Technical Document

# Containerized Niagara

**July 13, 2023**

niagara4

# Containerized Niagara

**Tridium, Inc.**
3951 Westerre Parkway, Suite 350
Richmond, Virginia 23233
U.S.A.

## Confidentiality



## Trademark notice



## Copyright and patent notice

# Contents

# About this guide

This topic contains important information about the purpose, content, context, and intended audience for this document.

### Product Documentation

This document is part of the Niagara technical documentation library. Released versions of Niagara software include a complete collection of technical information that is provided in both online help and PDF format. The information in this document is written primarily for Systems Integrators. To make the most of the information in this book, readers should have some training or previous experience with Niagara software, as well as experience working with JACE network controllers.

### Document Content

This document describes how to set up and use Containerized Niagara. Sections in this guide include chapters about container license, system requirements, and limitations.

# Document change log

Changes to this document are listed in this topic.

### Initial release publication: July 13, 2023

# Related documentation

Additional information is available in the following documents.

- *Niagara Platform Guide*

# Chapter 1 Overview

**Topics covered in this chapter**

♦ Features
♦ Using a container instead of a virtual machine
♦ Where to use containers
♦ Host system requirements
♦ Limitations

Containerized Niagara is a packaging mechanism for deploying and updating Niagara core. It is a Niagara framework stack that is packaged into a Docker® container with all required dependencies for easy deployment.

A container bundles the code defining a specific software application with its related configuration files, system libraries, system tools and dependencies such that the package is fully functional regardless of the cloud or non-cloud computing environment.

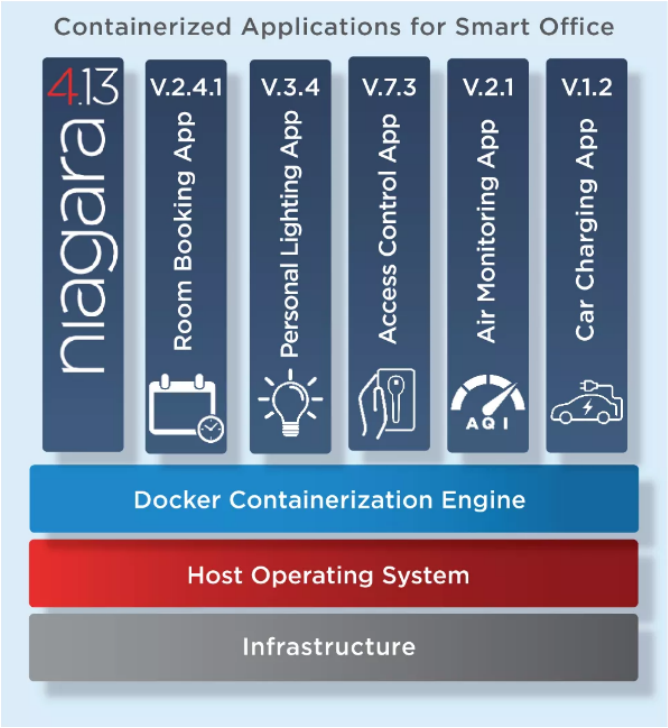
Advantages of using containers:

- Faster delivery of software
- More agile software development processes
- Easier portability of code from one computing environment to another

There are two primary use cases for deploying Niagara via container:

- Embedded deployment as an alternative to the Niagara Portability Software Development Kit. By bundling Niagara core (JRE, Niagara Framework, and modules) into a single deployment unit, you simplify the start and upgrade of Niagara systems compared to using a traditional Niagara Portability Software Development Kit (NPSDK) or Windows/Linux server installation.
- Niagara Development partner deploys Containerized Niagara to a third-party device or server.

**Figure 1** Example deployment of Containerized Niagara on a partner-built mobile hub




## Features

The following features are available in Containerized Niagara.

- The Niagara Framework is supplied in a Docker® container.
- You can use the hosting platform of your choice (for example, Azure, AWS, Google)
- If you use Enterprise Scale, you can choose the orchestration tool.
- AMD_x86 and Arm64 architecture are supported.
- Custom support is available for different architectures.
- Authorization models: file domain authorization or native domain authorization
- Subscription-based licensing using OPex instead of CAPex for purchasing flexibility
- Models that are released with Niagara 4.13:

| Type | Part # |
|---|---|
| Supervisor | NCC-SUP-0 |
| | NCC-SUP-1 |
| | NCC-SUP-10 |
| | NCC-SUP-100 |
| | NCC-SUP-500 |
| Supervisor Upgrade | NCC-SUP-UP-1 |
| | NCC-SUP-UP-10 |
| | NCC-SUP-UP-100 |
| Supervisor Device Pack | NCC-SUP-DEV-10 |
| | NCC-SUP-DEV-50 |
| | NCC-SUP-DEV-100 |
| | NCC-SUP-DEV-500 |
| | NCC-SUP-DEV-1000 |

| Type | Part # |
|---|---|
| Hardware Embedded | NCC-CPN-0001 |
| | NCC-CPN-0002 |
| | NCC-CPN-0005 |
| | NCC-CPN-0010 |
| | NCC-CPN-0025 |
| | NCC-CPN-0100 |
| | NCC-CPN-0200 |
| Hardware Embedded Device Pack | NCC-CPN-DEV-1 |
| | NCC-CPN-DEV-2 |
| | NCC-CPN-DEV-10 |
| | NCC-CPN-DEV-25 |
| | NCC-CPN-DEV-50 |

## Using a container instead of a virtual machine

Advantages using a container instead of a virtual machine are as follows:

- You do not need to maintain the host operating system.
- Containers require less storage and RAM overhead.
  - A container reuses the Linux Kernel of the host operating system.
  - Containers require less storage and RAM overhead.
    - A container reuses the Linux Kernel of the host OS.
    - Typically, a container is one hundredth of MB versus multiple GB for the virtual machine.
- Containers are more portable.

## Where to use containers

There are two primary use cases for deploying Niagara as a container: embedded deployment as an alternative for NPSDK (Niagara Portability Software Development Kit) and Supervisor deployment in a cloud service or on a server device locally.

Note that containers are not appropriate for devices targeted by Smart Edge SDK. If a device is capable of running a container, it can run full Niagara.

### Next generation NPSDK deployment

- OEM-specific engagement to support custom hardware
- Works out of box
- Differences from NPSDK
  - Faster onboarding time by OEM engineering team
  - Limited opportunities for customization. See also “Limitations”.

### Standard cloud service deployment

- N4Supervisor for AWS or Azure.

## Host system requirements

Before you start to use Containerized Niagara, check the host system requirements in this section.

