Preface

• Document Change Log

Document Change Log

Updates (changes or additions) to this document are listed as follows.
• Initial publication: August 30, 2012.
CONTENTS

Document Change Log .................................................................................. iii

SSL security for NiagaraAX ................................................................. 1–1
SSL Toolset features ............................................................................. 1–1
Frequently-asked questions ................................................................. 1–1
Best practices ......................................................................................... 1–2

Set up Workbench and stations for SSL .............................................. 2–1

Prerequisites ......................................................................................... 2–1
SSL setup checklists .......................................................................... 2–1
Set up Supervisor checklist ................................................................. 2–1
Set up JACE checklist ........................................................................ 2–2
Install SSL modules ........................................................................... 2–2
Install SSL modules ........................................................................... 2–2
Check for SSL license .......................................................................... 2–3

Enable SSL for the Supervisor and JACE platforms ......................... 2–3
Enable SSL .......................................................................................... 2–3
Open a secure platform connection (Niagarad) .................................. 2–4

Enable SSL for the Supervisor and JACE stations .............................. 2–6
Enable NiagaraNetwork ....................................................................... 2–6
Enable the Web Service connection .................................................. 2–7
Confirm the Fox Service connection .................................................. 2–7
Set up client/server relationships ....................................................... 2–8
Enable clients and configure them for the correct port ....................... 2–8

Configure email .................................................................................... 2–8
Configure email .................................................................................... 2–9

Upgrade from crypto.jar ..................................................................... 2–9
Upgrade from Crypto.jar .................................................................... 2–10

Create certificates ................................................................................ 3–1

Set up SSL security checklist ............................................................ 3–1
Create certificates ................................................................................ 3–1
Set up stations ...................................................................................... 3–2

Design the certificate chain of trust .................................................... 3–2
Set up a folder structure ...................................................................... 3–2
Establish a naming convention .......................................................... 3–2

Set up the root and intermediate certificates .................................... 3–2
Create the root and intermediate certificates ..................................... 3–3
Create a CSR for the intermediate certificate ..................................... 3–4
Sign the intermediate certificate using the root certificate’s private key 3–5
Import intermediate certificate back into the Key Store ....................... 3–6
Export the root and intermediate certificates ..................................... 3–6
Set up the JACE and Supervisor server certificates .................................................. 3–7
  Create new JACE and Supervisor server certificates ........................................ 3–7
  Create a CSR for each server certificate ......................................................... 3–8

Sign the server certificates ....................................................................................... 3–9
  Sign the server certificates using the intermediate certificates ....................... 3–9

Import the signed server certificate and configure each station .................. 3–10
  Import the server certificate into the Key Store ............................................ 3–10
  Select the platform server certificate ......................................................... 3–11
  Select the Https and Foss service certificates .............................................. 3–11

Set up the Trust Stores ......................................................................................... 3–12
  Set up the platform/station Trust Stores ....................................................... 3–12
  Set up the Workbench Trust Store .................................................................. 3–12

Install certificates in a client browser ................................................................. 3–12
  View the certificates in Internet Explorer 8 .................................................... 3–12
  Install a certificate in Internet Explorer 8 ....................................................... 3–13
  Install a certificate in Firefox 10 .................................................................... 3–13
  Install a certificate in Chrome 17 ................................................................. 3–13

Configure station-to-station .................................................................................. 3–13
  Configure the Supervisor for provisioning with SSL .................................... 3–13
  Configure each JACE station ......................................................................... 3–14

Updating a certificate .............................................................................................. 3–15
  Delete a certificate ............................................................................................ 3–15

Back up the stores ................................................................................................. 3–15
  To back up the stores ....................................................................................... 3–16

Managing allowed hosts ......................................................................................... 3–16

Test station health ................................................................................................. 3–16

About SSL—Alice, Bob, Cathy and Bart ................................................................. 4–1
  What makes any system secure? ................................................................. 4–1

Friends and enemies ............................................................................................... 4–1
  When trust is secure ....................................................................................... 4–1
  When things go wrong .................................................................................... 4–2
  Verifying authenticity ...................................................................................... 4–2

Identity verification ............................................................................................... 4–3
  More about certificates .................................................................................. 4–3
  Keys ................................................................................................................ 4–4
  Signing a certificate with a private key .......................................................... 4–4
  Creating a chain of trust ................................................................................. 4–6
  More about intermediate certificates ............................................................. 4–8

Cryptography .......................................................................................................... 4–8
  Encrypting and decrypting the handshake ..................................................... 4–8
  Data transmission after the handshake ......................................................... 4–8
  More about key size ....................................................................................... 4–9

About NiagaraAX SSL ............................................................................................ 5–1

NiagaraAX’s client/server architecture ............................................................... 5–1

About the certificate creation and signing process ........................................ 5–2
  Certificate creation ......................................................................................... 5–2
  Workbench certificate signing ....................................................................... 5–3
  JACE certificate signing .................................................................................. 5–4
  Supervisor certificate signing ....................................................................... 5–5
  Setting up the client Trust Stores ................................................................. 5–5
About NiagaraAX SSL certificates ........................................... 5–7
  Types of NiagaraAX certificates ........................................... 5–7
  About self-signed certificates .............................................. 5–7

About the SSL Toolset ......................................................... 5–8
  About the Key Stores ......................................................... 5–9
  About the Trust Stores ....................................................... 5–10
  About Allowed Hosts ......................................................... 5–10
  About the default certificate .............................................. 5–11
  Workbench keys and certificates ...................................... 5–11
    View the Workbench stores ............................................. 5–11
  Platform keys and certificates ........................................... 5–11
    Access platform stores using a platform connection ............. 5–11
    Access the platform stores from a station ....................... 5–12
    View the platform’s default certificate .............................. 5–12

NiagaraAX SSL best practices .............................................. 5–13
  How many keys and certificates do I need? ....................... 5–13
  Who should sign my certificates? ..................................... 5–13

Troubleshooting ................................................................. 6–1
  Fix error conditions ......................................................... 6–1

  Reset or replace a JACE securely ....................................... 6–2
    To reset or replace a JACE with security ......................... 6–2

Reference ........................................................................... 7–1
  SSL Toolset terminology ................................................... 7–1
  About TCP ports ............................................................. 7–2

  About the Certificate Management dialog .......................... 7–2
    About the Key Store tab ................................................ 7–3
    About the Private Key Password dialog ............................ 7–5
    About the Trust Store tab ............................................. 7–6
    About the Allowed Hosts tab ......................................... 7–6

  About the Certificate Signing dialog ................................ 7–7

  SSL configuration options ............................................... 7–7
    Platform SSL properties ................................................ 7–7
    Fox Service configuration properties ............................... 7–8
    Web Service configuration properties ............................... 7–9
SSL security for NiagaraAX

The NiagaraAX implementation of the industry-standard Secure Socket Layer (SSLv3) and Transport Layer Security (TLSv1) protocols provides:

- Server authentication.
- Encryption/decryption of data transmitted between client and server.

Note: NiagaraAX’s implementation of SSL does not secure data stored on a storage device and it is not available on the J9 JVM platform.

The NiagaraAX SSL Toolset consists of the menu items, dialogs and property sheets used to configure Workbench, each Supervisor, and JACE for SSL security.

This topic contains these sub-topics:

- “SSL Toolset features” on page 1-1
- “Frequently-asked questions” on page 1-1
- “Best practices” on page 1-2

SSL Toolset features

- Secure Platform—Niagarad over SSL for JACE network controllers that support HotSpot JVM
- Secure Fox Service—Fox over SSL (foxs, pronounced “fox-s”)
- Secure Web Service—Http over SSL (https, pronounced “h-t-t-p-s”)
- The creation of self-signed server certificates
- Certificate management for server and Certificate Authority (CA) certificates (both third-party and company-signed certificates) including:
  - Generation of Certificate Signing Requests (CSR)
  - Installation and management of certificates
  - Management of trusted CA certificates
- Allowed Hosts list for manually controlling the hosts (servers) to which a client can connect with a certificate that has not been validated (no matching certificate exists in the client’s Trust Store).
- Certificate signing tool for companies that decide to become their own Certificate Authority
- Email security

Frequently-asked questions

Q: What is SSL? What is TLS?
A: Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are cryptographic protocols for server authentication and secure encryption and decryption of data over the internet.

Q: What is the difference between SSL and TLS? How long have they been out? Is one better than the other?
A: Both standards have been out for a while and both offer the same level of security. The two standards do not compete.

SSL v. 3 is the currently-accepted SSL version. (Version 1 was actually never released. Version 2 had vulnerabilities and is no longer supported by browsers.)

TLS v. 1 is based on SSL v. 3, although the two are incompatible. TLS v. 1.11 and 1.12 provide additional minor feature enhancements.
Q: What organizations support SSL and TLS standards?
A: SSL originated with Netscape Corporation in the 1990s. TLS is developed and promoted by the Internet Engineering Task Force (IETF), a voluntary organization that cooperates closely with the World Wide Web consortium (W3C) and the International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC).

Q: Is an SSL v. 3 key 128-bit based?
A: The number of bits in the key depends on the ciphers used. SSL and TLS allow you to choose the key size. SSL Toolset supports a maximum 4096-bit key.

Q: How does SSL Toolset compare with credit card security?
A: Any technology is as secure as the guidelines followed. For data transport, SSL Toolset provides the same type of security that is provided by the credit card and banking industries.

Q: Our company has multiple locations. Each location has a network of JACEs, none of which is on the internet. How should we ensure that no unauthorized person can intercept communication at any of our sites?
A: Since the communication you are concerned with occurs “in house,” in other words, no JACE is on the internet, you can create and sign your own corporate root certificate and use its private key to sign a certificate for each location. Then, you use the location certificate’s private key to sign the certificate for each JACE, and distribute the signed certificate with each JACE or, if the JACES are already in the field, you import the certificate to each JACE’s Trust Store. When a JACE comes on line, the handshake validates the entire certificate chain of trust.

Q: Our company already has signed certificates. Can they be used on our JACEs?
A: In addition to signed client certificates, each JACE requires its own unique private key. While it is possible to request a certificate with a private key for each JACE from your IT department, you can also have the JACE generate its own unique pair of public and private keys. This avoids the requirement to transmit an externally-generated private key over an exposed connection.

Q: Do we have to use the SSL Toolset tools to generate and sign certificates?
A: No. Certificates, public and private keys conform to established standards. You can use any software tool to create them. The tools provided by Workbench are designed to be intuitive and easy to use.

Q: Can the same certificate be used for Foxs, Https, WebService, and Platform (Niagarad) security?
A: Yes, the same strong certificate for all three is usually adequate. But you may have your own reasons for wanting separate security for each service. For example, if you have a lot of people using a station, and the station connection is compromised, a separate certificate for the Niagarad connection would stop the breach immediately.

Q: Does using multiple keys slow performance?
A: No. Performance is primarily impacted by the size of the key and the frequency of handshakes. The slowest performance occurs when a JACE is generating a new key. Once the key is generated, performance is only slightly impacted.

Q: Since it takes a long time to generate a key on a JACE, is it acceptable to generate the key on a PC and download it into the JACE?
A: You can generate a key in Workbench on a PC, export it from the PC, and import it into the JACE, but be aware that the transmission may be over an exposed connection.

Q: Will there be an upgrade path for SSL security in future versions of Workbench?
A: Yes. You will be able to export a security folder and import it into new versions of Workbench.

Best practices

As you plan to implement security in your system, consider the following:

- To keep certificate usage straight, plan your station and certificate names carefully. For station names, include the words “supervisor” and “JACE.” For certificates and Certificate Signing Requests (CSRs), include the station name to which the certificate belongs. Consider adding the words “Server” or “CA” to differentiate certificate types and “Public” or “Private” to identify certificates that contain the private key. Certificates exported with their private keys should be heavily encrypted and password protected when stored on a company server.
- Computer equipment should be secured in a locked room. Wiring should be protected to prevent an unauthorized person from plugging in to it.
- Software passwords need to be stored and used securely. Access to the file system needs to be controlled.
• The transmission of data within a network, which may occur over wires or a wireless connection, needs to be secure. This is where SSL Toolset applies. It is concerned with the transport, not the storage of data.

