

Technical Document

Niagara 4 Platform Guide

August 18, 2015

niagara⁴

Niagara 4 Platform Guide

Tridium, Inc.

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

Confidentiality

The information contained in this document is confidential information of Tridium, Inc., a Delaware corporation ("Tridium"). Such information and the software described herein, is furnished under a license agreement and may be used only in accordance with that agreement.

The information contained in this document is provided solely for use by Tridium employees, licensees, and system owners; and, except as permitted under the below copyright notice, is not to be released to, or reproduced for, anyone else.

While every effort has been made to assure the accuracy of this document, Tridium is not responsible for damages of any kind, including without limitation consequential damages, arising from the application of the information contained herein. Information and specifications published here are current as of the date of this publication and are subject to change without notice. The latest product specifications can be found by contacting our corporate headquarters, Richmond, Virginia.

Trademark notice

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft, Excel, Internet Explorer, Windows, Windows Vista, Windows Server, and SQL Server are registered trademarks of Microsoft Corporation. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation. Tridium, JACE, Niagara Framework, NiagaraAX Framework, and Sedona Framework are registered trademarks, and Workbench, WorkPlaceAX, and AXSupervisor, are trademarks of Tridium Inc. All other product names and services mentioned in this publication that is known to be trademarks, registered trademarks, or service marks are the property of their respective owners.

Copyright and patent notice

This document may be copied by parties who are authorized to distribute Tridium products in connection with distribution of those products, subject to the contracts that authorize such distribution. It may not otherwise, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior written consent from Tridium, Inc.

Copyright © 2015 Tridium, Inc. All rights reserved.

The product(s) described herein may be covered by one or more U.S or foreign patents of Tridium.

Contents

About this guide	9
Document Change Log.....	9
Related documentation	9
Chapter 1 Niagara platform.....	11
Opening a secure platform connection.....	11
Platform overview	12
About a platform connection	12
Provisioning versus platform interface.....	14
Types of platform views	15
About platform differences.....	16
Embedded controllers	16
Windows-based	18
Models of platforms	20
File locations on Niagara 4 platforms	22
System home	23
Controller user home.....	23
Windows platform user homes.....	24
Station homes	26
Copying a new station to the daemon user home	27
Running a station from a Workbench User Home	28
Shared file strategy	28
Application Director	28
Installed applications (stations)	29
Application output	31
Station log levels (DebugService).....	33
Station log levels (spy:/logSetup).....	34
Application and output controls.....	35
Certificate Management view	39
Client/server relationships.....	39
User Key Store tab	40
Trust Store tabs.....	41
Allowed Hosts tab.....	43
Distribution File Installer.....	45
Operation of the Distribution File Installer.....	46
Restoring a backup dist	48
Wiping clean a JACE (Clean Dist)	50
Upgrading a controller	52
File Transfer Client	53
system.properties notes	55
Lexicon Installer	55
License Manager	57
License operations	58
About the licensing server	61
Synchronizing with the Supervisor license database	62

- Updating host licenses 63
- Updating licenses from the Network License Summary 63
- Platform Administration..... 64
 - Types of Platform Administration functions..... 65
 - View Details 66
 - User Accounts..... 66
 - Update Authentication 68
 - System Passphrase 70
 - Change HTTP Port 73
 - Change TLS Settings 73
 - Change Date/Time 75
 - Advanced Settings 76
 - Change Output Settings..... 76
 - View Daemon Output 77
 - Configure Runtime Profiles 78
 - Backup 81
 - Commissioning 83
 - Reboot 83
- Software Manager..... 83
 - Software Manager notes..... 84
 - About your software database..... 85
 - Default module listing and layout..... 86
 - Filtering displayed software..... 88
 - Software Import 89
 - Software actions..... 90
 - Right-click option to install earlier version..... 92
- Station Copier..... 93
 - Installing a station 93
 - Station copy direction..... 94
 - Station Copier Passphrase check..... 95
 - Station Copier dependencies check 96
 - Station Transfer Wizard 96
 - Renaming stations..... 103
 - Deleting stations 103
- TCP/IP Configuration 104
 - Configuring TCP/IP 104
 - TCP/IP Host fields 105
 - TCP/IP DNS fields 106
 - TCP/IP Interface fields..... 107
- Remote File System..... 110
- Chapter 2 Troubleshooting..... 111**
 - Station installation troubleshooting..... 111
- Chapter 3 Platform Services 113**
 - About Platform Services 113
 - Component differences for platform services 114

PlatformServiceContainer parameters	115
PlatformServiceContainer status values	115
PlatformServiceContainer configuration parameters	117
Model-specific PlatformServiceContainer properties	120
PlatformServiceContainer actions	120
SystemService (under PlatformServices)	121
Platform service types	122
Using platform services in a station	123
JACE power monitoring	123
PlatformServices binding and link caveats	123
About the NtpPlatformService	124
About the Ntp Platform Service Editor	125
About the Ntp Platform Service Editor Qnx	125
About the Ntp Platform Service Editor Win32	127
NTP port/firewall considerations	128
Chapter 4 Platform Component Guides	129
Components in platCrypto	129
platCrypto-CertManagerService	129
platform-DaemonSecureSession	129
Components in platDataRecovery module	129
platDataRecovery-DataRecoveryService	129
Components in platform module	130
platform-DefaultDaemonFileSpace	130
platform-DaemonSession	130
platform-LicenseDatabaseTool	130
platform-LicensePlatformService	130
platform-NtpPlatformServiceQnx	131
platform-NtpPlatformServiceWin32	131
platform-PlatformAlarmSupport	131
platform-PlatformServiceContainer	131
platform-SystemPlatformServiceQnxJavelina	131
platform-SystemPlatformServiceQnxNpm6xx	132
platform-SystemPlatformServiceWin32	132
platform-TcplpPlatformService	132
Components in platHwScan	132
platHwScan-HardwareScanService	132
Components in platPower module	133
platPower-ExternalSlaBattery	133
platPower-JavelinaBatteryPlatformService	133
platPower-NimhBattery	133
platPower-Npm2NimhBattery	134
platPower-NpmDualBatteryPlatformService	134
platPower-NpmExternalSlaBattery	134
platPower-PowerMonitorPlatformServiceQnx	134
Components in platSerialQnx module	134
platSerialQnx-SerialPortPlatformServiceQnx	134

platSerialQnx-SerialPortQnx	135
Chapter 5 Platform Plugin Guides.....	137
Plugin Reference Summary	137
Plugins in platCrypto	137
platCrypto-CertManagerView	137
Plugins in platDaemon module.....	137
platDaemon-ApplicationDirector	138
platDaemon-DistInstaller	138
platDaemon-DistributionView	138
platDaemon-FileTransferClient	138
platDaemon-LexiconInstaller	138
platDaemon-LicenseManager	139
Network License Summary	139
platDaemon-SoftwareManager	139
platDaemon-SoftwareView	140
platDaemon-PlatformAdministration.....	140
platDaemon-StationCopier	140
platDaemon-StationTextSummaryEditor	140
platDaemon-TcplpConfiguration.....	140
Plugin in platDataRecovery	140
platDataRecovery-DataRecoveryServiceEditor	140
Plugins in platform module	141
platform-LicensePlatformServicePlugin	141
platform-NtpPlatformServiceEditorQnx	141
platform-NtpPlatformServiceEditorWin32.....	141
platform-PlatformServiceContainerPlugin.....	141
platform-PlatformServiceProperties	141
platform-SystemDateTimeEditor.....	142
platform-SystemPlatformServicePlugin	142
platform-SystemPlatformServiceQnxPlugin	142
platform-TcplpPlatformServicePlugin.....	142
platform-WorkbenchLicenseManager.....	142
Plugins in platHwScan.....	142
platHwScan-HardwareScanServiceView.....	142
Plugins in platPower	143
platPower-JavelinaBatteryPlatformServicePlugin	143
platPower-PowerMonitorPlatformServicePlugin	143
Chapter 6 License Tools and Files	145
Workbench License Manager	146
Import File	146
Export File	147
Delete	148
Sync Online.....	148
Request License	149
About the local license database.....	150

Local license database rationale	151
Local license inbox	151
About license archive (.lar) files	152
About Niagara license files	152
Items common to all license files	153
JACE hardware features	154
Driver attributes.....	155
Driver types	156
Applications.....	159
Global capacity licensing	161
Example globalCapacity feature entry.....	162
Capacity licensing operation and recount	162
Checking capacity licensing status	162
Capacity licensing fault notifications.....	164
Capacity licensing notes about histories	165
Chapter 7 Time Zones and Niagara 4	167
Time zones and terminology	167
UTC	167
DST	167
Selecting a time zone in Niagara	168
About the historical time zone database.....	169

About this guide

This document provides information about Niagara platform services, components and plugins, license tools and other topics related to a Niagara host.

Document Change Log

Updates (changes/additions) to this document:

- Initial release publication: August 18, 2015

Related documentation

The following documents are related to the content in this guide and provide additional information.

- *AX to N4 Migration Guide*
- *Station Security Guide*

Chapter 1 Niagara platform

Topics covered in this chapter

- ◆ Opening a secure platform connection
- ◆ Platform overview
- ◆ About platform differences
- ◆ File locations on Niagara 4 platforms
- ◆ Application Director
- ◆ Certificate Management view
- ◆ Distribution File Installer
- ◆ Upgrading a controller
- ◆ File Transfer Client
- ◆ Lexicon Installer
- ◆ License Manager
- ◆ Platform Administration
- ◆ Software Manager
- ◆ Station Copier
- ◆ TCP/IP Configuration
- ◆ Remote File System

Platform is the name for everything that is installed on a Niagara host that is not part of a Niagara station. The platform interface provides a way to address all the support tasks that allow you to setup and support and troubleshoot a Niagara host.

Opening a secure platform connection

A platform (host) connection differs from a station connection in that when connected to a Niagara platform, Workbench communicates (as a client) to the host's platform daemon, niagarad (Niagara daemon), a server process. Unlike a station connection that uses the Fox/Foxs protocol, a client platform connection ordinarily requires full Workbench, meaning it is unavailable using a standard Web browser (Web Workbench applet).

Prerequisites: The platform (PC localhost or controller) has been physically installed and connected.

You are ready to begin working with your system.

Step 1 Launch Workbench.

Step 2 Right-click **My Host** in the Nav tree and click **Open Platform**.

The **Connect** window opens with the name of your computer as the **Host** name.

Step 3 To accept the host name, click **OK**.

The **Authentication** window opens.

Step 4 Do one of the following:

- If connecting to a controller, enter the credentials (user name and password) required by the controller.
- If connecting to your PC localhost, enter the credentials you use to log in to your computer.

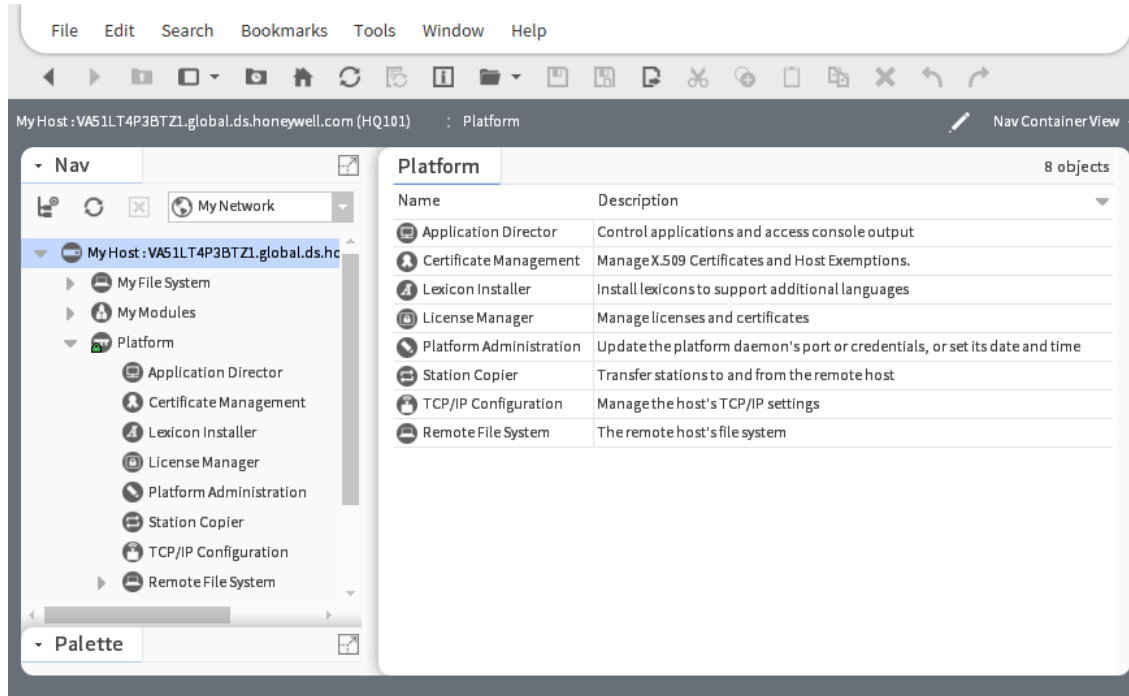
Step 5 Enable **Remember these credentials** and click **OK**

The system makes a secure connection between the host and Workbench.

Platform overview

In Workbench, when you open a platform connection to a Niagara host (whether controller or Supervisor), that host's available platform functions are listed in the platform's Nav Container View, as shown below.

Figure 1 Platform functions listed in platform's Nav Container View



Each platform function has its own Workbench view (plugin); you access it by simply double-clicking. Most of the same platform views exist whether a platform connection to a controller or a Supervisor, with these exceptions:

- If you open a *local* platform connection at your computer, note that some platform views appear to be missing, for example the **Distribution File Installer** and **Software Manager** are not in the list. These views have no application when working at your computer—instead, you simply use Windows Explorer.

Also, a few of the platform views differ depending on platform type. See [“About platform differences” on page 15, page 16](#) for details.

The following sections provide additional background on Niagara platform access:

- [About a platform connection, page 12](#)
- [Provisioning versus platform interface, page 14](#)
- [Types of platform views, page 15](#)
- [About platform differences, page 16](#)

About a platform connection

A platform connection is different than a station connection. When connected to a Niagara platform, Workbench communicates (as a client) to that host's *platform daemon* (also known as “niagarad” for Niagara daemon), a *server process*.

Unlike a station connection that uses the Fox protocol, a client platform connection ordinarily requires full Workbench, meaning it is *unavailable* using a standard Web browser (i.e. “Web Workbench” applet).

NOTE:



Browser access of a Supervisor station *can* provide platform connectivity, albeit indirectly, through its **ProvisioningService**. See [“Provisioning versus platform interface” on page 14, page 14.](#)

The following sections provide more details on a platform connection:

- [Platform connection session info, page 13](#)
- [Platform daemon \(niagarad\), page 13](#)
 - [Platform daemon port, page 14](#)
 - [Platform credentials, page 14](#)
 - [Platform access without a platform connection, page 14](#)
- [Platform daemon on a PC, page 14](#)

Platform connection session info

It is possible to open a *secure* (encrypted, TLS) platform connection to any Niagara 4 host, providing it is properly configured. The platform-connection session icon appears in the Nav tree with a small padlock to indicate this connection type, that is:

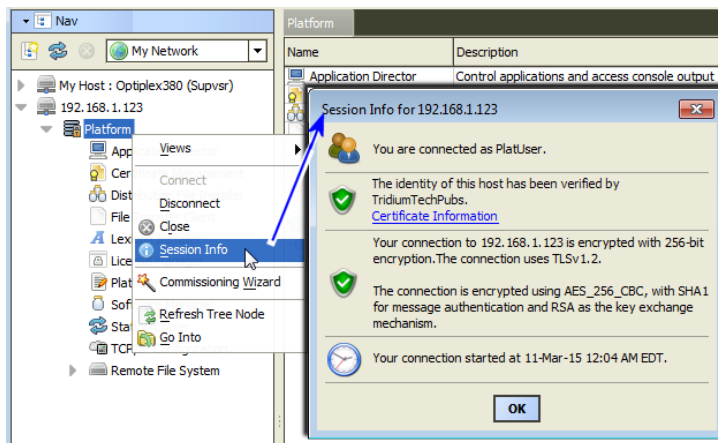
either  for secure platformtls using TLS (Transport Layer Security) encryption, or  for regular (unencrypted).

NOTE:

For best security, always use TLS. In Workbench, default **Open Platform** and **Open Station (Foxs)** assume a secure connection, where to connect in a regular (unencrypted) fashion you must change the connection **Type** first.

Once the platform is connected, the available platform functions are identical—regardless of connection method. Workbench provides a right-click **Session Info** action on any platform connection, as well as any station (Foxs) connection.

Figure 2 Example right-click Session Info dialog for a secure (TLS) platform connection



The figure above shows an example of this client session info from a secure (TLS) platform connection. In this example, the identity of the (server) has been verified by a signed certificate, and all data on this connection is being encrypted.



Platform daemon (niagarad)

The platform daemon is an executable that runs independently from Niagara core runtime, and is pre-installed on every JACE controller as factory-shipped, and runs whenever the JACE boots up. The daemon is

Java-based—running in its own Hotspot Java VM (Virtual Machine). An additional (and separate) Hotspot Java VM is used for the running the station process.

Platform daemon port

A Niagara host's platform daemon monitors a different TCP/IP port for client connections than does any running station. By *default*, this TCP port is either:

- 5011 - for a secure (TLS)  **Platform** connection (if available).
- 3011 - for a **Platform** connection that is not secure (unencrypted) .

If necessary, you can change either TCP port monitored to a different (non-default) port during Niagara platform configuration.

Platform credentials

Finally, as a platform client, you sign on using “host level” credentials for authentication. This means a user account and password separate from any station user account. Consider it the *highest level access to that host*.

CAUTION:

A new controller ships with *default* platform credentials that are widely known—and if left unchanged the controller is extremely susceptible to being hacked. Starting in the Niagara 4 startup commissioning process, you must change from defaults to something known only to your company and/or customers. For related details, see [“User Accounts” on page 53, page 66](#).

Platform access without a platform connection

A station user with admin-level permissions on the “Services” container (in the component **Config** space) of a running station also has access to a special subset of platform functions, via “Platform Services.” For details about this *different* type of platform access, see [“Platform Services” on page 87](#).

Platform daemon on a PC

When you install Niagara on your PC, one of the last “Would you like to?” install options is:

Install and Start Platform Daemon

The default selection is to install. You need the platform daemon locally installed and running to host a Niagara station on *your local PC*, such as for a Supervisor. This lets you open a Workbench client platform connection to your local (“My Host”) platform. It also allows *remote* client platform connections to your PC as well.

Once installed and started on a PC, you can see the platform daemon listed as a *Niagara service* from the Windows Control Panel, by selecting **Administrative Tools** → **Services**.

NOTE:

Alternatively, after Niagara installation on your PC, you can install and start the platform daemon at any time, if needed. From the Windows Start menu, do this with **Start** → **All Programs** → **Niagara <4.n.n>** → **Install Platform Daemon** (shortcut for “plat.exe installdaemon”).

In summary, your Workbench PC's local platform daemon is not necessary for making client platform connections to other Niagara hosts, only to provide the ability to run a station locally on your PC.

Provisioning versus platform interface

The focus in this document is about the Niagara platform user interface, meaning the different platform views and functions available when you (a Workbench user) open a direct platform connection to a Niagara host.

However, be aware that a Supervisor station can perform “provisioning”, which can automate some platform tasks. Provisioning typically applies to its subordinate JACEs, which are represented in the Supervisor station as Niagara Stations (devices) under its Niagara Network.

For more details, see the “Niagara Provisioning overview” in the *Niagara Provisioning Guide for Niagara Networks* document.

NOTE:

Some of the provisioning views provided by a Supervisor are nearly identical to platform views described in this document, including the **Software Manager** and **Application Director**, and work in the same fashion. However, if new to Niagara, it is recommended that you become familiar with “direct” platform views described in this document, before using provisioning in a Supervisor.

Types of platform views

A Workbench platform connection to a Niagara host, either JACE or Supervisor, provides various functional views.

NOTE:

In addition to the platform views listed below, a Commissioning Wizard is available as a right-click platform option. This wizard provides a “step-by-step” method to perform a sequence of platform tasks used for *Niagara commissioning* of a new JACE controller, or when *upgrading Niagara* in a JACE. For more details, see “About the Commissioning Wizard” in the *JACE Niagara 4 Install & Startup Guide*.

The following sections summarize the various Niagara platform functions and views, including typical usage:

- **Application Director**
To start, stop, restart, or kill a station on the Niagara platform. Output from the station displays in the view pane, useful for monitoring and troubleshooting. You also configure a station’s **Auto-Start** and **Restart on Failure** settings from this view. See [Application Director, page 28](#).
- **Certificate Management**
To import signed PKI certificates into the platform’s key store and trust store for TLS secure connections, and to perform related functions. Refer to *Station Security Guide*.
- **Distribution File Installer**
To restore a backup .dist file to the target controller, or to install a clean dist file to wipe the file system of a controller to a near-factory minimum state. See [Distribution File Installer, page 45](#).
- **File Transfer Client**
To copy files between your Workbench PC and the remote Niagara platform (in either direction). For example, you use this platform view when editing a controller’s `system.properties` file—once to copy it from the controller to your Workbench PC (for local editing), then afterwards to copy it back to the controller. See [File Transfer Client, page 53](#).
- **Lexicon Installer**
To install file-based Niagara lexicon sets from your Workbench PC to the remote Niagara platform, to provide non-English language support, or to customize English display of selected items. In Niagara 4, usage of this view and file-based lexicons may be atypical. See [Lexicon Installer, page 55](#).
- **License Manager**
To review, install, save, or delete licenses and (license) certificates on the remote Niagara platform. See [License Manager, page 57](#).
- **Platform Administration**
To perform configuration, status, and troubleshooting of the Niagara platform daemon. Included are commands to change time/date, backup all remote configuration, and reboot the host platform. Also included are functions to modify platform users, specify the TCP port monitored by the platform daemon, and various settings for a secure (TLS) platform connection. See [Platform Administration, page 18](#).
- **Software Manager**
To review, install, update, or uninstall Niagara modules (.jars) on the remote Niagara platform. The **Software Manager** compares modules installed on the connected platform against those available (locally) in **Sys Home** on your Workbench PC. See [Software Manager, page 83](#)

- Station Copier

To *install* (copy) a station from your Workbench **User Home** to a remote Niagara platform (or if a Supervisor, to the local PC's daemon **User Home**). Also to *backup* (copy) a station to your Workbench **User Home**, or to *delete* a remote station. You can also *rename* stations. See [Station Copier, page 93](#).

- TCP/IP Configuration

To review and configure the TCP/IP settings for the network adapter(s) of the Niagara platform. See [TCP/IP Configuration, page 104](#).

- Remote File System

For read-only access to folders and files on the remote platform, including all those under its Niagara system home (**Sys Home**) and daemon **User Home**. See [Remote File System, page 110](#).

About platform differences

Depending on the platform *type* opened, some platform views differ. In the initial Niagara 4.0 release, there are two main categories of platforms, by OS (operating system) used. These include:

- [Embedded controllers, page 16](#) (JACE controllers)
- [Windows-based, page 18](#) hosts (Supervisor hosts)

There are various controller host *models*, each with a "model" string descriptor. For a list of host models that support Niagara 4, current with this document, see ["Models of platforms" on page 19, page 20](#).

Embedded controllers

Sometimes called "embedded" JACE controllers, these include the newest JACE-8000 controllers as well as JACE-3,-6,-7 series models, all shipped with the *QNX operating system*. All use *flash memory* for file storage, Oracle's Sun Hotspot Java VM, and provide wired Ethernet connectivity.

The JACE-8000 platform, introduced with the release of Niagara 4, is the most powerful JACE controller. It provides a number of exclusive features such as integral WiFi (802.11b/g) support, backup and restore to/from a removable USB drive, and easy communications expansion using attachable modules.

JACE-8000 controllers initially support Niagara 4 only—a later update to AX-3.8 may provide JACE-8000 support. Whereas the JACE-3,-6,-7 series controllers were originally released as NiagaraAX, and may be migrated to Niagara 4 if already running AX-3.8, or else configured "from scratch" using Niagara 4.

See the following for further details on JACE controllers:

- ["Backup Battery \(or not\)" on page 16, page 16](#)
- ["Platform view differences, JACE controller vs. Windows-based host" on page 16, page 17](#)
- ["Models of platforms" on page 19, page 20](#)

Backup Battery (or not)

JACE-6 and JACE-7 controller models use an onboard NiMH backup battery (nickel metal hydride), used to preserve runtime data, and also allow continuous operation during brief power outages. The JACE-3E and JACE-6E controllers use integral SRAM to backup runtime data, but can also *optionally* use an onboard NiMH battery for continuous power event operation. The JACE-7 and JACE-603/JACE-645 models also support an additional external 12V SLA (sealed lead acid) battery for backup usage.

For any of the above controller models, the JACE station provides a "power monitoring" component to track its AC power and backup battery level, with a configurable delay for orderly shutdown of the JACE upon AC power failures. Access power monitoring of a JACE in the **PowerMonitorService** in a running *station*. See ["About Platform Services" on page 88](#).

NOTE:

A JACE-8000 controller has no backup battery provision—onboard SRAM preserves runtime data upon any power event. Thus, its station's PlatformServices has no **PowerMonitorService**. For continuous operation across power events, an external battery-backed UPS must be used to power the controller.

Battery-less JACE

The JACE-8000 controller and previous JACE-3E and JACE-6E controllers include integral *onboard* SRAM, which preserves runtime data upon a power loss. By default, these controller platforms are “battery-less”. An “SRAM option card” is also available for any JACE-6 and JACE-7 series controller. These controllers can operate *without* any backup battery, onboard NiMH or otherwise.

SRAM support works via a station platform service, the “DataRecoveryService.” This platform service continuously records all database changes in SRAM, and upon reboot from a power event, restores (plays back) these changes.

Except for a JACE-8000 controller, any SRAM-equipped JACE can *also* have a backup battery, and be configured to use either SRAM or backup battery, *or both*. By default, its station's PlatformServices contains *both* the PowerMonitorService *and* the DataRecoveryService.

For details, refer to the document *JACE Data Recovery Service (SRAM support)*.

Platform view differences, JACE controller vs. Windows-based host

For any JACE controller platform, the following platform views differ from Windows-based platforms:

- [Platform Administration, page 17](#)

Platform Administration

Platform Administration for a JACE platform differs as shown below:

Figure 3 Platform Administration for a controller

The screenshot shows the 'Platform Administration' window. The sidebar on the left contains the following buttons: View Details, User Accounts (highlighted with a blue border), System Password, Change HTTP Port, Change SSL Settings, Change Date/Time, SFTP/SSH (highlighted with a blue border), Change Output Settings, View Daemon Output, Configure Runtime Profiles (highlighted with a blue border), Backup, Commissioning, and Reboot. The main area displays the following configuration details:

Baja Version	Tridium 4.0.11.0
Daemon Version	4.0.11.0
System Home	/opt/niagara
User Home	/home/niagara
Host	192.168.1.59
Daemon HTTP Port	3011
Daemon HTTPS Port	5011
Host ID	Qnx-TITAN-1948-5518-1BA7-A3FF
Model	TITAN
Local Date	06-Mar-15
Local Time	10:05 Eastern Standard Time
Local Time Zone	America/New_York (-5/-4)
Operating System	qnx-jace-n4-titan-am335x (4.0.25.0)
Niagara Runtime	nre-core-qnx-armle-v7 (4.0.11.0)
Architecture	armle-v7
Enabled Runtime Profiles	rt,ux,wb
Java Virtual Machine	oracle-jre-compact3-qnx-arm (Oracle Corporation 1.8.0.0.8)
Niagara Stations Enabled	enabled
Number of CPUs	1
Current CPU Usage	5%
Overall CPU Usage	2%
Filesystem	Total Free Files Max Files

- A **User Accounts** button is available. For details, see “[User Accounts](#)” on page 53, page 66.

- An **Advanced options** button is available. For details, see [“SFTP/SSH” on page 59, page 76](#).
 - A **Configure Runtime Profiles** button is available. For details, see [“Configure Runtime Profiles” on page 61, page 78](#).
 - A **Commissioning** button and a **Reboot** button are available. For details, see [“Commissioning” on page 64, page 83](#) and [“Reboot” on page 65, page 83](#).
 - Various data in the view (repeated in “View Details”) differ greatly from that for Windows hosts.
- See [“Platform Administration” on page 51, page 64](#) for more details.

Windows-based

Windows-based platforms in Niagara 4 include Windows-based “Supervisor” PC hosts and/or engineering workstations.

NOTE:

All JACE controllers that can run Niagara 4 use the QNX operating system. Prior Windows-based JACE controllers (JACE-NXT, JACE-NXS) that run Windows XP Embedded are not supported in Niagara 4.

File storage on Windows-based Niagara 4 platforms is typically a hard drive, and the operating system is a minimum of Windows 7 Professional, with Windows 8 Professional often used. Alternatively, a Niagara 4 Supervisor may run Windows Server 2012.

Most platforms use a *64-bit* Windows OS (Win64-based). Although a Win64 Supervisor uses a 64-bit JVM (Java Virtual Machine) and different NRE core binaries, its Niagara platform interface is nearly *identical* to any Win32-based Supervisor. Therefore, you can equate a *Win64-based* Supervisor as a “Windows-based” host in various discussions in this document, unless particularly noted. For further details, see [“Win64-based Supervisor notes” on page 18, page 19](#).

For any Windows-based platform, the following platform views differ from JACE controller platforms:

- [Platform Administration, page 18](#)

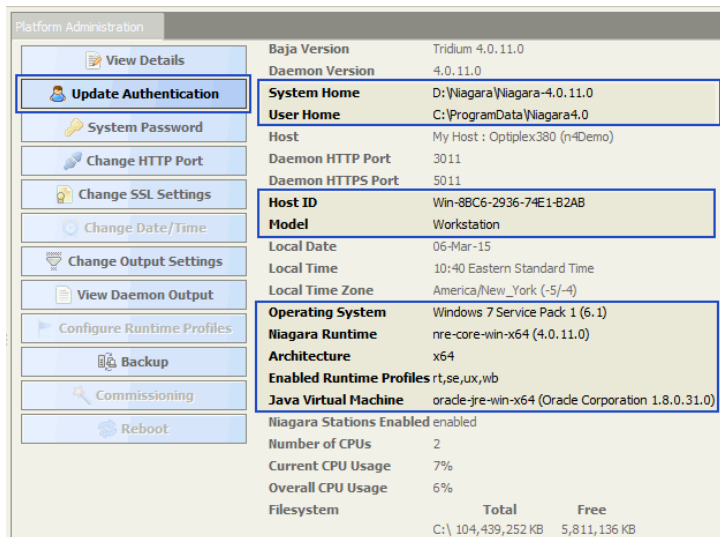
NOTE:

When connected to any Windows host, the TCP/IP Configuration platform view is always read-only. Intended configuration use is for JACE controllers only. On any Windows host, you configure TCP/IP and other network settings using the normal Windows Control Panel interface.

Platform Administration

Platform Administration for a Windows-based platform ([Figure 4 Figure 5, page 19](#)) differs as follows:

Figure 4 Platform Administration for Windows-based platform



- No **User Accounts** button is available, as platform authentication is handled differently, with credentials managed only in Windows. For details see ["Update Authentication" on page 54, page 68](#).
- No **SFTP/SSH** button is available (equivalent configuration can be done using Windows, if needed). More typically, the "Remote Desktop Connection" feature of Windows is used.
- The **Change Date/Time** button is dimmed (unavailable). This should be done using Windows.
- The **Configure Runtime Profiles** button is dimmed (unavailable). Windows hosts are invariably enabled for *all* runtime profiles.
- The **Commissioning** button is dimmed. The **Commissioning Wizard** is intended only for initial Niagara installation and startup in a remote JACE, or whenever *upgrading* a JACE.
- The **Reboot** button is dimmed. This is intended only for remote JACE platforms. Any Windows host must be rebooted using Windows.
- Various data in summary information (repeated in View Details) differs greatly from QNX hosts.

See ["Platform Administration" on page 51, page 64](#) for more details.

Win64-based Supervisor notes

Supervisor support for installations on PCs running 64-bit Windows operating systems is typical, for example Windows Server 2012 or Windows 7 or 8 Professional 64-bit. The primary application for 64-bit install is for a Supervisor station with a large NiagaraNetwork (job has large numbers of JACEs, each with many Niagara proxy points), and thus, a large station database.

In particular, the 64-bit Java VM (Virtual Machine) does not have a 2GB memory limit, unlike the Java VM on a Win32-based Supervisor. Typically, any PC with 64-bit Windows also has 4GB or more of RAM installed, and (unlike with a 32-bit Windows PC) the 64-bit OS can effectively utilize all of it. Therefore, a 64-bit Windows host may be the solution for the largest "enterprise level" Supervisor. See the following sections ["Known Limitations", page 19](#) and ["Installation and interface differences", page 20](#) for more details.

Known Limitations

Please note that at the time of this document update, there are several known limitations for a Supervisor running on a 64-bit Windows operating system. Although most of these do not apply to a "typical Supervisor", they should be understood before installation time. These 64-bit Windows platform limitations include the following:

- NRE serial support is available for a 64-bit Windows platform. However, serial-based drivers (e.g. modbusAsync, flexSerial, various legacy drivers) are not typically licensed on a Supervisor, and therefore are not fully tested or supported on a 64-bit platform.

Exceptions to such license rules can occur with 64-bit “engineering workstations” or “demo machines”. Again, in these cases note that 64-bit serial operation is not fully guaranteed.

Additionally, a known issue with the 64-bit serial library may present itself in initialization phases, as has been noticed with usage of a 64-bit Niagara Serial Tunnel client. For related details see the section “Client side (PC application)” in the latest *Drivers Guide*.

- Lonworks FTT-10 is not fully supported on a 64-bit Windows platform—although there are Echelon 64-bit drivers, most are 32-bit drivers in a “64-bit wrapper”, and are likely unsuitable. Further, a Supervisor is not typically licensed for Lonworks. However “LonIP” is supported.

Installation and interface differences

Installation of the Win64-based Supervisor is like the Win32-based installation, except that separate executables in the root of the Supervisor product image or CD are used to install (setup_x64.exe instead of setup_x86.exe, respectively).

A platform connection to a Win64-based Supervisor provides the identical collection of views as with a Win32-based host. Also, when opening a station running on a Win64 host, you see the same child platform services under its **PlatformServices** as with a station running on a Win32 host.

Models of platforms

Among the two groups of JACE controllers (embedded controllers and Windows-based), there are different *models*, each of which has a “host model” text descriptor. You see this descriptor in the **Station Manager** view of a Niagara Network (**Host Model** column), and also in platform views such as **Platform Administration**, as well as the **PlatformServices** container of a station running on that host.

[Table 1](#) lists various platform models (including JACE controllers), known at the time of this document, starting with the host model text descriptor.

NOTE:

A few JACE models (and the SoftJACE) listed are *not compatible with running Niagara 4*, only NiagaraAX, and are so noted. However, it is possible some may exist on a job site where the Supervisor was “migrated” from AX to N4 (N4.0), along with some number of JACE controllers that *are* compatible. For related background details, refer to the *AX to N4 Migration Guide*. In addition, several JACE models discontinued more than 10 years are not listed.

Host models of JACE platforms

Model desc.	Actual Model	Notes
JNXS	JACE-NXS series	Discontinued Win32-based (Windows XP Embedded), model before JACE-NXT series. NOTE: Not compatible with Niagara 4; must run NiagaraAX (AX-3.8 or earlier).
JNXT	JACE-NXT series	Discontinued Win32-based JACE controller (Windows XP Embedded). NOTE: Not compatible with Niagara 4; must run NiagaraAX (AX-3.8 or earlier).
Jsoft	SoftJACE installed on user-supplied PC	Windows-based. This is different than a Supervisor for example, which appears instead as “Workstation”. NOTE: Not available as a Niagara 4 product; it must run NiagaraAX (AX-3.8 or earlier).
JVLN	JACE-7 series (JACE-700)	Discontinued QNX- based, with more processing power than JACE-2/6 series. See the <i>JACE Niagara 4 Install and Startup Guide</i> for commissioning details. NOTE: The WiFi option for this controller is not supported in Niagara 4.
NPM2	JACE-2 series, including Security JACE (SEC-J-201), JACE-202 Express (M2M JACE 2)	Discontinued QNX- based. Uses the IBM J9 JVM (Java Virtual Machine). Models include the JACE-202 Express, or “M2M JACE 2” with onboard I/O points. NOTE: Not compatible with Niagara 4; must run NiagaraAX (AX-3.8 or earlier).
NPM3	JACE-3E series, introduced in mid 2013.	QNX-based controller, a JACE-3E is between the JACE-2 series and the JACE-6E series in performance, and includes onboard SRAM for battery-less operation (if desired). See the <i>JACE Niagara 4 Install and Startup Guide</i> for commissioning details.
NPM6	JACE-6 series, including Security JACE (SEC-J-601), JACE-602 Express (M2M JACE 6)	Discontinued QNX- based, with more processing power than the JACE-2 series. Includes the JACE-602 Express, or “M2M JACE 6” with onboard I/O points. See the <i>JACE Niagara 4 Install and Startup Guide</i> for commissioning details. NOTE: Security JACE controllers are not compatible with Niagara 4.
NPM6E	JACE-6E, as well as “retrofit board” controllers JACE-603 and JACE-645.	QNX-based controllers. The JACE-6E is like a JACE-6, but includes onboard SRAM for battery-less operation (if desired). The JACE-603 and JACE-645 are “retrofitted” Niagara R2 JACE-403 and JACE-545 controllers. Refer to the <i>JACE Niagara 4 Install and Startup Guide</i> for commissioning details.

Model desc.	Actual Model	Notes
TITAN	JACE-8000 series	Newest QNX-based controller, with the most processing power and resource capacity of any JACE controller. Includes onboard SRAM for battery-less operation, integral WiFi, and a USB port for backup/restore usage with a USB flash drive. Plug-in option modules provide additional communications ports. Currently supports Niagara 4 only. Refer to the <i>JACE-8000 Niagara 4 Install and Startup Guide</i> for commissioning details, also documents <i>JACE-8000 USB Backup and Restore</i> and <i>Niagara 4 JACE WiFi Operation</i> .
Workstation	User-supplied PC, e.g. a Supervisor or engineering workstation.	Windows-based customer supplied PC that runs Workbench, minimally.

File locations on Niagara 4 platforms

During the Workbench installation and platform commissioning processes, Niagara 4 differentiates between two types of files based upon the content of the files: *configuration* and *runtime* data. Files and folders that contain configuration data reside in separate locations from files and folders that contain runtime data. This separation enhances security by denying general access to the runtime files and allowing each user access to only their personal configuration files.

As a result of separating configuration and runtime data, the system supports multiple home directories on the Supervisor or engineering workstation. These homes may be identified as:

- The system home contains runtime files, such as core software modules, the JRE, and binary executables.
- A Workbench user home for each user contains configuration data, including option files, and Niagara registries.
- A platform daemon user home for the Supervisor or engineering workstation platform contains platform configuration data.
- Two station homes called, *protected station home* and *station home*, are part of each user home.

The following table provides a summary of the Supervisor or engineering workstation homes with shortcut information.

Table 1 Homes on a Supervisor or engineering workstation

Home in the Workbench Nav tree	Home in the Platform Administration view	Niagara 4 alias	Windows folder location and contents	File ORD shortcut
My Host→My File System→Sys Home	System Home	niagara_home	C:\niagara\Niagara-4.0.xx Executables and software files	! (as in NiagaraAX)
My Host→My File System→User Home	N/A	niagara_user_home	C:\Users\userName\Niagara4.0\brand Workbench user home for each human user contains that user's unique configuration files.	~ (unique to Niagara 4)
shared folder	N/A	station_home	C:\Users\userName\Niagara4.0\brand \shared	^ (as in NiagaraAX)
stations folder	N/A	protected_station_home	C:\Users\userName\Niagara4.0\brand \stations	^^ (unique to Niagara 4)
N/A	User Home	niagara_user_home	C:\ProgramData\Niagara4.0\brand Platform daemon user home (non-human user) holds platform daemon configuration files. Requires a local platform connection to view in the Platform Administration view.	~ (unique to Niagara 4)

On a controller there are two homes:

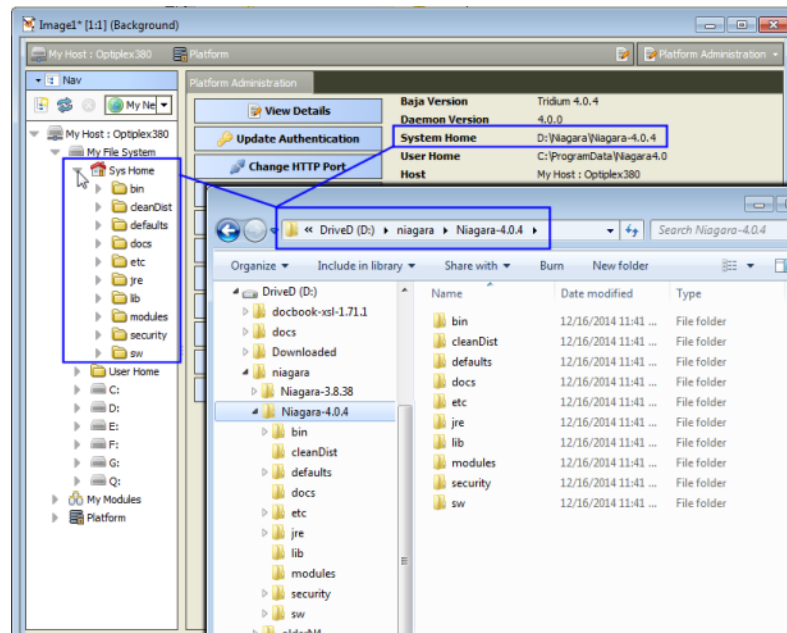
Table 2 Homes on a controller

Home in the Platform Administration view	Home in the Platform Administration view	Niagara 4 alias	OFD location and contents	File ORD shortcut
Platform→Remote File System→Sys Home (Read Only)	System Home	niagara_home	/opt/niagara Contains operating system data.	! (as in NiagaraAX)
Platform→Remote File System→User Home (Read Only)	User Home	Niagara_user_home	/home/niagara Contains configuration data and the installed and running station.	~ (unique to Niagara 4)

System home

Sometimes identified by its alias, `niagara_home`, the system home is the sole location to which the Workbench installation wizard and platform commissioning wizard install Niagara runtime components, such as core software modules, the JRE, and binary executables. License files and license certificates reside in the Workbench system home, under the `security` subfolder. A system home contains no configuration files that can be changed by a user. Except when it is time to upgrade, these runtime files are read-only.

Figure 5 Example Sys Home (`niagara_home`) on Supervisor platform



The example above shows the file system for a Niagara 4 Supervisor running on a Windows PC. In the Nav tree, the system home folder is referred to as its **Sys Home**. In the **Platform Administration** view, the same system home is referred to as **System Home**.

In the example, the actual location of the system home folder on this PC is: `D:\niagara\Niagara-4.0.15`.

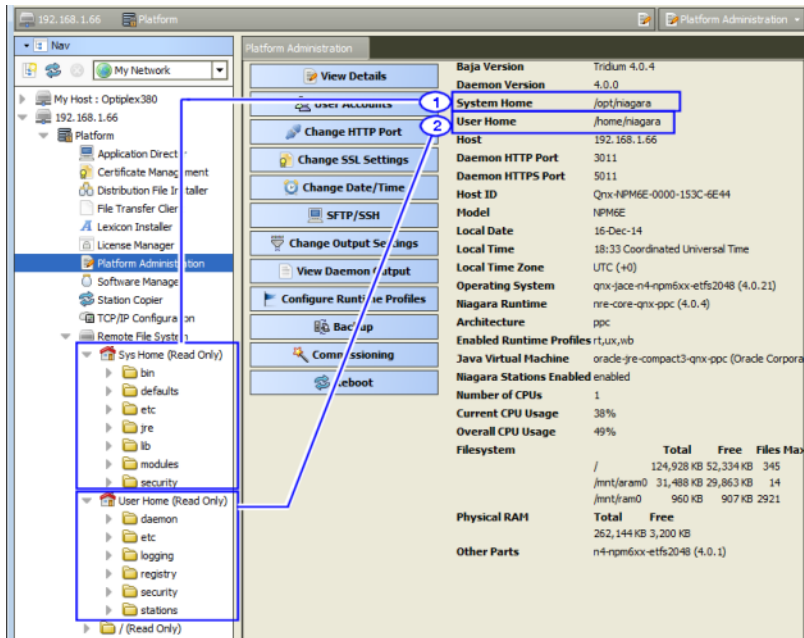
The system home on a controller appears as **System Home** in the **Platform Administration** view. The actual location of the system home folder for a controller is: `/opt/niagara`.

Controller user home

The user homes are the locations under which all configurable data reside. Included are stations, templates, registry, logs, and other data. Referred to by the alias `niagara_user_home`, the separation of these files from the runtime files stored in the system home folder is new in Niagara 4.

A JACE controller has one system home and one user home.

Figure 6 JACE Sys Home (niagara_home) and User Home (niagara_user_home) locations



Callout 1 above identifies a controller's system home (alias: niagara_home) in both the Nav tree and the Platform Administration view. In the Nav tree, you can find the controller's system home by expanding Platform→Remote File System. The actual folder for the system home is /home/niagara.

Callout 2 identifies the controller's user home or daemon user home (alias: niagara_user_home) that contains the installed and running station and other configuration files. The actual folder for the daemon user home is: /home/niagara.

Windows platform user homes

For security reasons, each person that is a user of a Windows platform, has their own user home. This means that each Supervisor platform has at least two user home locations: a Workbench **User Home** (for people), and a platform daemon **User Home** (for the daemon server processes).

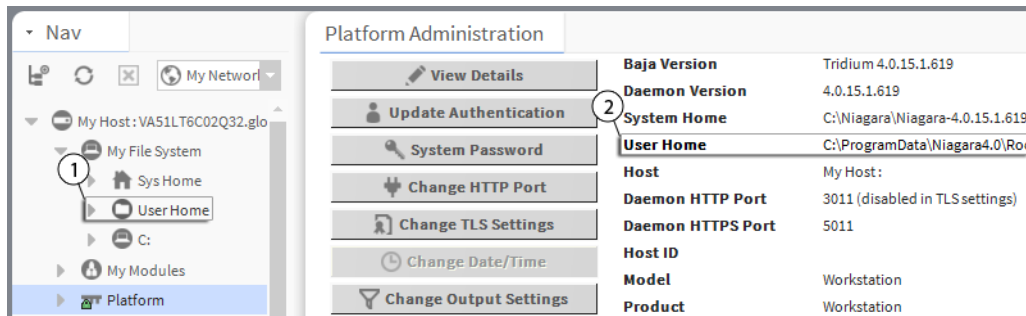
The Supervisor or engineering workstation that is licensed to run a station has a daemon user home. The daemon is a server process and represents a (non-human) user that manages the Supervisor's running station. The Supervisor's daemon user home contains daemon-specific configuration information. The actual location of the Supervisor's daemon user home is C:\ProgramData\Niagara4.0\brand. The platform daemon is installed to this location and started from this location as a Windows service.

In addition to this daemon user home, a Windows host has a separate Workbench user home for each person (operator, administrator) who logs on with credentials to a Windows-based platform licensed for Workbench, meaning a Supervisor or engineering workstation. Any given PC or workstation has at least one, and may contain multiple Workbench user homes.

Each person's Workbench user home is available in the Nav tree as a node under **My Host→My File System** and contains unique configuration information that is not shared. This is where to find any new Workbench station, as well as any remote station backups, templates and other configuration files. The actual location of each person's user home is in the Niagara4.0 folder under your Windows User account.

If you open a local platform connection to your Supervisor PC, expand **My File System** in the Nav tree, and go to the **Platform Administration** view, you can see both types of user homes at the same time.