- Linux x86_64 (ARM64) or Linux ARM64 system with Docker Engine installed.
  - Docker runtime on Windows or Mac is not supported.
  - (Optional) install Docker Compose to use `docker-compose.yml` for configuring containers.
- Docker volume support
  - Niagara stores station, history, and alarm data as standard files on volumes.
- RAM/disk requirements
  - RAM requirements vary by application. You can adjust the Java heap size allocated to Niagara station using container environment variables.
  - Standard Niagara container size on disk is about 350MB, which includes JRE and requires OS libraries and all standard modules.
- Container management
  - You can use Kubernetes or other orchestration tools to start, stop, upgrade, and configure the Niagara container.

## Limitations

The following limitations exist when using Niagara containers.

### Scalability

- Niagara containers do not provide horizontal scalability. It is a single instance of Niagara. You can increase the capacity by allocating additional memory or using a faster CPU.
- Containerized Niagara uses IP-based drivers only.
- Serial drivers and custom drivers such as BACnet MS/TP, CCN, BPort are not supported. The container engine does not permit applications running in container direct access to serial ports.

### No data recovery service

If the container is terminated or the hosting platform has a power loss, there is no support for data recovery. All data will be lost since the last station backup.

### Station only

Containerized Niagara is not suitable for deploying Workbench. You need another laptop or PC to interact with the container and run Workbench.

### Outbound internet connectivity required

A container deployment must be able to connect to the Entitlement Server to validate its license.

# Chapter 2 Licensing

**Topics covered in this chapter**

♦ Creating subscription license order
♦ Setting up a container license

Niagara container deployment is tightly coupled to the Niagara Entitlement Server (licensing service). The container has to reach out to the Entitlement Server at least once a day to re-authenticate and to receive the authentication token to continue running.

Traditional Niagara licensing uses a unique and permanent Host ID as a key to license. Containers are inherently portable and intended to operate in a variety of virtualized environments. Containers include a single hard-coded Host ID and node-locked license with an expiration date. A separate Host ID and thus container image is required for each license variant.

The Entitlement Server provides an online service to monitor and distribute licenses. While each container will be assigned a uniquely generated identity, the central Entitlement Server can detect abuses such as trying to reuse the same identity. Since a license must be periodically renewed, the opportunity for abuse is limited. In addition, the risk of abuse is mitigated by the following:

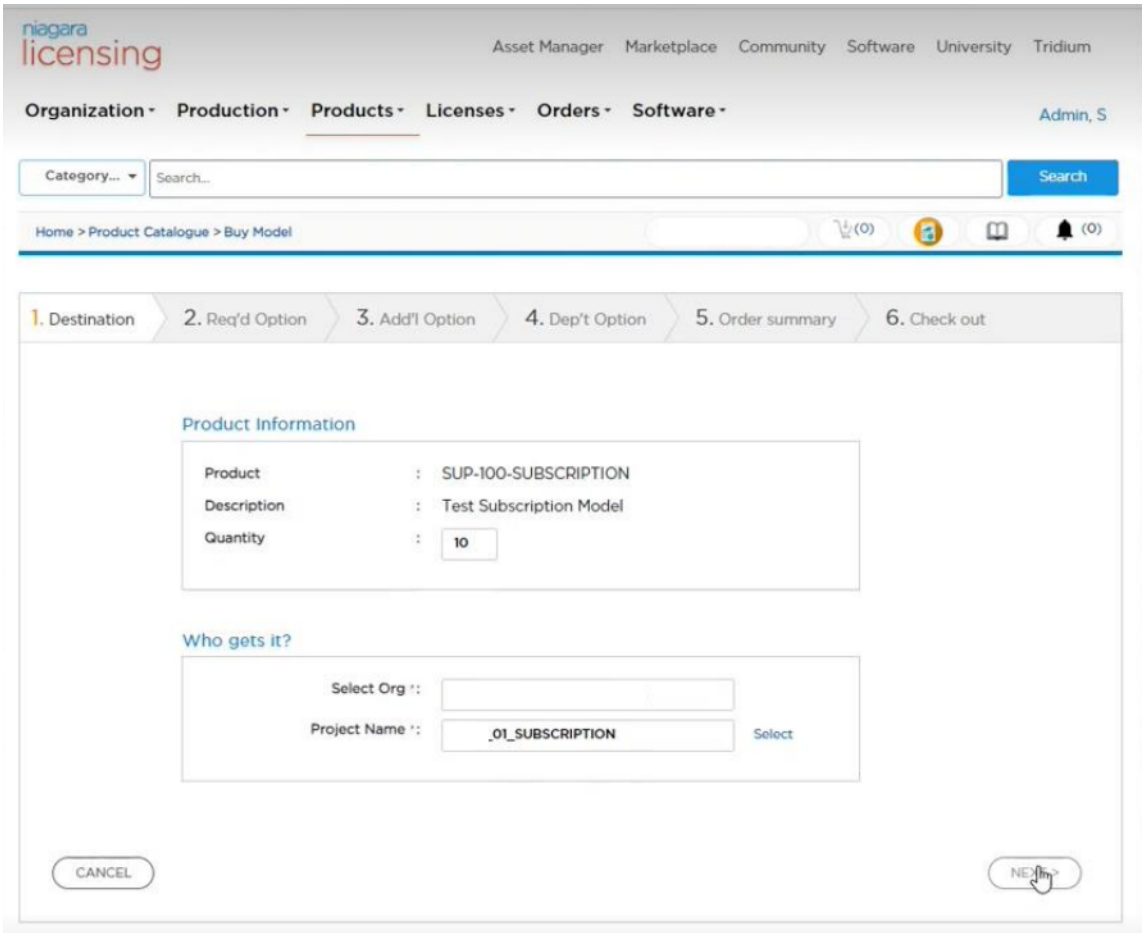
- Contractual auditing of units is deployed.
- Providing a unique container variant to customer.
- Each license has an expiration date.