• Communication security slows processing in two ways: 1) when generating a complex key on a JACE and 2) during the initial handshake to establish communication. Once communication is established, data are encrypted and decrypted using a single key, which speeds processing. Actual throughput depends on what you are doing.

• Third-party certificates, such as those from Certificate Authorities (CAs) VeriSign, Thawte, and Go Daddy, are installed with the browser. This requires trust in the browser installation program. If your company is acting as a CA, your signed client certificate(s) need to be separately installed in the user’s browser.

• A certificate chain of trust allows you to create a certificate, have it signed by a Certificate Authority (or become your own Certificate Authority), and then use that root certificate to sign the certificate generated in each JACE.

• Generating a key and certificate on a JACE is the most secure way to create the certificate for the JACE even though the generation process takes time.

• Although it can be done, it is not recommended to generate a certificate and key on a local computer, and then download it into the JACE.

• You can export all keys in case you have to reinstall them in the future. The Key Store for the station is not part of the station. Take the computer on which the Key Store is stored off the network, restore your keys, and bring it back on the network.

• To access the corporate JACE network from a remote location via the internet, use a VPN solution that incorporates RSA two-factor authentication.

• For high traffic stations (especially stations that provide public access to a JACE network), you can secure Niagara using a separate certificate from that used for Fox and Web service.

• If your network will contain a combination of SSL-secured and exposed JACEs (not SSL-secured), do not set up a secure Supervisor station.

Note: If you are really concerned about security, do not use self-signed certificates; do not use guest accounts; do not use the default password; and do not expose host stations publicly to the internet.
Set up Workbench and stations for SSL

This section explains how to install, enable and configure the SSL Toolset or upgrade to it from NiagaraAX CryptoService. Specifically, it contains these topics:

- "Prerequisites" on page 2-1
- "SSL setup checklists" on page 2-1
- "Install SSL modules" on page 2-2
- "Check for SSL license" on page 2-3
- "Enable SSL for the Supervisor and JACE platforms" on page 2-5
- "Enable SSL for the Supervisor and JACE stations" on page 2-6
- "Configure email" on page 2-8
- "Upgrade from crypto.jar" on page 2-9

Note: The platform on which SSL Toolset is to be installed or upgraded should be off line or located in a private, closed network. If an existing platform is on the internet or any company network, remove it before you get started.

Prerequisites

- A NiagaraAX platform that supports Hot Spot VM (virtual machine) from Sun Microsystems. (The JACE 2, 4 and 5 support the IBM J9 VM and require the previous Crypto module (crypto.jar)).
- NiagaraAX 3.7 or later.
- These modules installed: cryptoCore, daemonCrypto, and platCrypto.
- SSL Toolset license for the Supervisor and each JACE. This is the same license required by NiagaraAX version 3.6 and earlier to use the Crypto module.

SSL setup checklists

Note: If you are upgrading from a previous version of the software that supports the Crypto feature, see "Upgrade from crypto.jar" on page 2-9 before you continue with the checklists.

- "Set up Supervisor checklist" on page 2-1
- "Set up JACE checklist" on page 2-2

The best practice is to configure SSL for a Supervisor station or a JACE while the units are off line. Use a "crossover" cable from your laptop computer to connect to a JACE. If you must install SSL while a JACE is online, connect the crossover cable to a secondary port.

Set up Supervisor checklist

1. Supervisor PC off the company LAN and global internet.
2. Supervisor licensed for SSL.
   See "Check for SSL license" on page 2-3.
3. Platform (Niagarad) enabled.
   See "Enable SSL for the Supervisor and JACE platforms" on page 2-3.
4. SSL modules installed.
   See "Install SSL modules" on page 2-2
5. NiagaraNetwork enabled.
   See "Enable NiagaraNetwork" on page 2-6.
6. WebService enabled.
   See "Enable the Web Service connection" on page 2-7.
7. **FoxService** enabled and port assigned (defaults to 4911).  
   See “Confirm the Fox Service connection” on page 2-7.
8. JACE station set up as Supervisor client.  
   See “Set up client/server relationships” on page 2-8.
9. JACE client connection **Use Foxs** set to **true**, and JACE client connection port set correctly.  
   See “Enable clients and configure them for the correct port” on page 2-8
10. Certificates created, signed and imported.  
    See Chapter 3, “Create certificates”.
    See “Import the signed server certificate and configure each station” on page 3-9.

SSL is now set up for the Supervisor. Configuration continues with setting up the JACE platforms/stations (“Set up JACE checklist” on page 2-2), and setting up certificates (Chapter 3, “Create certificates”).

**Set up JACE checklist**
1. JACE PC off the network, global internet, and connected directly to the computer using a crossover cable.
2. JACE licensed for SSL.  
   See “Check for SSL license” on page 2-3.
3. Platform (Niagarad) enabled.  
   See “Enable SSL for the Supervisor and JACE platforms” on page 2-3.
4. **NiagaraNetwork** enabled.  
   See “Enable NiagaraNetwork” on page 2-6.
5. **WebService** enabled. See “Enable the Web Service connection” on page 2-7
6. **FoxService** enabled and port assigned (defaults to 4911)  
   See “Confirm the Fox Service connection” on page 2-7
7. Supervisor set up as JACE client.  
   See “Set up client/server relationships” on page 2-8.
8. Supervisor client connection **Use Foxs** set to **true** and Supervisor client connection port set correctly.  
   See “Enable clients and configure them for the correct port” on page 2-8.
9. Certificates created, signed and imported.  
   See Chapter 3, “Create certificates”.
    See “Import the signed server certificate and configure each station” on page 3-9.

SSL is now set up for the JACE.

**Install SSL modules**

Commissioning a new station installs these modules as part of the station installation:

- **cryptoCore**
- **daemonCrypto**
- **platCrypto**

See “Install SSL modules” on page 2-2

**Install SSL modules**

If the modules are missing, install them as follows:

1. **Step 1**  
   Double-click the **Platform** node in the Nav tree and double-click the **Software Manager**.
2. **Step 2**  
   Use the wizard to install the modules.
   You may need to import them if they are not in the database.

For more information about Software Manager and how to install modules, see “Software Manager” in the NiagaraAX Platform Guide.
Chapter 2 – Set up Workbench and stations for SSL

Check for SSL license

The license to use SSL is the same as the license to use the previous version, which was called Crypto-Service. In Services > PlatformServices view the license.

![License view](image)

If the crypto feature name is included with ssl="true", the platform is licensed to use SSL. For licensing information, see “Supervisor License Manager” in the NiagaraAX Provisioning Guide for Niagara Networks.

Enable SSL for the Supervisor and JACE platforms

Platform and station security are independent of one another. You can configure SSL for only your stations or for both your platforms (Niagarad) and stations (Fox).

A station’s “window” into the platform-resident SSL features is just like any other platform service under the station’s PlatformService node in the Nav tree. This means that anything configured in PlatformServices is independent of whatever station is running.

This topic explains how to:

- “Enable SSL” on page 2-3
- “Open a secure platform connection (Niagarad)” on page 2-4

To configure SSL for your stations, see “Enable SSL for the Supervisor and JACE stations” on page 2-6.

Enable SSL

This procedure involves making an exposed Fox platform connection, enabling SSL, disconnecting from the platform, and re-connecting using a secure connection.

**Step 1** Make an exposed connection to the platform.

**Step 2** Under Platform in the Nav tree, double-click Platform Administration.

The Platform Administration view appears.
Step 3  Click **Change SSL Settings**.

The **Platform SSL Settings** dialog appears as follows:

![SSL enabled dialog](image)

The default **Port** for platform connections over SSL is 4911.

**Certificate** provides a drop-down list of available certificates. Assuming this is a new platform, the only certificate in the list is **tridium**, the auto-generated self-signed certificate.

The **Protocol** list allows you to choose one protocol over the other (SSL or TLS). SSL Toolset supports both, which is the default way browsers work. There is no performance reason to choose one over the other. The option is provided in case your situation (contract or agreement) requires you to use one or the other.

Step 4  Change **State** to **Enabled** and click **Save**.

The system enables the SSL port and restarts the exposed connection, most likely using the TLS protocol. This restart occurs for reasons other than security.

Step 5  Disconnect from the platform session (right-click the platform in the Nav tree and click **Close**).

### Open a secure platform connection (Niagarad)

Now that SSL is enabled, you can open the platform securely.

Step 1  Click **File > Open > Open Platform**.

The **Open Platform** dialog appears.

![Open Platform dialog](image)

Step 2  Select **Platform SSL Connection** from the **Session Type** drop-down list.

Step 3  Enter your **Credentials** and click **OK**.
Chapter 2 – Set up Workbench and stations for SSL

Enable SSL for the Supervisor and JACE platforms
August 30, 2012
Set up JACE checklist

The system displays the unable-to-verify-host-identity message.

**Figure 2-5** Unable to verify server identity using self-signed certificate

This error message is expected for two reasons:

- The certificate's **Subject**, or Common Name (CN) is NiagaraAX, which does not match the host's name, which is usually its IP address.
- The certificate signature does not match the signature on any certificate in the client’s **Trust Store**. The fact that the **Issued By** and **Subject** are the same would indicate that the certificate has been self-signed.

**Step 4** Since this is the default **tridium** certificate, which can be trusted, click **Accept**.
Accepting the certificate creates an approved host exception in the **Allowed Hosts** list. Next, the system asks you to confirm your platform credentials again.

**Step 5** Enter your credentials and click **OK**.
The platform is now connected over a secure connection. All data transmitted is encrypted and decrypted but, server identity was not be validated.

**Step 6** To confirm this state, right-click **Platform** and click **Session Info**.
Enable SSL for the Supervisor and JACE stations

This topic explains how to enable SSL and how to turn on security for NiagaraAX Web Service (https) and Fox Service (foxs).

- “Enable NiagaraNetwork” on page 2-6
- “Enable the Web Service connection” on page 2-7
- “Confirm the Fox Service connection” on page 2-7
- “Set up client/server relationships” on page 2-8
- “Enable clients and configure them for the correct port” on page 2-8

Enable NiagaraNetwork

Step 1 Double-click Drivers in the Config folder.
Step 2 Select the Property Sheet view.

The system displays session information.

Figure 2-6  Session Information

- The red shield with the X indicates that the software was unable to verify the authenticity of the server certificate. It is a self-signed certificate and no matching root certificate exists in the platform Trust Store. To view the certificate, click the link.
- The green shield with the check mark indicates that encryption and decryption are enabled (this is a secure connection). In this example, the secure connection is using TLSv1 as the protocol and data is encoded/decoded using “AES_128_CBC with SHA1.”

Step 7 Click OK.

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

Step 8 To view this allowed host, click Tools > Certificate Management and check the Allowed Hosts tab.

Figure 2-7  Approved default tridium certificate in the Allowed Hosts list

The green shield indicates that the exception is approved.

Enable SSL for the Supervisor and JACE stations

Chapter 2 – Set up Workbench and stations for SSL

Set up JACE checklist

August 30, 2012
Enable SSL for the Supervisor and JACE stations

Set up JACE checklist

Chapter 2 – Set up Workbench and stations for SSL

August 30, 2012

Figure 2-8  NiagaraNetwork properties

Step 3  Confirm that **Enabled** is set to **true**

Enable the Web Service connection

Step 1  Expand the **Config > Services** node in the Nav tree and double-click **WebService**. The Web Service properties appear.

Figure 2-9  Web Service properties

Step 2  Set **Https Enabled** to **true**.

**Https** **Min Protocol** is already set to SSL and TLS.

Leave **Https Cert** configured to use the default **tridium** certificate until you have created the certificate to use here.

**Note:** Between the highlighted options above is **Https Only**. If you set this option to **true**, and attempt to connect using http, NiagaraAX redirects the connection to the SSL connection (https). If you are implementing SSL security into an existing system, you may have many pointers to the old http port number. This option saves having to manually change each occurrence of the http port. (Ideally you want to turn off http completely, but sometimes it is not practical.)