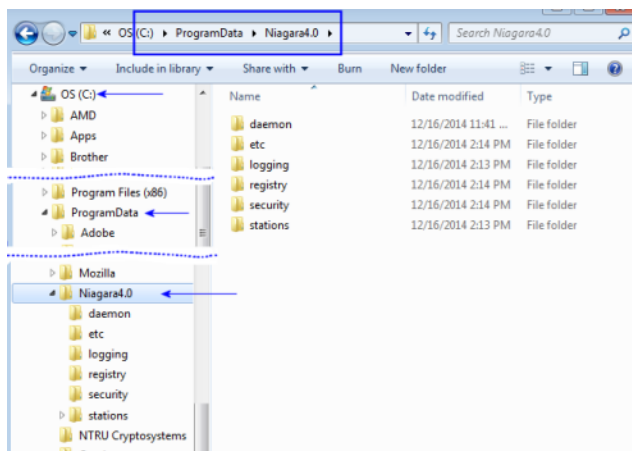
Figure 7 Local platform connection to a Supervisor station with Workbench and daemon user homes



- Callout 1 identifies a Workbench **User Home**.
- Callout 2 identifies the daemon **User Home**.

When you first install Niagara 4 on your PC and start the daemon (by choosing the install option *Install and Start Platform Daemon* on installation), the installation program creates this daemon **User Home** (Niagara4.0 folder). Initially, it contains an empty *stations* sub-folder, until you copy a station to it.

Figure 8 Example of a daemon User Home location in Windows Explorer



You can do this station copy in different ways. In Niagara 4, you can let the **New Station** wizard initiate this copy from its last Finish step. Or as needed, you can manually open a local platform connection and use the **Station Copier**.

The actual location of each user's home folder is under that user's personal Windows account. Some example Workbench user home locations are:

```
C:\Users\John\Niagara4.0\brand
```

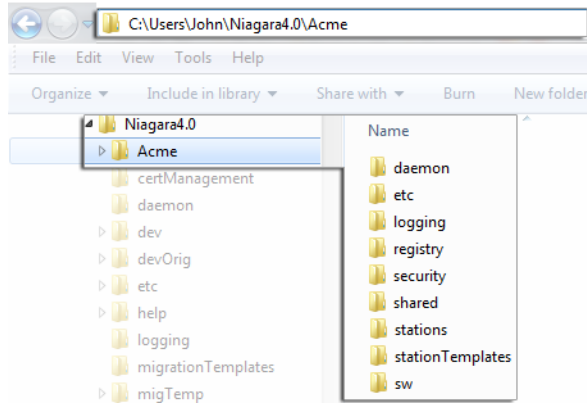
```
C:\Users\Mike\Niagara4.0\brand
```

where "John" and "Mike" are separate Windows user accounts. The first time a Windows user starts Workbench, the system automatically creates that user's unique user home folder.

- The person that installs Niagara 4 on a PC acquires the first such user home. If no other Windows users log on to that PC, this may be the only Workbench user home on the platform.
- However, if another person logs on to Windows on that computer and starts Workbench, that user also acquires their own Workbench user home.

The following figure shows an example Workbench user home location in Windows Explorer.

Figure 9 Example of an automatically-created Workbench User Home in Windows Explorer



Station homes

Niagara 4 uses the Java **Security Manager** to protect against malicious from accessing station or platform data and APIs. The **Security Manager** uses signed policy files that specify the permissions to be granted to code from various sources. Included are tighter controls about which applications have access to parts of the file system. Two folders under the **Workbench User Home** serve to protect sensitive data while allowing authorized access to data that can be shared.

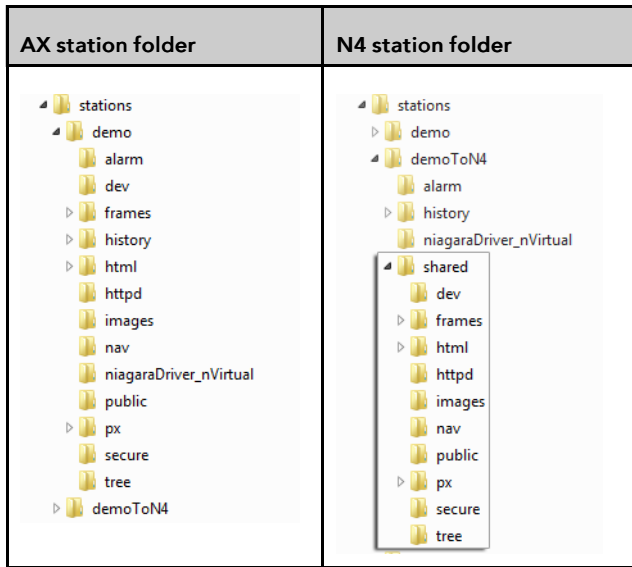
- The **stations** sub-folder, otherwise known as the *protected station home* (alias: `protected_station_home`) contains the running station's file system, and may be accessed only by core Niagara software modules. Station items that are always in protected station home, that is, items that are not under the `shared` sub-folder include the following folders, as applicable:
 - alarm
 - history
 - niagaraDriver_nVirtual
 - provisioningNiagara
 - dataRecovery

All files in the **stations** folder root (`config.bog`, `config.backup.timestamp.bog`, etc.) are always in protected station home. For this reason, in Niagara 4 it is no longer necessary to blacklist or whitelist station files or folders.

- The **shared** sub-folder, otherwise known as the *station home* (alias: `station_home`), allows all modules to have read, write, and delete permissions.

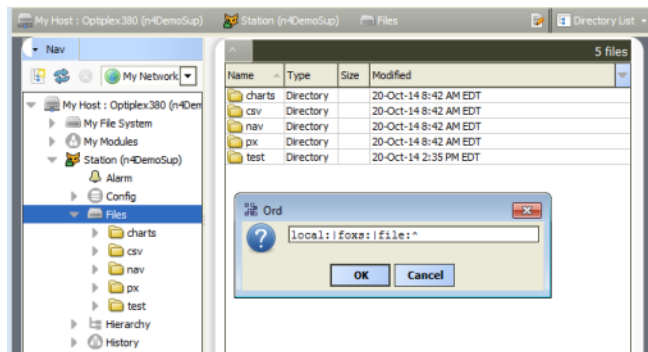
The alias `station_home` retains the same file ORD shortcut (^) as used in NiagaraAX—only in Niagara 4 this points to the station's **shared** sub-folder.

Figure 10 Example NiagaraAX station file folders compared to Niagara 4 station file folders



As shown in the figure above, comparing an AX station file folder structure (left side) to the same station migrated to Niagara 4, a number of folders are now under this **shared** sub-folder. Included are folders and files used in graphical (Px) views or navigation, such as images, px, nav and so on. Modules that are prevented from writing to the station root by the **Security Manager** must also write to the **shared** sub-folder.

Figure 11 File ORD for the station home in Niagara 4 now points to the shared folder



As shown in a station running above, the station home (alias: station_home) file ORD (^) now points to the contents of the **shared** sub-folder. Other items in protected station home are no longer accessible or visible.

Copying a new station to the daemon user home

In Niagara 4, the **New Station Wizard** tool finishes with an option to copy the station from the station home (the location for each new station) under yourWorkbench **User Home** to the **User Home** of the local platform daemon.

Prerequisites: The new station exists in the station home (under User Home).

- Step 1 When the **New Station Wizard** prompts you with the option to *Copy station*, select the option and click **Finish**.
- Step 2 Make a local platform connection and log on.
The **Station Copier** transfers the station and prompts you with the options to start the station after copying and enabling auto-start.
- Step 3 Select the option to start the station.

The **Application Director** opens with the new station present in the daemon **User Home**.

The new station now exists in two locations on your local host: the original location in your Workbench **User Home**, and also in the platform daemon **User Home**.

Once the station is running in the daemon **User Home**, you can make a backup of the running station, where the backup `.dist` file goes in the `backups` folder of your Workbench **User Home**. Or, you can use the platform **Station Copier** to copy the station back to the `stations` folder of your Workbench **User Home**.

NOTE:

Using the **Station Copier** to copy the station back to your Workbench **User Home** is highly recommended if you made any changes to the station. This is essential if you are installing it (copying it) to any remote platform. Remember, the copy of the station in your Workbench **User Home** is immediately obsolete as soon as you make changes to the copy of the station running in the daemon **User Home**.

Running a station from a Workbench User Home

Instead of running a station out of the daemon **User Home**, you can run a station directly from your Workbench **User Home** (outside of normal platform daemon control).

You do this using the Niagara console command:

```
station stationName
```

This is *not a recommended* way to run a production station, but instead more of a developer utility that allows quick access to station debug messages in the console window. If you run the station this way, be mindful of possible port conflicts with any *other station* that the daemon user may be running locally (in daemon **User Home**), meaning Fox ports, Web ports, and so on.

Shared file strategy

A sharing strategy makes it possible for multiple users of a single Supervisor or engineering workstation to share station files including backups.

If multiple people log on (differently) to Windows on a Niagara 4 host and use Workbench, each person has their own separate Workbench **User Home**.

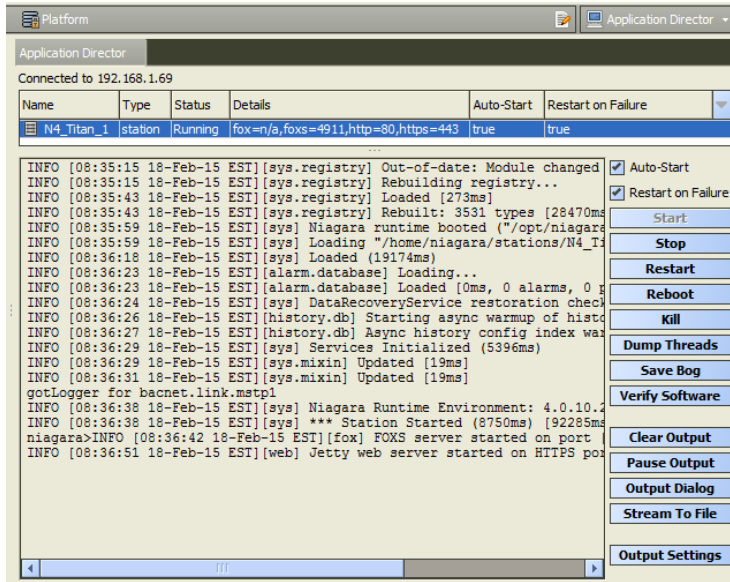
Windows users require permissions to access other users' files; yet it's possible that different users of a system (Supervisor or engineering workstation) may need to share items such as station backups, station copies, saved template files, and so on. Such items may be in multiple Workbench **User Home** locations in Niagara 4 (unlike in NiagaraAX where a single `!\backups` or `!\stations` folder applies to all users).

Therefore, in some scenarios you may need to establish a sharing strategy, perhaps re-copying such items to a commonly-accessible folder location on the Niagara 4 Windows PC. For example, you might create a shared folder under the Niagara 4 **Sys Home** location (a Workbench **User Home** is not shareable).

Application Director

The Application Director is one of several platform views. You use it to start and stop a station in any host (whether a remote JACE or a local or remote Supervisor PC), as well as see station output for troubleshooting purposes. From the Application Director, you also define a station's restart settings, plus have access to other station actions.

Figure 12 Application Director view, looking at Niagara station



As shown above, the **Application Director** is split into three main areas:

- [Installed applications \(stations\), page 29](#) — at top
- [Application output, page 31](#) — main area

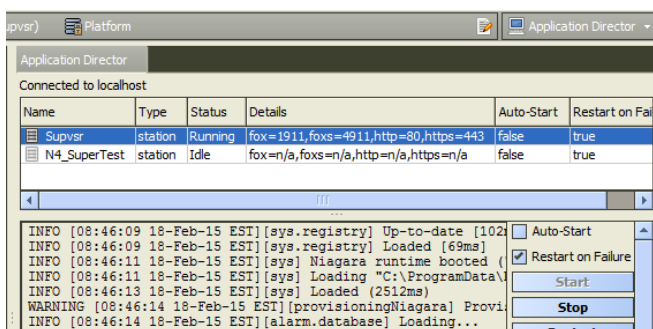
Related are log levels defined for the station. See [“Station log levels \(DebugService\)” on page 31, page 33](#).

- [Application and output controls, page 35](#) — right-side checkboxes and buttons

NOTE:

In the Application Director for any JACE, the installed applications area should show (at most) only one station, as shown above. However, the Application Director for a Windows platform (Supervisor, or engineering workstation) may show more than one station, as shown below.

Figure 13 Application Director for Supervisor host showing multiple stations



Even if a Windows platform is licensed for more than one station, running multiple stations at the same time requires you to use non-default ports for all but one of them to avoid port binding issues. For example, use a Fox and Foxs port other than 1911 or 3011 respectively, or Http and Https port other than 80 or 443 respectively.

Installed applications (stations)

The top area of the **Application Director** shows a table of installed applications (stations), as shown below.

Figure 14 Application Director installed applications

Name	Type	Status	Details	Auto-Start	Restart on Failure
N4_Titan_1	station	Running	fox=n/a,foxs=4911,http=80,https=443	true	true

Every 1.5 seconds, the platform daemon fetches data about the station(s) and updates this in the following columns:

- **Name**
The name of the station directory.
- **Type**
This is always “station” for a Niagara station.
- **Status**
One of the following, as applied to a station:
 - Idle — Station is not running, but can be started without a reboot.
 - Running — Station is running.
 - Starting — Platform daemon has started the station, but the station has not reported back its status back to the daemon.
 - Stopping — Daemon has ordered the station to stop, but its process has not yet terminated.
 - Halted — Station is not currently running, and cannot be restarted without a reboot.
 - Failed — Station terminated with a failure exit code.
- **Details**
For any station, shows four items:
 - `fox`= TCP/IP port monitored for regular (unencrypted) Fox connections to Workbench and other Niagara stations. Shows “n/a” if station is not running, or if Fox Enabled is set to false.
 - `foxs`= TCP/IP port monitored for secure Fox connections to Workbench and other Niagara stations, if so configured. Shows “n/a” if the host does not support (or is enabled) for a secure connection, or if the station is not running, or if Foxs Enabled is set to false.
 - `http`= HTTP port that the station’s WebService monitors for regular (unencrypted) browser connections to the station. Shows “n/a” if station is not running, or if it does not have a running WebService, or if Http Enabled is set to false.
 - `https`= HTTP port that the station’s WebService monitors for secure browser connections to the station, if so configured. Shows “n/a” if host does not support (or is enabled) for a secure connection, or if the station is not running, or if Https Enabled is set to false, or if the station does not have a running WebService.
- **Auto-Start**
Either true or false. If true, the station starts whenever the platform daemon starts. Configured with a right-side checkbox (see [“Start checkboxes” on page 33, page 36](#)).
- **Restart on Failure**
Either true or false. If true, the daemon automatically restarts the station after it terminates with a failure exit code. Configure with a right-side checkbox (see [“Start checkboxes” on page 33, page 36](#)).

Apart from the data shown in the table, application selections are possible. See [“Application selections”, page 31](#).

Application selections

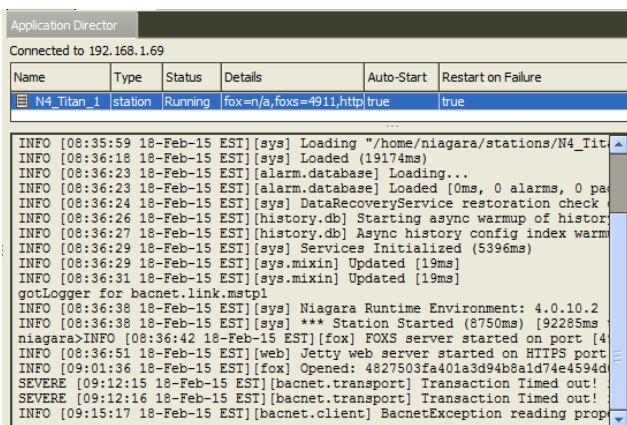
Click in the installed applications table for different results, as follows:

- click
Click a station to select it, highlighting it. When a station is selected, its standard output appears, and all enabled right-side buttons apply to it. For details, see [“Application output” on page 30, page 31](#) and [“Application and output controls” on page 33, page 35](#).
- right-click
Right-click a station for its shortcut menu (a *subset* of the application and output actions buttons). For details on included menu commands, see [“Application and output controls” on page 33, page 35](#).
- double-click
If running, double-click a station to open a Workbench (Fox) connection to it, showing its **Station Summary** view. Or, press Ctrl and double-click for a *new tab* showing the station connection.
If not running, a station double-click does not change the view.

Application output

The largest area in the **Application Director** view shows the “standard output / standard error” output text for the selected station, as shown below.

Figure 15 Station output in Application Director’s application output area



Depending on the status of the station selected, the standard output text is one of the following:

- If a running station, output updates in real time. As more text is written by the station, it is appended to the bottom of the output area.
- If the station is not running, output text is from the most recent execution of that station.
- If no station is selected, output text area is blank.

NOTE:

You can use the Windows copy shortcut (Ctrl + C) to copy output text to the clipboard. As needed, use scroll bars to view all text, and use the right-side output control buttons. One of these lets you stream station output to a file. For more details, see [“Output control buttons” on page 35, page 37](#).

The following sections provide more details related to a station’s standard output:

- [Standard output overview, page 32](#)
- [Station LogHistory \(LogHistoryService\), page 32](#)

Standard output overview

Station output log messages can include errors and warnings that let you why something is not working, as well as simple informational messages about events as they occur. If needed, you can also change the log “level” of station output—see [“Station log levels \(DebugService\)” on page 31, page 33](#).

The general format of a station output log message is:

```
TYPE [timestamp] [station_process] message_text
```

For example:

```
INFO [17:05:18 16-Feb-15 EST] [fox] FOXS server started on port [4911]
```

Message log types seen in station output include the following, by leading text descriptor

- **INFO**
Typical of most default station output log messages. Usually, each message lets you know some process milestone was started or reached, such as a service or the station itself.
Note **INFO** is equivalent to the **MESSAGE** level in AX station log output.
- **WARNING**
Informs you of a potential problem, such as inability to open a specific port. Typically, warnings do not keep a station from starting.
Note **WARNING** is equivalent to the (same) **WARNING** level in AX station log output.
- **SEVERE**
Informs you of a problem that might keep the station from starting. Or, if it can start, an error that prevents some function of the station from operating correctly. Often an exception is produced.
Note **SEVERE** is equivalent to the **ERROR** level in AX station log output.
- **FINE**
A verbose debug-level message that may be generated upon every process transaction. Typically this is useful only in advanced debugging mode. You see these for station processes only if you have set the log level at “**FINE**” or even finer (**FINER**, **FINEST**, **ALL**).
Note such levels (**FINE**, **FINER**, **FINEST**) are equivalent to the **TRACE** level in AX station log output.

In addition to the “typed” output messages described above, occasionally you may see a string of “java exception” text in the a station’s output. This indicates an unforeseen station execution issue, which can range from a licensing problem, a misconfiguration, or some other unexpected problem. If an unexplained exception reoccurs, copy the exception text and report the problem to Systems Engineering.

Note that station output logs in Niagara 4 use a standard Java logging API (`java.util.logging`), which has more log severity levels than the NiagaraAX Baja logging API (`javax.baja.log`). Also note any Niagara 4 station has a standard **DebugService** (`LoggingService`) for making changes. This in addition to the “spy” log setup used in NiagaraAX. See [“Station log levels \(DebugService\)” on page 31, page 33](#) and [“Station log levels \(spy:/logSetup\)” on page 32, page 34](#).

Station LogHistory (LogHistoryService)

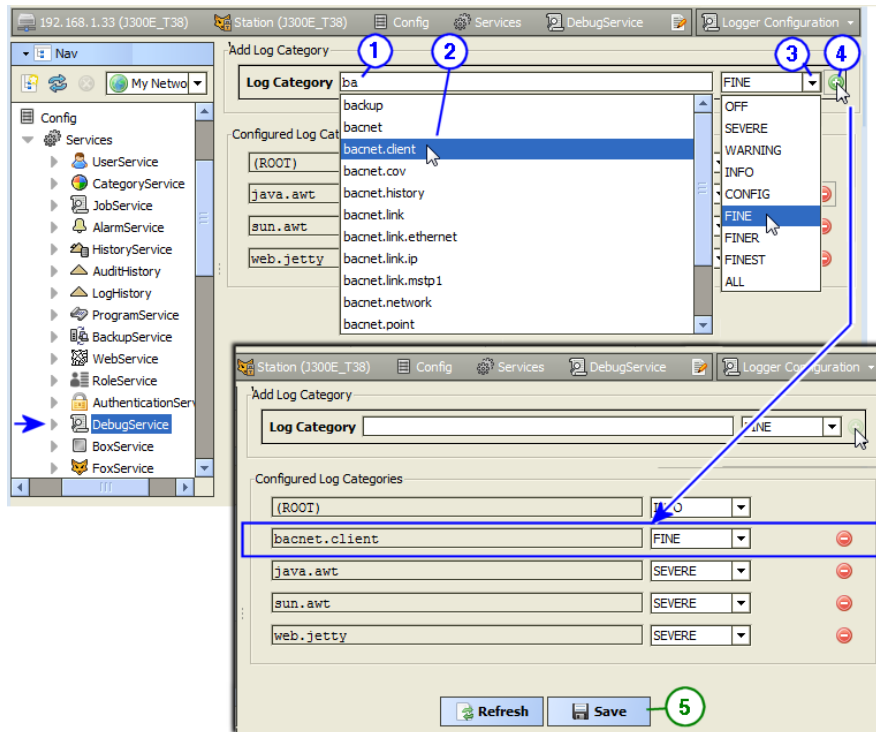
If a station is configured with the `LogHistoryService` (under its `Services` container), it maintains a buffered history (“`LogHistory`”) of *some* of the messages seen in the station’s standard output. In the `LogHistoryService`’s configuration, you specify its log level, meaning the minimum message type (from station output) to log. By default, the log level (property “`Minimum Severity`”) is `Info`. You may wish to change this to `Warning`.

This is mentioned because when looking at a station’s output, you are usually troubleshooting. As part of troubleshooting, you should always check the station’s histories for the `LogHistory`. It should contain recently recorded station errors and (if configured) warnings. This information may help when evaluating “live” output from the station.

Station log levels (DebugService)

Using the station's **DebugService** (LoggingService), you can review and change the "log level" of the station processes of interest, in order to "tune" station output seen in the **Application Director**.

Figure 16 Niagara 4 stations have a DebugService in which you can set log levels for modules/processes



In the example shown above, the "bacnet.client" process is being added with a log level of "FINE". Such an entry might be useful to debug (troubleshoot) errors about Niagara writes to Bacnet proxy points.

Adding log items in a station's DebugService

Prerequisite: Admin write permissions on the station's **DebugService**.

Prerequisites:

NOTE:

Steps in this task reference callouts in [Figure 16](#) [Figure 20](#), page 33.

- Step 1 In the DebugService's default **Logger Configuration** view, click in the **Log Category** field and start typing the name of a module or module.process you wish to add. A drop-down list appears.
- Step 2 *Double-click* the item you want in the list; this enters it in the **Log Category** field.
- Step 3 On the right, click the control and select a level, for example `FINE` (for more detail than the default `INFO`).
- Step 4 Click the **Add** control to add it to configured log categories. It now appears in the listed log categories. Repeat steps 1 through 4, if needed, to add other items, or else make other changes on levels, etc.
- Step 5 Click the **Save** button to save these settings to the host's `~/logging/logging.properties` file. Settings become immediately active, affecting station output as seen in the **Application Director**.

CAUTION:

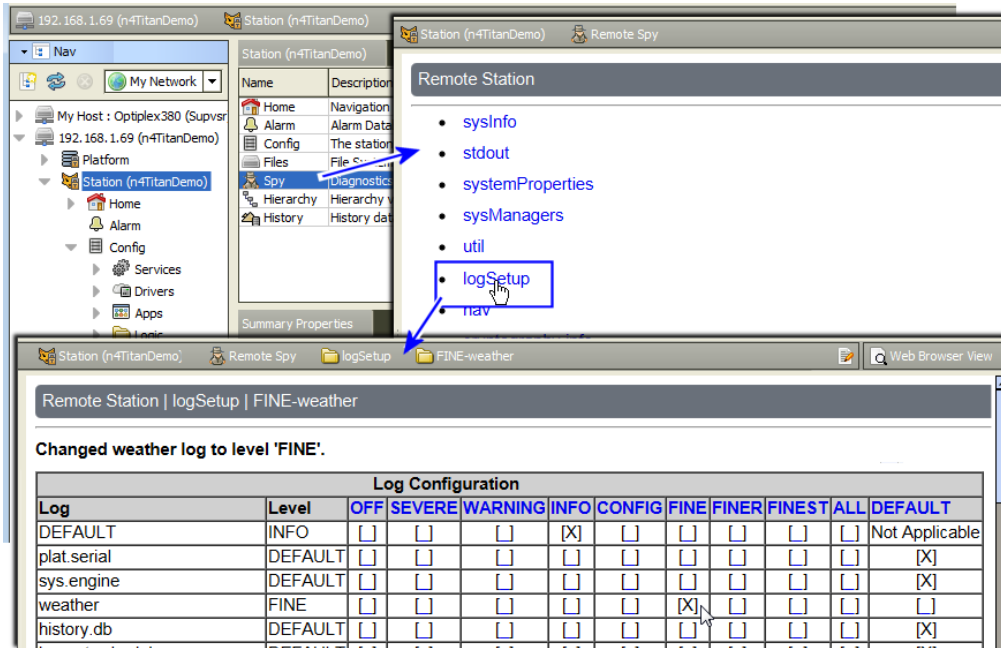
Be aware that persisted log settings are not part of a station's configuration, even though you access them through a station **DebugService**. Settings apply to *any station* run on the host, until changed and saved again. Therefore, be sure to return settings back to normal levels and/or delete additions after concluding a debug session. Otherwise, excessive station output could adversely impact station performance.

Note that Niagara 4 Workbench has a similar log interface for the *Workbench console*, available in the Tools menu (**Tools** → **Logger Configuration**). This affects output seen in the console window when you start Workbench with the shortcut "**Workbench (Console)**", for (wb.exe). Changes to it are stored in your Workbench User Home ~/logging/logging.properties file. For related details, see the documentation for *Niagara 4 Workbench Tools*.

Station log levels (spy:/logSetup)

As an alternative to using the station's **DebugService** to tune station log output, you can use the station "spy" HTML interface for log setup. (This is the only method available for a NiagaraAX station.) To access the station's logSetup page in Workbench, double-click the running station in the Nav tree for its **Station Summary** view. From there, double-click **Spy**, then click **logSetup**.

Figure 17 Station spy logSetup (from Station Summary)

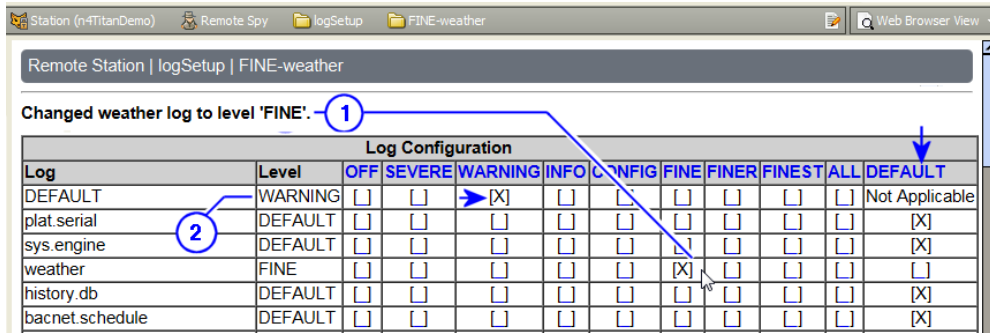


As shown above, spy logSetup is a table listing station processes, each showing its current log level, and starts with a (new) DEFAULT level. If familiar with spy logSetup in NiagaraAX, note these changes:

- Level selection columns are ordered left-to-right in *increasing* order of message volume.
- The number of severity levels has increased: 9 in N4, versus 5 in AX.
- Unlike in AX, log levels are *persisted each time you click to set/clear a checkbox*, saved in the host's ~/logging/logging.properties file. There is no separate "Save To File" in N4.0.
- The level given to the top DEFAULT row is global to any row with the far-right DEFAULT box set. See the next example figure.

The following figure shows the top of the spy logSetup page after the DEFAULT level has been changed from INFO to WARNING, and then the weather process set to the non-default level FINE.

Figure 18 Example spy logSetup for a station after a couple of changes



Callouts in the figure above show:

1. Last change made, reflected in this status line area (Changed weather log to level 'FINE' .)
2. The DEFAULT log level, which in this case has been set to WARNING. Note this log level now applies to all rows where one of the 9 non-default levels (OFF to ALL) has not been set.

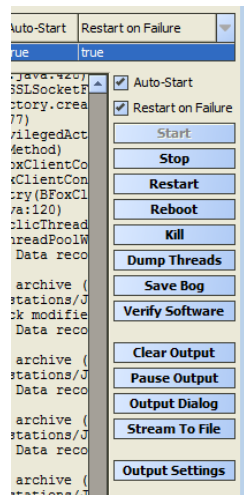
Increasing station output by assigning various log levels above INFO consumes extra station resources and may exact a performance penalty! After troubleshooting, always return log levels to default values.

Note you can also easily review, and if necessary, adjust log levels from the station's **DebugService**. See ["Station log levels \(DebugService\)" on page 31, page 33.](#)

Application and output controls

Unlike in most Workbench views, where changes are entered first and then applied with a "Save" button, in the **Application Director** when you click checkboxes and buttons, changes are applied *immediately* to the selected station. The figure below shows these controls with a running station selected.

Figure 19 Application Director checkboxes and buttons



From top-to-bottom, these controls are grouped as follows:

- [Start checkboxes, page 36](#) (Auto-Start, Restart on Failure)
- [Application control buttons, page 36](#) (Start, Stop, Restart, Reboot, Kill, Dump Threads, Save Bog, Verify Software)
- Output Control buttons, (Clear Output, Pause Output, Output Dialog)
- [Output Settings, page 38](#) button

Start checkboxes

For the currently selected station in the **Application Director**, you can enable (check) or disable (clear) two start settings using checkboxes, as shown above. Typically, for any JACE station you *enable both* checkboxes. In certain troubleshooting scenarios, you may clear **Restart on Failure** in order keep the station from constantly restarting after successive failures.

NOTE:

Changes reflect in the corresponding column of the Application Director's "installed applications" area.

The two start settings for a station are as follows:

- Auto-start

Specifies whether the station starts following platform daemon startup. This means following a host reboot, perhaps as a result of a power cycle, but possibly from a **Reboot** command.

NOTE:

For any JACE controller, a reboot also occurs following any installed dist file(s), as well any TCP/IP-related changes. However, installing new modules from the Software Manager—say, for a new driver, does not always result in a reboot (yet in a few cases, in order for a module to become effective, a reboot may be required—and so is prompted following the module install). At the same time, note that *changing* any existing module (upgrading or downgrading) *always* results in a reboot.

- Restart on Failure

Specifies whether the platform daemon restarts the station if its process exits with a nonzero return code (e.g., engine watchdog had killed the station because of a "deadlock" condition).

NOTE:

In Niagara 4, JACE controllers can have a station restart *without* a reboot. Therefore, if this setting is enabled on such a JACE, if the station fails (terminates with error), the station is restarted.

If a JACE continues to have 3 "automatic restarts" like this within 10 minutes (or however many specified in the station's PlatformService "Failure Reboot Limit" property), the station remains in a "Failed" state, regardless of the setting above. For related details, see "[PlatformServiceContainer configuration parameters](#)" on page 90.

Application control buttons

For the selected station in the **Application Director**, application control buttons apply as follows:

NOTE:

Be careful about using station controls, and understand the difference between them before using them.

- **Start**
Enabled if the selected station has an "Idle" or "Failed" status in the "installed applications" area. When pressed, that host's platform daemon immediately starts that station. Text in the "station output" area is cleared, and output messages begin with the new startup of that station.
- **Stop**
Enabled if the selected station has a "Running" status in the "installed applications" area. When pressed, a popup confirmation appears. If you confirm, the host's platform daemon shuts the station down *gracefully* (saving configuration to its config.bog file, and potentially saving history data).
- **Restart**
Enabled if the selected station has a "Running" status in the "installed applications" area. When pressed, a popup confirmation dialog appears. If you confirm, the host's platform daemon shuts the station down gracefully, then restarts it.
- **Reboot**
Always enabled. When pressed, a popup confirmation dialog appears. If you confirm, the selected host is rebooted. This is considered a drastic action. For details, see ["Reboot" on page 65, page 83](#).
- **Kill**
Enabled only if the selected station has a status of "Starting", "Stopping", or "Running" the "installed applications" area. When pressed, a popup confirmation dialog appears. If you confirm, the host's platform daemon terminates the station process immediately.

Always use **Stop** instead of **Kill**, unless unavailable (stuck for a long time as either "Starting" or "Stopping"). Unlike a station stop, a station kill does not cause the station to save its database (config.bog), histories or alarms, nor does it update the "station output" area.
- **Dump Threads**
Enabled only if the station has a "Running" status in the "installed applications" area. When pressed, the host's platform daemon has the station send a VM thread dump to its station output.
- **Save Bog**
Enabled only if the station has a "Running" status in the "installed applications" area. When pressed, the host's platform daemon has the station locally save its configuration to config.bog.
- **Verify Software**
Enabled regardless of station status. When pressed, Workbench parses the station's config.bog file and the host's platform.bog file, looking for module references. Workbench then checks to see if those modules (and any other software upon which they depend) are installed. Any missing software is listed in a popup dialog, and if available in your Workbench installation, the dialog offers to install the missing software into the remote host.

Only modules (or versions of modules) needed by the station are installed that do not require commissioning. If the station needs modules that require commissioning, meaning an upgrade of core Niagara software, those modules are not copied.

Output control buttons

For a selected station in the **Application Director**, output control buttons ([Figure 19](#) [Figure 23, page 35](#)) are as follows:

- **Clear Output**
Enabled regardless of station status. When pressed, all text in the "standard output" area is cleared.

NOTE:

Data in standard output area is fetched from a memory buffer in the platform daemon. Clearing the output does not actually clear the daemon's buffer. Therefore, if you change the selection away from, and then back to the station, it re-fetches all buffered data.

- **Pause Output**

Enabled if the selected station has a "Running" status. When pressed, the button toggles to "**Load Output**", and the next press back toggles back to "**Pause Output**" (and so on).

- During a paused output, text remains frozen in the "standard output" area. This is useful when the station is rapidly writing output.
- When you press **Load Output**, text in the "station output" area is reloaded with the station's buffered output, and output remains updating in real time.

- **Output Dialog**

Enabled regardless of station status. When pressed, it produces a separate "non-modal" output window displaying the same output text as in the Application Director's "standard output" area. Included are buttons to **Dump Threads**, **Pause Output**, **Clear Output**, and **Close** the window.

NOTE:

You may find this compact version of a station's standard output easier to work with than in the main area of the **Application Director**. Also, if needed you can open multiple output dialogs for comparison purposes.

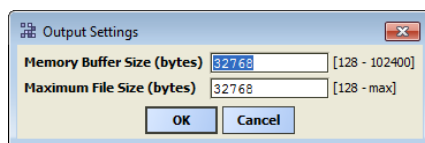
- **Stream To File**

Opens a browser window for assigning a file name. Once this file is opened, the system saves all application output to this file.

Output Settings

For the selected station in the **Application Director**, the **Output Settings** button produces a dialog in which you specify how the platform daemon buffers the output from that station.

Figure 20 Output Settings dialog



The two available settings are in bytes, and are:

- **Memory Buffer Size**

Size of the memory buffer for the station output. If the station creates more output than the size of the memory buffer, the oldest output is lost.

- **Maximum File Size**

When the station stops, its output buffer is written to a console.txt file. This setting defines the maximum size of that file.

NOTE:

Changes to either output setting may result in the output buffer's contents to be cleared.

Certificate Management view

The Certificate Management view allows you to create digital certificates and Certificate Signing Requests (CSRs). You also use this view to import and export keys and certificates to and from the Workbench, platform and station stores.

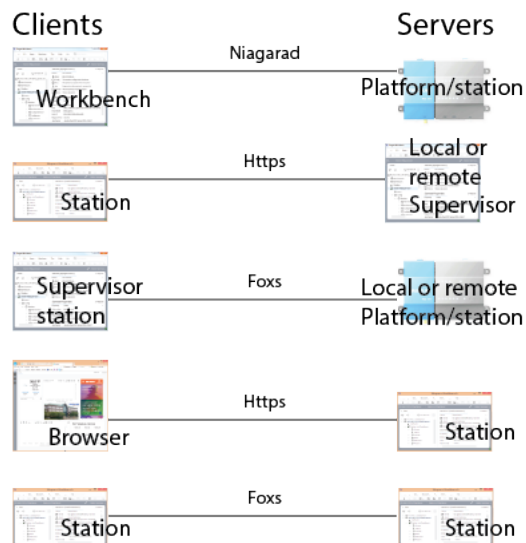
You use this view to manage PKI (Public Key Infrastructure) and self-signed digital certificates to secure communication within a Niagara network. Certificates secure TLS connections to this host.

Client/server relationships

Client/server relationships identify the connections that require protection. Niagara 4.0 client/server relationships vary depending on how you configure and use a system. Workbench is always a client. A platform is always a server. A station may be a client and a server.

With the exception of Workbench, Niagara entities may function as clients or servers in a NiagaraNetwork. Workbench only functions as a client.

Figure 21 Communication relationships



The system protocols that manage communications between these entities are:

- Platform connections from Workbench (client) to controller or Supervisor PC platform daemon (server) use Niagarad. A secure platform connection is sometimes referred to as platformtls. You enable platformtls using the Platform Administration view.
- Local station connections (Supervisor and platform) use Foxs. You enable these connections in a station's FoxService (**Config**→**Services**→**FoxService**).
- Browser connections use Https, as well as Foxs if you are using the Java Applet profile. You enable these connections using the station's WebService (**Config**→**Services**→**WebService**).
- Client connections to the station's email server, if applicable. You enable secure email using the station's EmailService (**Config**→**Services**→**EmailService**).

These relationships determine an entity's certificate requirements. For example, a station requires a signed server certificate, which it uses when it functions as a server, and a copy of the root CA certificate, which it uses when it functions as a client. Setting up digital certificates for identity verification involves creating separate certificates to verify the identity of each server. Each server's unique certificate, signed by a CA (Certificate Authority), resides in its **User Key Store**. Each client requires the root CA certificate used to sign each server certificate. The root CA certificate resides in the platform/station **System** or **User Trust Store**.

User Key Store tab

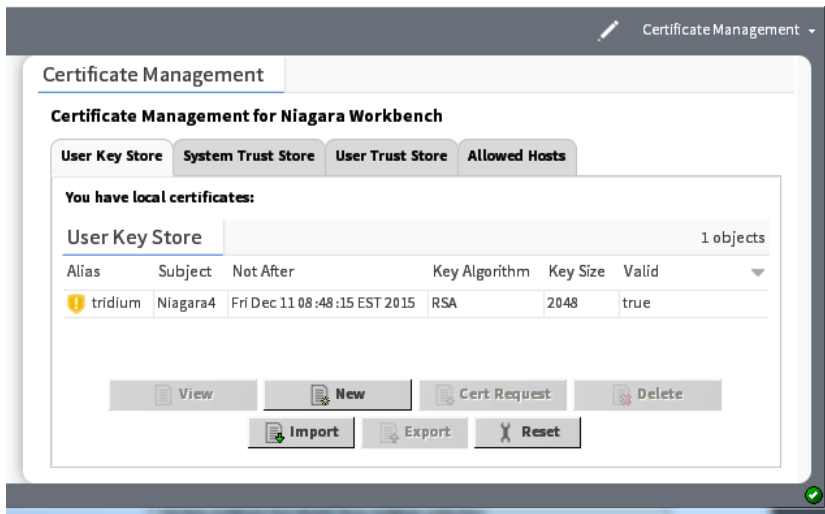
The **User Key Stores** contain server certificates and self-signed certificates with their matching keys. Each certificate has a pair of unique private and public encryption keys for each platform. A **User Key Store** supports the server side of the relationship by sending one of its signed server certificates in response to a client (Workbench, platform or station) request to connect.

If there are no certificates in a **User Key Store** when the server starts, such as when booting a new platform or station, the platform or station creates a default, self-signed certificate. This certificate must be approved as an allowed host. This is why you often see the certificate popup when opening a platform or station.

Default self-signed certificates have the same name in each **User Key Store** (`tridium`), however, each certificate is unique for each instance.

Clicking the **New** and **Import** buttons also adds certificates to the **User Key Store**.

Figure 22 Example of a Key Store



Name	Value	Description
Alias	text	A short name used to distinguish certificates from one another in the Key Store . This property is required. It may identify the type of certificate (root, intermediate, server), location or function. This name does not have to match when comparing the server certificate with the CA certificate in the client's Trust Store.
Issued By	text	Identifies the entity that signed the certificate.
Subject	text	Specifies the Distinguished Name, the name of the company that owns the certificate.
Not Before	date	Specifies the date before which the certificate is not valid. This date on a server certificate should not exceed the Not Before date on the root CA certificate used to sign it.
Not After	date	Specifies the expiration date for the certificate. This date on a server certificate should not exceed the Not After date on the root CA certificate used to sign it.
Key Algorithm	text	Refers to the cryptographic formula used to calculate the certificate keys.

Name	Value	Description
Key Size	number	Specifies the size of the keys in bits. Four key sizes are allowed: 1024 bits, 2048 bits (this is the default), 3072 bits, and 4096 bits. Larger keys take longer to generate but offer greater security.
Signature Algorithm	formula text	Specifies the cryptographic formula used to sign the certificate.
Signature Size	KB	Specifies the size of the signature.
Valid		Specifies certificate dates.
Self Signed	text	Read-only. Indicates that the certificate was signed with its own private key.

User Key Store buttons

Name	Value	Description
View	button	Displays details for the selected item
New	button	Opens the window used to create the entity you are working on.
Cert Request	button	Opens a Certificate Request window, which is used to create a Certificate Signing Request (CSR).
Delete	button	Removes the selected record from the database.
Import	button	Adds an imported item to the database.
Export	button	Saves a copy of the selected record to the hard disk. For certificates, the file extension is .pem.
Reset	button	Deletes all certificates in the User Key Store and creates a new default certificate. It does not matter which certificate is selected when you click Reset . CAUTION: Do not reset without considering the consequences. The Reset button facilitates creating a new key pair (private and public keys) for the entity, but may disable connections if valid certificates are already in use. Export all certificates before you reset.

Trust Store tabs

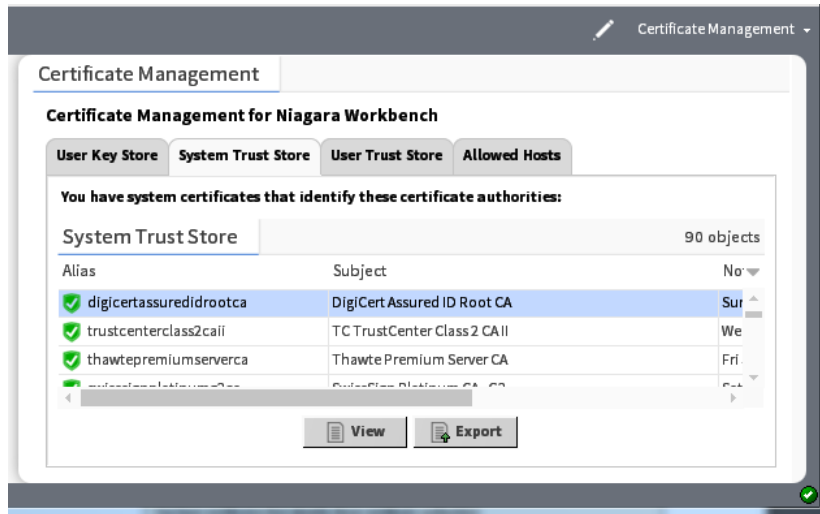
The **Trust Stores** contain signed and trusted root certificates with their public keys. These stores contain no private keys. A **Trust Store** supports the client side of the relationship by using its root CA certificates to verify the signatures of the certificates it receives from each server. If a client cannot validate a server certificate's signature, an error message allows you to approve or reject a security exemption (on the **Allowed Hosts** tab).

The **System Trust Stores** contain installed signed certificates by trusted entities (CA authorities) recognized by the Java Runtime Engine (JRE) of the currently opened platform. The **User Trust Stores** contain installed signed certificates by trusted entities that you have imported (your own certificates).

Only certificates with public keys are stored in the **Trust Stores**. The majority of certificates in the **System Trust Store** come from the JRE. You add your own certificates to a **User Trust Store** by importing them.

Feel free to pass out such root certificates to your team; share them with your customers; make sure that any client that needs to connect to one of your servers has the server’s root certificate in its client **Trust Store**.

Figure 23 Example of a Trust Store



Trust Store columns

Name	Value	Description
Alias	text	A short name used to distinguish certificates from one another in the Key Store . This property is required. It may identify the type of certificate (root, intermediate, server), location or function. This name does not have to match when comparing the server certificate with the CA certificate in the client’s Trust Store.
Issued By	text	Identifies the entity that signed the certificate.
Subject	text	Specifies the Distinguished Name, the name of the company that owns the certificate.
Not Before	date	Specifies the date before which the certificate is not valid. This date on a server certificate should not exceed the Not Before date on the root CA certificate used to sign it.
Not After	date	Specifies the expiration date for the certificate. This date on a server certificate should not exceed the Not After date on the root CA certificate used to sign it.
Key Algorithm	text	Refers to the cryptographic formula used to calculate the certificate keys.
Key Size	number	Specifies the size of the keys in bits. Four key sizes are allowed: 1024 bits, 2048 bits (this is the default), 3072 bits, and 4096 bits. Larger keys take longer to generate but offer greater security.
Signature Algorithm	formula text	Specifies the cryptographic formula used to sign the certificate.

Name	Value	Description
Signature Size	KB	Specifies the size of the signature.
Valid		Specifies certificate dates.
Self Signed	text	Read-only. Indicates that the certificate was signed with its own private key.

Trust Store buttons

Name	Value	Description
View	button	Displays details for the selected item
Delete	button	Removes the selected record from the database.
Import	button	Adds an imported item to the database.
Export	button	Saves a copy of the selected record to the hard disk. For certificates, the file extension is .pem.

Allowed Hosts tab

The **Allowed Hosts** tab contains security exemptions for the currently open platform. These are the certificates (signed or self-signed) received by a client from a server (host) that could not be validated against a root CA certificate in a client **Trust Store**. Whether you approve or reject the certificate, the system lists it in the **Allowed Hosts** list.

To be authentic, a root CA certificate in the client's **System** or **User Trust Store** must be able to validate the server certificate's signature, and the **Subject** of the root CA certificate must be the same as the **Issuer** of the server certificate.

Allowing exemptions makes it possible for a human operator to override the lack of trust between a server and client when the human user knows the server can be trusted.

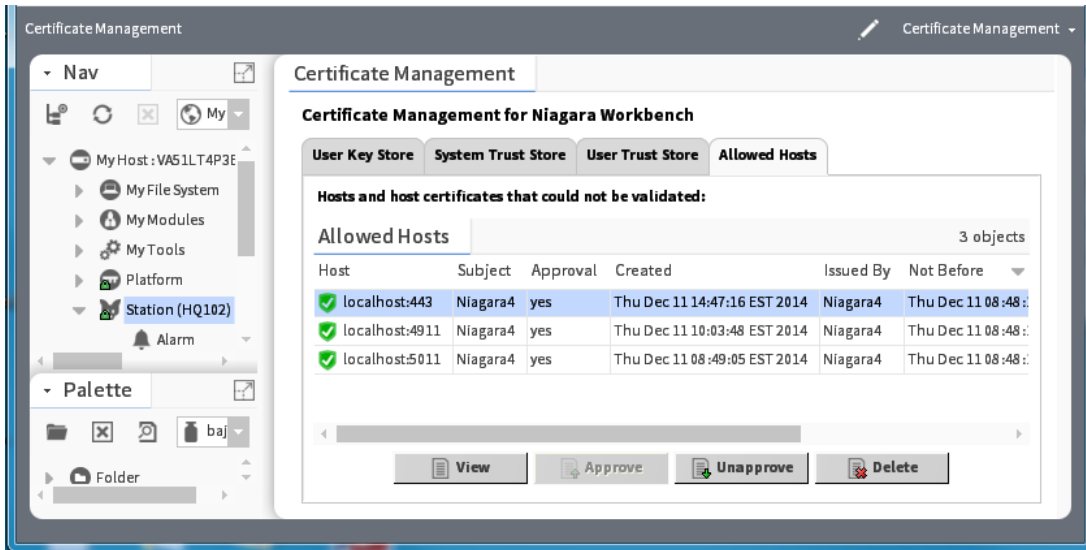
If this is a Workbench to station connection, the system prompts you to approve the host exemption. Workbench challenges server identity at connection for unapproved hosts and, unless specific permission is granted, prohibits communication. Once permission is granted, future communication occurs automatically (you still have to log in). Both approved and unapproved hosts remain in this list until deleted.