## Creating subscription license order

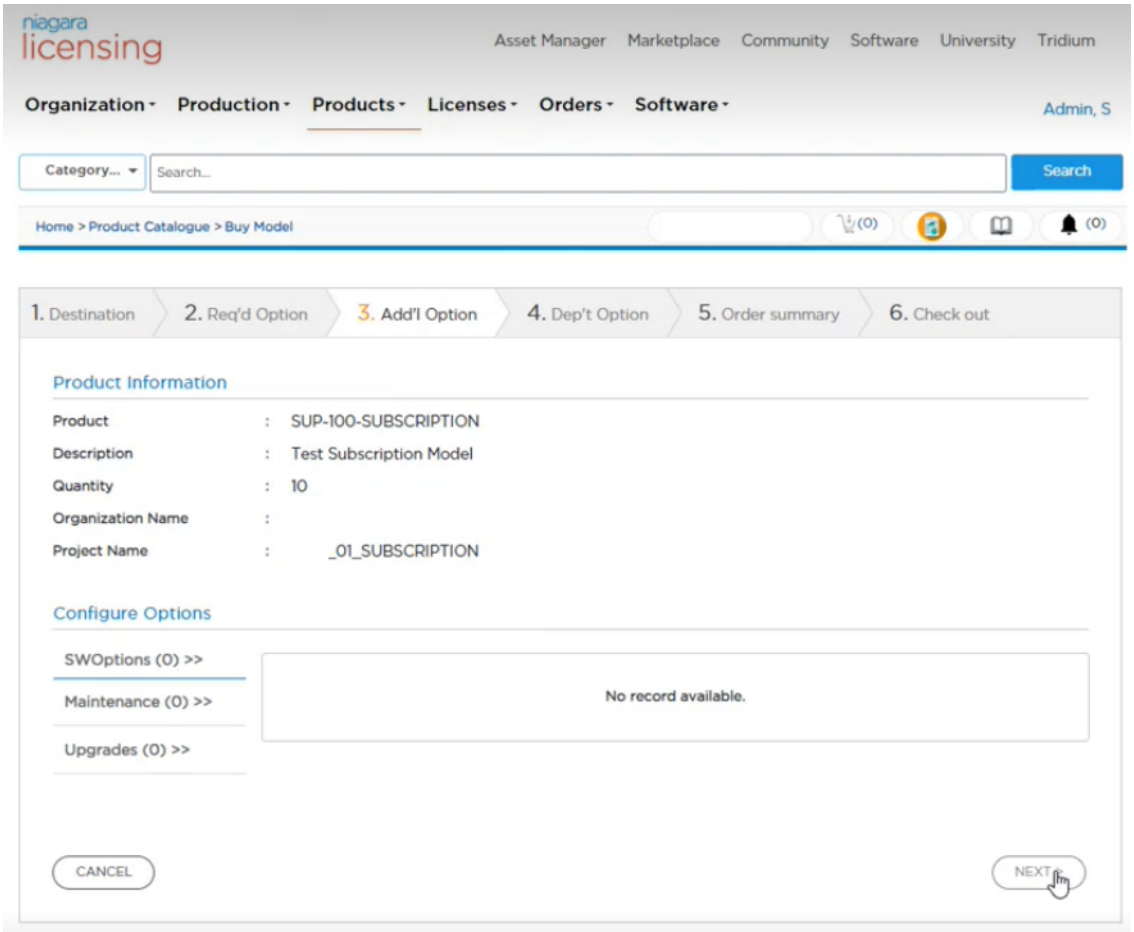
The following section describes how to create an order for a subscription license.

**Prerequisites:** You have created a Niagara Central account (https://www.niagara-community.com) and you have access to Niagara Licensing.

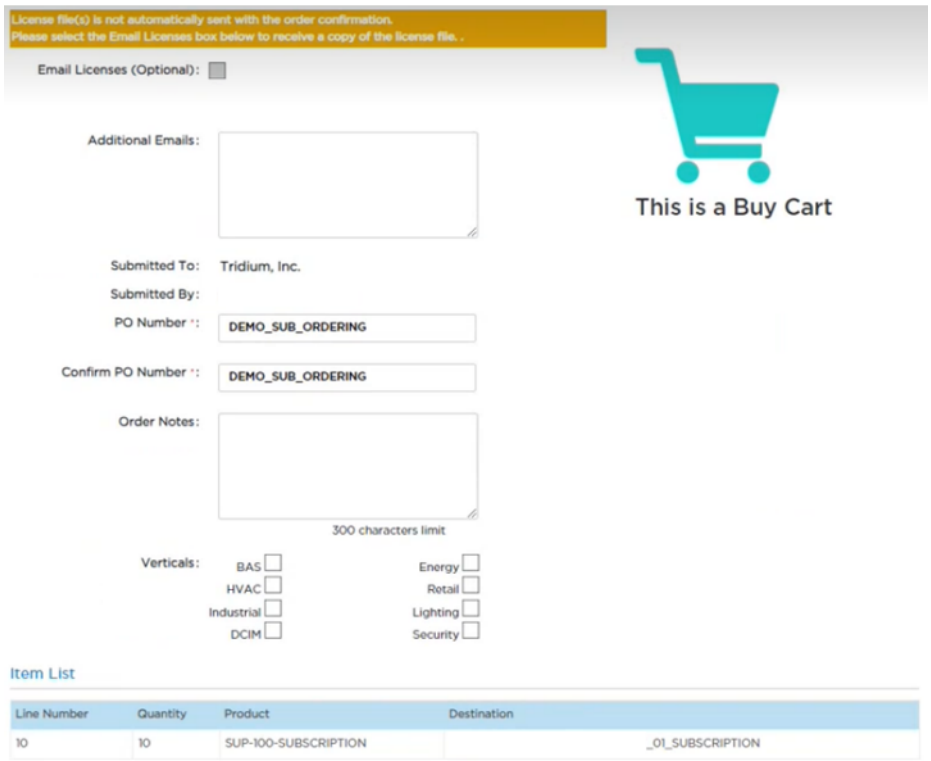
Step 1 To create a subscription license order, select **Products→Product Catalogue→Buy Model**, and click **Next**.



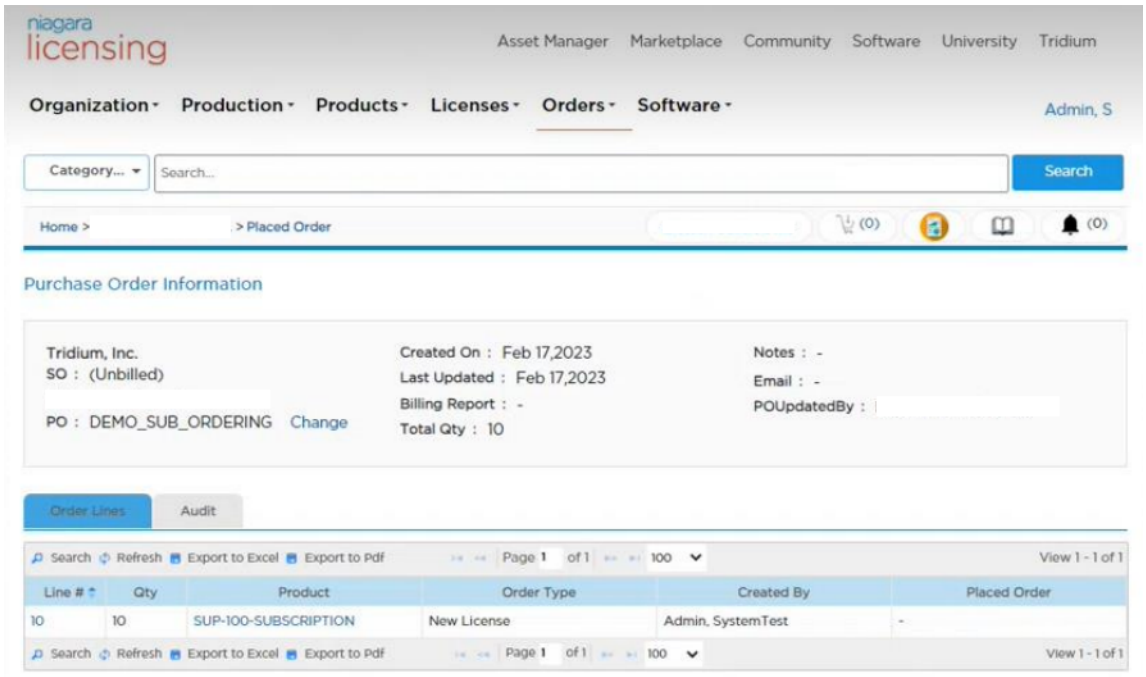

Step 2 On the **Add'l Option** tab, add upgrades and add-ons, and click **Next**.



