

Technical Document

# Station Security Guide

August 18, 2015

niagara<sup>4</sup>

# Station Security 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

<b>About station security</b> .....	<b>7</b>
Security precautions .....	7
Security best practices.....	8
Document change log .....	9
<b>Chapter 1 Secure communication (TLS)</b> .....	<b>11</b>
Client/server relationships .....	11
Certificates .....	12
Encryption .....	16
Naming convention .....	17
Certificate set up.....	17
Workbench checklist.....	17
Supervisor checklist.....	18
Platform and station checklist .....	19
Opening a secure platform connection (niagarad) .....	20
Enabling the NiagaraNetwork.....	21
Confirming client/server relationships .....	22
Certificate stores and certificate creation process.....	22
Creating a certificate .....	25
Password strength.....	27
Creating a CSR.....	27
Signing a certificate.....	28
Importing the signed certificate back into the User Key Store .....	28
Deleting .pem files .....	29
Exporting a certificate .....	29
Importing a certificate into a User Trust Store.....	29
Station health confirmation.....	30
Viewing session information.....	30
Allowed hosts management.....	31
Configuring secure platform communication .....	31
Configuring secure station communication .....	32
Setting up client/server relationships .....	32
Enabling clients and configuring them for the correct port .....	32
Securing email.....	32
Certificate renewal .....	34
Deleting a certificate .....	34
Secure communication troubleshooting .....	34
Default TCP ports.....	36
Certificate management when replacing a controller .....	37
<b>Chapter 2 User authentication</b> .....	<b>39</b>
User authentication checklist .....	39
Authentication schemes .....	40
Station-to-station users .....	40

- Adding or editing a user ..... 41
- Assigning authentication schemes to users ..... 41
- Password management ..... 41
  - Setting up password strength ..... 41
  - Setting up password options ..... 42
  - Setting up a user's password ..... 42
- Logging on to a station..... 42
- Changing your password ..... 44
- User authentication troubleshooting ..... 44
- Chapter 3 Authorization management.....45**
- Component permissions checklist ..... 46
- Component permission level..... 46
  - Changing a component Config flag ..... 46
  - UserService permission levels ..... 47
- Categories ..... 48
  - Basic categories ..... 48
  - Category management ..... 48
- Roles and permissions ..... 51
  - Adding roles and permissions ..... 51
  - Adding a component..... 53
  - Editing roles and permissions ..... 53
  - Assigning roles to users ..... 54
  - Confirming access security..... 54
  - Reviewing permissions ..... 54
  - Ancestor permissions ..... 55
  - File permissions ..... 55
  - History permissions ..... 55
- Component permissions troubleshooting ..... 56
- Chapter 4 Reference.....57**
- Components ..... 57
  - AuthenticationService ..... 57
  - CategoryService..... 59
  - RoleService ..... 61
  - UserService property sheet..... 62
  - FoxService ..... 63
  - WebService..... 65
- Windows..... 68
  - Certificate Signing window ..... 68
  - Edit history export window ..... 69
  - Generate Self-Signed Certificate window ..... 70
  - New category windows ..... 72
  - New/Edit roles window ..... 73
  - Platform Authentication window ..... 73
  - Platform Connect window ..... 74
  - Platform TLS settings ..... 75

- Permissions map .....75
- Session Info window .....76
- Station Authentication window .....77
- Station Connect window.....78
- Plugins (views).....78
  - Category Browser .....78
  - Category Manager .....79
  - Category Sheet .....80
  - Permissions Browser.....81
  - Certificate Management view .....82
- Glossary .....89**
- Index.....91**



## About station security

Cyber security is serious business. Almost daily, news of a breach somewhere in the world comes to light. These station security topics provide guidance and best practices designed to help you configure and maintain secure stations while taking advantage of the benefits that internet connectivity offers.

Security begins with the way you configure and monitor each station. It involves setting up secure communication, secure email, secure user credentials, and configuring components, categories, hierarchies, and roles to grant users access only to the system objects they need to do their jobs. Ultimately, your system will only be secure if you take full advantage of Niagara 4's enterprise security features, and if you configure your network effectively. Although the defaults are designed to be as secure as possible, your system will remain vulnerable if you rely solely on factory defaults. The aspects of station security that require configuration are settings for secure communication, user authentication and authorization management.

- *Secure communication* provides:
  - Server identity verification, which prevents man-in-the-middle and spoofing attacks. To set up the digital certificates that verify server identity, you use the **Certificate Manager** view.
  - Data encryption (foxs/https/platformtls), which prevents eavesdropping during the actual transmission of data. You define the key size used to encrypt data transmission when you create each certificate.
  - Secure email communication. To configure email security, you use the **EmailService**.
- *User authentication* protects against malicious access by ensuring that only legitimate users (human or station) can log in using Workbench or a web browser. You use the **AuthenticationService** to activate the authentication schemes the station needs, and the **UserService** to assign the authentication scheme and login credentials to individual users (human or another station). You can add multiple schemes, each of which may be used by a different user.
- *Authorization management* involves defining which component slots, files and histories are accessible, which users may modify them, and what modifications users may make. Niagara uses role-based access control, where users are assigned roles that are mapped to component permissions. You use the **CategoryService** to set up component categories (groups of components), the **RoleService** to assign permissions, and the **UserService** to assign roles to users.

**RESTRICTION:** Niagara 4.0 does not support the Tagged Categories feature, which is mentioned several times in this guide. Tagged Categories may be available in a future release.

**NOTE:** Platform security is documented elsewhere.

## Security precautions

Whether you are protecting assets in a single building or in a large, multi-site application, station security is critical. The practical implementation of a secure device network relies on basic common sense.



Do not connect any station directly to the Internet. If you need remote access, use a VPN (Virtual Private Network) solution where your devices are protected behind a fire wall, but remotely accessible. Your VPN solution should incorporate RSA two-factor authentication.



Do not share accounts (log on using someone else's credentials). Always log on as yourself.



Do not create a certificate (and key pair) on a local computer and download the certificate into the **User Key Store** of each remote controller. Each host requires its own unique certificate, public and private keys, which should be generated by the controller and should reside only in the controller or on a backup medium that is physically protected. Transmitting a certificate with its private key exposes the key to the risk of capture during transmission.



Do not commission a remote controller over the internet. If it becomes necessary to replace a controller, physically travel to the location, take the controller off the network, connect a cross-

over cable, and import the backed-up stores. While the Key and Trust Stores are backed up with the station, they are not part of a station copy.



Do not mix secure platforms with platforms that are not secure on the same network. All controllers and Supervisor stations must be secure.



Do not use self-signed certificates. In a CA-signed certificate, the **Issued By** property is not the same as the **Subject**.



Do not use guest accounts. They are easy to hack.



Do not use default passwords or passwords that can be easily guessed by attackers, such as birth dates, short words, and real words. Use different passwords for each entity. For example, use different usernames and passwords for your system password, platform credentials and station credentials. Implement strong passwords and change them frequently. Store and use passwords securely, strictly controlling access to file systems.



Do not rely on an NTP (Network Time Protocol) server that you do not directly control. If your Niagara network depends on an external NTP server for the time of day, and that server is compromised or spoofed, your Niagara system may be harmed. For example, locks may be turned off, the alarm system disabled, etc. If you use an NTP server, it must be an internal server that is physically controlled by your trusted organization.



Be warned. If your Niagara system is dependent on an external weather service, and if that weather service is compromised or spoofed, any logic in your system that uses the temperature for heating or cooling, or any other purpose may be harmed.

## Security best practices

In today's world, ensuring the security of your device network is extremely important. While managing digital certificates and passwords may seem like an excessive burden, the cost of the alternative is so substantial that you must assign resources and take the time to implement the best practices covered by this topic.



Always upgrade your platform and station to the latest software version. Install all patches and software updates. To verify that your systems are current, visit the Niagara Framework Security Center ([https://community.niagara-central.com/ord?portal:/dev/wiki/Security\\_Center#](https://community.niagara-central.com/ord?portal:/dev/wiki/Security_Center#)).



Physical security is crucial. Secure all computer equipment in a locked room. Make sure that each station is only accessible by authorized users.



Physically protect wiring to prevent an unauthorized person from plugging in to your network.



Use digital certificates to secure data transmission over wires or wireless connections. If you must connect a host station directly to the public Internet, make sure you are using CA-signed certificates.



If your company is acting as its own CA (Certificate Authority), your signed CA root certificate must be separately installed in each station's **User Trust Store** and each browser.



Physically protect the medium (usually a USB flash drive) you use to store exported certificates and keys.





Install browsers using only a trusted installation program. The program you use installs third-party certificates from CAs, such as VeriSign and Thawte. These must be trustworthy certificates.



For high-traffic stations (especially stations that provide public access to a controller network), secure **Niagara** with a separate certificate from that used for your **FoxService** and **WebService**.



Back up each station regularly. Embedded systems, such as JACE controllers write audit information to a rolling buffer. To avoid losing a station's audit trail, regularly export audit histories to a Supervisor station.

## Document change log

This log provides the date this document was released and listings of any document updates.

- Initial release document: August 18, 2015



# Chapter 1 Secure communication (TLS)

## Topics covered in this chapter

- ◆ Client/server relationships
- ◆ Certificate set up
- ◆ Configuring secure platform communication
- ◆ Configuring secure station communication
- ◆ Securing email
- ◆ Certificate renewal
- ◆ Deleting a certificate
- ◆ Secure communication troubleshooting

A Public Key Infrastructure (PKI) supports the distribution and identification of public encryption keys used to protect the exchange of data over networks, such as the Internet. PKI verifies the identity of the other party and encodes the actual data transmission. Identity verification provides non-repudiated assurance of the identity of the server. Encryption provides confidentiality during network transmission.

To provide secure networks using PKI, Niagara 4 supports the TLS (Transport Layer Security) protocol, versions 1.0, 1.1 and 1.2. TLS replaces its predecessor, SSL (Secure Sockets Layer).

Each Niagara installation automatically creates a default certificate, which allows the connection to be encrypted immediately. However, these certificates generate warnings in the browser and Workbench and are generally not suitable for end users. Creating and signing custom digital certificates allows a seamless use of TLS in the browser, and provides both encryption as well as server authentication.

**NOTE:** Verifying the server and encrypting the transmission does not secure data stored on a storage device. You still need to restrict physical access to the computers that manage your building model, set up user authentication with strong passwords, and secure components by controlling permissions.

Niagara 4.0 supports and uses secure communication by default. You do not need to purchase an additional license.

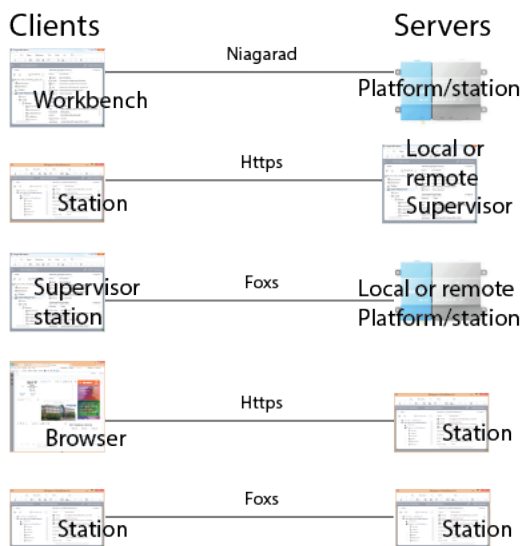
Security is an ongoing concern. While you will find much valuable information in the secure communication topics, expect future updates and changes.

## 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 1 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 Niagaraad. 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**.

## Certificates

Identity verification uses certificates to verify the identity of a server so that communication is trusted. Certificates serve a variety of purposes depending on how you configure the certificate's **Key Usage** property.

Niagara supports these types of certificates:

- A *CA (Certificate Authority) certificate* is a self-signed certificate that belongs to a CA. This could be a third party or a company serving as its own CA.
- A *root CA certificate* is a self-signed CA certificate whose private key is used to sign intermediate and server certificates in a trusted certificate tree. With its private key, a root CA certificate may be exported, stored on a USB drive in a vault, and brought out only when certificates need to be signed. A root CA certificate's private key requires the creation of a password on export and the provision of the same password when you use it to sign other certificates.

A variety of certificate-signing software tools are available. You are not required to use Niagara Workbench to sign your server certificates. When signing server certificates using Workbench, your root CA certificate should reside in the Workbench **User Key Store** on a physically secure computer.

When exported with only its public key, the root CA certificate may be freely distributed. You import it into a client's **User Trust Store**. When the `Subject` of the root CA certificate in a **User Trust Store** is the same as the `Issuer` of a server certificate, the keys match, and all other requirements are met, a client can trust that the server is authentic.

- An *intermediate certificate* is a CA certificate signed by a root CA certificate that is used to sign server certificates or other intermediate CA certificates. Using intermediate certificates isolates a group of server certificates.
- A *server certificate* represents the server-side of a secure connection. A unique server certificate resides with its public and private keys in the **User Key Store** of each server (platform or Supervisor). When a client initiates a connection, the server sends this signed certificate to the client. If a root CA certificate in either the client's **System** or **User Trust Store** is able to validate the signature of the server certificate, secure communication begins.

The **User** and **System Trust Stores** are associated with the client side of the relationship. If no third-party root CA certificate can be found in the client's **System Trust Store**, each entity that functions as a client requires a root CA certificate in its **User Trust Store**.

The **User Key Store** is associated with the server side. To function as a server, each entity (platform/station) requires its own unique server certificate that has been signed by the private key of the root CA certificate. If no such certificate exists in the client's trust stores, a user may choose to approve the certificate. This creates an exemption in the **Allowed Hosts** list. While communication is secure, it is better to use signed server certificates.

You may set up a separate certificate for each protocol.

**NOTE:** While you may configure a platform and station (as server) with separate server certificates, for simplicity they usually use the same server certificate.

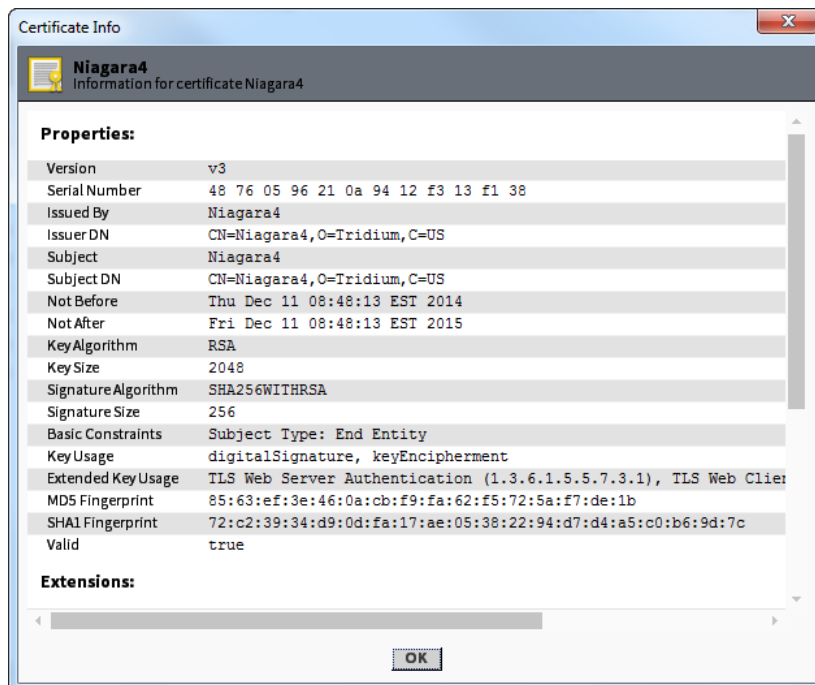
### ***Self-signed certificates***

A self-signed certificate is one that is signed by default using its own private key rather than by the private key of a root CA (Certificate Authority) certificate.

Two types of self-signed certificates are used in a NiagaraNetwork:

- A *root CA certificate* is implicitly trusted because there is no higher authority than the CA (Certificate Authority) that owns this certificate. For this reason, CAs, whose business it is to endorse other people's certificates, closely guard their root CA certificate(s) and private keys. Likewise, if your company is serving as its own CA, you should closely guard the root CA certificate you use to sign intermediate and server certificates.
- A *default, self-signed certificate*: The first time you start an instance of Workbench, a platform or station after installation (commissioning), the system creates a default, self-signed Niagara 4 server certificate with the alias of `tridium`.

Figure 2 Example of a self-signed certificate



Since the CN (Common Name) of the Issuer DN (Distinguished Name) and Subject DN are the same (Niagara4), the certificate is self-signed using its own 2048-bit, private key. The purpose of a self-signed certificate is to allow secure access to the platform and station before a trusted certificate tree with signed server certificates is established. Since a client cannot validate this type of certificate, it is not recommended for robust security if used as a server certificate.

When presented with a self-signed certificate, always confirm that it is the expected certificate before you manually approve its use. Once approved, you do not have to approve the certificate each time you make a connection to the server.

**NOTE:** Do not export this certificate and import it into any store of another platform or station. Although possible, doing so decreases security and increases vulnerability.

To minimize the risk of a man-in-the-middle attack when using self-signed certificates, all your platforms should be contained in a secure private network, off line, and without public access from the Internet.

#### CAUTION:

If you intend to use self-signed certificates, before you access the platform or station from Workbench for the first time, make sure that your PC and the platform are not on any corporate network or the Internet. Once disconnected, connect the PC directly to the platform, open the platform from Workbench, and approve its self-signed certificate. Only then should you reconnect the platform to a corporate network.

## Keys

A pair of asymmetric keys (one public and the other private) makes server verification and encryption possible. The term "asymmetric" means that the two keys are different, but related. The system can use the private key to read messages encrypted with the public key and vice versa.

The signing of certificates with the private key is required to verify authenticity. Both keys are required to encrypt information. In advance, key generation software running on a JACE controller or station generates this pair of asymmetric keys.

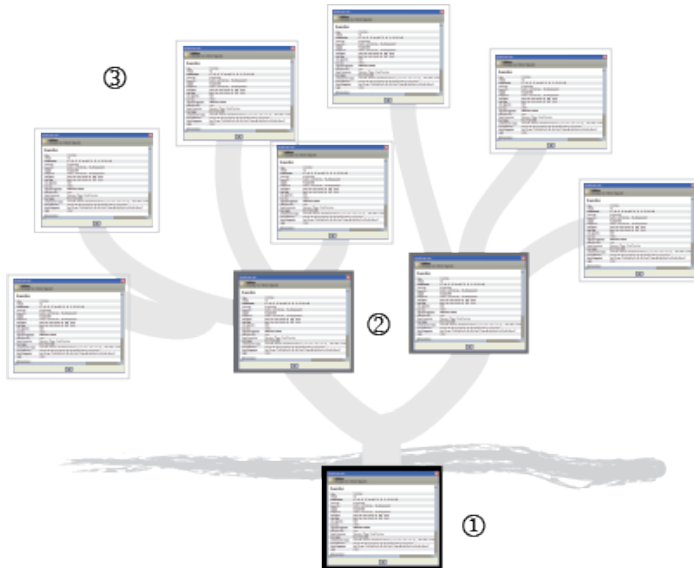
- A *public key* is a string of bytes included in the *certificate*. This key resides in the server's **System** or **User Trust Store** and is used to identify the authenticity of the connecting client certificate.

- A *private key* is also a string of bytes that resides on the server. The root CA certificate's private key must be physically protected for a certificate tree to remain secure. A private key must not be sent via email, and, if necessary, should be physically transported (on a flash drive or other medium that is not connected to the internet).

### About setting up identity verification

A certificate is an electronic document that uses a digital signature to bind a *public key* with a person or organization. Identity verification uses multiple certificates in a trusted *certificate tree*. Setting up identity verification may involve a third-party CA (Certificate Authority) or you may decide to serve as your own CA.

Figure 3 Certificate tree



In the illustration above:

1. Below the ground is the root CA certificate.
2. The major branches represent intermediate certificates.
3. The leaves are the server certificates.

How many certificates you need depends on your configuration. At a minimum you need a unique server certificate for each server (controller) and a single root CA certificate to sign your server certificates. If your company is large, you may decide to create an intermediate certificate for each geographical division or location. An individual server may have multiple certificates: one each to secure Fox, Http and Niagara (platforms) connections. Although each platform and station usually share the same certificate, you may create separate certificates for each platform and station.

If your network is large and getting thousands of certificates signed would be difficult, you may sign a wildcard certificate. Instead of identifying a specific IP or domain (for example, server1.domain.com), a wildcard certificate uses \*.domain.com.

This topic summarizes the process of securing a single server:

1. You create a server certificate that contains a pair of asymmetric keys (public and private) and company metadata (information, including name, address, etc.). At this stage the server certificate is *self-signed* using its own private key. You can tell that a certificate is self-signed if the certificate *Issuer* and *Subject* are the same.
2. You generate a CSR (Certificate Signing Request) from the server certificate using the **Certificate Management Cert Request** button and save the .csr file.

**NOTE:** Once the CSR has been created, do not delete the original certificate from the secure computer on which it is stored. Later in the process you will receive a signed certificate and import it back into the **User Key Store** where its public key must match the private key of the original certificate. Creating a new certificate with the same name does not generate the same key pair and results in errors when you try to import the signed certificate. If it is absolutely necessary (for example, if the computer on which the certificate is stored is vulnerable), you may export the original certificate with its private key and import it into the **User Key Store** when you receive the signed certificate. But, ideally, you should leave the original certificate in the **User Key Store** of the original secure host.

3. You send the CSR to a CA (Certificate Authority) with a request to verify company identity. (This transaction usually includes money.)

The CA owns a pair of keys and a trusted root CA certificate. This certificate is self-signed and serves as the primary root of the certificate tree. The CA stores this certificate usually in a vault on a computer that is not connected to the Internet.

4. After approving your company's information, the CA extracts the public key and metadata from the CSR to create a new certificate with the CA name as the *Issuer*. You can tell that a certificate has been signed by a CA because the *Issuer* and *Subject* are different.
5. The CA then uses the *private key* of its root CA certificate to sign this new certificate.
6. The CA securely returns the signed certificate to you.

For example, the CA may compress both the new server certificate and a copy of the root CA certificate containing only the public key with password protection, put both on a website, email the links to you, and phone you with the password for the compressed, password-protected files. The public key does not have to be protected and can be emailed.

You import the signed server certificate into the **User Key Store** on the server (controller). The imported certificate must match the certificate that created the CSR in the first place.

7. You may delete the self-signed certificate (that temporarily provided security until the signed server certificate was ready) from the **User Key Store**.

### ***How certificates verify identity***

Once you set up a certificate tree, identity verification takes place during the client/server handshake, before transmission begins and before the system authenticates each user by prompting for credentials (user name and password).

This is how digital certificates verify identity:

1. When a client connects to a server, the server sends its server certificate to the client.
2. The client station validates the server certificate's signature using the certificates in its **System** and **User Trust Store**. A client browser does the same. Each browser has a trust store of root CA certificates.
3. If the signature is valid, a trusted connection between server and client is established and encrypted communication begins. If the signature is not valid, the station or browser notifies the client and communication does not begin.
4. You may choose to approve a rejected certificate if you know that, although unsigned, it can be trusted.

**NOTE:** Always verify the issuer name on any certificate presented by the system as untrusted. Do not approve a certificate from an entity that you do not recognize.

