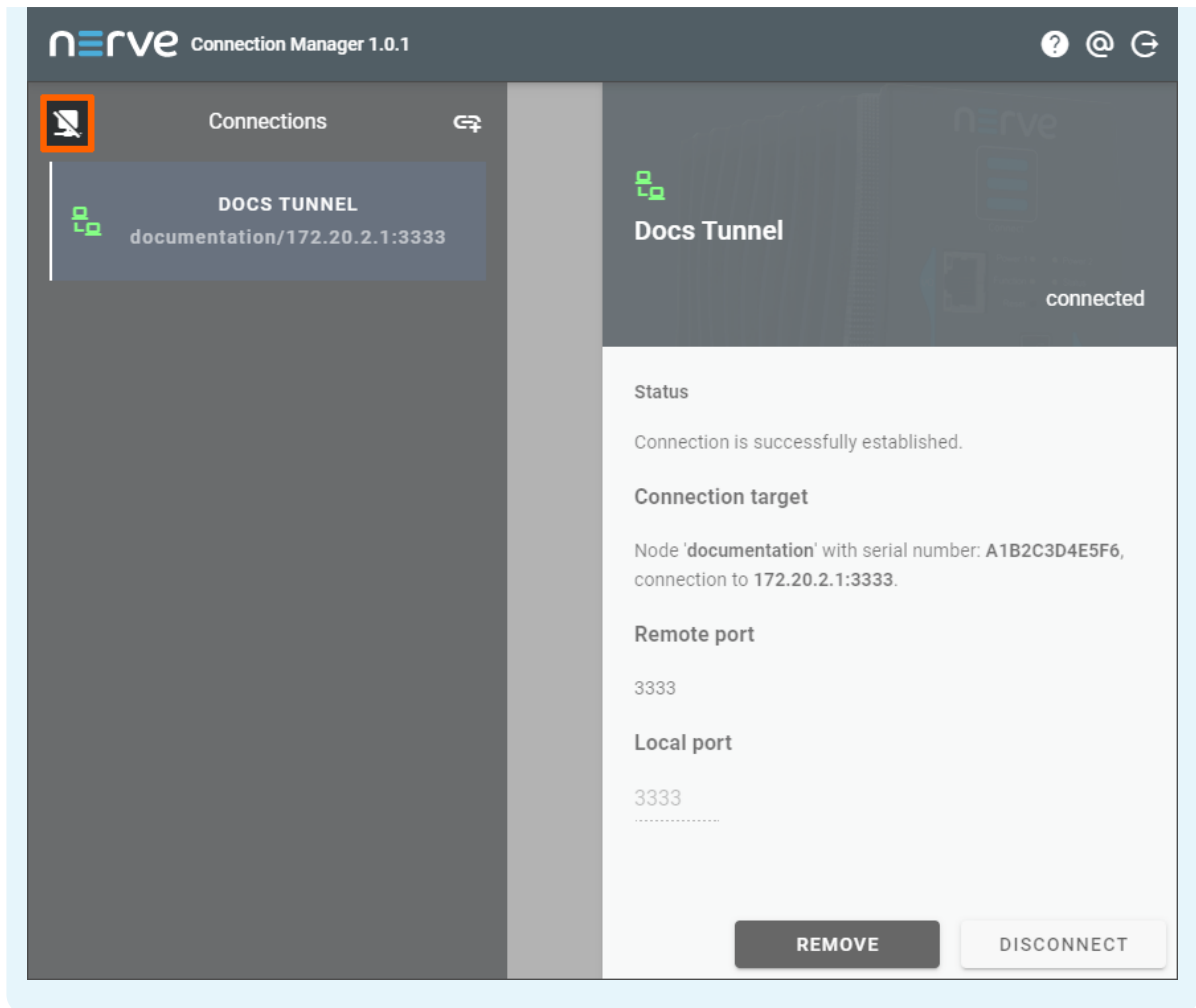


Disconnect from the remote tunnel by clicking **DISCONNECT** in the lower right corner of the Nerve Connection Manager.



## NOTE

Alternatively, all remote connections can be disconnected at once by clicking the **Disconnect all** symbol on the left side next to **Connections**.



Note that disconnecting does not terminate the connection. The connection will stay established until it is terminated in the Nerve Connection Manager, the Local UI or the Management System.

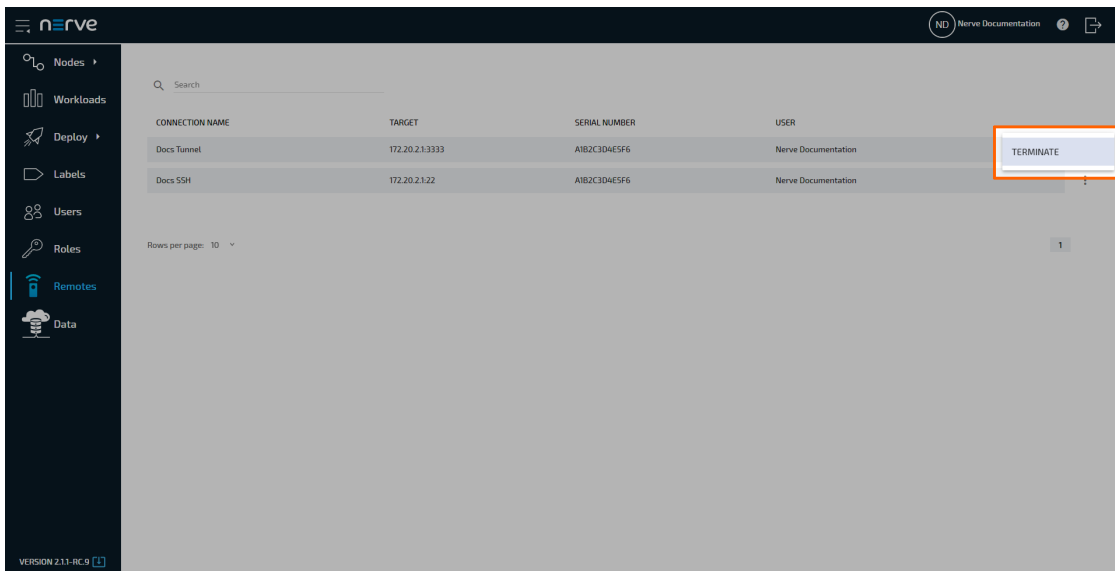
## Terminating remote connections

Remote connections are open and can be used as long as they are not terminated. A remote connection can be terminated from the Management System, in the Local UI or in the Nerve Connection Manager. Also, remote connections terminate automatically after 30 minutes of inactivity. Once a connection has been terminated, it has to be established again.

### Terminating an active remote connection in the Management System

Note that terminating an open remote connection does not remove the configuration of the remote connection from the node or workload. If a remote connection is terminated, it has to be re-established in the Management System to be used again.

1. Connect to the Management System.
2. Select **Remotes** in the navigation on the left.
3. Select the ellipsis menu to the right of an active remote connection.
4. Select **TERMINATE** in the overlay that appeared.



5. Select **OK** in the new window.

Terminating a connection in the Management System automatically removes the connection in the Management System, Local UI and Nerve Connection Manager.

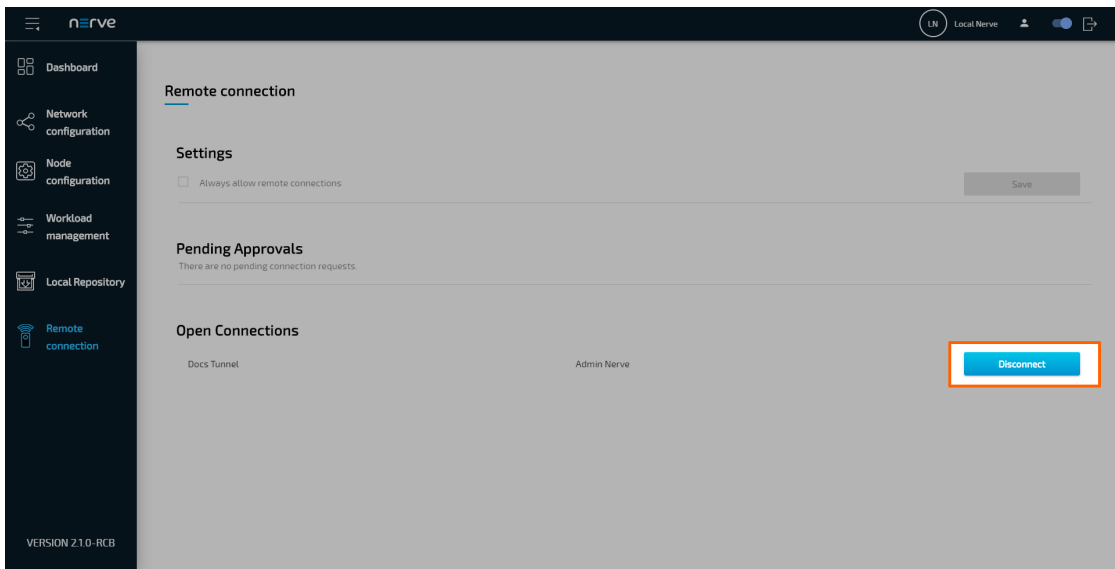
#### NOTE

Once a remote screen has been terminated while the browser tab is still open, a pop-up window will appear that offers the option to reconnect. Clicking **Reconnect** in the pop-up window has no effect. Close the window and re-established the connection in the Management System.

## Terminating an active remote connection in the Local UI

Note that terminating an open remote connection does not remove the configuration of the remote connection from the node or workload. If a remote connection is terminated, it has to be re-established in the Management System to be used again.

1. Connect to the Local UI.
2. Select **Remote Connections** in the navigation on the left.
3. Choose the remote connections that will be terminated.
4. Select **Disconnect**.



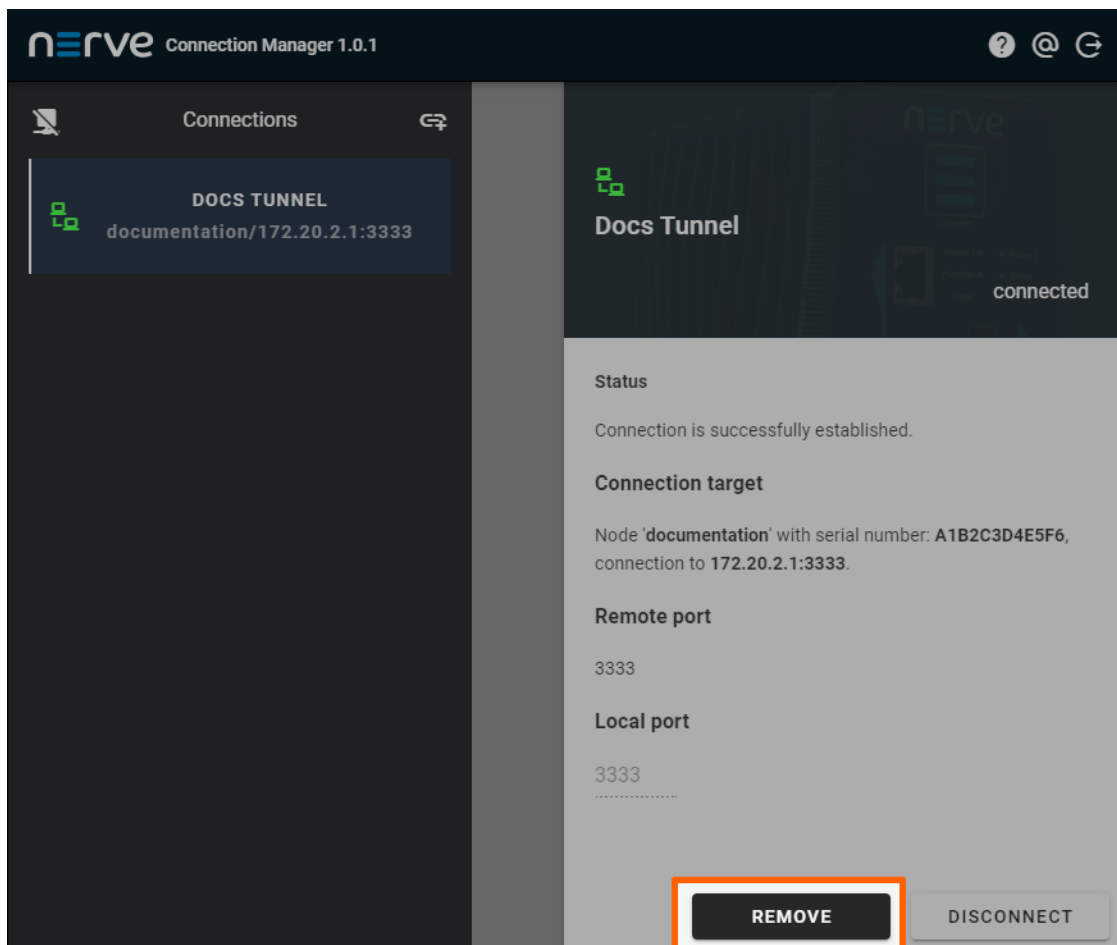
5. Select **YES** in the new window.

Terminating a connection in the Local UI automatically removes the connection in the Management System, Local UI and Nerve Connection Manager.

## Terminating a remote connection in the Nerve Connection Manager

Note that terminating an open remote connection does not remove the configuration of the remote connection from the node or workload. If a remote connection is terminated, it has to be re-established in the Management System to be used again.

1. Open the Nerve Connection Manager. Note that the Nerve Connection Manager will already be open if a remote tunnel has been established.
2. Select a remote connection that will be terminated in the list on the left.
3. Select **REMOVE** in the lower-right.



4. Select **YES** in the overlay that appeared.

Terminating a connection in the Nerve Connection Manager automatically removes the connection in the Management System, Local UI and Nerve Connection Manager. Exiting the Nerve Connection Manager terminates all remote tunnels

## Common error cases and known issues

Below is a list of most common error cases and known limitations. Hints how to avoid them or solve them the easiest way are given where applicable.

### Remote screens

- When trying to connect to a suspended workload, long loading times might occur. The connection can also seem established but the user will not be able to act in the remote screen window. The reason might be that a remote screen to a suspended workload was attempted. This is not supported.

Close the browser tab and terminate the connection in the Management System in that case. Make sure the workload is in the started state and re-establish the remote screen. If the behavior persists, investigate the workload settings or the node.

- Remote screens to workloads will be shown as active under **Remotes** if the workload is undeployed while the remote screen is being used.

## Remote tunnels

- Using two remote tunnels to two nodes, accessing the Local UI of each node at the same time is not possible. This is due to authentication conflicts.

Use the incognito mode of the current browser for the second tab or a second browser if both Local UIs have to be operated at the same time.

- Some systems might restrict usage of local ports lower than 1024. This is true for Linux systems especially. Enter ports higher than 1024 under **Port on PC** when configuring a remote tunnel to avoid port conflicts.

## Labels

Labels are a useful feature that help with the organization of nodes and workloads. They can be defined and used freely. All labels that have been created are listed in the labels menu.

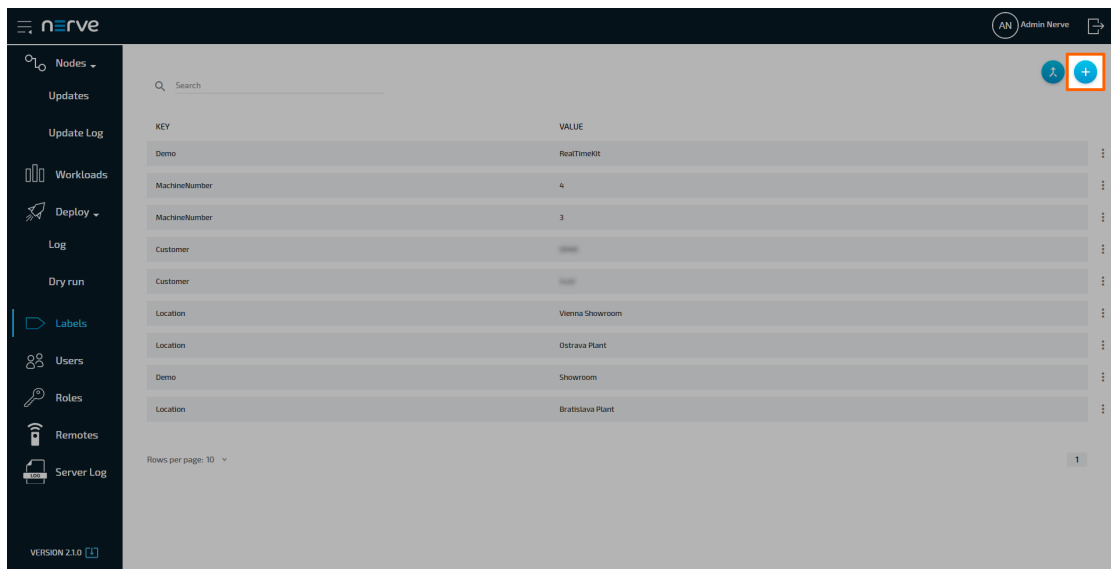
The screenshot shows the Admin Nerve interface. The sidebar on the left contains navigation icons for Nodes, Updates, Workloads, Deploy, Log, Dry run, Labels (highlighted), Users, Roles, Remotes, and Server Log. The main area features a search bar (1) and a table of labels. The table has two columns: KEY (4) and VALUE (5). The labels listed are: Demo (RealTimeKit), MachineNumber (4), MachineNumber (3), Customer (redacted), Customer (redacted), Location (Vienna Showroom), Location (Ostrava Plant), Demo (Showroom), and Location (Bratislava Plant). A vertical ellipsis menu (6) is on the right side of the table. At the top right, there are buttons for merge (2) and add new label (3). The footer shows 'VERSION 2.1.0'.

Item	Description
<b>Search bar (1)</b>	Use the search bar to filter labels by key.
<b>Merge labels (2)</b>	Clicking here leads to a page that allows to merge existing labels.
<b>Add new label (3)</b>	Click here to add a new label consisting of label key and label value.
<b>KEY (4)</b>	This is the key of the label. It can be understood as the "category" of the label. Examples of label keys are location, machine number or hardware.
<b>VALUE (5)</b>	This is the value of the label. It corresponds to the key of the label. Examples are Vienna, Machine 1 or MFN 100.
<b>Ellipsis menu (6)</b>	Clicking here opens an overlay that allows the deletion of labels.

## Adding a new label

Labels can be defined with any key and value. Note that labels are displayed in the <key>:<value> format in the Management System. The key of a label can be understood as the category of a label with the value being an item in that category. Example: Location:Vienna.

1. Select **Labels** in the menu on the left side.
2. Click the **Add new label** icon in the upper-right corner.



3. Enter the following information:

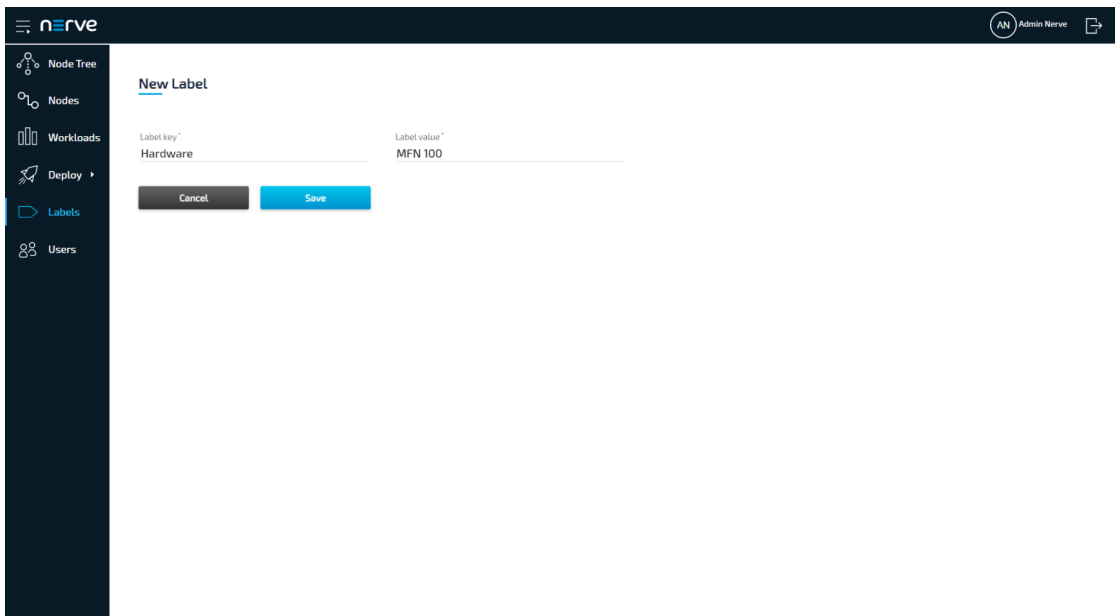
Item	Description
<b>Label key</b>	Enter the "category" of the label. Example: <b>Hardware</b> .
<b>Label value</b>	Enter the value for the label here. Example: <b>MFN 100</b> .

### NOTE

Label keys must consist of one word only. Use - and \_ as separators. Also, only use alphanumeric characters (a-z, A-Z, 0-9) and underscore (\_). Any other special characters are not allowed.

4. Click **Save** to add the new label.



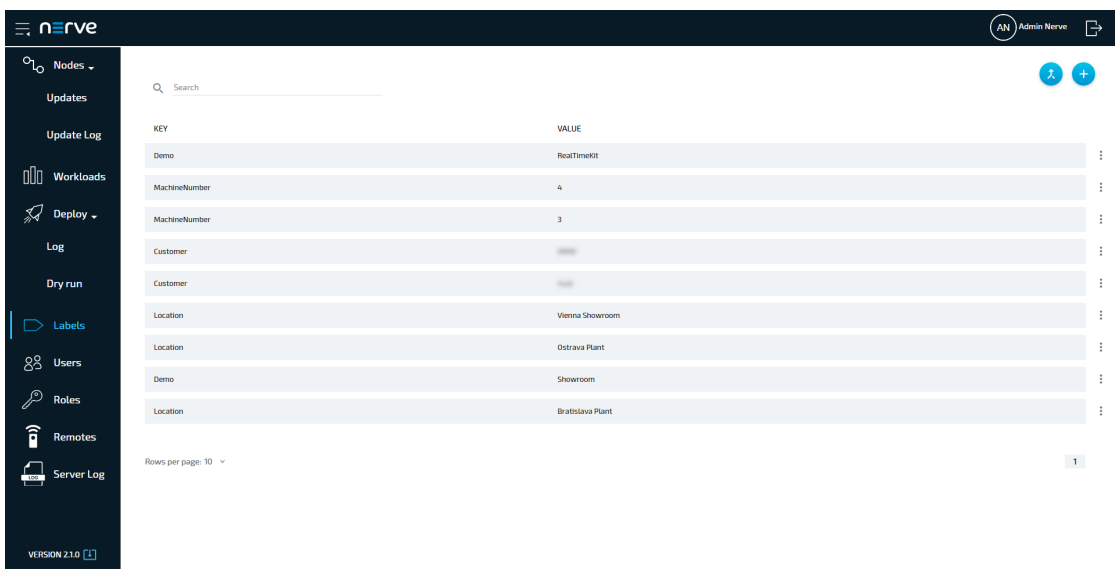


The label now appears in the label list and can be used in the Management System. Labels can be assigned to nodes when node details are edited. In the workload provisioning process they can be chosen as selectors.

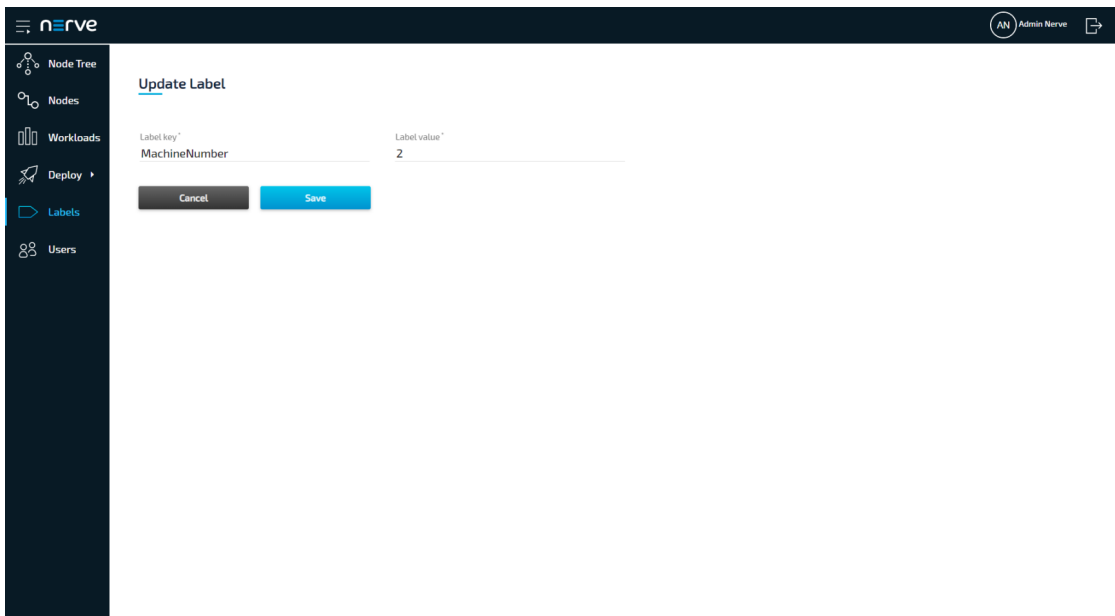
## Editing a label

Editing a label is virtually identical to the process of creating a new label. Note that labels will be edited even if they are currently assigned to nodes or used as selectors by workloads.

1. Select **Labels** in the menu on the left side.
2. Select a label to edit.



3. Edit **Label key** and **Label value**.



#### NOTE

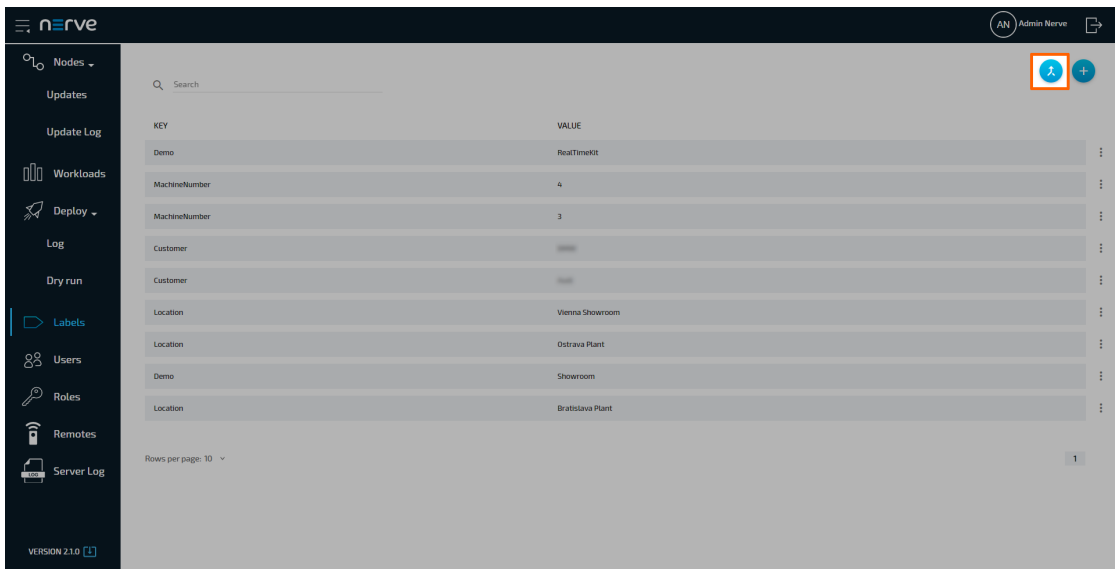
Label keys must consist of one word only. Use - and \_ as separators. Also, only use alphanumeric characters (a-z, A-Z, 0-9) and underscore (\_). Any other special characters are not allowed.

4. Click **Save** to update the label.

## Merging labels

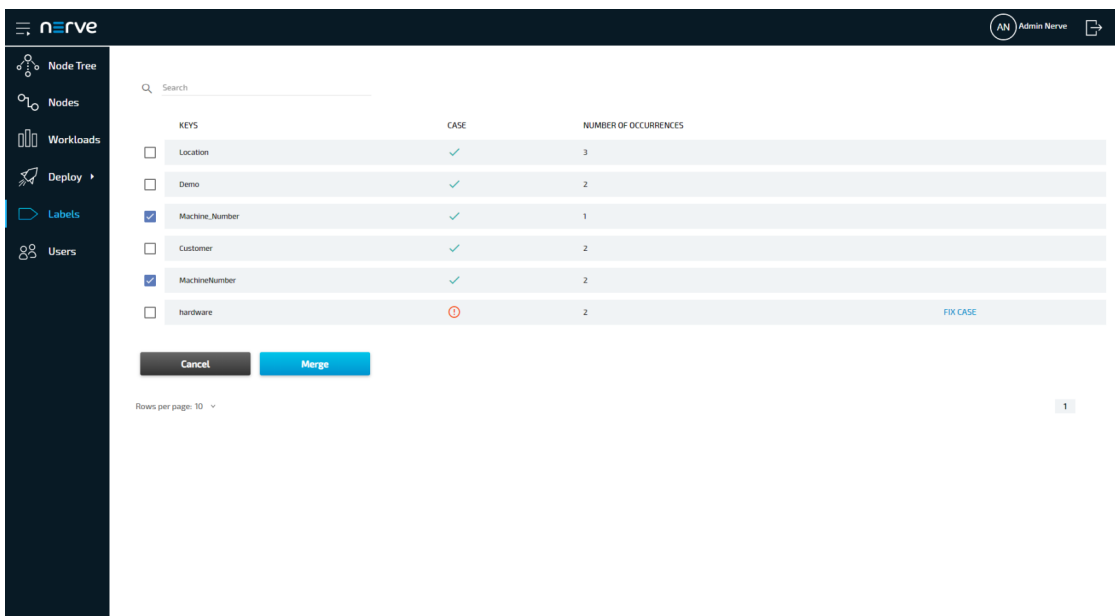
Labels with overlapping information or typos can be merged into one. However, only label keys are merged. The label values are left untouched and assigned to the new label key.

1. Select **Labels** in the menu on the left side.
2. Click the **Merge labels** icon in the upper-right corner.



3. Tick the checkboxes left of the labels that will be merged. Multiple labels can be selected.

4. Select **Merge**.

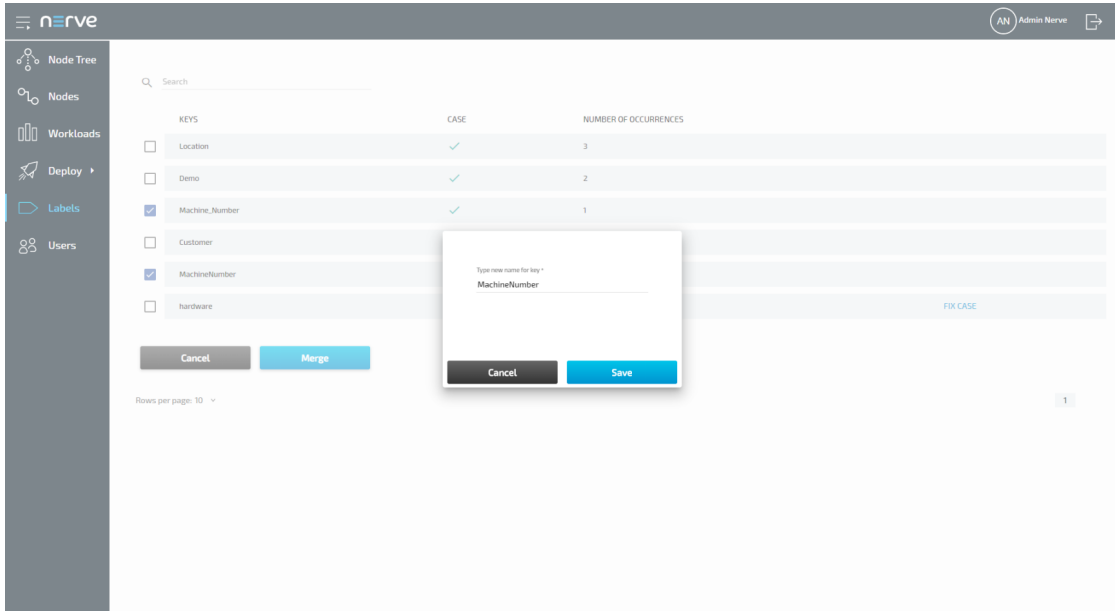


## NOTE

The table here gives the following information:

- CASE**  
 The system checks if labels that use the same characters are also written in the same case. A green check mark indicates that the case matches. A circled red exclamation mark appears if the case does not match.
- NUMBER OF OCCURRENCES**  
 The number here indicates how many labels have been defined with the same label key.

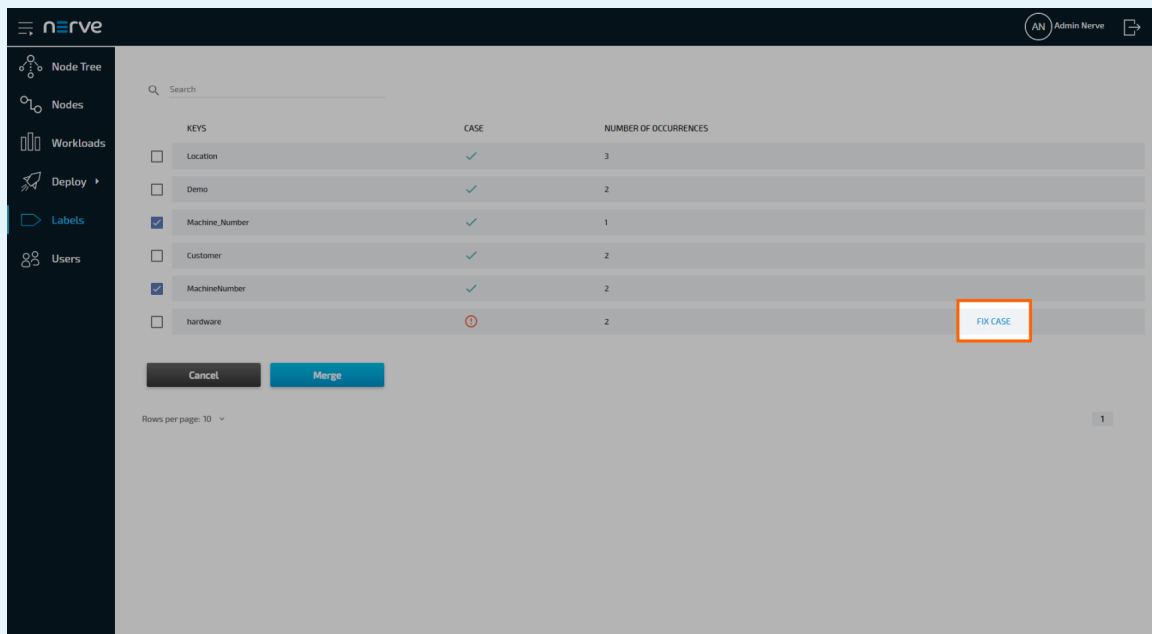
5. Type in the new name for the label key.



6. Click **Save** to save the new label.

## NOTE

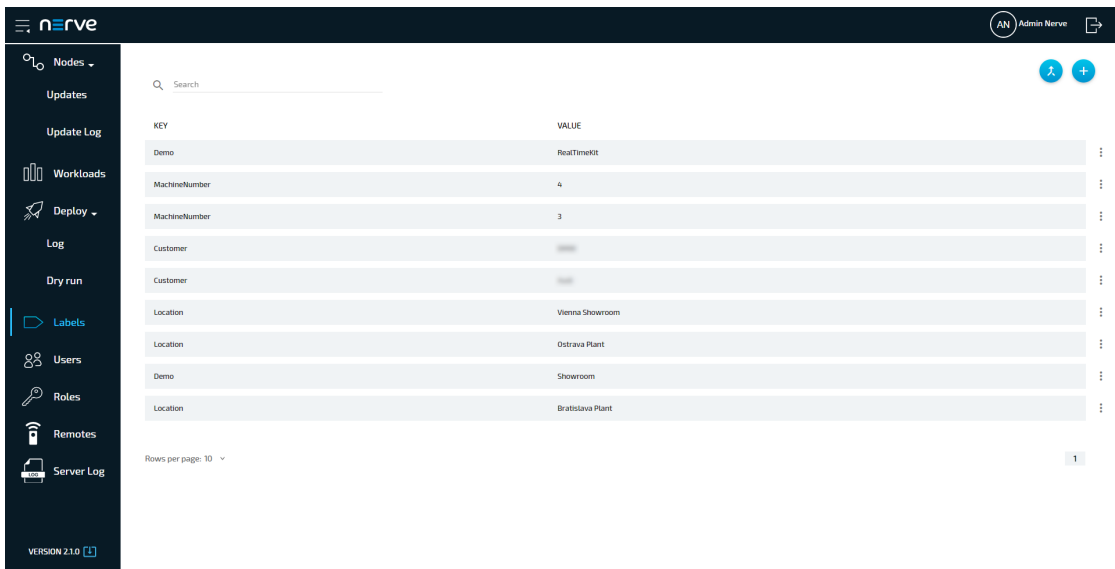
If labels have the same label key but are not written in the same case, the Management System will recognize it. In this case, select **FIX CASE** in the list, type in the new name for the label key and select **Save**.



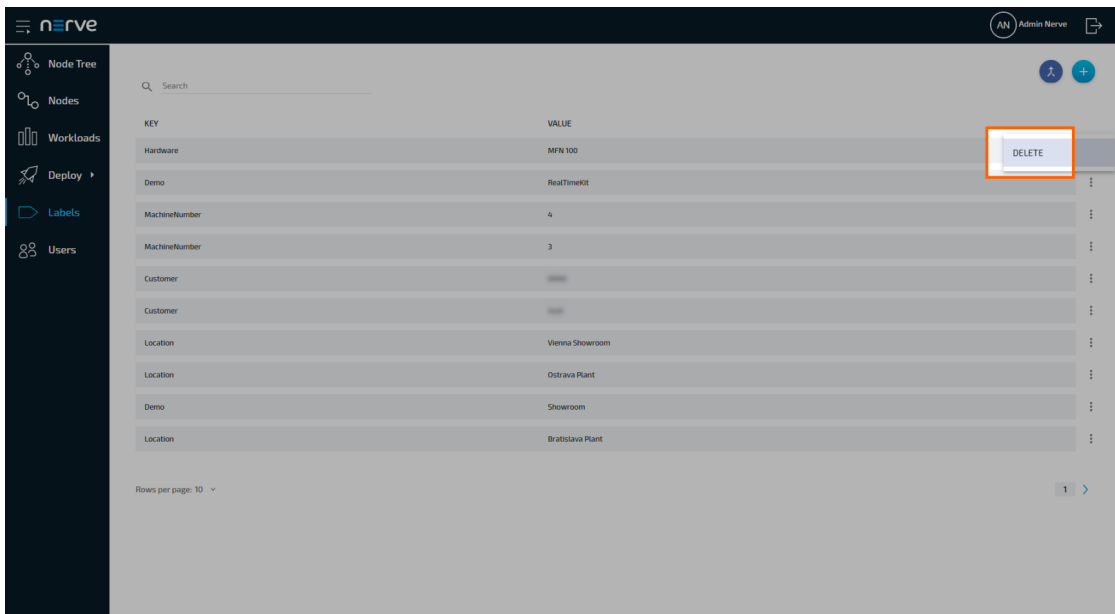
# Deleting a label

Note that when a label is deleted from the list, it will also be deleted from any node or workload that had the label assigned.

1. Select **Labels** in the menu on the left side.
2. Choose a label to delete.



3. Click the ellipsis menu on the right side of the label.
4. Select **Delete** in the overlay that appeared.



5. Click **OK** to delete the label.

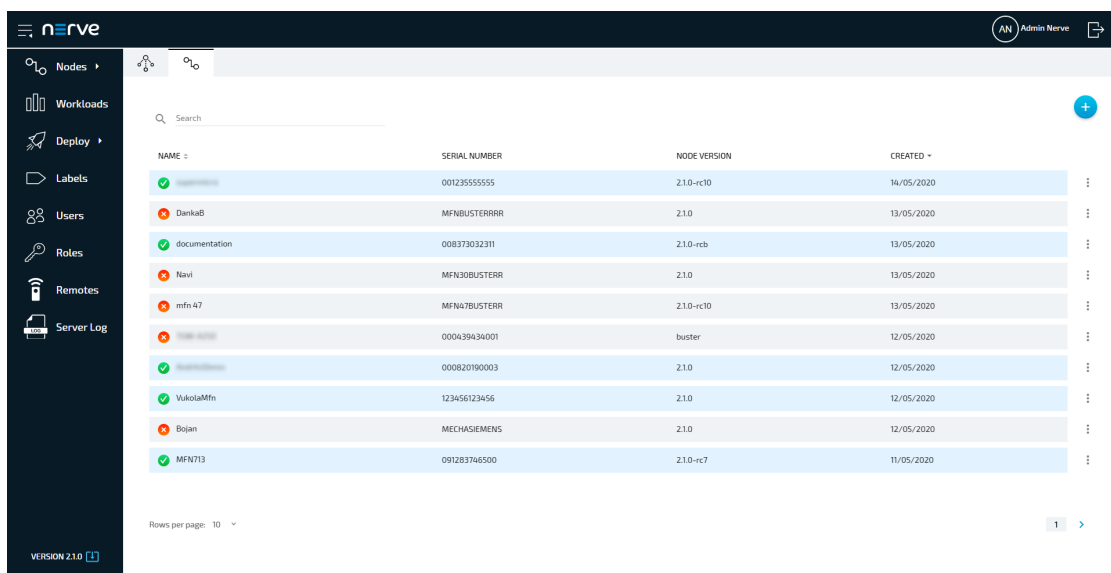
## Adding a label to a node

Adding a label to a node is done in the node menu. It is recommended to create labels before they are added to nodes.

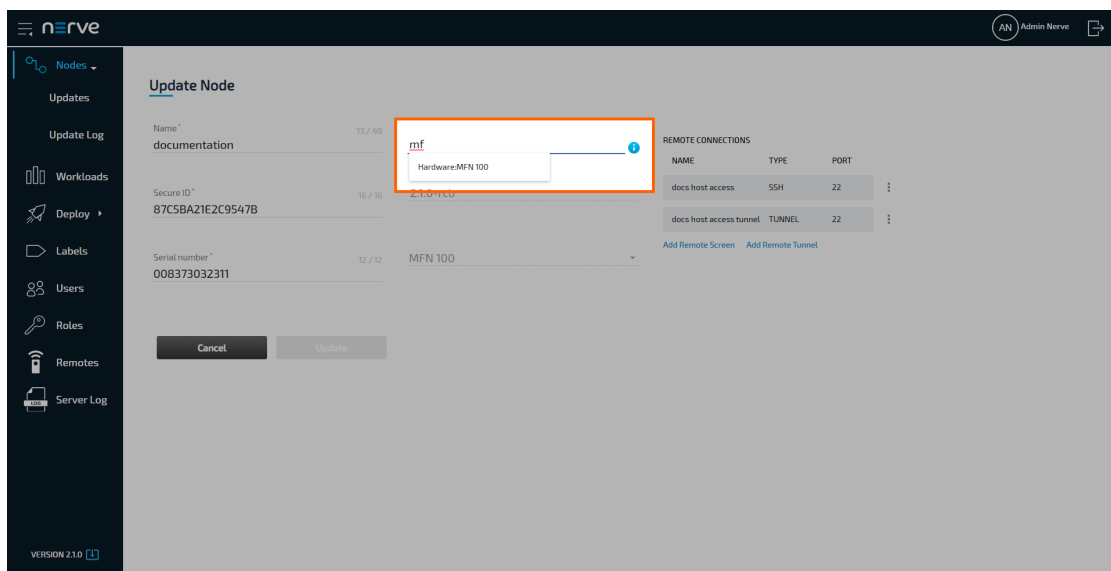
1. Select **Nodes** in the navigation on the left.
2. Select the nodes tab

on the right to display the list of registered nodes.

3. Select a node to which a label will be assigned.



4. Start typing in the **Insert label** field. Available labels will be displayed if they match the typed string.

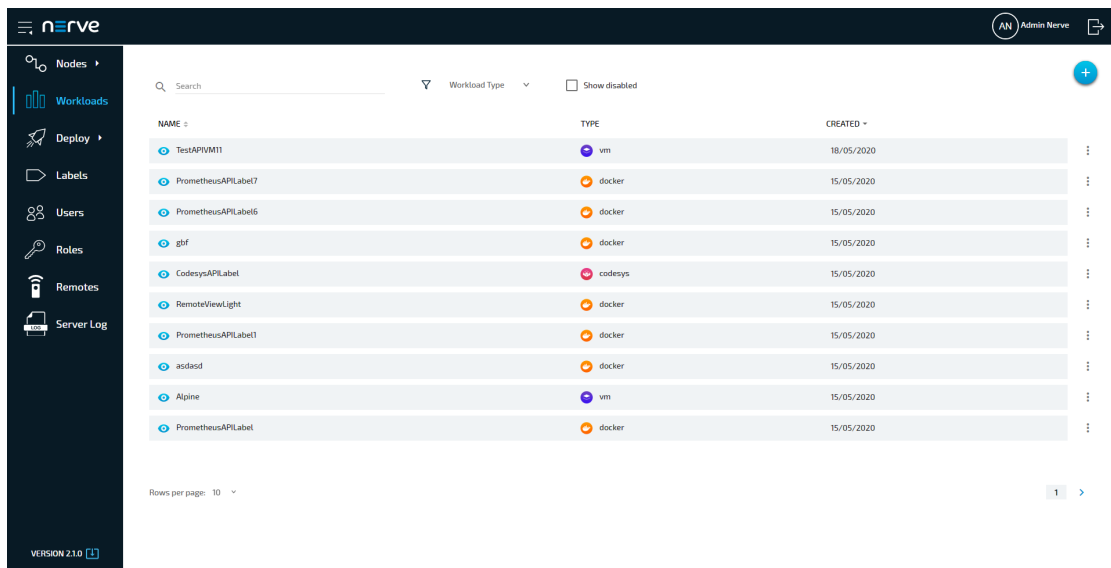


5. Select a label from the suggestions.
6. Select **Update** to save the changes to the node.

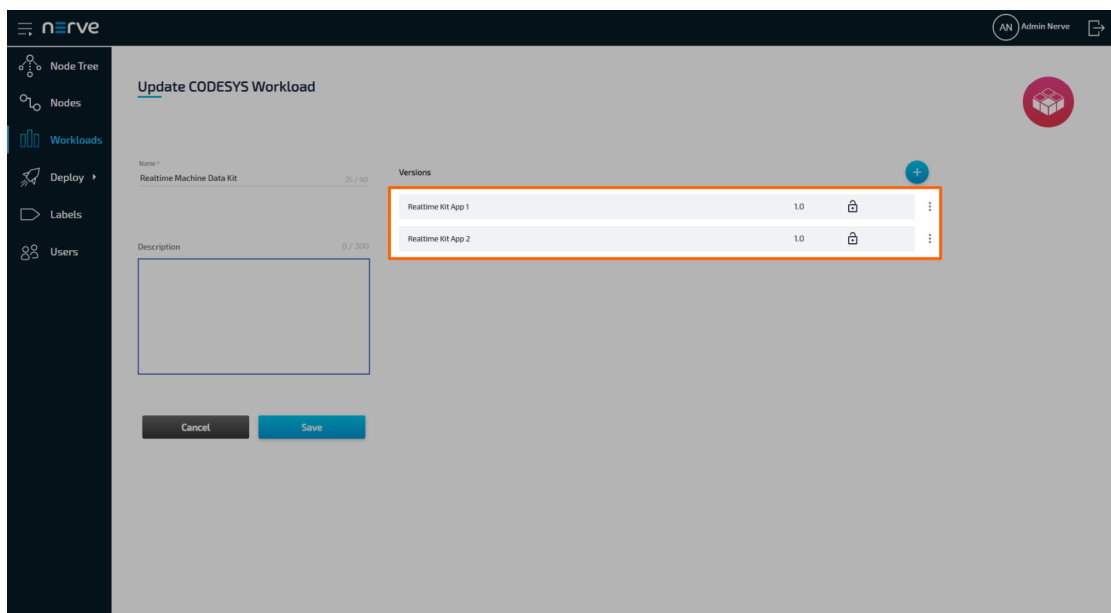
## Adding a label to a workload

Adding a label to a workload is done in the workloads menu. It is recommended to create labels before they are added to workloads. Note that only pre-defined labels can be added to workloads.

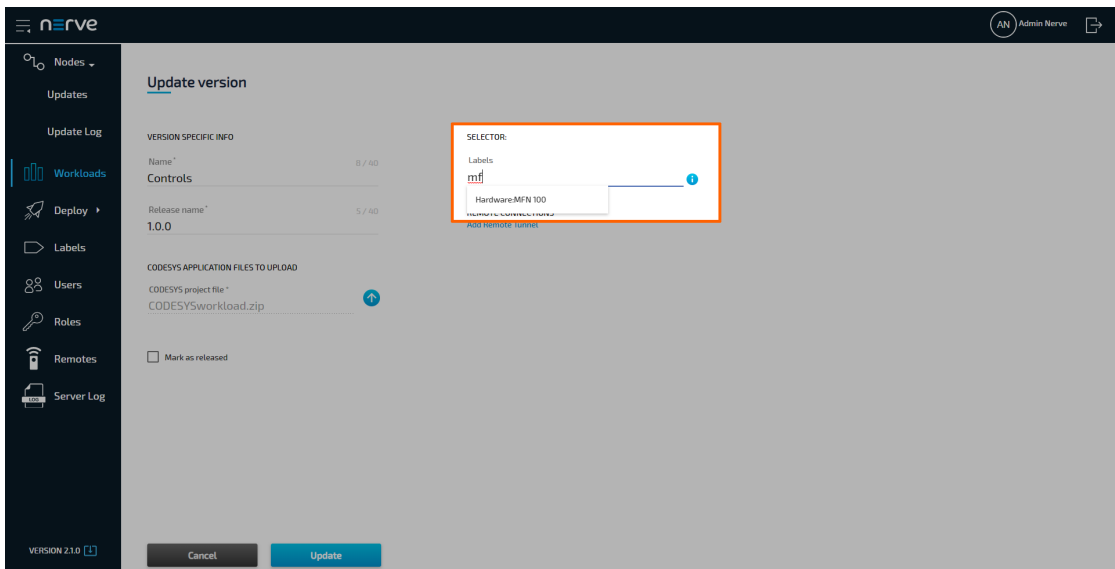
1. Select **Workloads** in the navigation on the left.
2. Select a workload to which a label will be assigned.



3. Select the workload version to which a label will be assigned.



4. Start typing in the **Insert label** field. Available labels will be displayed if they match the typed string.



5. Select a label from the suggestions.
6. Select **Update** to save the changes to the node.

## Users

### NOTE

This page describes the standard user management in the Nerve Management System. When LDAP authentication is active, user attributes and details are carried over from an LDAP server. Refer to [LDAP](#) for more information on LDAP synchronization.

This menu offers a list of all registered users. Every user has their own user profile with details about the user. There is one default user that is created with the Management System. The credentials for this user can be found in the customer profile.

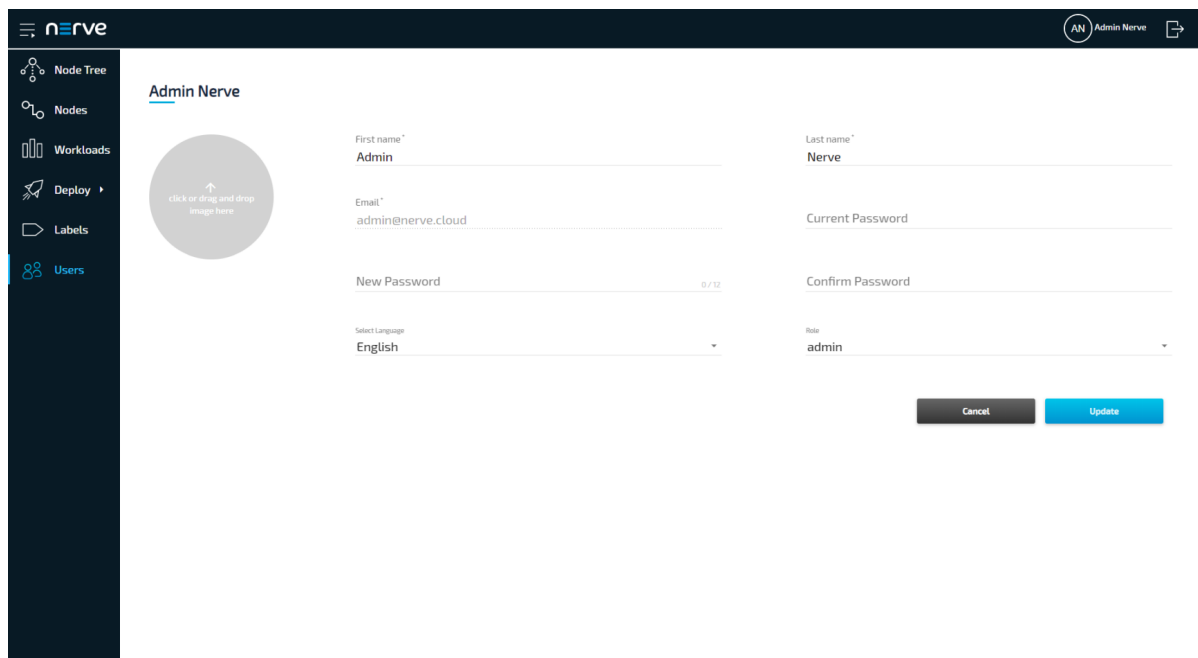


FIRST NAME	LAST NAME	EMAIL	TYPE	CREATED	USER ROLES
...	...	...	local	09/02/2021	Admin
...	...	...	local	05/02/2021	Admin, log
User	Nerve	...	local	05/02/2021	Admin
...	...	...	local	04/02/2021	Admin
...	...	...	local	04/02/2021	Admin
Nerve	Documentation	...	local	04/02/2021	Admin
...	...	...	local	04/02/2021	Admin
...	...	...	local	04/02/2021	Admin
...	...	...	local	04/02/2021	Admin
...	...	...	local	04/02/2021	Admin

Item	Description
<b>Search bar (1)</b>	Use the search bar to filter the list of users. The columns <b>FIRST NAME</b> , <b>LAST NAME</b> and <b>EMAIL</b> are the targets of the search.
<b>Type filter (2)</b>	Filter the list of users by authentication method. The target of the filter is the <b>TYPE</b> column and the options are <b>Local</b> , <b>LDAP</b> and <b>All</b> .
<b>Add new user (3)</b>	Select this icon to add a new user. Note that new users cannot be added when LDAP synchronization is active.
<b>FIRST NAME (4)</b>	This is the first name of the registered user. It is displayed in the upper-right corner when the user is logged in.
<b>LAST NAME (5)</b>	This is the last name of the registered user. It is displayed in the upper-right corner when the user is logged in.
<b>EMAIL (6)</b>	This is the e-mail address of the user. It is used as the username for logging in to the Management System in case of local authentication. When LDAP authentication is active, the username attribute that is carried over from LDAP synchronization is used to log in. The system also uses this e-mail for sending the activation link and the instructions on how to reset the login password. Note that passwords cannot be changed or reset if LDAP authentication is active.
<b>TYPE (7)</b>	This column shows the type of the user according to the authentication method used: <ul style="list-style-type: none"> <li>• <b>Local</b> This describes users that are created locally through the Management System. They authenticate against the Management System itself when logging in.</li> <li>• <b>LDAP</b> This describes users that are created following an LDAP synchronization. They authenticate against the defined LDAP server when logging in.</li> </ul>
<b>CREATED (8)</b>	This is the date the user was created. The date format is DD/MM/YYYY.

Item	Description
<b>USER ROLES (9)</b>	The roles that are assigned to a user are displayed in this column.
<b>Ellipsis menu (10)</b>	Clicking here opens an overlay that allows the deletion of users.

Clicking any of the users leads to their user profile. An admin can edit the details of any user profile if local authentication is active. This includes **First name**, **Last name**, **Select language** and **Role**. The e-mail address of a user cannot be changed. The password for an account can only be changed by the respective user. Note that users cannot be edited when LDAP authentication is active.



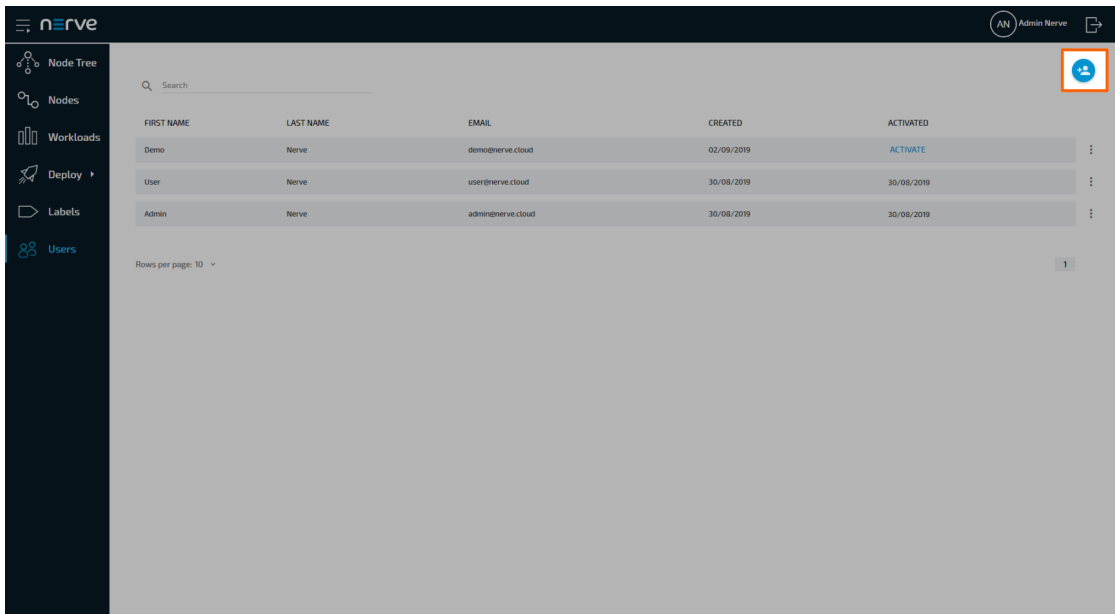
Item	Description
<b>Profile Picture</b>	Click here or drag and drop an image into the gray circle to upload a profile picture for the user. It is displayed in the upper-right corner when the user is logged in. Note that a profile picture cannot be set for users coming from LDAP synchronization.
<b>FIRST NAME</b>	This is the first name of the registered user. It is displayed in the upper-right corner when the user is logged in.
<b>LAST NAME</b>	This is the last name of the registered user. It is displayed in the upper-right corner when the user is logged in.
<b>EMAIL</b>	This is the e-mail address of the user. It is used as the username for logging in to the Management System in case of local authentication. When LDAP authentication is active, the username attribute that is carried over from LDAP synchronization is used to log in. The system also uses this e-mail for sending the activation link and the instructions on how to reset the login password. Note that passwords cannot be changed or reset if LDAP authentication is active.

Item	Description
<b>Current Password</b>	Enter the current password here when changing the password. This can only be done by the active user when local authentication is active.
<b>New Password</b>	Enter the new password here when changing the password. This can only be done by the active user when local authentication is active.
<b>Confirm Password</b>	Enter the new password again when changing the password. This can only be done by the active user when local authentication is active.
<b>Select Language</b>	Select the Management System language from the drop-down menu.
<b>Role</b>	This defines the role of the user. Refer to <a href="#">Roles and Permissions</a> for more information.

## Adding a new user

Admin users or users having the **UI\_USER:CREATE** permission assigned in a role need to create new users before these can set a password and log in to the Management System. Note that new users cannot be added when LDAP authentication is active.

1. Select **Users** in the menu on the left side.
2. Click the **Add new user** symbol in the upper-right corner.



3. Enter the required information: **First name**, **Last name** and **Email**.
4. Select one or more roles from the drop-down menu under **Role**.
5. Select **Save** to create the user.

The user account needs to be activated after it has been created by clicking an activation link. The activation link is sent to the e-mail address that was specified during the creation of the user. The activation e-mail is sent automatically by the Management System shortly after the user is created.

## Activating a user

Newly created users are sent an activation link to the specified e-mail address when the user was created. Following the link, the user will need to set their password in order to be activated and to be able to log in to the Management System.

1. Follow the activation link that was sent to the specified e-mail address.
2. Enter the new password to use for Management System access under **New Password** and **Confirm Password**.



The screenshot shows a web interface for setting a new password. On the left, there is a form with the following elements: a title 'Set new password', a 'New Password' input field, a 'Confirm Password' input field, and a blue button labeled 'SAVE NEW PASSWORD'. On the right, there is a large graphic with the 'nerve' logo in white and a stylized neural network or circuit pattern in orange and yellow against a dark blue background. A small blue button in the top right corner of the graphic reads 'Set new password and activate account.'

### NOTE

The password must contain at least one uppercase letter, one lowercase letter and one number. It must be at least 7 characters.

3. Select **SAVE NEW PASSWORD**.

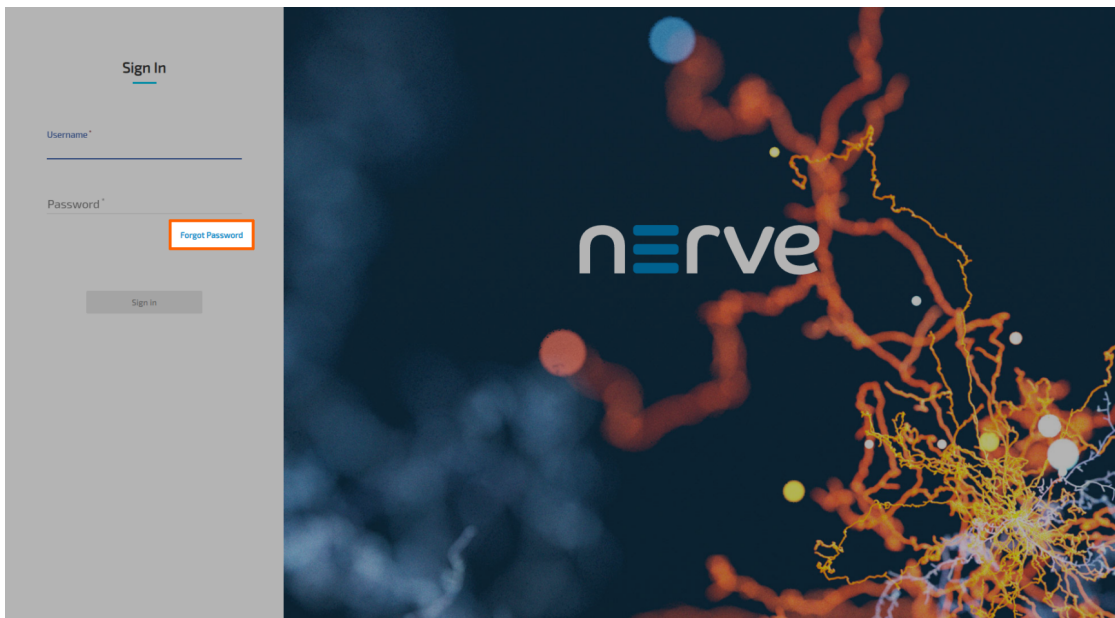
The new user is now activated and can log in immediately with their e-mail address and the password they have defined.

## Resetting the password

In case a user does not remember their password, a request to reset the password can be sent from the login page of the Management System.

Note that passwords cannot be reset if LDAP authentication is active.

1. Enter the URL of the Management System in a browser.
2. Select **Forgot Password**.



3. Enter the e-mail address of the account that needs to reset the password.



4. Select **Reset**. An e-mail with instructions is sent to the e-mail address of the account.

5. Follow the link from the e-mail to reset the password.

6. Enter the new password for Management System access under **New Password** and **Confirm Password**.

Reset Password

New Password\*

Confirm Password\*

Reset



7. Select **Reset** to save the new password.

The password has been changed and the user can log in immediately with their e-mail address and the new password.

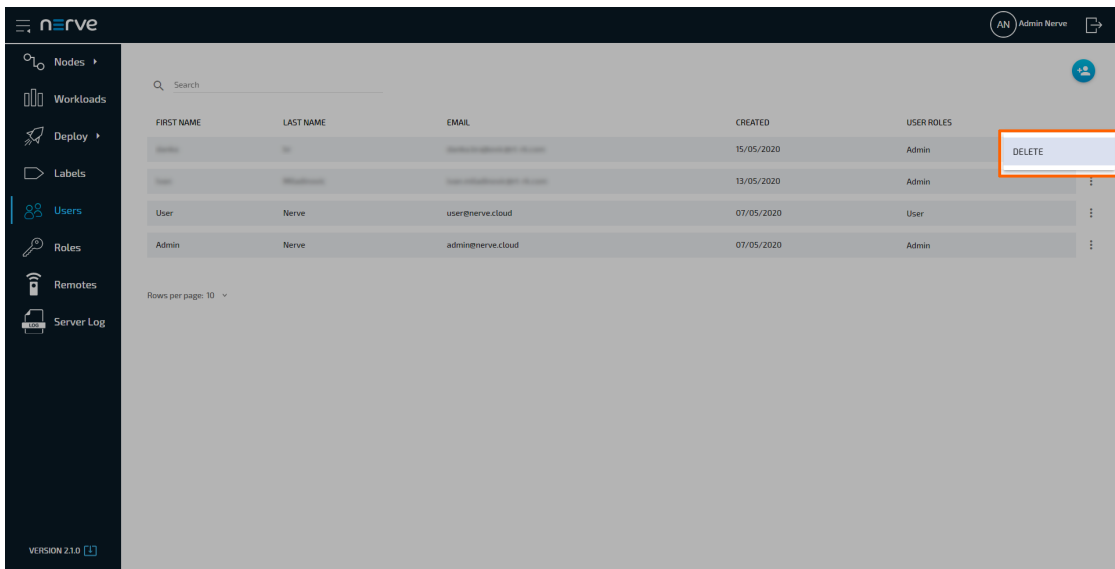
## Deleting a user

Admin users or users having the **UI\_USER:DELETE** permission assigned in a role can delete any user that is registered in the Management System. Note that users cannot be deleted when LDAP authentication is active.

1. Select **Users** in the menu on the left side.
2. Choose the user to delete.

FIRST NAME	LAST NAME	EMAIL	TYPE	CREATED	USER ROLES
User	Nerve		local	09/02/2021	Admin
User	Nerve		local	05/02/2021	Admin, log
User	Nerve		local	05/02/2021	Admin
			local	04/02/2021	Admin
			local	04/02/2021	Admin
Nerve	Documentation		local	04/02/2021	Admin
			local	04/02/2021	Admin
			local	04/02/2021	Admin
			local	04/02/2021	Admin
			local	04/02/2021	Admin
			local	04/02/2021	Admin

3. Select the ellipsis menu to the right of the user in the list.
4. Select **DELETE** from the overlay that appeared.



5. Click **OK** to confirm the deletion of the user.

## Roles and permissions

### NOTE

This page describes the standard roles and permission management in the Nerve Management System. When LDAP authentication is active, roles and permissions are applied according to the LDAP synchronization settings. Refer to [LDAP](#) for more information on LDAP synchronization.

Usage of the Management System is restricted by role-based access control (RBAC), meaning that users in the Management System are assigned roles. These roles are assigned a set of [UI permissions](#) and [API permissions](#). Three user roles — **Admin**, **User** and **Data Services Admin** — are available by default. Multiple roles can be assigned to one user. A user that is assigned multiple roles is granted the combined permissions of each role. Select **Roles** in the navigation on the left to reach a list of all available roles:

NAME	TYPE	NUMBER OF ASSIGNED USERS	DESCRIPTION
Admin	local	17	Admin role
User	local	0	User role
log	local	1	test

Item	Description
<b>Search bar (1)</b>	Use the search bar to filter roles by name.
<b>Type filter (2)</b>	Filter the list of roles by authentication method. The target of the filter is the <b>TYPE</b> column and the options are <b>Local</b> , <b>LDAP</b> and <b>All</b> .
<b>Add new role (3)</b>	Click here to add a new role.
<b>NAME (4)</b>	This is the name of the role that was defined when it was created. This name is also used when the role is assigned to users.
<b>TYPE (5)</b>	This column shows the type of the role according to the authentication method used: <ul style="list-style-type: none"> <li>• <b>Local</b> This describes roles that are created locally through the Management System.</li> <li>• <b>LDAP</b> This describes roles that are created following an LDAP synchronization.</li> </ul>
<b>NUMBER OF ASSIGNED USERS (6)</b>	The number of users that this role has been assigned to is displayed here.
<b>DESCRIPTION (7)</b>	This is a description that gives more information about each user role.
<b>Ellipsis menu (8)</b>	Clicking here opens an overlay that allows deleting roles.

The **Admin** role has all permissions assigned and cannot be edited. The **User** role has limited permissions. A user that has the **User** role assigned is not allowed to perform changes to the system such as adding or removing nodes, creating workloads or establishing remote connections among others. Users with the **User** role can work with the node tree and deploy workloads to nodes.



## UI permissions

UI permissions reflect the permissions of the frontend. They are relevant for a users interaction with the Management System. Below is the list of available UI permissions with descriptions.

As some actions depend on other actions, the system automatically selects and deselects permissions, including API permissions that are required for these actions. When creating a new role for a regular user that will operate the Management System, use the UI permissions as a starting point and change API permissions only if necessary. However, note that changes to API permissions should only be done by users with expert knowledge.

The tables below are separated by the part of the Management System the permissions affect.

### Workload deployment

Permission	Name	Description
<b>Deploy workload</b>	<b>UI_DEPLOY:DEPLOY</b>	Permission that grants the user the rights to deploy a workload to a node.
<b>Force stop campaign task</b>	<b>UI_DEPLOY:FORCE_CANCEL_ONE</b>	Permission that grants the user access to see the force stop button for a deployment campaign.
<b>Delete workload deploy log</b>	<b>UI_DEPLOY:LOG_DELETE</b>	Permission that grants the user the rights to delete a log entry in the workload deployment log.
<b>Reset deployment task</b>	<b>UI_DEPLOY:LOG_RESET</b>	Permission that grants the user the rights to reset a task in the workload deployment log.
<b>Reset all deployment tasks</b>	<b>UI_DEPLOY:LOG_RESET_ALL</b>	Permission that grants the user the rights to reset all tasks in the workload deployment log.
<b>Preview workload deploy log</b>	<b>UI_DEPLOY:LOG_VIEW</b>	Permission that grants the user the rights to view a log entry in the workload deployment log.

Permission	Name	Description
Access "Deploy" -> "Dry run"	<b>UI_SUBNAV_DEPLOY_DRY_RUN:VIEW</b>	Permission that grants the user the rights to see the dry run entry in the navigation menu and the rights to perform the dry run action with workloads to nodes.
Access "Deploy" -> "Log"	<b>UI_SUBNAV_DEPLOY_LOG:VIEW</b>	Permission that grants the user the rights to see the deployment log entry in the navigation menu and the rights to view the workload deployment log.

## Labels

Permission	Name	Description
Create new label	<b>UI_LABEL:CREATE</b>	Permission that grants the user the rights to create new labels.
Delete label	<b>UI_LABEL:DELETE</b>	Permission that grants the user the rights to delete labels.
Edit existing label	<b>UI_LABEL:EDIT</b>	Permission that grants the user the rights to edit labels.
Group labels by key	<b>UI_LABEL:GROUP</b>	Permission that grants the user the rights to group labels by key.
Merge labels to one	<b>UI_LABEL:MERGE</b>	Permission that grants the user the rights to merge multiple labels.
Preview list of labels	<b>UI_LABEL:VIEW</b>	Permission that grants the user the rights to view the details of a label.

## LDAP

Permission	Name	Description
Show edit button	<b>UI_LDAP:MANAGE_LDAP</b>	Permission that grants the user the rights to edit the LDAP configuration.

## Navigation menu

Permission	Name	Description
Access Data services Feature Preview	<b>UI_NAV_DATA_SERVICES:VIEW</b>	Permission that grants the user access to the Data Services feature.

Permission	Name	Description
<b>"Deploy" section</b>	<b>UI_NAV_DEPLOY:VIEW</b>	Permission that grants the user the rights to view the deployment menu.
<b>"Labels" section</b>	<b>UI_NAV_LABELS:VIEW</b>	Permission that grants the user the rights to see the labels entry in the navigation menu and the rights to list labels.
<b>Show default ldap configuration</b>	<b>UI_NAV_LDAP:VIEW</b>	Permission that grants the user the rights to see the LDAP entry in the navigation menu.
<b>"Nodes" section</b>	<b>UI_NAV_NODES:VIEW</b>	Permission that grants the user the rights to see the nodes entry in the navigation menu and the rights to list nodes.
<b>"Remotes" section</b>	<b>UI_NAV_REMOTE_CONNECTIONS:VIEW</b>	Permission that grants the user the rights to view active remote connections.
<b>"Roles" section</b>	<b>UI_NAV_ROLES:VIEW</b>	Permission that grants the user the rights to see the roles entry in the navigation menu and the rights to list roles.
<b>Access "Server Log"</b>	<b>UI_NAV_SERVER_LOGS:VIEW</b>	Permission that grants the user the rights to list internal server logs.
<b>"Users" section</b>	<b>UI_NAV_USERS:VIEW</b>	Permission that grants the user the rights to see the users entry in the navigation menu and the rights to list users.

Permission	Name	Description
<b>"Workloads" section</b>	<b>UI_NAV_WORKLOADS:VIEW</b>	Permission that grants the user the rights to see the workloads entry in the navigation menu and the rights to list workloads.

## Nodes

Permission	Name	Description
<b>Create new node</b>	<b>UI_NODE:CREATE</b>	Permission that grants the user the rights to create new nodes.
<b>Delete node</b>	<b>UI_NODE:DELETE</b>	Permission that grants the user the rights to delete nodes.
<b>Edit existing node</b>	<b>UI_NODE:EDIT</b>	Permission that grants the user the rights to edit nodes.
<b>Node logging and monitoring settings</b>	<b>UI_NODE:LOGGING_AND_MONITORING_SETTINGS</b>	Permission that grants the user the rights to access the logging and monitoring settings of a node.
<b>Node reboot</b>	<b>UI_NODE:REBOOT</b>	Permission that grants the user the rights to reboot a node.
<b>Show logs of node</b>	<b>UI_NODE:SHOW_LOGS</b>	Permission that grants the user the rights to view internal node logs.

Permission	Name	Description
Preview node	UI_NODE:VIEW	Permission that grants the user the rights to view details of a node.
Change a Node logging level configuration	UI_NODE_LOG_LEVEL:MANAGE_LOG_LEVELS	Permission that grants the user the rights to change the logging level settings of a node.
Delete node update log	UI_NODE_UPDATE:LOG_DELETE	Permission that grants the user the rights to delete a log entry in the node update log.
Show node update log	UI_NODE_UPDATE:LOG_VIEW	Permission that grants the user the rights to view the details of a node update log entry.
Update node	UI_NODE_UPDATE:UPDATE	Permission that grants the user the rights to update a node.
Access "Nodes" -> "Updates"	UI_SUBNAV_NODE_UPDATE:VIEW	Permission that grants the user the rights to see the updates sub-entry in the navigation menu.
Access "Nodes" -> "Log"	UI_SUBNAV_NODE_UPDATE_LOG:VIEW	Permission that grants the user the rights to see the log sub-entry in the navigation menu.

## Node tree

Permission	Name	Description
<b>Add new tree item</b>	<b>UI_NODE_TREE:ADD</b>	Permission that grants the user the rights to add new elements in the node tree.
<b>Delete tree item</b>	<b>UI_NODE_TREE:DELETE</b>	Permission that grants the user the rights to delete an element of the node tree.
<b>Edit tree item</b>	<b>UI_NODE_TREE:EDIT</b>	Permission that grants the user the rights to edit an element of the node tree.
<b>Node tree manipulation</b>	<b>UI_NODE_TREE:MANIPULATE</b>	Permission that grants the user the rights to manipulate the elements of the node tree structure, i.e perform changes to position and order.
<b>Preview node details in Node tree</b>	<b>UI_NODE_TREE:NODE_DETAILS</b>	Permission that grants the user the rights to view the node details of a node in the node tree.

## System notifications

Permission	Name	Description
<b>Create notification</b>	<b>UI_NOTIFICATION:CREATE</b>	Permission that grants the user the rights to create a system notification.
<b>Delete notification</b>	<b>UI_NOTIFICATION:DELETE</b>	Permission that grants the user the rights to delete a system notification.
<b>Edit notification</b>	<b>UI_NOTIFICATION:EDIT</b>	Permission that grants the user the rights to edit an existing system notification.
<b>View notification</b>	<b>UI_NOTIFICATION:VIEW</b>	Permission that grants the user the rights to view system notifications.

## Remote connections

Permission	Name	Description
<b>Connect over remote connection</b>	<b>UI_REMOTE_CONN:CONNECT</b>	Permission that grants the user the rights to connect to a host through a remote connection.

Permission	Name	Description
<b>Create remote connection</b>	<b>UI_REMOTE_CONN:CREATE</b>	Permission that grants the user the rights to establish a new remote connection.
<b>Delete remote connection</b>	<b>UI_REMOTE_CONN:DELETE</b>	Permission that grants the user the rights to delete remote connections.
<b>Edit remote connection</b>	<b>UI_REMOTE_CONN:EDIT</b>	Permission that grants the user the rights to edit existing remote connections.
<b>List all remote connections</b>	<b>UI_REMOTE_CONN:LIST</b>	Permission that grants the user the rights to list all remote connections in the node and workload details.
<b>Preview remote connection</b>	<b>UI_REMOTE_CONN:VIEW</b>	Permission that grants the user the rights to view the details of a remote connection.
<b>Terminate remote connection</b>	<b>UI_REMOTE_CONNECTIONS:TERMINATE</b>	Permission that grants the user the rights to terminate active remote connections.

## Roles

Permission	Name	Description
<b>Create new role</b>	<b>UI_ROLE:CREATE</b>	Permission that grants the user the rights to create new roles.
<b>Delete role</b>	<b>UI_ROLE:DELETE</b>	Permission that grants the user the rights to delete roles.
<b>Edit role</b>	<b>UI_ROLE:EDIT</b>	Permission that grants the user the rights to edit roles.
<b>Preview role</b>	<b>UI_ROLE:VIEW</b>	Permission that grants the user the rights to view the details of a role.

## Server log

Permission	Name	Description
<b>Preview server logs</b>	<b>UI_SERVER_LOGS:VIEW</b>	Permission that grants the user the rights to list internal server logs.

## Usage reports

Permission	Name	Description
<b>Access Usage Report Feature Preview</b>	<b>UI_USAGE_REPORT:VIEW</b>	Permission that grants the user the rights to access the usage report page.

## User menu

Permission	Name	Description
<b>Create new user profile</b>	<b>UI_USER:CREATE</b>	Permission that grants the user the rights to create new users.
<b>Delete user profile</b>	<b>UI_USER:DELETE</b>	Permission that grants the user the rights to delete users.
<b>Edit user profile</b>	<b>UI_USER:EDIT</b>	Permission that grants the user the rights to edit the profiles of other users.
<b>Preview user profile</b>	<b>UI_USER:VIEW</b>	Permission that grants the user the rights to view the details of a user.
<b>Edit user settings</b>	<b>UI_USER_SETTINGS:UPDATE</b>	Permission that grants the user the rights to update their user settings.
<b>Preview user settings</b>	<b>UI_USER_SETTINGS:VIEW</b>	Permission that grants the user the rights to view their user settings.

## Management System update

Permission	Name	Description
<b>List available Cloud app versions</b>	<b>UI_VERSION:LIST</b>	Permission that grants the user the rights to list all available versions of the Management System.
<b>Upload Cloud app versions</b>	<b>UI_VERSION:UPDATE</b>	Permission that grants the user the rights to update the Management System.

## Workload management

Permission	Name	Description
<b>Create new workload</b>	<b>UI_WORKLOAD:CREATE</b>	Permission that grants the user the rights to create new workloads.



Permission	Name	Description
<b>Delete workload</b>	<b>UI_WORKLOAD:DELETE</b>	Permission that grants the user the rights to delete workloads.
<b>Disable workload</b>	<b>UI_WORKLOAD:DISABLE</b>	Permission that grants the user the rights to disable workloads.
<b>Edit workload</b>	<b>UI_WORKLOAD:EDIT</b>	Permission that grants the user the rights to edit workload details (name and description).
<b>Preview workload</b>	<b>UI_WORKLOAD:VIEW</b>	Permission that grants the user the rights to view the details of a workload.
<b>Create workload version</b>	<b>UI_WORKLOAD:VERSION_CREATE</b>	Permission that grants the user the rights to create new workload versions.
<b>Delete workload version</b>	<b>UI_WORKLOAD:VERSION_DELETE</b>	Permission that grants the user the rights to delete workload versions.
<b>Edit workload version</b>	<b>UI_WORKLOAD:VERSION_EDIT</b>	Permission that grants the user the rights to edit workload versions.

Permission	Name	Description
<b>Preview workload version</b>	<b>UI_WORKLOAD:VERSION_VIEW</b>	Permission that grants the user the rights to view workload versions.
<b>Apply workload configuration</b>	<b>UI_WORKLOAD_CONFIGURATION:APPLY</b>	Permission that grants the user the rights to apply configuration files to a deployed workload.
<b>Update workload resources</b>	<b>UI_WORKLOAD_CONFIGURATION:UPDATE_RESOURCES</b>	Permission that grants the user the rights to change the allocated resources of a deployed workload.
<b>Control deployed workload</b>	<b>UI_WORKLOAD_CONTROL:CONTROL</b>	Permission that grants the user full control over status and life cycle of workloads deployed to nodes.
<b>List deployed workloads</b>	<b>UI_WORKLOAD_CONTROL:LIST</b>	Permission that grants the user the rights to list workloads that are deployed to a node.
<b>Preview deployed workloads</b>	<b>UI_WORKLOAD_CONTROL:VIEW</b>	Permission that grants the user the rights to view the details of a workload deployed to a node.

## API permissions

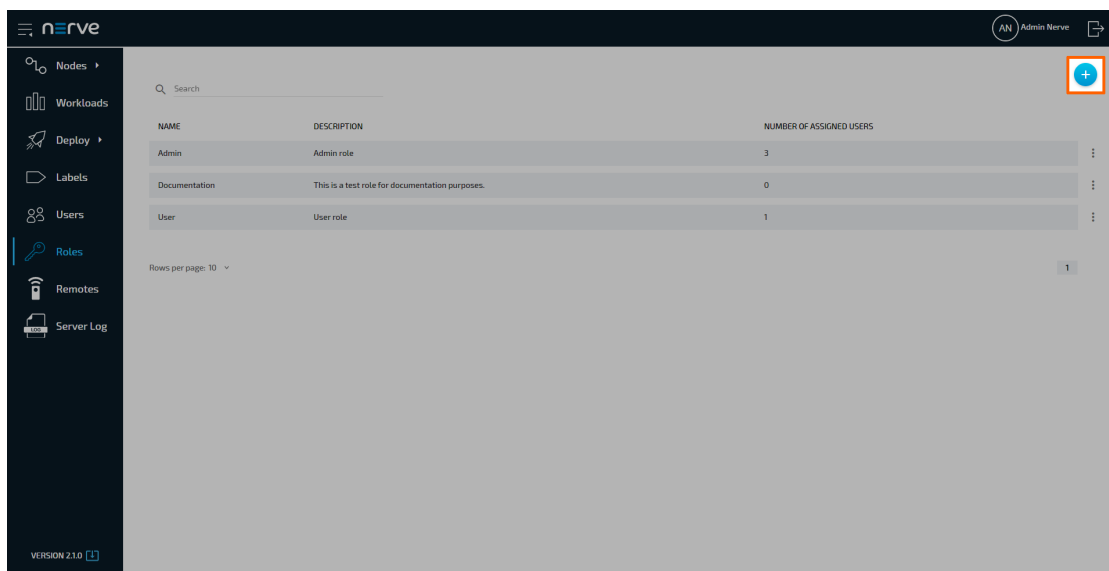
API permissions reflect the permissions of the server backend. They are primarily relevant for automating the Management System through API calls. When creating a role in the Management System for a program, they can be selected without selecting UI permissions beforehand. When creating a new role for a regular user that will operate the Management System, use the UI permissions as a starting point and change API permissions only if necessary. Note that API permissions should only be handled by persons with expert knowledge.

## Adding a new role

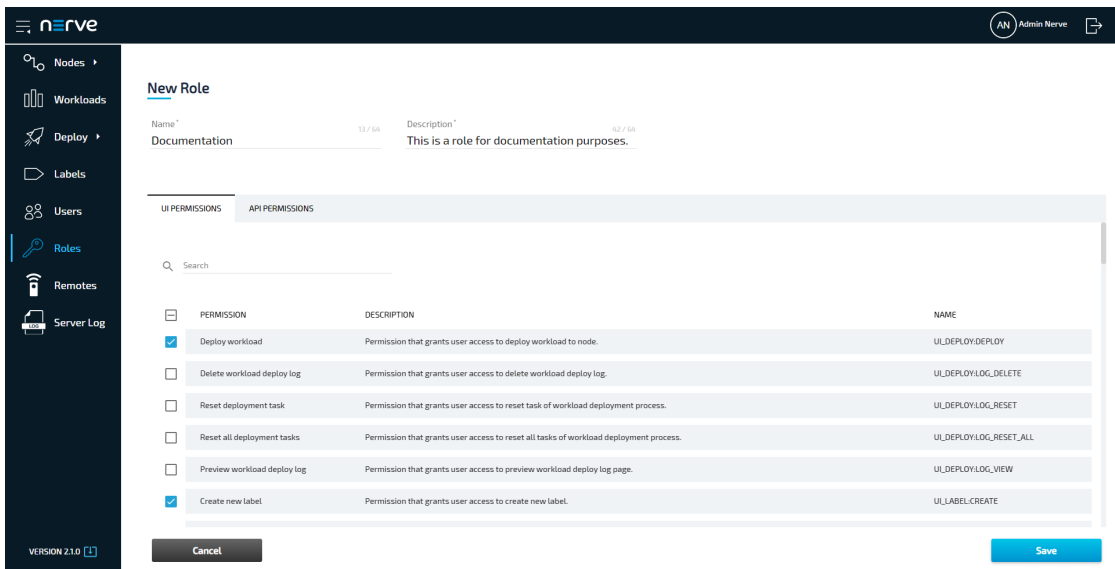
When adding a new role, it depends whether the role is going to be created for regular users or programs. When creating a new role for a regular user that will operate the Management System, use the UI permissions as a starting point and change API permissions only if necessary. When creating a role for a program, API permissions can be selected without selecting UI permissions beforehand. Note that API permissions should only be handled by persons with expert knowledge.

Selecting one permission might automatically select other permissions, which are needed to perform the task indicated by the selected permission. An example: if a user is permitted to deploy a workload, then the same user is also permitted to view the list of workloads. Associated API permissions will also be selected. Note that deselecting a permission might also deselect linked permissions.

1. Select **Roles** in the navigation on the left.
2. Click the plus symbol (**Add new role**) in the upper-right corner.

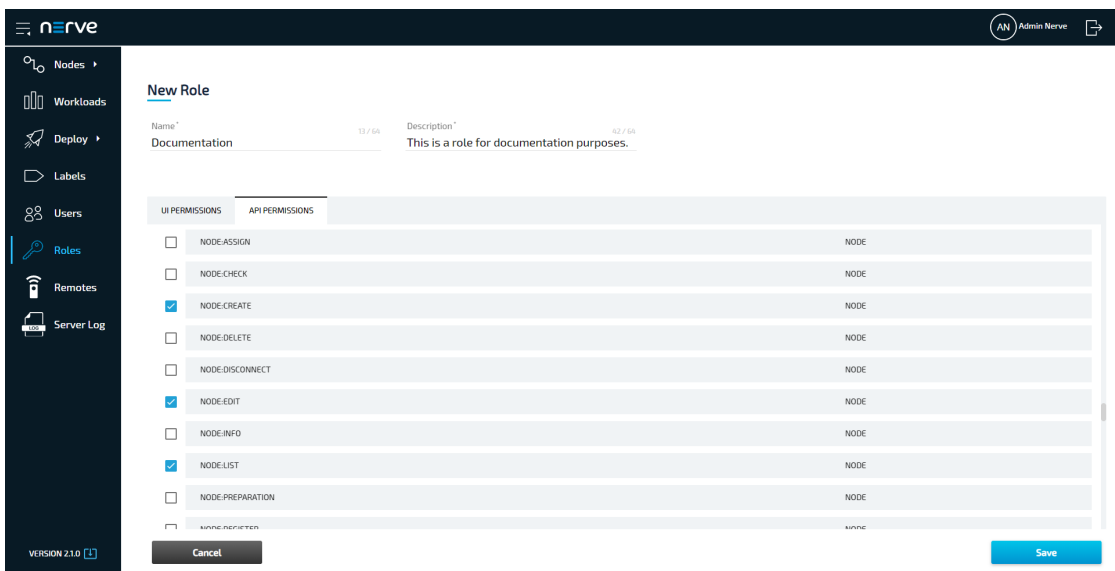


3. Enter a **Name** and a **Description** at the top.
4. Select the **UI PERMISSIONS** tab.
5. Tick the checkboxes next to the desired permissions.



6. Select the **API PERMISSIONS** tab.

7. Tick or untick the permissions that need to be changed.



#### NOTE

Make sure to review the selected permissions for completeness before saving the role. The system automatically selects and deselects permissions that are linked and might have added or removed desired permissions when permissions were selected or deselected.

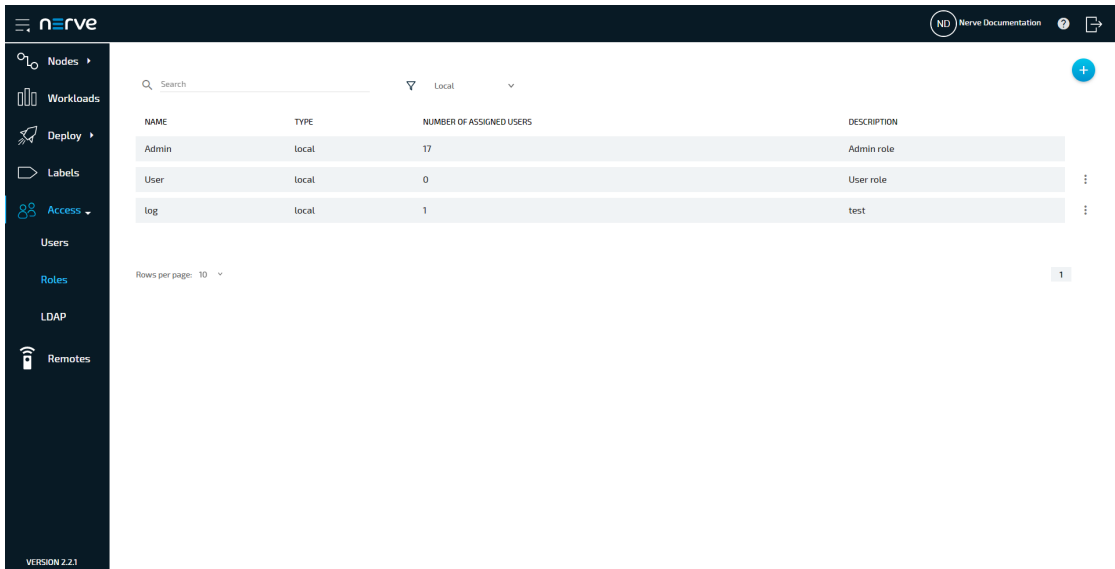
8. Click **Save**.

## Editing a role

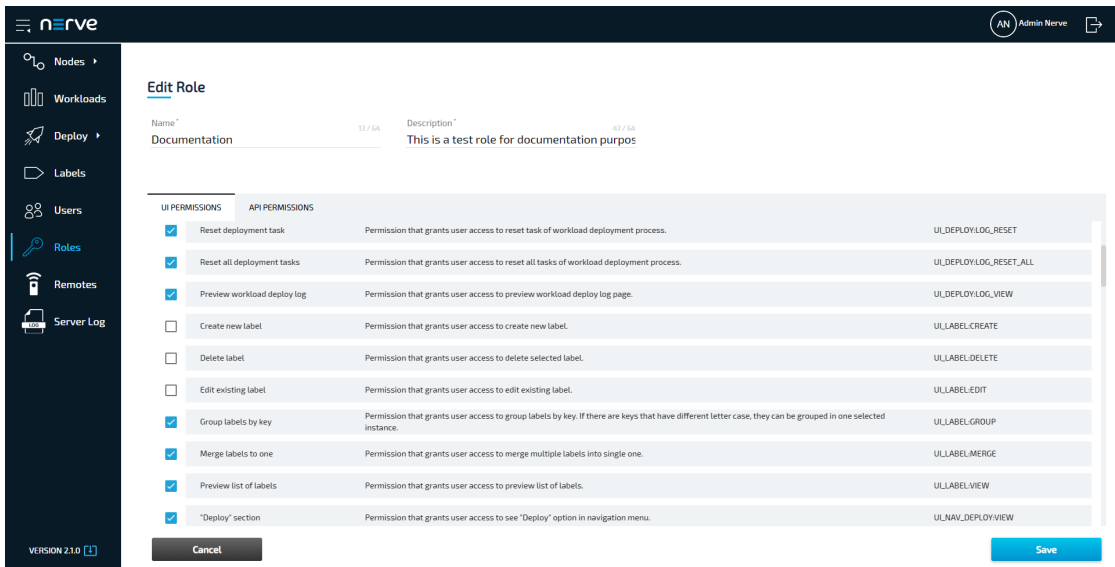
Note that editing the permissions of a role changes the permissions for users who are already assigned this role. Also, note that editing of roles coming from LDAP

synchronization is limited. The name and description of a role cannot be edited. Permissions, however, can be edited.

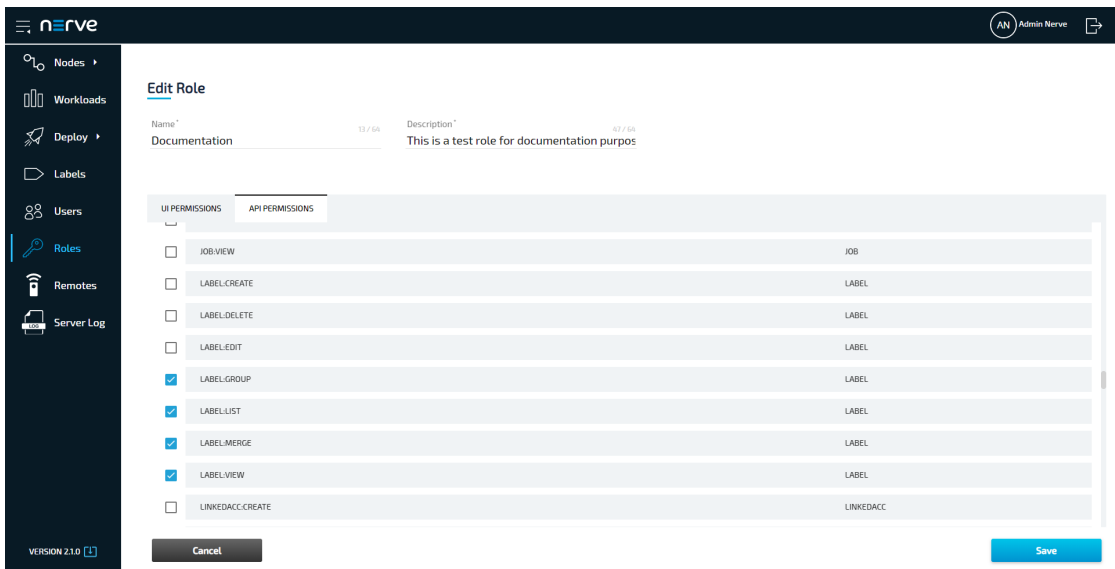
1. Select **Roles** in the navigation on the left.
2. Select a role from the list.



3. Edit **Name** and **Description** at the top.
4. Select the **UI PERMISSIONS** tab.
5. Tick or untick the permissions that need to be changed.



6. Select the **API PERMISSIONS** tab.
7. Tick or untick the permissions that need to be changed.



## NOTE

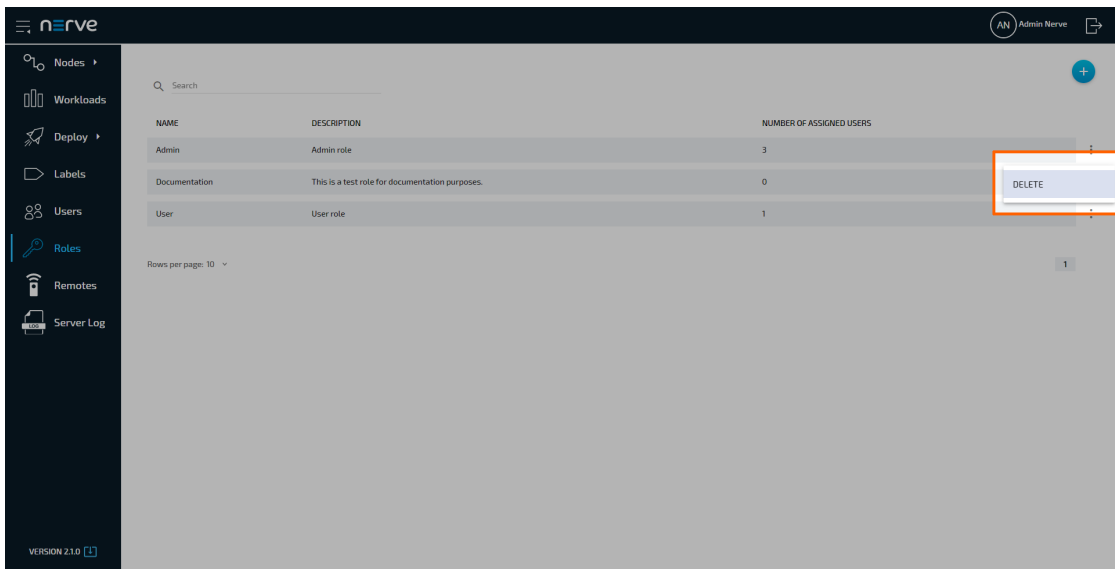
Make sure to review the selected permissions for completeness before saving the role. The system automatically selects and deselects permissions that are linked and might have added or removed desired permissions when permissions were selected or deselected.

8. Click **Save**.

## Deleting a role

Note that a role cannot be deleted if it is assigned to a user. Also, note that roles coming from LDAP synchronization cannot be deleted.

1. Select **Roles** in the navigation on the left.
2. Choose a role from the list.
3. Click the ellipsis menu next to the role.
4. Select **DELETE** in the overlay that appeared.

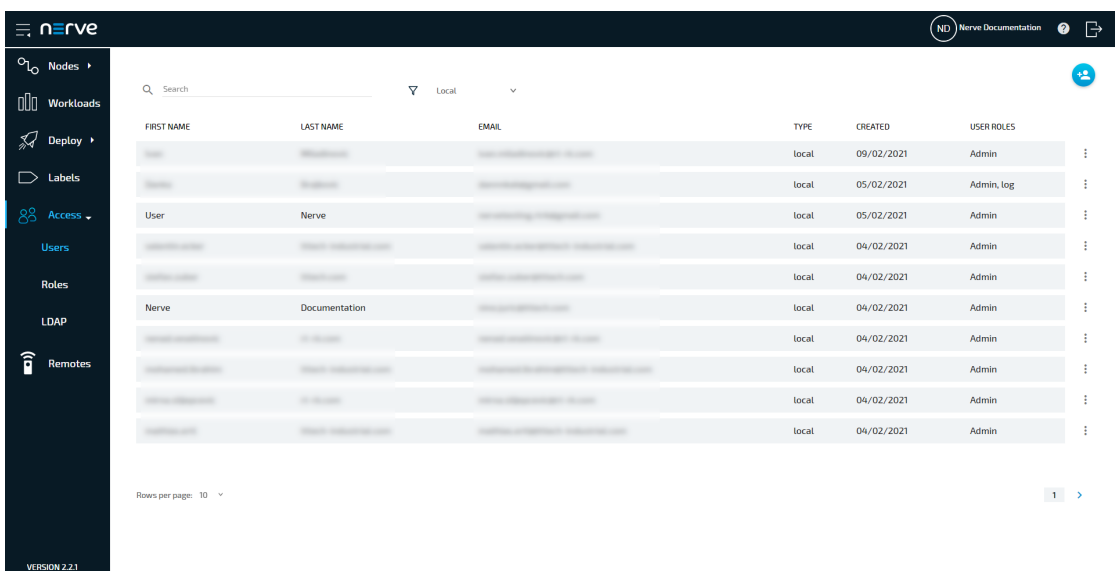


5. Select **OK** to delete the role.

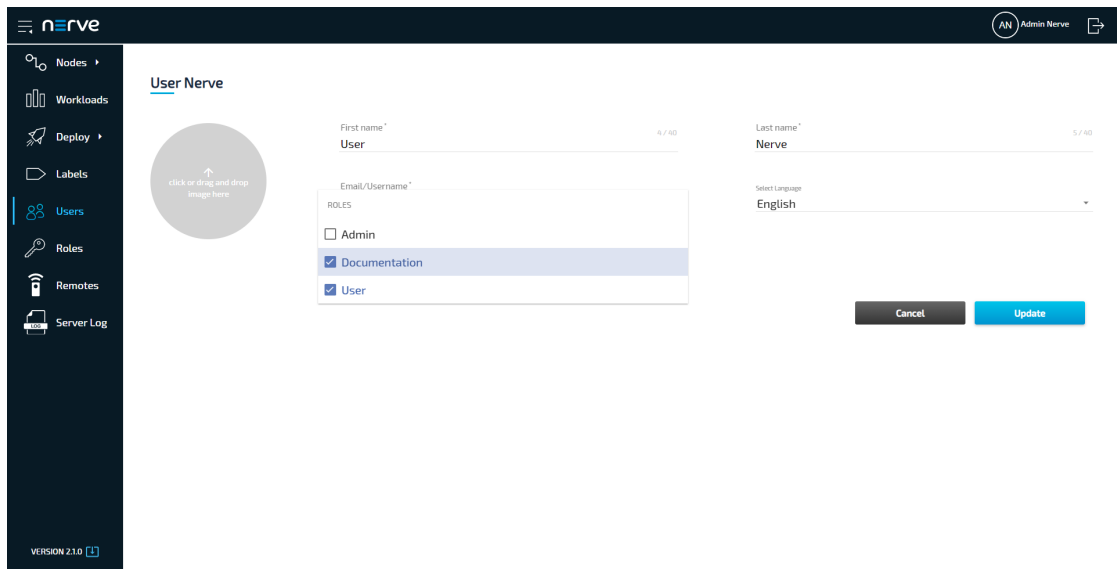
## Assigning a role to a user

Assigning a role is done in the users menu. Users can be assigned multiple roles. A user that is assigned multiple roles is granted the combined permissions of each role.

1. Select **Users** in the navigation on the left.
2. Select a user from the list.



3. Click the field under **Role** to open a drop-down menu.
4. Tick one or more roles that will be assigned to this user. Note that at least one role must be selected.



5. Select **Update** in the lower-right.

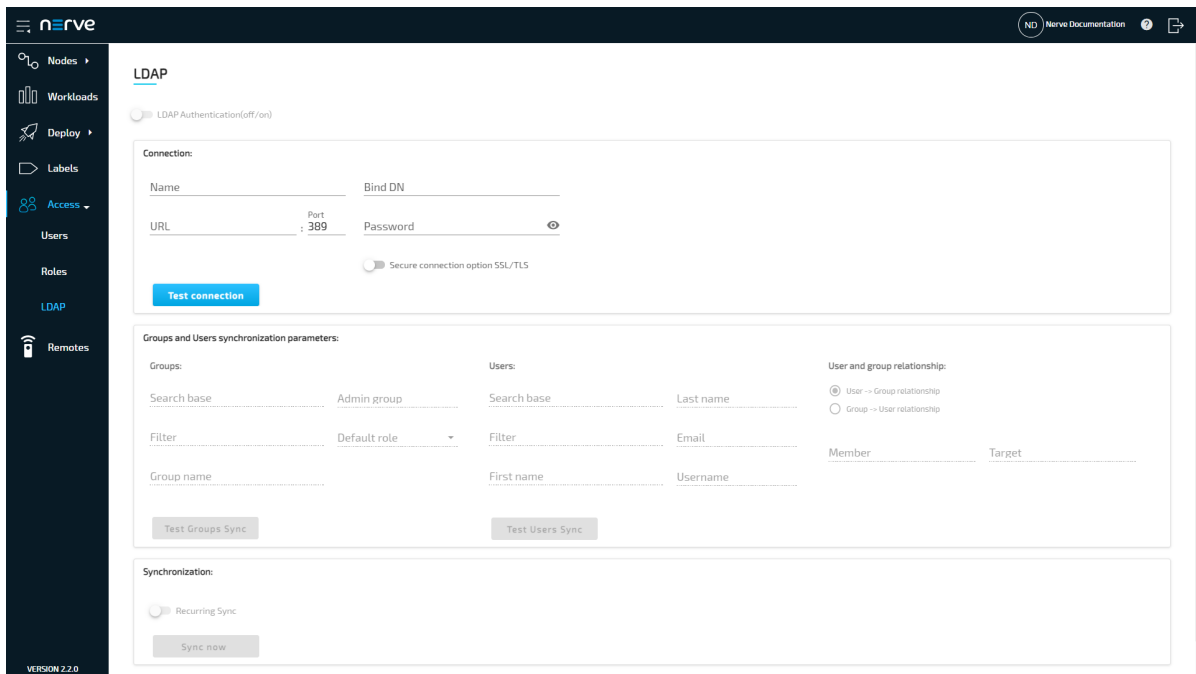
## Enabling access management with LDAP in the Management System

The Nerve Management System allows the import of users and roles for access management through its LDAP integration. Data is synchronized with an LDAP server and the authentication of users is done against this LDAP server instead of the Management System itself. The user authentication provided is LDAP Simple Authentication (username and password) with the option to enable SSL/TLS for a secure connection between the LDAP server and the Management System.

An existing LDAP server is required to enable active directory access management. The configuration of the LDAP synchronization in the Management System requires expert knowledge. It is recommended that the configuration is performed by an LDAP admin.

Expand **Access > LDAP** to reach the LDAP configuration. It is separated into three sections.





## Item

## Description

### LDAP Authentication(off/on)

Toggle this to activate or deactivate access management with LDAP once the configuration on this page has been completed.

Item	Description
<b>Connection:</b>	<p>This section contains the required parameters to connect to an LDAP server.</p> <p><b>Name</b> This is a user defined name for the configuration. Enter any name that accurately describes the connection.</p> <p><b>URL</b> This can be either an IP address or the hostname of the LDAP server.</p> <p><b>Port</b> Enter the port used for the LDAP service. Note that the standard port 389 is automatically filled in. Also, port 636 is automatically filled in if <b>Secure connection option SSL/TLS</b> is activated.</p> <p><b>Bind DN</b> Enter the user in LDAP that is used for establishing the connection. This is either a user that is used for external connections or an admin user. The user needs to be entered in the form of a distinguished name.</p> <p><b>Password</b> Enter the password of the <b>Bind DN</b> user.</p> <p><b>Secure connection option SSL/TLS</b> Toggle to enable a secure connection here. The secure connection is implicitly accepted and certificates are exchanged by enabling this option. The standard port 636 for secure connections over SSL/TLS is filled in automatically in the <b>Port</b> field.</p> <p><b>Test connection</b> This button is only available after the required fields have been filled in. Once clicked, a green check mark signifies a successful connection while a red cross signifies an unsuccessful connection attempt.</p>

Specify the required parameters for users and groups synchronization here. The options are grayed out by default. Once a connection to an LDAP server has been tested successfully, marked by a green check mark, the options will become available. Two query buttons are available here to test each configuration.

### Groups:

- **Search Base**  
This is the directory point where the groups synchronization is started from.
- **Filter**  
Enter a logical expression here to filter for desired groups. The filter has to be in the format that is used in LDAP. Hover over the text field to see the full filter displayed in a quick tip. An example filter could look like this: `(&(cn=Nerve*)(objectClass=groupOfNames))`
- **Group name**  
Enter the LDAP attribute that is used as a group name.  
Example: If cn is entered, roles in the Management System receive the value of the cn attribute as their name.
- **Admin group**  
Enter the group name from LDAP that will receive the Admin role in the Nerve Management System.
- **Default role**  
Specify a role in the Management System that is used as a permission template for all groups coming from LDAP. The only exception is the **Admin group**.  
Note that it is recommended to create a new role first. Refer to [Adding a new role](#) if roles have not been defined yet.
- **Test Groups Sync**  
Test whether the filter and settings for groups synchronization work with this button. A pop-up window opens where it is possible to check a preview list of groups that will be synchronized.

### Users:

#### Groups and Users synchronization parameters:

- **Search Base**  
This is the directory point where the users synchronization is started from.
- **Filter**  
Enter a logical expression here to filter for desired users. The filter has to be in the format that is used in LDAP. Hover over the text field to see the full filter displayed in a quick tip. An example filter could look like this: `(&(objectClass=person)(memberOf:1.2.840.113556.1.4.1941:=cn=Nerve,cn=Roles,dc=example,dc=com))`
- **First name**  
Enter the LDAP attribute that will be used as the first name for users.
- **Last name**  
Enter the LDAP attribute that will be used as the last name for users.
- **Email**  
Enter the LDAP attribute that will be used as the email for users.
- **Username**  
Enter the LDAP attribute that will be used as the username for authentication.
- **Test Users Sync**  
Test whether the filter and settings for users synchronization work with this button. A pop-up window opens where it is possible to check a preview list of users that will be synchronized.

### User and group relationship:

- **User -> Group relationship**  
Tick this option to select users as the source and groups as the target for synchronization.
- **Group -> User relationship**

Item	Description
<b>Synchronization:</b>	<p><b>Recurring sync</b> Activate <b>Recurring sync</b> to set an exact time or interval for recurring synchronization between the Management System and the LDAP server. This can be done in two ways:</p> <ul style="list-style-type: none"> <li>• <b>Exact time of day</b> If an exact time is specified, the process is executed once a day at the specified time.</li> <li>• <b>Interval in hours</b> Selecting an interval in hours executes the synchronization after the specified time has passed.</li> </ul> <p><b>Sync now</b> Select <b>Sync now</b> to perform an actual synchronization of users and groups. A result message is shown in the upper right corner at the end of the synchronization process.</p>

#### NOTE

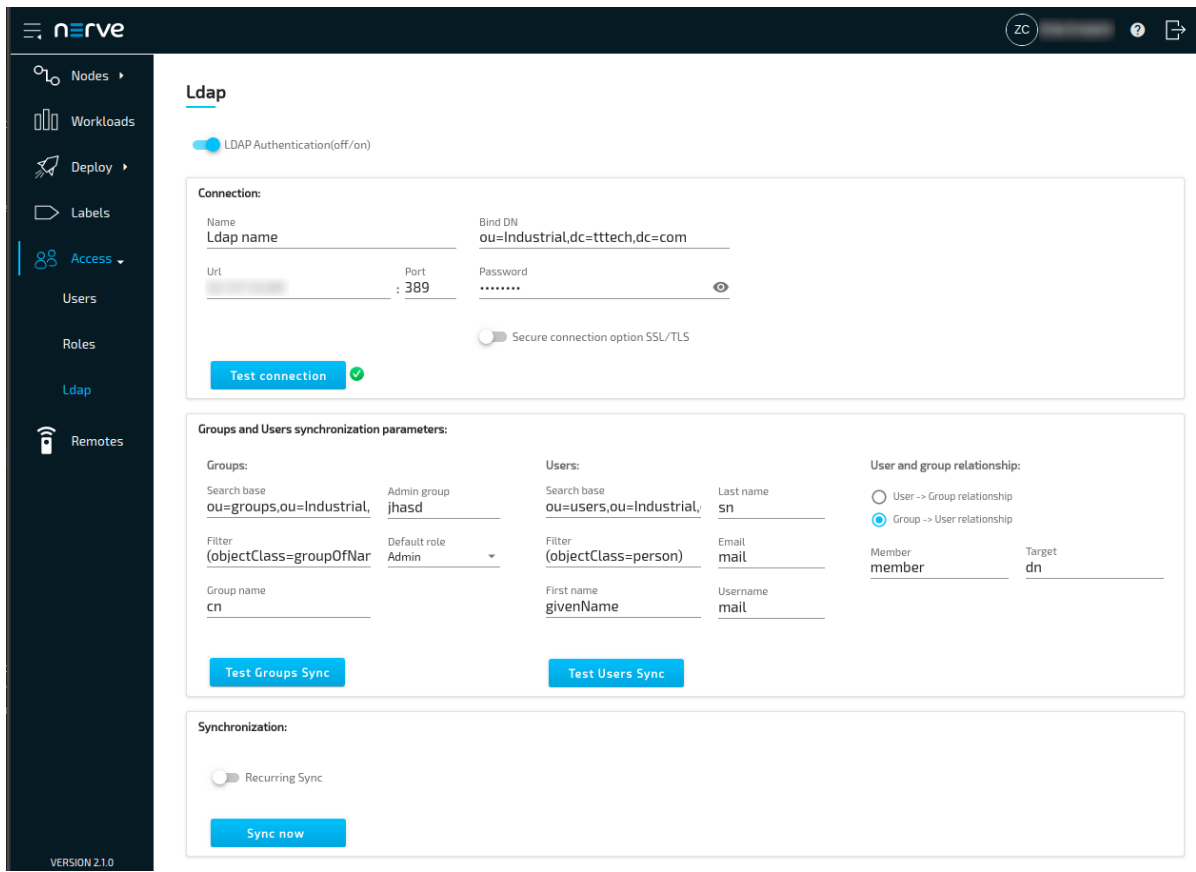
The configuration above is dependent on the LDAP server configuration. Consult the LDAP admin when configuring the LDAP integration in the Management System.

## Recommended workflow

As there are many fields and buttons that serve different purposes, consult the summarized workflow below for a quick overview.

1. Enter the required information in the **Connection** fields.
2. Select **Test connection**. If a green check mark appears, the connection was successful.
3. Fill in the **Groups** parameters under **Groups and Users synchronization parameters**.
4. Select **Test Groups Sync**. A pop-up window will appear.
5. Check the preview list of groups that will be synchronized.
6. Fill in the **Users** parameters under **Groups and Users synchronization parameters**.
7. Select **Test Users Sync**. A pop-up window will appear.
8. Check the preview list of users that will be synchronized.
9. Select **Sync now** to apply the configuration.
10. *Optional: Toggle **LDAP synchronization(off/on)** at the top.*
11. Select **Save** at the bottom to save the configuration.

Toggling the **LDAP synchronization(off/on)** switch is marked as optional here, as it can also be toggled at a later time. Save the configuration and toggle the switch to use the LDAP integration when desired. Below is an example configuration:



## Differences in user and role management when LDAP authentication is activate

When LDAP authentication is active, editing of users and roles is disabled or limited. The differences are:

- The LDAP username is used for authentication instead of the email address.
- Passwords cannot be changed or reset.
- New users cannot be added.
- Existing users, LDAP or local, cannot be edited or deleted.
- Profile pictures cannot be set.
- Names and descriptions of roles cannot be edited.
- Permissions of roles can be edited.
- Roles coming from LDAP synchronization cannot be deleted.