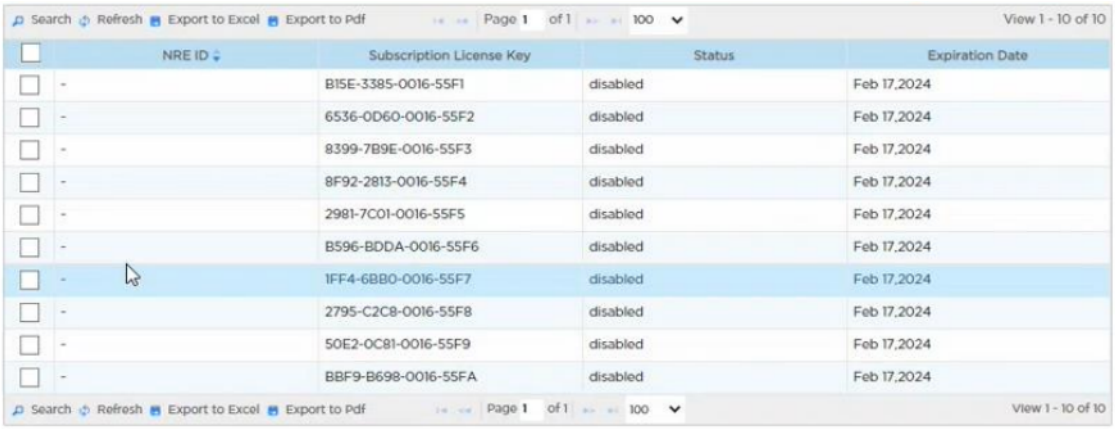

Step 3 On the subsequent tabs, enter additional order information, select the **Email Licenses (Optional)** checkbox to receive a copy of the license file, and on the **Check out** tab click the **Buy Cart** icon to place the order.



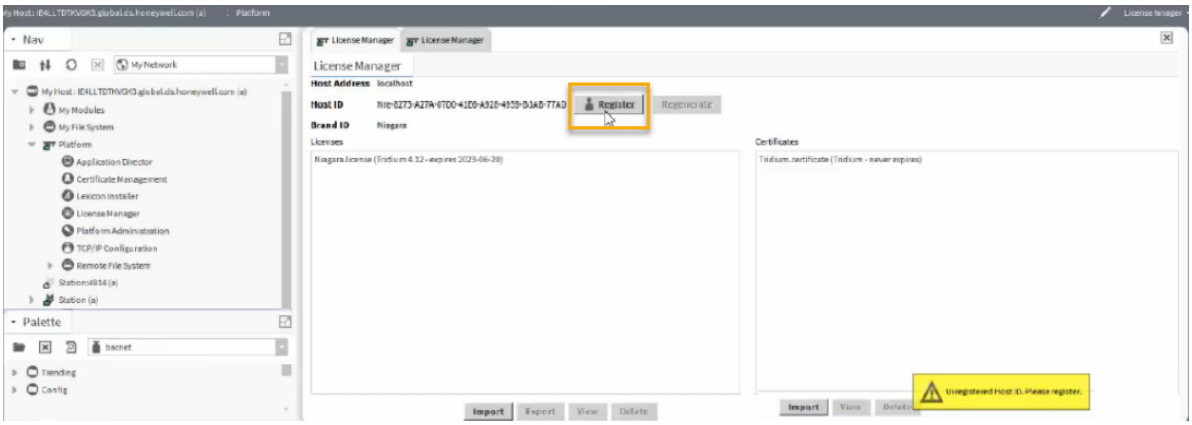

The subscription licensing order is complete.



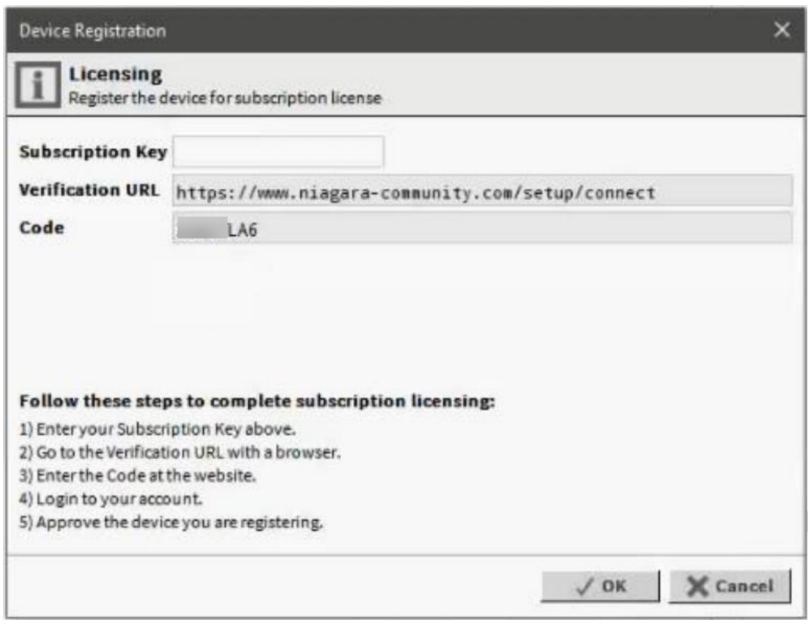

You can view the generated license keys, which are unbound, by clicking the appropriate license subscription in the **Product** column.