Step 3  Click **Save**.
Enable SSL for the Supervisor and JACE stations

Chapter 2 – Set up Workbench and stations for SSL

Set up JACE checklist

August 30, 2012

Confirm the Fox Service connection

Step 1
Under the station, expand the Config > Drivers node in the Nav tree, right-click Niagara Network, click Views > Property Sheet, and expand the Fox Service properties. The NiagaraNetwork properties contain the Fox Service properties.

Figure 2-10  Fox Service properties

Step 2
Notice that Foxs Enabled may already be set to true. If you enabled https in the WebService, the system automatically enabled foxs in the FoxService; there's nothing to change here.

Note: Between the highlighted options above is Foxs Only. If you set this option to true, and attempt to connect using Fox, NiagaraAX redirects the connection to the SSL connection (foxs). If you are implementing SSL security into an existing system, you may have many pointers to the old Fox port number. This option saves having to manually change each occurrence of the Fox port. (Ideally you want to turn off fox completely, but sometimes it isn't practical.)

Step 3
Notice that the same default 1024-bit certificate (Foxs Cert field), which was used for the platform is selected here.
This default server certificate is unique to this platform (and station). It is not the same as the default certificate used by Workbench.
If you choose to use a different certificate for your FoxService from that used with your WebService, this is where you would specify it.

Set 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 JACE station and a client for the JACE station exists in the Supervisor.

Step 1
Check the Drivers > NiagaraNetwork node in the Supervisor Nav tree. It should contain a node for the JACE station.
Figure 2-11  Nav tree with Supervisor and demo stations set up as clients of each other

Step 2  Check the Drivers > NiagaraNetwork node in the JACE Nav tree. It should contain a node for the Supervisor station.

Enable clients and configure them for the correct port

Step 1  If it is not already open, double-click the NiagaraNetwork node in the Nav tree of both the Supervisor and the JACE 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 JACE station as client; and for the JACE station, this is the Supervisor station as client.

Step 3  For each client, confirm that the correct Fox Port (4911) is being used and that Use Foxs is set to true.

Configure email

Email can use the platform Key Store to provide secure outgoing messaging. For incoming email the server functions as a client.

- “Configure email” on page 2-9

Configure email

Step 1  Double-click the EmailService node under Services in the station’s Nav tree, then double-click the IncomingAccount or OutgoingAccount in the Email Account Manager table.
The Edit email account dialog appears.

**Figure 2-12  Example of the outgoing EmailServices property sheet**

![Figure 2-12](image)

**Note:** If SSL is enabled for the SMTP Transport Protocol, the **Port** is different from that shown in the example above. The default port is 465. Do not change the **Use SSL** setting from true to false or vice versa without changing the Port.

**Step 2** Set **Use SSL** to true, change **Port** to the appropriate SMTP SSL port (default 465), and click **Save**.

**Step 3** Import the email server’s client certificate into the server **Trust Store**.

If a valid email client certificate is not available in the **Trust Store**, the system challenges the email message. Accepting the exception creates a record in the **Allowed Hosts** list. For how to accept the exception in the **Allowed Hosts** list, see “Managing allowed hosts” on page 3-13 and “About the Allowed Hosts tab” on page 7-6.

### Upgrade from crypto.jar

The CryptoService feature provided by versions of NiagaraAX prior to version 3.7 was station-based. All SSL configuration options were stored in the station database. Prior to version 3.7, CryptoService supported only SSLv3 HTTP communication. The Fox Service and Niagarad protocols were not protected.

The SSL Toolset is platform-based and can be configured without an open station. In addition to protecting HTTP, the SSL Toolset:

- Protects Fox Service and Niagarad.
- Provides for the creation and signing of certificates for each NiagaraAX service (Fox, HTTP and Niagarad).
- Unsecure connections can be denied.
- You have a choice of cryptographic protocols to use (SSL or TLS).
- Any port can be changed.

If you have been using the CryptoService with a version of NiagaraAX prior to version 3.7, follow this general procedure to upgrade to SSL Toolset. More information about upgrading a JACE can be found in the NiagaraAX 3.7 Platform Guide.

"Upgrade from Crypto.jar" on page 2-10

### Upgrade from Crypto.jar

**Step 1** Back up and save the station.

**Step 2** Put the platform in a safe, non-exposed network, such as a private, closed network.

**Step 3** Delete the **CryptoService** from the station and save the station.
Chapter 2 – Set up Workbench and stations for SSL

August 30, 2012

Step 4  Stop the station.
Step 5  Remove crypto.jar from the platform using Software Manager.
Step 6  Run the Commissioning Wizard to upgrade the platform.

Make sure that the following modules are selected for installation:

- cryptoCore
- daemonCrypto
- platCrypto

When commissioning is complete, the platform reboots and the station starts. Continue with “SSL setup checklists” on page 2-1.
Create certificates

The SSL Toolset provides the mechanism to create two types of certificates for NiagaraAX networks:

- A server certificate for each JACE and Supervisor platform.
- A Certificate Authority (CA) certificate that you can use to sign other certificates in a certificate chain of trust. Ideally, CA certificates are generated and kept in a vault where there is no network connectivity.

**Note:** SSL Toolset provides an easy-to-use interface for creating these certificates, but you may use other tools to create your certificates.

Third-party CAs, such as VeriSign, are unlikely to sign your CA certificates. This section explains how to become your own Certificate Authority, and includes:

- “Set up certificates checklist” on page 3-1
- “Design the certificate chain of trust” on page 3-2
- “Set up the root and intermediate certificates” on page 3-2
- “Set up the JACE and Supervisor server certificates” on page 3-7
- “Sign the server certificates” on page 3-8
- “Import the signed server certificate and configure each station” on page 3-9
- “Set up the Trust Stores” on page 3-11
- “Install certificates in a client browser” on page 3-12
- “Updating a certificate” on page 3-12
- “Back up the stores” on page 3-13
- “Managing allowed hosts” on page 3-13
- “Test station health” on page 3-13

**Set up certificates checklist**

This checklist assumes that within a given platform, a single server certificate will secure Niagara, Foxs, and Htts.

1. Folder structure created and naming convention designed.
   See “Set up a folder structure” on page 3-2.
   See “Establish a naming convention” on page 3-2

2. Root and intermediate certificates created.
   See “Create the root and intermediate certificates” on page 3-2.
   See “Create a CSR for the intermediate certificate” on page 3-4.
   See “Sign the intermediate certificate using the root certificate’s private key” on page 3-5.
   See “Import intermediate certificate back into the Key Store” on page 3-5.
   See “Export the root and intermediate certificates” on page 3-6.

3. Server certificate created for each JACE.
   See “Create new JACE and Supervisor server certificates” on page 3-7.
   See “Create a CSR for each server certificate” on page 3-8.
   See “Sign the server certificates using the intermediate certificates” on page 3-8.
   See “Import the server certificate into the Key Store” on page 3-10.

4. Certificates imported into client Trust Stores
   See “Set up the platform/station Trust Stores” on page 3-11.
   See “Set up the Workbench Trust Store” on page 3-12.

5. Certificate to use for each service selected.
   See “Import the signed server certificate and configure each station” on page 3-9.
Design the certificate chain of trust

To be your own CA, you will need a root certificate and possibly one or more intermediate certificates as well as a server certificate for every Supervisor and JACE in the network. The private key of your root certificate will be used to sign any intermediate certificates, which, in turn, will be used to sign your server certificates. Before you begin, consider these questions.

- How many intermediate certificates do you need? You might break them down by geography or department. Using intermediate certificates improves security. If one key is compromised, only the compromised chain is at risk. The rest of your network remains secure.
- How many Supervisor stations do you have?
- How many JACEs do you have?
- Can the Supervisor and JACE platform/stations use the default server certificate with the 1024-bit key pair or do you need to create a more secure key pair for each?

Creating a certificate chain of trust also involves setting up the Workbench, Supervisor and JACE Key and Trust Stores. What to import into each entity depends on the function of the entity. As noted in “NiagaraAX’s client/server architecture” on page 5-1, a given entity may serve as a client or a server. For an illustration of what goes in each store, see Chapter 5, “About the certificate creation and signing process”

Set up a folder structure

Before you begin, set up a folder structure on your computer’s hard disk with a separate folder for the root, intermediate and server certificates. For example, feel free to adapt the following:

- My_Company_Certificates
- Root_Certificate
- Intermediate_Certificate
- Server_Certificates

Establish a naming convention

The Key and Trust Stores form the heart of the SSL Toolset. The Alias field in the Key Store is a name you can use to differentiate your certificates and keys. Since certificates look a lot alike, use different aliases to differentiate the purpose of each certificate.

In addition to naming each alias, multiple passwords are required. For security’s sake your passwords need to be robust as well as easy to remember.

Set up the root and intermediate certificates

The procedures in this topic explain how to create a root certificate and a single intermediate certificate. The root certificate is a special case because it may be self-signed or signed by a third-party CA. If you are serving as your own CA, you create the root certificate, use it to sign your intermediate certificates and export it with only its public key for importing into each client Trust Store. Creating an intermediate certificate involves more steps.

The topic includes:
- “Create the root and intermediate certificates” on page 3-2
- “Create a CSR for the intermediate certificate” on page 3-4
- “Sign the intermediate certificate using the root certificate’s private key” on page 3-5
- “Import intermediate certificate back into the Key Store” on page 3-5
- “Export the root and intermediate certificates” on page 3-6

Caution

To ensure the security of your network, always perform these tasks using a computer that is disconnected from the internet and company network. It is recommended to maintain this computer in a secure physical location.

Create the root and intermediate certificates

The root certificate is the top of the certificate chain of trust. Its private key must be physically protected and should not be sent via email. The root certificate’s private key is used to sign each intermediate certificate. The root certificate’s public key is used to verify intermediate certificate authenticity.

Intermediate certificates are signed by the root certificate. Their private keys are used to sign the server certificate for each JACE. Their public keys are used to verify JACE server certificate authenticity.
The procedure for creating root and intermediate certificates is the same.

**Step 1** Open Workbench running on a stand-alone secure computer—one that is not connected to the JACE network, corporate intranet, or the internet.

**Step 2** Click **Tools > Certificate Management.** The Workbench **Certificate Management** view appears with the focus on the **Key Store** tab.

**Step 3** Check the title at the top of the **Certificate Management** view to ensure that you are viewing the Workbench **Key Store** and not a JACE **Key Store**. The Workbench and platform/station stores are separate.

![Figure 3-2 Workbench Key Store](image)

The Tridium certificate was automatically generated when you started Workbench.

**Step 4** Click **New**. The **Generate Self Signed Certificate** dialog appears.

![Figure 3-3 New certificate dialog](image)

**Step 5** Fill in the fields. **Alias** should identify root and intermediate certificates by company, and geography or department respectively. **Common Name (CN)** is the same as Distinguished Name and can be the same as the **Alias**.

For more information about each field, see “About the Generate Self-Signed Certificate dialog” on page 7-4.

**Step 6** Select the **CA Certificate** option for **Certificate Usage**

**Step 7** Click **OK**.
The system prompts you to create a password for the certificate's private key.

**Figure 3-4** Private Key Password dialog.

This password protects the private key and is required when using the certificate to sign other certificates. Choose robust passwords.

**Step 8** Type and confirm the private key password, and click **OK**.

**Note:** To ensure security, do not click **Skip** or the password that protects the private key will not be created.

**Step 9** To view the certificate, double-click it or select it and click **View**.

**Step 10** Confirm that the information is correct.

To change a certificate, you must delete it and create a new certificate.

**Create a CSR for the intermediate certificate**

Each intermediate certificate will be signed by the root certificate using the intermediate certificate's CSR. You don't need to create a CSR for the root certificate.