## **Encryption**

Encryption is the process of encoding data transmission so that it cannot be read by untrusted third parties. TLS uses encryption to transmit data between the client and server. While it is possible to make an unencrypted connection using only the `fox` or `http` protocols, you are strongly encouraged not to pursue this



option. Without encryption, your communications are potentially subject to an attack. Always accept the default Foxs or Hhttps connections.

The following summarizes how encryption works:

1. At the start of a TLS session, the system encrypts the server/client handshake using the client and server certificates' key pairs.
2. During the handshake the system verifies server identity and negotiates encryption keys.
3. Once communication is established, identity verification is no longer needed, and encrypted data transmission begins using the negotiated keys.

Key size is directly related to encryption security. The larger (more complex) the key, the more secure the data transmission. Large keys do not slow encryption, but they do take longer to initially generate.

## Naming convention

The **User Key Store**, **User Trust Store**, and **System Trust Store** form the heart of the configuration. Certificates look a lot alike, and the various default self-signed certificates are named identically. While developing a naming convention is not a requirement (the system will function just fine if the certificates are called "cert1," "cert2," etc.), a consistent naming scheme can make the process much easier to follow.

Consider using the **Alias** of each certificate to identify the certificate's purpose. Certificate aliases might include:

- The words "root," "intermediate," "server," "client"
- The geographic location
- The store in which the certificate resides (**User Key Store**, **User Trust Store**)
- The host name of the server
- The IP address of the server

## Certificate set up

Configuring server ID verification using digital certificates involves accessing the appropriate stores; creating, and signing certificates; exporting certificates for backup purposes; importing certificates into **User Key Stores**; and importing the root CA certificate into the client's **User Trust Store**.

As a best practice, you should use the target platform or station to create the initial server certificate and CSR (Certificate Signing Request). Then use the **Workbench User Key Store** to sign all server certificates at one time, exporting them and importing each into the appropriate platform/station **User Key Store**. This is most easily accomplished if you have all controllers together in your office before taking them out to the field.

### CAUTION:

To ensure security, always configure certificates using **Workbench** on a computer that is disconnected from the Internet and your company network. Use this computer in a secure physical location.

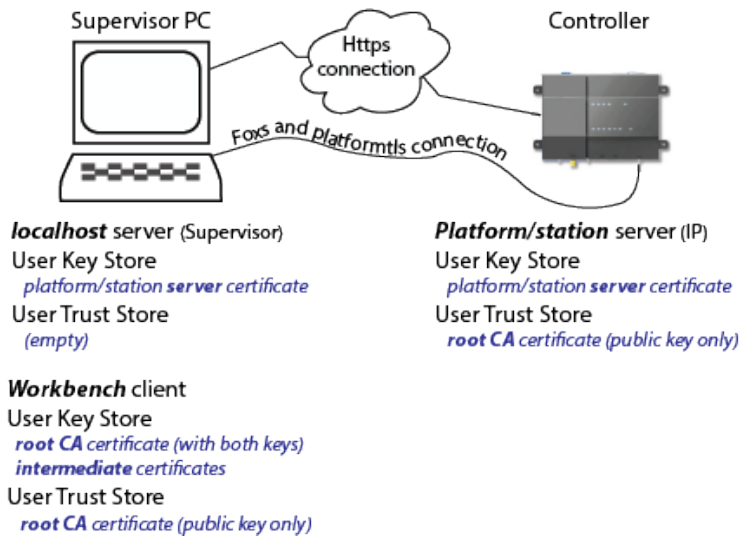
You may use a third-party CA (Certificate Authority), such as VeriSign or Thawte to sign your server certificates, or you may serve as your own CA.

## Workbench checklist

This checklist assumes that you are serving as your own CA (Certificate Authority). It summarizes the steps for setting up digital certificates using the **Workbench User Key Store** of your laptop computer.

If you are experienced managing digital certificates using the Niagara Framework, this checklist may be all you need to set up communication security for Niagara 4. If you are new to digital certificates, take a few minutes to browse through the topics.

Figure 4 Certificate locations



Use the illustration above to visualize the certificates you need and where to access them in the system. Use the check list to make sure you perform all necessary configuration tasks.

- Computer and device network disconnected from the company LAN and global Internet. To learn more about the importance of physical security, see [About station security, page 7](#) and [Security precautions, page 7](#).
- Required certificates identified: one root CA certificate, two or more intermediate certificates (optional) and one server certificate per controller. To learn more about the various certificates, see [Certificates, page 12](#), [About setting up identity verification, page 15](#), [How certificates verify identity, page 16](#), and [Certificate set up, page 17](#).
- Logical certificate naming convention established (a naming convention is not required, but it will help you keep track of your certificates). For more information see, [Naming convention, page 17](#).
- CSR folder structure under the **certManagement** folder in the `niagara_user_home` created. To learn more, see [Creating a CSR folder structure, page 24](#).
- Root CA certificate and any intermediate certificates created. For information about which certificates you need, see [Certificates, page 12](#)
- CSR for each intermediate certificate created. For more information, see [Creating a CSR, page 27](#).
- Any intermediate certificates signed using the root CA certificate. For how to sign certificates, see [Signing a certificate, page 28](#).
- Any signed intermediate certificates imported back into the Workbench User Key Store. For how to import certificates, see [Importing the signed certificate back into the User Key Store, page 28](#).
- Backup of the root CA certificate and the signed intermediate certificates created. For how to create a backup, see [Exporting a certificate, page 29](#).
- Root CA certificate with only its public key exported in preparation to import it into the platform/station Trust Stores. For more information, see [Exporting a certificate, page 29](#)

## Supervisor checklist

Use this checklist to verify that you completed all required tasks to set up a new Supervisor platform and station.

- NiagaraNetwork** enabled. For specific steps, see [Enabling the NiagaraNetwork, page 21](#).

- Controller(s) set up as Supervisor client(s). For specific steps, see [Confirming client/server relationships, page 22](#).
- Server certificate for the Supervisor platform/station created. For more information, see [Creating a certificate, page 25](#).
- CSR for the server certificate generated. For the specific procedure, see [Creating a CSR, page 27](#).
- Server certificate CSR signed using the root CA or intermediate certificate's private key. For the specific procedure, see [Signing a certificate, page 28](#).
- Signed server certificate imported back into the Supervisor platform/station's **User Key Store**. For more information, see [Importing the signed certificate back into the User Key Store, page 28](#).
- The existence of the third-party's root CA certificate in the Supervisor's Workbench **System Trust Store** confirmed or root CA certificate imported into the Supervisor's Workbench **User Trust Store**. For more information, see [Importing a certificate into a User Trust Store, page 29](#).
- Certificate .pem file deleted. For the specific procedure, see
- Confirmed platform (Niagarad) enabled. For the specific procedure, see [Configuring secure platform communication, page 31](#).
- Secure **FoxService** confirmed (the default) and **Foxs Cert** selected. For the specific procedure, see [Configuring secure station communication, page 32](#)
- Secure **WebService** confirmed (the default) and **Https Cert** selected. For the specific procedure, see [Configuring secure station communication, page 32](#)

## Platform and station checklist

Use this checklist to verify that you completed all required tasks to set up a new platform and station.

- NiagaraNetwork** enabled. See [Enabling the NiagaraNetwork, page 21](#).
- Supervisor set up as controller client. For more information, see [Confirming client/server relationships, page 22](#).
- Server certificate for each platform/station created in the User Key Store. If needed, separate server certificates for each communication protocol: foxs, https, and platformtls created. For how to create a certificate, see [Creating a certificate, page 25](#).
- CSR generated for each server certificate in the platform/station **User Key Store**. For the specific procedure, see [Creating a CSR, page 27](#).
- CSR(s) signed by root CA or intermediate certificate's private key. For the specific procedure, see [Signing a certificate, page 28](#).
- Signed server certificate(s) imported back into the platform/station **User Key Store**(s). For more information, see [Importing the signed certificate back into the User Key Store, page 28](#)
- Intermediate signed certificate(s) .pem files deleted. For the specific procedure, see
- Signed server certificate imported into each platform/station's **User Key Store**. For more information, see [Importing a certificate into a User Trust Store, page 29](#)
- If using a third-party CA, the existence of the third-party's root CA certificate in the platform/station's **System Trust Store** confirmed.
- If serving as the CA, company's root CA certificate imported into the platform/station's **User Trust Store**.
- Root CA certificate (.pem file) imported into each client's **User Trust Store**. You may distribute this public certificate via email.
- Confirmed platform (Niagarad) enabled and correct certificate assigned. For the specific procedure, see [Configuring secure platform communication, page 31](#).

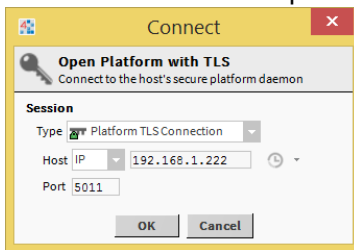
- Secure **FoxService** confirmed (the default) and **Foxs Cert** selected. For the specific procedure, see [Configuring secure station communication, page 32](#)
- Secure **WebService** confirmed (the default) and **Https Cert** selected. For the specific procedure, see [Configuring secure station communication, page 32](#)

## Opening a secure platform connection (niagarad)

Even before you configure digital certificates to provide server identity verification, every connection you make from a client to a server can be secure because you can manually verify the authenticity of the server.

**Step 1** Right-click **My Host** (for supervisor) or an IP address (for a controller) and click **Open Platform**.

The **Connect** window opens with **Platform TLS Connection** already selected.



This window identifies the entity to which you are connecting: your local computer, a supervisor platform, or a controller with an IP address.

**Step 2** If needed, enter the host IP and click **OK**.

If you are accessing the platform for the first time, the system displays an identity verification warning.

This message is expected for these reasons:

- The **Subject** or **CN (Common Name)** of the default **tridium** certificate (**Niagara4**) does not match the host name, which is usually the host IP address or domain name.
- The default certificate's **Issued By** and **Subject** are the same indicating that the certificate is self-signed. No third-party **CA (Certificate Authority)** has verified the server's authenticity.
- The certificate is signed, but no root **CA** certificate in the client's **User** or **System Trust Store** can verify its signature.

**Step 3** If you are presented with this warning and a certificate, make sure you recognize the certificate's **Issued By** and **Subject** properties.

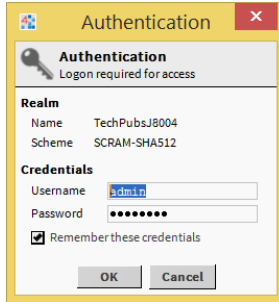
**CAUTION:** Do not approve a certificate if you do not recognize these properties. The weakest link in the security chain is the user who simply clicks **OK** without thinking.

**Step 4** Assuming that this is the default **tridium** certificate, which can be trusted, click **Accept**.

Accepting the certificate creates an approved host exemption in the platform/station **Allowed Hosts** list.

**NOTE:** Although the name of the default certificate (**tridium**) is the same for each controller and for **Workbench**, the content of each certificate is unique. Do not attempt to export and use the same **tridium** certificate for each controller in your network.

The system asks you to enter or confirm your platform credentials.

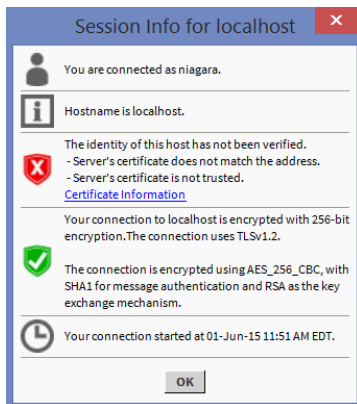


Step 5 Enter your platform credentials and click **OK**.

The platform is now connected over a secure connection. All data transmitted are encrypted. If you logged on for the first time and accepted the default certificate, only the server's identity cannot be validated.

Step 6 To confirm that you are using the self-signed certificate, right-click **Platform** in the Nav tree and click **Session Info**.

The system displays session information.



- The red shield with the X indicates that the handshake was unable to verify the authenticity of the server's certificate. To view the certificate, click the link ([Certificate Information](#)).
- The green shield with the check mark indicates that encryption is enabled. In this example, the secure connection is using `TLSv1.2` as the protocol and data is encrypted using `AES_256_CBC` (Advanced Encryption Standard) with `SHA1` (hash function) and `RSA` (Rivest-Shamir-Adleman), the most widely used public key cryptography algorithm.

Step 7 Click **OK**.

The tiny lock on the platform icon in the Nav tree indicates a secure, encrypted connection.

## Enabling the NiagaraNetwork

The NiagaraNetwork provides the physical connections for data transmission. Secure communication ensures that data are transmitted securely between trusted entities.

Step 1 Right-click the node in the **Drivers** container and click **Views**→**Property Sheet**.

Step 2 Expand the **NiagaraNetwork** property.

Step 3 Confirm that the `true` check box is selected for **Enabled**.

## Confirming client/server relationships

At any given time the Supervisor station may be the client of a controller station and vice versa. This procedure confirms that a client for the Supervisor station exists in the controller station and a client for the controller station exists in the Supervisor station.

### Prerequisites:

- Step 1 Expand the **Drivers→NiagaraNetwork** node in the Supervisor Nav tree. It should contain a node for each controller.
- Step 2 Expand the **Drivers→NiagaraNetwork** in a controller Nav tree. It should contain a node for the Supervisor station.
- Step 3 If either node does not exist, discover the station.

## Certificate stores and certificate creation process

Certificate management uses four stores to manage certificates: a **User Key Store**, **System Trust Store**, **User Trust Store** and **Allowed Hosts** list.

- The **User Key Store** contains a client's server certificate. For each certificate, this store contains both a public key, and its matching private key. In addition, this **User Key Store** contains the self-signed certificate initially created when you booted the platform for the first time.
- The **System Trust Store** comes pre-populated with standard public certificates: root CA certificates from well-known Certificate Authorities, such as VeriSign, Thawte and Digicert.
- The **User Trust Store** contains a company's own public root CA certificate(s).
- The **Allowed Hosts** list contains the certificates of servers for whom no trusted certificate exists in the client's **System** or **User Trust Stores**. This includes servers for whom the host name of the server is not the same as the `Common Name` in the server certificate. You can approve the use of these certificates on an individual basis.

The **Trust Stores** contain root CA certificates used to sign the server certificates stored in each server's **User Key Store**. If the `Subject` of a root CA certificate in one of the client's **Trust Stores** is the same as the `Issuer` on a server's certificate, the keys match, and all other requirements are met, then the client can trust that the server is authentic.

The most secure method for creating and signing server certificates is to connect your Supervisor PC directly to a controller using a crossover cable, then follow these general steps:

1. Create a server certificate in the **User Key Store** of the platform/station.
2. Using the server certificate in the **User Key Store** of the platform/station, create a CSR (Certificate Signing Request). A CSR is a file with the extension: `.csr`.
3. Use the root CA certificate in your Workbench **User Key Store** to sign the server certificate's CSR file. The signing process creates a new certificate file with the extension `.pem`.
4. Import the `.pem` file back into the **User Key Store** of the platform/station.

### Accessing the stores

The system supports two sets of stores: a set for Workbench, and another shared set for each platform and station. Workbench provides access to each set of stores.

- Step 1 Launch Workbench or Workbench in the browser.
- Step 2 Open a supervisor or remote station.
- Step 3 Do one of the following:
  - a. To access the Workbench stores, click **Tools→Certificate Management**.

Since Workbench is always a client, it requires only a trusted root CA certificate in its **System** or **User Trust Store**. Even though it functions only as a client, you may use it to create your root CA and any intermediate certificates, and to sign all server certificates at one sitting.

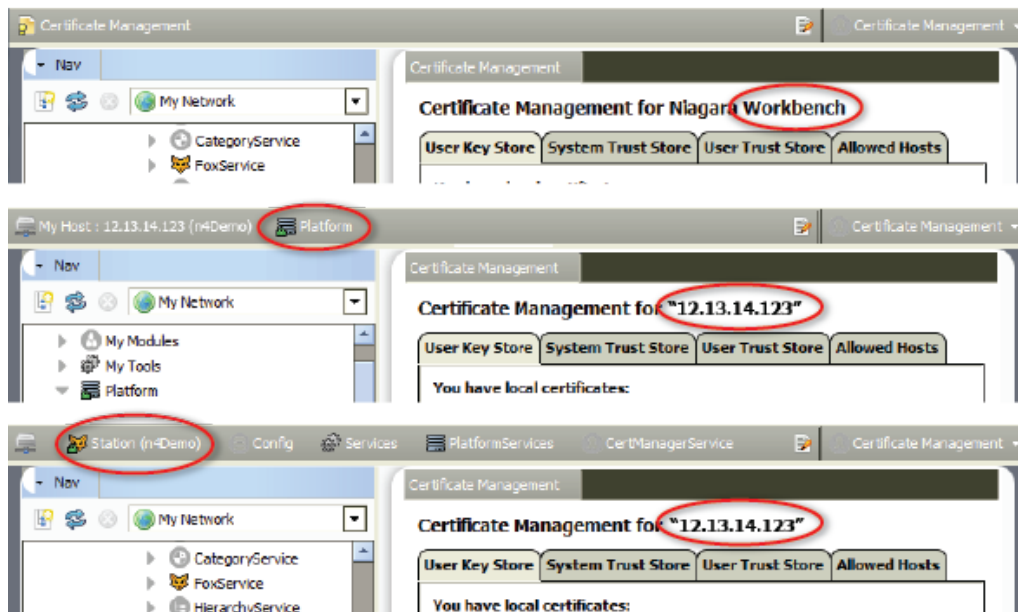
- b. To access the platform/station stores, expand **Platform** and double-click **Certificate Management** in the Nav tree. To access the stores this way, the station must be idle.

Since a platform/station may function at different times as a client or a server, it must have a server certificate in its shared **User Key Store** and a trusted root CA certificate in its **System** or **User Trust Store**.

- c. If your station is running, you access the platform/station stores by expanding **Station**→**Config**→**Services**→**PlatformServices** and double-clicking **CertManagerService** in the Nav tree.

The station shares the root CA certificate with the platform, but may have its own server certificate (when it functions as a server) located in the same **User Key Store**.

**Step 4** To confirm that you have accessed the correct stores, check the name in the screen title and above the stores.



The platform and station stores are actually the same stores.

### Stores folder structure

The Workbench and platform/station stores reside in separate locations on a Supervisor and platform.

The following table lists the default user homes that contain the stores, which are considered configuration files. If the folder paths have been changed, these no locations longer apply. `username` is the Windows user name of the person starting the Workbench application.

Table 1 Default user homes

Stores	Nav tree node	Default folder path
Workbench <sup>1</sup>	<b>My Host</b> → <b>My File System</b> → <b>User Home</b> → <b>security</b>	C:\Users\username\Niagara4.0\security
Supervisor or engineering workstation <sup>2</sup>	<b>Platform</b> → <b>RemoteFileSystem</b> → <b>User Home (Read Only)</b> → <b>security</b>	ProgramData\Niagara4.0\security
JACE controller <sup>3</sup>	<b>Platform</b> → <b>RemoteFileSystem</b> → <b>User Home (Read Only)</b> → <b>security</b>	\home\niagara\security

1. Each Workbench user has their own User Home.
2. The platform and station stores (same stores) are in the Platform daemon User Home of the Supervisor or any engineering workstation.
3. The platform and station stores (same stores) are in the Platform daemon User Home of the JACE controller,

### **Stores file names**

The folders that contain the Workbench and platform/station stores each contain a set of three data files, one per type of store.

**NOTE:** Only the appropriate Workbench, platform or station tools may be used to modify these data files. Attempts to modify them by other means will render them corrupt and unusable.

- `keystore.jceks` is the **User Key Store**. In Workbench it contains a company's root and intermediate certificates. In a server it contains the server's server certificate.
- `cacerts.jceks` is the **User Trust Store**. In a client it contains the root CA certificate(s) with only its public key.
- `exemptions.tes` is the **Allowed Hosts** list. In a client it contains the certificate for hosts (servers) with whom the client may securely communicate even though the client either:
  - does not have a root CA certificate in its **System** or **User Trust Store** for the server, or
  - may have a matching root CA certificate, but the `Common Name` or `Alternate Server Name` of the server certificate is not the same as the host name of the server being authenticated.

**NOTE:** A certificate in a Workbench **User Key Store** may have the same name as a certificate in a platform/station **User Key Store**, but they may not be the same certificate. Similarly, files in these stores may have differing alias names, and in fact contain the same public keys. It is the public/private key pair that defines a certificate, not the certificate's name.

### **CSR folder structure**

You may create CSRs (Certificate Signing Requests), store them in, and import signed certificates (.pem files) from any folder on your laptop. The default location is a working folder in the user file space.

The first time you access the Certificate Management view from Workbench, the system creates an empty **certManagement** folder in the following location:

`c:\Users\username\Niagara4.0\certManagement`, where `username` is the name you use to log in to the computer.

In the Nav tree, this location is: **My File System**→**User Home**→**certManagement**.

This folder, in Workbench's user space, is a working folder for storing CSRs and .pem files exported and imported by the **Certificate Manager**. Within this folder, you may create a structure for managing exports and imports or you may use a different location for exports and imports.

**NOTE:** Do not confuse the `certManagement` folder with the `certificates` folder that stores one or more certificates used to validate the authenticity of Niagara system licensing files. The `certificates` folder has nothing to do with secure communication.

The platform and station folders do not have a `certManagement` folder.

### **Creating a CSR folder structure**

A CSR (Certificate Signing Request) folder structure helps you organize a large number of server certificates for easy retrieval. You create this structure under the **certManagement** folder, which is an automatically-created folder in your personal `niagara_user_home`.

**Prerequisites:**



**Step 1** Using Windows Explorer, locate your `niagara_user_home`: `C:\Users\username\niagara4.0\brand\certManagement`, where

`username` is your name or other text used to identify you as the user of your computer.

`brand` is your company name or other name used to label personal information.

**Step 2** Create folders under `certManagement`.

For example:

```
certManagement
rootCertificate
intermediateCertificates
serverCertificates
```

## Creating a certificate

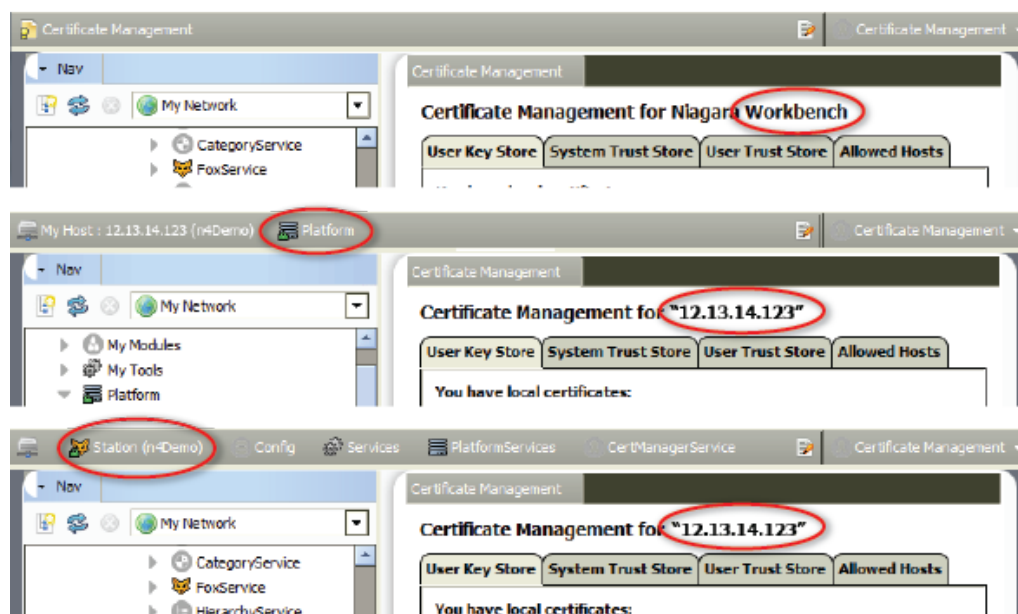
Certificates reside in the **User Key** and **Trust Stores**. The Workbench stores are separate from the single set of stores shared by the platform and station.