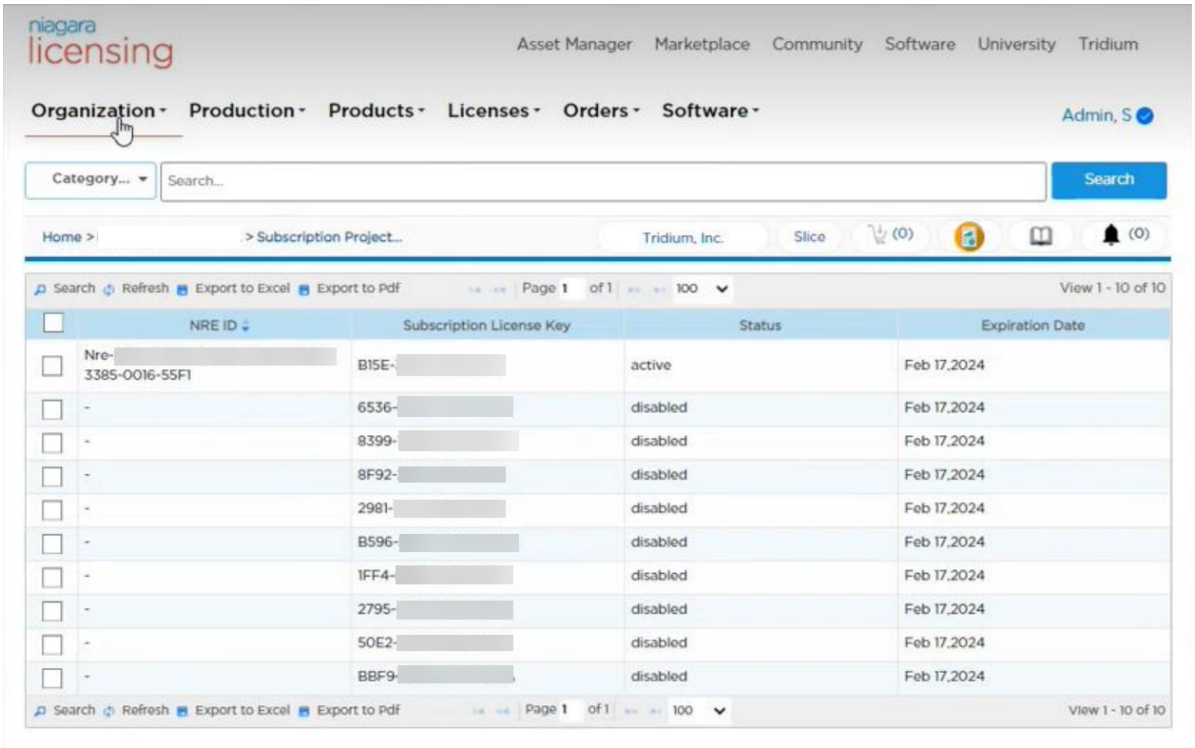

Step 4 To register the subscription instance, connect to the Workbench platform, navigate to the **License Manager** view, and click **Register**.




Step 5 From the newly created list of subscription license keys, paste the license key into the **Subscription Key** field, and follow the steps provided in the **Device Registration** window to complete the subscription licensing.



Step 6 To allow license binding, authorize the device.




Your subscription is activated and bound to Host ID (NRE-ID).

## Setting up a container license

In the following section, you learn how to set up a container license.

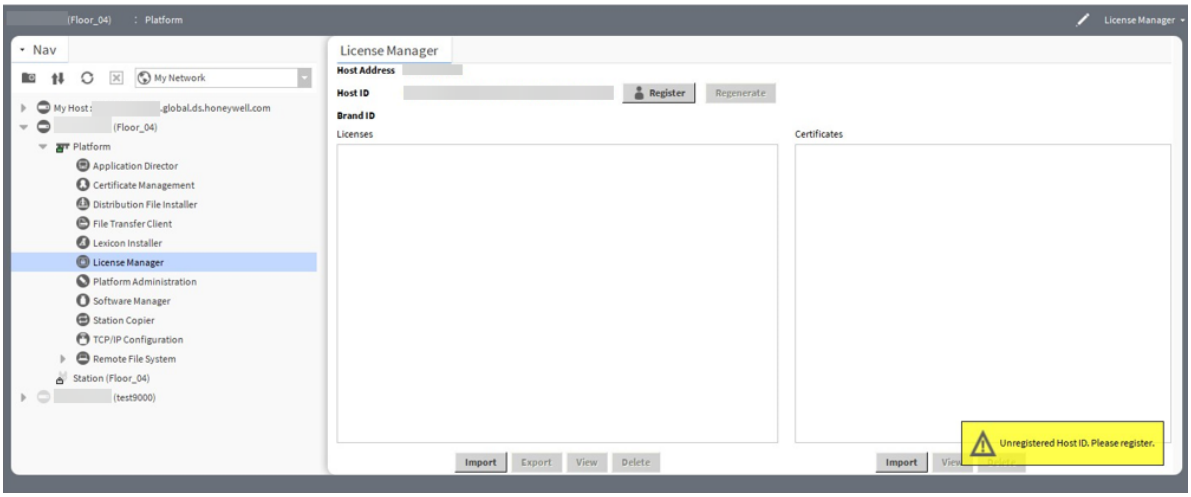
**Prerequisites:** You have created a Niagara Central account. Signing up is free and the account is free of charge. If you do not have an account, you can sign up here: *https://www.niagara-community.com/Sign_Up*.

Step 1 Start the Docker container.

Step 2 In Workbench, to open a platform connection, from the menu bar, select **File→Open→Open Platform**.

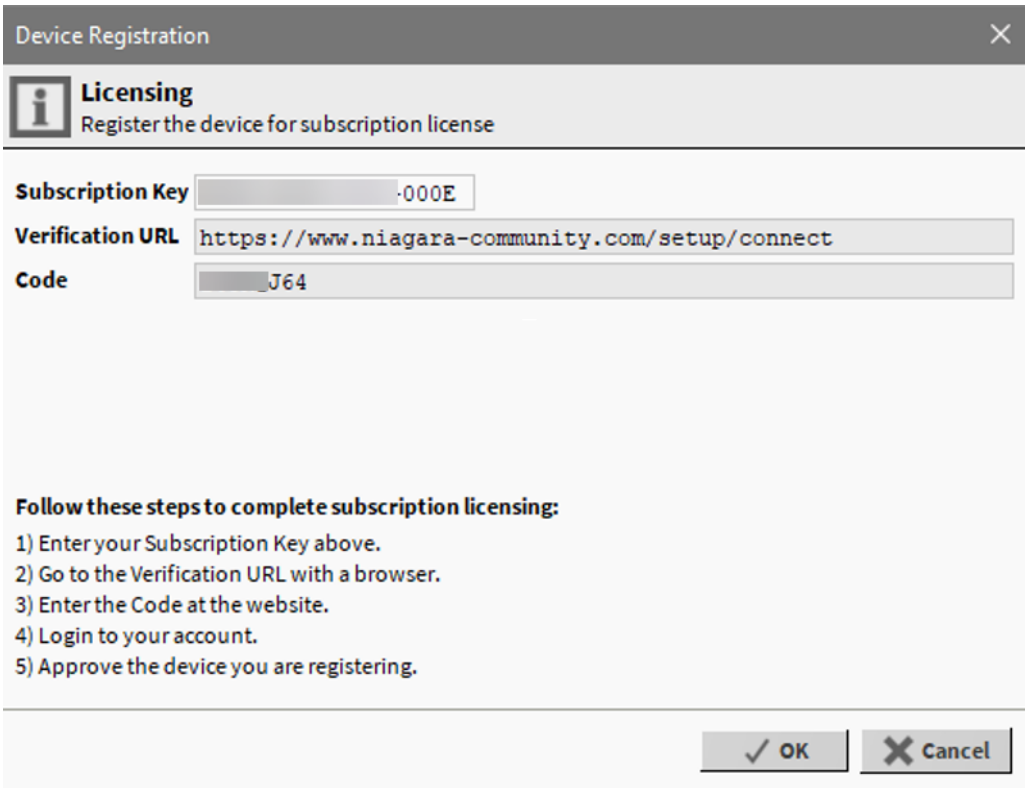
Step 3 Change default system passphrase and default platform credentials.