If this is a station to station connection, and there is a problem with the certificates, the connection fails silently. There is no prompt to approve the host exemption. However, the last failure cause in the station (expand the station **ClientConnection** under **NiagaraNetwork**) reports the problem.

The approved host exemption in the **Allowed Hosts** list is only valid when a client connects to the server using the IP address or domain name that was used when the system originally created the exemption. If you use a different IP address or domain name to connect to the server, you will need to approve an updated exemption. The same is true if a new self-signed certificate is generated on the host.

Allowed Hosts columns

Figure 24 Example of an Allowed Hosts list



Name	Value	Description
Host	text	Specifies the server, usually an IP address.
Subject	text	Specifies the Distinguished Name, the name of the company that owns the certificate.
Approval	Yes or No	Specifies the servers within the network to which the a client may connect. If approval is set to no, the system does not allow the client to connect.
Created	date	Identifies the date the record was created.
Issued By	text	Identifies the entity that signed the certificate.
Not Before	date	Specifies the date before which the certificate is not valid. This date on a server certificate should not exceed the Not Before date on the root CA certificate used to sign it.
Not After	date	Specifies the expiration date for the certificate. This date on a server certificate should not exceed the Not After date on the root CA certificate used to sign it.
Key Algorithm	text	Refers to the cryptographic formula used to calculate the certificate keys.
Key Size	number	Specifies the size of the keys in bits. Four key sizes are allowed: 1024 bits, 2048 bits (this is the default), 3072 bits, and 4096 bits. Larger keys take longer to generate but offer greater security.
Signature Algorithm	formula text	Specifies the cryptographic formula used to sign the certificate.
Signature Size	KB	Specifies the size of the signature.
Valid		Specifies certificate dates.

Allowed Hosts buttons

Name	Value	Description
View	button	Displays details for the selected item
Approve	Yes or No	Designates the server as an allowed host.
Unapprove	Yes or No	Does not allow connection to this server host. The system terminates any attempted communication.

Distribution File Installer

The Distribution File Installer is one of several platform views used to install dist (distribution) files. A dist file is a zip that contains other files and a manifest that describes the contents of the distribution. For technical details, see the “Distributions” section in the online *Niagara Developer Guide*. The Distribution File Installer is separate from the AX to N4 Migration Tool used to upgrade a station to N4. For related details, see the *AX to N4 Migration Guide*.

Use this view for either of these two tasks:

- To *restore* a locally available backup .dist file to a remote controller. Such a restore can be initiated using the **Backup** command from the platform’s Platform Administration view, or more commonly from the **BackupService** in the station running on that host. For details, see [“Restoring a backup dist” on page 39, page 48](#).
- To install a clean dist file. This downgrades a controller to an older Niagara 4 release level, or restores it to a known empty state. Following a clean dist install, you must *recommission* the controller, as this *wipes* the file system (almost all Niagara software, as well as all station files), leaving the controller in an empty near-factory state. For details, see [“Wiping clean a JACE \(Clean Dist\)” on page 41, page 50](#).

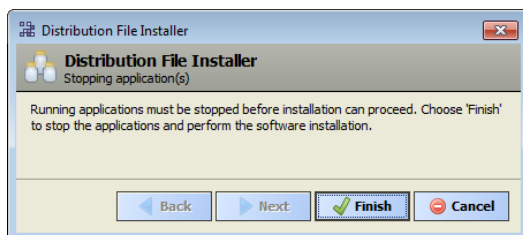
NOTE:

Do not use this view to upgrade a controller. Instead, use the **Commissioning Wizard** to upgrade Niagara in a controller. The **Commissioning Wizard** is a right-click option on a platform when opened in Workbench. See [“Upgrading a JACE” on page 42, page 52](#).

If a station is already running on the remote host, any dist file install requires that all applications be stopped, after which *all are invariably overwritten!* After selecting a dist file, the Installer provides a confirmation dialog for this, as shown in [Figure 25](#) [Figure 26, page 45](#). When you finalize the install (click **Finish**), the Installer automatically stops the station, then continues with the distribution file install process.

Before finalizing any dist installation, make sure that any controlled equipment, which might be adversely affected by the station stopping (and the removal of software) is put in a manually controlled state.

Figure 25 Stop station dialog in Distribution File Installer



The following sections provide more details on the **Distribution File Installer**:

- [Operation of the Distribution File Installer, page 46](#)
- [Restoring a backup dist, page 48](#)
- [Wiping clean a JACE \(Clean Dist\), page 50](#)

Operation of the Distribution File Installer

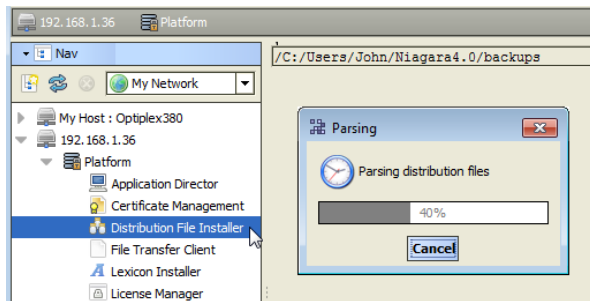
The following subsections explain more about dist file selection and the install process:

- [Dist file selection, page 46](#)
- [Distribution file install process, page 47](#)

Dist file selection

When you select the Distribution File Installer, it “parses” through the dist files on your local PC, using the last source folder selected. See the figure below.

Figure 26 Parsing dialog in Distribution File Installer

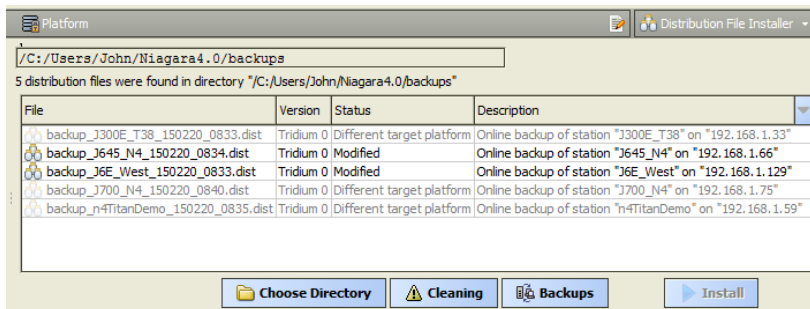


By default, the first time you use the Installer, the “backups” folder under your Workbench **User Home** (~\backups) is parsed. If that folder does not exist yet (no backups have been made), then the “cleanDist” folder under Niagara **Sys Home** (!\cleanDist) is parsed instead.

At the bottom of the view, the **Cleaning** and **Backups** buttons provide shortcuts to these two folder areas. If needed, you can also click the **Choose Directory** button for a **Change Directory** dialog, and point the Installer to that location.

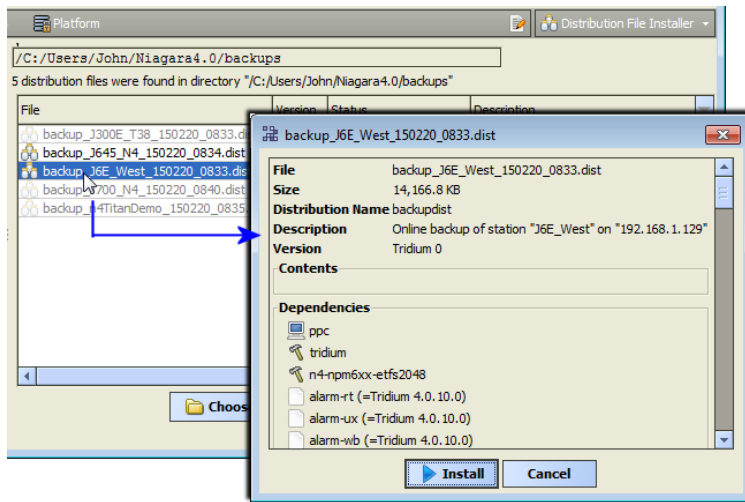
When parsing completes, a table of the found dist files appears, with the appropriate dist files available for selection in the table, as shown below (using default software location).

Figure 27 Available dist files in Distribution File Installer



Dist files that are inappropriate, for example that are for a different target platform or have unmet dependencies, are *dimmed*—the **Install** button does not become active if you select such a file. For details on any dist file, double-click it for a popup, including a list of its dependencies, as shown below.

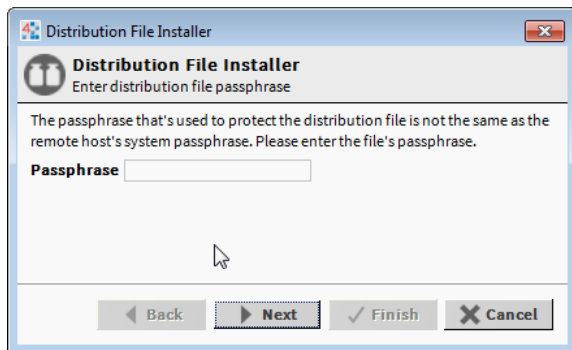
Figure 28 Example details dialog for a dist file, showing dependencies.



When you click **Install**, the system attempts to validate the file's passphrase.

- If the file passphrase and system passphrase are the same, the process continues without prompting for a passphrase.
- If the file passphrase and system passphrase are different, you are prompted to enter the file's passphrase, as shown here.

Figure 29 Distribution File Installer prompts for file passphrase

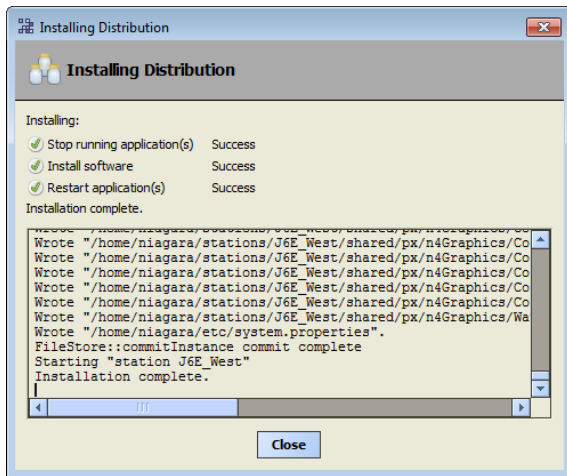


NOTE: If prompted for the .dist file passphrase and you do not know it, you cannot install the file.

Distribution file install process

After proceeding with **Finish**, the dist installation process appears in a dialog that tracks its progress as it continues, as shown below.

Figure 30 Distribution File Installer progress dialog



After the distribution file (and modules, if selected) are installed on the JACE platform, the JACE is re-booted, and the progress dialog indicates complete. You must click the **Close** button to continue. You can then reopen a platform connection, perhaps to view output in the **Application Director**.

Restoring a backup dist

- [About backup dist files, page 48](#)
- [Restoring a station backup, page 48](#)

About backup dist files

A backup dist includes not only the entire station folder, but all other Niagara configuration that may be customized for that platform. This allows for a complete replication from the one backup file.

Typically, station backups are done from Workbench *station connections* (station is running, and has the BackupService). In the Nav tree, right-click the opened station, and select **Backup Station**. Less typical is an “offline backup” from the **Platform Administration** view.

By default, station backup dist files are saved in your Workbench **User Home**, in a ~/backups folder.

Restoring a station backup


The following procedure describes restoring a backup for an Niagara 4 JACE controller.

NOTE:

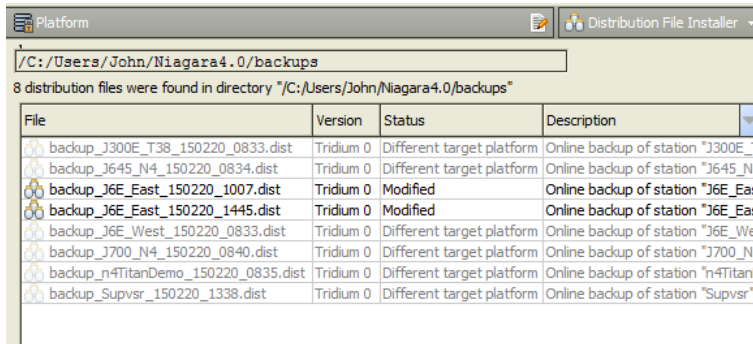
To be able to restore a backup dist file, your Workbench installation requires the same versions of software, including modules, to be available in its “software database.” Therefore, it is recommended that you make and keep frequent backups as you upgrade JACEs. Also for this reason, you may need to import the software database from prior revisions of Niagara into your current Workbench installation. For related details, see [“About your software database”, page 85](#)

Restoring a backup dist

Prerequisites: Station backup dist file for the target N4 JACE controller. The software database of your Niagara 4 installation must include matching versions of all software modules used by the station when the station backup was made, or else the backup restore will fail.

- Step 1 Using Niagara 4 Workbench, open a platform connection to the JACE.
- Step 2 In the **Distribution File Installer** click the  **Backups** button for the !/backups folder. or if needed, click the **Choose Directory** button to point to another backup dist file location.

The Installer parses through the dist files, and makes selectable only those files that are compatible with the opened JACE platform. When done parsing, available backup dists appear listed, as shown below.



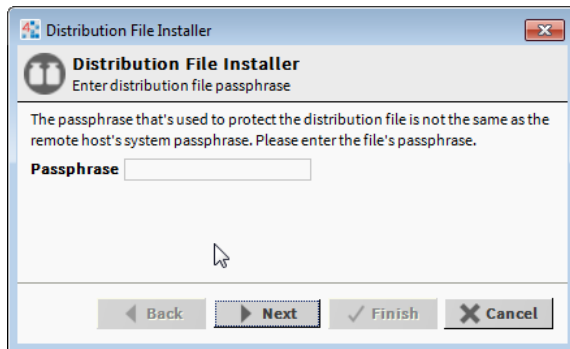
1. If needed, you can double-click any .dist for more details in a popup dialog.

Step 3 To restore any selected backup, click **Install**.

When you click **Install**, the system attempts to validate the file's passphrase.

- If the file passphrase and system passphrase are the same, the process continues without prompting for a passphrase.
- If the file passphrase and system passphrase are different, you are prompted to enter the file's passphrase, as shown here.

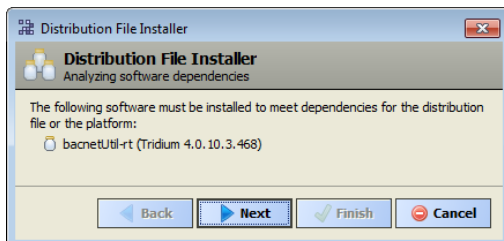
Figure 31 Distribution File Installer prompts for file passphrase



NOTE: If prompted for the .dist file passphrase and you do not know it, you cannot install the file.

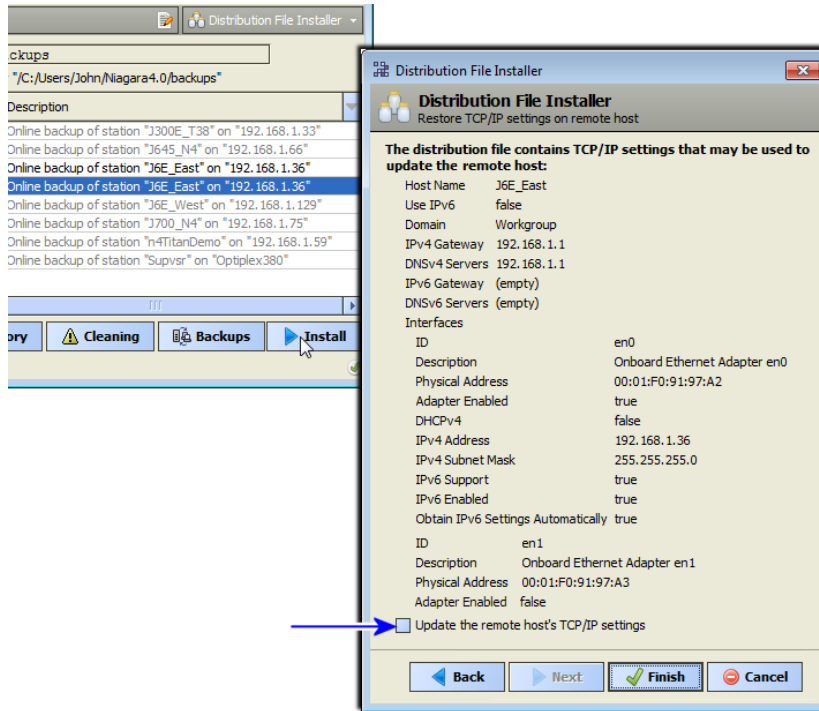
Proceeding with the Install, if the host is already running a station, a dialog appears telling you that it must be stopped.

2. If the station backup .dist file contains software modules different from (or in addition to) those already installed in the remote host, another dialog appears to inform you, as shown below.



Step 4 Click **Next** to continue.

3. As shown below, when restoring a backup, another dialog always asks if you wish to restore the TCP/IP settings stored in the dist file (as displayed) into the remote host.



TCP/IP settings contained in the dist file are listed, and by default, the checkbox “Update the remote host’s TCP/IP settings” is *cleared*.

- If you *leave* this cleared, after the dist file installs and host reboots, it *retains its current* TCP/IP settings. This allows you to use the same dist file on differently addressed hosts, if needed.
- If you *check* (select) this checkbox, after the dist file installs (and host reboots), it uses the TCP/IP settings stored in the dist file—meaning the same ones shown in this dialog.

Step 5 Click **Finish** to begin installation. See [“Distribution file install process”](#), page 47.

Wiping clean a JACE (Clean Dist)

At times it may be necessary to restore an N4 JACE controller to a known good “empty” state, either to re-commission with the current Niagara 4 release build, or else before recommissioning with an *earlier* Niagara 4 build. To do this, you can install a Niagara 4 “clean dist” (distribution) file.

NOTE: For controllers on which you have just installed clean dist, the system passphrase default value is the same as the default platform password. On the first commissioning of such controllers you are prompted to change the passphrase from the default value.

Wiping clean a JACE is typically unnecessary if upgrading an operational N4 JACE controller to a later Niagara 4 software build—simply using the **Commissioning Wizard** should be all that is necessary. However, if *downgrading* a JACE to an earlier N4 build, install a clean dist file first, to avoid compatibility problems. This applies especially to JACE controllers, as binaries for the (QNX) OS are included in dist files.

NOTE:

Clean dist files are unavailable to downgrade an N4 JACE controller to an AX controller, nor can they be installed in an AX JACE controller. Instead, there are special “AXtoN4” dist files you can use (one time only) to convert an AX controller to an N4 unit, using the Niagara 4 **Distribution File Installer**. However, before doing this, usually *much station migration work* is required. For related details, see the *AX to N4 Migration Guide*.

See the following for more details on JACE clean dist files:

- “Clean dist installation preparation”, page 51
- “Installing a clean dist file”, page 51

Clean dist installation preparation

Installing a clean dist wipes the entire file system and installs an appropriate version of Niagara platform daemon, resetting the unit to a near factory state. If the controller came with an appliance installed, installing a clean dist will also remove that application. Only the following settings are preserved:

- TCP/IP settings
- license files
- brand.properties
- most secure communication (TLS) configuration

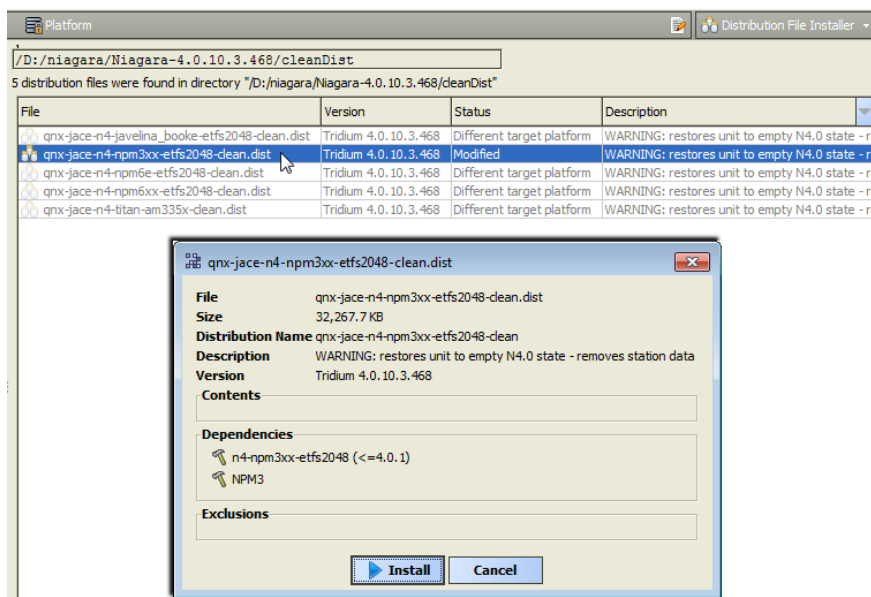
All *other data is deleted* from the file system, including station bog files, Px files, modules, etc. Note that the unit’s TLS private key information is also deleted. In addition, installing a clean dist deletes all configured platform users, restoring the factory-default platform credentials and port (3011).

Therefore before installing a clean dist file, make sure to *backup* station files plus any other items on the controller you wish to keep. You should always backup (export) certificate keys of any TLS-configured unit, such that if the controller needed to be replaced (hardware swap-out), you could re-import those keys.

Note that *after* installing a clean dist, you must always recommission the controller for Niagara 4, using the **Commissioning Wizard**. For related details, see “About the Commissioning Wizard” in the *JACE Niagara 4 Install & Startup Guide*.

Each clean dist file has the suffix `-clean` in its name. They are installed located in your Sys Home `!cleanDist` folder—apart from other dist files under your software database.

Figure 32 Clean dist file shown selected in the Distribution File Installer



Clean dist files appear listed with a “WARNING” in the Description, as shown in the one selected in the figure above. Only the appropriate one for the currently opened platform will be selectable.

Installing a clean dist file

The following procedure describes installing a clean dist file in a JACE controller.

Clean disting a controller

Prerequisites:

- The JACE controller is a Niagara 4 unit. You have backed up any station files as well as any other files needed later, for example digital certificate keys. See [“Clean dist installation preparation”, page 51](#).

Step 1 Using Niagara 4 Workbench, open a platform connection to the controller.

Step 2 Open the **Distribution File Installer** and clic the k **Cleaning** button to access the !cleanDist directory.

Step 3 Select the appropriate clean dist file for the platform and install.

- The file system clean will take a few minutes, then the controller will automatically reboot. Wait for the re-boot to complete.

NOTE:

After reboot from a clean dist install, the controller is using default platform credentials and port (3011).

Step 4 To re-install the software versions to the controller:

- Use a Niagara 4 Workbench that uses the same software versions that you want on the controller, and use the platform **Commissioning Wizard** to install the desired software build. For details, refer to the section *“About the Commissioning Wizard”* in the *JACE Niagara 4 Install & Startup Guide*.
- If you have a backup dist file for the controller that was made when it had the desired prior N4 software versions, use the **Distribution File Installer** to install it. See the previous section, [“Restoring a backup dist” on page 39, page 48](#).

Upgrading a controller

You must use the **Commissioning Wizard** to upgrade Niagara software in a JACE. This means either an “update” upgrade (say from build 4.0.101 to 4.0.106), or a full “minor” release upgrade, for example build 4.0.106 to build 4.1.88.

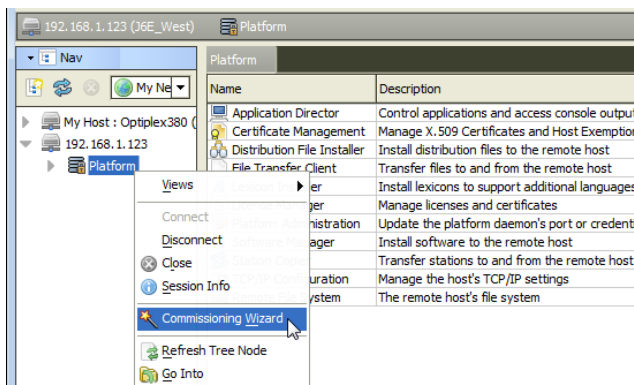
NOTE:

When updating a multi-station system for the first time to an update release, it is recommended to upgrade a Supervisor before its subordinate JACEs.

Any JACE to be upgraded from one minor version to another, say from 4.0.nn to 4.1.nn, *requires a license upgrade, purchased before starting the upgrade*. Otherwise, the Commissioning Wizard in Workbench will not perform the upgrade. This prevents the scenario where an upgraded JACE cannot start its station, due to a licensing error.

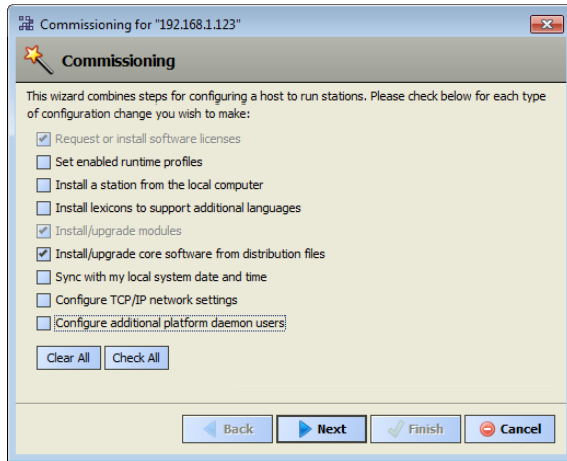
With a platform connection to any Niagara JACE, access the Commissioning Wizard by simply right-clicking on that platform and selecting it from the menu ([Figure 33](#) [Figure 32, page 52](#)).

Figure 33 Commissioning Wizard is right-click option of opened platform



If a controller upgrade, in the wizard's opening selection of steps you typically *deselect* most items that were previously run at the controller's initial commissioning time—for example to set enabled runtime profiles, set date and time, configure TCP/IP settings, and so on. See the figure below.

Figure 34 Typical upgrade selections for existing Niagara JACE (already running a station)



To upgrade a Niagara JACE, you *do select*:

- In the case where the upgrade requires an updated license installed:
"Request or install software licenses" (this may already be pre-selected).
- In the case where a station install also requires commissioning the JACE (i.e. upgrade):
"Install station from the local computer"
- And always:
"Install/upgrade core software from distribution files"

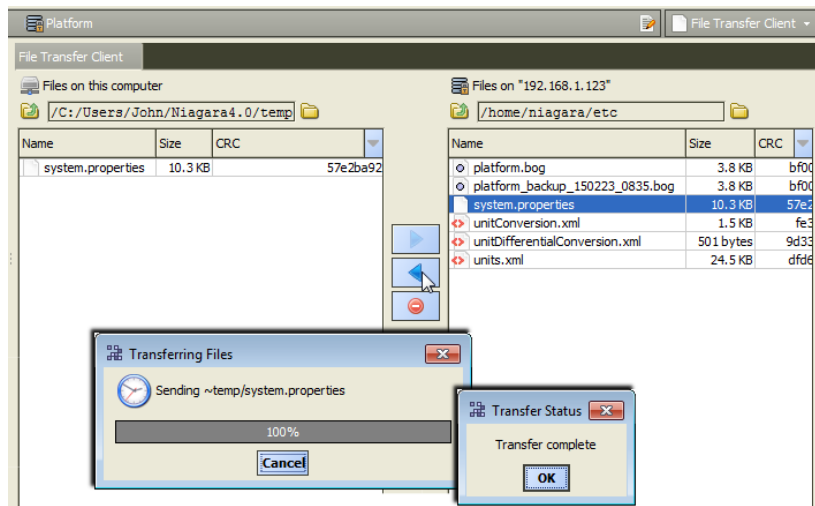
When you proceed in this manner, the wizard automatically finds and selects all core distributions needed for the JACE. Then, in the pre-selected "Install/Upgrade modules" step, the wizard provides the option to also upgrade all out-of-date software modules (always do that).

A final summary step allows you to review the upgrade before the wizard executes and performs its operations. For further details, refer to the section "About the Commissioning Wizard" in the *JACE Niagara 4 Install & Startup Guide*.

File Transfer Client

The File Transfer Client is one of several platform views. It allows you to copy files and/or folders between your Workbench PC and the remote Niagara platform. You can also use it to delete files and folders.

Figure 35 File Transfer Client



This may be useful if you wish to copy graphics images to a controller, as one example. Or, use it to copy a text file *from* a **User Home** folder on a remote controller (say, `~etc/system.properties`) to your local PC, to allow editing. Then use the File Transfer Client to copy the edited version back to the controller's `~etc` folder. An example of this is shown above. Also see [“system.properties notes”, page 55](#).

However, *do not use* the File Transfer Client to copy modules to a JACE, as “runtime profile types” are not applied, nor are module dependencies. As a result, incorrect or missing modules may result. Always use the platform **Software Manager** to install (or uninstall) software modules on a JACE controller.

For local-to-remote transfer of a file containing encrypted sensitive data, the file Transfer Client does not prompt you to enter a passphrase. Instead, the results of the transfer will be one of the following:

- Transfer completes successfully if:
 -
 - ◆ the file is protected with a passphrase that matches the system passphrase
- Transfer fails if
 - the file is protected with a passphrase that differs from the system passphrase
 - you include more than one protected file in the same transfer

The File Transfer Client provides a two-pane view, as shown in [Figure 34, page 54](#).

CAUTION:

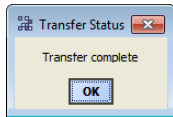
Be careful when using the File Transfer Client, especially when copying files to a target JACE platform, or whenever using the delete (X) control. Note that in either direction, when transferring a file and an identically-named file already exists, a popup confirmation dialog appears before the copy. A popup confirmation dialog also appears before any delete. However, after confirmation there is no “Undo.”

In the File Transfer Client view, the *left* pane provides access to *local* (Workbench PC) files, and the *right* pane provides access to files on the *remote* platform.

Use is straightforward, you simply click navigation controls at the top of each pane to go to the appropriate location for source and target. Then you click one or more items on one side (as source) to select for copying to the other side (target). Then, you click the appropriate transfer arrow.

A dialog appears when all files are transferred, as shown below.

Figure 36 Transfer complete dialog



system.properties notes

Occasionally there may be a need to edit the `system.properties` file used by any station running on a host, which for any Niagara 4 platform is located in its (platform daemon) **User Home**, `~/etc` folder.

- On a remote JACE (QNX OS), this folder is at `/home/niagara/etc`.
- On a local Supervisor (Windows OS), this folder is at `C:\ProgramData\Niagara4.0\etc`.

Note the `system.properties` file used by your Workbench is similar to, but different from this same file in either of those locations above. That file is in the `etc` folder under your Workbench **User Home**, for example at `C:\Users\userName\Niagara4.0\etc`.

You cannot directly edit a JACE's `system.properties` file in place. Instead, you must copy it to your PC first (using the platform **File Transfer Client**) to edit.

Only a few entries in a `system.properties` file are typically processed. Most lines in this file are comments, which start with a `#` and are not processed. Comments "inactivate" many entries in this file—and typically these entries should remain inactive. To activate such an entry, you must delete the leading `#` character on that line of code.

CAUTION:

Editing (and activating) `system.properties` entries is an operation for advanced users, with the possibility of undesirable results. Read all comment lines carefully, and consult your support channel before making a change! Always save a backup copy of this file, and test after implementing a change.

Station restart after changes to system.properties

A station must be *restarted* before changes to `system.properties` become effective. Thus, after copying (transferring back) an edited `system.properties` to any JACE using the platform **File Transfer Client**, do the following:

Prerequisites:

1. **Stop** the station using the platform **Application Director**.
Wait for the station to stop completely, ensuring that it saves its database.
2. From the platform **Platform Administration** view, select **Reboot**.
Allow sufficient time for the JACE to reboot and station to start.
3. Reconnect to the JACE's station with Workbench to verify operation.

Lexicon Installer

The Lexicon Installer is one of several Niagara platform views. Currently, this view lets you install *file-based* Niagara "lexicon sets" from your Workbench PC to a remote JACE platform, as needed.

NOTE:

In Niagara 4, usage of this view and file-based lexicon sets is typically *not recommended*. Instead, make one or more *modules* of customized lexicons using the **Lexicon Module Builder**, and install them in a remote platform using the **Software Manager**. Otherwise, issues may occur in browser access of the hosted station. A summary of lexicons as modules is provided below.

Lexicons can also be installed as *modules* (.jar files), in which case you use the platform **Software Manager** (instead) for installation in remote JACE platforms. In fact, "standard lexicons" are distributed as modules, using a module file name convention of:

niagaraLexiconLc-rt.jar

where **Lc** is the two-character “language code”, such as **Fr** for French or **Es** for Spanish. Workbench provides a **Lexicon Tool** with a special “Lexicon Module Maker” view that you can use to modify or make new lexicon modules, from edited text-based lexicon files. For complete details, refer to the *Niagara Lexicon Guide*.

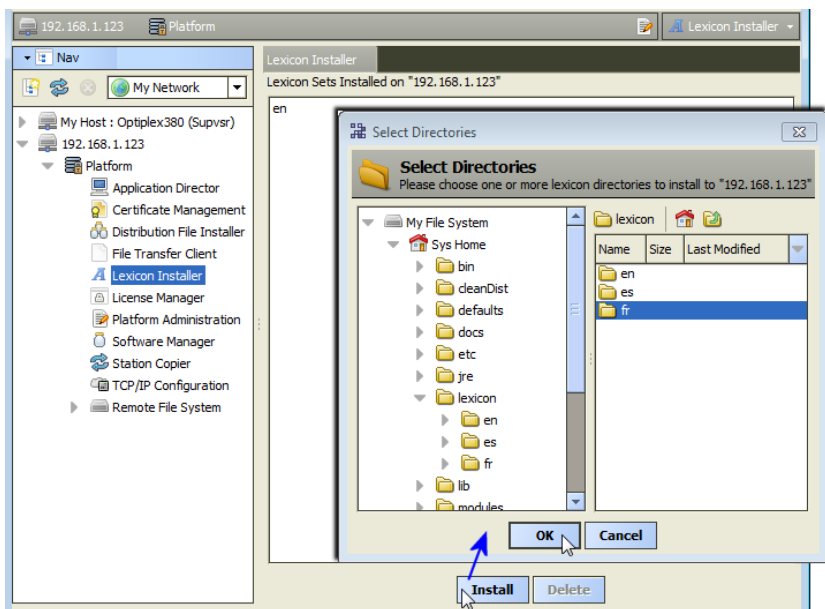
Lexicons in Niagara typically have one of two uses, depending on job location:

- International locations: For non-English language support
- Domestic (U.S.) locations: where you have modified the English (en) lexicon in order to change the wording used in default labels.

Beforehand, you typically use the **Lexicon Editor** view of the **Lexicon Tool** in Workbench to review and edit entries (or *keys*) in the individual lexicon files with localized values needed for language support.

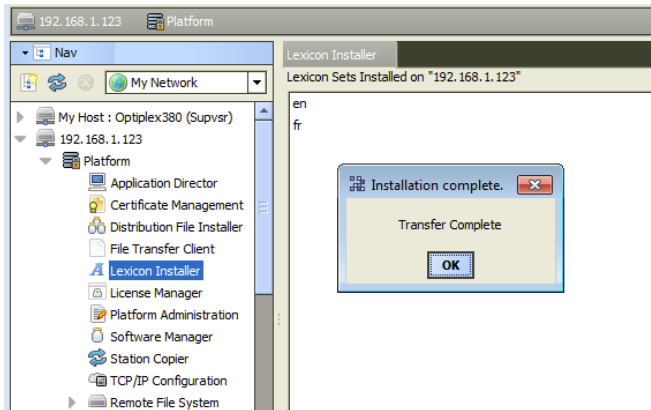
When you select Lexicon Installer, any existing file-based lexicon sets (already installed in that platform) are listed in the view pane. When you click **Install**, a “Select Directories” dialog appears for you to select lexicon sets (in the lexicon folder under your Workbench **Sys Home**) to install in the remote platform, as shown below.

Figure 37 Lexicon Installer, selecting lexicon



When you click **OK**, the selected lexicon directory or directories are installed in the remote JACE platform. An “Installation Complete” dialog appears when all files are transferred, as shown below.

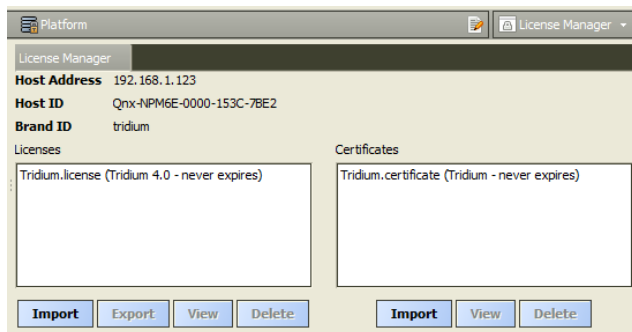
Figure 38 Lexicon Installer, lexicon installed



License Manager

The License Manager is one of several platform views. This view lets you install (import) licenses and certificates to a remote JACE platform, sourced either from your Workbench PC or the Niagara licensing server. You can also view contents of licenses and certificates, and if desired, delete them from a JACE.

Figure 39 License Manager lists existing Niagara licenses and certificates



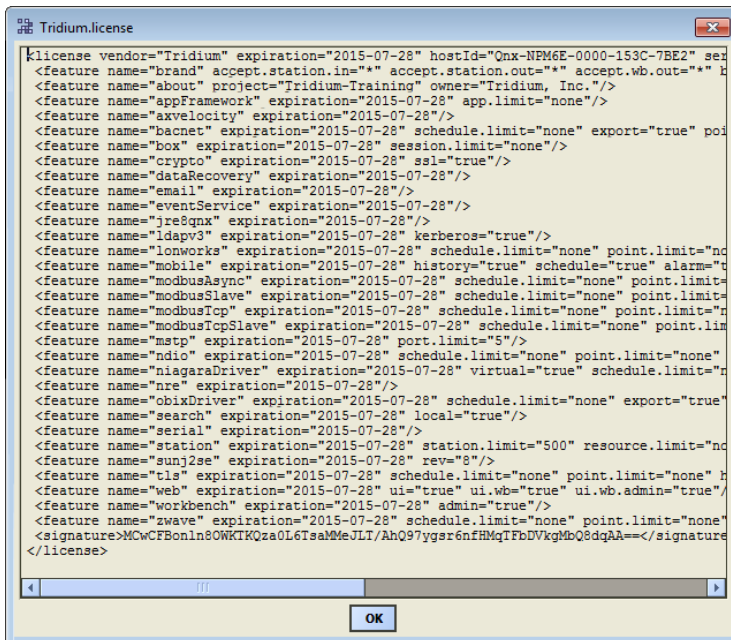
The License Manager lists any existing Niagara licenses and certificates (already installed in that platform), as shown in the figure above.

If selected, you can also view or delete an *existing* license file (**View** button is the same as simply double-clicking item, see figure below), or save (export) a license file as a "license archive" (.lar) file.

Buttons below each side let you install (import) a *new* license or certificate file. Typically, license files are imported from either the online licensing server or from your local license database.

Click a license or certificate to select it, or *double-click* to view in a dialog, as shown below.

Figure 40 Viewing a license in License Manager



A license and a certificate are each a digitally-signed text file, with differences briefly as follows:

- A *license* file is unique to a *specific Niagara host*, and enables a set of vendor *features*. All Niagara hosts require a branded “Tridium” license. If third-party modules are installed, one or more additional licenses may be needed. For details about license file contents, see the section “[About Niagara license files](#)” on [page 118](#).
- A *certificate* file varies by *vendor*, and matches that vendor to a public key used for encryption. It is used for verifying the authenticity of license files. All Niagara hosts require a “Tridium” certificate. If third-party modules are installed, one or more additional certificates may be needed.

CAUTION:

Do not delete an existing license or certificate without specific reason, as you will likely render the JACE inoperable until a proper license or certificate is reinstalled!

For further **License Manager** details, see the following sections:

- [License operations, page 58](#)
- [About the licensing server, page 61](#)

NOTE:

Workbench management of licenses uses a structured “local license database” and utilization of a “license archive file” format. In addition, a Workbench License Manager tool is available, which does not require a platform (or station) connection to use. These features are explained in an appendix to this document, “[License Tools and Files](#)”, along with details about the contents (features) of license files.

License operations

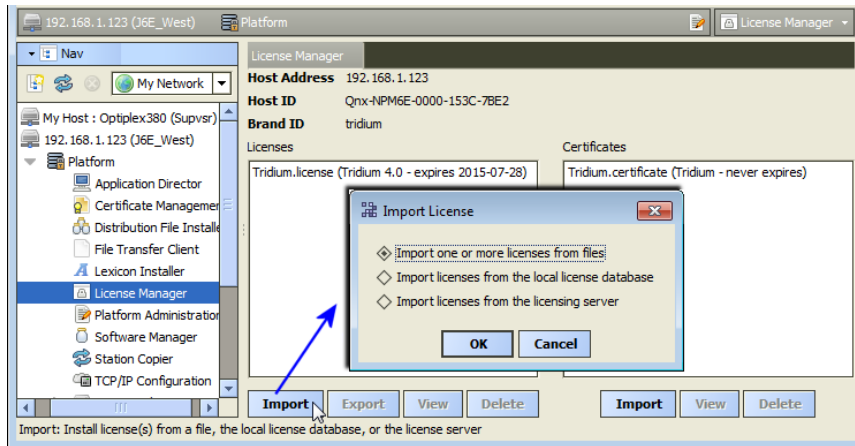
Below the left-hand *license side* of the **License Manager**, these two buttons (commands) are displayed in addition to **View** and **Delete**:

- [Import, page 59](#) — Always available, this provides various options for installing a license file from local files, from the licensing server, or from your “local license database.”
- [Export, page 59](#) — Available if you have a license selected, to save locally as a “license archive file.”

Import

If you choose **Import** from the License Manager, the Import License dialog asks you to select where the source license is, as shown below.

Figure 41 Import dialog from License Manager



Select *one* of the following options (depending on scenarios, some may be unavailable, as noted):

NOTE:

See [“License Import results”](#) on page 49, page 60 for details on results after making a selection below.

- Import one or more licenses from files

Always an available option, this provides a **Select File** dialog in which you can navigate to either a source license archive (.lar) file or an unzipped license file. When you select a license or license archive file, an attempt is made to install the license in the host platform.

- Import licenses from the local license database

This option will be unavailable (dim) if this host's license file is *not* in your local license database, or if the license in your local license database already *matches* the currently installed license. With this option selected, the license is immediately installed in the remote host platform. See [“About the local license database”](#) on page 116 for related details.

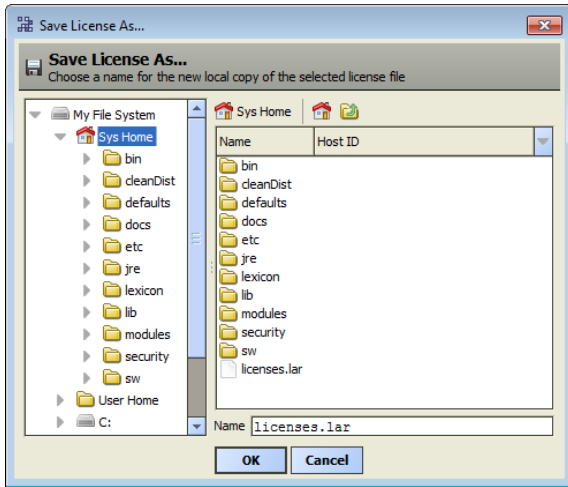
- Import licenses from the licensing server

Typically, this option is available if your Workbench PC has Internet connectivity. When you select this option, Workbench silently searches the licensing server and installs the license.

Export

With a license selected in the **License Manager**, the **Export** button provides a **Save License As...** dialog to save that license file locally on your Workbench PC, as a *license archive* (.lar) file, as shown below. For related details, see [“About license archive \(.lar\) files”](#) on page 117.

Figure 42 Save License As dialog



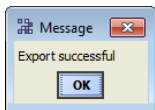
NOTE:

You can use the License Manager’s **Import** command to install any exported license archive, or the equivalent **Import File** command in the **Workbench License Manager** view of Workbench.

By default, a license archive file is saved in the root of your Niagara release directory. If needed, you can use the dialog’s navigation controls to specify another target folder or drive. Before saving, you can also *rename* the license archive file, to make it more identifiable. For example, instead of: licenses.lar, you could rename it MyJace6E.lar.

After exporting a license, a notification dialog appears in Workbench, as shown below.

Figure 43 Exported license archive notification dialog



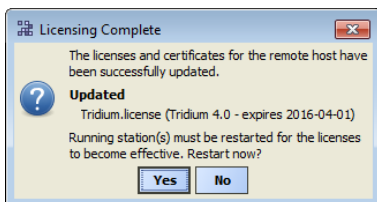
License Import results

Depending on the **Import** option chosen in the **License Manager** and the success of the import attempt, after you click **OK**, one of several dialogs may appear to signal completion, as follows:

- **Licensing Complete**

The license was successfully added, as shown below.

Figure 44 Licensing Complete dialog



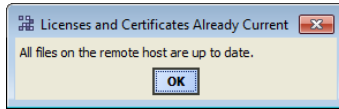
NOTE:

If a station is running on the host platform, this dialog informs you that the station must be restarted for the license(s) to become effective, and provides a **Yes** button to do this now. Or, you can select **No** and do this manually later.

- **Licenses and Certificates Already Current**

The license currently installed on the host already matches the source license (whether specifying any of the license import options). A dialog appears as shown below.

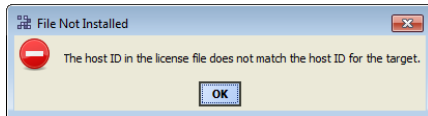
Figure 45 License and Certificates Already Current



- File Not Installed

No appropriate license (by host ID) was found in either the license file or the license archive specified when importing by file, noted with a dialog similar to below.

Figure 46 File Not Installed



- (License Request Form, in browser)

If importing from the license server, and an existing license was not found for this host platform, a separate window (of your default browser) opens with a license request form, showing the host ID for this host. See [Figure 47](#) [Figure 46 on page 50, page 62](#).

About the licensing server

For traditional Niagara license files validated against the Tridium certificate, installation can be automated from Workbench. All such purchased licenses (including JACEs, Supervisor, or Workstation-only) are stored and available to Workbench through the online *licensing server*.

NOTE:

The licensing server is the final license authority—the most current version of any Niagara host platform’s license is always stored there. In addition to accessing licenses via the licensing server when using the License Manager, operations in other Workbench views access the licensing server too. Examples include the Workbench **Local License Database** tool of Workbench, or the **Network License Summary** view of the “Licenses” slot of the NiagaraNetwork’s **ProvisioningNwExt**.

Providing that your PC *currently has Internet connectivity* while running a platform connection to any Niagara host, the **License Manager** lets you automatically retrieve and install any needed licenses.

Do this with the **Import** button, then selecting the license server option. As a side benefit, your “local license database” is also updated.

NOTE:

If sourcing from the license server while platform-connected to a host that has not yet been assigned a license by the server (or has a “pending” license), a license request form opens in your computer’s default browser, similar to that shown below.

Figure 47 License request form in browser (from Workbench, Tools > Request License)

This lets you submit a license request to the licensing server that includes the platform’s Niagara Host ID. In this dialog, be sure to enter your name, and email address.

If you already have been sent a “License Key”, note that a pending “unbound” license already exists on the licensing server. In this case, you can enter the license key along with the part number to activate that license, and make it immediately available.

Upon approval, the license file for the host is emailed back to the entered address. This license is typically in zipped format. At that point, it is also available for automatic retrieval using the corresponding “licensing server” operations from various views, such as the **License Manager**, **Workbench License Manager** view, and so forth.

Synchronizing with the Supervisor license database

The local Supervisor maintains a database that includes information about each host’s license. Periodically, it is a good idea to interrogate each host and update the Supervisor’s license database.

Prerequisites: You have a network of licensed hosts.

- Step 1 Expand the **ProvisioningNwExt** in the Nav tree to see its **Licenses** node.
- Step 2 Right-click **Licenses** and select **Views→Supervisor License Manager**.
The **Supervisor License Manager** window opens.
- Step 3 Click **Synchronize**.
The system prompts with the option to **Synch All Licenses?**
- Step 4 Click **Yes** and, at the **Synchronization Complete** prompt, click **OK**.