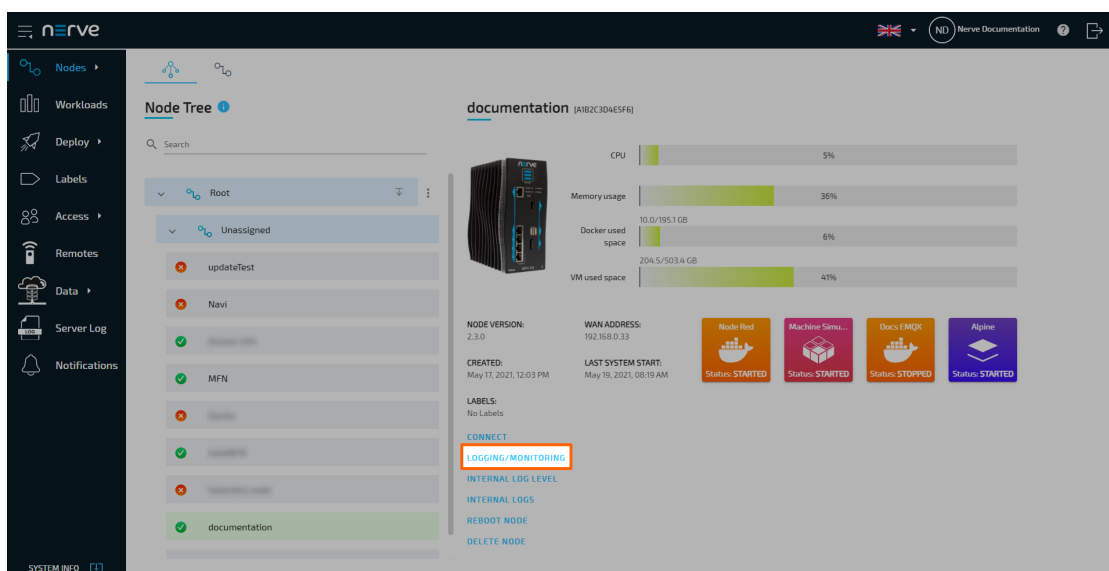
All changes to users are taken over from the LDAP server after a synchronization is performed. Once LDAP synchronization is deactivated, local users can be edited again. LDAP users, however, still cannot be edited. Note that LDAP users stay in the list of users even when LDAP synchronization is deactivated. However, the users and roles lists can be filtered by local or LDAP. Refer to [Users](#) and [Roles and permissions](#) for more information on local user and role management in the Management System.

## Logging and monitoring

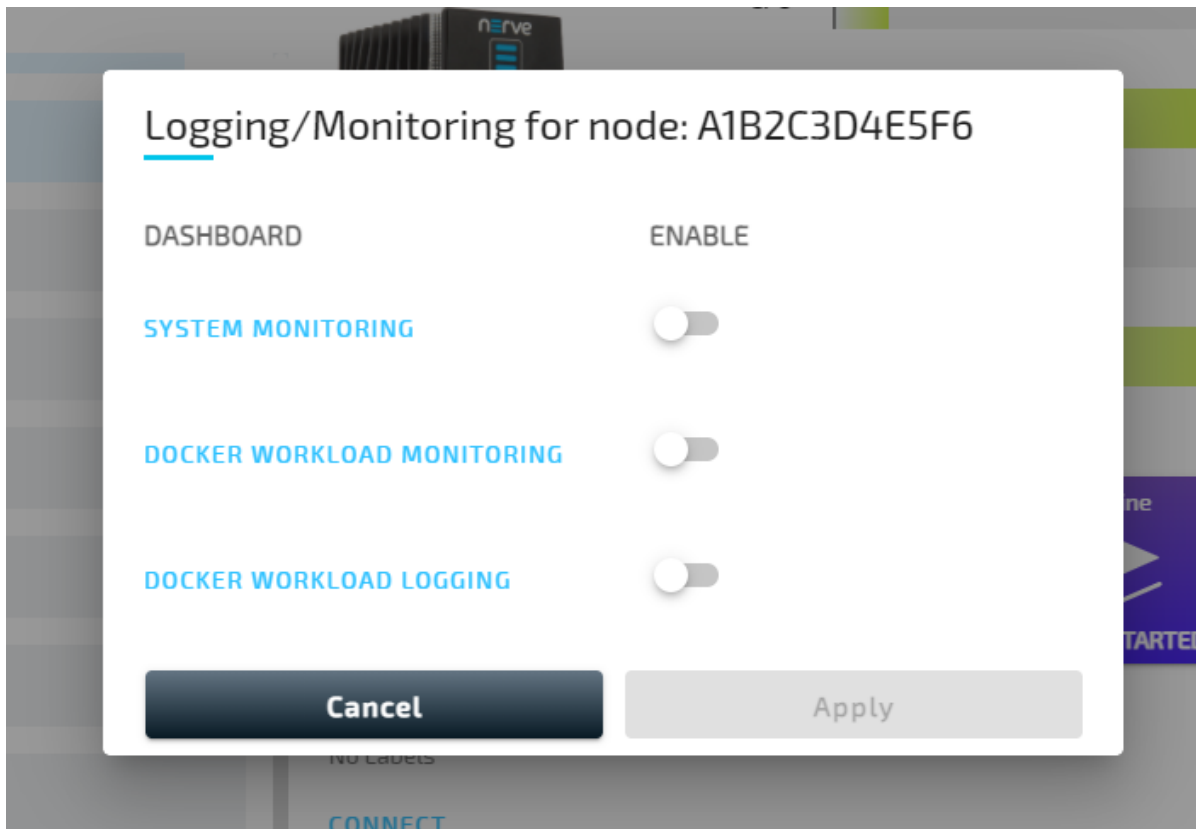
Data is gathered for Docker workloads and system parameters regarding resource usage. Metrics are then sent to the Elastic Stack in the Management System. Kibana is

used for visualization. Logging and monitoring options can be configured for each node in the node details screen in the Management System:

1. Log in to the Management System.
2. Select **Nodes** in the navigation on the left.
3. Select the node tree tab.
4. Select a node in the node tree.
5. Select **LOGGING/MONITORING**.



From here, toggle the corresponding slider of a dashboard on the right and select **Apply** to enable logging or monitoring on that node. The dashboards can be accessed by selecting the dashboard name on the left. A new browser tab opens, displaying the selected Kibana dashboard. Note that all dashboards are also available if the node is offline.

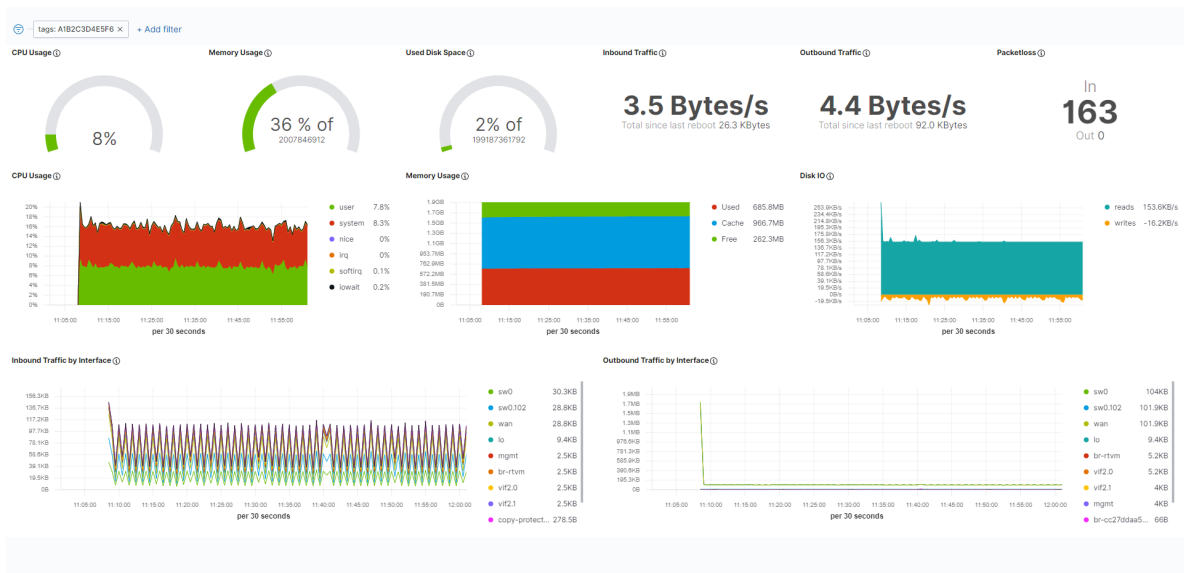


#### NOTE

Some Kibana knowledge could be beneficial when working with the dashboards. Refer to the official [Kibana guide](#) for more information on Kibana.

## System monitoring

Resource utilization of the system as a whole is tracked when system monitoring is enabled. Refer to the screenshot and the table below for more information on the data that is gathered and displayed. Toggle the **SYSTEM MONITORING** slider to enable the tracking of system data.



Item	Description
------	-------------

<b>CPU Usage</b>	<p><b>Gauge chart</b> This chart displays the currently used total percentage of the CPU.</p> <p><b>Line graph</b> The line graph displays the CPU usage in percent over time. Data is displayed scaled over time.</p> <p>In general, the data displayed here is according to how CPU usage is understood in Linux. For an explanation on how CPU usage is handled in Linux, refer to this <a href="#">link</a>.</p>
------------------	--

<b>Memory Usage</b>	<p><b>Gauge chart</b> This chart displays the currently used total percentage of memory, as well as the total memory available in byte. Memory used for virtualization is not included.</p> <p><b>Line graph</b> The line graph displays the total amount of memory used over time. Data is displayed scaled over time. Memory used for virtualization is not included.</p>
---------------------	---

<b>Used Disk Space</b>	This graph displays the currently used total percentage of disk space on the host, as well as the total disk space available in byte.
------------------------	---

<b>Inbound Traffic</b>	This is the current amount of incoming data, as well as the total amount of data transferred since the last reboot.
------------------------	---

<b>Outbound Traffic</b>	This is the current amount of outgoing data, as well as the total amount of data transferred since the last reboot.
-------------------------	---

<b>Packetloss</b>	Here the number of lost incoming packets and lost outgoing packets is displayed.
-------------------	--

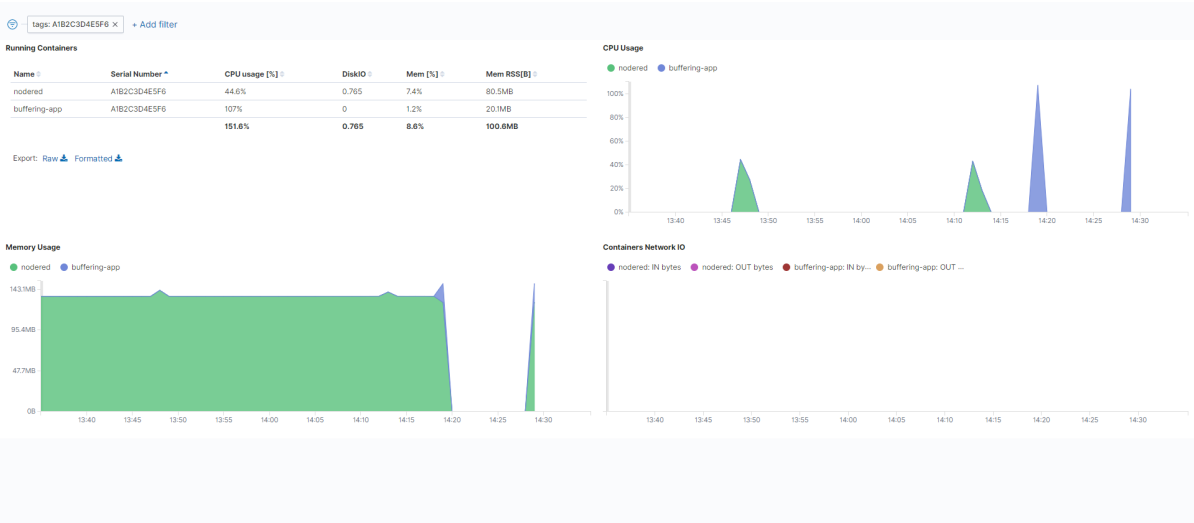
<b>Disk IO</b>	This graph displays the amount of reads and writes on the disk. Reads show how much data per second has been read while writes show the amount of data that has been saved or deleted.
----------------	--



Item	Description
<b>Inbound Traffic by Interface</b>	This is the amount of incoming data over time. The list to the right of the graph shows the average amount of traffic per interface. Note that this list also includes internal interfaces in this version.
<b>Outbound Traffic by Interface</b>	This is the amount of outgoing data over time. The list to the right of the graph shows the average amount of traffic per interface. Note that this list also includes internal interfaces in this version.

## Docker workload monitoring

Metadata of the overall state of Docker workloads is gathered in the Management System. A list of installed containers and their resource utilization is displayed in this dashboard. Toggle the **DOCKER WORKLOAD MONITORING** slider to enable the tracking of Docker workload data.



Item	Description
<b>Running Containers</b>	This is a list of user-installed Docker containers with details.
	<p><b>Name</b> This is the name of the Docker container as defined with the <b>Container name</b> setting when provisioning the workload.</p>
	<p><b>Serial Number</b> This is the serial number of the current node.</p>
	<p><b>CPU usage [%]</b> This is an average value of how much of the total CPU a Docker workload has used.</p>
	<p><b>DiskIO</b> This is the sum of reads and writes over the defined timespan.</p>
	<p><b>Mem [%]</b> This is an average percentage of how much of the total memory a Docker workload has used.</p>
	<p><b>Mem RSS[B]</b> This is an average value of how much resident set size (RSS) memory a Docker workload has used. Refer to this <a href="#">link</a> for a general explanation on RSS.</p>
<b>CPU Usage</b>	This is a graph of CPU usage in percentage over time. Note that the percentages here are in relation to the total amount of available CPU. Also, the display behaves according to standard Kibana behavior, meaning that CPU usage might be displayed as being at zero even though the CPU is busy. This is due to the graph showing only new data coming in and disregarding values that stay constant over a certain amount of time.
<b>Containers Network IO</b>	This is a graph showing the incoming and outgoing data for each container over time. Inbound and outbound traffic are marked separately per container.
<b>Memory Usage</b>	This is a graph of the total amount of memory used over time.

## Docker workload logging

The logs of the Docker workloads on a node are collected in the centralized logging system, allowing the analysis of logs from multiple workloads and nodes. Logs are collected from the standard Linux streams stdout (for debug messages) and stderr (for error messages). So for user created workloads this means that logs need to be sent to these streams to be collected. Note that the logs are most suitable to be read by developers with expert knowledge and should also be configured by developers. Toggle the **DOCKER WORKLOAD LOGGING** slider to enable the tracking of Docker workload logs.

To display logs of a certain workload, collected logs can be filtered in the Docker workload logging dashboard in Kibana.

tags: A1B2C3D4E5F6 X tags: docker X NOT container.name: is one of nerve-ds-node\_timescaledb\_1, nerve-ds-node\_supervisor-be\_1, nerve-ds-node\_grafana\_1, nerve-ds-node\_gateway\_1 X + Add filter

Docker Workload Logs 1-50 of 4390 < >

Time	@timestamp	container.name	message	syslog.severity_label
> May 26, 2021 @ 14:15:29.432	May 26, 2021 @ 14:15:29.432	-	CRITICAL:root:invalid configuration file. Exiting...	-
> May 26, 2021 @ 14:15:29.432	May 26, 2021 @ 14:15:29.432	-	[Errno 2] No such file or directory: '/nerve/config/config.json'	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	-	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	-	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	Nerve DS - influxDB Buffering	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	© TTTech-Industrial 2021	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	-----	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	v1.0 - 2021-05-26 12:15:29	-
> May 26, 2021 @ 14:15:29.427	May 26, 2021 @ 14:15:29.427	-	-	-
> May 26, 2021 @ 14:15:00.150	May 26, 2021 @ 14:15:00.150	-	2021-05-26 12:15:00.150 UTC [41] LOG: cron job 1 completed: 1 row	-
> May 26, 2021 @ 14:15:00.107	May 26, 2021 @ 14:15:00.107	-	2021-05-26 12:15:00.106 UTC [41] LOG: cron job 3 completed: 1 row	-
> May 26, 2021 @ 14:15:00.098	May 26, 2021 @ 14:15:00.098	-	2021-05-26 12:15:00.098 UTC [41] LOG: cron job 2 completed: 1 row	-
> May 26, 2021 @ 14:15:00.022	May 26, 2021 @ 14:15:00.022	-	2021-05-26 12:15:00.021 UTC [41] LOG: cron job 3 starting: SELECT dis_data_retention()	-
> May 26, 2021 @ 14:15:00.014	May 26, 2021 @ 14:15:00.014	-	2021-05-26 12:15:00.014 UTC [41] LOG: cron job 1 starting: SELECT check_space()	-
> May 26, 2021 @ 14:15:00.007	May 26, 2021 @ 14:15:00.007	-	2021-05-26 12:15:00.007 UTC [41] LOG: cron job 2 starting: SELECT data_buff_data_retention()	-
> May 26, 2021 @ 14:14:26.687	May 26, 2021 @ 14:14:26.687	-	[Errno 2] No such file or directory: '/nerve/config/config.json'	-
> May 26, 2021 @ 14:14:26.687	May 26, 2021 @ 14:14:26.687	-	CRITICAL:root:invalid configuration file. Exiting...	-
> May 26, 2021 @ 14:14:26.683	May 26, 2021 @ 14:14:26.683	-	-	-

# Device Guide

## Device Guide

The device guide is an extension of the user guide. It gives an overview of supported Nerve Devices and the device specific information that is required for operating Nerve software.

Each device chapter includes the following information:

- Links to documentation material from the manufacturer
- Hardware setup for getting the device Nerve ready
- A guide for installing Nerve on the device
- First steps after the installation
- Overview of physical ports and node internal networking

All devices share the base functionality of Nerve but differ in the extent of functionality.

### NOTE

The device guide chapter for the MFN 100 contains more information as it is the flagship device for Nerve. Nerve can be used to its full extent when operating on the MFN 100.

## MFN 100



The MFN 100 is a qualified Nerve Device that is optimized and tested for use with Nerve software. The device is designed for use in harsh industrial environments (-40°C to +70°C). It is based on an Intel Atom x5-E3940/50 CPU and offers 4 GB/8 GB RAM and up to 512 GB SSD storage. The MFN 100 offers one I/O port for Ethernet-based fieldbus connectivity, four GbE switch ports and one SFP port. Additional interfaces include two USB 2.0 ports and one DisplayPort.

## Technical data

<b>CPU</b>	<b>Intel E3940</b> 4 cores, 1.8 GHz, 4 GB RAM
	<b>Intel E3950</b> 4 cores, 2.0 GHz, 8 GB RAM
<b>Storage</b>	64 GB SSD MLC 256 GB SSD MLC 512 GB SSD MLC
<b>Performance</b>	1 ms control cycle time achievable with Nerve

<b>Interfaces</b>	<ul style="list-style-type: none"> <li>• 4 x RJ 45 Ethernet (1000/100/10 Mbit/s)</li> <li>• 1 x SFP (1000 Mbit/s) Optical transceivers / OFCS modules may be used which are in compliance with Class I device acc. 21 CFR 1040 and IEC/EN 60825-1</li> <li>• 1 x DP++</li> <li>• 2 x USB 2.0 1 A combined current</li> </ul>
<b>Mounting</b>	DIN rail or wall mount
<b>Dimensions</b>	(h x w x d): 179 x 87 x 143 mm
<b>Weight</b>	2.1 kg
<b>Power</b>	2 x 24 V redundant input, Average power consumption 12 W
<b>Environmental Parameters</b>	<ul style="list-style-type: none"> <li>• Operating Temperature Range: -40°C to 70°C</li> <li>• Shock and Vibration: ISO 60068-2-27: 15 g peak, 11 ms ISO 60068-2-6: 5 Hz ≤ f &lt; 8.4 Hz: 3.5 mm, 8.4 Hz ≤ f ≤ 150 Hz: 1.0 g</li> <li>• IP 40 according to IEC 60529</li> <li>• Indoor use only, intended for use in control cabinets</li> <li>• Use up to pollution degree 3</li> <li>• Use only in environments where no condensation will occur</li> <li>• Maximum relative humidity: 80% for temperatures up to 31°C decreasing linearly to 50% relative humidity at 40°C</li> <li>• Maximum altitude: 2000 msl</li> </ul>
<b>Certificates</b>	CE and UL certified (EN 61000-6-2/4, IEC/UL 61010, CSA C22.2 NO. 61010-1-12)

## Identifying the MFN 100

The label of the MFN 100 can be found on the back of the device, close to the DIN rail clip. Exact identification is possible through the combination of product number (P/N), serial number (S/N) and version number (V/N) that are printed on the label. The model number of the MFN 100 details the variant of the MFN 100:

Letter or Number	Description
<b>CODESYS indicator</b>	<p>This letter indicates whether the device has a CODESYS runtime pre-configured:</p> <ul style="list-style-type: none"> <li>• <b>C</b> — The CODESYS runtime is pre-configured</li> <li>• <b>X</b> — The CODESYS runtime is not pre-configured</li> </ul>

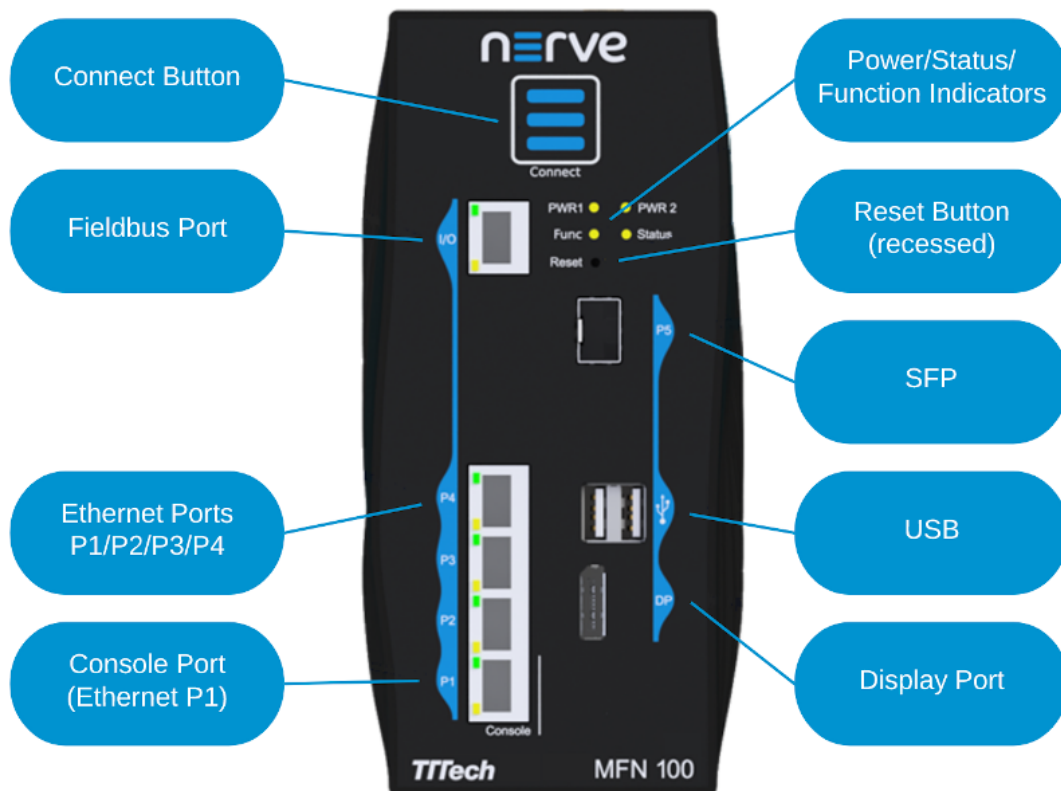
Letter or Number	Description
<b>SSD size</b>	This number indicates the size of the SSD:
	<ul style="list-style-type: none"> <li>• <b>6</b> — 64 GB SSD</li> <li>• <b>2</b> — 256 GB SSD</li> <li>• <b>5</b> — 512 GB SSD</li> </ul>

This indicates the CPU variant of the device:

<b>CPU variant</b>	• <b>4</b> — Intel E3940 (4 GB RAM)
	• <b>5</b> — Intel E3950 (8 GB RAM)

## Front panel controls and indicators

Below is an overview of the front panel of the MFN 100, describing physical interfaces, indicators and their labels.



Label	Description
<b>Connect Button</b>	The connect button interrupts the connection on ports P2 to P5 of the MFN 100. This is the behavior in the standard configuration. The function is configurable on request. The button may be configured to change the network configuration.

Label	Description
<b>Connection Indicator</b>	The connection indicator is the first fin in the MFN 100 housing. It lights up blue when all required services are initiated and the connection to the Management System is configured.
<b>Reset</b>	Holding the button for 4-8 seconds initiates a power cycle. Use a tool with a rounded tip to press the button.
<b>Power 1 Power 2</b>	Indicators showing power active on the power supply.  LED indicating system status
<b>Status</b>	<ul style="list-style-type: none"> <li>• Green: All device functions are ready.</li> <li>• Not lit: Device functions are not ready or the device is booting.</li> </ul>
<b>Function</b>	<p>LED indicating CODESYS runtime status</p> <ul style="list-style-type: none"> <li>• Green: CODESYS runtime is operational.</li> <li>• Not lit: CODESYS runtime is not operational.</li> </ul>
<b>P1 Console</b>	Ethernet port/console port. This port is typically used to connect a workstation to configure the MFN 100.
<b>P2/P3/P4</b>	Ethernet ports
<b>P5</b>	SFP port
<b>I/O</b>	Fieldbus interface
<b>USB</b>	Two USB 2.0 ports with 1.1 A maximum output current for both ports combined.
<b>DP</b>	DisplayPort supporting the DP++ standard.

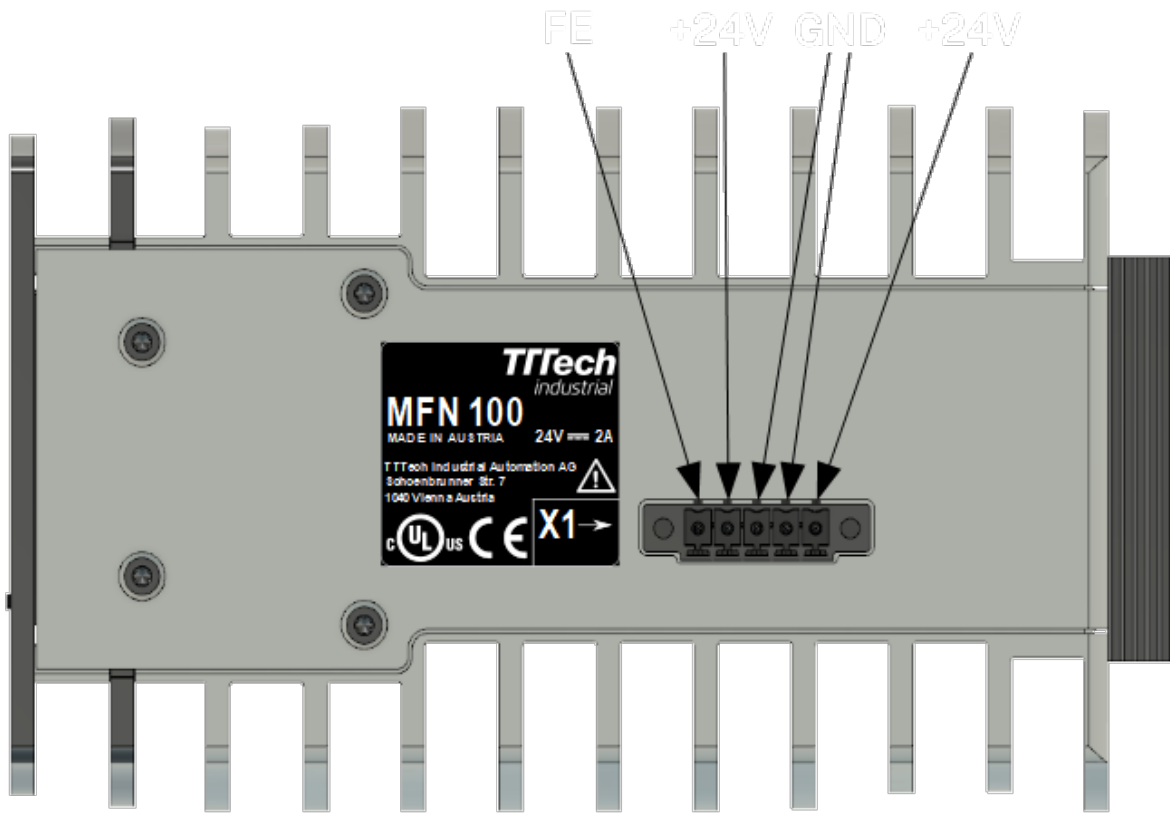
## NOTE

**P1** to **P5** are internally connected to a hardware switch and separated via VLAN tagging. This has two implications.

- Tagged VLAN frames cannot be used to communicate with any entity executed on the MFN 100.
- Do not connect two or more interfaces of the ports **P1** to **P5** to the same switch (nor to the same Ethernet subnet), as this creates a loop.

## Power connectors overview

The power connectors are located at the bottom of the MFN 100 next to the label. There are two separate 24 V inputs, two GND inputs and one Functional Earth (FE) input. The inputs are fused internally. The fuse cannot be replaced by the user. The power supply inputs are protected against reverse polarity.



Pin	Description
1	Power supply line 2
2	GND
3	GND
4	Power supply line 1
5	Functional Earth (FE)

#### NOTE

The GND and FE pins (pins 2, 3, and 5) are electrically connected to the housing.

### Power supply details

Parameter	Value
<b>Operating voltage</b>	18 - 30 VDC
<b>Start-up current</b>	7 A max.
<b>Consumption</b>	1.4 A continuous 2.1 A peak
<b>Dissipated power</b>	33.6 W at 24 VDC



## Installation and removal on a DIN rail

The MFN 100 is intended for mounting on a DIN rail inside a closed cabinet. Due to its weight it should be installed on a strong DIN rail. No tool is required to install or remove the MFN 100.

Follow these steps to install the MFN 100 on a DIN rail:

1. Engage the DIN rail mounting clip of the MFN 100 with the upper edge of the DIN rail.
2. Push the MFN 100 down into the DIN rail.
3. Place the MFN 100 in a vertical position so that the mounting clip engages the lower edge of the DIN rail.

Follow these steps to remove the MFN 100 from a DIN rail:

1. Push the MFN 100 down.
2. Rotate the MFN 100 upwards so that the lower edge of the DIN rail disengages.
3. Lift the MFN 100 slightly to remove it.

## Setting up the MFN 100

When delivered, Nerve is already installed on the MFN 100. Two network cables and a +24 V DC power supply are required to finish the setup and use Nerve on the MFN 100. This includes connecting the power supply to the mating connector which is delivered with the MFN 100.

1. Connect pin 1 of the mating connector to +24 V DC.
2. Connect pin 2 of the mating connector to GND.
3. Plug the mating connector into the bottom side of the MFN 100.
4. Connect port 2 of the MFN 100 to a DHCP-enabled network.

### NOTE

Port 2 is used for communication with the Management System. Make sure to connect the MFN 100 to the correct network, depending on whether the Management System is hosted on premise or by TTTech Industrial.

5. Plug in the power supply.

The MFN 100 will start after a few minutes and light up blue when all necessary services are initiated.

### NOTE

- Contact the IT administrator for help on how to allow external devices to connect to the network.
- To connect the MFN 100 to a fieldbus, connect a network cable to the I/O port of the MFN 100 and to a fieldbus interface.
- A second power supply can also be connected to the MFN 100 as a backup. To do so, connect pin 3 of the mating connector to GND and connect pin 4 of the mating connector to +24 V DC.

## Activating the Nerve license

The product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **P1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask.

### NOTE

Do not use 172.20.2.27 for the network adapter IP address. This IP address is used internally by the Nerve system.

Access the Local UI at <http://172.20.2.1:3333/> and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **P1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask. The credentials for the Local UI found in the customer profile are also required.

### NOTE

Do not use 172.20.2.27 for the network adapter IP address. This IP address is used internally by the Nerve system.

1. Follow this link to connect to the Local UI: <http://172.20.2.1:3333/>
2. Log in with the credentials from the customer profile.



Sign In

Username  
local@nerve.cloud

Password  
.....

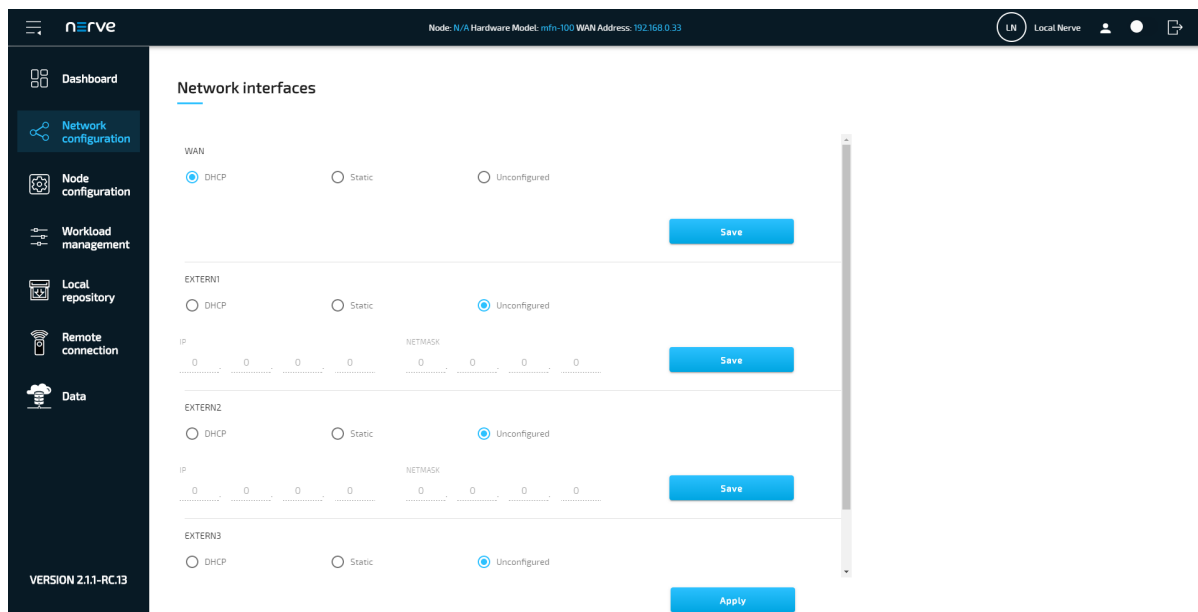
Sign In



Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Network configuration

The Ethernet ports of the Nerve Devices can be configured from the Local UI. For the MFN 100, the interfaces in the Local UI represent the physical ports 2, 3, 4 and 5. The console port **P1** and the I/O port of the MFN 100 are reserved and cannot be modified. The console port is used solely for configuration purposes. The I/O port is connected to the CODESYS runtime and used for fieldbus communication. Select **Network configuration** in the navigation on the left to reach this menu.



## CODESYS related information

For working with the CODESYS Development System, a device description for Nerve Devices is required. The device description can be downloaded from the [Nerve Software Center](#).

The MFN 100 has an Ethernet port that is reserved for machine data acquisition. Connect a network cable to the I/O port of the MFN 100 and to a fieldbus interface to acquire machine data. The CODESYS runtime can be reached at 172.20.2.2.

## Physical ports and network interfaces

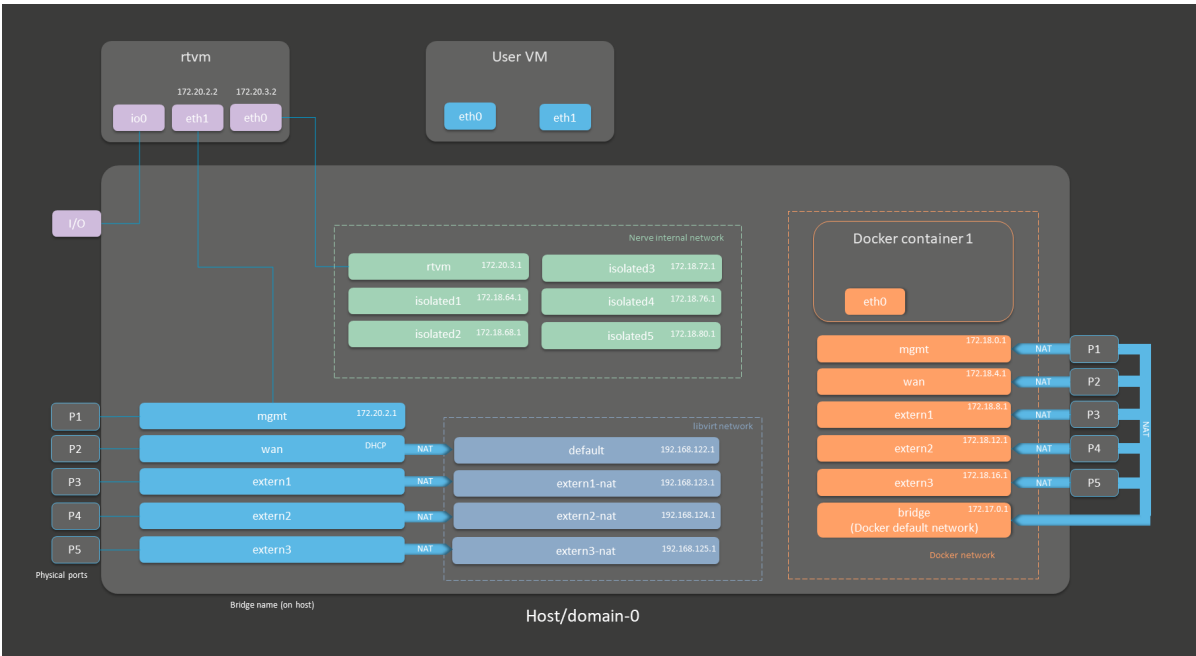
Below is a depiction of the node internal networking for the MFN 100. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the MFN 100.

Physical port	Network name
I/O	io0
P1	mgmt
P2	wan
P3	extern1
P4	extern2

Physical port	Network name
P5	extern3

Below is a graphic that details the available interfaces of the MFN 100 for use with Nerve. Pictured is how the physical interfaces translate to the Host and the CODESYS runtime.

The image shows an example node and how the physical interfaces translate to the Host and the CODESYS runtime. The node consists of the **host/domain-0** and the real-time VM running the CODESYS runtime (labeled **rtvm**). It also has one Virtual Machine workload and two Docker workloads deployed. The virtual machine is located outside of the host and the Docker containers are located in the Docker network inside of the host. However, the workloads are not yet connected.



Notable IP Adresses	
Host access	172.20.2.1
CODESYS runtime access	172.20.2.2

Refer to [Node internal networking](#) for more information on networking in the Nerve system.

## Updating Nerve from version 2.0 to 2.1

As updating nodes to newer versions through the Management System is a feature introduced in version 2.1, updates from version 2.0 to version 2.1 have to be performed manually. However, before updating the node, contact the sales representative or customer support for information on how to backup current data.

Requirements for updating Nerve on the MFN 100:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the MFN 100.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press F7 when the device is booting to enter the boot menu.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img`. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Power on the device.
6. Press F7 to enter the boot menu. Make sure that the device will boot from the USB drive.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_mfn-100.img bs=4M of=/dev/sd<drivena
sync
```

#### NOTE

Make sure to replace `<drivename>` with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Kontron KBox A-150-APL



The KBox A-150-APL is an industrial computer platform for process control and optimization with the Intel Atom series processors. It offers DIN Rail mounting positions in limited space.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@tttech-industrial.com](mailto:sales@tttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	EN00-03002-01
<b>CPU</b>	Intel Atom E3950
<b>Cores</b>	4
<b>RAM</b>	4 GB DDR3
<b>Storage</b>	128 GB 2.5" SATA SSD
<b>TPM</b>	TPM 2.0 included

Item	Description
<b>Interfaces</b>	<ul style="list-style-type: none"> <li>• 2x GB LAN</li> <li>• 2x RS232/422/485</li> <li>• 4x USB</li> <li>• 1x DisplayPort</li> <li>• 1x HDMI</li> </ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, update the BIOS on the KBox A-150-APL to the latest version. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Advanced &gt; CPU Chipset Configuration &gt; EIST</b>	<b>Disabled</b>
<b>Advanced &gt; CPU Chipset Configuration &gt; Active Processor Cores</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Chipset Configuration &gt; Intel Virtualization Technology</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Chipset Configuration &gt; VT-d</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Chipset Configuration &gt; C-States</b>	<b>Disabled</b>
<b>Advanced &gt; Network Stack &gt; Network Stack</b>	<b>Enabled</b>
<b>Security &gt; Secure Boot &gt; Attempt Secure Boot</b>	<b>Disabled</b>

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the KBox A-150-APL.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img` file. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-150-apl.img bs=4M of=
sync
```

#### NOTE

Make sure to replace `<drivename>` with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.



4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Finding out the IP address of the device

Due to the limited availability of ethernet ports, Nerve does not offer a designated port and interface for host access and management purposes on the Kontron KBox A-150-APL. The IP address of the **wan** interface that is mapped to physical port **LAN 1** is required to start using Nerve. Depending on the network access the node has, this needs to be done differently.

### The node has network access

If the node has network access, an IP address will be assigned to the **wan** interface by a DHCP server. Follow the instructions below to find out the IP address of the **wan** interface.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following command to display the IP address of the **wan** interface:

```
ip a s wan
```

The IP address is displayed next to **inet** in the output the system gives. This IP address is required to access the Local UI in the instructions below.

### The node does not have network access

In case of the node not having network access, the IP address of the **wan** interface has to be set manually. For simplicity, the IP address of the **wan** interface will be set to 172.20.2.1 — the IP address of the host.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following commands to open the **wan** interface configuration:

```
cd /etc/network/interfaces.d  
sudo nano wan
```

5. Enter the host access password if prompted.
6. Edit the configuration the following way:

```
auto wan  
iface wan inet static  
    bridge_ports eth0  
    address 172.20.2.1  
    netmask 255.255.255.0
```

7. Enter Ctrl+S to save the configuration.
8. Enter Ctrl+X to exit the Nano editor.
9. Enter the following command to apply the changes to the **wan** interface by restarting the networking services:  

```
/etc/init.d/networking restart
```
10. Enter the host access password if prompted.

With the **wan** interface IP address set to 172.20.2.1 the Local UI can be reached at <http://172.20.2.1:3333/>.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN 1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address.

Access the Local UI at <wanip>:3333 in a web browser and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN 1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address. The credentials for the Local UI found in the customer profile are also required.

1. Access the Local UI at <wanip>:3333 in a web browser.
2. Log in with the credentials from the customer profile.

Sign In

Username  
local@nerve.cloud

Password  
.....

Sign in

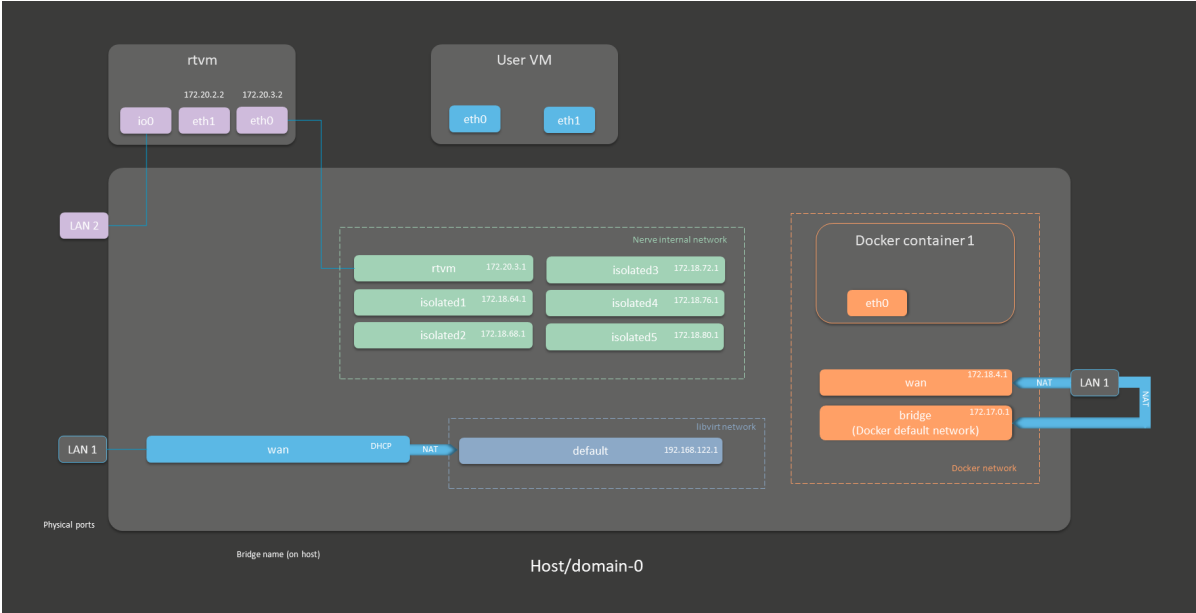


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the KBox A-150-APL hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the KBox A-150-APL.

Physical port	Network name
LAN 1	wan
LAN 2	io0



## Kontron KBox A-250



Based on a pITX-2.5" SBC with Intel Atom processors of the E3900 family, the fanless KBox A-250 has been designed as a gateway for IoT-Edge applications. Its fields of application are primarily found in industrial automation.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@ttech-industrial.com](mailto:sales@ttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	2-A0DM-009
<b>CPU</b>	Intel Atom E3950
<b>Cores</b>	4
<b>RAM</b>	4 GB DDR3L
<b>Storage</b>	128 GB M.2 MLC
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"> <li>• 2x GB LAN</li> <li>• 1x RS232/422/485</li> <li>• 2x USB</li> <li>• 1x DisplayPort</li> </ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, update the BIOS on the KBox A-250 to the latest version. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Advanced &gt; ACPI Settings &gt; Enable ACPI Auto Configuration</b>	<b>Disabled</b>
<b>Advanced &gt; ACPI Settings &gt; Enable Hibernation</b>	<b>Disabled</b>
<b>Advanced &gt; ACPI Settings &gt; ACPI Sleep State</b>	<b>Suspend Disabled</b>
<b>Advanced &gt; CPU Configuration &gt; Turbo Mode</b>	<b>Disabled</b>
<b>Advanced &gt; CPU Configuration &gt; Intel Virtualization Technology</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Configuration &gt; VT-d</b>	<b>Enabled</b>
<b>Advanced &gt; Network Stack Configuration &gt; Network Stack</b>	<b>Enabled</b>
<b>Advanced &gt; CSM Configuration &gt; CSM Support</b>	<b>Disabled</b>
<b>Advanced &gt; System Component &gt; DDR SSC</b>	<b>Disable</b>
<b>Advanced &gt; System Component &gt; HighSpeed SerialIO SSC</b>	<b>Disable</b>
<b>Chipset &gt; South Bridge &gt; OS Selection</b>	<b>Intel Linux</b>
<b>Chipset &gt; South Bridge &gt; Real Time Option</b>	<b>RT Enabled, Agent Disabled</b>

Path	Setting
<b>Security &gt; Secure Boot &gt; Attempt Secure Boot</b>	<b>Disabled</b>

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the KBox A-250.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img` file. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_kontron-kbox-a-250.img bs=4M of=/dev/
sync
```

#### NOTE

Make sure to replace <drivename> with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Finding out the IP address of the device

Due to the limited availability of ethernet ports, Nerve does not offer a designated port and interface for host access and management purposes on the Kontron KBox A-250. The IP address of the **wan** interface that is mapped to physical port **ETH 2** is required to start using Nerve. Depending on the network access the node has, this needs to be done differently.

### The node has network access

If the node has network access, an IP address will be assigned to the **wan** interface by a DHCP server. Follow the instructions below to find out the IP address of the **wan** interface.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following command to display the IP address of the **wan** interface:

```
ip a s wan
```

The IP address is displayed next to **inet** in the output the system gives. This IP address is required to access the Local UI in the instructions below.

### The node does not have network access

In case of the node not having network access, the IP address of the **wan** interface has to be set manually. For simplicity, the IP address of the **wan** interface will be set to 172.20.2.1 — the IP address of the host.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following commands to open the **wan** interface configuration:

```
cd /etc/network/interfaces.d  
sudo nano wan
```

5. Enter the host access password if prompted.
6. Edit the configuration the following way:

```
auto wan
iface wan inet static
    bridge_ports eth0
    address 172.20.2.1
    netmask 255.255.255.0
```

7. Enter Ctrl+S to save the configuration.
8. Enter Ctrl+X to exit the Nano editor.
9. Enter the following command to apply the changes to the **wan** interface by restarting the networking services:

```
/etc/init.d/networking restart
```

10. Enter the host access password if prompted.

With the **wan** interface IP address set to 172.20.2.1 the Local UI can be reached at <http://172.20.2.1:3333/>.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **ETH 2** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address.

Access the Local UI at <wanip>:3333 in a web browser and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **ETH 2** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address. The credentials for the Local UI found in the customer profile are also required.

1. Access the Local UI at <wanip>:3333 in a web browser.
2. Log in with the credentials from the customer profile.



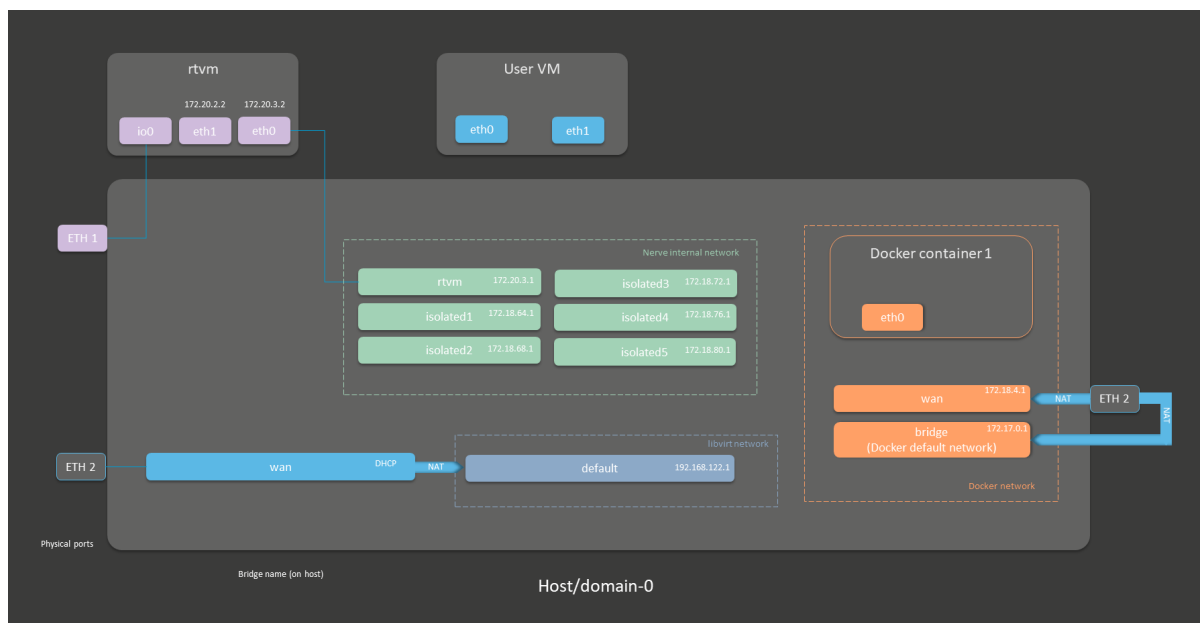


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the KBox A-250 hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the KBox A-250.

Physical port	Network name
ETH 2	wan
ETH 1	io0



# Maxtang AXWL10-8665U



The AXWL-10 is a small form factor fanless embedded system, which makes it ideal for space-sensitive applications. This Whiskey Lake based system is designed for various application as, but not limited to, automation, network security, communication and transportation, health-care, and retail.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [Datasheet](#)

Note that the manufacturer did not offer a dedicated manual at the time of writing. The device is based on the WL10 family of motherboards. Refer to the [Manuals](#) section on the manufacturers homepage for materials on the WL10 family or contact a Maxtang representative for more information.

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@ttech-industrial.com](mailto:sales@ttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	AXWL10-8665U
<b>CPU</b>	Intel Core i7-8665U
<b>Cores</b>	4
<b>RAM</b>	32 GB
<b>Storage</b>	256 GB

Item	Description
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"> <li>• 2x GB LAN</li> <li>• 5x RS232</li> <li>• 1x RS485</li> <li>• 3x USB 3.0</li> <li>• 2x USB 2.0</li> <li>• 8x GPIO</li> <li>• 5x COM</li> <li>• 1x DisplayPort</li> <li>• 1x HDMI 1.4</li> <li>• 1x VGA</li> </ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a DisplayPort, HDMI or VGA input
- a keyboard
- a USB drive formatted to FAT32

Refer to the materials of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, update the Setup Utility on the AXWL10-8665U to V2.20.1271 or later. Refer to the materials of the manufacturer for information on how to update the BIOS and Setup Utility versions.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Go to **Setup Utility**.
4. Change the following settings:

Path	Setting
<b>Advanced &gt; CPU Configuration &gt; Intel (VMX) Virtualization Technol</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Configuration &gt; Hyper-Threading</b>	<b>Disabled</b>
<b>Advanced &gt; CPU Configuration &gt; Active Processor Cores</b>	<b>All</b>
<b>Advanced &gt; CPU Configuration &gt; C states</b>	<b>Disabled</b>

Path	Setting
<b>Advanced &gt; Network Stack Configuration &gt; Network Stack</b>	<b>Enabled</b>
<b>Chipset &gt; System Agent (SA) Configuration &gt; VT-d</b>	<b>Enabled</b>
To enable boot from USB stick from all USB sockets:	
<b>Advanced &gt; USB Configuration &gt; Legacy USB Support</b>	<b>Enabled</b>
<b>Chipset &gt; PCH-IO Configuration &gt; USB Configuration &gt; USB Port Override</b>	<b>Select Per-Pin</b>

5. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort, HDMI or VGA input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the AXWL10-8665U.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img` file. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img.tar.gz` file from the [Nerve Software Center](#).

- Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_maxtang-axwl10-8665u.img bs=4M of=/dev/<drivename>
sync
```

#### NOTE

Make sure to replace `<drivename>` with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Finding out the IP address of the device

Due to the limited availability of ethernet ports, Nerve does not offer a designated port and interface for host access and management purposes on the Maxtang AXWL10-8665U. The IP address of the **wan** interface that is mapped to physical port **LAN1** is required to start using Nerve. Depending on the network access the node has, this needs to be done differently.

### The node has network access

If the node has network access, an IP address will be assigned to the **wan** interface by a DHCP server. Follow the instructions below to find out the IP address of the **wan** interface.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following command to display the IP address of the **wan** interface:

```
ip a s wan
```

The IP address is displayed next to **inet** in the output the system gives. This IP address is required to access the Local UI in the instructions below.

### The node does not have network access

In case of the node not having network access, the IP address of the **wan** interface has to be set manually. For simplicity, the IP address of the **wan** interface will be set to `172.20.2.1` — the IP address of the host.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.

3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.

4. Enter the following commands to open the **wan** interface configuration:

```
cd /etc/network/interfaces.d
sudo nano wan
```

5. Enter the host access password if prompted.

6. Edit the configuration the following way:

```
auto wan
iface wan inet static
    bridge_ports eth0
    address 172.20.2.1
    netmask 255.255.255.0
```

7. Enter Ctrl+S to save the configuration.

8. Enter Ctrl+X to exit the Nano editor.

9. Enter the following command to apply the changes to the **wan** interface by restarting the networking services:

```
/etc/init.d/networking restart
```

10. Enter the host access password if prompted.

With the **wan** interface IP address set to 172.20.2.1 the Local UI can be reached at <http://172.20.2.1:3333/>.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address.

Access the Local UI at <wanip>:3333 in a web browser and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address. The credentials for the Local UI found in the customer profile are also required.

1. Access the Local UI at <wanip>:3333 in a web browser.
2. Log in with the credentials from the customer profile.

[Sign In](#)

Username  
local@nerve.cloud

Password

[Sign In](#)

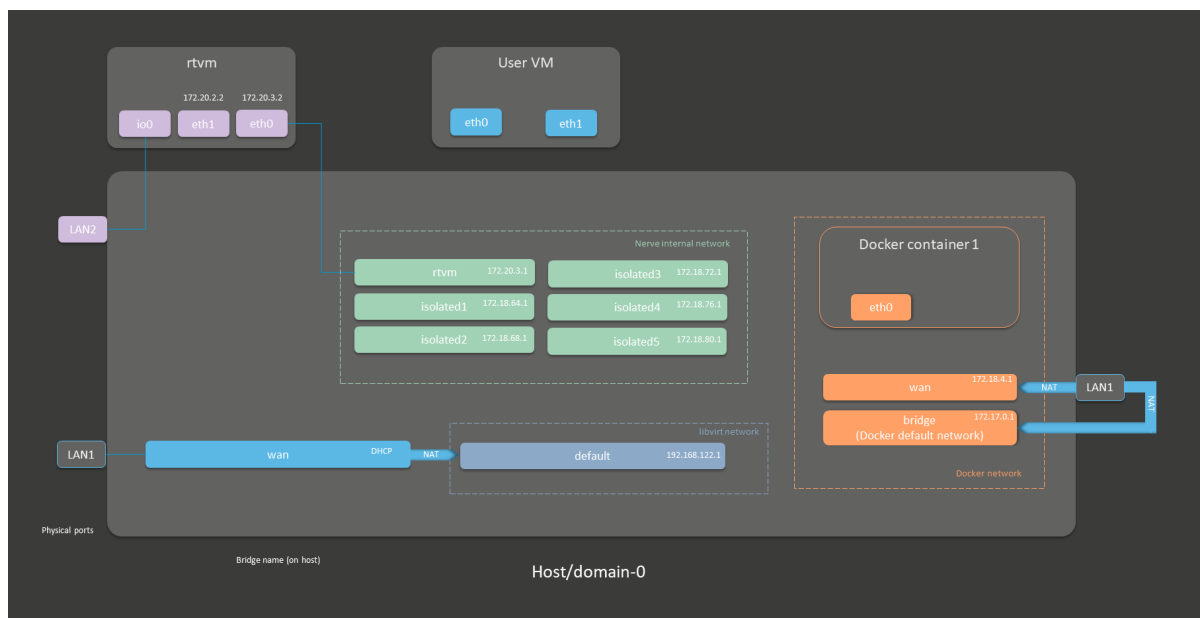


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the AXWL10-8665U hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the AXWL10-8665U.

Physical port	Network name
LAN1	wan
LAN2	io0



# Siemens SIMATIC IPC127E



The SIMATIC IPC127E can be integrated into a control cabinet or directly on the machine with minimal space requirements to record, collect, process, and transfer data directly in the production environment.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the



device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@ttech-industrial.com](mailto:sales@ttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	6AG4021-0AB12-1CA0 /SIMATIC IPC127
<b>CPU</b>	Intel Atom E3940
<b>Cores</b>	4
<b>RAM</b>	4 GB
<b>Storage</b>	128 GB
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"><li>• 3x GB LAN</li><li>• 4x USB 3.0</li><li>• 1x DisplayPort</li></ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, the BIOS version on the SIMATIC IPC127E needs to be version V27.01.03. or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Esc while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Power &gt; Advanced CPU Control &gt; VT-d</b>	<b>Enabled</b>
<b>Power &gt; Advanced CPU Control &gt; Active Processor Cores</b>	<b>Disabled</b>

Path	Setting
Power > Advanced CPU Control > C states	Disabled
Power > Advanced CPU Control > User Power Scenario	Max Performance
Power > Advanced CPU Control > Turbo Mode	Disabled
Boot > Network Stack	Enabled
Boot > USB Boot	Enabled

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the SIMATIC IPC127E.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Esc when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img`. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img.tar.gz` file from the [Nerve Software Center](#).

- Enter the following commands to extract the Nerve\_Blue\_USB-  
2. installer\_2.3.1\_for\_siemens-simatic-ipc127e.img.tar.gz file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc127e.img bs=4M of=
sync
```

#### NOTE

Make sure to replace <drivename> with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **X1 P1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask.

Access the Local UI at <http://172.20.2.1:3333/> and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **X1 P1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask. The credentials for the Local UI found in the customer profile are also required.

1. Follow this link to connect to the Local UI: <http://172.20.2.1:3333/>
2. Log in with the credentials from the customer profile.

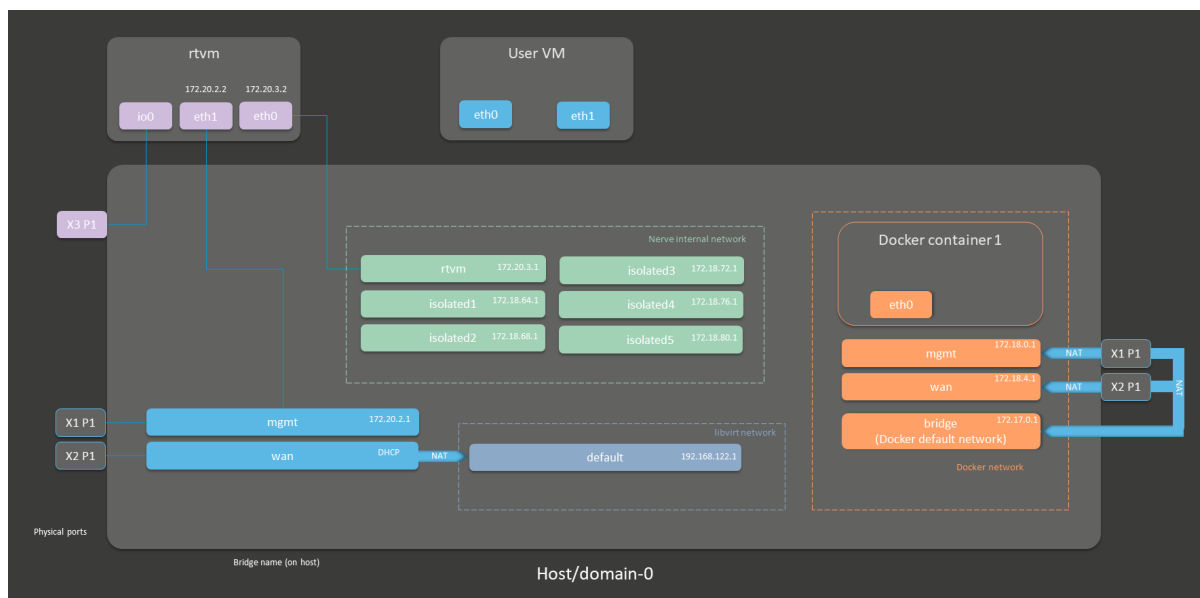


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

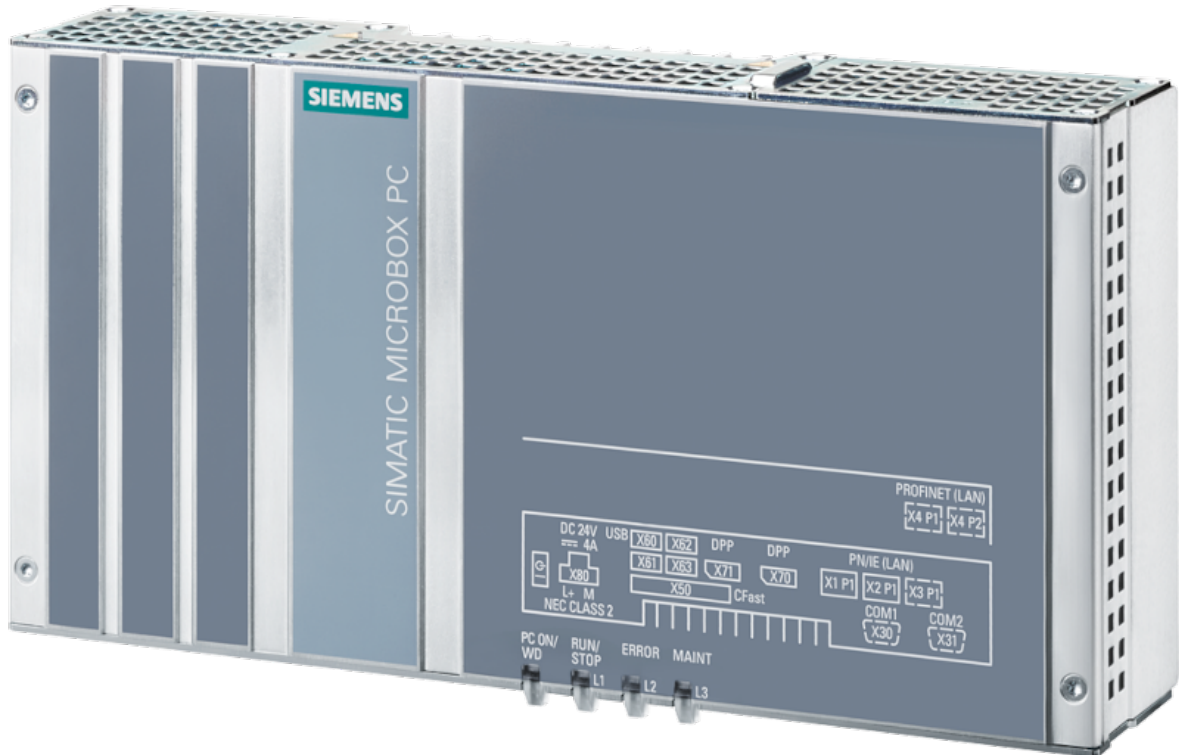
## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the SIMATIC IPC127E hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the SIMATIC IPC127E.

Physical port	Network name
X1 P1	mgmt
X2 P1	wan
X3 P1	io0



# Siemens SIMATIC IPC427E



The SIMATIC IPC427E is an embedded IPC with 6th generation Intel Core-i processors that can be used for control, data collection, or communication tasks at the machine or process level. It offers low mounting depth and different mounting possibilities, with all interfaces on one side for easy integration in existing plants.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@tttech-industrial.com](mailto:sales@tttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	6AG4141-5BC00-0GA8 /SIMATIC IPC427
<b>CPU</b>	Intel Core i5-6442EQ
<b>Cores</b>	4
<b>RAM</b>	16 GB
<b>Storage</b>	480 GB SATA SSD

Item	Description
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"> <li>• 3x GB LAN</li> <li>• 4x USB 3.0</li> <li>• 2x DisplayPort</li> </ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, the BIOS version on the SIMATIC IPC427E needs to be version 12 or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Esc while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Advanced &gt; System Agent (SA) Configuration &gt; VT-d</b>	<b>Enabled</b>
<b>Advanced &gt; Active Management Technology Support &gt; Intel AMT Configuration Screens</b>	<b>Disabled</b>
<b>Advanced &gt; Memory Configuration &gt; Max TOLUD</b>	<b>2 GB</b>
<b>Power &gt; CPU Configuration &gt; Intel (VMX) Virtualization Technology</b>	<b>Enabled</b>
<b>Power &gt; Power &amp; Performance &gt; CPU - Power Management Control &gt; CPU Power Level</b>	<b>Standard</b>
<b>Power &gt; Power &amp; Performance &gt; CPU - Power Management Control &gt; Intel (R) SpeedStep(tm)</b>	<b>Disabled</b>
<b>Power &gt; Power &amp; Performance &gt; CPU - Power Management Control &gt; Intel (R) Speed Shift Technology</b>	<b>Disabled</b>

Path	Setting
<b>Power &gt; Power &amp; Performance &gt; CPU - Power Management Control &gt; C states</b>	<b>Disabled</b>
<b>Boot &gt; Boot Type</b>	<b>Dual Boot Type</b>
<b>Boot &gt; Add Boot Options</b>	<b>Auto</b>
<b>Boot &gt; USB Boot</b>	<b>Enabled</b>
<b>Boot &gt; EFI Device First</b>	<b>Enabled</b>

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the SIMATIC IPC427E.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Esc when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img`. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img.tar.gz` file from the [Nerve Software Center](#).

- Enter the following commands to extract the Nerve\_Blue\_USB-  
2. installer\_2.3.1\_for\_siemens-simatic-ipc427e.img.tar.gz file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_siemens-simatic-ipc427e.img bs=4M of=
sync
```

#### NOTE

Make sure to replace <drivename> with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **X1 P1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask.

Access the Local UI at <http://172.20.2.1:3333/> and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **X1 P1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask. The credentials for the Local UI found in the customer profile are also required.

1. Follow this link to connect to the Local UI: <http://172.20.2.1:3333/>
2. Log in with the credentials from the customer profile.



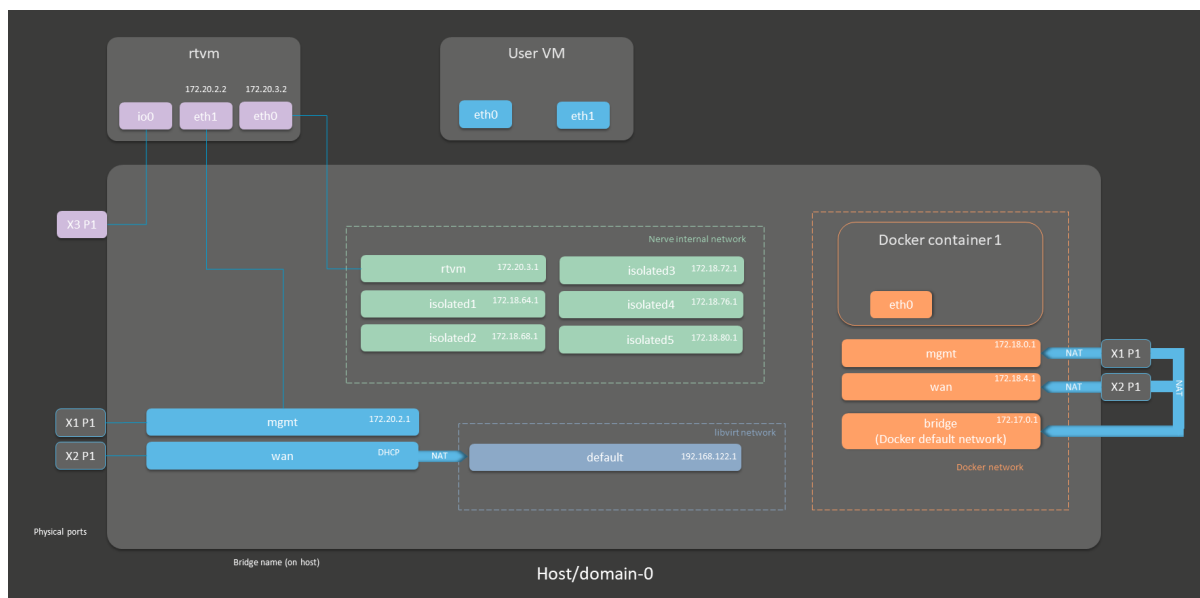


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the SIMATIC IPC427E hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the SIMATIC IPC427E.

Physical port	Network name
X1 P1	mgmt
X2 P1	wan
X3 P1	io0



# Supermicro SuperServer E100-9AP-IA



With an excellent price/performance ratio, the Supermicro Superserver devices offer a compact design with good connectivity, allowing optimal Nerve performance. The SuperServer E100-9AP-IA is a compact, Intel Atom class embedded IPC with many connection possibilities.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@ttech-industrial.com](mailto:sales@ttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	SYS-E100-9AP-IA
<b>CPU</b>	Intel Atom x5-E3940

Item	Description
<b>Cores</b>	4
<b>RAM</b>	8 GB
<b>Storage</b>	128 GB
<b>TPM</b>	TPM 1.2 and TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"> <li>• 2x GbE LAN ports via Intel I210-IT</li> <li>• 2x RS232</li> <li>• 2x RS232/422/485</li> <li>• 1x RS-485</li> <li>• 1x DIO via DB9</li> <li>• 4x USB 2.0</li> <li>• 2x USB 3.0</li> <li>• 1x VGA</li> <li>• 1x HDMI</li> </ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a VGA or HDMI input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, the BIOS version on the SuperServer E100-9AP-IA needs to be version 1.3 or later and Setup Utility needs to be V2.18.1263 or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Advanced &gt; CPU Configuration &gt; Intel Virtualization Technology</b>	<b>Enabled</b>
<b>Advanced &gt; Chipset Configuration &gt; VT-d</b>	<b>Enabled</b>

Path	Setting
<b>Advanced &gt; CPU Configuration &gt; Active Processor Cores</b>	<b>Enabled</b>
<b>Advanced &gt; ACPI Settings &gt; ACPI Sleep State</b>	<b>Suspend Disabled</b>

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a VGA or HDMI input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the SuperServer E100-9AP-IA.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img`. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.img.t
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-e100-9ap-ia.i
sync
```

#### NOTE

Make sure to replace <drivename> with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Finding out the IP address of the device

Due to the limited availability of ethernet ports, Nerve does not offer a designated port and interface for host access and management purposes on the Supermicro SuperServer E100-9AP-IA. The IP address of the **wan** interface that is mapped to physical port **LAN1** is required to start using Nerve. Depending on the network access the node has, this needs to be done differently.

### The node has network access

If the node has network access, an IP address will be assigned to the **wan** interface by a DHCP server. Follow the instructions below to find out the IP address of the **wan** interface.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following command to display the IP address of the **wan** interface:

```
ip a s wan
```

The IP address is displayed next to **inet** in the output the system gives. This IP address is required to access the Local UI in the instructions below.

### The node does not have network access

In case of the node not having network access, the IP address of the **wan** interface has to be set manually. For simplicity, the IP address of the **wan** interface will be set to 172.20.2.1 — the IP address of the host.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following commands to open the **wan** interface configuration:

```
cd /etc/network/interfaces.d
sudo nano wan
```

5. Enter the host access password if prompted.
6. Edit the configuration the following way:

```
auto wan
iface wan inet static
    bridge_ports eth0
    address 172.20.2.1
    netmask 255.255.255.0
```

7. Enter Ctrl+S to save the configuration.
8. Enter Ctrl+X to exit the Nano editor.
9. Enter the following command to apply the changes to the **wan** interface by restarting the networking services:

```
/etc/init.d/networking restart
```

10. Enter the host access password if prompted.

With the **wan** interface IP address set to 172.20.2.1 the Local UI can be reached at <http://172.20.2.1:3333/>.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address.

Access the Local UI at <wanip>:3333 in a web browser and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address. The credentials for the Local UI found in the customer profile are also required.

1. Access the Local UI at <wanip>:3333 in a web browser.
2. Log in with the credentials from the customer profile.

[Sign In](#)

Username  
local@nerve.cloud

Password

[Sign In](#)

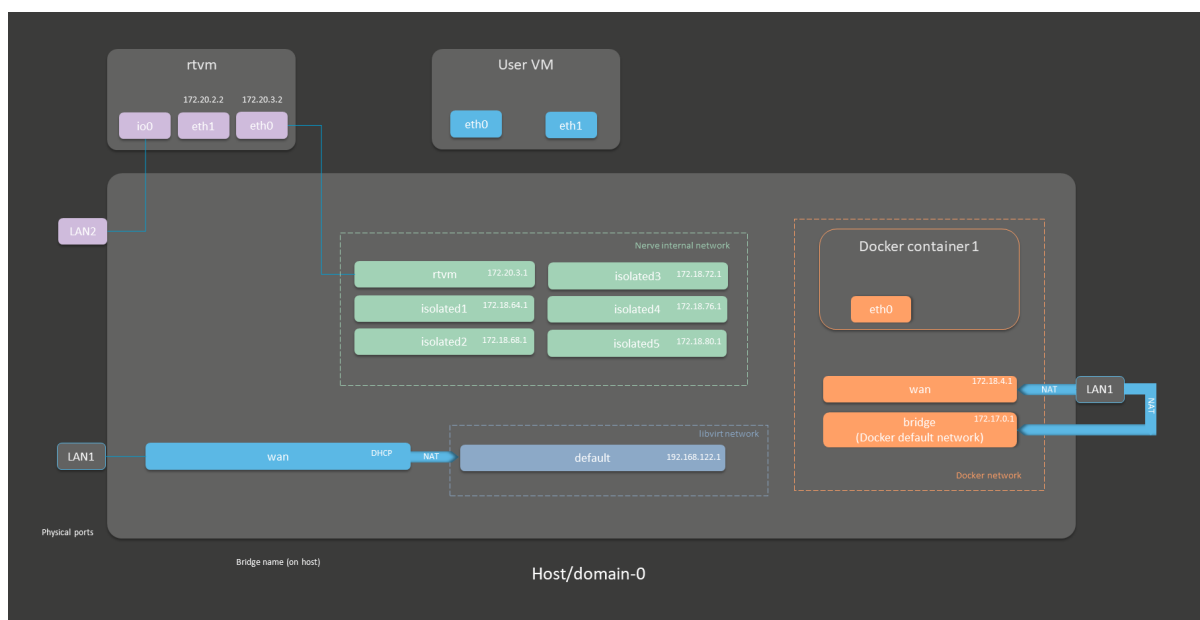


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the SuperServer E100-9AP-IA hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the SuperServer E100-9AP-IA.

Physical port	Network name
LAN1	wan
LAN2	io0



# Supermicro FHN13TP

# SuperServer

# 1019D-16C-



With an excellent price/performance ratio, the Supermicro Superserver devices offer a compact design with good connectivity, allowing optimal Nerve performance. The SuperServer 1019D-16C-FHN13TP features a 16 core Intel Xeon-D and a high count of Ethernet interfaces. It is 19" rack mountable, but still very compact.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@tttech-industrial.com](mailto:sales@tttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	SYS-1019D-16C-FHN13TP
<b>CPU</b>	Intel Xeon D-2183IT
<b>Cores</b>	16
<b>RAM</b>	32 GB
<b>Storage</b>	2x 256 G
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"><li>• 9x GbE</li><li>• 2x 10GbE</li><li>• 2x 10GbE SFP+</li><li>• 1 Dedicated IPMI LAN port</li><li>• 1x COM RS-232/422/485 (ESD 8KV)</li><li>• 1x COM via RJ45</li><li>• 2x USB 3.0</li><li>• 1x DisplayPort</li><li>• 1x VGA</li></ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:



- a monitor with a DisplayPort or VGA input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. <!--Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

## BIOS update

To avoid possible issues and complications, the BIOS version on the Superserver 1019D-16C-FHN13TP needs to be version 1.3 or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version. -->

## Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Advanced &gt; CPU Configuration &gt; Hyper-Threading [ALL]</b>	<b>Disabled</b>
<b>Advanced &gt; CPU Configuration &gt; Intel Virtualization Technology</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Configuration &gt; Advanced Power Management Configuration &gt; Power Technology</b>	<b>Disabled</b>

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort or VGA input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the SuperServer 1019D-16C-FHN13TP.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.



## On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img`. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-1019d-16c-fhn13tp.img of=/dev/<drivename> bs=1M sync
```

### NOTE

Make sure to replace `<drivename>` with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN3** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask.

Access the Local UI at <http://172.20.2.1:3333/> and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN3** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask. The credentials for the Local UI found in the customer profile are also required.

1. Follow this link to connect to the Local UI: <http://172.20.2.1:3333/>
2. Log in with the credentials from the customer profile.



Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the SuperServer 1019D-16C-FHN13TP hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the SuperServer 1019D-16C-FHN13TP.

Physical port	Network name
LAN1	extern1
LAN2	io0
LAN3	mgmt
LAN4	wan
LAN5	extern2
LAN6	extern3
LAN7	extern4
LAN8	extern5
LAN9	extern6

Physical port	Network name
LAN10	extern7
LAN11	extern8
LAN12	extern9
LAN13	extern10

#### NOTE

Due to the large number of network ports, contact TTTech Industrial customer support at [support@tttech-industrial.com](mailto:support@tttech-industrial.com) for any additional configuration requests.

## Supermicro SuperServer 5029C-T



With an excellent price/performance ratio, the Supermicro Superserver devices offer a compact design with good connectivity, allowing optimal Nerve performance. The SuperServer 5029C-T is a compact, mini tower server system featuring an Intel Core i3, and a high number of SSD/HDD mounting options.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

### Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@tttech-industrial.com](mailto:sales@tttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	SYS-5029C-T
<b>CPU</b>	Intel Core i3-8100
<b>Cores</b>	4
<b>RAM</b>	16 GB
<b>Storage</b>	HDD 2x 1 TB
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"><li>• 1x PCI-E 3.0 x16 slot</li><li>• 2x GbE</li><li>• 1 Dedicated IPMI LAN port</li><li>• 1x COM</li><li>• 4x USB 3.1</li><li>• 2x USB 2.0</li><li>• 1x VGA</li></ul>

## Setting up the device for Nerve usage

### NOTE

The device needs to have a RAID controller installed to be used with Nerve. Consult the manufacturer's user manual on how to install a RAID controller before attempting to install Nerve.

Requirements for the instructions below:

- a monitor with a VGA input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, the BIOS version on the SuperServer 5029C-T needs to be version 1.3 or later and Setup Utility needs to be V2.20.1276 or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

## Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
<b>Advanced &gt; CPU Configuration &gt; Intel (VMX) Virtualization Technol</b>	<b>Enabled</b>
<b>Chipset Configuration &gt; System Agent (SA) Configuration &gt; VT-d</b>	<b>Enabled</b>
<b>Advanced &gt; CPU Configuration &gt; C states</b>	<b>Disabled</b>
<b>Advanced &gt; CPU Configuration &gt; Active Processor Cores</b>	<b>All</b>
<b>Advanced &gt; PCIe/PCI/PnP Configuration &gt; CPU SLOT1 PCI-E 3.0 X16 OPROM</b>	<b>EFI</b> (this option is needed when the RAID controller is installed)

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a VGA input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the SuperServer 5029C-T.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img`. Depending on the program used, the file might need to be extracted more than once.

3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img.tar.g
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_supermicro-superserver-5029c-t.img b
sync
```

### NOTE

Make sure to replace `<drivename>` with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Finding out the IP address of the device

Due to the limited availability of ethernet ports, Nerve does not offer a designated port and interface for host access and management purposes on the Supermicro SuperServer 5029C-T. The IP address of the **wan** interface that is mapped to physical port **LAN1** is required to start using Nerve. Depending on the network access the node has, this needs to be done differently.

### The node has network access

If the node has network access, an IP address will be assigned to the **wan** interface by a DHCP server. Follow the instructions below to find out the IP address of the **wan** interface.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following command to display the IP address of the **wan** interface:

```
ip a s wan
```

The IP address is displayed next to **inet** in the output the system gives. This IP address is required to access the Local UI in the instructions below.

## The node does not have network access

In case of the node not having network access, the IP address of the **wan** interface has to be set manually. For simplicity, the IP address of the **wan** interface will be set to 172.20.2.1 — the IP address of the host.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following commands to open the **wan** interface configuration:

```
cd /etc/network/interfaces.d
sudo nano wan
```

5. Enter the host access password if prompted.
6. Edit the configuration the following way:

```
auto wan
iface wan inet static
    bridge_ports eth0
    address 172.20.2.1
    netmask 255.255.255.0
```

7. Enter Ctrl+S to save the configuration.
8. Enter Ctrl+X to exit the Nano editor.
9. Enter the following command to apply the changes to the **wan** interface by restarting the networking services:

```
/etc/init.d/networking restart
```

10. Enter the host access password if prompted.

With the **wan** interface IP address set to 172.20.2.1 the Local UI can be reached at <http://172.20.2.1:3333/>.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address.

Access the Local UI at <wanip>:3333 in a web browser and refer to [License activation](#) in the user guide for more information.



## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address. The credentials for the Local UI found in the customer profile are also required.

1. Access the Local UI at <wanip>:3333 in a web browser.
2. Log in with the credentials from the customer profile.

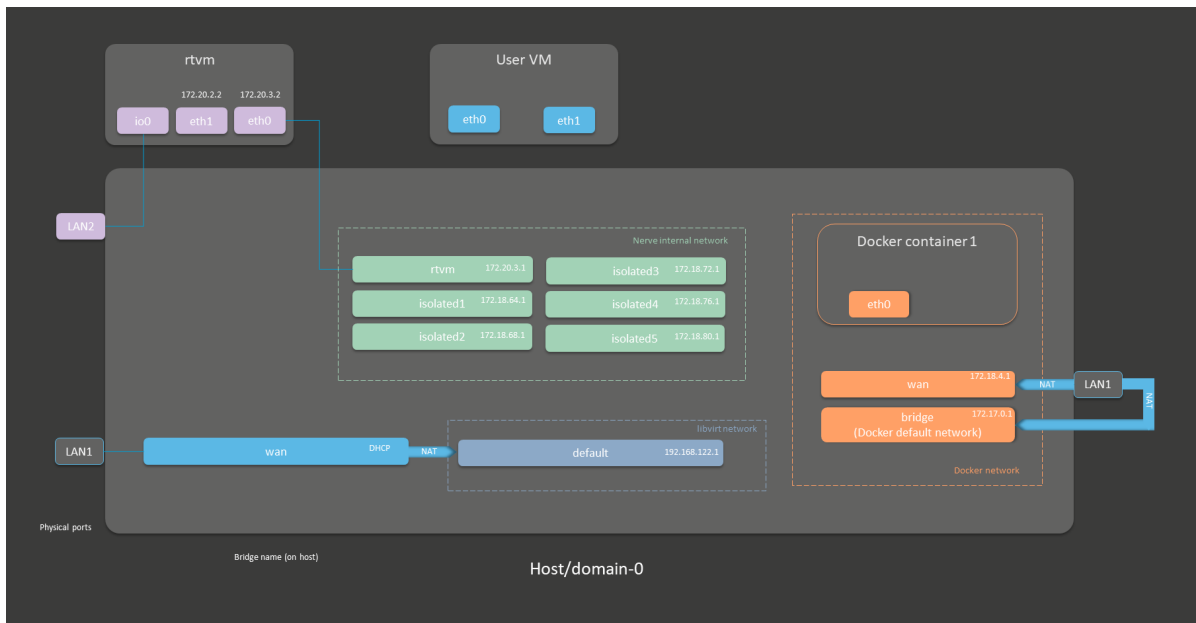


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the SuperServer 5029C-T hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the SuperServer 5029C-T.

Physical port	Network name
LAN1	wan
LAN2	io0



## Vecow SPC-5600-i5-8500



The Vecow SPC-5600 Series is powered by 8 Cores Intel® Core™ i5 and can be used for vehicle computing, smart manufacturing, in-vehicle infotainment, intelligent control or any IIoT application.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

## Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@tttech-industrial.com](mailto:sales@tttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	SPC-5600A-8500T32
<b>CPU</b>	Intel Core i5-8500T
<b>Cores</b>	6
<b>RAM</b>	32 GB DDR4
<b>Storage</b>	512 GB 2.5" SATA SSD MLC
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"><li>• 4x GB LAN</li><li>• 4x COM RS-232/422/485 (ESD 8KV)</li><li>• 4x USB 3.1 (external)</li><li>• 1x USB 2.0 (internal)</li><li>• 2x DisplayPort</li></ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, the BIOS version on the SPC-5600 needs to be version V27.01.03. or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Press Del while the device is booting to enter the BIOS menu.
3. Change the following settings:

Path	Setting
Advanced > CPU Configuration > Intel (VMX) Virtualization Technology	Enabled
Advanced > Power & Performance > CPU - Power Management Control > Boot performance mode	Max Non-Turbo Performance
Advanced > Power & Performance > CPU - Power Management Control > Intel (R) SpeedStep(tm)	Disabled
Advanced > Power & Performance > CPU - Power Management Control > Intel (R) Speed Shift Technology	Disabled
Advanced > Power & Performance > CPU - Power Management Control > C states	Disabled
Advanced > ACPI Settings > Enable Hibernation	Disabled
Advanced > ACPI Settings > ACPI Sleep State	Suspend Disabled
Advanced > Network Stack Configuration > Network Stack	Enabled
Chipset > System Agent (SA) Configuration > VT-d	Enabled
Security > Secure Boot > Secure Boot	Disabled

4. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with a DisplayPort input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the Vecow SPC-5600.

Before beginning with the installation, make sure that the device will boot from the USB drive. Press Del when the device is booting to enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img`. Depending on the program used, the file might need to be extracted more than once.

3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img.tar.gz
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_vecow-spc-5600.img bs=4M of=/dev/sd<
sync
```

### NOTE

Make sure to replace `<drivename>` with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN 1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask.

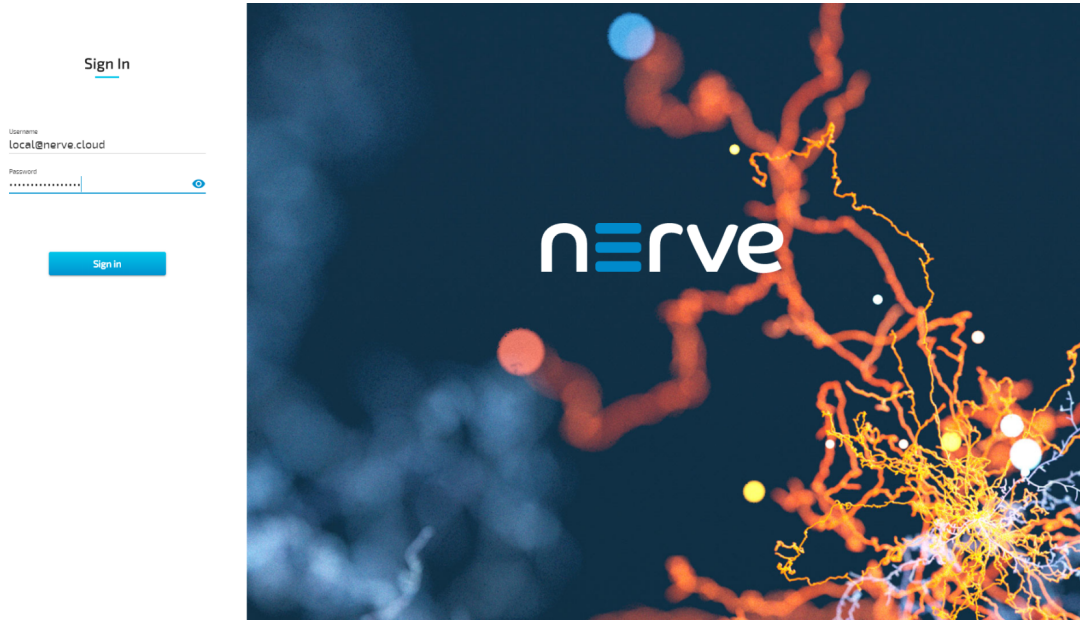
Access the Local UI at <http://172.20.2.1:3333/> and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN 1** and configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask. The credentials for the Local UI found in the customer profile are also required.

1. Follow this link to connect to the Local UI: <http://172.20.2.1:3333/>

2. Log in with the credentials from the customer profile.

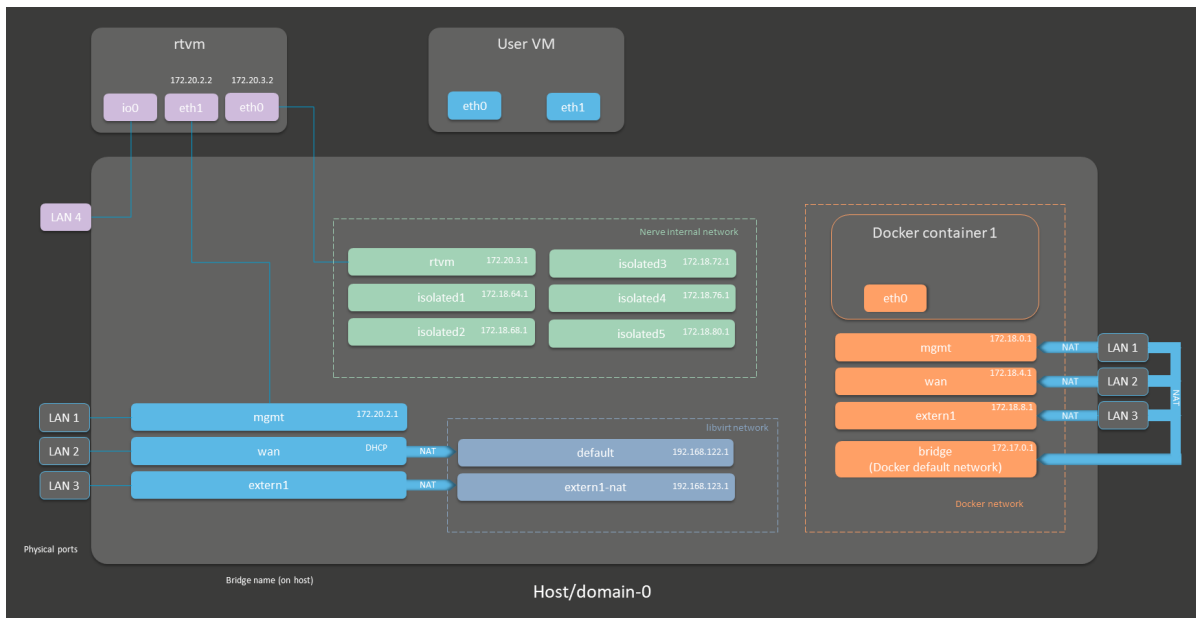


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the SPC-5600 hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the SPC-5600.

Physical port	Network name
LAN 1	mgmt
LAN 2	wan
LAN 3	extern1
LAN 4	io0



## Winmate IoT Gateway EAC Mini EACIL20



The EAC series gateway's hardware has a compact design and is based on rugged, reliable, and long-lasting industrial technology. It is a plug & play IoT gateway that simplifies deployment with multi-connectivity, optimized thermal solution, and easy mounting.

For more information refer to the information materials provided by the manufacturer:

- [Product page](#)
- [User manual](#)

### Device specifications

The table below contains the key specifications of the specific hardware model that has been certified for Nerve usage. Use the article number listed here when ordering the

device from the manufacturer only. Note that other device variants are not supported as Nerve Devices.

If required, contact [sales@ttech-industrial.com](mailto:sales@ttech-industrial.com) for help with ordering Nerve Devices.

Item	Description
<b>Article number</b>	<ul style="list-style-type: none"><li>• <b>Product Name</b> BOX PC</li><li>• <b>Model No.</b> EACIL20-100-A412</li></ul>
<b>CPU</b>	Intel Pentium N4200
<b>Cores</b>	4
<b>RAM</b>	4 GB
<b>Storage</b>	128 GB
<b>TPM</b>	TPM 2.0 included
<b>Interfaces</b>	<ul style="list-style-type: none"><li>• 2x GB LAN</li><li>• 2x USB 3.0</li><li>• 1x HDMI</li></ul>

## Setting up the device for Nerve usage

Requirements for the instructions below:

- a monitor with an HDMI input
- a keyboard
- a USB drive formatted to FAT32

Refer to the [user manual](#) of the manufacturer to set up the hardware. Connect a keyboard and a monitor to the device and make sure that the device is ready to be powered on. Also, prepare a USB drive in case the BIOS version of the device needs to be updated.

### BIOS update

To avoid possible issues and complications, update the BIOS on the EACIL20 to InsydeH20 Setup Utility, rev. 5.0. or later. Refer to the [user manual](#) of the manufacturer for information on how to update the BIOS version.

### Required BIOS settings for Nerve

Certain BIOS settings need to be changed to ensure the desired performance of the Nerve system.

1. Power on the device.
2. Enter **exit** to leave the EFI shell.
3. Go to **Setup Utility**.
4. Change the following settings:



Path	Setting
Power > CPU Configuration > VT-d	Enabled
Power > CPU Configuration > CPU Power Management > C-States	Disabled

5. Save the changes and exit BIOS.

## Installing Nerve

Requirements for installing Nerve on the device:

- a monitor with an HDMI input
- a keyboard
- a USB drive
- the `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img.tar.gz` which can be downloaded from the [Nerve Software Center](#)
- a tool for creating bootable USB drives like [Rufus](#) on Windows

In addition, a workstation is required to prepare the bootable USB drive. Connect the monitor and the keyboard to the EACIL20.

Before beginning with the installation, make sure that the device will boot from the USB drive. Enter BIOS and change the boot device settings.

### On Windows

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img.tar.gz` from the [Nerve Software Center](#) to a workstation.
2. Extract the `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img.tar.gz` file to retrieve the `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img` file. Depending on the program used, the file might need to be extracted more than once.
3. Transfer the extracted `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img` file to the USB drive using [Rufus](#).
4. Plug the USB drive into a USB port of the Nerve Device.
5. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

### On Linux

1. Download the `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img.tar.gz` file from the [Nerve Software Center](#).
2. Enter the following commands to extract the `Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img.tar.gz` file and transfer the extracted file to the USB drive:

```
tar xf Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20.img
sudo dd if=Nerve_Blue_USB-installer_2.3.1_for_winmate-iot-gateway-eac-mini-eacil20
sync
```

#### NOTE

Make sure to replace <drivename> with the system name of the USB drive.

3. Plug the USB drive into a USB port of the Nerve Device.
4. Make sure that the device will boot from the USB drive and power on the device.

The setup will start automatically and take a few minutes to complete. Select **OK** when the installation is complete and remove the USB drive. The device will reboot and reach a log in screen, asking for host access log in credentials. Make sure that the device will boot from the hard disk before rebooting the device.

## Finding out the IP address of the device

Due to the limited availability of ethernet ports, Nerve does not offer a designated port and interface for host access and management purposes on the Winmate EACIL20. The IP address of the **wan** interface that is mapped to physical port **LAN1** is required to start using Nerve. Depending on the network access the node has, this needs to be done differently.

### The node has network access

If the node has network access, an IP address will be assigned to the **wan** interface by a DHCP server. Follow the instructions below to find out the IP address of the **wan** interface.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following command to display the IP address of the **wan** interface:

```
ip a s wan
```

The IP address is displayed next to **inet** in the output the system gives. This IP address is required to access the Local UI in the instructions below.

### The node does not have network access

In case of the node not having network access, the IP address of the **wan** interface has to be set manually. For simplicity, the IP address of the **wan** interface will be set to 172.20.2.1 — the IP address of the host.

1. Connect a keyboard and a monitor to the device.
2. Power up the device once Nerve is installed.
3. Enter the login credentials for host access once the system asks for host login. The login credentials can be found in the customer profile.
4. Enter the following commands to open the **wan** interface configuration:

```
cd /etc/network/interfaces.d
sudo nano wan
```

5. Enter the host access password if prompted.
6. Edit the configuration the following way:

```
auto wan
iface wan inet static
    bridge_ports eth0
    address 172.20.2.1
    netmask 255.255.255.0
```

7. Enter Ctrl+S to save the configuration.
8. Enter Ctrl+X to exit the Nano editor.
9. Enter the following command to apply the changes to the **wan** interface by restarting the networking services:

```
/etc/init.d/networking restart
```

10. Enter the host access password if prompted.

With the **wan** interface IP address set to 172.20.2.1 the Local UI can be reached at <http://172.20.2.1:3333/>.

## Activating the Nerve license

After the installation, the product license needs to be activated so that Nerve can be used on the device. Connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address.

Access the Local UI at <wanip>:3333 in a web browser and refer to [License activation](#) in the user guide for more information.

## Accessing the Local UI and registering the device

With the license activated, the node needs to be registered for use in the Management System through the Local UI. To access the Local UI, first connect a workstation to port **LAN1** and configure the network adapter of the workstation. The IP address has to be in the same range as the IP address of the **wan** interface with a 255.255.255.0 subnet mask. Refer to the chapter [above](#) on how to find out this IP address. The credentials for the Local UI found in the customer profile are also required.

1. Access the Local UI at <wanip>:3333 in a web browser.
2. Log in with the credentials from the customer profile.

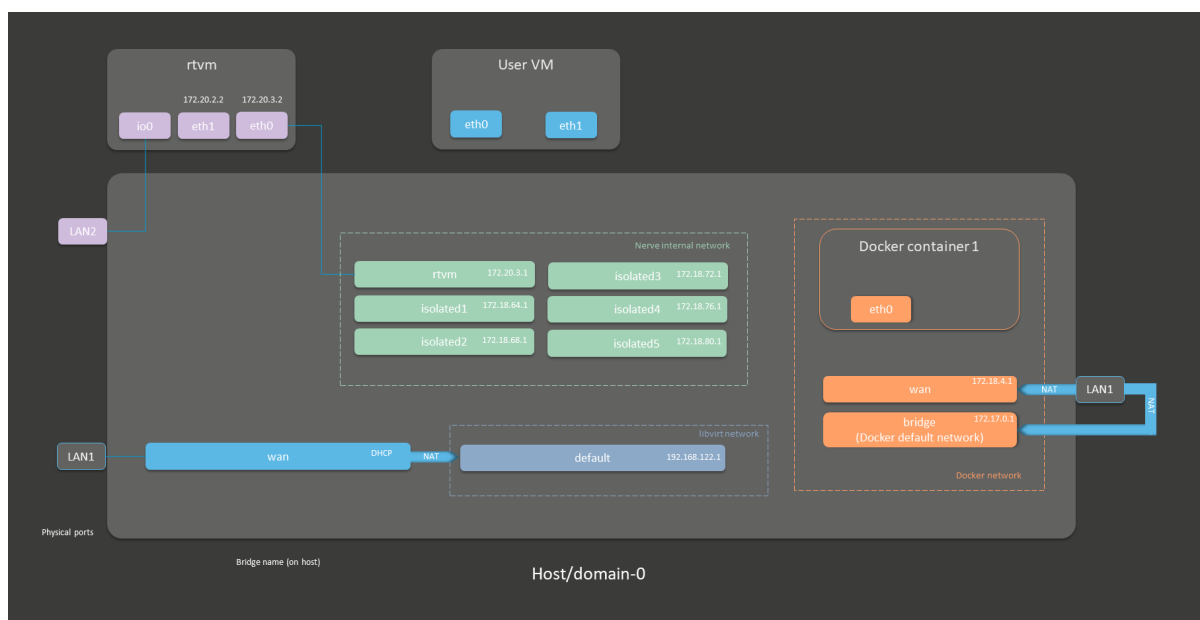


Continue with [Node configuration](#) for information on how to start registering the device in the Management System.

## Physical ports and network interfaces

Below is a depiction of the node internal networking adapted to the EACIL20 hardware. Refer to [Node internal networking](#) for more information. The table offers a quick overview of the network interfaces that can be reached through the physical ports of the EACIL20.

Physical port	Network name
LAN1	wan
LAN2	io0



# Developer Guide

## Developer Guide

The developer guide is a continuation of the user guide. It aims to give support to developers with tasks going beyond the regular usage of the Nerve system. In this version of the Nerve documentation it offers support with the following topics:

- an overview of node internal networking in the Nerve system
- first steps to programming CODESYS applications on Nerve Devices
- an introduction to the usage of the Management System API for working with workloads
- a full guide for the Nerve Data Services

This guide will be expanded with future releases.

## Nerve Data Services

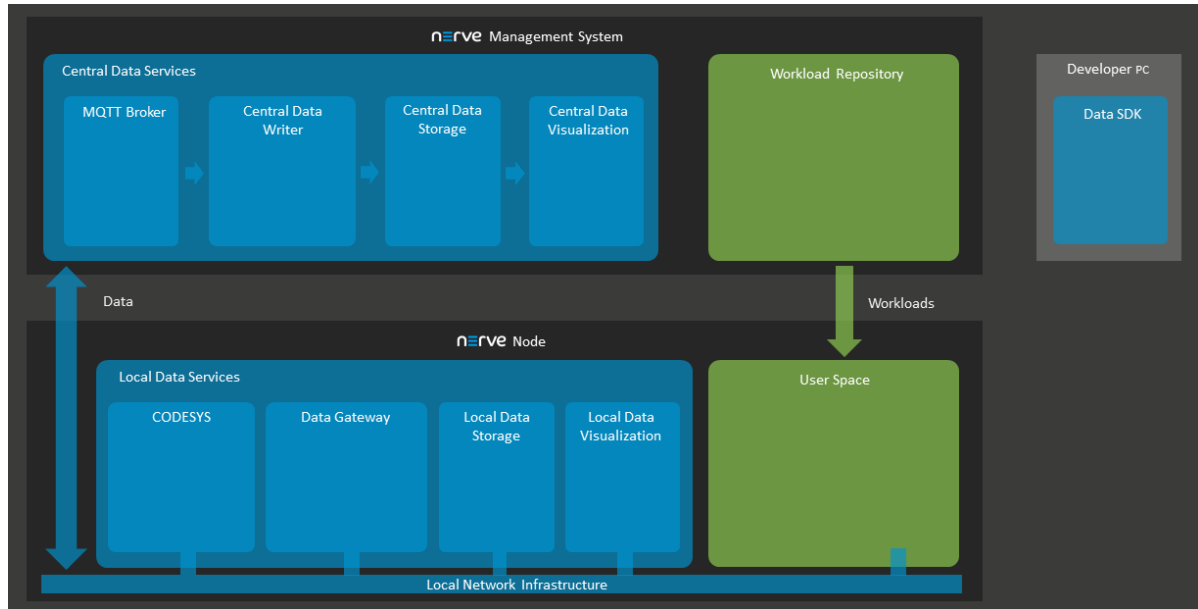
### Nerve Data Services

The Nerve Data Services are a collection of services and interfaces that allow to collect, store, analyze, visualize and distribute data. These services and interfaces are available on the Nerve node and in the Management System. Using Nerve Data Services, data can be collected at high speeds on the node and stored or visualized locally at the Node or centrally in the Management System.

### Elements of Nerve Data Services

Nerve Data Services	Description and elements
<b>Local Data Services</b>	The Local Data Services are accessible on the node. They include the following elements: <ul style="list-style-type: none"><li>• CODESYS</li><li>• Local Data Gateway</li><li>• Local Data Storage</li><li>• Local Data Visualization</li></ul>
<b>Central Data Services</b>	The Central Data Services are part of the Management System. They include the following elements: <ul style="list-style-type: none"><li>• Central Data Writer</li><li>• Central Data Storage</li><li>• Central Data Visualization</li></ul>
<b>User PC</b>	Data SDK With the Data SDK users can create analytics applications, which can be deployed to a node.

Most of the above elements are individually configurable. This allows for an application in a large number of scenarios and use cases. However, this also implies deeper knowledge of the elements and their possible interactions. Further explanations of the elements can be found in separate chapters. Find a high-level graphical overview below.



## Supported protocols

Inputs	Outputs
MQTT Subscriber	MQTT Publisher
OPC UA PubSub Subscriber	OPC UA PubSub Publisher
OPC UA Client	OPC UA Server
Modbus Server	ZeroMQ Publisher
S7 Server	Azure IoT Hub Device
ZeroMQ Subscriber	Kafka Producer
	NerveDB
	TimescaleDB
	Influx DB

## Model workflow

The following section gives an overview of a typical work flow, presenting what can be done with the Nerve Data Services on a high level:

- Setting up data sources
- Configuring the Gateway to collect data and defining where to distribute it
- Configuring Grafana on the node for local data visualization
- Configuring Grafana in the Management System for central data visualization
- Implementing an analytics app and configuring the Gateway again to send data to the analytics app
- Reconfiguring Grafana to visualize the result of the processing performed by the analytics app

The paragraphs below give short explanations to each step.

## **Ingesting data on the node**

CODESYS can be used to translate fieldbus protocols to OPC UA Server. The local Gateway can then be configured to collect data from CODESYS and distribute it to various destinations. Data sources that support protocols which are also native to the local Gateway can be linked there directly and do not need CODESYS.

## **Distributing data on the node**

Data is distributed by the local Gateway to user defined consumers. Unless the respective protocol of a Gateway output defines a specific data format, all data received by the Gateway is normalized to the [Nerve Data Services data format](#). Typical consumers of data are the local data storage on the node or the central data storage in the Management System, third party MQTT brokers or applications, or providing the data as OPC UA Server. To enable reuse of local Gateway configurations across multiple nodes, keywords can be used to abstract node specific information from the configuration.

## **Storage on the node**

Data is stored in the NerveDB on the node, a dedicated TimescaleDB database. Any application running on the node can access data stored in there. For application development, a Python API is provided in the Data SDK that simplifies reading and writing data. A time window can be configured for how long data shall be kept before deleting it.

## **Visualization on the node**

Visualization is available via Grafana which is pre-configured to allow access to data stored in the local data storage. It is also possible to configure access from Grafana to other user defined data sources without any restrictions, meaning that data in an influxDB that is installed on the node can also be visualized.

## **Using Data in analytics**

A Python SDK is available for creating custom analytics applications. An API is available that simplifies reading data from the local storage or receiving it directly from the Gateway when it arrives there. The API also allows to write back analytics results to the local storage for visualization or send it to an MQTT broker to distribute it to other consumers. It is also possible to use third party applications such as NodeRed or Crosser for analytics together with the Nerve Data Services.

## **Storing and visualizing in the Management System**

Data received on the Node can be forwarded to the Management for visualization. To allow this, the local Gateway and the central visualization must be configured. Keywords are in place to simplify the configuration process. Each node has a separate database in the Central Data Storage.

## **Data Services UI**

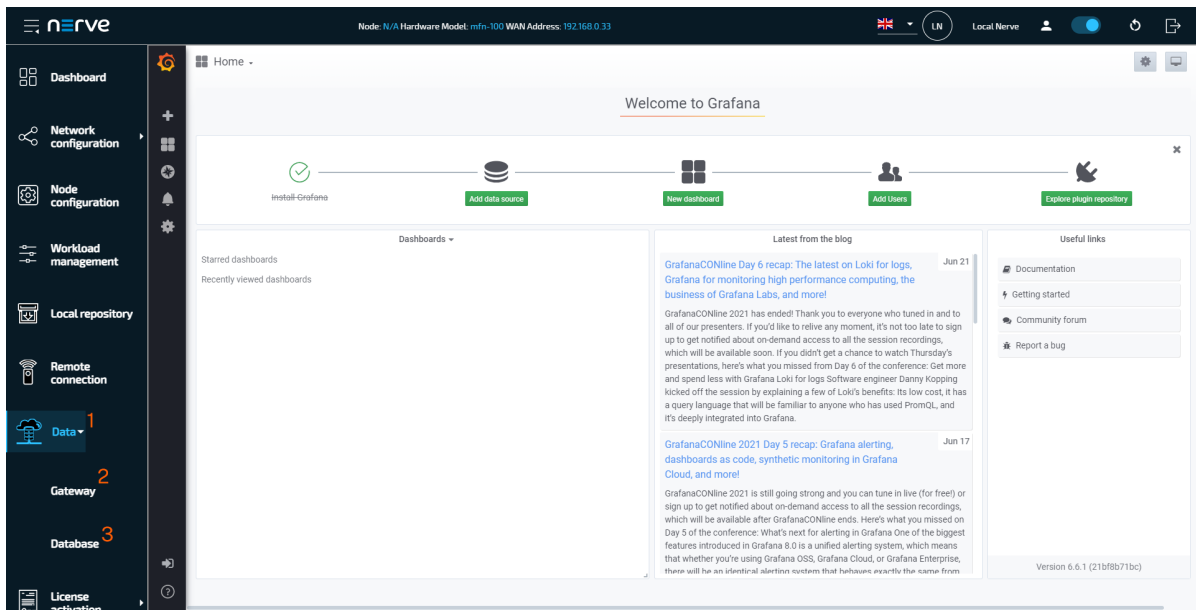
The Data Services are present on both the node and on the cloud. However, the instance on the node is the single point of configuration. The instance in the Management System is mainly used for viewing data.

Also, note that by default, the permission to access to Data Services in the Management System is not granted. Users that have the permission to create user roles can enable the Data Services entry in the navigation by adding the **UI\_NAV\_DATA\_SERVICES:VIEW** permission to a user role. A **Data Services** user role is also defined in the Management System by default that can be added to a user. Refer to [Users](#) for information on how to create new users and to [Roles and permissions](#) for information on how to add a new role.

Select the arrow next to **Data** in the navigation on the left to expand the following items:

## NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left corner to expand the navigation menu again.



Item	Description
<b>Data (1)</b>	Clicking here opens the visualization component of the Data Services and the Grafana landing page is displayed. For more information on visualization, refer to <a href="#">Data Visualization</a> .
<b>Configuration (2)</b>	Select this to reach the Gateway. The Gateway is the single point of configuration for data connectivity in the Nerve Data Services. Configure inputs, outputs and connections here. Refer to <a href="#">Nerve Data Services Gateway</a> for more information.
<b>Database (3)</b>	Data from databases can be viewed here. Every node has a local NerveDB database. In the Management System, databases are discerned by the serial numbers of the nodes and can be used to confirm the flow of data. Refer to <a href="#">Database</a> for more information.

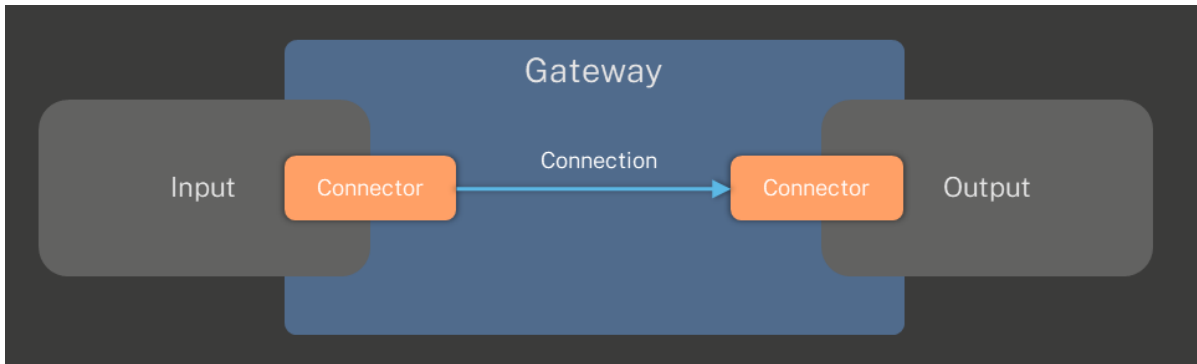


# Nerve Data Services Gateway

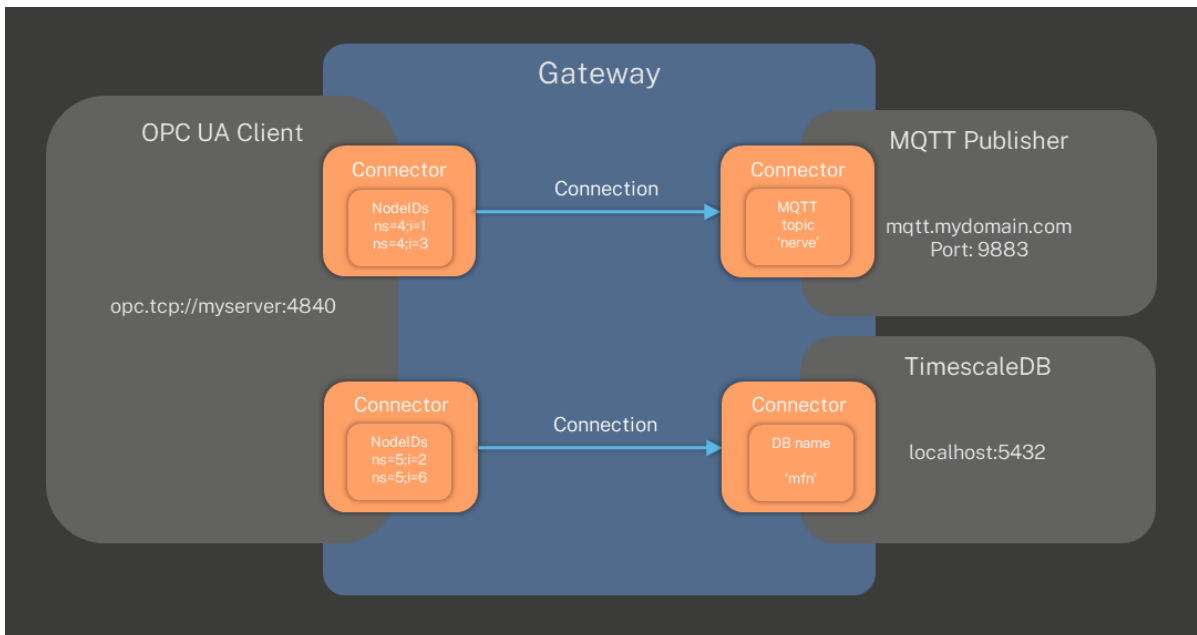
The Gateway is the central application of the Data Services. Its purpose is receiving data from a source via a certain protocol on an input interface and forwarding it to a destination using a different protocol on an output interface. This behavior is reflected in the user configuration as well. In general, the configuration file consists of inputs, outputs and connections between them:

- Inputs are collection interfaces where data is received.
- Outputs are providing interfaces where data can be received from.
- Connections are logical links between inputs and outputs.

Each input and output provides connectors where a connection can be attached to. A connector is a subset of the data available at an input. This subset is forwarded to the connector of an output where it is distributed further to the data receiver. On an abstract level, this concept is illustrated in the figure below.