Step 4 Open the **License Manager** view.



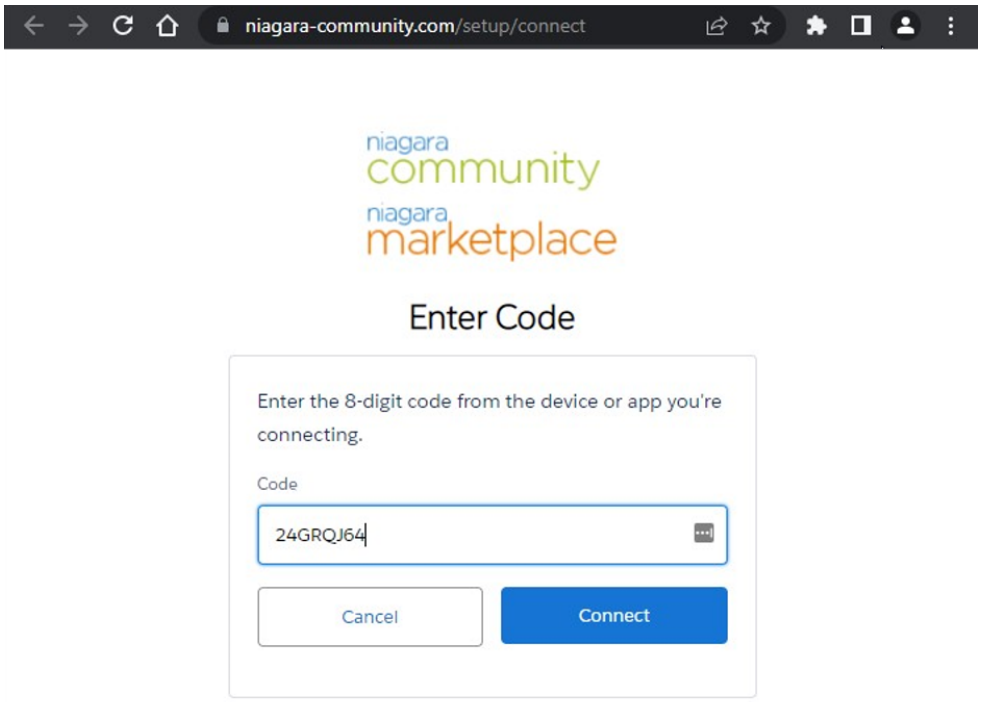

You should see the Host ID in the top left corner and a yellow warning dialog in the bottom right corner.

Step 5 Click **Register** and paste the subscription license key into the empty field.



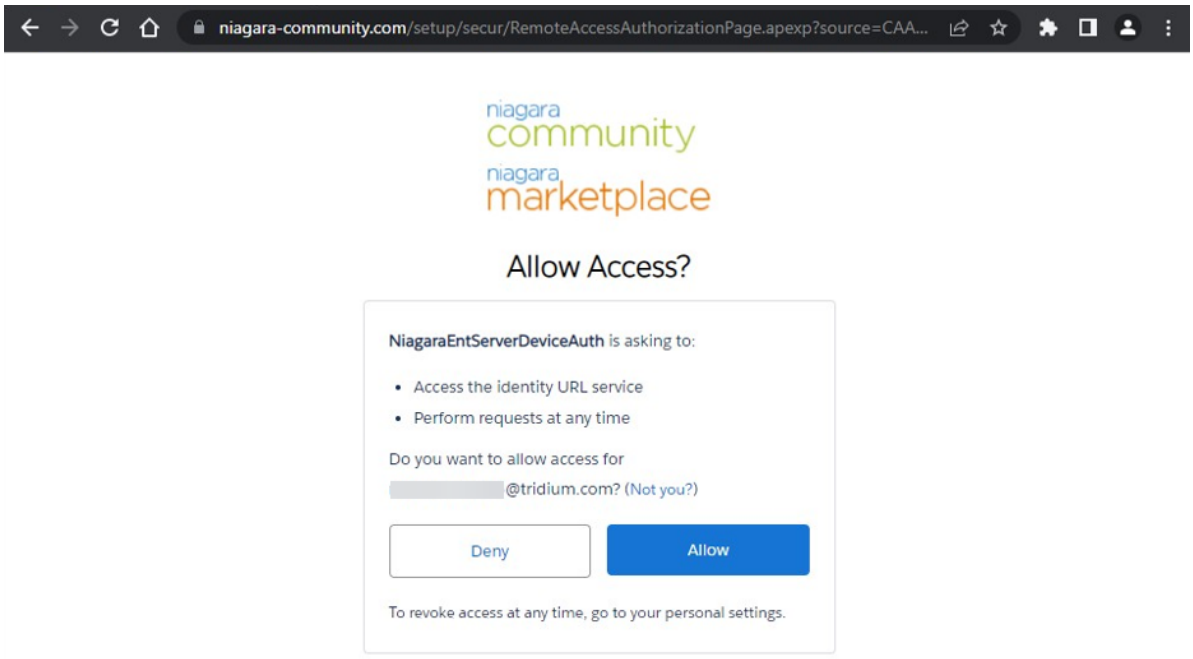

Step 6 To open the browser link, click in the **Verification URL** field.

Step 7 Paste the 8-digit code from the **Device Registration** dialog box into the **Code** field in the browser and click **Connect**.