The Supervisor's license database contains the license identifier for each host in the network.

Updating host licenses

This procedure uses the **Niagara Network Job Builder** to create a one-time provisioning job that updates one or more host licenses in a network.

Prerequisites: The BatchJobService and ProvisioningNwExt components are available under your NiagaraNetwork.

Step 1 Double-click **ProvisioningNwExt**.

The system displays the **Niagara Network Job Builder** view.

Step 2 In the top **Initial steps to run only once** pane, click the **Add** button.

Step 3 In the **New Job Step** popup window, click the **Update Licenses** step and click **OK**.

Step 4 In the lower **Stations to include in the job** pane, click the **Add** button,

Step 5 In the **Add Device** popup window, click to select the stations and click **OK**.

Step 6 To initiate the provisioning job, review your choices and click the **Run Now** button at the bottom of the **Niagara Network Job Builder View**.

The view changes to the **Niagara Network Job View**, where steps and results appear as they are executed.

The licenses for all selected hosts are up-to-date. To make updating licenses a regular, automatic event, you need to create a job prototype.

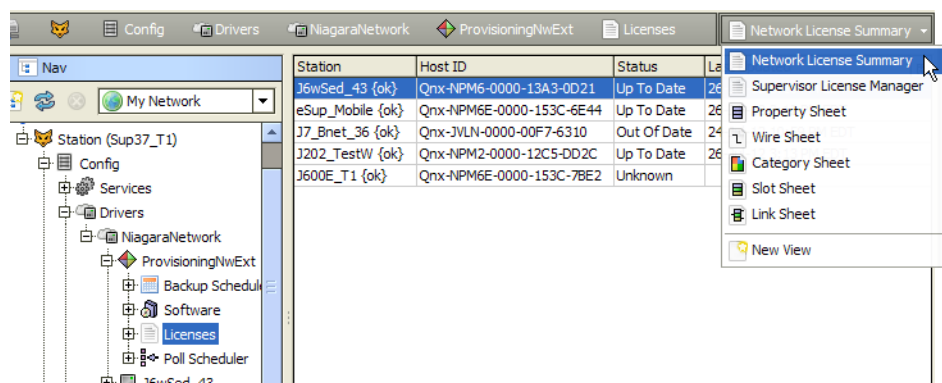
Updating licenses from the Network License Summary

Rather than create a one-time provisioning job, you can update the license on one or more remote hosts using the **Network License Summary**.

Prerequisites: You have synchronized the Supervisor's license database with the host controllers in your network, purchased a license upgrade for each host, and the upgrades are available on the online licensing server.

Step 1 Select the **Licenses** slot on the **ProvisioningNwExt**.

The system displays the **Network License Summary**.



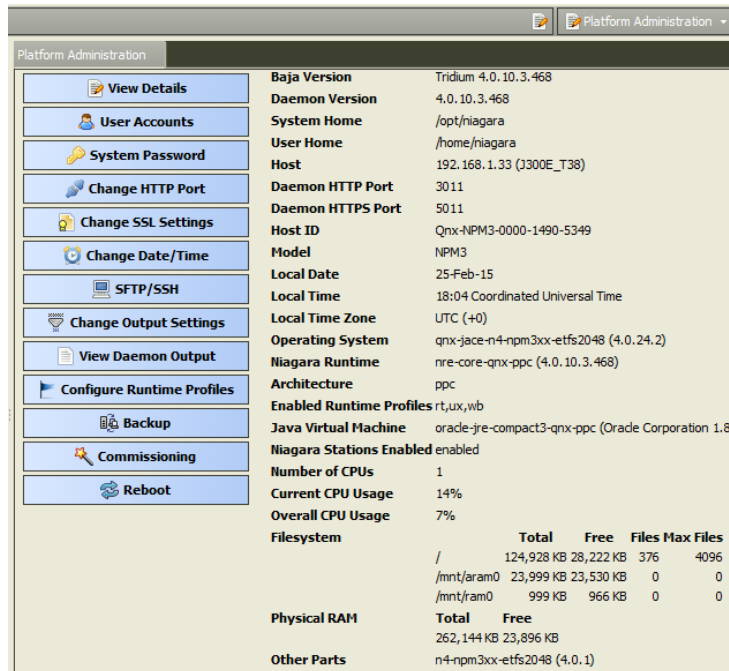
Step 2 Select one or more stations and click **Update**.

If a newer license is found (than that already installed), the system installs it in the remote host (s), updates the license(s) in the Supervisor's local license database, and resets the **Last Updated** timestamp to the time of the update.

Platform Administration

Platform Administration is one of several platform views. This view provides access to various platform daemon (and host) settings and summary information. As shown below, available functions appear as “buttons” on the left side, and summary information is listed in the right side. Typical use is when commissioning a new JACE, or when troubleshooting platform or host problems.

Figure 48 Platform Administration view



During a platform connection, upon first access to Platform Administration, a small delay occurs while downloading data about that platform’s installed modules. You may briefly see a “Loading Modules” dialog before the main view appears.

The following sections provide more details:

- [Types of Platform Administration functions, page 65](#) (summaries), with additional information as follows:
 - [“View Details” on page 52, page 66](#)
 - [“User Accounts” on page 53, page 66](#) (if JACE platform), or else [“Update Authentication” on page 54, page 68](#) (if Windows platform)
 - [“System Password” on page 56, page 70](#)
 - [“Change HTTP Port” on page 57, page 73](#)
 - [“Change SSL Settings” on page 57, page 73](#)
 - [“Change Date/Time” on page 58, page 75](#)
 - [“SFTP/SSH” on page 59, page 76](#)
 - [“Change Output Settings” on page 59, page 76](#)
 - [“View Daemon Output” on page 60, page 77](#)
 - [“Configure Runtime Profiles” on page 61, page 78](#)
 - [“Backup” on page 63, page 81](#)
 - [Commissioning, page 83](#)

- [“Reboot” on page 65, page 83](#)

NOTE:

Some functions vary by platform, see [About platform differences, page 16](#).

Types of Platform Administration functions

The following list summarizes platform administration functions, by button in the view:

- [View Details, page 66](#)
Provides platform summary data, available to the Windows clipboard. Includes all summary information shown in main **Platform Administration** view, plus installed modules, and so on.
- [User Accounts, page 66](#) (JACE platform) or [Update Authentication, page 68](#) (Windows platform)
For dialogs to change *platform login access* (user name and password).
- [System Password, page 70](#)
For a dialog to change the password used to encrypt client passwords in station database files (config. bog) and station backup .dist files. If a JACE-8000 platform, this is also used to encrypt data on its removable microSD flash drive, and when creating backup images to a USB flash drive.
- [Change HTTP Port, page 73](#)
For a dialog to change the HTTP port for the host’s platform daemon from (default) port 3011 to some other port.
- [Change SSL Settings, page 73](#)
For a dialog to enable/disable secure TLS connections to the host’s platform daemon, specify the TCP port used, plus other settings.
- [Change Date/Time, page 75](#)
For a dialog to change the hosts’s current date, time, and time zone, as used by that host’s OS.
- [SFTP/SSH, page 76](#)
(JACE controllers only) For a dialog to enable/disable both SFTP and SSH access to the JACE controller, or change the default port number shared by these protocols.
NOTE:
Enabling SFTP and SSH poses security risks. We strongly recommend you *keep this access disabled*, unless otherwise directed by Systems Engineering.
- [Change Output Settings, page 76](#)
Provides a dialog to change the log level of different processes that can appear in the platform daemon output.
- [View Daemon Output, page 77](#)
Provides a window in which you can observe debug messages from the platform daemon in real time, including the ability to pause.
- [Configure Runtime Profiles, page 78](#)
Provides a dialog to change the enabled runtime profile of the host. Used very infrequently.
- [Backup, page 81](#)
Make a complete backup of all configuration on the connected host platform, including all station files as well as other Niagara configuration.
- [Commissioning, page 83](#)
A way to launch the Commissioning Wizard (alternative to right-click on Platform in the Nav tree).

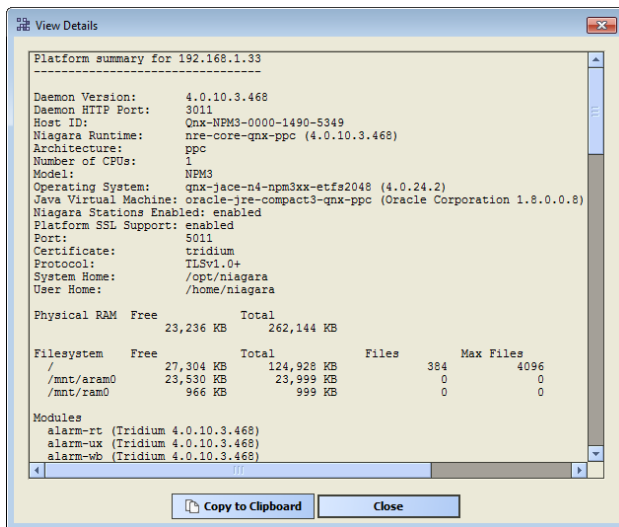
- [Reboot, page 83](#)

Provides a method to reboot a JACE platform, which restarts all software including the OS and JVM, then the platform daemon, then (if so configured in the **Application Director**) the installed station. If you click this, a confirmation dialog appears. If you answer yes, the JACE is rebooted and the platform connection drops.

View Details

This selection from the main **Platform Administration** view lists more platform information than shown in the main view.

Figure 49 View Details dialog in Platform Administration



Included in the **View Details** window is a listing of all installed modules, lexicons, licenses, and certificates. Included is a station line, listing configuration for autostart and autorestart, plus current status. Generally, information in this view is helpful when troubleshooting or asking for technical support. Buttons include:

- **Copy to Clipboard**

Puts all details in the dialog on your PC's Windows clipboard.

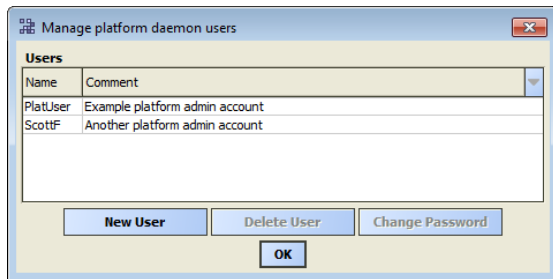
- **Close**

Exits the dialog, same as Windows close control (contents copied remain on clipboard).

User Accounts

This selection from the main **Platform Administration** view is available on Niagara 4 JACE controllers only. Unlike in NiagaraAX, the controller may have multiple platform administrator users (up to 20 maximum). All have the same full administrator permissions, can create additional users, and can change passwords of their own account.

Figure 50 Example where two platform admin accounts have been created

**NOTE:**

If commissioning a new unit, or a controller that has had a “cleanDist” file installed, only a well-known “default” platform admin account will exist. Any unit with the default platform admin user is extremely susceptible to unauthorized intrusion. Therefore, before you can complete other commissioning tasks, the N4 Commissioning Wizard requires you to first *replace the default platform user* account in a wizard step. For details see “Remove platform default user account” in the *JACE Niagara 4 Install & Startup Guide*.

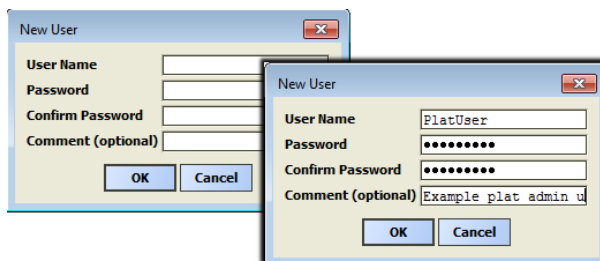
The following sections provide more details:

- [Platform admin account parameters, page 67](#)
- [Change notes for platform users, page 68](#)

Platform admin account parameters

Clicking **New User** produces the **New User** dialog, shown below.

Figure 51 Example New User dialog after typing user name, password, and comment



Parameters for adding new platform admin users in a controller are as follows:

- User Name

A maximum of 14 alphanumeric characters (a - z, A - Z, 0 - 9), where the first character must be alphabetic, and following characters either alphanumeric or underscore (_).

- Password

A *strong password* is required (must *match* in both password fields). The characters you enter display as asterisks (*).

Password must be a *minimum of 10 characters*, using:

- At least one *UPPER CASE* character.
- At least one *lower case* character.
- At least one *digit* (numeral).

An error popup reminds you if attempt to enter a password that does not meet minimum rules.

- Comment

(Optional) At most 64 alphanumeric characters, with these also allowed: - = + () @ . _

NOTE: At the time of this document, comment text cannot be re-edited after adding a user.

Some basic guidelines on strong passwords:

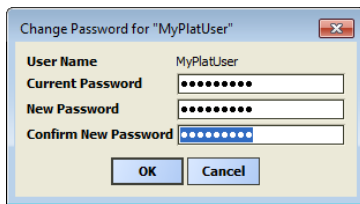
- Use both upper and lower case.
- Include numeric digits.
- Include special characters.
- Don't use dictionary words.
- Don't use company name.
- Don't make the same as the user name.
- Don't use common numbers like telephone, address, birthday, and so on.

Change notes for platform users

The following applies to password changes:

- Changing the *password* of any JACE platform user requires entering the current password, then entering the new password (twice) in the popup dialog. Again, a *strong password* is required.

Figure 52 Change Password dialog for platform admin user in



If you enter an incorrect current password, an “Invalid login credentials” popup error results, and after clicking **OK** you return back to the change password dialog above.

- If changing *your password* (used in your current platform session), your new credentials become immediately effective when you click **OK**. If you previously had “Remember these credentials,” selected in the Authentication login dialog, the cached credentials are automatically updated. For related details, see the “Credentials manager” section in the *User Guide*.

Other notes on changes to JACE platform users are as follows:

- Any platform user can *delete* any other platform user *except*:
 - The user active in the current platform session.
 - The original platform user, meaning the one created in the “Remove platform default user account” step when using the **Commissioning Wizard** to commission the JACE controller.

Such users cannot be selected to delete.

Update Authentication

For Niagara 4 platforms, this selection from the **Platform Administration** view is available on Windows-based hosts only. Use it to specify the Windows users group for platform administrator access.

NOTE:

This selection is also available in a platform connection to a NiagaraAX JACE, to change the credentials for the single (digest) platform user. It is *unavailable* in a platform connection to any AX Windows host.

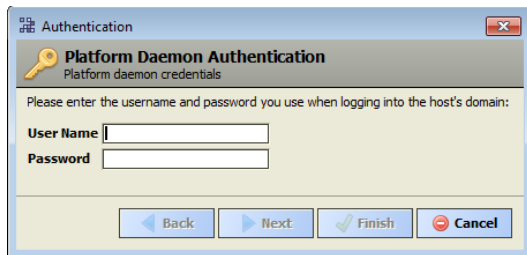
Unlike in NiagaraAX, authentication in Niagara 4 uses only basic (native Windows OS user based) authentication for Niagara platform access—“file/digest” platform access is no longer an option. Nor is there a

platform **User Manager** view for any Niagara 4 Windows host, as there was in NiagaraAX. Instead, you must use native Windows tools to create and manage Windows OS users and groups.

This theme also applies to the **TCP/IP Configuration** view for a Niagara 4 Windows host, which is available, but is *read-only* (allowing you to review current TCP/IP settings). Also unlike in NiagaraAX, there is only one level of platform access for any Niagara 4 host—“admin” level, which now applies to Windows platforms as well as JACE platforms.

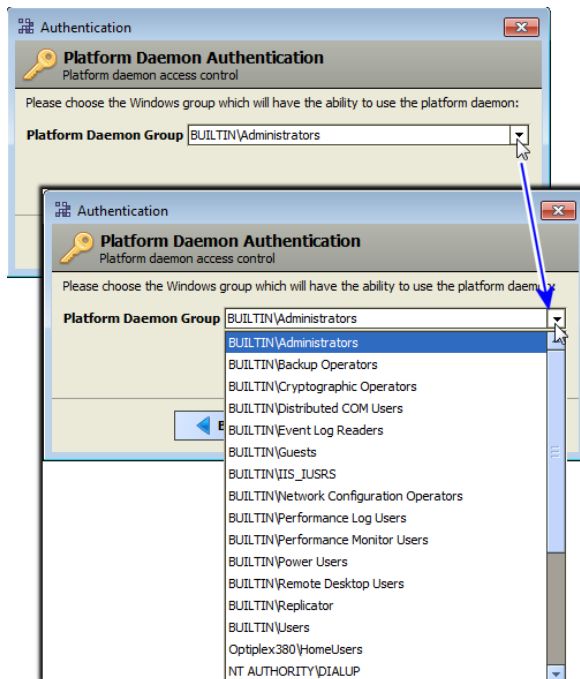
When you click **Update Authentication** on a Windows host, you see a login dialog, as below.

Figure 53 Login dialog for Update Authentication on Niagara 4 Windows platform



Use your standard Windows login credentials—if the host is on a Windows domain, login using the credentials you use when logging into that domain. This is necessary to limit the number of possible domain groups to only those groups in which you are a member. Such groups will be selectable in the next dialog to choose the sole Windows users group for platform daemon access, as shown in the figure below.

Figure 54 Windows platform daemon group selection dialog



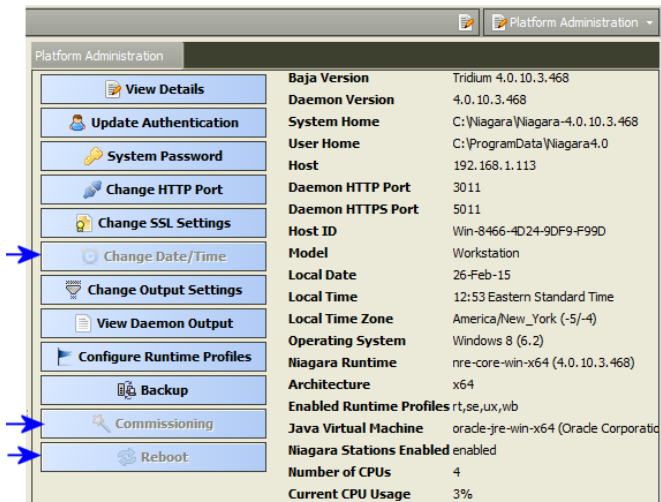
This dialog lets you select the one Windows users group that can make platform connections to this host. Groups include Windows “built-in” user groups (include “BUILTIN” or “NT AUTHORITY” prefix), as well as any locally-defined user groups. If the host has been added to a Windows *domain*, groups defined in that domain are also listed and available.

NOTE:

Domain groups are limited to only those in which the login user is a member.

When platform-connected to a Niagara 4 Windows host, some **Platform Administration** view buttons are unavailable, as shown in the figure below.

Figure 55 Platform Administration view in platform connection to remote Niagara 4 Windows host



As shown above, “Change Date/Time”, “Commissioning”, and “Reboot” are unavailable when platform connected to any Niagara 4 Windows host (remote or local). If a local platform connection, note that “Configure Runtime Profiles” is also unavailable.

Station access to Windows platform notes

Station (Fox) and/or HTTP access of a station running on a Niagara 4 Windows platform also prevents any date, time, or time zone host changes via the station’s **PlatformServices** (differing from NiagaraAX). In PlatformServices, this is reflected by the following:

- Properties of the PlatformServiceContainer for System Time, Date, and Time Zone are read-only, as are those same properties in the **System Date Time Editor** view on PlatformServices.
- The child NTP platform service (**NtpPlatformServiceWin32**) comes up as both disabled and read-only, and thus has no application.

Any necessary changes to these items you must make through the Windows OS on the Niagara 4 host. However, note access of a Niagara 4 station running on a QNX host (JACE) provides write ability to all such items, depending on the permissions of the user.

System Passphrase

All Niagara 4 platforms have a system passphrase (password), used to encrypt sensitive information, such as client passwords stored in BOG files and station databases (config.bog files) or station backup distribution (.dist) files. The passphrase increases security for the files that contain critical information. In various Workbench operations, you are prompted to enter the passphrase, such as when copying stations or restoring station backups in remote platforms.

The following areas of the framework are affected by passphrase implementation:

- Provisioning
- Distribution File Installer
- File Transfer Client
- Station Copier
- Back up
- Commissioning

- Export Tags

The sensitive information in files is protected with encryption, either by encrypting the information within the file or by encrypting the whole file. How encryption is applied depends on the expected portability of the file. Files located under the daemon User Home (files that “belong to the system”) are encrypted using a strong, randomly generated key that exists only on that system. Files located under a Workbench User Home (“portable” files that can be sent to many systems) are encrypted using a key derived from the user-defined system passphrase entered during software installation or when the system passphrase is changed.

Due to the different types of encryption that are used for the “*system*” or “*portable*” locations, when transferring files between the daemon User Home and another Workbench User Home you must use the Workbench platform tools (Station Copier, File Transfer Client or Backup) which convert files to use the correct encryption key for the target location.

CAUTION: Do not use Windows Explorer to copy files between the daemon User Home and other User Homes because without the proper encryption those files may not be readable.

- For system-to-portable transfers

You can get “portable” copies of files located under the daemon User Home by any of these methods:

- Make a backup from the **Platform Administration** view
- Make a backup from a running station
- Use either Station Copier or File Transfer Client from the **Platform Administration** view

The resulting local, portable copies and backup files are protected with a passphrase.

- For portable-to-system transfers

Alternately, when you use the Distribution File Installer to restore a backup .dist file, or Station Copier to transfer a station from your Workbench directory to a controller, the file’s passphrase is validated and used to translate the data back into the proper “system” encryption format for use under the daemon User Home.

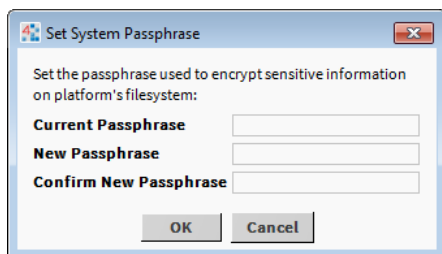
CAUTION: It is important to remember the system password and keep it safe. If you lose the system passphrase, you will lose access to encrypted data.

Updating the system passphrase

To change the system passphrase use either the **Commissioning Wizard** or the **Platform Administration** view as described here.

In the **Platform Administration** view, when you click **System Password** for any Niagara 4 platform, the dialog below appears.

Figure 56 Update System Password dialog



A strong password is required (must *match* in both password fields). The characters you enter are obscured. Password rules are the same as for JACE platform users, using a minimum of 10 characters, with:

- At least one *UPPER CASE* character.
- At least one *lower case* character.
- At least one *digit* (numeral).

An error popup reminds you if attempt to enter a password that does not meet minimum rules.

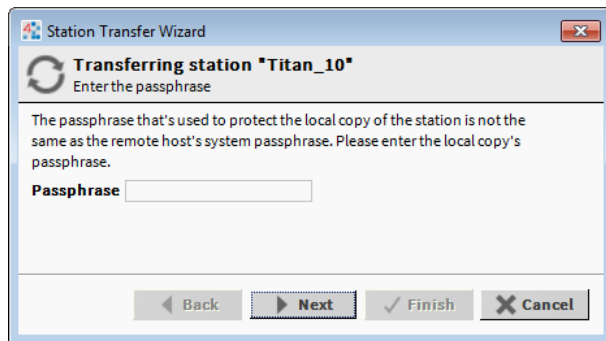
NOTE: It is important to remember the system password and keep it safe. If you lose the system passphrase, you will lose access to encrypted data.

System passphrase usage in backups and station copies

When using either the Distribution File Installer to restore a backup .dist file, or the Station Copier to transfer a local file, note the following:

- If the file passphrase and system passphrase are the same, the station copy proceeds without prompting for a passphrase.
- If the file passphrase is not the same as the target host system passphrase then you are prompted to enter the file passphrase, as shown.

Figure 57 Station Transfer Wizard prompt for bog file passphrase



NOTE:

- If you do not know the passphrase for a BOG file, you can edit it offline.
- If you do not know the passphrase for a .dist file you cannot install it

Editing BOG files offline

Files created in Workbench initially have no passphrase at all since the files do not yet contain sensitive data. You can add passphrase protection to offline BOG files by clicking the **Bog File Protection** icon in the toolbar.

If you change or add a passphrase value, attempts to **Save** will prompt you to enter the file passphrase. You can **Save** only if you enter the correct passphrase or add a new one.

If a BOG file is protected with an unknown passphrase, you can use the Workbench toolbar icon to unlock (force-remove) the passphrase, making the file unprotected, or “force-change” the passphrase to enter a new value. When you choose either of these options, *any sensitive data in the file is cleared*.

CAUTION: Be aware that unlocking (force-remove) and changing (force-change) the passphrase on a BOG file results in the loss of sensitive data in the file.

System passphrase usage in JACE-8000

The JACE-8000 makes additional use of its system passphrase, to encrypt sensitive information on its removable microSD flash drive, as well as when writing backup images to a USB flash drive. The passphrase is assigned as the file passphrase for portable copies of backups and station copies.

NOTE: The system passphrase default value is the same as the default platform password for controllers that you have just converted from NiagaraAX, and controllers on which you have just installed clean dist. On the first commissioning of such controllers you are prompted to change the passphrase from the default value.

Inserting a JACE-8000 SD card into a replacement unit

When inserting a JACE-8000 SD card into a replacement unit, note the following:

- If the replacement unit is preconfigured with the same system passphrase, the unit will start.
- If the replacement unit has a different system passphrase, the unit will not boot, and the status LED will flash with a 1-second period. To resolve, you must make a serial connection and when prompted, select either: **Update the system passphrase**, or **Remove all encrypted data**.

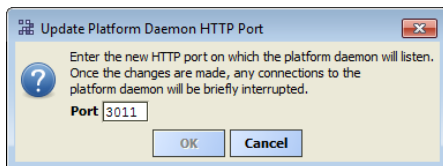
Change HTTP Port

This selection from the **Platform Administration** view lets you change the HTTP port monitored by the host's platform daemon for regular platform client connections (connections that are not secure). By default, port 3011 is monitored for such connections. This differs from any port used for station (Foxs) connections that are secure. For more details, see ["About a platform connection" on page 12, page 12](#).

CAUTION:

If there is a firewall on the host (or its network), *before changing* this port make sure that it will allow traffic to the new port.

Figure 58 Update Platform Daemon HTTP Port dialog



If needed, you can change the daemon monitored port to another HTTP port. You may choose to do this for specific firewall reasons, or perhaps for additional security. As shown in the figure above, you can type in the new port number in the Port field, which enables the **OK** button.

When you click **OK**, the platform daemon restarts, and your platform connection reopens (note this does not affect the operation of any running station). If previously connected on the port without security, the platform icon shows in the Nav tree with the new HTTP port number (:n) in parenthesis.

NOTE:

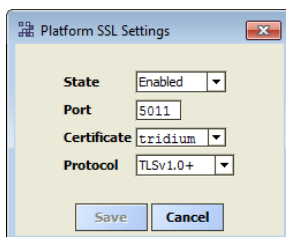
Before closing the host (removing it from the Nav tree), *carefully note* the new (non-default) port number you entered. You must specify that port number whenever reopening a platform using a connection that is not secure. You can check this port number in a station running on the host, by opening its Config, Services, **PlatformServices** property sheet.

Change TLS Settings

This selection from the **Platform Administration** view lets you configure for secure (TLS) platform connections, as well as change related secure platform connection (platformtls) parameters.

The figure below shows the dialog with default values.

Figure 59 Platform TLS Settings with default values (enabled)



Fields in this dialog are as follows:

- State

Either Disabled, Enabled, or Tls Only, to specify how Workbench clients can connect to this host's platform daemon.

- Disabled — Secure platform connections not possible (only regular platform connections).
- Enabled — Secure platform connections permitted, *as well as* regular platform connections.
- Tls Only — Only secure platform connections are allowed. Any attempt to connect without security goes unresolved (errors out).

This state is reflected among the properties listed on the main Platform Administration view, as "Platform TLS Support" state.

NOTE:

The "Tls Only" setting provides the best security. Note that in Niagara 4, all platforms support secure (TLS) platform connections, even if a freshly "clean disted" controller.

- Port

Software port monitored by the platform daemon for a *secure* platform connection, where port 5011 is the default. Note this is different than the default HTTP port (3011) for a regular platform connection that is not secure.

CAUTION:

Again, if there is a firewall on the host (or its network), *before changing* this port make sure that it will allow traffic to the new port.

- Certificate

The "alias" for the server certificate in the platform's "key store" to use for any platformtls connection. The default is the `tridium` self-signed certificate, which is automatically created when Niagara 4 is first loaded. If another certificate has been imported in the platform's key store, you can use the drop-down control to select it instead.

Certificates on the platform are managed via the platform **Certificate Management** view. For general information in this document, see *Station Security Guide*.

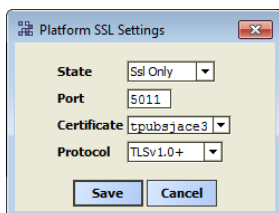
- Protocol

The minimum TLS protocol (Transport Layer Security) that the platform daemon's secure server will accept to negotiate with a client for a secure platform connection. During the handshake, the server and client agree on which protocol to use.

- TLSv1.0+ — (default) Includes TLS versions 1.0, 1.1, and 1.2, providing most flexibility.
- TLSv1.1+ — Only TLS versions 1.1 or 1.2 are accepted.
- TLSv1.2 — Only TLS versions 1.2 is accepted.

The figure below shows an example dialog for a controller enabled for platform TLS (only).

Figure 60 Example settings for a controller enabled for TLS, with a signed certificate



In this example, the controller uses a signed certificate with alias `tpubsjace3` (previously imported), with the port and protocol settings left at defaults.

Save notes on TLS platform settings

When you click **Save** after making any changes in Platform TLS Settings, those changes are immediately applied. Often this means your current platform connection *closes*, and then *reopens* in Workbench. For example if you change state from `Tls Only` to `Disabled`, your secure connection closes and opens again as a regular platform connection without security. Or, if while securely connected, you change the `Port` from (default) 5011 to another port number, your reopened platformtls connection uses this new port, shown in parentheses (nnnn) to indicate a port other than the default is being used.

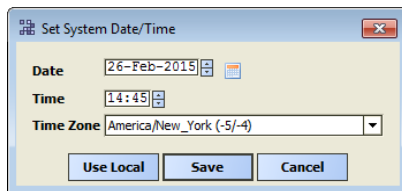
NOTE:

Before closing the host (removing it from the Nav tree), *carefully note* the new secure platform port number you entered. In the future you must specify that port number whenever making a secure connection to this platform.

Change Date/Time

This selection from the main **Platform Administration** view lets you change the date and time in the Niagara platform, as well as specify its time zone, as shown below. The **Save** button becomes available after you change one or more fields in the dialog, or when you click **“Use Local”**.

Figure 61 Set System Date/Time dialog



Upon **Save**, any change is processed by that host’s operating system.

The three dialog fields are as follows:

- **Date**
Click in a day-month-year position to select, then click up/down controls, or click and type in numerals directly, or click the calendar icon for a popup dialog to select the date from a calendar.
- **Time**
Always displays in 24-hour format. Click in a hour or minute position to select, then click up/down controls, or click and type in numerals directly.
- **Time Zone**
Provides a drop-down selection list of all available time zones in Java. Each time zone provides a text description, and in parenthesis the “hour offset” from UTC (and if daylight savings time is used) the “offset plus daylight savings.” For example: `America/New_York (-5, -4)`.

For related details, see [“Time Zones and Niagara 4” on page 129](#).

Use Local

Typically, if your Workbench PC’s current date/time setting are accurate, you click the **“Use Local”** button to synchronize the remote host’s date, time, and time zone with your Workbench PC. Upon **Save**, the remote host will have the identical settings.

NOTE:

To keep time synchronized across multiple Niagara platforms, configure the `NtpPlatformService` in the `PlatformServices` of the *station* running on each platform, as appropriate.

Advanced Settings

This **Platform Administration** view selection appears for Niagara 4 JACE platforms only, to enable, disable, or configure SFTP (Secure File Transfer Protocol) or SSH (Secure Shell Protocol) access. For Windows-based hosts, you typically use Windows “Remote Desktop Connections” instead.

NOTE:

This *replaces* an “FTP/Telnet” selection available for QNX-based platforms running NiagaraAX, which are both inherently less secure services.

This window contains two options:

- SFTP/SSH Enabled Port

As factory-shipped, a Niagara 4 controller has the SFTP and SSH service disabled — this may be best, especially if the platform is exposed to the public Internet. However, in some cases you may wish to temporarily enable the single port shared by these services, perhaps to facilitate debugging.

CAUTION:

Even SFTP and SSH pose security risks. Before enabling, we strongly recommend you configure for platform TLS only, and *keep this function disabled*, unless otherwise directed by Systems Engineering.

Note that SSH access to a controller provides “system shell” access, providing (after login using platform credentials) the same menu as “serial shell access” to its RS-232 port. For related details, see the “System shell” section in the *JACE Niagara 4 Install & Startup Guide*.

You can also change the TCP/IP port shared by these services from the “well-known” port to some other port. However, be sure that any firewalls being used on your network will allow traffic to that port.

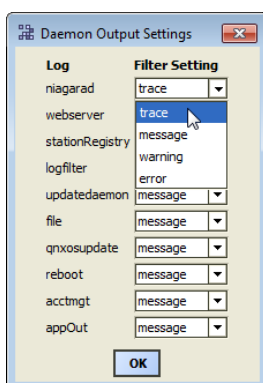
- Debug Enable checkbox

Enables a QNX webserver to accept incoming browser connections on :3011 or :5011.

Change Output Settings

This selection from the main **Platform Administration** view lets you “tune” the amount and content of the platform daemon output (see [“View Daemon Output” on page 60, page 77](#)). You can do this by changing the log filter settings of the various daemon processes. See [“Log filter settings” on page 60, page 77](#).

Figure 62 Daemon Output Settings dialog for JACE controller



By default, all daemon processes have a “Message” log filter level, and include the following:

- niagarad — Log for the platform daemon (niagarad) process, with “high level” entries like “niagarad starting”, “baja home = ...”, “niagarad stopping”.
- webservice — Log for HTTP server for incoming platform client connections. Entries are often generic, before the daemon hands off to the appropriate platform servlet.

- `stationregistry` — Log for platform daemon management of stations, including startup, shutdown, and watchdog actions.
- `logfilter` — Logs changes to daemon log states, meaning it tracks changes made in this dialog.
- `updatedaemon` — Log for handling Workbench requests for current platform daemon configuration, used mainly by Platform Administration view.
- `file` — Logs requests made to the platform daemon's file servlet, used in platform views like the File Transfer Client, Commissioning Wizard, Software Manager, Station Copier, and so on. Many different things can print on this log, like `"request for file xxx"`, or `"wrote file xxx"`.
- `qnxosupdate` — Log for the OS upgrade servlet created by the platform daemon. Workbench uses this servlet to upgrade the QNX OS in the host JACE when using the Commissioning Wizard or Distribution File Installer. Entries here can reflect a problem when updating the QNX OS, such as `"os crc isn't right"`, or `"waitpid when launching osupdate command failed"`.
- `reboot` — Log for the reboot servlet, one of the servlets the platform daemon manages.
- `appOut` — Log for the thread managing buffers associated with station output, making that output visible in the Application Director view. Entries may reflect buffer size changes (available in Application Director interface), or if a problem occurs streaming the output to Workbench.

Log filter settings

For any item, use the **Filter Setting** drop-down to select one of the following:

- Trace
Returns all message activity (*verbose*). This includes all transactional messages, which may result in too many messages to be useful. Be careful using Trace!
- Message
(Default) Returns informational "MESSAGE"s, plus all "ERROR" and "WARNING" types.
- Warning
Returns only "ERROR" and "WARNING" type messages (no informational "MESSAGE"s).
- Error
Returns only "ERROR" type messages (no "WARNING" or informational "MESSAGE"s).

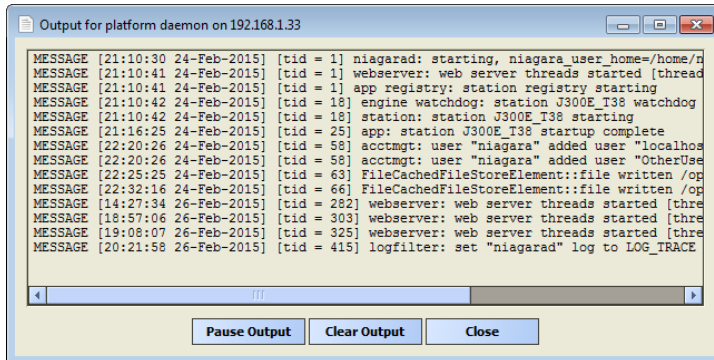
View Daemon Output

This selection from the main **Platform Administration** view lets you examine standard output from the host's *platform daemon* in *real time*. It is available for troubleshooting purposes.

NOTE:

Output is different from the output of a running *station*, as seen in the **Application Director**.

Figure 63 Example Output for platform daemon



Depending on the log filter settings set in platform administration's **Daemon Output Settings** dialog, the activity level in the output window will vary. Output is "non-modal," meaning that you can leave this window open and still do other Workbench operations (including change output settings).

As needed, use the scroll bars to navigate through messages, which will have headings "TRACE," "MESSAGE," "WARNING," or "ERROR," depending on message type. Each message includes a timestamp and a thread id number.

Use the Windows copy shortcut (CTRL + C) to copy text of interest to the Windows clipboard.

Click **Pause Output** to freeze the output from updating further (no longer in real time). When you do that, note that the button changes to **Load Output**. This means that daemon messages are still collected. When you click **Load Output**, the display loads the collected messages and continues again in real time.

Click **Clear Output** to clear all collected messages from the current daemon output window. This not a "destructive clear," as another (or new) daemon output window retains daemon messages.

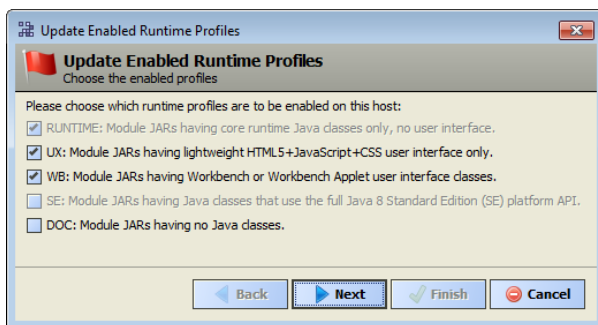
Configure Runtime Profiles

This selection from the **Platform Administration** view lets you globally change the "runtime profile" types for software modules on the connected platform. It is similar to the former "module filter" setting for NiagaraAX hosts, in that it affects the file space consumed by installed Niagara modules.

NOTE:

Typically, enabled runtime profiles are set *once* during initial JACE commissioning, then never changed.

Figure 64 Update Enabled Runtime Profiles dialog



The figure above shows typical settings for JACE controllers in the initial Niagara 4 release (N4.0).

- For any Windows-based host (providing it has hard drive for file storage), you typically want all runtime profiles enabled—including “DOC” if it hosts Workbench.
- For a JACE controller, with more limited flash-based file storage, in certain scenarios you may wish to change the enabled runtime profiles. Selection produces the dialog above.

NOTE:

For N4.0, the selection of UX will automatically includes WB, and vice-versa. This is likely to change in a future Niagara 4 release.

The following sections provide further details:

- [“Runtime profiles for Niagara 4 modules”, page 79](#)
- [“Results from a change in enabled runtime profiles”, page 80](#)

Runtime profiles for Niagara 4 modules

Software modules in Niagara 4 are distributed with a “runtime profile” type, designated by a suffix on the module’s JAR file name. In this “refactoring” of modules, many now often have *multiple* runtime profiles.

For example, the `alarm` module is distributed as three JARs: `alarm-rt`, `alarm-se`, and `alarm-wb`: each a separate `.jar` file. The runtime profile describes each JARs contents based on what systems are *able* to use them, where “`rt`” module JARs are a baseline among all Niagara 4 platforms.

This differs from NiagaraAX software modules, where a single module file (`alarm.jar` for example) holds all possible content. At installation time, AX platform tools can strip content from each JAR, based on the platform’s *module content filter*. In Niagara 4, each `.jar` file is digitally signed. This security measure ensures that the content cannot be changed at commissioning time. Thus, Niagara 4 has more software module JARs than NiagaraAX.

The following table lists the types of software module runtime profile types in Niagara 4.

Types of Runtime Profiles for Niagara 4 software modules. Only a very few modules do not have the `-rt` extension. One of those is `baja.jar`.

Runtime Profile	Example module name	Minimum JRE Version (1) Dependencies	Description
rt	alarm-rt	Java 8 compact3	Module JARs for data modeling and communications. These have core runtime Java classes only, with no user interface. This is the largest runtime profile group.
ux	webchart-ux	Java 8 compact3	Module JARs for BajaUX, any Java classes implementing lightweight HTML5, JavaScript, CSS user interface interaction, also theme modules.
wb	report-wb	Java 8 SE or Java 8 compact3	Module JARs with Java classes for Workbench or Workbench applet (WbApplet) user interface; views, field editors, widgets, and so on. Includes Hx and HTML5 Hx views.
se	test-se	Java 8 SE	Module JARs with Java classes that use the full Java 8 Standard Edition (SE) platform API. Currently, these can run on Windows-based hosts only.
doc	platformguide-doc	not applicable	Module JARs without Java code (classes), typically for documentation.

(1) JACE controllers use a “Java 8 compact3” compliant VM, whereas Windows-based hosts use the full Java 8 Standard Edition (SE) VM.

Currently, the runtime profile type `rt` is by far the most common of Niagara 4 software JARs. An inventory of module JAR files by type in one Beta build `!/modules` folder (4.0.11.0) yielded counts of:

- `*-rt`: 378
- `*-ux`: 17

- *-wb: 116
- *-se: 6
- *-doc: 20

Where the majority of modules with two runtime profiles had both "rt" and "wb", with only a few modules having three runtime profiles, as follows:

- alarm: rt, wb, se
- hierarchy: rt, ux, wb
- history: rt, ux, wb
- platCrypto: rt, se, wb
- search: rt, ux, wb
- seriesTransform: rt, ux, wb

NOTE:

Niagara 4 module refactoring was done for several reasons, notably:

- Niagara 4 software module files are *digitally-signed*, to improve security. This does not allow for module files to be installed with removed content during commissioning.

To simplify dependencies between different modules.

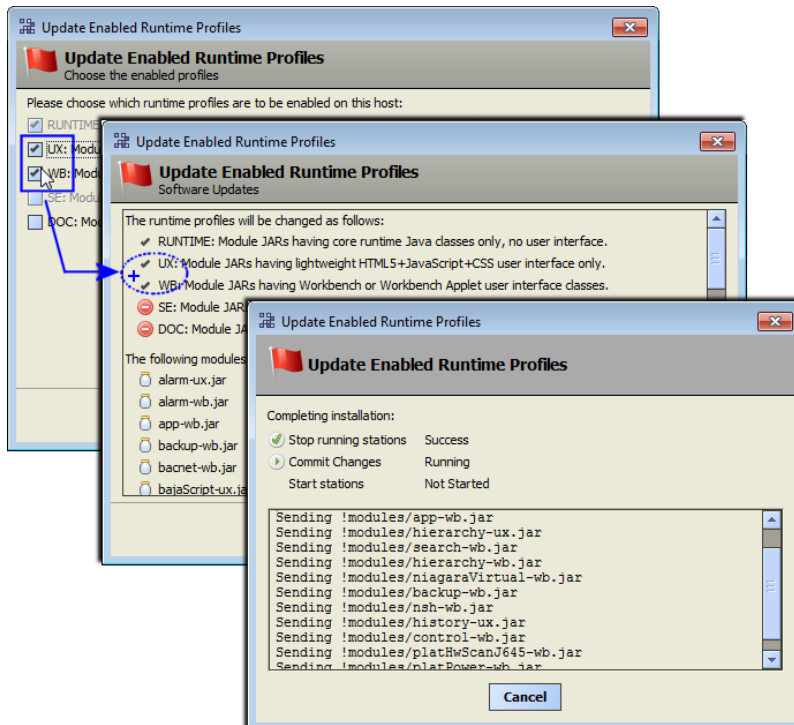
Results from a change in enabled runtime profiles

Depending on how you *change* enabled runtime profile, operations on the platform vary:

- If you *enable* additional profiles (say, go from "rt" only to "rt", "ux" and "wb" on a controller), additional modules will need to be *installed*—and these are listed in a confirmation window with a **Finish** button. Upon confirming, any running station is stopped, the modules are installed, and the station is restarted.
- If you *disable* currently enabled runtime profiles (say, from "rt", "ux" and "wb" to just "rt") some modules will need to be *uninstalled*, as they are no longer supported. Again, these modules are listed in a confirmation dialog with a Finish button. Upon confirming, any running station is stopped, the modules are uninstalled, and the station is restarted.

NOTE: Niagara 4 does not permit disabling currently enabled runtime profiles to only "rt" and "ux."

Figure 65 Dialog example after enabling more runtime profiles



Backup

This selection from the **Platform Administration** view performs a complete backup of the connected JACE, saved as a .dist file on your PC. The backup dist contains the entire station folder plus the specific NRE config used by that JACE platform, including license(s) and certificate(s). The dist also contains pointers to the appropriate NRE core, Java VM, modules, and OS. If ever needed, you restore a backup dist using the platform **Distribution File Installer** view.

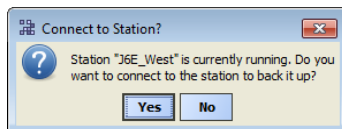
NOTE:

The backup dist file also contains the TCP/IP configuration of the host when it is backed up. When restoring the backup, you can select to restore these settings, or retain the TCP/IP settings currently in use by the target host. See ["Restoring a backup dist" on page 39, page 48.](#)

You can perform a backup with a station running on the target host, or when no station is running.

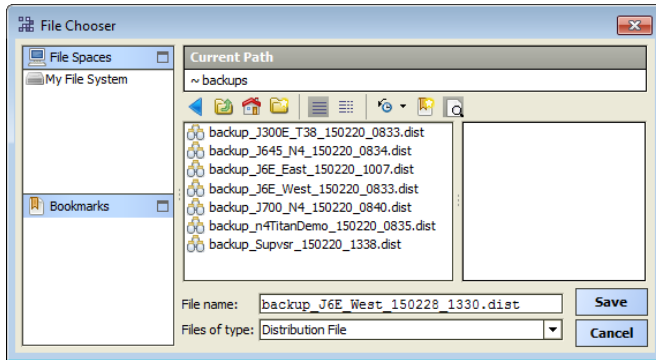
- If the JACE is running station, a confirmation dialog appears to connect to it, as shown below. This routine uses that station's **BackupService** to perform an "online backup." (If the station is not already open in Workbench, you must then logon as a station user.)
- If no station is running on the JACE, the platform daemon performs its own "offline backup."

Figure 66 Backup with station running, station connection



After station login and connection to the station (or if no station is running), the **File Chooser** appears, as shown below. Navigate to a target location to save the backup file, and to rename if desired.

Figure 67 File Chooser to select target folder and dist file name



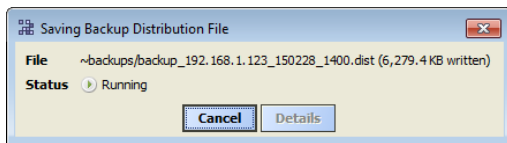
By default, the Backup function automatically creates (if not already present) a backups subdirectory under your Workbench User Home. The default file name for a backup file uses a format of: `backup_station-Name_YYMMDD_HHMM.dist`

For example, “`backup_J6E_West_150228_1330.dist`” for a backup made of station “`J6E_West`” on February 28, 2015 at 1:30 pm.

After you click **Save** the backup starts.

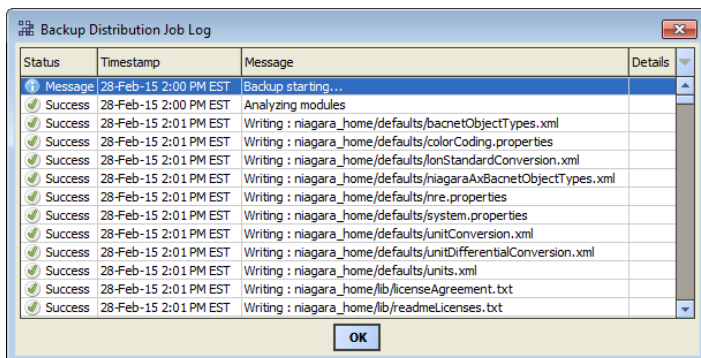
- If the station is running, a Fox Backup job is performed. A notification popup appears in the lower right of your display when the backup is done. This job is recorded in the station’s **BackupService** and visible in that component’s **BackupManager** view. Details are also available by accessing the job in the station’s **Job Service Manager**.
- If doing an “offline backup” (no station running), the platform daemon provides another progress dialog during the backup to a dist file, as shown below.