There are several ways to create a certificate.

- You may use the default server certificate that is automatically generated when you start Workbench or boot a platform for the first time.
- To create a new server certificate for the current *platform* and *station*, use either the platform's **Certificate Management** tool or the station's **CertManagerService** (under the **PlatformServices** node in the station Nav tree). Both methods access the same stores.
- To create a root CA certificate, sign server certificates, and export them as needed, use the Workbench stores, which you access on a secure computer by clicking **Tools**→**Certificate Management**.

**TIP:** While this is not a requirement, as a best practice you should disconnect both your computer and the platform from the Internet and company LAN, then connect your Supervisor computer to the platform using a crossover cable.

**Step 1** Check the title at the top of the view to ensure that you are focused on the **User Key Store** for the correct stores (Workbench or platform/station).



The platform and station stores are the same stores. Which to use depends on how you are connected to the platform/station.

Step 2 Click the **New** button.

The **Generate Self Signed Certificate** window opens

Step 3 Fill in the properties.

- Use **Alias** to identify the type of certificate (root, intermediate), server, company, geography or department. The **Alias** can be the same as the **Common Name (CN)**.
- **Common Name (CN)** should be the same as the host name, which is how a server identifies itself. The common name becomes the **Subject** (also known as the Distinguished Name). The IP address of a controller or its Fully Qualified Domain Name (FQDN) is an appropriate **Alias** and **Common Name** for a JACE controller or Supervisor station.

An FQDN is the **Hostname** plus the **Primary Dns Suffix**. For a computer, you can see this name in My Computer Properties: "Full computer name." For a controller, there is no good place to see this name, but it would be something like: mycontroller.mydomain.com or mycontroller.mydomain.net.

**NOTE:** Do not use the same name for **Common Name (CN)** of a server certificate that you use for a root or intermediate certificate's **Common Name (CN)**.

- Although **Locality** and **State/Province** are not required and are arbitrary, leaving them blank generates a warning message. Third-party CAs may not sign certificates without these properties defined.
- The two-digit **Country Code** is required and must be a known value, such as: US, IN, CA, FR, DE, ES, etc. (See [countrycode.org](http://countrycode.org) for a list.)
- **Not Before** and **Not After** define the period of validity for the certificate.
- **Key Size** defaults to 2048. A larger key improves security and does not significantly affect communication time. The only impact it has is to lengthen the time it takes to create the certificate initially.

If a third-party will sign the certificate, consult with your CA (Certificate Authority) to determine the acceptable key size. Some CAs support a limited number of key sizes.

- For **Certificate Usage**, select **CA Certificate** for a root or intermediate certificate. Select **Server** as needed for a platform/station.
- For server certificates, if **Common Name** is an IP address, use a Fully Qualified Domain Name for the **Alternate Server Name**.

Step 4 When you have filled in all information, click **OK**.

One of two things happens:

- If you are creating a root or intermediate certificate, the system prompts you to create a password for the certificate's private key. You should create and confirm a strong password.

**NOTE:** Create a root or intermediate certificate only in a Workbench **User Key Store** on a very specific and secure Workbench host. The root and intermediate CA certificates in this **User Key Store** contain the private keys used to sign all your server certificates.

- If you are creating a server certificate, the system begins the certificate generation process without requiring the creation of a password.

A pop-up window in the lower right of your screen advises you regarding the time it may take to generate the certificate. The length of time it takes depends on the key size and the platform's processing capability.

Step 5 To view the certificate, double-click it or select it and click **View**.

Step 6 Confirm that the information is correct.

**NOTE:**

To change a certificate you just created, delete it and create a new certificate. Do not delete a certificate that is already in use.

Repeat this procedure to create additional certificates.

## Password strength

To protect each certificate's private key you must supply a private key password. When backing up certificate private keys by exporting certificates, you may use an additional encryption password. (The default encryption password is the same as the private key password.) To prevent unauthorized access, your passwords need to be strong.

A strong platform or station password:

- Has 10 or more characters.
- Includes letters, punctuation, symbols, and numbers.
- Is unique for each set of credentials.

**NOTE:** You should not reuse passwords.

- Avoids dictionary words in any language, words spelled backwards or words that use common misspellings and abbreviations, sequences or repeated characters, personal information such as your birthday, driver's license, passport number, etc.

These precautions were adapted from information at microsoft.com, which provides a secure password checker you can use to test the strength of any password. Niagara 4 allows you to control password strength for user authentication. The password strength configuration for user authentication does not apply to certificate passwords.

## Creating a CSR

A CSR (Certificate Signing Request) prepares a certificate for signing by the root or intermediate CA certificate by creating a .csr file.

**Prerequisites:** For creating intermediate certificates you are viewing the Workbench stores. For creating server certificates you are viewing the platform/station stores.

**Step 1** Select the intermediate or server certificate to sign, and click **Create CSR**.

The **Certificate Request Info** window opens

**Step 2** Confirm that the certificate properties are correct and click **OK**.

One of the following happens:

- If you are preparing a CSR for a server certificate, the system displays the **certManagement** folder for you to choose the location to store the CSR.
- If you are creating a CSR for a CA certificate (root or intermediate), the **Certificate Manager** prompts you for the private key password. Enter the password and click **OK**. The system displays the **certManagement** folder for you to choose the location to store the CSR.

The **Alias** for the certificate is used as the file name of the CSR.

**Step 3** Use the default folder, or select a different folder in which to store the CSR and click **Save**.

The system displays, *CSR generation complete*.

**Step 4** To confirm completion, click **OK**.

**Step 5** If an external CA, such as VeriSign or Thawte, will sign your server certificates, follow the CSR submission procedure as required by the CA.

The CA verifies that you are who you claim to be, that each certificate is for a server your organization actually maintains, and other important information. They then return a signed server certificates (one for each server).

## Signing a certificate

In a large installation that serves at its own CA (Certificate Authority), you use your root CA certificate to sign any intermediate certificates and the intermediate certificates to sign your server certificates. In a small configuration, you may use your root CA certificate to sign all server certificates.

**Prerequisites:** The Workbench stores are open, the root CA certificate exists and a CSR for the certificate has been created.

**NOTE:** To ensure network security, always sign certificates using Workbench on a computer that is disconnected from the Internet and from the company LAN. Maintain this computer in a physically secure location.

Step 1 In Workbench, click **Tools→Certificate Signer Tool**.

The **Certificate Signing** window opens

Step 2 Click the browser icon, locate, and open the CSR for the certificate you wish to sign.

The **Certificate Signing** window expands to show the certificate details.

Step 3 Confirm that this is the correct certificate.

Step 4 Select the date on which the certificate becomes effective (**Not Before**) and the date after which it expires (**Not After**).

Step 5 For **CA Alias**, use the down arrow to select the certificate (root or intermediate) whose private key will sign this certificate.

Step 6 Supply the CA certificate's password and click **OK**.

Signing is done by the private key of the root or intermediate certificate.

The same file folder, `C:/Users/[username]/Niagara4.0/certManagement`, displays with the file name (extension: .pem) filled in for you.

You may modify this file structure to aid in the management of these files.

Step 7 To complete the signing, click **Save**.

Repeat this procedure for each CSR.

## Importing the signed certificate back into the User Key Store

Signing a certificate creates a .pem file, which is only intended for importing back into the **User Key Store** that contains the original certificate with the matching private key. For a server certificate this is the platform/station **User Key Store** that originally created the certificate and CSR. For an intermediate certificate this is the Workbench **User Key Store** on your Supervisor computer.

**Prerequisites:** You have the signed .pem files. The focus is on the User Key Store in the appropriate stores location (Workbench or platform/station).

Step 1 Click **Import**.

Step 2 Locate and select the signed certificate's .pem file (the output of the certificate signer or the .pem file you received from a third-party CA) and click **Open**.

The **Certificate Import** window opens.

Step 3 Confirm that you are importing the correct certificate and click **OK**.

If the **Alias** of the certificate you are importing is not the same as the **Alias** of the certificate you are replacing, the system prompts you for the **Alias** of the certificate to replace.

Step 4 If needed, enter the **Alias** and click **OK**.

The green shield icon appears next to the certificate `Alias` in the **User Key Store**.

Step 5 Using the operating system, delete the `.pem` file(s) from the Supervisor computer.

## Deleting `.pem` files

### Prerequisites:

Step 1

## Exporting a certificate

There are two reasons to export certificates: 1) to create a public root CA certificate for each client's **User Trust Store** and browser, and 2) to back up the company's root CA certificate and all signed intermediate and server certificates with their private keys.

As soon as you create your company's root CA certificate, any intermediate certificates and finish importing each signed server certificate (`.pem` file) back into each server **User Key Store** make a backup of all of certificates and store the backup on a flash drive in a physically secure location. You back up each certificate one at a time.

**NOTE:** To protect your backups create strong passwords and store backup media in a vault. These backups contain the keys used to sign all server certificates.

Step 1 Open the stores that contain the certificate(s) to export.

Step 2 On the **User Key Store** tab, select the certificate and click **Export**.

The system opens the **Certificate Export** window.

Step 3 Do one of the following:

In addition to the private key password, you should use an encryption password to provide double-password protection. The default encryption password is the same as the private key password.

- To create a CA certificate (root or intermediate) for importing into a client **User Trust Store**, just click **OK** (do not select `Export the private key`).
- To back up a certificate with its private key, click `Export the private key` and supply the private key password.
  - a. To use the additional protection, **deselect** `Reuse password to encrypt private key under Encrypt exported private key` and supply the additional encryption password.
  - b. To export the certificate with its private key, click **OK**

Step 4 Use the default path in the user space, or locate a different folder and click **Save**.

The system reports that the export was successful.

Step 5 To complete the action, click **OK**.

## Importing a certificate into a User Trust Store

If your **System Trust Stores** already contain the root CA certificate of the CA (Certificate Authority) that signed your intermediate and server certificates, you do not need to import anything. If you are serving as your own CA, you import only the root CA certificate into the **User Trust Store** of each client. Each platform and station share the same stores.

**Prerequisites:** The focus is on the User Trust Store in the appropriate stores location (Workbench or station).

If you are acting as a local CA, you will need to import the root CA certificate (the one with only its public key) into the **User Trust Store** of each client instance (Workbench or station) that may connect to each server. If you are using a root CA certificate signed by a third party, each client's **System Trust Store** should

already contain the vendor's root CA certificate. If not, you must obtain one from the vendor and import it into the **User Trust Store** of each client.

**NOTE:** There is no need to import the server certificates or the intermediate CA certificates to any **User Trust Store**. Each signed server certificate carries within it any intermediate and root CA certificate information.

Step 1 Select the **User Trust Store** tab.

Step 2 Click **Import**.

Step 3 Locate and select the root CA certificate's .pem file (signed by a third-party or your company's root CA certificate) and click **Open**.

The **Certificate Import** window opens.

Step 4 Confirm that you are importing the correct certificate and click **OK**.

Repeat this procedure for Workbench and each client station.

## Station health confirmation

When you finish configuring a client or server, stop and restart a secure station and check station health. The system does not validate existing connections against new certificates until you restart the station. The system does not automatically change connections from **Http** and **Fox** to **Https** and **Foxxs**, even when you enable **Https Only** and **Foxxs Only**, until you reestablish the connection.

## Viewing session information

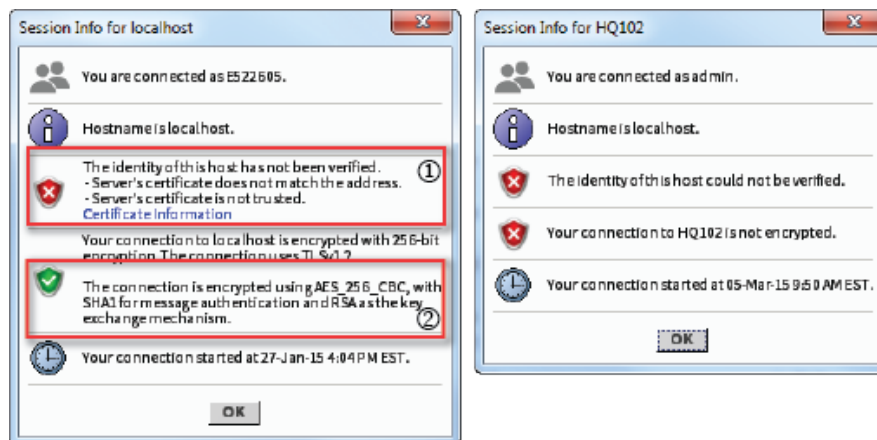
The **Session Info** window provides useful information and a graphical representation of certificate status (green and red icons).

**Prerequisites:** Workbench is running.

Step 1 To view session information, do one of the following:

- Click the Session Info icon (i) in the row of icons at the top of the page.
- Right-click the station name in the Nav tree and click **Session Info**.

The system displays one of two **Session Info** windows.



- The **Session Info** message on the left indicates that you have made a secure connection. For the Server identity section (1), a red shield with a white X indicates that the client is unable to verify the authenticity of the Fox host. There are multiple reasons why this host may not be authentic.

A green shield with a white check mark indicates that a root CA certificate in the client's **System** or **User Trust Store** was able to validate the signature on the server certificate, verifying the authenticity of the Fox host.

For the Connection encryption section (2), a red shield with a white X indicates that the Fox session connection is not sufficiently encrypted.

A green shield with a white check mark indicates that the Fox session connection is sufficiently encrypted.

Communication is the most secure when both shields are green.

- The Session Info message on the right indicates that you have made a regular connection (a connection that is not secure). Communication is the least secure when both shields are red.

**Step 2** Click the **Certificate Information** link.

The system displays the details of the Fox server certificate.

## Allowed hosts management

If you used self-signed certificates to get started, more than one exemption may be allowed in your **Allowed Hosts** list. Once you have set up signed certificates for all hosts, delete the exemptions from each **Allowed Hosts** list (Workbench, and platform/station).

- To access the Workbench **Allowed Hosts** list, click **Tools**→**Certificate Management**, and click the **Allowed Hosts** tab.
- To access the platform/station **Allowed Hosts** list, expand **Platform** and double-click **Certificate Management** in the Nav tree. Then, click the **Allowed Hosts** tab.
- You may also access the platform/station stores by expanding **Station**→**Config**→**Services**→**PlatformServices** and double-clicking **CertManagerService** in the Nav tree.

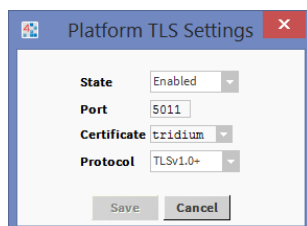
## Configuring secure platform communication

Platform and station security are independent of one another. The system defaults to enabling secure communication for both platform and station. Configuring a platform (Niagarad) for secure communication (platformtls) involves confirming the port, selecting the signed server certificate to use, and, if required, restricting the TLS protocol version.

A station's "window" into the platform-resident secure communication features is just like any other **Platform Service** under the station's **Platform Administration** node in the Nav tree. This means that anything configured for a platform is independent of whatever station is running. Follow this procedure for the Supervisor platform and all JACE controller platforms.

**Step 1** Double-click **Platform**→**Platform Administration** and double-click **Change TLS Settings**.

The **Platform TLS Settings** window opens.



The default **State** is **Enabled**. If you are using a separate certificate for verifying niagarad communications, this is where you select the certificate.

**Step 2** Configure the properties and click **Save**.

## Configuring secure station communication

This topic explains how to set up secure Foxs and Https communication for Supervisor and controller stations.

Follow this procedure for both Foxs and Webs.

- Step 1 Make a secure connection to the station.
- Step 2 Right-click **FoxService** or **WebService** under **Config→Services** in the Nav tree.  
The **Property Sheet** opens and click **Views→Property Sheet**.
- Step 3 Confirm that the `true` check box is selected for **Foxs Enabled** or **Https Enabled**.
- Step 4 From the **Foxs Cert** or **Https Cert** list, select the appropriate certificate.  
Each platform/station should have its own unique, signed server certificate. Do not use the same server certificate for more than one platform/station. If you choose to use a different certificate for your **FoxService** from that used for your **WebService**, this is where you specify it.

## Setting up client/server relationships

At any given time, the Supervisor station may be the client of the JACE station and vice versa. This procedure confirms that a client for the Supervisor station exists in the station and a client for the JACE station exists in the Supervisor.

- Step 1 In the Supervisor Nav tree, expand the **NiagaraNetwork** node under the **Drivers** container. It should contain a node for the station.
- Step 2 In the station Nav tree, expand the **NiagaraNetwork** node under the **Drivers** container. It should contain a node for the Supervisor station.

## Enabling clients and configuring them for the correct port

While not directly related to secure communication, setting up each platform/station as a client and server is important for setting up basic communication relationships.

- Step 1 If it is not already open, double-click the **NiagaraNetwork** node in the Nav tree of both the Supervisor and the controller stations.  
The **Station Manager** view opens.
- Step 2 Double-click the client station under the client in the **Database** pane.  
For the Supervisor station, this is the controller station as client; and for the controller station, this is the Supervisor station as client.
- Step 3 For each client, confirm that the **Fox Port** is set to 4911, and that **Use Foxs** is set to `true`.

## Securing email

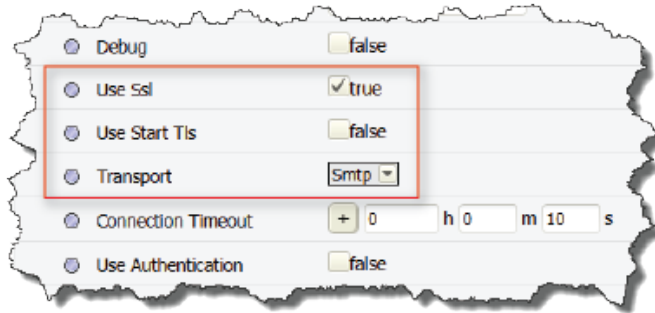
Niagara supports secure outgoing and incoming email using TLS (Transport Layer Security).

**Prerequisites:** The **EmailService** is in your **Services** container with both **IncomingAccount** and **OutgoingAccount** components. If not, add the **EmailService** component from the **email** palette before you begin. You may have multiple incoming and outgoing accounts, which allow you to set up connections to servers that support secure communication and others that may not.

Follow this procedure for both your incoming and outgoing accounts.

- Step 1 In the station's Nav tree, right-click the **IncomingAccount** or **OutgoingAccount** node under the **EmailService** container and click **Views→Property Sheet**.  
The account **Property Sheet** opens.





The system provides two secure communication options:

- The default, **Use Ssl**, encrypts the connection before it is ever opened. To do the encryption, it automatically uses either SSL v3 or TLS (depending on email server requirements). This provides the most secure data transmission since the connection is encrypted from the start.
- **Use Start Tls** makes it possible to connect to an unprotected email server. The handshake occurs without encryption, then switches to encrypt the message itself.

**Use Ssl** and **Use Start Tls** are mutually exclusive. Both may be `false`.

**Step 2** To provide secure email, set one property to `true`, and the other `false`.

The example shows the configuration when **Transport** is set to `SmtP`.

Incoming and outgoing messages use different ports for secure communication as follows:

Table 2 Email ports based on transport type

	Outgoing (SMTP)	Incoming (IMAP)	Incoming (POP3)
Not encrypted	25	143	110
Use Start Tls	587	143	110
Use Ssl	465	993	995

Not all servers follow these rules. You may need to check with your ISP (Internet Service Provider).

**NOTE:**

Do not enable or disable the **Use Ssl** or **Use Start Tls** properties without configuring the **Port**.

**Step 3** Change the **Port** to the appropriate port number (defaults are: 25 for outgoing and 110 for incoming email).

The system also provides server identity verification. For most email servers, the root certificate is already in the **System Trust Store**.

**Step 4** If no root CA certificate for the email server is in the station's **System Trust Store** (third-party signed certificate) or in the **User Trust Store** (your own certificate if you provide your own secure email server), either:

- Import your own or a third-party signed root CA certificate into the station's **User Trust Store**.
- Or, if you do not have a signed certificate yet, accept the system-generated, self-signed certificate when challenged. This creates an exemption in the **Allowed Hosts** list. Later, import the root CA certificate and delete this temporary exemption.

## Certificate renewal

Certificates expire and need to be renewed and imported again.

To renew, follow the procedures to create a new server certificate, create a CSR, obtain signing by a root CA or intermediate certificate, and import the signed certificate (.pem file) back into the **User Key Store** that contains the original certificate.

## Deleting a certificate

As a general rule, third-party certificates may be renewed but not changed. Some CAs will not allow any changes once the certificate is signed. If you need to make a change, delete the certificate and start again with a new certificate.

### ATTENTION:

Do not delete a certificate until its replacement is in place and configured. If you delete a certificate that is in use, the platform, **FoxService** or **WebService** could fail to restart. If you have the services configured for **Https Only** a secure platform connection using TLS (Platform TLS settings) or **Foxs Only**, a missing certificate could prohibit connectivity using encrypted connections. Workbench gives no warning if you delete a certificate that is currently being used by Workbench or the platform/station.

- Step 1 Open the platform.
- Step 2 Click **Tools**→**Certificate Management** .
- Step 3 Select the certificate in the **User Key Store** and click **Delete**.

## Secure communication troubleshooting

This topic suggests solutions for common connection security problems.

**When I attempt to import the signed server certificate back into the host User Key Store, I get errors.**

This may happen if you deleted the original certificate created on the host from which you created the CSR. If you backed up this certificate, import it back into the **User Key Store** and import the CSR again. Generating a new certificate with the same name does not generate the same key pair and will result in errors when you attempt to import a signed certificate whose keys do not match.

**For months I have been able to log in without being prompted to accept a certificate. All of a sudden the software is asking me to accept the certificate again.**

One or more of the following may be occurring:

- The client may no longer contain the host's root CA certificate in the **User Trust Store** (for whatever reason). Check the certificate and import a matching root CA certificate into the **User Trust Store**.
- The root CA certificate may have expired or changed and you need to import new certificates. Check the server certificate carefully to make sure it is trusted and temporarily approve it, creating an exemption in the **Allowed Hosts** list. Create or acquire a new root CA certificate and create new server certificates. Get the new server certificates signed by the new root CA certificate. Finally, import the certificates into the appropriate stores, deleting any expired certificates and any temporary exemptions you approved in the **Allowed Hosts** list.
- There may be a problem with the Fox port. Check the **FoxService** on the client **NiagaraNetwork** to ensure the correct Fox port: 4911 for Foxs; 1911 for Fox.
- You may be subject to a man-in-the-middle attack and no trusted root CA certificate exists for the attacker in the platform/station **Trust Stores**. Check the certificate's **Issued By** and **Issuer DN** (Distinguished Name) carefully. Do not manually approve a certificate for an issuer you do not recognize.