**Step 1** Select the intermediate certificate you just created, and click **Cert Request**.

**Step 2** The **Certificate Request Info** view appears.

**Figure 3-5** CSR information view

**Step 3** Confirm that the certificate properties are correct.

**Step 4** To save the CSR, click **OK**, select the folder for intermediate certificates and click **OK**.

The **Alias** for the certificate is used as the file name with the extension: .csr.

Repeat this procedure for each intermediate certificate.
Chapter 3 – Create certificates

Set up the root and intermediate certificates

Establish a naming convention

August 30, 2012

3.7

NiagaraAX SSL Connectivity Guide

Note: If an external Certificate Authority, such as VeriSign or Thawte, will sign your root certificate, follow the CSR submission procedure as required by the CA. They will verify that you are who you claim to be, that this certificate is for a server that your organization actually maintains, and other important information. They will then return the signed certificate to you.

Sign the intermediate certificate using the root certificate’s private key

This procedure uses the Workbench SSL tools and the root certificate you created to sign your intermediate certificates.

Step 1 In Workbench, select Tools > Certificate Signer Tool. The Certificate Signing dialog appears.

Figure 3-6 Certificate Signing dialog

Step 2 Click the browser icon, locate, and open a CSR for an intermediate certificate you created. The Certificate Signing dialog expands to show the certificate details.

Figure 3-7 Certificate Signing dialog

Step 3 Confirm that this is the intermediate certificate you created.

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

Step 5 Select the root certificate for CA Alias, type the root certificate’s private key password for CA Password, and click OK.

Signing is done by the private key of the root certificate, which is why the password you created for the root certificate’s private key is required.
Repeat this procedure for each intermediate CA certificate.

**Import intermediate certificate back into the Key Store**

**Note:** The certificate you import back into the Workbench Key Store must match the original Alias.

**Step 1** To view the Workbench Key Store click Tools > Certificate Management.

**Step 2** Click → Import, locate the certificate .pem file and click → Open.

**Step 3** Enter the intermediate certificate’s private key password 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.

The Certificate Import dialog appears.

**Step 5** Confirm that this is the certificate you expect and click OK.

The green shield icon 🟢 appears next to the certificate Alias in the Key Store.

**Note:** If your root certificate was signed by an external CA, you will need to follow this same procedure to import it back into the Workbench Key Store.

Repeat this procedure for each intermediate certificate.

**Export the root and intermediate certificates**

You export the root certificate so that you can import it along with the intermediate certificates into the Trust Store of the client and browser. For this purpose you do not need to export the intermediate certificates because a signed .pem file already exists for them.

You would export both your root and intermediate certificates to back them up with their private keys.

**Step 1** On the Key Store tab, select the root certificate and click → Export.

The system prompts for the private key password.

**Figure 3-8** Export private key password dialog

**Step 2** Do one of the following:

- To create a client certificate for importing into the client’s Trust Store and into the browser, click → Skip. This exports the certificate with only its public key.
- To create a backup of the certificate and its private key, type the password and click OK. This password is required each time a CA certificate (with its private key) is presented. CA certificates are used by human beings who can supply the password. Server certificates use the password that protects access to the Key Store. By enabling Reuse password for exported key, this password is saved to the file system. Otherwise, you would be prompted again to supply it.

**Step 3** Locate the root or intermediate certificate folder and click OK.

The system displays the Save As dialog with the Alias as the file name, and .pem as the extension for the signed certificate file.

**Step 4** Change the name or accept it as is and click Save.

The system reports that the export was successful.

**Figure 3-9** Export successful message

**Step 5** To complete the action, click OK.
If you are exporting for the purpose of importing into the client Trust Stores you only need to export the root certificate. Your intermediate certificate(s) already have signed .pem files ready to import.

Set up the JACE and Supervisor server certificates

For each JACE, follow these procedures using Workbench running on a computer that is connected by a crossover cable to the JACE.

For each Supervisor station, disconnect the Supervisor station from the internet and company network before following these procedures.

- “Create new JACE and Supervisor server certificates” on page 3-7
- “Create a CSR for each server certificate” on page 3-8

Create new JACE and Supervisor server certificates

You need a server certificate for each JACE and Supervisor in the network. All signed server certificates (with their public keys) reside in the Workbench, Supervisor, and other client Trust Stores.

There are multiple ways to create a server certificate.

- For each JACE, you can use the default server certificate that is automatically generated when you boot the JACE for the first time. However, this certificate provides only a 1024-bit key.
- If you are connected to a JACE using a crossover cable, you can use PlatformServices to create a new server certificate with a larger (more secure) key.
- Using Workbench on a secure computer, you can create a server certificate, sign the certificate and download it into the JACE, preferably using a crossover cable directly connected to the JACE.

This procedure creates a new 2048-bit server certificate on a JACE.

Step 1 Launch Workbench and connect to the JACE platform.

Step 2 Locate the JACE station in the Nav tree and click CertManagerService under PlatformServices.

The Certificate Management view appears with the focus on the Key Store tab.

Step 3 Check the title at the top of the Certificate Management view to ensure that you are viewing the JACE's Key Store and not the Workbench Key Store, then click New.

The Generate Self Signed Certificate dialog appears.

Figure 3-10 New certificate Distinguished Name fields

Step 4 Fill in the fields and click OK.

Common Name (CN) is the same as Distinguished Name and can be the same as the Alias. Follow these recommendations:

- Create a name that identifies the JACE. You might use the JACE's IP address or a location code. Do not use the same Common Name that you also used for the root or intermediate certificates.
- The Common Name should match the Subject and host name, which is how the server identifies itself. The IP address of the JACE or domain name may be an appropriate Alias and Common Name for a JACE.
For more information about each field, see "About the Generate Self-Signed Certificate dialog" on page 7-4.

**Note:** The option for **Certificate Usage** defaults to **Server Certificate** and **Key Size** defaults to **2048**. A larger key takes longer to generate, but provides more robust security.

A pop-up in the lower right corner indicates certificate creation success or failure, Workbench displays an information message, and adds the certificate to the **Key Store**.

**Figure 3-11  Information message**

![Image of information message]

Step 5  Click **OK** to the **Info** message.

The length of time it takes to generate the certificate depends on the key size and the platform. When finished, you will have a key pair (public and private keys).

Step 6  To view the certificate, double-click it or select it and click **View**.

Step 7  Confirm that the information is correct.

To change a certificate, you must delete it and create a new certificate.

### Create a CSR for each server certificate

For each server certificate to be signed by an intermediate certificate (or the root certificate if your installation is small), a Certificate Signing Request (CSR) is required. This procedure is the same if you are using the default server certificate or a server certificate that you created.

**Step 1**  While you are connected to the JACE and are viewing the station's **Key Store**, select the certificate and click **Certificate Request Info**.

The **Certificate Request Info** view appears.

**Step 2**  Confirm that certificate properties are correct.

**Step 3**  To save the CSR, click **OK**, select the folder for server certificates on your computer, and click **OK**.

The system uses the **Alias** as the certificate file name and the extension of: .csr. This file does not contain the server's unique private key.

**Step 4**  Copy the CSR to a flash drive or store it on the laptop Workbench computer for transport to the secure, standalone Workbench computer that contains the root and intermediate certificates for signing.

### Sign the server certificates

Each CSR on the flash drive or laptop Workbench computer needs to be signed by the private key of the root or an intermediate certificate. Although each server certificate CSR does not contain its private key, transportation to the Workbench computer on which the root and intermediate certificates are stored should be secure.

**Caution**  To ensure the security of your network, always sign certificates using a computer that is disconnected from the internet and company network. It is recommended to maintain this computer in a secure physical location.

See “Sign the server certificates using the intermediate certificates” on page 3-8.

### Sign the server certificates using the intermediate certificates

This procedure uses the Workbench tools and the intermediate certificates to sign server certificates.

**Step 1**  Select **Tools > Certificate Signer Tool**.

The **Certificate Signing** dialog appears.
Chapter 3 – Create certificates

August 30, 2012

Establish a naming convention

Chapter 3 – Create certificates

3.7 NiagaraAX SSL Connectivity Guide

Chapter 3 – Create certificates

3–9

Import the signed server certificate and configure each station

Establish a naming convention

Chapter 3 – Create certificates

3.7 NiagaraAX SSL Connectivity Guide

Chapter 3 – Create certificates

3–9

Figure 3-12 Certificate Signing dialog

Step 2 Click the browser icon, locate, and open the CSR for a server certificate. The Certificate Signing dialog expands to display the certificate details.

Figure 3-13 Certificate Signing dialog

Step 2 Click the browser icon, locate, and open the CSR for a server certificate. The Certificate Signing dialog expands to display the certificate details.

Step 3 Confirm that this is the correct server certificate.

Step 4 Select valid dates.

Step 5 Select the intermediate certificate that applies for CA Alias, type the intermediate certificate’s private key password for CA Password and click OK. Since the signing is done by the private key of the intermediate certificate, the password you created for the private key needs to be provided.

This generates a new certificate file with the extension of: .pem. This file contains only the public key associated with the certificate.

Repeat this procedure for each server certificate.

Import the signed server certificate and configure each station

When signing is complete, the .pem files for the server certificates need to be transported (on a flash/thumb drive or laptop computer) back to where they came from. Even though these files do not contain their private keys, you should transport them securely between locations.

3.7 NiagaraAX SSL Connectivity Guide

3–9
Two tasks remain to set up SSL on a JACE or Supervisor:
- “Import the server certificate into the Key Store” on page 3-10
- “Select the platform server certificate” on page 3-10
- “Select the Https and Foxs service certificates” on page 3-10

**Import the server certificate into the Key Store**

The JACE platform and station share the same Key and Trust stores. This procedure demonstrates how to use station **PlatformServices** to import a signed server certificate back into each platform **Key Store**.

**Note:** *The certificate you import back into the platform must match the originalAlias.*

**Step 1** Connect to the platform.

**Step 2** In Workbench, click **Station > Config > Services > PlatformServices > CertManagerServices**. **Certificate Management** opens with the focus on the **Key Store**.

**Step 3** Click , locate the certificate .pem file and click . **Certificate Import** displays the certificate details.

**Step 4** Enter the server certificate's private key password 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.

The **Certificate Import** dialog appears.

**Step 5** Confirm that this is the certificate you expect and click **OK**. The green shield icon  appears next to the certificate Alias in the **Key Store**.

**Select the platform server certificate**

**Step 1** Connect to the platform.

**Step 2** Click **Platform** and double-click **Platform Administration**.

**Step 3** Click . The **Platform SSL Settings** dialog appears.

*Figure 3-14 Platform SSL Settings dialog*

**Step 4** Change the **Certificate** and click **Save**.

**Select the Https and Foxs service certificates**

**Step 1** Connect to the station.

**Step 2** Select the property sheet for the service in the Nav tree:
- For the Fox service select **Station > Config > Drivers > Niagara Network**.
- For HTTP service select **Station > Config > Web Service**.

The property sheet appears.
Set up the Trust Stores

Each JACE and Supervisor can also serve as a client. Workbench and the browser are always clients. This topic explains how to set up the Trust Stores.

- "Set up the platform/station Trust Stores" on page 3-11
- "Set up the Workbench Trust Store" on page 3-12
- "Install certificates in a client browser" on page 3-12.

In addition to the above Trust Stores, you also need to import the certificates into the Java Trust Store (Java Control Panel > Security > Certificates). This is necessary for foxs connections to be trusted.

Caution

If your only recourse is to email a root certificate, use a heavily encrypted ZIP file and communicate the ZIP file password over the phone. Otherwise, the receiver of the ZIP file has no way to verify that the ZIP contains the actual certificate. You do not want Bart to be able to replace the root certificate that signed Bob’s credentials with his own root certificate.

If you are installing a brand new network, certificate transfer can be done in the shop during initial commissioning of a group of JACEs that will be later installed on site (commission the JACEs first and then install the certificates).

Set up the platform/station Trust Stores

Step 1 To view the JACE or Supervisor station’s Trust Store click PlatformServices > CertManagerService in the Nav tree.
Step 2 Click the Trust Store tab.
Step 3 Click and locate the certificate .pem file and click Open. Certificate Import displays the certificate details.
Step 4 Confirm that this is the certificate you expect and click OK.