Figure 68 Backup from Platform Administration, no station running



Upon completion, you can click **Close** to return to the **Platform Administration** view, or click **Details** to see another popup with a log of actions performed in the backup, as shown below.

Available Details from backup using platform daemon (no station running)



Commissioning

This selection from the **Platform Administration** view launches the **Commissioning Wizard**, an ordered sequence of various platform steps. Included steps are platform views similar to a subset of those listed in [“Types of platform views” on page 14, page 15](#).

Typically, you use the **Commissioning Wizard** for the following:

- The *initial* Niagara 4 installation and startup of a JACE controller. For related details, see “About the Commissioning Wizard” in the *JACE Niagara 4 Install & Startup Guide*.
- To *upgrade* a JACE. For related details, see [“Upgrading a JACE” on page 42, page 52](#).

In Niagara 4, the **Commissioning Wizard** is intended for a remote JACE controller only. Note that this button is *unavailable* whenever you are connected to any Windows platform.

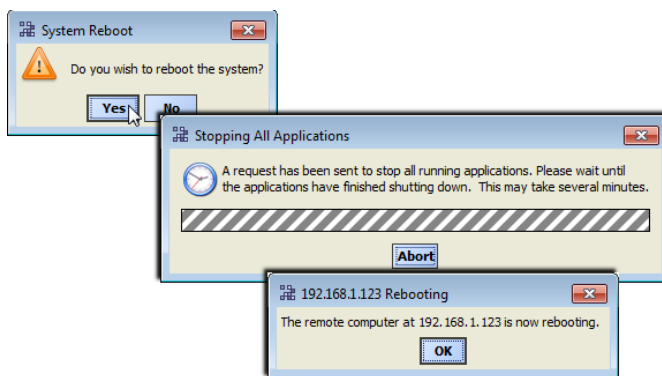
Reboot

This selection from the **Platform Administration** view *reboots the host* of a connected platform.

NOTE:

In Niagara 4, reboot is available only for remote JACE platforms. Reboot is *always unavailable* whenever you are connected to any Windows platform, either local or remote.

Figure 69 Reboot performs operating system reboot



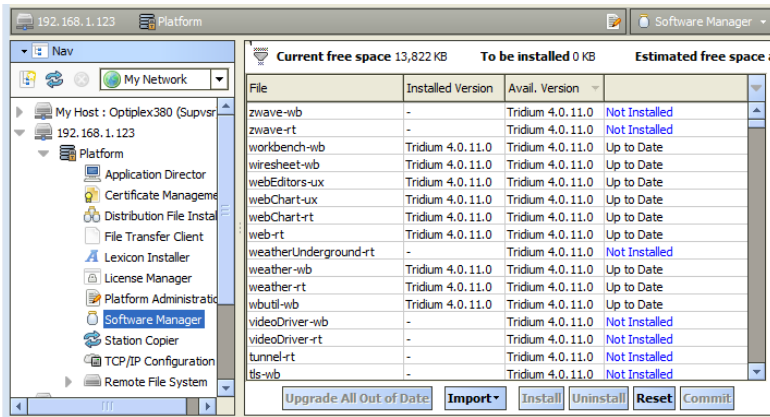
As shown above, a confirmation dialog appears, after which the daemon attempts to stop any running station before issuing the final reboot. A reboot restarts the QNX OS, Java VM, platform daemon, and finally the station (providing it is configured to “Auto-restart,” see [Application Director, page 28](#)).

When the platform reboots, your Workbench platform connection to it is dropped. Depending on the platform type, it may take from several seconds to a couple of minutes before you can connect again.

Software Manager

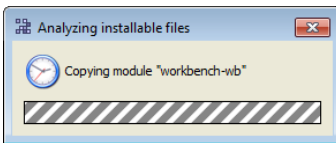
As shown in the figure below, the **Software Manager** is one of several platform views. This view lets you install, uninstall, or simply review all Niagara software modules installed in a remote JACE platform. By default, this view compares the platform’s modules against your “locally available” modules, meaning the most current Niagara modules in the software database on your Workbench PC.

Figure 70 Software Manager compares remotely installed modules to locally available



The *first time* you run the **Software Manager**, it copies modules from your **Sys Home** `!/modules` folder into a build-named subfolder in your “software database” (`!sw`), for example `!sw/4.0.11.0`. Note this can take several seconds, with a popup similar to the one below.

Figure 71 Copying modules into your software database

**NOTE:**

Copying also occurs whenever you “import” software into your Workbench’s software database.

Then every time you access the Software Manager it rebuilds the modules list, reflecting the latest revision of your available modules, as well modules currently installed in the opened Niagara platform.

The following sections explain further:

- [Software Manager notes, page 84](#)
- [About your software database, page 85](#)
- [Default module listing and layout, page 86](#)
- [Filtering displayed software, page 88](#)
- [Software actions, page 90](#)

Software Manager notes

The Niagara 4 **Software Manager** operates like it did in NiagaraAX, noting that Niagara 4 software module files now have separate “runtime profiles”. For related details, see [“Runtime profiles for Niagara 4 modules” on page 62, page 79](#). Apart from that difference, note the following still applies:

- Only software modules are shown, versus all “installable parts” including dist files, etc. Note that (as in later releases of NiagaraAX) “standard lexicons” are distributed in Niagara 4 builds as *modules*, named (by convention) as `niagaraLexiconLc-rt.jar` (where `Lc` is a two-character language code). For details, see the *Niagara Lexicon Guide*.
- Module statuses of “Out of Date” and “Not Installed” can include “Requires Commissioning” too, for example “Out of Date (Requires Commissioning)”. You cannot install such modules without first commissioning (upgrading) the JACE, using the **Commissioning Wizard**.
- In some cases you can install a new module or modules without rebooting the JACE, with its station kept running. This does not apply if upgrading (or downgrading) an existing module on the JACE.

- If needed, you can install an *earlier* Niagara 4 version of a module, versus its latest “Available” version—providing earlier versions are in your Workbench’s software database. See [“Right-click option to install earlier version” on page 72, page 92.](#)

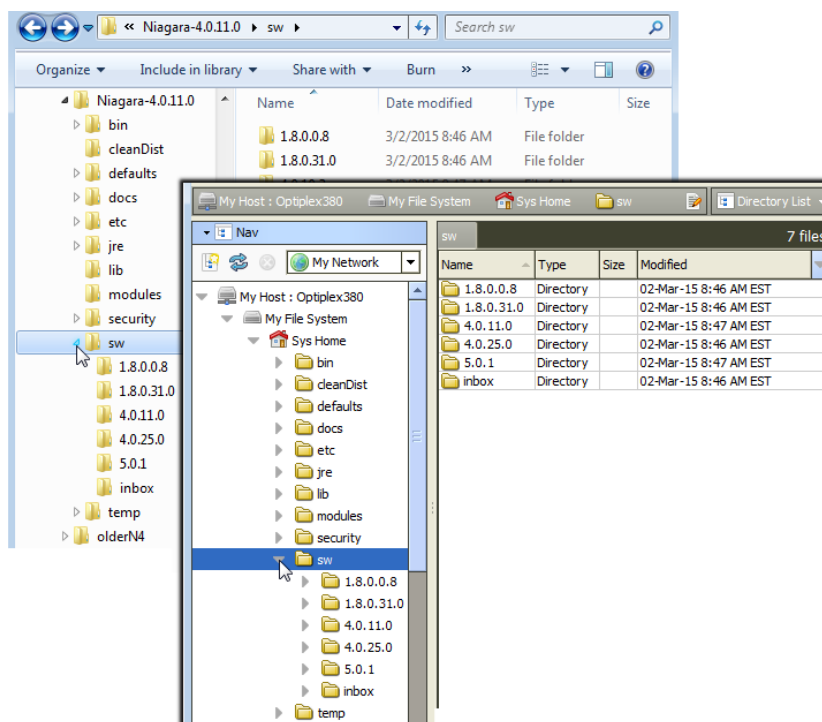
These changes are described and noted in other sections of this document, and are summarized here only to assist if you are already familiar with previous Workbench versions.

About your software database

The software database for your Niagara 4 Workbench is located under the **Sys Home** `sw` subdirectory. If Workbench was installed using the `use as an installation tool` option, this directory contains several subdirectories for various distribution (`.dist`) files, with each subdirectory named using version numbers.

You can see your `sw` subdirectory structure using either Windows Explorer, or in the Workbench Nav tree, **My File System, Sys Home** as shown below.

Figure 72 Software database is everything under `sw`



NOTE:

Numbers of subdirectories and version number names in your `sw` subdirectories will be different, this is only a simple example. Do not manually create or rename subdirectories in this area for proper operation—instead, let the **Software Manager** automatically administer this database.

Using the [Figure 72, page 85](#) example of a Niagara 4.0 installation (4.0.11.0), this `sw` software database has several versioned subdirectories, which are described in this example as follows:

- `1.8.0.0.8` — Reflects the version of dist files for the Oracle Java 8 “compact3” JRE for JACE controllers (2 files, one for PPC processor JACE controllers, one for ARM processor JACE-8000).
- `1.8.0.31.0` — Reflects the version of dist files for the Oracle Java 8 “Standard Edition” JRE for Windows platforms (2 files, one for 64-bit Windows, one for 32-bit Windows).
- `4.0.11.0` — Reflects the current *Niagara release, by build number*. Contains numerous Niagara `nre` “config” and “core” dist files, installed by the “installation tool” Workbench installation option. Also, after the **Software Manager** is first used, the contents of the build’s modules directory (module .jars) are automatically copied here too.

- 4.0.25.0 — Reflects version of dist files for QNX operating system for JACE controllers, with 4 different dist files.
- 5.0.1 — Reflects a version of a few “prototype” Workbench help modules.
- `inbox` — Provides a means for you to copy any installable file here, and have the **Software Manager** automatically create a proper “versioned” subdirectory for it. Or, if the correct subdirectory already exists, the Software Manager will copy the inbox file(s) there.

As an equivalent to the inbox feature, you can use the **Import** button at the bottom of the **Software Manager** to add to your Workbench software database. For details, see [“Software Import” on page 70, page 89](#).

When you add different-versioned installable files, the number of different subdirectories under your `sw` directory will continue to increase. By default, the Software Manager displays only the most recent version of any module as the **Avail. Version**.

NOTE:

You can select to install an *older version* of any module listed in the Software Manager, if available in your software database. See [“Right-click option to install earlier version” on page 72, page 92](#).

Note that older software files (modules, dists) are also useful in your software database when restoring a backup dist for a JACE, if the backup was made using a previous software release. You use the platform **Distribution File Installer** to restore a backup.

Default module listing and layout

By default, the **Software Manager** lists all the JACE’s *out-of-date* modules at the *top* of the table, then *uninstalled* modules, and lastly *up-to-date* modules (sorted alphabetically); see the figure below.

Figure 73 Software Manager default listing out-of-date, then uninstalled modules

File	Installed Version	Avail. Version	Status
app-rt	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date
app-wb	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date
axvelocity-rt	-	Tridium 4.0.11.0	Not Installed
axvelocity-wb	-	Tridium 4.0.11.0	Not Installed
backup-rt	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date
backup-wb	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date
bacnet-rt	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date
bacnet-wb	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date
bacnetAws-rt	-	Tridium 4.0.11.0	Not Installed
bacnetAws-wb	-	Tridium 4.0.11.0	Not Installed
bacnetMigrator-wb	-	Tridium 4.0.11.0	Not Installed
bacnetOwls-rt	-	Tridium 4.0.11.0	Not Installed
bacnetOwls-wb	-	Tridium 4.0.11.0	Not Installed
bacnetUtil-rt	-	Tridium 4.0.11.0	Not Installed
baja	Tridium 4.0.10.0	Tridium 4.0.11.0	Out of Date

- **Out of Date** modules are older than what you have in your PC software database.
- **Not Installed** modules do not exist on the platform, but are in your PC software database.
- **Up to Date** modules are the same (or possibly newer) than that in your PC software database.

NOTE:

Both “out of date” and “not installed” modules may also show a “Requires Commissioning” status. This indicates you must upgrade the JACE first, before installing that module version. For more details, see status descriptions for **Software Manager** table columns below.

As needed, you can scroll down the table or click on headers of table columns to resort alphabetically.

Software Manager table columns

The **Software Manager** lists modules using four columns, from left-to-right labeled as follows:

- File — File name of locally available module file, or blank if the module is on the remote host only.
- Installed Version — Version of the module installed in the remote host, or blank if not installed.
- Avail. Version — Latest version of locally available module, or blank if the software is on the remote host only.
- <unlabeled> — *Status* of the module in the remote JACE platform. For each module, status is one of the following:
 - Not Installed — Module is not in remote platform, but is available locally.
Blue text is used for this status.
 - Not Installed (Requires Commissioning)— Module is not in remote platform, but is available locally.
Blue text is also used for this status.
Dependencies prevent you from installing it, unless you first upgrade the JACE, using the Commissioning Wizard. See [“Upgrading a JACE” on page 42, page 52.](#)
 - Up to Date — Module is installed in the remote platform, and is equal to (or higher) than locally available module version.
 - Out of Date — Module is installed in remote platform, and is *older* than your local version.
Red text is used for this status.
 - Out of Date (Requires Commissioning)— Module is installed in remote platform, and is *older* than your local version shown. Red text is also used for this status.
Dependencies prevent you from installing it, unless you first upgrade the JACE, using the Commissioning Wizard. See [“Upgrading a JACE” on page 42, page 52.](#)
 - Not Available Locally — Module installed in remote platform is not in your software database.
 - Cannot Install — Local module is unreadable or has a bad manifest; you cannot install it.
 - Bad Target — Remotely installed module is unreadable or has a bad manifest, and is therefore unusable by a station. Software in this state should probably be fixed, since it could cause the station to not work correctly.
 - Downgrade to <version> — Remotely installed software is intended to be replaced with a module having a lower version.
 - Install <version> — Module is intended to be installed; it does not currently exist on the remote platform.
 - Re-Install <version> — Remotely installed module is intended to be replaced with a module having a the same version.
 - Uninstall <version> — Remotely installed module is intended to be uninstalled.
 - Upgrade to <version> — Remotely installed module is intended to be replaced with a module having a higher version.

NOTE:

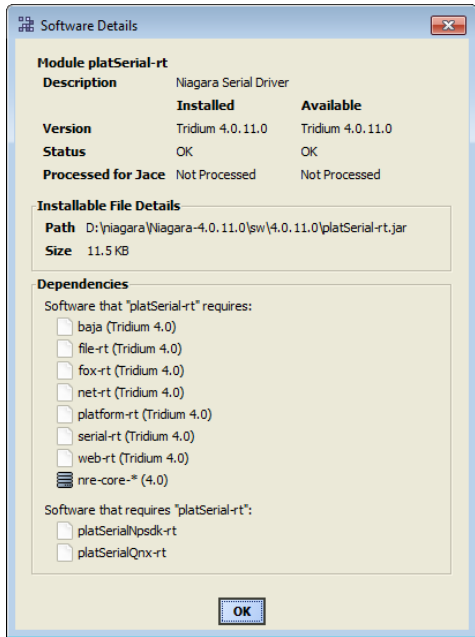
“Intended” status values like “Install <version>” reflect *un-committed* actions made during your Software Manager session. Blue text is used to list these statuses.

You can also view *software details* about any item in the table. In addition, you can filter (reduce) the number of software items listed, based on text included in file name or the softwares’ status values. See [“Filtering displayed software” on page 69, page 88](#) for more details.

Software Details

From the **Software Manager**, double-click any module to see a popup dialog with details.

Figure 74 Software Details dialog from Software Manager



Details include a brief module description, comparisons between installed and available module, module file and size, and whatever module *dependencies* exist, by part names. Dependencies are listed for both cases: what software is required *by* this module, plus software that is dependent *on* this module.

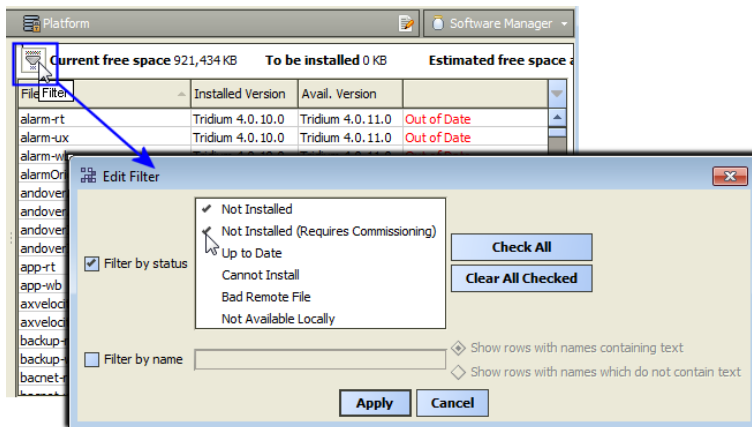
NOTE:

Essentially, dependency details are for information only. When installing modules from the Software Manager, all dependent modules are *automatically* included when you select a module to install.

Filtering displayed software

By default, the **Software Manager** lists all remotely installed and locally available modules, which can produce a very large table. A filter control provides an **Edit Filter** dialog, in which you select items for listing, thereby filtering undesired items. See the following figure.

Figure 75 Filter control and dialog to limit displayed modules



You can use either “Filter by status” or “Filter by name”, or a combination of the two.

Filter by status

Modules with an “Out of Date” or “Out of Date (Requires Commissioning)” status always appear in the **Software Manager**. So do any with uncommitted (intended) status values, such as “Install,” “Uninstall,” and so on.

When you enable filter by status, you can *check* other statuses *to include* (or clear to omit) the listing of associated items in the table, as follows:

- Not Installed — Modules on your PC that can be installed, but are not in the remote platform.
- Not Installed (Requires Commissioning) — Modules on your PC, but not in the remote platform. The remote JACE must be upgraded (using **Commissioning Wizard**) first.
- Up to Date — Modules on your PC *and* in the remote platform, where the software is not older.
- Cannot Install — Local module is unreadable or has bad manifest, you cannot install it.
- Bad File — Remote module is unreadable or has bad manifest.

NOTE:

With status filtering enabled, you can also simply “check all” and “clear all checked.”

- If all status items are cleared, only “Out of Date” and uncommitted status modules appear.
- If all status items are checked, the display is similar to disabled status filtering, except “non-module” items are not listed.

Filter by name

Name filtering lets you include *or* exclude items based on character string portion of module File name. When enabled (checked), you can *type* in a string of characters, and then check one of the following:

- Show rows with names containing text — Only items with file name containing this string.
- Show rows with names which do not contain text — Only items with file name that does not contain this string.

This feature can be useful to filter many modules with common name characters, for example “lon” or “doc” part-named modules.

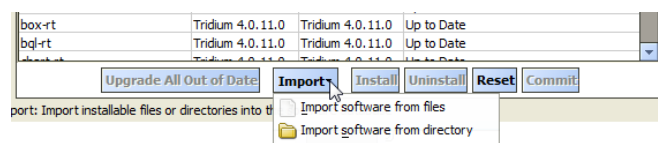
Software Import

As shown below, an **Import** button at the bottom of the **Software Manager** provides two menu choices for you to add new (or earlier) installable software files (module .jars, .dists) in your software database.

NOTE:

Also see the next section, [“Import vs. copy into modules”, page 90.](#)

Figure 76 Import choices to bring in file(s) or entire folders



The two import options are:

- Import software from files

This produces the standard **File Chooser** dialog, in which you navigate to the proper location and select one or more software files for import.

- Import software from directory

This produces the standard **Directory Chooser** dialog, in which you navigate to the proper location and select a directory, for inclusion of any contained software files. For example, you might do this for an *earlier* installed build of Niagara, selecting its “sw” folder, or a portion thereof.

Upon import, the software list is again rebuilt by the Software Manager (popup dialogs appear while software files are copied). Afterwards, any modules that are newer-versioned, or that did not previously exist, will now be represented by default in the software table.

If imported modules are *earlier* versions, they are also available for installation in the Software Manager. See [“Right-click option to install earlier version” on page 72, page 92.](#)

Import vs. copy into modules

When receiving updated or new *module* jar files (say sent from Systems Engineering or downloaded from Niagara Central), you have two basic options when copying them to your Workbench PC, as follows:

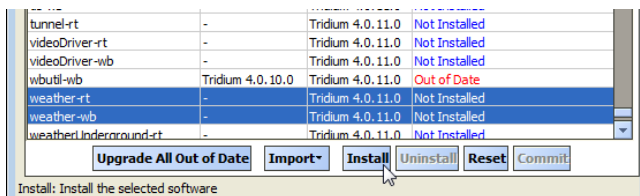
1. Copy directly into your `!/modules` directory. This makes the module(s) available in your Workbench environment, and also available to install in other remote platforms (when the installer runs, the module(s) are also copied into your *software database*, available for installation). This is the typical choice.
2. Copy into your `!/sw/inbox` directory (or, use the equivalent software “**Import**” feature in the **Software Manager**). In this case, the module(s) are *not* used in your Workbench environment, but *are* available in your software database for installation in remote platforms.

This would be the choice where you want to keep using a newer (or older) version of the received module(s) in your Workbench environment. A scenario that fits here, is if you received *older* versions of modules, perhaps needed to restore an older backup dist file in a certain remote platform.

Software actions

As needed, from the **Software Manager** you can take actions on modules, such as install, uninstall, upgrade, downgrade, and re-install. You flag intended actions on software items using *action buttons* near the bottom of the manager’s view pane, as shown below. Action buttons become enabled when you have one or more items selected.

Figure 77 Software Manager action buttons



Included in action buttons are **Reset** and **Commit**. When you reset, all flagged module changes (since the last commit) are cleared. Commit is how you actually *launch* the flagged changes.

When you **Commit**, one of these two things happens:

- If upgrading (or downgrading) modules, a confirmation popup dialog appears, telling you the station must be stopped and the host rebooted. After the software operation completes, the host is rebooted.

NOTE:

Before committing, make sure that controlled equipment that might be adversely affected by the JACE’s station stopping and then host rebooting (from software changes) is put in a manually controlled state.

- In many cases, if only installing *new* module(s), meaning modules not previously installed, the station continues running on that platform. The software is immediately installed.

The following action buttons are explained in further detail:

- [Upgrading out-of-date modules, page 91](#)

- [Install, page 91](#)
- [Uninstall, page 91](#)
- [Re-Install, Upgrade, Downgrade, page 92](#)
- [Commit and Reset, page 92](#)

NOTE:

Also see ["Right-click option to install earlier version" on page 72, page 92.](#)

Upgrading out-of-date modules

Whenever one or more local modules are newer than in the modules in an opened platform, the **Software Manager** enables an **Upgrade All Out of Date** button. This allows you to flag *all* out-of-date modules to be upgraded. Unlike other action buttons, specific item(s) do not need selection first.

The platform is configured and running.

Step 1 Open a secure platform connection to a target controller.

Step 2 Expand the **Platform** container in the Nav tree and double-click the **Software Manager** container.

The **Software Manager** compares the modules on the Supervisor platform with the modules in each open platform. If a module on the Supervisor side is newer than its equivalent on the controller side, the **Software Manager** enables the **Upgrade All Out of Date** button.

Step 3 Click the **Upgrade All Out of Date** button.

The status of all out-of-date modules changes to `Upgrade to version`, where `version` is the latest version available.

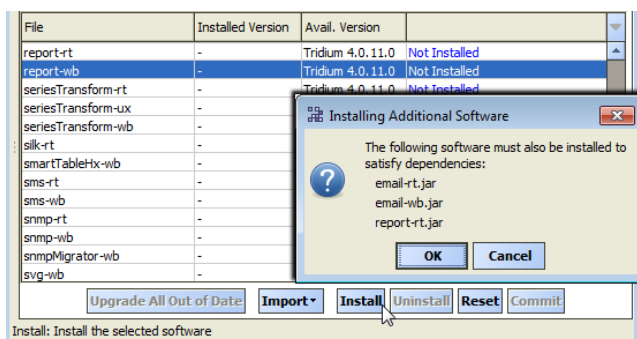
Install

This button is available in the **Software Manager** when you have one or more modules selected with a status of "Not Installed." When you click it, the status of the selected modules changes to "Install <version>," and if selected again, the button changes to **Cancel Install**.

NOTE:

If a selected module has *dependencies* on modules not already installed (or also flagged to install), a dialog appears explaining additional software is needed, as shown below. After you click **OK** from this dialog, the additional modules are flagged, the status of all affected modules changes to "Install <version>".

Figure 78 Installing Additional Software dialog

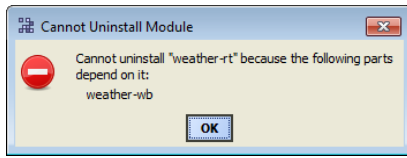
**Uninstall**

This button is available in the **Software Manager** when you have one or more *installed* modules selected (status of either "Up to Date" or "Out of Date"). If the selected module(s) are not dependencies of other installed modules, when you click **Uninstall** the module(s) status changes to "Uninstall <version>," and the button changes to **Cancel Uninstall**.

NOTE:

If other installed modules have *dependencies* on one or more modules you selected, a dialog appears explaining the uninstall cannot occur, as shown below. You can then decide if you want to reflag another uninstall, selecting also all modules that are dependent.

Figure 79 Cannot Uninstall dialog



Re-Install, Upgrade, Downgrade

In the **Software Manager**, when you have one or more *installed* software items selected, the “install” button changes to show one of these options.

- **Re-Install** appears if the installed item is the same version as your locally available one.
- **Upgrade** appears if the installed item is an earlier version than your locally available one.
- **Downgrade** appears if the installed item is an newer version than your locally available one.

When you click this button, the software’s status correspondingly changes to either “Re-Install <version>”, “Upgrade <version>”, or “Downgrade <version>”, and the button changes to **Cancel <action>**, for example: **Cancel Re-Install**.

Commit and Reset

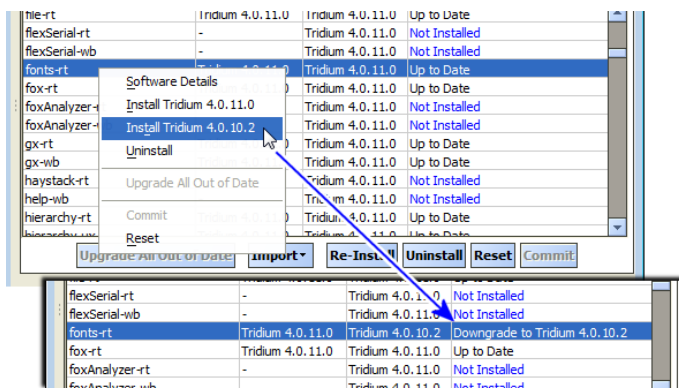
In the Software Manager, when you have one or more *pending* actions in place on software items, the **Commit** button is available. This is how you initiate the software action.

At any time before you commit, you can also click the **Reset** button. This removes all pending actions in place on software items, and makes the **Commit** button unavailable again.

Right-click option to install earlier version

In addition to button-based software actions in the Software Manager, you can also select an *earlier* version of a module to install, providing one is in your Workbench’s *software database*.

Figure 80 Right-click option to install earlier module version in Software Manager



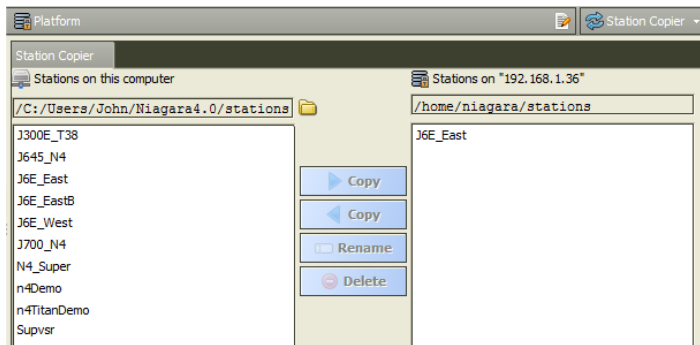
Simply right-click a module row, and from the shortcut menu select any “Install →vendor→ 4.→n→.→nn” items as shown above. Note if a “downgrade”, a host reboot/station restart will result after you commit.

Station Copier

The **Station Copier** is one of several platform views. In Niagara 4, you use it to install a station in any Niagara 4 platform (remote or local), as well as make a copy in your Workbench **User Home** of any running station (remote or local). You can also rename and delete stations, either locally or remotely.

You see this view even when opening a local platform connection at your Supervisor computer as well as when opening a remote Niagara host. The following figure shows the **Station Copier** in a platform connection to a controller.

Figure 81 Example Station Copier view for remote JACE platform



As shown above, the **Station Copier** view is split into two main areas:

- Stations on your Workbench PC, typically your *Workbench* User Home (left side)
- Station in the *daemon* User Home of the opened platform (right side)

By default, contents of your Workbench User Home stations folder is shown on the (left) side. If you have station folders located elsewhere, click the folder icon for a **Change Directory** window, and point the **Station Copier** there. That changed location is used the next time you access the **Station Copier**.

The following sections provide more details:

- ["Station copy direction", page 94](#)
- ["Station Copier dependencies check", page 96](#)
- ["Station Transfer Wizard", page 96](#)
- ["Renaming stations", page 103](#)
- ["Deleting stations" on page 81, page 103](#)

Installing a station

Station installation usually involves copying an existing station from a Workbench user home to a target controller platform. The procedure uses the **Station Transfer Wizard**.

Prerequisites: The following conditions have been met:

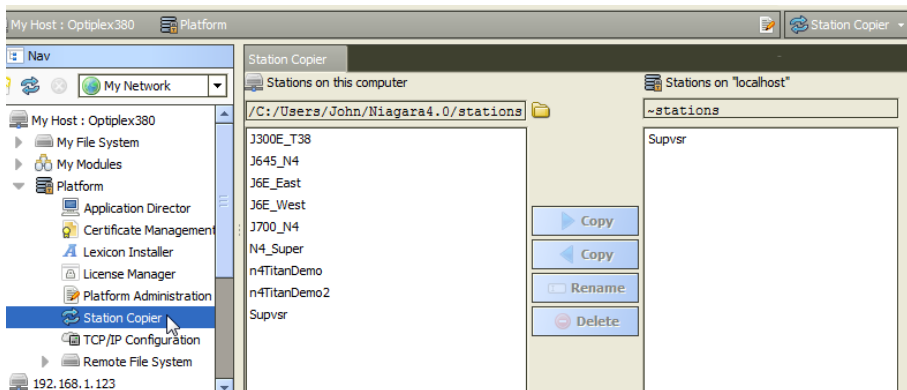
- All hardware (PC or controller) has been installed and connected.
- The controller has been commissioned and network communication configured.
- The station you wish to install exists in a Workbench user home.
- You are prepared to answer the **Station Transfer Wizard** questions.

This topic is part of configuring a station for the first time or upgrading a station from a previous version of Niagara. The Station Copier is a platform service.

Step 1 Open a secure platform connection to the target controller, the one on which you wish to install the station.

Step 2 Expand the **Platform** container in the Nav tree and double-click the **System Copier**.

Figure 82 Example Station Copier view at local Supervisor host



Step 3 On the left side (Workbench user home), select the station and click the **Copy** button that points to the right side of the window.

The Station Copier displays “Loading module information” and, if all needed modules are available, launches the **Station Transfer Wizard**.

If any module needed by the station has a dependency that requires the controller (platform) to be commissioned (commissioning upgrades core software or the operating system), the station installation stops immediately, displays a message, and provides the option to start the **Commissioning Wizard** instead. The need to commission a controller arises if you are installing a brand new controller or upgrading a controller from NiagaraAX to Niagara 4.0 and forgot to run the **Commissioning Wizard**.

Step 4 Follow the wizard prompts clicking **Next** or **Back** as necessary.

If the station is running, the wizard informs that it will stop and restart the station.

Step 5 Review all the changes you selected and click **Finish**.

The wizard displays installation progress in the **Transferring station** window.

Step 6 To complete the operation, click **Close**.

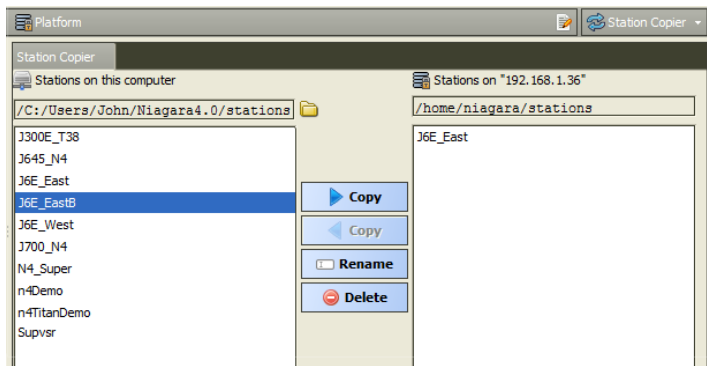
The Station Copier prompts you to open the Application Director now. **what is the benefit of viewing the Application Director now?**

Step 7 To view the station log, click **Yes**.

Station copy direction

The copier works in either direction. In other words, click a station on one side (to copy to the other side). When you click a station, the station is selected (highlighted) and the appropriate **Copy** button, by direction, becomes enabled to clarify the source and target. See the figure below.

Figure 83 Copy direction by station side selection



To perform the following station operations, you:

- Click in left side for a copy from Workbench User Home-to-daemon User Home.
Do this to install a station in a JACE. This is called “installing” in remaining document subsections.
- Click in right side for a copy from daemon User Home-to-Workbench User Home.
Do this to make a local backup copy of a station, saved to your Workbench computer. This is described as a “backup” in the following document subsections.

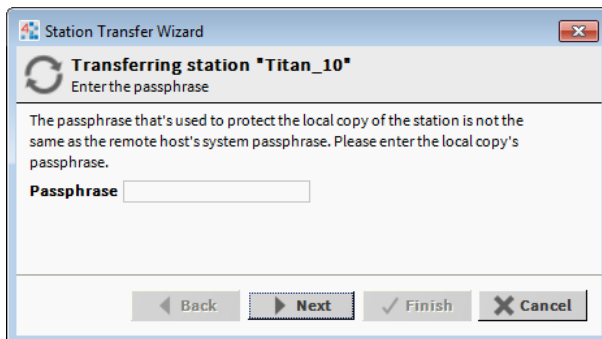
When you click **Copy**, the **Station Transfer Wizard** appears and guides you through the steps of the station transfer process.

Station Copier Passphrase check

When you click **Copy**, the **Station Transfer Wizard** attempts to validate the Bog file’s passphrase with the target host’s system passphrase. If they are the same, then it guides you through the rest of the station transfer process.

If the file passphrase is not the same as the target host system passphrase then you are prompted to enter the file passphrase, as shown.

Figure 84 Station Transfer Wizard prompt for bog file passphrase



If a BOG file is protected with an unknown passphrase, you can use the Workbench toolbar icon to unlock (force-remove) the passphrase, making the file unprotected, or “force-change” the passphrase to enter a new value. When you choose either of these options, *any sensitive data in the file is cleared*.

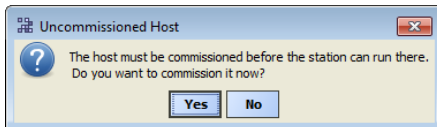
CAUTION: Be aware that unlocking (force-remove) and changing (force-change) the passphrase on a BOG file results in the loss of sensitive data in the file.

Station Copier dependencies check

The **Station Copier** checks, whenever installing a station, to determine if the target JACE platform does not already have all modules installed that are required by that station. Such dependencies may prevent the installation of a selected station. Changes are summarized as follows:

If any module needed by the station has a dependency that requires the JACE to be commissioned (upgrade core Niagara software or QNX OS), the station install immediately stops, upon station selection. Steps in the Station Transfer Wizard *do not* appear. A dialog explains the JACE needs commissioning, and provides the option to start the **Commissioning Wizard**. See figure below.

Figure 85 Selected station cannot be installed without first commissioning the JACE.



Click **Yes** to start the **Commissioning Wizard**, or **No** to simply return to the Station Copier.

This may occur if are trying to install a station in a new, uncommissioned JACE-8000, or in another JACE controller that has been converted from AX to N4, but still not yet commissioned. Despite documentation to first commission any new JACE using the platform **Commissioning Wizard**, this continues to occasionally come up. For complete details, see the *JACE Niagara 4 Install and Startup Guide*.

If all modules needed by the station are found on your Workbench computer, the Station Transfer Wizard starts normally. However, upon reaching the "Modules step", in some cases you may see a caution. For further details see "[Modules step](#)", [page 99](#).

Station Transfer Wizard

This wizard assists with any station copy (installing or backing up) by presenting a number of *steps*. The exact steps vary by the direction of copy, as well your selections in wizard step dialogs. In each step, click **Next** to advance to the next step. As needed, click **Back** to return to a previous step and make changes, or click **Cancel** to exit from the wizard (no station copy performed).

NOTE:

Use Cancel if you need to select a *different* station to copy; this reruns the wizard.

The wizard's **Finish** button is enabled only in the final step. When you click Finish, the related operations begin, and you see progress updates in the wizard's "Transferring Station" dialog. When complete, you click **Close** in that dialog to exit the wizard.

The following sections describe all possible steps in the Station Transfer Wizard:

- [Name step, page 97](#)
- [Delete step, page 97](#)
- [Content step, page 97](#)
- [Disposition step, page 98](#)
- [Station settings step, page 98](#)
- [Details step, page 99](#)
- [Modules step, page 99](#)
- [Stop station step, page 100](#)
- [Review step, page 101](#)

NOTE:

In the unlikely case where the source station config.bog file is currently in use ("locked"), the wizard opens in a state where you must **Cancel** to exit (no other steps are given).

- If *installing* a station, the source config.bog is locked if it contains unsaved changes (it is being edited elsewhere in Workbench). After saving changes, you can try the copy again.
- If *backing up* a station, the source config.bog is locked if currently in process of being saved. You can retry the copy later.

Name step

The first step in the **Station Transfer Wizard** is to confirm the name (or type a new name) for the copied station directory.

Figure 86 Station Transfer Wizard dialog, name step



Default name is the station directory being copied. If you rename the station, it will be identical to the source (copied) station in every way *except* name of its station directory.

Delete step**NOTE:**

This step is skipped for any station backup, or if a station install in either of these cases:

- No existing station exists on the target.
- The existing station is named the same as the one you are installing.

This step occurs because *all* JACE platforms have a support limit of one (1) installed station. The delete step simply cautions you that the existing station will be deleted, as shown below.

Figure 87 Station Transfer Wizard dialog, delete step

**NOTE:**

The entire remote station directory (all subdirectories and files) is deleted when the station install starts. If unsure, it may be best to **Cancel**, then backup the remote JACE station first.

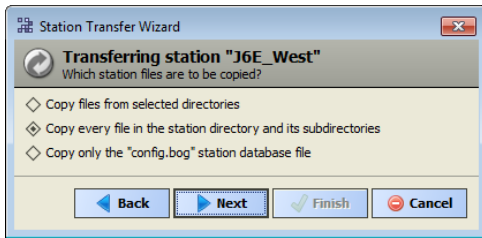
The next step is the "Content step".

Content step**NOTE:**

This wizard step is skipped if the source station consists of *only* a config.bog file.

After the “name step” and possibly “delete step”, the wizard asks you to select what station files to copy, with the default selection being “all” files and folders under that station directory, as shown below.

Figure 88 Station Transfer Wizard dialog, content step



The three possible selections are:

- Copy files from selected directories (not shown if source station has no subdirectories).
If you select this, a later “Details step” allows you to select the source subdirectories.
- Copy every file in the station directory and its subdirectories.
- Copy only the “config.bog” station database file.

The next step is the “Disposition step”.

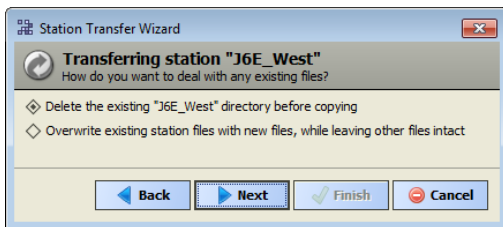
Disposition step

NOTE:

This wizard step occurs only when an identically-named target station already exists.

If the target station already exists, a disposition step asks what is to be done with it, as shown below.

Figure 89 Station Transfer Wizard dialog, disposition step



The two possible selections are:

- Delete existing station directory before copying.
- Overwrite existing station files with new files, while leaving other files intact.

If you previously selected “copy everything” from the “Content step”, the default pre-selection is the first (delete existing station directory). Otherwise, the second selection (overwrite) is pre-selected.

The next step is the “Station settings step”.

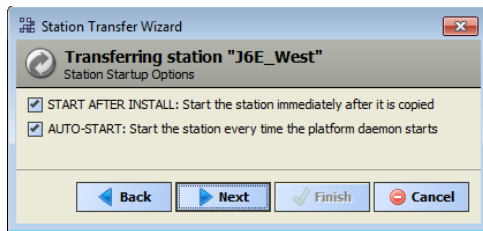
Station settings step

NOTE:

This wizard step is skipped for any station backup.

This step specifies the station’s Auto-Start setting.

Figure 90 Station Transfer Wizard dialog, station settings



Two items are listed:

- **START AFTER INSTALL:** Start the station immediately after it is copied.
- **AUTO-START:** Start the station every time the platform daemon starts.

Auto-start is one of two station settings for any station, as specified in the **Application Director** view by using “start checkboxes”. See [“Application and output controls” on page 33, page 35](#).

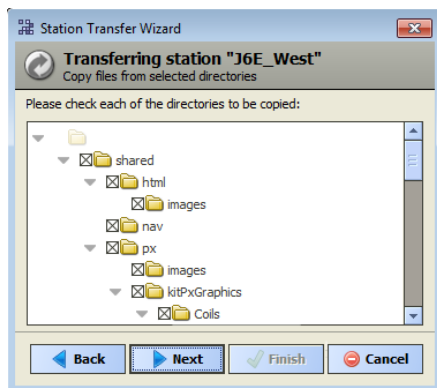
Typically, you enable both settings and go to the next step, either the “Details step” or “Modules step”.

Details step

NOTE:

This wizard step is skipped for any station backup, as well as for a station install—unless you selected “copy selected directories” in the “Content step”.

Figure 91 Station Transfer Wizard dialog, details step



As shown above, this step provides a tree to select station subdirectories (folders) to include in the copy. By default, all selectable folders are both expanded and selected, while unselectable folders are not (note that if present, a station’s `alarm` and `history` folders are unselectable).

For any selectable folder, click to toggle it as either selected (with `x`) or unselected (no `x`).

Modules step

NOTE:

This wizard step is skipped if a station backup, or if all modules required by the station to be installed are already in the JACE controller. In this case, you see either the “Stop station step” or “Review step” instead.

This step occurs if the target JACE platform is missing one or more of the modules required by the station being copied (installed). It lists the missing modules/versions that will be installed during the station copy operation. If included, this is the *final step* before the station copy process starts.

NOTE:

Dependencies of the missing modules are compared against the software that is already installed in the target JACE platform. The Station Copier looks for versions of those missing modules in your Workbench software database that can be installed *without* re-commissioning the JACE, by default.

There are two possible results when the wizard reaches this step:

- [Station can be installed with most current modules, page 100](#)
- [Station can be installed with “out of date” modules, page 100](#)

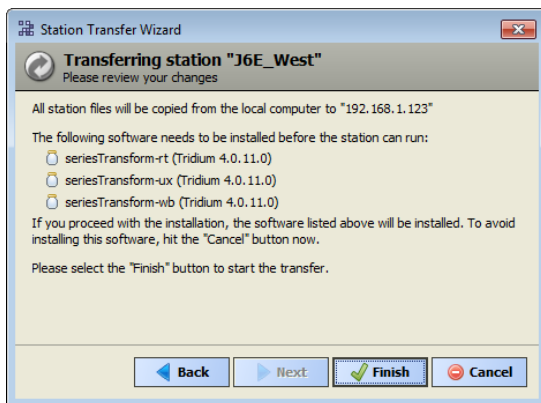
In either case, to continue you click either:

- **Finish** — Start the local-to-remote copy, including installation of the listed modules. Progress updates appear in a “Transferring station” dialog.
- **Cancel** — Exit from the Station Transfer Wizard, then either select another station to install, or if a JACE upgrade is possible (and you have purchased an upgrade license for it) run the **Commissioning Wizard** to upgrade the JACE controller, including the installation of a station.

Station can be installed with most current modules

If all missing modules can be installed using the most current versions, they list without any warning, as shown below.

Figure 92 Station install example, all missing modules are most current versions

**Station can be installed with “out of date” modules**

If any module to be installed is not the most current version, you have the option to cancel the station install. A dialog explains that you can use the **Commissioning Wizard** to upgrade the JACE.

This may occur at some *future point* after a “point release” of Niagara 4, say N4.1, where you have previously imported the software database of an N4.0 installation.

For related details, see Software Manager topics [“About your software database” on page 66, page 85](#) and [“Software Import” on page 70, page 89](#). Also see [“Upgrading a JACE” on page 42, page 52](#).

Stop station step

You can see this wizard step in any of these scenarios:

- You are copying the station running in a remote platform to your local computer, and you selected either “copy files from selected directories” or “copy only the config.bog station database file” in the previous “Content step”. Note this step is skipped if you elect to “copy every file in the station directory and its subdirectories”. However, a station save occurs before the station copy transfer starts.
- If installing a “same-named” station.

This step reminds you that the station must be stopped while it is copied, as shown below.

Figure 93 Station Transfer Wizard dialog, stop station step



Click **Next** to go to the “Review step”.

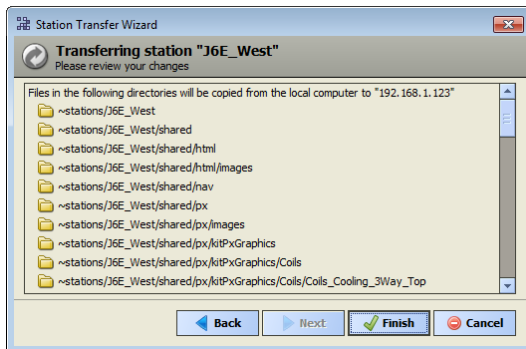
Review step

NOTE:

This wizard step is skipped when installing a station where additional modules are required. (Instead, the “Modules step” provides the **Finish** button. See [Figure 92](#) [Figure 91 on page 78](#), [page 100](#) for an example.

This step provides a summary of choices from previous steps, and a **Finish** button to begin the station copy process.

Figure 94 Station Transfer Wizard dialog, review step

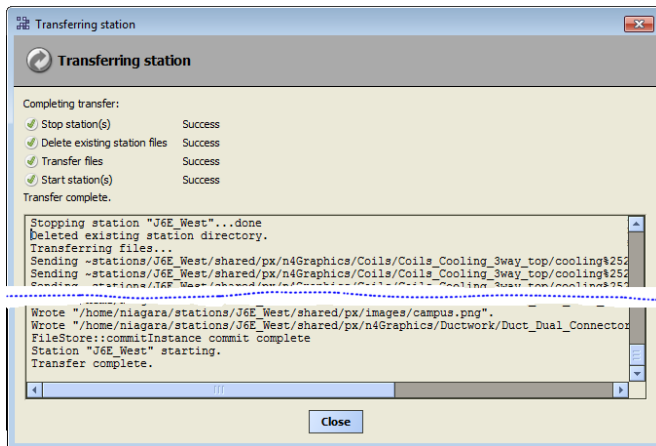


As shown above, if you selected only *specific* station subdirectories to copy (from the “Details step”), they are listed. If needed, click **Back** to make changes, or click **Finish** to begin the copy process and observe progress in the “Transferring station” dialog.

Transferring station

After clicking **Finish** in the “Modules step” or “Review step” of the Station Transfer Wizard, the station copy process begins and updates appear in a this dialog, as shown below.