**The Session Info window (right-click Station→Session Info) shows a red shield with a white x in the section that reports host identity authentication.**

There may be a number of reasons for this:

- If you are serving as your own Certificate Authority, a platform and station requires your root CA certificate in its **User Trust Store**. When you start the platform or station for the first time the **User Trust Store** is empty. This causes the system to generate a self-signed certificate and display it for you to approve before it establishes the connection. Compare the `Issued By` and `Subject` properties. They are the same for a self-signed certificate. If you recognize the name, you can manually approve the certificate and rest assured that communication is secure. If you do not recognize the name, do not manually approve the certificate. If you are configuring the host for the first time, import your root CA certificate to the host's **User Trust Store** as soon as possible and delete the default, self-signed certificate.
- If, in a hurry, you allowed a certificate without checking its `Issued By` and `Issuer DN` (Distinguished Name), and you are worried about what you approved, open the platform/station stores (**Config→Services→Platform Services→CertManagerService**); click the **Allowed Hosts** tab; locate the certificate and, if you do not recognize it, click **Unapprove**.

**When running in a browser, the Https in the address is crossed through with a red X next to it.**

This indicates that you are using a self-signed certificate for which no client certificate exists in the browser's trust store. Using Google Chrome, the browser caches nothing. You can still access the platform and station, but system performance is less than desirable. To speed performance, set up and import your own root CA certificate into the browser's trust store, or purchase and install a signed client certificate from a CA (Certificate Authority).

**I enabled SSL and logged in using a secure connection, but the platform icon does not include the lock symbol. Why did the platform boot with a connection that is not secure?**

Most likely there is something wrong with the certificate. If a certificate fails, or for any reason secure communication cannot start, rather than lock you out of the platform, the system enables a connection without security. Restart the platform.

**NOTE:** If you have to replace a platform certificate, assuming you exported the keys, you can import them to configure the new platform for secure communication.

**I enabled SSL and logged in using a secure connection. The platform icon shows the lock symbol, but no communication is occurring.**

A firewall or secure router may be blocking or ignoring a port. Consult your firewall or router documentation for a list of blocked ports, then either unblock the port in the firewall or router, or change the port using Workbench.

**I'm using a signed server certificate, but the message "Unable to verify host identity" still appears when connecting to the platform.**

The system cannot find a root CA certificate in a **Trust Store** that matches the server certificate. Import the root CA certificate used to sign the server certificate into the **User Trust Store**.

**My platform or Supervisor private key has been compromised, what should I do?**

Get on site as quickly as possible. Take the entire network off the Internet. Configure security again for each compromised platform creating and signing all new certificates.

**When importing a root CR certificate into a client User Trust Store I get the message, "The 'Import' command encountered an error" or the certificate simply did not import.**

Click the **Details** button to view the Workbench console. Investigate these possibilities:

- You may be attempting to import a private key into the **User Trust Store**. This cannot be done.  
Export the root CA certificate from the Workbench **User Key Store** without its private key and try to import it again into the client **User Trust Store**.
- The `Issuer` of the certificate you are importing must be the same as the `Subject` of the certificate that is below it in the certificate tree (the certificate used to sign the one that is causing the error). This may

be an intermediate certificate or the root CA certificate. Beginning at the bottom of the tree, the issuer-subject relationship is something like this:

Issuer, SubjectC, DB, CA, B

Where "A" is the root CA (Certificate Authority) at the root of the certificate tree. "D" is the subject of the final server certificate in the tree. The rest are intermediate certificates.

If necessary, delete the certificate, create a new certificate, sign it using the certificate below it in the trusted certificate tree, and attempt to import again.

### I'm trying to get two stations to connect and it is not working.

If this is the first time you are making this connection, check the **Allowed Hosts** list. The station serving as the client may not have a certificate in its **User Trust Store** for the station that is serving as the server. In the **Allowed Hosts** list, analyze the exemption, then select the certificate to make sure that you recognize its **Issued By** and **Issuer DN** (Distinguished Name) and click **Approve**. Check the certificate for the correct name and port number in the **Host** column.

If you have been connecting successfully but suddenly you are unable to connect, try to figure out what changed. Check the daemon logs for an error message.

If you are using root and intermediate certificates, check the **Issuer** name on your signed intermediate and server certificates. It should be the same as the **Subject** name on the root CA certificate. When it is unable to validate the certificate tree, the software prevents communication.

**We use self-signed certificates. All hosts are approved in the Allowed Hosts list, and we've been able to connect to our platforms without getting the message that our hosts are not trusted. All of a sudden we're getting that message again. What happened?**

If the IP address of the platforms changed, the entry in the **Allowed Hosts** list is no longer valid.

**I get the message, "Cannot connect. Ensure server is running on specified port." when I attempt to log in to a secure station:**

This is a general message. A number of things could be wrong:

- There may be a problem with the controller. Ensure that the controller is connected to power and the power is on.
- You entered invalid credentials or, for some other reason, you are having difficulty logging on (the station may have stopped running). Confirm your credentials, start the platform and use the Platform Application Director to start the station, then connect again.
- Your secure **WebService** (Https) or **FoxService** (Foxy) may not be enabled (set to `true`). Both must be enabled to make a secure station connection. Make a regular station connection by clicking **File→Open→Open Station**, select **Station Connection**, provide credentials, then, on the **FoxService Property Sheet**, confirm that **Foxy** is enabled (set to `true`), close the station and connect to it again. Make sure you select a **Station TLS Connection** for **Type** in the **Connect** window.

## Default TCP ports

The various system protocols (fox, foxy, etc.) manage communication across specific ports. If a firewall or router blocks a port, communication fails. Be aware of this potential and make appropriate exemption rules where necessary.

Following commissioning, the default TCP port numbers are:

Protocol	Default port	Type of communication	Security
fox	1911	station	not secure
foxy	4911	station	secure
platform	3011	niagarad (platform)	not secure

Protocol	Default port	Type of communication	Security
platformtls	5011	niagarad (platform)	secure
http	80	browser	not secure
https	443	browser	secure
email	25	send and receive	not secure
email tls	587, 465	send and receive	secure

You may change these defaults as needed.

## Certificate management when replacing a controller

When replacing a JACE controller in the field, you may reuse backups of the **User Key Store** and **User Trust Store** from the old controller. If no station backup is available, you must generate a new server certificate and sign it or get it signed.

**Prerequisites:** You are on site. Remotely importing a security backup into a JACE controller is not recommended because you should not restore the **User Key Store** and **User Trust Store** while the station is connected to the Internet.

- Step 1 Make sure that the JACE controller is not on the Internet.
- Step 2 Reboot the controller and restore the station.
- Step 3 Either restore from the station backup, or import the stores from a previously exported file.



# Chapter 2 User authentication

## Topics covered in this chapter

- ◆ User authentication checklist
- ◆ Authentication schemes
- ◆ Station-to-station users
- ◆ Adding or editing a user
- ◆ Assigning authentication schemes to users
- ◆ Password management
- ◆ Logging on to a station
- ◆ Changing your password
- ◆ User authentication troubleshooting

User authentication is validating the identity of a subject, which can be a human user, a system, or an application. The system's approach to user authentication is designed to be extensible by supporting a variety of authentication schemes. You configure authentication properties using the **AuthenticationService** in the Nav tree.

All stations must have an **AuthenticationService**, with the **Authenticator** property for each user set to one of the supported schemes.

When a station attempts a connection, it checks the user's login credentials (user name, password) against the users under the station's **UserService**. This process is called *user authentication*. The actual process depends on the authentication scheme and on the type of connection:

- Workbench-to-station (**FoxService**)

When a user opens a station (**File**→**Open**→**Open Station**), Workbench prompts for user name and password. When using Niagara 4, this type of authentication defaults to the **DigestScheme**. Connections to older software versions (NiagaraAX) default to the **LegacyDigestScheme**.

- HTTPs browser-to-station (**WebService**)

When a user opens a station from a browser, the system prompts for user name and password. The authentication mechanism used depends on the scheme selected in the **AuthenticationService**.

- Station-to-station (**FoxService**)

As for a Workbench-to-station connection, a station-to-station connection requires an assigned authentication scheme and a pre-configured user name and password. The role assigned to a station user (machine-to-machine communication) should grant only the permissions needed by the accessing station.

## User authentication checklist

Use this checklist to verify that you completed all required tasks to set up user authentication.

- Connections are secure (https rather than http; foxs rather than fox).
- Each user has been created.
- The authentication scheme has been selected for each user.
- Credentials (user name and password) have been set up for each user.
- User roles has been identified. You need to determine what each user can do with each component in the system. Objects to protect are components, files, and histories. Each of these is assigned a category.

- Roles have been created and assigned to each user. This assignment grants permission for the user to access each category of object. The user's role defines exactly what each user can do with each object in the system.
- The audit log has been set up for later analysis.

## Authentication schemes

The authentication scheme verifies that a user is authorized to access a station. A station can support more than one authentication scheme. Schemes may be added to or removed from the **AuthenticationSchemes** container in the **AuthenticationService** under the **Services** container. Additional schemes are in the **baja** and **ldap** palettes. Other schemes may be found in other palettes, and developers may create new authentication schemes. Legacy connections, such as a Niagara 4 supervisor connected to a NiagaraAX station use LegacyDigest, while N4 connections use the Digest scheme.

All authentication requests are routed through the system's **AuthenticationService**. These default authentication schemes are provided as standard components of the **AuthenticationService**:

- **DigestScheme**: With this scheme, a user password is never directly sent to the station. Instead, proof is sent that the user knows the password.
- **LegacyDigestScheme**: With this scheme, several messages are passed back and forth to prove the client knows the password. The client's password is never actually transmitted, which helps protect the system if another layer of security, such as secure TLS communication fails.

This scheme provides compatibility with stations running previous software versions. Stations running NiagaraAX must have been upgraded with the following security updates: 3.8, 3.7u1, 3.6u4, or 3.5u4.

These schemes are available in the **ldap** palette:

- **LdapScheme**
- **KerberosScheme**

The ldap schemes require the use of an LDAP server and the setting of additional properties.

Schemes may be located in other modules. Third-party schemes may also be available.

Each user account is associated with a specific scheme as set up in the **UserService Property Sheet**. This allows some user accounts to use one scheme, while other accounts use different schemes. For example, a Digest or Basic scheme is appropriate for human users, whereas a Certificate or HTTP-Basic scheme is more appropriate for devices. The system supports only schemes that have been added to the **AuthenticationService**.

**NOTE:** Deleting a scheme may leave your users with an invalid reference to a non-existent scheme.

## Station-to-station users

A station-to-station user requires a machine user as opposed to a human user.

By convention, a station-to-station user should be named something memorable (perhaps a name that is unique to your company or even to a job site).

**NOTE:** A station-to-station user should have only the permissions it requires. To improve system security, do not make a station-to-station user a super user.

As with all user, human and machine, you should carefully guard user passwords. Although frequently a station-to-station user is assigned a role with many `admin-level` Write permissions, every user, human and machine should be assigned roles that permit them (it) to access only the components required to do their job.

When adding this user, properties, such as **Facets**, **Nav File**, and **Web Profile**, which apply to browser access are inconsequential.

**NOTE:** Do not use a station-to-station user to log in as a human user to a station! Instead, you reference this user in another station, when adding a NiagaraStation device under a NiagaraNetwork.



## Adding or editing a user

Users define possible connections to the station. Under the station's `Services` container, a `UserService` provides a default `User Manager` view for you to add, delete, and edit users.

**Prerequisites:** The `User Manager` view is open.

**Step 1** Double-click the `UserService` node in the station Nav tree.

The `User Manager` view opens.

**Step 2** To create a new user, click the **New** button, otherwise, to edit an existing user select the user and click the **Edit** button.

The **New** window opens.

## Assigning authentication schemes to users

Each user is assigned their own `AuthenticationScheme`. This allows different users to use different schemes appropriate to the user type. Some schemes apply to both Fox and Web. Other schemes, such as HTTP-Basic, apply only to certain web logins (for example, Obix clients). These schemes do not work over Fox or even via the form login.

**Prerequisites:** The authentication scheme to use has been added to the `AuthenticationService`.

By default, each new station comes with the `DigestScheme` and `LegacyDigestScheme` already installed. The `DigestScheme` is assigned to all users, so that in simple cases no additional setup is required.

**Step 1** Right-click `UserService` in the Nav tree and click **Views→User Manager**.

The `User Manager` view opens.

**Step 2** Select the user and click **Edit**.

The `UserService Property Sheet` opens.

**Step 3** Scroll down to **Authentication Scheme Name** and expand the **Authenticator** section.

**Step 4** To assign an authentication scheme, select the scheme from the **Authentication Scheme Name** drop-down list.

Once these setup steps are complete, the station is ready for authentication.

## Password management

Managing passwords involves configuring the strength of the passwords to be used authentication scheme, establishing the period of time after which the password expires, setting the warning period, and setting up the password for each user.

The system supports three password features designed to strengthen access security:

- Password strength that may be configured for each authentication scheme.
- An expiration interval for a password
- Password history

### Setting up password strength

Strong passwords are recommended. Along with the other password features, password strength will frustrate any attempt to breach your system.

**Prerequisites:** Authentication scheme has been added to the `AuthenticationService`.

Password strength is associated with the selected authentication scheme, for example, `Digest` or `Baisc`, but not `LDAP`, for which password strength is managed by the `LDAP` server. You can create different strengths

for different schemes and apply those schemes to different classes of user. For example, an administrator could have stricter password strength requirements.

Once the New Station wizard completes, you can adjust the scheme's password strength properties as needed. If changed for a scheme, any future password change for any station user (including the `admin` user) requires the minimum values specified in the **Password Strength** properties.

**NOTE:** Although you may reduce password strength by entering zeros for its property values, it is strongly recommended that you retain a level of password strength similar to the default level, if not greater. For example, you may wish to require at least one special and at least two upper case characters.

You configure password strength for each authentication scheme.

- Step 1 Right-click the **AuthenticationService** in the Nav tree and click **Views→Property Sheet**.  
The **AuthenticationService Property Sheet** window appears.
- Step 2 Expand the scheme and the **Global Password Configuration→Password Strength** container for the scheme.
- Step 3 Configure the minimum character requirements, **Expiration Interval**, **Warning Period**, and **Password History Length** (5 or 10 characters).
- Step 4 Do the same for any other scheme you plan to use and click **Save**.

## Setting up password options

In most cases, users create their own passwords. You may create a temporary password for each user in the **UserService** and require them to change the password at their next login. You may also configure the password expiration date.

**Prerequisites:** The authentication scheme you need is available in the **AuthenticationService**.

- Step 1 Right-click **UserService** and click **Views→Property Sheet** in the Nav tree.
- Step 2 Open the user's **Property Sheet**.
- Step 3 Expand the user whose password you want to set.
- Step 4 Scroll down and expand the **Authenticator→Password Config** container under the user name.  
**Force Reset At Next Login** defaults to `true`.
- Step 5 To allow the user to continue using the same password, set **Force Reset At Next Login** to `false`.
- Step 6 Set the password expiration date, scroll down and click **OK**.

## Setting up a user's password

You configure user passwords through the **UserService**. If you are accessing the **UserService** from a browser, your connection must be secure (https) or you will be unable to set the password.

- Step 1 Double-click **UserServices** in the Nav tree and double-click the user record.
- Step 2 To view the password properties, expand the **Authenticator**.
- Step 3 Enter and confirm the password, then click **OK**.

## Logging on to a station

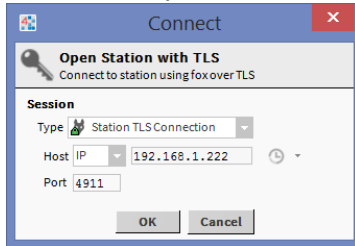
Using TLS, a secure communication session is established before the system asks for your user credentials. When you log on using the station Authentication window, the system confirms your identity, which determines your Nav tree configuration and the components you have permission to access. The system is designed to require minimum interaction while providing a secure connection and ensuring authorized access.

**Prerequisites:** An authentication scheme has been assigned to each user, and a user name and password created.

This procedure demonstrates user authentication using the default DigestScheme.

**Step 1** Open the station.

The system opens a station **Connect** window.



This window initiates the process of verifying the server.

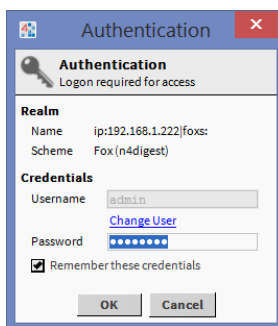
**Step 2** Enter the IP address or confirm the default address and click **OK**.

If no matching root CA certificate can be found in the client's System or User Trust Stores, the system presents a default certificate for your approval.

**Step 3** If you are presented with a certificate, make sure you recognize the certificate's **Issued By** and **Subject** properties.

**CAUTION:** Do not approve a certificate if you do not recognize these properties. The weakest link in the security chain is the user who simply clicks OK without thinking.

The system displays the station **Authentication** window.



**Step 4** If you are logging on for the first time, enter your user name.

Stations can have many authentication schemes. The first time you log on to a new station the system allows you to enter the **Username**. It uses this information to determine what authentication scheme to use. After that initial logon, you cannot change the user because another user may use a different scheme with different credential requirements. The [Change User](#) link provides a way for a different user with a different authentication scheme to log on.

**Step 5** To change to a different user, click the [Change User](#) link and enter a different name.

**Step 6** Enter your station password, select **Remember these credentials** and click **OK**.

When you select the `Remember these credentials` check box, the system saves the last user name and password you entered and defaults to them the next time you log on.

This procedure establishes a secure TLS connection to the station using the Foxs protocol over port 4911 (this is the default port).

The default logon threshold is five attempts. If you make five unsuccessful attempts to log in during a 30-second period the system locks you out for 10 seconds. You may change the logon threshold in the **UserService**.

To log off, close Workbench or the browser.

Each authentication scheme supports its own audit log, including saving date, user, and event. This information is written to the AuditHistory in the following location: **Station→History→<station name>→AuditHistory**. Permission must be assigned to this file in the **RoleService** to grant a user access to view it.

## Changing your password

Based on the password configuration, the system warns you when your password is about to expire. You can only change your password when required to do so by the system. Follow the login prompts.

## User authentication troubleshooting

Once authentication is configured and passwords assigned, very little can go wrong.

### I am attempting to set up new users using a browser, and the New button is not available.

This is a security control. Most likely the **Secure Only Password Set** is on (set to `true`), and you have made a regular connection (`http`) to the platform/station. When a secure connection is required, you must make a secure connection (`https`) to the platform/station to set the password.

### My credentials were working, but they no longer allow me to log in.

- Your password may have expired.
- You may not have permission to access this station. Check with your manager.
- The credentials cache may have become corrupted causing the saved credentials to no longer work. Clear the `Remember these credentials` check box, re-type the password or clear your computer's cache.
- There is a problem with the configuration of the authentication scheme. For example, if you are using the LDAP scheme, the port is blocked by a firewall or the wrong LDAP host name or port has been configured. Refer to the documentation for the LDAP scheme.

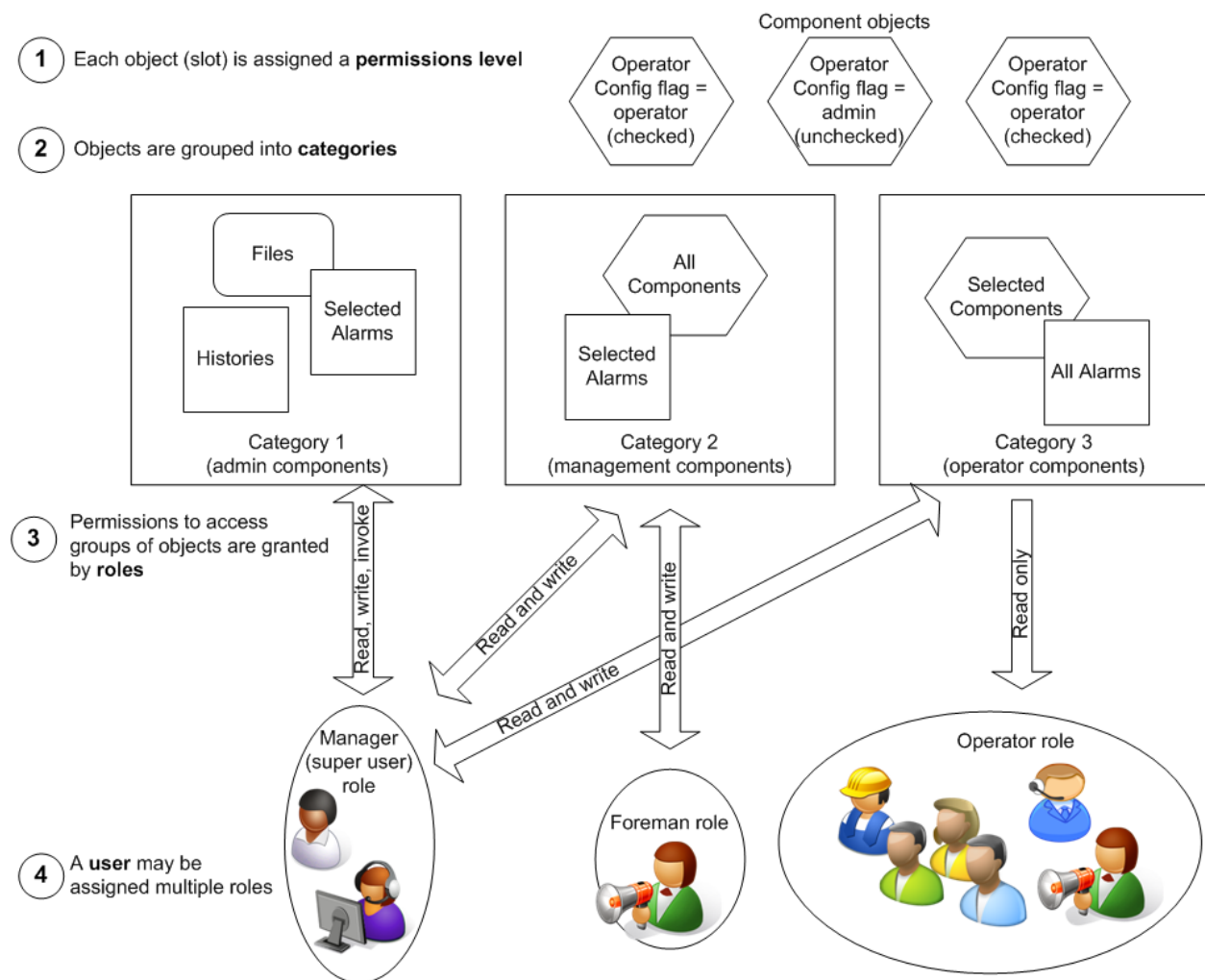
# Chapter 3 Authorization management

## Topics covered in this chapter

- ◆ Component permissions checklist
- ◆ Component permission level
- ◆ Categories
- ◆ Roles and permissions
- ◆ Component permissions troubleshooting

Once a human or remote station user is authenticated, authorization to access station components is based upon the permission level assigned to each slot, the category(ies) into which components are grouped, and the role assigned to each user. All configuration is stored in the system database, using services, components, and component views.