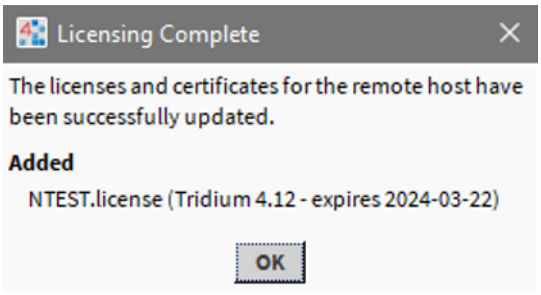



**NOTE:** If you were not previously logged into Niagara Central, provide your login credentials here.

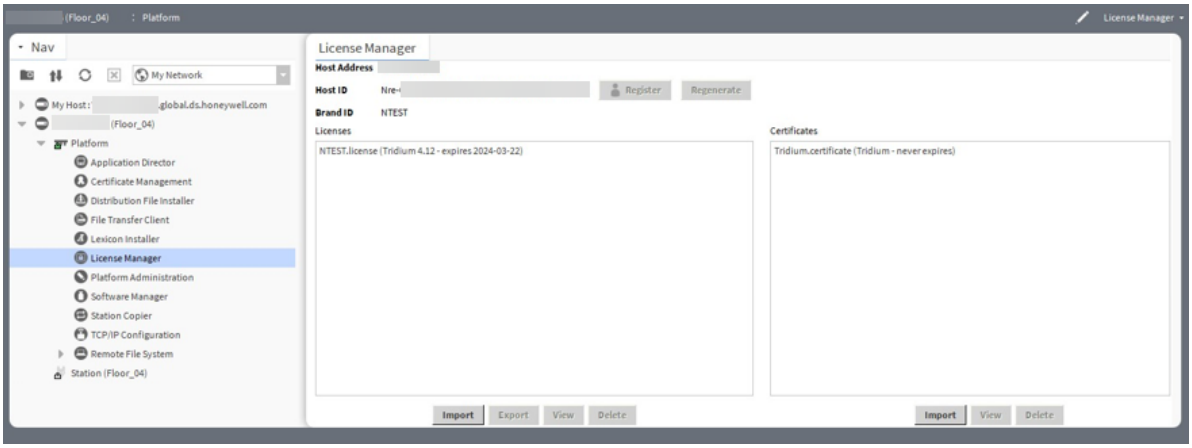
Step 8 On the **Remote Access Authorization** page, click **Allow** to complete license binding in Niagara Central, and click **Continue**.




Step 9 Return to Workbench and click **OK** in the **Device Registration** dialog box. A dialog box indicates whether the licensing of the device was successful or failed.



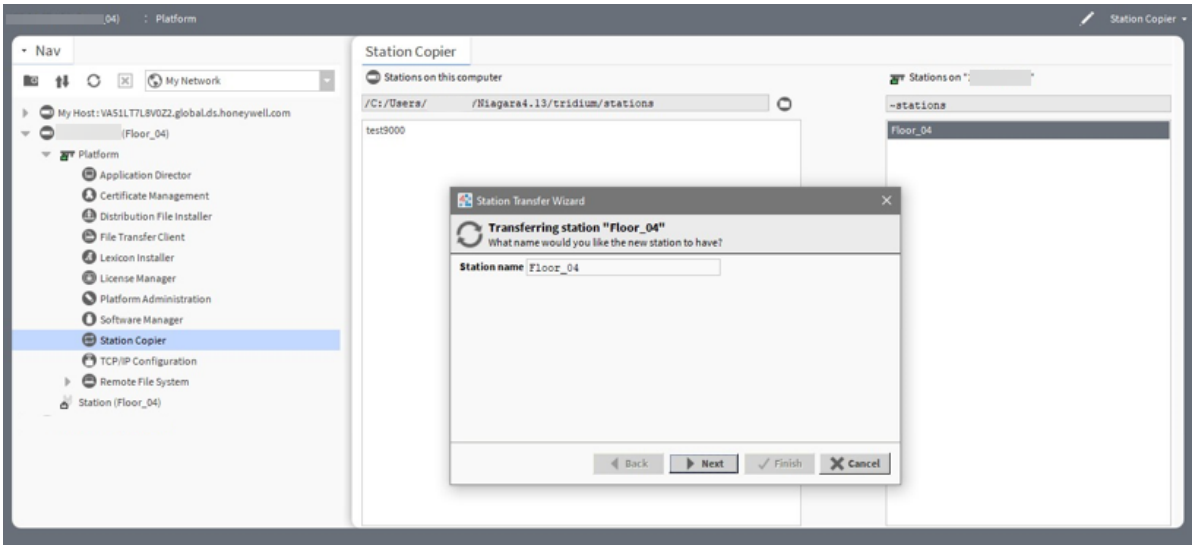

Both the license and the certificate are displayed in the **License Manager** view.



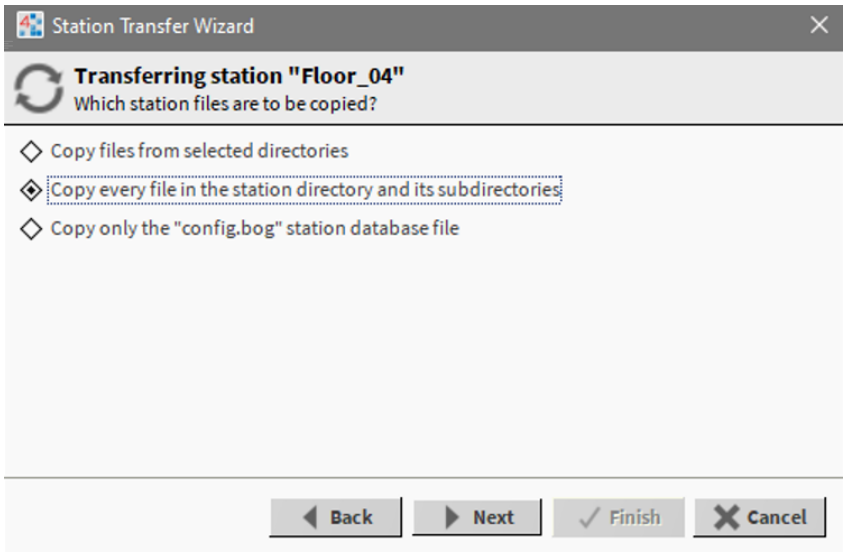

## Copying a station from a container version

This section describes how to copy a station from a previous container version.

Step 1 Before upgrading the container, open the station copier from the platform and copy the station to your local files to store it on your Workbench PC.




Step 2 Select the `Copy every file in the station directory and its subdirectories` option and click **Finish**.

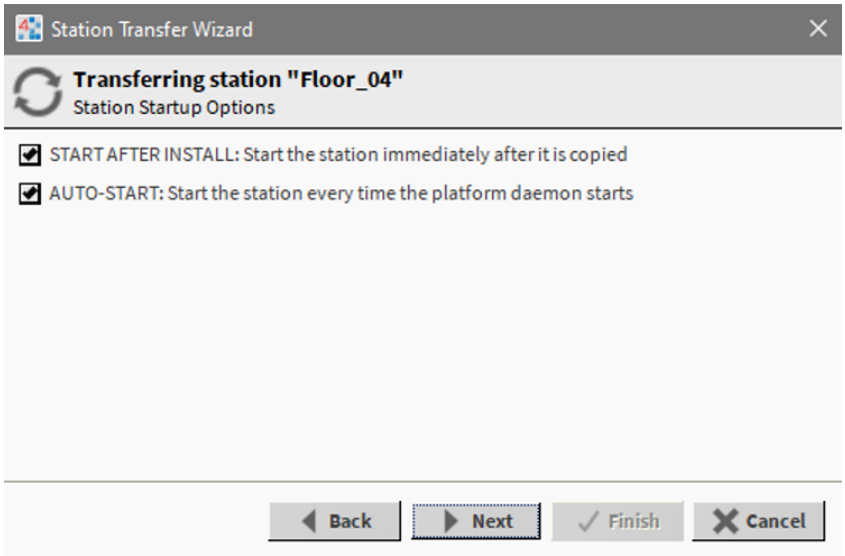



Step 3 Start a new container image and make a platform connection. Repeat the licensing steps above if necessary.