The certificate appears in the Trust Store. All servers that have server certificates signed by the private key associated with this certificate will be trusted automatically.

Follow this procedure for each certificate to be trusted by a client platform/station.
Set up the Workbench Trust Store

Step 1 To view the Workbench Trust Store click Tools > Certificate Management and click the Trust Store tab.
Step 2 Click Certificate Import and locate the certificate .pem file and click Open. Certificate Import displays the certificate details.
Step 3 Confirm that this is the certificate you expect and click OK. The certificate appears in the Trust Store. All servers that have server certificates signed by the private key associated with this certificate will be trusted automatically. Follow this procedure for each certificate to be trusted by Workbench.

Install certificates in a client browser

Installing your signed client certificate in the browser ensures that security will be enabled automatically.

• “View the certificates in Internet Explorer 8” on page 3-12
• “Install a certificate in Internet Explorer 8” on page 3-12
• “Install a certificate in Firefox 10” on page 3-12
• “Install a certificate in Google Chrome 17” on page 3-12

View the certificates in Internet Explorer 8

Each browser maintains the list of approved certificates. How to view this list varies with the browser. To confirm that the NiagaraAX certificate for https is stored in Internet Explorer 8’s approved list:

Step 1 Launch Internet Explorer and click Tools > Internet Options.
Step 2 On the Content tab click the Certificates button.
Step 3 On the Certificates tab click the Intermediate Certification Authorities tab and look for the name of your intermediate CA certificate in the list.

Install a certificate in Internet Explorer 8

Step 1 In Internet Explorer 8.0, click Tools > Internet Options > Content.
Step 2 In the Certificates section, click Certificates > Intermediate Certificate Authorities > Import and follow the wizard.

Install a certificate in Firefox 10

Step 1 Using the Firefox menu in the upper left corner of the page, click Options > Options > Advanced.
Step 2 Click the Encryption tab.
Step 3 Click View Certificates > Import.
Step 4 Locate the certificate file and click Open.

Install a certificate in Google Chrome 17

Step 1 Click the wrench icon in the upper right corner of the page.
Step 2 Click Options > Under the Hood.
Step 3 In the HTTPS/SSL section, click Manage certificates....
Step 4 Click Import and follow the wizard.

Updating a certificate

As a general rule, third-party certificates are not changed. Some CAs will not allow any changes once the certificate is signed. If a change needs to be made, the original certificate is deleted and a new one created.

Follow these procedures:

• “Delete a certificate” on page 3-12

Delete a certificate

Step 1 Open the platform.
Step 2 Click Tools > Certificate Management.
Step 3 Select the certificate in the Key Store and click Delete.
Back up the stores

To back up the Key and Trust Stores, select the certificates and export them to a secure location on your computer's hard disk or on a flash/thumb drive.

- “To back up the stores” on page 3-13

To back up the stores

Step 1 Access both sets of stores:
- To access the Workbench stores, click Tools > Certificate Management.
- To access the platform stores through the Nav tree use either Platform > Certificate Management, or Station > Config > Services > PlatformServices > CertManagerServices.

Step 2 Select each certificate and click Export. You export each certificate one at a time.

Step 3 When exporting from a Key Store, export the private key along with the certificate, creating a password for each private key.

Step 4 Store the off-line storage medium (for example, a thumb drive) in a safe place.

Managing allowed hosts

If you used the self-signed certificates to provide initial encryption and decryption, more than one exception may be allowed in your Allowed Hosts list. Once you have set up signed certificates for all hosts and clients, it is a good idea to delete the exceptions from the Allowed Hosts lists both for Workbench and for the platform/station.

Test station health

When you finish configuring a server, stop and restart the station and check station health.
About SSL—Alice, Bob, Cathy and Bart

This topic introduces basic security concepts:

- “What makes any system secure?” on page 4-1
- “Friends and enemies” on page 4-1
- “Identity verification” on page 4-3
- “Cryptography” on page 4-8

What makes any system secure?

A secure system requires:

- Physical security: Your JACE network should be located in a guarded location with appropriate locks, security systems, and physical access control.
- Protection against hacking: Your computers need firewalls, passwords and other software access controls.
- Verification of authenticity: Certificates verify that the contacted server is the expected server. This verification of authenticity thwarts imposters who attempt to steal trust for the purpose of disrupting network operations.
- Data transmission security: Data encryption and decryption defeat hackers who would use listening devices to capture transmitting data.

The third and fourth bullets are the security requirements configured by the SSL Toolset that is part of NiagaraAX 3.7.

Friends and enemies

Alice, Bob and Cathy are remote friends. Bart is an imposter. This topic explains how internet security systems work.

- “When trust is secure” on page 4-1
- “When things go wrong” on page 4-2
- “Verifying authenticity” on page 4-2

When trust is secure

Alice connects to Bob remotely using the internet. For example, Bob may run a commercial website from which Alice wants to purchase goods or services.

Figure 4-1 Communication in a trusted world

In Figure 4-1, Alice and Bob are both authentic and legitimate. Communication between them is safe.
When things go wrong

In Figure 4-2, Bart is pretending to be Bob so he can steal Alice’s identity and confidence.

Figure 4-2  When things go wrong

```
Hello, are you Bob?
```

Alice

```
I'm Bob
```

Bart

How does Alice know that the person claiming to be “Bob” is really Bart impersonating Bob?

Figure 4-3  How the bad get caught

```
Hello, are you Bob?
```

Alice

```
Don't bet on it!
```

Cathy

```
I'm Bob
```

Bart

The answer is that she and Bob have a mutual friend, Cathy, who authenticates Bob’s credentials and warns Alice when the person to whom she is connected is not Bob.

Verifying authenticity

In reality, there is no person (or computer) on the internet whose job it is to monitor each client/server transaction. So, who or what is Cathy and how did Cathy know to warn Alice?

Figure 4-4  Cathy, a third-party Certificate Authority signs Bob’s credentials

```
Hi, will you sign my certificate of authentication?
```

Bob

```
Yes, Bob, I know who you are.
```

Cathy

Cathy is a file known as a certificate of authentication owned and distributed by a Certificate Authority (CA). Bob also is a certificate of authentication. Ahead of time, Bob’s company sent its certificate to Cathy’s company, which verified Bob’s company identity and signed his certificate.

When Alice installed her browser, Cathy’s certificate was installed in Alice’s browser’s Trust Store. As the name implies, a Trust Store contains certificates from trusted entities.
Chapter 4 – About SSL—Alice, Bob, Cathy and Bart

August 30, 2012

Identity verification

More about certificates

### Figure 4-5  Secure communication: the signatures match

As soon as Alice contacts Bob, he sends her his certificate. Alice’s browser checks the signature on Bob’s certificate against the signature on Cathy’s certificate in its Trust Store. The signatures match and Alice’s browser authorizes the beginning of a trusted connection between Alice and Bob.

### Figure 4-6  Rejected communication: the signatures do not match

Alice’s browser immediately rejects Bart’s certificate because its signature does not match the signature on Cathy’s certificate in Alice’s browser’s Trust Store.

### Identity verification

Public Key Infrastructure (PKI) is the name of the technology that employs two security processes to ensure secure communication:

- Identity verification, which is described in these topics:
  - “More about certificates” on page 4-3
  - “Keys” on page 4-4
  - “Signing a certificate with a private key” on page 4-4
  - “Creating a chain of trust” on page 4-6
  - “More about intermediate certificates” on page 4-8
  - “Cryptography” on page 4-8, which is the general term for scrambling and unscrambling data to ensure it is not intercepted during transmission.

### More about certificates

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 **chain of trust**. The example of Alice, Bob, Cathy and Bart involves at least three certificates:

- During the handshake with Alice, Bob presents his **server certificate**.
- The Trust Store in Alice’s browser contains a copy of the **root certificate** that Cathy used to sign her own and Bob’s server certificates. The successful matching of the signature on Bob’s server certificate with the signature on Cathy’s root certificate allows communication to begin.
- Bart also has a signed server certificate, but it was not signed by Cathy, therefore, his attempt to impersonate Bob is not trusted. Each certificate contains metadata that identifies the certificate owner and the purpose of the certificate. **Figure 4-7** shows a certificate as it appears in Windows 7.
Figure 4-7  Certificate with metadata

The **General** tab identifies to whom the certificate was issued (axlicensing.tridium.com), who the trusted Certificate Authority (CA) was that issued the certificate (VeriSign), and for how long the certificate is valid (until 7/24/2013). It is typical for certificates to be valid for a year or two. It is unusual for a certificate to be valid indefinitely.

The **Details** tab provides more information, including the Subject, which is also known as the Common Name (CN). In addition to signatures matching, Server and client CNs must match for secure communication to begin.

**Keys**

A pair of asymmetric keys (one public and the other private) makes SSL authenticity verification and encryption/decryption possible. The term “asymmetric” means that each key is unique but they match each other. The signing of certificates with the private key is required to verify authenticity. Both keys are required to encrypt/decrypt information. In advance, key generation software running on a stand-alone computer generates this pair of asymmetric keys.

Figure 4-8  Asymmetric keys

- A **public key** is a string of bytes wrapped by a certificate. A root certificate with its public key is installed in the browser’s Trust Store either when the browser is installed, or it can be added to the Trust Store later. If sent by email, a root certificate must be password protected. It is usually part of an installation or disk imaging process.
- A **private key** is also a string of bytes, but is not wrapped by a certificate. The private key resides on the authentic server. Private keys must be physically protected for a chain of trust to remain secure. They must not be sent via email, and, if necessary, should be physically transported (on a thumb drive or other medium that is not connected to the internet) from the computer that generated them to the server.

**Signing a certificate with a private key**

Cathy’s company, a **Certificate Authority (CA)**, verified Bob’s identity and signed his server certificate with Cathy’s private key. Here’s how it happened:
1. Bob created a pair of asymmetric keys and a certificate that contained his credentials (his name, address, etc.).

**Figure 4-9  Bob's keys and self-signed server certificate**

Bob’s server certificate was not yet signed by a CA, and at this stage is considered to be *self-signed* using his own private key. Notice that the **Issuer** and **Subject** are the same.

2. Bob sent this certificate to Cathy with a request that she verify his identity. (He probably also sent money with his request.) He did not send Cathy his private key.

3. As a CA, Cathy owns a pair of keys and a trusted root certificate.

**Figure 4-10  Cathy's keys and root certificate**

Cathy’s root certificate is also a self-signed certificate. It serves as the top of the chain of trust. Cathy stores it in a vault along with her private key.
4. After thoroughly checking Bob’s credentials, Cathy extracted Bob’s public key and metadata from his self-signed certificate and created a new certificate with her name as the **Issuer**. Notice how the **Issuer** and **Subject** are different from the self-signed certificate that Bob sent Cathy.

5. Cathy then used her **root private key** to sign this new certificate.

6. Cathy compressed both the new server certificate and a copy of her root certificate (Figure 4-10) with password protection, put both on a website and emailed the links to Bob.

7. Then she phoned Bob and gave him the password for the two compressed, password-protected files.

8. Bob expanded the files and imported his signed server certificate into the **Key Store** on his web server. This action replaced his self-signed certificate.

9. There are several ways for Alice to install Cathy’s root certificate in her browser:
   - Cathy’s root certificate may have been installed when Alice installed her browser.
   - Since Alice is a customer of Bob’s, he may have created an installation program, which Alice downloaded and installed in her browser.
   - If Alice is a JACE controller, Bob may have installed Cathy’s root certificate in the office, when he set up each JACE before taken them out to the field.

**Creating a chain of trust**

A certificate chain is a structure of trust with a root certificate at the top-most level. A root certificate is made trustworthy by securing its physical distribution.
Figure 4-13  A chain of trust with a Root Certificate Authority (CA) certificate at the top.

The certificate chain of trust is also known as the **certification path**. At the top of the chain is the CA root certificate (the one that has been self-signed by a CA and stored in the Trust Store of the client’s browser).