Figure 5 Station security configuration includes categories, roles and users



1. Beginning at the top of the diagram, the *permission level* may be configured on each component as needed. You change the default permission level for a component by turning the `Operator` config flag for the slot on or off.

2. *Categories* organize components, files and histories into groups. You may assign each component to a group or create a NEQL query (a search) that assigns components to groups based on an identifying tag already associated with the component. This is a tagged category. You set up categories using the **Category Manager** view (`CategoryService`).
3. *Roles* associate permissions to read, write, or invoke an action on a category of system components with a generic name, such as *Manager*, *Foreman* or *Maintenance crew*. You set up roles and permissions using the **Role Manager** view (`RoleService`). The **New Station** wizard installs the `Admin` role. This special super user cannot be modified or configured, and does not appear in the **Role Manager**.
4. Human and machine *users* are assigned to roles for the purpose of granting users the right to read, write and invoke actions on components. You assign roles to individual users using the **User Manager** view (`UserService`).

## Component permissions checklist

Use this checklist to verify that you completed all required tasks to set up roles and permissions.

- Categories have been set up and basic categories assigned to components.
- System components that require access control have been identified.
- The Permission level config flag has been set on component slots.
- Basic and tagged categories have been set up.
- Basic categories have been assigned to components.
- Roles have been created and permissions granted.
- Roles have been assigned to users.
- Station security has been tested.

## Component permission level

The component permission level is a config flag associated with the slot. The configuration of this flag begins the process of granting permission to access individual component slots.

- If the component slot's `Operator` config flag is cleared (unchecked), the slot is configured for the `admin permission level`. A user must be assigned a role with at least the minimum permission set in the **Role Manager** view to `admin-level Read (R)`.
- If the slot's `Operator` config flag is set (checked), the slot is configured for the `operator permission level`, and can be accessed by a user who has been assigned a role with the minimum permission set in the **Role Manager** view to `operator-level read (r)`. In other words, any user assigned this role may access the slot at least to read it.

With the `admin` permission level, users can see and change slot flags from the slot sheet of any component. By default, most slots are configured for the `admin` permission level (the out slot is typically set to the `operator` permission level).

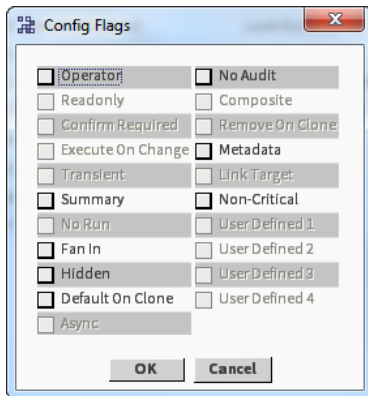
## Changing a component Config flag

Config flags set up system features at the component level.

Step 1 Display the component slot sheet.

Step 2 Right-click the slot and click **Config Flags**.

The Config Flags editor appears.



Step 3 Click in the `Operator` check box to set the config flag and click **OK**.

## UserService permission levels

By design, the `UserService` component enjoys a special permission level scheme—one that varies from the scheme described for other component access.

By default, these user properties appear as slots in the `UserService`:

- Email
- Password
- Cell Phone Number
- Facets (time format and unit conversion)

The `Operator` config flag for these slots may be enabled (checked) and disabled (unchecked) just as you would configure the permission level on any other slot. The special scheme that applies only to the `UserService` component yields the following results:

- If the `operator` permission level is enabled (`Operator` checked) on the slot, and the role assigned to the user grants read permission (`r`), the user is allowed read-only access to the user properties (email, password, etc.) on their own user account (all other users are hidden).
- If the `operator` permission level is enabled (`Operator` checked) on the slot, and the role assigned to the user grants write permissions (`rw`), the user is allowed both read and write access to the user properties on their own user account (all other users are hidden). This is the configuration required to allow a user to change their own password.
- If the `admin` permission level is enabled (`Operator` unchecked) on the slot, and the role assigned to a user grants read permission (`rR`), the user is allowed read-only access to all user properties for all available users.
- If the `admin` permission level is enabled (`Operator` unchecked) on the slot, and the role assigned to a user grants write permissions (`rwRW`), the user is allowed both read and write access to all properties for all available non-super users. Moreover, they have access to the **User Manager**, and can add new users and delete selected users. In addition, the **Permissions Browser** view of the `UserService` is available to them.

**NOTE:** A user cannot assign permissions to other users that they do not have themselves. For example, a non-super user cannot make another user a super user.

To allow each user to change their own password, but not have access to other users' passwords, you would set the config flag for the `Authenticator` slot to the `operator` permission level (checked; this is the default for this slot), and assign a role to the user that grants `operator`-level write (`rw`) permissions.

All non-super user roles should be configured for `operator`-level write (`rw`) permissions applied to the category that contains the `UserService`. (By default, the **New Station Wizard** assigns the `UserService` to the `Admin` named category (category 2), along with the `CategoryService`.)

**NOTE:** Any user granted super user permissions has access to all components, and can add more super users.

## Categories

Categories are groups to which each station component may be assigned for the purpose of managing who has permission to access the component.

The system provides three types of categories:

- **Basic (or explicit) categories** are groups (collections) to which you assign each component. Each component maintains a bitmap for basic category membership. The default eight categories consume one byte and each increment of eight categories you add consumes an additional byte (1–8, 9–16, 17–24, and so on) in the component record.
- **Inherited categories** are passed down from component parent to component child.
- **Tagged categories** do not require direct assignment. Instead they use a **Tag Query** (NEQL, Niagara Entity Query Language query) to identify the components that belong in each category based on the tags set up for the component when it was added to the system.

All components must be assigned to at least one basic category, either an explicit assignment, or an inherited assignment from a parent component. Beyond this assignment, which type of category to use depends on your needs. You may use explicit, inherited and tagged categories at the same time.

**TIP:** If you are installing a new system, tagged categories may take time to set up initially, but they will save you time when adding new components in the future. To use tagged categories, the tag must be on the component, then the category automatically groups all components that share the same tag. This saves you having to assign each new component to a category

### Basic categories

The system maintains basic categories as indexes in an array. You individually assign components in the station to each category. Subsequently, you set up roles to grant permission to access components based on the category you associated with each component. Finally, you assign a role to each user.

A new station (created using the New Station wizard) comes with two default basic categories:

- `User` (Category 1)
- `Admin` (Category 2)

As the names imply, regular users may view and modify some station objects. Only administrators should have permission to access other objects. All objects default to the `User` category except for these, which default to the `Admin` category:

- The configuration services: `UserService`, `CategoryService`, and `ProgramService`
- All files (the entire file space)

You may add basic categories as needed. For example, you could group components by equipment type, such as `Lighting`, `Door access`, and `HVAC`. An alternate scheme might group components by geography: `Floor 1`, `Floor 2`, and `Floor 3`. How you group components depends on your overall building model, and specifically on how you plan to set up roles, permissions and users.

**NOTE:** Basic categories use station memory. To improve system performance, minimize the number of categories, and keep category indexes contiguous.

### Category management

The procedures for managing categories are the similar for both basic and tagged categories.

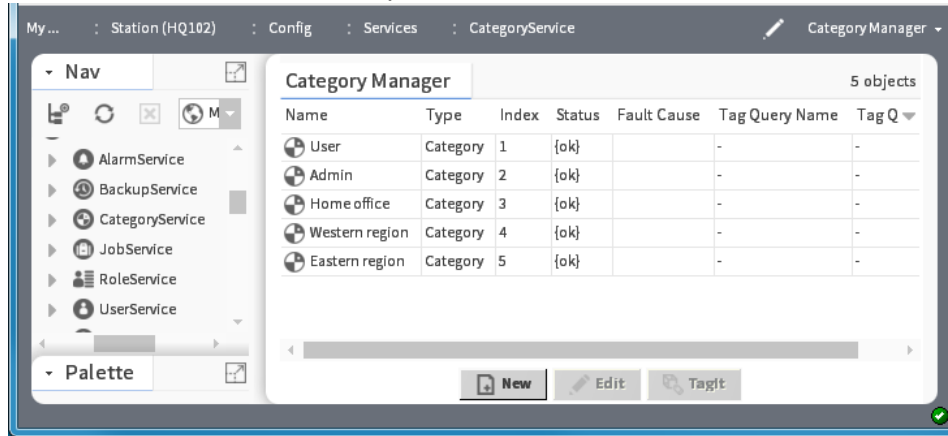


### Adding and editing a basic category

You add and edit basic categories using the **Category Manager**.

Step 1 Right-click the **CategoryService** and click **Views**→**Category Manager**.

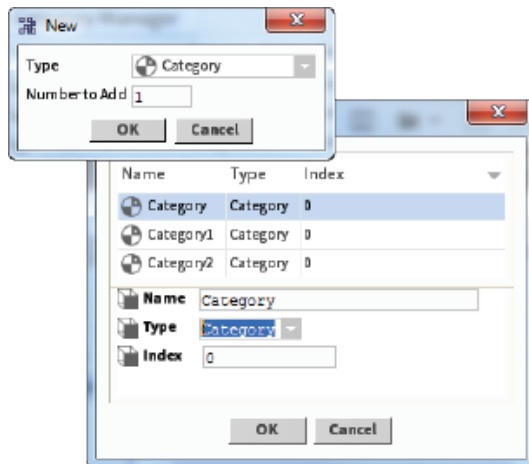
The **Category Manager** view opens.



The rows in the **Category Manager** view represent existing categories.

Step 2 Do one of the following:

- To edit the name or index of an existing category, double-click the category row in the table.
- To create a new category, click the **New** button.



Step 3 In the **New** window, choose **Category** for **Type**, select the number of categories to add, and click **OK**.

Selecting **Category** from the **Type** list allows you to assign a name to one or more basic categories.

Step 4 Enter a name and an index for each category and click **OK**.

The name(s) replace the default "Category 1," "Category 2", etc. names.

### Assigning a component to a basic category

You use the **Category Browser** to assign individual components to basic categories. Components may belong to more than one category at the same time. You cannot explicitly assign objects to tagged categories.

Step 1 Right-click Expand **CategoryService** in the Nav tree and click **Views**→**Category Browser** in the Nav tree.

The **Category Browser** view appears. Use this view to assign components to categories.

	Inherit	User	Admin	Operator	Viewer	Signal	Temp
Custom Config	n/a		•				
Services	✓		•				
Drivers	✓		•				
Apps		•	•				
category			•				
Ramp	✓		•			•	
Noise	✓		•			•	
SineWave	✓		•			•	
Threshold			•	•			
AddRamp	✓		•				
AddSine	✓		•				
GreaterThan	✓		•				
Temp1	✓		•				•
Temp2	✓		•				•
Alarm	✓		•				
AverageRoomTemp	✓		•				

By default this view shows the component types collapsed into rows in a tree structure: Alarm, Config (components), File, and History. The columns represent the categories in the station. The column titles identify the categories.

Gray columns identify tagged categories, which do not require manual component-to-category associations. This column cannot be edited in the **Category Browser**. Any component with a tag that satisfies the NEQL query is visible in the **Category Browser**.

Step 2 To view all components that have category assignments, click the binocular icon (🔍) on the toolbar.

All components appear in the table.

Step 3 Use the expandable tree to navigate to components of interest in the table. To return to the previous collapsed view, select the **Category Browser** from the drop-down list of views.

Step 4 As needed, in any component row, click either:

- In the category column to assign a component to a category or click again (toggle) to remove the component assignment from the category.
- In the Inherit column to assign a component to any of the categories its parent is assigned to.

**NOTE:** With the exception of the root components, Alarms, Config, Files, and History, each object must belong to at least one basic category or inherit its parent’s category assignments. The root objects cannot inherit. They must belong to one or more categories.

Using the **Category Browser** to assign a component to a basic category updates the component’s category bitmap. Copying the component to another station or saving it in a bog for reuse includes the category bitmap.

**Deleting a category name**

You delete a category name using the **Category Manager**.

**Prerequisites:** The category has been added. You are viewing the Category Manager.

**Step 1** To delete a category name, click the row in the **Category Manager** view and press **Delete** or right-click the row and click **Delete**.

Deleting a category name changes the name back to the generic index name of Category 1, 2, etc. It does not remove the category index from the object(s) with which it is associated.

## Roles and permissions

Once a user has been authenticated, the user is granted or denied the right to access each protected object in the system using a permissions map (a category-based matrix), which is defined by the role assigned to the user. Permissions define the rights a user has within each category of station objects.

Roles ease the management of permissions for a large number of users. The permissions for a group of users who are assigned to the same role can be updated by changing the role. This saves having to update each user's permissions individually.

For example, if 40 operators need access to a new component in the station, you may need to update only their shared operator role, and then only if a category has been added or permissions need to be changed. The initial configuration of a station's security, which involves object permission levels, object categories, roles (permissions) and users may take time to design and set up in some configurations, but the trade off is worth the future time saved when updating the permissions of more than one user at a time.

## Adding roles and permissions

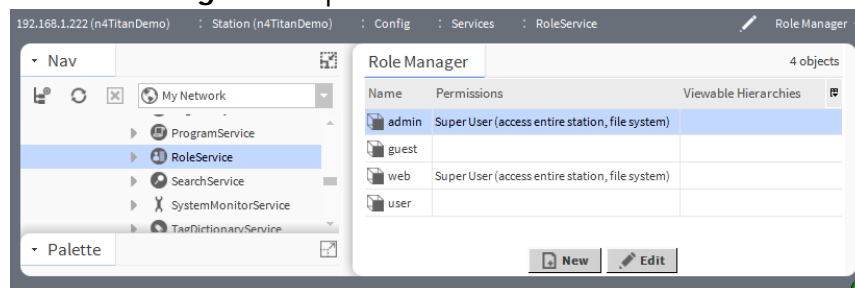
You add roles using the station's **Role Manager** view (**RoleService**).

**Prerequisites:** `Operator` config flag enabled for any restricted components. Categories created and any basic categories assigned to components.

Most companies require as a minimum, an administrator (super user) role, a manager role, and a regular user or operator role.

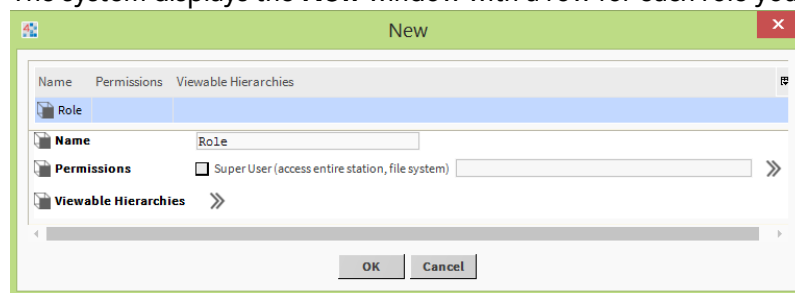
**Step 1** Right-click **RoleService** in the Nav tree, click **Views**→**Roll MANager**.

The **Role Manager** view opens.



**Step 2** Click the **New** button, enter the number of roles to create in the pop-up window and click **OK**.

The system displays the **New** window with a row for each role you are creating.



Step 3 Name each role.

Step 4 To configure a role as a super user, click the `Permissions` check box for `Super User`.

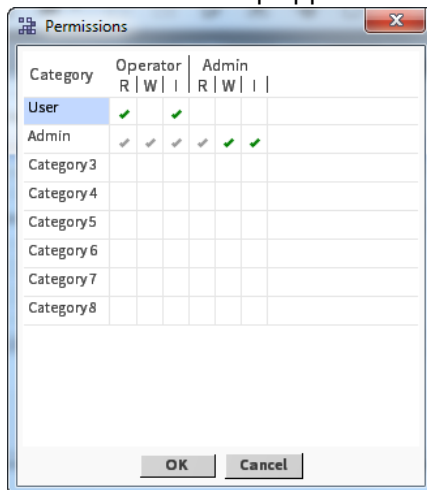
The built-in `Admin` role grants all possible rights for every category (super user). Only when logged in as the `Admin` user, or another super user, can you assign super user rights using the `Super User` check box.

In general, assigning super user rights should be strictly limited and based on special needs. For example, a Supervisor station may need super user rights to connect with other station clients (machine login vs. login by a person) *in scenarios where Program objects are exported from stations using ExportTags*. Human users may need super user rights to add and edit Program or Robot components.

**CAUTION:** Do not make it a common practice to give station-to-station users admin privileges. If your network is breached, a station-to-station user could occasion significant damage without drawing attention to what is happening.

Step 5 To set up individual permissions, click the chevron at the end of the `Permissions` property.

The `Permissions` map appears.



The first column, `Category`, lists the groups to which you may grant permission. The `Operator` and `Admin` columns relate to the permissions level configured on each component. Below these headings are the cells to use for assigning one of three permissions to each category:

- R = Read allows the user to view the object.
- W = Write allows the user to change the object.
- I = Invoke allows the user to initiate an action related to the object.

Depending on how the permission level is set on the slot, six permissions are derived:

- To allow a user to view `operator`-level information, check the `Operator` config flag on the slot and select the `Operator R` column on the permission map.
- To allow a user to modify `operator`-level information (if it is not read-only), check the `Operator` config flag on the slot and select the `Operator W` column on the permission map.
- To allow the user to view and invoke `operator`-level operations (actions), check the `Operator` config flag on the slot and select the `Operator I` column on the permission map
- To allow the user to view `admin`-level information, leave the `Operator` config flag unchecked on the slot and select the `Admin R` column on the permission map.
- To allow the user to modify `admin`-level information (if it is not read-only), leave the `Operator` config flag unchecked on the slot and select the `Admin W` column on the permission map.

- To allow the user to view and invoke `admin`-level operations (actions), leave the `Operator` config flag unchecked on the slot and select the `Admin I` column on the permission map.

When you assign permissions, higher-level permissions (green check marks) automatically include the lower-level ones (gray check marks). For example, if you enable `admin`-level write (W), the system automatically enables `admin`-level read (R), as well as `operator`-level read and write (rw).

**Step 6** Click the cell to assign a permission and click **OK**.

The permissions appear in the **Permissions** property.

**Step 7** To finalize permissions, click **OK**.

## Adding a component

Adding a component to your building model involves dragging the component from a palette, possibly setting the **Config Flag** on the component slot, and configuring the component to assign it to a category. A component may be a new network, device or service.

### Prerequisites:

- If required, you have a license to add the component to your model.
- Any categories, roles (permissions) to assign to the component have been set up.
- The users who will access the component exist in the system.

**Step 1** Open the palette that contains the component module.

**Step 2** Expand the Nav tree to view the **Services** or **Drivers** container.

**Step 3** Do one of the following:

- Drag the component from the palette to the **Property Sheet** or **Driver Manager**.
- Drag the component to the appropriate **Services** or **Driver** container in the Nav tree

The **Name** window opens.

**Step 4** Change the name of the component, or use the default name and click **OK**.

**Step 5** If you need to configure the permission level for the new component slot, right-click the component in the Nav tree and click **Slot Sheet**, then right-click the slot and click **Config Flags**.

- Leave the `Operator` config flag unchecked for the `admin` permission level (this level allows read and write access).
- Toggle this flag (checked) for the `operator` permission level (this level restricts user access to a minimum of read-only permission).

**Step 6** To assign the component to a category, do one of the following:

- If a tagged category exists, set up a tag on the component that will satisfy the tagged category's NEQL query.
- If you are using basic categories, add the component to the category using the **Category Browser** or directly to the component using the component's **Category Sheet**.

**Step 7** If you created a new basic category, update the role assigned to users of this component to include permissions for the new component, otherwise confirm that the role includes the category.

**Step 8** Confirm that component **Status** is `{ok}`.

You are ready to configure the component.

## Editing roles and permissions

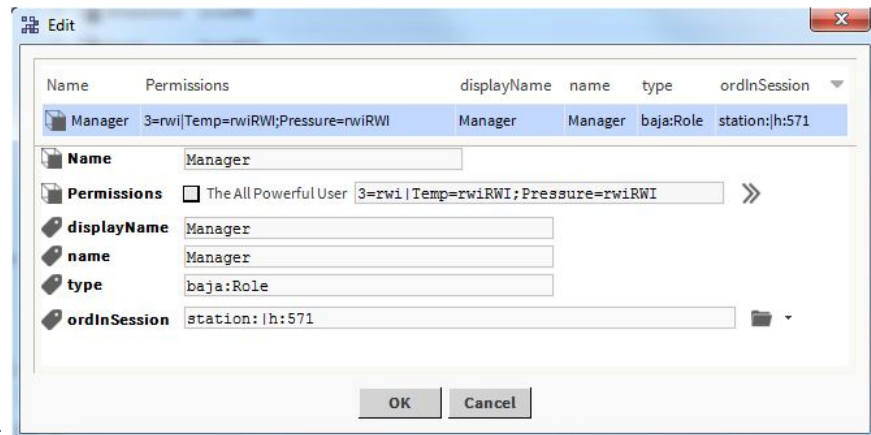
The primary reason for editing a role is to update permissions.

**Prerequisites:** The permission level is set appropriately on all components. Categories have been created. Roles have been created.

Step 1 Right-click Expand **RoleService** in the Nav tree and click **Views→Roll Manager** .

The **RoLe Manager** view opens.

Step 2 Double-click the row for the role to edit or select the row and click the **Edit** button.



The **Edit** window opens.

Step 3 To change permissions, click the chevron to the right of the **Permissions** property.

The **Permissions** map appears.

Step 4 Update the permissions as needed and click **OK**.

Step 5 To finalize the changes, click **OK**.

## Assigning roles to users

This task associates the permissions defined by a specific role with each user.

**Prerequisites:** Roles and users have already been created.

Step 1 Right-click **UserService** in the Nav tree, click **Views→User Manager** .

Step 2 Double-click the user's row in the **User Manager**.

The user's **Property Sheet** appears.

Step 3 For the **Roles** property, select one or more role names by clicking in the check box and click **OK**.

## Confirming access security

You should test each user's access rights before allowing users to use the system.

Step 1 Log out of the station.

Step 2 Log back in as a user that represents each role.

Step 3 Confirm that the Nav tree shows only those components to which the user has read, write or invoke action rights.

## Reviewing permissions

The **Permissions Browser** displays the rights granted to each role.

**Prerequisites:** Roles exist with permissions granted.

Step 1 Log in to the station as a super user or as the **Admin** user.

Step 2 Right-click **RoleService** in the Nav tree and click **Views→Permissions Browser** .

Step 3 Expand the Nav tree to view role permissions.

**NOTE:** Although you may double-click any object row in the **Permissions Browser**, view the permissions map and update permissions for an individual user, this method of updating permissions does not change the permissions as configured in the user's role.

## Ancestor permissions

Often, you may wish to grant a user the right to access components using categories that are not included in the component's parent, such that, permissions to a component's ancestor tree are not explicitly granted. In this case, the system automatically grants `operator` permission level read-only access to all ancestor components in the component tree. Otherwise, a user would be unable to navigate to target component in the Nav tree.