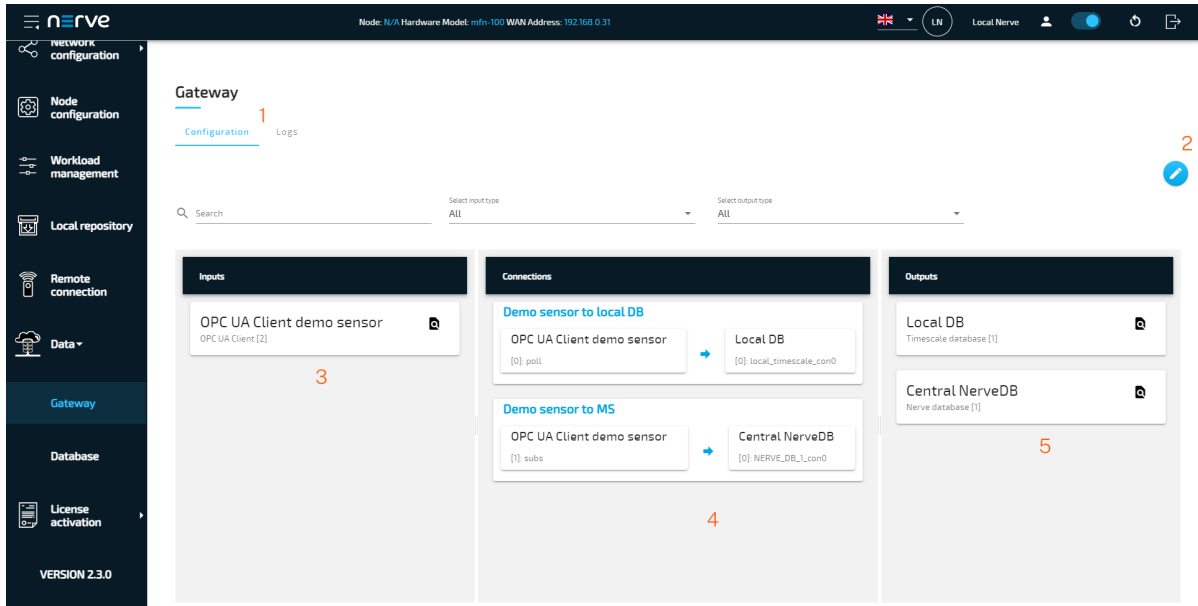


Inputs and outputs hold general configuration data for the protocol they implement. This can be a URL to a server, a port number or something related to timing. A connector of an input or output holds more specific information on how to get or where to send data, for example the node IDs of an OPC UA Server an OPC UA Client input connects to, or the name of a topic for an MQTT Publisher. Each input and output can provide multiple connectors at a time. Data from an input connector can also be connected to multiple output connectors at the same time to distribute data to multiple data consumers. The figure below shows a more specific example of that concept.



# Data Gateway UI

The Data Gateway UI is the central page of the Data Services and can be reached by selecting **Data > Gateway** in the navigation on the left in either Local UI or Management System. Gateway configurations are applied through the interface in the middle, either through the graphical configuration tool or with a pre-written JSON file that is uploaded. When accessing this page, the panes of the configuration tool displays the last deployed configuration.



Item	Description
<b>Configuration and Logs tabs (1)</b>	The <b>Configuration</b> tab is displayed by default, displaying the graphical configuration tool. The latest logs can be viewed within the <b>Logs</b> tab. The entirety of the logs can be exported into a file using the <b>Download</b> button in this tab.

Item	Description
<p><b>Edit configuration (2)</b></p>	<p>Select this button to enable editing mode and enter a configuration into the editor to configure the Gateway. In editing mode, the configuration with the graphical configuration tool is enabled, as well as options to import or export JSON configurations. After selecting the edit button, these buttons appear instead:</p> <div data-bbox="632 533 1251 663" style="text-align: center;"> </div> <p>The following descriptions describe the buttons from left to right.</p> <ul style="list-style-type: none"> <li>• <b>Deploy</b> Select this to apply the current configuration to the Gateway. The button becomes active once changes have been performed.</li> <li>• <b>Import</b> Select this to upload a Gateway configuration as a JSON file from the workstation into the Gateway. Once the file is uploaded, the graphical configuration tool will reflect the configuration from the JSON file.</li> <li>• <b>Export</b> Select this to download the currently deployed Gateway configuration as a JSON file.</li> <li>• <b>Discard current changes</b> Discard all the changes that have been done in editing mode. This will revert everything to the state before the edit button was selected.</li> <li>• <b>Import certificates</b> Import certificates and keys from a workstation to be used by the Gateway. When configuring inputs and outputs, only the name of the certificates and key files is required. Note the following: <ul style="list-style-type: none"> <li>◦ All certificates are stored in the same directory on the node. Therefore, certificates must have unique names when imported.</li> <li>◦ Certificates need to be in DER format.</li> </ul> </li> </ul> <p>To exit editing mode, select the arrow on the left.</p>
<p><b>Inputs (3)</b></p>	<p>Configured inputs are displayed here. Details can be viewed by selecting the magnifying glass icon. Editing options appear when editing mode is active.</p>
<p><b>Connections (4)</b></p>	<p>Configured connections are displayed here. Details can be viewed by selecting the magnifying glass icon. Editing options appear when editing mode is active.</p>
<p><b>Outputs (5)</b></p>	<p>Configured outputs are displayed here. Details can be viewed by selecting the magnifying glass icon. Editing options appear when editing mode is active.</p>

## Applying a configuration to the Gateway

The configuration of the Gateway is applied through the graphical configuration tool in the Gateway section of the Data Services UI. The current configuration of the Gateway is displayed in the separate input, connection and output panes if there is a configuration present in the Gateway. The panes will be empty on initial startup. A new configuration can be entered using the GUI or imported from an existing JSON file. After applying a configuration, check the **Logs** tab for any warnings or errors.

The basic structure of the configuration consist of a list of inputs such as an OPC UA server or an MQTT subscriber, outputs such as a Timescale database or an MQTT publisher, and connections between these inputs and outputs. Refer to [Gateway configuration parameter descriptions](#) below for the exact syntax. While studying the syntax is required to write a Gateway configuration in JSON format, note that the syntax is also helpful if the graphical configuration tool is used.

The examples in the [Examples](#) section show step by step applications in common use cases.

### Applying a configuration through the GUI

The graphical configuration tool offers a guided way to configure the Gateway. Required fields are marked with a red asterisk. The recommended order is defining an input, defining an output and then defining the connections last.

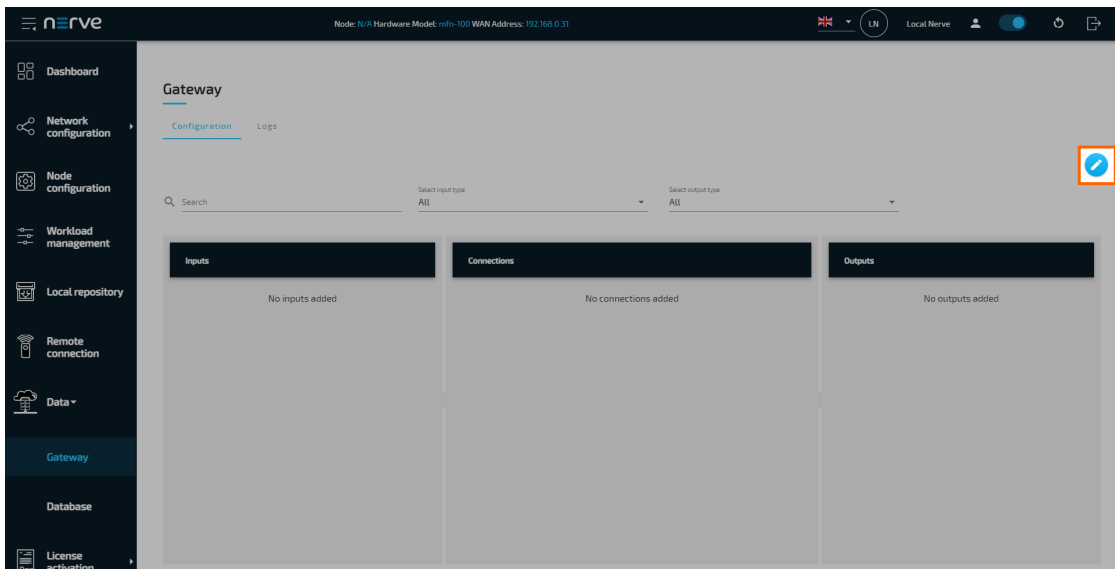
#### NOTE

While not marked with a red asterisk, defining a connector for the input and output is mandatory as it is required when configuring a connection.

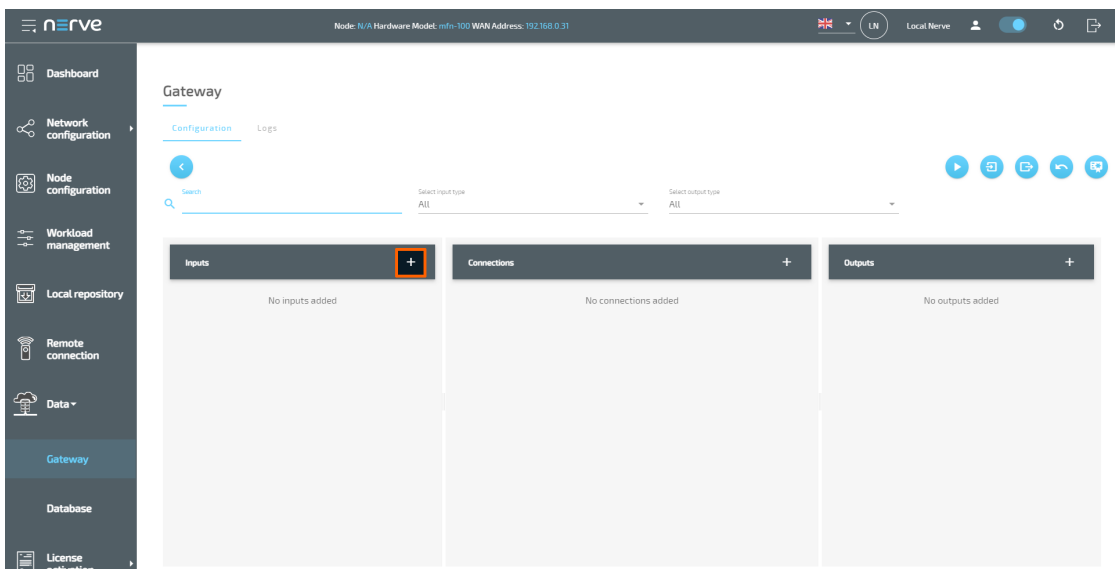
As a demonstration, the instructions below show how to recreate the pre-written JSON configuration file from the [OPC UA Server to cloud for visualization](#) example with the graphical configuration tool.

First, inputs need to be configured. This input will have two connectors configured that are required for connecting to two database outputs.

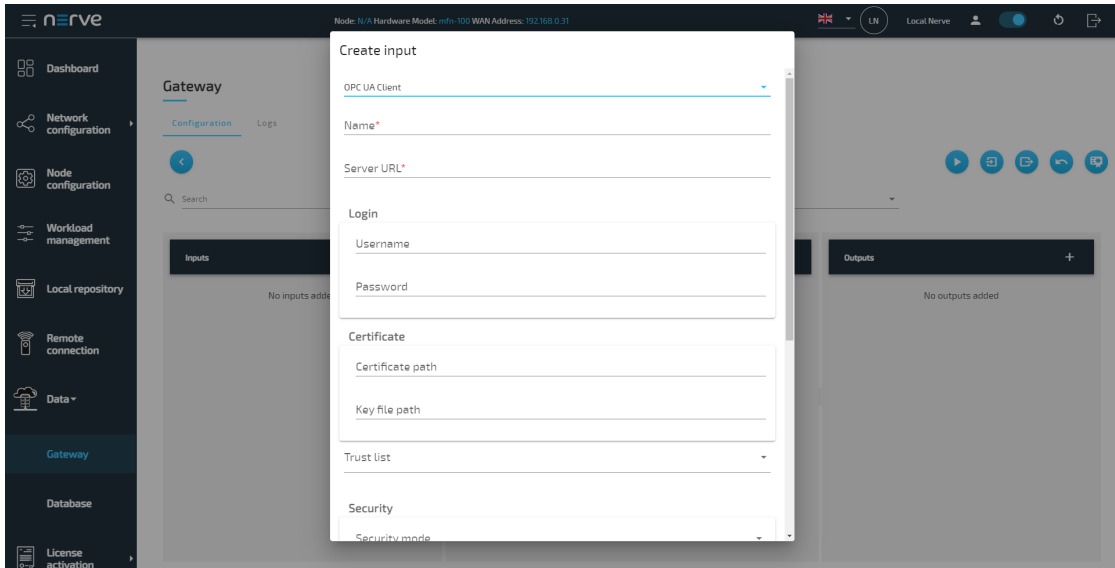
1. Select the **Edit configuration** icon on the right.



2. Select the plus icon next to **Inputs**.



3. Select **OPC UA Client** from the drop-down menu. A list of parameters will appear.

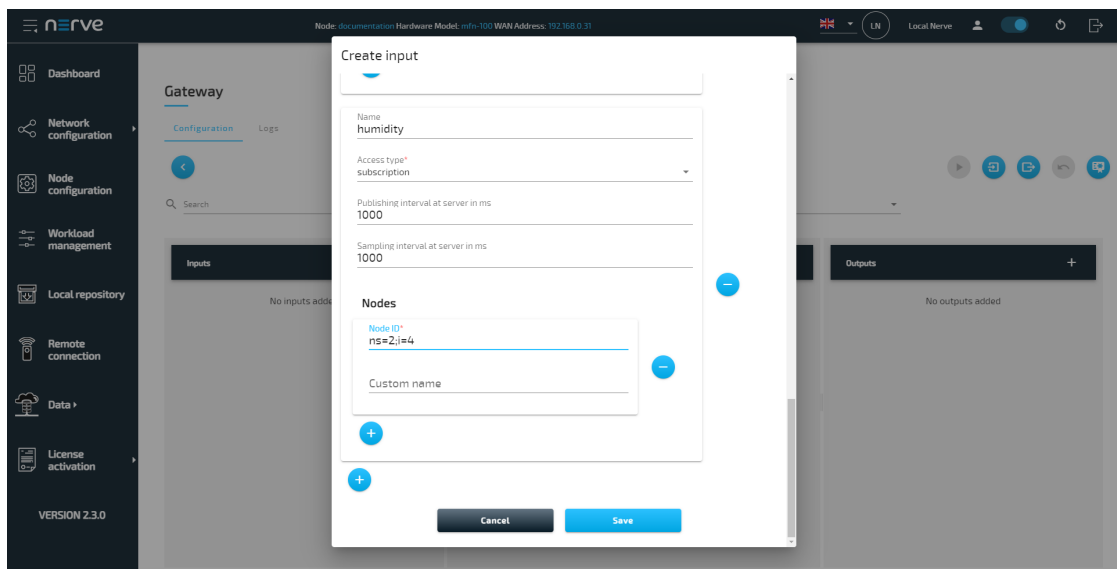


4. Enter the following information:

Setting	Value
<b>Name</b>	Enter any name, for example OPC UA Client.
<b>Server URL</b>	opc.tcp://localhost:4848
<b>Polling interval in ms</b>	1000
<b>Connector 1</b>	<p>Select the plus icon and enter the following information:</p> <p><b>Name</b> Enter a name for the connector. This example uses temperature.</p> <p><b>Access type</b> polling</p> <p><b>Nodes</b> Select the plus icon and enter the following information:</p> <ul style="list-style-type: none"> <li><b>Node ID</b> ns=2;i=2</li> </ul>

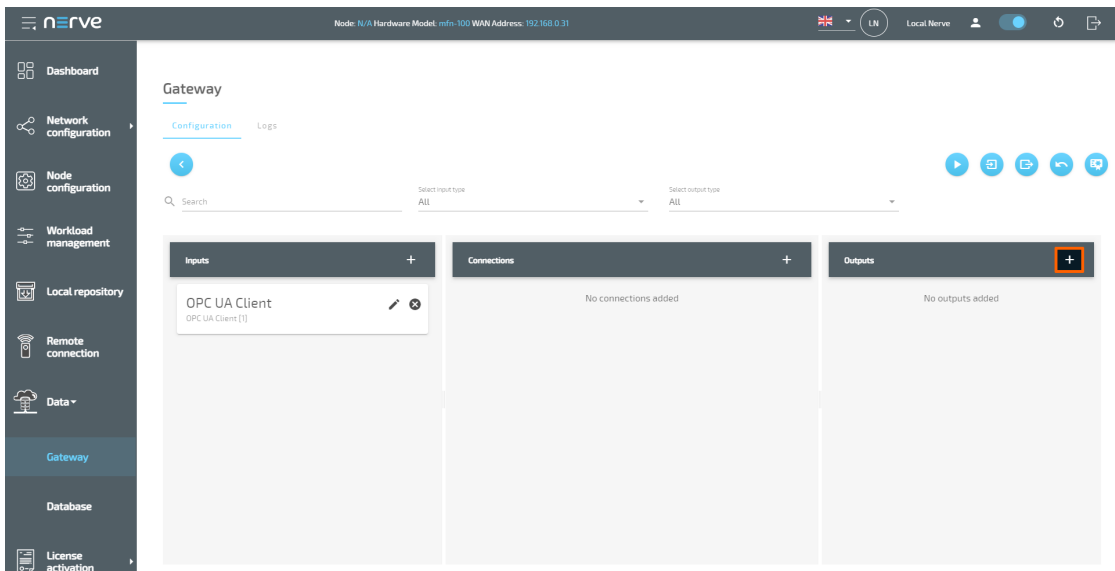
Setting	Value
	Select the plus icon and enter the following information:
	<b>Name</b> Enter a name for the connector. This example uses humidity.
	<b>Access type</b> subscription
<b>Connector 2</b>	<b>Publishing interval at server in ms</b> 1000
	<b>Sampling interval at server in ms</b> 1000
	<b>Nodes</b> Select the plus icon and enter the following information:
	<ul style="list-style-type: none"> <li><b>Node ID</b> ns=2;i=4</li> </ul>

5. Select **Save**.



The graphical configuration tool now reflects the configuration above, showing an OPC UA Client input. Next, it is required to define an output before defining connections. In the OPC UA Server to cloud example, two outputs are required, as temperature data from a demo sensor will be saved in the local NerveDB for visualization while humidity data from the same demo sensor will be saved in the central NerveDB in the Management System and be visualized there.

1. Select the plus icon next to **Outputs**.



2. Select **Nerve database** from the drop-down menu. A list of parameters will appear.
3. Enter the following information:

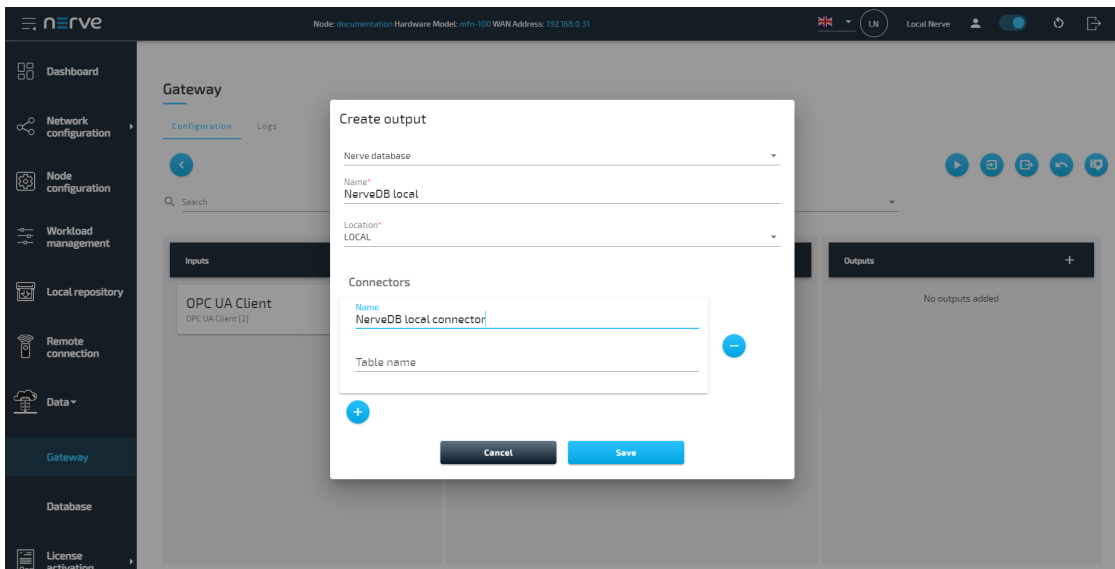
Setting	Value
<b>Name</b>	Enter any name, for example NerveDB local.
<b>Location</b>	LOCAL

Select the plus icon and enter the following information:

<b>Connector</b>	<p><b>Name</b> Enter a name for the connector. This example uses NerveDB local connector.</p> <p><b>Table name</b> Enter a name for the data table. If no table name is provided, the name of the connection will be used as a table name. Note that this field is optional when using the local NerveDB.</p>
------------------	---

4. Select **Save**.





5. Select the plus icon next to **Outputs** to define a second output.
6. Select **Nerve database** from the drop-down menu. A list of parameters will appear.
7. Enter the following information:

Setting	Value
<b>Name</b>	Enter any name, for example NerveDB central.
<b>Location</b>	CENTRAL

Select the plus icon and enter the following information:

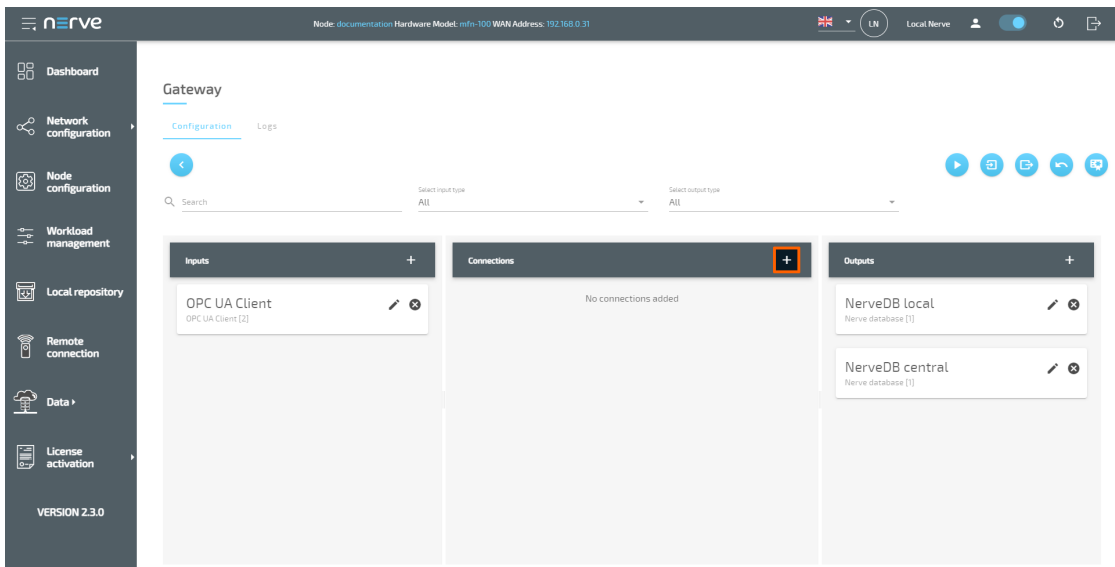
**Connector**  
**Name**  
 Enter a name for the connector. This example uses NerveDB central connector.

**Table name**  
 Enter a name for the data table. This example uses Demo sensor to central NerveDB.

8. Select **Save**.

The graphical configuration tool now reflects the configuration above, showing two database outputs in addition to the OPC UA Client input. With inputs, outputs and their respective connectors defined, connections can now be set up to finalize the Gateway configuration. In this example temperature data is sent to the local NerveDB and humidity data is sent to the central NerveDB.

1. Select the plus icon next to **Connections**.



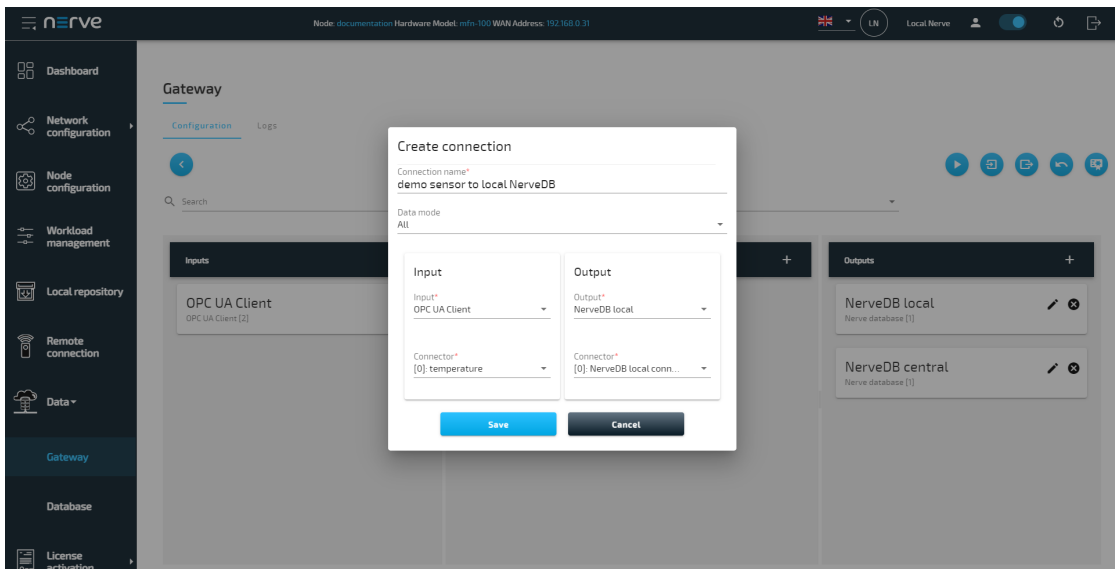
2. Enter a connection name, for example **demo sensor to local NerveDB**.
3. Select the following information from the drop-down menus under **Input**:

Setting	Value
<b>Input</b>	Select the OPC UA Client input, named <b>OPC UA Client</b> in this example.
<b>Connector</b>	Select the first connector, named <b>[0]: temperature</b> in this example.

4. Select the following information from the drop-down menus under **Output**:

Setting	Value
<b>Output</b>	Select the local NerveDB output, named <b>NerveDB local</b> in this example.
<b>Connector</b>	Select the connector, named <b>[0]: NerveDB local connector</b> in this example.

5. Select **Save**.



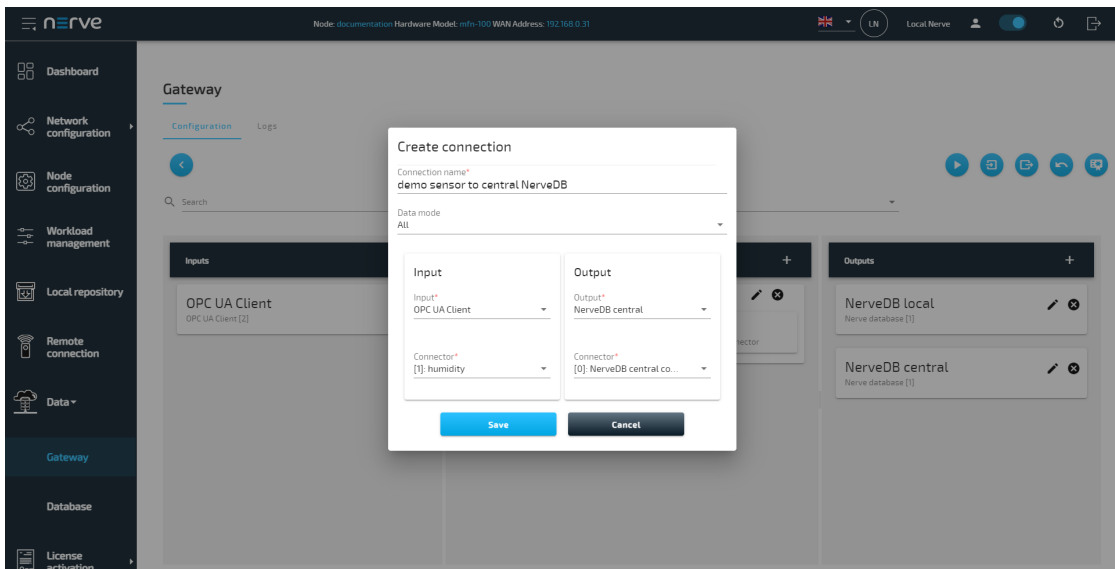
6. Select the plus icon next to **Connections** to add the second connection.
7. Enter a connection name, for example **demo sensor to central NerveDB**.
8. Select the following information from the drop-down menus under **Input**:

Setting	Value
<b>Input</b>	Select the OPC UA Client input, named <b>OPC UA Client</b> in this example.
<b>Connector</b>	Select the second connector, named <b>[1]: humidity</b> in this example.

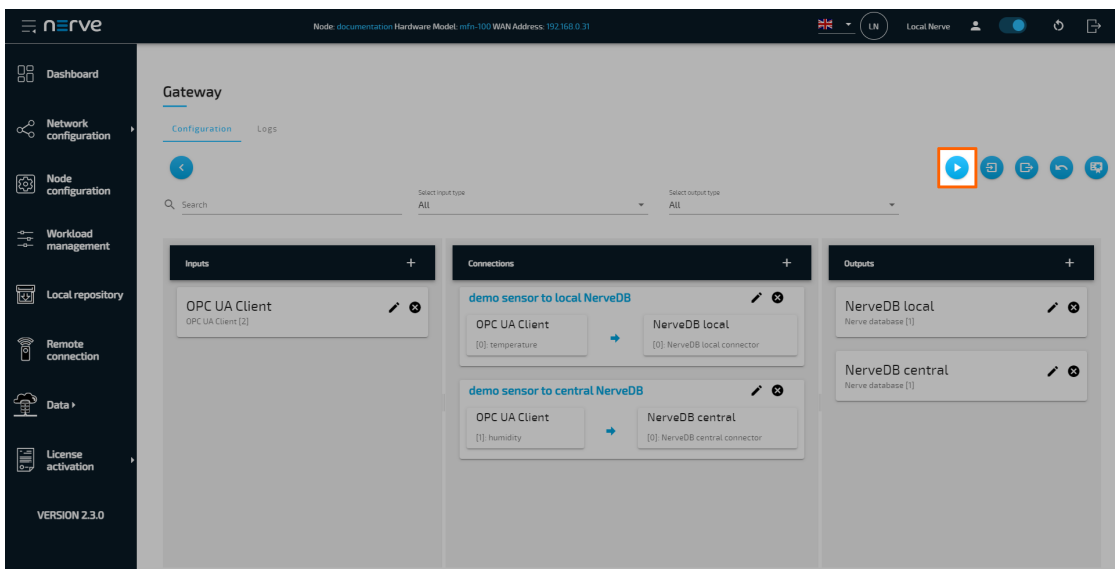
9. Select the following information from the drop-down menus under **Output**:

Setting	Value
<b>Output</b>	Select the NerveDB output, named <b>NerveDB central</b> in this example.
<b>Connector</b>	Select the connector, named <b>[0]: NerveDB central connector</b> in this example.

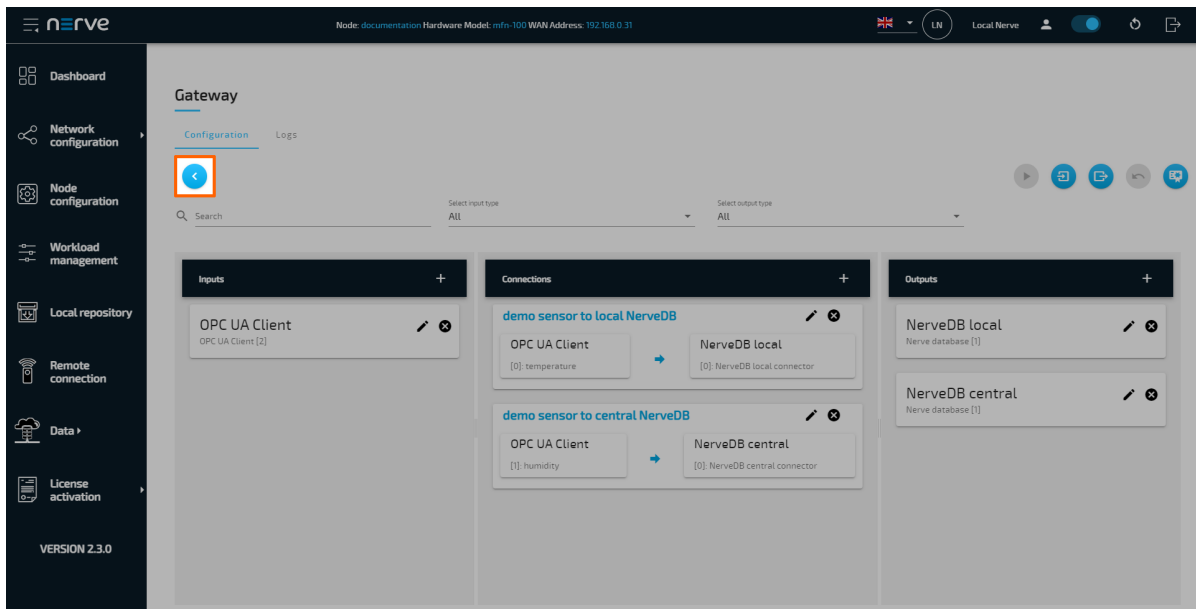
10. Select **Save**.



11. Select the **Deploy** button. A success message pops up in the upper-right corner.



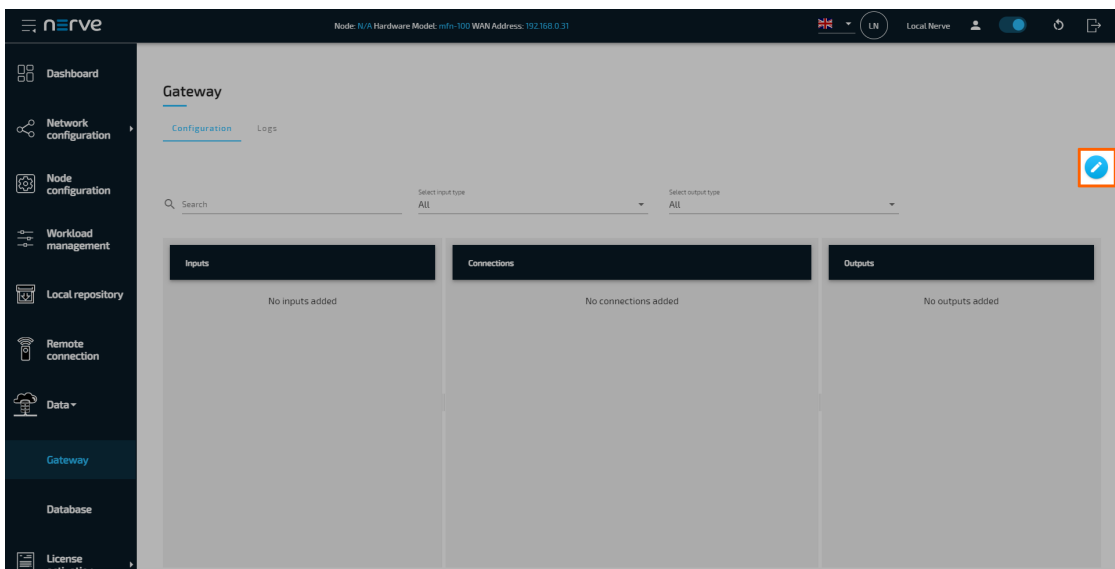
After deploying the configuration, the graphical configuration tool is still in editing mode. Select the arrow on the left side to exit editing mode.



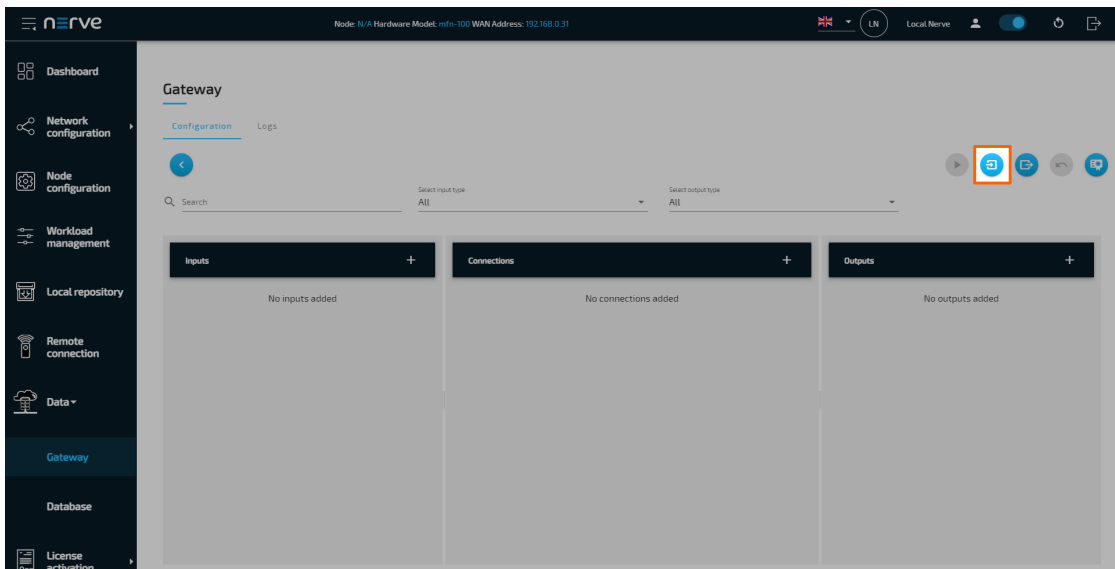
## Uploading an existing JSON configuration file

If writing a JSON configuration file is the preferred method of configuring the Gateway, a pre-written JSON file can be uploaded and applied to the Gateway. This method is also useful when following any of the [examples](#).

1. Log in to the Local UI.
2. Expand **Data > Gateway** in the navigation on the left.
3. Select the edit button on the right.

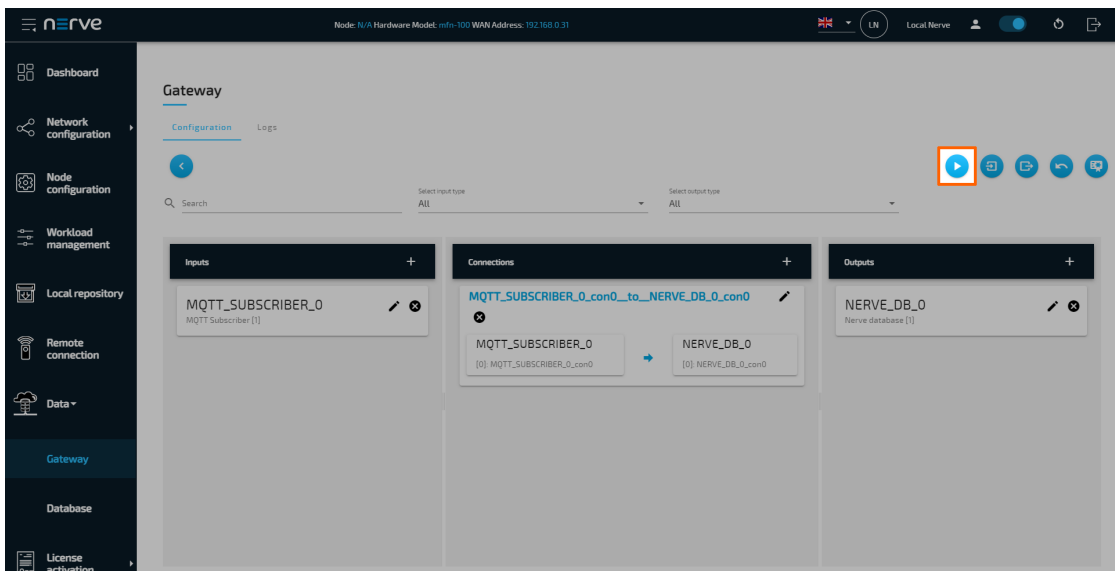


4. Select the **Import** button.

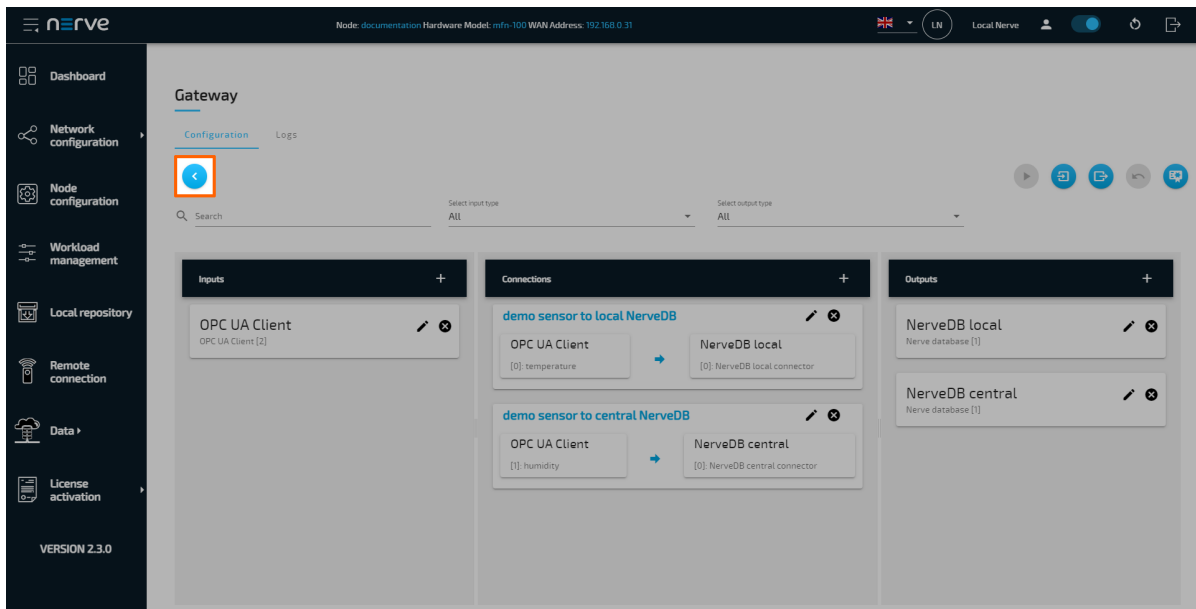


5. Add a pre-written JSON configuration file in the file browser.

6. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the settings from the JSON configuration file. Edit the elements in each pane by selecting the edit button in each element. Exit editing mode by selecting the arrow on the left.



For writing JSON configuration files, have a look at the [Nerve Data Services Data Format](#) below and consult the [list of parameters](#) for more information on all input, output and connection parameters.

## Writing a Gateway configuration file in the Nerve Data Services data format

Inputs and Outputs at the local Gateway, using a protocol that does not define a data format (i.e. MQTT), send or receive data in JSON format. The same applies to the Python API in the Analytics Data SDK. As a result, data within the Nerve Data Services is normalized to this format.

The JSON schema below describes the data format in more detail. The JSON schema is also used to validate data upon reception. If a data frame is received that does not comply with the schema, it is silently dropped by the Nerve Data Services.

```
{
  "$schema": "http://json-schema.org/draft/2019-09/schema#",
  "$id": "https://nerve.cloud/dp/dp_data_model.schema.json",
  "title": "Nerve Data Services JSON data model schema",
  "description": "Schema that represents default data model that Nerve Data Services",
  "type": "object",
  "properties": {
    "variables": {
      "type": "object",
      "additionalProperties": {
        "anyOf": [
          {
            "type": [ "boolean", "number", "string" ]
          },
          {
            "type": "array",
            "items": {
              "type": "boolean"
            }
          }
        ],
        "type": "array",

```

```

        "items": {
            "type": "number"
        }
    },
    {
        "type": "array",
        "items": {
            "type": "string"
        }
    }
]
},
"timestamp": {
    "type": [ "number", "string", "array" ],
    "items": {
        "type": [ "number", "string" ]
    }
}
},
"required": [ "variables" ],
"additionalProperties": false
}

```

The schema defines and allows only two properties in the root object: timestamp and variables. Timestamp is optional, and can be a single value or an array of values. Variables is an object containing arbitrary number of properties, which can be single values or arrays of values.

## Custom JSON format

In order to be able to fetch values from all kinds of JSON messages, the Nerve Data Services Gateway supports custom JSON data formats. The value of a field from a JSON message can be fetched using a so-called JSON path. A JSON path is comparable to a file path in a Unix-like file system. As an example, if a file is located in the following file structure:

- Documents
  - Personal
    - journal.txt

its file path would be: /Documents/Personal/journal.txt. Based on this analogy, a JSON field from the following JSON:

```

{
  "Values": {
    "Temperature": 20
  }
}

```

would have the following JSON path: .Values.Temperature. Arrays are represented with square brackets:

```

{
  "Values": {
    "Temperature": [ 20, 30, 40 ]
  }
}

```



The third value of the Temperature field would be represented the following way:  
.Values.Temperature[2]

## Gateway configuration for custom JSON formats

### NOTE

Custom JSON formats are supported in all inputs using JSON messages.

The fields name and path of the variables array objects inside of the connectors object are used to construct the JSON path. The full JSON path is always constructed by adding name to path:

The screenshot shows a 'Create input' configuration window. It has a 'Connectors' section with fields for 'Name', 'Topic\*' (set to 'demo-sensor-topic'), and 'Timestamp' (with a 'Path' field set to '.info.measurement\_time[]'). Below this is a 'Variables' section, which is highlighted with an orange border. The 'Variables' section contains fields for 'Name' (set to 'temperature'), 'Type\*' (set to 'int32'), 'Path' (set to '.measurement'), and 'Maximal values'. There are also several blue circular buttons with minus signs and a plus sign at the bottom left of the configuration area.

The example above will construct the following JSON path: `.measurement.temperature`

- The variable name through the rest of the Gateway will always be just the value of the name field. (temperature in the example above).
- If name is omitted, the value of path will become the name of the variable.
- If path is omitted, the value of path will be the default path `.variables`. This allows the Gateway to use the default Data Services data model JSON schema.
- To define a path for timestamps, a special `timestamp` object is used in the root of the input. The only field in the `timestamp` object is `path`, which is used to specify the path where to fetch timestamp values.
- This object is optional, and if omitted, the Gateway will look for timestamps under the default JSON path, `.timestamp`. This also allows the Gateway to use the default Data Services data model JSON schema.

### Timestamp JSON path example

The screenshot shows a configuration interface for creating an input. It is divided into three main sections: 'Connectors', 'Timestamp', and 'Variables'.  
1. **Connectors**: Contains a 'Name' field and a 'Topic\*' field with the value 'demo-sensor-topic'.  
2. **Timestamp**: This section is highlighted with an orange border. It contains a 'Path' field with the value '.info.measurement\_time[]'.  
3. **Variables**: Contains a variable configuration with 'Name' 'temperature', 'Type\*' 'int32', and 'Path' '.measurement'.  
The interface also features several blue circular buttons: a '+' button at the bottom left, and '-' buttons on the right side of the 'Variables' section.

The variable expansion operator [\*] cannot be used in a timestamp path. All additional requirements from the section above still apply.

### Extended JSON paths

The Gateway extends the JSON paths explained above with two new operators for fetching values, in order to reduce the need to write JSON paths for recurring fields. The following data is used for both expansion examples below:

```
{
  "machineB": {
    "sensor1": [
      {
        "temperature": 10,
        "humidity": 51
      },
      {
        "temperature": 20,
        "humidity": 52
      },
      {
        "temperature": 30,
        "humidity": 53
      }
    ]
  }
}
```

### Operator [] - List expansion

The list expansion is an operator used to fetch multiple values of the same variable. It allows processing large chunks of data in a simple way.

The data above, when received with the following configuration:

## Create input

### Variables

Name  
**temperature**

Type\*  
int32

Path  
.machineB.sensor1[]

Maximal values

Name  
**humidity**

Type\*  
int32

Path  
.machineB.sensor1[]

Maximal values

+

-

-

generates 3 entries on any Gateway output.

temperature	humidity
10	51
20	52
30	53

### Operator [\*] - Variable expansion

The variable expansion is an operator used to fetch multiple values of a variable as separate variables. It allows processing large chunks of data in a compact way.

The data above, when received with the following configuration:

### Create input

#### Variables

Name  
**temperature**

---

Type\*  
int32

---

Path  
**.machineB.sensor1[\*]**

---

Maximal values  
**3**

---

Name  
**humidity**

---

Type\*  
int32

---

Path  
**.machineB.sensor1[\*]**

---

Maximal values  
**4**

---

+
-
-
-

generates all data in one entry. Note that in the configuration above, only the operator was changed from [] to [\*].

temperature-0	temperature-1	temperature-2	humidity-0	humidity-1	humidity-2
10	20	30	51	52	53

An additional parameter is required when working with the variable expansion, `maxValues`. `maxValues` is required for outputs that need to know the exact number of data before actually receiving it (TimescaleDB, for example). If more values are received than specified in `maxValues`, the Gateway will simply ignore them. If less values than specified are received, the Gateway will write either the last known value at that

position or a default value based on the variable type. In the example above, humidity-3 is set to 0 because of this.

## Gateway configuration parameter descriptions

The following list gives more information on the parameters that can be set when configuring the Gateway. Parameters are listed by available inputs and outputs in the list below. The description of each parameter also includes information on additional parameters. The following inputs and outputs are included in the list:

Inputs	Outputs
MQTT Subscriber	MQTT Publisher
OPC UA PubSub Subscriber	OPC UA PubSub Publisher
OPC UA Client	OPC UA Server
Modbus Server	ZeroMQ Publisher
S7 Server	Azure IoT Hub Device
ZeroMQ Subscriber	Kafka Producer
	NerveDB
	TimescaleDB
	Influx DB

### Inputs

In the graphical configuration tool, mandatory fields are marked with a red asterisk. Note that defining connectors is mandatory even though it is not marked in the UI.

#### OPC UA Client

Object	Values and descriptions
<b>Name</b>	Enter a name for the input. This name is also used in log messages.
<b>Server URL</b>	Enter the URL of the server the input will connect to, i.e. <code>opc.tcp://myserver.com:4840</code> . This requires an OPC UA Server source to be configured.
<b>Login</b>	Enter <b>Username</b> and <b>Password</b> of the source OPC UA Server if authentication parameters have been defined.
<b>Certificate</b>	Enter the file names of the certificate under <b>Certificate path</b> and key file under <b>Key file path</b> for client authentication. This requires a certificate and key file to be uploaded through the <b>Import certificates</b> button in the Gateway's edit mode. Certificates must be in DER format. Example: <code>certificate.der</code> or <code>keyfile.der</code>
<b>Trust list</b>	Enter the certificate file names of trusted OPC UA Servers here. With added certificates here, the Gateway will only connect to OPC UA Servers presenting these certificates. Note that this requires the certificates to be uploaded through the <b>Import certificates</b> button in the Gateway's edit mode. Certificates must be in DER format. Example: <code>server_a_cert.der</code> , <code>server_b_cert.der</code> , <code>server_c_cert.der</code>

Object	Values and descriptions
<p><b>Security</b></p> <p><b>Security mode</b> Choices are None, Sign, SignAndEncrypt.</p> <p><b>Security policy</b> Choices are None, Basic128Rsa15, Basic256 and Basic256Sha256.</p> <p>This is the description of the OPC UA Client.</p> <p><b>Application URI</b> The application URI of the client is the globally unique identifier for the application instance, used in the secure connection certificate. If a certificate has been added earlier, this field must match the URI in the certificate.</p> <p><b>Description</b> Example: urn:myfactory.com:Machine54:UA Server</p> <p><b>Application Name</b> This is the OPC UA Client's string representation for a server. An OPC UA application delivers this when information on the client is requested. The default value will be used if this field is left empty. The default value of the Gateway OPC UA Client is Nerve Data Services - Gateway OPC UA Client.</p>	
<p><b>Force session to be kept alive</b></p>	<p>Tick this checkbox in order to force the server to keep the session alive by constantly polling an arbitrary server value. Some OPC UA servers have been observed having problems keeping the session alive.</p>
<p><b>Polling interval in ms</b></p>	<p>This is the interval at which the Gateway polls values from the server in milliseconds. This setting is in effect when the polling access type is used.</p>

Object	Values and descriptions
<b>Connectors</b>	<p>Select the plus icon to add a connector. Note that adding a connector is required for establishing a connection between inputs and outputs. Add another connector by selecting the plus icon at the end of the <b>Connectors</b> field that opened.</p> <p><b>Name</b> Enter a name for the connector. This makes the connector easy to identify when defining connections.</p> <p><b>Access type</b> This is the access mode of the Gateway OPC UA Client. Options are polling and subscription.</p> <ul style="list-style-type: none"> <li>• <b>polling</b> When the polling access type is used, the Gateway OPC UA Client sends requests to the OPC UA server for reading values of variables. The interval at which the Gateway OPC UA Client sends requests is defined in the <b>Polling interval in ms</b> setting above.</li> <li>• <b>subscription</b> When the subscription access type is used, the Gateway OPC UA Client subscribes to notifications on changes of variable values at the OPC UA Server. The sampling and publishing intervals are set below.</li> </ul>
	<p><b>Publishing interval at server in ms</b> Define the interval at which the server is publishing values in milliseconds. The publishing interval determines the interval at which the OPC UA server sends notification of changes in the values of variables, if there are any, to the Gateway OPC UA Client.</p> <p><b>Sampling interval at server in ms</b> Define the sampling interval at which the server updates its values in milliseconds. The sampling interval determines the interval at which the OPC UA server internally checks the values of variables for changes that have occurred.</p> <p><b>Nodes</b> Select the plus icon to add variable nodes:</p> <ul style="list-style-type: none"> <li>• <b>Node ID</b> Enter the Node ID string, e.g. <code>ns=4;i=1, ns=mynamespace;s=myvar.int0</code>. This string has to be entered according to the OPC UA Node ID string notation syntax. The format is <code>ns=&lt;namespaceIndex&gt;;&lt;identifiertype&gt;=&lt;identifier&gt;</code>. Refer to the <a href="#">official documentation</a> for more information.</li> <li>• <b>Custom name</b> Enter the custom name of the variable node. In case of node name collisions, the custom name can be used to resolve those manually. The custom name can also be used to shorten a node's name if it does not fit the requirements.</li> </ul> <p>If the Node ID of an object is provided, all variables of that object are used.</p>



## MQTT Subscriber

Object	Values and descriptions
<b>Name</b>	Enter a name for the input. This name is also used in log messages.
<b>Client ID</b>	Client ID of the MQTT_SUBSCRIBER.
<b>Server URL</b>	Protocol and URL of the server to connect to, i.e. tcp://myserver.com:4840. Possible or wss.
<b>Username</b>	Enter the username for the MQTT broker.
<b>Password</b>	Enter the password for the MQTT broker.
<b>Keep alive interval</b>	Define the maximum time allowed of no communication between client and server in
<b>Clean session</b>	Tick this checkbox to define whether the server should remember the state of the client
<b>QOS</b>	Quality of Service value. Possible values are 0, 1 or 2.
<b>SSL Options</b>	Secure connection options:
	<b>CA certificate path</b> Path to CA certificate file.
	<b>Certificate path</b> Path to certificate file.
	<b>Key file path</b> Path to key file.
	<b>Key password</b> Password for the given key.
	<b>Server authentication required</b> Tick this checkbox to activate server authentication.

Object	Values and descriptions
	<p>Select the plus icon to add a connector. Note that adding a connector is required for es and outputs. Add another connector by selecting the plus icon at the end of the <b>Conn</b> connectors for the MQTT_SUBSCRIBER:</p> <p><b>Name</b> Enter a name for the connector. This makes the connector easy to identify when defini</p> <p><b>Topic</b> Enter the name of the MQTT topic that was defined in the MQTT Publisher.</p> <p><b>Timestamp</b> Timestamp related options</p> <ul style="list-style-type: none"> <li>• <b>Path</b> JSON path pointing to the timestamp value(s).</li> </ul> <p><b>Variables</b> Array of variables expected to be received:</p> <ul style="list-style-type: none"> <li>• <b>Name</b> Enter a variable name as found in the received message. If omitted, <b>Path</b> is req</li> <li>• <b>Type</b> Select the data type the Gateway uses for further processing. This type must ma Default type is int64. Possible values are bool, sbyte, byte, int8, uint8, int16, int32, int64, uint64, double, float, string, bytestring, datetime or guid.</li> <li>• <b>Path</b> JSON path pointing to the value(s). If both <b>Name</b> and <b>Path</b> are provided, the pat</li> <li>• <b>Maximal values</b> Enter a maximum value that is required when using the expansion operator [*]</li> </ul>
<b>Connectors</b>	

## ZeroMQ Subscriber

Object	Values and descriptions
<b>type</b>	Type of input (ZEROMQ_SUBSCRIBER).
<b>Name</b>	Name of the ZEROMQ_SUBSCRIBER input used in log messages.
<b>serverUrls</b>	Array of server URLs for this subscriber instance to connect to, i.e. [tcp://myserver:

Object	Values and descriptions
	Array of connectors for ZEROMQ_SUBSCRIBER:
	<p><b>Name</b> Connector name.</p> <p><b>topic</b> Name of the ZeroMQ topic to use.</p> <p><i>timestamp</i> Timestamp related options.</p> <ul style="list-style-type: none"> <li>• <b>path</b> JSON path pointing to the timestamp value(s).</li> </ul>
<b>Connectors</b>	<p><b>variables</b> Array of variables expected to be received:</p> <ul style="list-style-type: none"> <li>• <b>Name</b> Variable name as found in the received message. If omitted, path is required and</li> <li>• <b>type</b> Data type the Gateway uses for further processing. It must match the receiving bool, sbyte, byte, int8, uint8, int16, uint16, int32, uint32, int64, uint64, datetime or guid.</li> <li>• <i>path</i> JSON path pointing to the value(s). If both name and path are provided, path will</li> <li>• <i>maxValues</i> Maximum values required when using the expansion operator [*] in the path. O</li> </ul>

## OPC UA PubSub Subscriber

Object	Values and descriptions
<b>type</b>	Type of input (OPC_UA_PS_SUBSCRIBER).
<b>Name</b>	Name of the OPC_UA_PS_SUBSCRIBER input used in log messages.
<i>mqttClientId</i>	Client ID of the OPC_UA_PS_SUBSCRIBER when MQTT transport is used. Required
<b>networkAddressUrl</b>	URL of the OPC UA PubSub connection to subscribe to.
<i>networkInterface</i>	Name of the network interface to use.
<i>encoding</i>	Type of message encoding (JSON/UADP).

**Object****Values and descriptions**

Array of connectors for the OPC\_UA\_PS\_SUBSCRIBER:

**Name**

Connector name.

*publisherId*

Publisher ID for the writer group.

*writerGroupId*

Writer group ID.

*dataSetWriterId*

Data set writer ID.

*fieldsReversePublished*

Some publishers send fields within a data set message in a reversed order compared to the publisher. If this field is set to true, the user can use field indexes in the configuration to these publishers. Must be set to true when connecting to a publisher using

*mqttTopic*

Name of the MQTT topic if transport protocol is MQTT, ignored otherwise. Required

**variables**

Array of variables expected to be received:

- *fieldIndex*  
Field index of the variable in the data set.
- **name**  
Name of the variable as found in the received message.
- **type**  
Data type of the variable. Possible values are bool, sbyte, byte, int8, uint64, double, float, string, bytestring, datetime or guid.

**Connectors****S7 Client**

Object	Values and descriptions
<b>type</b>	Type of output (S7_CLIENT).
<b>Name</b>	Name of the S7_CLIENT instance used in log messages.
<b>serverUrl</b>	URL of the S7 server to connect to.
<i>port</i>	Port at which to connect to.
<b>connectionType</b>	Connection type (PG, OP, S7_BASIC).
<i>localTsap</i>	Local tsap, mandatory if s7 basic connection type.
<i>remoteTsap</i>	Remote tsap, mandatory if s7 basic connection type.
<i>pollingInterval_ms</i>	Interval for polling values from server in milliseconds.
<i>rack</i>	S7 device rack.
<i>slot</i>	S7 device slot.

Array of connectors for S7\_CLIENT:

**Name**

Connector name.

*markers*

Array of S7 markers:

- **name**  
S7 merker name.
- **offset**  
Address of first data.
- *quantity*  
Number of markers to be read.
- **type**  
S7 merker data type.

*inputs*

Array of S7 inputs:

- **name**  
S7 input name.
- **offset**  
Address of first data.
- *quantity*  
Number of inputs to be read.
- **type**  
S7 input data type.

*outputs*

Array of S7 outputs:

- **name**  
S7 output name.
- **offset**  
Address of first data.
- *quantity*  
Number of outputs to be read.
- **type**  
S7 output data type.

**Connectors**

*timers*

Array of S7 timers:

- **name**  
S7 timer name.
- **offset**  
Address of first data.
- *quantity*  
Number of timers to be read.

*counters*

Array of S7 counters:

- **name**  
S7 counter name.
- **offset**  
Address of first data.
- *quantity*  
Number of counters to be read.

**datablocks**  
Array of S7 datablocks:

- **name**  
S7 datablock name.

## Modbus Client

Object	Values and descriptions
<b>type</b>	Type of output (MODBUS_CLIENT).
<b>Name</b>	Name of the MODBUS_CLIENT instance used in log messages.
<b>serverUrl</b>	URL of the Modbus server to connect to.
<b>port</b>	Port at which to connect to.
<i>pollingInterval_ms</i>	Interval at which to poll values from the server in milliseconds.

Object	Values and descriptions
<b>Connectors</b>	<p>Array of connectors for the MODBUS_CLIENT:</p> <p><b>Name</b> Connector name.</p> <p><i>coils</i> Array of Modbus coil type:</p> <ul style="list-style-type: none"> <li>• <b>name</b> Coil name, single string or an array of string.</li> <li>• <b>address</b> Starting address of first instance.</li> <li>• <i>quantity</i> Number of coils to be read.</li> </ul> <p><i>discreteInputs</i> Array of Modbus discrete input type:</p> <ul style="list-style-type: none"> <li>• <b>name</b> Discrete input name, single string or an array of string.</li> <li>• <b>address</b> Starting address of first instance.</li> <li>• <i>quantity</i> Number of discrete inputs to be read.</li> </ul>
	<p><i>inputRegisters</i> Array of Modbus input register type:</p> <ul style="list-style-type: none"> <li>• <b>name</b> Input register name, single string or an array of string.</li> <li>• <b>address</b> Starting address of first instance.</li> <li>• <i>quantity</i> Number of input registers to be read.</li> <li>• <b>type</b> Input register data type.</li> </ul> <p><i>holdingRegisters</i> Array of Modbus holding register type:</p> <ul style="list-style-type: none"> <li>• <b>name</b> Holding register name, single string or an array of string.</li> <li>• <b>address</b> Starting address of first instance.</li> <li>• <i>quantity</i> Number of holding registers to be read.</li> <li>• <b>type</b> Holding register data type.</li> </ul>

## Outputs

In the graphical configuration tool, mandatory fields are marked with a red asterisk. Note that defining connectors is mandatory even though it is not marked in the UI.

## TimescaleDB

Object	Values and descriptions
<b>type</b>	Type of output (DB_TIMESCALE).
<b>Name</b>	Name of the DB_TIMESCALE instance used in log messages.
<b>url</b>	URL of the TimescaleDB server to connect to.
<i>port</i>	Port at which to connect to.
<i>dataRate_MBps</i>	Hypertable data rate in MegaBytes per second.
<i>chunkTimeInterval_s</i>	Hypertable chunk time interval in seconds.
	Array of connectors for DB_TIMESCALE:
	<b>Name</b> Connector name.
	<i>dbName</i> Database name to connect to.
<i>connectors</i>	<i>tableName</i> Table name to write into.
	<i>user</i> Username for authentication.
	<i>password</i> Password for authentication.
	<i>booleanAsSmallint</i> Whether boolean should be represented as PostgreSQL small int data type.

## InfluxDB

Object	Values and descriptions
<b>type</b>	Type of output (DB_INFLUX).
<b>Name</b>	Name of the DB_INFLUX instance used in log messages.
<b>url</b>	URL of the InfluxDB server to connect to.
<i>port</i>	Port at which to connect to.
	Array of connectors for DB_INFLUX:
	<b>Name</b> Connector name.
<b>Connectors</b>	<b>dbName</b> Database name to connect to.
	<i>user</i> Username for authentication.
	<i>password</i> Password for authentication.



## MQTT Publisher

Object	Values and descriptions
<b>Name</b>	Name of the MQTT_PUBLISHER instance used in log messages.
<b>Client id</b>	Client ID of the MQTT_PUBLISHER.
<b>Server URL</b>	Protocol and URL of the server to connect to, i.e. tcp://myserver.com:4840. Possible protocols are: tcp, mqtt, mqtts, ssl, ws or wss.
<b>Username</b>	MQTT broker authentication username.
<b>Password</b>	MQTT broker authentication password.
<b>Keep alive interval in s</b>	Maximum time allowed of no communication between client and server.
<b>Clean session</b>	Defines whether the server should remember state for the client across reconnect.
<b>Qos</b>	Quality of Service value (0-2).
	Secure connection options:
	<b>CA certificate path</b> Path to CA certificate file.
	<b>Certificate path</b> Path to certificate file.
<b>SSL options</b>	<b>Key file path</b> Path to key file.
	<b>Key password</b> Password for the given key.
	<b>Server authentication required</b> Defines whether server authentication is required.

Object	Values and descriptions
	<p>Array of connectors for MQTT_PUBLISHER:</p> <p><b>Name</b> Connector name.</p> <p><b>Topic</b> Name of the MQTT topic to use.</p> <p><b>Timestamp required</b> Defines whether a timestamp is added to each message or not.</p> <p><b>Timestamp format</b> Format of the timestamp in a message, either <code>iso</code> (ISO 8601) or <code>unix_ns</code> (UNIX time in nanoseconds since Jan 01 1970 (UTC)).</p> <p><b>Max list size</b> Maximum array size of variables, if sent as arrays. If size is larger than the max data will be sent in chunks of Max list size.</p>
<b>Connectors</b>	<p><b>Timestamp</b> Timestamp options:</p> <ul style="list-style-type: none"> <li>• <b>Path</b> Timestamp JSON path in the outgoing message.</li> </ul> <p><b>Variables</b> Array of variables for which a custom JSON path is desired:</p> <ul style="list-style-type: none"> <li>• <b>Name</b> Name of the variable.</li> <li>• <b>New name</b> New name of the variable.</li> <li>• <b>Path</b> JSON path of the variable.</li> <li>• <b>Default value</b> Default value of the variable which is being constantly sent until a value is received at the input.</li> <li>• <b>toCombine</b> Array of names of variables to combine into this one.</li> </ul>

### OPC UA PubSub Publisher

Object	Values and descriptions
<b>type</b>	Type of output (OPC_UA_PS_PUBLISHER).
<b>Name</b>	Name of OPC_UA_PS_PUBLISHER instance used in log messages.
<b>networkAddressUri</b>	URL of the OPC UA PubSub connection to publish to.
<i>networkInterface</i>	Name of the network interface to use.
<i>encoding</i>	Type of message encoding (JSON/UADP).

Object	Values and descriptions
<b>Connectors</b>	Array of connectors for OPC_UA_PS_PUBLISHER:
	<b>Name</b> Connector name.
	<b>publisherId</b> Publisher ID of OPC UA PubSub connection.
	<b>writerGroupId</b> Writer group ID in the OPC UA PubSub connection.
	<b>dataSetWriterId</b> Data set writer ID in the writer group.
	<i>publishInterval_ms</i> Interval at which publisher publishes messages.
	<i>keyFrameCount</i> Key frame count for messages.
	<i>mqttTopic</i> Topic to publish to in case of MQTT connection.
	<i>mqttClientId</i> MQTT client ID for PubSub connection in case of MQTT connection. This must be unique for each connector. Required when MQTT is used.

## OPC UA Server

Object	Values and descriptions
<b>type</b>	Type of output (OPC_UA_SERVER).
<b>Name</b>	Name of OPC_UA_SERVER instance used in log messages.
<i>customHostname</i>	Custom hostname of the server, used for discovery URL.
<i>port</i>	Port to listen at (defaults to 4840).
<i>logins</i>	Array of username/password pairs available for authentication:
	<b>username</b> Username for authentication.
	<b>password</b> Password for authentication.
<i>securities</i>	Array of available security modes and policies:
	<b>securityMode</b> Security Mode ("all" keyword is supported to create all available endpoints for chosen Security Policy).
	<b>securityPolicy</b> Security Policy ("all" keyword is supported to create all available endpoints for chosen Security Mode).

Object	Values and descriptions
	Server's certificate:
<i>certificate</i>	<p><b>certFilePath</b> Path to Server Instance Certificate.</p> <p><b>keyFilePath</b> Path to private key.</p>
<i>description</i>	<p>Server Description:</p> <p><i>applicationUri</i> Server Application URI.</p> <p><i>applicationName</i> This is the client's string representation for a server.</p>
<i>trustList</i>	Array of trusted certificates.
<i>addressSpaceFolders</i>	<p>OPC-UA_SERVER address space tree:</p> <p><b>browseName</b> OPC UA Folder browse name.</p> <p><i>displayName</i> OPC UA Folder display name.</p> <p><i>description</i> OPC UA Folder description.</p> <p><i>namespaceURI</i> OPC UA Folder namespace URI.</p> <p><b>identifier</b> OPC UA Folder nodeID identifier.</p> <p><i>children</i> Array of OPC UA childFolders, same format as addressSpaceFolders field.</p> <p><i>connectorIndicies</i> Array of indicies of connectors, placed in the OPC UA Folder.</p>

Object	Values and descriptions
<b>Connectors</b>	Array of connectors for OPC_UA_SERVER:
	<b>Name</b> Connector name.
	<b>browseName</b> OPC UA node browse name.
	<i>displayName</i> OPC UA node display name.
	<i>description</i> OPC UA node description.
	<i>namespaceURI</i> OPC UA node namespace URI.
	<b>identifier</b> OPC UA node nodeID identifier.

### ZeroMQ Publisher

Object	Values and descriptions
<b>type</b>	Type of output (ZEROMQ_PUBLISHER).
<b>Name</b>	Name of the ZEROMQ_PUBLISHER instance used in log messages.
<b>serverUrl</b>	Server endpoint to publish to. Subscribers connect to it.

Object	Values and descriptions
	<p>Array of connectors for ZEROMQ_PUBLISHER:</p> <p><b>Name</b> Connector name.</p> <p><b>topic</b> Name of the ZeroMQ topic to publish to.</p> <p><i>timestampRequired</i> Defines whether a timestamp is added to each message or not.</p> <p><i>timestampFormat</i> Format of the timestamp in a message, either <i>iso</i> (ISO 8601) or <i>unix_ns</i> (UNIX time in nanoseconds since Jan 01 1970 (UTC)).</p> <p><i>maxListSize</i> Maximum array size of variables, if sent as arrays. If size is larger than the max data will be sent in chunks of <i>maxListSize</i>.</p> <p><b>Connectors</b> <i>timestamp</i> Timestamp options:</p> <ul style="list-style-type: none"> <li>• <i>path</i> Timestamp JSON path in the outgoing message.</li> </ul> <p><i>variables</i> Array of variables for which a custom JSON path is desired:</p> <ul style="list-style-type: none"> <li>• <b>Name</b> Name of the variable.</li> <li>• <i>newName</i> New name of the variable.</li> <li>• <i>path</i> JSON path of the variable.</li> <li>• <i>defaultValue</i> Default value of the variable which is being constantly sent until a value is received at the input.</li> <li>• <i>toCombine</i> Array of names of variables to combine into this one.</li> </ul>

## Azure IoT Hub Device

Object	Values and descriptions
<b>type</b>	Type of output (AZURE_IOT_HUB_DEVICE).
<b>Name</b>	Name of the AZURE_IOT_HUB_DEVICE instance used in log messages.
<b>deviceConnectionString</b>	Connection string of the device created in the MS Azure IoT Hub.
<i>mqttOverWss</i>	Determines if the AZURE_IOT_HUB_DEVICE instance should use MQTT over WSS (port 433) or regular MQTT over TLS/SSL (port 8883) to connect to the Azure IoT Hub broker.

Object	Values and descriptions
<i>sslOptions</i>	Secure connection options for MQTT transport:
	<b>certFilePath</b> Path to certificate file.
	<b>keyFilePath</b> Path to key file.
	<i>keyPassword</i> Password for the given key.

Object	Values and descriptions
<i>connectors</i>	<p>Array of connectors for AZURE_IOT_HUB_DEVICE:</p> <p><b>Name</b> Connector name.</p> <p><i>timestampRequired</i> Defines whether a timestamp is added to each message or not.</p> <p><i>timestampFormat</i> Format of the timestamp in a message, either iso (ISO 8601) or unix_ns (UNIX time in nanoseconds since Jan 01 1970 (UTC)).</p> <p><i>maxListSize</i> Maximum array size of variables if sent as arrays. If size is larger than the maximum, data will be sent in chunks of maxListSize.</p> <p><i>customProperties</i> Custom properties array that is included with every published message:</p> <ul style="list-style-type: none"> <li>• <b>name</b> Name of the custom property.</li> <li>• <b>value</b> Value of the custom property. May be a constant value, or the value from one of the input variables of the message, in which case it should be the variable name enclosed in { }.</li> </ul> <p><i>timestamp</i> Timestamp options:</p> <ul style="list-style-type: none"> <li>• <i>path</i> Timestamp JSON path in the outgoing message.</li> </ul> <p><i>variables</i> Array of variables for which a custom JSON path is desired:</p> <ul style="list-style-type: none"> <li>• <b>Name</b> Name of the variable.</li> <li>• <i>newName</i> New name of the variable.</li> <li>• <i>path</i> JSON path of the variable.</li> <li>• <i>defaultValue</i> Default value of the variable which is being constantly sent until a value is received at the input.</li> <li>• <i>toCombine</i> Array of names of variables to combine into this one.</li> </ul>



## NerveDB

Object	Values and descriptions
<b>type</b>	Type of output (NERVE_DB).
<b>Name</b>	Name of the NERVE_DB instance used in log messages.
<b>location</b>	Location of the Nerve database. This can be CENTRAL or LOCAL.
<i>connectors</i>	Array of connectors for NERVE_DB:  <b>Name</b> Connector name.  <i>tableName</i> Table name to write into.  <i>dataBuffer</i> Enables data buffering option. This is only usable if location is set to CENTRAL.  <b>bufferExpTime</b> Data retention time of the buffer in minutes. This is mandatory if data buffering is enabled and the location is CENTRAL.

## Kafka Producer

Object	Values and descriptions
<b>type</b>	Type of output (KAFKA_PRODUCER).
<b>Name</b>	Name of the KAFKA_PRODUCER instance used in log messages.
<b>bootstrapServers</b>	Array of bootstrap servers URLs.
<i>clientId</i>	Client ID to use for the Kafka client.
<i>securityProtocol</i>	Security protocol to use when connecting to the Kafka broker. Options are plaintext and ssl.

Object	Values and descriptions
<i>sslOptions</i>	<p>Object describing SSL options. Only relevant if securityProtocol is SSL.</p> <p><i>sslCertificateLocation</i> Path to the certificate.</p> <p><i>sslKeyLocation</i> Path to the certificate key.</p> <p><i>sslKeyPassword</i> Key password.</p> <p><i>sslCaLocation</i> Path to the CA certificate.</p> <p><i>sslCrlLocation</i> Path to the CRL.</p> <p><i>sslKeyStoreLocation</i> Path to the key store directory.</p> <p><i>sslKeyStorePassword</i> Key store password.</p>

Object	Values and descriptions
<b>Connectors</b>	Array of connectors for KAFKA_PRODUCER:
	<p><b>Name</b> Connector name.</p>
	<p><b>topic</b> Name of the Kafka topic to produce to.</p>
	<p><i>compressionType</i> Compression algorithm to use to compress the outgoing messages. Options are none, gzip, snappy, lz4 and zstd.</p>
	<p><i>timestampRequired</i> Defines whether a timestamp is added to each message or not.</p>
	<p><i>timestampFormat</i> Format of the timestamp in a message. Options are iso (ISO 8601) or unix_ns (UNIX time in nanoseconds since Jan 01 1970 (UTC)).</p>
	<p><i>maxListSize</i> Maximum array size of variables, if sent as arrays. If size is larger than the max data will be sent in chunks of maxListSize.</p>
	<p><i>timestamp</i> Timestamp options:</p> <ul style="list-style-type: none"> <li>• <i>path</i> Timestamp JSON path in the outgoing message.</li> </ul>
	<p><i>variables</i> Array of variables for which a custom JSON path is desired:</p> <ul style="list-style-type: none"> <li>• <b>Name</b> Name of the variable.</li> <li>• <i>newName</i> New name of the variable.</li> <li>• <i>path</i> JSON path of the variable.</li> <li>• <i>defaultValue</i> Default value of the variable which is being constantly sent until a value is received at the input.</li> <li>• <i>toCombine</i> Array of names of variables to combine into this one.</li> </ul>

### Connections

Object	Values and descriptions
<b>Name</b>	Name of the connection, will define the name of the TABLE within TimescaleDB if tableName field is omitted.

Object	Values and descriptions
	<p>Mode used by the output to handle data in the buffer.</p> <p><i>ALL</i> This is the default mode. All variables that are in the data set are used by the output, regardless of the value having changed or not. If no new value has been received, the previous one is used. All inputs and outputs support this mode.</p> <p><i>INCREMENTAL</i> Only variables for which new values were received are considered. Currently this is only supported by OPC UA Client, JSON inputs (ZeroMQ Subscriber and MQTT Subscriber) and JSON outputs (ZeroMQ Publisher, MQTT Publisher and Kafka Producer).</p>
	<p>Input of the connection.</p>
<b>input</b>	<p><b>index</b> Index in the array of INPUTs.</p> <p><b>connector</b> Index in the array of connectors at the input.</p>
	<p>Output of the connection.</p>
<b>output</b>	<p><b>index</b> Index in the array of OUTPUTs.</p> <p><b>connector</b> Index in the array of connectors at the output.</p>

## Gateway configuration file keywords

Parts of the Gateway configuration are eased by the use of keywords. All keywords must be put in angled brackets, for example <SN>. To be able to use keywords, the node must be registered in the Management System.

Refer to the list below for keywords that can be used with usage examples right after. Note that keywords can also be used in the graphical configuration tool.

Keyword	Description
<b>SN</b>	Replaces the keyword with the serial number of the node.
<b>SID</b>	<p>Replaces the keyword with the secure ID of the node.</p> <ul style="list-style-type: none"> <li>• Replaces the keyword with localhost.</li> <li>• Used in url field of TimescaleDB output.</li> <li>• Adds port field to the TimescaleDB output.</li> <li>• Adds connectors array with a single connector to the TimescaleDB output if none exist. Adds or replaces name, dbName, user and password fields for all connectors.</li> </ul>
<b>LOCAL</b>	<p>Note that the usage of this keyword is an old way for configuring a local database instance. Use the NerveDB output with location set to LOCAL instead for usage of a pre-configured local database. The TimescaleDB output should only be used for custom TimescaleDBs installed by users on the node or on an external device.</p>

## Examples of keyword usage

### LOCAL

```
{
  "type": "DB_TIMESCALE",
  "name": "timescaledb_0",
  "url": "<LOCAL>"
}
```

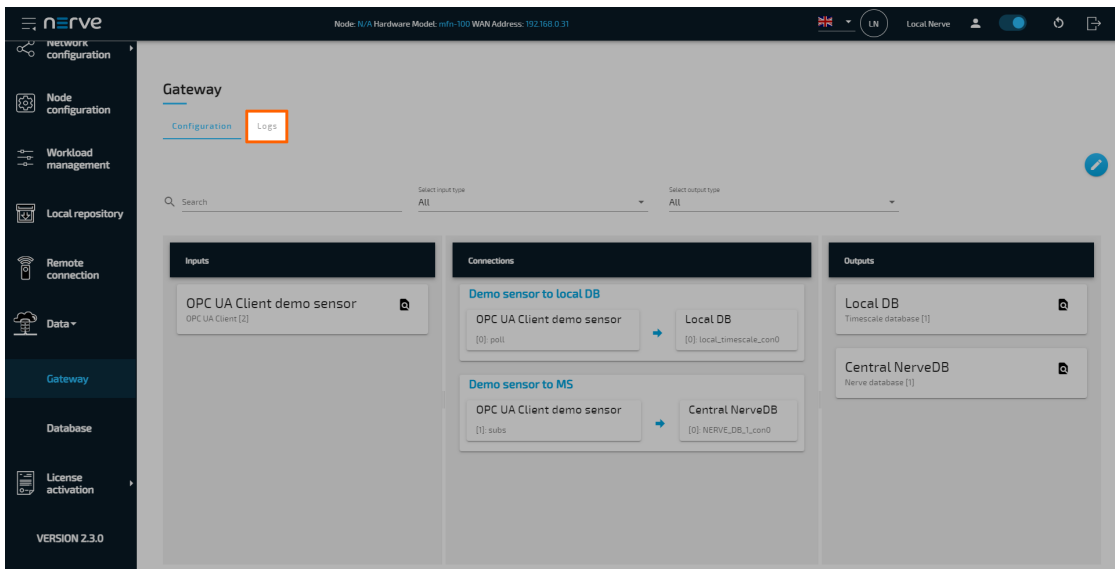
### SN and SID

```
{
  "type": "MQTT_SUBSCRIBER",
  "serverUrl": "tcp://localhost:1883",
  "username": "<SN>",
  "password": "<SID>",
  "keepAliveInterval_s": 20,
  "qos": 1,
  "cleanSession": true,
  "connectors": [
    {
      "topic": "<SN>",
      "variables": [
        {
          "name": "temperature",
          "type": "uint64"
        }
      ]
    }
  ]
}
```

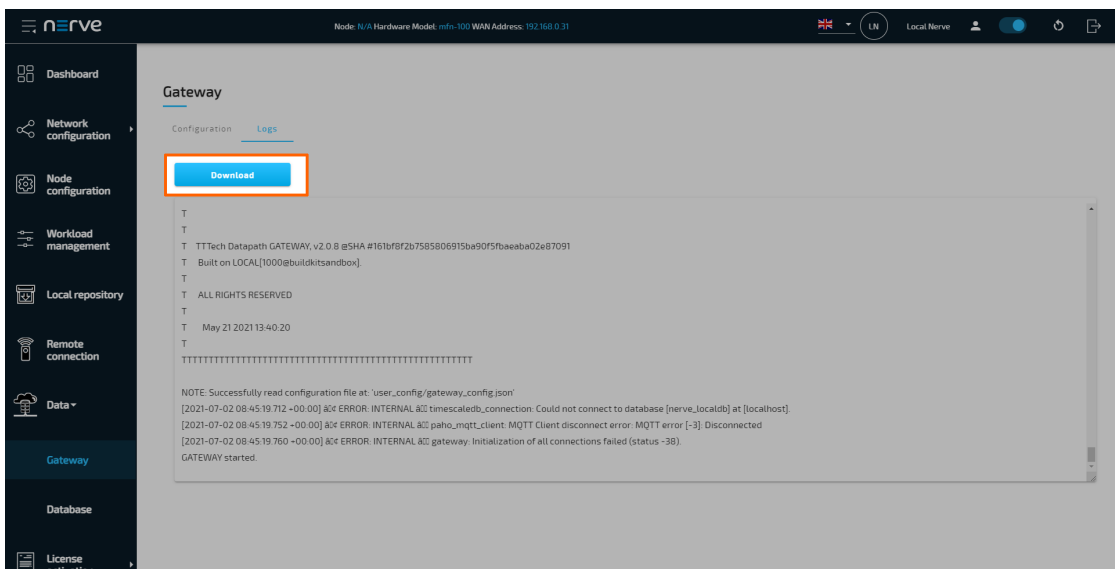
## How to export a log file

Every element of the Data Services is producing log files locally and centrally that can be viewed in the **Logs** tab. These logs can also be downloaded through the Data Services UI.

1. Select **Gateway** in the navigation on the left.
2. Select the **Logs** tab.



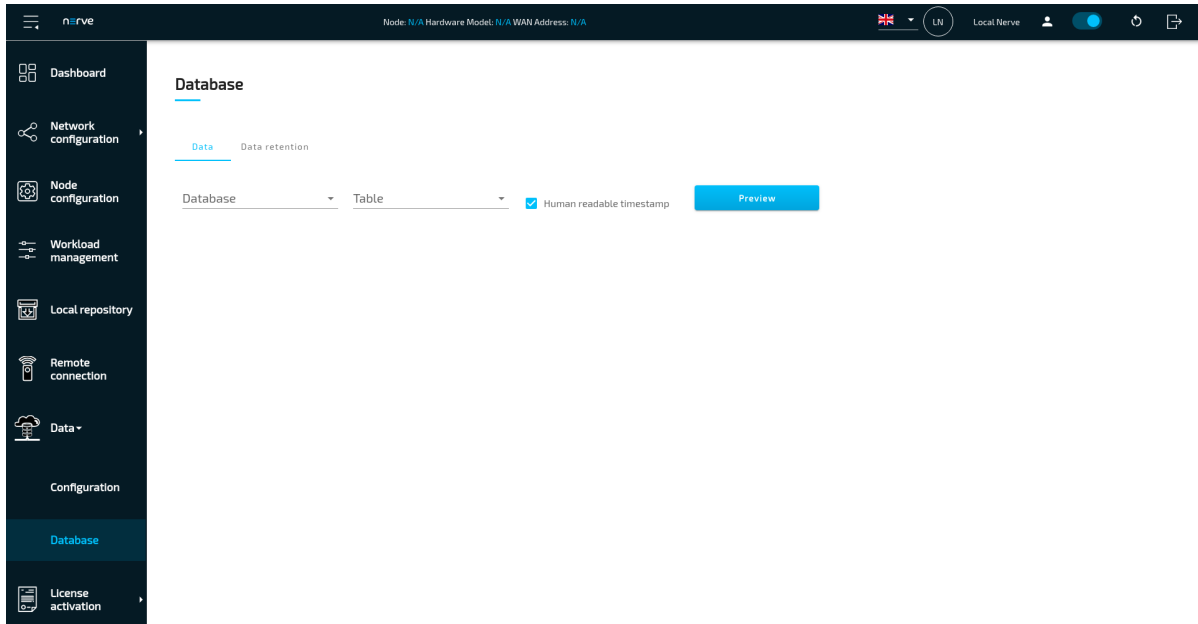
3. Select **Download** to export a LOG file.



## Nerve Data Services Database

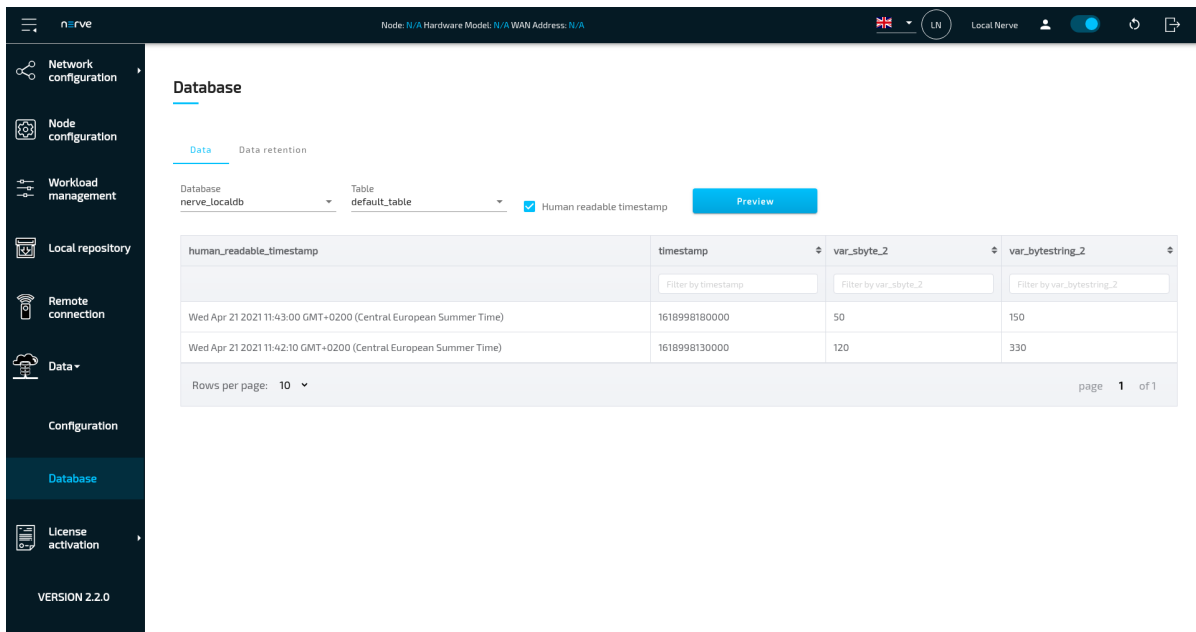
Every node has a local NerveDB database pre-configured. The local NerveDB is also the database that should be used if local database storage is required. In the Management System, databases are discerned by the serial numbers of the nodes and can be used to confirm the flow of data. Select the arrow next to **Data** in the navigation on the left in both the Local UI and the Management System and select **Database** to reach the database menu. The database menu can be used to preview data collected in the NerveDB of a node.

The **Data** tab is the first active tab when selecting the database menu in the navigation.



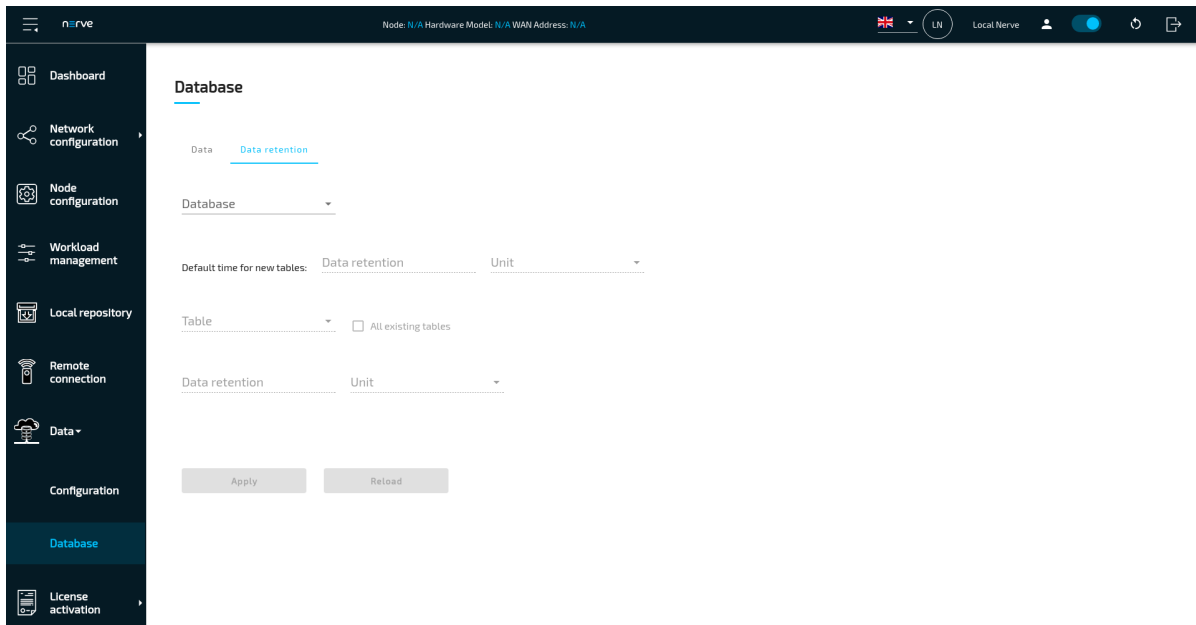
Item	Description
<b>Database</b>	Select a database from the drop-down menu. In the Local UI this will only contain the default database <b>nerve_localdb</b> . In the Management System, this will contain a list of the serial numbers of all registered nodes.
<b>Table</b>	Select a table containing data from the drop-down menu. The selection here depends on the configuration of the Gateway. Data needs to be written to <b>nerve_localdb</b> for something to be listed here.
<b>Human readable timestamp</b>	Tick the checkbox here to add a timestamp to the data preview of the data contained in the database.
<b>Preview</b>	Select this to show a preview of data. The data shown depends on the selections in the <b>Database</b> and <b>Table</b> drop-down menus.

Here is an example of a data preview.



## Data retention tab

Due to limitation in storage, a data retention policy is in place to delete old data after a certain amount of time. The default time is one day. Select the **Data retention** tab for settings. Refer to the table below for more information on the settings.



Item	Description
<b>Database</b>	Select a database from the drop-down menu. In the Local UI this will only contain the default database <b>nerve_localdb</b> . In the Management System, this will contain a list of the serial numbers of all registered nodes.

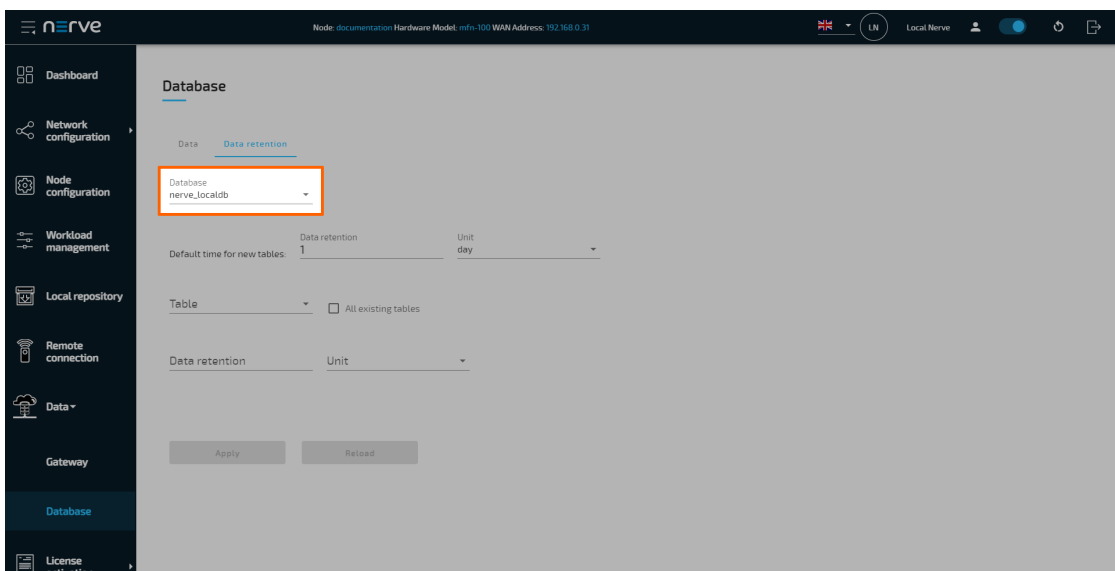


Item	Description
<b>Default time for new tables</b>	Set the data retention time for new tables here. Times can be set in <b>minutes, hours</b> and <b>days</b> . When setting the data retention parameters for the first time, the default value of 1 day is filled in automatically. This default value is also automatically applied to existing tables when the retention time is initially set. After that, this value applies to tables added in the future when the Gateway configuration is changed.
<b>Table</b>	Select a table from the drop-down menu to configure its data retention time. The drop-down menu contains table names defined in the Gateway configuration if data has been written into the database that was selected above.
<b>All existing tables</b>	Tick this checkbox to apply the data retention time to all tables in the drop-down menu.
<b>Data retention</b>	Set the data retention time for the selected tables here. Times can be set in <b>minutes, hours</b> and <b>days</b> .
<b>Apply</b>	Use this button to save the configuration.
<b>Reload</b>	Use this button to load the current settings of the selected tables into the UI.

## Setting data retention time

The following example shows a possible way of setting different data retention settings for different tables in a database.

1. Apply a Gateway configuration.
2. Select **Data > Database** in the navigation on the left.
3. Select the **Data retention** tab.
4. Select a database from the drop-down menu.



### NOTE

The drop-down menu is labelled **Database** in the Local UI and contains the **nerve\_localdb** database

In the Management System, this drop-down menu is labelled **Nodes** and contains the serial numbers of all registered nodes.

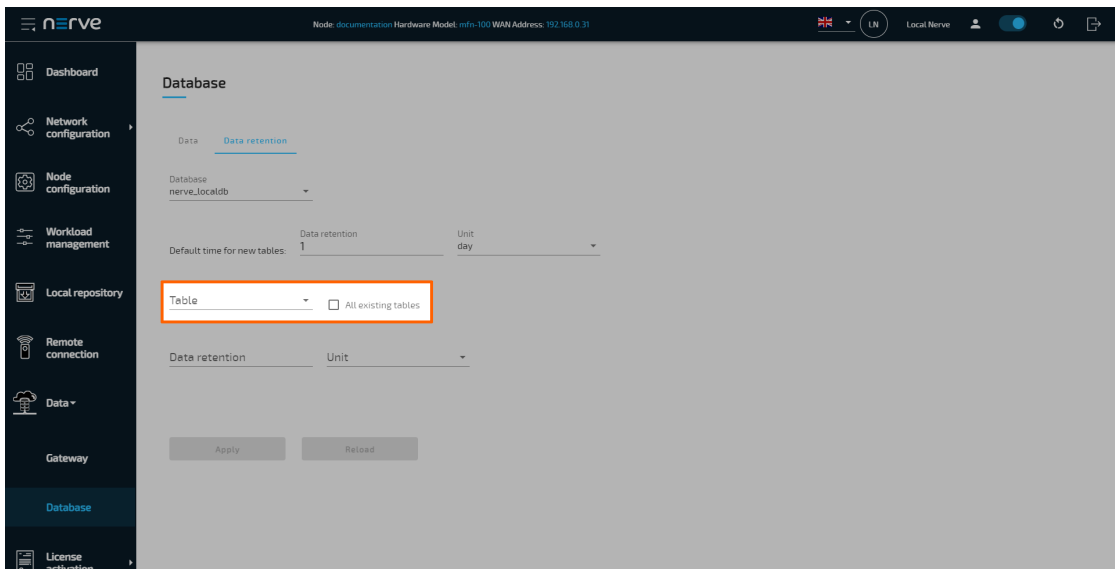
5. Set the **Default time for new tables** to the desired value.

#### NOTE

When setting the data retention parameters for the first time, the default value of **1 day** is filled in automatically. This default value is also automatically applied to the currently existing tables. After that, this value applies to tables added in the future when the Gateway configuration is changed.

Example: If the value is changed from **1 day** to **2 days**, newly defined tables that are added with the next Gateway configuration will have a data retention time of **2 days** while the existing tables remain at a data retention time of **1 day**. Change this value to a desired value for future tables. The retention time of the existing tables is configured below.

6. Select a table from the drop-down menu or tick the checkbox next to **All existing tables**.



7. Set the value to the desired amount of time.

#### NOTE

This setting allows the definition of retention time for existing tables. The time value below this selection is applied to table selected in step 6 above. The retention time of all currently available tables can be defined one by one by selecting each table from the drop-down menu, setting a data retention time below and selecting **Apply**. If all existing tables are to have the same data retention time, tick the checkbox next to **All existing tables**, set a data retention time below and select **Apply**.

After selecting **Apply**, data retention settings are applied. Note that the currently data retention time of an existing table is displayed in the settings when the table is selected from the drop-down menu.

## Data Visualization

Data stored in the Data Services can be visualized via Grafana, an open source web application which provides charts, graphs and alerts for data visualization. An instance of Grafana is available on each node and in the Management System to visualize data stored in databases on the node or in the Management System. This chapter describes only how Grafana is configured and how it can be accessed. Refer to the [Grafana documentation](#) for general information on how to use Grafana.

### NOTE

To use Grafana, the user must be logged in to the Local UI or the Management System respectively. When logged out, Grafana is no longer able to perform its internal operations and reports random errors. To continue working with Grafana, close the browser tab, log in again at either the Local UI or the Management System, depending on which Grafana instance is being accessed, and re-open Grafana.

## Central Data Visualization in the Management System

In the Management System, each registered node is represented as a separate data source in Grafana. The data source is created during the registration of a node. A data source is named after the node it represents, formatted as <nodename> (<serialnumber>). When creating a new panel, select the node which sent the data to be displayed as the data source. The name of the table must match the name provided in the Gateway configuration file.

1. Log in to the Management System.
2. Select **Data** in the navigation on the left.

### NOTE

If the menu item **Data** is not available, make sure the logged in user has the permission to access the Data Services. Refer to [Assigning a role to a user](#) for more information.

The navigation on the left collapses to show the home screen of Grafana. Here dashboards can be created as described [below](#). Select the burger menu in the top-left to expand the navigation again. Note that the Grafana UI theme is set to light by default.

## Access levels for Data Visualization in the Management System

Management System users are assigned Grafana permissions when they receive permissions to access the Data Services. The roles can be either viewer, editor or admin:

- Viewers can view dashboards in Grafana.

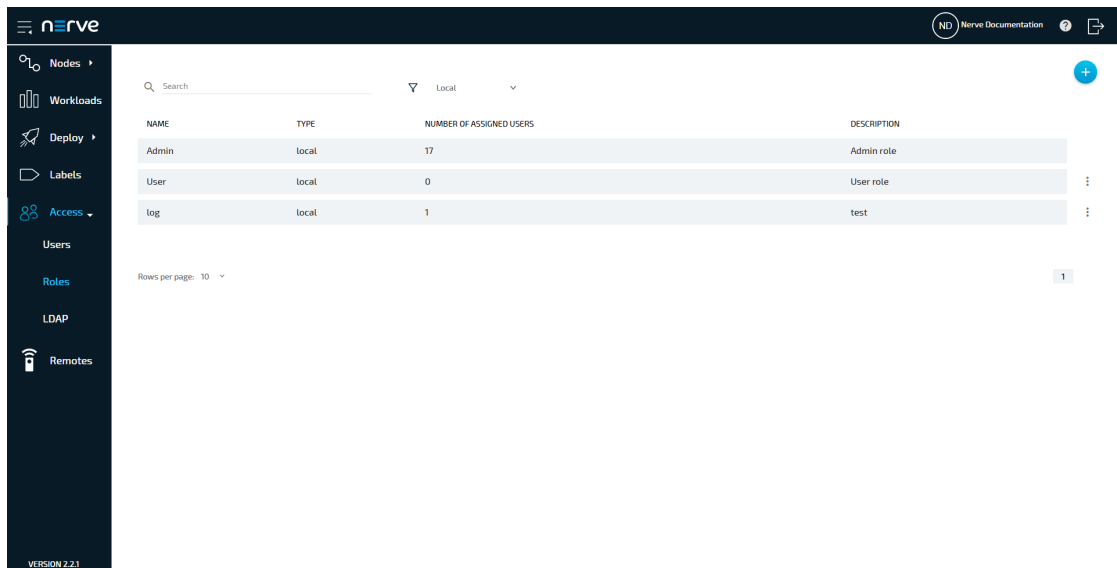
- Editors can view, create and edit dashboards.
- Admins can manage data sources and users.  
Note that this is not recommended to do as Nerve components do this automatically.

## NOTE

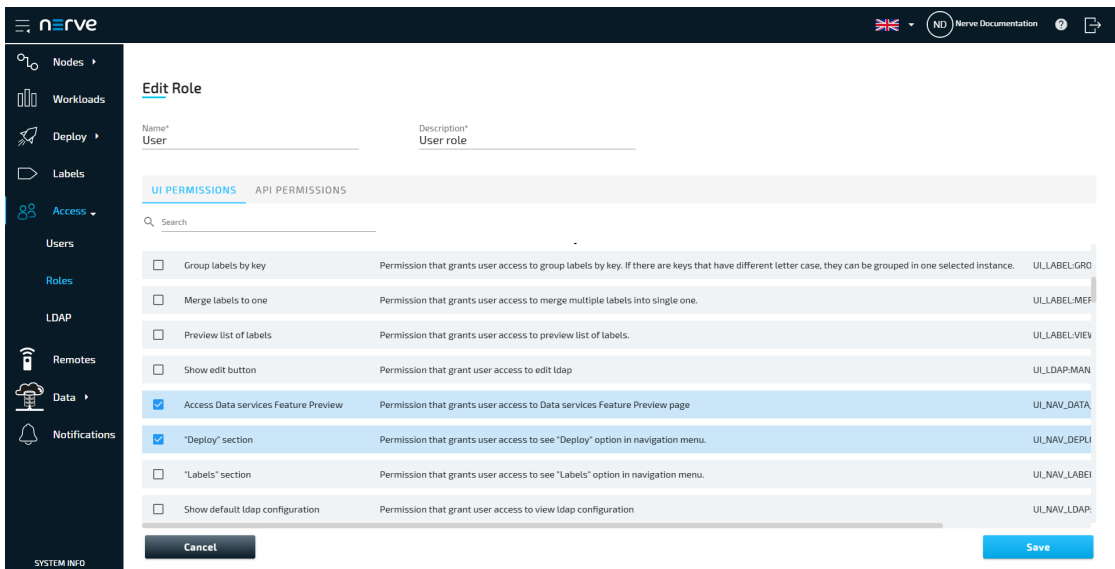
Users with the **GRAFANA:ADMIN** permission shall not delete the default admin user, as the system relies on it. If the default admin user is deleted, the assigned roles from the Management System will not work.

The Grafana Viewer role is given by default. To give a user the editor or admin role:

1. Log in to the Management System.
2. Select **Access > Roles** in the navigation on the left.

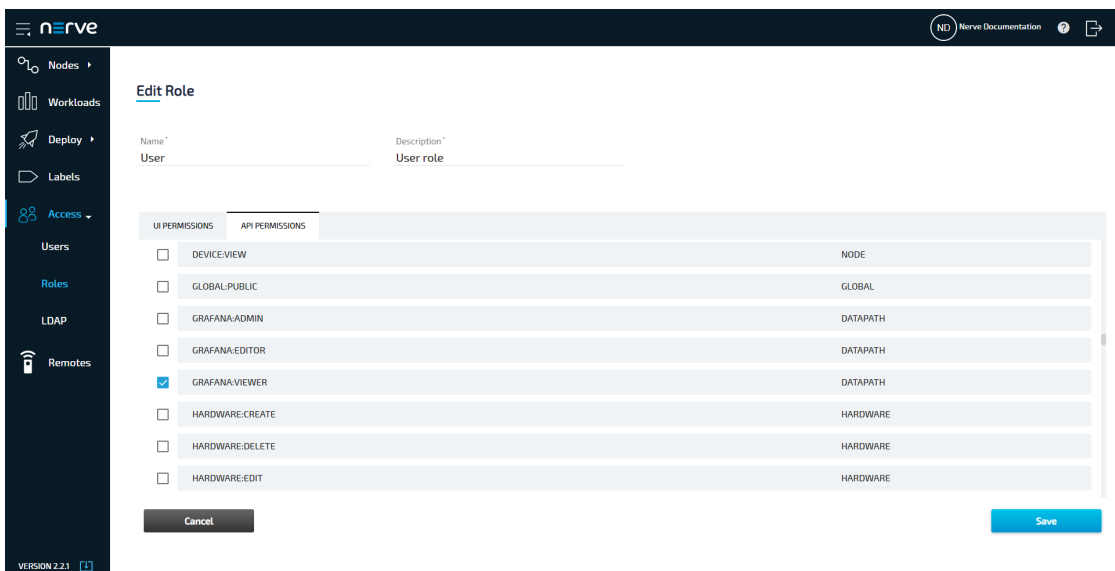


3. Select the role that has permission to access the Data Services.
4. Make sure that **Access Data services Feature Preview** is checked.



5. Select the **API PERMISSIONS** tab.

6. Look for **GRAFANA:ADMIN** and **GRAFANA:EDITOR**



7. Select the desired permission.

8. Select **Save**

## Local Data Visualization on the node

On the node, only a single data source exists in Grafana, formatted as <serialnumber>. Use this data source in all panels to visualize data received by the Data Services.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
MFN 100	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>

Nerve Device	Physical port	Local UI
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.
<b>Kontron KBox A-250</b>	ETH 2	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.
<b>Maxtang AXWL10</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer 5029C-T</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Winmate EACIL20</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

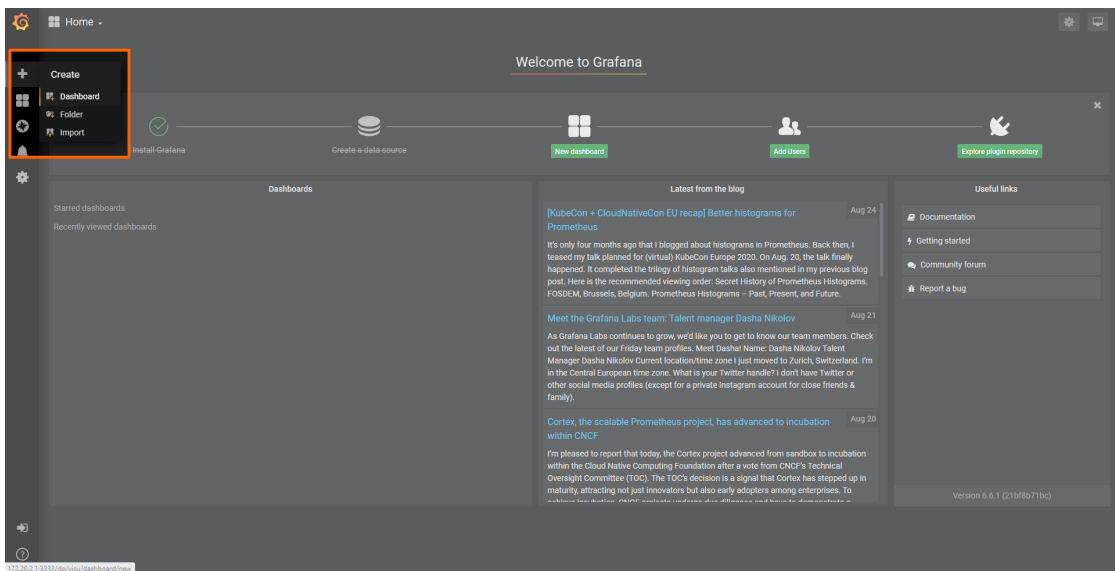
2. Select **Data** in the navigation on the left.

The navigation on the left collapses to show the home screen of Grafana. Here dashboards can be created as described [below](#). Select the burger menu in the top-left to expand the navigation again. Note that the Grafana UI theme is set to light by default.

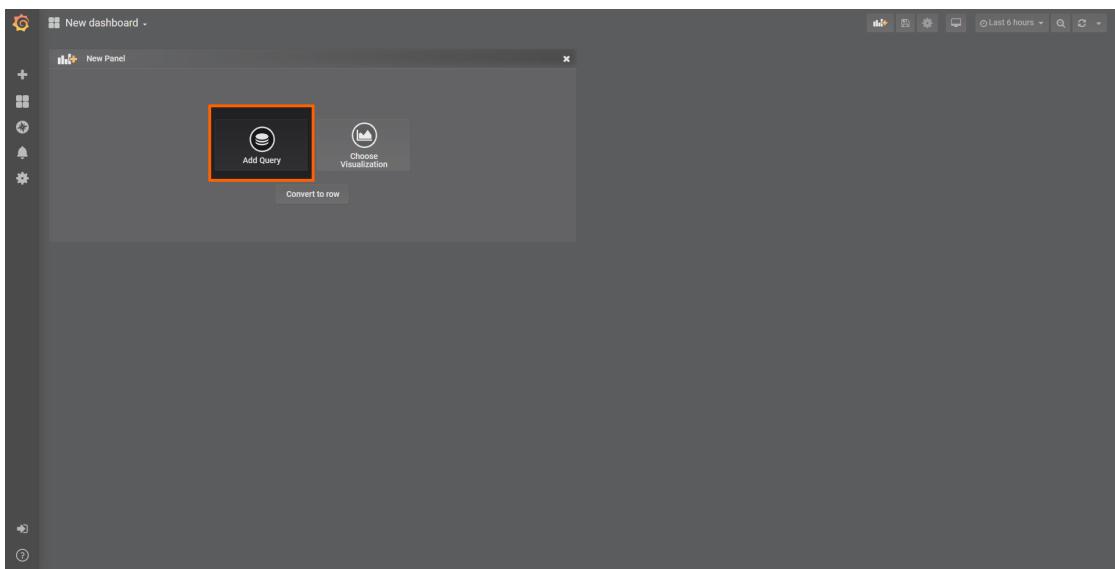
## Creating a dashboard

The instructions below cover the general workflow of creating a dashboard. Refer to [Examples](#) for specific use cases.

1. Access the visualization element on either the node or in the Management System.
2. Select **+ > Dashboard** in the navigation on the left. A box will appear.



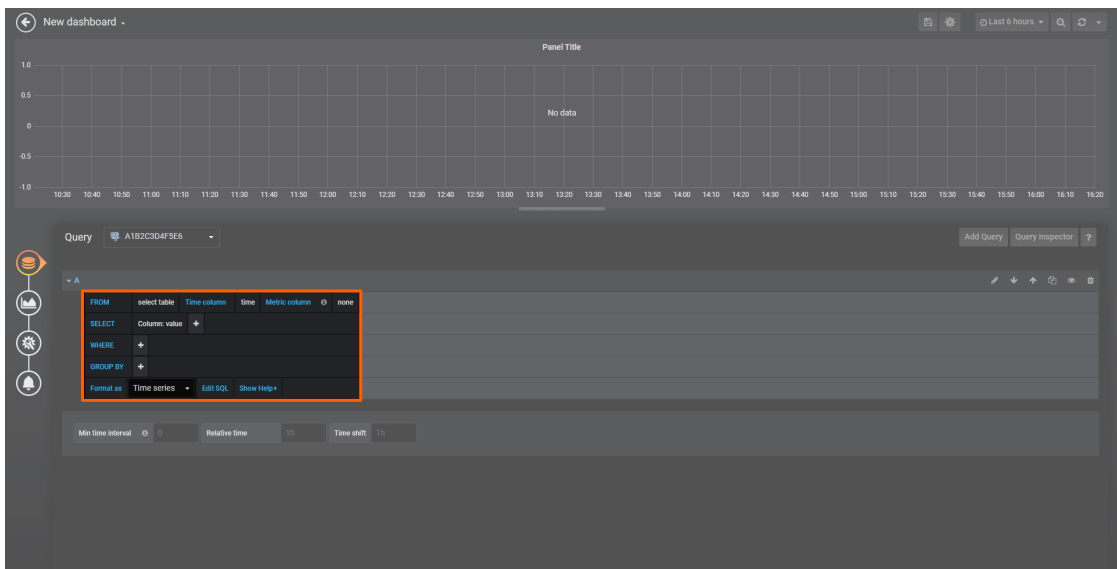
3. Select **Add Query** in the **New Panel** box.



4. Select the data source from the drop-down menu. The name of the data source is the serial number of the node.



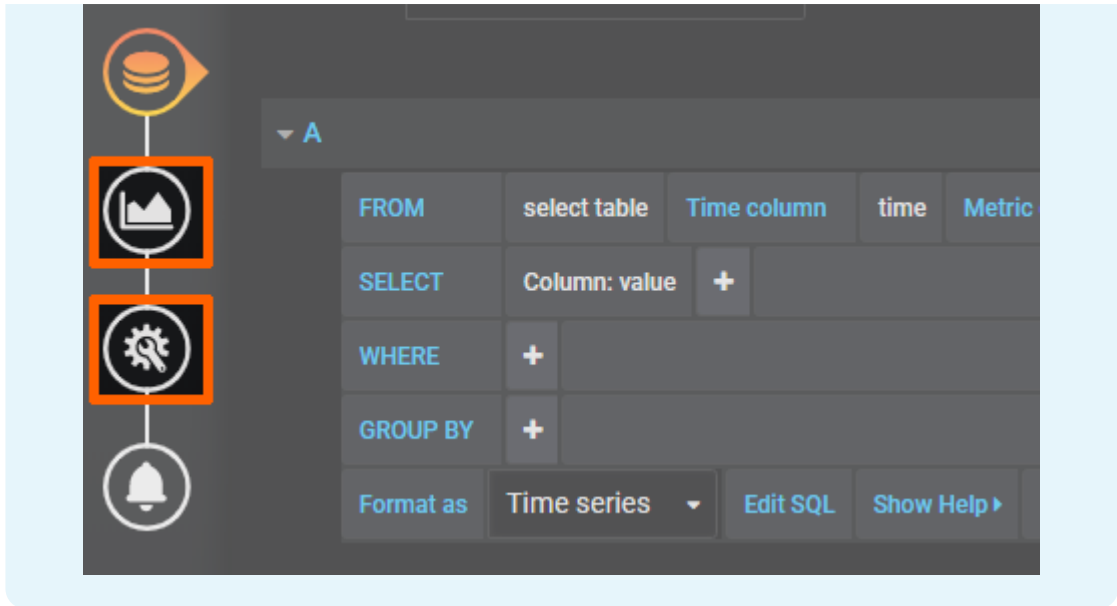
5. Fill in the query information below that appears below. This information depends on the use case and the Gateway configuration.