Step 4 Expand the platform container in the Nav tree, double-click the System Copier, select the station that you want to copy to the container and click **Copy**.

Step 5 Click **Next**, select the `Copy every file in the station directory and its subdirectories` option, and click **Next** again.

Step 6 Select `Start After Install: Start the station immediately after it is copied` and `Auto-Start: Start the station every time the platform daemon starts` options, click **Next**, and click **Finish** to complete.




You can check the status of the transfer in the **Transferring station** window.

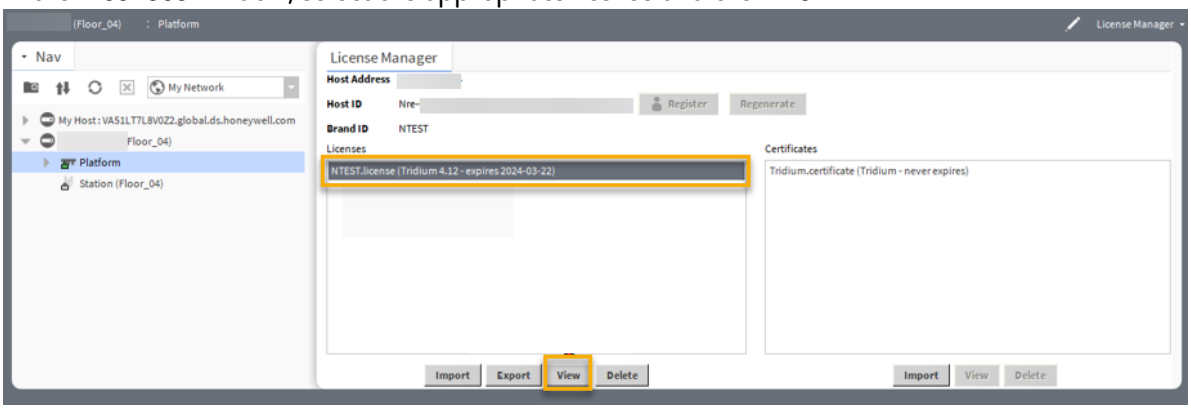
Step 7 Check the station status in the platform's **Application Director**.

## Verifying license check-in parameters

This section describes how to verify the check-in parameters of a license.

Step 1 Connect to the platform and open the **License Manager**.

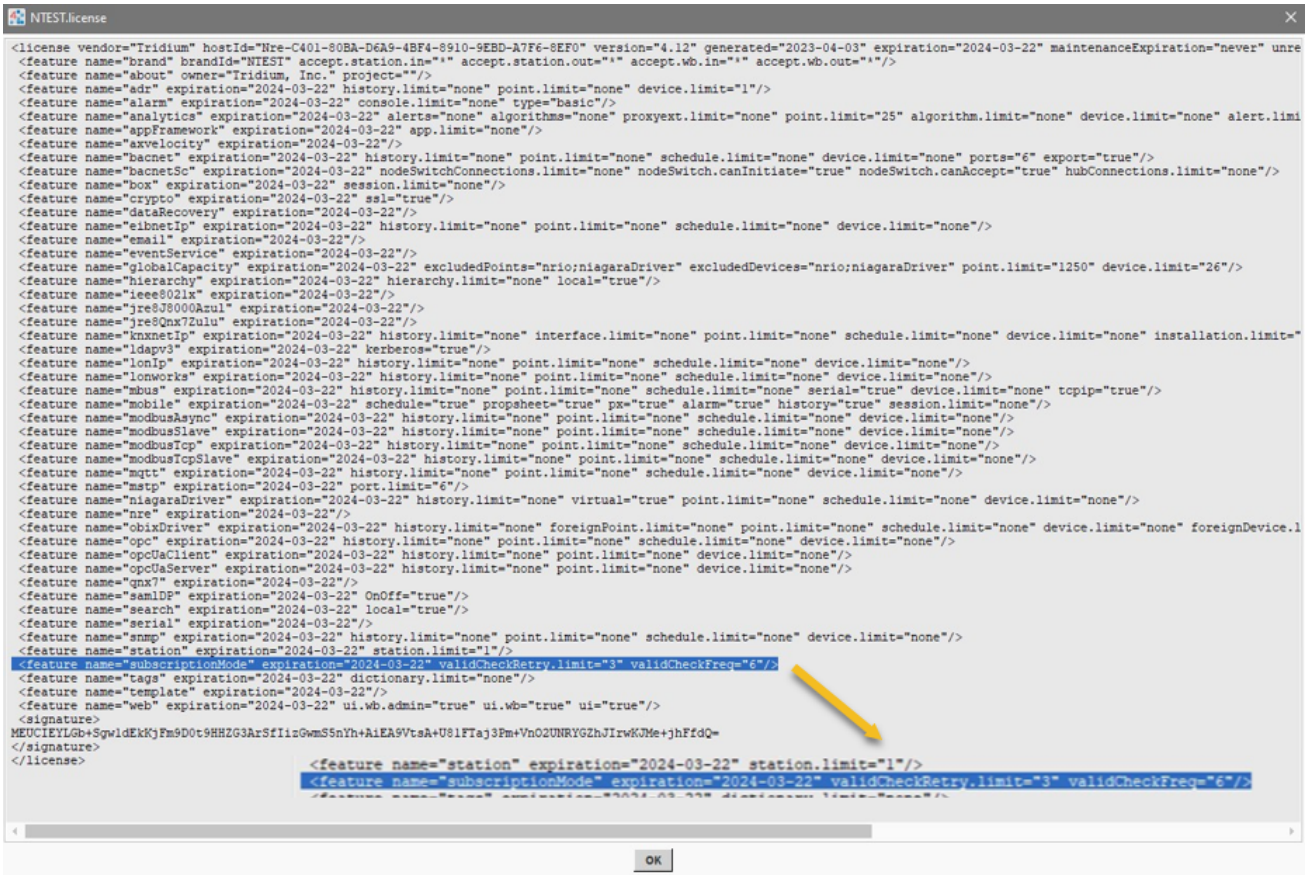
Step 2 In the **Licenses** window, select the appropriate license and click **View**.




Towards the bottom of the file, look for the `subscriptionMode` license feature. It indicates your check-in interval and number of allowed failed check-ins prior to the station shutdown.

**Example:**

```
<feature name="subscriptionMode" expiration="2024-03-22" validCheckRetry.limit="3" validCheckFreq="6"/>
```



# Glossary

| | |
|---|---|
| Carrier Communication/ Comfort Network (CCN) | The Carrier Communication/Comfort Network (CCN driver) integrates CCN devices and data into the Niagara Framework environment. |
| Niagara Portability Software Development Kit (NPSDK) | |
| Java SE Runtime Environment (JRE) | |

# Index