Figure 95 Station Transfer Wizard, Transferring station (copy) process



Depending on the type of copy, the following operations may be included in this process:

- If installing a station (copy Workbench User Home-to-daemon User Home):
 - Stop all stations — whenever modules require to be installed.
 - Stop one station — any JACE where same station is being reinstalled.
 - Delete station(s) — if you chose to delete station in the “Disposition step”, or if a station needs to be deleted to stay under maximum number of stations (only one for any JACE platform)
 - Transfer files — includes station and module files (actual copy portion).
 - Start station — if a station had required to be stopped (module installation), or if you chose to start the station in the “Station settings step”.
- If backing up a station (copy daemon User Home-to-Workbench User Home):
 - Save station — whenever remote station is currently running.
 - Transfer files — includes station files (actual copy portion).

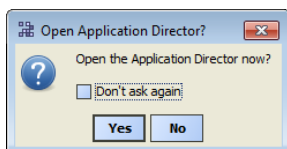
NOTE:

A popup explaining that the existing station must be saved (if a backup) or stopped (if installing) may appear for a few seconds. Following, and during execution of the various operations, a **Cancel** button is available. If you click Cancel before all operations complete, the installation (or backup) is not valid.

After all operations are finished, a **Close** button is available and the last update in the dialog is “Transfer complete.” Click **Close** to exit the wizard.

By default, after *installing* a station, the wizard exits with a popup asking if you wish to switch to the **Application Director** platform view.

Figure 96 Switch to Application Director popup



Because it is a good idea to observe a station’s output upon first startup, you typically select **Yes**. To *always* automatically switch to the **Application Director** after installing a station, click the checkbox to “Don’t ask again” before selecting **Yes**. Then, you do not see this popup again.

Renaming stations

The **Station Copier** lets you rename any station, either in your *Workbench User Home* (left side) or in the opened platform's *daemon User Home* (right side).

As shown above, a Rename dialog appears when you select a station and click **Rename**.

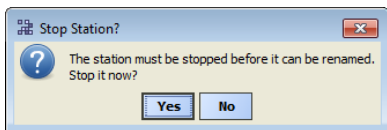
Figure 97 Rename station dialog



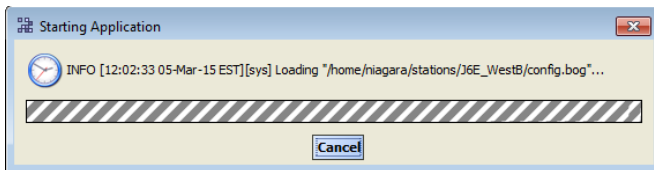
NOTE:

Be careful when renaming stations, as there is *no undo*. Furthermore, please note the following:

- Any *running* station that is renamed must first be *stopped*—a confirmation popup dialog informs you of this after you enter the new station name and click OK.



After the station stops it becomes renamed, and then automatically restarts. A series of other popups appear, each showing a station startup message.



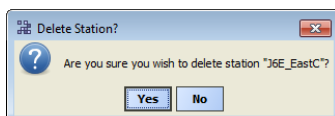
- If a renamed running station is already included in the NiagaraNetwork of other stations, its corresponding NiagaraStation component *will remain "down" until renamed* to match the new name. Thus, all child components (Niagara proxy points and so on) will also be down until this is done. In addition, other unforeseen consequences may result from changing the name of a station that has already been integrated into other stations.

Therefore, station renames are best done on the *Workbench User Home* (left side) stations, or when initially configuring a job site network, such as when first installing (copying) a station.

Deleting stations

The **Station Copier** lets you delete *any* station, either in your *Workbench User Home* (left side) or in the opened platform's *daemon User Home* (right side).

Figure 98 Confirm delete station dialog



As shown above, a confirmation dialog appears when you select a station and click **Delete**.

NOTE:

Be careful when deleting stations, as there is *no undo*. Furthermore, note the following:

- The entire selected station directory gets deleted, including all subdirectories and file contents.
- Special notification does *not* occur if you choose to delete a *running* station (you may briefly see a “stop station” popup, with opportunity to **Abort**).
- Also in general (as a precaution), *before* deleting a running station, it is generally recommended to make a backup copy first. If desired, when backing up you can rename it using some “temp” convention to flag it for later housekeeping.

TCP/IP Configuration

TCP/IP Configuration is one of several platform views. Typically, you use it to initially configure a remote controller’s TCP/IP settings.

NOTE:

If connected to any Windows-based platform, all settings in this view are read-only. You typically use the Windows Control Panel for making these changes on a PC.

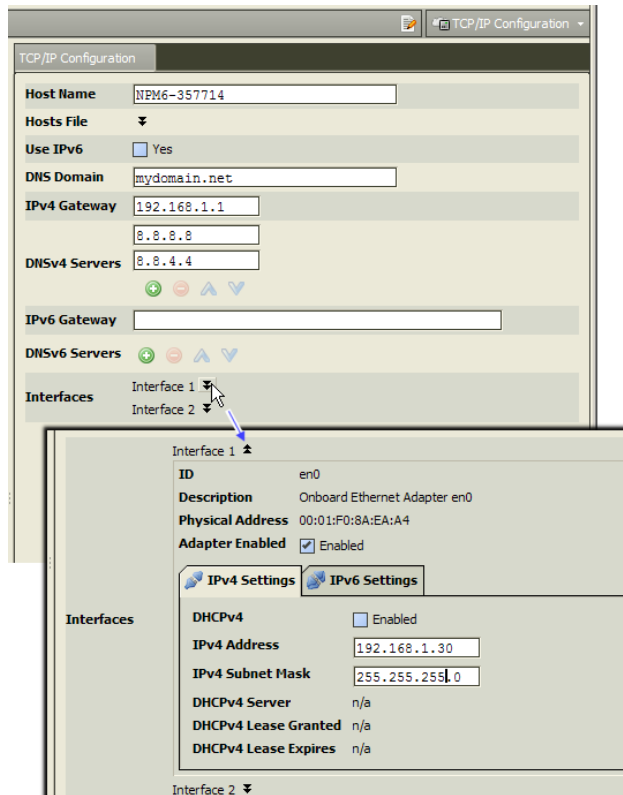
- [Configuring TCP/IP, page 104](#).
- [TCP/IP Host fields, page 105](#)
- [TCP/IP DNS fields, page 106](#) (host-level for JACE controller platforms only)
- [TCP/IP Interface fields, page 107](#)

Configuring TCP/IP

Configuring TCP/IP communication settings is a task for the systems integrator when initially setting up a controller.

Prerequisites:

- Step 1 Open a secure connection to the platform.
- Step 2 Expand the **Platform** container in the Nav tree and double-click the **TCP/IP Configuration** container.

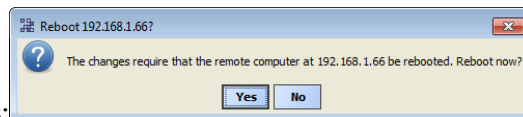


The system displays the main TCP/IP properties.

Step 3 Click the drop-down arrows to expand a group of properties.

To save yourself time when making multiple changes, enter *all* changes before you continue.

Step 4 When you finish the configuration, click **Save**.



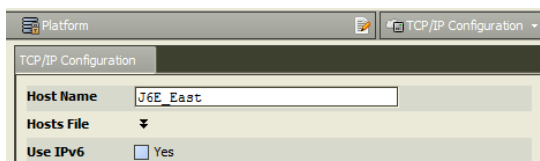
The system displays:

Step 5 Reboot the controller for the changes to take effect.

TCP/IP Host fields

The top of the **TCP/IP Configuration** view provides the platform's TCP/IP host settings.

Figure 99 Hosts fields on platform TCP/IP Configuration view



These available host fields are as follows:

- Host Name

Synonymous with "computer name," this is a string that can be processed by a DNS server to resolve to an IP address. On Windows-based systems, this hostname is the computer's identification in its work-group or domain. If using hostnames, each Niagara platform should have a *unique* hostname.

- Hosts File

The hosts file is a standard TCP/IP hosts file, where each line associates a specific IP address with a known hostname. To review, click the expand control to see all entries.

If a JACE controller, you can edit its host file.

- To *add* an entry, click at the end of the last line and press Enter.
Then type the IP address, at least one space, then the known hostname.
- To *delete* an entry, drag to highlight the entire line, then press Backspace.
Click the expand control again to collapse the Hosts File editor.

- Use IPv6

Default is No (unchecked). If set to Yes (checked), Niagara (platform daemon and station) respond to IPv6 requests, that is, creates IPv6 server sockets (daemon) and IPv6 fox multicast sockets.

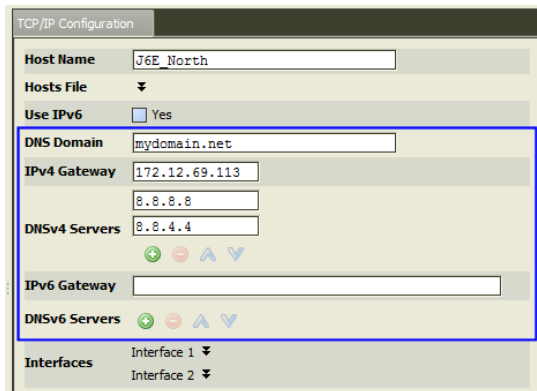
TCP/IP DNS fields

If connected to a JACE controller, the DNS and gateway settings are also “host-level” parameters in the TCP/IP Configuration view, as shown below.

NOTE:

If a Windows-based host, DNS and gateway settings are available under each Interface section.

Figure 100 Host-level fields for any JACE controller includes DNS and gateway



The available fields for JACE controllers are as follows:

- DNS Domain
The TCP/IP Domain Name System (DNS) domain this host belongs to, if used.
- IPv4 Gateway
The IP address of the router that forwards packets to other IPv4 networks or subnets. A valid gateway address is required in multi-station (JACE) jobs to allow point discoveries under NiagaraNetworks.
- DNSv4 Servers
The IP address of one or more DNS servers (if available), where each can automate associations between hostnames and IPv4 addresses. Included are icon-buttons to Add (to enter IP address of server), Delete, and move Up/Down (to set the DNS search order).
- IPv6 Gateway
The IPv6 address for the router that forwards packets to other IPv6 networks or subnets.
- DNSv6 Servers

The IPv6 address for one or more IPv6 DNS servers (if available), where each can automate associations between hostnames and IPv6 addresses. Included are icon-buttons to Add (to enter IP address of server), Delete, and move Up/Down (to set the DNS search order).

TCP/IP Interface fields

For each Ethernet port on the connected platform, the **TCP/IP Configuration** platform view provides an expandable **Interface n** section.

All Niagara 4-compatible JACE controllers have two Ethernet ports: LAN1 and LAN2. In the **TCP/IP Configuration** view, they are listed as **Interface 1** (en0) and **Interface 2** (en1).

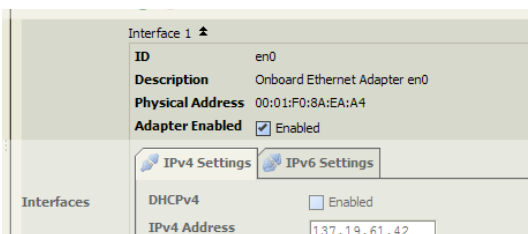
NOTE:

A JACE-8000 controller includes an onboard “WiFi” adapter. If enabled, it appears in the **TCP/IP Configuration** view with multiple **Interface n** selections—where two have significance:

- tiw_sap — (Titan wireless supplicant Access Point configuration)
- tiw_sta — (Titan wireless supplicant Client mode configuration)

Where only one mode above can be enabled, according to the WiFi switch position on the controller.

Figure 101 TCP/IP Interface fields, top properties



As shown above, each **Interface** has the following properties at the top:

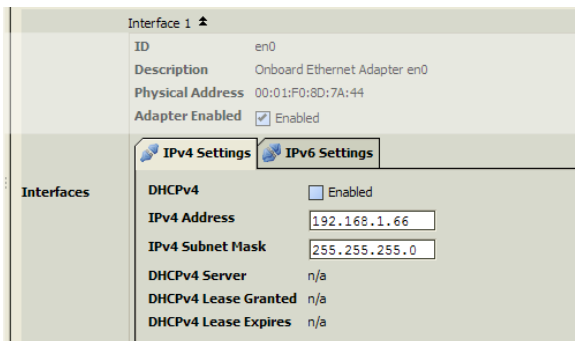
- **ID**
A read-only OS identifier for the hardware interface, such as “en0” if a JACE controller, or if a Windows platform, either a 128-bit GUID (globally unique identifier) or a Windows network connection name, such as “Local Area Connection 2”.
- **Description**
A read-only text string such as “Onboard Ethernet Adapter en0” for a JACE controller, or “Intel(R) PRO/100 VE Network Connection” for a Win32-based host, describing a NIC model.
- **Physical Address**
The unique 48-bit MAC address of the Ethernet adapter, in six two-hexadecimal digits. For example, for the “en0” Interface 1 port of a JACE controller: 00:01:F0:80:13:E6
- **Adapter Enabled**
Checkbox to specify whether the Ethernet port is usable.

Below the properties above, each **Interface** has two *separate tabs*, as follows (each with properties):

- [IPv4 Settings, page 108](#)
- [IPv6 Settings, page 109](#)

IPv4 Settings

Figure 102 IPv4 tab for Interface of JACE controller, in platform TCP/IP Configuration view



The following properties are on the **IPv4 Settings** tab of the selected Interface:

- DHCPv4

NOTE:

Only *ONE* adapter of any JACE controller may have DHCP enabled.

A checkbox to specify DHCP (Dynamic Host Configuration Protocol) instead of static IP addressing. Successful use requires a DHCP server installed on your network. If enabled, other interface fields such as IP Address and Subnet Mask become read-only, as these are assigned by the DHCP server after the platform reboots.

In general (for stability), static IP addressing is recommended over DHCP. If configuring for DHCP it is recommended that you reserve a specific, fixed IP address for this JACE host in the network's DHCP server/router configuration, noting the MAC address of this adapter as shown above.

CAUTION:

Do not enable DHCP unless sure that your network has one or more DHCP servers! Otherwise, the JACE may become unreachable over the network.

- DNS Domain

(Windows hosts only) The TCP/IP Domain Name System (DNS) domain the host belongs to, if used.

- IPv4 Address

The "static" IP address for this host, unique on your network.

Be careful to understand the following:

NOTE:

- If enabling multiple ports, note that IP address must be on *different subnets*, otherwise the ports will not function correctly.

For example, with a typical "Class C" subnet mask of 255.255.255.0, setting Interface 1=192.168.1.99 and Interface 2=192.168.1.188 is an *invalid* configuration, as both addresses are on the *same subnet*.

- A JACE controller *does not* provide IP routing or bridging operation between different Interfaces (LAN ports, GPRS, dialup, WiFi).

- IPv4 Gateway

(Windows hosts only) IP address for the device that forwards packets to other networks or subnets.

- IPv4 Subnet Mask

The "static" IP subnet mask used by this host.

- DHCPv4 Server

Applies only if DHCP is enabled. Shows read-only address of the DHCP server from which this host last obtained its IP address settings.

- DHCPv4 Lease Granted

Applies only if DHCP is enabled. Shows a read-only timestamp of when the DHCP lease started.

- DHCPv4 Lease Expires

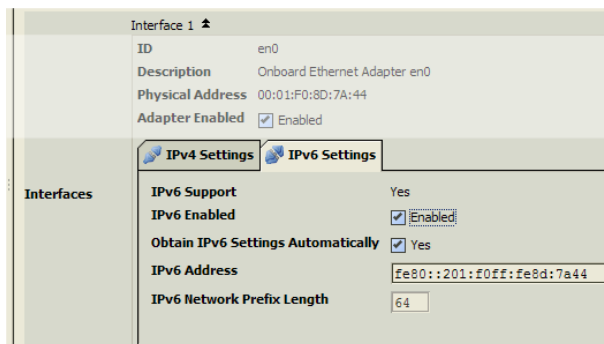
Applies only if DHCP is enabled. Shows a read-only timestamp of when the DHCP lease will expire, and will need renewal.

- DNSv4 Servers (DNS Servers)

(Windows hosts only) The IP address for one or more DNS servers, each of which can automate associations between hostnames and IP addresses. Included are icon-buttons to Add (to enter IP address of server), Delete, and move Up/Down (to set the DNS search order).

IPv6 Settings

Figure 103 IPv6 tab for Interface of JACE controller, in platform TCP/IP Configuration view



The following properties are on the **IPv6 Settings** tab of the selected Interface:

- IPv6 Support

Yes or No, as read-only. Indicates if host platform's OS supports IPv6.

- IPv6 Enabled

Checkbox for Enabled, where default is cleared (disabled). If a Windows host, this indicates if it is configured with the IPv6 protocol.

- Obtain IPv6 Settings Automatically

Checkbox for Enabled (default). Provides for "auto-configuration" of IPv6 address, if acceptable. If enabled on a JACE controller, the next two properties are read-only. If cleared, the two properties below must be entered manually.

- IPv6 Address

The host's IP address in IPv6 format, to be unique on its network.

- IPv6 Network Prefix Length

The number of left-most contiguous bits of the IPv6 address (in decimal) that compose the subnet prefix.

- DNSv6 Servers

(Windows hosts only, providing host's OS has IPv6 enabled) Read-only IPv6 address for one or more DNS servers, each of which can automate associations between hostnames and IPv6 addresses.

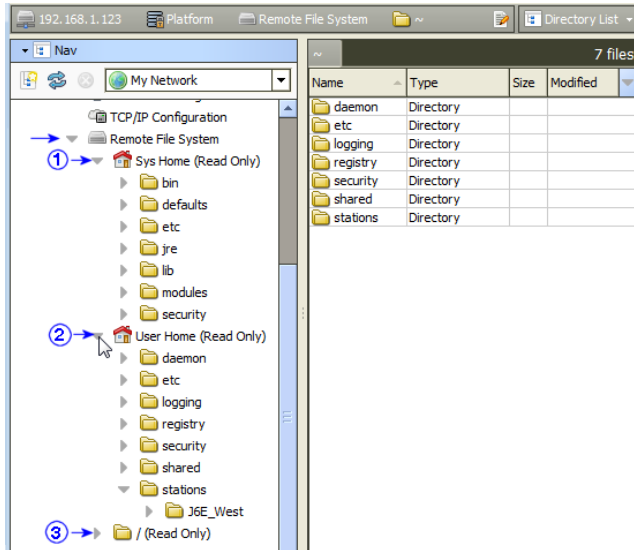
Remote File System

The **Remote File System** view is one of several platform views. It provides a *read-only* view of the remote platform's file system. As needed, you can expand folders and examine and/or copy files to your local computer.

NOTE:

To edit or write files on the remote Niagara platform, you must use the platform **File Transfer Client** view. See ["File Transfer Client"](#) on page 44, page 53.

Figure 104 Remote File System for JACE controller platform



As shown above, included in the Nav tree under the **Remote File System** are main nodes for:

1. The "system home" (**Sys Home**) root folder, under which all installation/runtime files are installed.
2. The "user home" (**User Home**) root folder for the platform daemon, under which all configuration files are stored.
3. (JACE controllers only) The root folder for the entire filesystem, with browse capability.

For details on **Sys Home** and **User Home**, see ["Niagara 4 directory \(homes\) architecture"](#) on page 20, page 22.

Chapter 2 Troubleshooting

Topics covered in this chapter

- ◆ Station installation troubleshooting

Station installation troubleshooting

These troubleshooting tips concern the Station Copier.

I started the Station Copier, selected the station and clicked Copy, and got a message prompting me to enter a file passphrase.

The bog file's passphrase is not the same as the target host's system passphrase. You must enter the correct file passphrase to proceed with the station copy. An alternative is to edit the bog file offline to either unlock the file, making it unprotected, or to change the passphrase value. However, either of these choices will clear any sensitive information in the file.

I started the Station Copier, entered the station name and got a message indicating that the controller requires commissioning.

The controller may be new or you may be upgrading from NiagaraAX to Niagara 4.0 and you forgot to commission the platform first. Run the Commissioning Wizard and come back to the Station Copier later.

I started the Station Copier, selected the station to copy and clicked Next, but the only option available is to Cancel and exit the wizard.

The station database (config.bog) is locked. This happens if:

- The config.bog has been edited elsewhere in Workbench and contains unsaved changes. After saving changes, try the copy again. [how do I save the changes?](#)
- The system is in the process of saving the config.bog using the BackupService. Wait a while and try the copy again.

Chapter 3 Platform Services

Topics covered in this chapter

- ◆ About Platform Services
- ◆ PlatformServiceContainer parameters
- ◆ SystemService (under PlatformServices)
- ◆ Platform service types
- ◆ Using platform services in a station
- ◆ About the NtpPlatformService

This section explains the platform access available in a running *station*—in other words, the *station's perspective* on its host platform. Unlike the various platform views, a platform connection is *not needed* to access platform services. Instead, you need only a standard station (Fox) connection.

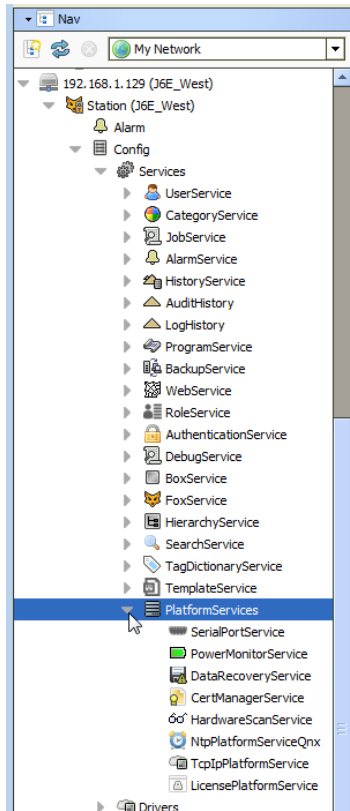
The following main sections provide more details:

- ["About Platform Services" on page 88, page 113](#)
 - ["Component differences for platform services" on page 88, page 114](#)
- ["PlatformServiceContainer parameters" on page 89, page 115](#)
 - ["PlatformServiceContainer status values" on page 89, page 115](#)
 - ["PlatformServiceContainer configuration parameters" on page 90, page 117](#)
 - ["Model-specific PlatformServiceContainer properties" on page 92, page 120](#)
 - ["PlatformServiceContainer actions" on page 92, page 120](#)
- ["SystemService \(under PlatformServices\)" on page 93, page 121](#)
- ["Platform service types" on page 94, page 122](#)
- ["Using platform services in a station" on page 95, page 123](#)
 - ["JACE power monitoring" on page 95, page 123](#)
 - ["PlatformServices binding and link caveats" on page 95, page 123](#)
- ["About the NtpPlatformService" on page 96, page 124](#)
 - ["About the Ntp Platform Service Editor" on page 96, page 125](#)
 - ["About the Ntp Platform Service Editor Qnx" on page 97, page 125 \(most JACEs\)](#)
 - ["About the Ntp Platform Service Editor Win32" on page 99, page 127](#)
 - ["NTP port/firewall considerations" on page 99, page 128](#)

About Platform Services

Under **Config, Services**, every *running* Niagara station has a **PlatformServices** container, which any station user with admin-level permissions to this component can access.

Figure 105 Example Niagara 4 JACE station's PlatformServices



Platform services in a running station provide two main types of functionality:

- A subset of platform views available in a platform connection. Platform services do *not* provide the full set of functions available in Workbench platform connection. For example, you cannot install or upgrade software, or transfer stations and files. However, a number of platform configuration views are available under a station's PlatformServices.
- Certain platform configuration settings accessible *only* through PlatformServices—that is, not available in a client platform connection.

NOTE:

When engineering station security, be *careful* about assigning user permissions to PlatformServices and its child service components. In general, you should regard this portion of the station as *most critical*, as it allows access to items such as host licenses and TCP/IP settings. Furthermore, right-click actions on the PlatformServices include "Restart Station". For related details, see the *Niagara 4 Station Security Guide*.

PlatformServices and all child components are *unique from all other station components*. For details, see "[Component differences for platform services](#)", page 114

Component differences for platform services

PlatformServices is *different from all other components* in a station in the following ways:

- It acts as the *station interface* to specifics about the *host platform* (whether JACE or a PC).
- It is built *dynamically* at station *runtime*—you do not see PlatformServices in an offline station.
- Changes you make to PlatformServices and all child services are *not stored in the station database*. Instead, changes are stored in other files on that platform, such as its platform.bog file, or within the platform's operating system.

NOTE:

Do not attempt to edit platform.bog directly; always use PlatformServices' views!

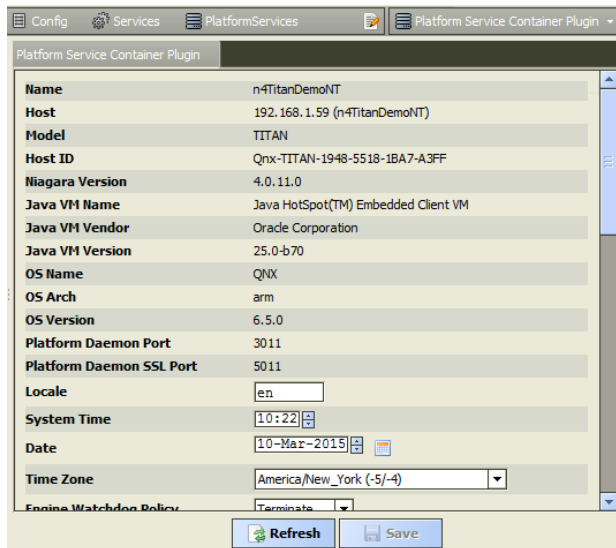
In summary, when you make changes under a station's PlatformServices, those changes are independent of the running station. If you install another station, platform services are dynamically recreated again when the new station starts, based upon the last settings.

In addition, understand that some changes in platform services views may require the host to be rebooted to become effective. Examples include TCP/IP changes, or some NTP-related changes in a JACE controller. A "Reboot Now?" popup dialog appears upon saving such a change.

PlatformServiceContainer parameters

In addition to being a container, the default **Platform Service Container Plugin** view provides various status and configuration entries for the host platform. In the Nav tree, double-click **PlatformServices** to access this view, as shown below.

Figure 106 PlatformServicesContainerPlugin view (many entries not shown)



Included are many read-only status values as well as configuration parameters. Each is described in separate sections as follows:

- [PlatformServiceContainer status values, page 115](#)
- [PlatformServiceContainer configuration parameters, page 117](#)

By default, any **PlatformServiceContainer** also provides three right-click actions. See "[PlatformService-Container actions](#)" on [page 92](#), [page 120](#).

PlatformServiceContainer status values

Status values in a station's **PlatformServices** container include the following:

- Name
Name of running station.
- Host
IP address of host platform.
- Model

Model of host platform type, such as NMP6, JACE-8000, or Workstation. See [“Models of platforms” on page 19](#) for further details.

- **Host ID**
Niagara host identifier, a string unique to this one machine.
- **Niagara Version**
Version and build number of the Niagara distribution running in the host platform.
- **Java VM Name**
Java virtual machine used, for example, “Java HotSpot(TM) Embedded Client VM” for any N4 controller, or “Java HotSpot(TM) 64-Bit ServerVM” for a Supervisor on a Windows host.
- **Java VM Vendor**
Vendor for Java VM: Oracle Corporation.
- **Java VM Version**
Version of Java VM, for example, “25.0-b 70” for the Java 8 compact3 VM on a controller, or “25.31-b07” for the Java 8 SE VM on a Windows host.
- **OS Name**
Operating System name, such as “QNX” or “Windows 7.”
- **OS Arch.**
Machine architecture for OS, such as “arm” or “ppc” (controller hosts) or “amd64” (Windows hosts).
- **OS Version**
Operating System version, such as “6.5.0” (QNX) or “6.1” (Windows 7).
- **Platform Daemon Port**
Port number on which the platform daemon that started the station is listening for its platform server (3011, or another port number). This can prove useful in case you changed the platform port (see [“Change HTTP Port” on page 57](#)), but then forgot what the new port is.
- **Platform Daemon TLS Port**
Port number on which the platform daemon is listening for its platform TLS server (5011, or another port number, provided that platform TLS enabled). If platform TLS is disabled, it reads `Unknown`. This can prove useful in case you changed the platform TLS port (see [“Change SSL Settings” on page 57](#)), but then forgot what the new port is.

NOTE:

In the container plugin, most of the remaining entries are *configuration* parameters. However a few status values are also mixed in, and are described below.

- **Number of CPUs**
Number of CPUs used in the host platform (typically 1 if a controller, more if a Windows host).
- **Current CPU Usage**
Percentage of CPU utilization in the last second.
- **Overall CPU Usage**
Percentage of CPU utilization since the last reboot.
- **Filesystem**
File storage statistics for the host, including total file space, available (free) space, and file block size (minimum size for even the smallest file). For a JACE-8000 host, it may look similar to:

```

Total          Free          Files  Max Files

```

```

/          3,476,464 KB 3,039,088 KB 602      108640
/mnt/aram0 393,215 KB 381,019 KB 0         0
/mnt/ram0 8,192 KB 8,192 KB 0         0

```

- **Physical RAM**

Current total and free RAM statistics for the host. For a JACE-8000, it may look similar to:

```

      Total      Free
1,048,576 KB 113,424 KB

```

- **Serial Number**

(Appears only if a JACE host). The controller's unique serial number.

- **Hardware Revision**

(Appears only if a JACE host). Hardware revision of the controller.

- **Hardware Jumper Preset**

(Applies only if a JACE host, except for a JACE-8000) Either true or false—indicates whether or not the mode jumper is installed for "serial shell mode" access. Read at boot time only. See "System shell" in the *JACE Niagara 4 Install & Startup Guide*.

Also see the section ["PlatformServiceContainer configuration parameters"](#), page 117.

PlatformServiceContainer configuration parameters

Configuration properties of a station's **PlatformServices** Container are listed below. If needed, you can change any in the container plugin view (property sheet)—click **Save** to write to the host platform.

NOTE:

It is recommended that you leave engine-related parameters and other advanced settings at *default* values, unless you have been directed otherwise by Systems Engineering.

- **Locale**
Determines locale-specific behavior such as date and time formatting, and also which lexicons are used. A string entered must use the form: language ["_" country ["_" variant]]. For example, U.S. English is "en_US" and traditional Spanish would be "es_ES_Traditional".
- **System Time**
Current local time in host (read-only if a Windows host).
- **Date**
Current local date in host (read-only if a Windows host).
- **Time Zone**
Current local time zone for host (read-only if a Windows host). For more details, see ["Time Zones and Niagara 4" on page 129](#).
- **Engine Watchdog Policy**
The *engine watchdog* is a platform daemon process, to which the station periodically reports its updated engine cycle count. The watchdog purpose is to detect and deal with a "hung" or "stalled" station, and is automatically enabled when the station starts.
The Engine Watchdog Policy defines the response taken by the platform daemon if it detects a station engine watchdog timeout. Watchdog policy selections include:
 - Log Only — Generates stack dump and logs an error message in the system log. (The station should ultimately be restarted if a watchdog timeout occurs with the "Log Only" setting).
 - Terminate — (Default) Kills the VM process. If "restart on failure" is enabled for the station (typical), the station is restarted.
 - Reboot — Automatically reboots the host JACE platform. If "auto-start" is enabled for the station, the station is restarted after the system reboots.
- **Engine Watchdog Timeout**
Default is 1 minute, and range is from 0 ms to infinity. If the station's engine cycle count stops changing and/or the station does not report a cycle count to the platform daemon within this defined period, the platform daemon causes the VM to generate a stack dump for diagnostic purposes, then takes the action defined by the Engine Watchdog Policy.
- **Enable Station Auto-Save**
Either Enable (default) or Disable. Allows for "auto save" of running station to "config_backup_<YYMMDD>_<HHMM>.bog" file at the frequency defined in next property. Auto-saved backup files are kept under that station's folder.
- **Station Auto-Save Frequency**
Default is every 24 hours for any JACE platform, or every (1) hour if a Windows host. Range is from 1 to many hours.
- **Station Auto-Save Backups to Keep**
Oldest of kept backups is replaced upon next manual save or auto-save backup, once the specified limit is reached. The default value for JACE platform is 0 (none), and should be kept low.
However, changing to 1 provides a benefit in the case where a catastrophic (yet inadvertent) station change is made, such that a station "kill" can be issued to revert back to the backup copy on the JACE.
In Windows hosts, the default is 3, and typically can be safely adjusted up, if desired.

- Battery Present

(Applies only if a JACE host *other* than a JACE-8000) Applies to configuration of a JACE's backup battery. Used to specify whether the controller has an integral backup battery, typically an onboard NiMH battery. The default property value is true—which is recommended *unless* the controller is *both* SRAM-equipped *and is without an attached backup battery* (there is no way to detect the latter through software).

If set to false and saved, upon the next reboot the station's PowerMonitorService no longer monitors for a backup battery, with the underlying "power daemon" stopped. This prevents nuisance "battery bad" alarms. Station backup is dependent totally on SRAM and the station's DataRecoveryService (the JACE must have the platDataRecovery module installed, and be licensed for DataRecovery).

The configuration described above is only *one* of three possible backup options for an SRAM-equipped controller that can also have a backup battery installed (e.g. JACE-6E or JACE-3E, or else a JACE-6 or JACE-7 with an SRAM option card). The two other options are to use *both* backup battery *and* SRAM for backup, or to use *backup battery only* (and *not* SRAM). These other two options require that this Battery Present property is set to true.

For related details, refer to the document *JACE Data Recovery Service (SRAM support)*.

- Failure Reboot Limit

(JACE platforms only) Limits the number of station restarts that can be triggered by station failures, within the Failure Reboot Limit Period, below (if the host is so configured using the **Application Director**, see "[Start checkboxes](#)" on page 33). Default value is 3.

- Failure Reboot Limit Period

(JACE platforms only) Specifies the repeating frequency of the Failure Reboot Limit period, with a default value at 10 minutes.

These two "Failure Reboot" settings are also adjustable (in any version of QNX-based host) within that JACE's !daemon/daemon.properties file, in the following two properties:

- failureRebootLimit=x (where x is integer, default is 3)
- failureRebootLimitPeriod=y (where y is long in milliseconds, default is 3600000)

- RAM Disk Size

Has one configurable field and one read-only field:

- Min Free — minimum allowable free size in %. If status is not Ok, a "Low RAM disk space" warning is overlaid in all Workbench views of the station.
- Size —Read-only in MB, where default is 32 for a JACE-3E or JACE-6 or JACE-6E series, or 48 for a JACE-7 series, or 394 for a JACE-8000 series. Specifies the size of RAM disk used to store history and alarm files.

- Java Heap

Has one configurable "Min Free" field, in MB. Specifies the *minimum* free Java heap size, in MB, against which the station compares (tests) for low memory conditions, that is excessive Java heap. The default varies according to JACE model. This test automatically runs once a minute. If the heap free byte count is less than the defined minimum free heap size, a "low memory warning" appears in *all Workbench views* of the station. The warning is a yellow message box overlaid on any new view accessed, or on any current view that is refreshed. This warning is removed when the heap free byte count rises above the defined minimum size—such as might occur if enough components are deleted from the station.

All memory statistics, including those for heap, are accessible on a station opened in Workbench, via the **Resource Manager** view of the Station component.

- Open File Descriptors

Has one configurable "Min Free" field, related to number of files (and/or open sockets). Specifies the maximum amount of file descriptors that can be used. That is, the read-only "Max Open" number minus the "Min Free" amount. File descriptors are used for histories, modules, and Fox connections. If exceeded a "Station has too many open files or sockets" warning is overlaid in all Workbench views of the station.

- **Free RAM**
Has one configurable “Min Free” field, in KB. Specifies the minimum RAM that can be left free during station operation. If status is not Ok, a “Low free RAM” warning is overlaid in all Workbench views of the station.
- **Disk Space**
Has one configurable “Min Free” field, in %. Specifies the minimum percentage of disk storage that can be left free during station operation. Below this amount, a “Platform running low on disk space” warning is overlaid in all Workbench views of the station.
- **Files**
Has one configurable “Min Free” field, to specify the minimum number of free files available during station operation. Below this amount, a related platform warning appears. Note that the PlatformServiceContainer status property “Filesystem” includes both the current number of files and the maximum number of files for each partition on a JACE controller.

Also see the section [“Model-specific PlatformServiceContainer properties”](#), page 120.

Model-specific PlatformServiceContainer properties

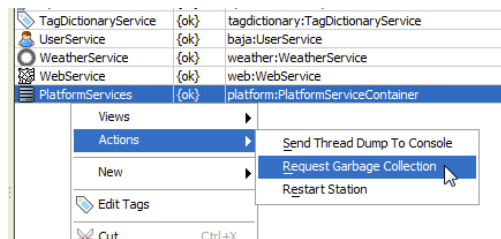
Some JACE controller models may have yet more **PlatformServices** properties, specific to special hardware features. This is in addition to the standard and additional properties described above. Typically, these are configured at JACE commissioning time.

For more details, see “Controller-specific PlatformServices properties” in the *JACE Niagara 4 Install & Start-up Guide*.

PlatformServiceContainer actions

The **PlatformServices** Container also provides three (right-click) actions, as shown below.

Figure 107 PlatformServicesContainer actions.



These actions are described as follows:

- **Send Thread Dump to Console**
Causes that host’s platform daemon to have the station send a VM thread dump to its standard output (console), equivalent to the “Dump Threads” command in the platform **Application Director** view. Typically used only during troubleshooting.
NOTE:
Apart from Application Director (platform access) to view station output, you can also view a “snapshot” of station output in a browser. Do this via the “stdout” link in the **spy** utility, at URL `http://<hostIP>/ord?spy:/stdout`
- **Request Garbage Collection**
Causes the JVM running the station to perform garbage collection. This results in a “best effort” towards releasing unused objects and making more memory available on the “heap”. Note that current heap and memory statistics for any running station are available on the ResourceManager view of the station component.

- Restart Station

Produces a popup confirmation dialog. Applies directly to any Niagara 4 station, whether running on a JACE controller or a Windows platform. It is equivalent to issuing a “Restart” command from the platform **Application Director** view (station is saved on its host, then restarted). Note that unlike in NiagaraAX, this *does not* result in a reboot of a JACE controller.

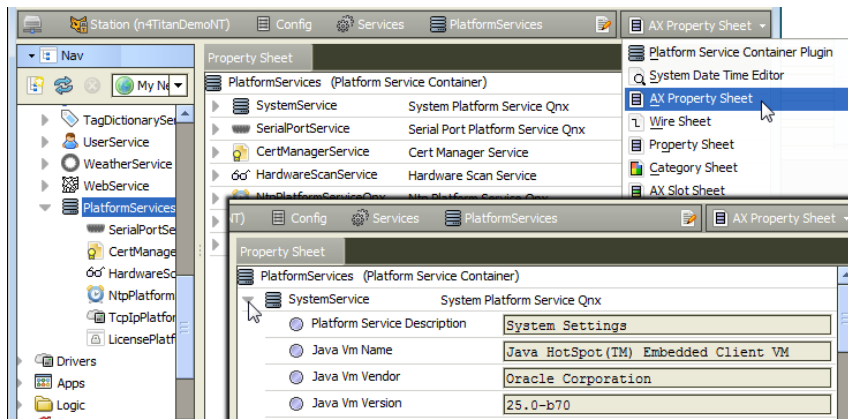
NOTE:

Also, most *child* services under the **PlatformServices** Container have an available “Poll” action, which refreshes their property values. See “[Platform service types](#)”, page 122 for a listing of possible child services.

SystemService (under PlatformServices)

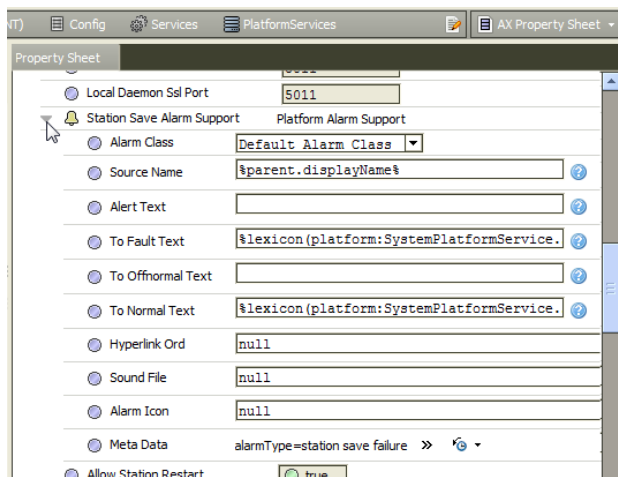
PlatformServices also contains a child “**SystemService**” container, accessible from its property sheet as shown below. Unlike other child services, **SystemService** does not appear in the Nav tree.

Figure 108 SystemService from property sheet of PlatformServices.



When you expand SystemService, you see most of the same properties available in the default Platform Service Container Plugin view (see “[PlatformServiceContainer parameters](#)” on page 89, page 115). In addition, as shown below, there is a container slot “Station Save Alarm Support”.

Figure 109 Station Save Alarm Support expanded in property sheet of SystemService.



Properties under “Station Save Alarm Support” allow you to configure the alarm class and other parameters to use for “station save” alarms. Such an alarm may occur, for example, if there is insufficient disk space to complete the save.

Properties work the same as those in an alarm extension for a control point.

NOTE:

Other platform warnings from defined limits, such as for low memory, low disk space, and so on are not really alarms—they simply generate a yellow overlay in the lower right corner when viewing the station in Workbench. If you need actual alarms, you can link from an appropriate boolean slot of the `SystemService` component (for example, “LowHeap”) into other persisted station logic in another area of the station.

If linking to PlatformServices, be aware that you should change the link type from “handle” to “slot path”. For related details, see [“PlatformServices binding and link caveats” on page 95, page 123](#).

Platform service types

In addition to the `SystemService` found under its property sheet, the `PlatformServices` Container has various child services, of which different types are listed below.

NOTE:

Some platform services are intended to support installations where *all* configuration must be done using only a browser connection (and not a Workbench platform connection to a JACE’s platform daemon). Examples include types `TcpIpService` and `LicenseService`.

The list of visible platform service types includes the following:

- `CertManagerService`
For management of PKI certificate stores and/or allowed host exceptions, used in certificate-based TLS connections between the station/platform and other hosts. For details, see the *Niagara 4 Station Security Guide*.
- `TcpIpPlatformService`
Provides access to the same configuration using the platform’s **TCP/IP Configuration** view. See [“TCP/IP Configuration” on page 81](#).
- `LicensePlatformService`
Provides access to the same configuration using the platform’s **License Manager** view. See [“License Manager” on page 46](#).
- `SerialPortService`
(JACE platforms only) Allows review of available serial ports on the host platform.
- `PowerMonitorService`
(All platforms except for JACE-8000 series) Provides configuration and status of the controller’s battery monitoring and AC power-fail shutdown routines. See [“JACE power monitoring” on page 95, page 123](#) for details.
- `NtpPlatformService`
Provides the Niagara interface to the NTP (Network Time Protocol) service or daemon of the platform’s OS (QNX or Windows), including several configuration parameters and a list specifying one or more NTP time servers. For details, see [“About the NtpPlatformService” on page 96, page 124](#).
- `DataRecoveryService`
(JACE platforms only) Allows monitoring the service that automatically creates and manages static RAM buffers in the controller, allowing “battery-less” operation (if so configured), or usage of the SRAM along with an installed backup battery (if applicable). For details, refer to the document *JACE Data Recovery Service (SRAM support)*.
- `HardwareScanService`
(JACE platforms only) Optional platform service that provides a graphical diagram of communication ports and other features on the hosting JACE platform, including callouts to a table that explain the

location, description (such as COM2), port type, and status/usage of each item. Requires installation of the modules `platHwScan` and a corresponding `platHwScanType`. Refer to the document *JACE Hardware Scan Service*.

Using platform services in a station

Apart from configuration usage, some platform services under the [PlatformServices, page 113](#) Container provide status values that you can further incorporate. Typically, each value also provides built-in alarm features. Usage is typical for the following:

- [JACE power monitoring, page 123](#)

JACE power monitoring

By default, through the **PowerMonitorService**, any JACE provides status monitoring of the following items, via “Boolean” type slots:

- AC power
 (“Primary Power Present” slot) — True whenever AC power is currently supplied to the JACE.
- Battery level
 (“Battery Good” slot) — True if last JACE test of NiMH backup-battery was good.
 Also included is a “Time of Last Test” slot that provides a timestamp for the last battery test.

If needed, you can make Px bindings or links to these slots (however, see [“PlatformServices binding and link caveats” on page 95, page 123](#)).

In addition to these read-only status slots, the **PowerMonitorService** provides related *configuration* slots, which you typically review at commissioning time. For more details and a related procedure, see “JACE power monitoring configuration” in the *JACE Niagara 4 Install & Startup Guide*.

Battery monitoring disabled

(Does not apply to a JACE-8000 series controller) An SRAM-equipped JACE can be configured for “battery-less” operation (the `platDataRecovery` module must be installed, and JACE licensed with for the “data-Recovery” feature). The **PowerMonitorService** will continue to monitor for an (optional) backup battery, and upon loss of AC power allows continuous operation on battery power until the Shutdown Delay time is reached—unless you set the “Battery Present” property (of its PlatformServiceContainer) from true (the default) to false. This disables backup battery support and prevents ongoing “battery bad” nuisance alarms—when there is no backup battery. For related details see [“PlatformServiceContainer configuration parameters” on page 90, page 117](#).

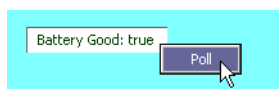
PlatformServices binding and link caveats

Because any station’s **PlatformServices** are dynamically built upon startup, if binding its slots to Px widgets (or linking to other station components), be aware of the following limitations/guidelines:

- Subscription behavior is unique to a station’s PlatformServices slots, in that property values initially load, but do *not automatically update*. To explicitly refresh such properties, you must invoke the “poll” action of the container for those properties.

For example, if on a Px page you bind a BoundLabel to the PowerMonitorService’s “Battery Good” slot, it will display text as “true” or “false.” However, this value does not update until the user right-clicks for the “Poll” action, which forces a fresh read.

Figure 110 Poll action on bound PlatformServices property

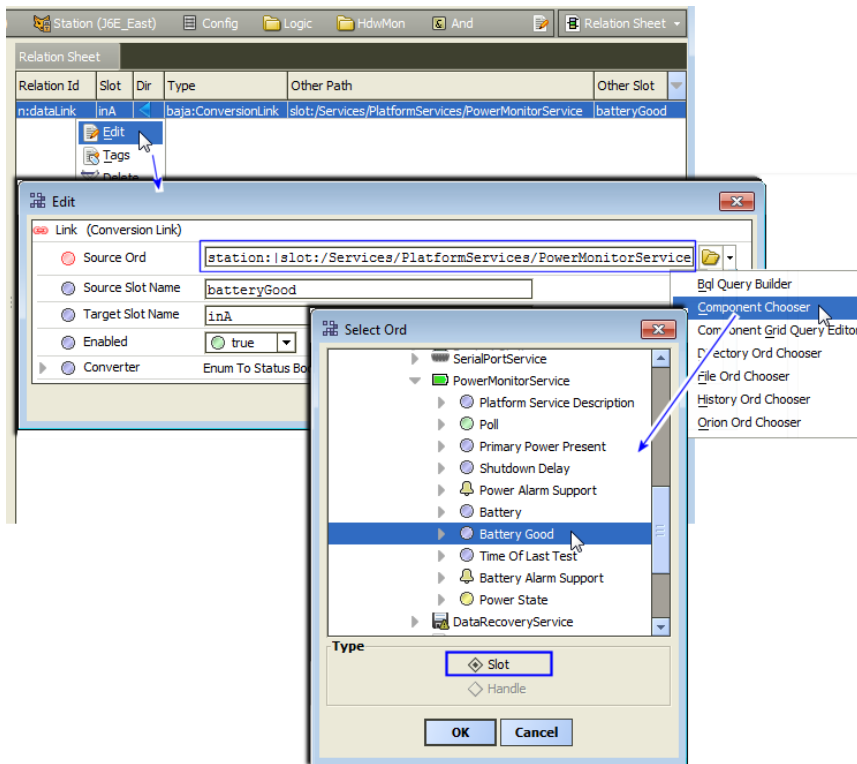


- Links from PlatformServices (and child slots) to other station components must use a source ord "*slot path*", versus "*handle*". Otherwise, after a station restart or host reboot, handle-sourced links may be lost. An example link being edited to use slot path is shown in the figure above.

NOTE:

Consider the "update limitation" before *linking* PlatformServices slots into other components that provide control logic. Linked slot values may well be outdated shortly after station startup, yet still "subscribed" and not marked as "stale."

Figure 111 From RelationSheet of target component, editing link to use slot path for source ord.



However, note that the station's plugins (views) for the **PlatformServices** *do* provide updated property values, as they work in concert with the special polling used for platform-resident data.