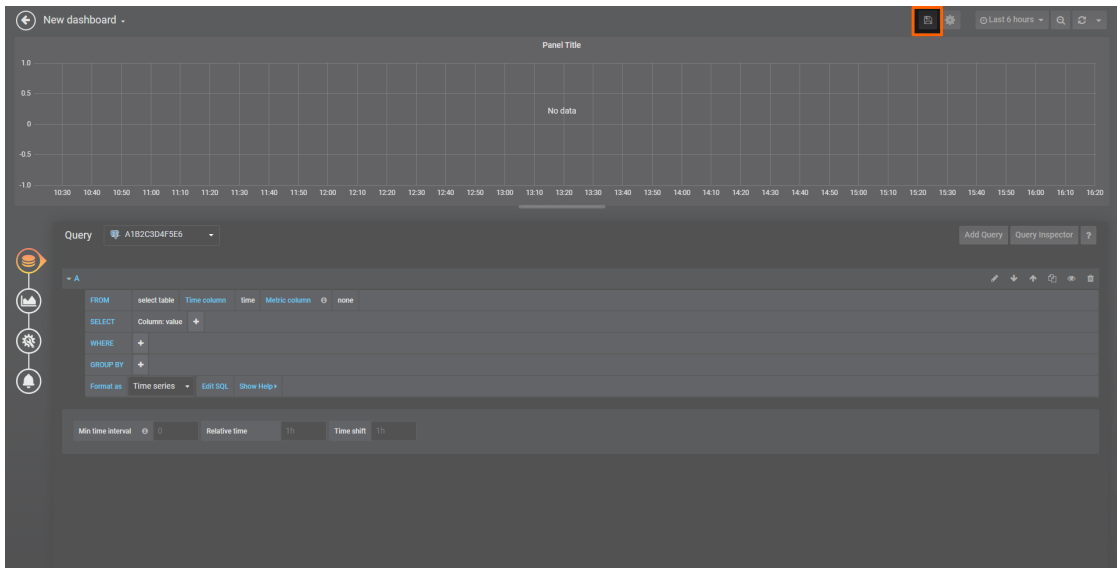
## NOTE

Customize general settings and visualization settings by selecting **General** and **Data** in the navigation on the left. Refer to the [Grafana documentation](#) for more information.

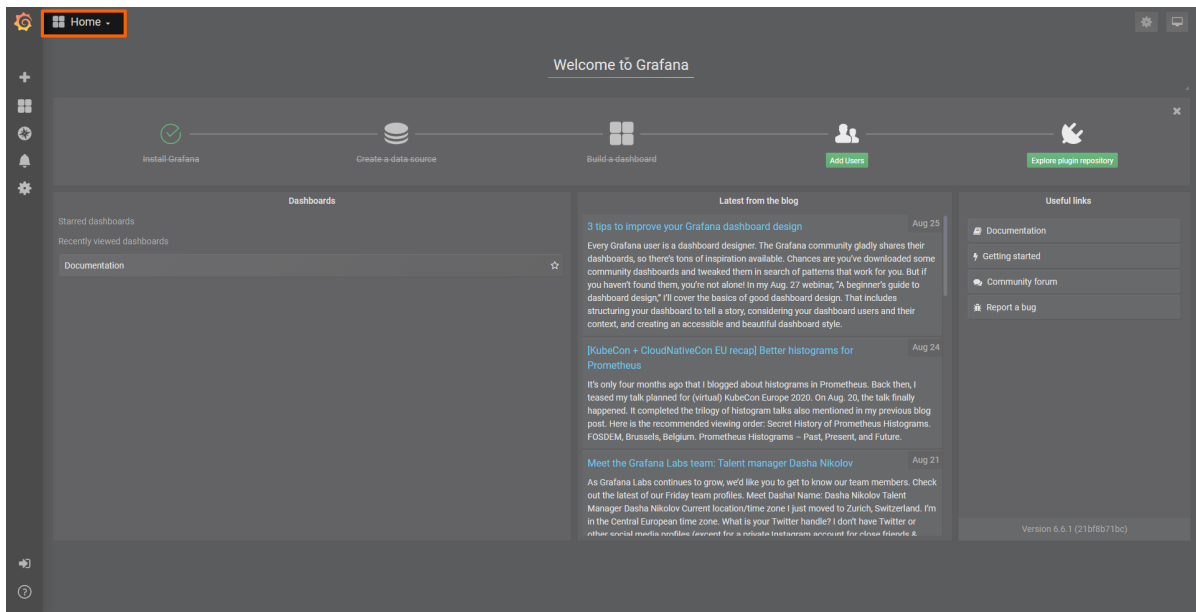




6. Select the save icon in the upper-right corner to save the dashboard.



Access the dashboard from the home menu.



For specific examples on how to use Grafana with Nerve Data Services, refer to [Examples](#).

## Data analytics

The analytics element of the Nerve Data Services is primarily used for processing and analysis of data collected by the Gateway, and provides an easy way to collect and store that data. It consists of a Software Development Kit (SDK) and an Application Programming Interface (API) both written in Python. Analytics are supported on all operating systems based on Debian Buster (tested on Ubuntu 18.04 LTS and Ubuntu 20.04 LTS). The recommended Python version to use when writing analytics apps is 3.7 or later.

The release package contains:

- API documentation
- an installation script
- the installation script README file
- the SDK python wheel (containing the API python wheel)

The Nerve Data SDK can be downloaded from the [Nerve Software Center](#).

## Software Development Kit

### NOTE

The Software Development Kit can only be used on Linux and requires `sudo`, `python` and `pip` for the installation.

The Analytics Data SDK is used to create, build, push and provision analytics apps as Docker images. These functions will be explained in detail in the following sections. The SDK is installed using the installation script. Navigate to the folder containing the installation script and execute the following command:

```
source ./nerve_dp_analytics_install.sh
```

Follow the instructions in the installation script. It will ask for confirmation before installing the following requirements:

- Docker
  - docker-ce
  - curl
  - apt-transport-https
  - ca-certificates
  - gnupg-agent
  - software-properties-common
- Miniconda

It will also add an official Docker GPG key to the system and the official Docker stable repository to apt repositories. The Conda environment `nerve-ds-analytics` will be created and activated upon installation of the Analytics Data SDK and API wheels, as well as all their dependencies.

#### NOTE

Before working with the SDK, make sure that the Conda environment is active. If the Conda environment is active it will be displayed in parentheses in front. Activate the Conda environment by entering the following command:

```
source miniconda/bin/activate <environmentname>
```

The Conda environment automatically deactivates after a restart.

## Create command

The `create` command is used to initialize the environment for analytics app development.

Item	Description
<b>Usage</b>	<code>nerve-analytics create [-h] [-c [CLONE]] [-t {minimal,slim,standard,intel,custom}] app_name path</code>
<b>Positional arguments</b>	<code>app_name</code> The application name. <code>path</code> Path where to create the application.
<b>Optional arguments</b>	<code>-h</code> or <code>--help</code> Use this argument to show the help message in the command line. <code>-c [CLONE]</code> or <code>--clone [CLONE]</code> Clones base Nerve analytics environment into a new environment. If an environment name is not passed with the argument, the new environment name will be auto-generated. <code>-t {minimal,slim,standard,intel,custom}</code> or <code>--type {minimal,slim,standard,intel,custom}</code> Defines the project type to be created.

Using the create command will create a directory named after the app name. Inside, it will create a Dockerfile, a Python script, and add the analytics API wheel. If type is not provided, standard will be used.

## Build command

The build command is used to build the analytics app Docker image.

Item	Description
<b>Usage</b>	<code>nerve-analytics build [-h] [-t TAG] [-p PATH]</code>
<b>Optional arguments</b>	<code>-h</code> or <code>--help</code> Use this argument to show the help message in the command line.  <code>-t TAG</code> or <code>--tag TAG</code> Name of the Docker tag that will be used when tagging the Docker image.  <code>-p PATH</code> or <code>--path PATH</code> Path to the application folder.

If tag is not provided, latest will be used. If path is not provided, the build command will look for a Dockerfile in the current directory.

## Push command

The push command is used to push the previously built Docker image to a Docker registry.

Item	Description
<b>Usage</b>	<code>nerve-analytics push [-h] [-t TAG] [-r RE_TAG] [-n NEW_NAME] [-u USERNAME] [-e EMAIL] [-p PASSWORD] reg_path</code>
<b>Positional arguments</b>	<code>reg_path</code> Path of the Docker registry.

Item	Description
<b>Optional arguments</b>	-h or --help Use this argument to show the help message in the command line.
	-t TAG or --tag TAG Tag name of the Docker image to be pushed.
	-r RE_TAG or --re-tag RE_TAG Define tag name of the Docker image in the registry.
	-n NEW_NAME or --name NEW_NAME New name of the repository.
	-u USERNAME or --username USERNAME Username which is used to log in into registry.
	-e EMAIL or --email EMAIL Email which is used to log in into the registry.

-p PASSWORD or --password PASSWORD  
 Password which is used to log in into the registry.

If tag is not provided, latest will be used. If password is not provided, a prompt to enter it will be displayed to the user.

### Provision command

The provision command is used to upload the previously built Docker image as a Docker workload to a Nerve Management System. The Docker image is also archived as a TAR.GZ file.

Item	Description
	<pre data-bbox="438 315 1348 533">nerve-analytics provision [-h] [-f CFG_FILE] -u URL -vn VERSION_NAME -rn RELEASE_NAME [-usr USERNAME] [-pas PASSWORD] [-n NAME] [-i IMAGE] [-d DEPLOYED_CONTAINER_NAME] [-net NETWORKS] [-desc DESCRIPTION] [-p PORT] [-e ENV] [-m MEM_LIMIT] [-cl CPU_LIMIT] [-r {no,on-failure,always,unless-stopped}] [- rel] [--verbose]</pre> <p data-bbox="438 562 1348 656">Arguments that start with -- (eg. --url) can also be set in a config file (specified via -f &lt;filepath&gt;). The config file syntax accepts the following separated by line breaks:</p> <p data-bbox="220 683 319 712"><b>Usage</b></p> <ul data-bbox="502 683 710 779" style="list-style-type: none"> <li>• key=value</li> <li>• flag=true</li> <li>• stuff={a,b,c}</li> </ul> <p data-bbox="438 801 774 929">Example:  name=test-workload  verbose=true  restart={on-failure}</p> <p data-bbox="438 958 1364 1079">For details on the syntax refer to <a href="https://pypi.org/project/ConfigArgParse/">https://pypi.org/project/ConfigArgParse/</a>. If an argument is specified in more than one place, then the command line values override the config file values which override the defaults.</p>

## Optional arguments

-h or --help

Use this argument to show the help message in the command line.

-f CFG\_FILE or --cfg-file CFG\_FILE

Configuration file for the provision command.

-u URL or --url URL

URL of the Nerve Management System to which the workload will be uploaded. This flag is mandatory.

-vn VERSION\_NAME or --version-name VERSION\_NAME

Version name of the Docker workload. This flag is mandatory.

-rn RELEASE\_NAME or --release-name RELEASE\_NAME

Release name of the Docker workload. This flag is mandatory.

-usr USERNAME or --username USERNAME

Username which is used to log in to the Nerve Management System.

-pas PASSWORD or --password PASSWORD

Password which is used to log in to the Nerve Management System.

-n NAME or --name NAME

Name of the Docker workload which will be uploaded to the Nerve Management System.

-i IMAGE or --image IMAGE

This is the of the Docker image that has been created with the create and build commands and will be provisioned to the Nerve Management System.

-d DEPLOYED\_CONTAINER\_NAME or --deployed-container-name DEPLOYED\_CONTAINER\_NAME

Name of the Docker container upon deployment to a node.

-net NETWORKS or --networks NETWORKS

Docker network names. This is set to nerve-ds by default if not specified.

-desc DESCRIPTION or --description DESCRIPTION

Description for the Docker workload to be uploaded.

-p PORT or --port PORT

Port mapping for the container to be deployed, for example 5432:5432 or 123:234/udp.

-e ENV or --env ENV

Environment variables for the container to be deployed, for example VAR1=myval.

-m MEM\_LIMIT or --mem-limit MEM\_LIMIT

Memory limit in MB for the workload on the node.

-cl CPU\_LIMIT or --cpu-limit CPU\_LIMIT

Max number of CPUs to be used by the workload on the node. 1 by default.

-r {no,on-failure,always,unless-stopped} or --restart {no,on-failure,always,unless-stopped}

Restart policy for the container to be deployed on the node.

-rel or --released

This sets the workload version to released. Released versions cannot be updated.

The user that is used to provision the analytics app must already exist in the Management System. All parameters used when uploading a workload from the Management System can also be specified in the provision command.

If password is not provided, a prompt to enter the password will be displayed. If a workload with the same name already exists in the Management System, it must be a Docker workload in order to add a new version to it. Otherwise, the provisioning cannot be done. When adding a new version to an existing Docker workload in the Management System, version must be different than that of the existing version.

## Application Programming Interface

The Analytics API is used for writing Python applications that collect and store data from and to other elements of the Data Services. The recommended Python version to use when writing analytics apps is 3.7 or later. The Analytics API provides the following modules:

Module	Description
<b>Batch Input TimescaleDB</b>	Used to fetch data from a TimescaleDB database.
<b>Batch Output TimescaleDB</b>	Used to store data in a TimescaleDB database.
	ZeroMQ Subscriber used to collect data from a ZeroMQ Publisher. Can work in both asynchronous and synchronous modes.
<b>Stream Input ZeroMQ</b>	<b>NOTE</b> In order to connect the ZeroMQ Publisher output of the Gateway and the Stream Input ZeroMQ of the analytics, the IP address 172.20.10.1 must be used. This is the case only if the analytics container is running inside the Docker network nerve-ds. If the Docker network is host, localhost can be used.
<b>Stream Output MQTT</b>	MQTT Publisher used to send data to a MQTT broker.

More in depth documentation regarding the API modules can be found in the release package of the analytics element that has been downloaded from the Nerve Software Center.

## Examples

### Gateway configuration examples

This chapter covers use cases for the supported protocols of the Nerve Data Services, showcasing the usage of each Data Services element in the process. This section is subject to change and further examples will be added in the near future. Currently available examples are:

- [OPC UA Server to cloud](#)
- [OPC UA Server security](#)
- [S7 Client to cloud](#)
- [MQTT Publisher to OPC UA Server at the node](#)



- [Receiving data via MQTT for Analytics and Visualization](#)
- [Custom JSON format example](#)
- [Sending data to MS Azure IoT Hub](#)
- [Modbus server data to InfluxDB for visualization](#)
- [NerveDB with data buffering](#)

## OPC UA Server to cloud for visualization

In this example, a sensor is providing temperature and humidity data as an OPC UA Server. Temperature data is displayed at the node while humidity data is presented at the Management System. Data is visualized with the visualization element of the Data Services.

The instructions below cover the following steps:

- Provisioning an OPC UA Server demo sensor as a Docker workload
- Deploying the provisioned Docker workload to the target node
- Configuring the Data Services Gateway
- Local data visualization at the node
- Central data visualization in the Management System

### Provisioning and deploying an OPC UA Server at the node

First, the temperature sensor simulation OPC UA Server must be deployed to the node as a Docker workload. Download the **Data Services OPC UA demo sensor** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload by following [Provisioning a Docker workload](#). This example uses **TTTech OPC UA Server - demoSensor** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>DOCKER SPECIFIC INFO</b>	Select <b>New port</b> and enter the following settings: <ul style="list-style-type: none"> <li>◦ <b>Protocol:</b> TCP</li> <li>◦ <b>Host Port:</b> 4848</li> <li>◦ <b>Container Port:</b> 4848</li> </ul>
<b>Container name</b>	tttech-opcua-server-demosensor
<b>Network name</b>	bridge

3. Deploy the provisioned Docker workload by following [Deploying a workload](#).

## Configuring the Data Services Gateway

The configuration defines the OPC UA Server demo sensor that was deployed above as a data input. The local NerveDB storage and the cloud NerveDB storage are defined as data outputs. An OPC UA Client connects to the OPC UA Server demo sensor and periodically reads the value of temperature and humidity. Temperature values are forwarded to the storage on the node and humidity values are sent to the Management System.

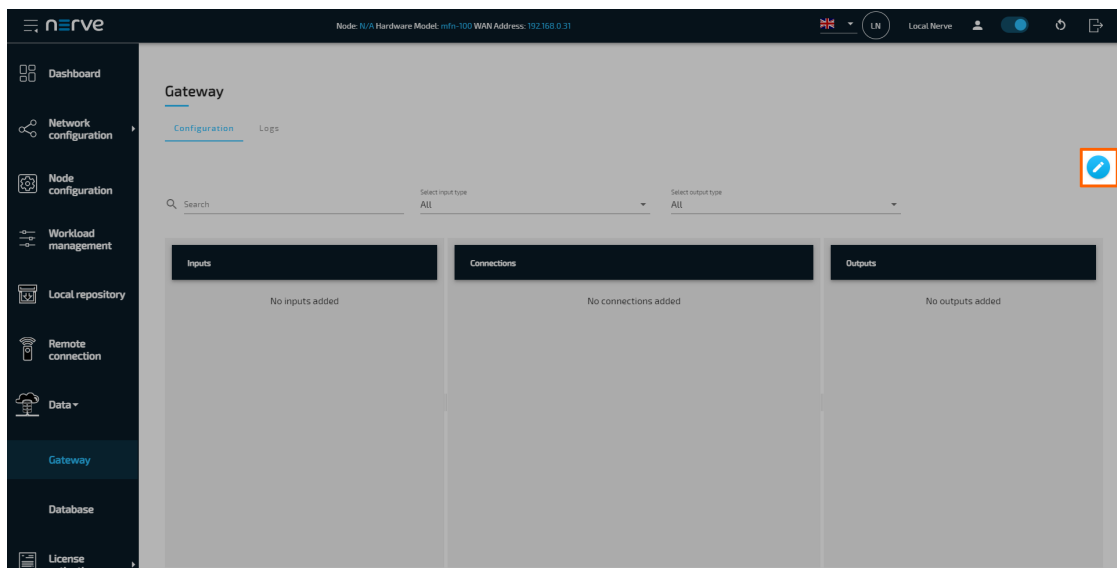
The configuration can be applied with the graphical configuration tool or by importing a pre-written JSON configuration file. In this example, a pre-written JSON configuration is used to configure the Gateway for the sake of simplicity.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Kontron KBox A-150-APL</b>	LAN 1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide. <wanip>:3333
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide. <wanip>:3333
<b>Maxtang AXWL10</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>

Nerve Device	Physical port	Local UI
<b>Supermicro SuperServer 5029C-T</b>	LAN1	<wanip>:3333  To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>  <wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

```
{
  "inputs": [
    {
      "name": "client_demo_sensor",
      "type": "OPC_UA_CLIENT",
      "serverUrl": "opc.tcp://localhost:4848",
      "pollingInterval_ms": 1000,
      "connectors": [
        {
          "name": "poll",
          "accessType": "polling",

```

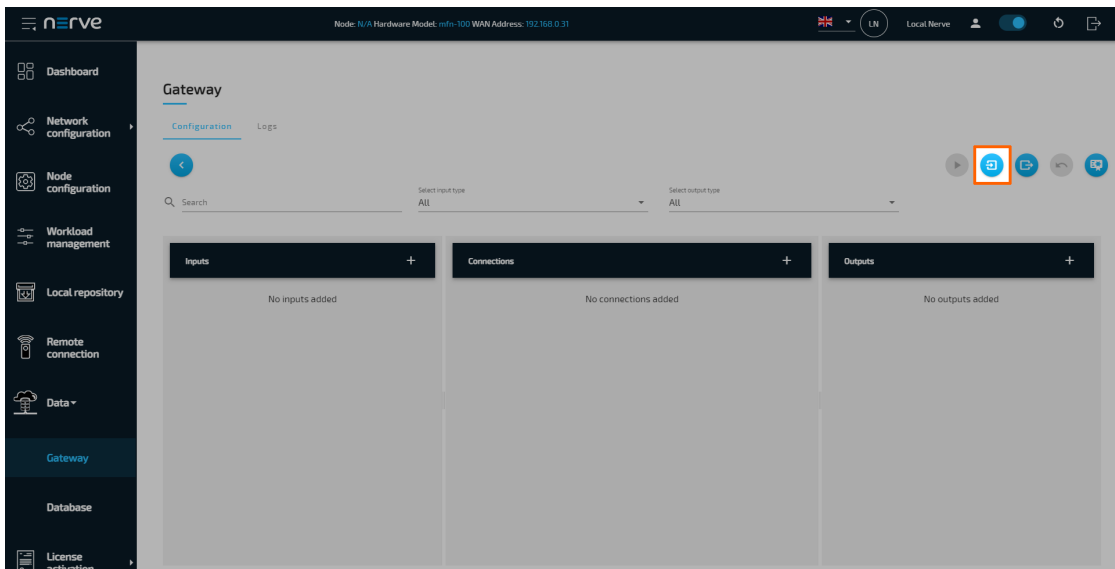
```

        "nodes": [
            "ns=2;i=2"
        ]
    },
    {
        "name": "subs",
        "accessType": "subscription",
        "publishingIntervalAtServer_ms": 1000,
        "samplingIntervalAtServer_ms": 1000,
        "nodes": [
            "ns=2;i=4"
        ]
    }
]
},
],
"outputs": [
    {
        "type": "NERVE_DB",
        "location": "LOCAL",
        "connectors": [
            {
                "name": "default connector",
                "tableName": "default"
            }
        ]
    },
    {
        "type": "NERVE_DB",
        "location": "CENTRAL",
        "connectors": [
            {
                "tableName": "ms_mqtt_broker_to_cloud_timescale_db"
            }
        ]
    }
],
"connections": [
    {
        "name": "demo_sensor_to_local_db",
        "input": {
            "index": 0,
            "connector": 0
        },
        "output": {
            "index": 0,
            "connector": 0
        }
    },
    {
        "input": {
            "index": 0,
            "connector": 1
        },
        "name": "demo_sensor_to_mgmsys",
        "output": {
            "index": 1,
            "connector": 0
        }
    }
]

```

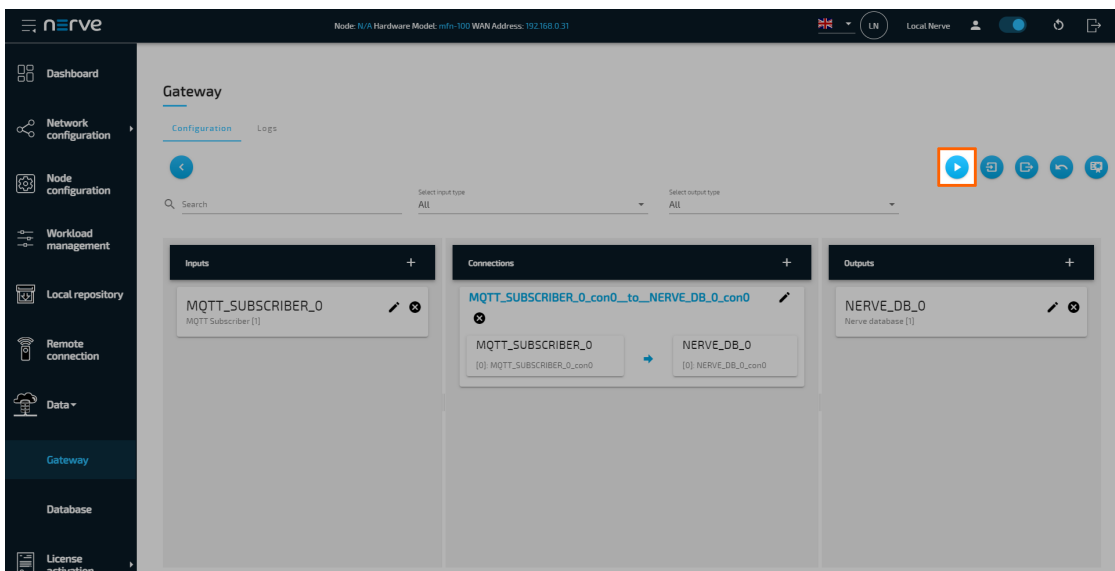
```
}  
]  
}
```

6. Select the **Import** button.



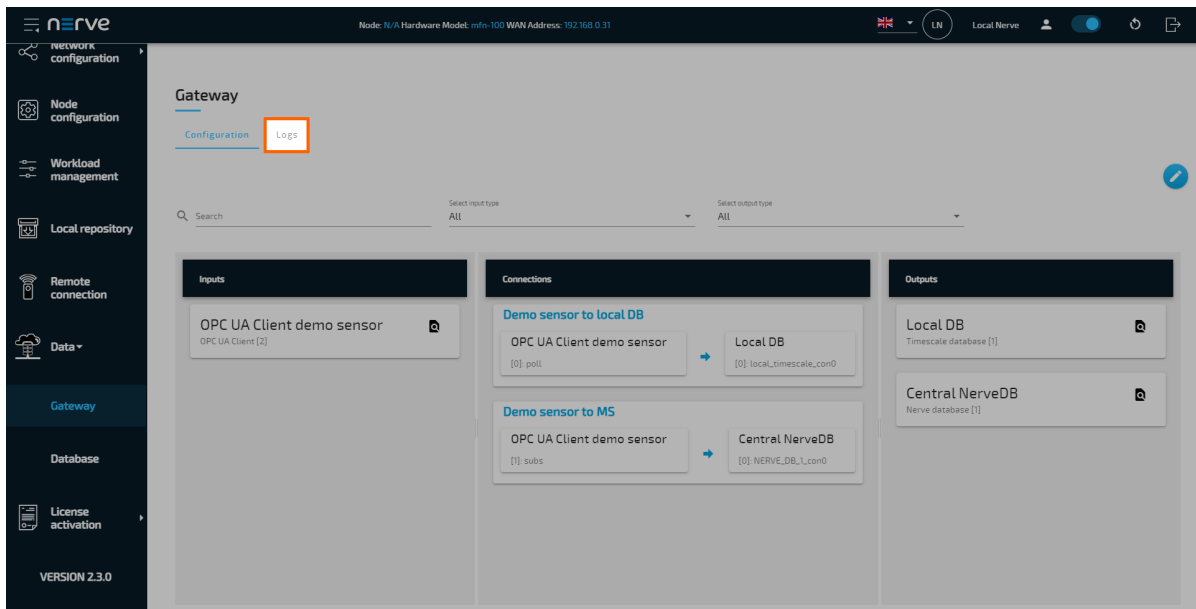
7. Add the JSON configuration file containing the code above from the file browser.

8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Local data visualization at the node

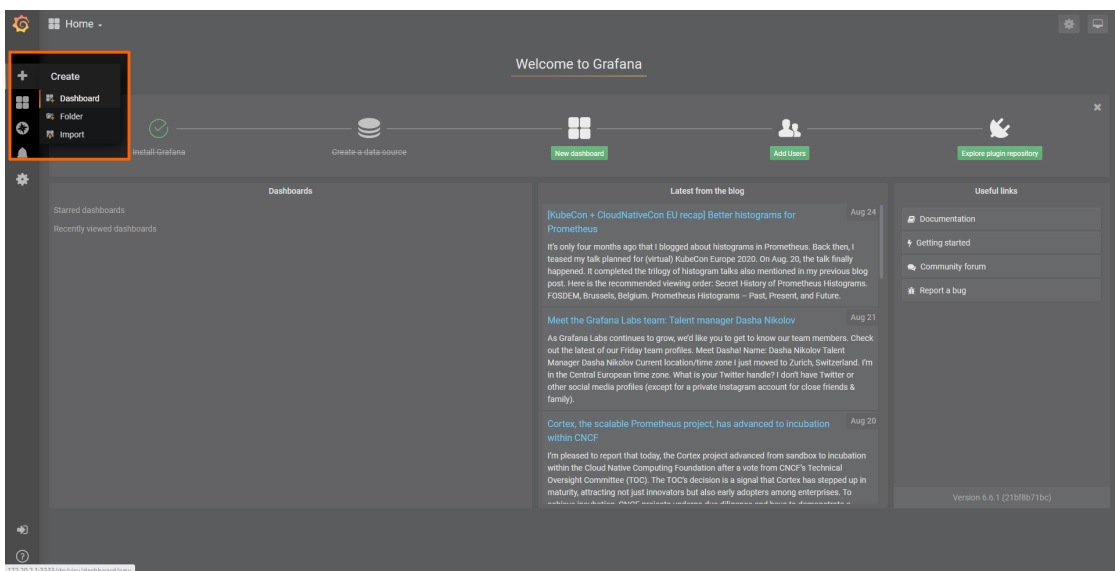
To visualize temperature data at the node, open the local data visualization element through the Data Services UI on the node.

1. Select **Data** in the navigation on the left. The Grafana UI will open.

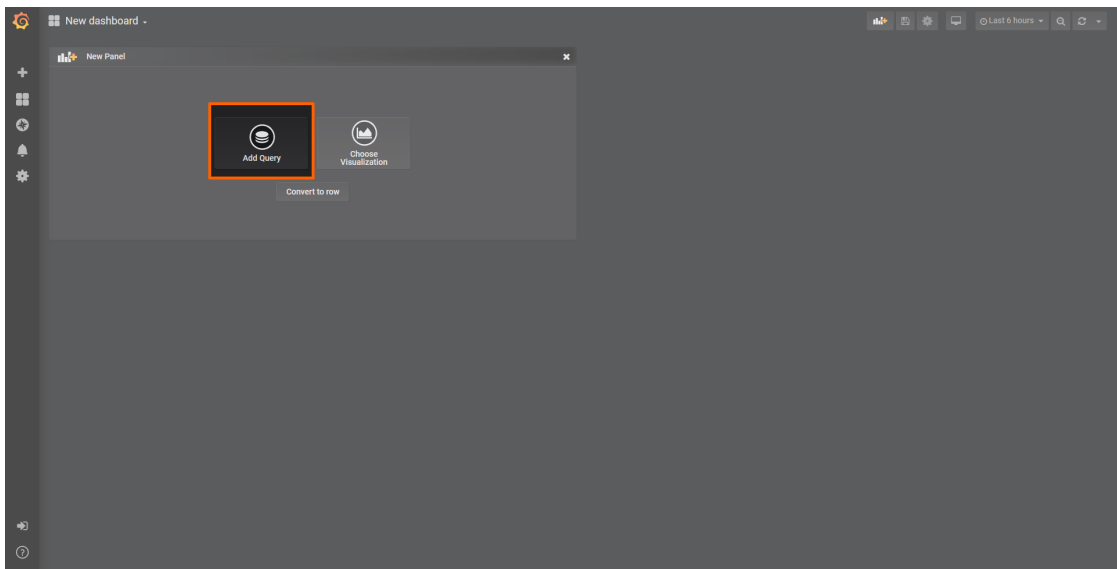
### NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

2. Select **+ > Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.



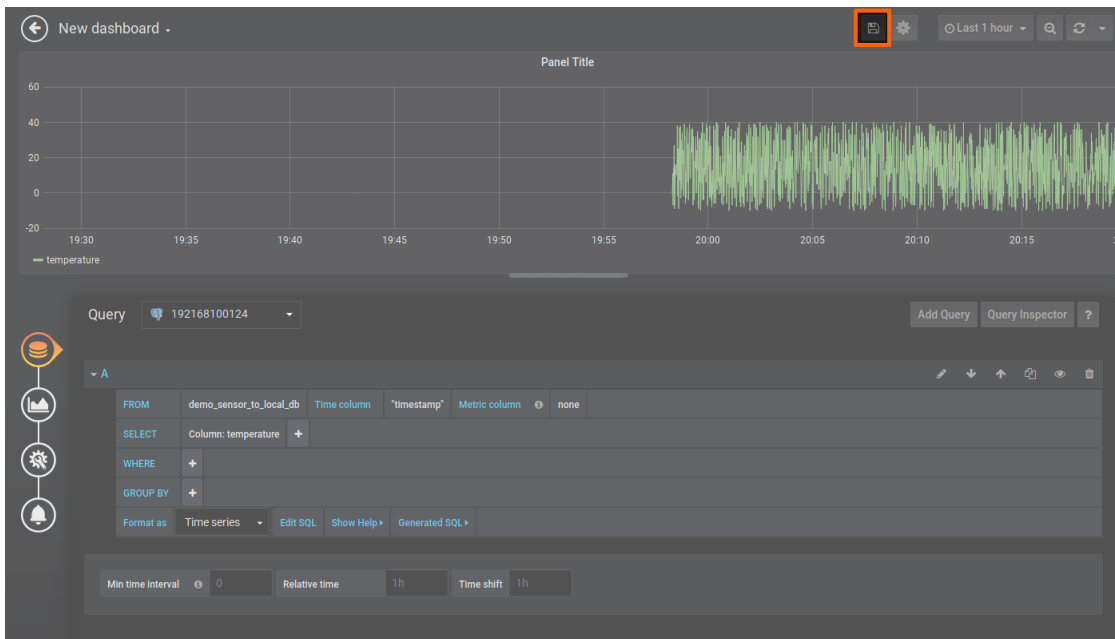
4. Select the data source from the drop-down menu. The name of the data source is the serial number of the node.



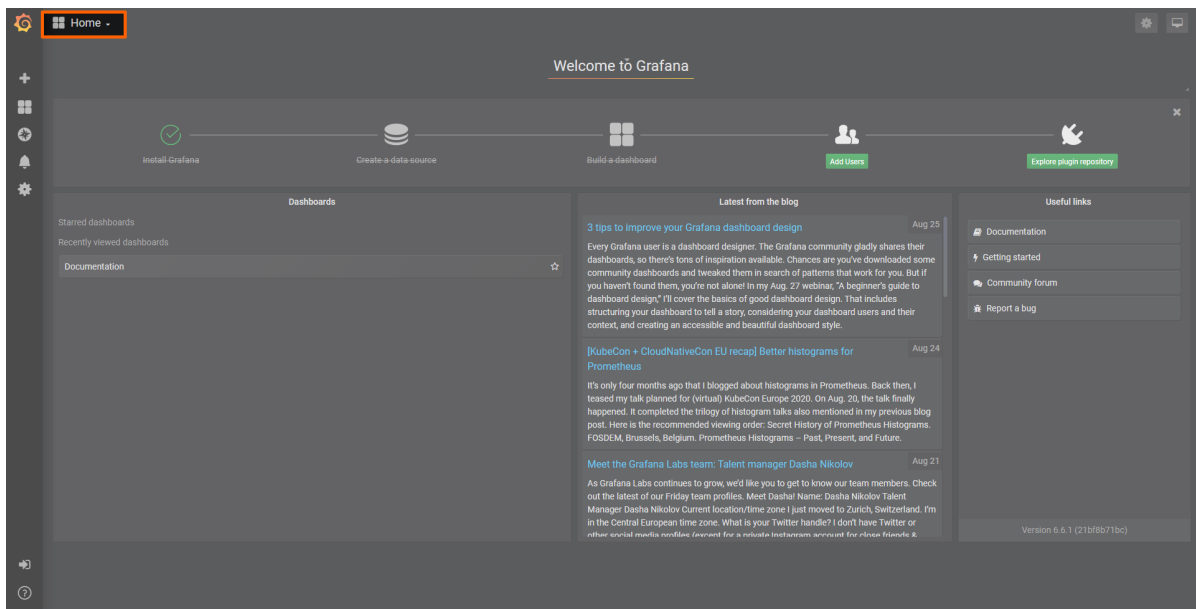
5. Fill in the following query information:

Setting	Value
<b>FROM</b>	demo_sensor_to_local_db
<b>SELECT</b>	Column: temperature
<b>Format as</b>	Time series

6. Select the save icon in the upper-right corner to save the dashboard.



The dashboard can be accessed from the Grafana home menu.



## Central data visualization in the Management System

To visualize humidity data in the Management System, open the central visualization element through the Data Services UI in the Management System.

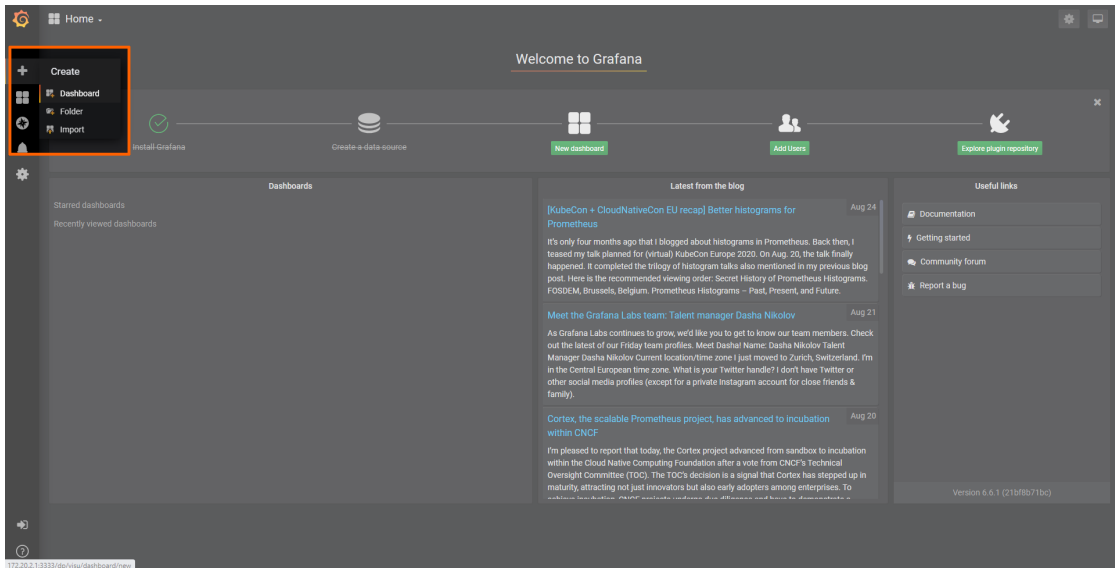
1. Select **Data** in the navigation on the left. The Grafana UI will open.

### NOTE

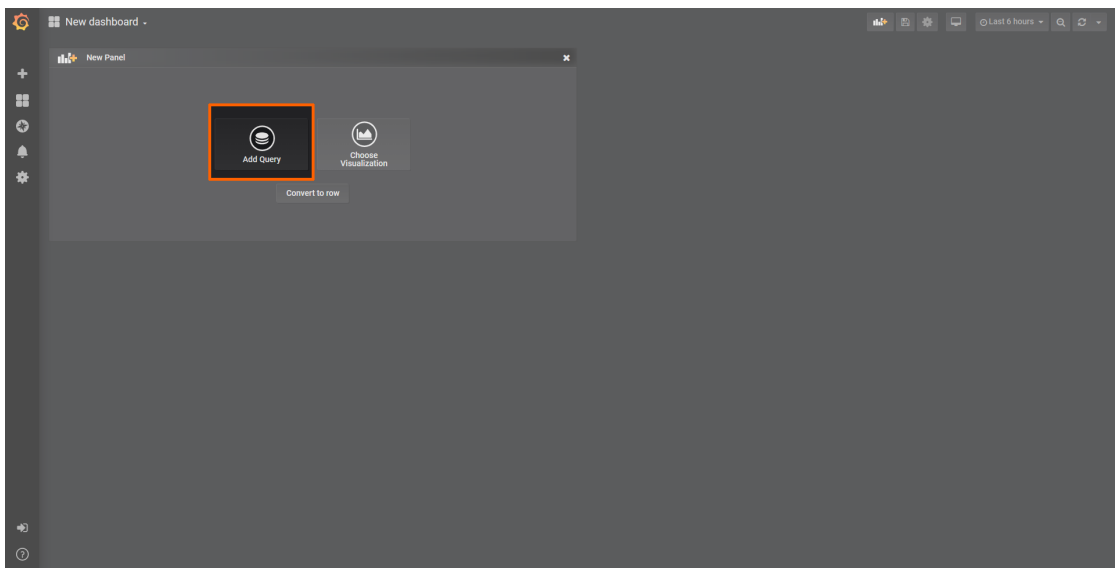
Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.



2. Select **+ > Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.



4. Select the data source from the drop-down menu. The name of the data source is the name and serial number of the node, formatted as `<nodename>` (`<serialnumber>`).



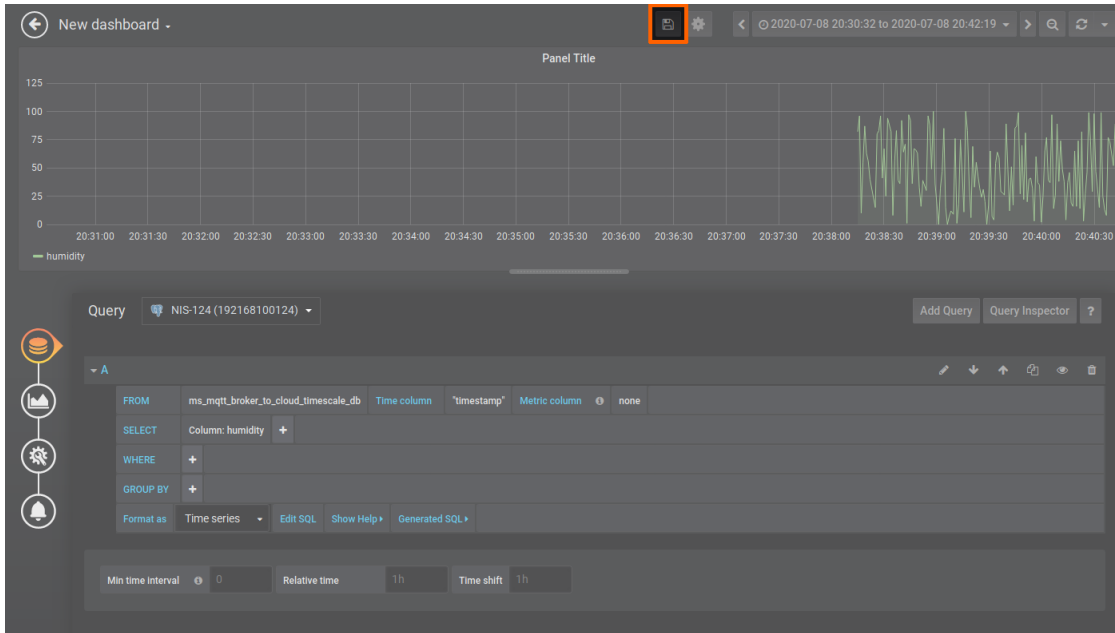
## NOTE

Note that multiple data sources can be selected in the Grafana instance of the Management System, depending on the number of nodes that are registered. Make sure to remember the serial number of the node that was used for workload deployment before.

5. Fill in the following query information:

Setting	Value
<b>FROM</b>	ms_mqtt_broker_to_cloud_timescale_db
<b>SELECT</b>	Column: humidity
<b>Format as</b>	Time series

6. Select the save icon in the upper-right corner to save the dashboard.



## OPC UA Server security

The OPC UA Server security example demonstrates an OPC UA Server configuration with:

- all available security end-points configured (security mode and security policy), including unsecured end-points
- two users added (logins)
- certificates added for security connection
- application URI set
- the certificate of an external OPC UA client added to the trust list.

To try this example, an OPC UA Server and an OPC UA client are needed. This example uses UaExpert. The OPC UA Server temperature sensor simulation is deployed in form of a Docker workload next.

## Provisioning and deploying an OPC UA Server at the node

First, the temperature sensor simulation OPC UA Server must be deployed to the node as a Docker workload. Download the **Data Services OPC UA demo sensor** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload by following [Provisioning a Docker workload](#). This example uses **TTTech OPC UA Server - demoSensor** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.

Setting	Value
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>DOCKER SPECIFIC INFO</b>	Select <b>New port</b> and enter the following settings: <ul style="list-style-type: none"> <li>◦ <b>Protocol:</b> TCP</li> <li>◦ <b>Host Port:</b> 4848</li> <li>◦ <b>Container Port:</b> 4848</li> </ul>
<b>Container name</b>	tttech-opcua-server-demosensor
<b>Network name</b>	bridge

3. Deploy the provisioned Docker workload by following [Deploying a workload](#).

## Obtaining certificates

Once the OPC UA demo sensor is deployed, the certificate files have to be obtained and imported to the node. In this example, a script to generate a self-signed certificate key pair is used. The script can be downloaded under **Example Applications** from the [Nerve Software Center](#). It uses the OpenSSL library in order to make the generation of the certificate more convenient. If preferred, other tools or certificates can be used. Consult the IT administrator before continuing.

The script can be used the following way. Note that Python version 3.7.0 or higher is required to run the script:

Item	Description
<b>Usage</b>	<code>create_ss_certificate.py [-h] [-k KeySize] [-a NetworkAddresses] [-d Duration] ApplicationURI CertificateName [OutputPath]</code>
<b>Arguments</b>	<ul style="list-style-type: none"> <li>-k Key length, defaults to 2048</li> <li>-a Network addresses in comma separated list, defaults to empty list</li> <li>-d Number of days to certify the certificate for, defaults to 30</li> <li>ApplicationURI Enter the Application URI, mandatory argument</li> <li>CertificateName Enter the name of the generated files, mandatory argument.</li> <li>OutputPath Destination where certificate and private key files are placed after generation, defaults to current working directory</li> </ul>

Enter the following command to generate a self-signed certificate and private key pair:

```
python create_ss_certificate.py -a 172.20.2.1 urn:gateway.server server <outputpath>
```

## NOTE

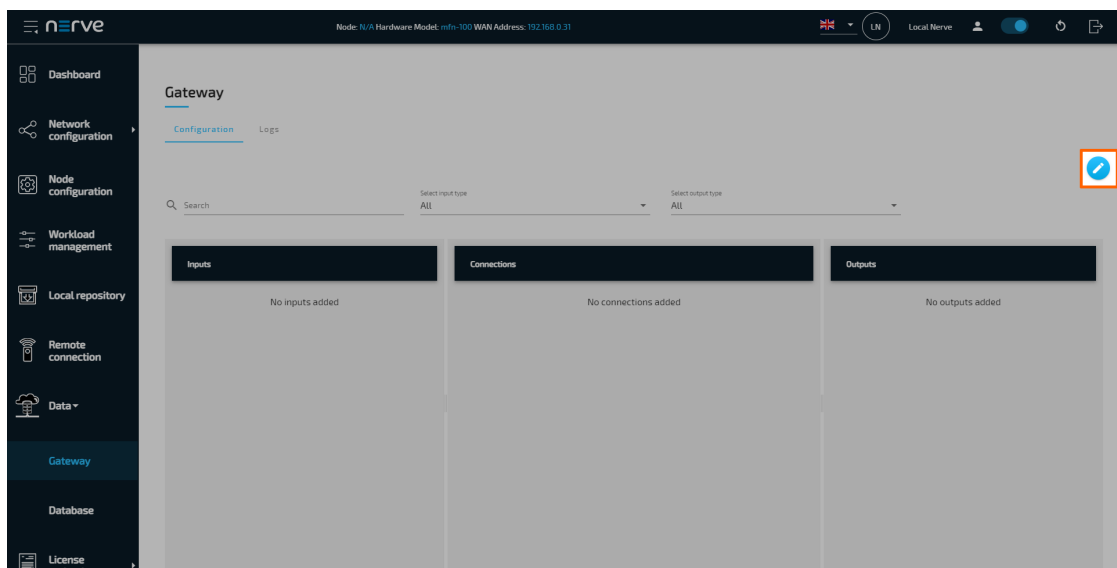
- Note that a hostname can be used when creating a certificate. However, an entry in `/etc/hosts` must be created on the Linux machine so that the hostname can be resolved. The OPC UA Server output of the Gateway must be accessed with the hostname that was given when creating the self-signed certificate.
- Note that `172.20.2.1` applies to Nerve Devices that have a physical mgmt port like the MFN 100. If the used device does not have a mgmt port, refer to the [device guide](#) page of the device to find out which IP address to use.

To obtain the certificate from UaExpert, start UaExpert and go to **Settings > Manage certificates**. There a list of current certificates and their location is displayed. The standard location of the `uaexpert.der` certificate is `.../unifiedautomation/uaexpert/PKI/own/certs/uaexpert.der`.

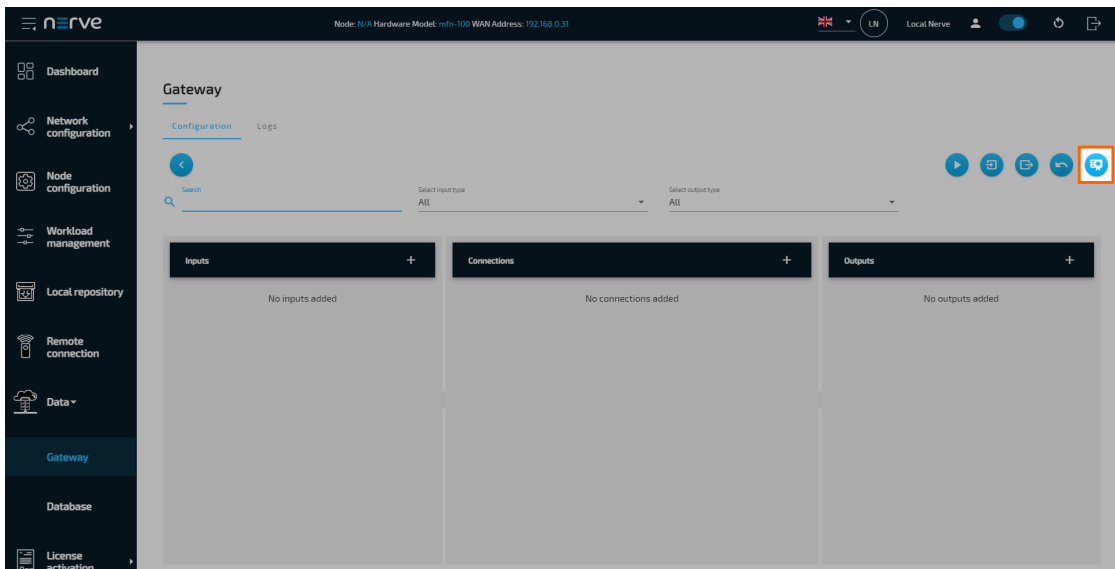
## Importing certificates

Importing certificates is done using the graphical configuration tool on the node. The instructions below show the import of the UaExpert certificate as well as the self-signed certificate and key. Note that the certificates have to be in the DER format. If other certificates are used, convert certificates first before importing. OpenSSL can be used for converting certificates. Also, note that this example uses the UaExpert client. Any OPC UA Client can be used. Import the corresponding certificate of the used OPC UA Client in DER format instead.

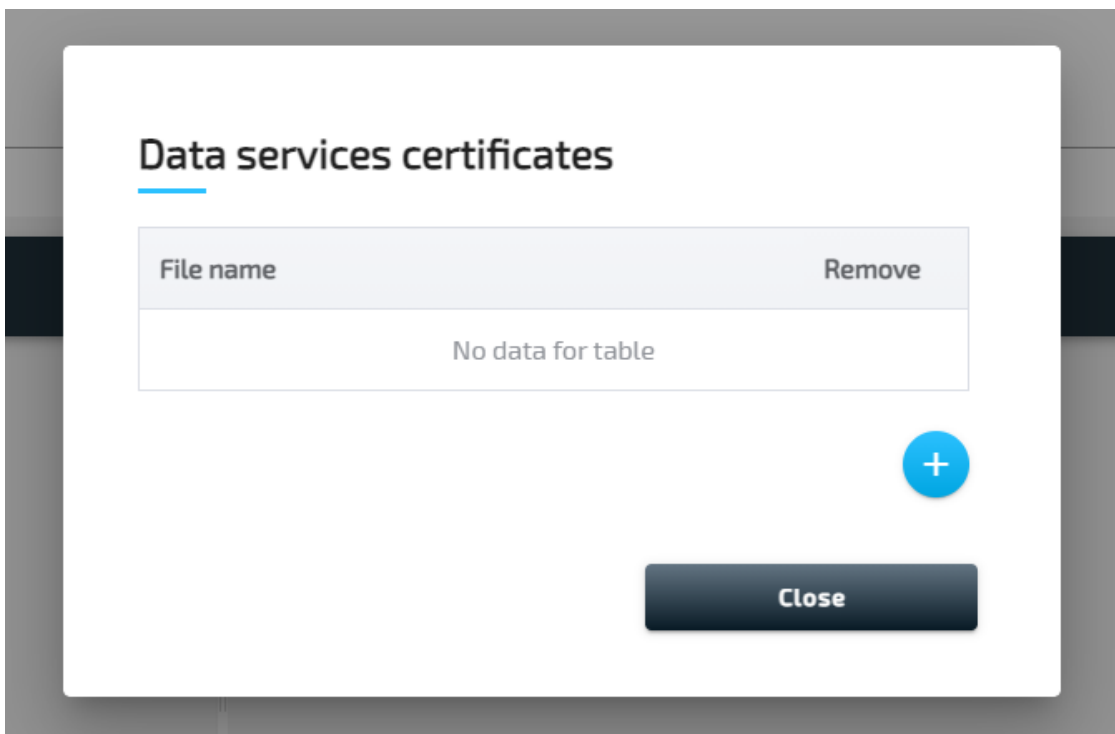
1. Log in to the Local UI.
2. Select the arrow next to **Data** to expand the sub menu.
3. Select **Gateway**.
4. Select the **Edit configuration** symbol on the right to enter editing mode.



5. Select the **Import certificates** symbol.

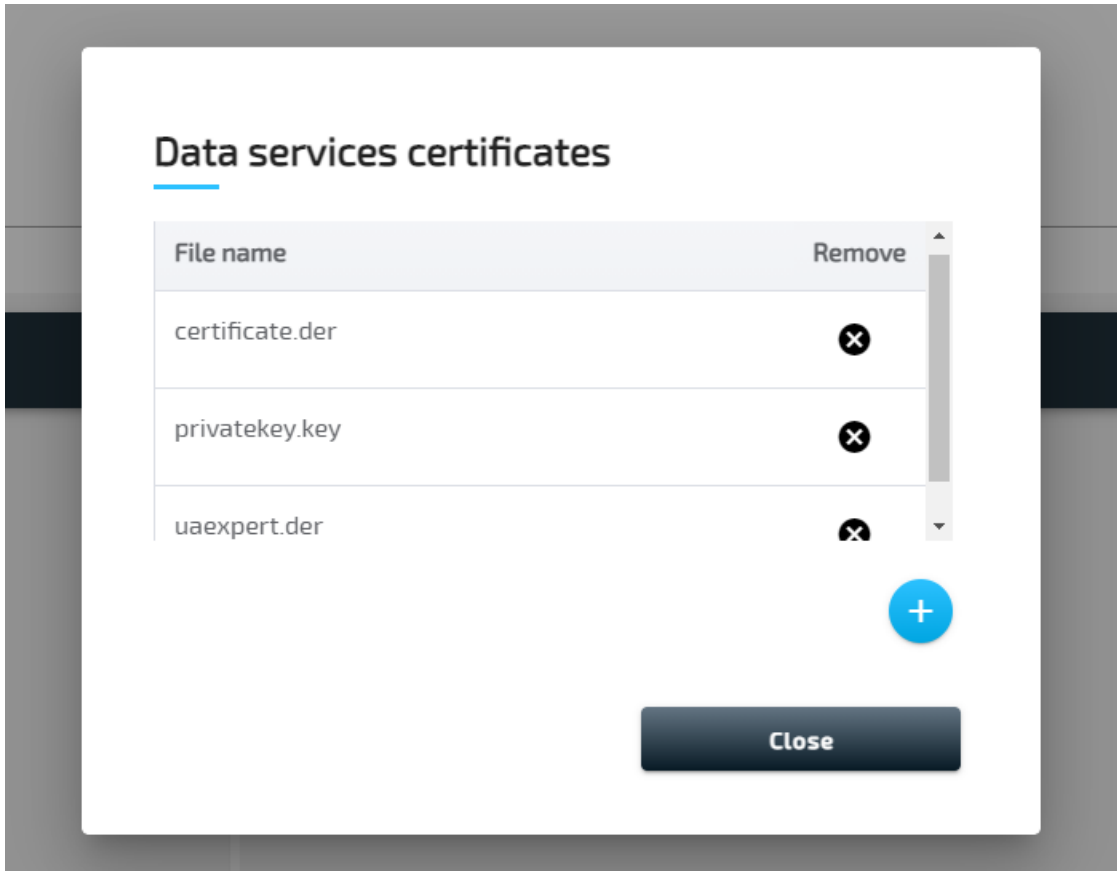


6. Select the plus symbol in the overlay that appeared.



7. Add the UaExpert certificate as well as the self-signed certificate and key.

8. Select **Close**



## Configuring the Data Services Gateway

Through the JSON configuration below, the OPC UA Client connects to the OPC UA Server demo sensor and periodically reads the values of temperature and humidity. That data is published as an OPC UA Server output. In the OPC UA Server output configuration example, the following security settings are used:

- Two username/password entities are added for logging in to the server.
- `securityMode` and `securityPolicy` settings are both set to `all` to create security endpoints for all available combinations.
- Certificate and private key file names are set according to the files that were imported to the node earlier.
- `applicationUri` is set in the description section. It must comply with the application URI set in the certificate.
- The UaExpert client certificate is added to the trust list.

Follow the instructions below to apply the Gateway configuration.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>

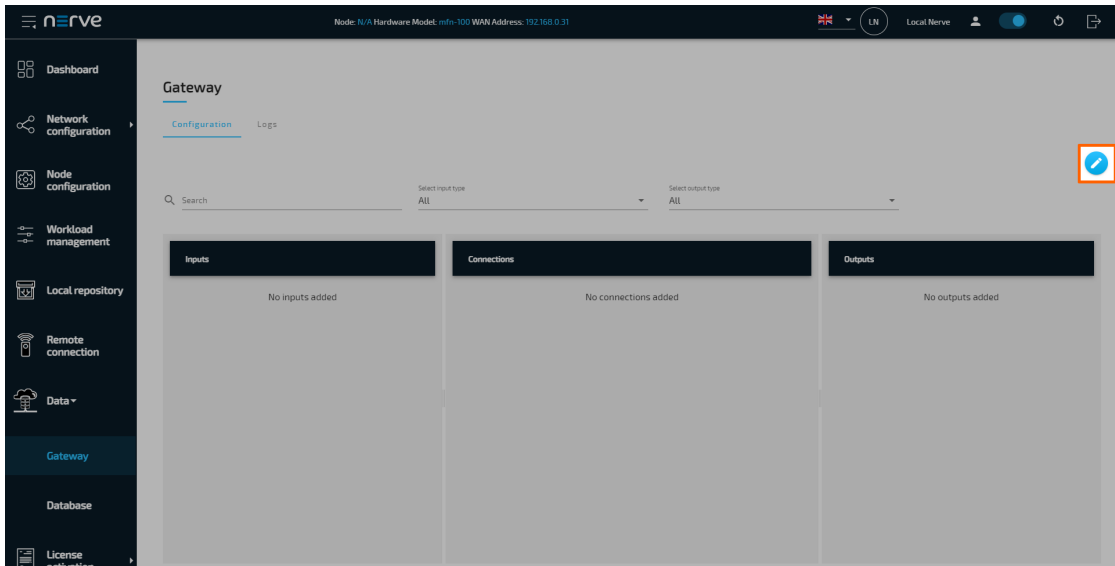
Nerve Device	Physical port	Local UI
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.
<b>Kontron KBox A-250</b>	ETH 2	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.
<b>Maxtang AXWL10</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer 5029C-T</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Winmate EACIL20</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.



3. Select **Gateway**.

4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

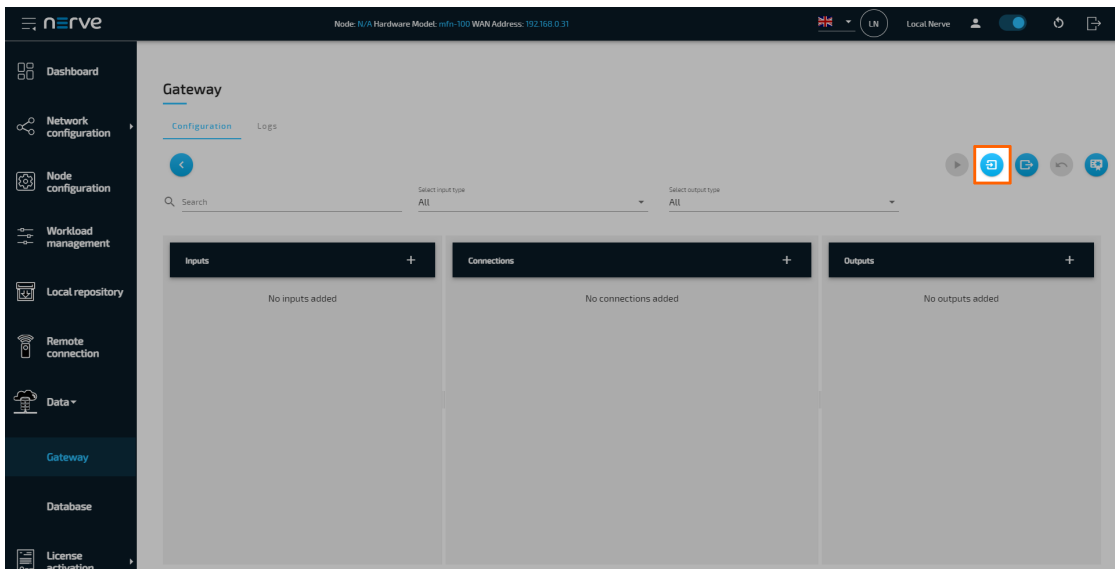
```
{
  "inputs": [
    {
      "name": "client_demo_sensor",
      "type": "OPC_UA_CLIENT",
      "serverUrl": "opc.tcp://localhost:4848",
      "pollingInterval_ms": 1000,
      "connectors": [
        {
          "name": "poll",
          "accessType": "polling",
          "nodes": [
            "ns=2;i=2",
            "ns=2;i=4"
          ]
        }
      ]
    }
  ],
  "outputs": [
    {
      "name": "output_opc_ua_server",
      "type": "OPC_UA_SERVER",
      "port": 4840,
      "logins": [
        {
          "username": "usr1",
          "password": "pwd1"
        },
        {
          "username": "usr2",
          "password": "pwd2"
        }
      ]
    }
  ],
}
```

```

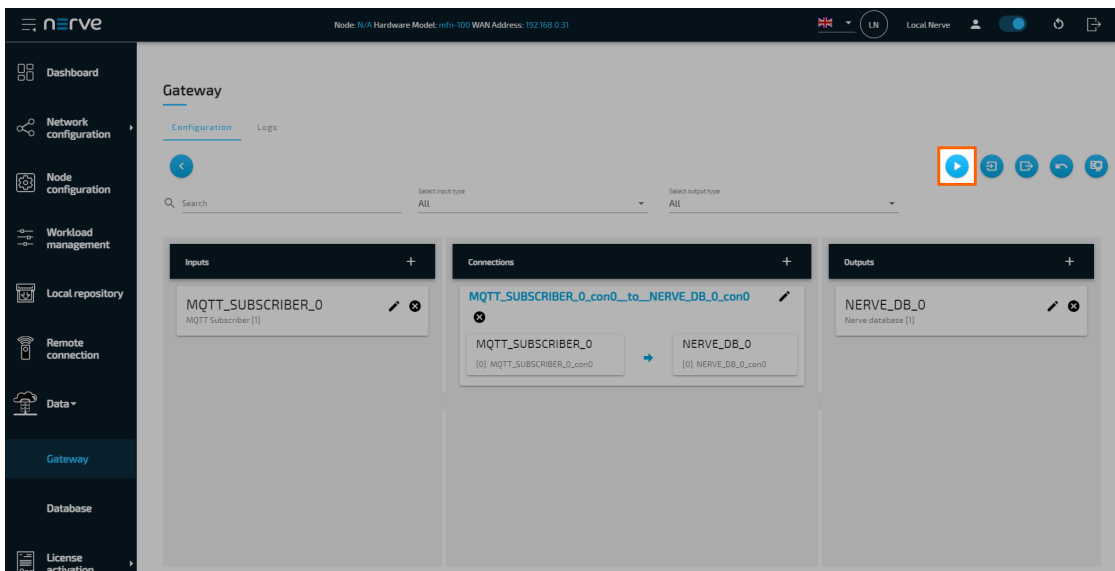
"securities": [
  {
    "securityMode": "all",
    "securityPolicy": "all"
  }
],
"certificate": {
  "certFilePath": "server_cert.der",
  "keyFilePath": "server_key.der"
},
"description": {
  "applicationUri": "urn:gateway.server"
},
"trustList": [
  "uaexpert.der"
],
"connectors": [
  {
    "name": "OutputFromDemoSensor",
    "browseName": "OutputFromDemoSensor",
    "identifier": "s=OutputFromDemoSensor"
  }
]
},
"connections": [
  {
    "name": "opcua_client_to_opcua_server",
    "input": {
      "index": 0,
      "connector": 0
    },
    "output": {
      "index": 0,
      "connector": 0
    }
  }
]
}

```

6. Select the **Import** button.

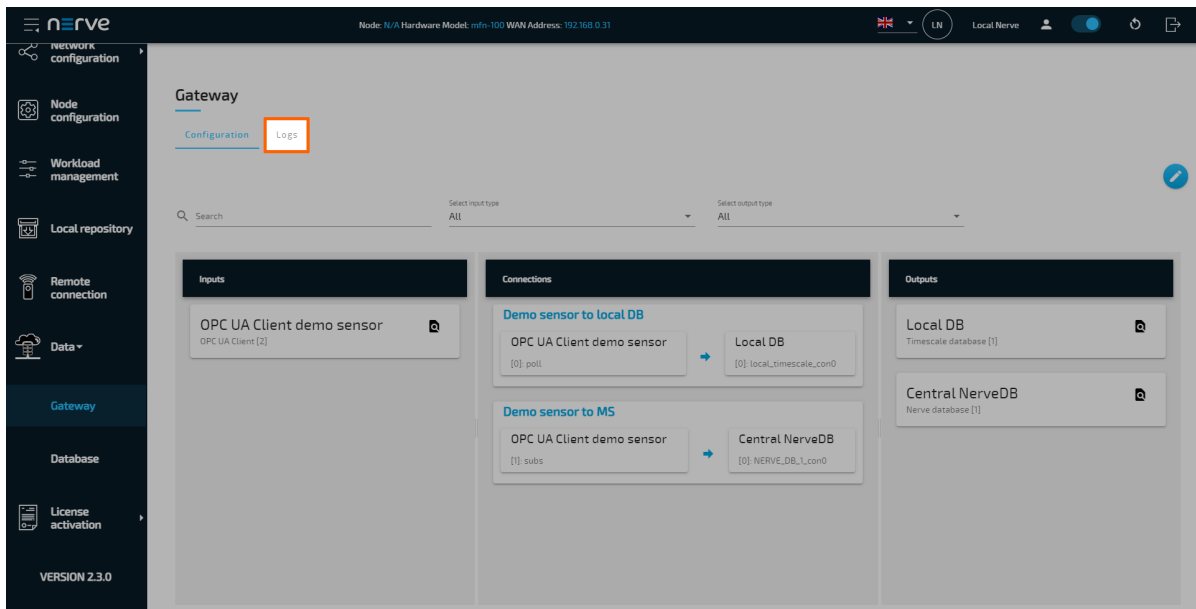


7. Add the JSON configuration file containing the code above from the file browser.
8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Verifying data with the UaExpert OPC UA Client

The temperature and humidity simulation data can be checked visually using a third party application. This example uses the [UaExpert OPC UA Client](#). The desired variables are located in the address space, and by adding them to the data access view, their values can be observed.

### NOTE

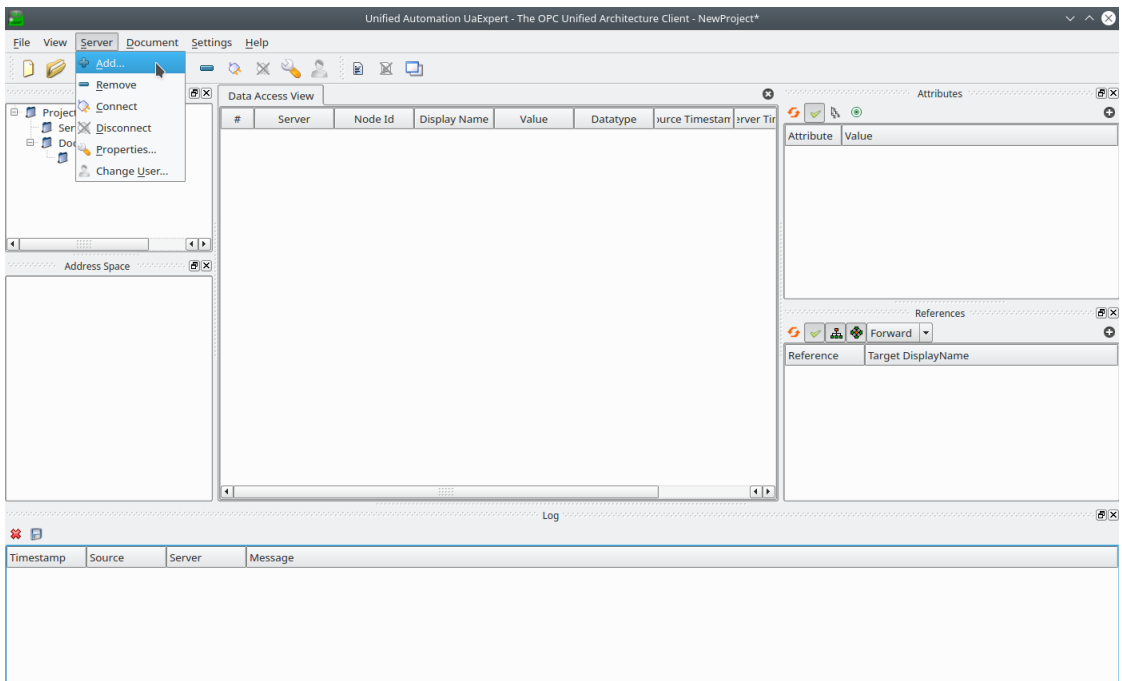
In order to download the UaExpert OPC UA Client, it is required to create a free account at [unified-automation.com](https://unified-automation.com).

Before continuing, make sure to follow the [UaExpert documentation](#) through the first steps with the UaExpert client. Afterwards follow the instructions below:

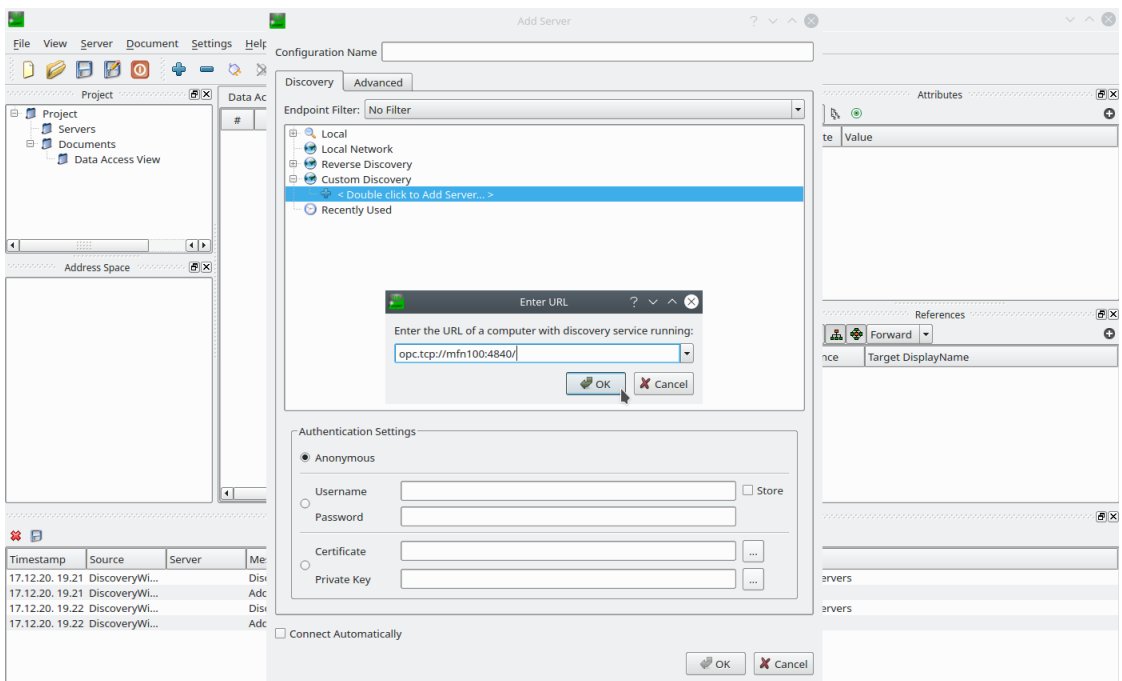
### NOTE

In the screenshots below, `opc.tcp://mfn100:4840` is used to connect to the OPC UA Server output of the Gateway, as `mfn100` had been defined as the hostname when the self-signed certificate was created. Use `172.20.2.1` if this example was followed or replace `mfn100` with the IP address or hostname that was used when creating the certificate earlier.

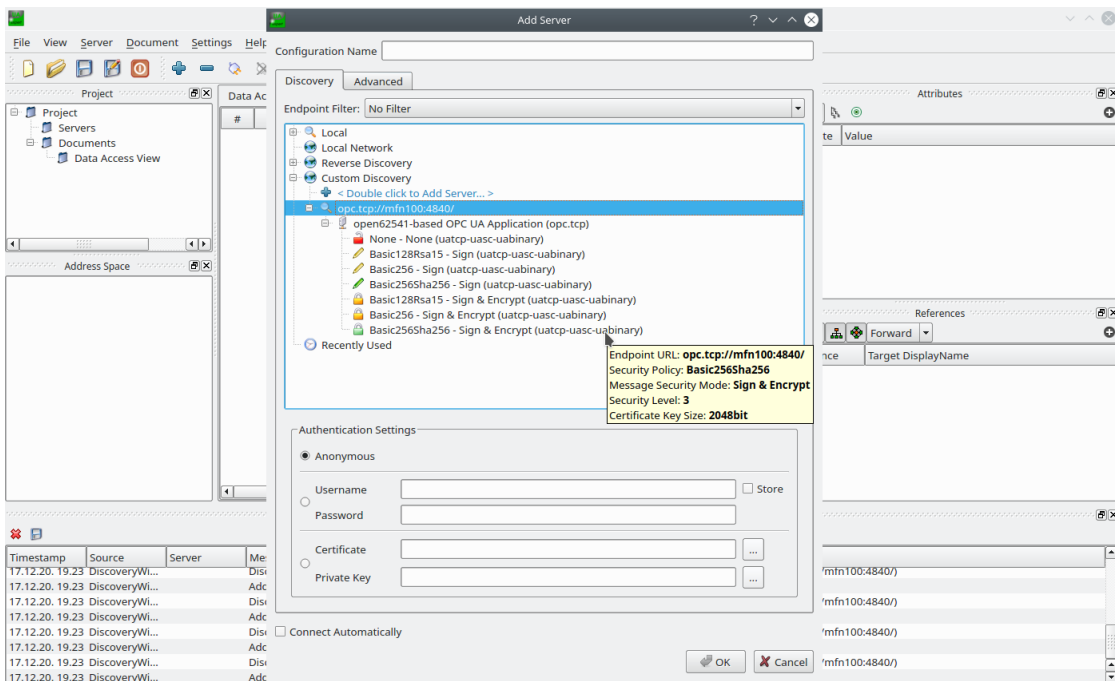
1. Select **Server > Add...** in the main menu or select the plus icon in the toolbar.



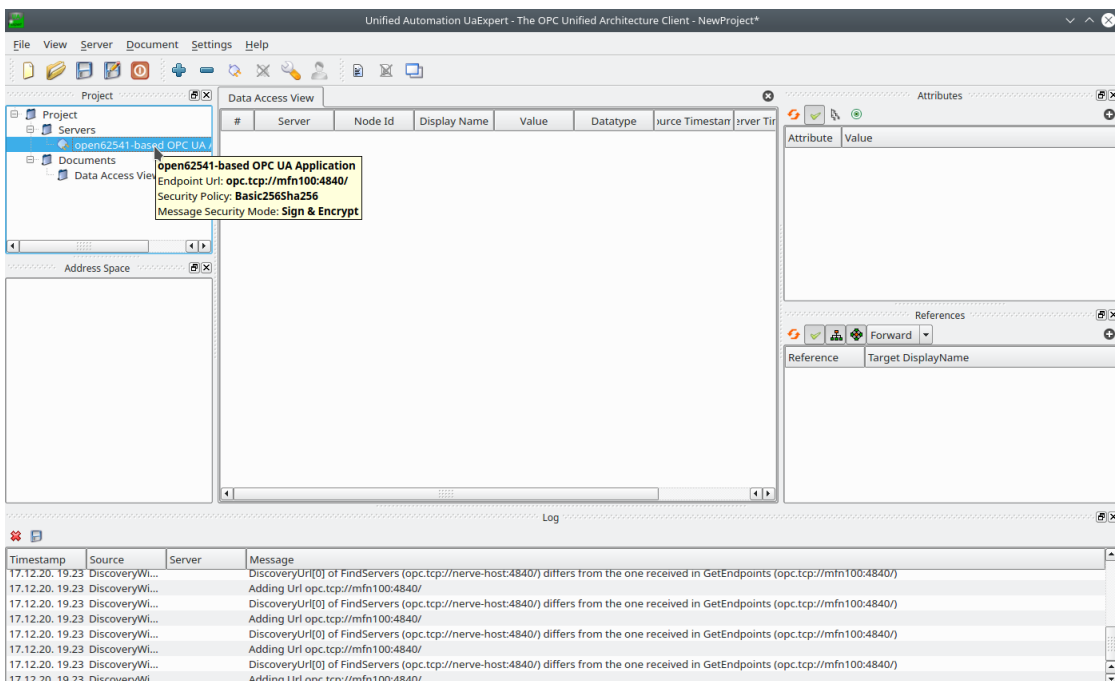
2. Select **Double click to Add Server...** under **Custom Discovery**.
3. Enter `opc.tcp://<hostname>:4840`. The new server now appears under **Custom Discovery**.



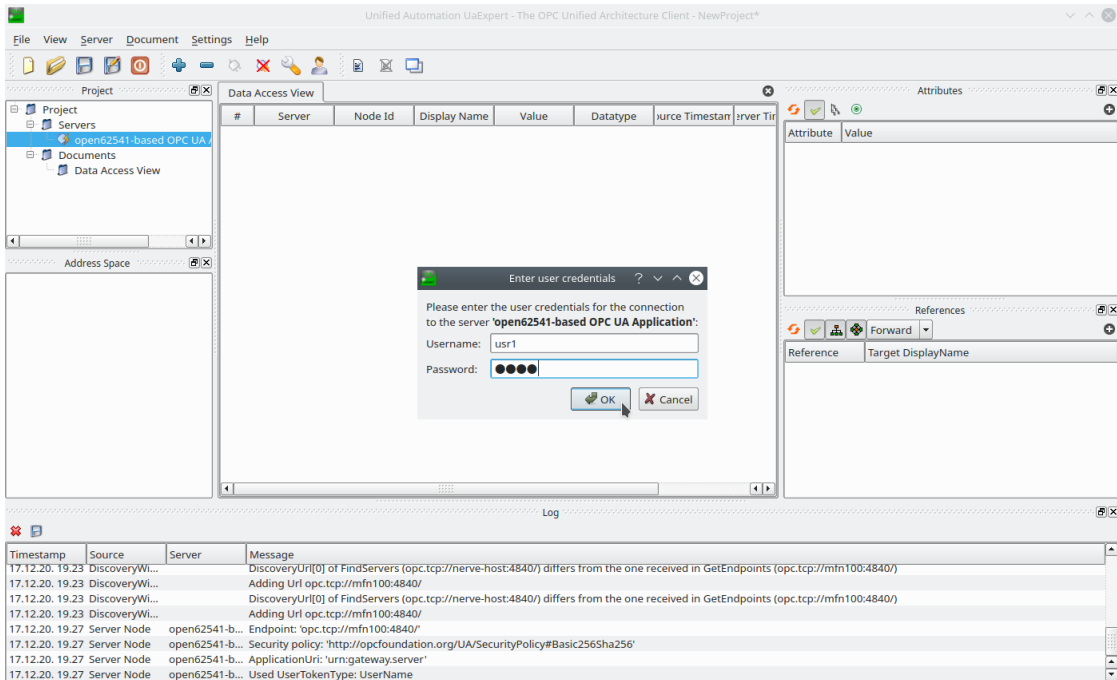
4. Expand the arrow next to the left of the new server to show all accessible endpoints of the server.



5. Double-click the desired security policy from the list. This example uses the **Basic256Sha256** security policy. The server configuration now appears in the **Project** panel under **Servers** on the left side.

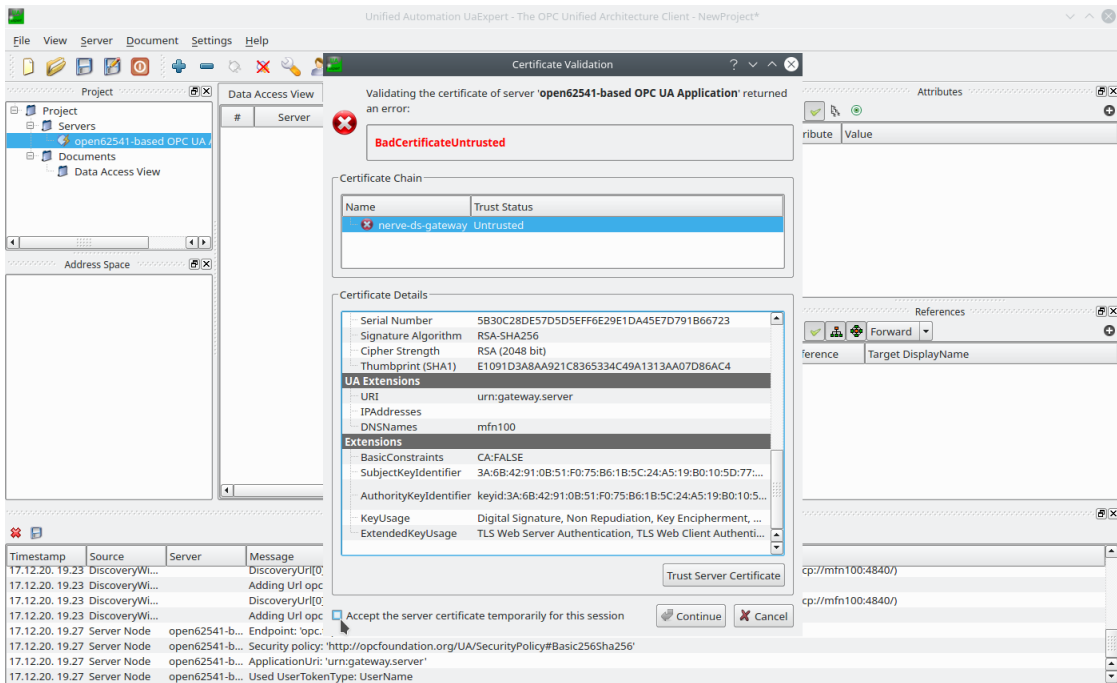


6. Select the server configuration in the **Project** panel.
7. Select **Server > Connect** in the main menu or select the **Connect Server** icon in the toolbar to establish a connection to the server.
8. Enter the username and password that were set in the OPC UA Server output configuration.



9. Select **OK**. The OPC UA Server certificate is retrieved next. Details are shown in the **Certificate Validation** window.

10. Tick the checkbox next to **Accept the server certificate temporarily for this session**.



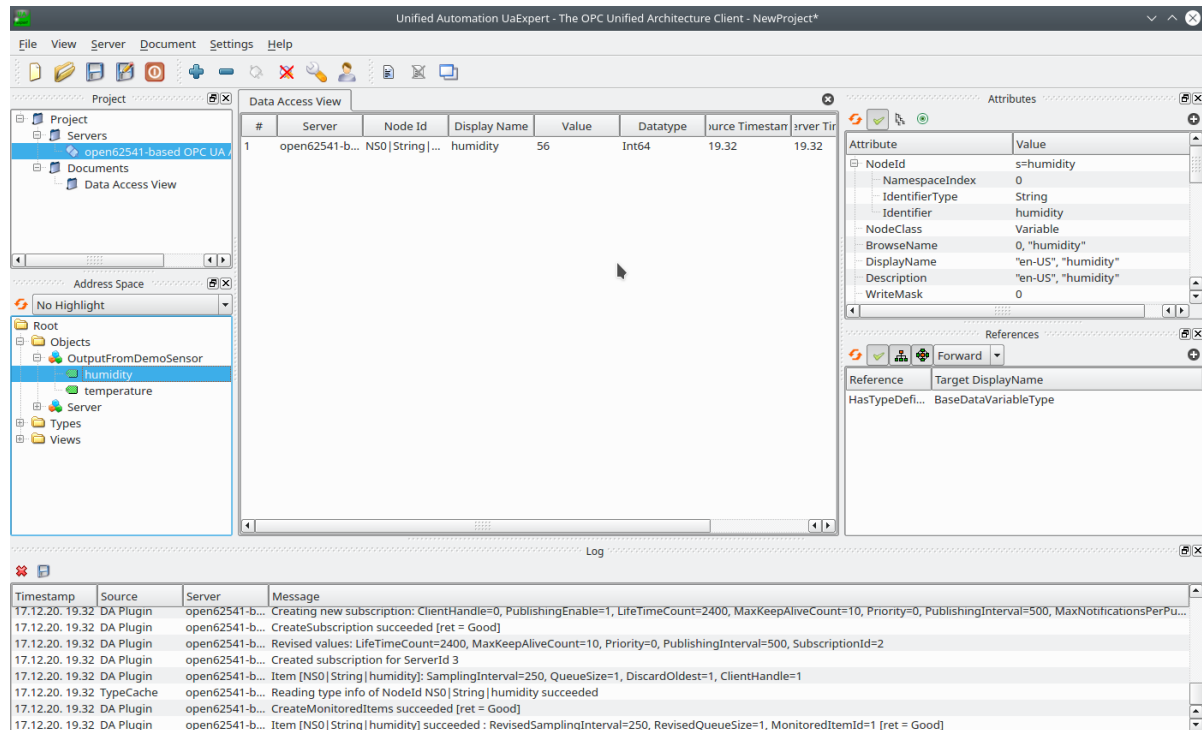
#### NOTE

Alternatively, select **Trust Server Certificate** to always trust this certificate in future sessions.

Select **Continue**.

11.

All connection steps are logged in the **Log** panel at the bottom. Once connected, the OPC UA Server's address space appears in the **Address Space** panel on the left. Expand **Objects > OutputFromDemoSensor** in the **Address Space** panel on the left and drag-and-drop the temperature and humidity variables to the **Data Access View** in the middle to monitor their values.



## S7 Client to cloud for visualization

This example demonstrates how to read data from an S7 demo server that provides temperature, revolutions and active digital inputs of a motor connected to an S7 PLC. The data is then visualized with the Central Data Services in the Management System.

The instructions below cover the following steps:

- Provisioning an S7 demo server as a Docker workload
- Deploying the provisioned Docker workload to the target node
- Configuring the Data Services Gateway
- Central data visualization in the Management System

## Provisioning and deploying an S7 demo server at the node

First, the S7 demo server must be deployed to the node as a Docker workload. Download the **Data Services S7 demo server** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo server as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.



- Provision a Docker workload by following [Provisioning a Docker workload](#). This example uses **s7-demo** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the S7 demo sensor that has been downloaded from the Nerve Software Center.
<b>DOCKER SPECIFIC INFO</b>	Select <b>New port</b> and enter the following settings: <ul style="list-style-type: none"> <li>◦ <b>Protocol:</b> TCP</li> <li>◦ <b>Host Port:</b> 102</li> <li>◦ <b>Container Port:</b> 102</li> </ul>
<b>Container name</b>	tttech-s7-server-demo
<b>Network name</b>	bridge

3. Deploy the provisioned Docker workload by following [Deploying a workload](#).

#### NOTE

Remember the node name and serial number of the target node. They are needed for the JSON configuration.

## Configuring the Data Services Gateway

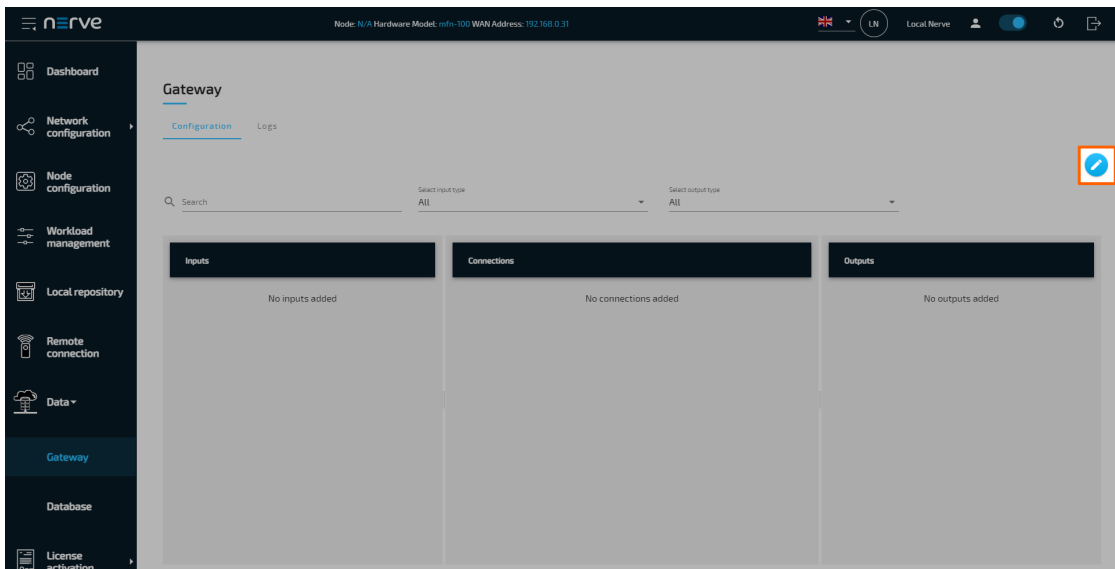
The configuration defines the S7 demo server that was deployed above as a data input and the NerveDB in the Management System as a data output. The S7 client will periodically read data from the S7 demo server and send it to the Management System. The temperature of the motor is stored in the datablock 0 with offset 0 and data type float. The number of revolutions of the motor is stored in the merker with offset 0 and data type uint32. Eight digital inputs from the demo server are also read as eight different boolean values.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.

Nerve Device	Physical port	Local UI
		<wanip>:3333
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.
		<wanip>:3333
<b>Maxtang AXWL10</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
		<wanip>:3333
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
		<wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
		<wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

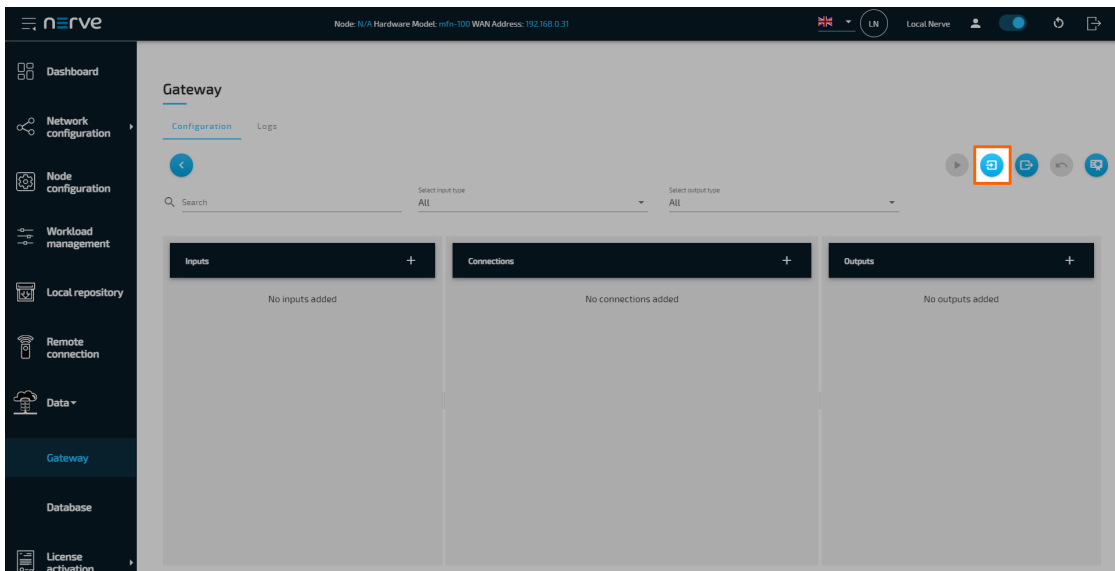
```
{
  "inputs": [
    {
      "type": "S7_CLIENT",
      "name": "s7_client_on_node",
      "serverUrl": "127.0.0.1",
      "port": 102,
      "connectionType": "PG",
      "pollingInterval_ms": 1000,
      "connectors": [
        {
          "name": "s7_demo_connector",
          "datablocks": [
            {
              "name": "motor_temperature",
              "datablock": 0,
              "offset": 0,
              "type": "float"
            }
          ],
          "merkers": [
            {
              "name": "motor_revolutions",
              "offset": 0,
              "type": "uint32"
            }
          ],
          "inputs": [
            {
              "name": "motor_input",
              "offset": 0,
              "quantity": 8,
              "type": "bool"
            }
          ]
        }
      ]
    }
  ]
}
```

```

    ],
    "outputs": [
      {
        "type": "NERVE_DB",
        "location": "CENTRAL",
        "connectors": [
          {
            "tableName": "s7_data"
          }
        ]
      }
    ]
  ],
  "connections" : [
    {
      "name" : "s7_client_to_cloud",
      "input": { "index" : 0, "connector" : 0 },
      "output": { "index" : 0, "connector" : 0 }
    }
  ]
}

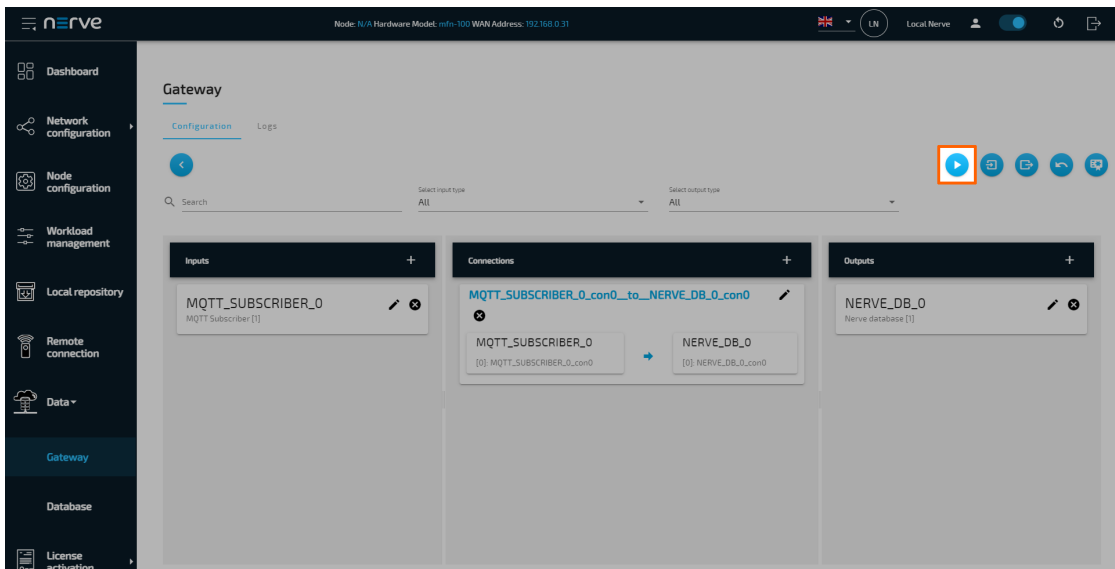
```

6. Select the **Import** button.



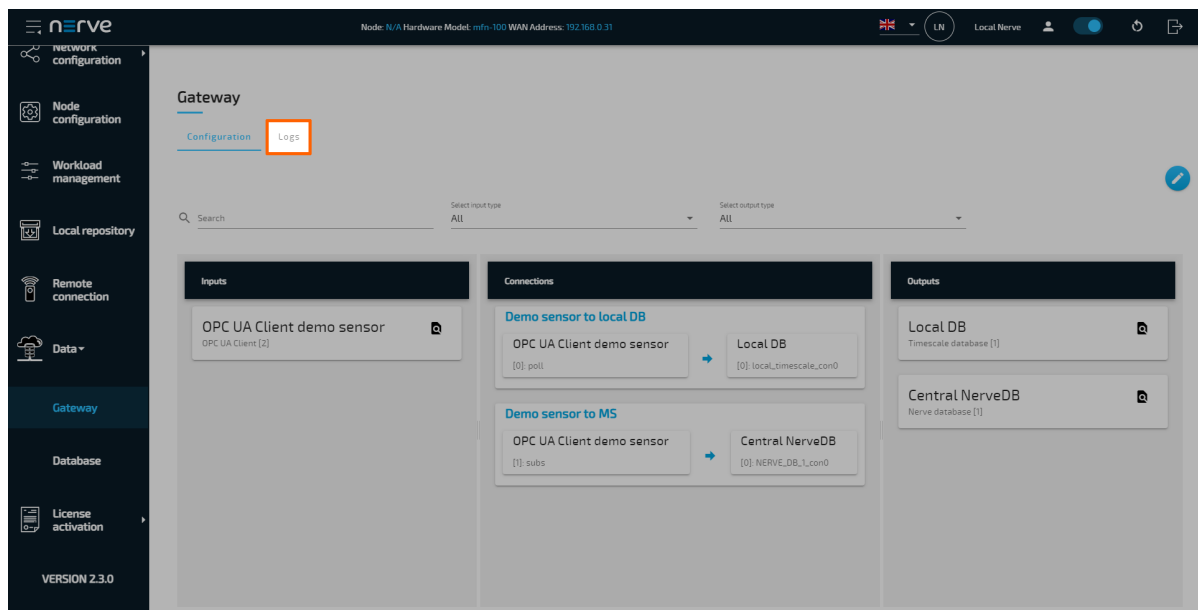
7. Add the JSON configuration file containing the code above from the file browser.

8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Central data visualization in the Management System

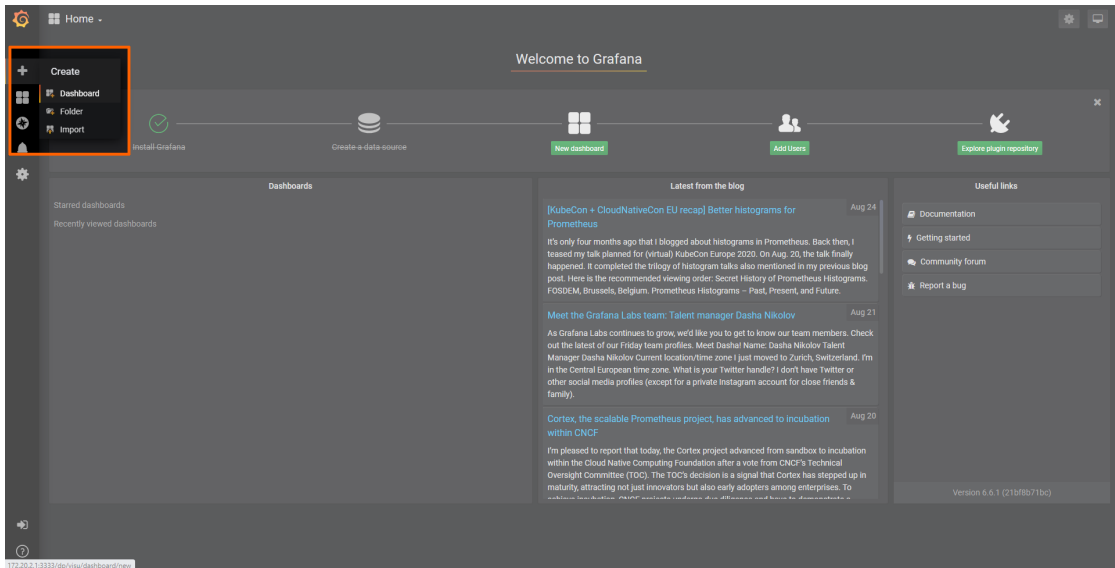
To visualize the motor temperature data in the Management System, open the central visualization element through the Data Services UI in the Management System.

1. Select **Data** in the navigation on the left. The Grafana UI will open.

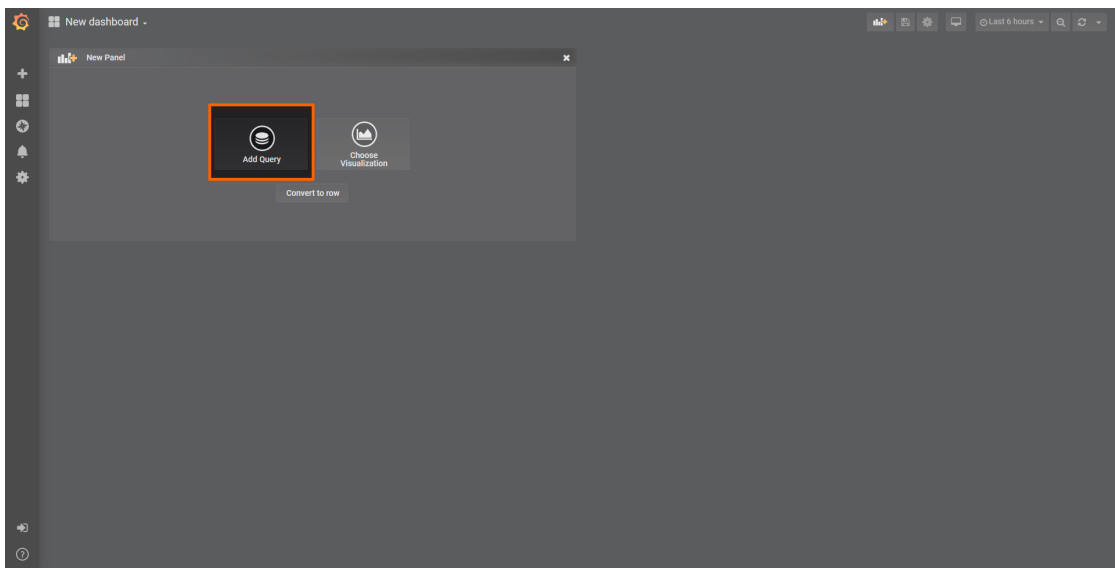
### NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

2. Select **+ > Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.



4. Select the data source from the drop-down menu. The name of the data source is the name and serial number of the node, formatted as `<nodename>` (`<serialnumber>`).

#### NOTE

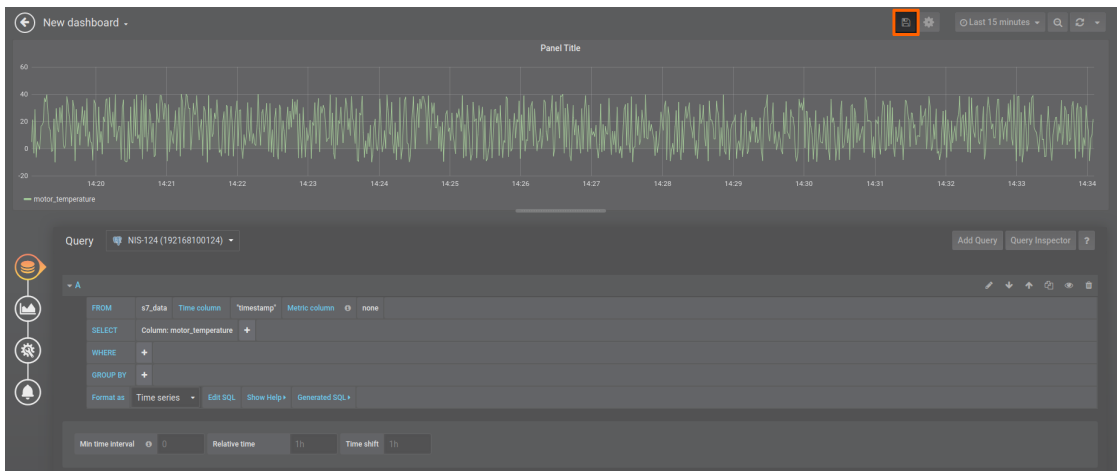
Note that multiple data sources can be selected in the Grafana instance in the Management System, depending on the number of nodes that are registered. Make sure to remember the serial number of the node that was used for workload deployment before.



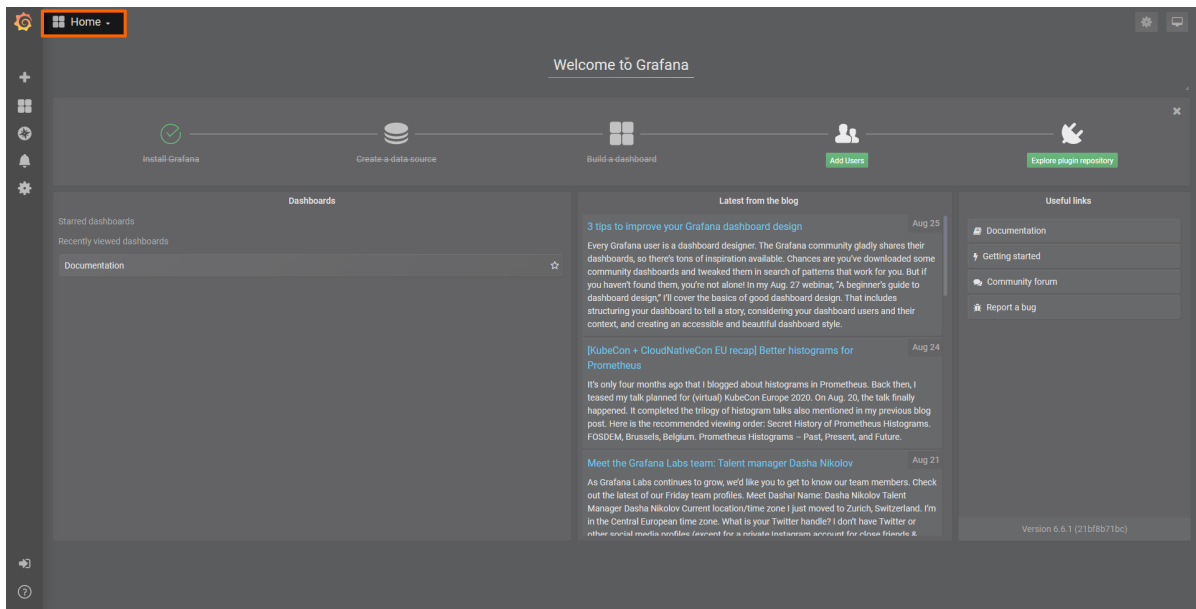
5. Fill in the following query information:

Setting	Value
<b>FROM</b>	s7_data
<b>SELECT</b>	Column: motor_temperature
<b>Format as</b>	Time series

6. Select the save icon in the upper-right corner to save the dashboard.



The dashboard can be accessed from the Grafana home menu.



## MQTT publisher to OPC UA Server at the node

In this example, temperature and humidity data is provided from a sensor simulation as an MQTT Publisher. The sensor simulation is running on the node. With the Data Services, the data is then provided as OPC UA Server on the node. A development PC with the UaExpert OPC UA Client is used to connect to the OPC UA Server on the node to read the data.

The instructions below cover the following steps:

- Provisioning an MQTT broker as a Docker workload
- Provisioning an MQTT Publisher simulation as a Docker workload
- Deploying the provisioned Docker workloads to the target node
- Configuring the Data Services Gateway
- Reading data with the UaExpert OPC UA Client on a development PC

## Provisioning and deploying the sensor simulation and the MQTT broker

In the instructions below two Docker workloads will be provisioned and deployed:

An MQTT broker must be deployed to the node first in order for the sensor simulation to function. The EMQX MQTT broker is used in this example that can be downloaded from the Docker Hub registry.

Afterwards the temperature and humidity sensors simulation MQTT publisher is deployed. Download the **Data Services MQTT demo sensor** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.



- Provision a Docker workload for the EMQX MQTT broker by following [Provisioning a Docker workload](#). This example uses **emqx-4.1.0** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>From registry</b> and enter emqx/emqx:v4.1.0.
<b>Container name</b>	emqx
<b>Network name</b>	host

3. Provision a Docker workload for the sensor simulation by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>New environment variable</b>	Select the + icon and enter the following information: <ul style="list-style-type: none"> <li>◦ <b>Env. variable</b> MQTT_PUB_TOPIC</li> <li>◦ <b>Variable value</b> demo-sensor-topic</li> </ul>
<b>Container name</b>	ttt-mqtt-demo-sensor-1.0
<b>Network name</b>	host

4. Deploy both provisioned Docker workloads above by following [Deploying a workload](#).

## Configuring the Data Services Gateway

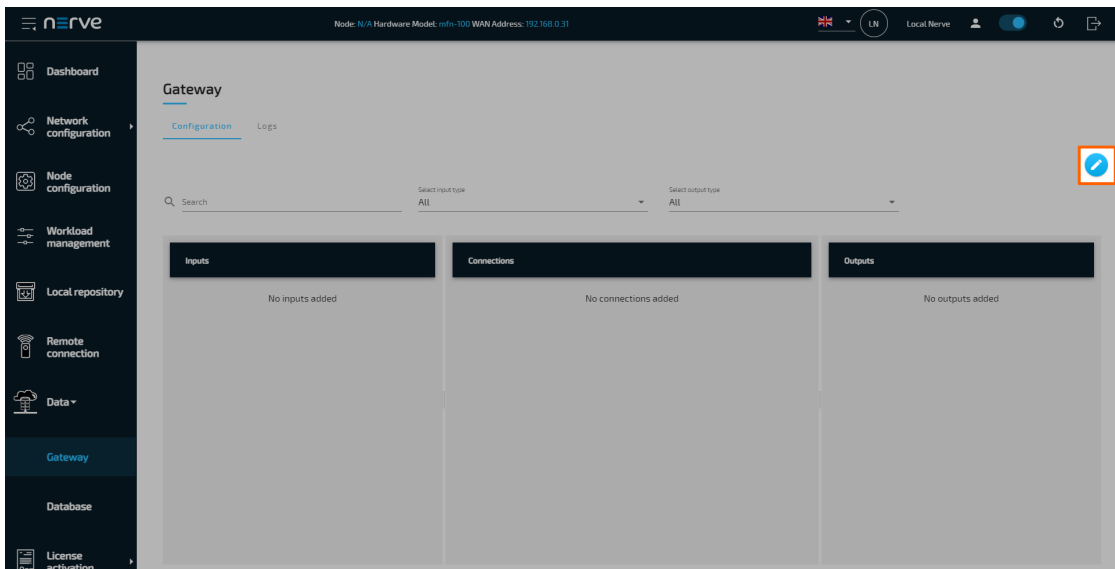
The input data is a subscription to temperature and humidity variables on the MQTT broker. An OPC UA Server is set up as data output. The input and output are linked in connections.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333  To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.

Nerve Device	Physical port	Local UI
		<wanip>:3333
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.
		<wanip>:3333
<b>Maxtang AXWL10</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
		<wanip>:3333
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
		<wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
		<wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

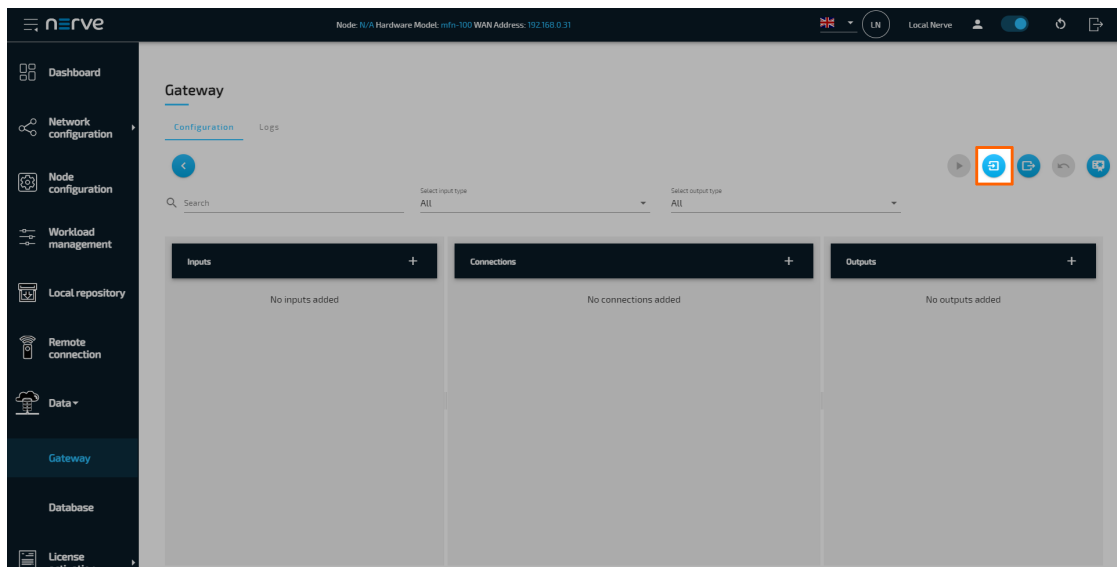
```
{
  "inputs": [
    {
      "type": "MQTT_SUBSCRIBER",
      "name": "emqx_broker_input",
      "clientId": "mqtt_subscriber_0",
      "serverUrl": "tcp://localhost:1883",
      "keepAliveInterval_s": 20,
      "cleanSession": false,
      "qos": 1,
      "connectors": [
        {
          "name": "demo_sensor_connector",
          "topic": "demo-sensor-topic",
          "variables": [
            {
              "name": "temperature",
              "type": "int16"
            },
            {
              "name": "humidity",
              "type": "uint16"
            }
          ]
        }
      ]
    }
  ],
  "outputs": [
    {
      "type": "OPC_UA_SERVER",
      "name": "opcua_server_output",
      "connectors": [
        {
          "name": "demo_sensor",
          "browseName": "demoSensor",
          "identifier": "s=demoSensor"
        }
      ]
    }
  ]
}
```

```

    }
  ],
  "connections": [
    {
      "name": "mqtt_publisher_T0_opcua_server",
      "input": {
        "index": 0,
        "connector": 0
      },
      "output": {
        "index": 0,
        "connector": 0
      }
    }
  ]
}

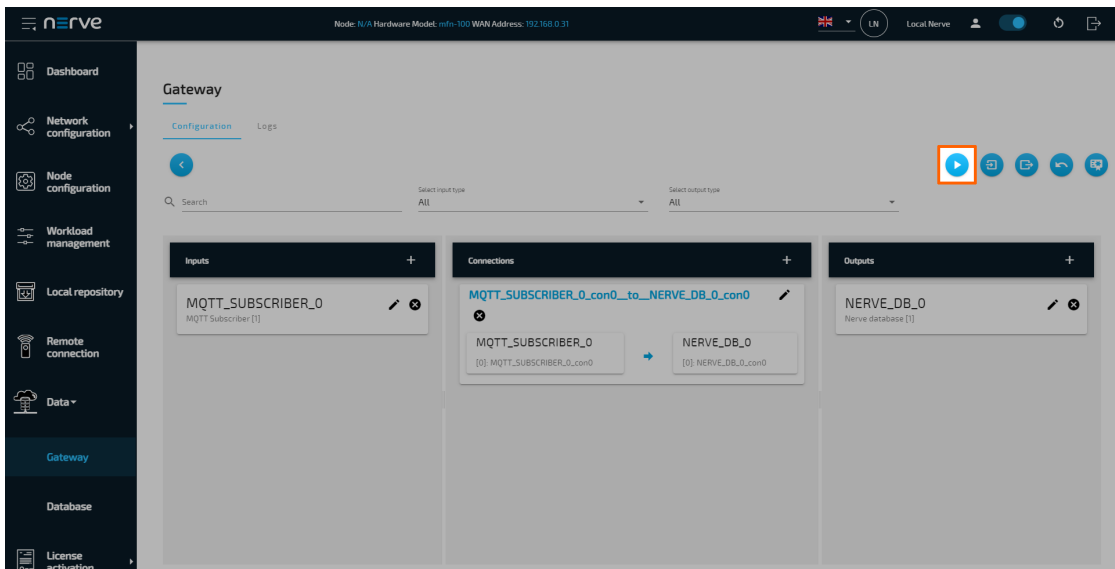
```

6. Select the **Import** button.



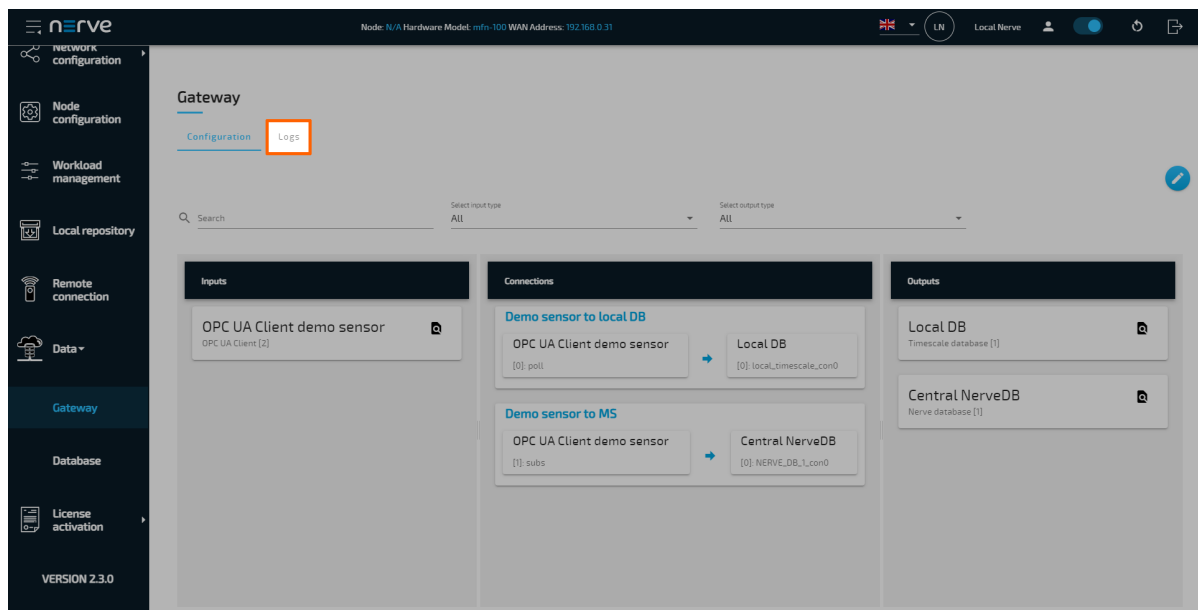
7. Add the JSON configuration file containing the code above from the file browser.