This automatic ancestor permission level assignment is done by the station periodically, but you can force it at any time with a right-click **Update** action on the **CategoryService** node in the Nav tree.

## File permissions

By default, the New Station Wizard assigns the entire station's File space to category 2 (Admin). A station's `config.bog` file and `config.bog.backup` files are not accessible (even by super users) in the station's file space. If needed, other station files and folders may be hidden from a remote station.

Users typically require that the role assigned to them have `operator`-level read permission on station file folders, such as `^nav`, `^px`, `^images`, `^html`, and so on. However, permissions higher than an `operator`-level read on the `Admin` category should only be assigned to selected users on an as-needed basis. In most situations, creating a new category containing *only* the components a user needs to access is a more appropriate solution.

Largely, rights granted to access categories that are used by files and folders are `operator`-level permissions as follows:

- Files require `operator`-level read (`r`) access to view, and `operator`-level write (`rw`) access to edit a file (if applicable). For instance, a user with `operator`-level write and write (`rw`) access to an `.html` file can modify it using the Text File Editor in Workbench.
- Folders (directories) require `operator`-level read (`r`) access to list and to copy child files, and `operator`-level write (`rw`) access to create new (or delete existing) child files.

A few views of files require `admin`-level Write (`rwRW`) permissions to access, such as the **Nav File Editor** for a Nav file. There are also these special case file permissions:

- The system automatically restricts any system module files to `operator`-level read (`r`) access.
- If a user is not a super user, the system denies all access outside of the station's home directory.
- Users need `admin`-level Read (`rwRW`) access to see a Supervisor station's `^provisioningNiagara` folder (written to by the Supervisor's provisioning mechanism).
- If you have a developer license, your system includes an additional category called `NModuleDevFilePermission`. This category grants `rwRW` permission to access all system modules.

## History permissions

Histories require that the `operator` permission level be set and `operator`-level read (`r`) permission granted by the role to access all available views.

History views include History Chart, Collection Table, and History Table). This includes the ability to rename a history.

## Component permissions troubleshooting

To begin troubleshooting component permissions, go back and review your original design and double-check all configuration properties.

**I can successfully log in to a station, but I get the message, "User username does not have access to the station. Check permissions."**

The user name you used to log in with is an authentic user name, but either no role has been associated with the user or the permissions contained in the role configuration do not allow the user to access the station. Log in with a user that has permission to access the station and update the configuration. A role needs permissions on at least one component for a user to be able to access the station.

**I set up categories and roles, but my users can still access some components they should not be able to access and cannot access others that they should be able to access.**

Confirm that you configured the component permission level correctly for each component.

Open the **Category Browser** and review the categories the user has permission to access. Verify that there are no unexpected components in the categories.

**My users cannot change their own passwords.**

You need to assign a role to each user that grants write access (rw) to the **Password** slot on the **UserService**.



# Chapter 4 Reference

## Topics covered in this chapter

- ◆ Components
- ◆ Windows
- ◆ Plugins (views)

The topics that follow provide detailed documentation for each component and plugin that supports this system feature.

## Components

Components include services, folders and other model building blocks. They may be dragged and dropped onto a property or wire sheet from a palette.

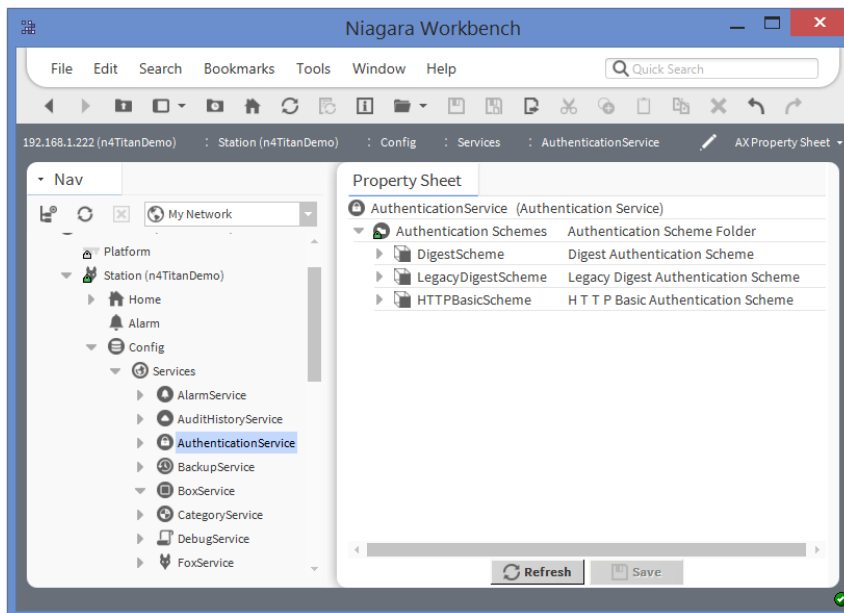
The descriptions included in the following topics appear as headings in documentation. They also appear as context-sensitive help topics when accessed by:

- Right-clicking on the component and selecting **Views→Guide Help**
- Clicking **Help→Guide On Target**.

## AuthenticationService

This component manages how users verify their identity to the station, using authentication schemes. Some schemes require password configuration, others do not. The **AuthenticationService** node is located in the **Services** container.

Figure 6 Authentication Schemes Property Sheet



The **New Station** wizard installs two default authentication schemes:

- **DigestScheme** provides SCRAM-SHA256 (Salted Challenge Response Authentication Mechanism) technology for connecting Niagara 4 entities. Several messages are passed back and forth to prove the client knows the password.
- **LegacyDigestScheme** provides compatibility with stations running a previous software version.

Schemes available in the ldap palette include:

- **LdapScheme**
- **KerberosScheme**

Additional schemes may be added in the future. Some may reside in other palettes. Developers may also create authentication schemes for special circumstances. You pick the one or two schemes you wish to use, drag them from the palette and drop them directly under the **AuthenticationService** in the Nav tree.

### DigestScheme component

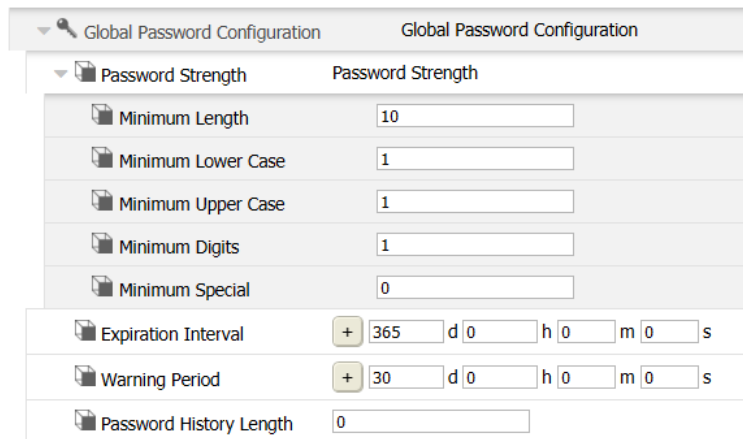
This is an authentication scheme that uses SCRAM-SHA256 (Salted Challenge Response Authentication mechanism). One of the default schemes, this component is located in the **ba ja** palette.

When using the DigestScheme, the password is never sent across the wire. Instead, the client sends proof that they know the password.

### Global Password Configuration

These properties configure password requirements for a particular authentication scheme. You access them by expanding **Station→Config→Services→Authentication→Authentication Schemes** and double-clicking one of the schemes.

Figure 7 Global Password Configuration properties



### Scheme properties

Property	Value	Description
Password Strength	several sub-properties	See <a href="#">Password strength properties, page 59</a>
Expiration Interval	number of days, hours, minutes and seconds	Defines the length of time from when the password is created until it is no longer valid. When this period of time expires, the system denies access.

Property	Value	Description
Warning Period	number of days, hours, minutes and seconds	Defines how many days of warning a user receives prior to the expiration of the password.
Password History Length	number	Defines how many previously used passwords the system remembers.

### Password strength properties

Property	Value	Description
Minimum Length	number	Indicates the total number of characters required.
Minimum Lower Case	number	Indicates the minimum number of lower case letters required.
Minimum Upper Case	number	Indicates minimum number of upper case letters required.
Minimum Digits	number	Indicates the minimum number of digits (1, 2, 3 etc.)
Minimum Special	number	Indicates the number of special characters required. For example: ! @ # \$ % ^ , . ; etc.

### *HTTPBasicScheme*

This authentication scheme performs HTTP-Basic authentication using standard HTTP headers. It only works via the web, and is intended for clients that cannot use cookies. In this authentication scheme, the user name and password are sent over the connection. This component is located in the **ba ja** palette.

### *LegacyDigestScheme*

This default authentication scheme provides backward compatibility with stations running a previous software version. This component is located in the **ba ja** palette.

This authentication scheme provides compatibility with these : NiagaraAX versions:

- 3.5u4
- 3.6u4 and up
- 3.7u1 and up
- any 3.8 version

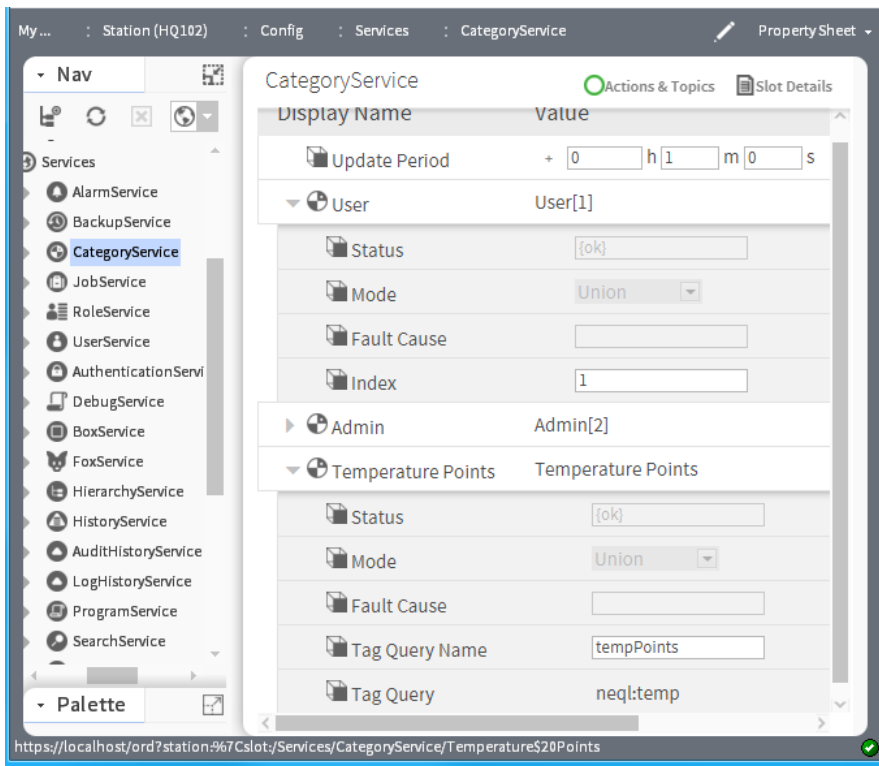
Earlier version of NiagaraAX do not support the LegacyDigestScheme.

### **CategoryService**

The **CategoryService** allows you to add and edit component categories. It is located in a station's **Services** container.

## Primary property

Figure 8 CategoryService property sheet



In addition to being the container for child categories, the **CategoryService** has only one slot: **Update Period**.

Property	Value	Description
Update Period	hours minutes seconds	Sets the interval at which the system automatically assigns ancestor permissions. The default value is one (1) minute. If you assign a zero value, the system disables this feature.

### User, Admin and additional basic category properties

Property	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. <ul style="list-style-type: none"> <li>{ok} indicates that the component is polling successfully.</li> <li>{down} indicates that polling is unsuccessful, perhaps because of an incorrect property.</li> <li>{disabled} indicates that the <b>Enable</b> property is set to false.</li> <li>fault indicates another problem.</li> </ul>
Mode		

Property	Value	Description
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Index	integer	Sequential number that identifies the property in the station.

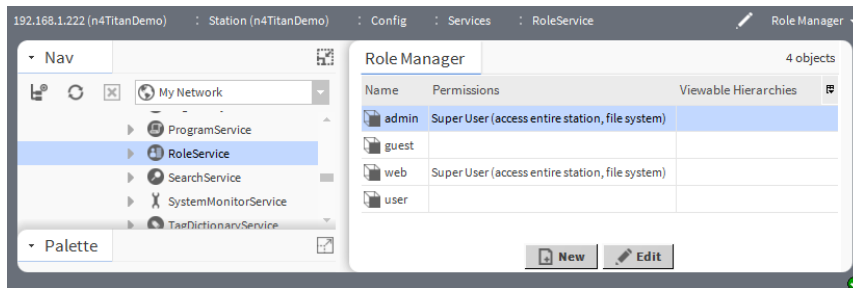
**Tagged category properties**

Property	Value	Description
Tag Query Name	text	A descriptive name to represent the results of the search.
Tag Query	NEQL	A NEQL query. This property is required when Type is Tagged Category.

**RoleService**

The **Role Manager**, which is the default view for this service, allows you to create, edit and delete roles. It is located in the station's **Services** container.

Figure 9 Role Manager view

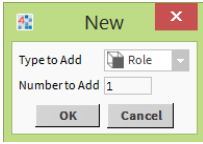


The system creates the `admin` role by default and grants it super user permissions. The `admin` role does not appear in the **Role Manager** view and you may not delete it.

Column or field	Value	Description
Name	text	Identifies the role to be assigned to one or more users. Role names are case sensitive.
Permissions	text	Associates a name with a specific set of permissions.
Viewable Hierarchies	text	Identifies the hierarchies this user may view.
Type	Role (default)	Identifies the type of entity being created.
Number to add	number	Allows you to create many rows at once in the <b>Role Manager</b> view's table.

### New role window

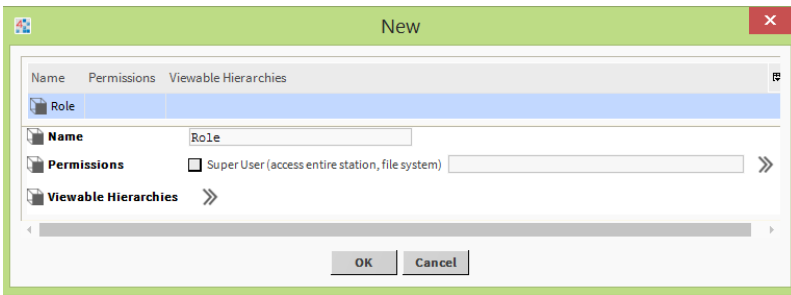
Figure 10 New Role window



Column or field	Value	Description
Type	Role (default)	Identifies the type of entity being created.
Number to add	number	Allows you to create many rows at once in the <b>Role Manager</b> view's table.

### New role properties

Figure 11 New role properties



Property	Value	Description
Name		
Permissions		
Viewable Hierarchies		

### UserService property sheet

This service manages all system users: human and machine. It is located in the station's **Services** container.

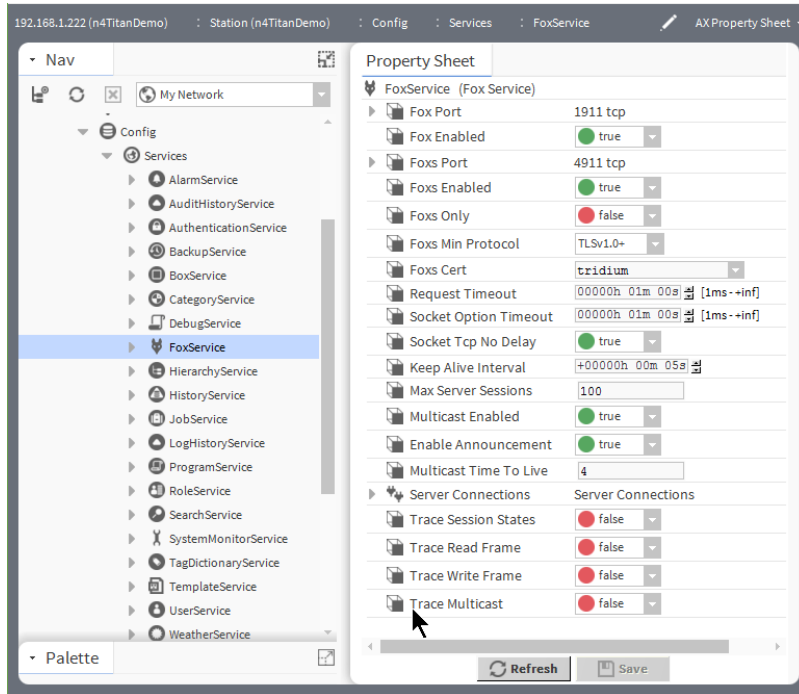
#### Property Sheet

Property	Value	Description
Lock Out Enabled		
Lock Out Period		
Max Bad Logins Before Lock Out		
Lock Out Window		
Secure Only Password Set		
User Prototypes	multiple properties	See

## FoxService

To view these properties expand the **Services** folder in the Nav tree, right-click **FoxService**→**Views**→**Property Sheet**.

Figure 12 Example of FoxService properties



Type	Value	Description
Fox Port	1911 (default) Tcp (default)	Public Server Port specifies the port number for standard Fox communication.  Ip Protocol specifies the protocol.
Fox Enabled	true (default) or false	When set to true, turns on a standard Fox connection (no communication encryption) using port 1911. When enabled, <b>Http Enabled</b> in the <b>WebService</b> must also be set to true (for wbaapplet use).  When set to false, turns off the standard Fox connection causing the system to ignore attempts to connect using Fox port 1911. If <b>Foxs Only</b> is enabled, this setting (false for <b>Fox Enabled</b> ) is irrelevant.
Foxs Port	4911 (default) Tcp (default)	Public Server Port specifies the port number for standard Fox communication.  Ip Protocol specifies the protocol.
Foxs Enabled	true (default) or false	When set to true, turns on secure Fox communication using port 4911. When enabled, <b>https Enabled</b> in the <b>WebService</b> must also be set to true (for webaapplet use).  When set to false, turns off the secure Fox connection causing the system to ignore attempts to connect using Foxs port 4911.

Type	Value	Description
Foxs Only	true or false (default)	When set to true redirects any attempt to connect using a connection that is not secure (Fox alone) to Foxs.  When set to false, does not redirect attempts to connect using Fox alone.
Foxs Min Protocol	drop-down list TLsv1.0+ (default) TLsv1.1+ TLsv1.2	The minimum level of the TLS (Transport Layer Security) protocol to which the server will accept negotiation. The default includes versions 1.0, 1.1 and 1.2. It works with most clients, providing greater flexibility than an individual version.  During the handshake, the server and client agree on which protocol to use.  Change <b>Protocol</b> from the default if your network requires a specific version or if a future vulnerability is found in one of the versions.
Foxs Cert	drop-down list of server certificates; defaults to tridium	Specifies the alias of the host platform's server certificate, which the client uses to validate server authenticity. The default identifies a self-signed certificate that is automatically created when you initially log in to the server. If other certificates are in the host platform's key store, you can select them from the drop-down list.
Request Timeout	hours, minutes, seconds (default: 1 minute)	Specifies the time to wait for a response before assuming a connection is dead.
Socket Option Timeout	hours, minutes, seconds (default: 1 minute)	Specifies the time to wait on a socket read before assuming the connection is dead.
Socket Tcp No Delay	true (default) or false	Used to disable Nagle's algorithm, which may cause issues with delayed acknowledgements that occur in TCP socket communications between Fox clients and servers. The default disables Nagle's algorithm.  In Workbench, you can enter this line in the <code>system.properties</code> file to adjust this setting: <code>niagara.fox.tcpNoDelay=true</code> .
Keep Alive Interval	hours, minutes, seconds (default: 5 seconds)	Sets the amount of time between messages, which indicate that the connection is alive. This value should be well below the request timeout and socket option timeout.
Max Server Sessions	number (default: 100)	Sets the maximum number of Fox/Foxs server connections before additional client connections error with busy.
Multicast Enabled	true (default) or false	Allows the station to initiate UDP (User Datagram Protocol) multicasting. This feature is necessary for a discovery from this station. This type of multicasting differs from Workbench UDP multicast support, which can be optionally disabled via an entry in the Workbench host's <code>system.properties</code> file.
Enable Announcement	true (default) or false	Turns on support for UDP multicast announcement messages received by the station in support of learn/discover.
Multicast Time to Live	number (default: 4)	Specifies the number of hops to make before a multicast message expires.

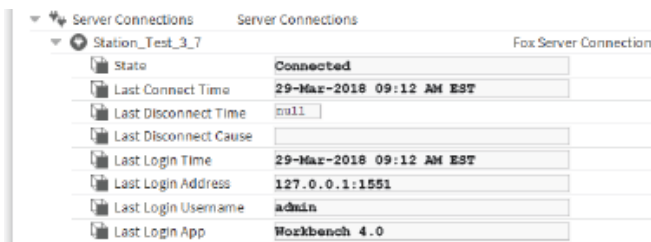


Type	Value	Description
Server Connections		See <a href="#">Server Connections, page 65</a>
Trace Session States	true or false (default)	Debug usage for tracing session state changes.
Trace Read Frame	true or false (default)	Debug usage for dumping frames being read from the wire.
Trace Write Frame	true or false (default)	Debug usage for dumping frames being written to the wire.
Trace Multicast	true or false (default)	Debug usage for tracing multicast messaging.

### Server Connections

These properties provide status information about current Workbench client connections to the local station. They do not reflect station-to-station Fox connections. Each connection provides the same set of properties.

Figure 13 Example of Server Connections properties

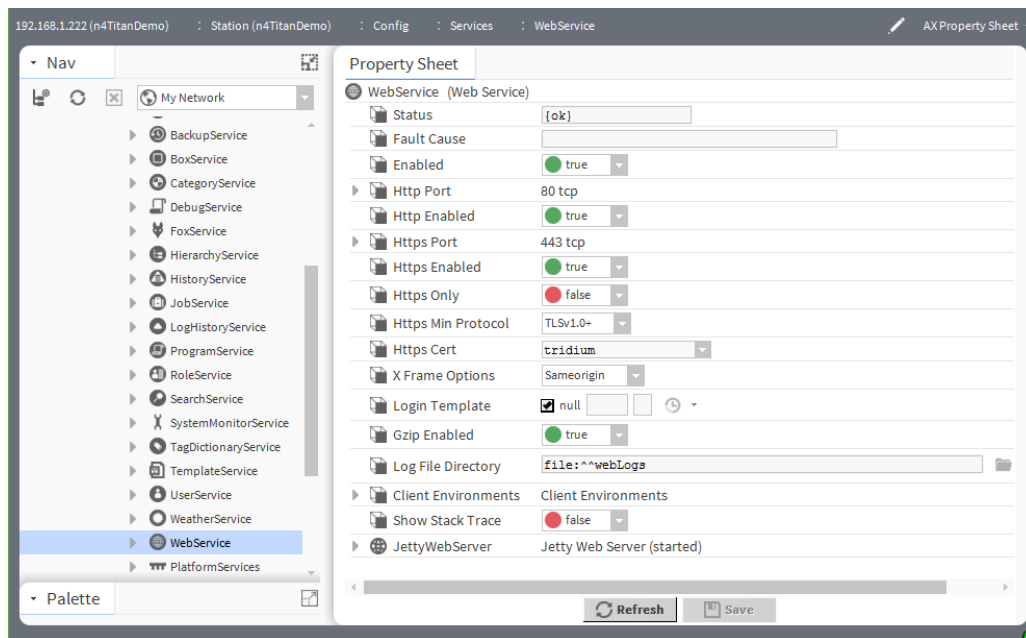


Property	Value	Description
State	Not connected, Connected	Reports the current status of the connection.
Last Connect Time	hours, minutes, seconds	Reports the last time the client successfully connected to the server.
Last Disconnect Time	hours, minutes, seconds	Reports when the client last disconnected from the server.
Last Disconnect Cause	text	Provides a brief explanation.
Last Login Time	hours, minutes, seconds	Reports the last time a client logged in to the server.
Last Login Address	IP address	Identifies the platform.
Last Login Username	text	Identifies the user who made the connection.
Last Login App	text	Identifies the version of Workbench.