About the NtpPlatformService

PlatformServices in any station contains a child **NtpPlatformServicesOS**, which provides an interface to the RFC 1305-compliant NTP (Network Time Protocol) service or daemon running on that host platform. NTP is the currently recommended time synchronization protocol to use between inter-networked devices, offering more accuracy than the older RFC 868 Time Protocol.

By default, this platform service is disabled.

- If left disabled, this platform service does nothing.

NOTE:

All Windows hosts running Niagara 4 are left in this state. If NTP operation is needed, say for a Niagara 4 Supervisor, you must configure this using native Windows tools. While different than in NiagaraAX, note this is consistent with Niagara 4 platform access of Windows hosts for other administrative tasks, say for TCP/IP configuration, or setting the date, time, or time zone.

- If enabled, this platform uses NTP as a client to sync its clock with time values retrieved from one or more NTP time servers, according to other configuration properties.

See the following sections for more details:

- [About the Ntp Platform Service Editor, page 125](#)
- [NTP port/firewall considerations, page 128](#)

About the Ntp Platform Service Editor

For either platform OS type (Windows or QNX), the default view for any **NtpPlatformService** is an **Ntp PlatformService Editor OS** view, your typical interface. Double-click any **NtpPlatformService** to see this editor.

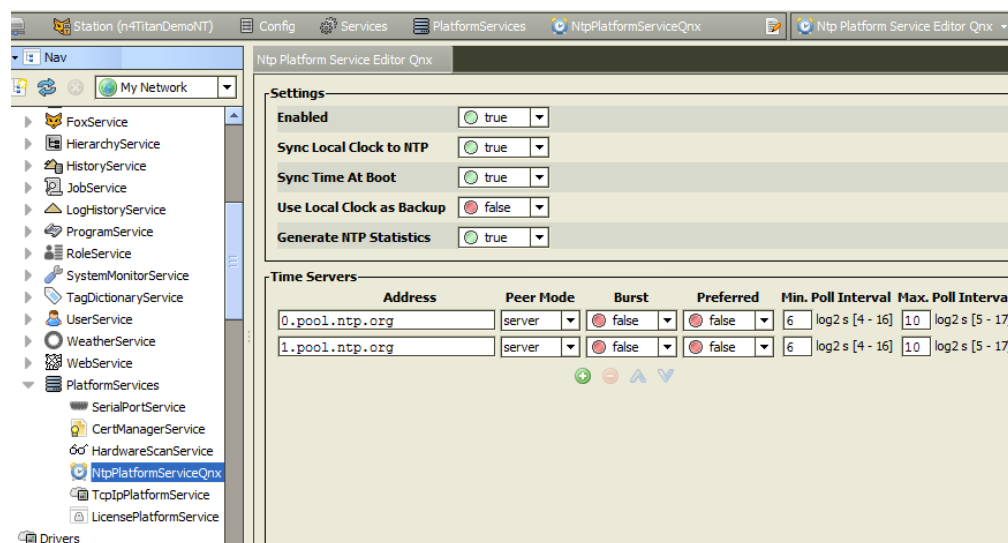
In Niagara 4 the **NtpPlatformService Editor** on any Windows platform is disabled and read-only.

- [Ntp Platform Service Editor QNX, page 125](#)
- [Ntp Platform Service Editor Win32, page 127](#)

About the Ntp Platform Service Editor Qnx

An example **Ntp Platform Service Editor** for a JACE controller is shown below. This is the default view for the **NtpPlatformServiceQnx**.

Figure 112 Ntp Platform Service Editor Qnx



This dialog provides access to some of the key settings of the NTP daemon (ntpd) of the QNX OS running on the host JACE platform.

There are two main areas: **Settings** at top, **Time Servers** at bottom. The **NtpPlatformServiceQnx** also has an available “Sync Now” action. For more details, see [“Sync Now action” on page 98, page 127](#).

Ntp Platform Service Editor Qnx settings

Settings in the **Ntp Platform Service Editor Qnx** include the following properties:

- **Enabled**
If true, the host will use NTP to sync its clock with time values retrieved from other servers.
- **Sync Local Clock to NTP**
If true, this enables the host to adjust its local clock by means of NTP. If disabled (false), the local clock free-runs at its intrinsic time and frequency offset. This flag is useful in case the local clock is controlled by some other device or protocol and NTP is used only to provide synchronization (as server) to other

clients. In this case, the local clock driver can be used to provide this function and also certain time variables for error estimates and leap-indicators.

- Sync Time At Boot

Default is false. If true, when the JACE boots, before the stations starts or the ntpd starts, it executes the `ntpdate` command. This updates the system local time.

- Use Local Clock as Backup

If true, should the specified NTP server(s) become unavailable at the time of a poll, the time used is provided by the system clock. This prevents the timing of the polling algorithm in the ntpd (which is executed at specified/changing intervals) from being reset.

A true value does not result in any change to the NTP daemon's polling interval (frequency). In fact, by using the local system clock the NTP-calculated polling time would remain the same, and

NOTE:





thus not result in more polling.

- Generate NTP Statistics

If true, the NtpPlatformService reports whatever information it can about its operation. To access these statistics with the station opened in Workbench, right-click the NtpPlatformServiceQnx and select **Views** → **SpyRemote**. Keep in mind that the ntpd is a QNX process; thus Niagara has no control over what it reports.

Ntp Platform Service Editor Qnx time servers

Each entry in the time servers list in the **Ntp Platform Service Editor Qnx** specifies a server to which the host's clock will be sync'ed when the service is Enabled (true), and "Sync Local Clock to NTP" is also true. These servers are *not* used if either of these properties are false.

Controls below the list allow you to add  and delete  servers, as well as reorder up  or down  (to establish priority order, highest at top). Fields for each time server includes the following:

- Address

Fully qualified domain name, IP address, or host files alias for the NTP time server.

- Peer Mode

Peer mode to use with the server, as either server or peer (symmetricActive).

- Burst

False by default. If true, when server is reachable, upon each poll a burst of eight packets are sent, instead of the usual one packet. Spacing between the first and second packets is about 16 seconds to allow a modem call to complete, while spacing between remaining packets is about 2 seconds.

- Preferred

If true, designates a server as preferred over others for synchronization. Note also that priority order (top highest, bottom lowest) is also evaluated if multiple servers are entered.

- Min. Poll

Minimum poll interval for NTP messages, from 4 to 16. Note units are in "log-base-two seconds," or 2 to the power of n seconds (NTP convention), meaning from 2 to the 4th (16 seconds) to 2 to the 16th (65,536 seconds).

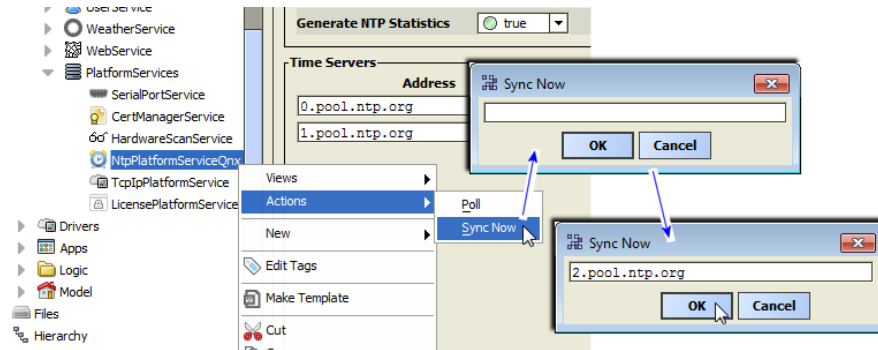
- Max. Poll

Maximum poll interval for NTP messages, from 10 to 17. Note units are in "log-base-two seconds," or 2 to the power of n seconds (NTP convention), meaning from 2 to the 10th (1,024 seconds) to 2 to the 17th (131,072 seconds).

Sync Now action

In addition to the “Poll” action present on any NtpPlatformService, the NtpPlatformServiceQnx component has an additional “Sync Now” action.

Figure 113 Sync Now action on NtpPlatformServiceQnx



As shown in Figure 113 Figure 114, page 127, this action produces a popup Sync Now dialog, which is blank.

To use, type in the fully qualified domain name of a public NTP server (as shown above), or else the IP address of any accessible NTP server, and then click OK.

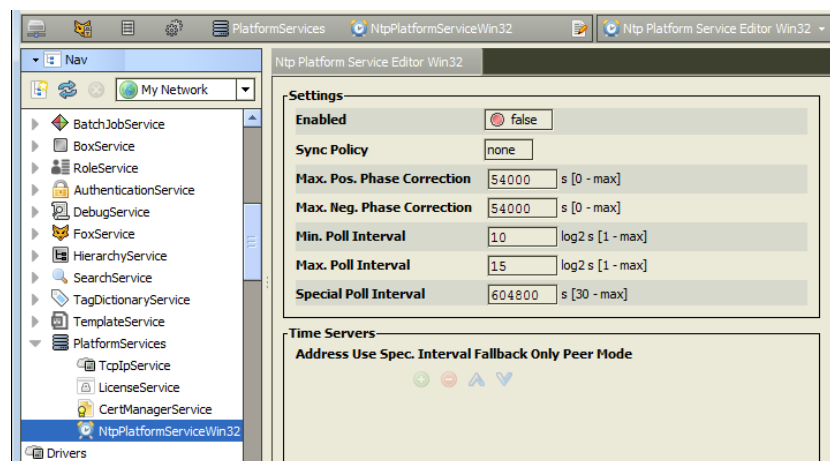
To verify, look for a related entry in the station’s spy “platform diagnostics” log. Do this in Workbench by right-clicking the station, then selecting **Spy** → **platform diagnostics** → **log** or from the Workbench File menu, **File** → **Open ord** (Ctrl + L) and enter:

```
ip:JACE_IP_address|fox:|spy:/platform diagnostics/log
```

About the Ntp Platform Service Editor Win32

An example Ntp Platform Service Editor Win 32 is shown below. This is the default view for the NtpPlatformService on a Windows-based (Win32 or Win64) host.

Figure 114 Ntp Platform Service Editor Win32



NOTE:

In Niagara 4 this service remains disabled and read-only. Settings shown are only a small subset of those possible to configure—if needed to configure this time service, Windows registry settings can be set according to Microsoft’s latest instructions. Visit the Microsoft tech support site for more information on a particular Windows OS, searching on the “NTP Time service”.

NTP port/firewall considerations

On any host, NTP requires the use of UDP port 123—this port is not configurable. On a JACE platform this is not an issue.

However, on a Windows host platform, in addition to configuring NTP using Windows native tools, typically you need to make the necessary firewall exception or “iptables” entry to allow UDP port 123 traffic. Otherwise, NTP time synchronization can fail because of firewall-blocked messages.

Chapter 4 Platform Component Guides

Topics covered in this chapter

- ◆ Components in platCrypto
- ◆ Components in platDataRecovery module
- ◆ Components in platform module
- ◆ Components in platHwScan
- ◆ Components in platPower module
- ◆ Components in platSerialQnx module

Component Guides provides summary information on platform-related components.

Components in platCrypto

- [CertManagerService, page 129](#)
- [DaemonSecureSession, page 129](#)


platCrypto-CertManagerService



The **CertManagerService** is a platform service of any Niagara 4 station. It has few visible properties, but provides a default **Certificate Management** view, equivalent to that same-named platform view.

The **Certificate Management** view provides the means to import and export signed certificates (for TLS secure connections) into the platform's key store and trust store, and to perform other related functions. For complete details, refer to the document *Niagara 4 Station Security Guide*.

platform-DaemonSecureSession

 DaemonSecureSession represents a *secure* platform connection to a host made in Workbench. In the Nav tree view, the platform session icon is labeled **Platform**, shows a small padlock, and is directly under the host to which the platform session is in progress. To support such connections, the host must have its "Platform TLS Settings" enabled (accessed in its **Platform Administration** view).

As in a regular (un-encrypted) platform connection, the default view is the **Nav Container View**, which provides a table of all the various platform views. For related details, see "[Niagara platform](#)" on page 11, including "[Platform overview](#)" on page 12 and "[About a platform connection](#)" on page 12.

NOTE:

For details on TLS connections, refer to the document *Niagara 4 Station Security Guide*.

Components in platDataRecovery module

- [DataRecoveryService, page 129](#)

platDataRecovery-DataRecoveryService



The DataRecoveryService automatically creates and manages buffers in a JACE controller's available SRAM (Static RAM), allowing "battery-less" operation. It applies to most recent JACE controller models with integral SRAM, (JACE-8000, JACE-3E, JACE-6E, JACE-603, JACE-645) as well as a JACE-6 or JACE-7 controller with an installed "SRAM option card".

Some SRAM-equipped JACE controller can additionally (and optionally) use a backup battery—such as an NiMH onboard battery pack, and (if applicable) and external 12V sealed lead-acid battery. In this case, both the **DataRecoveryService** and **PowerMonitorService** can run in the station's PlatformServices container, operating independently or in unison, as configured.

For details, see the "About the DataRecoveryService" section in the document *JACE Data Recovery Service (SRAM support)*.

Components in platform module

- [DaemonFileSpace](#), page 130
- [DaemonSession](#), page 130
- [LicensePlatformService](#), page 130
- [NtpPlatformServiceQnx](#), page 131
- [NtpPlatformServiceWin32](#), page 131
- [PlatformAlarmSupport](#), page 131
- [PlatformServiceContainer](#), page 131
- [SystemPlatformServiceQnxJavelina](#), page 131
- [SystemPlatformServiceQnxNpm6xx](#), page 132
- [SystemPlatformServiceWin32](#), page 132
- [TcplpPlatformService](#), page 132

platform-DefaultDaemonFileSpace



The **Remote File System** (DefaultDaemonFileSpace) represents the files accessible for read-only access when platform-connected to a remote Niagara host. For more details, see "[Remote File System](#)" on page 86.

platform-DaemonSession



DaemonSession represents a platform connection to a host made in Workbench. In the Nav tree view, the DaemonSession icon is labeled **Platform**, and is directly under the host to which the platform session is in progress.

The default view is the **Nav Container View**, which provides a table of all the various platform views. For more details, see "[Niagara platform](#)" on page 11.

platform-LicenseDatabaseTool



The LicenseDatabaseTool (Local License Database) represents your Workbench PC's "local license database." The default view is the Workbench License Manager, which allows you to manage locally-stored licenses. For more details, see "[Workbench License Manager](#)" on page 112.

platform-LicensePlatformService



The LicensePlatformService provides station access to the host platform's license(s) and certificate(s). This service is found under the running station's **PlatformServices** container. From the default plugin (view), you can perform the same operations as from the License Manager view using a platform connection. For more details, see ["License Manager" on page 46](#).

platform-NtpPlatformServiceQnx



The NtpPlatformServiceQNX is the Niagara interface to the NTP (Network Time Protocol) daemon of the QNX OS running on a JACE. If enabled, it provides client and server support for NTP. The default view of this platform service is the **Ntp Platform Service Editor Qnx** plugin, in which you can adjust a few settings, as well as specify time servers.

For more details, see ["About the NtpPlatformService" on page 96](#).

platform-NtpPlatformServiceWin32



The NtpPlatformServiceWin32 is the Niagara interface to the Windows Time service (W32Time) on a Windows platform. This Windows service uses the SNTP (Simple Network Time Protocol) to synchronize to one or more designated time servers. The default view of this platform service is the **Ntp Platform Service Editor Win32** plugin.

In Niagara 4, *this platform service remains disabled and read-only*. For more details, see ["About the NtpPlatformService" on page 96](#).

platform-PlatformAlarmSupport



PlatformAlarmSupport is a container slot that appears for each alarmable value under a Platform Service, such as the **PowerMonitorService** for many JACE controllers.

For a JACE platform, example PlatformAlarmSupport components include:

- Battery Alarm Support
To configure how "low battery level" alarms are handled in the station.
- Power Alarm Support
To configure how "AC power loss" alarms are handled in the station.

Properties under each PlatformAlarmSupport container are used to designate the station's Alarm Class to be used, and also to populate the alarm record when the specific alarm occurs. These properties work in the same fashion as those in an alarm extension for any control point.

platform-PlatformServiceContainer



PlatformServiceContainer (**PlatformServices**) provides a container for a station's PlatformService instances. The **Platform Service Container Plugin** is its primary view. The PlatformServiceContainer is available when online with any running station, under its Config, **Services** folder. For more details, see ["Platform Services" on page 87](#) and ["PlatformServiceContainer parameters" on page 89](#).

platform-SystemPlatformServiceQnxJavelina



SystemPlatformServiceQnxJavelina (**SystemService**) is the QNX implementation of SystemPlatformService in a station running on a JVLN-based (JACE-700) controller. For more details, see [“SystemService \(under PlatformServices\)” on page 93](#).

platform-SystemPlatformServiceQnxNpm6xx



SystemPlatformServiceQnx (**SystemService**) is the QNX implementation of SystemPlatformService in a station running on a JACE controller. For more details, see [“SystemService \(under PlatformServices\)” on page 93](#).

platform-SystemPlatformServiceWin32



SystemPlatformServiceWin32 (**SystemService**) is the Win32 implementation of SystemPlatformService. For more details, see [“SystemService \(under PlatformServices\)” on page 93](#).

platform-TcplpPlatformService



TcplpPlatformService provides station access to the host platform’s TCP/IP settings. This service is found under the running station’s **PlatformServices** container. From the default plugin (view), you can perform the same operation as from the **TCP/IP Configuration** view using a platform connection. For more details see [“TCP/IP Configuration” on page 81](#).

Note in Niagara 4, for any Windows platform this view is read-only.

Components in platHwScan

platHwScan-HardwareScanService



The **Hardware Scan Service** is an available platform service on a JACE station, providing that the JACE platform has the `platHwScan` module installed.

To function correctly, the appropriate `platHwScanType` module also needs to be installed on the JACE. Otherwise, the default **Hardware Scan Service View** will simply display:

Jar file `platHwScanType` is required to support this platform

Where the appropriate `platHwScanType` is as follows:

Controller Series	platHwScanType module
JACE-6E, JACE-6, JACE-3E	platHwScanNpm
JACE-7 (700)	platHwScanJvln
JACE-603 (JACE-403 with retrofit board)	platHwScanJ603
JACE-645 (JACE-545 with retrofit board)	platHwScanJ645
JACE-602 Express (J-602-XPR or M2M)	platHwScanXpr
JACE-8000	platHwScanTitan

This default **Hardware Scan Service View** provides a diagram of the controller that shows its communication ports and other features (including, if applicable, installed communication options such as modules or cards). The diagram has callouts to a table that explains each item's location, description (such as COM2), port type, usage, and status.

NOTE:

For complete details, refer to the document "JACE Hardware Scan Service".

Components in platPower module

- [ExternalSlaBattery](#), page 133
- [JavelinaBatteryPlatformService](#), page 133
- [Npm2NimhBattery](#), page 134
- [NpmDualBatteryPlatformService](#), page 134
- [NpmExternalSlaBattery](#), page 134
- [PowerMonitorPlatformServiceQnx](#), page 134

platPower-ExternalSlaBattery

ExternalSlaBattery is one of two "Battery" slots in the **JavelinaBatteryPlatformService** in a JACE-700 (JACE-7 series) controller station's PlatformServices container. This slot indicates the host JACE platform can use an optional, sealed-lead acid (SLA) battery, *in addition to* the onboard NiMH backup battery.

platPower-JavelinaBatteryPlatformService



JavelinaBatteryPlatformService (PowerMonitorService) applies to a station running in a JACE-700 (JACE-7 series) controller. It can monitor primary power status and backup battery levels in *both* the onboard 12V NiMH battery and an optional 12V sealed-lead acid (SLA) battery. In addition, it can monitor alarm contacts of an external, customer-supplied UPS— if enabled and wired to the two corresponding onboard contact inputs (CIs) of the controller. Note the JACE-7 controller has three onboard CIs, with the intended use for UPS AC power lost, UPS low battery, and (door) tamper switch.

NOTE:

The tamper switch CI on the JACE-7 controller is enabled/monitored by two properties in the PowerMonitorService's parent **PlatformServices** Container).

Configuration properties in this PowerMonitorService allow changing the shutdown delay time, and also specifying whether external equipment is connected (12V SLA battery, UPS). Separate alarm source configuration properties are available for all five types of alarms (low NiMH battery level, low SLA battery level, primary power lost, UPS AC power lost, UPS low battery).

Typically, support is enabled and configured at JACE *commissioning time*. For related details, see "JACE power monitoring configuration" in the latest *JACE Niagara 4 Install & Startup Guide*.

platPower-NimhBattery

NimhBattery is a "Battery" container slot under the PowerMonitorService in a JACE-700 (JACE-7 series) station's PlatformServices container. This slot indicates the host JACE platform uses a nickel-metal hydride (NiMH) battery. Included are two status properties that show the current "State" (Idle, Charging, Discharging, Unknown) and "Charge Time Left" (in hours and minutes, if state is charging).

platPower-Npm2NimhBattery

● Npm2NimhBattery is a “Battery” container slot under the PowerMonitorService in a JACE-2/6 series or JACE-x02 Express (NPM2 or NPM6) station’s PlatformServices container. This slot indicates the host JACE platform uses a nickel-metal hydride (NiMH) battery. Included are two status properties that show the current “State” (Idle, Charging, Discharging, Unknown) and “Charge Time Left” (in hours and minutes, if state is charging).

This slot also appears in the **NpmDualBatteryPlatformService** (“dual battery” PowerMonitorService) of a JACE that is capable and enabled for dual battery support.

platPower-NpmDualBatteryPlatformService



NpmDualBatteryPlatformService (PowerMonitorService) applies to a station running in a JACE platform that is capable and enabled for “dual battery” support, such as a JACE-x02 Express series (M2M JACE). It is used to monitor primary power status and backup battery levels in *both* the onboard NiMH battery as well as the optional sealed-lead acid (SLA) battery. A few configuration parameters allow changing the shutdown delay time, as well as alarm source configuration for all three types of alarms (low NiMH battery level, low SLA battery level, primary power lost).

Typically, support is enabled and configured at JACE *commissioning time*. For related details, see “JACE power monitoring configuration” in the latest *JACE Niagara 4 Install & Startup Guide*.

platPower-NpmExternalSlaBattery

● NpmExternalSlaBattery is one of two “Battery” slots under the **NpmDualBatteryPlatformService** in a “dual battery enabled” JACE’s station’s PlatformServices container. This slot simply indicates the host JACE platform can use an optional, sealed-lead acid (SLA) battery, *in addition to* the onboard NiMH backup battery.

platPower-PowerMonitorPlatformServiceQnx



PowerMonitorPlatformServiceQnx (**PowerMonitorService**) is used to monitor the primary power status and backup battery level in many JACE controllers. A few configuration parameters allow changing the shutdown delay time, as well as alarm source configuration for both types of alarms (low battery level, primary power lost).

This PowerMonitorService is found under the **PlatformServices** container in a station running on many JACE controllers *except* for those models that are capable and/or enabled for “dual battery” support. Typically, support is enabled and configured at JACE *commissioning time*. For related details, see “JACE power monitoring configuration” in the latest *JACE Niagara 4 Install & Startup Guide*.

Components in platSerialQnx module

- [SerialPortPlatformServiceQnx](#), page 134
- [SerialPortQnx](#), page 135

platSerialQnx-SerialPortPlatformServiceQnx



SerialPortPlatformServiceQnx is the station’s interface to the platform’s serial port configuration, such as used by a JACE-3,-6,-7 series host. This service is found under the running station’s **PlatformServices** container as the **SerialPortService**.

platSerialQnx-SerialPortQnx

SerialPortQnx contains properties that describe how a serial port (RS-232 or RS-485) on a JACE controller is being used in software as COMn. Each one is a child of that JACE's SerialPortService (**SerialPortPlatform-ServiceQnx**). Properties are as follows:

- **Owner** — The driver network or function currently associated with that COM port, for example, "Nrio-Network", "dialup", "none", "ModbusAsyncNetwork", or "dbgjmp" (latter indicated for COM1 when "serial shell" jumper is installed on JACE).
- **Os Port Name** — How the port is known to the QNX OS and associated low-level drivers.
- **Port Index** — Unique serial port index number, starting with 1 for COM1.

Chapter 5 Platform Plugin Guides

Topics covered in this chapter

- ◆ Plugin Reference Summary
- ◆ Plugins in platCrypto
- ◆ Plugins in platDaemon module
- ◆ Plugin in platDataRecovery
- ◆ Plugins in platform module
- ◆ Plugins in platHwScan
- ◆ Plugins in platPower

There are many ways to view plugins (*views*). One way is directly in the tree. In addition, you can right-click on an item and select one of its views. Plugins provide views of components.

Access the following summary descriptions on any plugin by selecting **Help**→→**On View** (F1) from the menu, or by pressing F1 while the view is open.

Plugin Reference Summary

Summary information is provided on views in the following modules:

- [platCrypto, page 137](#)
- [platDaemon, page 137](#)
- [platDataRecovery, page 140](#)
- [platform, page 141](#)
- [platHwScan, page 142](#)
- [platPower, page 143](#)

Plugins in platCrypto

- [Certificate Management, page 137](#)

platCrypto-CertManagerView



The **Certificate Management** view is a platform view on any Niagara 4 host. It is also the default view of the **CertManagerService** under a station's **PlatformServices**.

The **Certificate Management** view provides the means to import and export signed certificates (for TLS secure connections) into the platform's key store and trust stores, and to perform other related functions. For a brief overview in this document, see "[Certificate Management](#)" on page 35.

NOTE:

Niagara 4 Workbench also provides a similar view, via **Tools**→→**Certificate Management**. Also included is a related **Tools**→→**Certificate Signer Tool** view.

For complete details, refer to the document *Niagara 4 Station Security Guide*.

Plugins in platDaemon module

- [Application Director, page 138](#)

- [Certificate Management, page 137](#)
- [Distribution File Installer, page 138](#)
- [Distribution View, page 138](#)
- [File Transfer Client, page 138](#)
- [Lexicon Installer, page 138](#)
- [License Manager, page 139](#)
- [Software Manager, page 139](#)
- [Software View, page 140](#)
- [PlatformAdministration, page 140](#)
- [Station Copier, page 140](#)
- [TCP/IP Configuration, page 140](#)

platDaemon-ApplicationDirector



The Application Director is the platform view that allows you to start, stop, restart, and kill a station on the connected Niagara platform. You also use it to examine standard *station output*, for troubleshooting and debug purposes. For more details, see [“Application Director” on page 28](#).

platDaemon-DistInstaller



The platform **Distribution File Installer** allows you to install distribution (.dist) files from your Workbench PC to the remote Niagara host. Typical use is for restoring backups, or for installing a “clean dist” file to essentially wipe clean the filesystem of a Niagara 4 JACE controller. For more details, see [“Distribution File Installer” on page 37](#).

platDaemon-DistributionView



Distribution View is the dialog that appears when you double-click a distribution file listed in the platform’s **Distribution File Installer** view. A number of details is provided about the selected distribution file, including all contents and any dependencies.

platDaemon-FileTransferClient



The **File Transfer Client** is the platform view that allows you to *copy* files and/or folders between your Workbench PC and the remote Niagara platform, as needed. For more details, see [“File Transfer Client” on page 44](#).

platDaemon-LexiconInstaller



Lexicon Installer allows you to install text-based lexicon file sets (for Niagara localization) on a remote host. For more details, see [“Lexicon Installer” on page 45](#).

NOTE:

Standard lexicons in Workbench are distributed as *modules*, for example: `niagaraLexiconFr` as the French lexicon, or `niagaraLexiconDe` for German. Workbench lexicon tools include a lexicon module maker, to make new or updated lexicon modules from lexicon files.

You can still install lexicon *files* using the **Lexicon Installer**, but to install lexicon *modules* you must use the platform **Software Manager** view. For more details, see the *Niagara Lexicons Guide*.

platDaemon-LicenseManager



The **License Manager** allows you to view and install files required for Niagara licensing. For more details, see “[License Manager](#)” on page 46.

Network License Summary

This view provides a summary table listing the currently known license information for each station (**NiagaraStation**) in the network. It is the default view for the **SupervisorLicenses** slot on the **ProvisioningNwExt** under the Supervisor’s **NiagaraNetwork**.

Figure 115 Network Licenses Summary

Station	Host ID	Status	Last Updated
J6wSed_43 {ok}	Qnx-NPM6-0000-13A3-0D21	Up To Date	26
eSup_Mobile {ok}	Qnx-NPM6E-0000-153C-6E44	Up To Date	26
J7_Bnet_36 {ok}	Qnx-JVLN-0000-00F7-6310	Out Of Date	24
J202_TestW {ok}	Qnx-NPM2-0000-12C5-DD2C	Up To Date	26
J600E_T1 {ok}	Qnx-NPM6E-0000-153C-7BE2	Unknown	

Each row contains the license information for a host running a station. The **SupervisorLicenses** device extension of each child station populates the table. If you double-click on a row, the view changes to the **SupervisorLicenses** extension property sheet for that particular **NiagaraStation**.

Column	Value	Description
Station	text	Identifies the name of the station.
Host ID	text	A 20-character identifier that provides unique identification for each host.
Status	Up-To-Date	A status of <code>Up To Date</code> means that the license on the remote host agrees with the license that the Supervisor has for it in its (own) local license database. It may be possible that a more recent license is available for it on the licensing server.
Last Updated	date and time	The timestamp when the station’s license was last updated.

platDaemon-SoftwareManager



The **Software Manager** is the platform view you use to install, upgrade, or remove modules in the connected Niagara platform. For more details, see [“Software Manager” on page 65](#).

platDaemon-SoftwareView



Software View is the dialog that appears when you double-click an item (for example, module) listed in the platform’s **Software Manager** view. A number of details is provided about the selected item.

platDaemon-PlatformAdministration



The **Platform Administration** view provides access to various platform daemon (and host) settings and summary information. Included are buttons to perform various platform operations. For more details, see [“Platform Administration” on page 51](#).

platDaemon-StationCopier



The **Station Copier** is the platform view used to install a station in either a remote or local Niagara platform, as well as make a local Workbench copy of a remote Niagara JACE station or a locally running station. You can also delete and rename stations using this view. For more details, see [“Station Copier” on page 73](#).

platDaemon-StationTextSummaryEditor



StationTextSummaryEditor is the dialog that appears when you click the export tool button when using the [Application Director](#) view. Setup in this dialog allows you to include/exclude the platform summary data, platform daemon console output, station console output, as well as limit both the daemon and station output.

platDaemon-TcplpConfiguration



TCP/IP Configuration is the platform view you use to configure a remote JACE host’s TCP/IP settings. Typically, you make initial settings when you first commission a JACE for Niagara, where this view is one step in the platform’s Commissioning Wizard. Note for Windows platforms, this view is read-only in Niagara 4. For more details, see [“TCP/IP Configuration” on page 81](#).

Plugin in platDataRecovery

- [Data Recovery Service Editor, page 140](#)

platDataRecovery-DataRecoveryServiceEditor



The Data Recovery Service Editor is the default view on the **DataRecoveryService**, as found in the PlatformServices of JACE controllers with onboard static RAM (SRAM or FRAM), or an installed SRAM option card.

This view allows monitoring of the “battery-less” support provided by this service. In a few cases, an SRAM-equipped JACE can additionally (and optionally) use a backup battery—such as an NiMH onboard battery pack, and (if applicable) and external 12V sealed lead-acid battery. In this case, both the

DataRecoveryService and **PowerMonitorService** can exist in the station's PlatformServices container, operating independently or in unison, as configured.

For details, see the "About the DataRecoveryService" section in the document *JACE Data Recovery Service (SRAM support)*.

Plugins in platform module

- [License Platform Service Plugin, page 141](#)
- [Ntp Platform Service Editor Qnx, page 141](#)
- [Ntp Platform Service Editor Win32, page 141](#)
- [Platform Service Container Plugin, page 141](#)
- [Platform Service Properties, page 141](#)
- [System Platform Service Plugin, page 142](#)
- [System Platform Service Qnx Plugin, page 142](#)
- [TcpIp Platform Service Plugin, page 142](#)

platform-LicensePlatformServicePlugin



License Platform Service Plugin allows you to manage the host's licenses and certificates under a station's **PlatformServices** container. It provides the same interface as the **License Manager** view in a platform connection. See "[License Manager](#)" on page 46.

platform-NtpPlatformServiceEditorQnx



Ntp Platform Service Editor Qnx is the default view of the station's **NtpPlatformServiceQnx**, which provides the platform interface to the NTP daemon (process) running on a JACE controller. This view provides access to a few related settings, plus allows specifying one or more remote time servers. For more details, see "[About the Ntp Platform Service Editor Qnx](#)" on page 97.

platform-NtpPlatformServiceEditorWin32



Ntp Platform Service Editor Win32 is the default view of a Windows platform station's **NtpPlatformServiceWin32**, which provides the platform interface to the Windows Time service (W32Time) on the host platform's Windows OS. In Niagara 4, this platform service *remains disabled and read-only*. For details, see "[About the Ntp Platform Service Editor Win32](#)" on page 99.

platform-PlatformServiceContainerPlugin



The **Platform Service Container Plugin** allows you to view and edit platform parameters on the host running the opened station. It is the default view for a station's **PlatformServices** container. For related details, see "[PlatformServiceContainer parameters](#)" on page 89.

platform-PlatformServiceProperties



PlatformServiceProperties allows you to view and edit platform parameters on the host running the opened station, using a property sheet. See ["PlatformServiceContainer parameters" on page 89](#).

platform-SystemDateTimeEditor



As an available view on a station's **PlatformServices** container, the **System Date Time Editor** allows you to set the date, time, and time zone for the JACE platform running the station. If the station is running on a Windows platform, this view is read-only. For related time zone details, see ["Selecting a time zone in Niagara" on page 130](#).

platform-SystemPlatformServicePlugin



System Platform Service Plugin allows you to view and edit platform parameters on a Windows based host running the station, and is the default view on the station's **SystemService** (SystemPlatformServiceWin32). See ["SystemService \(under PlatformServices\)" on page 93](#).

platform-SystemPlatformServiceQnxPlugin



System Platform Service Qnx Plugin allows you to view and edit platform parameters on a JACE platform running the station, and is the default view on the station's **SystemService** (SystemPlatformServiceQnx). See ["SystemService \(under PlatformServices\)" on page 93](#).

platform-TcpIpPlatformServicePlugin



Tcp Ip Platform Service Plugin allows you to manage the host's TCP/IP settings under a station's **PlatformServices** container. It provides the same interface as the **TCP/IP Configuration** view in a platform connection. If the station is running on a Windows platform, this view is read-only. See ["TCP/IP Configuration" on page 81](#).

platform-WorkbenchLicenseManager



Workbench License Manager allows you to browse and manage the contents of your Workbench PC's "local license database." For more details, see ["Workbench License Manager" on page 112](#).

Plugins in platHwScan

platHwScan-HardwareScanServiceView



The **Hardware Scan Service View** is the default view on the platform service **HardwareScanService** in a station, providing that the JACE platform has the `platHwScan` module installed, along with the appropriate `platHwScanType` module. This view provides a graphical diagram of communication ports and other features on the hosting JACE platform, including callouts to a table that explain the location, description (such as COM2), port type, and status.

NOTE:

For complete details, refer to the document ["JACE Hardware Scan Service"](#).

Plugins in platPower

platPower-JavelinaBatteryPlatformServicePlugin



The **Javelina Battery Platform Service Plugin** is the default view on the platform service PowerMonitorService in a JACE-7 (700) series controller. This view provides parameters for changing the shutdown delay time, as well as alarm source configuration settings. For related details in this document, see [“JACE power monitoring” on page 95](#).

Typically, support is enabled and configured at JACE *commissioning time*. For related details, see “JACE power monitoring configuration” in the latest *JACE Niagara 4 Install & Startup Guide*.

platPower-PowerMonitorPlatformServicePlugin



The **Power Monitor Platform Service Plugin** is the default view on the platform service PowerMonitorService in most JACE controller models. This view provides parameters for changing the shutdown delay time, as well as alarm source configuration settings. For related details in this document, see [“JACE power monitoring” on page 95](#).

Typically, support is enabled and configured at JACE *commissioning time*. For related details, see “JACE power monitoring configuration” in the latest *JACE Niagara 4 Install & Startup Guide*.

Chapter 6 License Tools and Files

Topics covered in this chapter

- ◆ Workbench License Manager
- ◆ Request License
- ◆ About the local license database
- ◆ About license archive (.lar) files
- ◆ About Niagara license files
- ◆ Global capacity licensing

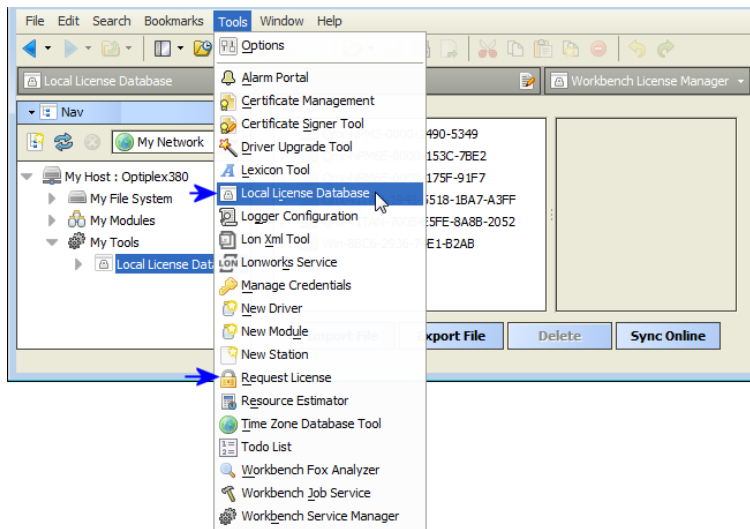
This appendix provides details about the Workbench tools related to Niagara license files, including license-management. Also included are details on the *contents* of license files.

The following subsections are included:

- License-related Workbench tools

Unlike platform views (which require a platform connection), or equivalent **PlatformServices** plugin views (requiring a station connection), Workbench tools are available whenever running full Workbench. Find Workbench tools on the **Tools** menu, as shown below.

Figure 116 Tools menu in Niagara 4 Workbench includes licensed-related tools



License-related tools include:

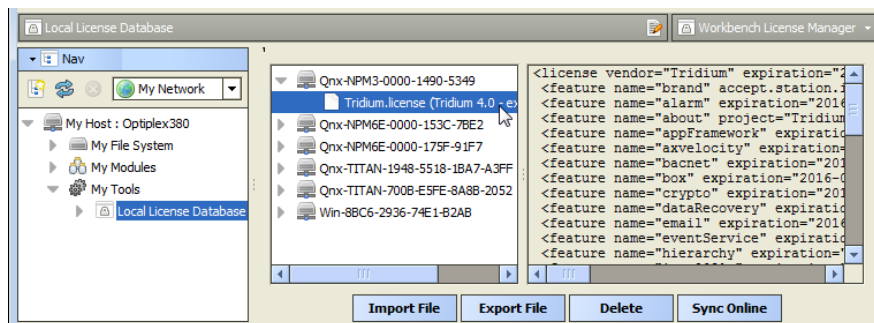
- **Workbench License Manager** tool (see [“Workbench License Manager”](#) on page 112, page 146)
- **Request License** tool (see [“Request License”](#) on page 115, page 149)
- License management topics (in addition to Workbench License tools):
 - [“About the local license database”](#) on page 116, page 150
 - [“About license archive \(.lar\) files”](#) on page 117, page 152
- [“About Niagara license files”](#) on page 118, page 152
 - [“Items common to all license files”](#) on page 118, page 153
 - [“JACE hardware features”](#) on page 119, page 154
 - [“Driver attributes”](#) on page 120, page 155

- “Driver types” on page 121, page 156
- “Applications” on page 123, page 159
- “Global capacity licensing” on page 125, page 161
 - “Capacity licensing operation and recount” on page 125, page 162
 - “Checking capacity licensing status” on page 125, page 162
 - “Capacity licensing fault notifications” on page 127, page 164
 - “Capacity licensing notes about histories” on page 128, page 165

Workbench License Manager

The **Workbench License Manager** view is available via the Workbench Tools menu, by selecting **Local License Database**.

Figure 117 Workbench License Manager



As shown above, this view lets you browse and manage the contents of your “local license database.”

NOTE:

For details about the license database structure, see [“About the local license database” on page 116, page 150](#).

This view provides a two-pane window into all the license files and parent “host ID” folders, where

- Left pane provides tree navigation, where you can expand folders and click (to select) license files.
- Right pane shows the text contents of any selected license file.

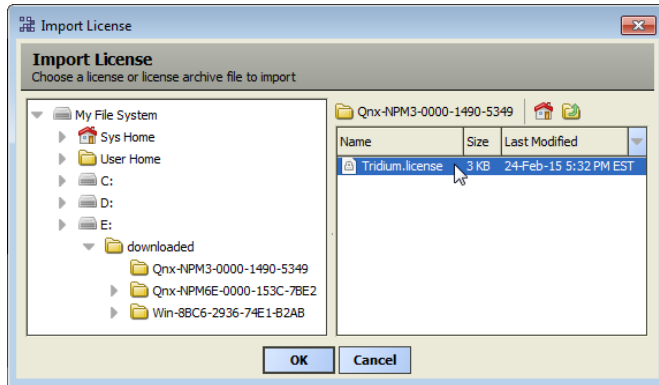
Buttons at the bottom of this view provide a way to manage the contents of your local license database, and are described as follows:

- [Import File, page 146](#) — Always available, this allows you to add license file(s) from a local license file or license archive (.lar) file.
- [Export File, page 147](#) — Always available, this allows you to save all licenses (or any selected licenses) locally, as a license archive file.
- [Delete, page 148](#) — This allows you to delete licenses from your Workbench local license database.
- [Sync Online, page 148](#) — Typically available if you have Internet connectivity. This lets you *update* all licenses (or any selected licenses) in your local license database with the *most current* versions, via the on-line licensing server.

Import File

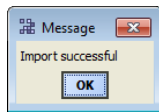
The **Import File** button in the **Workbench License Manager** is always enabled, and produces the **Import License** dialog for you to navigate to a source file (.license or .lar), as shown below. Note only these two types of files appear for selection.

Figure 118 Import License dialog to find local license file or license archive file



Select a license file and click **OK** to add to (or update in) your local license database. As shown below, a pop-up dialog informs you of success, and the license(s) are added or updated in your database.

Figure 119 Import success

**NOTE:**

If any of the license(s) you select to import are *older* than the ones currently in your local database, meaning that the “generated” attribute timestamp is earlier, newer license(s) in your local license database are *not* overwritten. However, the same “Import successful” message popup appears for such file import operations.

Export File

The **Export File** button in the **Workbench License Manager** allows you to save any number (or all) licenses in your local license database locally on your Workbench PC, as a *license archive* (.lar) file.

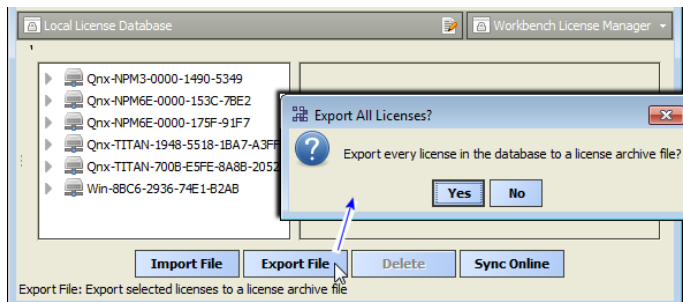
NOTE:

The license archive format allows you to *easily share* saved .lar files (however named) among multiple Workbench PCs without overwriting a license file for a different host platform. You can use the “Import File” command in the **Workbench License Manager** to add/update licenses in a license archive, or the equivalent “Import” command from the platform **License Manager** (or similar License Platform Service Plugin).

For more details, see [“About license archive \(.lar\) files” on page 117, page 152.](#)

If you click **Export File** without first selecting any licenses (and/or) host IDs, every license in your local license database will be included in the archive, as noted in a confirmation dialog. See below.

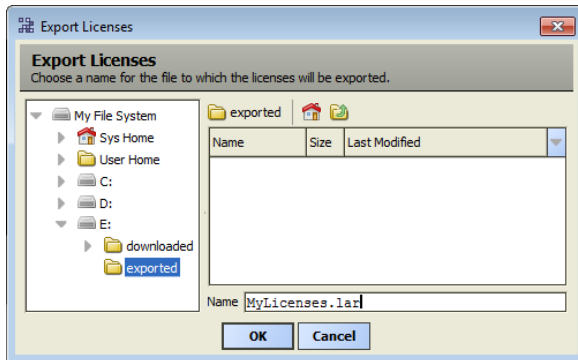
Figure 120 Export All Licenses confirmation dialog



Or, you can select one or more entries in the left pane (host IDs or license files) to include only those selected (highlighted) licenses to be in the exported archive file.

When you click **Yes** (if all) or **Export File** for selected licenses, an **Export Licenses** dialog lets you navigate to the spot to save the .lar file, as shown below.

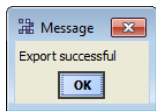
Figure 121 Export Licenses dialog



Use the dialog's navigation controls to specify another target folder or drive, as needed. Before saving, you can also *rename* the license archive file, to make it more identifiable. For example, instead of: licenses.lar, you could rename it MyJace6s.lar.

Upon export of license(s) to a license archive file, a popup dialog appears, as shown below.

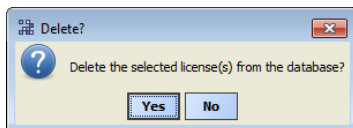
Figure 122 Export file success



Delete


The **Delete** button in the **Workbench License Manager** is enabled when you have one or more host IDs and/or license files selected in the left pane, and produces a confirmation dialog to delete licenses from your local license database, as shown below.

Figure 123 Delete licenses confirmation



Click **Yes** to delete the license(s), or **No** to leave the local license database unchanged.

NOTE:

Following a delete, you may need to click the  Refresh button in order to update the left pane contents. Note that if the selected "host ID" folder contained only a .license file, the entire folder is removed with a delete. However, if the folder contained other files (or subfolders), only the .license file is actually deleted, but it will no longer appear in the left pane.

Sync Online

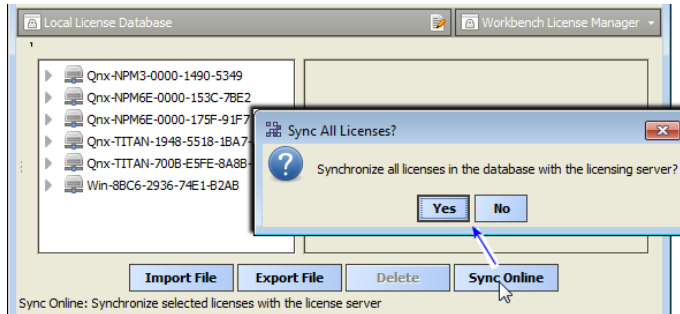
The Sync Online feature in the **Workbench License Manager** allows you to update any number (or all) licenses in your local license database with the most current license, available *online* from the *licensing server*. This feature requires Internet connectivity from your Workbench PC.

NOTE:

For related details, see [“About the licensing server”](#) on page 49.

If you click **Sync Online** without first selecting any licenses (and/or) host IDs, every license in your license database will be included in the sync request, as noted in a confirmation dialog. See below.

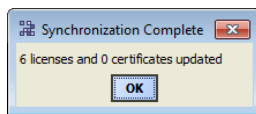
Figure 124 Sync All Licenses confirmation dialog



Or, you can select one or more entries in the left pane (host IDs or license files) to include only those selected (highlighted) licenses to be included in the sync request.

When you click **Yes** (if all) or **Sync Online** for selected licenses, an immediate request is sent to the licensing server. Intermediate popup dialogs may briefly appear while the sync request is handled. The operation concludes with a **Synchronization Complete** dialog, which summarizes the number of licenses and certificate files that were updated in your Workbench local license database. See below.

Figure 125 Synchronization Complete dialog



If all licenses (and certificates) were already up-to-date, this dialog will say “0 licenses and 0 certificates updated”.

Request License

The **“Request License”** item on the Workbench Tools menu simply opens a **“Bind License form”** in your Workbench PC’s default browser. By default, the only pre-filled field in this form is the host ID of your Workbench PC. See below.