8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Reading data with the UaExpert OPC UA Client

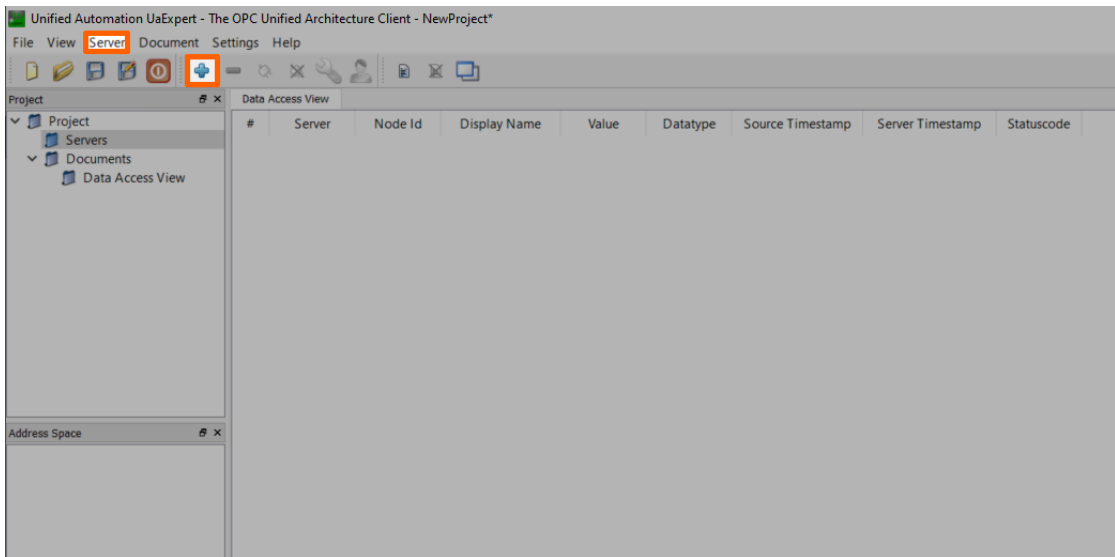
The temperature and humidity simulation data can be checked visually using a third party application. This example uses the [UaExpert OPC UA Client](#). Add a new server with the address `opc.tcp://172.20.2.1:4840` (here port 4840 is specified as the default of the Gateway) and connect to it. The desired variables are located in the address space, and by adding them to the data access view, their values can be observed.

## NOTE

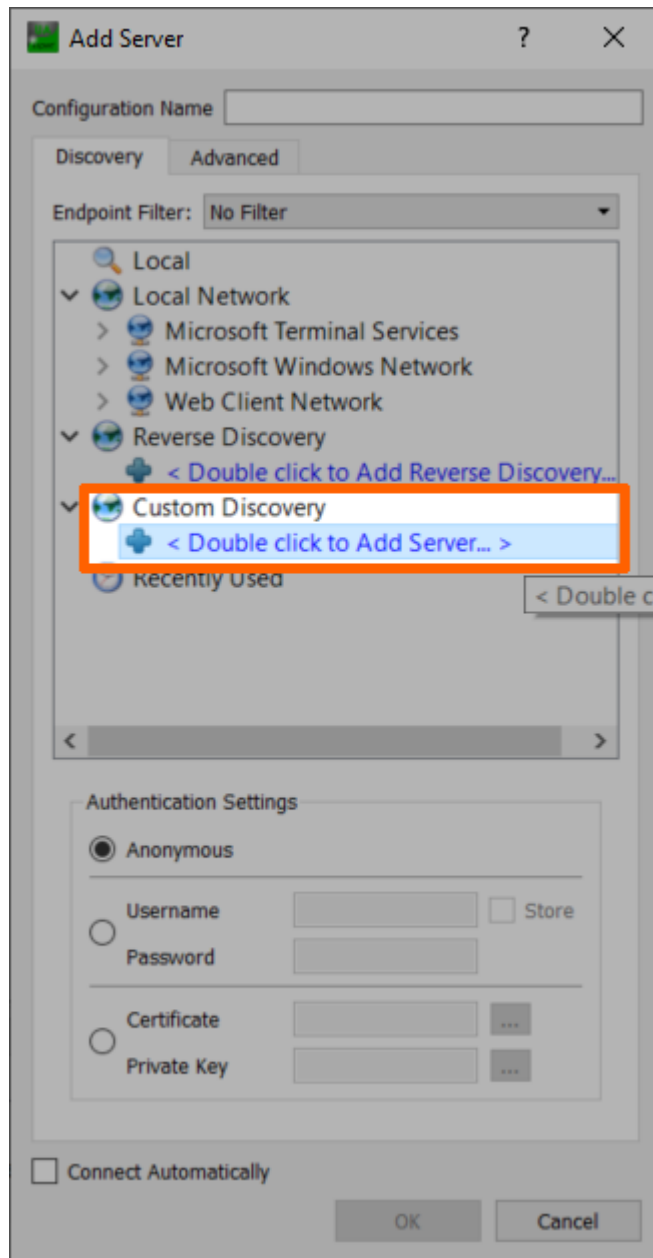
In order to download the UaExpert OPC UA Client, it is required to create a free account at [unified-automation.com](https://unified-automation.com).

Before continuing, make sure to follow the [UaExpert documentation](#) through the first steps with the UaExpert client. Afterwards follow the instructions below:

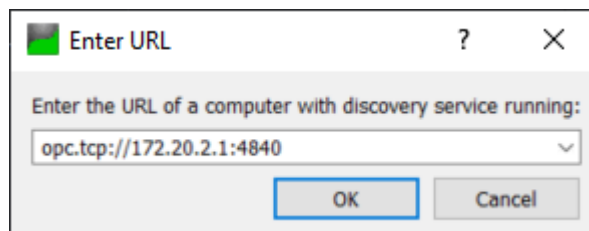
1. Select **Server > Add...** in the main menu or select the plus icon in the toolbar.



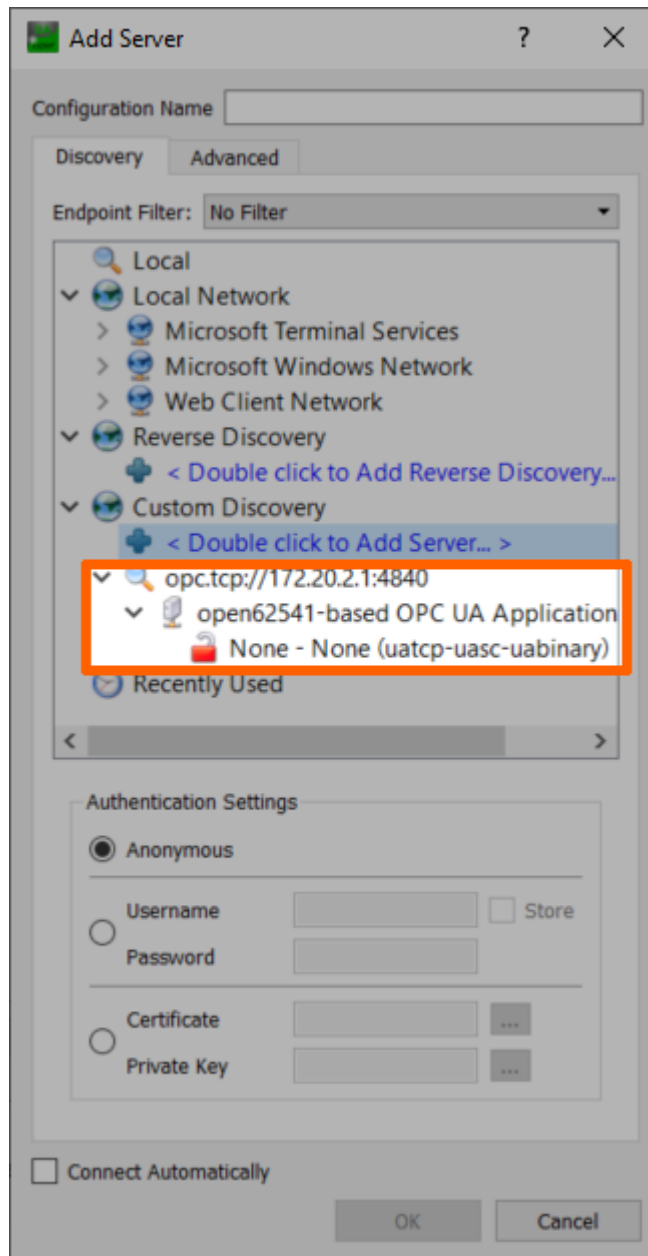
2. Select **Double click to Add Server...** under **Custom Discovery**.



3. Enter `opc.tcp://172.20.2.1:4840`. The new server now appears under **Custom Discovery**.

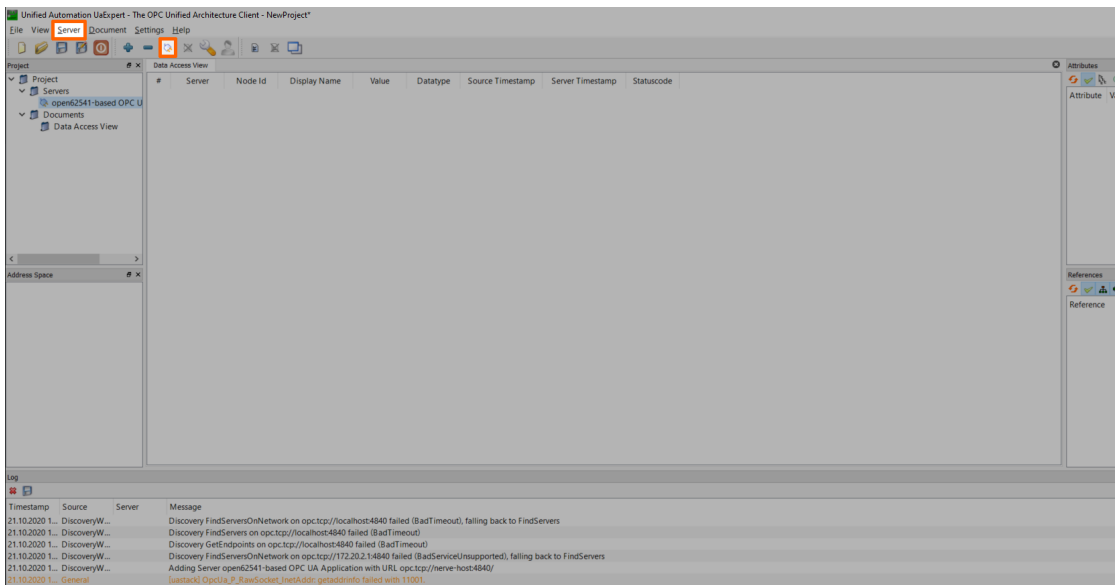


4. Expand the arrow next to the left of the new server.

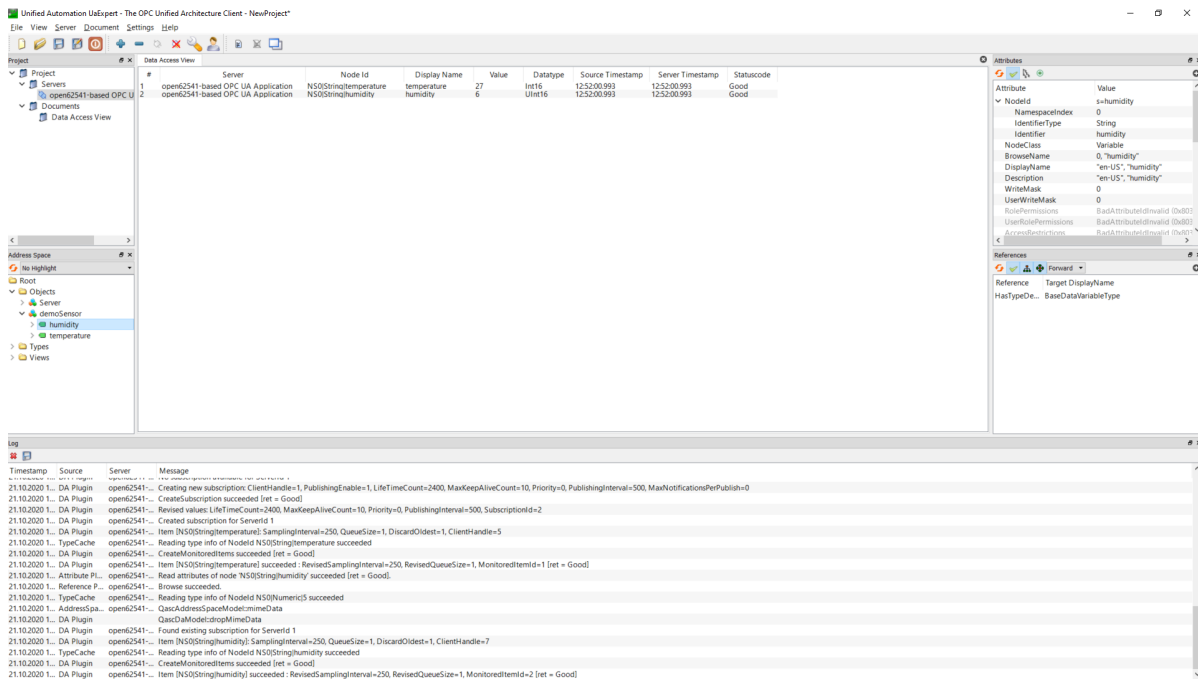


5. Double-click the **None** security policy. The server configuration now appears in the **Project** panel under **Servers** on the left side.
6. Select the server configuration in the **Project** panel.
7. Select **Server > Connect** in the main menu or select the **Connect Server** icon in the toolbar to establish a connection to the server.





All connection steps are logged in the **Log** panel at the bottom. Once connected, the OPC UA Server's address space appears in the **Address Space** panel on the left. Expand **Objects > demoSensor** in the **Address Space** panel on the left and drag-and-drop the temperature and humidity variables to the **Data Access View** in the middle to monitor their values.



## Receiving data via MQTT for Analytics and Visualization

In this example, data from a sensor providing temperature and humidity as an MQTT Publisher is visualized and processed via the Data Services and the analytics element. Data will be displayed before and after processing.

The instructions below cover the following steps:

- Provisioning an MQTT broker as a Docker workload
- Provisioning an MQTT Publisher simulation as a Docker workload
- Deploying the provisioned Docker workloads to the target node
- Creating and provisioning an analytics app with the Nerve Data SDK
- Deploying the analytics app as a Docker workload
- Local Data Visualization of temperature data before and after processing

## Provisioning and deploying the sensor simulation and the MQTT broker

In the instructions below two Docker workloads will be provisioned and deployed:

An MQTT broker must be deployed to the node first in order for the sensor simulation to function. The EMQX MQTT broker is used in this example that can be downloaded from the Docker Hub registry.

After that the temperature and humidity sensors simulation MQTT publisher is deployed. Download the **Data Services MQTT demo sensor** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload for the EMQX MQTT broker by following [Provisioning a Docker workload](#). This example uses **emqx-4.1.0** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>From registry</b> and enter <code>emqx/emqx:v4.1.0</code> .
<b>Container name</b>	<code>emqx</code>
<b>Network name</b>	<code>host</code>

3. Provision a Docker workload for the sensor simulation by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>New environment variable</b>	Select the + icon and enter the following information: <ul style="list-style-type: none"><li>◦ <b>Env. variable</b> <code>MQTT_PUB_TOPIC</code></li><li>◦ <b>Variable value</b> <code>demo-sensor-topic</code></li></ul>
<b>Container name</b>	<code>ttt-mqtt-demo-sensor-1.0</code>

Setting	Value
Network name	host

4. Deploy both provisioned Docker workloads above by following [Deploying a workload](#).

## Preparing the Nerve Data SDK

The Nerve Data SDK is required for working with analytics apps. They are created, built and provisioned with it. Download the **Nerve Data SDK** found under **Nerve Tools** from the [Nerve Software Center](#). Refer to [Data analytics](#) for more information.

## Creating and provisioning an analytics app

Before working with the SDK, make sure that the Conda environment is active. If the Conda environment is active it will be displayed in parentheses in front. The default name of the Conda environment is `nerve-ds-analytics`. Activate the Conda environment by entering the following command:

```
source miniconda/bin/activate <environmentname>
```

The Conda environment automatically deactivates after a restart so it needs to be activated whenever it is used.

1. Enter the following command to create an analytics app. `demo_sensor_analytics_app` is the name used for this example:

```
nerve-analytics create demo_sensor_analytics_app .
```

2. Enter `cd demo_sensor_analytics_app` to navigate to the newly created folder.
3. Edit the `demo_sensor_analytics_app.py` file and insert the following code:

```
import signal
import sys

from nerve_dp_analytics.stream.inputs.input_zeromq import Stream_Input_Zeromq
from nerve_dp_analytics.batch.outputs.output_timescaledb import Batch_Output_Timescaledb

running = True

def sig_hdlr(signal, frame):
    global running
    running = False

    if siz:
        try:
            siz.clear()
        except Exception as e:
            print(e)

    if bot:
        try:
            bot.clear()
        except Exception as e:
            print(e)

    print('Exiting...')
```

```

    sys.exit(0)

# catch CTRL+C
signal.signal(signal.SIGINT, sig_hdlr)

def celsius_to_kelvin(value):
    return value + 273.15

def normalize_humidity(value):
    return value / 100

try:
    siz = Stream_Input_Zeromq('demo-sensor-analytics-app-siz',
                              host='172.20.10.1',
                              port=5555,
                              topic='demo-sensor-topic')

    bot = Batch_Output_Timescaledb('demo-sensor-analytics-app-bot',
                                    table_name='demo_sensor_analyzed_data',
                                    vars={'temperature': 'real',
                                          'humidity': 'real'})

    while(running):
        try:
            data = siz.receive(Stream_Input_Zeromq.DTYPE_LIST)
            new_data = list()

            for d in data:
                nd = dict()

                nd['timestamp'] = d['timestamp']
                nd['temperature'] = celsius_to_kelvin(d['temperature'])
                nd['humidity'] = normalize_humidity(d['humidity'])

                new_data.append(nd)

            for nd in new_data:
                bot.send(nd)
            except Exception as e:
                print(e)
except Exception as e:
    print(e)

```

This analytics app receives data from the Gateway through the ZeroMQ Stream Input, which is a default way of transferring data between the Gateway and analytics. Processed data is stored in a TimescaleDB via the TimescaleDB Batch Output. When only `table_name` is provided for this output, the analytics write data into the default database of the node that has the node serial number as a name.

In this example, basic processing is done on the data provided by the demo sensor. Temperature data is converted from Celsius to Kelvin while humidity data is normalized to a range between 0 and 1.

#### NOTE

The ZeroMQ Publisher output of the Gateway must publish messages on 172.20.10.1 if analytics are running on the node in the `nerve-ds` Docker

network. Consequently, the ZeroMQ Stream Input of the analytics must listen on the same IP address.

4. Edit the Dockerfile and insert the following:

```
FROM python:3.8.3-slim-buster
WORKDIR /nerve
COPY nerve_dp_analytics_api-1.0-py3-none-any.whl .
RUN pip install wheel nerve_dp_analytics_api-1.0-py3-none-any.whl
WORKDIR /
COPY demo_sensor_analytics_app.py .
CMD [ "python", "-u", "demo_sensor_analytics_app.py" ]
```

5. Enter the following command to build the Docker image containing the analytics app. `nerve-ds-2.1.1` is used as the name in this example:

```
nerve-analytics build -t nerve-ds-2.1.1
```

6. Enter the following command to provision the analytics app as a Docker workload in the Management System:

```
nerve-analytics provision -u https://<MS-URL> -n "Data Services Analytics - demoSe
```

This will provision a Docker workload with the following settings:

Setting	Description
<b>Workload name</b>	Data Services Analytics - demoSensor App
<b>Description</b>	Docker container running a Nerve Data Services analytics app that processes temperature and humidity data.
<b>Version name</b>	nerve-ds-2.1.1
<b>Release name</b>	nerve-ds-2.1.1
<b>CPU resource in percentage</b>	1
<b>Container name</b>	analytics-demo-sensor-app
<b>Network name</b>	nerve-ds

#### NOTE

Due to version differences, the workload is created with a maximum of 1% of allowed CPU usage. Change this setting to a value between 10 and 25.

All settings except **Container name** and **Network name** in the command above or in the Management System are suggestions and can be changed freely.

With the analytics app provisioned in the Management System, the app needs to be deployed to the node to analyze data coming from the demo sensor. Deploy the app to the node that has the demo sensor and the MQTT broker deployed by following [Deploying a workload](#).

## Configuring the Data Services Gateway

The configuration below defines an MQTT Subscriber connection to a ZeroMQ Publisher. Upon receiving data from the demo-sensor-topic topic of the MQTT Subscriber, the Gateway forwards said data to the ZeroMQ Publisher. The ZeroMQ Publisher in turn publishes the data to the demo-sensor-topic ZeroMQ topic that the analytics app listens to. The configuration also defines a connection from the MQTT Subscriber to the TimescaleDB database, which means that the same data received at the MQTT Subscriber end is also written directly into the TimescaleDB.

### NOTE

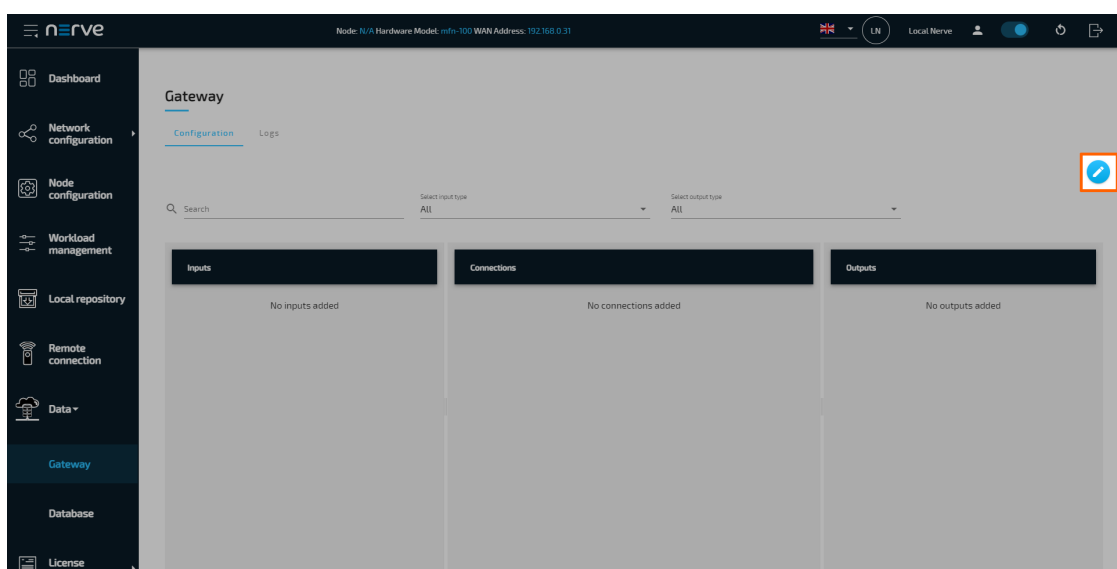
Note that both topics have the same name in this example. However, they are different as they are topics of two different protocols, MQTT and ZeroMQ.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide. <wanip>:3333
<b>Maxtang AXWL10</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>

Nerve Device	Physical port	Local UI
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

```

{
  "inputs": [
    {
      "type": "MQTT_SUBSCRIBER",
      "name": "mqtt_subscriber",
      "clientId": "mqtt_subscriber_0",
      "serverUrl": "tcp://localhost:1883",
      "keepAliveInterval_s": 20,
      "cleanSession": false,
      "qos": 1,
      "connectors": [
        {
          "name": "mqtt_subscriber_connector_0",
          "topic": "demo-sensor-topic",
          "variables": [
            {
              "name": "temperature",
              "type": "int16"
            },
            {
              "name": "humidity",
              "type": "uint16"
            }
          ]
        }
      ]
    }
  ],
  "outputs": [
    {
      "type": "ZEROMQ_PUBLISHER",
      "name": "zeromq_publisher_0",
      "serverUrl": "tcp://172.20.10.1:5555",
      "connectors": [
        {
          "name": "zeromq_publisher_connector_0",
          "topic": "demo-sensor-topic",
          "timestampRequired": true,
          "timestampFormat": "unix_ns"
        }
      ]
    },
    {
      "type": "DB_TIMESCALE",
      "name": "timescaledb_0",
      "url": "<LOCAL>"
    }
  ],
  "connections": [
    {
      "name": "mqttsub_zmqpub_0",
      "input": {
        "index": 0,
        "connector": 0
      },
      "output": {
        "index": 0,
        "connector": 0
      }
    }
  ]
}

```

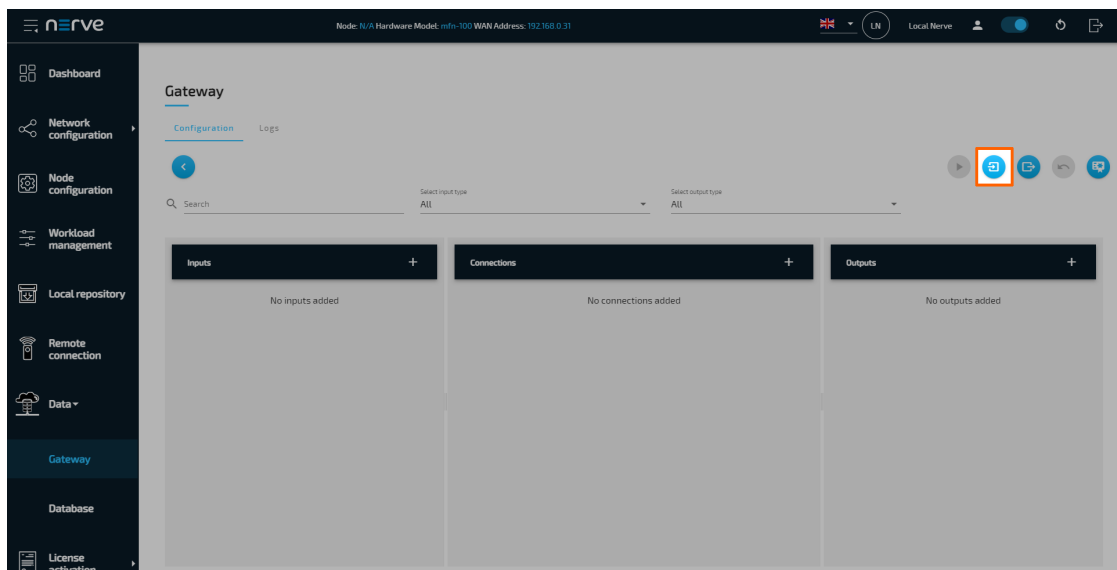


```

    },
    {
      "name": "mqttsub_timescaledb_0",
      "input": {
        "index": 0,
        "connector": 0
      },
      "output": {
        "index": 1,
        "connector": 0
      }
    }
  ]
}

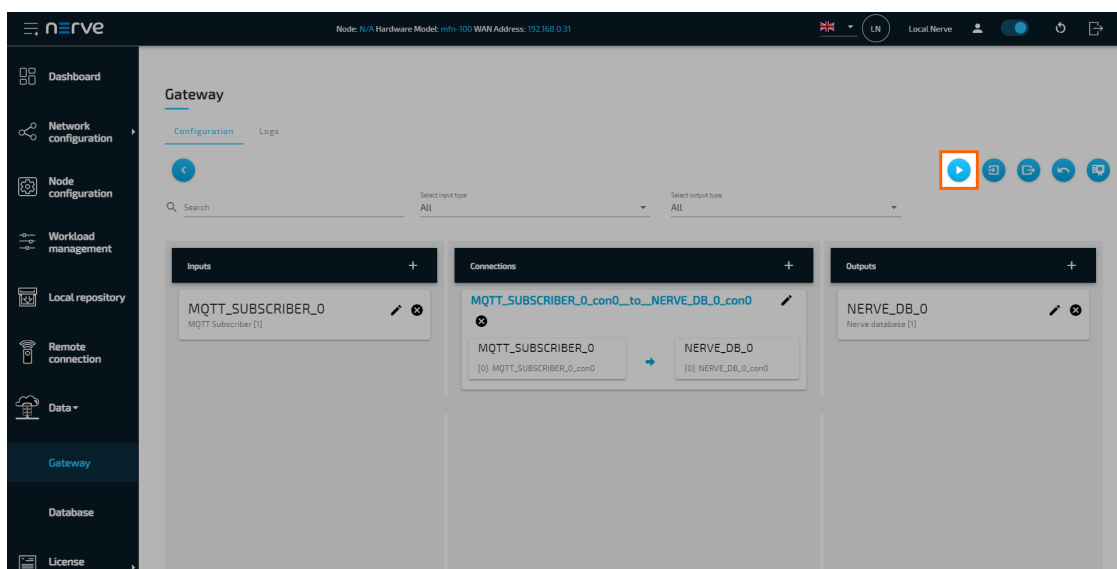
```

6. Select the **Import** button.



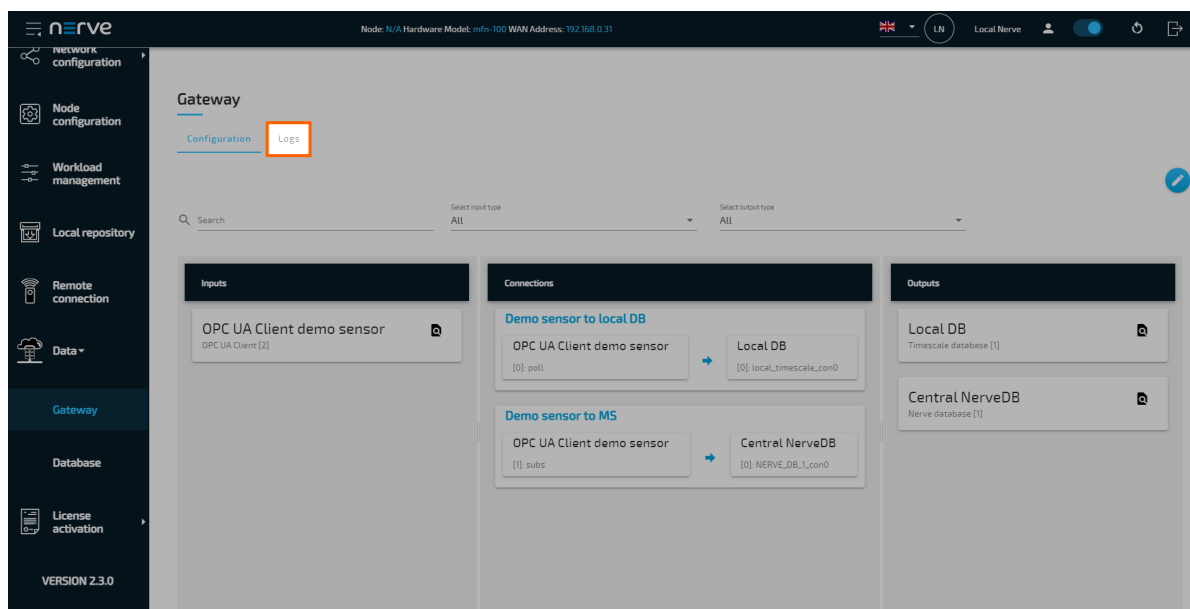
7. Add the JSON configuration file containing the code above from the file browser.

8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Local data visualization at the node

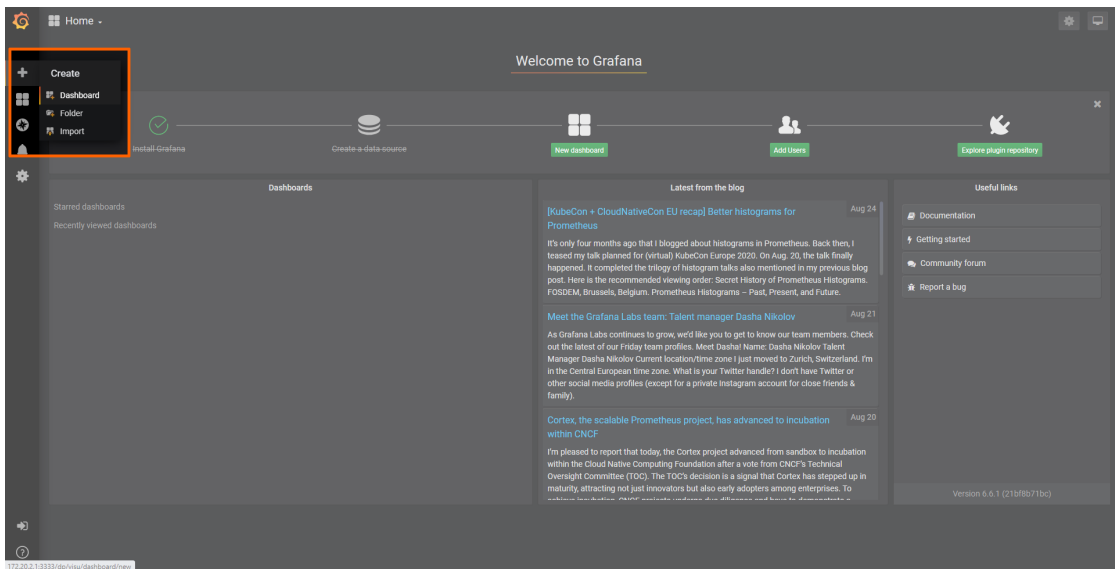
To visualize the data received by the Gateway and data processed by the analytics app, open the local data visualization element through the Data Services UI on the node. Two queries will be added in the instructions below.

1. Select **Data** in the navigation on the left. The Grafana UI will open.

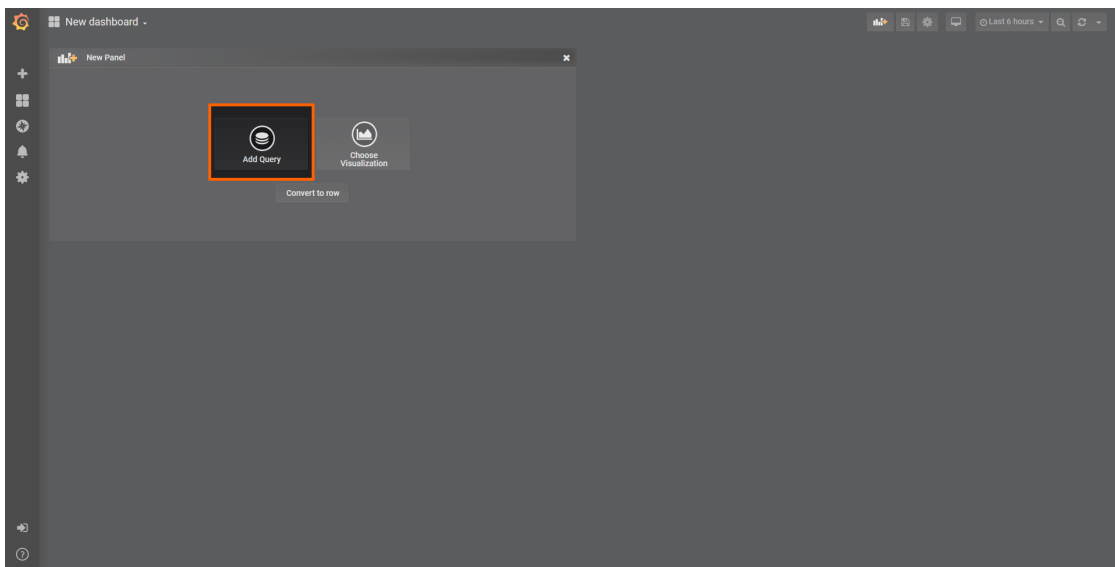
### NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

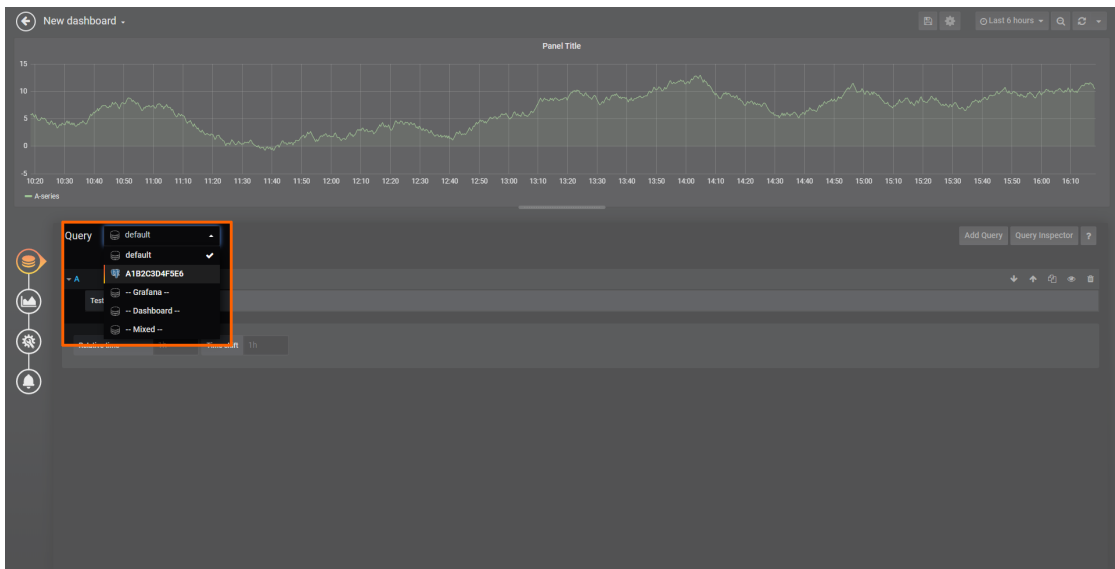
2. Select + > **Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.



4. Select the data source from the drop-down menu. The name of the data source is the serial number of the node.



5. Fill in the following query information to add the temperature data from the MQTT Subscriber:

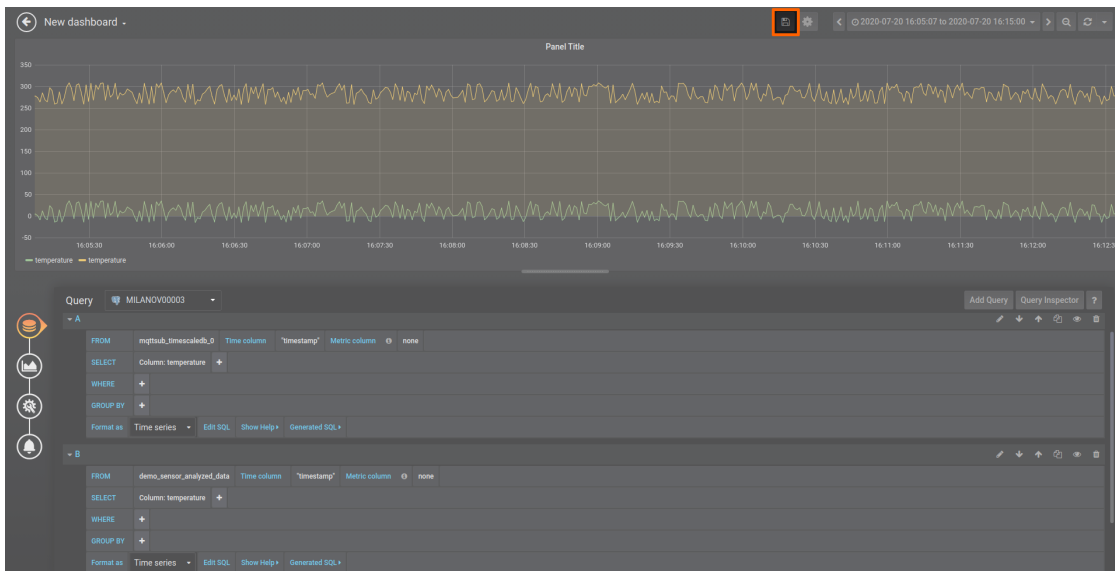
Setting	Value
<b>FROM</b>	mqttsub_timescaledb_0
	<b>Time column:</b> "timestamp"
<b>SELECT</b>	Column: temperature
<b>Format as</b>	Time series

6. Select **Add Query** to the right to add query B for temperature data analyzed by the analytics app.

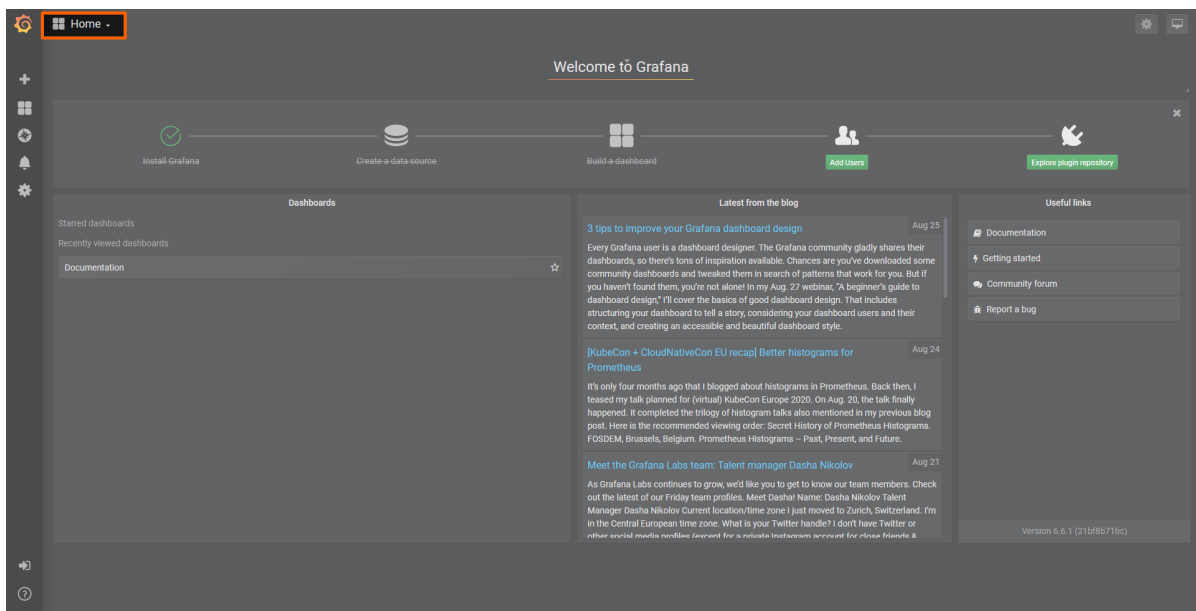
7. Fill in the following query information:

Setting	Value
<b>FROM</b>	demo_sensor_analyzed_data
	<b>Time column:</b> "timestamp"
<b>SELECT</b>	Column: temperature
<b>Format as</b>	Time series

8. Select the save icon in the upper-right corner to save the dashboard.



The dashboard can be accessed from the Grafana home menu.



## Custom JSON format example

In this example, data is provided to the Nerve Data Services Gateway in a custom JSON format. Compared to the previous version of the Gateway, data can be input more freely, as it was previously required to strictly adhere to the JSON format of the Gateway.

The data in the custom JSON format is provided by an MQTT Publisher in form of a demo sensor to be processed and stored by the Gateway for visualization at the node.

## Provisioning and deploying the sensor simulation and the MQTT broker

In the instructions below two Docker workloads will be provisioned and deployed:

An MQTT broker must be deployed to the node first in order for the sensor simulation to function. The EMQX MQTT broker is used in this example that can be downloaded from the Docker Hub registry.

Afterwards the temperature and humidity sensors simulation MQTT publisher is deployed. Download the **Data Services MQTT demo sensor** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload for the EMQX MQTT broker by following [Provisioning a Docker workload](#). This example uses **emqx-4.1.0** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>From registry</b> and enter emqx/emqx:v4.1.0.
<b>Container name</b>	emqx
<b>Network name</b>	host

3. Provision a Docker workload for the sensor simulation by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>New environment variable</b>	Select the + icon and enter the following information: <ul style="list-style-type: none"> <li>◦ <b>Env. variable</b> MQTT_PUB_TOPIC</li> <li>◦ <b>Variable value</b> demo-sensor-topic</li> </ul>
<b>Container name</b>	ttt-mqtt-demo-sensor-1.0
<b>Network name</b>	host

4. Deploy both provisioned Docker workloads above by following [Deploying a workload](#).

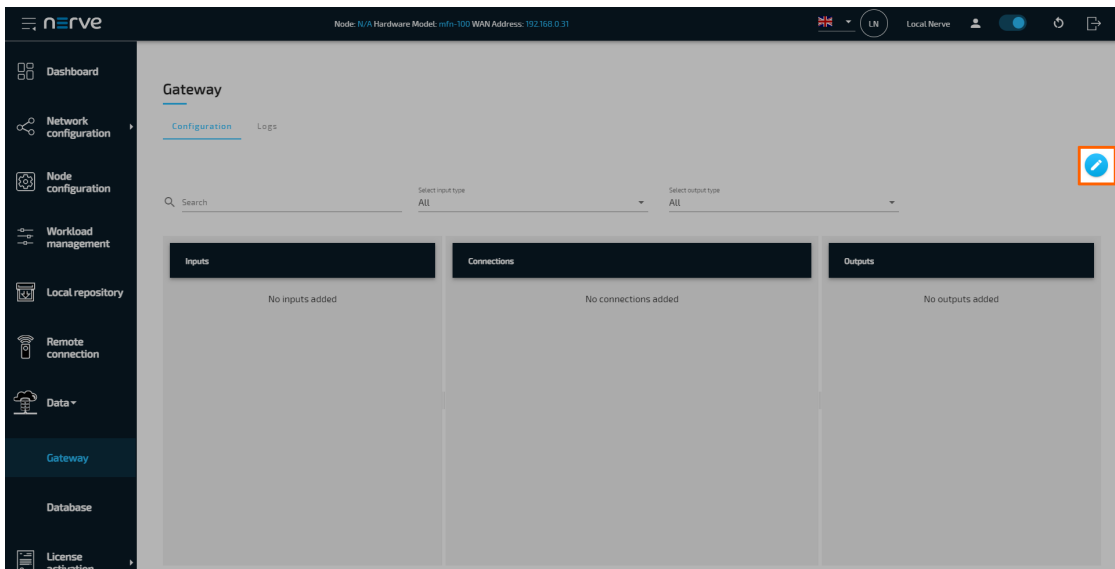
## Configuring the Data Services Gateway

The Gateway configuration in the instructions below defines an MQTT subscriber as an input that receives data in a custom JSON format on four different topics. The output is a TimescaleDB database with four different tables, one per topic.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide. <wanip>:3333
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide. <wanip>:3333
<b>Maxtang AXWL10</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide. <wanip>:3333
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

```
{
  "inputs": [
    {
      "type": "MQTT_SUBSCRIBER",
      "name": "mqtt_subscriber",
      "clientId": "mqtt_subscriber_0",
      "serverUrl": "tcp://localhost:1883",
      "keepAliveInterval_s": 20,
      "cleanSession": true,
      "qos": 1,
      "connectors": [
        {
          "name": "mqtt_subscriber_connector_0",
          "topic": "machineA",
          "variables": [
            {
              "name": "temperature",
              "type": "int32",
              "path": ".machineA.temperatureSensor"
            },
            {
              "name": "top-left.distance",
              "type": "double",
              "path": ".machineA.distance_sensors"
            },
            {
              "name": "top-right.distance",
              "type": "double",
              "path": ".machineA.distance_sensors"
            }
          ]
        }
      ]
    }
  ]
}
```



```

        "name": "bottom-left.distance",
        "type": "double",
        "path": ".machineA.distance_sensors"
    },
    {
        "name": "bottom-right.distance",
        "type": "double",
        "path": ".machineA.distance_sensors"
    }
]
},
{
    "name": "mqtt_subscriber_connector_1",
    "topic": "machineB",
    "timestamp": {
        "path": ".info.measurement_time[]"
    },
    "variables": [
        {
            "type": "int32",
            "path": ".machineB.sensor1[].temperature"
        },
        {
            "type": "uint32",
            "path": ".machineB.sensor1[].humidity"
        },
        {
            "type": "int32",
            "path": ".machineB.sensor2[].temperature"
        },
        {
            "type": "uint32",
            "path": ".machineB.sensor2[].humidity"
        }
    ]
},
{
    "name": "mqtt_subscriber_connector_2",
    "topic": "machineC",
    "variables": [
        {
            "type": "string",
            "path": ".machineC[*].entry",
            "maxValues": 4
        }
    ]
},
{
    "name": "mqtt_subscriber_connector_3",
    "topic": "all_combined",
    "variables": [
        {
            "type": "uint8",
            "path": "[*].[]",
            "maxValues": 11
        },
        {
            "type": "uint8",
            "path": "[].[*]",

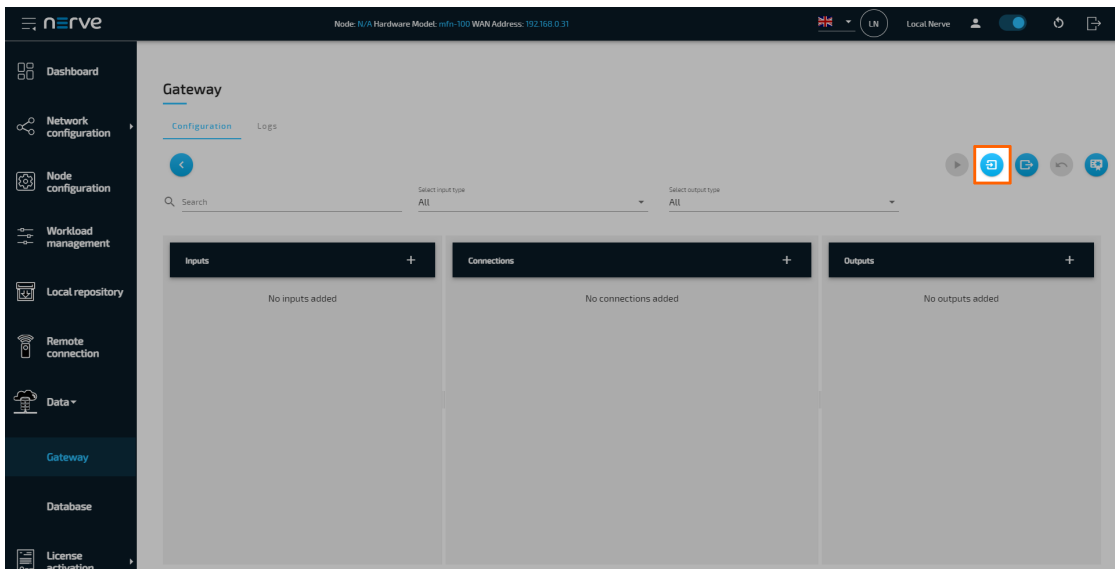
```

```

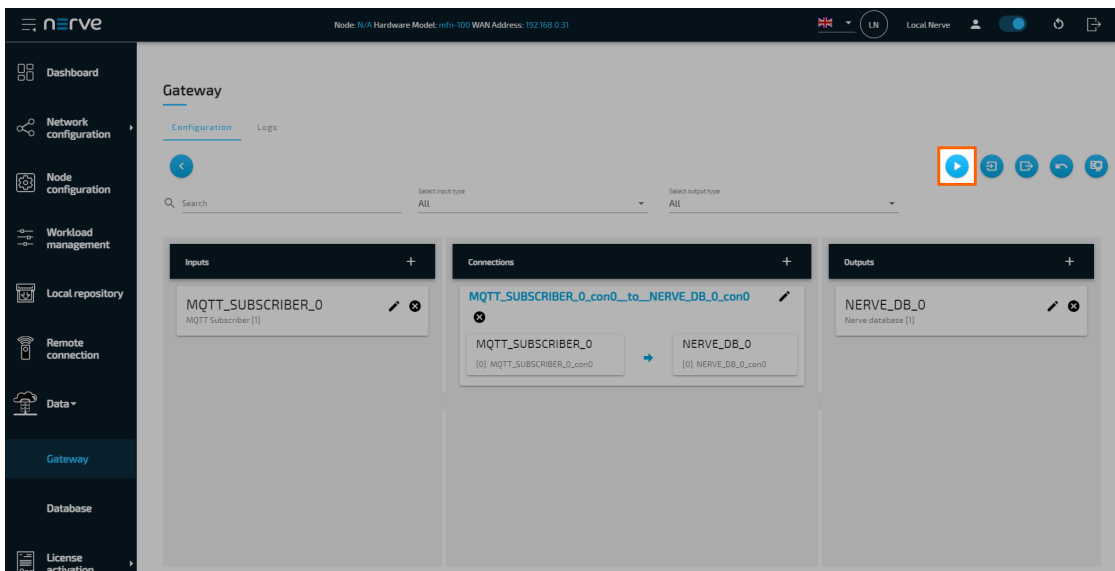
        "maxValues": 11
      }
    ]
  }
],
"outputs": [
  {
    "type": "DB_TIMESCALE",
    "name": "outp_timescale",
    "url": "<LOCAL>",
    "connectors": [
      {
        "tableName": "machineA_data"
      },
      {
        "tableName": "machineB_data"
      },
      {
        "tableName": "machineC_data"
      },
      {
        "tableName": "all_combined_data"
      }
    ]
  }
],
"connections": [
  {
    "name": "connection_0",
    "input": { "index": 0, "connector": 0 },
    "output": { "index": 0, "connector": 0 }
  },
  {
    "name": "connection_1",
    "input": { "index": 0, "connector": 1 },
    "output": { "index": 0, "connector": 1 }
  },
  {
    "name": "connection_2",
    "input": { "index": 0, "connector": 2 },
    "output": { "index": 0, "connector": 2 }
  },
  {
    "name": "connection_3",
    "input": { "index": 0, "connector": 3 },
    "output": { "index": 0, "connector": 3 }
  }
]
}

```

6. Select the **Import** button.

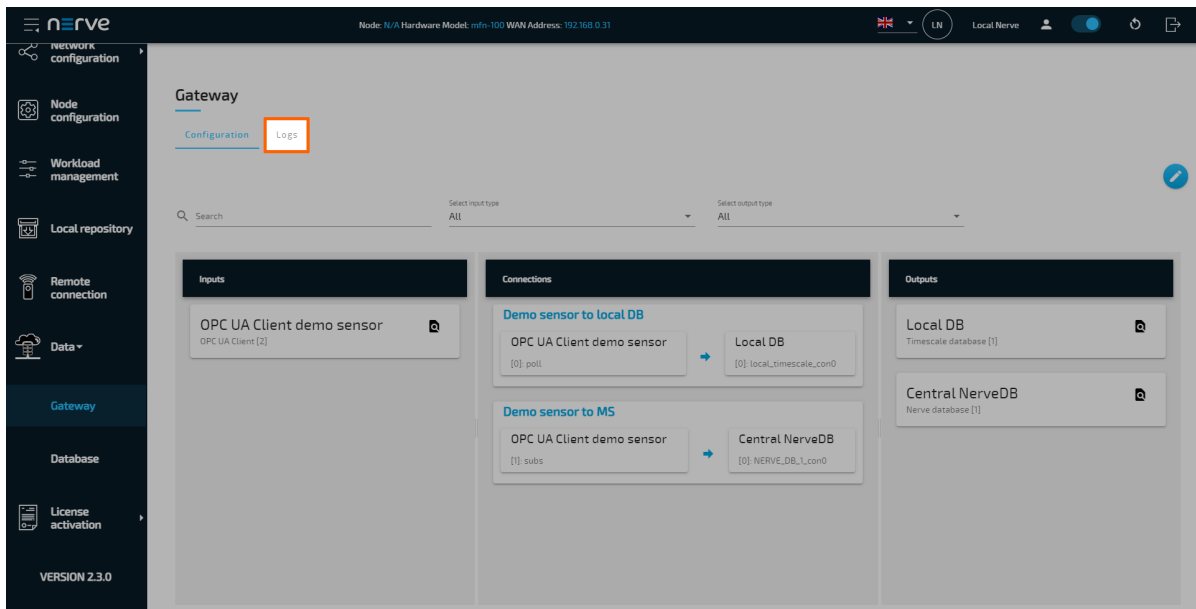


7. Add the JSON configuration file containing the code above from the file browser.
8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Taking a closer look at the data

The input is an MQTT Subscriber that receives data in custom JSON format on four different topics. The output is a TimescaleDB database with four different tables, one per topic or data.

### First connector data

The first connector receives data from a simulation sensor:

```
{
  "machineA": {
    "type": "molding",
    "date_installed": "2018-07-03",
    "temperatureSensor": {
      "state": "MEASURING",
      "hasError": false,
      "temperature": 34
    },
    "distance_sensors": {
      "top-left": {
        "distance": 0.26210331244354845,
        "hasError": false
      },
      "top-right": {
        "distance": 0.26046464329587893,
        "hasError": false
      },
      "bottom-left": {
        "distance": 0.6323130733349438,
        "hasError": false
      },
      "bottom-right": {
        "distance": 0.8431831637166814,
        "hasError": false
      }
    }
  }
}
```

```
}  
}
```

With the configuration above only data that is defined in the Gateway configuration is extracted and written into the database. By specifying exactly which fields value is required, all other data is eliminated. This gives flexibility to the Gateway to support all kinds of JSON formats.

### Example data table

top-left.distance	top-right.distance	bottom-left.distance	bottom-right.distan
0.26210331244354845	0.26046464329587893	0.6323130733349438	0.8431831637166814

### Second connector data

The second connector also receives data from a simulation server, but in batches:

```
{  
  "info": {  
    "measurement_time": [  
      1611936105084752100,  
      1611936105084770000,  
      1611936105084773000  
    ]  
  },  
  "machineB": {  
    "sensor1": [  
      {  
        "temperature": 25,  
        "humidity": 24  
      },  
      {  
        "temperature": 26,  
        "humidity": 100  
      },  
      {  
        "temperature": 28,  
        "humidity": 78  
      }  
    ],  
    "sensor2": [  
      {  
        "temperature": -11,  
        "humidity": 84  
      },  
      {  
        "temperature": 33,  
        "humidity": 5  
      },  
      {  
        "temperature": -11,  
        "humidity": 88  
      }  
    ]  
  }  
}
```

With the configuration for this connector, the Gateway extracts lists of data. For example, the `sensor1[].temperature` list expansion extracts all temperature field values from `sensor1`. Note that the `sensor1` array must have objects with the same fields. This also applies to `sensor1[].humidity`, `sensor2[].temperature` and `sensor2[].humidity`. Combining all list expansions for the data example above, the Gateway inserts three rows in the database. Each row is constructed by the field index in the JSON.

### Example data table

<code>.machineB.sensor1[].temperature</code>	<code>.machineB.sensor1[].humidity</code>	<code>.machineB.sensor2[]</code>
25	24	-11
26	100	33
28	78	-11

### Third connector data

The third connector also receives data in batches:

```
{
  "machineC": [
    {
      "entry": "mywclgbwcl"
    },
    {
      "entry": "ctuswukfqp"
    },
    {
      "entry": "maaqtyssvq"
    },
    {
      "entry": "usepjkpylg"
    }
  ]
}
```

Here only a single array of data is received to show the use of the variable expansion. The Gateway reads `machineC[*].entry` as a list of entries. However, instead of inserting four rows into the database, it inserts one row with four columns. Each column is labelled with the name of the values field and extended with an index of the field in the array. By supporting variable expansion, the Gateway can fetch multiple variables from a JSON without specifying them all.

### Example data table

<code>.machineC[*].entry-0</code>	<code>.machineC[*].entry-1</code>	<code>.machineC[*].entry-2</code>	<code>.machineC[*].entry-</code>
mywclgbwcl	ctuswukfqp	maaqtyssvq	usepjkpylg

### Fourth connector data

The last connector shows the use case of combining list and variable expansions in a single Gateway variable:

```
[
  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
  [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
]
```

```

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
]

```

The first variable specifies the variable expansion first and the list expansion second ([\*]. []). This means that a variable is each individual array and that array values are fetched as a list. Ten columns are created with ten rows inserted. Each row holds its respective value (first row, all 0, second row, all 1...).

[*]. []-0	[*]. []-1	[*]. []-2	[*]. []-3	[*]. []-4	[*]. []-5	[*]. []-6	[*]. []-7	[*]. []-8	[*]. []-9	[*]. []-10
0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9	9
10	10	10	10	10	10	10	10	10	10	10

The second variable specifies the list expansion first and the variables expansion second ([] . [\*]). This means that a variable is a single element in an array and that values are fetched from all arrays for that element. Ten columns are created with ten rows inserted. Each column holds its respective value (first column, all 0, second column, all 1...).

[], [*]-0	[], [*]-1	[], [*]-2	[], [*]-3	[], [*]-4	[], [*]-5	[], [*]-6	[], [*]-7	[], [*]-8	[], [*]-9	[], [*]-10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	4	5	6	7	8	9	10

## Local data visualization at the node

To visualize the data received by the Gateway, open the local data visualization element through the Data Services UI on the node. Two separate queries will be added in the instructions below.

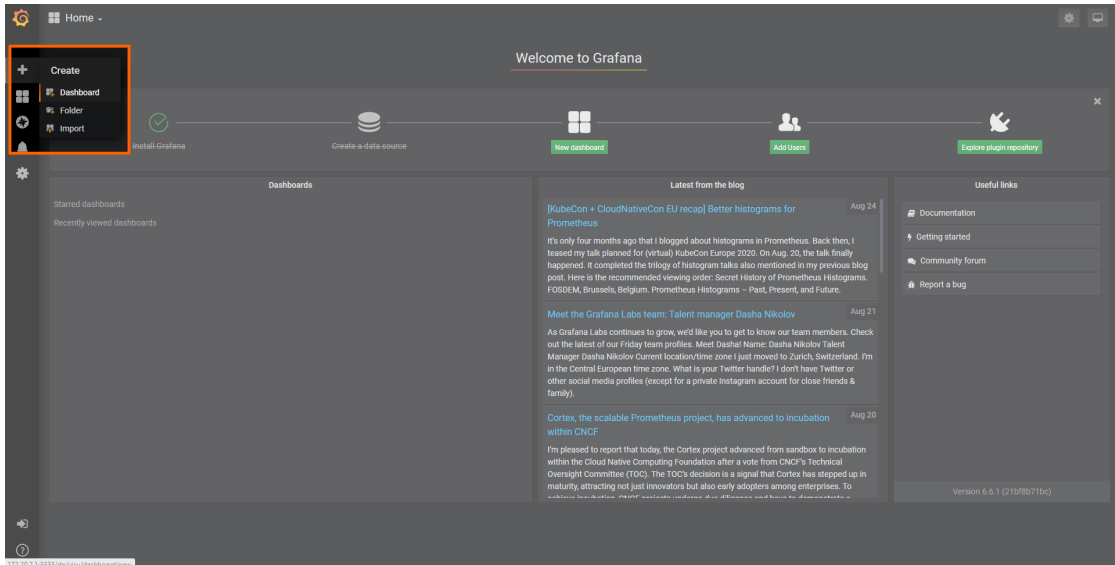
### Visualizing machineA data

1. Select **Data** in the navigation on the left. The Grafana UI will open.

#### NOTE

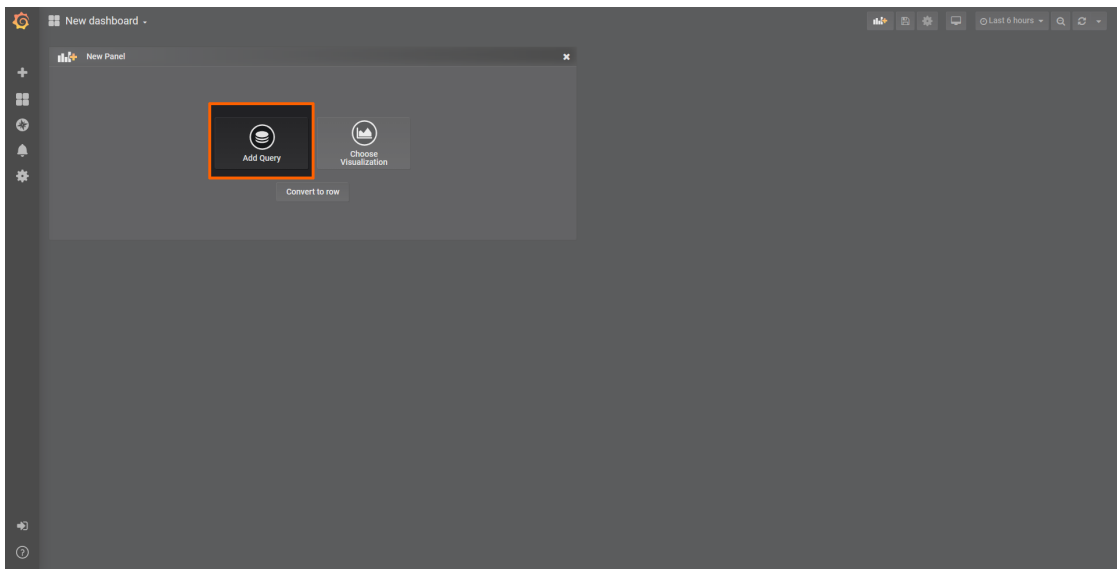
Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

2. Select **+ > Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.





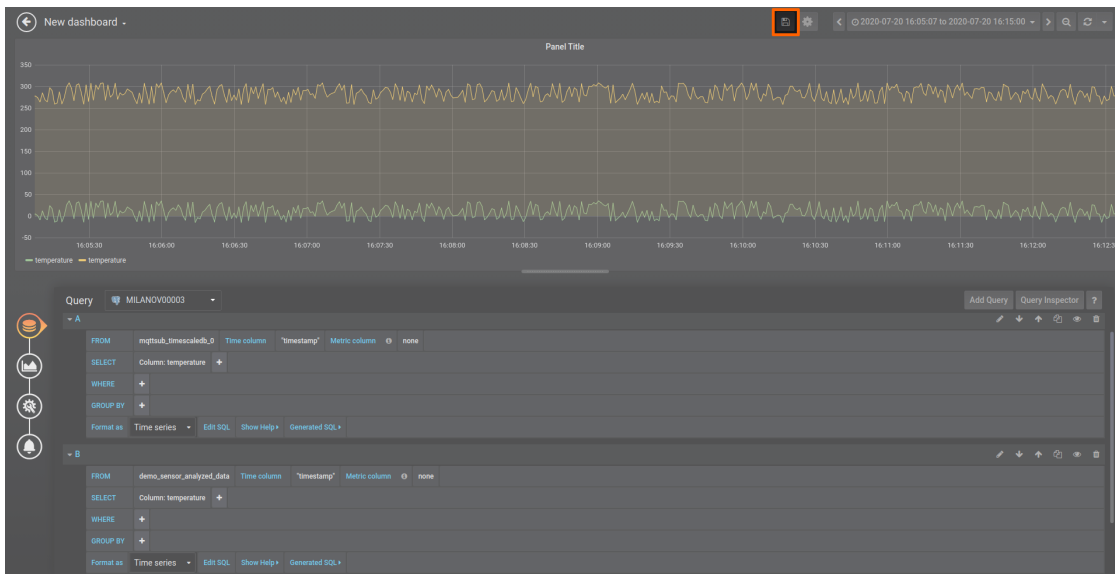
4. Select the data source from the drop-down menu. The name of the data source is the serial number of the node.



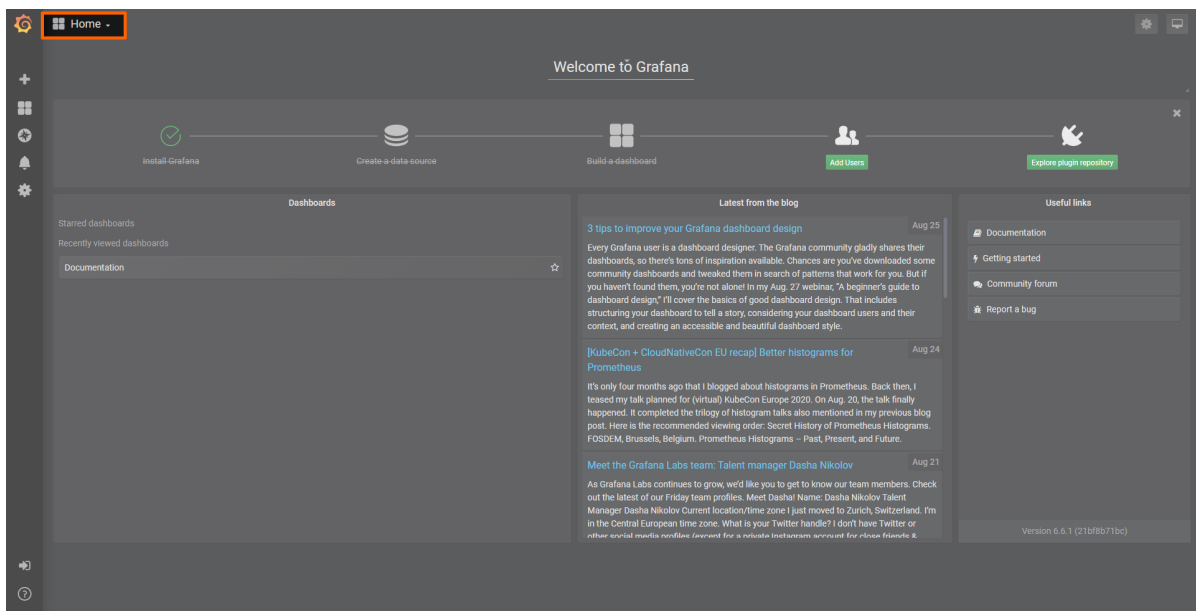
5. Fill in the following query information to add the data of Machine A:

Setting	Value
<b>FROM</b>	machineA_data
<b>SELECT</b>	Column: "bottom-left.distance" Column: "bottom-right.distance" Column: "top-left.distance" Column: "top-right.distance"
<b>Format as</b>	Time series

6. Select the save icon in the upper-right corner to save the dashboard.



The dashboard can be accessed from the Grafana home menu.



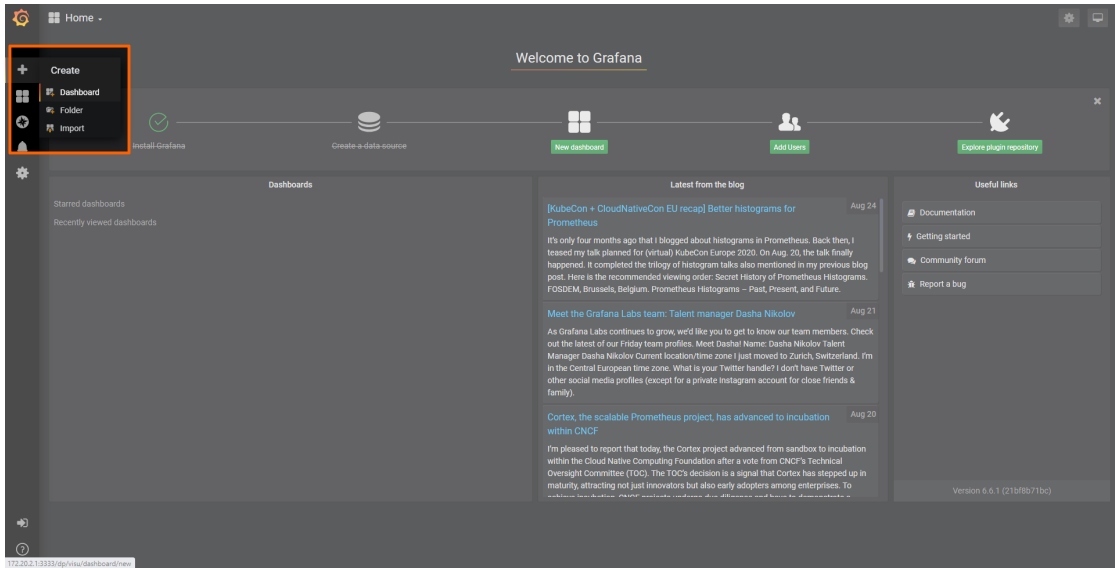
## Visualizing machineB data

1. Select **Data** in the navigation on the left. The Grafana UI will open.

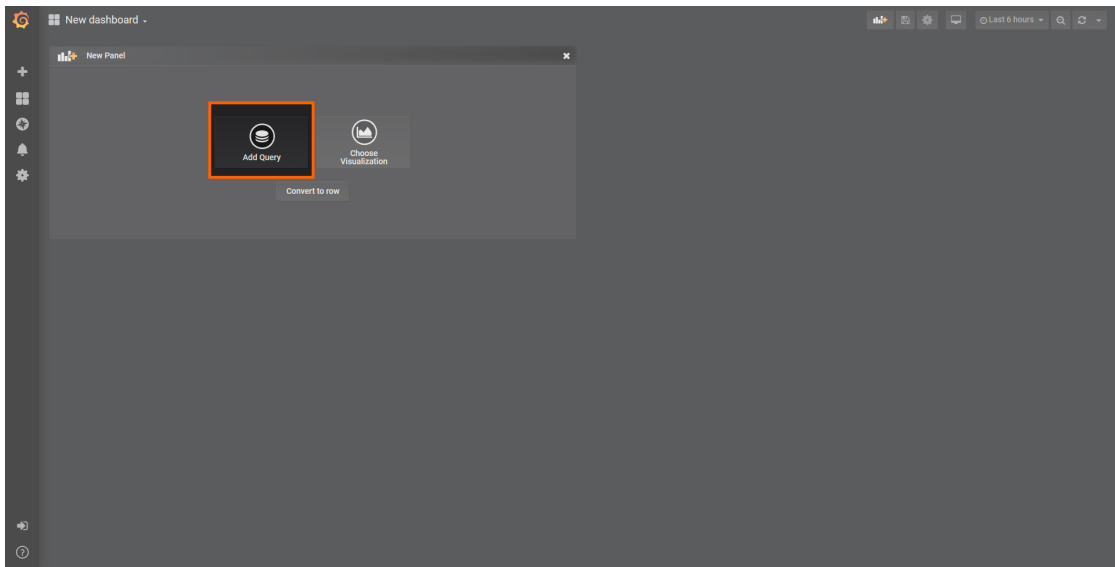
### NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

2. Select + > **Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.



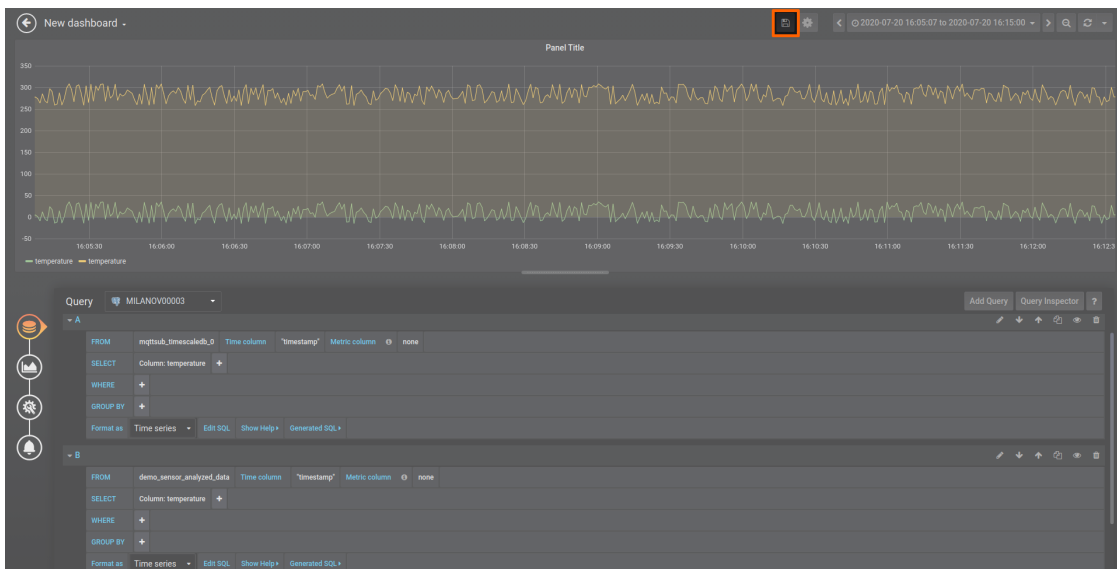
4. Select the data source from the drop-down menu. The name of the data source is the serial number of the node.



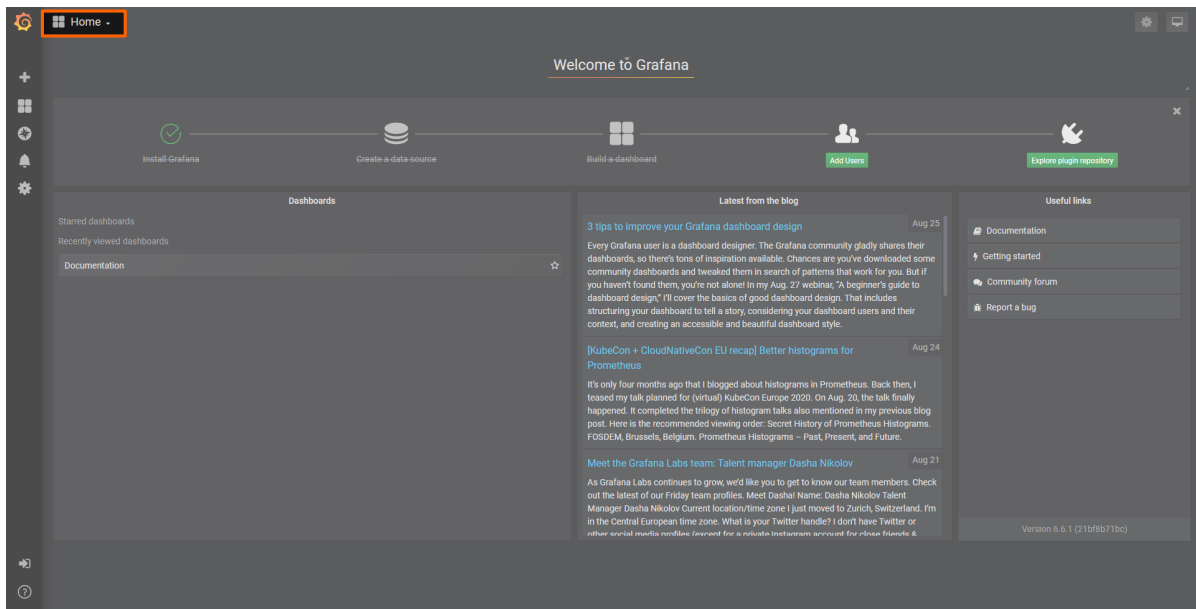
5. Fill in the following query information to add the data of Machine B:

Setting	Value
<b>FROM</b>	machineB_data
<b>SELECT</b>	Column: ".machineB.sensor1[.].humidity" Column: ".machineB.sensor1[.].temperature" Column: ".machineB.sensor2[.].humidity" Column: ".machineB.sensor2[.].temperature"
<b>Format as</b>	Time series

6. Select the save icon in the upper-right corner to save the dashboard.



The dashboard can be accessed from the Grafana home menu.



## Sending data to MS Azure IoT Hub

This example demonstrates the basic usage of the Gateway's Azure IoT Hub device output in order to show how to provide a bridge between Nerve and Microsoft's IoT infrastructure. The instructions below describe an easy-to-use way to send data from an MQTT demo sensor to an MS Azure IoT Hub device.

### NOTE

An MS Azure IoT Hub device is required before attempting the following example. Refer to the official Microsoft Azure IoT Hub documentation in the section [Create a hub using the Azure portal](#) for more information.

## Provisioning and deploying the sensor simulation and the MQTT broker

In the instructions below two Docker workloads will be provisioned and deployed:

An MQTT broker must to be deployed to the node first in order for the sensor simulation to function. The EMQX MQTT broker is used in this example that can be downloaded from the Docker Hub registry.

Afterwards the temperature and humidity sensors simulation MQTT publisher is deployed. Download the **Data Services MQTT demo sensor** found under **Example Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload for the EMQX MQTT broker by following [Provisioning a Docker workload](#). This example uses **emqx-4.1.0** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>From registry</b> and enter emqx/emqx:v4.1.0.
<b>Container name</b>	emqx
<b>Network name</b>	host

3. Provision a Docker workload for the sensor simulation by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>New environment variable</b>	Select the + icon and enter the following information: <ul style="list-style-type: none"> <li>◦ <b>Env. variable</b> MQTT_PUB_TOPIC</li> <li>◦ <b>Variable value</b> demo-sensor-topic</li> </ul>
<b>Container name</b>	ttt-mqtt-demo-sensor-1.0
<b>Network name</b>	host

4. Deploy both provisioned Docker workloads above by following [Deploying a workload](#).

## Configuring the Data Services Gateway

The example demonstrates the simplest configuration for the Azure IoT Hub Device output and omits any parameters that are not required. Since the definition of a connector has been omitted, the Gateway will generate a default connector at index 0 with the following fields:

- **name**: automatically generated name that is used in logs
- **timestampRequired**: false

Follow the instructions below to apply the Gateway configuration.

1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

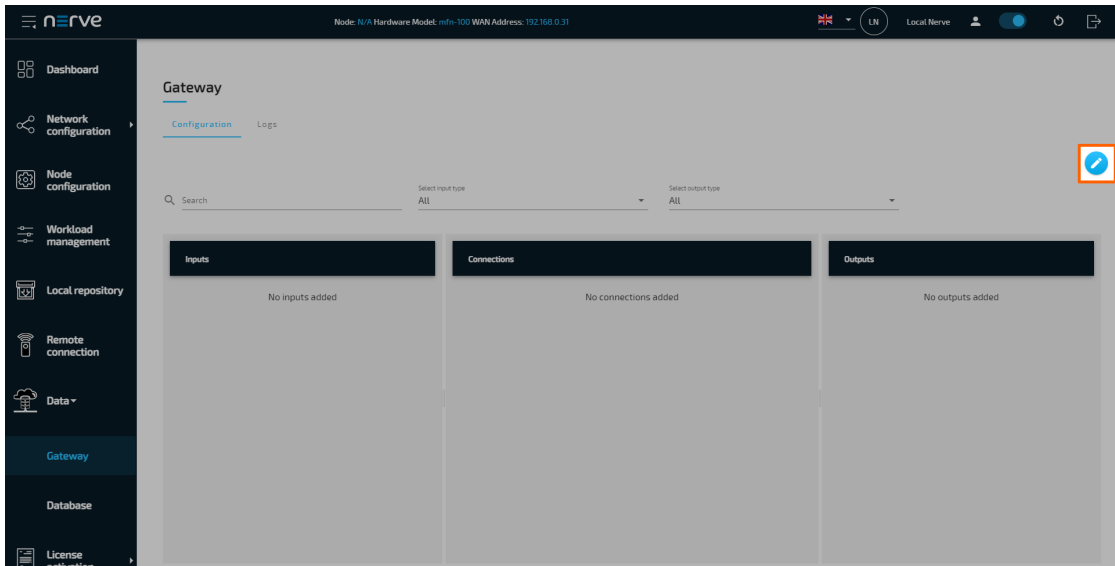
Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>

Nerve Device	Physical port	Local UI
<b>Kontron KBox A-150-APL</b>	LAN 1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.
<b>Kontron KBox A-250</b>	ETH 2	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.
<b>Maxtang AXWL10</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Supermicro SuperServer 5029C-T</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Winmate EACIL20</b>	LAN1	<wanip>:3333 To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.

3. Select **Gateway**.

4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

```
{
  "inputs": [
    {
      "type": "MQTT_SUBSCRIBER",
      "clientId": "mqtt_subscriber_0",
      "serverUrl": "tcp://localhost:1883",
      "keepAliveInterval_s": 20,
      "cleanSession": true,
      "qos": 1,
      "connectors": [
        {
          "topic": "demo-sensor-topic",
          "variables": [
            {
              "name": "temperature",
              "type": "int16"
            },
            {
              "name": "humidity",
              "type": "uint16"
            }
          ]
        }
      ]
    }
  ],
  "outputs": [
    {
      "type": "AZURE_IOT_HUB_DEVICE",
      "deviceConnectionString": "HostName=<YOUR_IOT_HUB_HOSTNAME>;DeviceId=<YOUR_IOT_HUB_DEVICE_ID>"
    }
  ],
  "connections" : [
    {
```

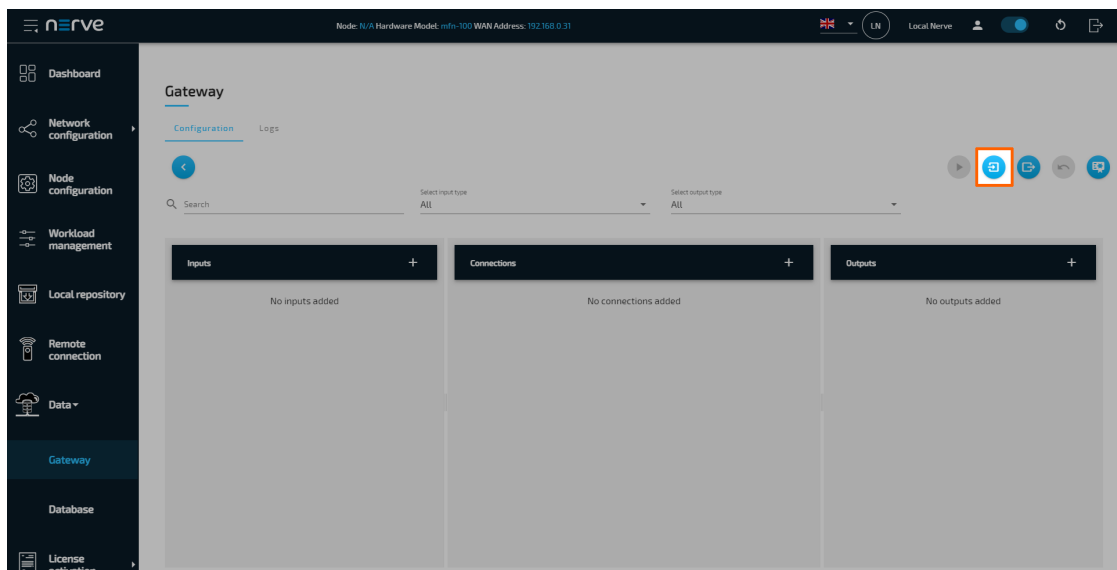


```
    "input": { "index" : 0, "connector" : 0 },
    "output": { "index" : 0, "connector" : 0 }
  }
}
```

#### NOTE

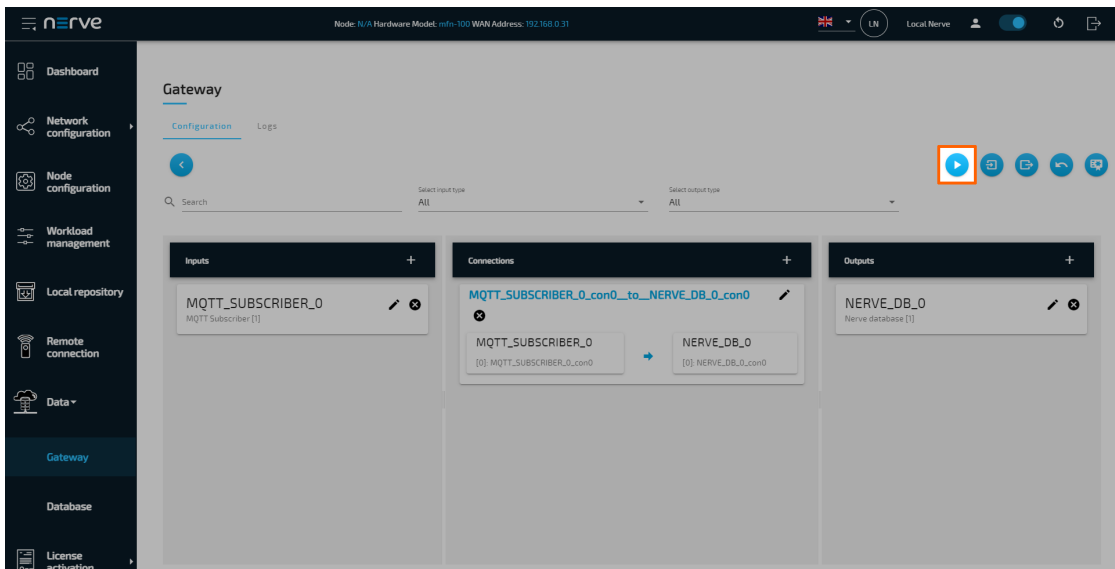
Replace the value of deviceConnectionString in the configuration above with the connection string of the device that was created before in the MS Azure IoT Hub. Make sure to copy the string that is marked as PRIMARY. Refer to [Register a new device in the IoT hub](#) in the official Microsoft Azure IoT Hub documentation for more information.

6. Select the **Import** button.



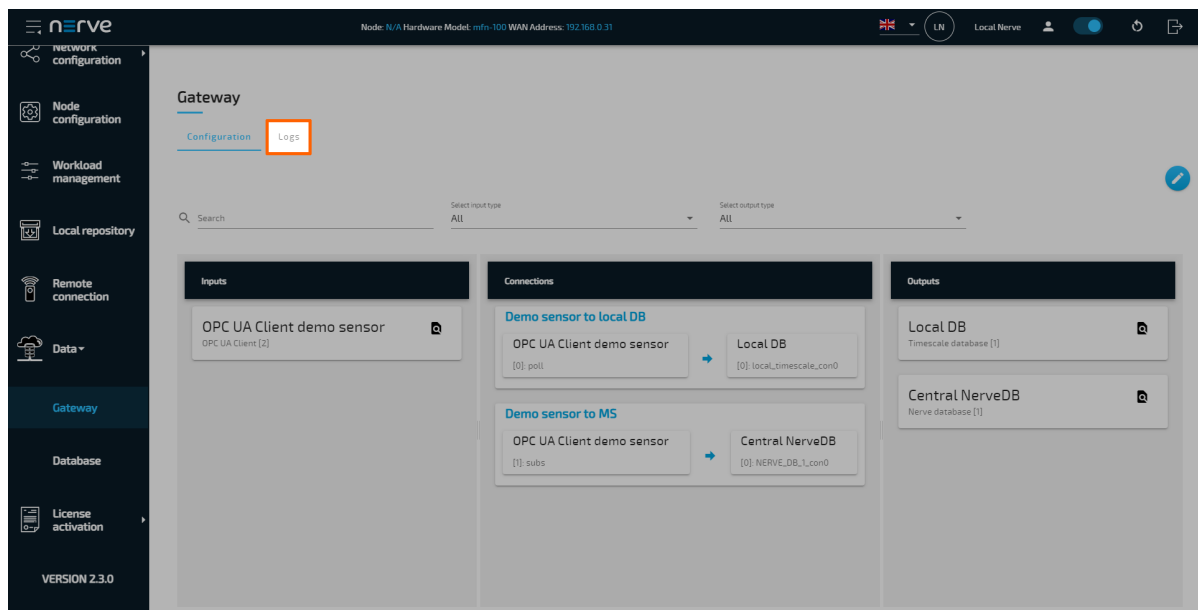
7. Add the JSON configuration file containing the code above from the file browser.

8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Verifying the data transmission

The official Microsoft Azure IoT Hub documentation describes multiple ways of verifying if data is being sent from the Nerve node to the Azure IoT Hub device. This example uses a python script utilizing the built-in event hub that is available by default for every created Azure IoT Hub device. This method is valid for every type of Azure subscription, including the free or basic tier. The script is taken from the official Azure sample repository on GitHub. To get it, clone the repository by opening the terminal and running the following command. This requires Git to be installed on the workstation:

```
git clone https://github.com/Azure-Samples/azure-iot-samples-python
```

If Git is not installed, visit the [repository page](#) and select **Code > Download ZIP** to download the repository as a ZIP file.

#### NOTE

Python version 3.7.0 or higher is required to successfully run the example.

1. Navigate to `iot-hub/Quickstarts/read-d2c-messages/` inside the cloned repository.
2. Open the `Readme.md` file.
3. Read the instructions carefully.
4. Choose either the `read_device_to_cloud_messages_async.py` or `read_device_to_cloud_messages_sync.py`. This example works with both scripts.
5. Replace the placeholder variables below in the chosen script with information from the Event Hub-compatible connection string. This string is obtained when the Azure IoT Hub device is created.

```
EVENTHUB_COMPATIBLE_ENDPOINT  
EVENTHUB_COMPATIBLE_PATH  
IOTHUB_SAS_KEY
```

6. Enter the following command to install the `azure-eventhub` package:

```
pip install azure-eventhub
```

7. Run the script that was chosen earlier with one of the following commands or executing the file:

```
python read_device_to_cloud_messages_async.py
```

or

```
python read_device_to_cloud_messages_sync.py
```

The result of the script will look similar to the screenshot below:

```
1/1 + B TiliX: python read_device_to_cloud_messages_async.py
1:pythonread_device_to_cloud_messages_async.py
1607940835028}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":10,"temperature":31}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940835963", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612105", "x-opt-offset": "b'184683726968'", "x-opt-enqueued-time": "1607940836669"}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":30,"temperature":14}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940836979", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612106", "x-opt-offset": "b'184683727368'", "x-opt-enqueued-time": "1607940837107"}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":22,"temperature":-3}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940837963", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612107", "x-opt-offset": "b'184683727752'", "x-opt-enqueued-time": "1607940838171"}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":86,"temperature":26}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940838979", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612108", "x-opt-offset": "b'184683728144'", "x-opt-enqueued-time": "1607940838999"}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":3,"temperature":3}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940839979", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612109", "x-opt-offset": "b'184683728536'", "x-opt-enqueued-time": "1607940840047"}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":30,"temperature":-12}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940840995", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612110", "x-opt-offset": "b'184683728928'", "x-opt-enqueued-time": "1607940841094"}
Received event from partition: 3.
Telemetry received: {"variables":{"humidity":24,"temperature":5}}
Properties (set by device): None
System properties (set by IoT Hub): {"iothub-connection-device-id": "b'nerve-dp-test-dev-02'", "iothub-connection-auth-method": "b'{"scope":"device","type":"sas","issuer":"iothub","acceptingIPFilterRule":null}", "iothub-enqueuedtime": "1607940842042", "iothub-message-source": "b'Telemetry'", "x-opt-sequence-number": "612111", "x-opt-offset": "b'184683729320'", "x-opt-enqueued-time": "1607940842141"}
```

# Modbus server data to InfluxDB for visualization

This example shows how to set up Modbus as an input for the Gateway. It uses a Modbus simulation server that holds temperature and humidity data inside its holding registers. This server is provided as a Docker container. Data is then written into an InfluxDB that is also provided as a Docker container and visualized at the node.

**NOTE**  
Note that InfluxDB version 1.X is supported by the Data Services. In this version, InfluxDB 2.0 is not supported.

## Provisioning and deploying the simulation server and the InfluxDB

In the instructions below two Docker workloads will be provisioned and deployed:  
A Modbus simulation server is deployed to the node that holds temperature and humidity data inside its holding registers. The simulation server is provided as a Docker container.

After that an InfluxDB that gathers data from the Modbus simulation server is deployed to the node. The InfluxDB is also provided as a Docker container.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload for the server simulation by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the server simulation that has been downloaded from the Nerve Software Center.
<b>Container name</b>	modbus-demo-server
<b>Network name</b>	host

3. Provision a Docker workload for the InfluxDB by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>From registry</b> and enter influxdb:1.8.3.
<b>Docker volumes for persistent storage</b>	influxdb:/var/lib/influxdb
<b>Container name</b>	influxdb
<b>Network name</b>	host

4. Deploy both provisioned Docker workloads above by following [Deploying a workload](#).

## Creating a database over a remote connection

A database needs to be created first that will receive data from the Modbus simulation server. In this example this is done with a remote tunnel for port 8086. The connection is then established through the Nerve Connection Manager so that an HTTP requests can be sent to the InfluxDB.

1. Configure a remote tunnel by following [Configuring a remote tunnel to a workload](#). Use the following settings:

Setting	Value
<b>Name</b>	Enter any name for the remote connection.
<b>Local acknowledgment</b>	Select <b>Yes</b> or <b>No</b> from the drop-down menu.
<b>Port on node</b>	8086
<b>Port on PC</b>	Enter a free port on the local workstation.

2. Use the remote tunnel by following [Using a remote tunnel to a workload](#). Make sure to select the InfluxDB Docker workload.
3. Open a command-line interface.

- Enter the following command to create a database in the InfluxDB. Note that this
- command requires cURL to be installed.

```
curl -POST http://localhost:8086/query --data-urlencode "q=CREATE DATABASE data"
```

## Configuring the Data Services Gateway

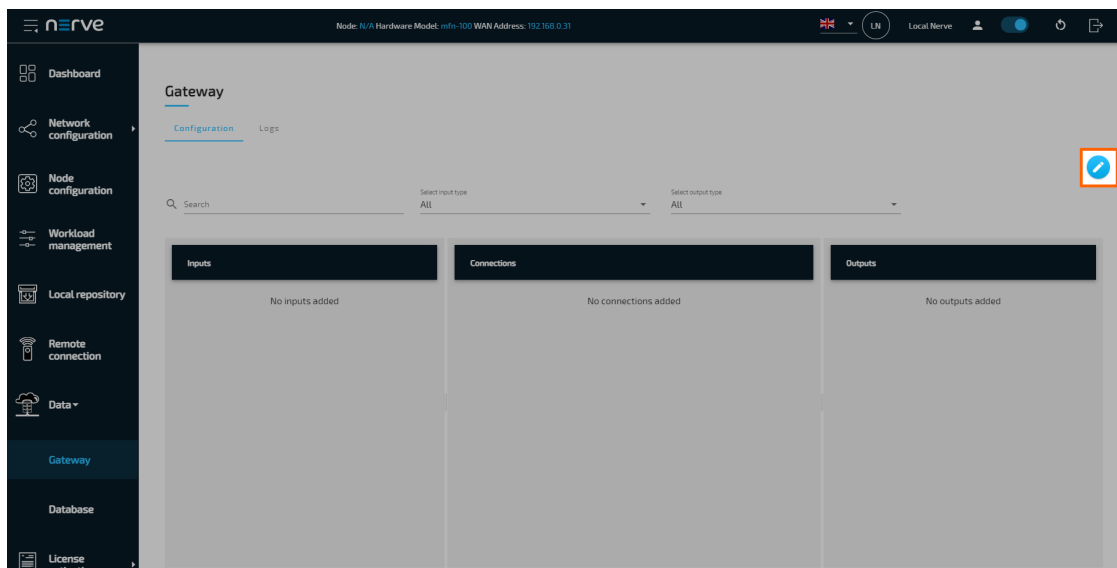
The configuration below defines the Modbus simulation as an input that delivers humidity and temperature data. The InfluxDB is defined as the output.

- Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Kontron KBox A-150-APL</b>	LAN 1	<p>&lt;wanip&gt;:3333</p> <p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide.</p> <p>&lt;wanip&gt;:3333</p>
<b>Kontron KBox A-250</b>	ETH 2	<p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide.</p> <p>&lt;wanip&gt;:3333</p>
<b>Maxtang AXWL10</b>	LAN1	<p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide.</p>
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<p><a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a></p> <p>&lt;wanip&gt;:3333</p>
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	<p>To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide.</p>
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>

Nerve Device	Physical port	Local UI
<b>Supermicro SuperServer 5029C-T</b>	LAN1	<wanip>:3333  To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide.
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>  <wanip>:3333
<b>Winmate EACIL20</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

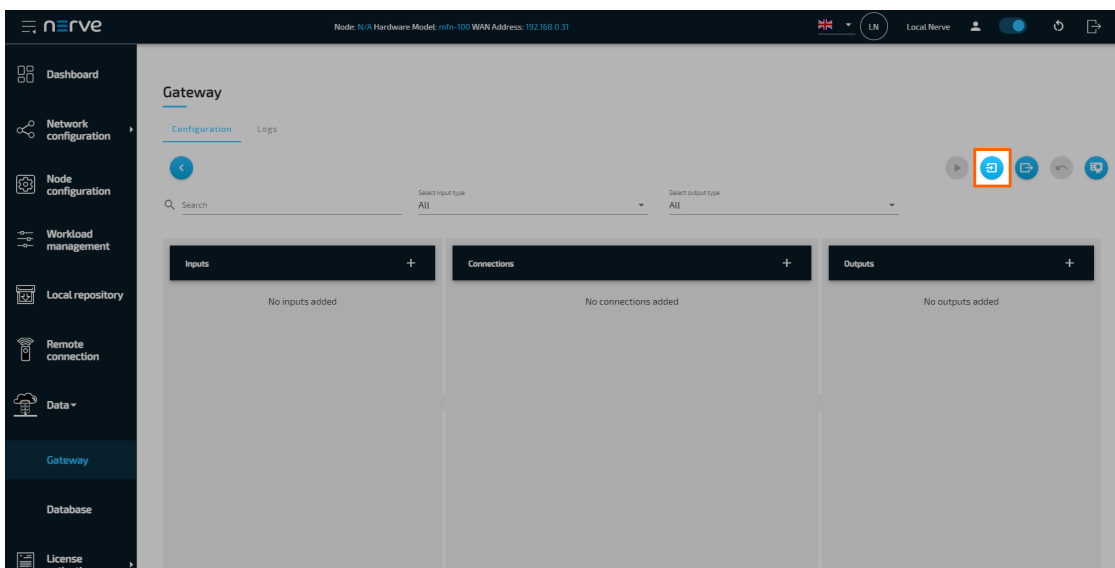
```
{
  "inputs": [
    {
      "type": "MODBUS_CLIENT",
      "name": "modbus_client",
      "serverUrl": "127.0.0.1",
      "port": 10503,
      "pollingInterval_ms": 1000,
      "connectors": [
        {
          "name": "modbus_connector_0",
```

```

        "holdingRegisters": [
            {
                "name": "temperature",
                "address": 0,
                "type": "int16"
            },
            {
                "name": "humidity",
                "address": 4,
                "type": "int16"
            }
        ]
    },
    ],
    "outputs": [
        {
            "type": "DB_INFLUX",
            "name": "influxdb_output",
            "url": "localhost",
            "port": 8086,
            "connectors": [
                {
                    "name": "modbus_data",
                    "dbName": "data"
                }
            ]
        }
    ],
    "connections" : [
        {
            "name": "connection_0",
            "input": { "index" : 0, "connector" : 0 },
            "output": { "index" : 0, "connector" : 0 }
        }
    ]
}

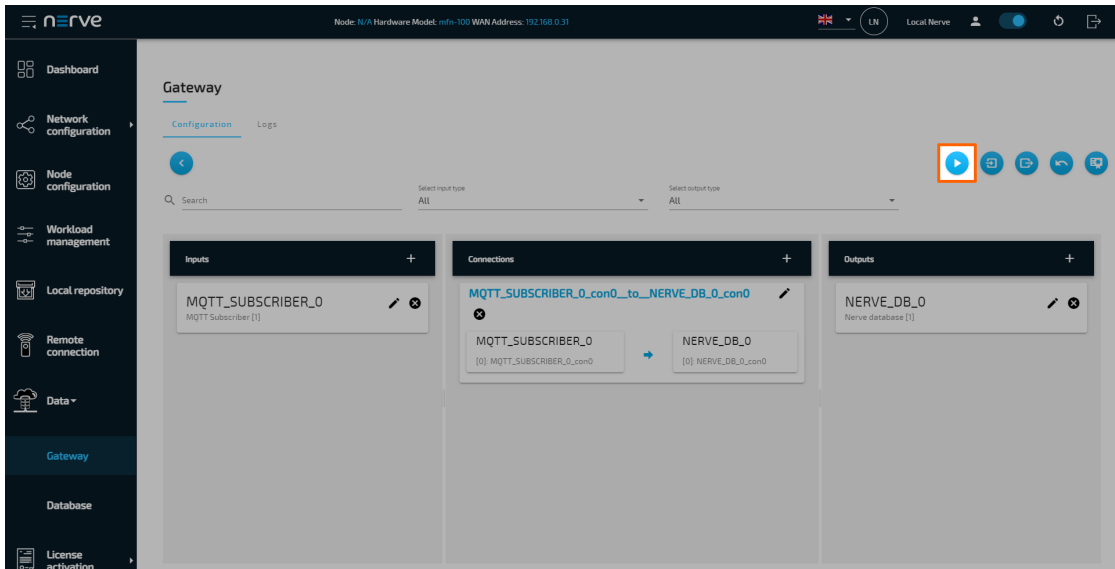
```

6. Select the **Import** button.



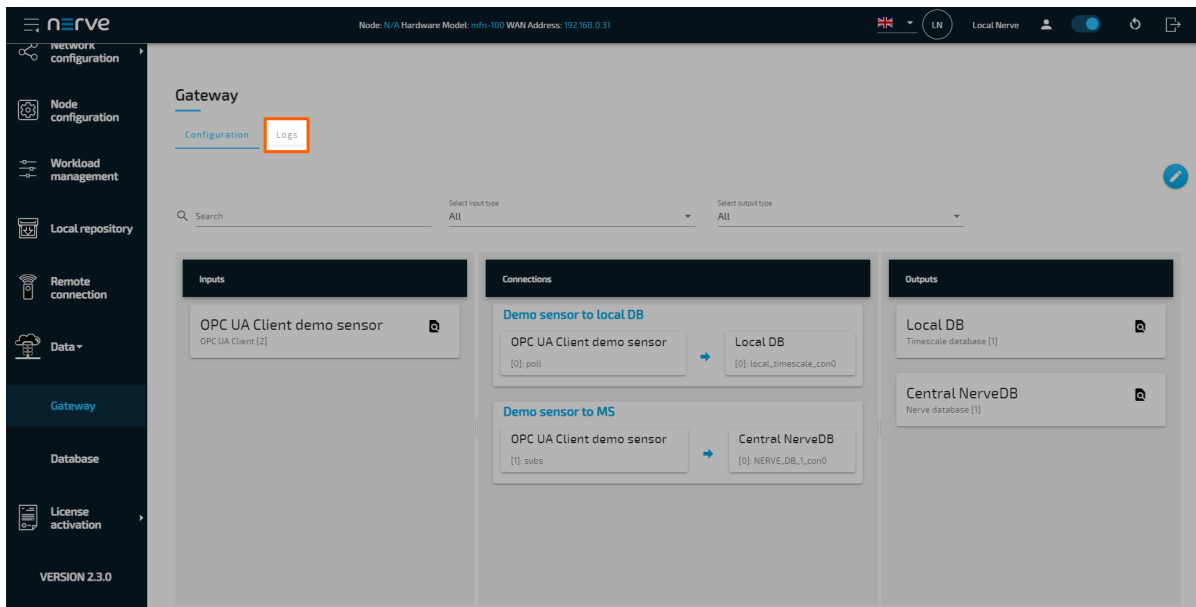


7. Add the JSON configuration file containing the code above from the file browser.
8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



## Local data visualization at the node

The Gateway collects data from the Modbus simulation server and stores it inside the InfluxDB. The data can now be visualized at the node. Open the local data visualization element through the Data Services UI on the node by selecting **Data** on the left.

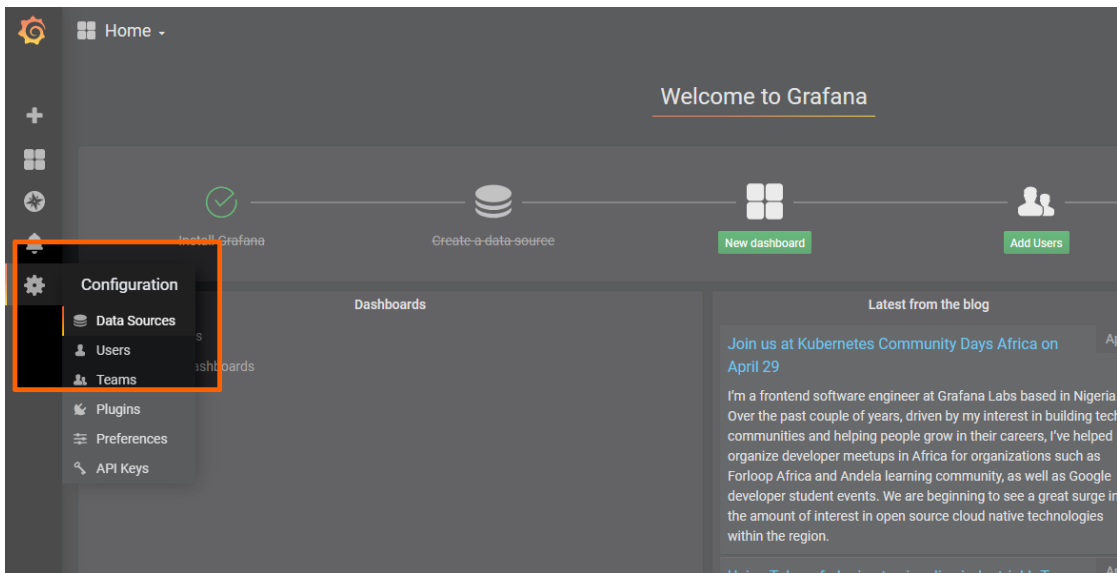
### NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

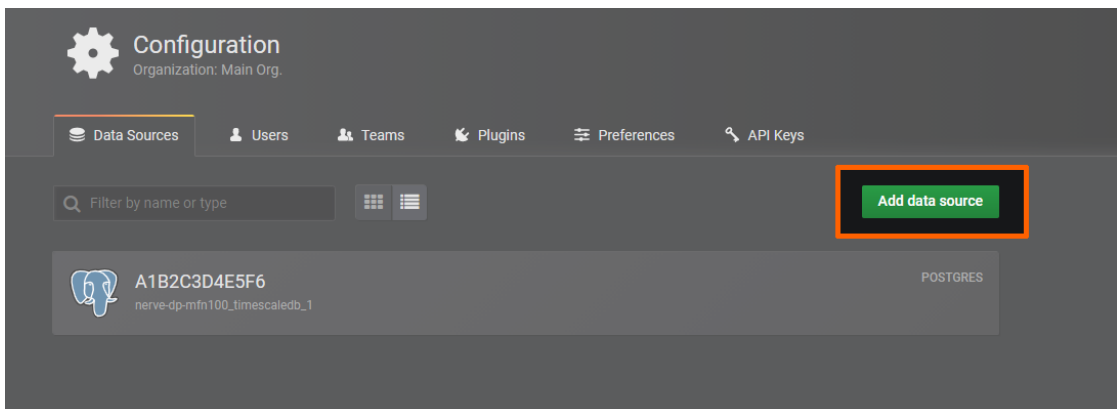
## Configuring the InfluxDB as a data source

First, a new data source needs to be created since the data is collected into an InfluxDB.

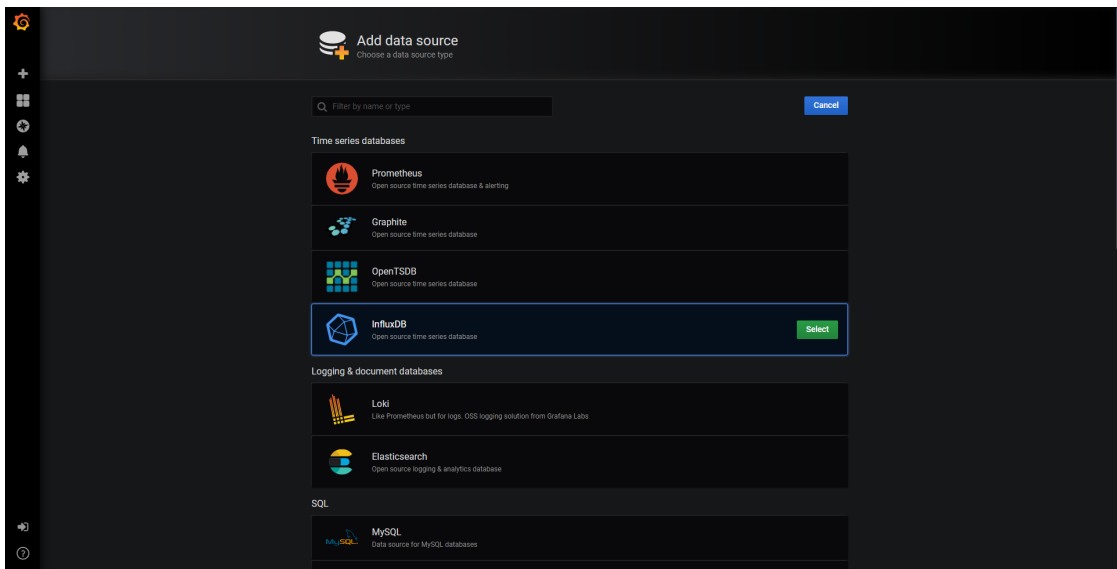
1. Select the gear symbol in the navigation on the left.
2. Select **Data Source**.



3. Select **Add data source** on the right.



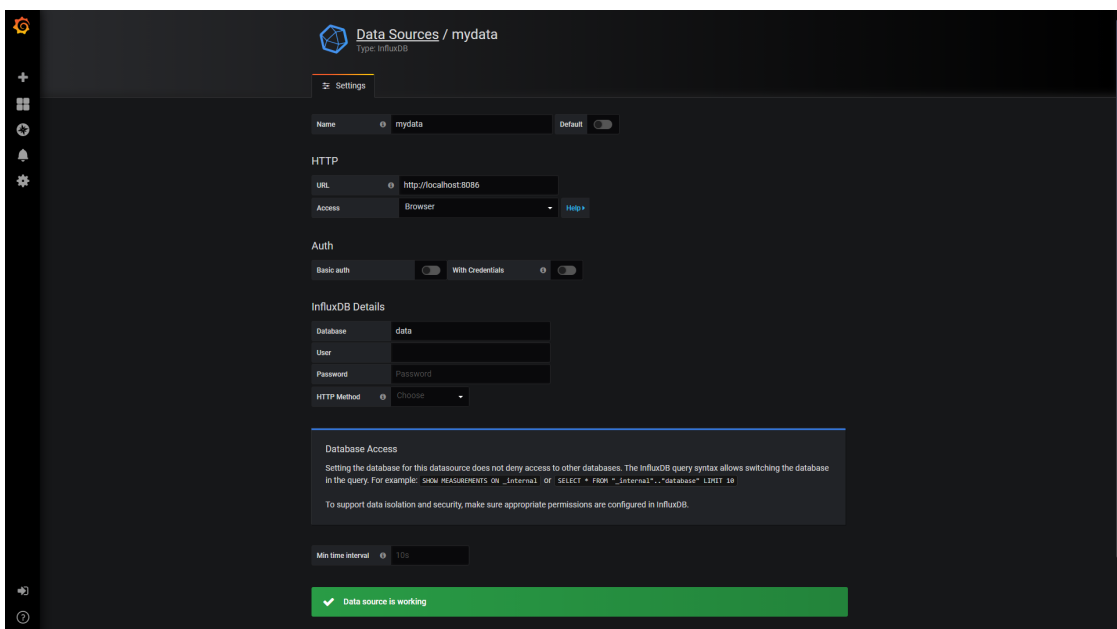
4. Select **InfluxDB**.