### WebService

To view these properties, expand the **Services** container in the Nav tree, and right-click **WebService**→**Views**→**Property Sheet**.

Figure 14 Example of Webservice properties



Name	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. <ul style="list-style-type: none"> <li>• {ok} indicates that the component is polling successfully.</li> <li>• {down} indicates that polling is unsuccessful, perhaps because of an incorrect property.</li> <li>• {disabled} indicates that the <b>Enabled</b> property is set to false.</li> <li>• fault indicates another problem.</li> </ul>
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Enabled	true or false	Activates and deactivates use of the function.
Http Port	80 (default)	Specifies the port number for standard Http communication.
Http Enabled	true (default) or false	When set to true, turns on a standard Http connection (no communication security) using port 80. When enabled, <b>Fox Enabled</b> in the <b>FoxService</b> must also be set to true (for wbapplet use).  When set to false, turns off the standard Http connection causing the system to ignore any attempts to connect using Http port 80. If <b>Https Only</b> is enabled, this setting (false for <b>Http Enabled</b> ) is irrelevant.
Https Port	443 (default)	Specifies the port number for secure Http communication
Https Enabled	true (default) or false	When set to true, turns on secure Http communication using port 443. When enabled, <b>Foxs Enabled</b> in the <b>FoxService</b> must also be set to true (for webapplet use).

Name	Value	Description
		When set to <code>false</code> , turns off the secure Https connection causing the system to ignore any attempts to connect using Https port 443.
Https only	<code>true</code> (default) or <code>false</code>	When set to <code>true</code> redirects any attempt to connect using a connection that is not secure (Http alone) to Https. When set to <code>false</code> , does not redirect attempts to connect using Http alone.
Https Min Protocol	drop-down list TLSv1.0+ (default) TLSv1.1+ TLSv1.2	The minimum level of the TLS (Transport Layer Security) protocol to which the server will accept negotiation. The default includes versions 1.0, 1.1 and 1.2. It works with most clients, providing greater flexibility than an individual version. During the handshake, the server and client agree on which protocol to use. Change <b>Protocol</b> from the default if your network requires a specific version or if a future vulnerability is found in one of the versions.
Https Cert	drop-down list of server certificates; defaults to <code>tridium</code>	Specifies the alias of the host platform's server certificate, which the client uses to validate server authenticity. The default identifies a self-signed certificate that is automatically created when you initially log in to the server. If other certificates are in the host platform's key store, you can select them from the drop-down list.
X Frame Options		
Login Template		
Gzip Enabled	<code>true</code> (default) or <code>false</code>	
Log File directory		
Client Environments		See <a href="#">Client Environments—Mobile, page 67</a>
Show Stack Trace	<code>true</code> or <code>false</code> (default)	
JettyWebServer		

### Client Environments—Mobile

The Client Environments container slot allows the station to automatically detect the user agent of an incoming client and use the appropriate Web Profile for the user:

- Default Web Profile if using a Java-enabled device, such as a PC
- Mobile Web Profile if using a mobile device, such as a cell phone or tablet

Name	Value	Description
Enabled	<code>true</code> or <code>false</code>	Activates and deactivates use of the function.
Status [component]	<code>text</code>	Read-only field. Indicates the condition of the component at last polling.

Name	Value	Description
		<ul style="list-style-type: none"> <li>{ok} indicates that the component is polling successfully.</li> <li>{down} indicates that polling is unsuccessful, perhaps because of an incorrect property.</li> <li>{disabled} indicates that the <b>Enable</b> property is set to false.</li> <li>fault indicates another problem.</li> </ul>
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
User Agent Pattern	text separated by the pipe symbol ( )	A list of one or more user agents separated by the pipe symbol that identify the target display types.

## JettyWebServer

Name	Value	Description
Server State		
Min Threads	3 (default)	
Max Threads	30 (default)	
N C S A Log	NCSA Request Log	See <a href="#">N C S A Log, page 68</a>

## N C S A Log

This is a common format of a standardized text file that web servers use to keep track of processed requests.

Name	Value	Description
Enabled	true or false	Activates and deactivates use of the function.
Retain Days	7 (default)	Limits the size of the log by defining how many days to save log information.
Extended Format	true (default) or false	
Log Cookies	true or false (default)	
Log Time Zone	list of time zones	Identifies the time zone to use for time stamps.

## Windows

### Certificate Signing window

This window selects the root or intermediate certificate to be used for signing a client certificate. You access this window by clicking **Tools**→**Certificate Signer Tool**.

Figure 15 Example of a Certificate Signing dialog box



Name	Value	Description
Select a certificate signing request to sign:	text	This is the CSR you created.
Not Before	date	Specifies the date before which the certificate is not valid. This date on a server certificate should not exceed the <b>Not Before</b> 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 <b>Not After</b> date on the root CA certificate used to sign it.
CA Alias	text	This short name specifies the CA certificate whose private key will be used to sign this certificate.
CA Password		This is the password that protects the private key of the CA certificate being used to sign this certificate.

### Edit history export window

The Edit window shows the configuration properties of the history export descriptor, plus **Name**, which is equivalent to the right-click **Rename** command on the descriptor. To access all properties, including all status properties, view the AuditHistoryService property sheet.

### NiagaraHistoryExport

The **NiagaraHistoryExport** line above the properties summarizes the properties.

[Screen capture here](#)

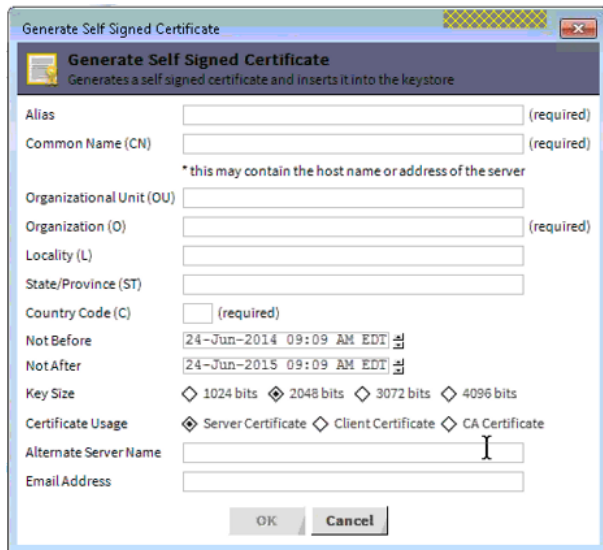
Property	Value	Description
Name	Text string followed by numbers	For a history originating in the local host station, the name begins with <code>Local_</code> . If Discovered for import, typically left at default. For a system history export, originating in the remote station, the name begins with <code>NiagaraSystemHistoryExport</code> .
Execution Time — Manual	N/A	Requires human intervention to initiate a history export or import.
HistoryId	Text in two parts: <code>/stationname/historyname</code>	Specifies the history name in the local station's History space, using two parts: <code>"/&lt;stationName&gt;"</code> and <code>"/&lt;history-Name&gt;"</code> . If Discovered, station name is <code>"^"</code> (a character

Property	Value	Description
		representing the device name of the parent container) and history name reflects the source history name. Typically, you leave both fields at default values, or edit the second (<history-Name>) field only.
Execution Time — Daily (default)	Time Of Day hours:minutes:seconds AM/PM timezone Randomization Days Of Week	Defines when the daily export or import automatically takes place. The hours follow a 24-hour clock.
Execution Time — Interval	Interval hours:minutes:seconds Time Of Day Days Of Week	Defines the amount of time between automatic exports or imports. Hours may number in the thousands.
Enabled	true or false	Activates and deactivates use of the function.

### Generate Self-Signed Certificate window

This window defines the important information required to create a certificate. It appears when you click **New** at the bottom of the **User Key Store** tab.

Figure 16 Default view of the Generate Self-Signed Certificate dialog box



You use this window to create your own certificates along with a key pair (public and private).

There is a limit of 64 characters for each of the following properties. Although blank fields are permitted, it is recommended to correctly fill in all fields, as not doing so may generate errors, or cause third-party CAs to reject your certificate. Spaces and periods are allowed. Enter full legal names.

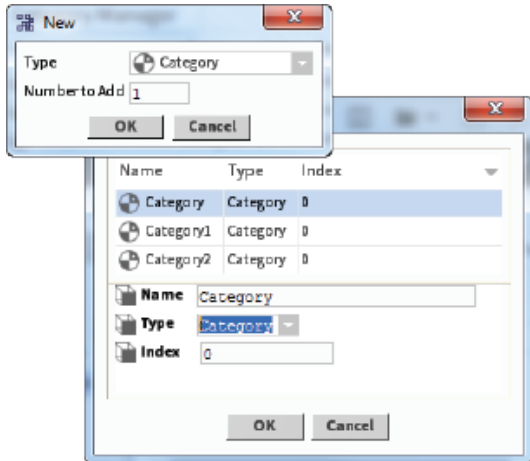
Name	Value	Description
Alias	text	A short name used to distinguish certificates from one another in the <b>Key Store</b> . This property is required. It may identify the type of certificate (root, intermediate, server), location or function. This name does not have to match when comparing the server certificate with the CA certificate in the client's Trust Store.
Common Name (CN)	text, required, alphanumeric; do not use "*" or "?" as part of the name	Also known as the Distinguished Name, this field should be the host name. It appears as the <code>Subject</code> in the <b>User Key Store</b> .
Organizational Unit (OU)	text	The name of a department within the organization or a Doing-Business-As (DBA entry). Frequently, this entry is listed as "IT", "Web Security," "Secure Services Department" or left blank.
Organization (O)	text	The legally registered name of your company or organization. Do not abbreviate this name. This property is required.
Locality (L)	text	The city in which the organization for which you are creating the certificate is located. This is required only for organizations registered at the local level. If you use it, do not abbreviate.
State/Province (ST)	text	The complete name of the state or province in which your organization is located. This property is optional.
Country Code (C)	two-character ISO-format country code.	If you do not know your country's two-character code, check <a href="http://www.countrycode.org">www.countrycode.org</a> . This property is required.
Not Before	date	Specifies the date before which the certificate is not valid. This date on a server certificate should not exceed the <b>Not Before</b> 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 <b>Not After</b> date on the root CA certificate used to sign it.
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.
Certificate Usage:	text	Specifies the purpose of the certificate: server, client or CA certificate. Other certificate management software utilities may allow other usages.
Alternative Server Name	text	This property provides a name other than the <code>Subject</code> (Common Name) that the system can use to connect to the server. Like the <code>Common Name</code> , the system uses the <code>Alternative Server Name</code> to validate the server certificate making it possible to specify both an IP (Internet Protocol) and FQDN (Fully Qualified Domain Name).
Email Address	email address	The contact address for this certificate. It may also be the address to which your signed certificate (.pem file) will be sent.

## New category windows

Each category type (basic and tagged) has a version of the **New** category window. One of these windows appears when you click the **New** button at the bottom of the **Category Manager** view and select the type of category to create.

### Basic category window

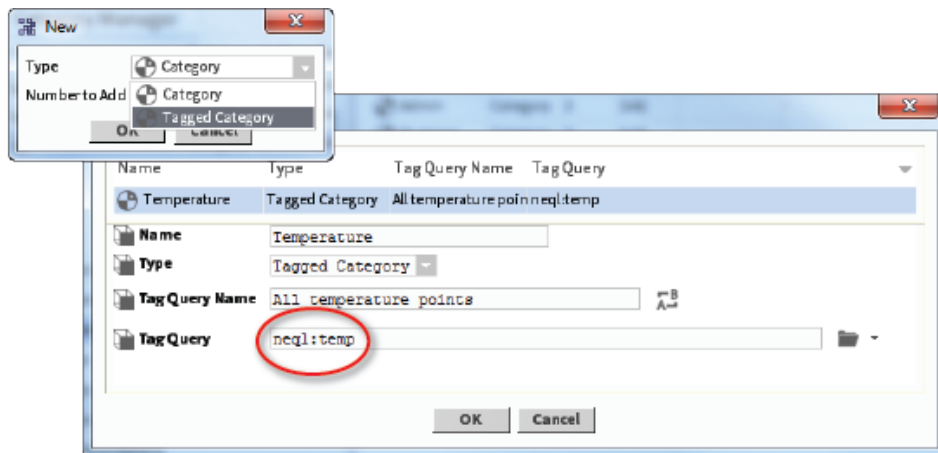
Figure 17 Creating basic categories



Property	Value	Description
Name	text	A name for the given category. This can be any name that describes how you are using the category.
Type	Category or Tagged Category	A basic Category is a name that is associated with individual objects. A Tagged Category includes a NEQL query that returns objects with tags that satisfy the query..
Index	number	A unique number for the category, as it is known to the station.

### Additional Tagged Category properties

Figure 18 Creating a tagged category



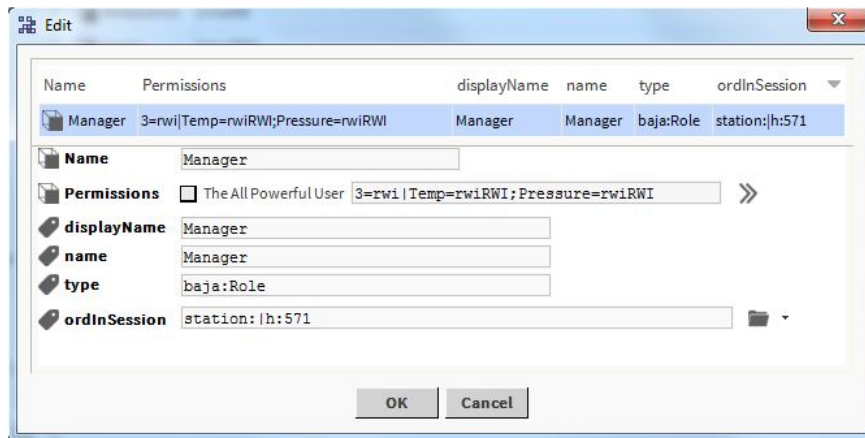


Property	Value	Description
Tag Query Name	text	A descriptive name to represent the results of the search.
Tag Query	NEQL	A NEQL query. This property is required when Type is Tagged Category.

### New/Edit roles window

This window creates and edits roles and permissions. You access it from the **Role Manager** view (**RoLeService**).

Figure 19 Example of an Edit role dialog box

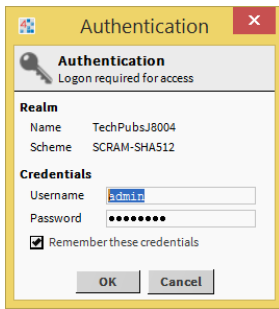


Property	Value	Description
Name	text string	Descriptive text that reflects the purpose of the entity or logical grouping.
Permissions	check box and chevron	<p>Checking the <b>All Powerful User</b> (super user) check box sets up a role that allows the user to access the entire station and file system. To set individual permissions for a role, click the chevron.</p> <p>The box between The All Powerful User and the chevron displays all categories. Basic categories display by index number (for example: 3=rwi). A vertical bar ( ) separates tagged categories, which display by name (for example: Temp=rwiRWI; Pressure=rwiRWI).</p>
tags	various	Any tags associated with the object appear at the end of the property sheet. The tag icon identifies them. The systems integrator sets up tags to provide additional, searchable information about the object.

### Platform Authentication window

This window opens after a secure platform connection is made. Its purpose is to authenticate the platform user.

Figure 20 Platform authentication window

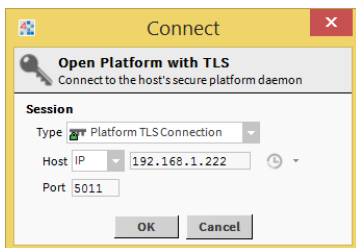


Name	Value	Description
Name	text	The name may include your company network name (when connecting to a Windows-based computer), the IP address of the controller, or host name of the controller.
Scheme	text	This information identifies the authentication scheme used to log on to a platform: HTTP-Basic, HTTP-Digest, or SCRAM-SHA512.
Credentials—Username	text	This is where you enter the platform user name created when the controller was commissioned. Access to this name is through <b>Platform Administration</b> → <b>User Accounts</b> .
Credentials—Password	text	This is the password created when the controller was commissioned. Access to this password is through <b>Platform Administration</b> → <b>User Accounts</b> .
Remember These Credentials	check box	Select this check box to have the system automatically fill in the user name and password when you log on the next time.

### Platform Connect window

This window opens when you open a platform (supervisor PC or controller).

Figure 21 Platform connect window

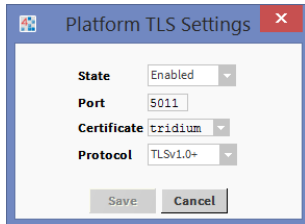


Name	Value	Description
Type	drop-down list	Defaults to Platform TLS Connection.
Host (type)		Defaults to IP.
IP address	text	This is where you enter the IP address or URL of the host platform.
Port	number	The port for secure platform communication. Defaults to 5011.

## Platform TLS settings

This window sets up the platformtls (niagarad) properties that provide server authentication and encryption. To access it, right-click **Platform**→**Views**→**Platform Administration** and double-click **Change SSL Settings**.

Figure 22 Platform SSL Settings dialog box

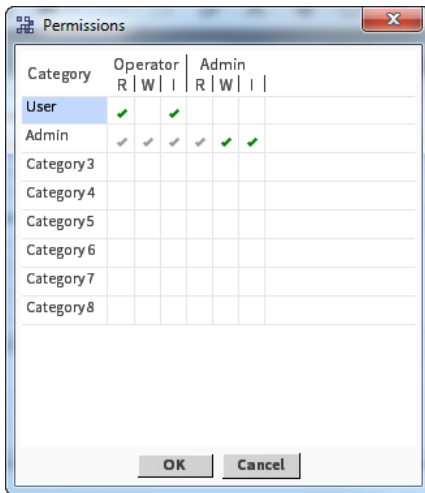


Name	Value	Description
State	Enabled or Disabled	Defaults to Enabled.
Port	number	The port for secure communication. Defaults to 5011
Certificate	drop-down list	Provides a list of available certificate aliases. The <code>tridium</code> certificate is the default, self-signed certificate created when you first accessed the platform.
Protocol	drop-down list TLSv1.0+ (default) TLSv1.1+ TLSv1.2	The minimum level of the TLS (Transport Layer Security) protocol to which the server will accept negotiation. The default includes versions 1.0, 1.1 and 1.2. It works with most clients, providing greater flexibility than an individual version.  During the handshake, the server and client agree on which protocol to use.  Change <b>Protocol</b> from the default if your network requires a specific version or if a future vulnerability is found in one of the versions.

## Permissions map

This window associates permissions with categories and permission levels. To access it, add or edit a role and click the chevron next to the **Permissions** property.

Figure 23 Permissions map

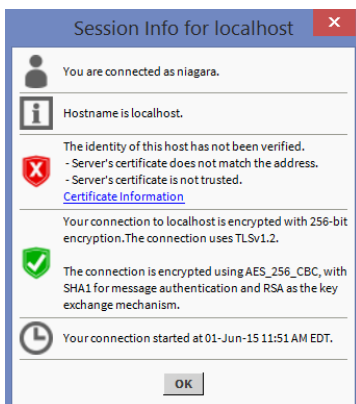









Column	Value	Description
Category	table row	User and Admin are default categories created by the <b>New Station</b> wizard. Each category occupies a single row in the Permissions map.
Operator	permission level	Provides a way to set access rights for components that are configured with the <code>operator</code> permission level.  Permission level is set by the <code>Operator</code> config flag on the component's <b>Property Sheet</b> .
Admin	permission level	Indicates that the object may be read, written or an action invoked by only system users that have been granted <code>admin</code> rights.  The Admin permission level is set by turning off the <code>Operator</code> config flag on the component's <b>Property Sheet</b> .

### Session Info window

The **Session Info** window reports the security status of the current communication session. You view this window by right-clicking the Session Info icon (i)

Figure 24 Example of Session Info when using a self-signed certificate

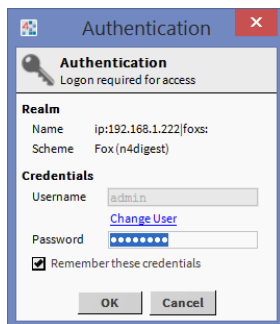


Screen element	Value	Description
	Connection	Identifies the user account that is logged in to the station.
	Hostname	Identifies the host name of the server.
 or 	Identity verification	Reports on the attempt to verify the authenticity of the server. A green shield indicates that a root CA certificate in the client's Trust Store verified the authenticity of the server certificate. A red shield indicates that the client system found no matching root CA certificate in a <b>Trust Store</b> with which to verify the server certificate. Clicking <i>Certificate Information</i> displays the certificate.
 or 	Encryption	Describes the Foxs session encryption strength. A green shield indicates that the transmission is encrypted. A red shield indicates that the transmission is not encrypted.
	Time	Logs when the Foxs session began.

### Station Authentication window

This window opens after a secure station connection is made. Its purpose is to authenticate the station user.

Figure 25 Station Authentication window



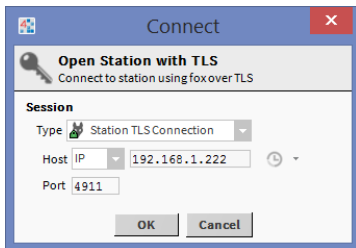
Name	Value	Description
Name	text	The station name, which, for a controller, is usually its IP address.
Scheme	text	This information identifies the authentication scheme used to log on to this station. The scheme used depends upon the user. Configuration is through the UserService.
Credentials—Username	text	This is where you enter your station user name.

Name	Value	Description
Credentials—Password	text	This is your platform password, which is the same as your personal station password.
Remember These Credentials	check box	Select this check box to have the system automatically fill in the user name and password when you log on the next time.

### Station Connect window

This window opens when you open a station.

Figure 26 Station Connect window



Name	Value	Description
Type	drop-down list	Defaults to Station TLS Connection.
Host (type)		Defaults to IP.
IP address	text	This is where you enter the IP address or URL of the host platform.
Port	number	The port for secure station (foxs) communication. Defaults to 4911.

### Plugins (views)

Plugins provide views of components and can be accessed in many ways. For example, double-click a component in the Nav tree to see its default view. In addition, you can right-click on a component and select from its **Views** menu.