Figure 126 License request form in browser (from Workbench, Tools > Request License)

Request/Bind License

License Details

Host Id : Win-5BE1-B094-FC24-3440

License Key :

Requester Details

Name :

Company :

E-mail :

Cancel Submit

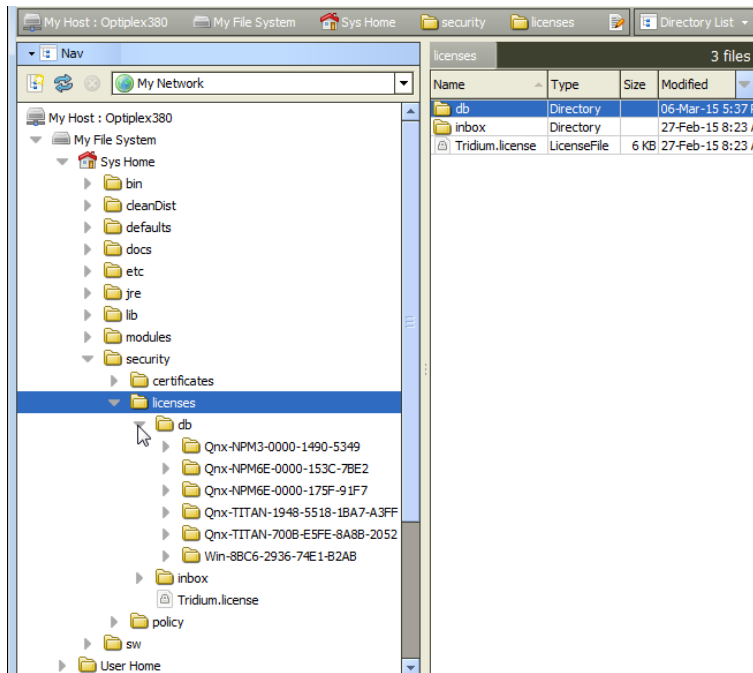
Typically, your Workbench PC is already licensed. Otherwise, you would not be able to successfully start Workbench, and then select **Request License** from the “Tools” menu.

However, you could use this as quick method to request a license for *another* PC on which you have installed Niagara. In that case, you could substitute (type in) the host ID for the other PC in this form, along with other pertinent information.

About the local license database

Any Workbench PC (including a Supervisor) has a “local license database”, a *structured* collection of subfolders and files under its Niagara release (**Sys Home**) `!security/licenses/db` directory. Each subdirectory has a unique Niagara “host ID” name, matching that for some remote host platform. The figure below shows an example license database structure, as viewed in the Workbench Nav tree.

Figure 127 Workbench local license database is everything under !/licenses/db



Your local license database is created and managed *automatically* by Workbench, and updated whenever you perform license operations from platform connections, **PlatformServices** plugins, or when using Workbench tools such as the **Workbench License Manager**. Note that you can see the same directory/file structure when looking at this location on your Workbench PC using Windows Explorer.

NOTE:

The license required for your (local) Workbench PC operation is in the *root* of the licenses folder, named simply by your brand, for example Tridium.license.

Local license database rationale

The local license database design makes it easier for Workbench to store licenses for multiple host platforms—without inadvertently overwriting one license file with another. This saves you from having to make special license folders (subdirectories), and/or rename license files uniquely. The related “license archive” storage file format (.lar) also facilitates the exchange of licenses among different Workbench PCs, and is used in updating/synchronizing licenses to the online licensing server, as well as with provisioning features for Niagara Networks. See [“About license archive \(.lar\) files” on page 117, page 152.](#)

Local license inbox

In addition to the !security/licenses/db folder, there is also a !security/licenses/inbox folder. The inbox allows “drag and drop” importing into your license database of both individual license files and “license archive” (.lar) files, which may have been “saved” or “exported” from other Workbench computers, or perhaps sent to you from the licensing server.

After copying license files and/or .lar files into your inbox subfolder, you need to *close and restart Workbench*. Then, the appropriate “host ID” named subfolders are automatically created in your local license database, each with the appropriate license file(s). Contents of the inbox folder are then deleted.

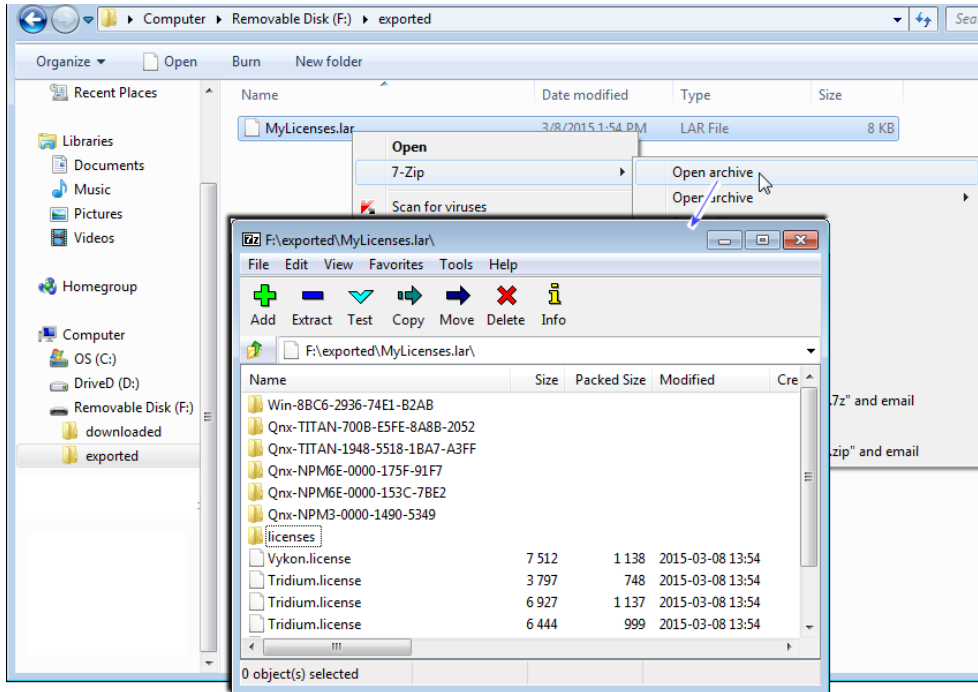
NOTE:

After you restart Workbench, your local license database will be correctly structured. In addition, now you can use the “Sync Online” feature of the **Workbench License Manager** to ensure you have the latest version of all your licenses. See [“Sync Online” on page 114, page 148.](#)

About license archive (.lar) files

When you use the platform **License Manager** view or Workbench's **Workbench License Manager** view (under Tools) to *export* one or more license files, they are saved in a compressed (Zip compatible) format known as a *license archive*, that is a file with a ".lar" file extension. Any .lar file is simply a zip of the exported license file(s) that includes the complete "licenses/hostID" folder (subdirectory) structure for any included licenses.

Figure 128 License archive (.lar) is license file(s) in zip format, including folder paths relative to sys home.



The figure above shows a .lar file in Windows Explorer being opened using 7-Zip, and its subsequent contents. In this case where the archive contains multiple licenses, it was created by an export performed using the **Workbench License Manager** tool. However, if you export a license in the **License Manager** when platform-connected to a remote host, the license archive file contains just the license(s) for that host.

About Niagara license files

A Niagara license file is a structured XML file that has a .license file extension. It enables a set of vendor specific *features*. Each license file is valid for one specific host platform (JACE, PC), matched by that host's unique *host ID*. License files are "digitally signed" by the vendor to prevent tampering.

The following sections provide more details on the contents of a Niagara 4 license file that validates against the Tridium certificate:

- [Items common to all license files, page 153](#) ([license, page 153](#), [about, page 154](#), [brand, page 153](#), [signature, page 154](#))
- [JACE hardware features, page 154](#) (e.g. [dataRecovery, page 154](#), [mstp, page 155](#), [ndio, page 155](#), [serial, page 155](#), others)
- [Driver attributes, page 155](#) ([name, page 156](#), [expiration, page 156](#), [device.limit, page 156](#), [history.limit, page 156](#), [point.limit, page 156](#), [schedule.limit, page 156](#), [parts, page 156](#))
- [Driver types, page 156](#) (many types, including [bacnet, page 157](#), [lonworks, page 157](#), [modbusTcp, page 158](#), [obixDriver, page 158](#), [niagaraDriver, page 158](#))

- [Applications, page 159](#) ([email, page 160](#), [provisioning, page 161](#), [station, page 159](#), [web, page 160](#), [workbench, page 160](#), others)

Items common to all license files

license

All license files require an opening `<license >` line, where the last line in the license file is the closing `</license>` tag, and all contents (lines) in between are `<feature >` elements, plus one signature element.

In the first `<license >` line, there are a number of common attributes, described below.

```
<license vendor="Tridium" expiration="never" version="4.0" hostId="Qnx-NPM6E-0000-153C-6E44"
vendor
```

`vendor="Tridium"` - This is always Tridium.

expiration

`expiration="never"` - The expiration date of the license file. After the expiration date the Workbench software will fail start due to a license expired error. Typically, engineering copies of Workbench have expiration dates which expire on an annual basis. License files for actual projects are issued with non-expiring licenses, where this attribute value is "never".

version

`version="4.0"` - The highest Niagara 4 release version of software which can run in the JACE. If a newer version of software is installed, the JACE will fail on startup with a license version error.

NOTE:

Niagara 4 licenses, starting at version "4.0", are *not* backward compatible with NiagaraAX (version "3.x") software. At some future date when Niagara version 4.1 is released, a host with a `version="4.1"` license will be able to run Niagara 4.0 software as well as Niagara 4.1 software.

hostId

`hostId="Qnx-NPM6E-0000-153C-6E44"` - Alphanumeric code unique to the specific host. On a Windows-based platform, host ID is generated upon installation of the Niagara software, and typically begins with "Win-", for example "Win-5BE1-B094-FC24-3440".

JACE controllers are assigned a host ID at the factory. The first two segments are "Qnx-Model-" such as "Qnx-NPM6-" for a JACE-6 or "Qnx-TITAN-" for a JACE-8000 controller. The `hostId` in the license file must match the `hostId` of the JACE controller, otherwise the JACE cannot run a station.

serialNumber

`serialNumber="329696"` - Applies to a license for a JACE controller only. Designates its unique serial number assigned from the factory. The serial number in the license file must match the serial number of the JACE.

generated

`generated="2015-01-27"` - The date upon which the license file was generated.

brand

For any license with `vendor="Tridium"`, the NiCS (Niagara Compatibility Structure) provides a structure (or schema) that OEMs can use to define the various levels and types of Niagara interoperability that their products will support.

NiCS definitions are contained in this feature item, which is checked by a station or tool when it starts up. There are five attributes to the NiCS: BrandID, Station Compatibility In, Station Compatibility Out, Tool Compatibility In, and Tool Compatibility Out. These elements can be combined in a variety of ways to achieve unlimited flexibility, and are described below.

```
<feature name=accept.station.in="*" accept.station.out="*" accept.wb.out="*" "brand" brand
```

accept.station.in

`accept.station.in="*"` - A list of brands that this local station will allow Niagara data to come in from. Simply stated from a JACE perspective, "this is the list of brands that I can accept data from". The "*" is a wildcard designation to allow all brands.

accept.station.out

`accept.station.out="*"` - A list of brands that this local station will allow Niagara data to be shared with. Simply stated, "This is the list of brands that I can share data with".

accept.wb.out

`accept.wb.out="*"` - A list of brands that this tool is allowed to connect to and engineer. Simply stated, "This is the list of brands that I can engineer".

brandId

`brandId="MyBrand"` - Every licensed station and tool has a Brand Identifier (BrandID). This field holds a text descriptor that the OEM chooses as the identifier for its product line. Each station or tool can have only one BrandID entry.

accept.wb.in

`accept.wb.in="*"` - A list of brands that this station will allow to be connected to it for engineering of its application. Simply stated, "This is the list of brands that can engineer me".

about

The "about" feature is used to designate optional information, and does not affect station operation in any way. This information can be useful for filtering records when searching the license database. Two attributes in this feature are typically designated when ordering product:

```
<feature name="about" project="Tridium Testing" owner="Tridium Tech Pubs"/>
```

project

`project="Tridium Tech Pubs"` - Optional attribute to designate a project. This grouping should typically be assigned to all JACE controllers used for a particular project.

owner

`owner="Tridium Tech Pubs"` - Optional attribute to designate the name of a person or group responsible for the project, or possibly an end user.

signature

This ending element contains a digital signature which is created when the license file is generated. It prevents tampering with the license file. Attempts to edit the license file to enable additional features will render the license file useless.

Typically, the signature element is the last element contained in the license, so it is followed by the closing license tag as the last line in the license file.

```
<signature>MCwCFFOdq4wJcYgvhTVtrf0oSyuCDCwjAhRj+H9pNxQGStBnhEkIqK8rONB10g==</signature>
</license>
```

JACE hardware features

Some license features are specific to JACE controller hardware capabilities.

Alphabetically, these include features [dataRecovery, page 154](#), [jre8qnx, page 155](#), [mstp, page 155](#), [ndio, page 155](#), [nrio, page 155](#), and [serial, page 155](#).

dataRecovery

This feature licenses a station's **DataRecoveryService**, sourced from the its `platDataRecovery` module. This is required to support installed SRAM (Static RAM), whether integral "onboard SRAM" (such as for more recent controllers) or another JACE controller with an installed SRAM option card.

```
<feature name="dataRecovery" expiration="never" parts="NPB-SRAM">
```

jre8qnx

This feature licenses the (Oracle) Sun Hotspot Java 8 virtual machine (VM) to be able to run on the Niagara 4 JACE controller. There are no attributes

```
<feature name="jre8qnx" expiration="never">
```

mstp

This feature determines how many of the available serial ports may be used for BACnet MS/TP communications. Note that features bacnet and serial must also exist in the license file.

```
<feature name="mstp" expiration="never" port.limit="5" parts="DR-MSTP-AX"/>
```

port.limit

`port.limit="5"` - This specifies the number of serial ports which may be used for MSTP communications. Typically this number matches the number of physical ports. Some JACE controller models have option card modules or slots with serial ports.. If additional ports are added then the port limit may be less than the number of physical ports (if the port activation has not been ordered as well).

ndio

This feature enables the NDIO (Niagara Direct Input Output) driver, required to configure and use a JACE controller's Ndio-type I/O modules. Not all JACE controllers support such I/O modules (which attach/chain directly to the controller, using 20-pin connectors); refer to specific JACE controller data sheets to confirm whether this is an available option. Note that in the ndio features line (below), a "device" equates to an "Ndio Board", and that history and schedule limits have no practical application.

```
<feature name="ndio" expiration="never" device.limit="none" history.limit="none" point.limit="none">
```

Refer to the *Niagara NDIO Guide* for related details.

nrio

This feature enables the NRIO (Niagara Remote Input Output) driver, required to configure and use a JACE controller's Nrio-type I/O modules and/or any onboard I/O of a controller. All N4 JACE controllers support NRIO modules (which communicate via RS-485). Note that in the nrio features line (below), a "device" equates to an "Nrio16Module", and that history and schedule limits have no practical application.

```
<feature name="nrio" expiration="never" device.limit="16" history.limit="none" point.limit="none">
```

Refer to the *Niagara Nrio Guide* for related details.

serial

This feature enables the use of JACE serial ports for various drivers, for example aapup or modbusAsync. Note that the JACE license needs this serial feature in addition to any specific driver feature. Only one serial feature line is needed, regardless of number of serial-based drivers. Note that in the case of a JACE used for BACnet MS/TP, it would require this serial feature and driver features bacnet and mstp.

```
<feature name="serial" expiration="never"/>
```

Driver attributes

Each driver is enabled by a feature line (element) in the license file. Most of the drivers utilize the same *attributes* within that feature. The most common driver attributes are shown below.

```
<feature name="driverName" expiration="expirationDate" device.limit="none" history.limit="none">
```

The various "limit type" attribute values can be either "none" or a numerical (limit) value, for example `device.limit=32`. Note that a limit value of `none` means unlimited, whereas a limit value of `0` means none allowed.

For many drivers, only the `point.limit` and `device.limit` attributes are applicable; yet most drivers include all `.limit` attributes. For example, none of the Modbus-related drivers have any history or schedule

import/export capability, due to the simplicity of the Modbus protocol. Thus, "history.limit" and "schedule.limit" values have no significance in the feature for a Modbus driver.

NOTE:

In Niagara 4 (and depending on license), limit attributes in individual drivers may be superceded by "global capacity" limits, using a licensing model that sets limits that span across multiple drivers. This allows more flexibility to allocate the number of devices, points, and so on—without requiring ongoing license changes. For more details, see ["Global capacity licensing" on page 125, page 161](#).

name

Feature name of the driver, often the same as the actual module (.jar file) name, for example bacnet, lonworks, etc.

expiration

Each driver has an expiration date which is typically the same as the expiration property of the license feature. In some cases such as beta testing agreements, individual drivers may be set to expire where the main license file is non-expiring.

device.limit

This attribute designates a license limit on the number of devices which may be added to this specific driver network in the station database. Above this limit, any added device component (and all its child components) will be in fault.

This limit has no impact on the actual physical limitation of a field bus. For example just because the lonworks feature is set to device.limit="none", this does not mean that you can exceed the normal limit of 64 devices per segment.

history.limit

This attribute limits the number of Niagara histories that can be imported from remote histories (logs or trends) into the station's history space, and/or exported from station histories to appear as histories in remote devices. Above this limit, any added history import descriptor (or history export descriptor) will be in fault, and the associated import/export will not be successful.

point.limit

This attribute designates the maximum number of proxy points that may be added to the station database for a particular driver. Above this limit, any added proxy point will be in fault.

schedule.limit

This attribute limits the maximum number of Niagara schedules that can be imported from remote schedules into the station's database, and/or exported from station schedules to appear as schedules in remote devices. Above this limit, any added schedule import descriptor (or schedule export descriptor) will be in fault, and the associated import/export will not be successful.

parts

This is an alphanumeric part code which is automatically assigned when generating the license file and is for Tridium internal use.

Driver types

Each driver type is enabled by a separate feature element (or line, starting with *name* attribute), and has common attributes.

NOTE:

New Niagara drivers are continually developed and offered as products. This section includes some, but not all drivers available. It is included in this section to illustrate how driver features appear in licenses.

Alphabetically, driver types listed here include [aaphp, page 157](#), [aapup, page 157](#), [bacnet, page 157](#), [bacnetAws, page 157](#), [bacnetOws, page 157](#), [fileDriver, page 157](#), [lonworks, page 157](#), [modbusAsync, page 158](#), [modbusCore, page 158](#), [modbusSlave, page 158](#), [modbusTcp, page 158](#), [modbusTcpSlave, page 158](#), [obixDriver, page 158](#), [opc, page 158](#), [niagaraDriver, page 158](#), [rdbOracle, page 158](#), [rdbSqlServer, page 159](#), [snmp, page 159](#), [videoDriver, page 159](#) and [zwave, page 159](#).

aaphp

Enables the American Auto-Matrix Public Host Protocol (PHP) driver. The serial feature is also required.

aapup

Enables the American Auto-Matrix Public Unitary Host (PUP) driver. The serial feature is also required.

bacnet

Enables functionality of the BACnet driver for BACnet/Ethernet and BACnet/IP. If a JACE controller, other features can be added to enable BACnet MS/TP communications over serial ports: `mstp` and `serial`.

```
<feature name="bacnet" expiration="never" device.limit="none" export="true" history.limit=
```

Refer to the *Niagara BACnet Guide* for details on all BACnet integration with Niagara.

export

`export="true"` - When set to "true" this field enables BACnet server operation. When the field is set to "false" only BACnet client operation is permitted.

NOTE:

When BACnet export is enabled, any station histories and/or schedules that are exported to BACnet do not count towards any `history.limit` or `schedule.limit` values in the license (if any).

bacnetAws

Provides added functionality as *BACnet AWS Supervisor* with BTL-certification, as described in the BACnet "Advanced Operator Workstation" specification (B-AWS). Available for PC platforms only (not JACE platforms). The `bacnet` feature is also required in the license. More details are available in an appendix in the *Niagara BACnet Guide*.

bacnetOws

Provides added functionality as *BACnet OWS Supervisor* with BTL-certification, as described in the BACnet "Operator Workstation" specification (B-OWS). Available for PC platforms only (not JACE platforms). More details are available in an appendix in the *Niagara BACnet Guide*.

fileDriver

Enables the File driver, used to import comma or tab delimited text files and convert into Niagara histories. For more details, see the "file-FileNetwork" section in the *Niagara Drivers Guide*.

lonworks

Enables the Lonworks driver. Utilizing the driver also requires a LON interface on the JACE controller. Most JACE controller models require an optional Lonworks interface card to be installed. More details are available in the *Niagara Lonworks Guide*.

modbusAsync

Enables the Modbus Master Serial driver. The JACE controller operates as the Modbus Master device communicating via an available serial port using either Modbus RTU or Modbus ASCII. The `modbusCore` and serial features are also required.

modbusCore

Required by a JACE controller or Modbus Supervisor host for any of the Modbus drivers (Async, Slave, TCP, TCP Slave). For details on any Modbus driver, refer to the *Niagara Modbus Guide*.

modbusSlave

Enables the Modbus Slave Serial driver. The JACE controller operates as a Modbus Slave communicating via an available serial port using either Modbus RTU or ASCII to a Modbus Master device. The `modbusCore` and serial features are also required.

modbusTcp

Enables the Modbus Master TCP driver. The JACE controller or Modbus Supervisor operate as a Modbus Master device communicating via Modbus TCP/IP. The `modbusCore` feature is also required

modbusTcpSlave

Enables the Modbus Slave TCP driver. The JACE controller or Modbus Supervisor operates as a Modbus Slave device communicating via Modbus TCP/IP. The `modbusCore` feature is also required

obixDriver

Enables the oBIX driver. The driver supports the oBIX protocol, which is M2M (Machine-to-Machine) communications via XML over TCP/IP. Refer to the *Niagara Obix Guide* for related details.

```
<feature name="obixDriver" expiration="never" device.limit="none" export="true" history.limit=
export
```

`export="true"` When set to "true" this field enables oBIX server operation. When the field is set to "false" only oBIX client operation is permitted.

opc

Enables the OPC client driver, and is only available on Windows-based platforms because of the protocol's dependency of Windows. Refer to the *Niagara OPC Guide* for related details.

niagaraDriver

Enables communication via the Fox protocol to other Niagara stations, and allows creation of a NiagaraNetwork, including proxy points, importing/exporting histories and schedules, and routing alarms.

```
<feature name="niagaraDriver" expiration="never" virtual="true" schedule.limit="none" point.1
```

For more details, refer to the *Niagara Drivers Guide* section "About the Niagara Network".

rdbOracle

Enables the Relational Database Driver using the Oracle database format. This driver allows exporting of histories from the Niagara station to an Oracle database. The driver does not include the Oracle software, which must be purchased separately from a third party source.

```
<feature name="rdbOracle" expiration="never" parts="ENG-WORKSTATION"/>
```

rdbSqlServer

Enables the Relational Database Driver using the Microsoft SQL database format. This driver allows importing and exporting of histories to and from the Niagara station, and to and from a Microsoft SQL database. The driver does not include the Microsoft SQL software, which must be purchased separately from a third party source. The driver does work with the MSDE version which is free from Microsoft; however, the normal Microsoft imposed limitations on the MSDE version still apply.

```
<feature name="rdbSqlServer" expiration="never" history.limit="10" historyImport="true" pa
```

snmp

Enables the SNMP (Simple Network Management Protocol) driver, which allows sending and receiving SNMP messages. Refer to the *Niagara SNMP Driver Guide* for related details.

```
<feature name="snmp" expiration="never" device.limit="none" history.limit="none" point.lim
```

videoDriver

Enables the Niagara Video Framework driver (modules `nvideo`, `videoDriver`, `nDriver`) that provide the foundation to integrate select commercial off-the-shelf video surveillance and recording systems into a Niagara station. Depending on the specific video hardware used, one or more vendor-specific license feature entries are also typically required. Refer to the *Niagara Video Framework Guide* for related details.

zwave

Applies to a JACE controller with an installed Z-Wave option card, or any host platform with a third-party, serially-connected, Z-wave gateway device. The serial feature is also required. Enables a network of wireless Z-Wave devices (ZWaveNetwork), including device and point limits. Refer to the *Niagara Z-Wave Driver Guide* for related details.

Applications

Alphabetically, application types listed here include [box, page 160](#), [email, page 160](#), [ldapv3, page 161](#), [mobile, page 161](#), [provisioning, page 161](#), [search, page 161](#), [template, page 161](#), [station, page 159](#), [web, page 160](#), and [workbench, page 160](#). Applications [station, page 159](#), [web, page 160](#), and [workbench, page 160](#) have special importance, and are summarized first.

NOTE:

The application feature “`crypto`”, as used in NiagaraAX licenses is no longer required. All Niagara 4 hosts are capable of TLS operation without this license feature. For more details, refer to the *Niagara 4 Station Security Guide*.

station

Enables a station to be run, and is present in any JACE platform, as well as a Supervisor.

```
<feature name="station" expiration="2015-04-01" resource.limit="none" guestEnabled="true"/>
```

The station feature may not be present in a license for an engineering workstation (PC), unless specifically ordered with it. Optional attributes are listed below.

resource.limit

`resource.limit="none"` - If the `resource.limit` flag is specified (in kRUs), then the station displays a warning on startup if the actual resource units exceed the limit resource units. If the limit is exceeded by 110% then the station will not boot at all. This limit is normally only specified in (NiagaraAX) SoftJACE license files.

NOTE:

In Niagara 4, the station feature attribute `resource.limit` is superceded by “global capacity” limits, using a licensing model that sets limits on the numbers of devices, points, histories, and so on that span across multiple drivers in the station. This allows a clearer measure of resource capacity in a license than the “resource limit” method. For more details, see [“Global capacity licensing” on page 125, page 161](#).

guestEnabled

`guestEnabled="true"` - Must be present and true, or else the station's `UserService` has its built-in user "guest" hidden upon first station start up, as a security measure. Only hosts licensed as "demo hosts" can enable and use the guest user—thus is unavailable on any host with a "non-expiring" license.

web

The web feature must be present to start the `WebService` in a running station (to access the web server via a browser HTTP connection). If not licensed, the server is set to fault with appropriate `faultCause`.

NOTE:

Full Workbench can connect to a station (via Fox connection) even if the web feature is missing or expired.

```
<feature name="web" expiration="never" ui="true" ui.wb="true" ui.wb.admin="true"/>
```

ui

`ui="true"` - This flag allows browser access to users with an HTML5 Hx Profile.

Note: If `ui="false"`, users cannot access the browser UI with either HTML5 Hx or Wb web profiles. No browser access is allowed, except for Spy pages.

ui.wb

`ui.wb="true"` - This flag allows browser access to users with a Wb web profile.

Note: If `ui.wb="false"`, users with an HTML5 Hx web profile still have browser UI access, as long as `ui="true"`.

ui.wb.admin

`ui.wb.admin="true"` - This flag allows browser users with a Wb web profile access to admin-only views on components, providing they have admin permissions on components with such views. Admin-only views include most types of views, except for property sheet views. For example, wire sheets and most manager views require this option. Browser access to such views is unavailable for any user with an HTML5 Hx web profile. Or, if this flag is false, such views are also unavailable to Wb web profile users.

Note: If `ui.wb.admin="false"`, users still have access to the station with a browser, subject to the "ui" and "ui.wb" flags. Property sheet views are available on components. Slot sheets may be available too, providing a user has admin-level permissions on components.

workbench

The workbench feature must be present to start the full version of Workbench (for example, a copy of Tri-dium's Niagara Workbench or an OEM-specific Workbench-based application). If the admin flag is false, then all views requiring admin access are unavailable. This feature is included for PC platforms only, with the sole exception of a (NiagaraAX) SoftJACE.

```
<feature name="workbench" expiration="never" admin="true"/>
```

box

This enables a host for Bajascript, a Javascript API (read and write) for Niagara data access from Javascript enabled environment like web browsers. Along with the mobile feature, this license feature is required for mobile application support.

```
<feature name="box" expiration="never" session.limit="none" parts="ENG-WORKSTATION"/>
```

Messaging features

The devices monitored by the system and the services that do the monitoring can communicate status, alarms and reports as needed using direct email and SMS messaging.

The system supports these features:

- The email feature enables a station to communicate with an SMTP server: `<feature name="email" expiration="never"/>`

If the feature is not present, the system marks the **EmailService** and all incoming and outgoing accounts as in `{fault}`.

- The SMS messaging feature enables a station to send text messages to a phone.

ldapv3

This feature enables a host to use authentication schemes of LDAP and/or Kerberos for station users under the standard **UserService**. This allows users to be authenticated using LDAP in coordination with the site's existing Active Directory server or LDAPv3 server.

Note this departs from the former usage of special user services, such as "LdapUserService" or "LdapV3ADUserService" in place of the standard **UserService** in NiagaraAX.

If the `kerberos` attribute is "true", the Niagara 4 host is licensed for Kerberos authentication with LDAPv3.

```
<feature name="ldapv3" expiration="never" kerberos="true" parts="ENG-WORKSTATION"/>
```

Refer to the *Niagara LDAP / Active Directory Guide* for complete details.

mobile

This enables the host to support the Niagara Mobile application framework, for station support of web browser access from mobile devices like cell phones or tablets. The host also requires to be licensed with the box feature for Bajascript support.

```
<feature name="mobile" expiration="never" history="true" schedule="true" alarm="true" px="1"/>
```

provisioning

Enables the operation of Niagara host provisioning, typically used to automate routine maintenance of a Niagara system such as JACE software upgrades, file distribution and backups. It applies to an Supervisor platform only. Provisioning uses the **BatchJobService** and a "network extension model" (e.g. a "ProvisioningExt" under the **NiagaraNetwork**), sourced respectively from modules **batchJob** and **provisioningNiagara**.

```
<feature name="provisioning" expiration="never"/>
```

search

(New in Niagara 4) Enables the **SearchService** and the use searches in the station. Without this feature, the **SearchService** remains in fault, and the "Quick Search box" and Search sidebar are unavailable.

The "local" attribute must be true to allow searches local to this station. The "system" attribute is for a future release of Niagara 4 (e.g. 4.1) where searches can span across multiple stations.

```
<feature name="search" local="true" system="true" expiration="never"/>
```

template

(New in Niagara 4) Enables the **TemplateService** and the use of templates in the station. Without this feature, the **TemplateService** remains in fault, as well as templates.

```
<feature name="template" expiration="never"/>
```

Global capacity licensing

In Niagara 4, a new licensing model is more widely-used for some platforms, among them the newest JACE controller (JACE-8000): Global capacity licensing. Specific resources in the station are tracked using global counters, providing more flexibility in how resources are allocated.

Using a license feature named "globalCapacity", the station keeps a global count of all networks, devices, proxy points, links, histories and schedules. When one of these resources goes over a licensed limit, the resource either goes into fatal fault or becomes inactive.

The content of any particular global capacity license depends on the feature purchased for the controller.

Example globalCapacity feature entry

```
<feature name="globalCapacity" expiration="2016-04-01" point.limit="1250" device.limit="50" excludedDevices="" excludedPoints="" excludedNetworks="" excludedLinks="" excludedHistories="" excludedSchedules="" />
```

The example above sets global limits on points (1250) and devices (50), but no limits on networks, links, histories, or schedules. The `excludedDevices` and `excludedPoints` attributes mean that there is no limit on the number of devices and points from these modules: `ndio`, `nrio` or `niagaraDriver`.

A more restrictive global capacity example feature could look like below.

```
<feature name="globalCapacity" expiration="never" network.limit="3" device.limit="25" point.limit="500" link.limit="400" history.limit="124" schedule.limit="10" />
```

The example indicates:

- A limit of three networks of any kind
- A limit of 25 devices of any kind,
- A limit of 500 points of any kind
- A limit of 400 links of any kind.
- A limit of 124 histories and 10 schedules of any kind.

Attributes allow for `excludedNetworks`, `excludedDevices`, and `excludedPoints`, each as a comma-separated list of modules. This means that there is no limit for `nrio` networks, Devices and ports. Any modules in these attributes are excluded from the respective global capacity limit. However all links, histories, and schedules from these modules are *not excluded* from any other global capacity counts.

NOTE:

All stations include an `AuditHistory` and `LogHistory`, which are *included* in the history limit.

More details are in the following sections:

- ["Capacity licensing operation and recount" on page 125, page 162](#)
- ["Checking capacity licensing status" on page 125, page 162](#)
- ["Capacity licensing fault notifications" on page 127, page 164](#)
- ["Capacity licensing notes about histories" on page 128, page 165](#)

Capacity licensing operation and recount

Often you delete as well as add resources in the process of engineering a station. It is important to note that capacity licensing *never decrements* any resource counter. This could result in inaccurate counts over a long period of time.

NOTE:

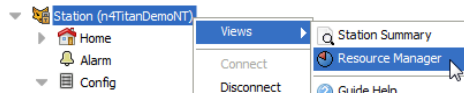
Therefore, any resource created with an over-capacity error will *never* get out of fault; you must delete it.

Global capacity counts are corrected on station restart. However since this is often inconvenient, a global capacity *recount* is done approximately every 10 minutes to correct any inflated counts. You can always see the recount status, along with the current global licensing counts and limits in the station's **Resource Manager** view (or "spy" view). See ["Checking capacity licensing status", page 162](#).

Checking capacity licensing status

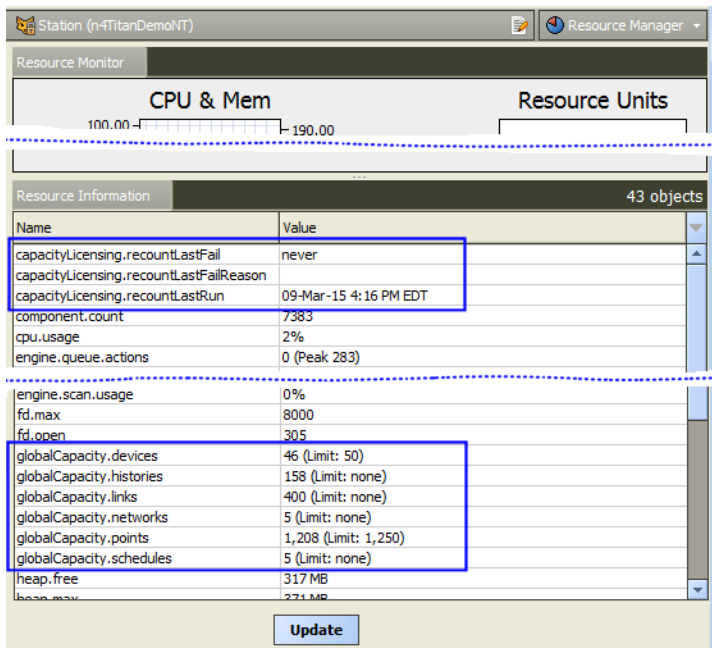
Any station running on a platform with global capacity licensing keeps a running tally on all corresponding resources in the **Resource Manager** view.

Figure 129 Accessing a station's Resource Manager view



In Workbench, right-click the opened **Station** and select **Views**→→**Resource Manager**. Any station with global capacity licensing has specific entries in the lower "Resource Information" table.

Figure 130 Example entries for global capacity licensing in a station's Resource Manager view.



As shown above, there are three "capacityLicensing.recount" statistics, including a timestamp for when the global capacity recount last ran.

Another group of "globalCapacity.resource" statuses show the current counts along with respective license limits (if any). The station view above reflects the first globalCapacity example given, namely:

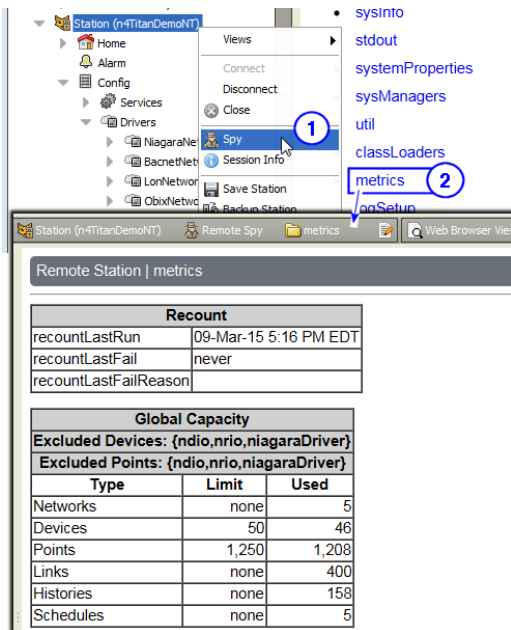
```
<feature name="globalCapacity" expiration="2016-04-01" point.limit="1250" device.limit="50"
```

where global limits exist only on devices (Limit: 50) and proxy points (Limit: 1,250) with ndio, nrio and niagaraDriver devices and points being excluded from any limits.

Alternatively, if you are a super user, you can get this same information from the right-click spy page on a station, at the following location: **Spy**→→**metrics**, as shown in [Figure 131](#) [Figure 131](#), page 164.

NOTE: Special permission is required to view spy pages.

Figure 131 Capacity licensing information in the station Spy pages

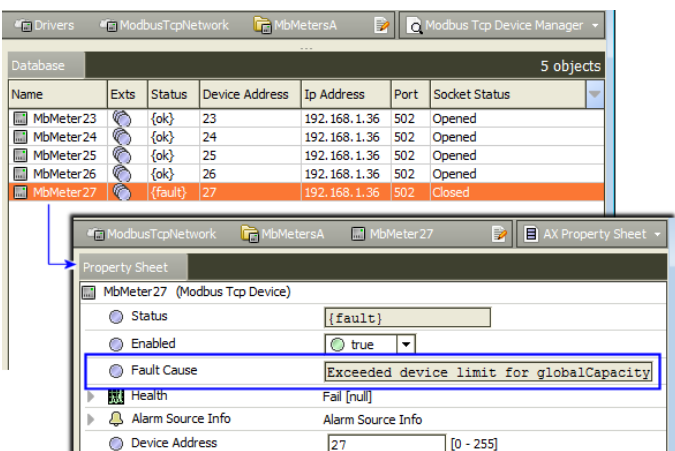


Information in the spy page above matches the Resource Manager view example shown in [Figure 130 Figure 130 on page 126, page 163](#), with the exception of the “recountLastRun” timestamp shown (one hour later).

Capacity licensing fault notifications

Added components that exceed global capacity limits provide a “Fault Cause” explaining the reason. In the prior example, where there are 46 global existing devices globally, if five (5) new ModbusTcpDevices are added this results in a fault, as 51 devices is one over the 50 limit.

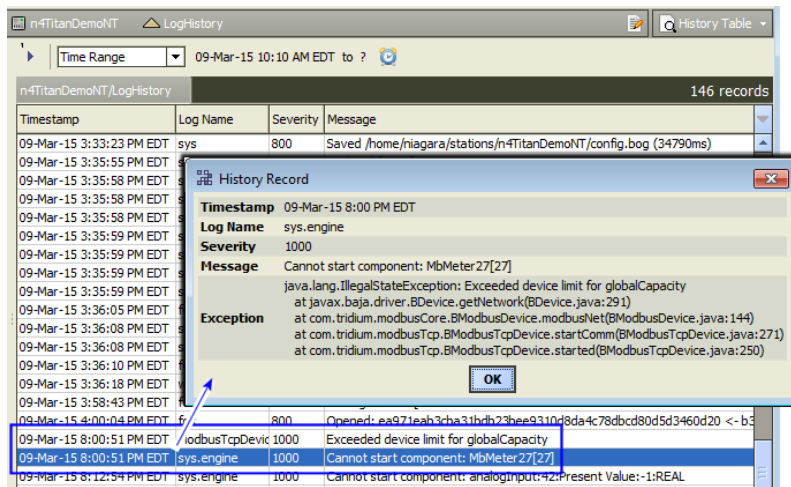
Figure 132 Example of an added device exceeding the global capacity device limit



As shown above, the property sheet of the device shows this reason in **Fault Cause**. You must delete this (or any) component with a similar fault cause, as otherwise it *remains in fault*.

Note that corresponding events are also entered in the station’s LogHistory, as shown in [Figure 133 Figure 133, page 165](#).

Figure 133 LogHistory entries from exceeding globalCapacity



As shown below, the globalCapacity count for any exceeded resource appears in the corresponding entry in the station's **Resource Manager**.

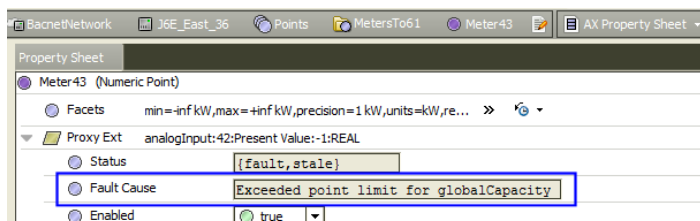
Figure 134 Example globalCapacity entry for a resource over license limit

Name	Value
fd.open	305
globalCapacity.devices	51 (Limit: 50)
globalCapacity.histories	158 (Limit: none)
globalCapacity.links	400 (Limit: none)
globalCapacity.networks	5 (Limit: none)

Note the necessary deletion of an over-limit device (51) *does not decrement the count* back at that time—in-
stead, a periodic recount (about every 10 seconds) or station restart is needed for this.

Similar component faults and error logs apply to networks, points, links, and schedules.

Figure 135 Example globalCapacity fault for proxy point



As shown above, the ProxyExt for a proxy point shows a **Fault Cause** reason. The property sheet of a network in global capacity fault is similar Exceeded network limit for globalCapacity, and this applies also to a schedule in global capacity fault: Exceeded schedule limit for globalCapacity.

Exceeding the globalCapacity *link* limit produces a popup **Capacity Licensing** window on the wire sheet or active view, saying "Exceeded Link Limit", and the link is not functional.

Histories exceeding globalCapacity limits can manifest in different ways. See ["Capacity licensing notes about histories"](#), page 165.

Capacity licensing notes about histories

Histories in the station are being counted. If the number of histories in the station is greater than the global-Capacity limit, then the resource counter in the station prevents more histories from being created.

In the default **History Ext Manager** view on the **HistoryService**, you see a table of all history *extensions* along with the state of each (enabled, disabled, fault). Included is a total count of all history extensions. However, there is no easy, intuitive way to get a count of those in a particular state or set of states. An extension in a fault state with a **Fault Cause** of `Exceeded history limit` indicates a related problem. Simply disabling such an extension does not fix the issue—you must delete the history in the history database, which cannot be done from the **HistoryService**.

You delete histories from the **Database Maintenance** view of the **History** space; but note there are no counts available there. If you delete one or more histories to reduce history count, you must wait until the periodic recount (approximately every 10 seconds) comes around to see the reduced count. However, you cannot delete the history extension that created the history from there. So it is likely that the history is going to be recreated in the future, unless you delete or modify the history extension that created the history database table.

Note that in addition to history extensions in the same station, other items can create histories, including:

- History imports in the same station
- History exports in another station
- AuditHistory service
- LogHistory service

These activities add to the count. This is the reason why a station quickly exceeds its history limit.

Chapter 7 Time Zones and Niagara 4

Topics covered in this chapter

- ◆ Time zones and terminology
- ◆ Selecting a time zone in Niagara

Platform configuration of a Niagara host includes specifying its time zone. This affects both real time clock accuracy used in station control, as well as how timestamps appear in items like histories and alarms. This appendix provides details on time zone selection in Niagara 4, including the currently used “historical time zone database.”

The following main sections are included:

- [“Time zones and terminology” on page 129, page 167](#)
- [“Selecting a time zone in Niagara” on page 130, page 168](#)

NOTE:

Workbench provides a special “Time Zone Database Tool” that lets you explore the historical time zone database on the local Workbench host.

Time zones and terminology

A time zone is a region in the world that uses the same standard time, often referred to as the *local time*. There are many different time zones, owing to the combinations of geographic locations and political/cultural differences. Time zones calculate their local time as an offset from UTC (Coordinated Universal Time). In addition, many time zones apply DST (Daylight Saving Time). See [“UTC”, page 167](#) and [“DST”, page 167](#)

UTC

Coordinated Universal Time (UTC) is the recognized atomic-clock standard of reference time, largely replacing GMT (Greenwich Mean Time) as reference time. Time zones are commonly expressed as negative or positive offsets from UTC time.

DST

Daylight Saving Time (DST) is used as a means of maximizing daylight hours during normal waking hours, and is used by many (but not all) time zones. DST is a twice-yearly event acting upon local time, as follows:

- Start of DST adds an offset (typically 1 hour) to local time. During this period of the year, local time may be called “daylight time.”
- End of DST removes the DST offset from local time. During this period of the year, local time may be called “standard time.”

Any time zone using DST has specific rules that define the exact days and times when DST starts and ends. These rules vary widely from zone to zone, since DST policies are set by national and regional governments. Also, DST policies are subject to *change* for this same reason—as in the recent 2007 change for all U.S. time zones that observe DST.

In the 2007 U.S. DST changes, the DST start time was changed to “first Sunday on or after the 8th in March” (from “first Sunday on or after the 1st in April” for 2006 and prior years). The DST end time was changed to “first Sunday on or after the 1st in November” (from “last Sunday in October” for 2006 and prior years).

NOTE:

A change in DST rules for a time zone can cause *issues* in Niagara when displaying historical data (histories and alarm records), particularly when applying new (current) DST rules to records collected using prior (old) DST rules. For more details, see [“About the historical time zone database”](#) on page 130, page 169.

Selecting a time zone in Niagara

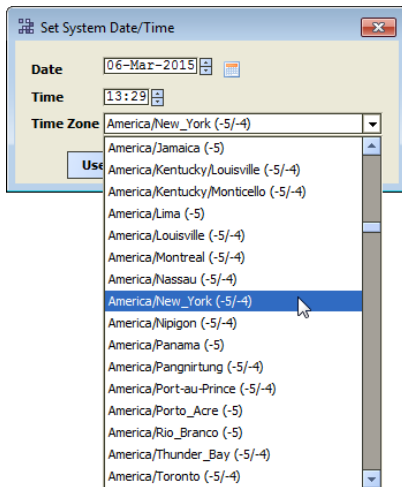
Since the first Niagara release, platform configuration of a Niagara host includes the setting of date and time, which includes specifying its local time zone.

NOTE:

For any Niagara 4 Windows host, you must use *Windows tools* to specify a time zone and/or change date and time. Such tasks, as well as TCP/IP configuration on a Windows host, are read-only.

Typically, you specify time zone in a JACE controller during its initial commissioning in a platform connection, when running the **Commissioning Wizard**. Or, you can do this at any time using the **Platform Administration** view (function “Change Date/Time”).

Figure 136 Selecting time zone from Change Date/Time selection in Platform Administration View



After a station is installed and running, you can also specify a JACE’s time zone using one of the station’s **PlatformService** views (“Platform Service Container Plugin” or “System Date and Time Editor”).

In any case, time zones appear on a selection list with a format such as:
Zone ID (± hours UTC offset DST, ± hours UTC offset UST).

For example:

```
America/Chicago (-6, -5)
Europe/Berlin (+1, +2)
Asia/Tokyo (+9)
```

Note there is no DST observance in Japan, so the selection with zone ID “Asia/Tokyo” shows only the UTC offset of +9 hours. This selection list of time zones is from a historical “*time zone database*”.

NOTE:

Unlike in NiagaraAX, in Niagara 4 there is no separately maintained “timezones.jar” distributed in Niagara builds for a time zone database, nor associated entries in a platform’s system.properties file. Instead, time zones are directly sourced from the Java VM (virtual machine) in the host platform.

This means there is no longer a workflow for updating time zone definitions independently from Java updates that may be included in Niagara 4 updates.

About the historical time zone database

The Java-sourced time zone database has a “historical perspective,” where a history of *changes* for applicable time zones are stored. Thus, *multiple definitions* for a time zone may exist, including past definitions as well as its current definition.

This allows display of a station’s timestamped data (histories and alarms) collected in time zones under “prior rules” (typically DST-related) to display with the original (and correct) collected time.

NOTE:

On all Niagara 4 JACE controller platforms, the Java-sourced time zone database is historically accurate only back to year 2010. Any pre-2010 historical data is displayed using 2010 rules. This was done to improve Java heap usage on these platforms.

However, note the Java-sourced time zone database on Windows Niagara 4 platforms extends further back, for example, to year 1995.

In Niagara 4 Workbench, use the **Time Zone Database Tool (Tools→ →Time Zone Database Tool)** to navigate the Java time zone database, where you can explore DST rules for any timezone. If a local station is running on the same host (Supervisor), this is time zone database that is utilized. For more information about the Time Zone Database Tool, see the section by the same name in the *User Guide*.