5. Enter the following settings:

Setting	Value
<b>Name</b>	Enter any name for the remote connection. This example uses mydata.
<b>URL</b>	http://localhost:8086
<b>Access</b>	Browser
<b>Database</b>	data

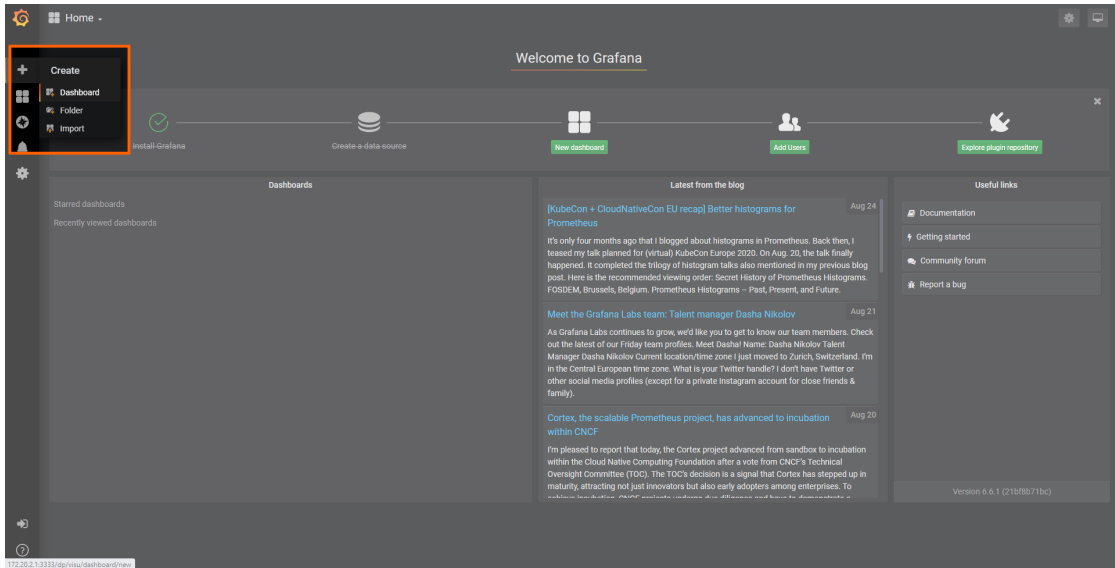
6. Select **Save & Test**. A green box will appear if the configuration was successful.



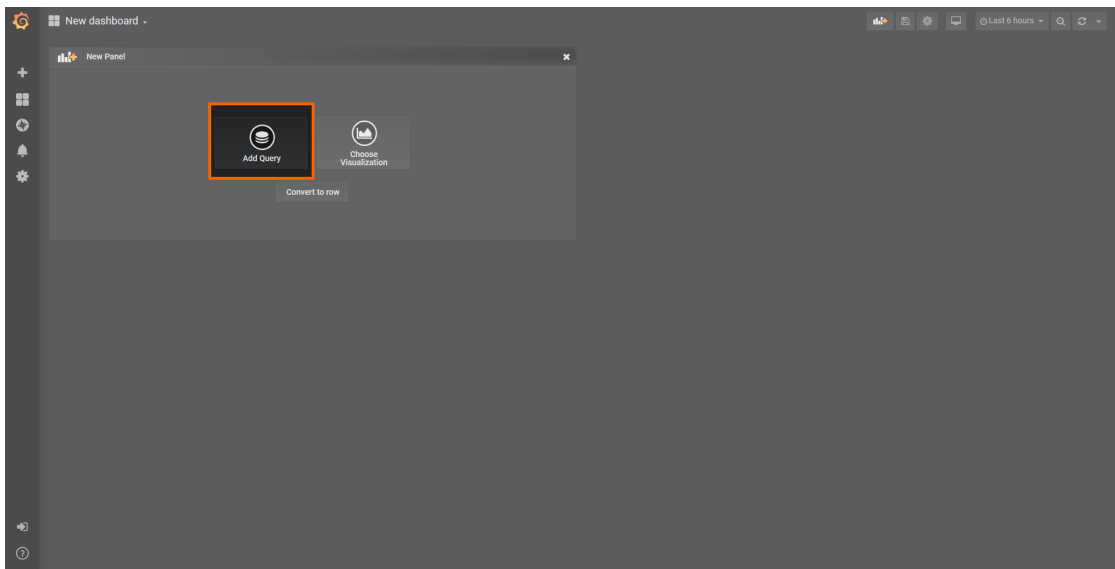
## Creating a dashboard for visualization

With the data source defined, temperature and humidity data can now be visualized in a dashboard.

1. Select **+ > Dashboard** in the navigation on the left. A box will appear.



2. Select **Add Query** in the **New Panel** box.



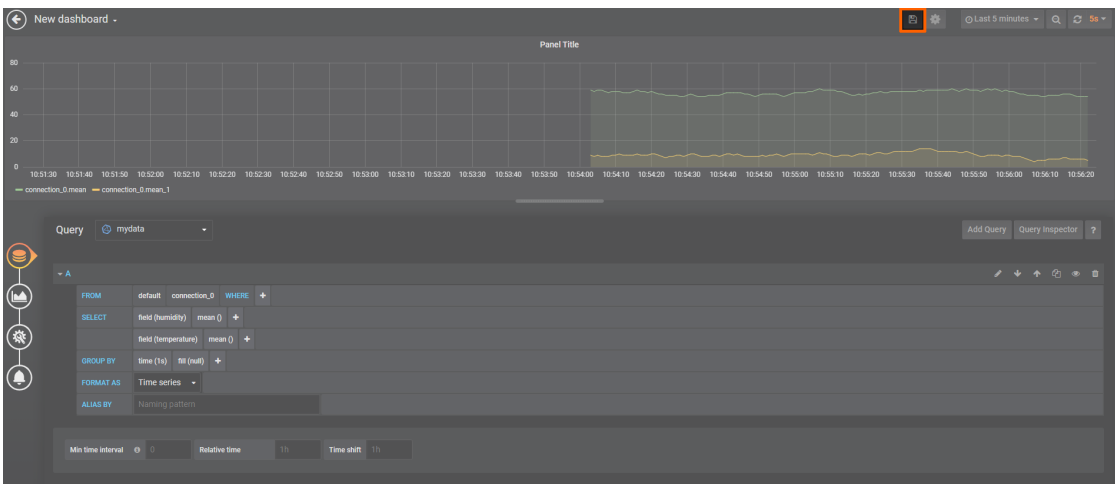
3. Select the InfluxDB data source that was created before. This example uses **mydata** as the data source.



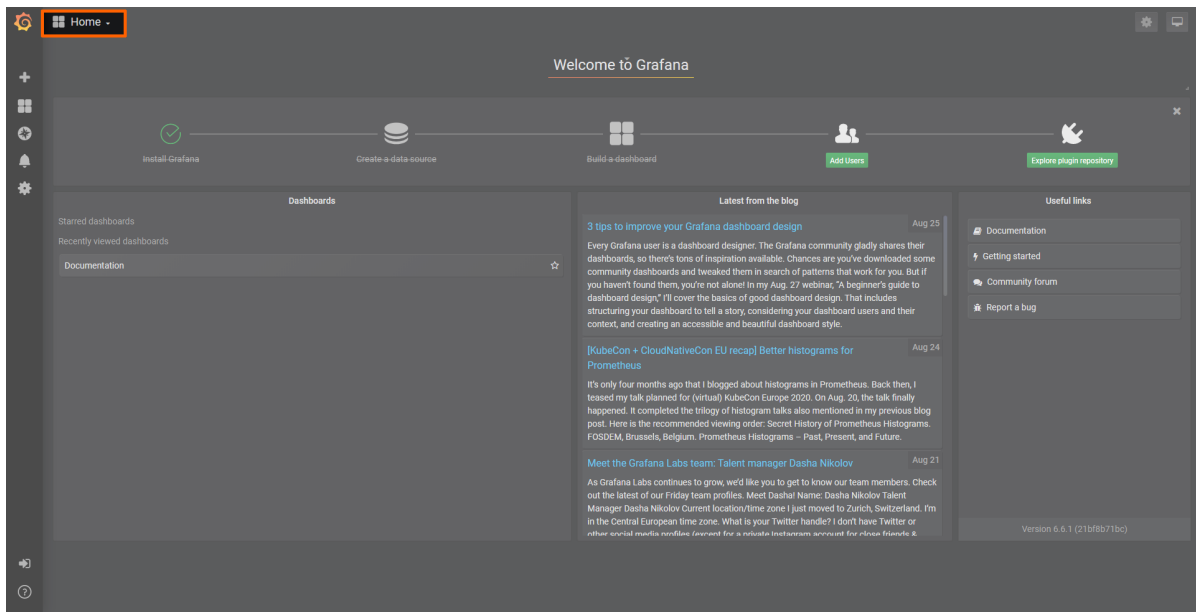
4. Fill in the following query information:

Setting	Value
<b>FROM</b>	default connection
<b>SELECT</b>	field (humidity) field (temperature)
<b>GROUP BY</b>	time (1s)
<b>Format as</b>	Time series

5. Select the save icon in the upper-right corner to save the dashboard.



The dashboard can be accessed from the Grafana home menu.



## NerveDB with data buffering

### NOTE

Note that the data buffering feature is only available when location is set to CENTRAL when using the NerveDB as an output.

Data buffering is a feature of the NerveDB output. Its purpose is to give a guarantee that data will always reach the Management System even when a node is connected to an unstable network. When this feature is enabled, data is buffered into persistent local storage first and then sent to the Management System in chunks. New chunks are not sent until it is confirmed that the previous chunk has been stored in a database in the Management System. After the confirmation, the chunk is removed from the persistent local storage. This slows down the insert rate for very fast data sources but in exchange provides a strong guarantee that no data is lost.

This example shows how to use the data buffering feature of the NerveDB output. It uses an MQTT sensor simulation as a data source. This also requires an MQTT broker. First, the sensor simulation and the broker need to be provisioned in the Management System and deployed to the node.

## Provisioning and deploying the sensor simulation and the MQTT broker

In the instructions below two Docker workloads will be provisioned and deployed:

An MQTT broker must to be deployed to the node first in order for the sensor simulation to function. The EMQX MQTT broker is used in this example that can be downloaded from the Docker Hub registry.

After that the temperature and humidity sensors simulation MQTT publisher is deployed. Download the **Data Services MQTT demo sensor** found under **Example**

**Applications** from the [Nerve Software Center](#). This is the Docker image that is required for provisioning the demo sensor as a Docker workload.

1. Log in to the Management System. Make sure that the user has the permissions to access the Data Services.
2. Provision a Docker workload for the EMQX MQTT broker by following [Provisioning a Docker workload](#). This example uses **emqx-4.1.0** as the workload name. Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>From registry</b> and enter emqx/emqx:v4.1.0.
<b>Container name</b>	emqx
<b>Network name</b>	host

3. Provision a Docker workload for the sensor simulation by following [Provisioning a Docker workload](#). Use the following workload version settings:

Setting	Value
<b>Name</b>	Enter any name for the workload version.
<b>Release name</b>	Enter any release name.
<b>DOCKER IMAGE</b>	Select <b>Upload</b> to add the Docker image of the sensor simulation that has been downloaded from the Nerve Software Center.
<b>New environment variable</b>	Select the + icon and enter the following information: <ul style="list-style-type: none"><li>◦ <b>Env. variable</b> MQTT_PUB_TOPIC</li><li>◦ <b>Variable value</b> demo-sensor-topic</li></ul>
<b>Container name</b>	ttt-mqtt-demo-sensor-1.0
<b>Network name</b>	host

4. Deploy both provisioned Docker workloads above by following [Deploying a workload](#).

## Configuring the Data Services Gateway

The input data is a subscription to temperature and humidity variables on the MQTT broker. The NerveDB is set as an output, with location set to CENTRAL in order for the data buffering feature to be usable. The input and output are linked in connections.

The Gateway configuration below shows that data buffering is enabled on a table level. The table demo-sensor-data has data buffering enabled with a 60 minutes expiration time. This means that if the data point is not confirmed by the Management System side in 60 minutes, it will be dropped regardless if it is written into the Management System database. Expiration time is set in minutes and in theory can be any positive value.

Follow the instructions below to apply the Gateway configuration.

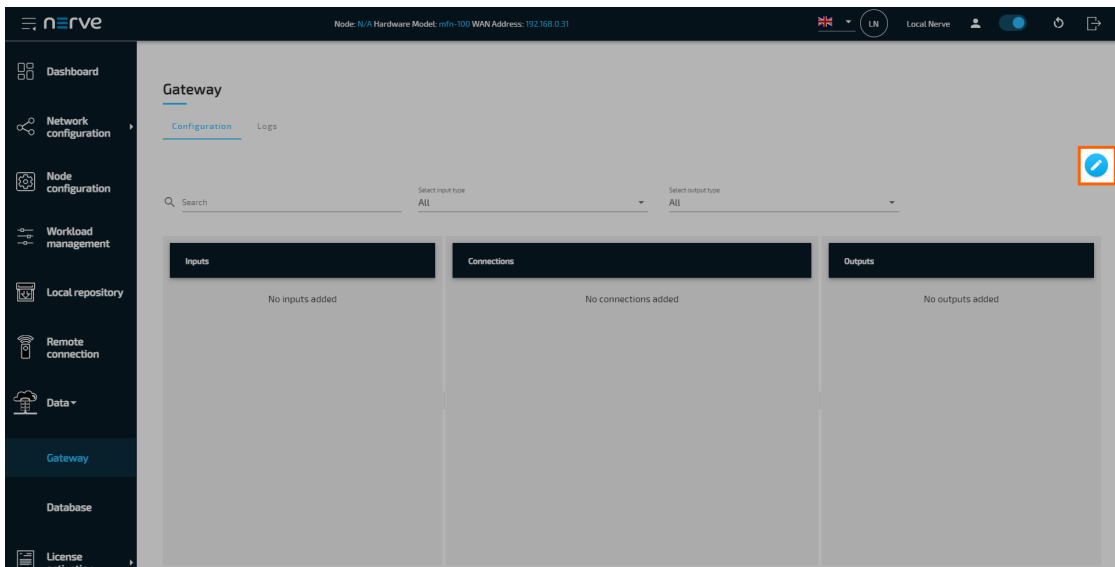
1. Access the Local UI on the node. This is Nerve Device specific. Refer to the table below for device specific links to the Local UI. The initial login credentials to the Local UI can be found in the customer profile.

Nerve Device	Physical port	Local UI
<b>MFN 100</b>	P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Kontron KBox A-150-APL</b>	LAN 1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-150-APL chapter of the device guide. <wanip>:3333
<b>Kontron KBox A-250</b>	ETH 2	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Kontron KBox A-250 chapter of the device guide. <wanip>:3333
<b>Maxtang AXWL10</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Maxtang AXWL10 chapter of the device guide. <wanip>:3333
<b>Siemens SIMATIC IPC127E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>
<b>Siemens SIMATIC IPC427E</b>	X1 P1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer E100-9AP-IA</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer E100-9AP-IA chapter of the device guide. <wanip>:3333
<b>Supermicro SuperServer 1019D-16C-FHN13TP</b>	LAN3	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a> <wanip>:3333
<b>Supermicro SuperServer 5029C-T</b>	LAN1	To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Supermicro SuperServer 5029C-T chapter of the device guide. <wanip>:3333
<b>Vecow SPC-5600-i5-8500</b>	LAN 1	<a href="http://172.20.2.1:3333">http://172.20.2.1:3333</a>



Nerve Device	Physical port	Local UI
Winmate EACIL20	LAN1	<wanip>:3333  To figure out the IP address of the WAN interface, refer to <a href="#">Finding out the IP address of the device</a> in the Winmate EACIL20 chapter of the device guide.

2. Select the arrow next to **Data** to expand the Data Services sub menus in the navigation on the left.
3. Select **Gateway**.
4. Select the **Edit configuration** icon on the right to enter editing mode.



5. Create a JSON file out of the following Gateway configuration:

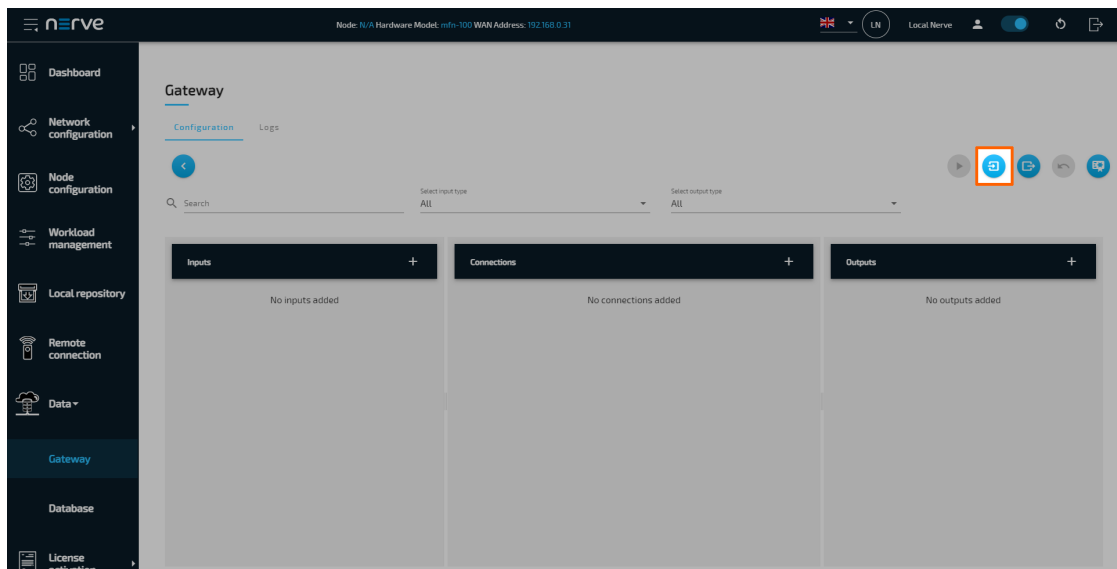
```
{
  "inputs": [
    {
      "type": "MQTT_SUBSCRIBER",
      "clientId": "mqtt_sub_dbuf-in",
      "serverUrl": "tcp://localhost:1883",
      "keepAliveInterval_s": 20,
      "cleanSession": true,
      "qos": 1,
      "connectors": [
        {
          "topic": "demo-sensor-topic",
          "variables": [
            {
              "name": "temperature",
              "type": "int16"
            },
            {
              "name": "humidity",
              "type": "uint16"
            }
          ]
        }
      ]
    }
  ]
}
```

```

    ]
  }
}
],
"outputs": [
  {
    "type": "NERVE_DB",
    "location": "CENTRAL",
    "connectors": [
      {
        "tableName": "demo-sensor-data",
        "dataBuffer": true,
        "bufferExpTime": 60
      }
    ]
  }
],
"connections" : [
  {
    "input": { "index" : 0, "connector" : 0 },
    "output": { "index" : 0, "connector" : 0 }
  }
]
}

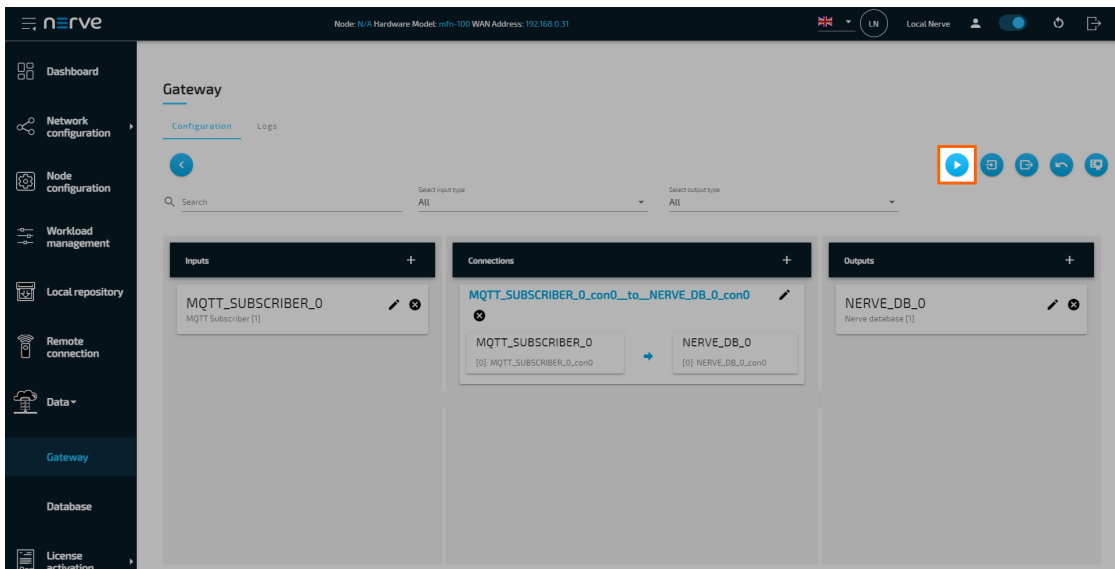
```

6. Select the **Import** button.



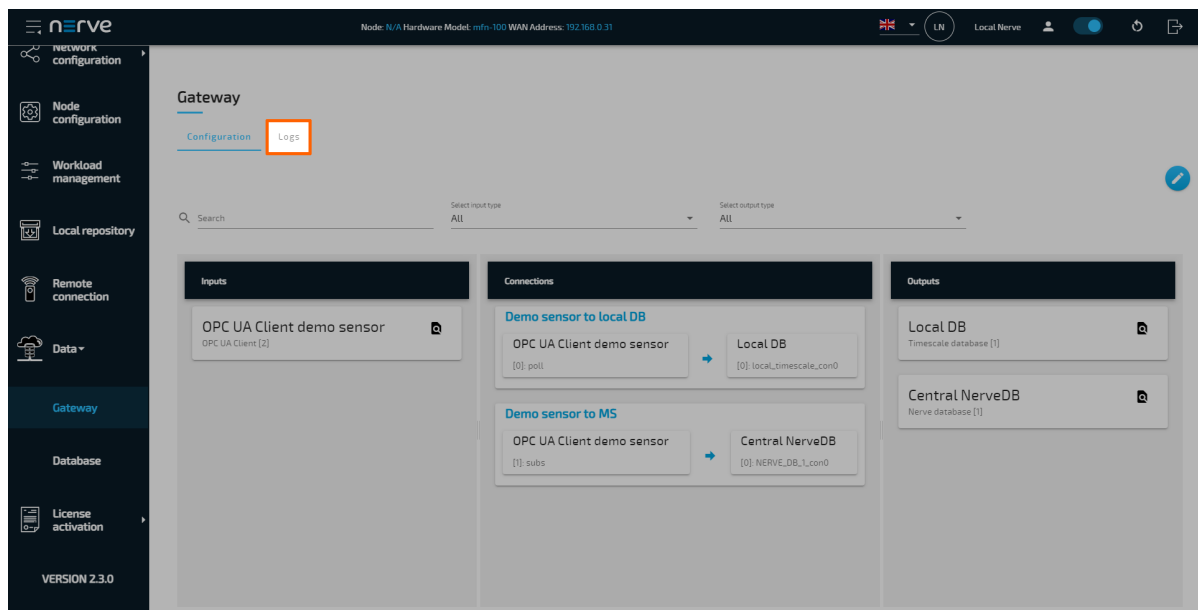
7. Add the JSON configuration file containing the code above from the file browser.

8. Select the **Deploy** button. A success message pops up in the upper-right corner.



The configuration is now deployed. The graphical configuration tool now reflects the contents of the JSON file. Exit editing mode by selecting the arrow on the left. Select the magnifying glass symbol next to the inputs and outputs to take a look at details.

Select the **Logs** tab to view the Gateway logs for more information.



To test whether data buffering is working as intended a faulty network can be simulated, for example by unplugging the network cable of the device. Since the data source is deployed as a workload, it will not be affected by an outside connection and the data should keep flowing. Once the connection has been restored, depending on the amount of data that is buffered on the device it should reach the Management System in due time. This process can be repeated multiple times.

# Central data visualization in the Management System

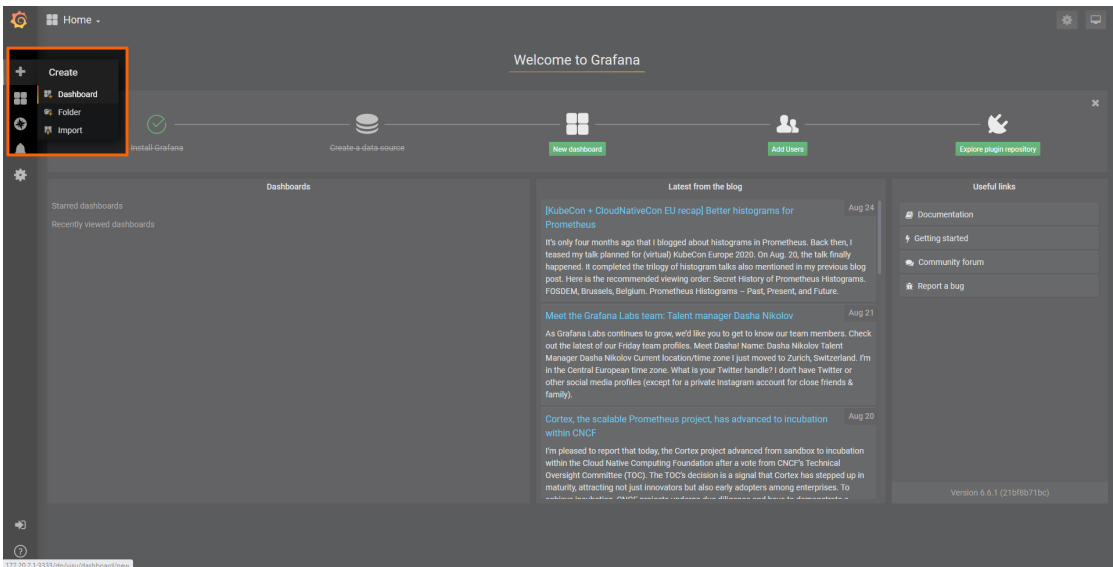
The arrival of data may be monitored in real time using Grafana. To visualize the data received by the Gateway, open the central data visualization element through the Data Services UI in the Management System.

1. Select **Data** in the navigation on the left. The Grafana UI will open.

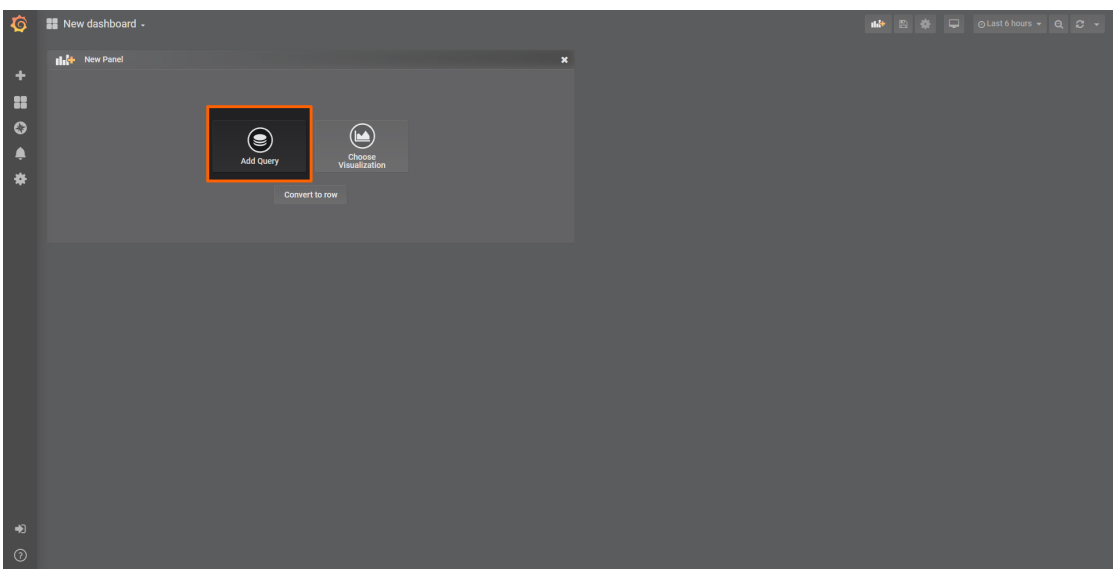
## NOTE

Note that the navigation on the left collapses when **Data** is selected. Select the burger menu in the top-left to expand the navigation again.

2. Select **+ > Dashboard** in the navigation on the left. A box will appear.



3. Select **Add Query** in the **New Panel** box.



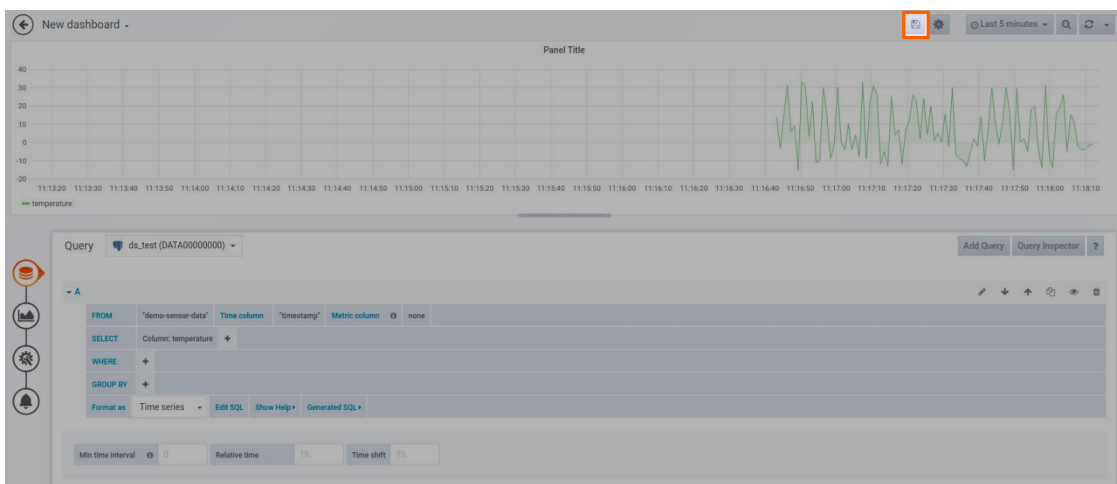
- Select the data source from the drop-down menu. The name of the data source is the serial number of the node.



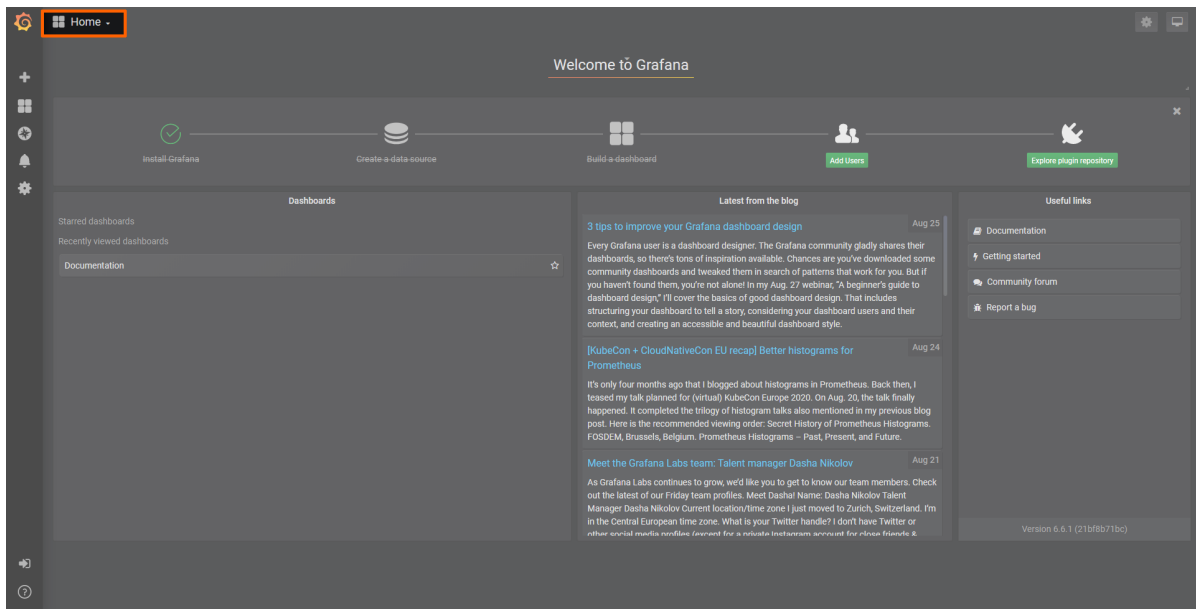
- Fill in the following query information to add the temperature data from the MQTT Subscriber:

Setting	Value
<b>FROM</b>	mqttsub_timescaledb_0
<b>SELECT</b>	Column: temperature
<b>Format as</b>	Time series

- Select the save icon in the upper-right corner to save the dashboard.



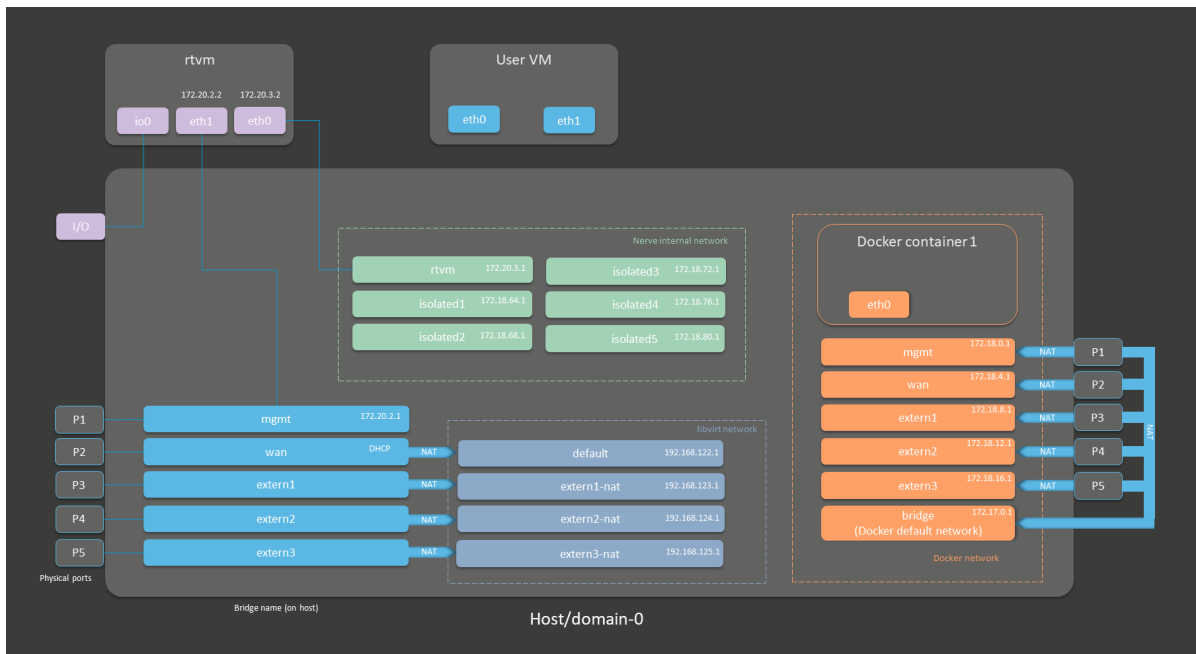
The dashboard can be accessed from the Grafana home menu.



## Node internal networking

This chapter explains how a user can connect workloads (VMs and Docker containers) to services and network ports of a node. In order to do this, it explains the internal networking concepts in detail. Most workloads will need to be connected to a network as networking is the main form of communication for workloads. They either want to connect to external servers or they are servers themselves, in which case they need to be made visible for their communication partners. The Nerve networking system enables both use cases.

The image below shows an example node consisting of the **host/domain-0** and the real-time VM running the CODESYS runtime (labeled **rtvm**). To further clarify the networking example it also has one Virtual Machine workload and one Docker workload deployed. The virtual machine is depicted outside of the host and the Docker container is depicted in the Docker network inside of the host. For the sake of explanation, however, the workloads are not yet connected. This is done in the examples further below.



Interface color	Description
Gray with blue frame	Physical network ports
Light blue	Linux network bridges displayed with their bridge names on the host
Dark blue	libvirt network interfaces for Virtual Machine workloads
Orange	Docker network bridges for Docker workloads
Green	Nerve internal network bridges shared between virtual machines, Docker containers and the host. They are not connected to any external, physical interface.
Purple	Interfaces related to the RTVM

The physical ports **P1** to **P5** and **I/O** of the Nerve Device (the MFN 100 in this case) are displayed on the left, touching the large dark rectangle that represents the host. The light blue interfaces connected to them inside the host are Linux bridged interfaces displayed with their names on the host. Highlighted by a dark blue dashed frame is the libvirt network with NAT interfaces. The system is set up so that the network bridges can be reached by connecting through the physical ports and that virtual machines can reach outside of the system through the NAT interfaces.

Highlighted by an orange dashed frame is the Docker network including the default Docker network (the orange interface labeled **bridge**), as well as Docker network equivalents of the Linux bridged interfaces. For the sake of easier representation, the physical ports **P1** to **P5** are duplicated to the right of the Docker network, again touching the host. This is done to show that Docker network interfaces can also be reached directly by connecting to the physical ports while also making sure to show that the libvirt network and the Docker network are separate from each other.

The Nerve internal network is highlighted by the green frame. It consists of the **rtvm** interface and the isolated interfaces. The isolated interfaces are designed for communication inside of the system, enabling communication between workloads. Using the isolated interfaces, communication can be established between Docker workloads,

between Virtual Machine workloads, or between Docker and Virtual Machine workloads. The **rtvm** interface is designed for communication with the RTVM and can be used to enable communication of the RTVM with Docker and Virtual Machine workloads.

All interfaces colored in purple are related to the RTVM. Interfaces labeled **eth** are symbolic representations of interfaces that are used by virtual machines and Docker containers for communication with the Nerve system. The actual used interfaces depend on the Docker container or virtual machine.

Connections are displayed in three ways. Blue lines are connections that are predefined by the system. Blue arrows are used between bridged interfaces and the libvirt network to indicate NAT. Further down below, green lines are used as example connections that can be defined by the user.

As mentioned above, the image above represents the MFN 100. Refer to the [device guide](#) for information on the Nerve Device as the physical ports and the connection to their respective interfaces differ.

See the tables below for more information on the interfaces, their usage and their IP ranges. The table is structured by workload type and shows which interface can be used for each workload type.

## Networks for Virtual Machine workloads

---

To connect to any of the bridged interfaces below, select **Bridged** from the drop-down menu and enter the name of the interface when provisioning a Virtual Machine workload.

### Bridged Interfaces

- **mgmt**  
This interface is intended for the configuration of the system. There is no DHCP server present on this interface. To connect to it, configure the network adapter of the workstation. The IP address has to be in the range from 172.20.2.5 to 172.20.2.254 with a 255.255.255.0 subnet mask. The **mgmt** interface is connected to the **eth1** interface of the RTVM.
- **wan**  
This interface is designated for internet connection and configured as DHCP client by default.
- **extern1, extern2, extern3**  
These interfaces can be used for customer specific external connections.



## NAT Interfaces

To connect to any of the NAT interfaces below, select **NAT** from the drop-down menu and enter the name of the interface when provisioning a Virtual Machine workload.

If a deployed virtual machine uses one of the predefined NAT interfaces, the IP address of the respective interface is assigned by a DHCP server with a subnet mask of 255.255.255.0. The DHCP pool contains the lower half of the respective address space, e.g. 192.168.122.2 to 192.168.122.128. The only exception is the **default** network.

- **default**

This network is the NAT interface of the **wan** bridged interface with an IP address in the range from 192.168.122.2 to 192.168.122.254.

- **extern1-nat, extern2-nat, extern3-nat**

These networks are the NAT interfaces of the **extern1**, **extern2** and **extern3** bridged interfaces. The DHCP IP address ranges of these interfaces are the following:

- **extern1-nat**

192.168.123.2 to 192.168.123.128

- **extern2-nat**

192.168.124.2 to 192.168.124.128

- **extern3-nat**

192.168.125.2 to 192.168.125.128

---

To connect to any of the interfaces in the Nerve internal network below, select **Bridged** from the drop-down menu and enter the name of the interface when provisioning a Virtual Machine workload.

### Isolated interfaces

Isolated interfaces can be used to allow Virtual Machine workloads to communicate with Docker workloads and other Virtual Machine workloads. These networks cannot communicate outside of the system. The IP addresses of these interfaces are assigned by DHCP servers in the following ranges:

## Nerve internal network

- **isolated1**  
172.18.64.2 to 172.18.67.254
- **isolated2**  
172.18.68.2 to 172.18.71.254
- **isolated3**  
172.18.72.2 to 172.18.75.254
- **isolated4**  
172.18.76.2 to 172.18.79.254
- **isolated5**  
172.18.80.2 to 172.18.83.254

Note that when creating a VM on a Nerve Device according to [Provisioning a Virtual Machine workload](#), these networks need to be specified as br-isolated1.

### rtvm

Use this interface for communication with the RTVM and to establish connections between Virtual Machine workloads and the RTVM. The IP addresses are assigned in the range from 172.20.3.3 to 172.20.3.254. 172.20.3.2 is reserved for **eth0** in the RTVM. Note that when creating a VM on a Nerve Device according to [Provisioning a Virtual Machine workload](#), this network needs to be specified as br-rtvm.

## Networks for Docker workloads

---

Since interfaces in the Docker network are behind NAT by the default, there is no drop-down menu to select **Bridged** or **NAT** when provisioning a Docker workload. Enter the network name of one of the networks below to connect a Docker workload to an interface.

### Docker network

- **bridge**  
This is the default Docker network designated for communication with Docker. Interfaces of Docker containers that are connected to the gateway receive IP addresses in the default Docker network, ranging from 172.17.0.2 to 172.17.0.244. By default, this interface can be reached through any physical port (excluding **io0**). Note that in this version the **bridge** network is automatically assigned to a Docker workload when any Docker network (**mgmt**, **wan** or **extern1** to **extern3**) is assigned.
- **mgmt, wan, extern1, extern2, extern3**  
These interfaces are the Docker network equivalents of the bridged interfaces. They can be reached through the physical ports **P1** to **P5**. The DHCP IP address ranges of these interfaces are the following:
  - **mgmt**  
172.18.0.2 to 172.18.3.254
  - **wan**  
172.18.4.2 to 172.18.7.254
  - **extern1**  
172.18.8.2 to 172.18.11.254
  - **extern2**  
172.18.12.2 to 172.18.15.254
  - **extern3**  
172.18.16.2 to 172.18.19.254

Since there is no drop-down menu to select **Bridged** or **NAT** when provisioning a Docker workload, enter the network name of one of the networks below to connect a Docker workload to an interface.

#### Isolated interfaces

Isolated interfaces can be used to allow Docker workloads to communicate with Virtual Machine workloads and other Docker workloads. These networks cannot communicate outside of the system. The IP addresses of these interfaces are assigned by DHCP servers in the following ranges:

#### Nerve internal network

- **isolated1**  
172.18.64.2 to 172.18.67.254
- **isolated2**  
172.18.68.2 to 172.18.71.254
- **isolated3**  
172.18.72.2 to 172.18.75.254
- **isolated4**  
172.18.76.2 to 172.18.79.254
- **isolated5**  
172.18.80.2 to 172.18.83.254

#### rtvm

Use this interface for communication with the RTVM and to establish connections between Docker workloads and the RTVM. The IP addresses are assigned in the range from 172.20.3.3 to 172.20.3.254. 172.20.3.2 is reserved for **eth0** in the RTVM.

## Physical ports, other interfaces and connections

The physical ports are device dependent. They are included here for clarification of the image above. The MFN 100 is used as an example. Refer to the [device guide](#) for information on the specific hardware model of the Nerve Device.

#### Physical ports

- **P1 to P5**  
Ethernet ports of the MFN 100. Note that P5 is an SFP port.
- **I/O**  
While also an Ethernet port, this port is reserved for communication of the RTVM with the fieldbus.

#### Other interfaces

##### eth

These interfaces are symbolic representations of interfaces that are used by virtual machines and Docker containers for communication with the Nerve system. The actual interfaces used depend on the Docker container or virtual machine.

The only exceptions are **eth0** and **eth1** for the RTVM as they are always defined with these names.

##### io0

This interface is defined for communication between the I/O port and the RTVM.

---

<b>Connections</b>	<b>Blue lines</b> Blue lines signify connections that are predefined by the Nerve system.
	<b>Green lines</b> Green lines are example connections that could be defined by a user to connect virtual machines or Docker containers to the network.
	<b>Blue arrows</b> Blue arrows indicate a NAT between bridged interfaces and the libvirt network.

The following sections are conceptual explanations. Workloads are attached to internal networks during the provisioning process. Refer to the provisioning chapters ([Virtual Machine workloads](#) and [Docker workloads](#)) in the user guide on how to provision workloads.

## Addressing workloads by DNS name

Docker containers and virtual machines can be addressed by their DNS name, without the need of knowing the exact IP address. The DNS name of a Docker container and a virtual machine consists of the hostname or container name and the name of the network they are attached to. This behavior is explained further within the examples below, highlighting the DNS name that can be used where applicable.

### Addressing a Docker workload by DNS name

The DNS name of a Docker workload is defined in the provisioning process of the workload. Following the [Provisioning a Docker workload](#) chapter, the DNS name is determined by the **Container name** and **Network name** settings formatted as <containername>.<networkname>. As an example, the following screenshot shows these settings of a Docker workload:

Container restart policy ▼

---

Container name \*


**docker-container**

---

Network name \*

**isolated1**

---


 Docker network

According to these settings, this Docker container can be reached under `docker-container.isolated1` after being deployed to a node.


### Addressing a Virtual Machine workload by DNS name


For Virtual Machine workloads the same principle applies, but the hostname part of the DNS name does not come from the workload provisioning settings in the Management System. The hostname is defined within the virtual machine, usually at creation time. As an example, if a Windows VM is generated on a Nerve Device, the hostname that is defined during the installation process of the Windows OS is required for addressing the Virtual machine per DNS name, e.g. `DOCUMENTATION-PC`.

The network name is determined in the Management System by the **New interface** setting. As an example, the following shows this setting of a Virtual Machine workload:

 PCI passthrough

New interface \*

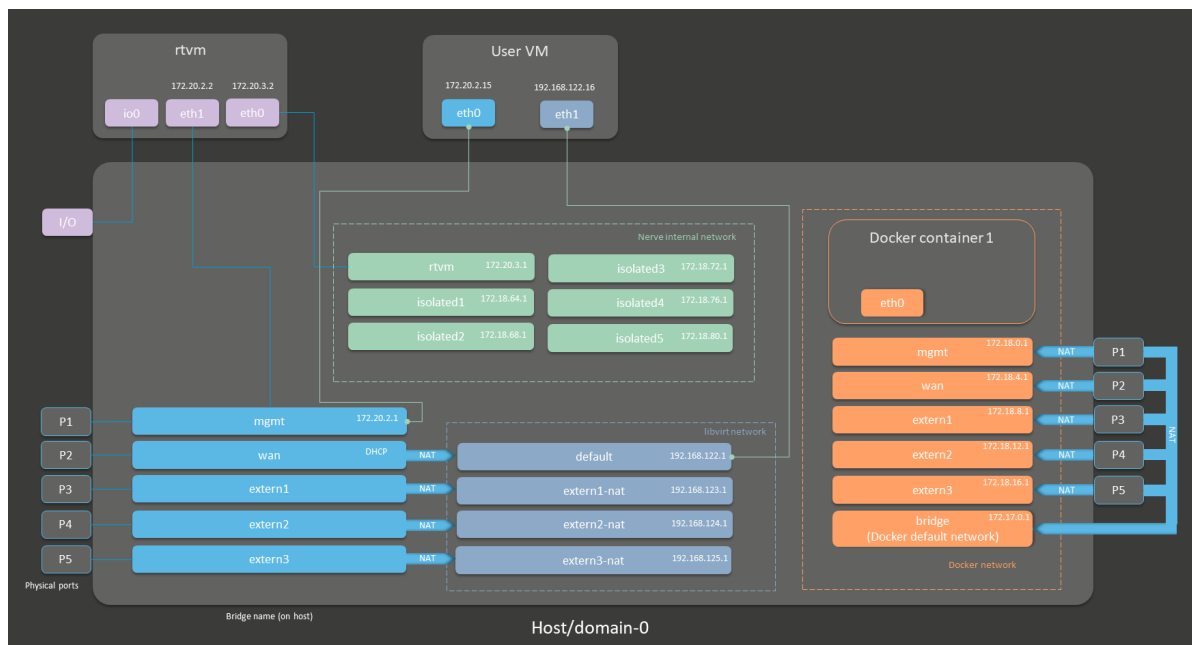
**Bridged** ▼ **isolated1** 

 New interface

According to the machine name or hostname setting of the virtual machine and the network setting in the Management System, this virtual machine can be reached under `DOCUMENTATION-PC.isolated1`

## Attaching virtual machines to a network

Virtual machine networking is comparable to installing a network card in the virtual machine and attaching it to the network with the network name given in the network drawing. For this example, there are two "network cards" installed in a user deployed virtual machine. They are located in the **User VM** and are labeled **eth0** and **eth1** in this example. Green lines indicate a user established connection.



There are two connections established here:

- **eth0** of the **User VM** is connected to the **mgmt** bridged interface so that the virtual machine can be reached through physical port **P1**. If the **User VM** has an SSH server set up, a service technician could gain access through **P1** of the MFN 100. Technically, the **User VM** could communicate with the RTVM through the **mgmt** network as well. However, it is recommended to use the Nerve internal network **rtvm** for communication with the RTVM.
- **eth1** of the **User VM** is connected to the **default** NAT interface for an internet connection protected by NAT on **P2** of the Nerve Device.

Both interfaces have IP addresses in the designated ranges. 172.20.2.15 for **eth0** was manually configured in the virtual machine and 192.168.122.16 for **eth1** was assigned by the DHCP server.

### Settings example

To achieve the functionality above, configure the interfaces of the Virtual Machine workload the following way during the provisioning process in the Management System:

## VIRTUAL MACHINE SPECIFIC INFO


Number of virtual CPUs \*


2

System memory to reserve \*

4

GB ▾

 New data disk

 PCI passthrough

New interface \*

Bridged ▾

mgmt





New interface \*

NAT ▾

default



 Add ports

 New interface

## Communication of a virtual machine with the RTVM

A virtual machine can communicate with the RTVM by connecting an interface to the bridged interface **rtvm** in the Nerve internal network. In the example below this is done with the interface **eth0** of the **User VM** that has the IP address 172.20.3.15. The IP address was manually configured.





#### VIRTUAL MACHINE SPECIFIC INFO

Number of virtual CPUs \*

2

Limit memory to \*

4

GB



New data disk



PCI passthrough

New interface \*

Bridged



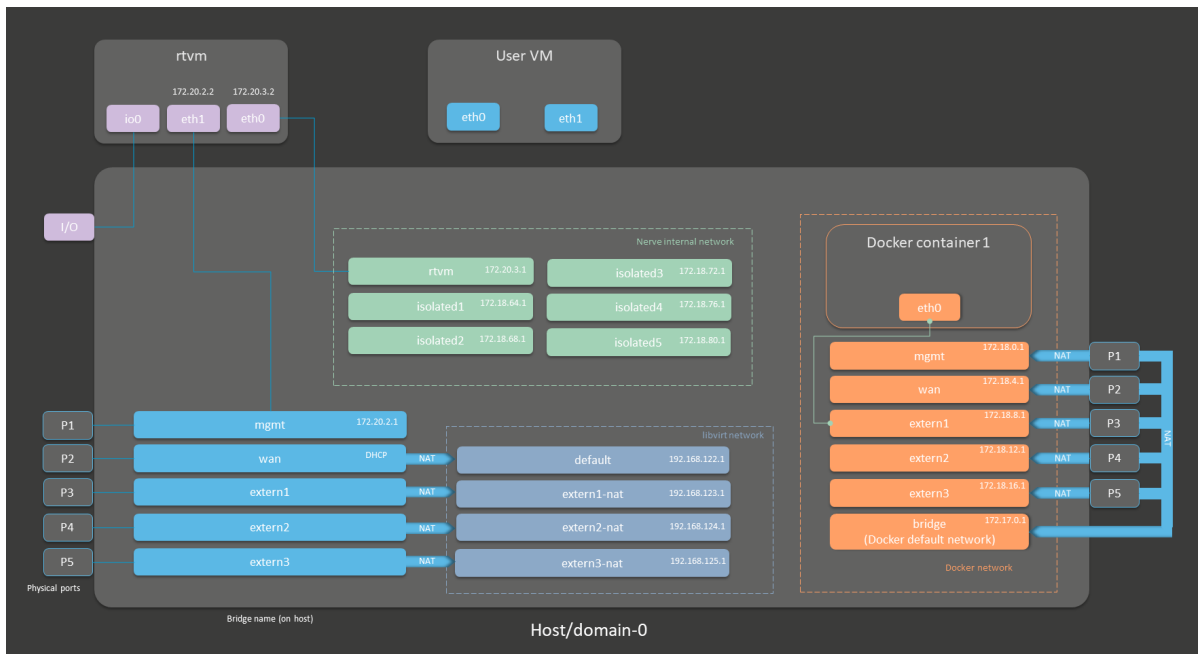
rtvm



New interface

## Communication of a Docker container with external devices

Docker containers can be attached to the Docker default network or respective Docker network interfaces to access other parts of the system or to communicate outside of the system. The Docker default network is called **bridge** and has the IP address 172.17.0.1 assigned. This interface is available on all physical ports (here **P1** to **P5**). Therefore, when a Docker is connected to the Docker default network, the forwarded ports, as defined during workload provisioning in the port mapping section, are exposed on all physical interfaces. For this example, the Docker container will be connected to the **extern1** interface. In order to make a Docker port accessible from outside, specify the port and protocol during workload provisioning in the port mapping section. The chosen network, here **extern1**, defines on which physical port the exported Docker port can be accessed. The Docker container is connected to the **extern1** interface in the Docker network, which makes the exported port available at physical port **P3** to external devices.



The Docker container can be reached from the outside by connecting to the specified, exported port on the IP address of the **P3** interface from an IP address in the range from 172.18.8.2 to 172.18.8.254.

## Settings example

To achieve the functionality above, configure the Docker network name the following way during the provisioning process in the Management System. The **Container name** and the port mapping settings are placeholders chosen for this example:

## DOCKER SPECIFIC INFO

Protocol\*    Host Port\*    Container Port\*  
TCP    9999    : 9999



New port



New environment variable

## Docker volumes for persistent storage

Limit memory to    MB

CPU resource in percentage

Container restart policy

Container name\*  
docker-container-1

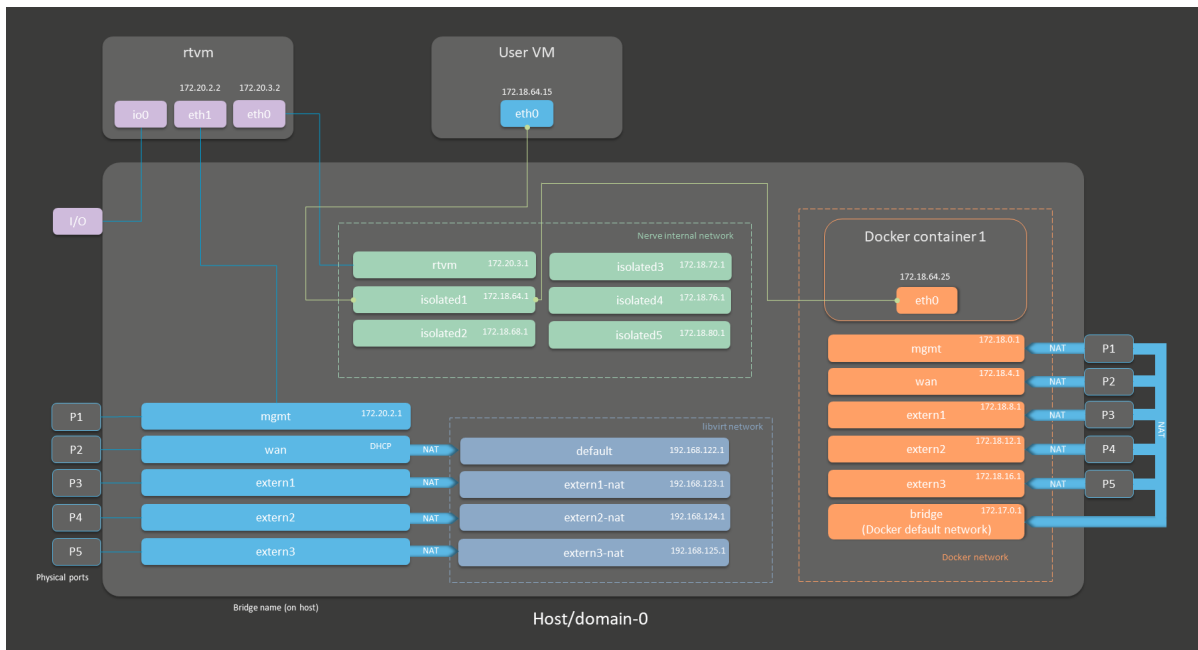
Network name\*  
extern1



Docker network

## Communication of a Docker container and a virtual machine through isolated networks

Nerve offers isolated network interfaces in the Nerve internal network for communication of workloads inside of the system. They can be used to establish communication between Docker containers, between virtual machines, or as in this example, between a Docker container and a virtual machine. Note that these interfaces do not communicate outside of the system.



The virtual machine has a "network card" installed. **User VM 1** is connected to the **isolated1** interface through **eth0**. **Docker container 1** is connected through its interface **eth0** to the same network interface, **isolated1**. Each interface has been assigned an IP address by a DHCP server in the designated range: 172.18.64.15 for **eth0** of **User VM1** and 172.18.64.25 for **eth0** of **Docker container 1**.

### Settings example

To achieve the functionality above, configure the interfaces of the Virtual Machine workload and the Docker workload the following way during the provisioning process in the Management System:

#### Virtual machine

## VIRTUAL MACHINE SPECIFIC INFO

Number of virtual CPUs \*

2

System memory to reserve \*

4

GB ▾



New data disk



PCI passthrough

New interface \*

Bridged ▾

isolated1



New interface

Assuming that the virtual machine is running Windows with the hostname DOCUMENTATION-PC which was defined in the creation process, the virtual machine can be reached under DOCUMENTATION-PC.isolated1.

### Docker container

Container restart policy 

Container name \*

docker-container-1

Network name \*

isolated1



Docker network

The Docker container can be reached under `docker-container-1.isolated1` due to the **Container name** and the isolated network chosen in the Docker workload settings.

## List of reserved TCP/UDP ports

In general, Nerve reserves the port range 47200 — 47399 on both TCP and UDP for internal usage. The following list states ports that are reserved in version 2.1.

Port	Interface	Protocol	Reserved for
22	none	TCP	SSH daemon
3000	172.20.2.1	TCP	Previous location of Local UI
3333	172.20.2.1	TCP	Local UI
47200	127.0.0.1	TCP/UDP	System Log
47201	127.0.0.1	UDP	Filebeat
47300	127.0.0.1	TCP	Local MQTT broker
47301	127.0.0.1	TCP	Local MQTT broker

## First steps with CODESYS

### NOTE

This chapter uses the MFN 100 as an example.

This chapter will give an introduction on how to start working with the integrated soft PLC in the MFN 100. First, some configuration and installation of files and libraries are required.

## NOTE

- Download the CODESYS Development System V3 from [store.codesys.com](https://store.codesys.com) for this chapter.  
We recommend version 3.5 SP14 (32 bit) or newer.
- Connect the workstation to the console port **P1** of the MFN 100.

## Installing the device descriptions

After downloading and installing the CODESYS Development System on the workstation, install the device description of the MFN 100 in the CODESYS Development System. The device description has the following filename:

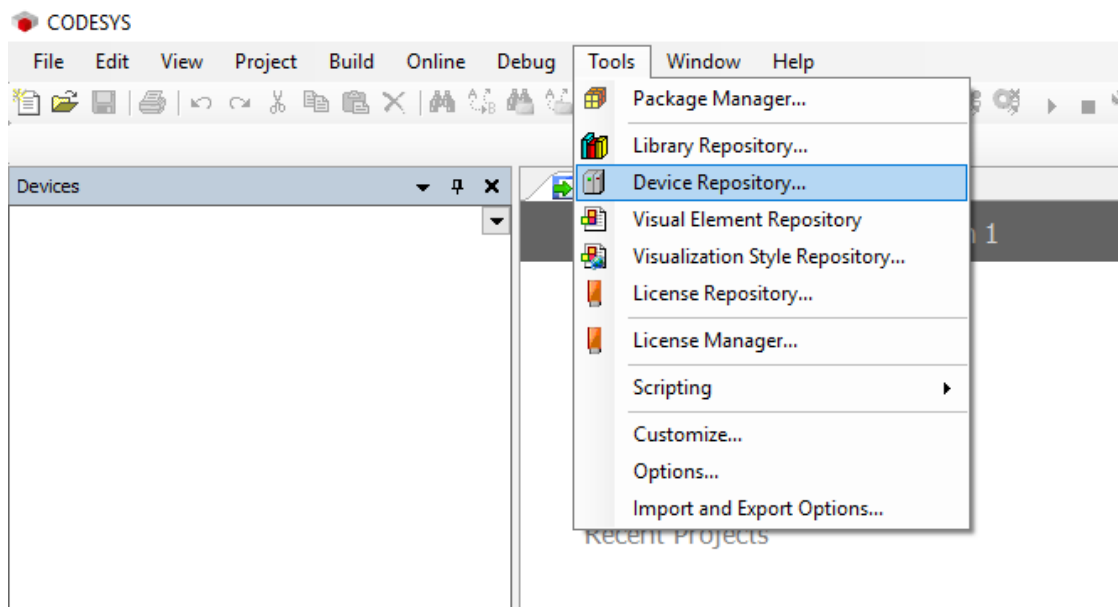
- Nerve\_MFN\_100\_V3.5.XX.X.devdesc.xml

## NOTE

XX.X stands for the current version of the CODESYS Development System

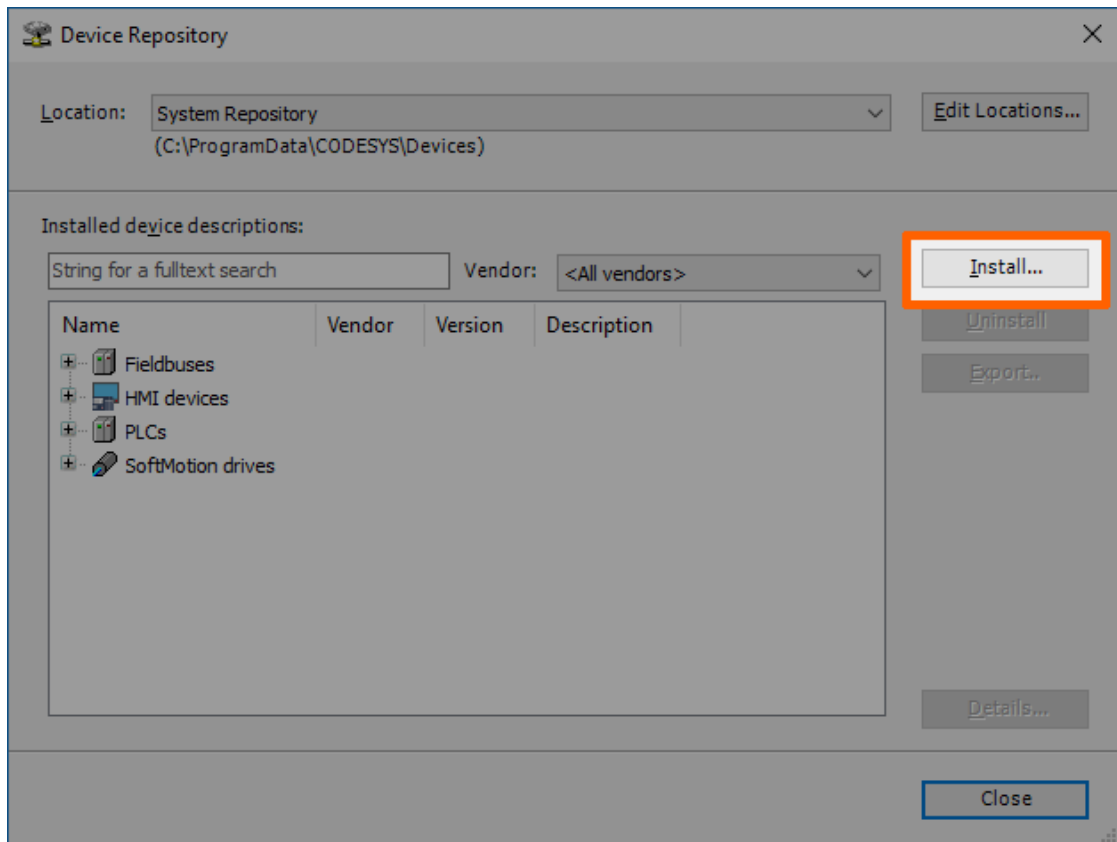
The device description of the MFN 100 is available at the [Nerve Software Center](#). Remember where the device description is saved for the following steps.

1. Start the CODESYS Development System.
2. Go to **Tools > Device Repository**.



3. Click **Install**.



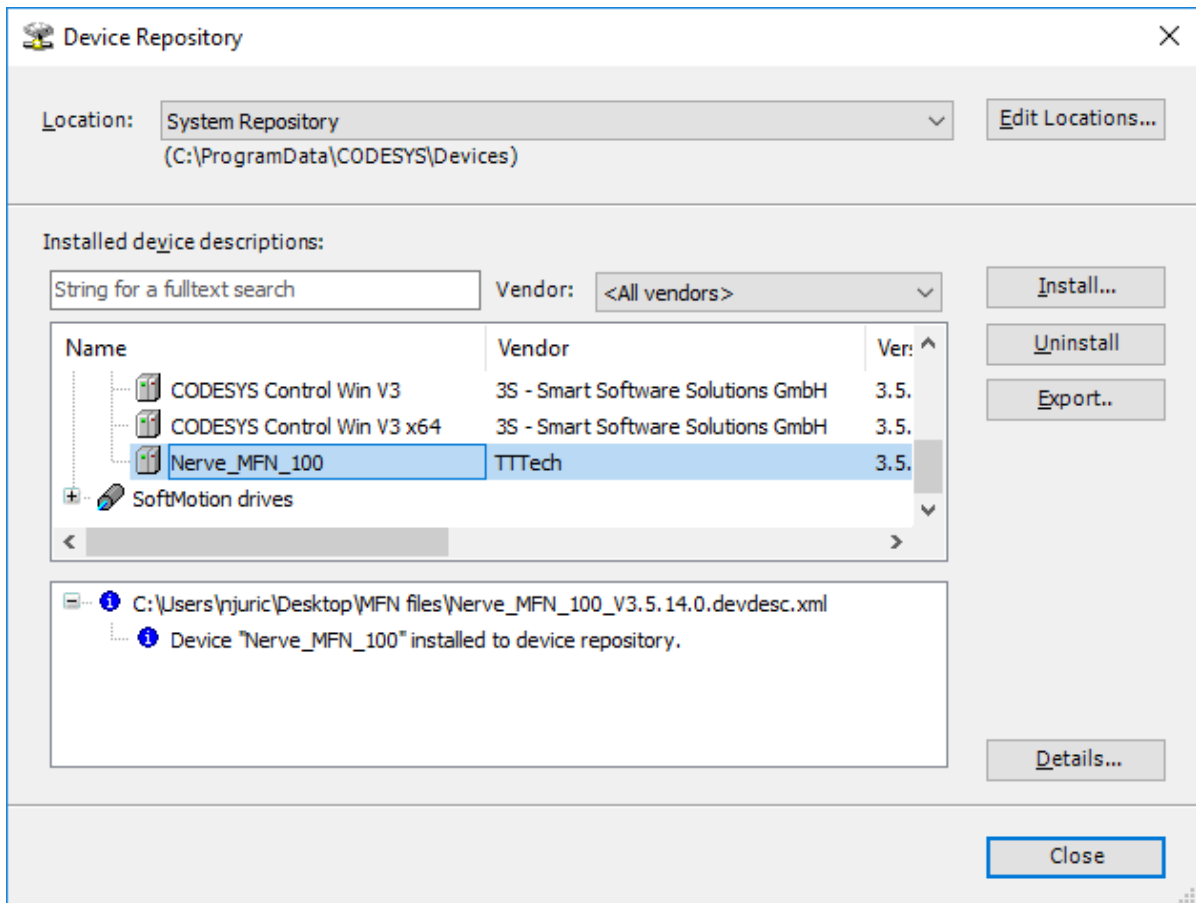


4. Go to the directory of the previously downloaded device description.
5. Select the device description of the MFN 100.

The device description will look like this: Nerve\_MFN\_100\_V3.5.XX.X.devdesc.xml

6. Click **Open**.

When the installation was successful, the MFN 100 will appear in the list of device descriptions in the middle of the window.



After installing the device description, work with the CODESYS Development System can be started. However, libraries and device descriptions of generic devices might be missing so that the CODESYS Development System can work properly. The following chapters cover the download process.

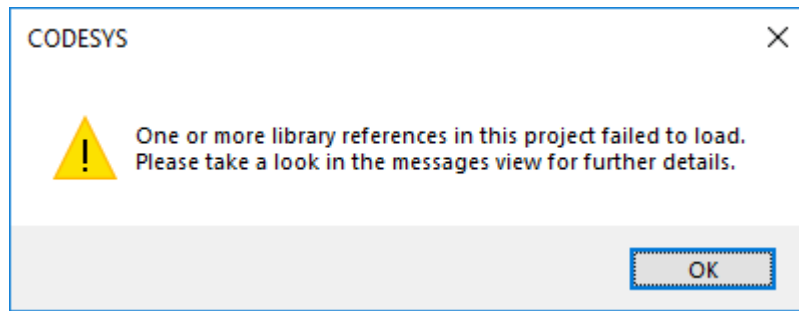
#### NOTE

The device description might need to be updated if this is not the first time working with MFN 100 and the CODESYS Development System:

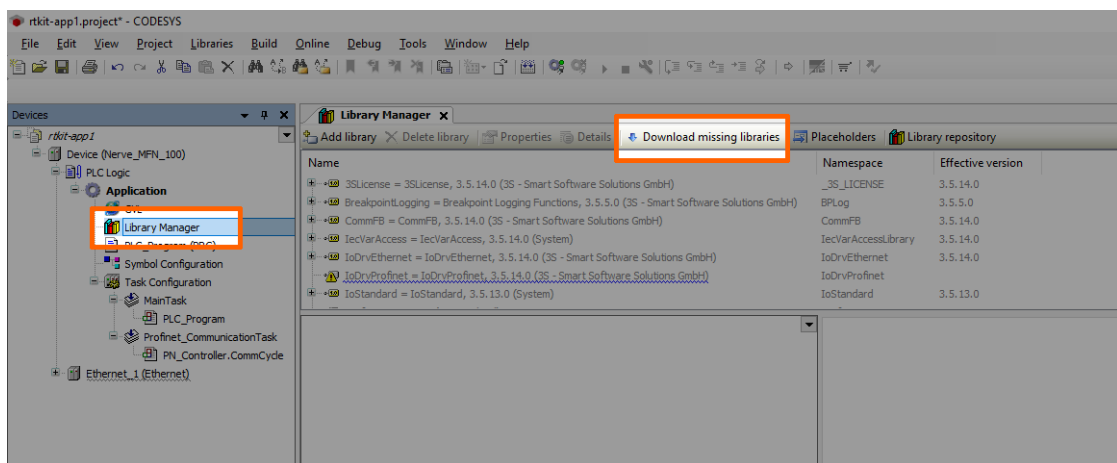
1. Follow the steps above to install the newest device description.
2. Right-click **Device (Nerve\_MFN\_100)** on the left side.
3. Select **Update Device...**
4. Select the current device description in the new window.
5. Click **Update Device** in the lower-right.

## Downloading missing libraries

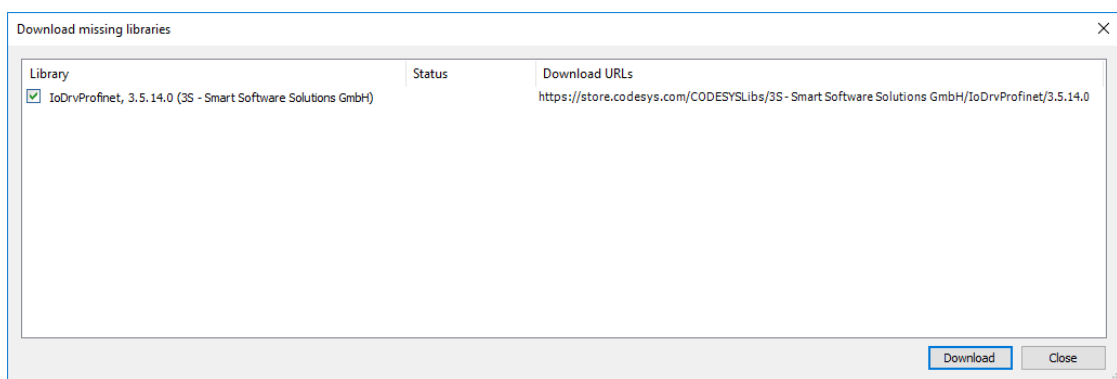
The error message for missing libraries might appear when opening or creating a CODESYS project. The CODESYS Development System identifies the missing libraries automatically but the following process might need to be repeated a few times.



1. Open or create a CODESYS project.
2. If the error message about missing libraries appears, click **OK**.
3. Double-click **Library Manager** in the tree view on the left.
4. Click **Download missing libraries**.



5. Click **Download** in the new window.



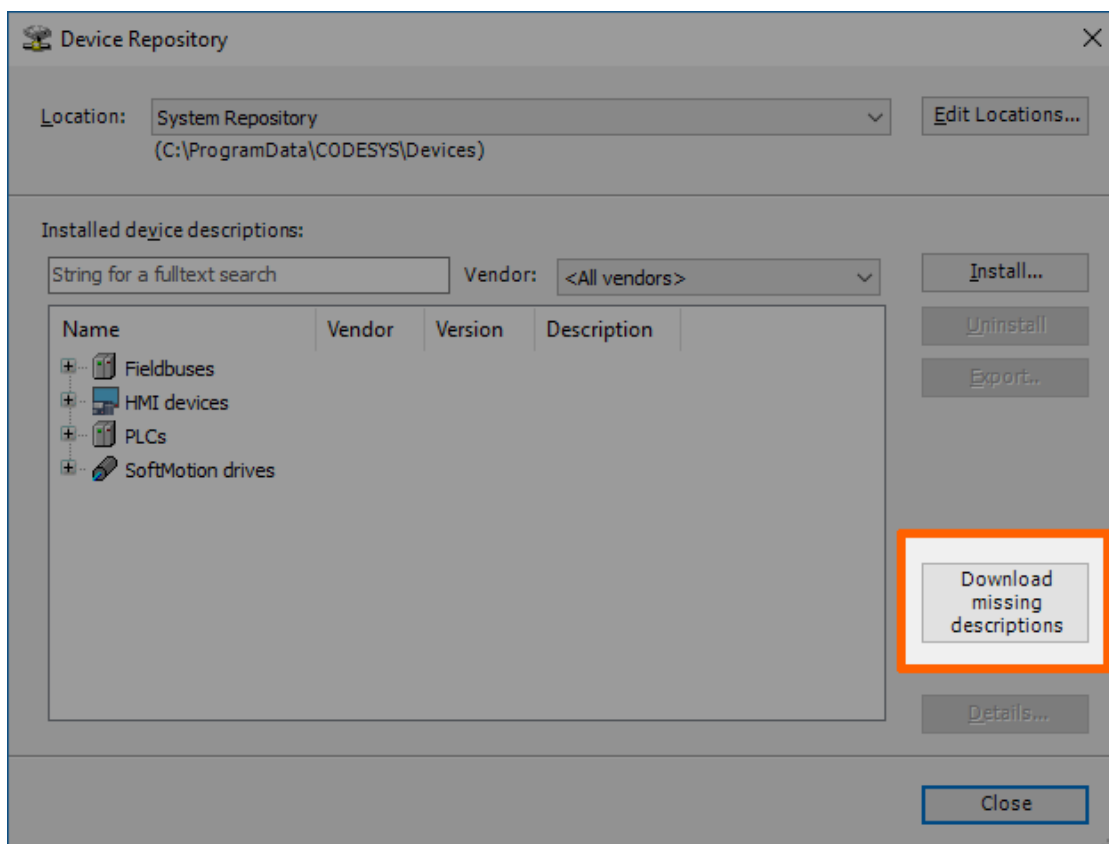
6. Click **Close** when the download is finished.
7. Repeat steps 3 to 5 until no more libraries appear in the download window.

## Downloading missing device descriptions

Apart from the device description for the MFN 100 that have been installed manually before, device descriptions of generic devices may be missing for the CODESYS Development System to function as intended. The CODESYS Development System will

identify the missing device descriptions automatically but this time it will not generate an error message unless a CODESYS application is being loaded into the MFN 100.

1. Click **Tools > Device Repository**.
2. Click **Download missing descriptions**.



#### NOTE

The button for downloading missing descriptions will not appear if no device descriptions of generic devices are missing. Close the window and continue if that is the case.

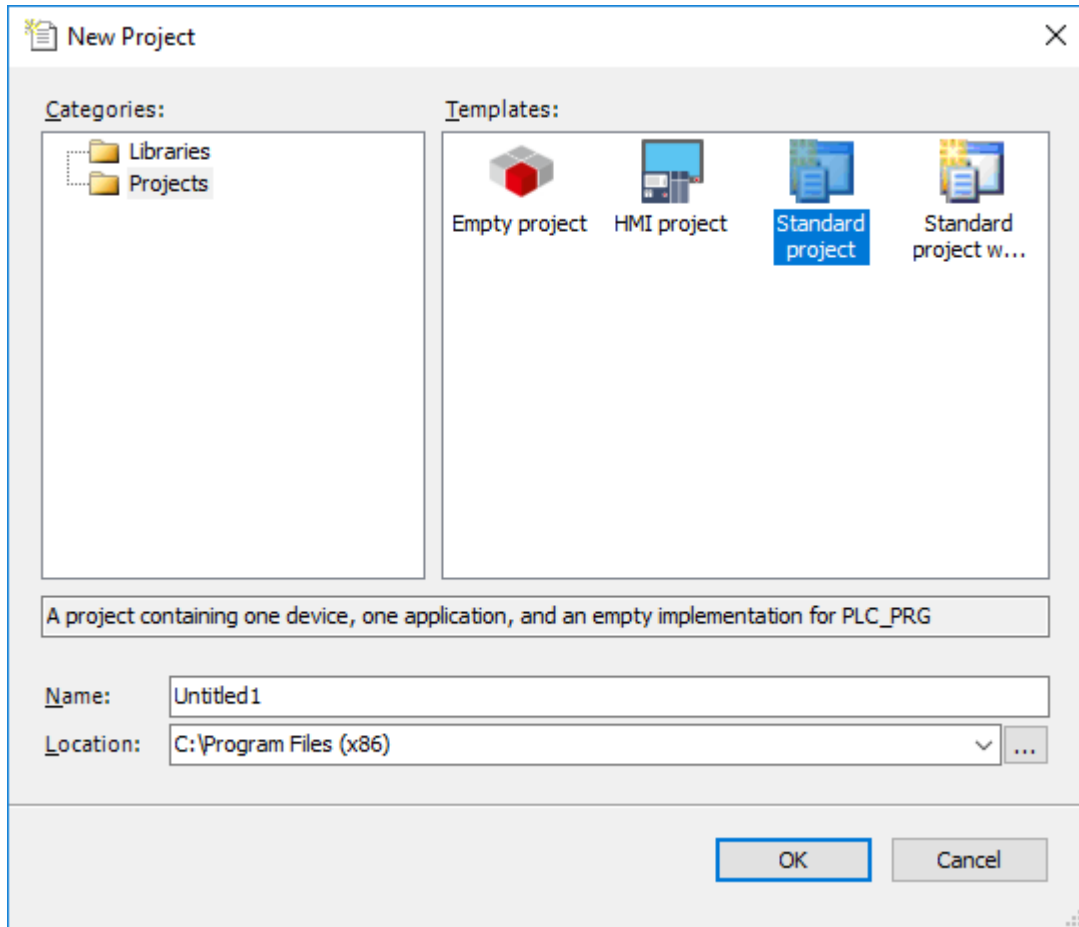
3. Click **Download** in the new window.
4. Click **Close** when the download is finished.

## Creating a new CODESYS project

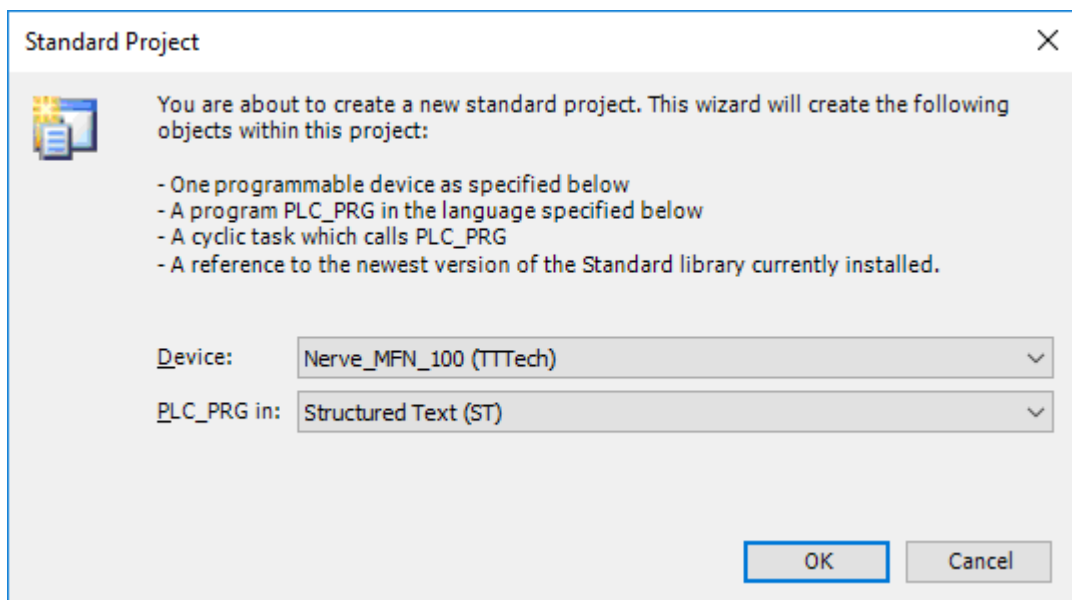
This example shows how to create a new project in the CODESYS Development System. The easiest way to get started is to create a **Standard project**.

1. Start CODESYS
2. Go to **File > New Project**.
3. Click **Standard project** on the right side among the templates.
4. Enter a name for the project.
5. Choose a **Location** where the project will be saved.

6. Click **OK** to save the project.

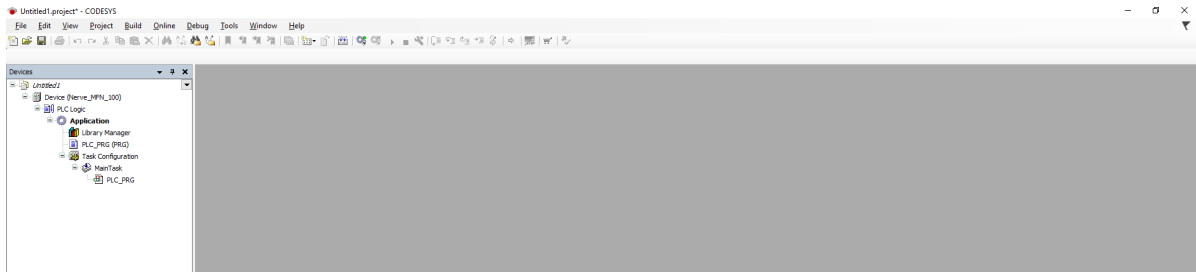


7. Select **Nerve\_MFN\_100 (TTTech)** as the device.



8. Click **OK**.



The result is an empty project that is open in the main view of CODESYS.



## Working with the default applications

To work with existing applications first, modify the default applications `app1.project` and `app2.project`. They have been sent as part of the delivery.

1. Start CODESYS.
2. Go to **File > Open Project**.
3. Select the location where the default applications are saved.
4. Select the application to work with.

Name	Date modified	Type	Size
 <code>app1.project</code>	24.07.2019 14:55	CODESYS project	287 KB
 <code>app2.project</code>	24.07.2019 14:57	CODESYS project	288 KB

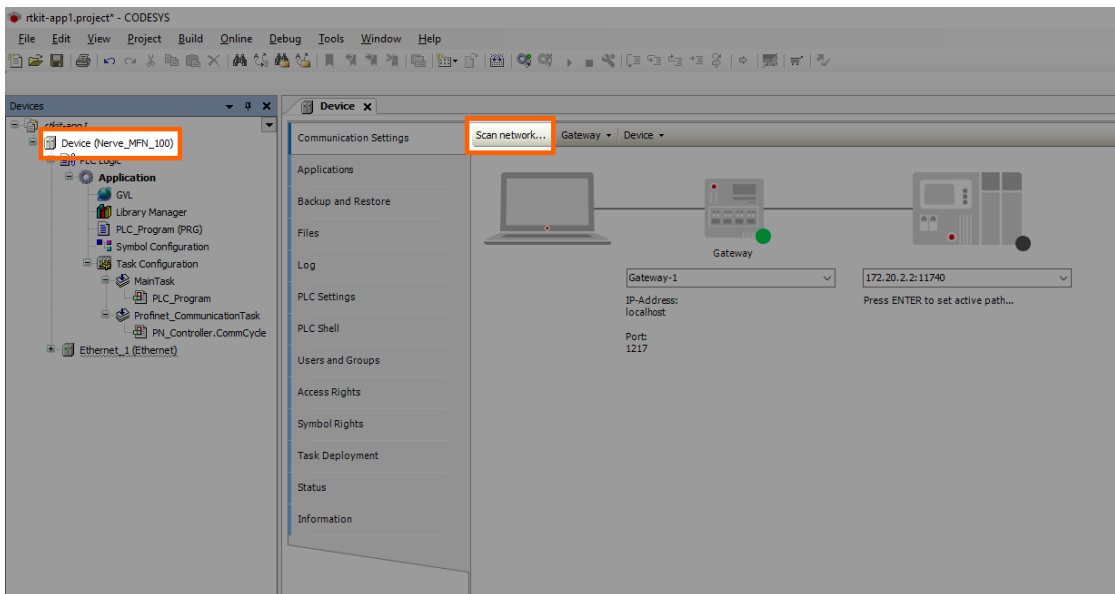
5. Click **Open**.

If the default applications are opened for the first time, some libraries and device descriptions will be missing. Follow the instructions [above](#) to see how to download the missing files.

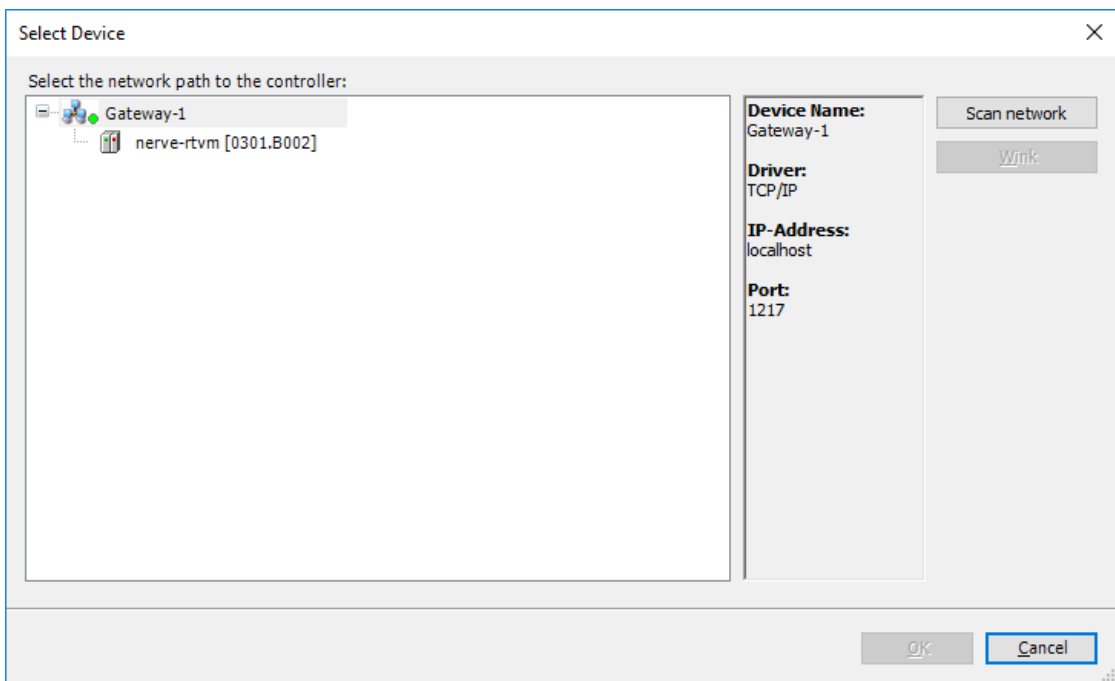
## Connecting to the MFN 100

Before downloading CODESYS applications to the MFN 100, make sure that the device description of the MFN 100 is installed in the CODESYS Development System.

1. Open or create a CODESYS project.
2. Double-click **Device (Nerve\_MFN\_100)** in the tree view on the left.
3. Go to **Communication Settings > Scan network....**



4. Select the MFN 100 (here **nerve-rtvm [XXXX.XXXX]**) in this window.



**NOTE**

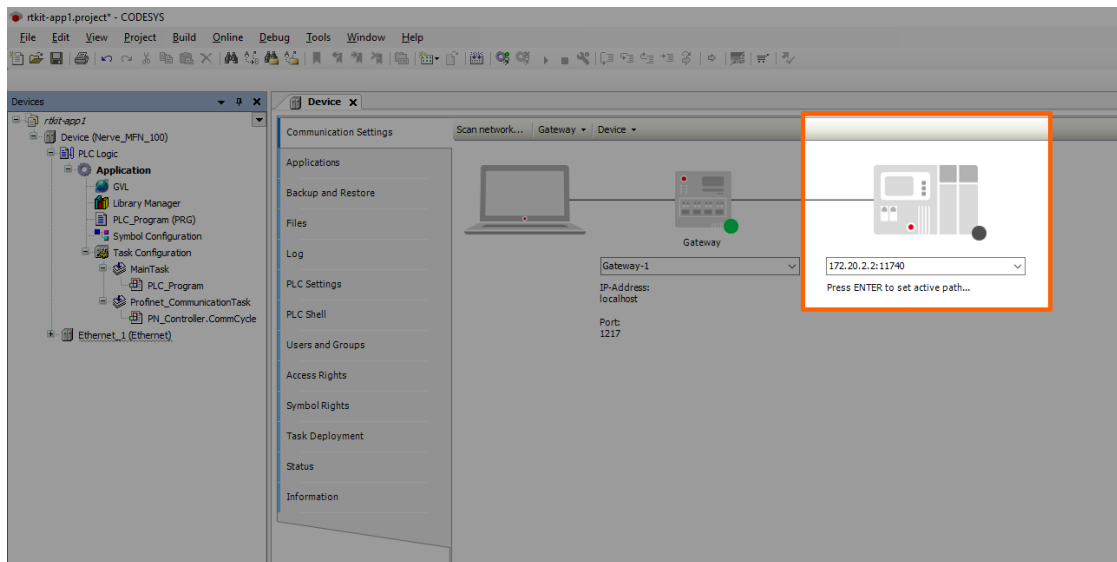
When more than one network is active on the workstation, it sometimes happens that the MFN 100 cannot be found. Continue reading if the MFN 100 does not appear in this window.

5. Click **OK**.

Typically the MFN 100 will be found automatically. If the MFN 100 cannot be found, enter the IP address and port of the CODESYS runtime manually.

1. Double-click **Device (Nerve\_MFN\_100)** in the tree view on the left.

2. Go to **Communication Settings** in the middle of the window.
3. Enter 172.20.2.2:11740 in the text box under the device on the right.



4. Press Enter.

The CODESYS Development System is now connected to the MFN 100 and applications can be downloaded into the CODESYS runtime.

## Downloading an application to the MFN 100

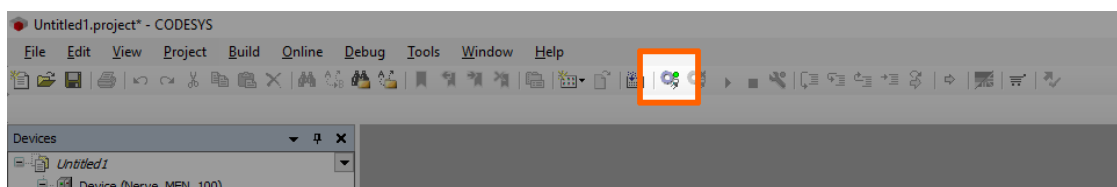
CODESYS applications can be loaded directly into the MFN 100. However, before downloading an application into the MFN 100 it needs to be free of errors.

The process of downloading an application is slightly different if an entirely new application is downloaded into the MFN 100 or if an application is being updated that has already been downloaded into the MFN 100. If updating an application that has been downloaded to the MFN 100 before, continue with [Downloading an Updated Application to the MFN 100](#) further down below.

## Downloading a new application to the MFN 100

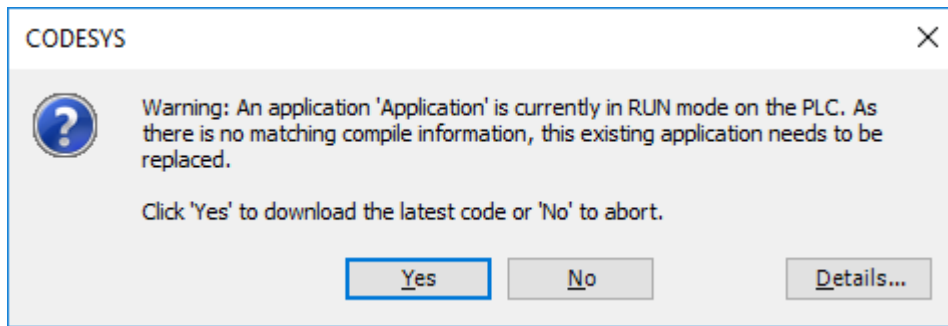
Once a project has been created and programming is finished, the CODESYS application can be downloaded to the MFN 100 directly.

1. Open the CODESYS project to load into the MFN 100.
2. Click the **Login** symbol in the CODESYS menu bar.

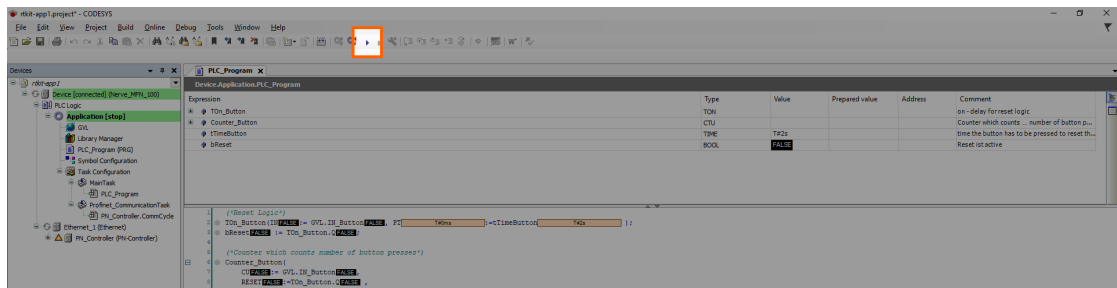


3. Click **Yes** in the pop-up window.





4. The application is stopped now. Click the **Play** symbol in the CODESYS menu bar.

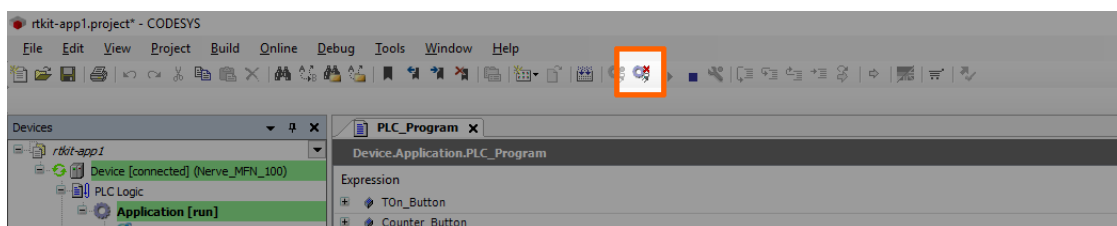


The application is now loaded to the MFN 100.

## Downloading an updated application to the MFN 100

If updating an application after loading it into the MFN 100, it needs to be downloaded into the MFN 100 again. The download process is slightly different from downloading a new application into the MFN 100.

1. Stop the CODESYS application that has been loaded into the MFN 100 through the **Local UI**.
2. Click the **Logout** button in the CODESYS toolbar.

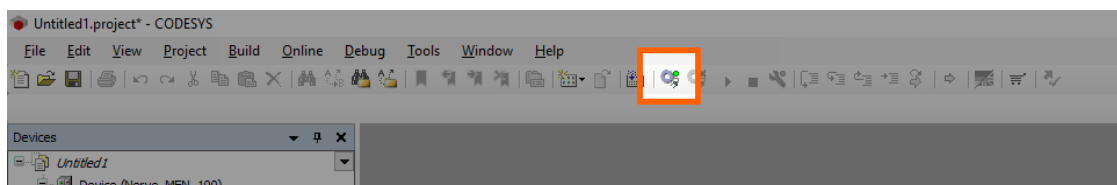


3. Expand **Device (Nerve\_MFN\_100) > PLC Logic > Application**.

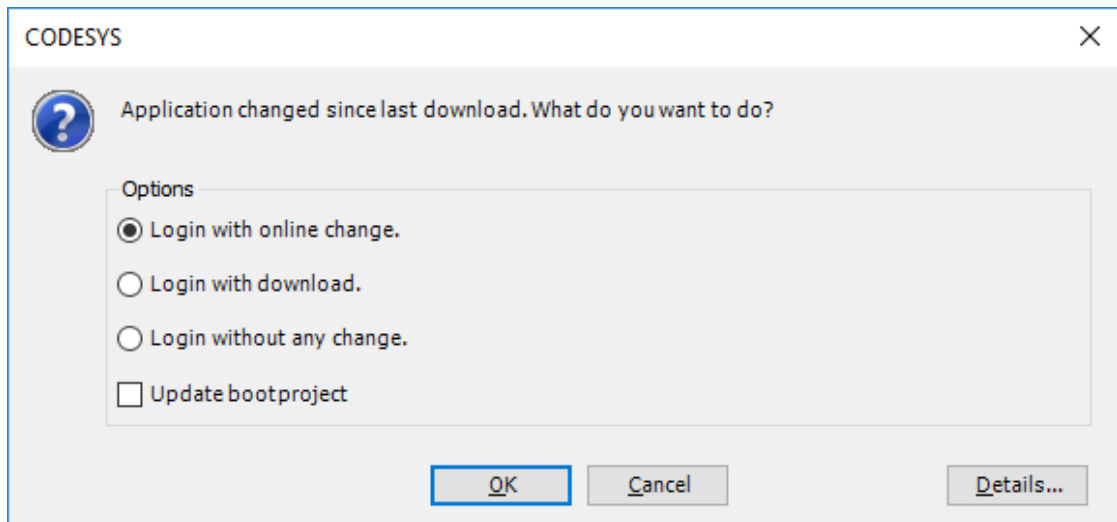
4. Double-click **PLC Program (PRG)**.

5. Perform changes.

6. Click the **Login** symbol in the CODESYS menu bar.



7. In the pop-up window, select one of the options.



Item	Description
<b>Login with online change.</b>	The updated application will be loaded into the MFN 100. Variable values will not be reset. If the application was running before, it will be running after the download.
<b>Login with download.</b>	The updated application will be loaded into the MFN 100. Variable values will be reset. The application is stopped.
<b>Login without any change.</b>	The updated application will not be loaded into the MFN 100 but the code will keep the changes.

8. Click **OK**.

The application is now loaded to the MFN 100.

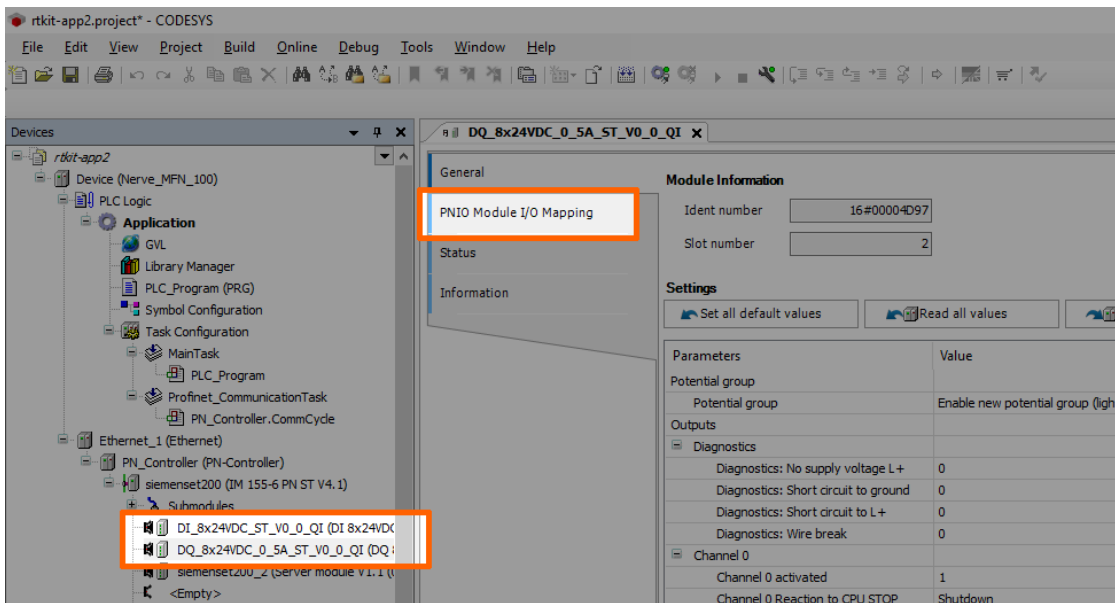
#### NOTE

For more help with programming PLC applications in the CODESYS Development System go to [help.codesys.com](http://help.codesys.com).

## Allocating variables to inputs or outputs

After connecting new sensors and actuators, assign variables to the I/O channel in CODESYS.

1. Open a CODESYS project.
2. Expand **Device (Nerve\_MFN\_100) > PLC Logic > Ethernet\_1 > PN\_Controller > siemenset200 (IM 155-6 PN ST V4.1)** in the tree structure on the left.
3. Double-click **DI\_8x24VDC\_ST\_V0\_0\_QI (...)** for digital inputs.  
Double-click **DQ\_8x24VDC\_0\_5A\_ST\_V0\_0\_QI (...)** for digital outputs.
4. Select **PNIO Module I/O Mapping**.

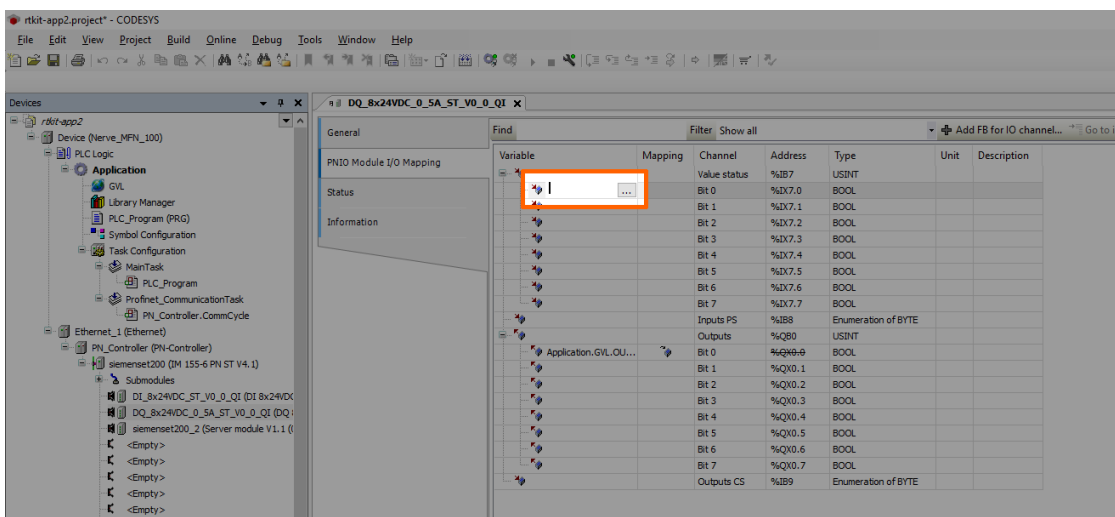


5. Fully expand the tree view.
6. Double-click the variable slot to assign.

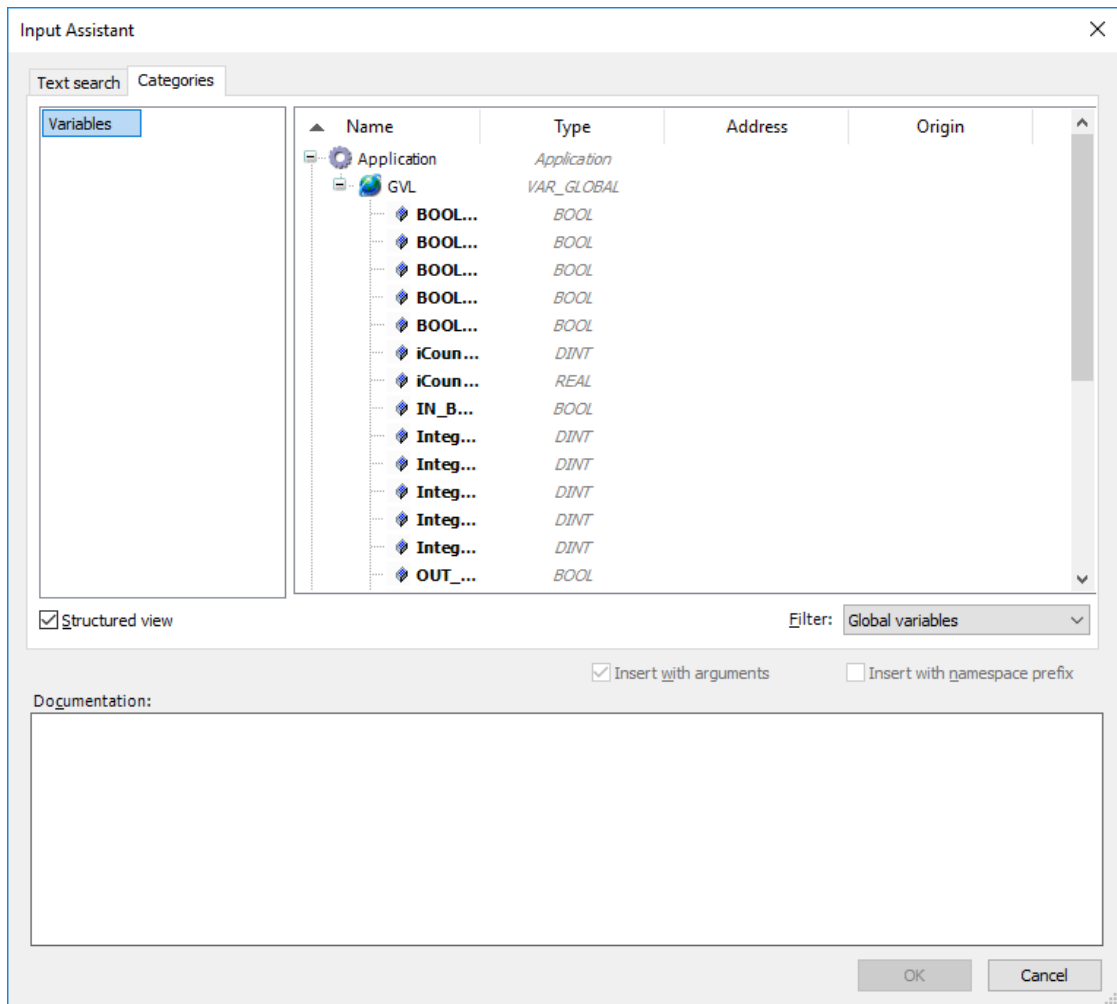
#### NOTE

The inputs in this view do not match the physical inputs of the I/O module on the kit. The inputs here go from 0 to 7. The physical inputs go from 1 to 8. Therefore input 0 in this view represents the physical input 1 on the I/O module. This also applies to outputs.

7. Click the three dots next to the variable slot.



8. Expand **Application** > **GVL** in the new window.



9. Select the variable to assign.

#### NOTE

Make sure to select a variable of the same type as the input, i.e., a **BOOL** variable for a **BOOL** input or output.

10. Click **OK**.

Use the assigned variables to read data from connected sensors or to control actuator functionality.

#### NOTE

For more help with programming PLC applications in the CODESYS Development System go to [help.codesys.com](http://help.codesys.com).

## Enabling retain variables

NerveCodesysRetainVar is a library for storage and restoration of retain variables with prevention of data loss in case of system crash (e.g. power outage). To prevent data loss, the retain variables are stored in two file copies under `/opt/data/var/lib/nerve-codesys/PlcLogic/`.

The retain variables library (`NerveCodesysRetainVar.compiled-library`) and an example project showcasing typical use (`demo.project`) are available at the [Nerve Software Center](#). Follow the instructions below to install and import the retain variables library into a project.

### Installing and importing the NerveCodesysRetainVar library

Download the retain variables library (`NerveCodesysRetainVar.compiled-library`) from the [Nerve Software Center](#) first before following the instructions below.

1. Open or create a CODESYS project.
2. Select **Tools > Library Repository** in the toolbar.
3. Click **Install...** in the new window.
4. Navigate to the folder containing the retain variables library and select the `NerveCodesysRetainVar.compiled-library` file.
5. Select **OK**.

The library will appear in the **Application** element after it has been installed. Close the library repository window by selecting **Close** in the lower right. Next the library needs to import into the project.

1. Double-click **Library Manager** in the tree view on the left.
2. Select **Add library** in the **Library Manager** tab that opened on the right.
3. Enter `NERVE Retain` in the search field to search for the library.
4. Select the **NERVE Retain Variables Library**.
5. Select **OK**.

The **Nerve Retain Variables Library** now appears in the **Library Manager** tab.

### Example project and use

The example project (`demo.project`) is meant to show how the library functions can be used. It includes two variants of a simple program counting up multiple counters stored as permanent variables:

- **PLC\_PRG1** stores and restores the retain variables just once based on a trigger activated by boolean flags (store and restore).
- **PLC\_PRG2** shows a typical use case, where the retain variables are restored once during the initialization phase and then periodically stored every cycle after all the counters are updated.

The `NerveCodesysRetainVar` library provides two functions:

- `StoreRetains` to store retain variables
- `RestoreRetains` to restore retain variables

The namespace of the library is `NerveCodesysRetainVar`.

Functions have a return value of type `RTS_IEC_RESULT` defined in the **SysTypes Interfaces** library. It is required to either include this library or define the return variables as type `NerveCodesysRetainVar.RTS_IEC_RESULT`. The possible error codes are defined by [CmpErrors2 Interfaces](#).

Note that RestoreRetains may fail with error code 0x32 when the application runs for the first time, before any retain variables were stored. This is because the files with stored retain variables do not exist yet.

During the startup of the device, the CODESYS application uses the function RestoreRetains. This loads the last stored state of retain variables.

During execution, the CODESYS application periodically saves the current state of retain variables with the function StoreRetains. This could be done in a periodically executed task or at certain points during calculations, when it is meaningful to store the current state.

In case of a sudden crash of the CODESYS application (e.g. power failure), the last saved state of the retain variables will be restored.

#### NOTE

The type of memory must be taken into consideration for periodical storing of retain variables. Writing too often for longer periods of time may result in damage to disks.

## Integrating Nerve into the build system

The Management System can be controlled with the API. In this version of the Management System API documentation, the focus lies on working with workloads. As a demonstration, download the [Nerve\\_API\\_2.3.0.zip](#) from the [Nerve Software Center](#). The login credentials for the download can be found in the customer profile.

With the Python script it is demonstrated how to:

- provision CODESYS, Docker and Virtual Machine workloads
- deploy workloads
- undeploy workloads
- delete workloads

Full API documentation will be made available in future versions.

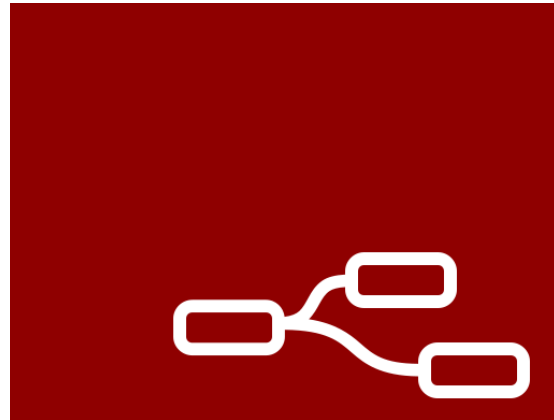
## Supported Applications

## Supported Applications

Nerve is a platform for applications running on machines and accessing machine data. Supported applications increase the functionality of Nerve to accustom a larger number of use cases. The applications are tested and regularly maintained. More applications will be added in the near future, as well as usage examples for each application.

Each section of an application below provides assistance in the usage of the supported application with Nerve. It offers an overview and an explanation of each application, as well as step-by-step examples and useful links.

Select an application from the table below for more information:



# Node-RED

For questions and information not covered by the documentation, contact a sales representative or write an email to [support@tttech-industrial.com](mailto:support@tttech-industrial.com)

## Crosser

Crosser designs and develops streaming analytics and integration software for edge, on-premise or cloud. The Crosser platform enables real-time processing of streaming or batch data for Industrial IoT, data transformation, analytics, automation and integration. The Crosser platform consists of two core parts.

- Crosser Cloud is the heart of the platform where all design and orchestration takes place. It is a multi-tenant SaaS service hosted by Crosser but also exists in an on-premise version that customers can run on a private cloud, inside the internal firewall.
- Crosser Node is the real-time engine that customers install where they need it and anywhere a Docker container can be used.

Refer to the [official Crosser homepage](#) for more information on Crosser.