The arrows on the right of the drawing show how the chain of trust was established using the private keys to sign the certificates at the next level down in the chain. All certificates immediately below the root certificate inherit the trustworthiness of the root certificate. Certificates further down the chain depend on the trustworthiness of the intermediate certificate(s).

The server certificate’s private key is not part of establishing the chain of trust. It is used to encrypt and decrypt data after identity is established.
The arrows on the left side of the drawing show how identity is verified during the handshake. Working up the chain, the server sends the client browser its server and intermediate certificates. The client browser:

1. Verifies that the server certificate was signed by the intermediate certificate and that the Distinguished Names match.
2. Verifies that the intermediate certificate was signed by the root certificate, which it has in its Trust Store.

Assuming all signatures match, communication begins.

**Note:** Certificates have an expiration date. Every couple of years they must be renewed.

**More about intermediate certificates**

Any number of intermediate certificates may be set up in the chain of trust. A certain amount of overhead is carried to verify each certificate during the handshake. The benefit of multiple intermediate certificates is that it reduces the impact of a security compromise.

**Cryptography**

Once Bob’s identity is confirmed during the handshake with Alice, they no longer worry about confirming identities. Bob’s private key takes over the encryption and decryption task. This prevents eavesdropping on their “conversation.”

Public Key Infrastructure (PKI) encrypts and decrypts in two steps:

- At the start of communication, the handshake receives extra protection. See “Encrypting and decrypting the handshake” on page 4-8
- Once communication is established, encrypted data transmission begins. See “Data transmission after the handshake” on page 4-8
- Key size is directly related to security. See “More about key size” on page 4-9
- NiagaraAX combines identity verification with encryption and decryption.

**Encrypting and decrypting the handshake**

Encryption using both keys protects the exchange of the identity-verifying certificates.

*Figure 4-14* The handshake uses asymmetric keys to encrypt and decrypt the exchange of certificates

The private key on the server side encrypts the opening handshake, and the client’s matching public key decrypts it. This action (called asymmetric cryptography) protects the exchange of certificates used to establish identity.

- The advantage of using asymmetric keys, is that they can be larger (more secure) than symmetric keys (the same key used at both ends).
- The disadvantage of using asymmetric keys is that their size and complexity make encryption and decryption slow. This is why they are used only to establish the connection.

**Data transmission after the handshake**

Once server identity has been established using asymmetric cryptography, the transmission continues with a negotiated symmetric key that both encrypts and decrypts the data (symmetric cryptography).

*Figure 4-15* A single key speeds symmetric encryption/decryption
• The advantage of symmetric cryptography is that it is simpler and faster than asymmetric cryptography.

• The disadvantage of using a single key for symmetric encryption/decryption is that there is no easy way to share the key (it should not be sent via email).

**More about key size**

The size of the key is directly related to the quality of security it offers. Bigger keys are more secure. Your options for key size are:

• 1024, the size of the default key generated at JACE, Workbench and Supervisor at start-up.

• 2048, the default if you generate a key manually. At the time of issuing this guide, this size is required by most Certificate Authorities.

• 3072

• 4096

To generate a 1024-bit key on a JACE takes about 15 seconds. To generate a 4096-bit key on a JACE can take a while. The reason is that key generation is math-intensive (it randomly generates prime numbers that meet certain criteria). Ideally, this type of security should be installed in person. If you are sitting next to each JACE with your laptop directly connected to the JACE, you can safely generate a complex key on your laptop and import it into the JACE.
This topic describes how SSL security works in a NiagaraAX network.

- “NiagaraAX’s client/server architecture” on page 5-1
- “About the certificate creation and signing process” on page 5-2
- “About NiagaraAX SSL certificates” on page 5-7
- “About the SSL Toolset” on page 5-8
- “NiagaraAX SSL best practices” on page 5-13

NiagaraAX’s client/server architecture

The typical internet client/server relationship with a third-party verifying the server becomes a little more complicated when considering the relationships between the various programs (processes) within the NiagaraAX Framework.

NiagaraAX’s SSL implementation is platform based. All SSL Toolset functions can be configured using only a platform connection and without any station running.

Three NiagaraAX programs (processes) require communication protection:

- Fox Service, a proprietary program that is used for all network communication between stations as well as between the Workbench and stations.
- Web Service for the station. This service is used to download the Workbench applet (wbapplet) and the modules that are required for running the station in the browser. It is also used to display hx pages, mobile pages, etc.
- Platform for Niagarad, which provides HTTP connectivity to a platform.

For more information about how the NiagaraAX programs provide NiagaraAX functionality, see Types of NiagaraAX Programs in the Networking and IT Guide.
Figure 5-1 illustrates the possible client/server relationships within the NiagaraAX Framework:

- Workbench (client) logging in to a JACE (server) using Niagarad (platform connection).
- JACE station (client) connecting to a Supervisor station (server) using Fox Service.
- Vice versa: Supervisor station (client) connecting to a JACE station (server) using Fox Service.
- JACE station (client) logging in to another JACE station (server) using Fox Service.
- Web browser (client) connecting to the station (server) via HTTP.
- Station (client) connecting to the remote Supervisor (server) over the internet (HTTP).
- Vice versa: Supervisor station (client connecting to a station (client) over the internet (HTTP).

Note: The JACE B in figure Figure 5-1 has all the same connections to Workbench and the Supervisor as does JACE A. For simplicity the illustration shows only the peer-to-peer connection between it and JACE A.

About the certificate creation and signing process

The illustrations in this section are intended to help you visualize what needs to be done. The actual steps you will take may vary. For example, to save time it makes sense to do everything in Workbench at one time rather than go back and forth between the Workbench and platform/station tools.

- "Certificate creation" on page 5-2
- "Workbench certificate signing" on page 5-3
- "JACE certificate signing" on page 5-4
- "Supervisor certificate signing" on page 5-5
- "Setting up the client Trust Stores" on page 5-5

Certificate creation

To set up a chain of trust you begin by creating a root CA certificate and a server certificate for each JACE and Supervisor. You may also require several intermediate CA certificates.
1. If you use intermediate certificates, you probably have more than one. The illustrations show only one `IntermCA Cert` for the sake of simplicity.

The Workbench stores are separate from each platform/station stores. You access the Workbench SSL tools by using a menu option. Click `Tools > Certificate Management` for the procedure, see “Create the root and intermediate certificates” on page 3-2.

2. You access the platform/JACE stores by double-clicking `CertManagerService` under `Platform Services` in the station Nav tree.

For the procedure, see “Create new JACE and Supervisor server certificates” on page 3-7.

3. The `tridium` certificates you see in the Supervisor and JACE Key Stores are the default 1024-bit certificates that are created at initial station startup. Although the certificates are all named “tridium,” each is unique to the platform on which it was created. In their unsigned, default state these certificates do not provide server authentication, but they do provide encryption and decryption while you are setting up your certificates.

You could sign these as server certificates and use them for your Supervisors and JACES. This example, however, demonstrates creating server certificates for a Supervisor and JACE using 2048-bit keys, which are more secure than the 1024-bit keys of the default certificates.

To save space, the remainder of the illustrations do not show the `tridium` server certificate. After the new certificates are signed and imported, you will select them for each station. At that time you can delete the `tridium` certificate.

**Workbench certificate signing**

Using the SSL Toolset the intermediate certificates are signed by the private key of the root certificate and prepared for import into the client Trust Stores.

---

**Figure 5-2** Creating the CA and server certificates

![Diagram showing steps to create CA and server certificates](image)

**Figure 5-3** Setting up the Workbench certificates

![Diagram showing steps to set up Workbench certificates](image)
JACE certificate signing

1. Notice that in the Workbench Key Store the root certificate always shows the caution shield . As the highest authority in the chain of trust, this certificate must be self-signed. For this reason, the root certificate must be physically protected for the security system to provide any protection to the network.

   For the procedure, see “Create a CSR for the intermediate certificate” on page 3-4

2. The certificate signing tool uses the private key of the root certificate to sign the intermediate certificate.

   For the procedure, see “Sign the intermediate certificate using the root certificate’s private key” on page 3-5

3. This step marries the signed certificate with its private key, which never left the Key Store. The green shield  indicates that the intermediate CA certificate has been signed.

4. The intermediate certificate (IntermCACert) does not need to be exported because its .pem file (without its private key) was created when it was signed.

   For the procedure, see “Export the root and intermediate certificates” on page 3-6

JACE certificate signing

Each JACE has its own separate and unique platform/station stores.

Figure 5-4  Signing the JACE server certificate

1. Creating the CSR prepares the certificate for signing without its private key.

   For the procedure, see “Create a CSR for each server certificate” on page 3-8

2. Signing the server certificate is done using the private key of the root certificate.

   For the procedure, see “Sign the server certificates using the intermediate certificates” on page 3-8

3. This step marries the signed certificate with its private key, which never left the Key Store. The green shield  indicates that the intermediate CA certificate has been signed.
Supervisor certificate signing

Figure 5-5  Signing the Supervisor certificate

The Supervisor procedure is the same as that for the JACE.
1. Creating the CSR prepares the certificate for signing without its private key.
   For the procedure, see “Create a CSR for each server certificate” on page 3-8
2. Signing the server certificate is done using the private key of the root certificate.
3. Importing the signed certificate back into the Key Store changes the caution shield to the green shield.

Setting up the client Trust Stores

After creating and signing the certificates, the final step is to import them into the client Trust Stores.
In addition to importing them into the Workbench, JACE and Supervisor Trust Stores, you must also import them into your browser Trust Store.
1. When Workbench functions as a client to a JACE or Supervisor, it checks its Trust Store for the relevant, signed certificates.

2. A JACE can be a client to the Supervisor. It can also be the client of another JACE. In this case, the signed server certificate for the server JACE would appear in this JACE’s Trust Store (not illustrated).

3. A Supervisor can be a client to another Supervisor or to a JACE.

For the procedure, see “Set up the platform/station Trust Stores” on page 3-11
About NiagaraAX SSL certificates

NiagaraAX certificates are presented in an easy-to-read format.

Figure 5-7  An example of the default Tridium certificate

A certificate contains a public key, a signature and metadata including: subject, issuer, valid date ranges, the algorithm and key size, and the purpose of the certificate (key usage).

- “Types of NiagaraAX certificates” on page 5-7
- “About self-signed certificates” on page 5-7

Types of NiagaraAX certificates

Certificates serve a variety of purposes depending on how the certificate’s Key Usage field is configured. To reduce complexity, NiagaraAX 3.7 arbitrarily manages these types of certificates:

- A server certificate resides with its matching private key in the Key Store on the server (JACE or Supervisor). No password is required to use this certificate.
- The private key of a CA certificate is used to sign other certificates. The private key requires the creation of a password on export and the provision of a password on import. A signed root CA certificate exported with only its public key serves as a client certificate in the Workbench and station Trust Stores of each client.

About 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 that is owned by a CA. This type of certificate cannot be validated by a signed client certificate in a client’s Trust Store and is not recommended for robust security. There is no procedure for self-signing a certificate. Each is created self-signed.

Two self-signed certificates are used in a JACE network:

- A default self-signed certificate for each JACE: When a JACE starts up for the first time, it creates its own unique, self-signed certificate, the primary purpose of which is to provide immediate encryption and decryption. The first time a client connects to the JACE, the software displays a message indicating that the certificate is not trusted. Whether you approve or disapprove the certificate, the software lists it in the Allowed Hosts list. If approved, the host is identified in the future as trusted and you will not have to approve the connection each time it is made. This form of security may be acceptable for small private networks (one or two JACEs). Self-signed certificates are not secure for connections made over the internet.
If you intend to use self-signed certificates, before you access the JACE from Workbench for the first time, make sure that your PC and the JACE are not on any corporate network or the internet. Once disconnected, connect the PC directly to the JACE, access the JACE from Workbench, and approve its self-signed certificate. Only then should you reconnect the JACE to the corporate network or internet.

If the IP address of the server changes, the self-signed certificate loses its trustworthiness as an approved host in the **Allowed Hosts** list.