For summary documentation on any view, select **Help→On View (F1)** from the menu or press **F1** while the view is open.

### Category Browser

This view is the default view of the station’s **CategoryService**, and typically where you spend most of your time assigning categories to components after initially creating the categories.

Figure 27 Category Browser

	Inherit	User	Admin	Operator	Viewer	Signal	Temp
Custom Config	n/a		•				
Services	✓		•				
Drivers	✓		•				
Apps		•	•				
category			•				
Ramp	✓		•			•	
Noise	✓		•				
SineWave	✓		•			•	
Threshold			•	•			
AddRamp	✓		•				
AddSine	✓		•				
GreaterThan	✓		•				
Temp1	✓		•				•
Temp2	✓		•				•
Alarm	✓		•				
AverageRoomTemp	✓		•				

There is no need to define component-to-category associations for Tagged Categories, so each Tagged Category column is grayed out and cannot be edited in the **Category Browser**. Any component with a tag that satisfies the NEQL query is visible in the **Category Browser**.

### Columns

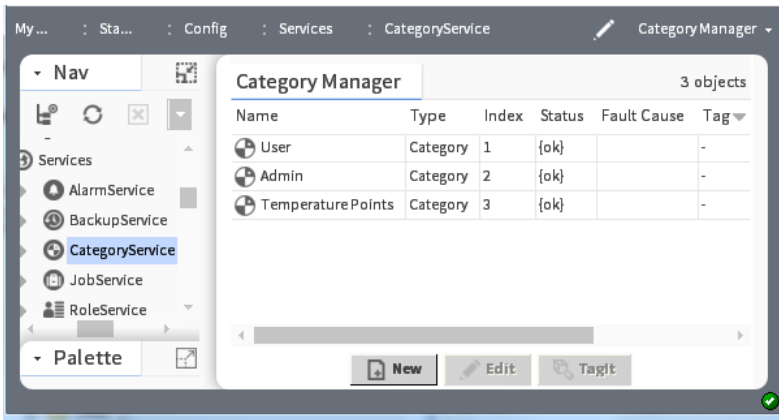
Column	Value	Description
Inherit	check mark or blank	A check mark indicates that the object inherits the category from its parent in the table.
User	Category 1	All system objects except for those listed as assigned to Admin are assigned to this category.
Admin	Category 2	These objects default to the Admin category: <ul style="list-style-type: none"> <li>The configuration services: <b>UserService</b>, <b>CategoryService</b>, and <b>ProgramService</b></li> <li>All files (the entire file space)</li> </ul>
Categories 3–10	bold bullet, grayed out bullet, or blank	Objects with a bold bullet have tags that satisfy a tagged category. A bold bullet indicates that the object is assigned to the category. A grayed out bullet indicates inheritance. Blank indicates that the category has not been assigned.

### Category Manager

**Category Manager** view provides the mechanism used to create, rename, and delete the groups that the security model uses to control access to the objects in a station. Once you create categories, you use the

**Category Browser** view to centrally assign system objects to them. Or, at the individual component level, you use a component’s **Category Sheet** view to assign it to categories.

Figure 28 Category Manager with tagged category, Temperature Points as Category 3



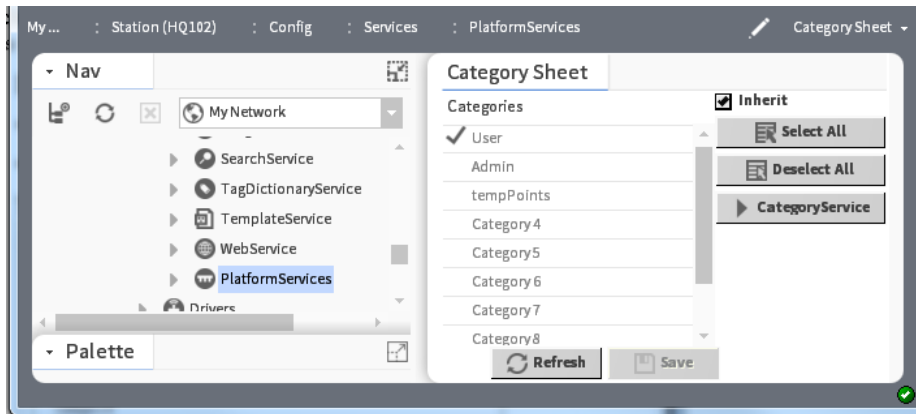
Column	Value	Description
Name	text string	Descriptive text that reflects the purpose of the entity or logical grouping.
Type	Category or Tagged Category	A basic Category is a name that is associated with individual objects. A Tagged Category includes a NEQL query that returns objects with tags that satisfy the query..
Index	number	A unique number for the category, as it is known to the station.
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. <ul style="list-style-type: none"> <li>• {ok} indicates that the component is polling successfully.</li> <li>• {down} indicates that polling is unsuccessful, perhaps because of an incorrect property.</li> <li>• {disabled} indicates that the <b>Enable</b> property is set to false.</li> <li>• <code>fault</code> indicates another problem.</li> </ul>
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Tag Query Name	text	A descriptive name to represent the results of the search.
Tag Query	NEQL	A NEQL query. This property is required when Type is Tagged Category.

### Category Sheet

In the **Category Sheet**, the component from which you accessed this sheet must be assigned to at least one category, or must inherit the category(ies) to which it belongs from its parent.



Figure 29 Category Sheet

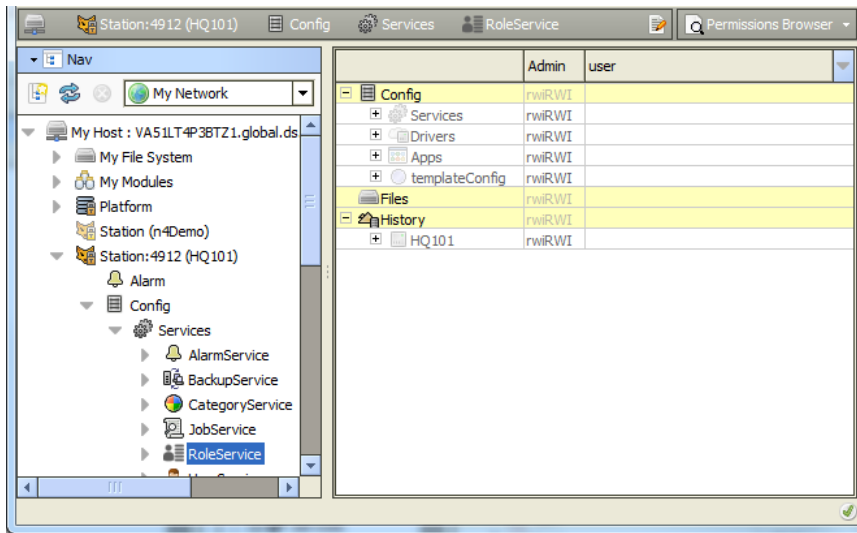


Option/button	Value	Description
Categories	text	Provides one table row for each category name.
Inherit	toggle	A check mark indicates that the component belongs to the same categories as its parent component. No check mark allows you to make explicit category assignments for this component.
Select All	button	Effective if <code>Inherit</code> is cleared, clicking this button assigns this component to all categories in this station.
Deselect All	button	Effective if <code>Inherit</code> is cleared, clicking this button removes this component from all categories.
CategoryService	button	Opens the <b>Category Browser</b> .
Refresh	button	Re-displays the <b>Category Sheet</b> .
Save	button	Records the changes made.

### Permissions Browser

This view allows you to quickly review the objects that someone assigned to a given role has the right to access and the action rights granted. You access this view by right-clicking `RoleService` in the Nav tree and clicking **Views→Permissions Browser**.

Figure 30 Permissions Browser view



Columns represent individual roles defined in the station.

Column	Value	Description
First column	Nav tree for station Config, Files and History	Each Nav tree node occupies a row in the table. This expandable tree lets you navigate to objects of interest to review current permissions.
Admin	permissionsR = readW = writel = invoke actionadmin level permissions appear in upper case.	Reports the rights assigned to the admin role. As this is a super user, admin has rights to read, write and invoke an action for all objects.
user	permissionsr = readw = writei = invoke actionoperator permissions appear in lower case.	Reports the rights assigned to the user role. The default is no rights assigned.

Row coloring in the **Permissions Browser** provides the same data as in the **Category Browser**, where:

- Yellow rows are objects explicitly assigned to one or more categories.
- Dimmed rows represent objects that inherit their parent’s category or categories.

Double-clicking on any column opens the user’s permissions map.

### 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.

## User Key Store tab

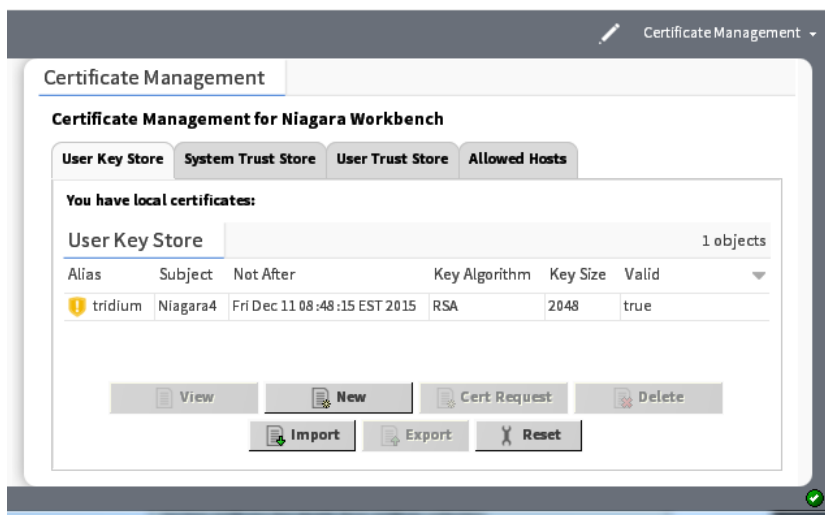
The **User Key Stores** contain server certificates and self-signed certificates with their matching keys. Each certificate has a pair of unique private and public encryption keys for each platform. A **User Key Store** supports the server side of the relationship by sending one of its signed server 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 31 Example of a Key Store



Name	Value	Description
Alias	text	A short name used to distinguish certificates from one another in the <b>Key Store</b> . 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 <b>Not Before</b> 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 <b>Not After</b> 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 <b>Certificate Request</b> 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 <b>User Key Store</b> and creates a new default certificate. It does not matter which certificate is selected when you click <b>Reset</b> .  <b>CAUTION:</b>  Do not reset without considering the consequences. The <b>Reset</b> 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

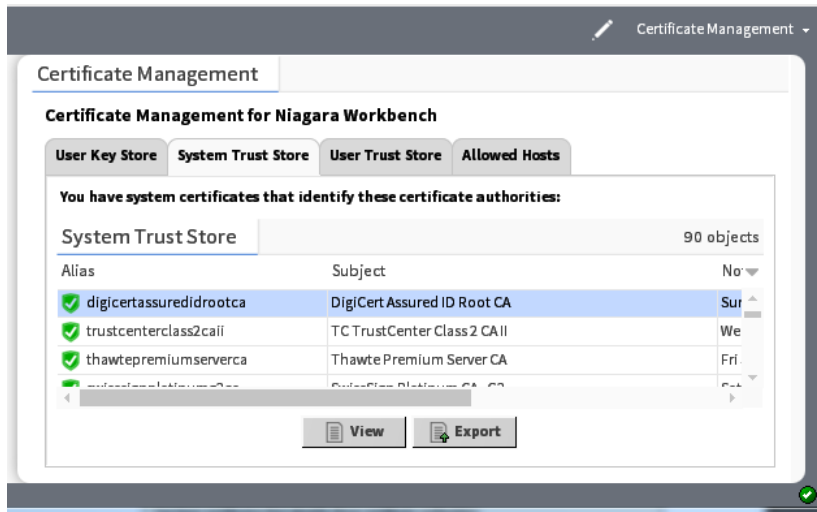
The **Trust Stores** contain signed and trusted root certificates with their public keys. These stores contain no private keys. A **Trust Store** supports the client side of the relationship by using its root CA certificates to verify the signatures of the certificates it receives from each server. If 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 32 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 <b>Key Store</b> . 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 <b>Not Before</b> 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 <b>Not After</b> 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.

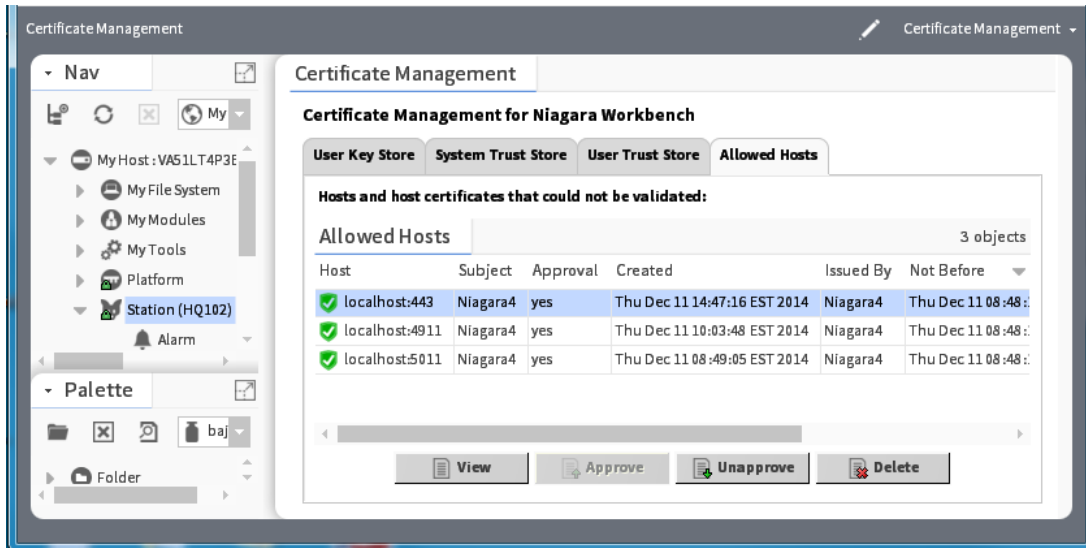
If this is a Workbench to station connection, the system prompts you to approve the host exemption. Workbench challenges server identity at connection for unapproved hosts and, unless specific permission is granted, prohibits communication. Once permission is granted, future communication occurs automatically (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 33 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 <b>Not Before</b> 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 <b>Not After</b> 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**

<b>Name</b>	<b>Value</b>	<b>Description</b>
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.



# Glossary

certificate	A PKI (Public Key Certificate) or digital certificate is an electronic document used to prove ownership of a public key. The certificate includes information about the key, the identity of its owner, and the digital signature of an entity that verified the validity of the certificate's contents. If the signature is valid, and the client can trust the signer, the client can be confident that it can use the public key contained in the certificate to communicate with the server.
Certificate Authority (CA)	An entity that issues the digital certificates used to certify the ownership of a public key by the named subject of the certificate. This allows system users to rely upon signatures or assertions made by the private key that corresponds to the certified public key. In this relationship model, the party that relies on the certificate trusts that the subject (owner) of the certificate is authentic because of the relationship of both parties to the CA.
chain of trust	Also called a web of trust, certification path, or trusted certificate tree is an approach to server verification that uses a self-signed certificate owned by a CA (Certificate Authority) to begin the authorized relationships. The private key of this root CA certificate signs a company's server certificate(s). Intermediate certificates may be used to further isolate relationships, such as by geographic location or corporate division.
key	A digital key is a very large, difficult-to-predict number surrounded by a certificate. Keys serve these purposes: 1) The public key of a root CA certificate in a client's <b>System</b> or <b>User Trust Store</b> verifies the authenticity of each server. 2) The private key of a trusted root CA certificate may sign other certificates. 3) After server authentication, matching public and private keys encrypt and decrypt data transmission.
NEQL	Niagara Entity Query Language provides a simple mechanism for querying objects with tags. Whereas BQL supports the tree semantics and pathing of Workbench component space (for example parent.parent) and BFormat operations, NEQL queries only for tags using the Niagara 4 tagable and entity APIs.
object	An object is the base class required for all system entities that conform to the baja model. Objects group information used to construct a model that includes building devices, virtual devices, individual points, users, system features and services. Objects appear in the Nav tree as files, modules, installers, administrators, copiers, drivers and apps. Metadata associated with objects, including categories, roles (permissions), and hierarchies, provide access control and configuration options to manage automated buildings efficiently.
permission	The right to access a component slot, folder, file or history. Three rights may be granted: the right to read information provided by the object, the right to write (change) the object, and the right to invoke an action on the object. Rights are granted using the <b>Role Manager's</b> permissions map. Permissions are applied to users by assigning each user a role. A super user is automatically granted every permission in every category for every object.
permission level	A slot config flag that indicates who is allowed to access the slot. When unchecked (the default) at least <code>admin-Read (R)</code> permission is required (as defined in the <b>Role Manager's</b> permissions map). When checked a user with a minimum of <code>operator-read</code> permission ( <code>r</code> ) may access the slot.
role	A logical grouping that is assigned as metadata to system users (human and machine) for security purposes. For example, roles may be used to group users as administrators ( <code>admin</code> ), regular users ( <code>user</code> ), and operators ( <code>operator</code> ).

	<p>Roles speed the management of permissions for a large number of users. The permissions of a group of users that share the same role can be updated by changing the role's permissions instead of updating each user's permissions individually.</p> <p>You manage roles using the <b>RoleService</b>.</p>
signature	<p>A digital signature combines a unique hash that is created using a cryptographic algorithm (such as SHA-512) with a public key. This is done by using a matching private key to encrypt the hash. The resulting signature is unique to both the certificate and the user. Finally, the signing process embeds the digital signature in the certificate.</p>
user permission	<p>See permission.</p>

# Index

## A

access rights	
reviewing .....	54
allowed hosts .....	31
Allowed Hosts .....	22
Allowed Hosts tab .....	86
ancestor permission level .....	55
audit log.....	42
authentication	
audit log .....	42
DigestScheme.....	58
HTTPBasicScheme.....	59
LegacyDigestScheme .....	59
of users.....	39
scheme, assigning to users .....	41
schemes.....	40
authentication scheme	
password management .....	41
AuthenticationService.....	57
authorization checklist .....	46
authorization management .....	7

## B

backup, <i>See</i> certificate, exporting	
basic categories.....	48
best practices.....	8

## C

categories .....	48
category	
adding and editing .....	49
basic.....	48
deleting .....	50
for assigning components.....	49
management.....	48
New window.....	72
Category	
Browser .....	78
Crowser .....	49
Category Manager.....	80
Category Sheet .....	80
CategoryService .....	59
certificate.....	16
backing up, <i>See</i> certificate, exporting	
create .....	70
creating .....	25
deleting .....	34
exporting.....	29
importing into a User Trust Store .....	29
importing into the User Key Store .....	28
naming convention .....	17
renewal.....	34

self-signed .....	13
Session Info.....	30
signing.....	28
stores locations .....	22–23
TLS	
session info.....	30
types .....	12
Certificate Management plugin .....	82
Certificate Signing Request	
folder structure .....	24
Certificate Signing window .....	68
certificate tree .....	15
certificates	
signing.....	15
certManagement folder .....	24
checklist	
certificate creation and signing .....	17
platform/station for server verification .....	19
Supervisor station for server identity.....	18
user authentication.....	39
client certificate.....	12
client/server relationships .....	11, 22, 32
component	
adding a component .....	53
assigning to a basic category .....	49
permission level .....	46
permissions.....	7, 45
permissions, troubleshooting.....	56
component permissions checklist .....	46
Components .....	57
config flag, setting.....	46
configuring client port .....	32
controller,	
replacing securely .....	37
cryptography.....	11
CSR, <i>See</i> Certificate Signing Request	
folder structure .....	24

## D

DigestScheme .....	39–40, 57–58
document change log .....	9

## E

edit.....	69
Edit roles window .....	73
email	
securing using TLS.....	32
encryption.....	14, 17

**F**

file permissions.....	55
Foxs properties .....	32
FoxService .....	63

**G**

Generate Self-Signed Certificate window.....	70
global password configuration .....	58

**H**

history export window .....	69
history permissions .....	55
HTTPBasicScheme .....	57, 59
https required to set password in browser .....	42
Https properties .....	32

**I**

identity verification .....	11, 15–16
inherited categories.....	48

**K**

keys public and private.....	14
---------------------------------	----

**L**

legacy authentication.....	40
LegacyDigestScheme.....	40, 57, 59
Legal notices .....	2

**N**

naming convention for certificates .....	17
New category window .....	72
niagara_user_home .....	24

**O**

Operator config flag .....	46
----------------------------	----

**P**

password expiration .....	41
global configuration .....	58
management.....	41
renewing an expired password.....	44

setting up .....	42
setting up for a user .....	42
strength.....	27
strength, setting up.....	41
troubleshooting .....	44
warning period.....	41
permission level.....	46
permissions .....	51
editing .....	53
to access files.....	55
to view history files.....	55
Permissions map .....	75
Permissions Browser .....	81
permissions checklist .....	46
PKI certificate creating .....	25
platform authentication window .....	73
connect window .....	74
opening a secure connection.....	20
stores .....	22
TLS properties.....	75
platform/station checklist for server verification.....	19
plugins Certificate Management view .....	82
Plugins .....	78
precautions .....	7
private key .....	14
public key.....	14–15

**R**

reference .....	57
roles about their configuration .....	51
add and edit.....	73
adding .....	51
and permission level.....	46
assigning to users.....	54
editing .....	53
reviewing access rights.....	54
testing .....	54
RoleService .....	61

**S**

secure communication .....	7, 32
configuring station security properties .....	32
configuring Supervisor and platforms.....	31
enabling clients .....	32
encryption .....	17
Foxs properties .....	32
FoxSrvce configuration properties .....	63
Https properties.....	32
NiagaraNetwork enabling.....	21

securing outgoing email .....	32
Workbench check list.....	17
security precautions.....	7
server authentication	
fixing error conditions .....	34
TCP ports.....	36
server certificate.....	12
server identity	
Supervisor setup check list.....	18
server identity verification	
setting up .....	17
server verification	
station setup check list .....	19
server/client relationships .....	32
Session Info .....	30
window.....	76
station	
authentication window .....	77
connect window.....	78
health .....	30
logging off.....	42
logging on .....	42
security model .....	45
set up server verification.....	17
stores .....	22
station-to-station user.....	40
stations	
configuring security properties .....	32
stores	
accessing .....	22
file names .....	24
locations on the Supervisor platform .....	23
System Trust Store .....	22

## T

tagged categories .....	48
TCP ports.....	36
TLS, <i>See</i> Transport Layer Security protocol	
keys.....	14
platform properties .....	75
Transport Layer Security protocol.....	11
troubleshooting	
component permissions.....	56
server authentication.....	34
Trust Store tab .....	84

## U

user	
adding and editing .....	41
authentication .....	40
password .....	41
user authentication .....	7
user authentication checklist .....	39
User Key Store .....	22
tab.....	83
User Trust Store .....	22

importing a certificate .....	29
UserService	
component permission level .....	47
property sheet .....	62
setting up a password .....	42

## V

views, *See* plugins

## W

WebService configuration properties.....	65
Workbench.....	17
stores .....	22