## Requirements for usage with Nerve

To use Crosser with Nerve, a Crosser Cloud account is required. Use one of the provided test options or get in touch with Crosser directly:

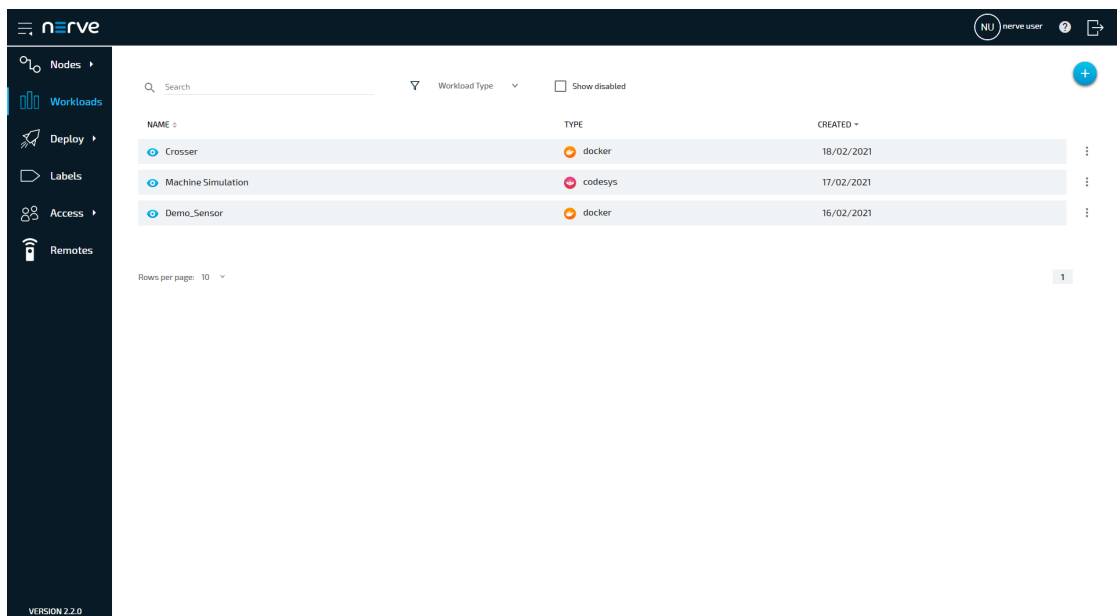
- [Self-service free trial](#)
- [Supported trial](#)
- [sales@crosser.io](mailto:sales@crosser.io)

## Getting started

The Crosser workload is pre-configured in the Nerve Management System as part of the delivery so that the application can be used once the account for the Crosser Cloud is set up. Follow the steps below to finalize the configuration. The pre-configured workload is set up with the following settings:

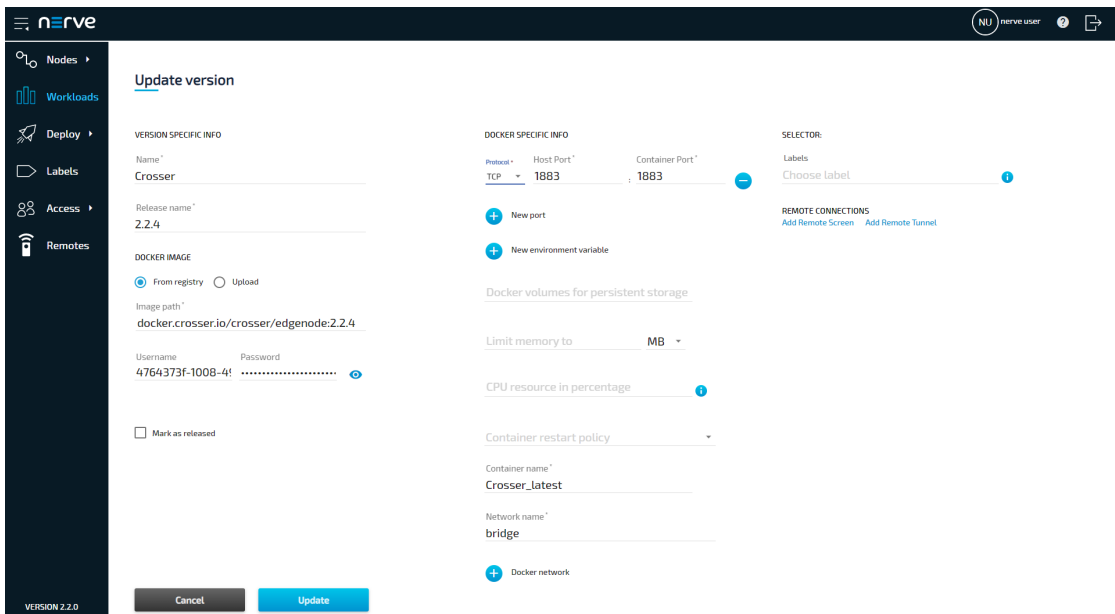
Setting	Value
<b>Name</b>	Crosser
<b>Release name</b>	2.2.4
<b>DOCKER IMAGE</b>	<b>From Registry</b> is selected with an <b>Image path</b> and login credentials to the Docker image.
<b>Port settings</b>	<b>Protocol</b> TCP
	<b>Host Port</b> 1883
	<b>Container Port</b> 1883
<b>Container name</b>	Crosser_latest
<b>Network name</b>	bridge

1. Log in to the Nerve Management System. Note that the initial login credentials can be found in the customer profile.
2. Select **Workloads** in the navigation on the left.

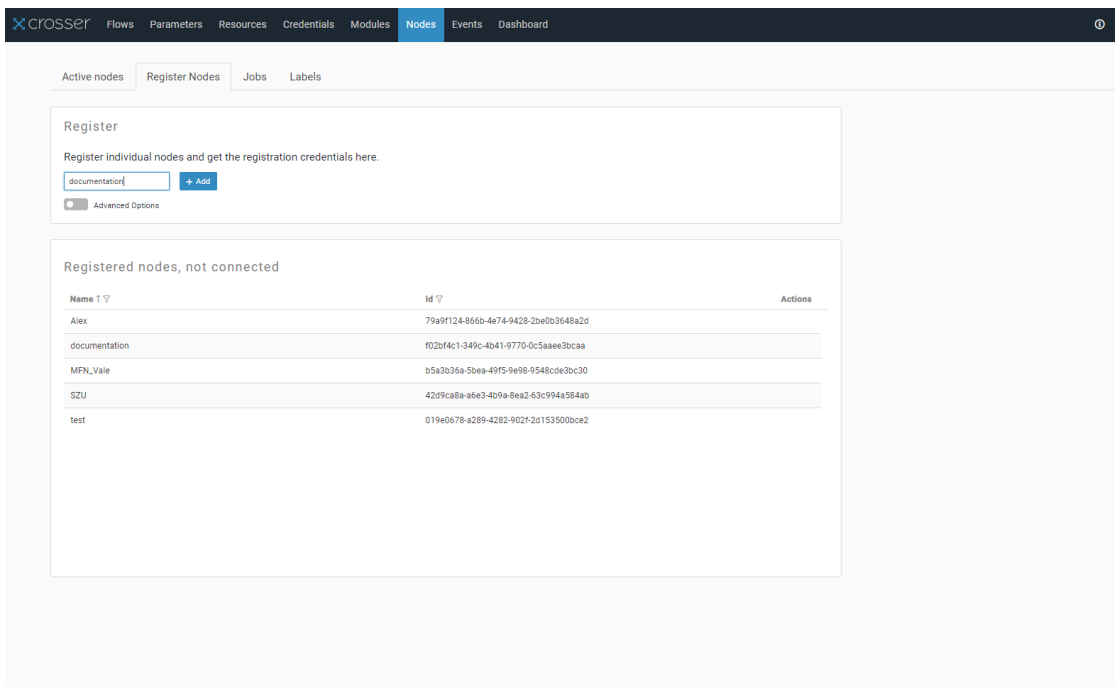


3. Select the Crosser workload.
4. Select the latest version on the right to edit the settings of the version.

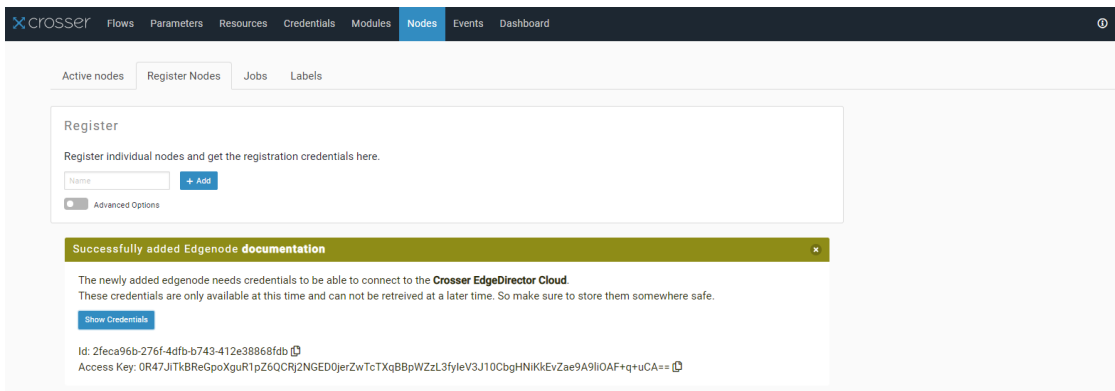




5. Log in to [Crosser Cloud](#).
6. Select **Nodes** in the navigation at the top.
7. Select the **Register Nodes** tab.
8. Enter a name for the node in the field.



9. Select **+ Add**. A notification saying **Successfully added Edgenode** appears below when the registration is successful.
10. Select **Show Credentials** to display the NodeID and the Access Key. They are needed for the configuration of the Crosser workload.

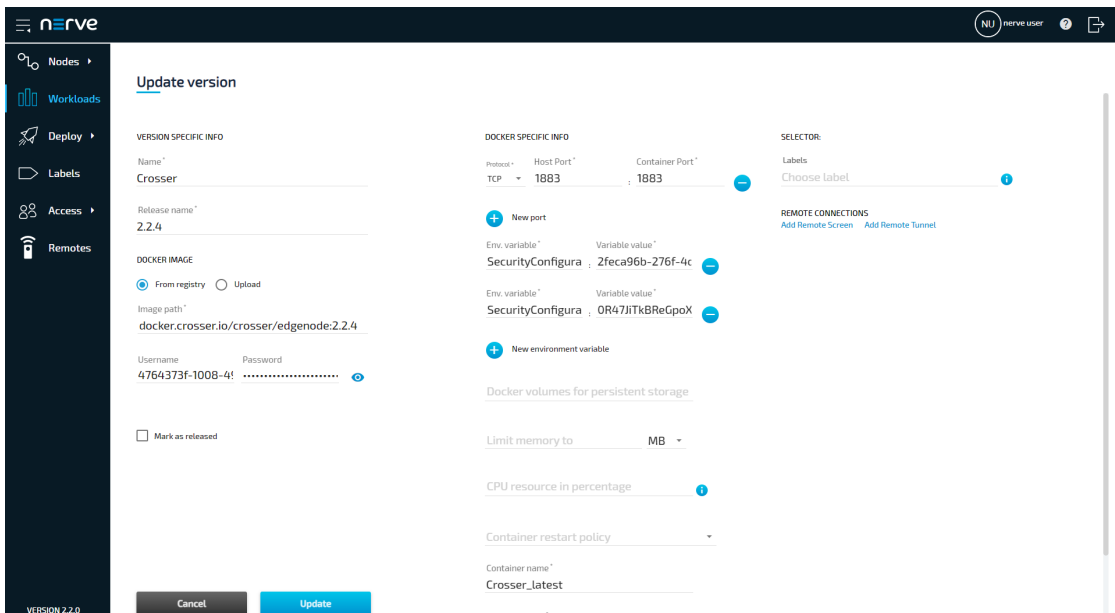


11. Switch back to the Management System.
12. Select + next to **New environment variable**.
13. Enter the following values:

Field	Value
<b>Env. variable</b>	SecurityConfiguration__Credentials__NodeId
<b>Variable value</b>	Enter the value next to <b>Id</b> in the Crosser Cloud credentials window.

14. Select + next to **New environment variable** again.
15. Enter the following values:

Field	Value
<b>Env. variable</b>	SecurityConfiguration__Credentials__AccessKey
<b>Variable value</b>	Enter the value next to <b>Access Key</b> in the Crosser Cloud credentials window.

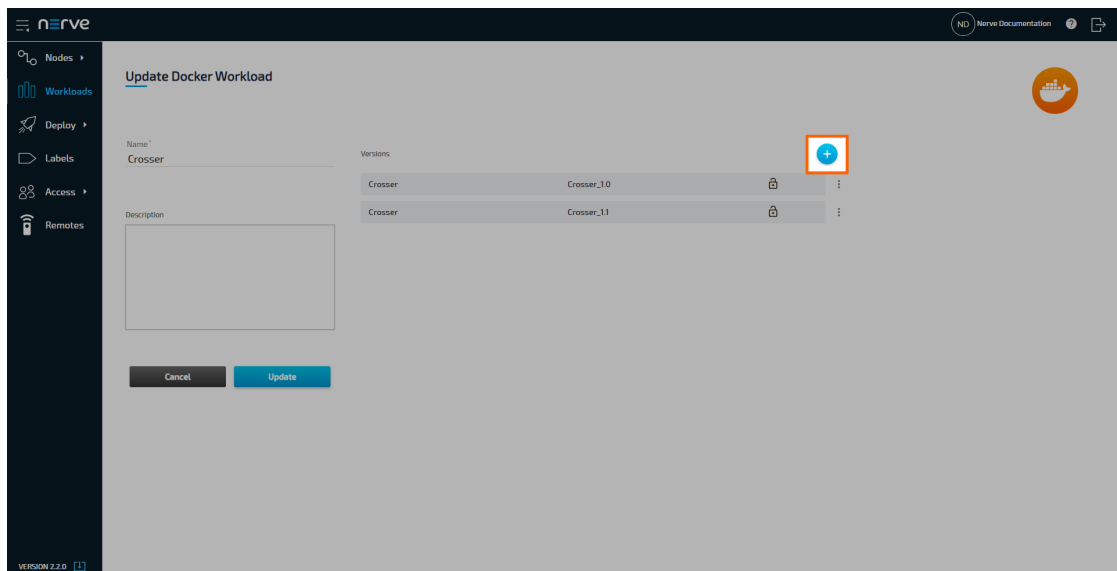


16. Select **Update**.

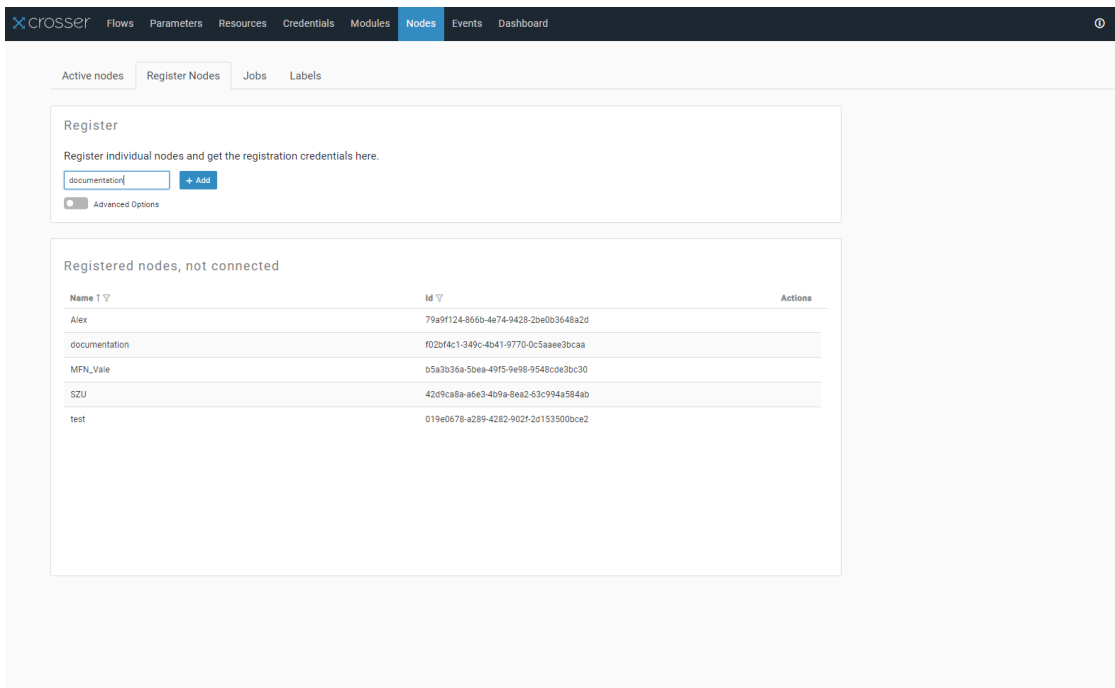
## Onboarding more nodes to Crosser Cloud

In order to onboard further nodes to Crosser Cloud, a new version of the Crosser workload has to be created in the Management System.

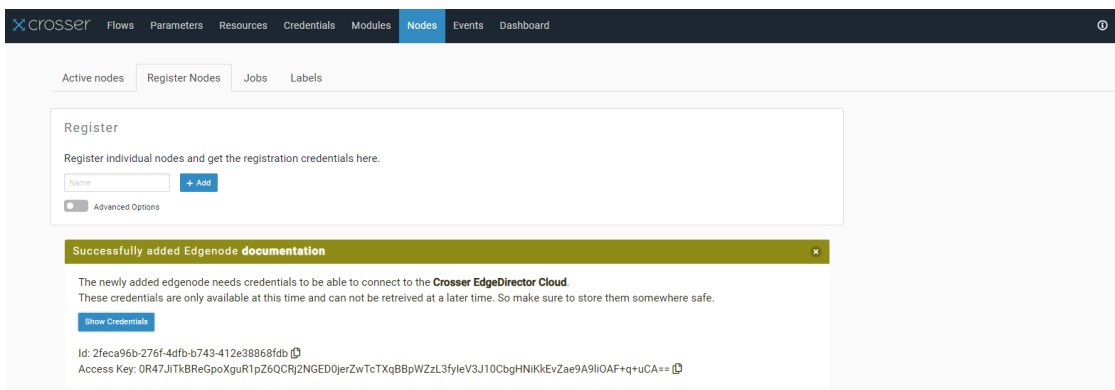
1. Select **Workloads** in the navigation on the left.
2. Select the **Crosser** workload from the list of workloads.
3. Click the plus symbol on the right. The settings from the latest version are automatically filled in.



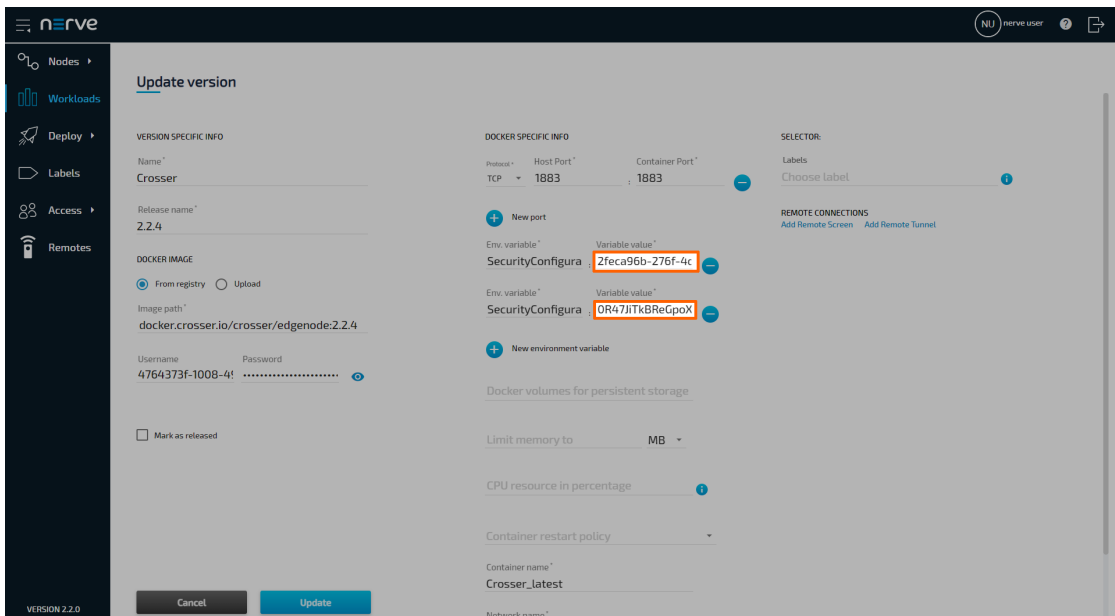
4. Switch to [Crosser Cloud](#).
5. Select **Nodes** in the navigation at the top.
6. Select the **Register Nodes** tab.
7. Enter a name for the node in the field.



8. Select **+ Add**. A notification saying **Successfully added Edgenode** appears below when the registration is successful.
9. Select **Show Credentials** to display the NodeID and the Access Key.



10. Switch back to the Management System.
11. Replace the value of the `SecurityConfiguration_Credentials_NodeId` environment variable with the new value of **Id** in the Crosser Cloud credentials window.
12. Replace the value of the `SecurityConfiguration_Credentials_AccessKey` environment variable with the new value of **Access Key** in the Crosser Cloud credentials window.



13. Update the **Name** of the version.

#### NOTE

Choose the name of the node in Crosser Cloud for easy identification of the workload version.

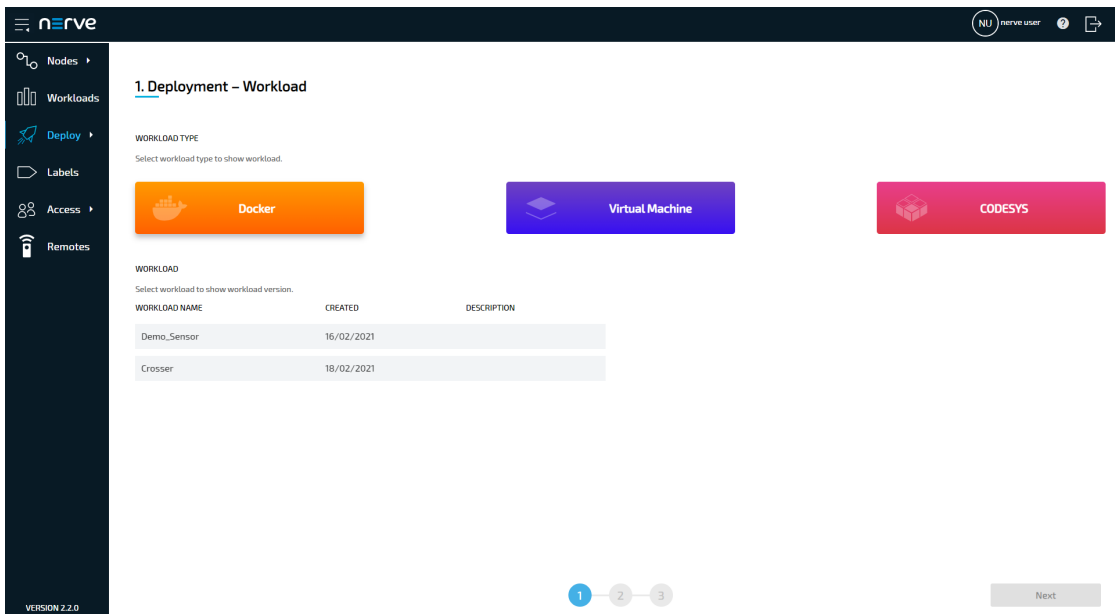
14. Click **Save** to save the new version of the workload.

15. Select **Update**.

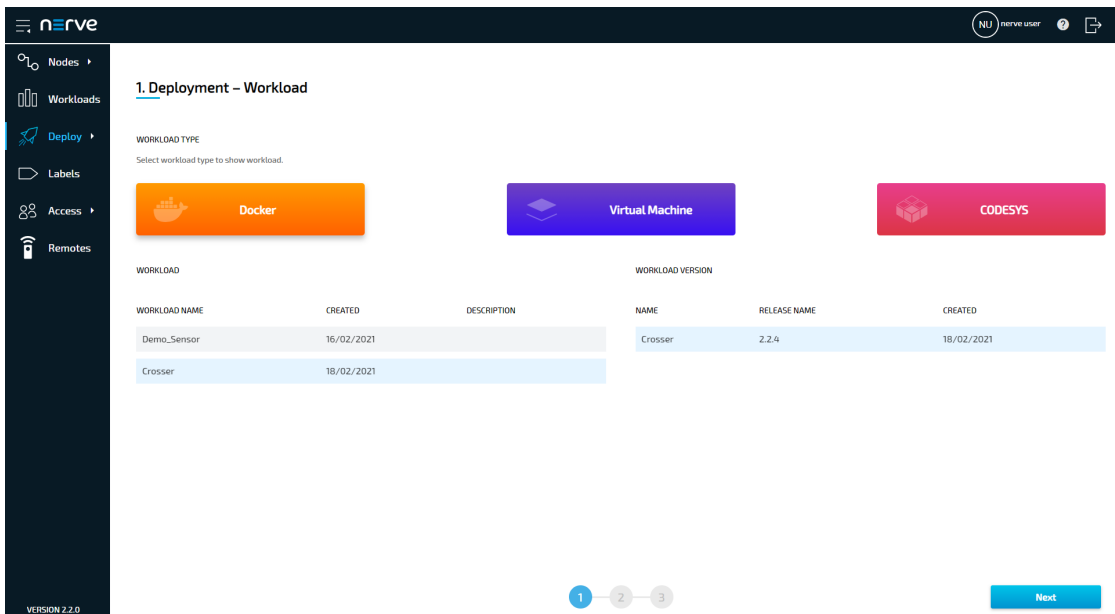
## Deploying the Crosser workload

After configuring the Crosser workload in both the Management System and Crosser cloud, follow the instructions below to deploy the Crosser workload to a Nerve Node in order to start using Crosser with Nerve.

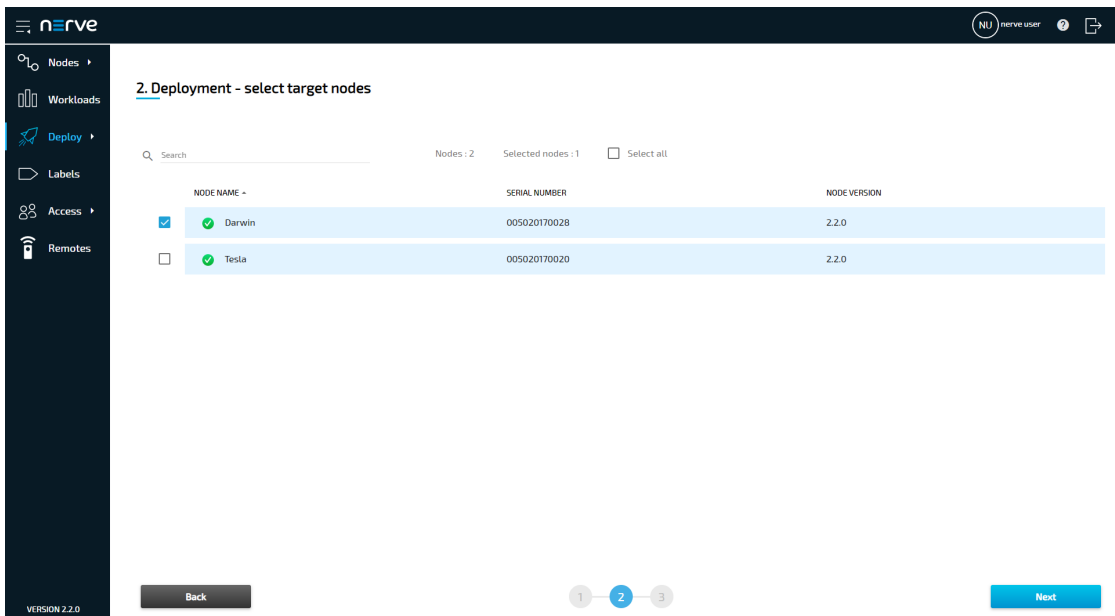
1. In the Management System select **Deploy** in the navigation on the left.
2. Select the orange Docker tab. A list of Docker workloads will appear below.



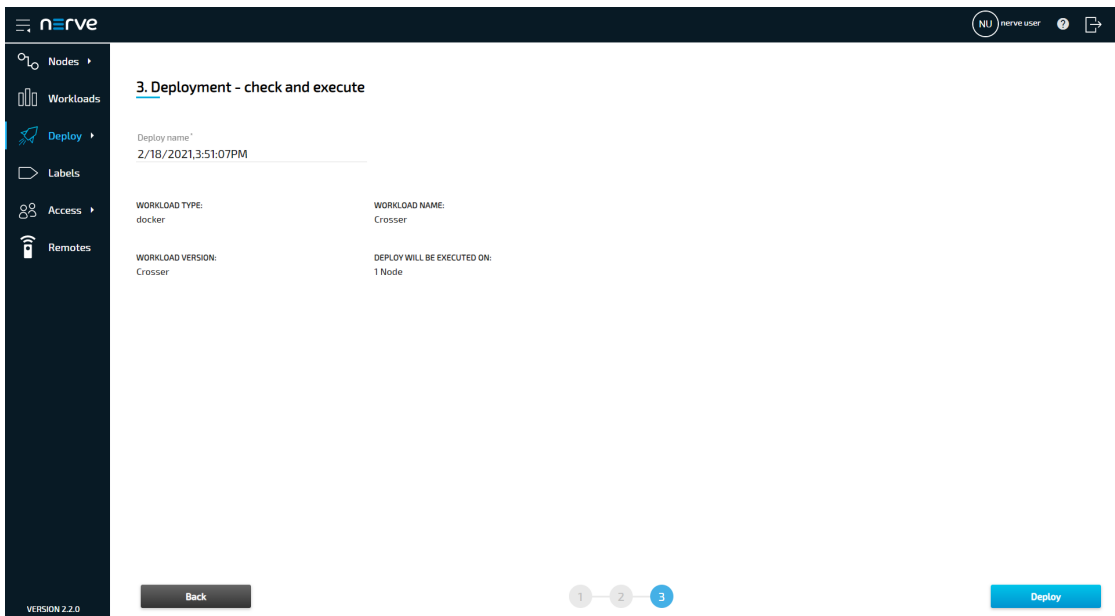
3. Select **Crosser** in the list of workloads. A list of versions of this workload will appear to the right.
4. Select the corresponding version of the workload that is connected to a node in the Crosser Cloud.



5. Select **Next** in the lower-right corner.
6. Tick the checkbox next to the node that Crosser shall be deployed to.
7. Select **Next** in the lower-right corner.

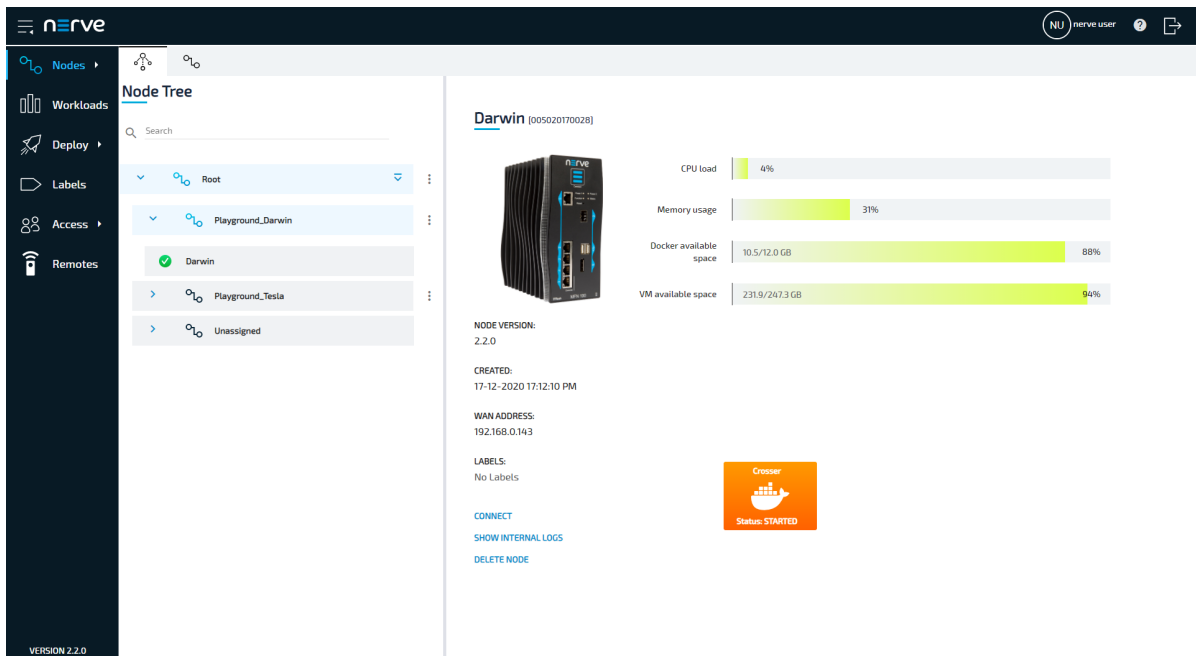


8. Select **Deploy** to execute the deployment.  
*Optional: Enter a Deploy name above the Summary of the workload to make this deployment easy to identify. A timestamp is filled in automatically.*

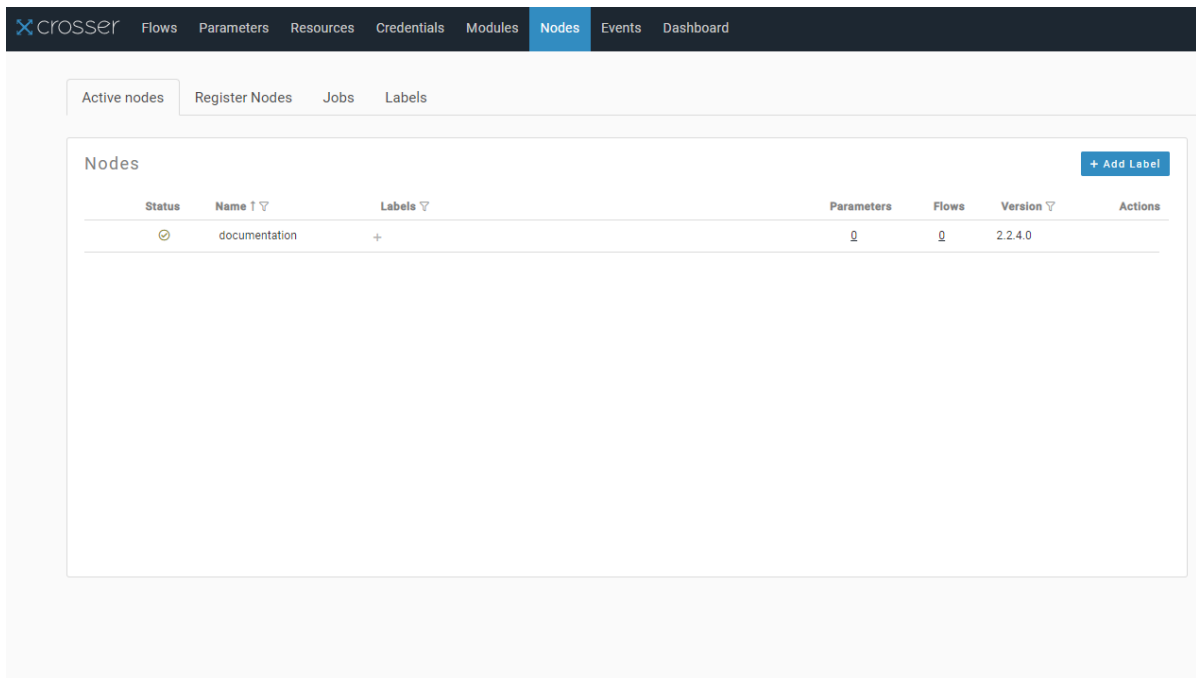


The deployment should now be visible at the top of the deployment log. Click the log entry of the deployment to see a more detailed view.

To confirm if the workload has been deployed successfully, select **Nodes** in the navigation on the left. Select the corresponding node in the node tree and confirm if the orange Crosser workload tile is showing underneath the bar graph. The workload should show the status **STARTED**.



To confirm if the workload has been successfully connected to the Crosser Cloud, select **Nodes** in the navigation at the top. The node appears in the **Active nodes** list and is marked with a check mark in the **Status** column.



## Node-RED

Node-RED is an open source programming tool for wiring together sensor inputs, APIs and online services. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click.



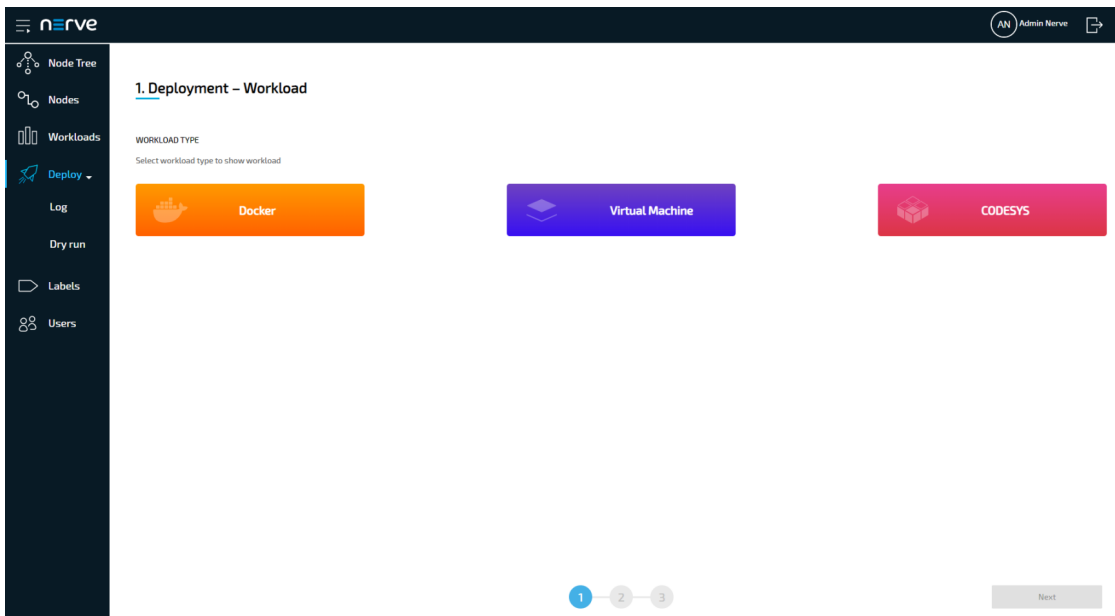
Refer to the [official Node-RED homepage](#) and the [Node-RED documentation](#) for more information on Node-RED.

## Deploying Node-RED

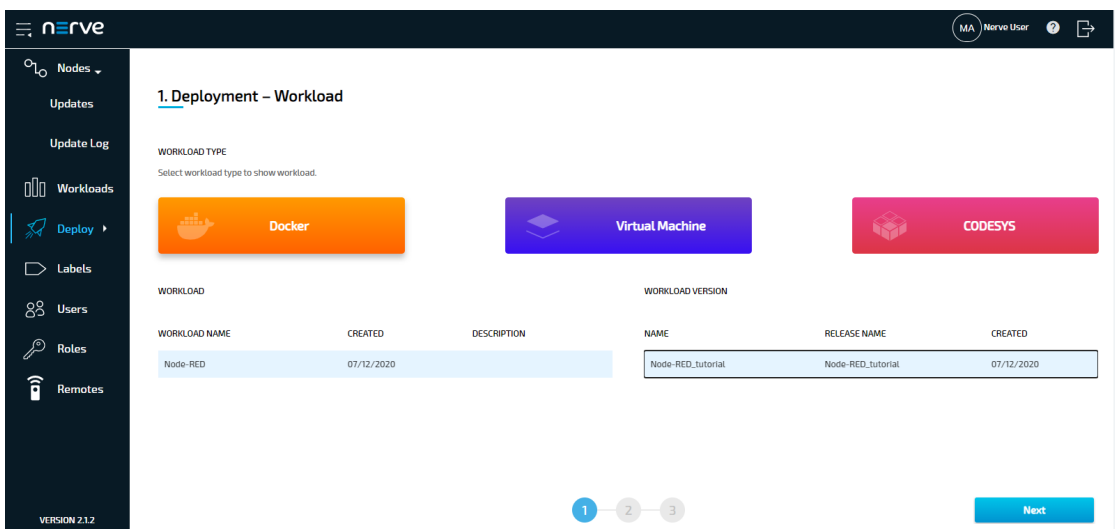
The Node-RED workload is pre-configured in the Nerve Management System as part of the delivery so that the application can be used out of the box. Follow the steps below to deploy Node-RED to a Nerve Node. The pre-configured workload is set up with the following settings:

Setting	Value
<b>Name</b>	Node-RED_Tutorial
<b>Release name</b>	1.2.7-12
<b>DOCKER IMAGE</b>	<b>From Registry</b> is selected with an <b>Image path</b> to the Docker image.
<b>Port settings</b>	<b>Protocol</b> TCP <b>Host Port</b> 1880 <b>Container Port</b> 1880
<b>Container name</b>	Node-RED
<b>Network name</b>	wan
<b>Network name</b>	bridge
<b>REMOTE CONNECTIONS</b>	<b>NAME</b> Node-RED <b>TYPE</b> TUNNEL <b>PORT</b> 1880

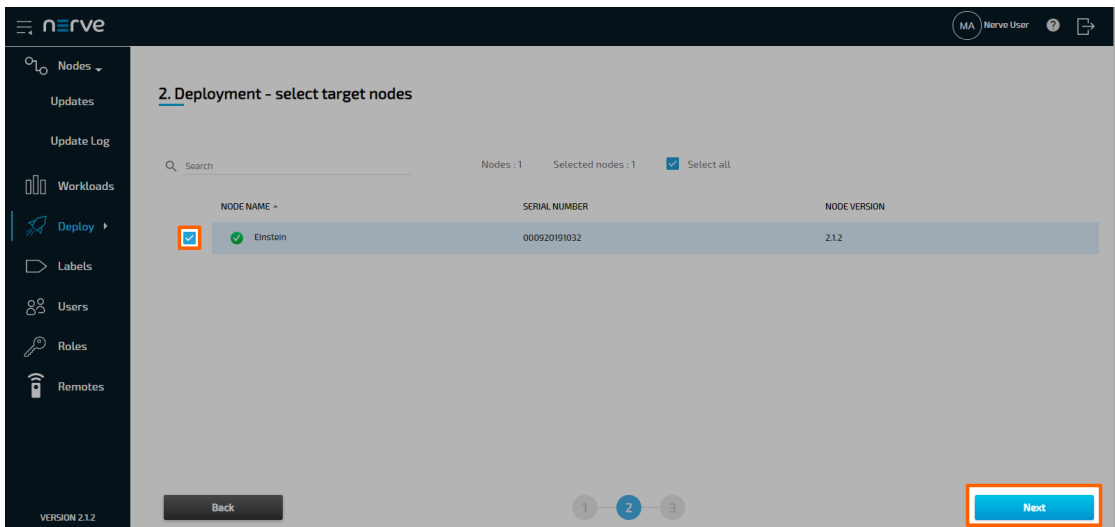
1. Log in to the Management System.
2. Select **Deploy** in the navigation on the left.



3. Select the orange Docker tab. A list of Docker workloads will appear below.
4. Select **Node-RED** in the list of workloads. A list of version of this workload will appear to the right.
5. Select the latest version of the workload.



6. Select **Next** in the lower-right corner.
7. Tick the checkbox next to your node.
8. Select **Next** in the lower-right corner.



9. Select **Deploy** to execute the deployment.

*Optional: Enter a Deploy name above the Summary of the workload to make this deployment easy to identify. A timestamp is filled in automatically.*


The deployment should now be visible at the top of the deployment log. Click the log entry of the deployment to see a more detailed view.

To confirm if the workloads have been deployed successfully, select **Nodes** in the navigation on the left. Select your node in the node tree and confirm if two workload tiles are showing underneath the bar graph. The workloads should show the status **STARTED**.

#### NOTE

It can take a number of seconds until the status of workloads switches from **IDLE** to **STARTED** after deployment.

**Einstein** [000920191032]



CPU load: 12%

Memory usage: 33%

Docker used space: 2.8/12.0 GB, 23%

VM used space: 13.8/247.3 GB, 5%

NODE VERSION:  
2.1.2

CREATED:  
04-12-2020 11:26:51 AM

WAN ADDRESS:  
192.168.0.142

LABELS:  
No Labels

CONNECT

SHOW INTERNAL LOGS

DELETE NODE

Machine Simulation Status: STARTED

Node-RED Status: STARTED

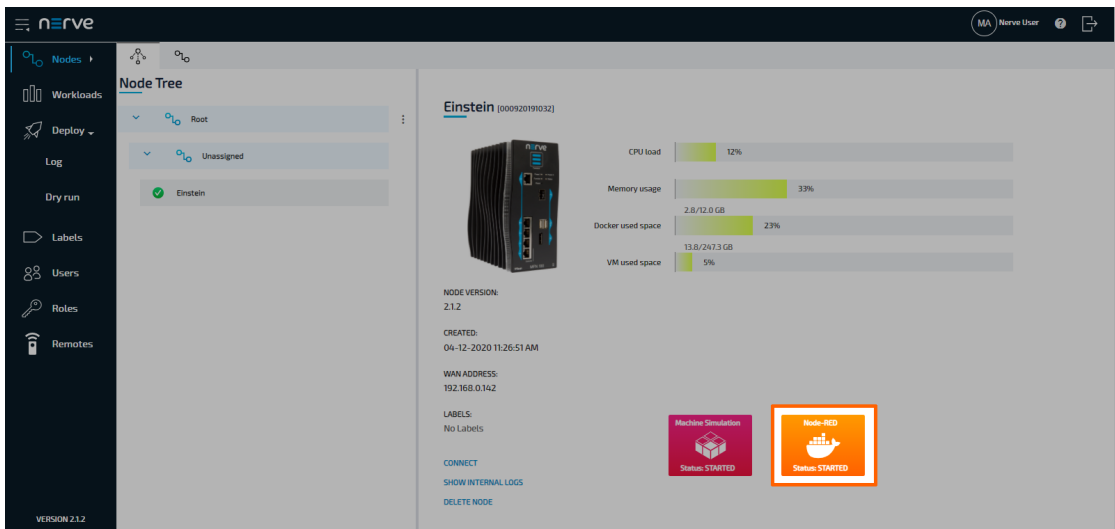
## Connecting to Node-RED

After deployment, Node-RED is accessible at port 1880 on the Nerve Device through a web user interface. The pre-configured remote tunnel will use this port and port 1880 on the local computer to create a connection between the computer and the Nerve Device to access the interface of Node-RED. Before continuing, make sure the Nerve Connection Manager that was part of the delivery is installed, as it is required for using remote tunnels.

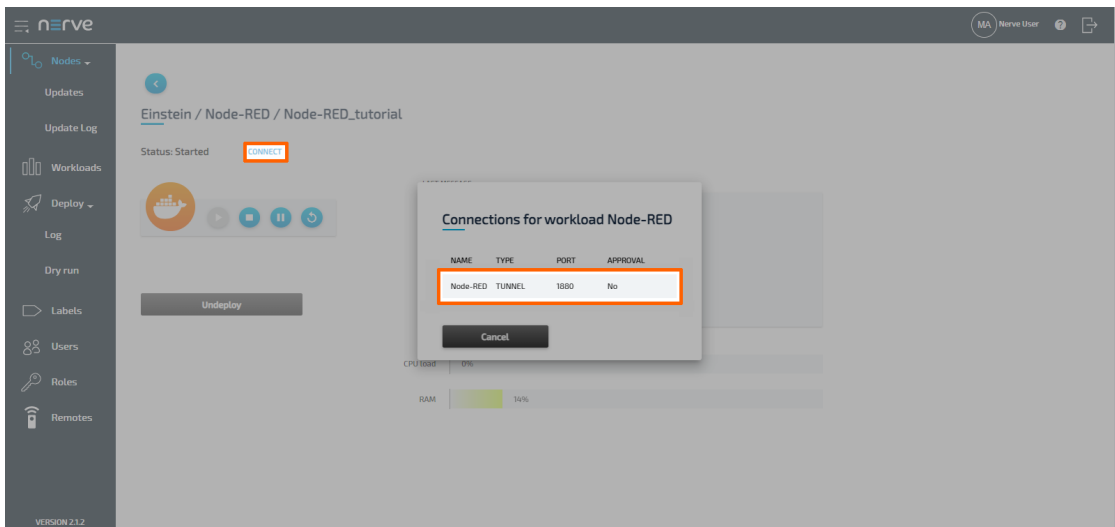
### NOTE

Local admin rights might be required to successfully install the Nerve Connection Manager. In case a remote tunnel is not pre-configured, refer to [Remote tunnels](#) for more information on remote tunnels.

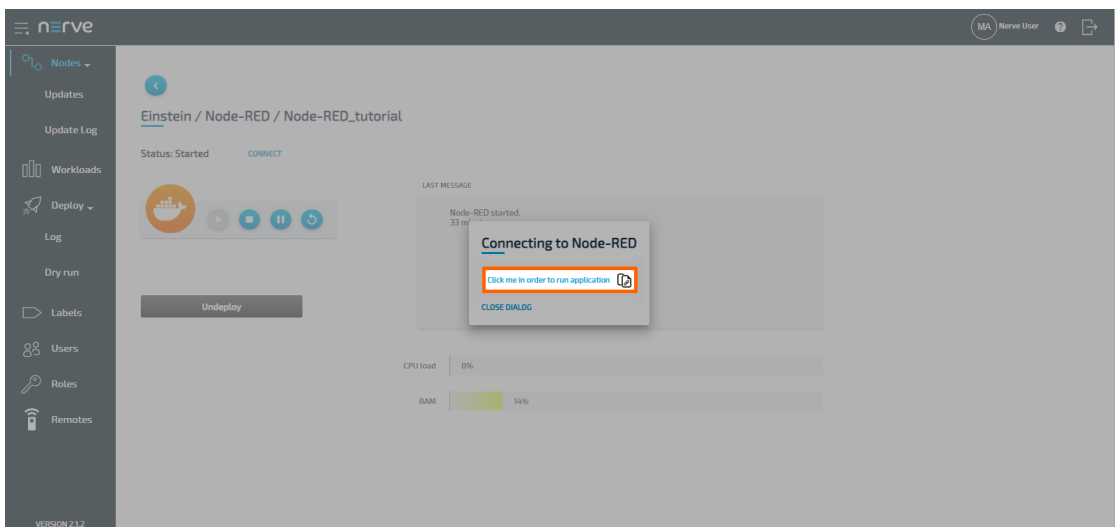
1. Select **Nodes** in the navigation on the left.
2. Select the node with the deployed Node-RED workload in the node tree.
3. Select the Node-RED Docker workload.



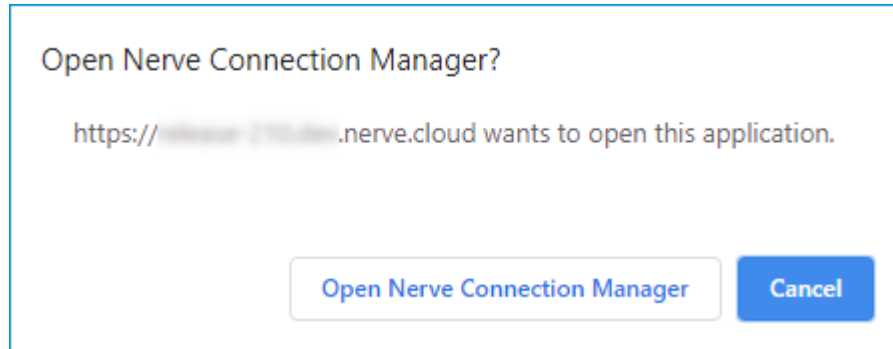
4. Select **CONNECT** next to the workload status. Available connections will appear in a window.
5. Select the **Node-RED** remote tunnel from the list.



6. Select **Click me in order to run application** in the new window.



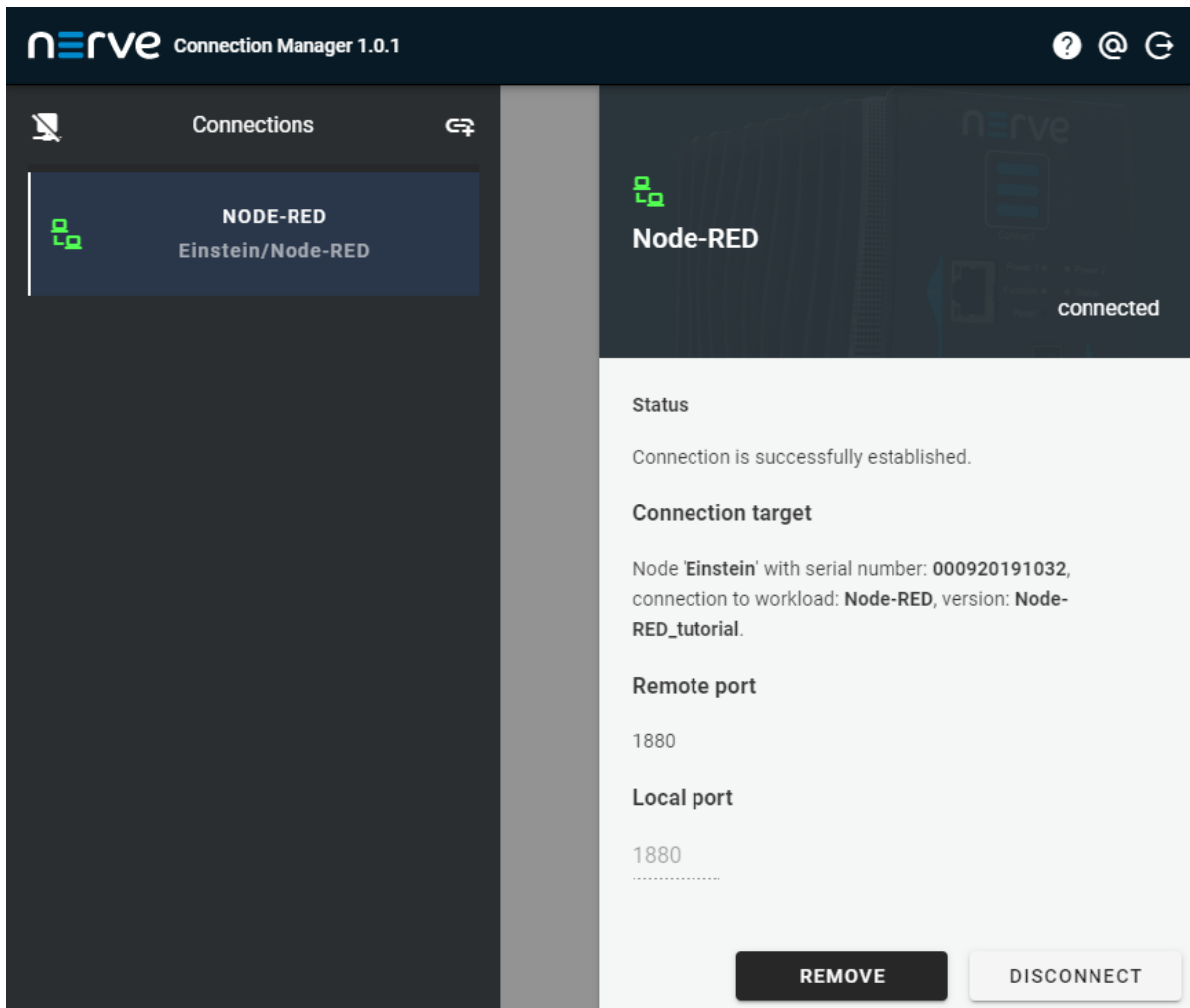
- If the Nerve Connection Manager installed correctly, confirm the browser message
7. that the Nerve Connection Manager shall be opened.  
Depending on the browser that is used, this message will differ. The Nerve Connection Manager will start automatically once the message is confirmed.



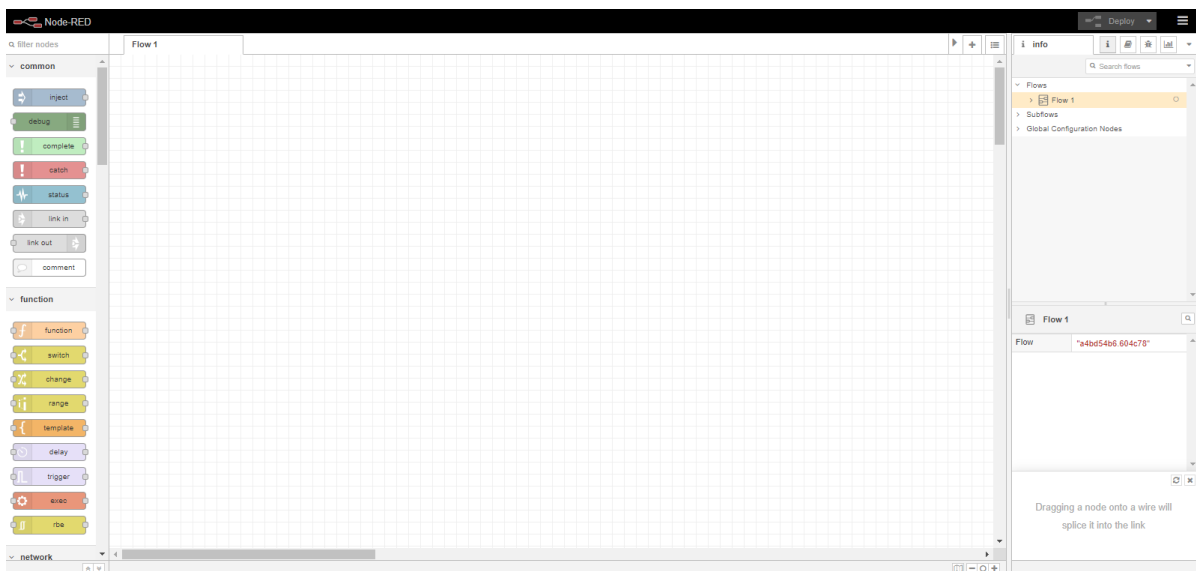
**NOTE**

If the Nerve Connection Manager does not start automatically, refer to [Using a remote tunnel to a workload](#) in the user guide.

The remote tunnel will be established once the Nerve Connection Manager starts and the remote connection will turn green in the Nerve Connection Manager. The remote tunnel is now ready to be used.



Open a new browser tab and enter localhost:1880 to open Node-RED.



## Examples of using Node-RED with Nerve

The following is a list of documentation examples of how Node-RED is used with Nerve. New examples will be added in the near future.

## Tutorial: Machine efficiency insight

Node-RED is part of the Nerve trial, which demonstrates how to gain insight on machine efficiency. There, Node-RED is used to analyze machine performance and display machine efficiency data in an OEE and quality dashboard. Refer to [Tutorial: Machine efficiency insight](#) for more information.

# Security Recommendations

## Security recommendations

In order for Nerve to operate in a secure manner, there is a set of measures that needs to be taken by implementers of Nerve. The information below is split into secure installation, operation and disposal, each focusing on the measures that need to be taken. The current state of the Nerve system in regards to the measure is also summarized.

### Secure installation

The measures described here need to be taken into consideration in the planning phase of the environment.

#### Installation

##### Measure

Provide physical protection against physical access to the device to avoid an unauthorized user accessing sensible data on the disk.

#### Network connection

On the node the **wan** interface is used for the connection to the Management System. The communication towards the Management System uses HTTPS to ensure protection and is always initiated from the node, never from the Management System.

##### Measure

Place the node behind a firewall allowing access to port 443. If workloads provide access to additional ports, the workloads should be hardened to prevent unauthorized access and the firewall configuration should be adapted.

To ensure compatibility with fieldbus (Profinet, Modbus), communication towards the machine network is not encrypted.

##### Measure

Limit physical access to the network cables in order to protect the network within the machine. Whenever possible, select a secure connection to devices.

By default the networks in Nerve are isolated and no monitoring is implemented. Island mode can be achieved by resetting the network configuration on the node.



## Measure

Ensure a process during network design to limit traffic crossing boundaries to the strict minimum.

## Secure operation

The measures described here need to be taken into consideration when setting up the Nerve system for operation and during the operation of the system.

### User management and permissions

In Nerve authentication is implemented in the Management System and on the node using email and password. The user accounts for the node and the Management System are not synchronized and must be managed separately.

The Management System can use its own user management system or can be connected to an LDAP server providing the necessary information for authentication and allowing the use of an already implemented password policy. In case of delegation, the LDAP groups can be mapped to the roles in the Management System. The nodes cannot be connected to an external user management system.

A minimum set of password requirements is used in the Management System. The lifetime of a password cannot be enforced in the Management System, as password lifetime restrictions are not considered best practice anymore. The Management System enforces the following password policy:

- At least one uppercase letter
- At least one lowercase letter
- At least one number
- Minimum length of seven characters

Stronger password requirements or lifetime restrictions can be done by delegating authentication to an LDAP or Active Directory server with the desired configuration. Also, support the use of long passwords by deploying appropriate password stores.

## Measures

- Select local user management or LDAP server synchronization based on the available infrastructure, security policy and password policy.
- Periodically review all user accounts and remove the ones which are not needed anymore.

The Management System does not have a default password. When users are created they receive an email instructing them to create a new password on a dedicated page. Passwords are transmitted only over HTTPS and not stored in clear text.

The Nerve system implements user management, associated with a fine-grained Role Base Access Control system (RBAC). This is meant to prevent unauthorized operations by following the least privileged rule.

## Measures

- Implement a process to define the roles as needed for operation (e.g. admin, operator, etc.) with the minimum possible number of access rights. The permissions for each role and the assignment of the roles to the different users should be reviewed periodically, at least once a year.
- Consider the necessity for user access to audit logs when defining roles and permissions.

## Management System operations

The Management System needs certificates to encrypt communication with nodes over HTTPS. The node is based on Debian and only accepts the certificate authority root certificates provided by Debian. The certificates used by the Management System are provided by the user when deployed on premise or by LetsEncrypt when hosted by TTTech Industrial.

## Measures

- Ensure a periodic (e.g. 90 days) and timely renewal of the certificates of the Management System if the Management System is hosted on premise.
- Develop and implement a backup mechanism for the Docker volumes of the Management System when operating the Management System on premise.

## Node operations

### Measure

Integrate the onboarding of a Nerve node into the machine commissioning procedure. Add a manual verification of the serial number to the procedure.

The node authenticates itself to the Management System using a serial number as the identifier and a secure ID as the password.

The number of [remote sessions](#) to the node can be limited. The limit of sessions is configurable per connection.

### Measure

Implement a process to define and enforce a limited number of parallel sessions for each connection in the Management System.

The Nerve node can be configured to [collect logs from Docker workloads](#).

### Measure

Implement adequate logging when creating control workloads.

## Secure disposal

The disk used by the Nerve node stores information in clear text, no encryption is used.

## Measure

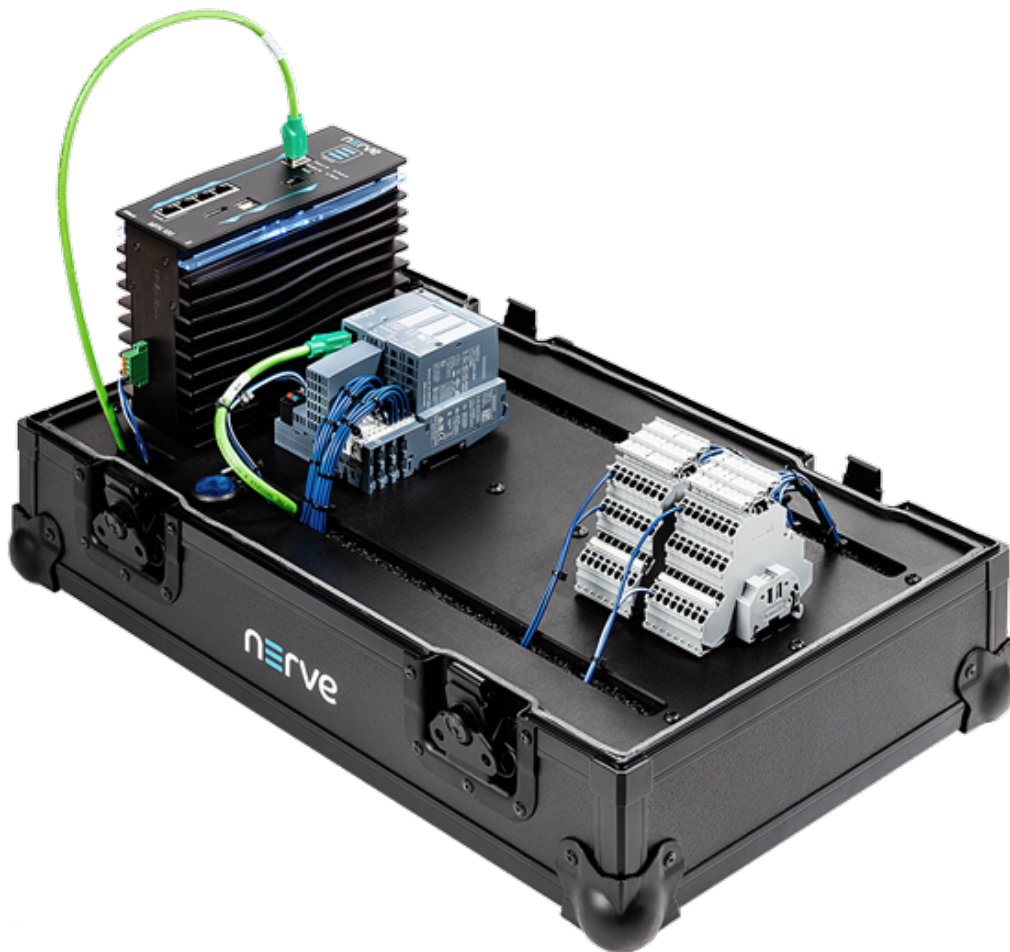
Remove the disk from the node before disposal and wipe it with adequate tools or ensure physical destruction.

## Release Notes

### Nerve

### Nerve Kit

### Nerve Kit



The Nerve Kit is a ready-to-use hardware and software package, designed to support learning and training needs around digitalization. The kit provides everything needed to start collecting, storing and analyzing data, and can be expanded with custom sensors and IOs e.g. IO Link Master. Nerve provides a virtualized software environment, enabling the implementation and remote management of multiple containers or virtual machines for specific use case. In addition, the kit can be used to connect to real-time data sources and update real-time control applications remotely via the Management System.

## Key features

The Nerve Kit is fully integrated in a robust, portable case suitable for lab use and demonstrations. Start with default control applications, then develop and deploy custom applications and extend the kit's functionality by adding sensors and actuators.

### Plug and play

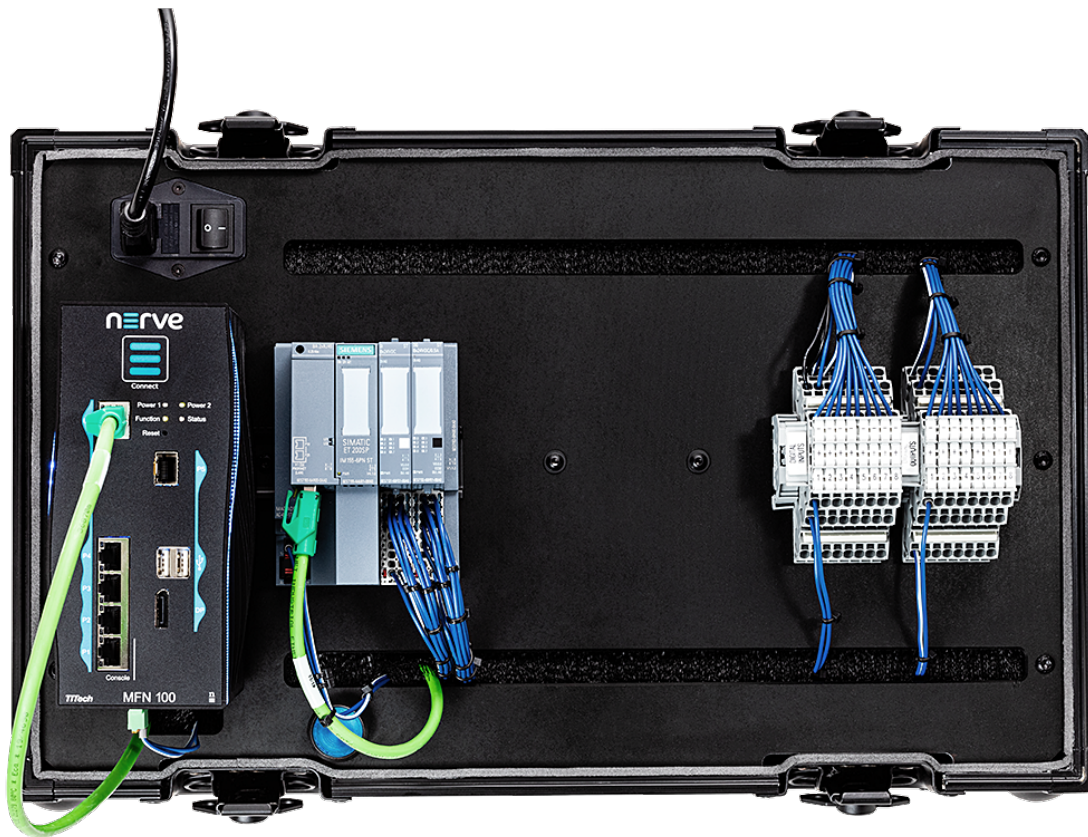
- Fully configured Data Services for ingesting PROFINET, MQTT and OPC UA data
- Two pre-installed CODESYS control applications for demo purposes
- Pre-configured Grafana dashboard for visualization of data (available locally and in the cloud)
- Workloads (CODESYS programs) available for deployment from the Management System to the Nerve Device

### Extendable

- Extra digital I/Os available for adding sensors to extend functionality
- Data Services can be configured for new data streams including EtherCAT and Modbus TCP/IP
- Customizable Grafana dashboard for visualization of new sensor data
- Newly created workloads (virtual machines / containers / CODESYS programs) can be deployed from the Management System to the Nerve Device

## Nerve Kit contents

The contents of the kit are delivered in two separate boxes, a small box and a large box. The small box contains the MFN 100 including the Hardware Installation Guide and a mating connector. The large box has the remaining components of the kit: the mounting plate, the SIMATIC I/O module and the digital I/O terminal block.



A power cord and two network cables are also required to finish the setup.

## Content overview

### Hardware

- MFN 100-C64xx
- SIMATIC ET200SP I/O system including:
  - 1x SIMATIC ET 200SP bundle PROFINET IM, IM 155-6PN ST
  - 1x SIMATIC ET 200SP 8x24V DC digital input
  - 1x SIMATIC ET 200SP 8x24V DC digital output
- Digital I/O terminal block
- Illuminated push button
- 24V DC Power supply
- 1x Mating connector
- 1x Network cable

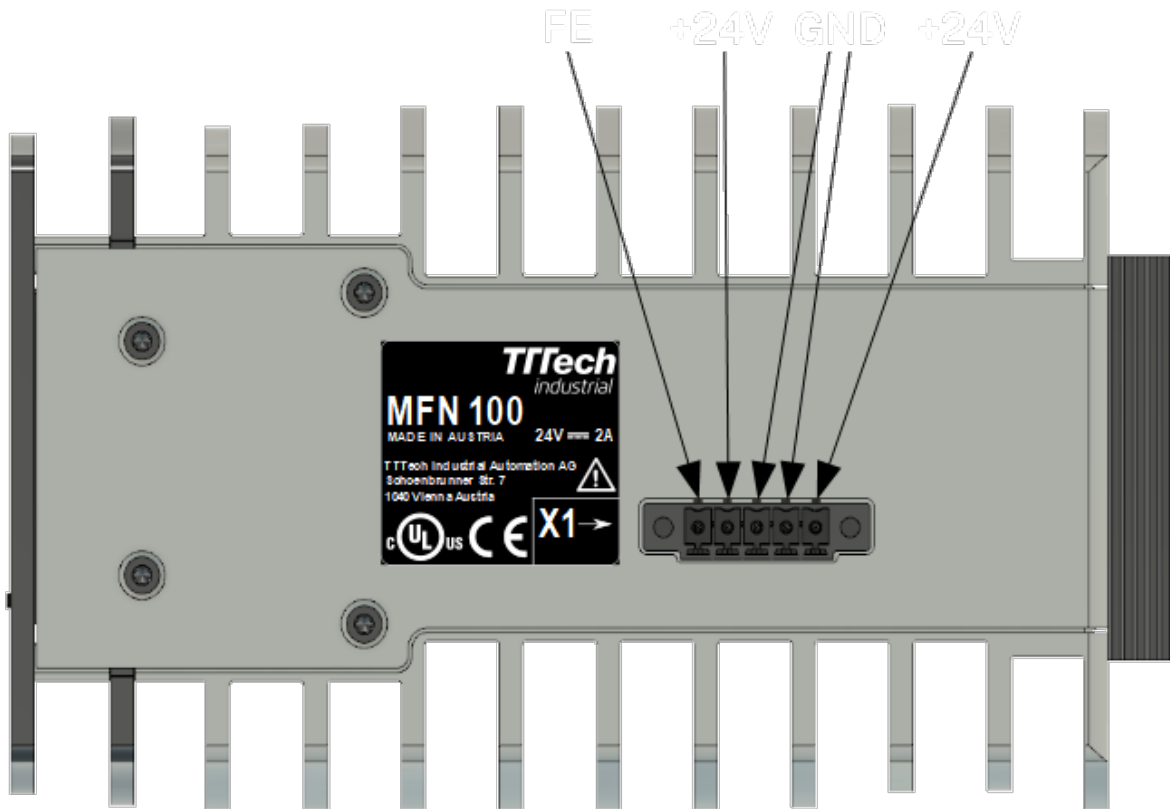
### Software

- MFN 100-C64xx device description file
- SIMATIC ET200SP I/O device description file
- Two default CODESYS projects

The device descriptions and the default CODESYS applications can be downloaded from the [Nerve Software Center](#). The login credentials for the Nerve Software Center can be found in the customer profile that has been sent as part of the delivery.

## Power connectors overview

The power connectors are located at the bottom of the MFN 100 next to the label. There are two separate 24 V inputs, two GND inputs and one Functional Earth (FE) input. The inputs are fused internally. The fuse cannot be replaced by the user. The power supply inputs are protected against reverse polarity.



Pin	Description
1	Power supply line 2
2	GND
3	GND
4	Power supply line 1
5	Functional Earth (FE)

### NOTE

The GND and FE pins (pins 2, 3, and 5) are electrically connected to the housing.

## Setup and default operation

This section covers the hardware setup, explains the default operation of the kit and gives an introduction to the Management System.

## Setting up the hardware

The MFN 100 is delivered in a separate box and has to be mounted on the kit. A mating connector is already connected to the power supply. Also, the SIMATIC ET 200SP I/O module is already connected to the I/O port of the MFN 100 with a network cable. In order to completely set up the kit, a power cord and two network cables are also required.

1. Mount the MFN 100 on the DIN rail on the left side of the kit.

### NOTE

For help with mounting the MFN 100 on the DIN rail, refer to the Hardware Installation guide enclosed in the box of the MFN 100.

2. Plug the mating connector that is connected to the power supply into the bottom side of the MFN 100.
3. Connect port 2 of the MFN to a DHCP-enabled network with access to the Management System or internet access if the Management System is hosted by TTTech Industrial.

### NOTE

Contact the IT administrator for help on how to allow external devices to connect to the network.

4. Connect the power cord to the power supply and to a power outlet. Make sure the power supply is turned off.
5. Push the button to switch on the power supply.

The MFN 100 will start after a few minutes and the blue light will turn on. All necessary services are initiated and after that, data is sent to the Management System.

## Default operation

The Nerve Kit comes with two default applications which can replicate scenarios found in industrial automation. The applications allow for data to be generated at the I/O, be acquired by the MFN 100 via Ethernet fieldbus, translated to OPC UA and be sent to the cloud for visualization. The data can be generated as a continuous flow throughout the cycle, as a regular data flow, or as irregular event based data flow. These different data flows mimic various types of industrial process and show the functionality of the Data Services from I/O to cloud.

The applications use the push button that is connected to the SIMATIC ET200 SP I/O module via the terminal block. Data is sent between the SIMATIC I/O module and the MFN 100 I/O port via PROFINET and the illumination of the button is controlled via the CODESYS soft PLC running on the MFN 100.

The two default CODESYS applications: **app1** and **app2** are used to alter flow of data from the I/O modules. Both apps control the illumination of the button and record two values: `iCountNumber` and `iCountButton`.

	app1	app2
<p><b>iCountNumber</b> (continuous data flow from the application to the Management System)</p>	<ul style="list-style-type: none"> <li>• Continuous count upwards from 0 to 1000.</li> <li>• Resets to 0 automatically when it has reached 1000.</li> </ul>	<ul style="list-style-type: none"> <li>• Continuous count downwards from 1000 to 0.</li> <li>• Resets to 1000 automatically when it has reached 0.</li> </ul>
<p><b>iCountButton</b> (irregular and regular data flow from the application to the Management System)</p>	<ul style="list-style-type: none"> <li>• Counts the number of times that the button has been pressed (irregular data flow).</li> <li>• The push button lights up after 10 button presses.</li> <li>• The value resets to 0 when the button is held for two seconds. The light of the button goes out.</li> </ul>	<ul style="list-style-type: none"> <li>• Counts the duration that the button is held (regular data flow).</li> <li>• The counter continuously increases from 0. The push button lights up when the value reaches 25.</li> <li>• The value continuously decreases to 0 when the button is released.</li> </ul>

In both application modes the calculated values are sent to the Management System and displayed in Grafana for visualization. The Management System allows users to deploy either of the default applications from the repository to the MFN 100. This demonstrates the ability to update CODESYS applications remotely and alter applications running on machines wherever they are in the world.

Beyond the two default applications, the kit can be used to develop custom CODESYS applications, which can be uploaded to the repository in the Management System and then deployed to the MFN 100. These applications can then be used in conjunction with corresponding actuators and/or sensors.

#### NOTE

The default application **app1** is loaded automatically during the initial startup of the MFN 100.

## Connecting to the Management System

The Nerve Management System is a web-based service that permits management of Nerve nodes that are registered.

#### NOTE

Google Chrome or Firefox Version 63 or later are recommended for the usage of the Management System.

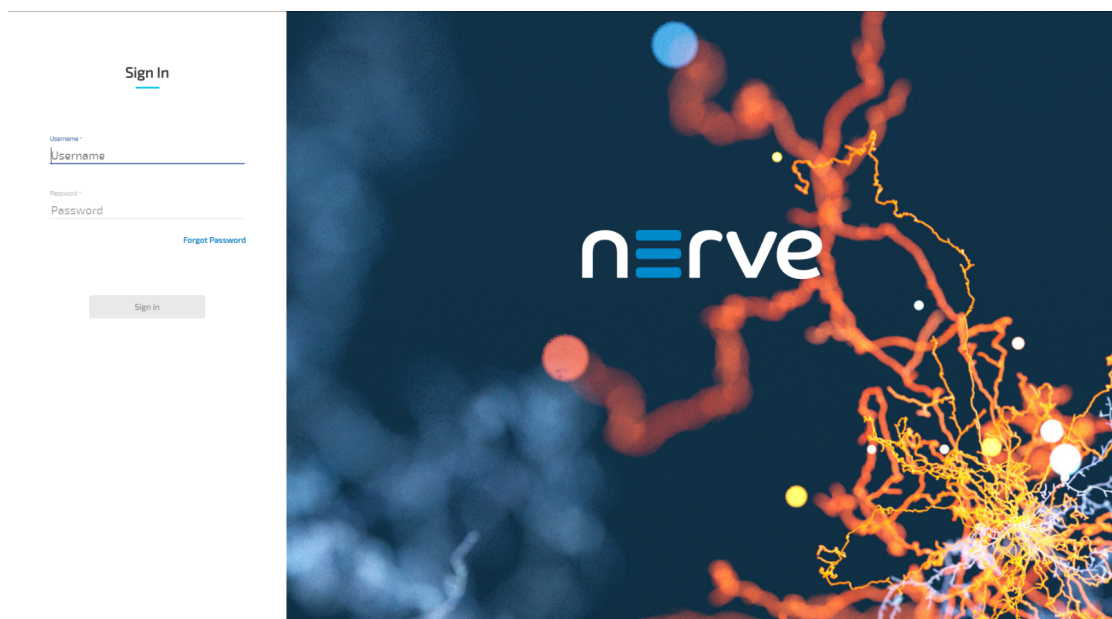


Before connecting, make sure that the MFN 100 of the starter kit is connected to the network through port 2 and that an IP address has been assigned by the DHCP server. Contact the IT administrator for help with assigning an IP address. The login credentials for the Management System are in the customer profile. The customer profile has been sent in form of a PDF as part of the delivery.

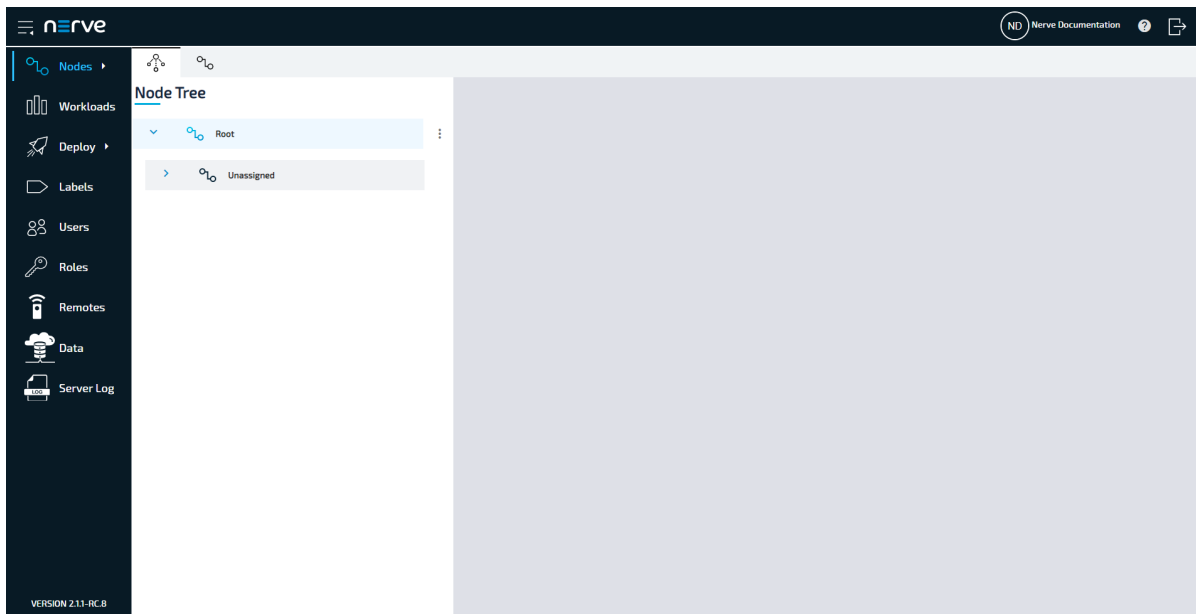
#### NOTE

If a customer profile has not been part of the delivery, contact a sales representative or TTTech Industrial customer support at [support@tttech-industrial.com](mailto:support@tttech-industrial.com).

1. Go to the URL of the Management System in the customer profile.
2. Log in with the credentials provided in the customer profile.



The Management System will show the node tree by default.



One element in the node tree is already created at first login, containing the MFN 100 of the Nerve Kit. All newly registered nodes will be located under **Root > Unassigned** by default.

Manage nodes, provision workloads and deploy workloads among other options from here. Refer to the [user guide](#) for more information on the Management System.

#### NOTE

Port 443 (HTTPS) and port 8883 (MQTTS) of the corporate firewall have to be open for communication between nodes and the Management System.

## Moving a node from one tree element to another

Moving nodes in the node tree is possible by drag and drop. Make sure to create a new tree element before attempting to move a node.

1. Select **Nodes** in the navigation on the left.
2. Select the node tree tab



- on the right.
3. Expand the tree element of the node that will be moved. The default element is **Root > Unassigned**.
4. Choose the node to move.
5. Drag and drop the node to the newly created element. Elements expand automatically when they are hovered over.
6. Select **APPLY CHANGES (n)** in the upper-right corner of the node tree.

## NOTE

(n) is a placeholder for the number of changes made to the node tree. For three performed changes, (3) will be displayed in the button above the node tree.

The node has now been moved to the new element. Note that a node cannot be moved back to **Unassigned** once it has been moved to another element.

## Viewing data in the Management System

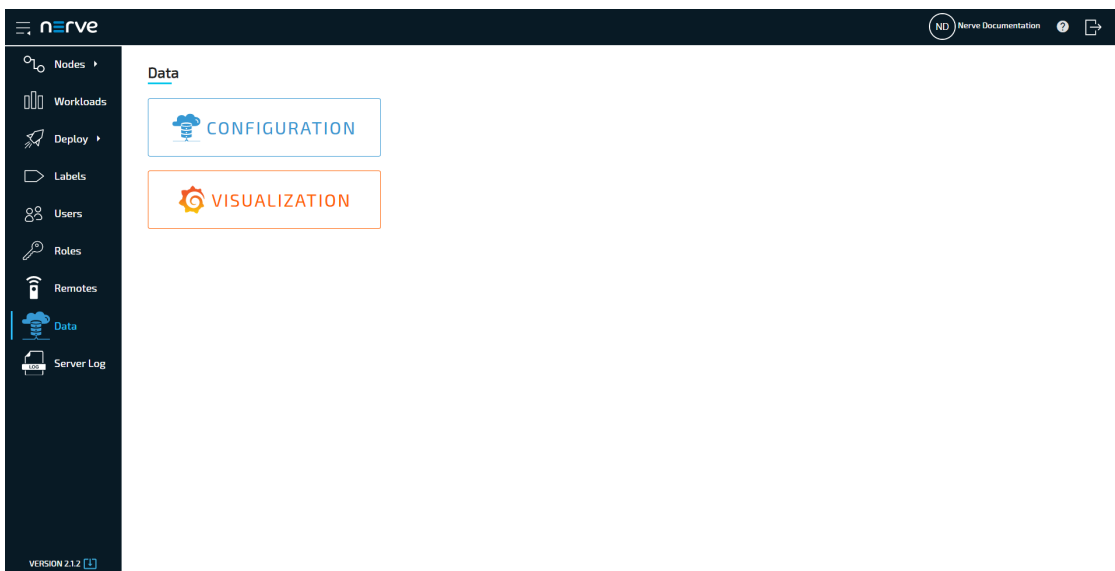
Data that is being sent to the Management System can be viewed through the Grafana element in the Data Services instance of the Management System. The default values `iCountNumber` and `iCountButton` are set up by default. The URL of the Management System and the login credentials can be found in the customer profile. It has been sent in form of a PDF as part of the delivery.

1. Log in to the Management System.
2. Select **Data** in the navigation on the left.

## NOTE

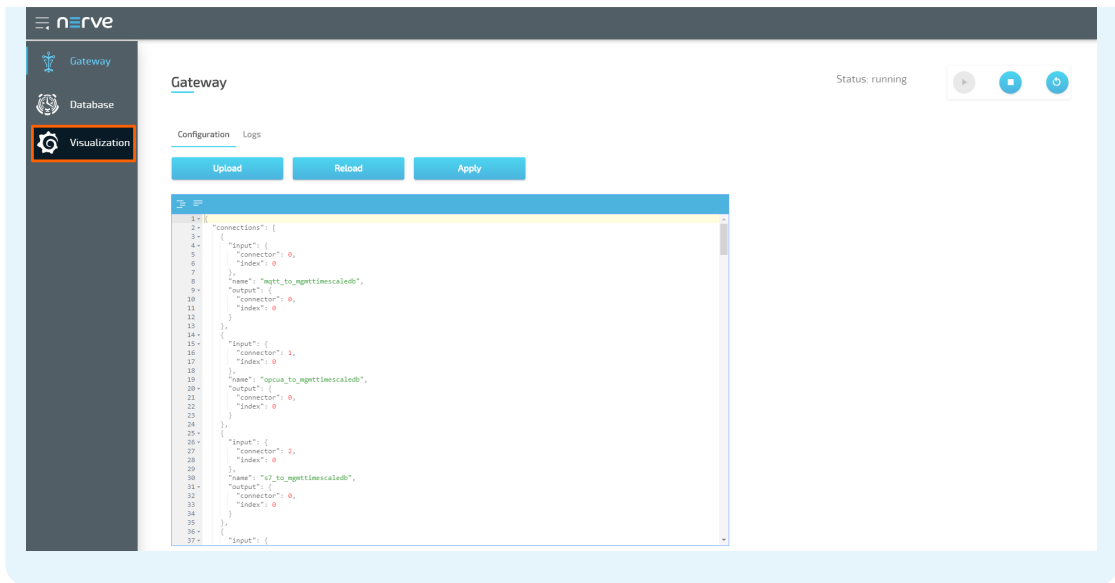
If the menu item **Data** is not available, make sure the logged in user has the permission to access the Data Services. Refer to [Assigning a role to a user](#) for more information.

3. Select **Data**.

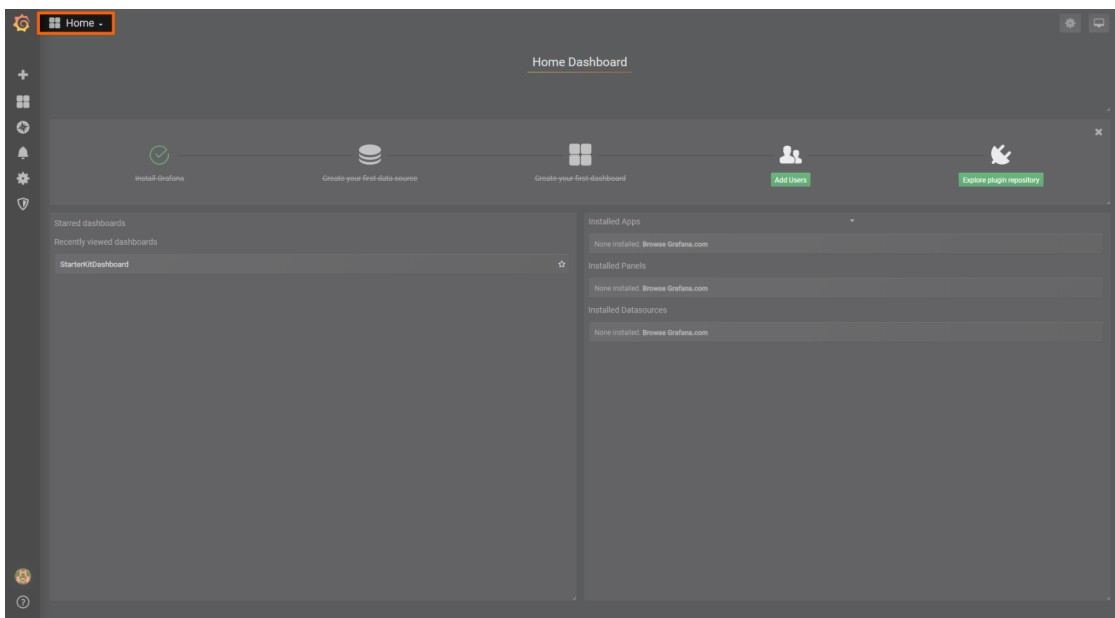


## NOTE

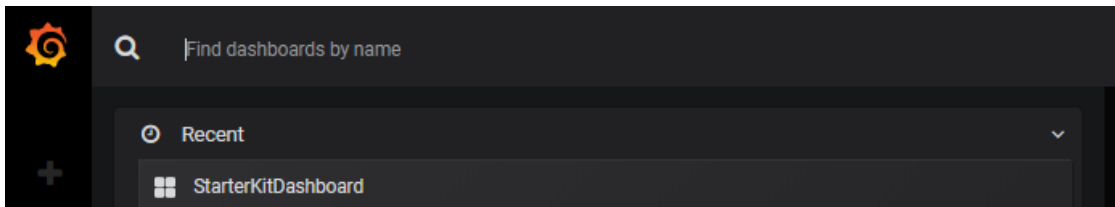
The visualization element can also be reached from the Data Services UI. When in the Data Services UI, select **Data** in the navigation on the left and select **Open** to reach the Grafana UI.



4. Select **Home** in the upper-left corner.



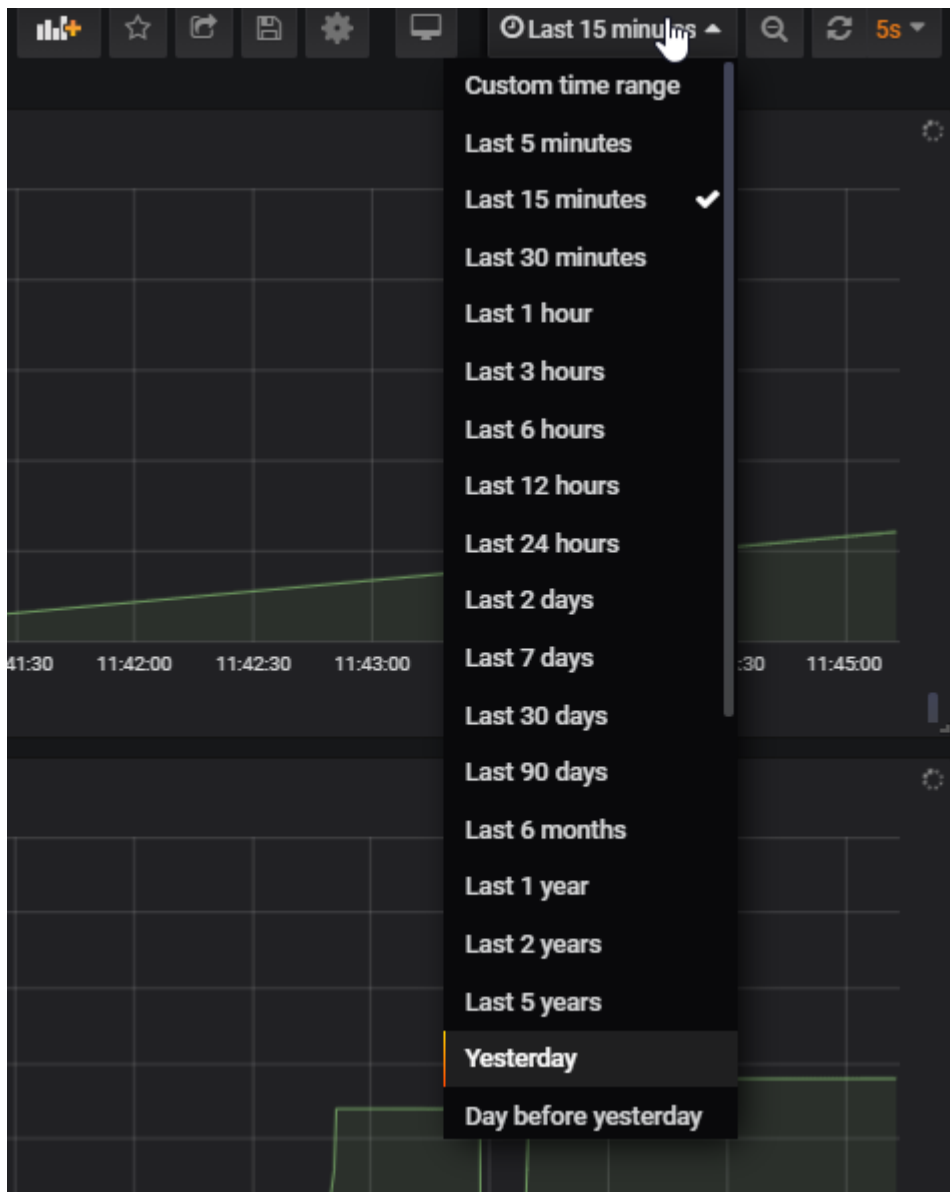
5. Select **Nerve Kit** underneath the search bar.



This is the visualization of live data from the MFN 100. The calculated values, `iCountNumber` and `iCountButton`, that are generated in the default CODESYS applications **app1** and **app2** are displayed here by default.



The data automatically updates every 5 seconds. To change the update rate and the visible time range, access the settings by clicking the clock symbol in the upper-right.



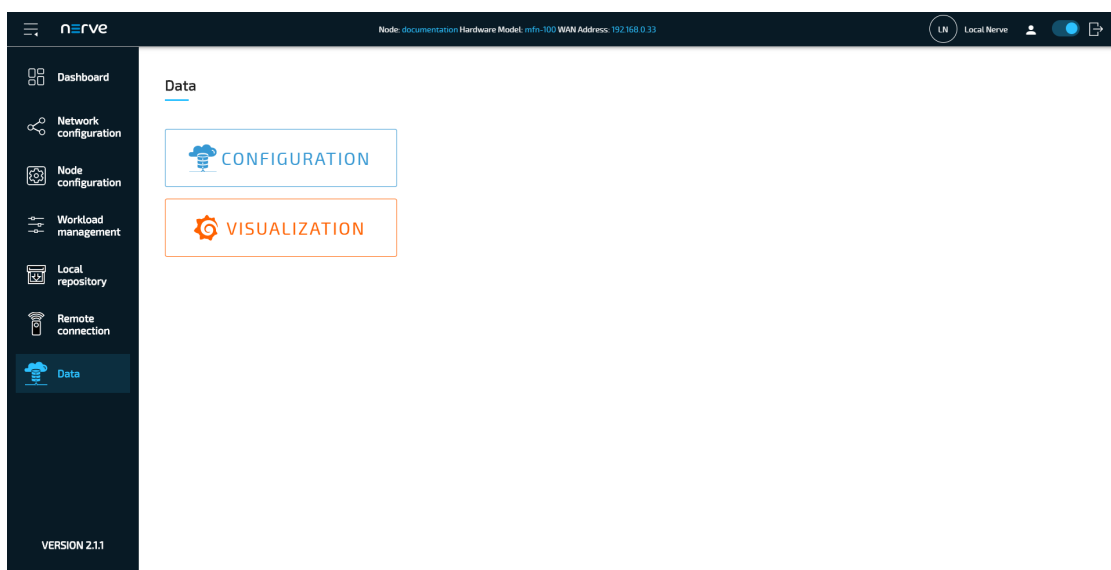
## Viewing data on the Nerve Device

Data can also be viewed locally on the Nerve Device. The default values `iCountNumber` and `iCountButton` are set up by default.

1. Connect the workstation to the console port **P1** of the MFN 100.
2. Configure the network adapter through which the workstation is connected to the MFN 100 the following way:

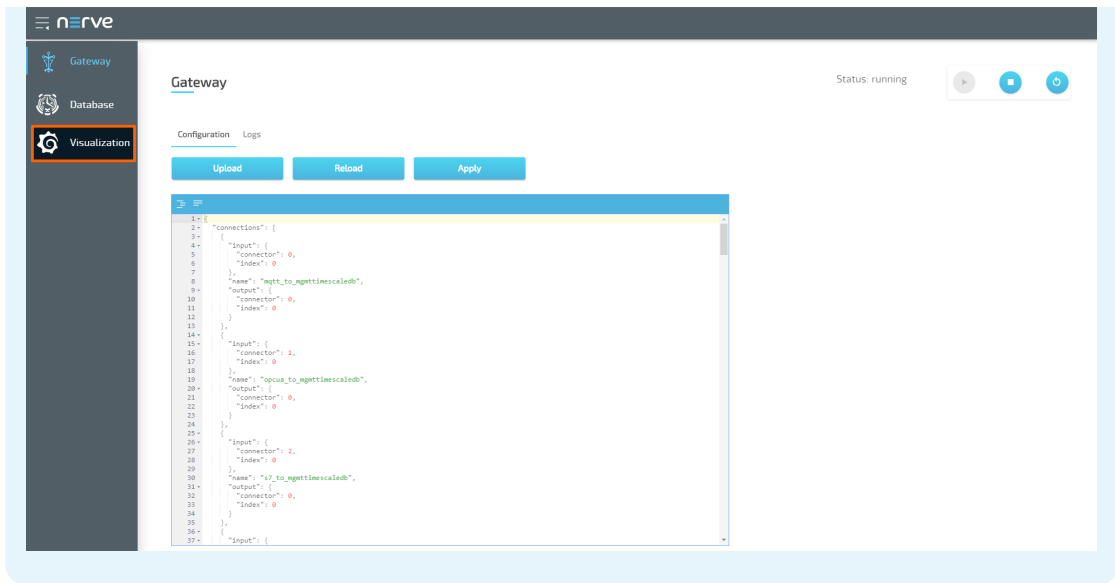
<b>IP address</b>	172.20.2.90
<b>Subnet mask</b>	255.255.255.0

3. Follow this link to reach the Local UI: <http://172.20.2.1:3333>.
4. Log in with the credentials from the customer profile.
5. Select **Data** in the navigation on the left.
6. Select **Data**.

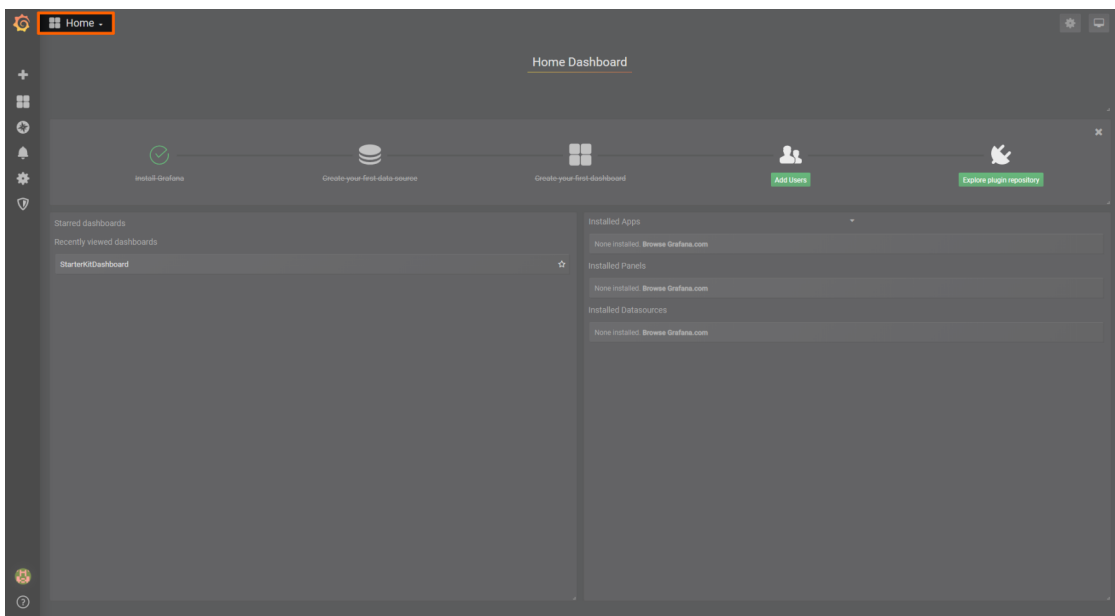


### NOTE

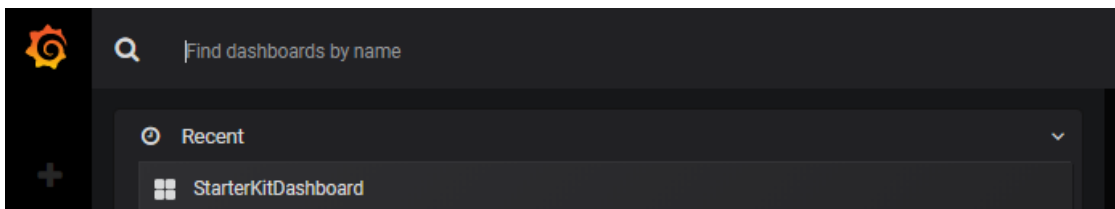
The visualization element can also be reached from the Data Services UI. When in the Data Services UI, select **Data** in the navigation on the left and select **Open** to reach the Grafana UI.



7. Select **Home** in the upper-left corner.



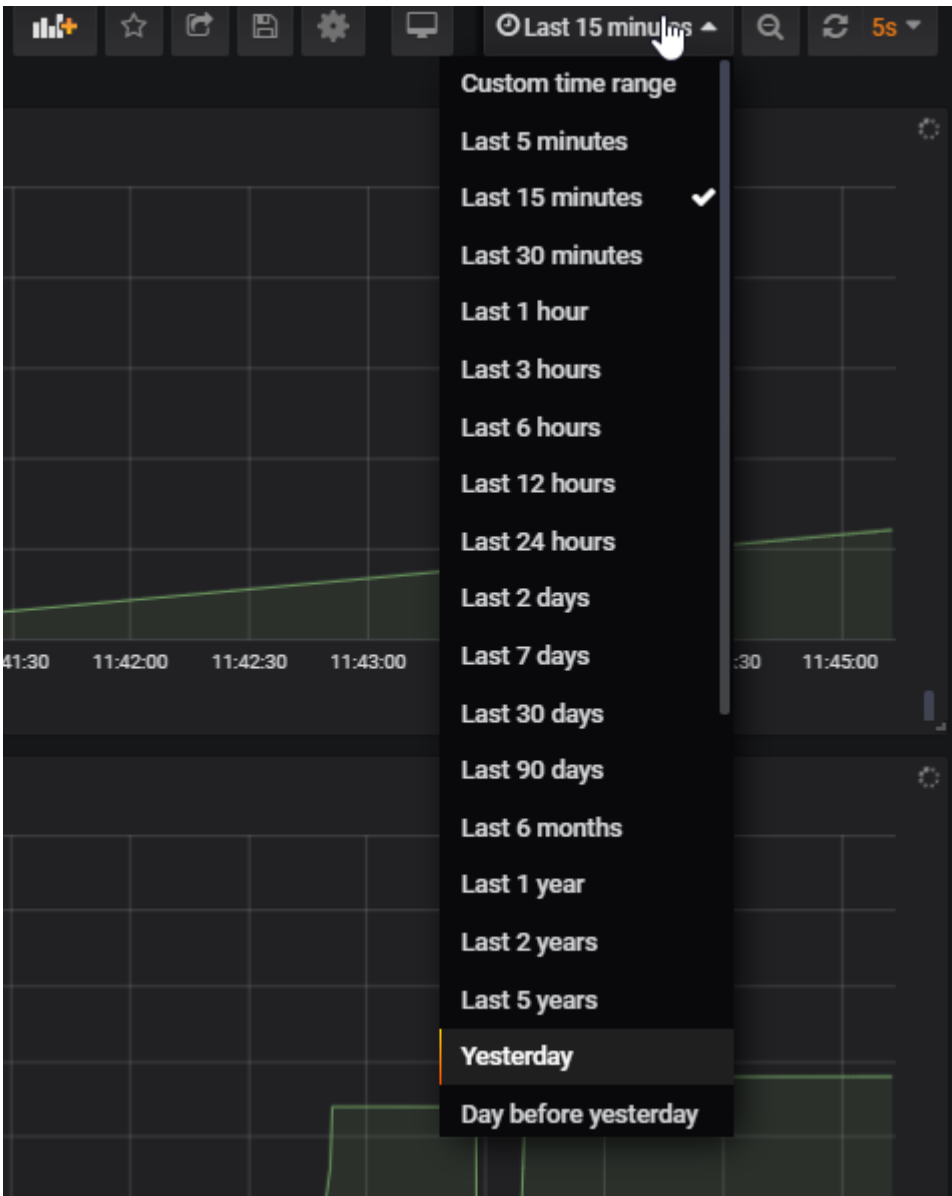
8. Select **Nerve Kit** underneath the search bar.



This is the visualization of live data from the MFN 100. The calculated values, `iCountNumber` and `iCountButton`, that are generated in the default CODESYS applications **app1** and **app2** are displayed here by default.



The data automatically updates every 5 seconds. To change the update rate and the visible time range, access the settings by clicking the clock symbol in the upper-right.





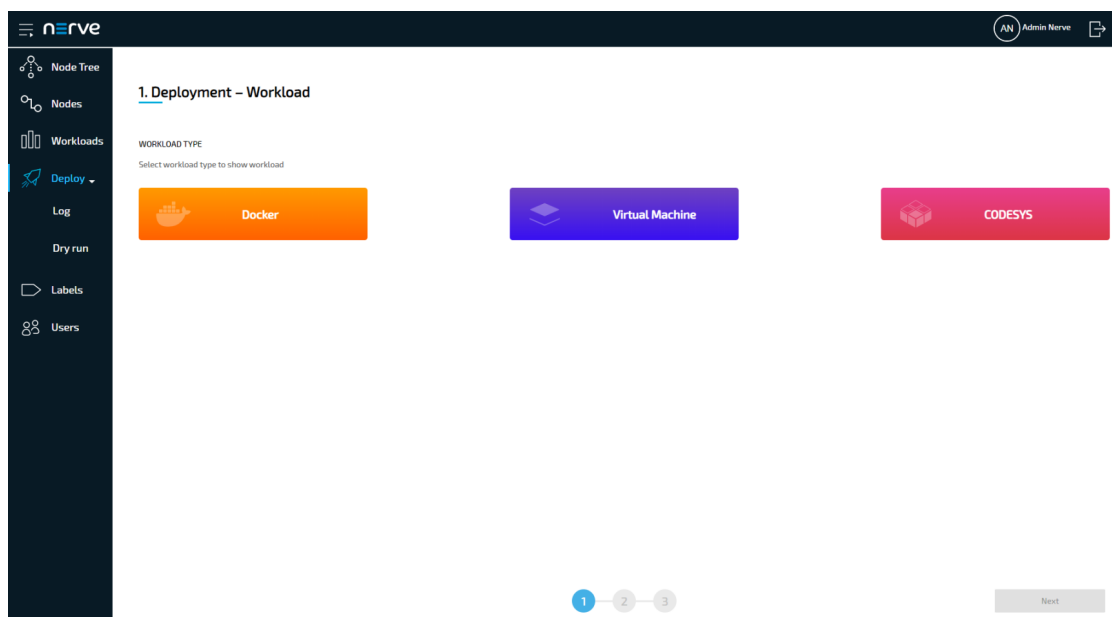
# Downloading & uploading CODESYS applications from the Management System

With the kit CODESYS applications can be deployed from the Management System to the MFN 100. The two default CODESYS applications are already available in the Management System. Own applications can also be uploaded to the repository in the Management System and made ready for deployment.

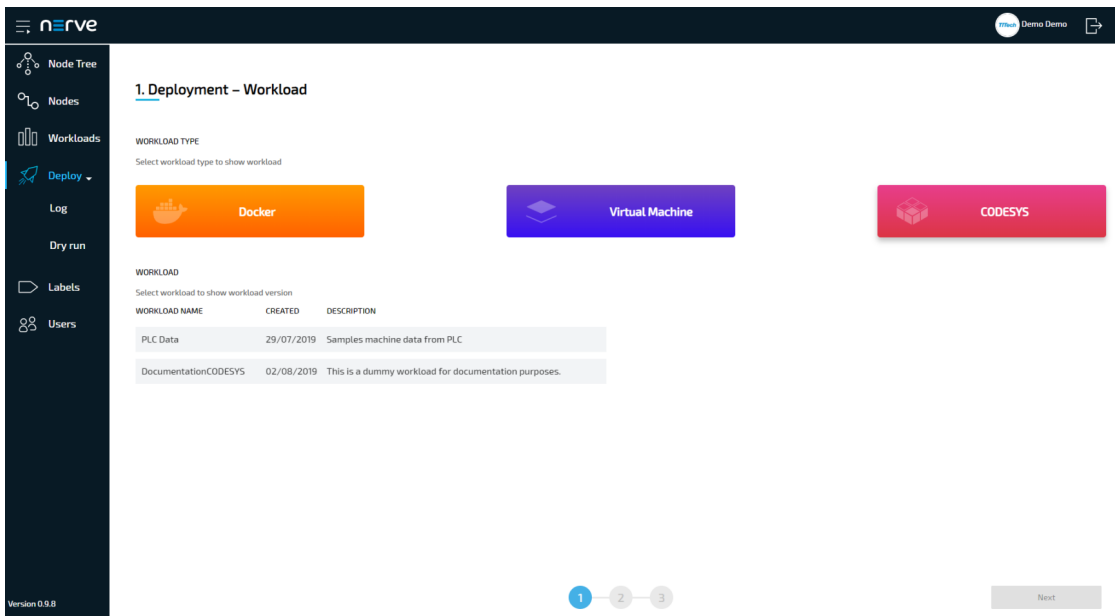
## Deploying a CODESYS workload

One CODESYS workload is available with first login that can be used with the kit immediately. The workload is named **Nerve Kit** and it has two versions: **Nerve Starter Kit App1** and **Nerve Starter Kit App2**. For more information on the deployment process for all workload types refer to the [user guide](#).

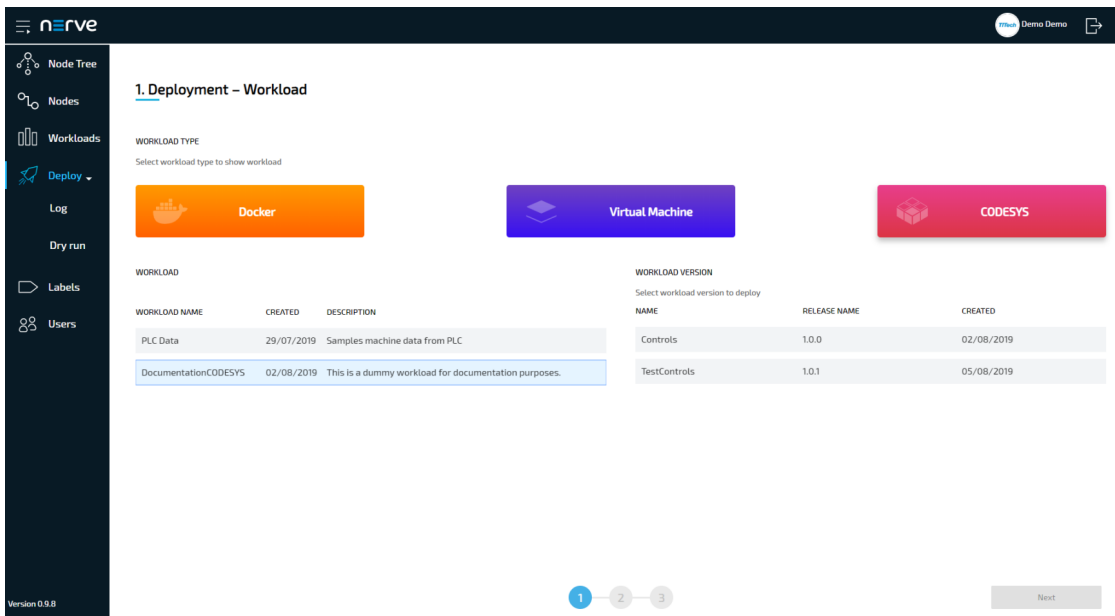
1. Select **Deploy** in the left-hand menu.



2. Select the CODESYS workload icon on the right. A list of CODESYS workloads will appear below.

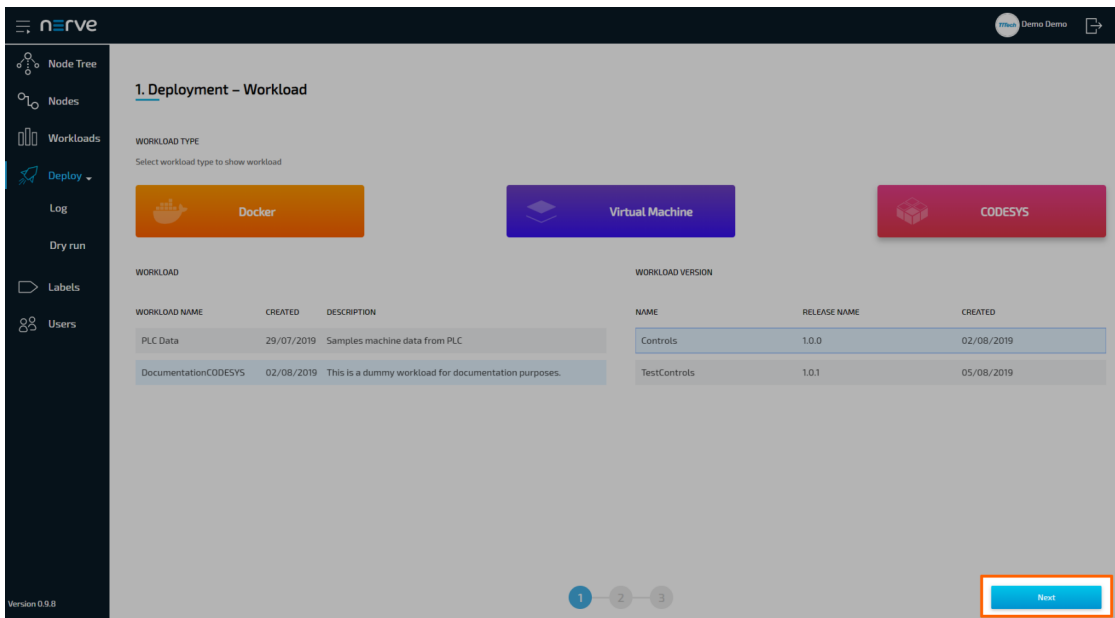


3. Select a workload from the list. A list of versions of this workload will appear to the right.



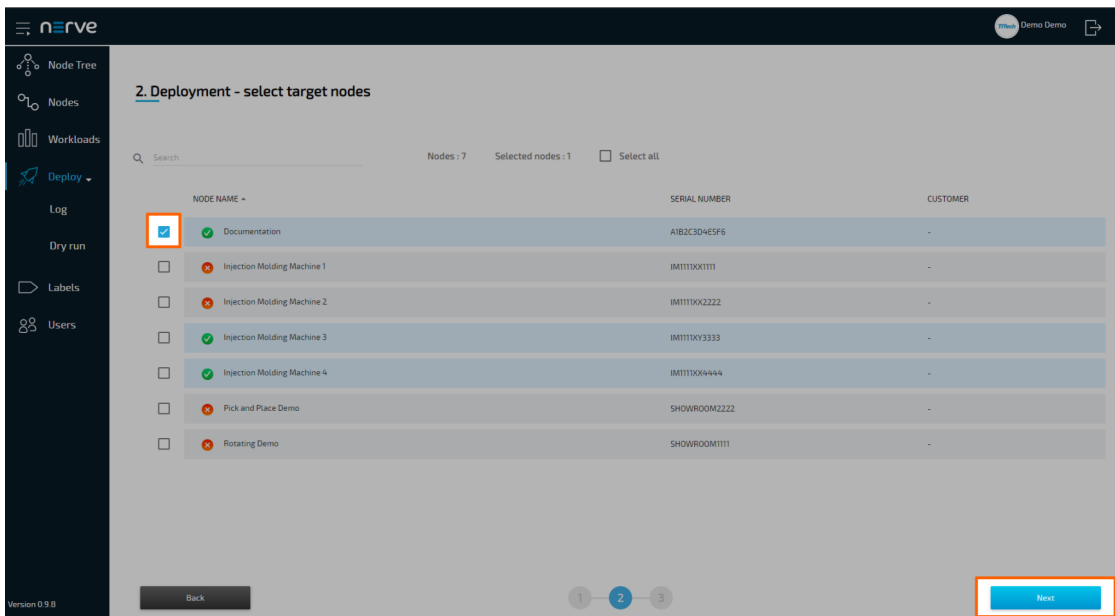
4. Select the version of the workload.