- **Root certificate**: The entity that creates a self-signed certificate is implicitly trusted. The root certificate at the top of a certificate chain of trust is a self-signed certificate because there is no higher authority than the entity that created this certificate. For this reason, Certificate Authorities, whose business it is to endorse other people’s certificates, closely guard their root certificate(s).

**About the SSL Toolset**

Two sets of certificates are maintained by the SSL Toolset.

- Workbench, which always functions as a client, has its own set of certificates and keys that are separate from the platform/station certificates and keys.

**Figure 5-8** Workbench SSL tools

You manage these using Workbench’s **Tools > Certificate Management**, as well as **Tools > Certificate Signer Tool**.

- Through **PlatformServices**, each Supervisor and JACE station shares the same set of certificates and keys with its platform.

**Figure 5-9** Two ways to access the platform/station SSL tools

You manage these with either:
• A platform connection to the Niagara host (Certificate Management view), or
• A station (Fox) connection to the Niagara host, under its Services > PlatformServices > CertManagerService view. This is the same view as that accessed from the platform connection.

Whether you are accessing the Workbench or platform certificates and keys, the interface is essentially the same. Certificates and keys appear as rows in one of three tables:

• The Key Store list (table), see "About the Key Stores" on page 5-9
• The Trust Store list (table), see "About the Trust Stores" on page 5-10
• The Allowed Hosts list (table), see "About Allowed Hosts" on page 5-10

This topic continues with these sub-topics:

• "About the default certificate" on page 5-11
• "Workbench keys and certificates" on page 5-11
• "Platform keys and certificates" on page 5-11

### About the Key Stores

#### Figure 5-10   Example of a Key Store list

The Key Store in Workbench, and in the JACE and Supervisor stations contains one or more server certificate(s), each with its pair of private and public cryptographic keys. The default server certificate (tridium) has the same name in each key store, however, its keys are unique for each instance.

A key is added to this table automatically (when you start Workbench, a JACE or Supervisor), and manually when you import or create a new server certificate.

Double-clicking the certificate row in the table allows you to view certificate details.

**Note:** Once created, you cannot edit a certificate. To correct an error you must delete the certificate and start again. This is important for security.

### Key Store buttons

- **View** displays certificate details for the selected certificate.
- **New** opens the Generate Self Signed Certificate dialog, which is used to create CA and server certificates.
- **Cert Request** opens a Certificate Request dialog, which is used to create a Certificate Signing Request (CSR).
- **Delete** removes the certificate from the Key Store.
- **Import** adds the certificate (.pem file) to the Key Store.
- **Export** saves a copy of the selected certificate to the hard disk. The file extension is .pem.
- **Reset** deletes all keys in the Key Store and creates a new default key pair and certificate. It does not matter which certificate is selected when you click Reset. Reset does not reboot the JACE.
The Reset button facilitates creating a new key pair (private and public keys) for the entity, but may have unintended consequences if you delete valid certificates. Export all certificates before you reset.

**About the Trust Stores**

**Figure 5-11** Example of a Trust Store list

The Trust Store contains signed and trusted CA (client) certificates with their public keys. The Trust Store contains no private keys. Trust Store CA certificates are used by the client Workbench or station/platform to validate the digital signature on the server certificate to which the client is connecting. If the certificate names and signatures match, communication begins.

You add certificates to Workbench's Trust Store or a station's Trust Store by importing them.

**About Allowed Hosts**

**Figure 5-12** Example of Allowed Hosts list

The Allowed Hosts list contains security exceptions. These are hosts that submitted server certificates (during the handshake), which could not be validated (the private key used to sign the server certificate does not match the public key of a root or intermediate certificate in the Trust Store).

Regardless of the response to the message (Allow or Reject), the host that sent the unmatched server certificate is listed in the Allowed Hosts list.

For unapproved hosts, the browser or Workbench challenges server identity at startup 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.

**Note:** Host identity includes the IP address and port number. Port numbers are different for secure platform and station connections, thus you can have two certificates for the same IP host. If the IP address of an approved host changes, and no matching certificate exists in the Trust Store, the browser again challenges server identity at startup and enters a “new” host in the Allowed Hosts list.

You should only approve exceptions if you are sure of the identity of the host and know why the matching certificate was not found in the Trust Store.
About the default certificate

When you start up a JACE, Supervisor or Workbench for the first time, each automatically generates a default 1024-bit pair of keys, surrounds the public key with a self-signed certificate, and places the public key in its Key Store. This takes about 15 seconds.

Unless you replace it with a certificate that you create, this certificate is used to encrypt and decrypt data. Each platform has its own default certificate and key pair stored in its own Key Store. You should not copy this certificate (or a certificate you create for this server) from one platform to another.

Self-signed certificates are inherently exposed because they cannot authenticate the server. If you are going to use them, all your platforms should be contained in a secure private network, off line, and not publicly accessible from the internet.

Caution

Self-signed certificates provide data encryption and decryption. They do not validate the identity of the server. For this reason they are not recommended as a complete security solution.

Workbench keys and certificates

Workbench functions purely as a client with its own stores and Allowed Hosts list. This topic contains these procedures:

- "View the Workbench stores" on page 5-11

View the Workbench stores

Step 1 Launch Workbench.

Step 2 Click the Tools menu.

Figure 5-13 Workbench SSL tools

Two menu items manage the Workbench keys and certificates.

- Certificate Management is used to create certificates, create Certificate Signing Requests (CSRs), and to import and export keys and certificates to the Workbench stores.
- Certificate Signer Tool is used to sign certificates.

Step 3 To view Workbench’s client stores, click Certificate Management.

The Certificate Management view displays the three lists: Key Store, Trust Store and Allowed Hosts.

Platform keys and certificates

The same Key and Trust Stores, which are separate from the Workbench client stores, serve both the platform and station.

- "Access platform stores using a platform connection” on page 5-11
- "Access the platform stores from a station” on page 5-12
- "View the platform's default certificate” on page 5-12

Access platform stores using a platform connection

Step 1 Open a platform connection to the JACE (right-click My Host > Open Platform).

Step 2 Double-click Certificate Management in the Nav Container View.

The stores and Allowed Hosts lists appear.
**Access the platform stores from a station**

Step 1: If the platform is not already connected, open a platform connection to the JACE.

Step 2: If it is not already running, start the station.

Step 3: In the **Nav Container View**, click **Station > Config > Services > PlatformServices > CertManagerService**. The stores and Allowed Hosts lists appear.

**View the platform’s default certificate**

Step 1: After accessing the platform stores, double-click the default certificate (**tridium**) in the **Key Store**.
The Certificate Info view appears.

**Figure 5-16  Certificate Info view**

![Certificate Info view](image)

Step 2  Notice the **Issuer DN** and **Subject DN** properties.

The **Issuer DN** (Distinguished Name) is logically the same as the **Subject DN**. This indicates that this certificate is signed with its own private key, resulting in a self-signed certificate. Because it is self-signed, the client **Trust Store** does not contain a certificate with a public key that matches this self-signed certificate's signature.

Certificates that are automatically generated when you start Workbench and a JACE for the first time are self-signed certificates. The root certificate in a certificate chain of trust is also a self-signed certificate.

The **Issuer DN** and **Subject DN** are different for a certificate signed by a Certificate Authority (CA).

**NiagaraAX SSL best practices**

Consider these topics before you design your security system:

-  “How many keys and certificates do I need?” on page 5-13
-  “Who should sign my certificates?” on page 5-13

**How many keys and certificates do I need?**

Each JACE within a Niagara network requires its own unique private key and signed server certificate, which is stored in its **Key Store**.

Each client within a Niagara network requires a copy of the root certificate, intermediate certificate and server certificates that it needs to communicate with supervisors and other clients in the network. These certificates reside in the client **Trust Store**.

One certificate may be used to secure Fox station communication, Web Service and Niagarad or each service may have its own server certificate and private key.

The number of certificates you need depends on the number of platforms in your network and the type of communication you need to secure.

For more on how many certificates you may need, see “Design the certificate chain of trust” on page 3-2

**Who should sign my certificates?**

All certificates are signed in one of these ways:

- For a self-signed certificate, the certificate's private key is used to sign itself. This type of signature is not recommended because trust cannot be verified.

  **Note:** A self-signed certificate provides only data encryption and decryption.
• If your network is self-contained, you can serve as your own Certificate Authority (CA). When a company serves as its own CA, a certificate signed by the private key associated with one of its CA certificates must be installed in the client browser, and imported into the client platform/station’s Trust Store.

• If your network is exposed to the internet, a third-party CA provides the most secure communication, however at a cost. Third-party companies that provide certificate signing services include VeriSign® and Thawte. Certificates that contain only a public key and are signed by the third party are distributed with a user’s browser. NiagaraAX Workbench comes with a number of these certificates. No separate step is required to install them, but you must trust that the browser installation was secure. See “Install certificates in a client browser” on page 3-12).

You have your server certificates signed by creating a Certificate Signing Request (CSR) that contains your subject (Distinguished Name) and your public key. The process of generating a CSR also creates the private key, which must be kept secure. You send the CSR to the Certificate Authority.

⚠️ Warning Do not send your private key to the CA and do not distribute it via email.

When the CA receives the CSR file, it extracts certain information from the CSR, verifies your identity, creates a new certificate with itself as the Issuer and signs the certificate with its root or an intermediate private key. The CA returns to you the signed client certificate(s) (root and intermediate).
Troubleshooting

- “Fix error conditions” on page 6-1
- “Reset or replace a JACE securely” on page 6-2

Fix error conditions

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

Most likely there is something wrong with the certificate. If a certificate fails, or for any reason SSL does not enable, rather than lock you out of the platform, the system enables a connection without security.

Reset the JACE.

Note: If you have to reset a JACE, assuming you exported the keys, you can import them to the reset JACE. If you have to replace a JACE, the most secure practice is to create a new certificate and set of keys.

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

A port may be blocked or ignored by your firewall or secure router. See “About TCP ports” on page 7-2 for the list of default port numbers. 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 server certificate does not have a matching public key in the Trust Store. Import the certificate into the Trust Store.

My network has been hacked, what should I do?

Get on site as quickly as possible. Take the entire network off the internet. Reset each JACE and configure security again creating and signing all new certificates.

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.

If the certificate has changed and you did not change it, something is wrong. Carefully check the certificate to make sure it is what you expect. To be on the safe side, clear the stores, and have the platform generate a new key pair.
When importing a client certificate into a client Trust Store I get the message, "The ‘Import’ command encountered an error."

Check the certificate and re-create it if necessary. Click the Details button to view the Workbench console. You may be attempting to import a private key into the Trust Store. This cannot be done.

I’m trying to get two stations to connect and it’s 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 Trust Store for the station that is serving as the server. In the Allowed Hosts list, select the certificate and click Approve. Make sure the certificate has the correct name and port number in the Host column.

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

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

Reset or replace a JACE securely

To reset a JACE, you should be on site. Resetting a JACE remotely is not recommended because restoring the private key is exposed.

To replace a JACE, you can reuse the certificate and keys generated and signed for the old JACE. If no backup was available, use the generated, self-signed certificate to create a new certificate.

- “To reset or replace a JACE with security” on page 6-2

To reset or replace a JACE with security

Step 1 Make sure that the JACE is not on the internet.
Step 2 Reboot the JACE and restore the station.
Step 3 Import the Key and Trust Stores from the backup file.
The Key and Trust Stores are not part of a station backup.
Reference

This topic provides quick reference information for SSL Toolset tools and features.

- "SSL Toolset terminology" on page 7-1
- "About TCP ports" on page 7-2
- "About the Certificate Management dialog" on page 7-2
- "About the Certificate Signing dialog" on page 7-7
- "SSL configuration options" on page 7-7

SSL Toolset terminology

**Allowed host**  A host whose server certificate cannot be validated by a trusted certificate in the Trust Store.

**Base certificate**  A certificate that will be used to create a Certificate Signing Request. A base certificate may be a default certificate created by Workbench or a JACE on system start-up, or a certificate you create using the SSL Toolset.

**CA certificate**  A certificate whose private key is used to digitally sign other certificates. This certificate should not be emailed unless it is heavily encrypted in a ZIP file with a strong password.

**Client**  A browser, Workbench, JACE, Supervisor or program (process) that seeks information from a server in a NiagaraAX network.

**Client certificate**  The certificate with its public key (no private key) that resides on the client Trust Store. See also CA certificate, Intermediate certificate, and Root certificate.

**Certificate Authority (CA)**  An entity (certification authority) that issues digital certificates. Digital certificates certify the ownership of a public key by the named subject of the certificate. This allows others to rely upon the signature presented by the subject and use his key pair (public and private) to encrypt and decrypt data.

**Certificate/certificate of authentication**  A general name for an electronic file that establishes a user’s credentials when doing business or other transactions over the internet. A digital certificate contains the subject, expiration dates, copy of the certificate holder’s public key (used to decrypt messages and digital signatures), and the purpose of the certificate (server, intermediate or root certificate). The certificate may be signed by a Certificate Authority (CA), or it may be self-signed.

See also CA certificate, Self-signed certificate, Root certificate, and Server certificate.

**Key Store**  A location for storing a certificate with both its public and private keys.

**Encryption/decryption**  The process of using a pair of keys to scramble data at the sending end of a communication, and unscramble it at the receiving end.

**Handshake**  The initial exchange of certificates between a client and server that establishes a communication session.

**Intermediate certificate**  A certificate between the root certificate and server certificate in a certificate chain of trust. An intermediate certificate is signed by the private key of the root certificate or another intermediate certificate. During identity verification, the signature of the server certificate is validated using the signature of the intermediate certificate, whose signature is, in turn, validated against the root certificate’s signature.
The use of intermediate keys isolates servers. If one key is compromised the entire network is not threatened.

**Private key** A software entity based on prime numbers and used to encrypt and decrypt data. For encryption and decryption to remain secure the private key must be physically protected.

**Program (or process)** One of these NiagaraAX entities that provide communication services within a NiagaraAX network: Niagarad, Fox Service, and Web Service.

**Protocol** A set of rules that facilitates information exchange within a computer system, between computers, and between a client and server.

**Public key** A software entity based on prime numbers and used to encrypt and decrypt data. This key matches its private key and may be distributed freely.

**Root certificate** A self-signed certificate that is implicitly trusted and used to sign other certificates. See also CA certificate.

**Secure Socket Layer (SSL)** A commonly-used protocol for managing the security of message transmission over the internet. SSL uses encryption keys and includes a digital certificate.

**Self-signed certificate** A certificate that has not been signed by a CA. It is signed, but by its own private key. In NiagaraAX no separate step or procedure is required to self-sign a certificate. When first generated each certificate is self-signed by default.

**Server** A platform or program (process) that offers information to another platform, program browser or Workbench in a NiagaraAX network. See “NiagaraAX’s client/server architecture” on page 5-1 for a description of the various roles within a NiagaraAX network.

**Server certificate** A certificate that is used primarily by the JACE or Supervisor station/platform for encryption and decryption. In NiagaraAX, the private key of a server certificate is not used to sign other certificates. See also Certificate, CA Certificate.

**Transport Layer Security (TLS)** A commonly-used protocol for managing the security of message transmission over the internet. TLS uses encryption keys and includes a digital certificate.

**Trust Store** A location for storing a trusted certificate with its public key. The Trust Store contains no private keys.

### About TCP ports

Ports may be blocked or ignored by firewalls or secure routers. You must be aware of these blocked ports, and make appropriate exception rules where necessary. If a port is blocked by a firewall or router, communication will not succeed.

The default Niagara TCP port numbers are:

- **fox**: 1911
- **foxs**: 4911
- **platform**: 3011
- **platformssl**: 5011
- **http**: 80
- **https**: 443
- **email**: 25
- **email ssl**: 587, 965

### About the Certificate Management dialog

The Certificate Management view of the SSL Toolset allows you to create PKI (Public Key Infrastructure) digital certificates; to create Certificate Signing Requests (CSRs); and to import and export keys and certificates to and from the Workbench and JACE key stores.

- "About the Key Store tab" on page 7-3
- "About the Private Key Password dialog" on page 7-5
- "About the Trust Store tab" on page 7-6
- "About the Allowed Hosts tab" on page 7-6
About the Key Store tab

**Figure 7-1  Key Store tab**

- "Key Store columns" on page 7-4
- "Key Store buttons" on page 7-4

**Key Store columns**
- **Alias** is a name used to distinguish certificates from one another in the **Key Store**. Use it to identify certificates by location or function.
- **Issued By** identifies the entity that created the key.
- **Subject** is the Distinguished Name.
- **Not Before** displays the date before which the key cannot be used.
- **Not After** displays the expiration date for the key.
- **Key Algorithm** refers to the mathematical formula used to calculate the key.
- **Key Size** shows the size of the key in bits. Four key sizes are allowed: 1024 bits, 2048 bits (this is the default), 3072 bits, and 4096 bits. The bigger the key, the longer it takes to generate.
- **Signature Algorithm** refers to the mathematical formula used to sign the certificate.
- **Signature Size** shows the size of the signature.
- **Valid** shows certificate dates.
- **Self Signed** indicates that the certificate was signed with its own private key.

**Key Store buttons**
- **View** displays certificate details for the selected certificate.
- **New** opens the **Generate Self Signed Certificate** dialog, which is used to create CA and server certificates.
- **Cert Request** opens a **Certificate Request** dialog, which is used to create a Certificate Signing Request (CSR).
- **Delete** removes the certificate from the **Key Store**.
- **Import** adds the certificate (.pem file) to the **Key Store**.
- **Export** saves a copy of the selected certificate to the hard disk. The file extension is .pem.
- **Reset** deletes all certificates in the **Key Store** and creates a new default certificate. It does not matter which certificate is selected when you click **Reset**.

**Caution**
The **Reset** button facilitates creating a new key pair (private and public keys) for the entity, but may have unintended consequences if you delete valid certificates. Export all certificates before you reset.
About the Generate Self-Signed Certificate dialog

Figure 7-2  Distinguished Name dialog

You use this dialog 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 fields. Do not enter blank in any field. Spaces and periods are allowed. Enter full legal names.

**Alias** is a short name used to identify the certificate. This field is required. Use this field to indicate where the certificate will be used.

**Common Name (CN)** is your Distinguished Name and can be the same as the Alias. Do not use the symbols “*” or “?” as part of this name. This name appears as the **Subject** in the **Key Store**. This field is required.

**Organizational Unit (OU)** is 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)** is the legally registered name of your company or organization. Do not abbreviate this name. This field is required.

**Locality (L)** is 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)** is the complete name of the state or province in which your organization is located. This field is optional.

**Country Code (C)** is a two-character ISO-format country code. If you do not know your country’s two-character code, check the internet. This field is required.

**Not Before** indicates the date on which the certificate becomes valid.

**Not After** indicates when the certificate expires.

**Key Size** establishes the size of the key in bits. The larger the key, the longer it takes to generate.

**Certificate Usage** identifies the purpose of the certificate. In NiagaraAX, certificates are either server or CA certificates. Other open-source certificate management software utilities may allow other usages.

**Email Address** is the address to which your CSR will be sent.
**About the Private Key Password dialog**

**Figure 7-3 Private Key Password dialog**

Password is the credential that will be required for each action that involves the private key. 
Confirm ensures you meant what you typed for Password.

- **OK** creates a certificate (.pem file) with the private key and enables the password.
- **Skip** creates a certificate (.pem file) without the private key and enables the password.
- **Cancel** aborts the export process and displays the Key Store tab view.

**About the Trust Store tab**

The trust store contains a list of trusted certificates.

**Figure 7-4 Trust Store tab**

- “Trust Store columns” on page 7-6
- “Trust Store buttons” on page 7-6

**Trust Store columns**

Trust Store columns are similar to those for the Key Store. See “Key Store columns” on page 7-4

**Trust Store buttons**

- **View** displays certificate details.
- **Delete** removes the certificate from the Trust Store.
- **Import** imports a certificate into the Trust Store. If the private key is part of the certificate, it is ignored. Private keys do not belong in the Trust Store.
- **Export** saves a copy of the selected certificate(s) to the local hard disk using the .pem extension.
About the Allowed Hosts tab

Figure 7-5  Allowed Hosts tab

- “Allowed hosts columns” on page 7-6
- “Allowed hosts buttons” on page 7-7

Allowed hosts columns
- **Host** identifies the server.
- **Subject** displays the Distinguished Name.
- **Approval** indicates that servers within the network to which the client may connect. If approval is set to **no**, the system will not allow the client to connect.
- **Created** is the date the record was created.
- **Issued By** identifies the Certificate Authority.
- **Not Before** is the date before which the certificate is valid.
- **Not After** is the date after which the certificate becomes invalid.
- **Key Algorithm** identifies the algorithm used to generate the key.
- **Key Size** identifies the size of the key.
- **Signature Algorithm** identifies the algorithm used to digitally sign the certificate.
- **Signature Size** identifies the signature size.
- **Valid** indicates if this server is trusted.

Allowed hosts buttons
- **View** displays server details.
- **Approve** designates server as an allowed host.
- **Unapprove** designates server as an unallowed host. Any attempted communication will be terminated.
- **Delete** deletes the server from the list.

About the Certificate Signing dialog

Figure 7-6  Certificate signing options

Signing fields:
- **Select a certificate signing request to sign**: This is the CSR you created.
**Not Before** Date this certificate becomes valid.

**Not After** Expiration date

**CA Alias** is the short name you assigned to the CA certificate whose private key will be used to sign this certificate.

**CA Password** is the password that protects the private key of the CA certificate being used to sign this certificate.

### SSL configuration options

- "SSL configuration options" on page 7-7
- "Fox Service configuration properties" on page 7-8
- "Web Service configuration properties" on page 7-9

### Platform SSL properties

To access this dialog, expand the **Platform** node in the Nav tree, double-click **Platform Administration** and click **Change SSL Settings**.

![Platform SSL properties](image)

**State** Enabled or Disabled

**Port** The port for SSL.

**Certificate** The **Alias** for the server certificate.

**Protocol** The security service to use. The default is to use both standard protocols (**SSLv3+TLSv1**). During the handshake, the server and client agree on which protocol to use. It may be one or the other.
**Fox Service configuration properties**

To view this screen expand the Station in the Nav tree to the **Station > Config > Drivers** level, right-click **Niagara Network** and click **Views > Property Sheet**

**Figure 7-8  Fox Service properties**

![Fox Service properties](image)

Fox properties related to SSL:

**Fox Enabled** should always be set to **true**.

**Foxs Port** identifies the port number for secure communication. The default is 4911.

**Foxs Enabled** turns secure Fox communication on.

**Foxs only** causes the system to redirect to a secure connection (foxs) if an attempt is made to connect using a connection that is not secure (fox alone). Once you set this property to **true**, it makes no difference what type of connection you select at login. All communication is secure.

**Foxs Min Protocol** selects the communication protocol. The default is to use both standard protocols (**SSLv3+TLSv1**). During the handshake, the server and client agree on which protocol to use. It may be one or the other.

**Foxs Cert** allows you to select a certificate that is different from the one being used by the platform.
Web Service configuration properties

To view this screen expand the Station in the Nav tree to the Station > Config > Services and double-click WebService.

Figure 7-9 Web Service properties

`Https Port` indicates the port number for secure communication. The default is 80.

`Https Enabled` turns secure http communication on.

`Https only` causes the system to redirect to a secure connection (https) if an attempt is made to connect using an insecure connection (http alone). Once you set this property to true, it makes no difference what type of connection you select at login. All communication is secure.

`Https Min Protocol` selects the security protocol. The default is to use both standard protocols (SSLv3+TLSv1). During the handshake, the server and client agree on which protocol to use. It may be one or the other.

`Https Cert` allows you to select a certificate that is different from the one being used by the platform.