5. Click **Next** in the bottom-right corner.

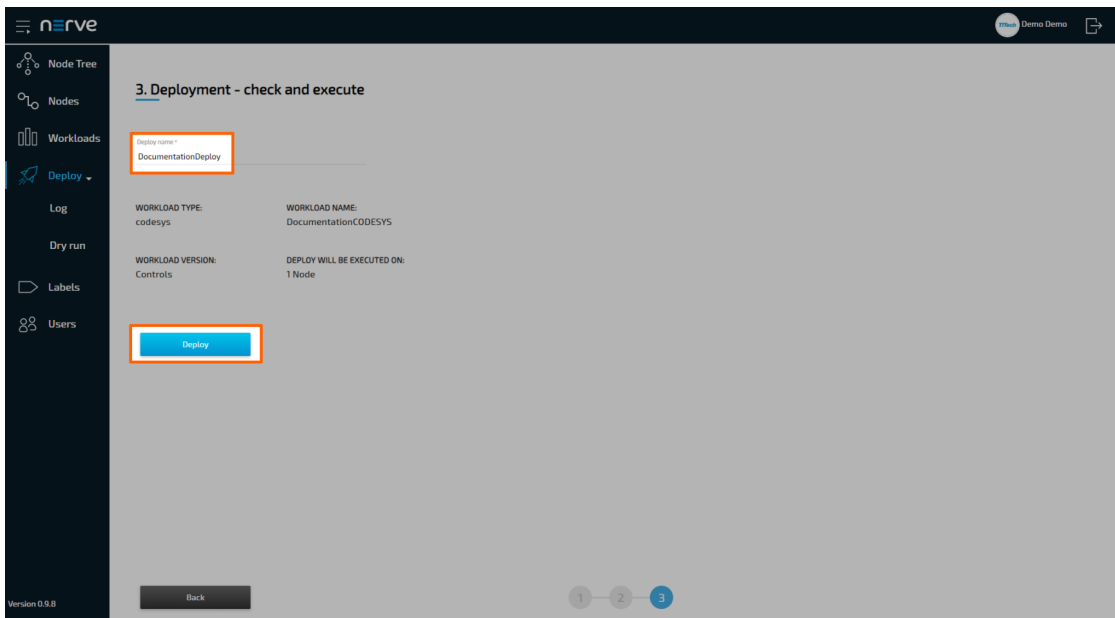


6. In the next window, select one or more nodes from the list for deployment by ticking the checkboxes on the left.

7. Select **Next** in the lower-right corner.



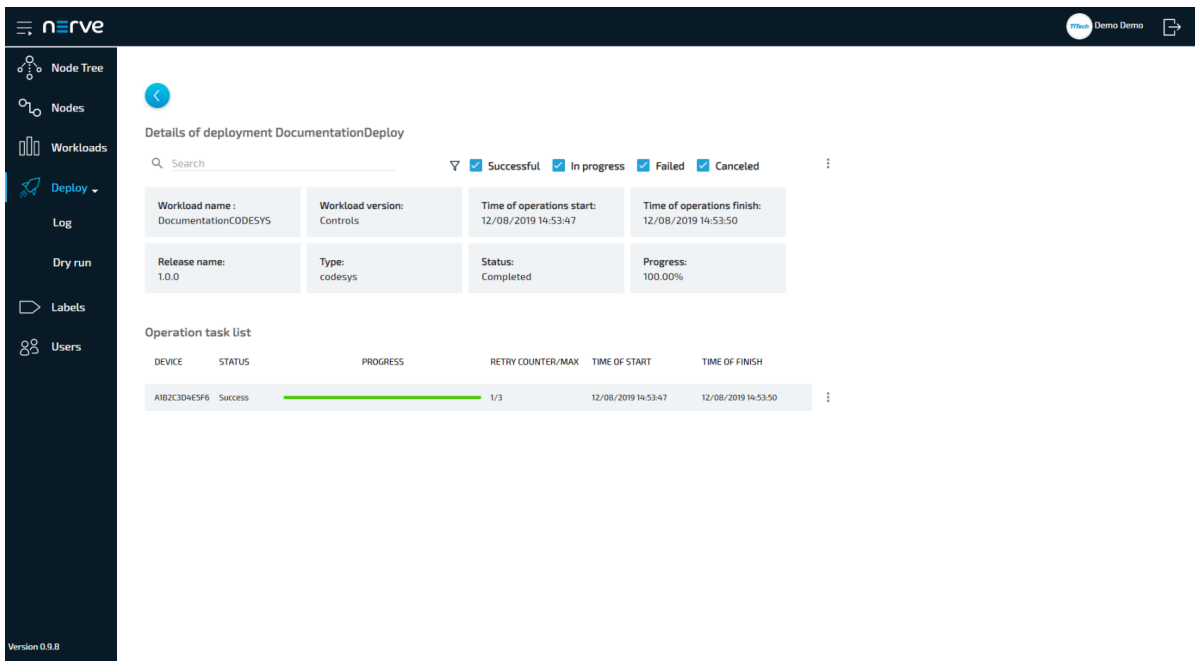
8. Select **Deploy** to execute the deployment.  
*Optional: Enter a **Deploy name** above the **Summary** of the workload to make this deployment easy to identify. A timestamp is filled in automatically.*



The Management System will deploy the log next. The current deployment is at the top of the list. The **Deploy name** chosen before is the name that identifies the deployment in the log.

DEPLOYMENT NAME	ACTION	PROGRESS	STARTED	FINISHED
5/28/2021,9:41:30AM	Deploy	0.00% <span>In progress</span>	28/05/2021 09:41	in progress
5/28/2021,9:35:19AM	Deploy	100.00% <span>Complete</span>	28/05/2021 09:35	28/05/2021 09:35
27/05/2021,12:14:41	Deploy	100.00% <span>Failed</span>	27/05/2021 13:14	27/05/2021 13:15
5/27/2021,1:01:04PM	Deploy	100.00% <span>Failed</span>	27/05/2021 13:01	27/05/2021 13:07
27/05/2021,11:56:40	Deploy	100.00% <span>Complete</span>	27/05/2021 12:56	27/05/2021 12:57
5/26/2021,3:31:06PM	Deploy	100.00% <span>Complete</span>	26/05/2021 15:31	26/05/2021 15:31
5/26/2021,11:37:54AM	Deploy	100.00% <span>Complete</span>	26/05/2021 11:38	26/05/2021 11:39
5/26/2021,11:29:08AM	Deploy	100.00% <span>Complete</span>	26/05/2021 11:29	26/05/2021 11:29
5/26/2021,11:09:57AM	Deploy	100.00% <span>Complete</span>	26/05/2021 11:10	26/05/2021 11:12
5/26/2021,10:54:11AM	Deploy	0.00% <span>In progress</span>	26/05/2021 10:54	26/05/2021 11:00

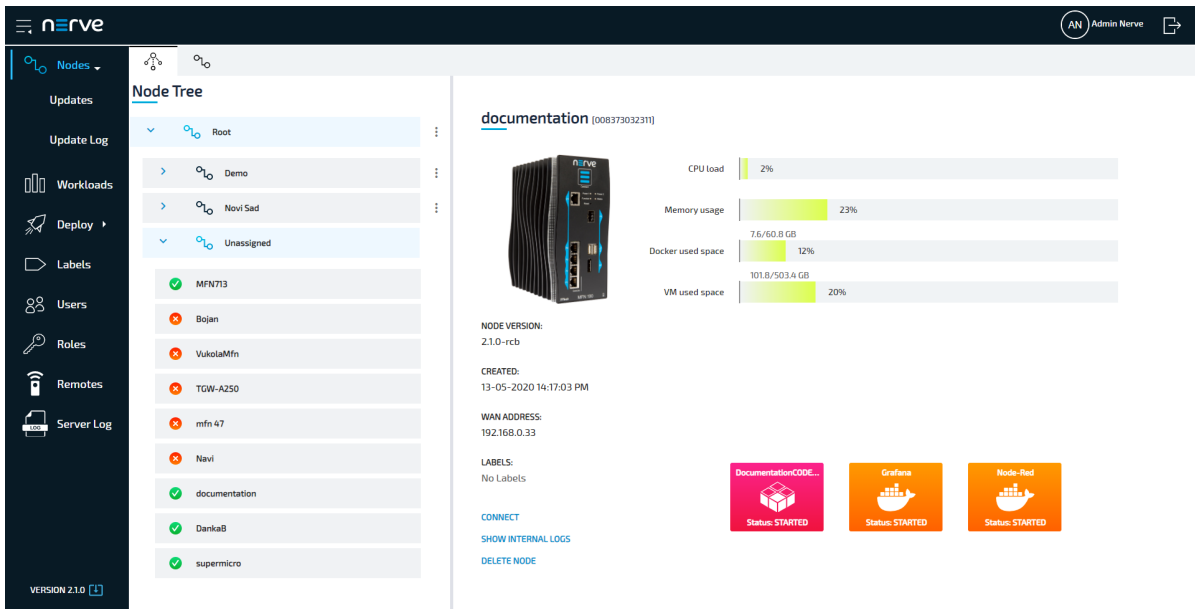
Check the progress of the current deployment and click the workload to see a more detailed view.



Confirm the deployment by viewing the workload in the node details view in the node tree. Select **Nodes** in the navigation on the left and select the node tree tab



on the right. Select the node that has workloads deployed.



Clicking a workload tile leads to the workload control screen. This is where workloads can be controlled. However, CODESYS workloads can only be controlled through the Local UI.

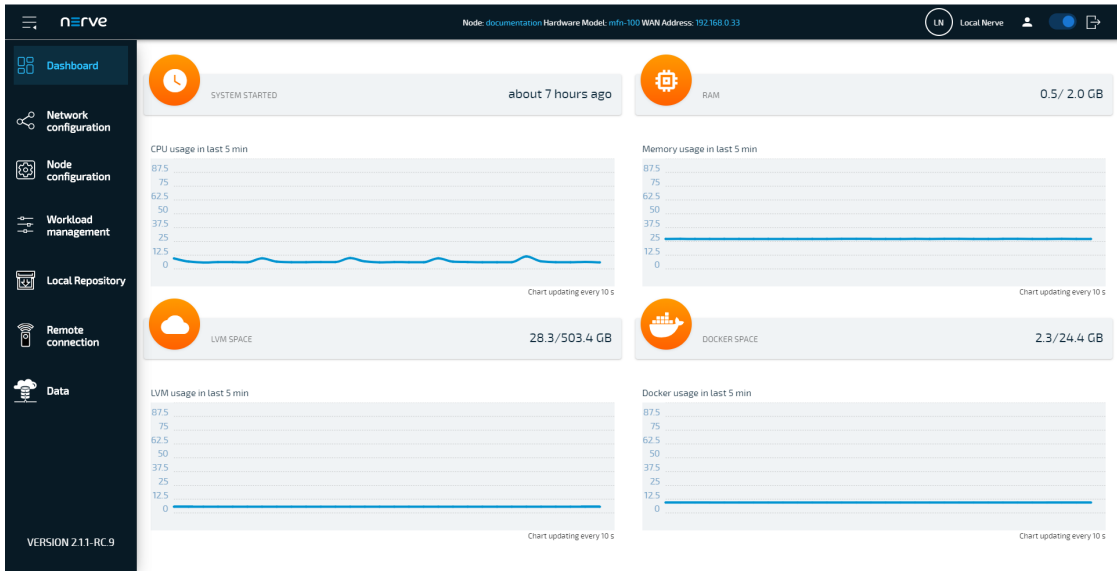
All workloads are started as soon as they are deployed.

## Connecting to the Local UI

In order to access the Local UI, connect a workstation to the console port **P1** of the MFN 100 and configure the network adapter of the workstation. The IP address of the network adapter has to be in the range from 172.20.2.5 to 172.20.2.254 with a

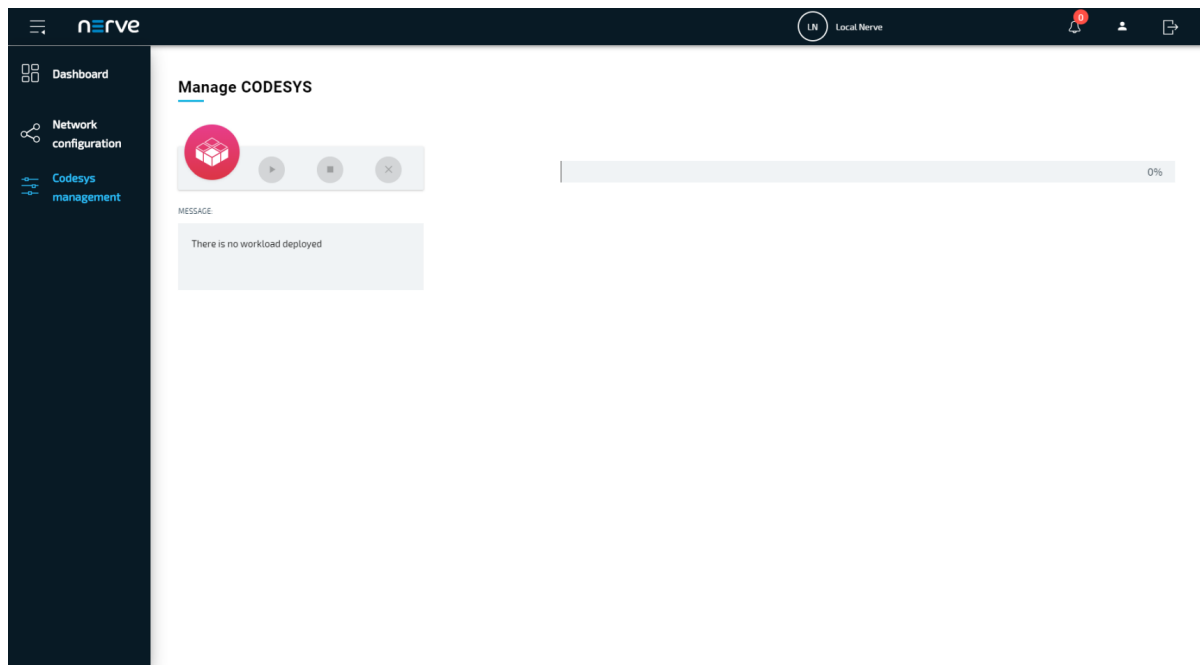
255.255.255.0 subnet mask. The credentials for the Local UI found in the customer profile are also required.

1. Follow this link to connect to the Local UI: <http://172.20.2.1:3333/>
2. Log in with the credentials from the customer profile to reach the main page of the Local UI.



## Control of CODESYS applications

CODESYS workloads can only be controlled in the Local UI, as operation of a CODESYS workload may have an impact on machine operation and therefore should not be controlled remotely. Select **Workload management** in the menu on the left-hand side to reach the interface for controlling a CODESYS application running on the Nerve Device:



Function Name	Description
<b>Start</b>	This starts the CODESYS application.
<b>Stop</b>	This stops the CODESYS application and it is reset to its initial values.
<b>Remove</b>	This removes the CODESYS application from the Nerve Device. To deploy the CODESYS application again, do so through the Management System.
<b>Message</b>	<p>CODESYS workloads have the following set of messages:</p> <ul style="list-style-type: none"> <li>• "Preparing files for installation"</li> <li>• "Starting CODESYS application"</li> <li>• "CODESYS application started"</li> <li>• "Stopping CODESYS application"</li> <li>• "CODESYS application stopped"</li> <li>• "Removing CODESYS application file"</li> <li>• "An unexpected error has occurred. &lt;errormessage&gt;"</li> </ul> <p>Here, &lt;errormessage&gt; is a message that is sent by the CODESYS Development System.</p>

#### NOTE

It takes a moment before CODESYS applications are actually started, stopped or removed.

## Uploading new CODESYS applications

In order to work with new CODESYS applications on the MFN 100, new CODESYS workloads need to be provisioned in the Management System. Here, provisioning is the creation of a workload and its storage in the workload repository of the Management System so that it can be deployed to nodes. This requires configuration of the workload and the upload of the required files to the Management System. After that, the workload can be deployed to nodes.

Before the workload can be provisioned, however, a CODESYS application has to be loaded into the CODESYS runtime first. Refer to the [introduction to working with CODESYS and the MFN 100](#) first before continuing.

#### NOTE

Note that **app1** is already loaded into the CODESYS runtime by default.

Once a CODESYS application has been loaded into the MFN 100, the following steps have to be taken before provisioning a CODESYS workload:

- Creating the ZIP file of the CODESYS application
- Transferring the ZIP file to a local workstation

Also the workstation needs to be connected to the console port **P1** of the MFN 100 and the network adapter of the workstation needs to be configured. The IP address of the network adapter has to be in the range from 172.20.2.5 to 172.20.2.254.

## Creating the ZIP file on the Nerve Device

First, the CODESYS project needs to be zipped on the Nerve Device before it can be copied from the CODESYS runtime. This is done through the Local UI.

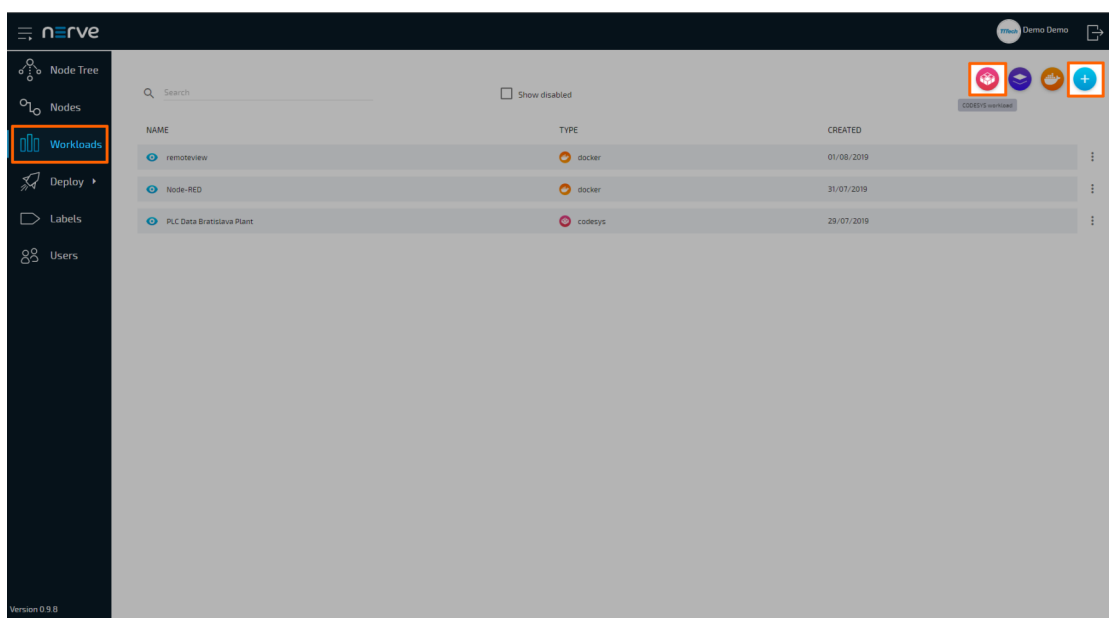
1. Connect to the Local UI as described [above](#).
2. Select **Workload management** in the navigation on the left.
3. Click **Download CODESYS app archive**.
4. Select **YES** in the pop-up. Note that the CODESYS application will be stopped.

The ZIP file is automatically downloaded to the workstation and a CODESYS workload can now be provisioned in the Management System.

## Provisioning a CODESYS workload

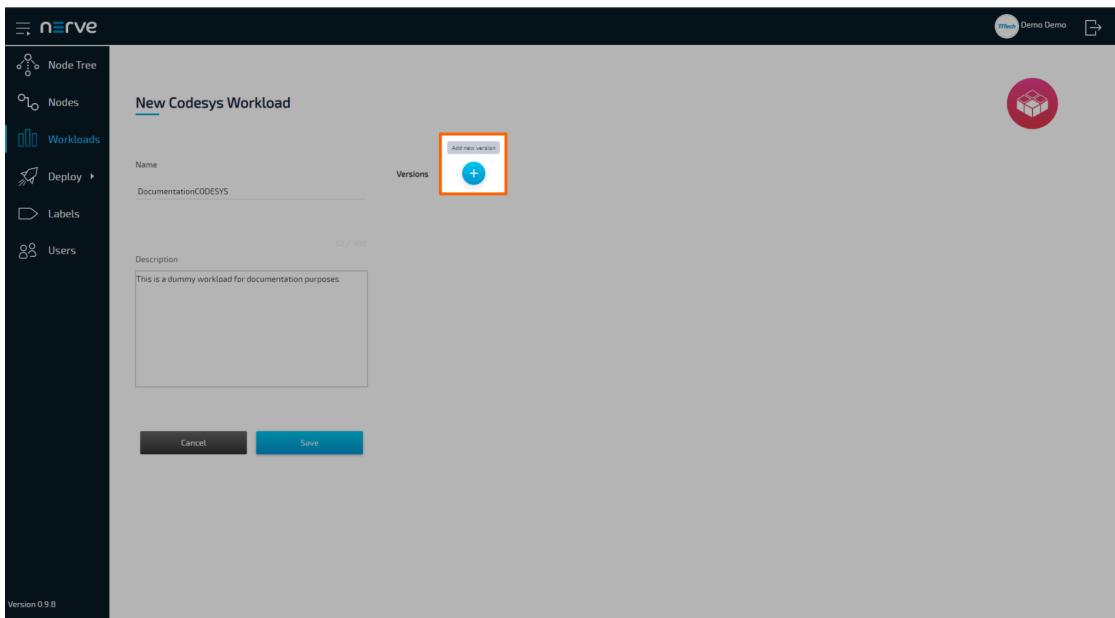
The following instructions cover the basic requirements for provisioning a CODESYS workload. Optional settings will be left out. Extended options are addressed in the [user guide](#).

1. Log in to the Management System.
2. Select **Workloads** in the left-hand menu.
3. Select the plus symbol in the upper-right corner.
4. Select the CODESYS symbol (**CODESYS workload**) on the left of the three symbols that expanded.



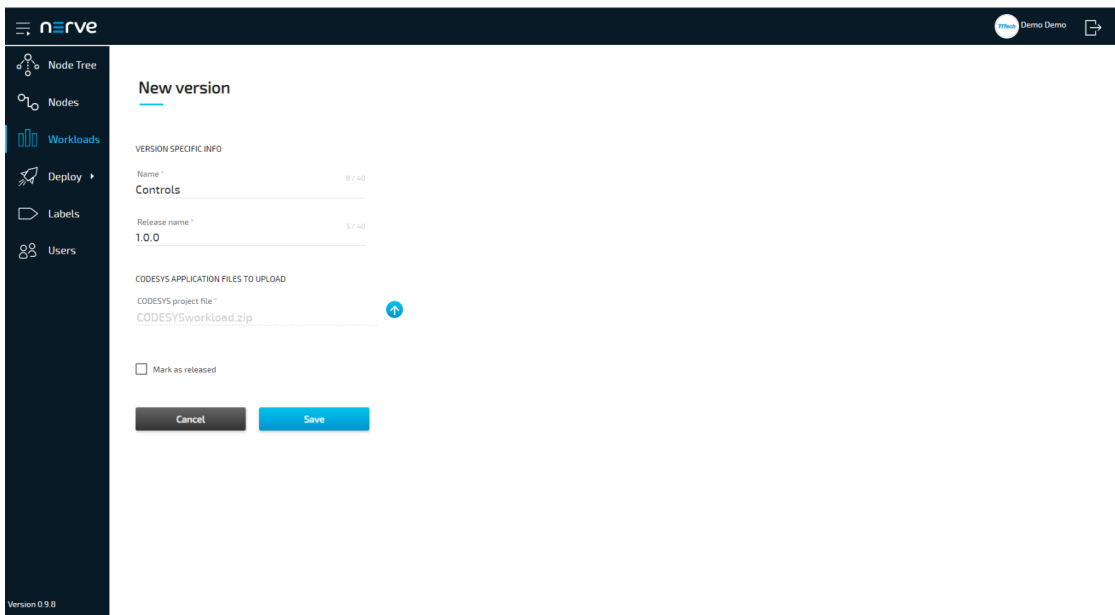
5. Enter a name for the workload in the new window.
6. Select the plus symbol next to **Versions** to add a new version of the workload.





7. Enter the following information in the new window:

Item	Description
<b>Name</b>	Enter a <b>Name</b> for the version of this workload.
<b>Release name</b>	Enter a <b>Release name</b> for the version of this workload.
<b>CODESYS project file</b>	Click the <b>upward arrow</b> symbol to open the file browser and add the CODESYS application ZIP file. This is the ZIP file that has been created before.



8. Click **Save**.

The workload has now been provisioned and is ready to be deployed in the **Deploy** menu.

# Connecting new sensors and actuators

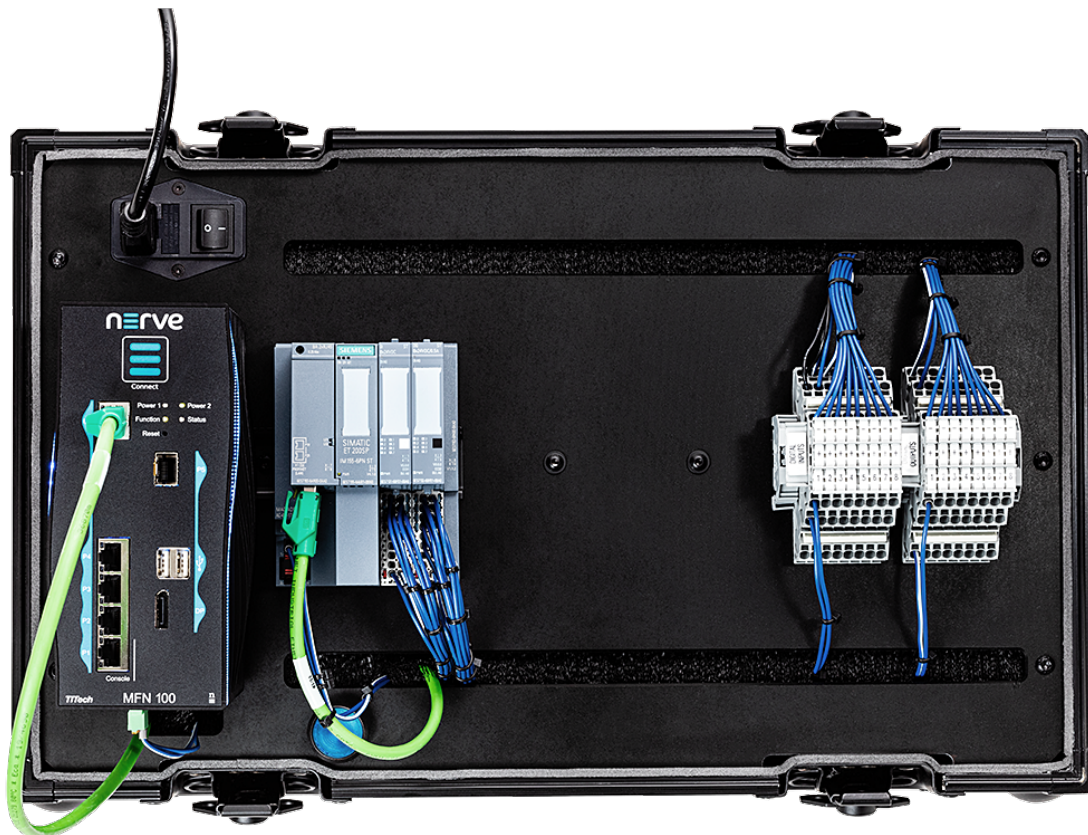
Add sensors and actuators to the kit to execute control applications and visualize the corresponding data. With the delivered set-up it is possible to add up to 7 additional inputs and outputs. I/O blocks can also be added to increase the number or type of inputs and outputs.

## Wiring a new sensor or actuator

### NOTE

- Before wiring any new components, review the Nerve Kit Circuit Diagram. Contact a sales representative for more information.
- Disconnect the power supply from the power outlet before wiring new I/O devices to prevent injury to persons or damage to equipment.
- Only staff with knowledge about electrical circuits should perform the tasks described in this section.

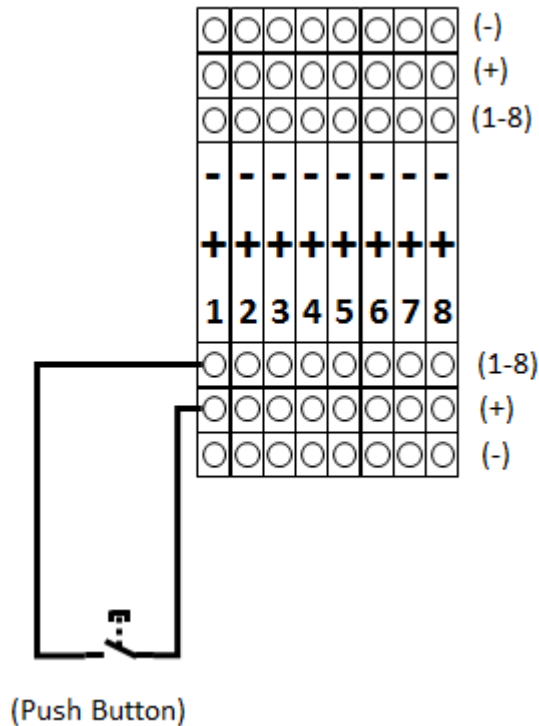
The inputs and outputs of the SIMATIC ET200 SP I/O module are wired to the terminal blocks on the right hand side of the kit. The left terminal block is used to connect digital inputs. The right terminal block is used to connect digital outputs.



## Connecting a digital input

This section shows how to connect an additional digital input to the kit. A push button is used for demonstration purposes.

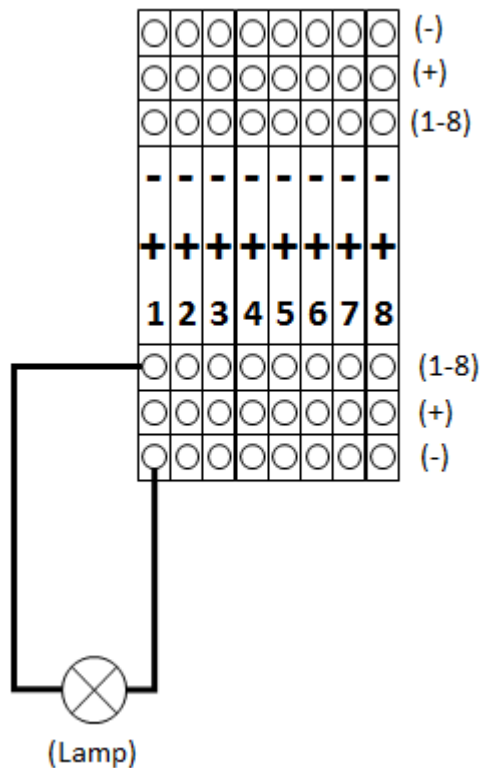
1. Connect the power supply of the button to the middle row of the I/O module (+24V).
2. Connect the input to the top row.



## Connecting a digital output

This section shows how to connect an additional digital output to the kit. A lamp is used for demonstration purposes.

1. Connect the lamp to the top row of the I/O module.
2. Connect the common wire to the bottom row to close the electrical circuit.



After wiring the sensors or actuators to the inputs or outputs respectively switch the kit back on.

The next chapter describes how to [assign variables to the inputs and outputs](#) in order to read data from newly connected sensors or control actuator functionality.

## First steps with CODESYS and the MFN 100

This chapter will give an introduction on how to start working with the integrated soft PLC in the MFN 100. First, some configuration and installation of files and libraries are required.

### NOTE

- Download the CODESYS Development System V3 from [store.codesys.com](https://store.codesys.com) for this chapter.  
We recommend version 3.5 SP14 (32 bit) or newer.
- Connect the workstation to the console port **P1** of the MFN 100.

## Installing the device descriptions

After downloading and installing the CODESYS Development System on the workstation, install the device descriptions of the MFN 100 and the SIMATIC ET200 SP I/O module in the CODESYS Development System. The device descriptions have the following filenames:

**MFN 100**

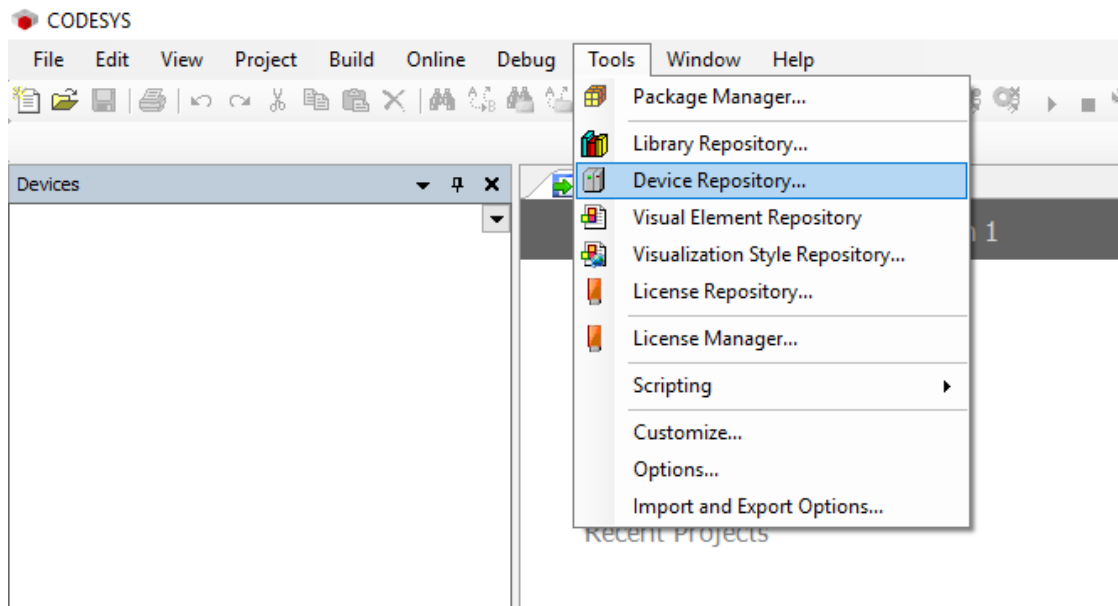
Nerve\_MFN\_100\_V3.5.XX.X.devdesc.xml

**SIMATIC ET200 SP**

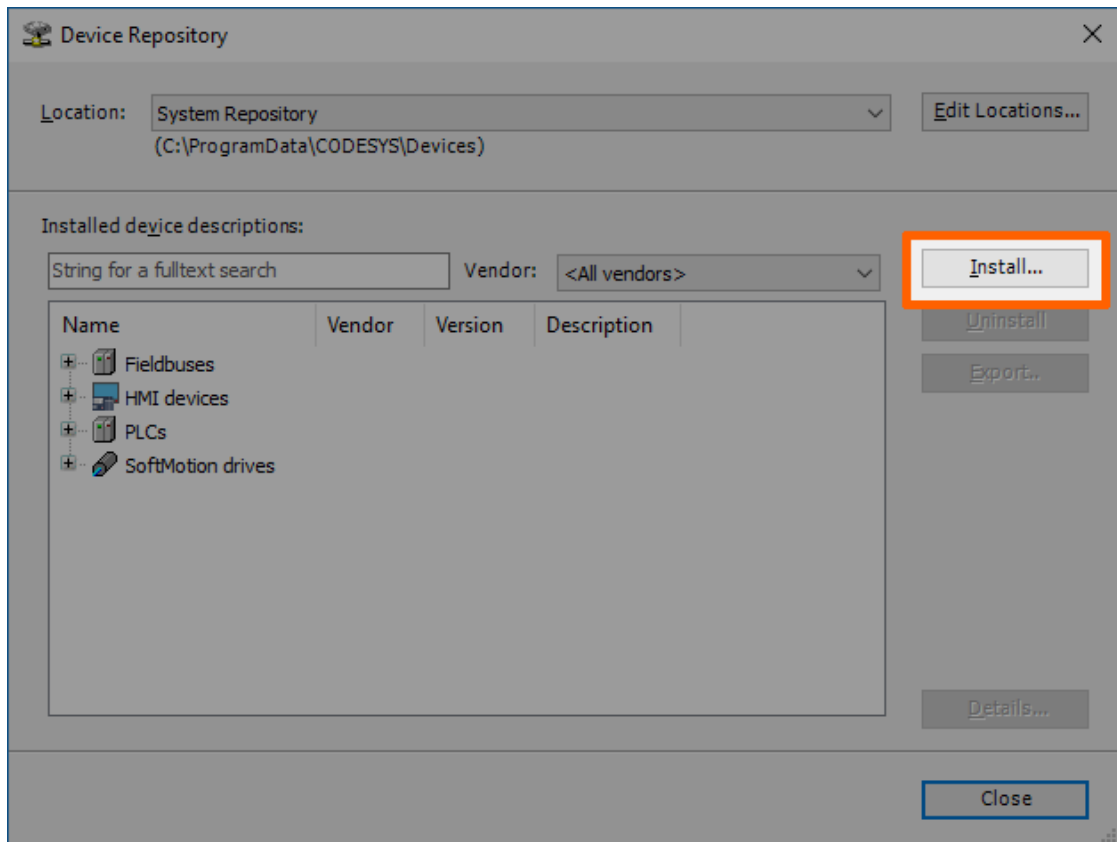
GSDML-V2.34-Siemens-ET200SP-20180926.xml

XX.X stands for the current version of the CODESYS Development System. The device descriptions of the MFN 100 and the SIMATIC ET200 SP I/O module are available at the [Nerve Software Center](#).

1. Start the CODESYS Development System.
2. Go to **Tools > Device Repository**.

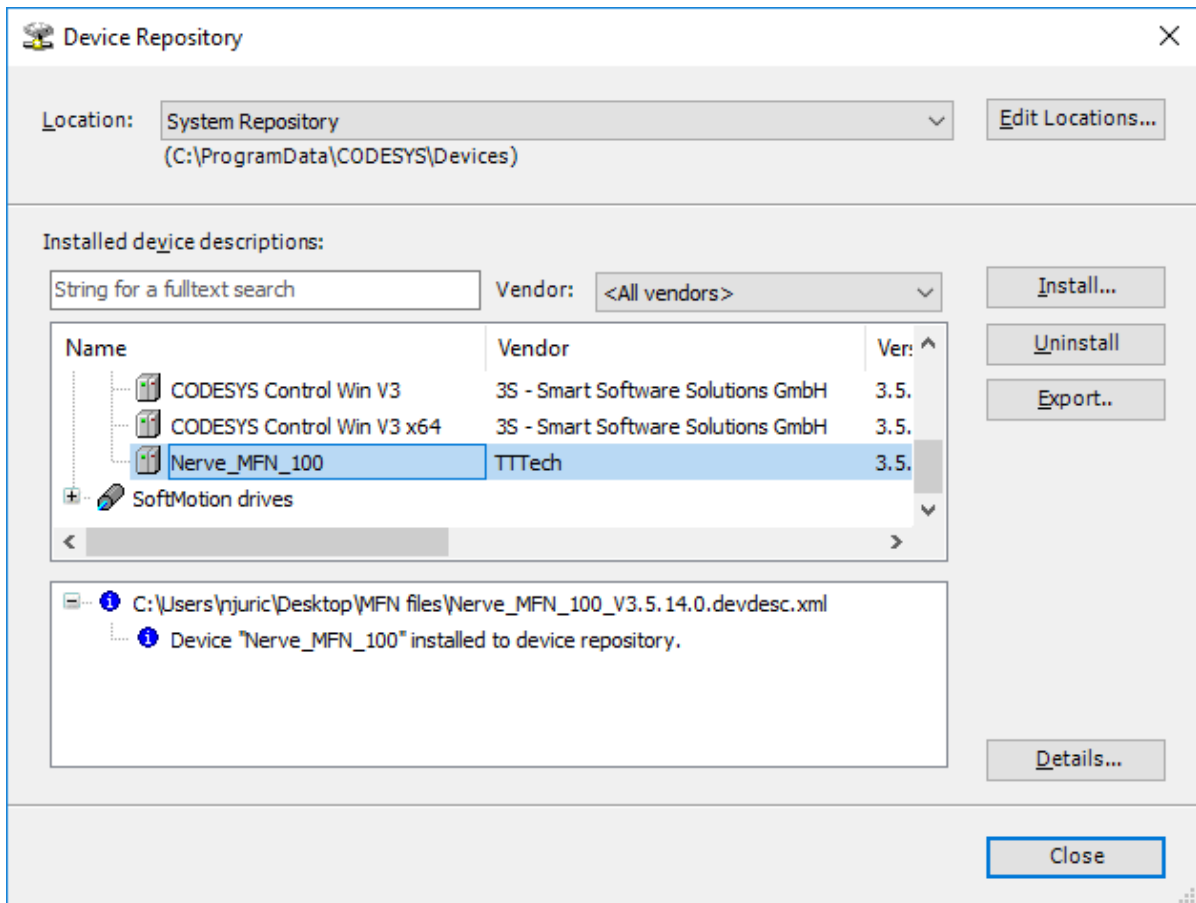


3. Click **Install**.



4. Go to the directory of the previously downloaded device description.
5. Select the device description of the MFN 100.
6. Click **Open**.
7. Repeat steps 3 to 6 to install the device description SIMATIC ET200 SP I/O module.

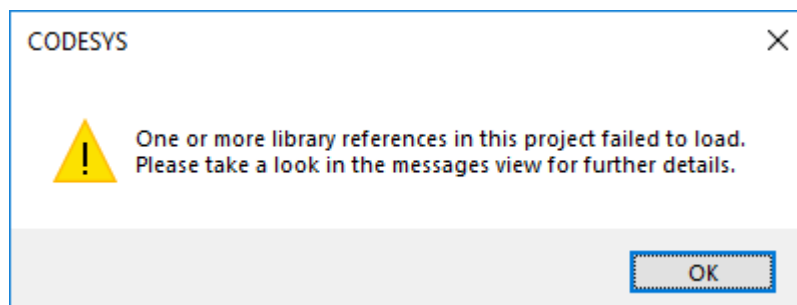
When the installation was successful, the MFN 100 and the SIMATIC ET200 SP I/O module will appear in the list of device descriptions in the middle of the window. Close the window afterwards.



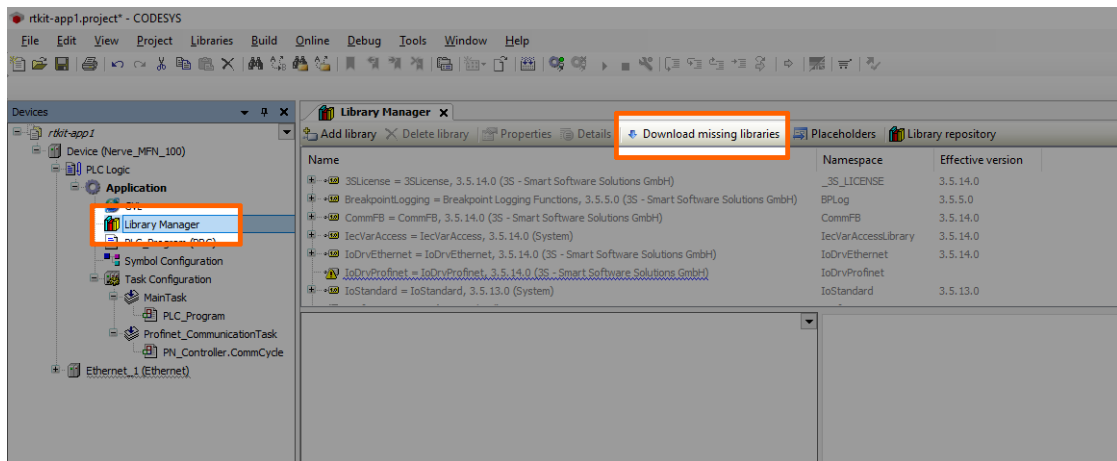
After installing the device description the CODESYS Development System can be worked with. However, libraries and device descriptions of generic devices might be missing so that the CODESYS Development System can work properly.

## Downloading missing libraries

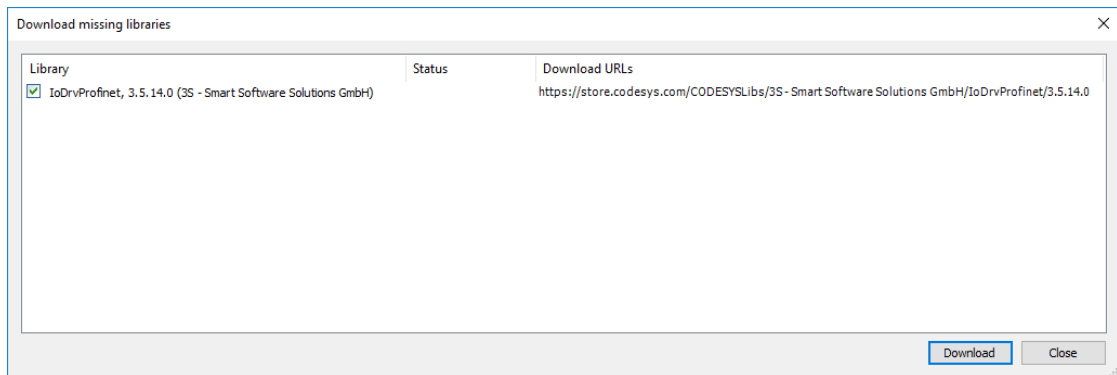
The error message for missing libraries might appear when opening or creating a CODESYS project. The CODESYS Development System identifies the missing libraries automatically but the following process might have to be repeated a few times.



1. Open or create a CODESYS project.
2. If the error message about missing libraries appears, click **OK**.
3. Double-click **Library Manager** in the tree view on the left.
4. Click **Download missing libraries**.



5. Click **Download** in the new window.



6. Click **Close** when the download is finished.

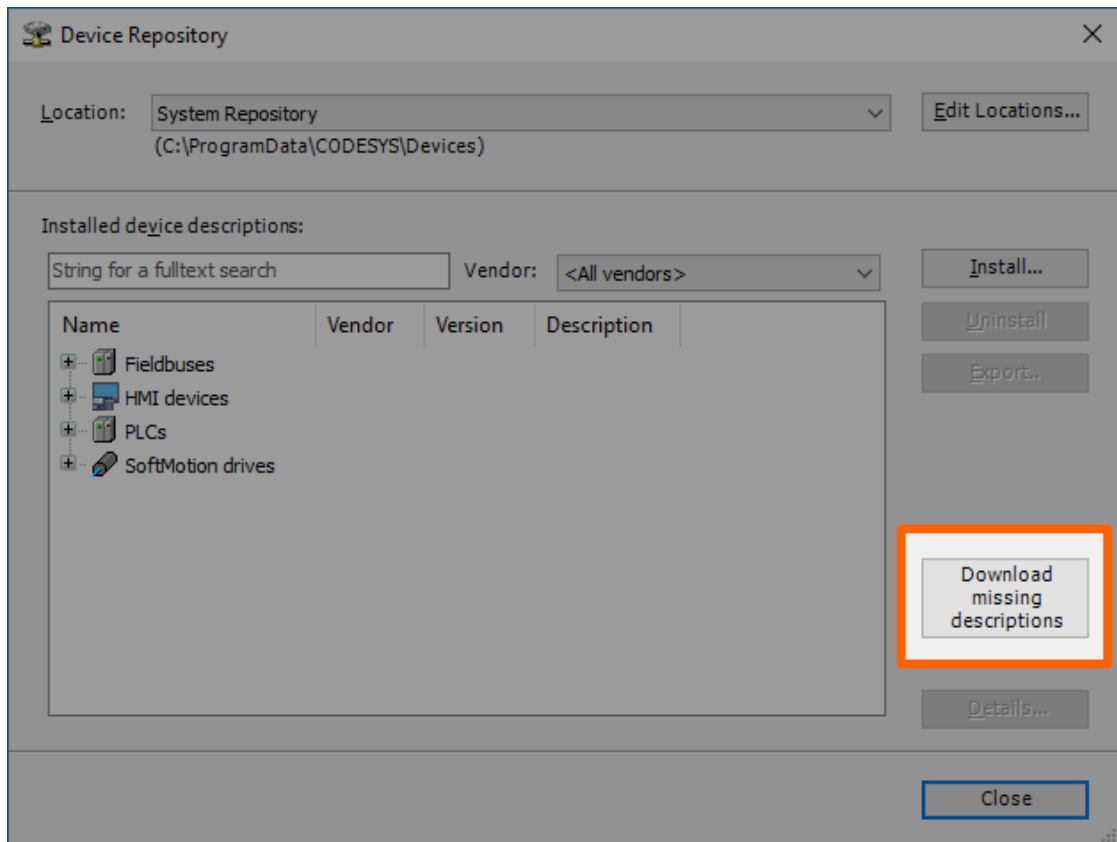
7. Repeat steps 3 to 5 until no more libraries appear in the download window.

## Downloading missing device descriptions

Apart from the device description for the MFN 100, device descriptions of generic devices may be missing for the CODESYS Development System to function as intended. The CODESYS Development System will identify the missing device descriptions automatically but this time it will not generate an error message unless a CODESYS application is loaded into the MFN 100.

1. Click **Tools > Device Repository**.
2. Click **Download missing descriptions**.





#### NOTE

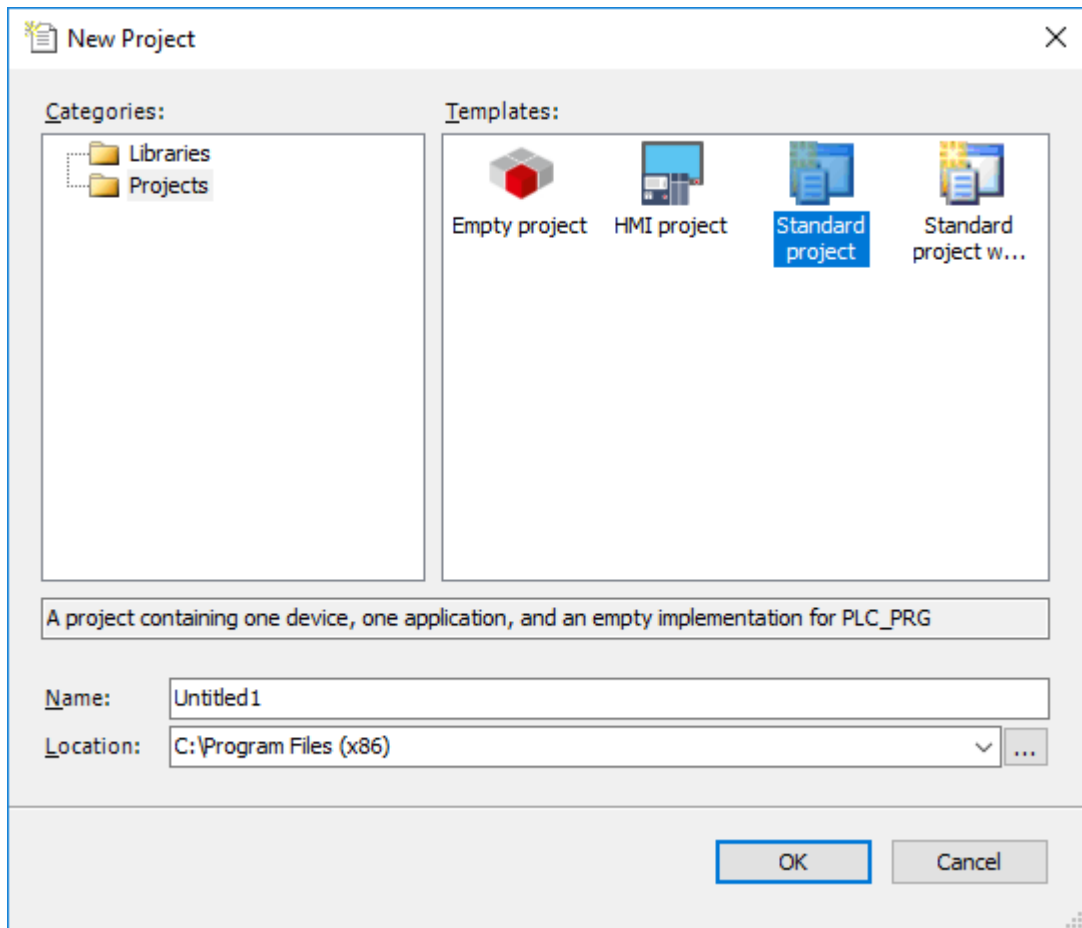
If no device descriptions of generic devices are missing, the button for downloading missing descriptions will not appear.

3. Click **Download** in the new window.
4. Click **Close** when the download is finished.

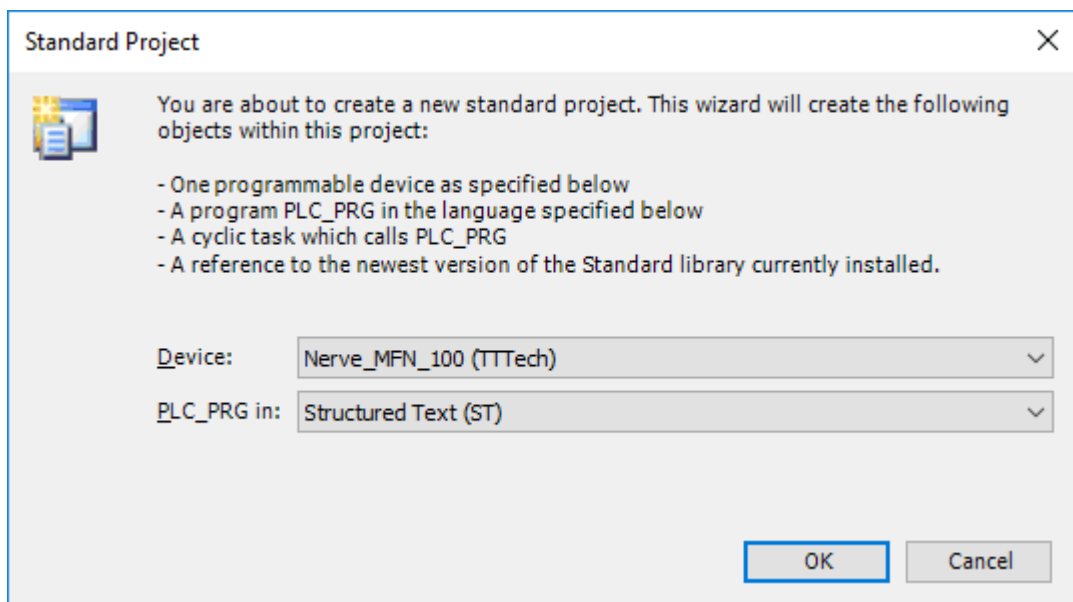
## Creating a new CODESYS project

This example shows how to create a new project in the CODESYS Development System. The easiest way to get started is to create a **Standard project**.

1. Start CODESYS
2. Go to **File > New Project**.
3. Click **Standard project** on the right side among the templates.
4. Enter a name for the project.
5. Choose a **Location** where the project will be saved.
6. Click **OK** to save the project.

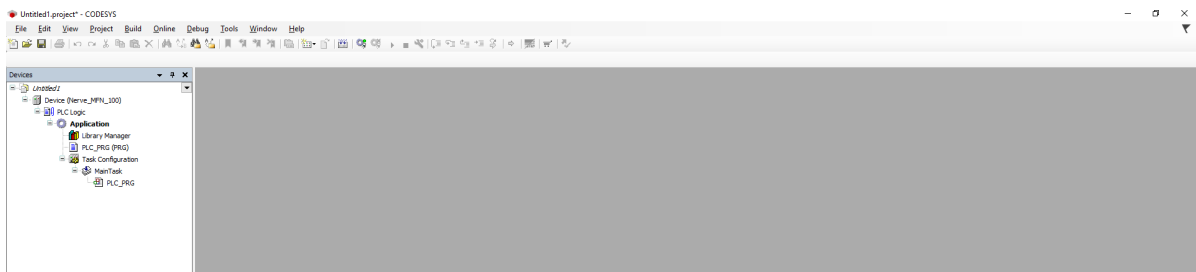


7. Select **Nerve\_MFN\_100 (TTTech)** as the device.



8. Click **OK**.



The result is an empty project that is open in the main view of CODESYS.



## Working with the default applications

To work with existing applications first, modify the default applications app1.project and app2.project. They can be downloaded from the [Nerve Software Center](#) under **Example applications**.

1. Start CODESYS.
2. Go to **File > Open Project**.
3. Select the download location of the default applications.
4. Select the application to work with.

Name	Date modified	Type	Size
 app1.project	24.07.2019 14:55	CODESYS project	287 KB
 app2.project	24.07.2019 14:57	CODESYS project	288 KB

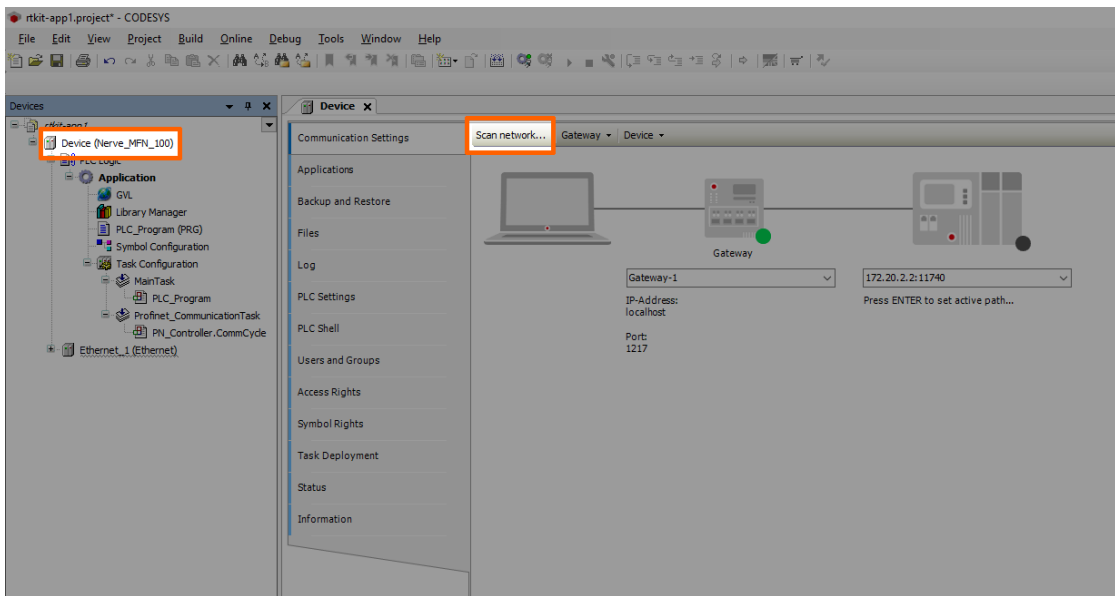
5. Click **Open**.

When opening the default applications for the first time, some libraries and device descriptions will be missing. Follow the instructions [above](#) to see how to download the missing files.

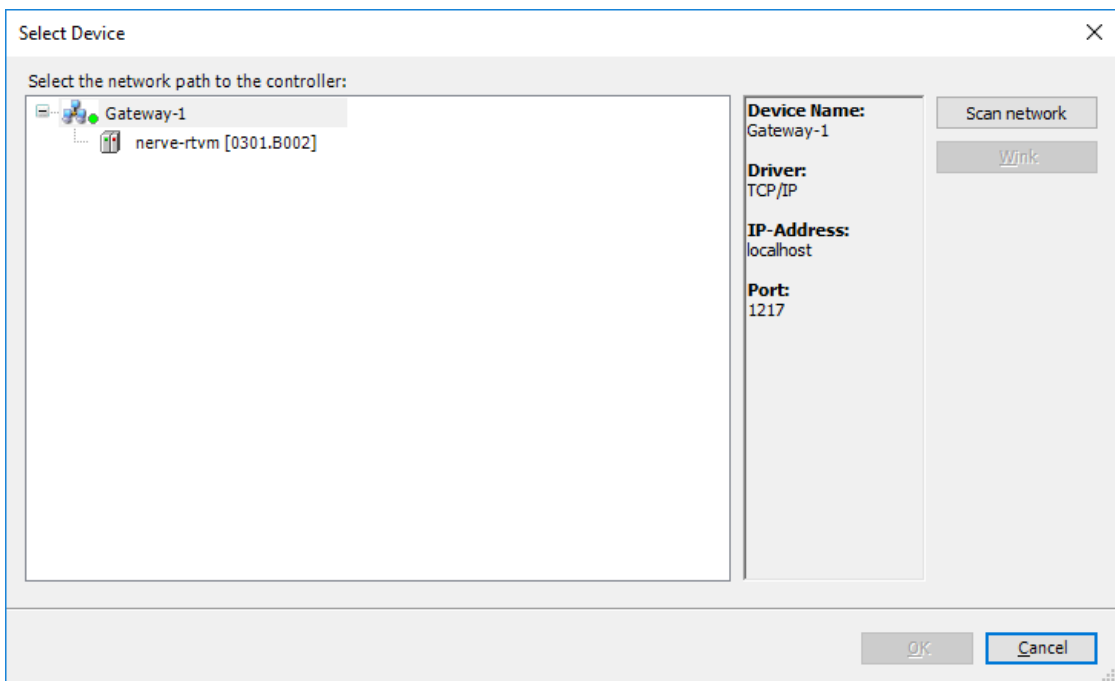
## Connecting to the MFN 100

Before downloading CODESYS applications to the MFN 100, make sure that the device description of the MFN 100 is installed in the CODESYS Development System.

1. Open or create a CODESYS project.
2. Double-click **Device (Nerve\_MFN\_100)** in the tree view on the left.
3. Go to **Communication Settings > Scan network....**



4. Select the MFN 100 (here **nerve-rtvm [XXXX.XXXX]**) in this window.



**NOTE**

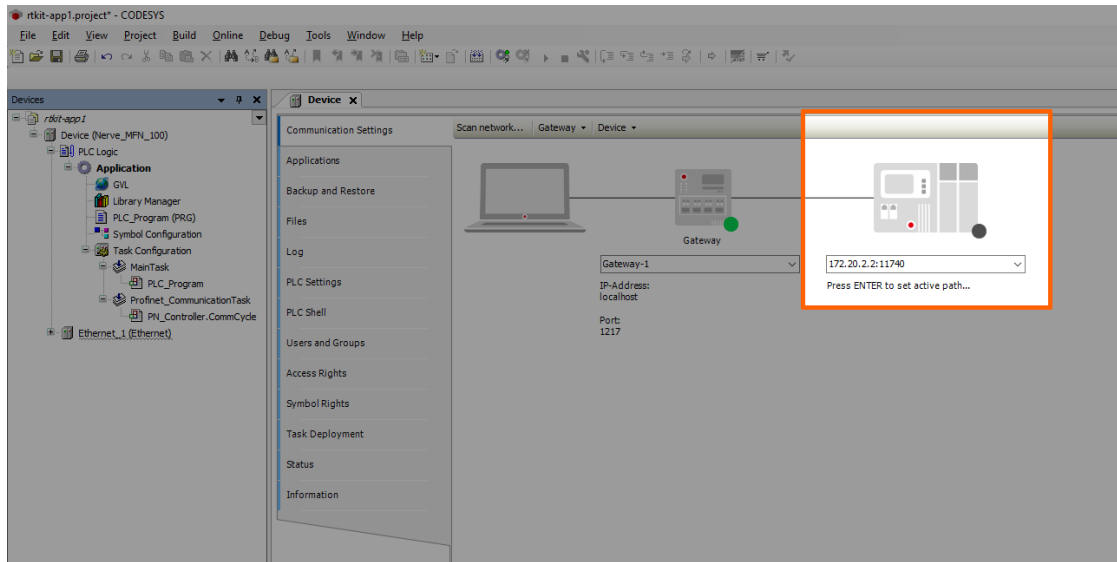
When more than one network is active on the workstation, it sometimes happens that the MFN 100 cannot be found. Continue reading if the MFN 100 does not appear in this window.

5. Click **OK**.

Typically the MFN 100 will be found automatically. If the MFN 100 cannot be found, enter the IP address and port of the CODESYS runtime manually.

1. Double-click **Device (Nerve\_MFN\_100)** in the tree view on the left.

2. Go to **Communication Settings** in the middle of the window.
3. Enter 172.20.2.2:11740 in the text box under the device on the right.



4. Press Enter.

The CODESYS Development System is now connected to the MFN 100 and applications can be downloaded into the CODESYS runtime.

## Downloading an application to the MFN 100

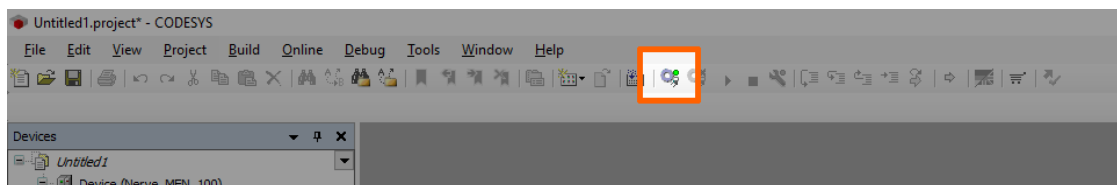
CODESYS applications can be loaded directly into the MFN 100. However, before downloading an application into the MFN 100 it needs to be free of errors.

The process of downloading an application is slightly different when downloading an entirely new application into the MFN 100 or updating an application that has already been downloaded into the MFN 100. Continue with [Downloading an Updated Application to the MFN 100](#) further down below if an application is being updated.

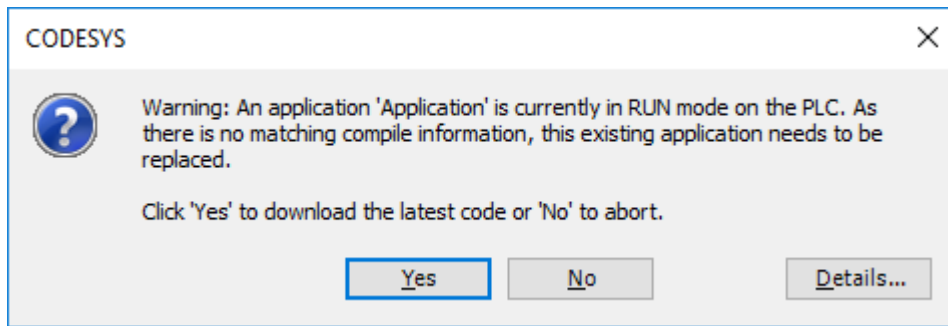
## Downloading a new Application to the MFN 100

After creating a project and finishing with programming, the CODESYS application can be downloaded into the MFN 100 directly.

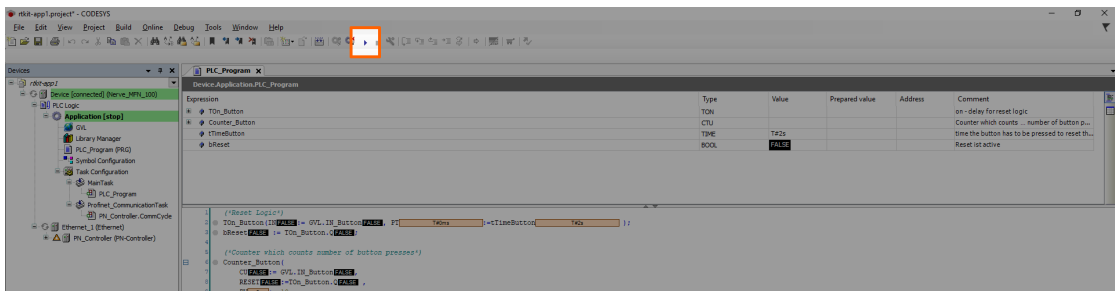
1. Open the CODESYS project to load into the MFN 100.
2. Click the **Login** symbol in the CODESYS menu bar.



3. Click **Yes** in the pop-up window.



4. The application is stopped now. Click the **Play** symbol in the CODESYS menu bar.

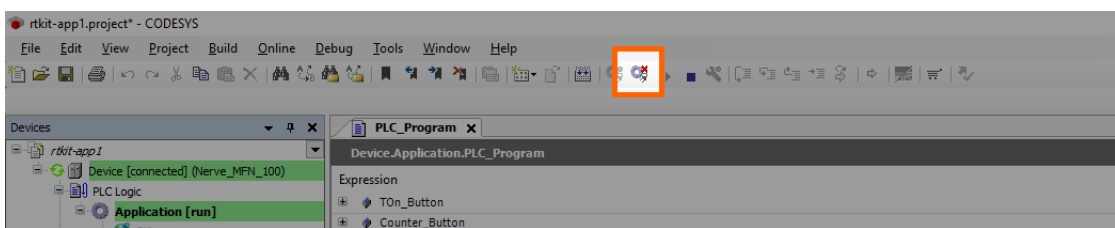


The application is now loaded to the MFN 100.

## Downloading an updated application to the MFN 100

Updating an application after loading it into the MFN 100 requires another download into the MFN 100. The download process is slightly different from downloading a new application into the MFN 100.

1. Stop the CODESYS application that has been loaded into the MFN 100 through the **Local UI**.
2. Click the **Logout** button in the CODESYS toolbar.

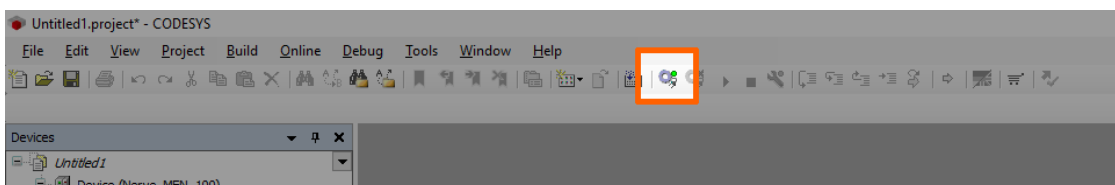


3. Expand **Device (Nerve\_MFN\_100) > PLC Logic > Application**.

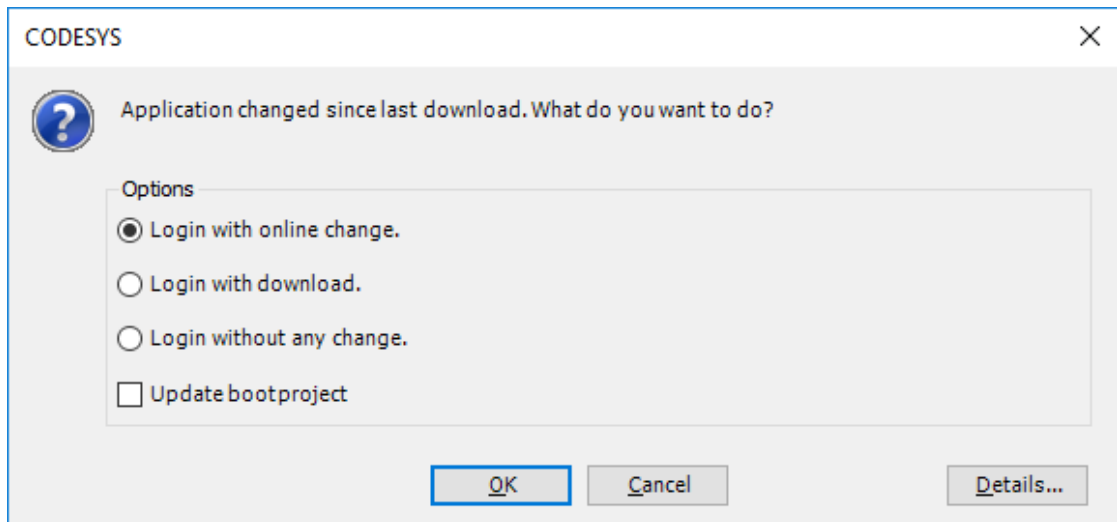
4. Double-click **PLC\_Program (PRG)**.

5. Perform the changes.

6. Click the **Login** symbol in the CODESYS menu bar.



7. In the pop-up window, select one of the options.



Item	Description
<b>Login with online change.</b>	The updated application will be loaded into the MFN 100. Variable values will not be reset. If the application was running before, it will be running after the download.
<b>Login with download.</b>	The updated application will be loaded into the MFN 100. Variable values will be reset. The application is stopped.
<b>Login without any change.</b>	The updated application will not be loaded into the MFN 100 but the code will keep the changes.

8. Click **OK**.

The application is now loaded to the MFN 100.

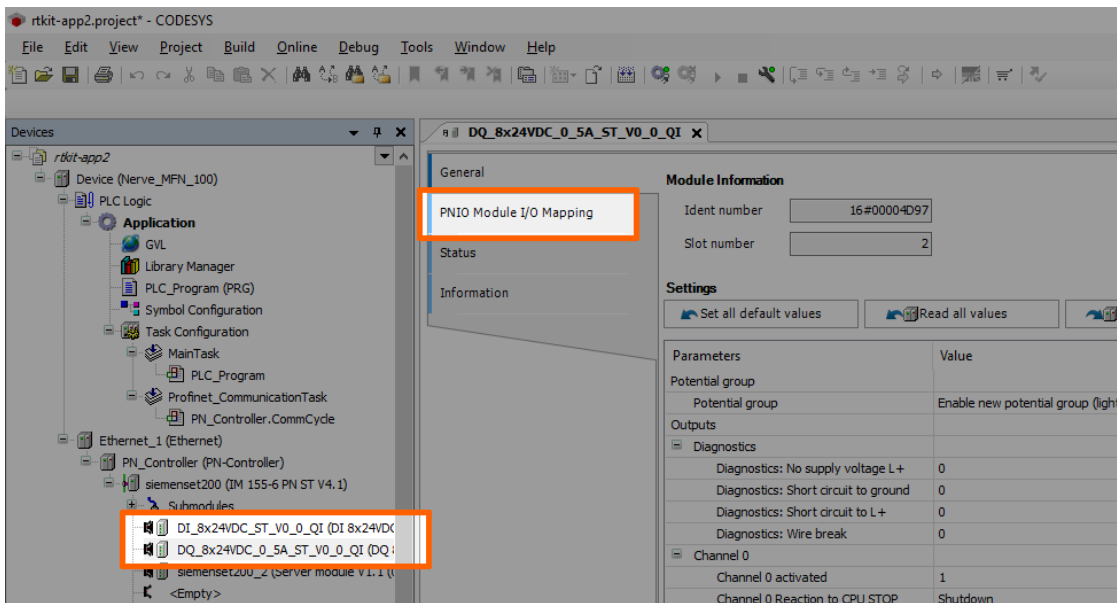
#### NOTE

For more help with programming PLC applications in the CODESYS Development System go to [help.codesys.com](http://help.codesys.com).

## Allocating variables to inputs or outputs

After [connecting new sensors and actuators](#), variables need to be assigned to the I/O channel in CODESYS.

1. Open a CODESYS project.
2. Expand **Device (Nerve\_MFN\_100) > PLC Logic > Ethernet\_1 > PN\_Controller > siemenset200 (IM 155-6 PN ST V4.1)** in the tree structure on the left.
3. Double-click **DI\_8x24VDC\_ST\_V0\_0\_QI (...)** for digital inputs.  
Double-click **DQ\_8x24VDC\_0\_5A\_ST\_V0\_0\_QI (...)** for digital outputs.
4. Select **PNIO Module I/O Mapping**.

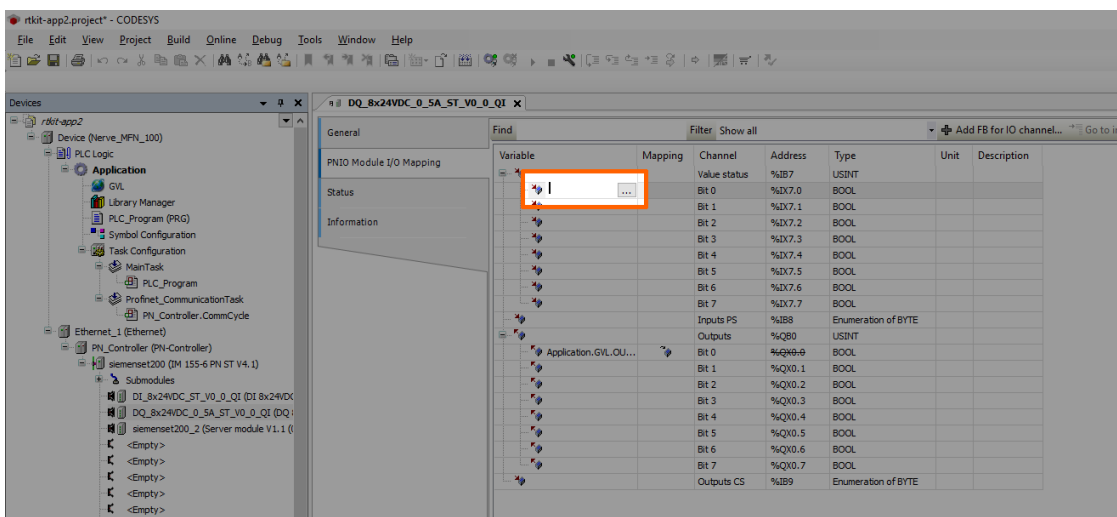


5. Fully expand the tree view.
6. Double-click the variable slot to assign.

#### NOTE

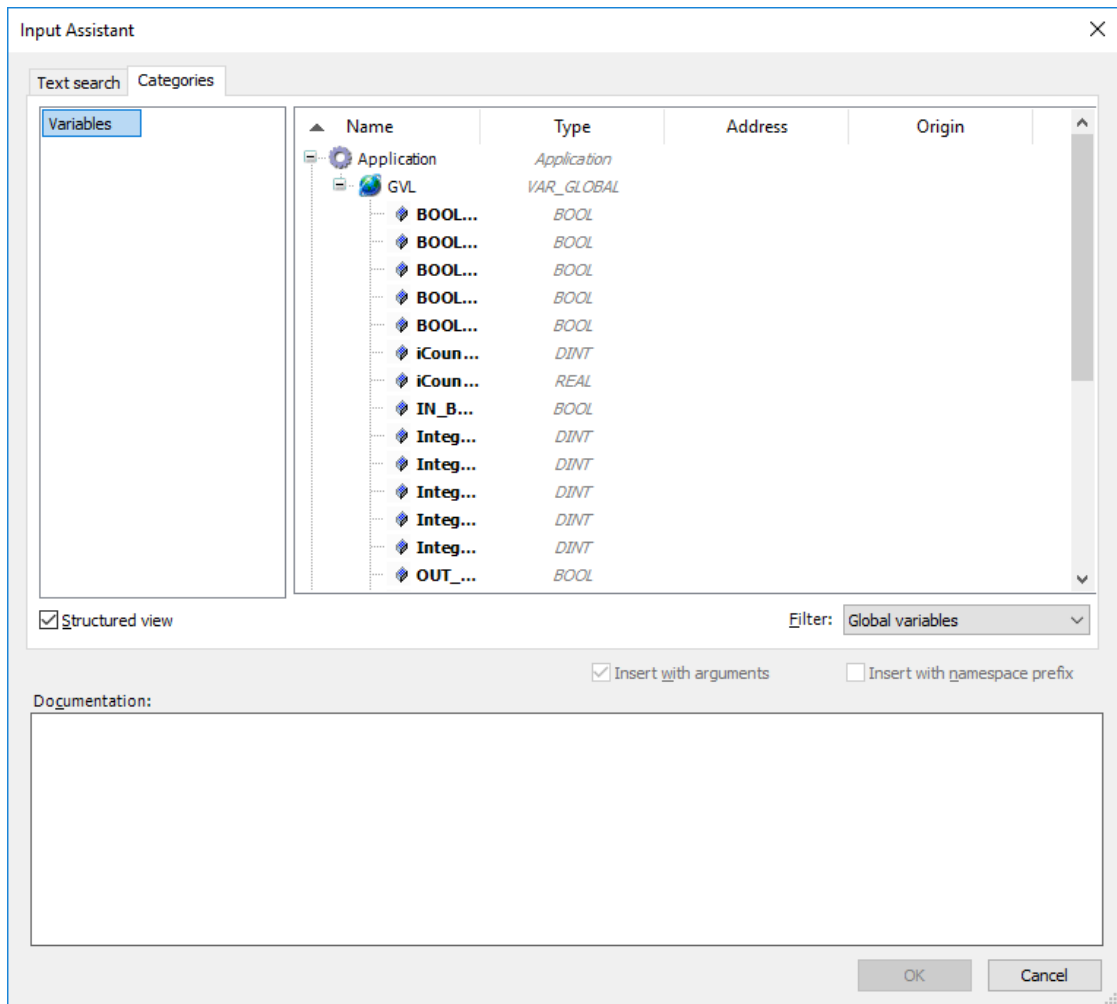
The inputs in this view do not match the physical inputs of the I/O module on the kit. The inputs here go from 0 to 7. The physical inputs go from 1 to 8. Therefore input 0 in this view represents the physical input 1 on the I/O module. This also applies to outputs.

7. Click the three dots next to the variable slot.



8. Expand **Application** > **GVL** in the new window.





9. Select the variable to assign.

#### NOTE

Make sure to select a variable of the same type as the input, i.e., a **BOOL** variable for a **BOOL** input or output.

10. Click **OK**.

Use the assigned variables to read data from connected sensors or to control actuator functionality.

#### NOTE

For more help with programming PLC applications in the CODESYS Development System go to [help.codesys.com](http://help.codesys.com).

# Data transfer from CODESYS to the Management System

The kit sends data that is generated in CODESYS from the MFN 100 to the Management System using variables. The kit has a pre-configured Gateway configuration for the Data Services that can be further configured to send new values to the Management System. This chapter gives a quick overview on how to configure new variables.

## Overview

The CODESYS runtime has an integrated OPC UA server which can be configured via the CODESYS Development System. The data from the OPC UA server is received by an OPC UA client, ingested by the Data Services and written into the database in the Management System.

The Data Services are pre-configured to transmit a set of different kinds of CODESYS variables to the Management System. Use the following global variables to transmit data to the Management System:

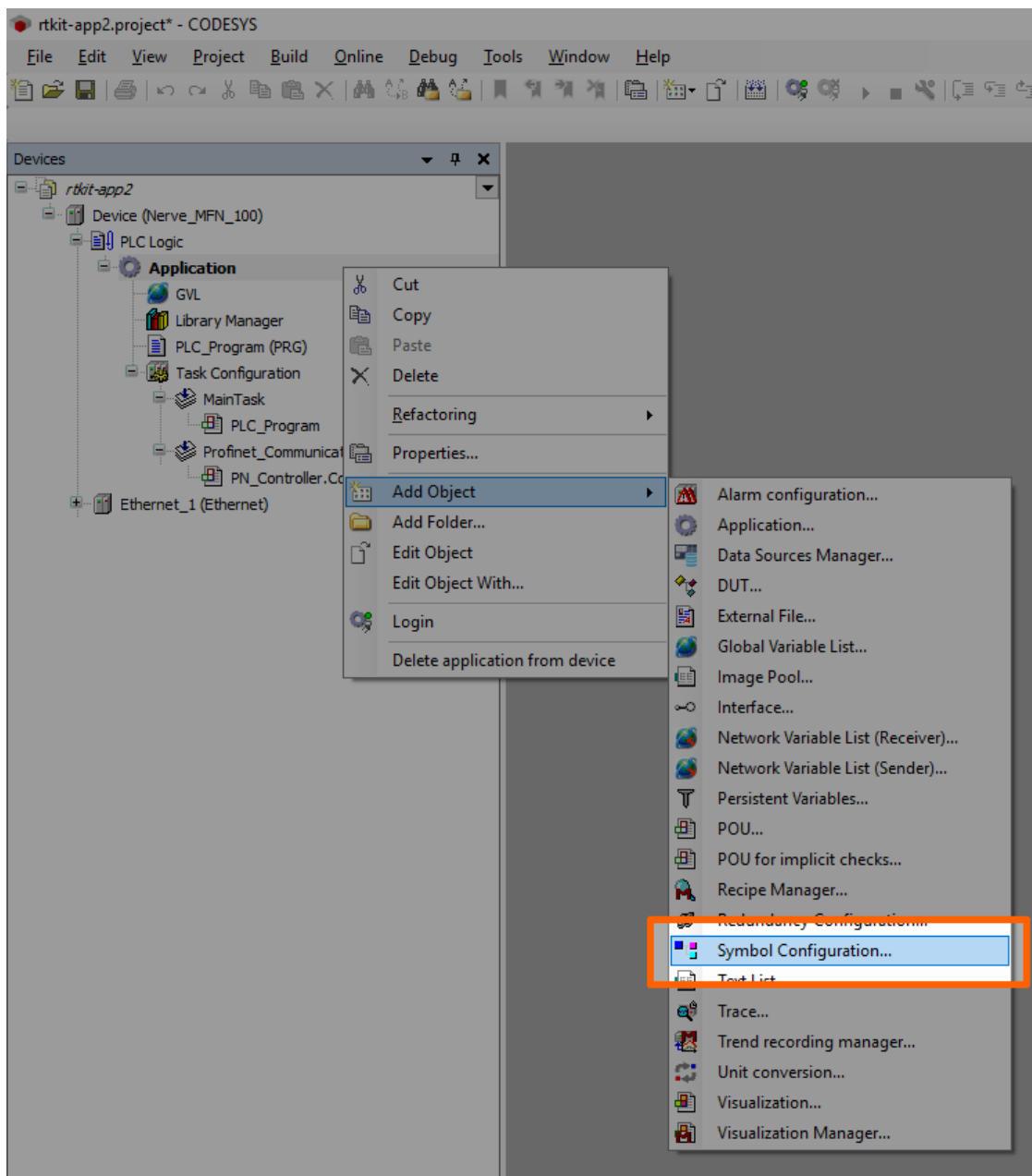
Global Variables			
BOOL_1	Integer_1	Real_1	STRING_1
BOOL_2	Integer_2	Real_2	STRING_2
BOOL_3	Integer_3	Real_3	STRING_3
BOOL_4	Integer_4	Real_4	STRING_4
BOOL_5	Integer_5	Real_5	STRING_5

The following instructions are only needed if new variables are added or a completely new CODESYS project is created.

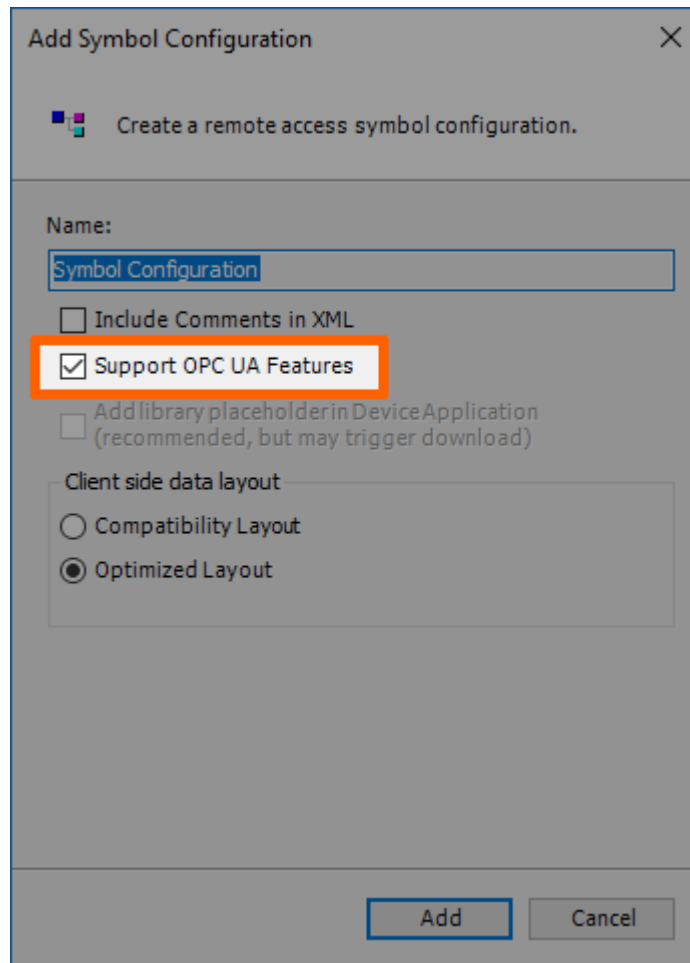
## Configuring the CODESYS OPC UA server

In the CODESYS Development System it is possible to configure which variables are available in the OPC UA server. Configuring variables requires an object called **Symbol Configuration**.

1. Open a CODESYS project to configure variables.
2. Expand **Device (Nerve\_MFN\_100) > PLC Logic**.
3. Right click **Application**.
4. Select **Add Object**.
5. Click **Symbol Configuration...**



6. Tick **Support OPC UA Features**.

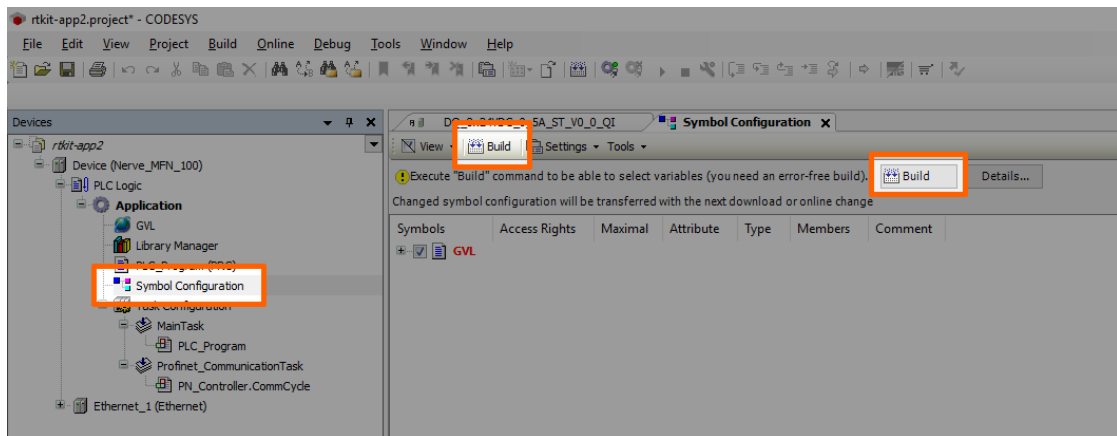


7. Click **Add**.

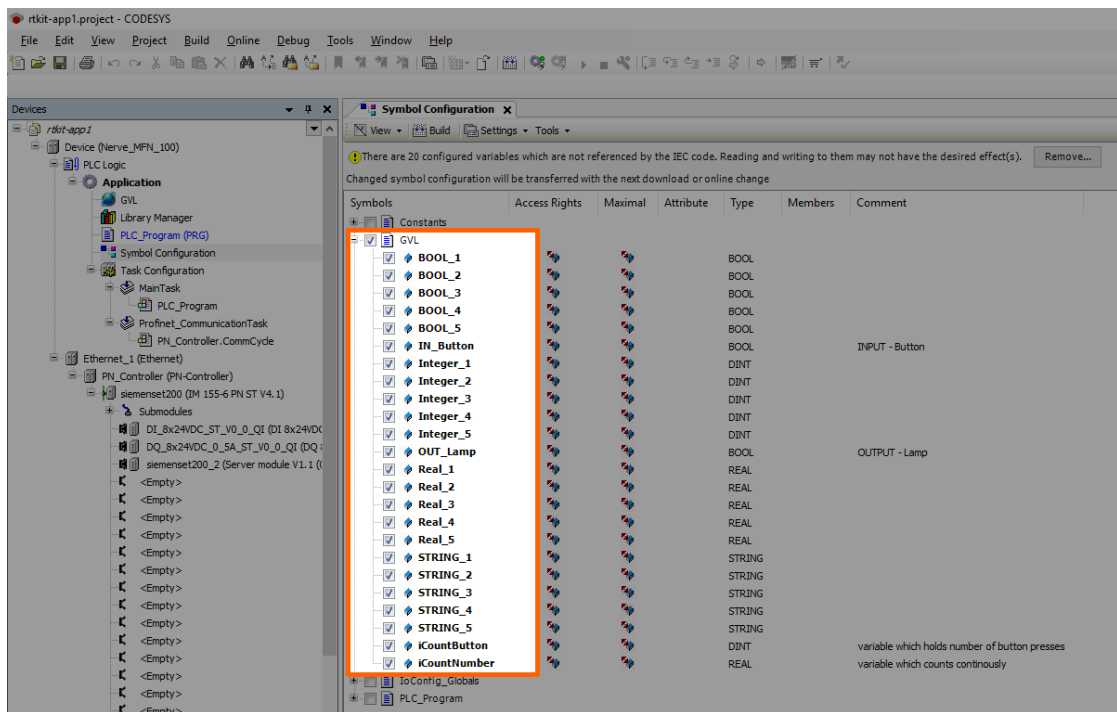
## Adding variables to the OPC UA server

Before adding variables to the OPC UA server, add the object **Symbol Configuration** to the tree structure. Refer to the instructions above on how to add the Symbol Configuration to the tree structure.

1. Open a CODESYS project to configure variables.
2. Expand **Device (Nerve\_MFN\_100) > PLC Logic > Application**.
3. Double-click **Symbol Configuration**.
4. Click **Build**.



5. Expand **GVL**.
6. Tick the variables to add to the OPC UA server.



## Adding variables to the Data Services Gateway

The Nerve-app Data Services Gateway has a pre-configured Gateway configuration loaded. This configuration needs to be adapted when new values are added through sensors, actuators and in the CODESYS Development System.

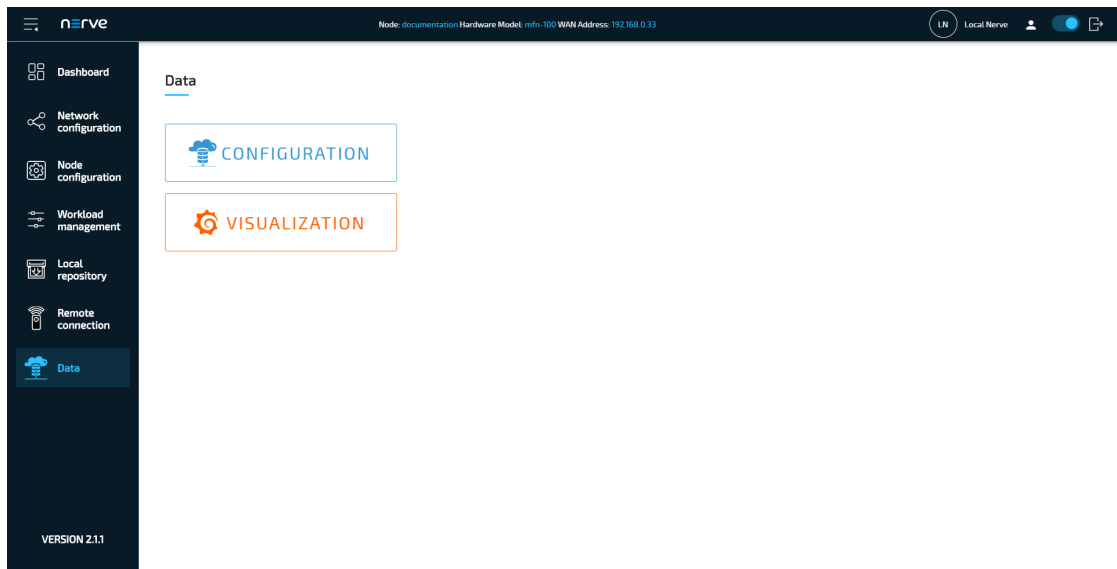
1. Connect the workstation to the console port **P1** of the MFN 100.
2. Configure the network adapter through which the workstation is connected to the MFN 100 the following way:

<b>IP address</b>	172.20.2.90
<b>Subnet mask</b>	255.255.255.0

3. Follow this link to reach the Local UI: <http://172.20.2.1:3333>.

4. Log in with the credentials from the customer profile.
5. Select **Data** in the navigation on the left.

6. Select **Gateway**.



7. Look for the following part of the currently loaded Gateway configuration:

```
"samplingIntervalAtServer_ms": 500,
"nodes": [
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.iCountButton",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.iCountNumber",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.BOOL_1",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.BOOL_2",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.BOOL_3",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.BOOL_4",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.BOOL_5",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Integer_1",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Integer_2",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Integer_3",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Integer_4",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Integer_5",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Real_1",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Real_2",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Real_3",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Real_4",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.Real_5",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_1",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_2",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_3",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_4",
  "ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_5"
```

8. Edit the last line the following way to add a new variable called myVar1:

```
"ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_2",
"ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_3",
"ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_4",
"ns=4;s= var Nerve_MFN_100 .Application.GVL.STRING_5",
"ns=4;s= var Nerve_MFN_100 .Application.GVL.myVar1"
```

The variable is added as a fully qualified nodeId. Refer to [Unified Automation](#) for more information.

#### NOTE

This nodeId can be found out with an OPC UA Client such as UA Expert. Connect a workstation to **P1** of the MFN 100 and connect to `opc.tcp://172.20.2.2:4840` in the OPC UA Client.

9. Select **Apply** to save the Gateway configuration. The Gateway will restart automatically.

![Apply Gateway configuration]../img/data\_opcua-ms01.png)

The data is automatically stored in the local TimescaleDB and also sent to the Management System. Refer to [Nerve Data Services](#) for more information.

## Visualizing new variables

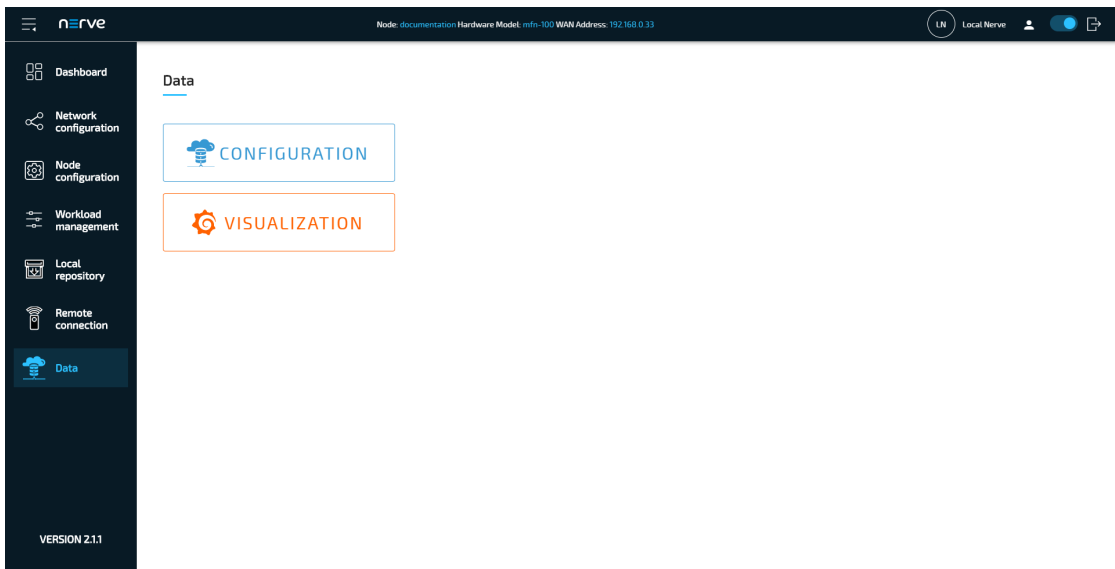
Newly added variables can also be visualized through the visualization element of the Data Services. The instructions below describe how to add new variables to the existing dashboard by adding a query. New dashboards can also be created for the new variables. Refer to [Creating a dashboard](#) for more information.

## Adapting the local data visualization on the node

1. Connect the workstation to the console port **P1** of the MFN 100.
2. Configure the network adapter through which the workstation is connected to the MFN 100 the following way:

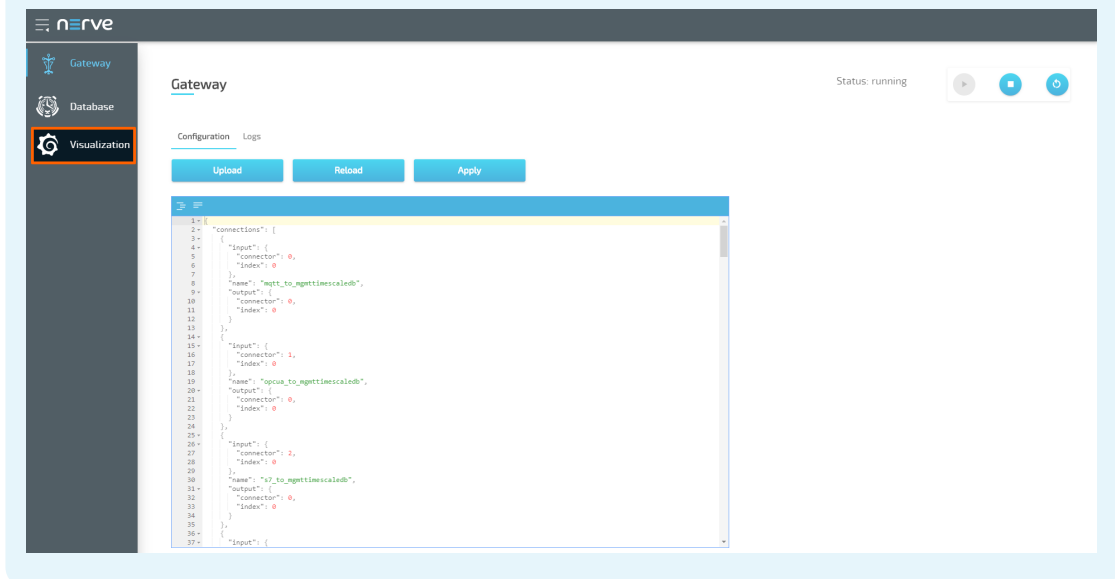
<b>IP address</b>	172.20.2.90
<b>Subnet mask</b>	255.255.255.0

3. Follow this link to reach the Local UI: <http://172.20.2.1:3333>.
4. Log in with the credentials from the customer profile.
5. Select **Data** in the navigation on the left.
6. Select **Data**.



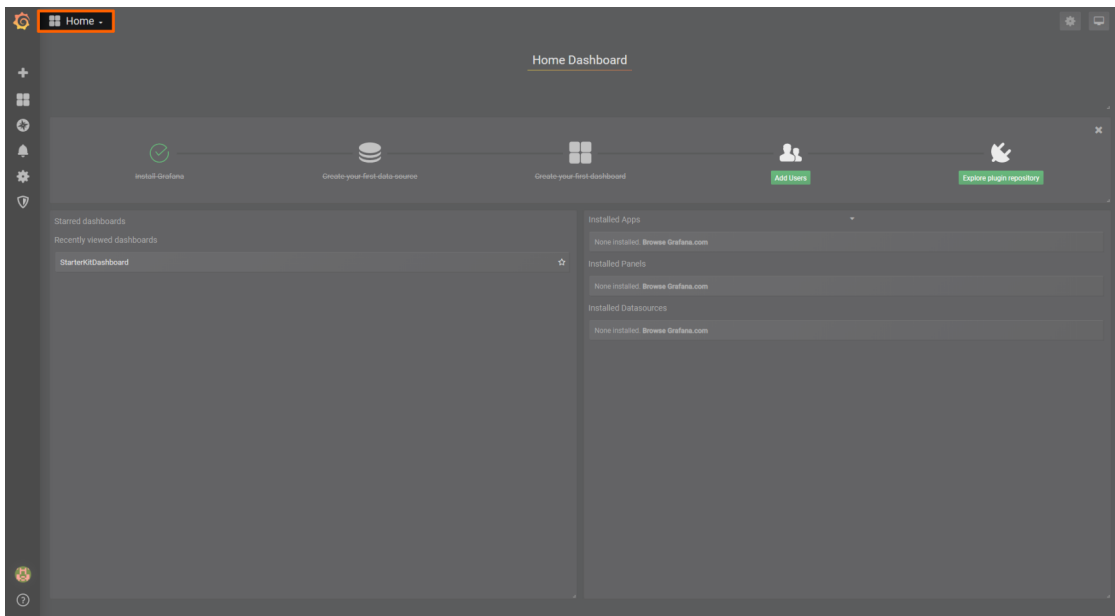
## NOTE

The visualization element can also be reached from the Data Services UI. When in the Data Services UI, select **Data** in the navigation on the left and select **Open** to reach the Grafana UI.

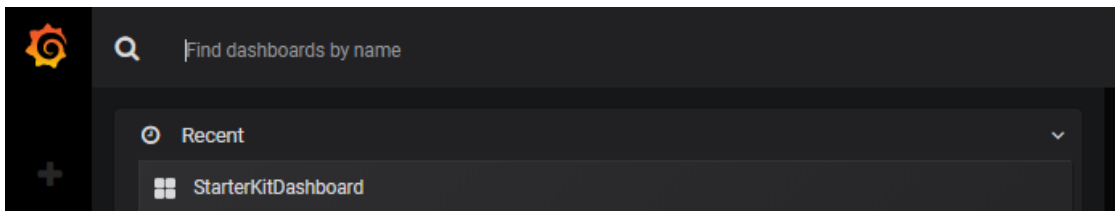


7. Select **Home** in the upper-left corner.





8. Select **Nerve Kit** underneath the search bar.

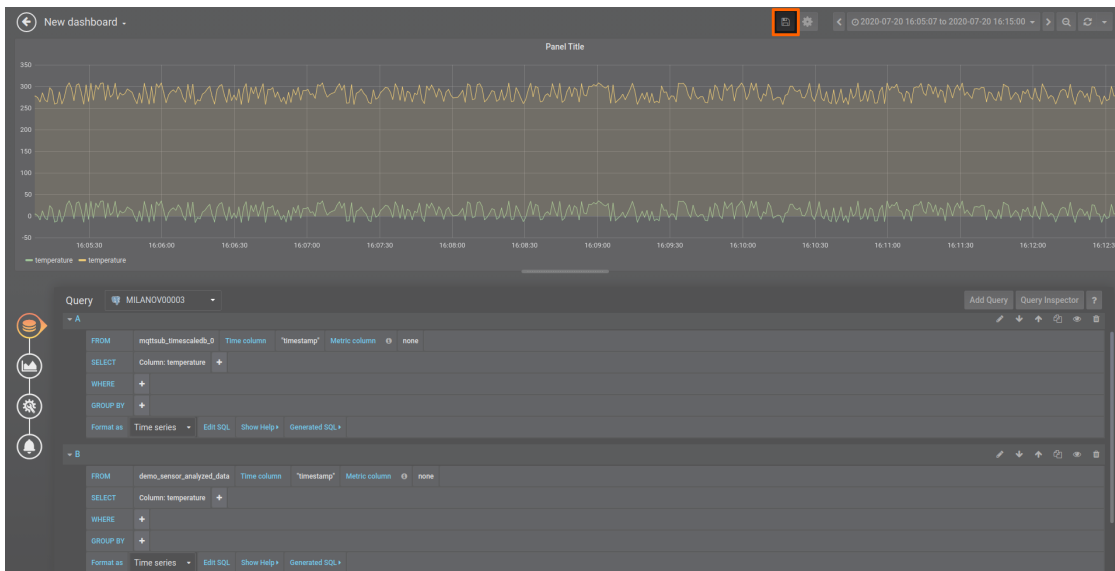


9. Select **Add Query** to the right to add a query for the new variable.

10. Fill in the following query information:

Setting	Value
<b>FROM</b>	Codesys_to_localdb
<b>SELECT</b>	Time column: "timestamp"
<b>Format as</b>	Column: myVar1
	Time series

11. Select the save icon in the upper-right corner to save the dashboard.



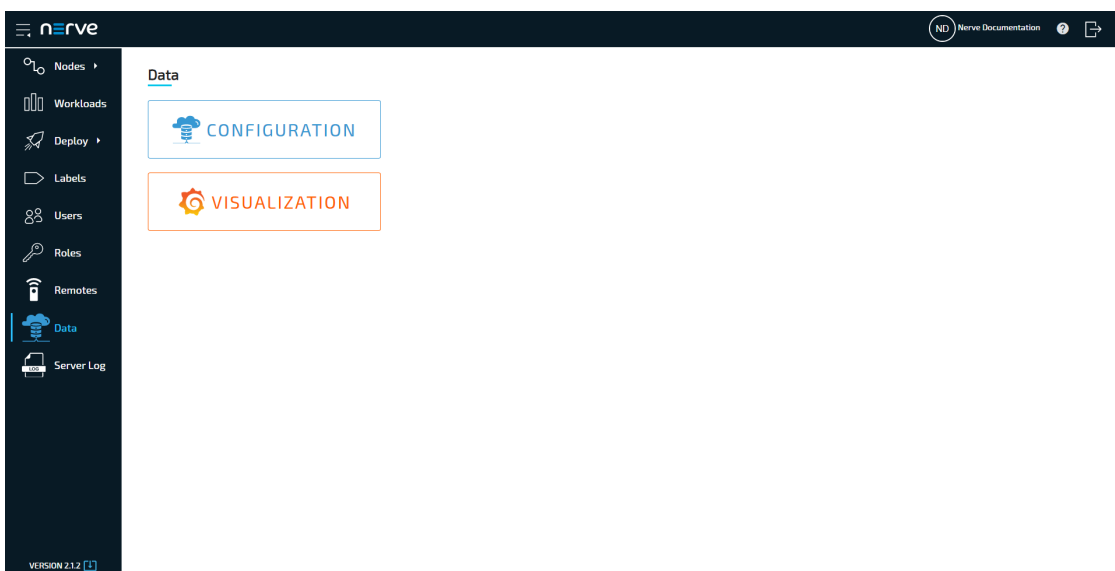
## Adapting the central data visualization in the Management System

1. Log in to the Management System.
2. Select **Data** in the navigation on the left.

### NOTE

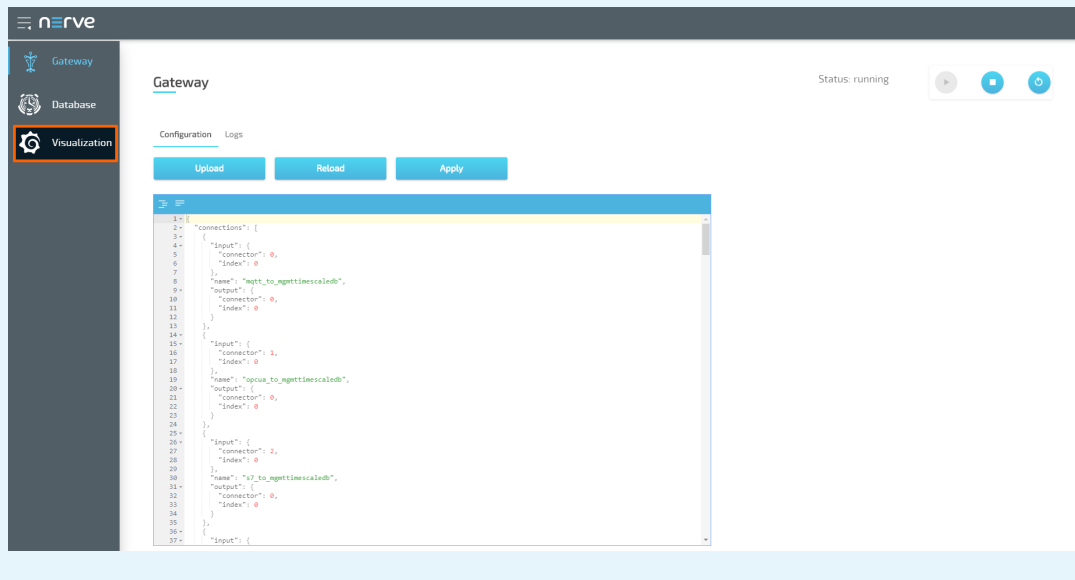
If the menu item **Data** is not available, make sure the logged in user has the permission to access the Data Services. Refer to [Assigning a role to a user](#) for more information.

3. Select **Data**.

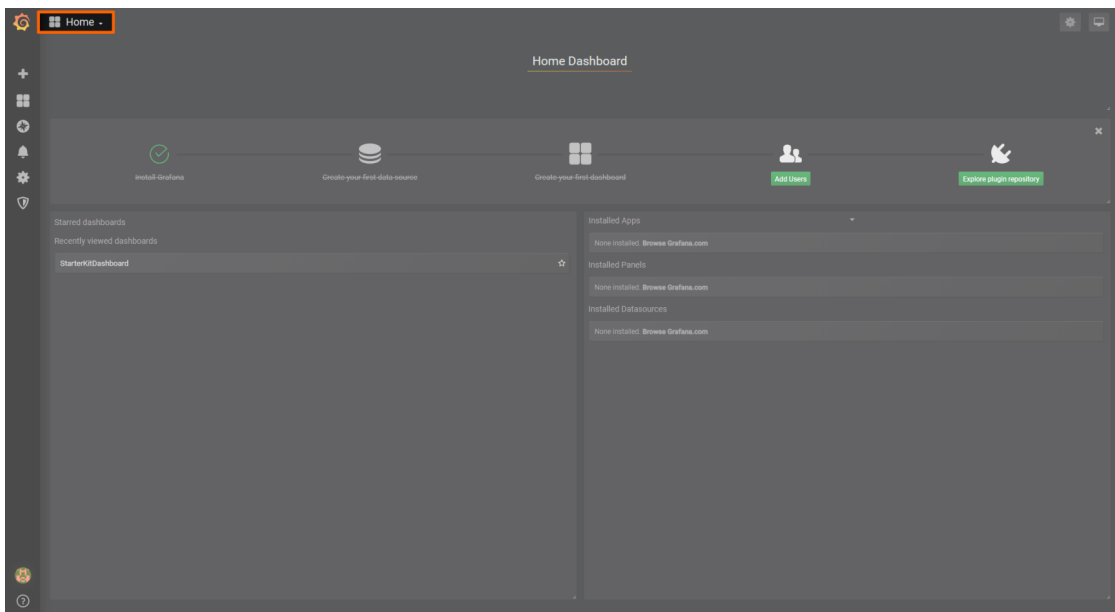


### NOTE

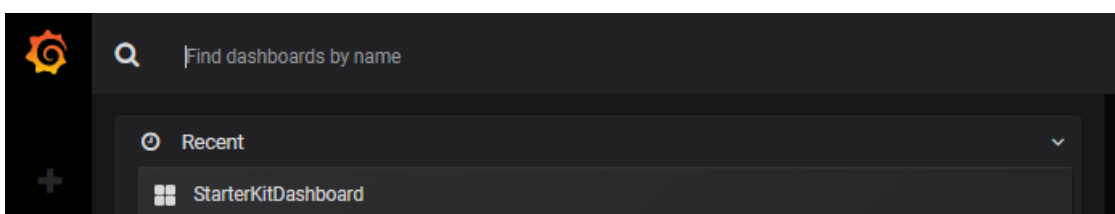
The visualization element can also be reached from the Data Services UI. When in the Data Services UI, select **Data** in the navigation on the left and select **Open** to reach the Grafana UI.



4. Select **Home** in the upper-left corner.



5. Select **Nerve Kit** underneath the search bar.



6. Select **Add Query** to the right to add a query for the new variable.

7. Fill in the following query information:

Setting	Value
<b>FROM</b>	ms_mqtt_broker_to_cloud_timescale_db
<b>SELECT</b>	Column: myVar1
<b>Format as</b>	Time series

8. Select the save icon in the upper-right corner to save the dashboard.

