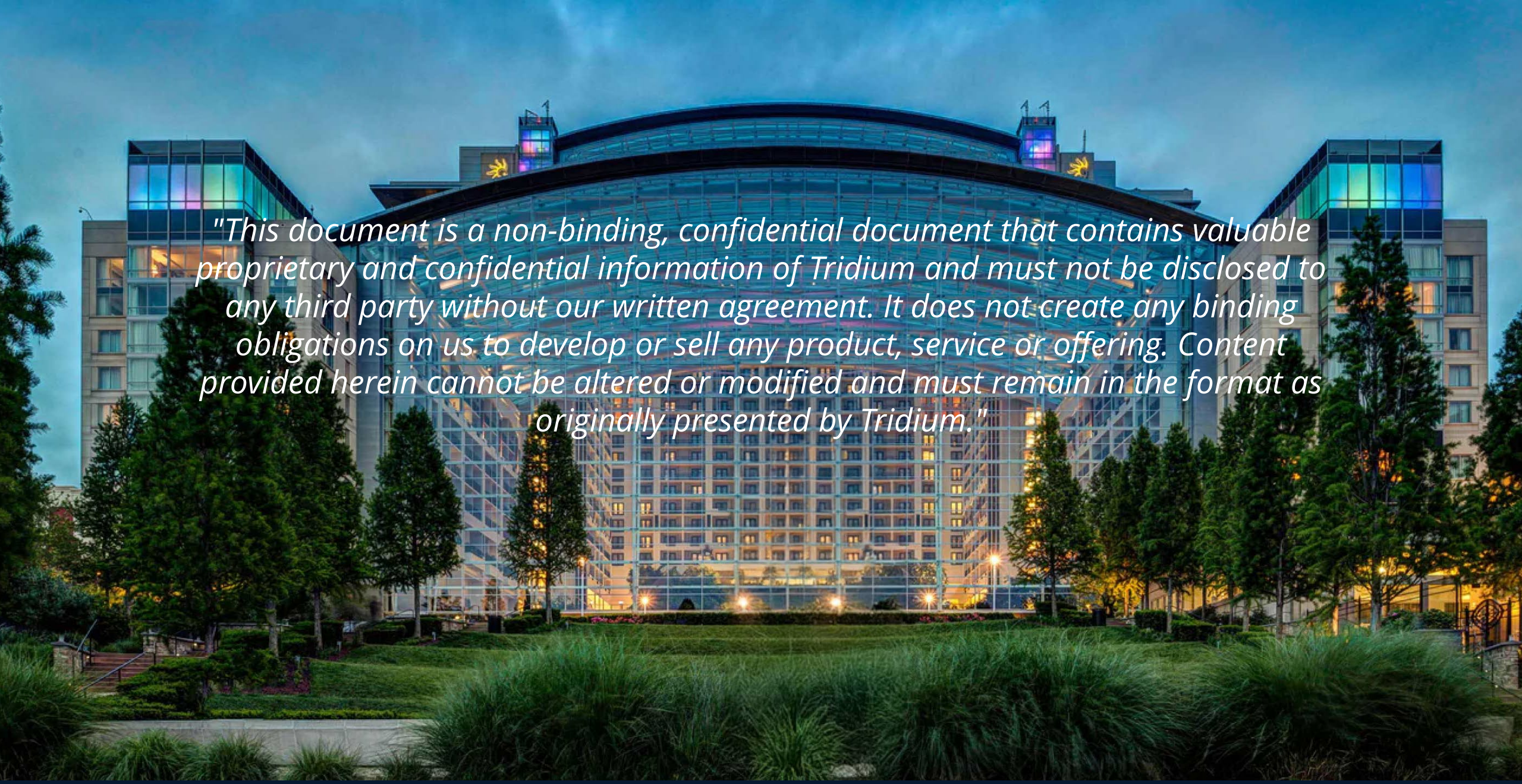

NIAGARA
SUMMIT
2026
SEAMLESS CONNECTIVITY,
POWERFUL INTELLIGENCE
TRIDIUM

Design Fundamentals:
Leveraging Enterprise Features

NS26

TRIDIUM


"This document is a non-binding, confidential document that contains valuable proprietary and confidential information of Tridium and must not be disclosed to any third party without our written agreement. It does not create any binding obligations on us to develop or sell any product, service or offering. Content provided herein cannot be altered or modified and must remain in the format as originally presented by Tridium."

Design Fundamentals:

Leveraging Enterprise Features

Kyle Sardinia - Tridium

NS62

TRIDIUM

Objectives:

- Enterprise Authentication
- Provisioning
- Virtual Policies and nspace
- SystemDb
- Export Tags

NS2 6

Authentication

Validates the identity of a subject, which can be:

- a human user
- a system
- or an application.

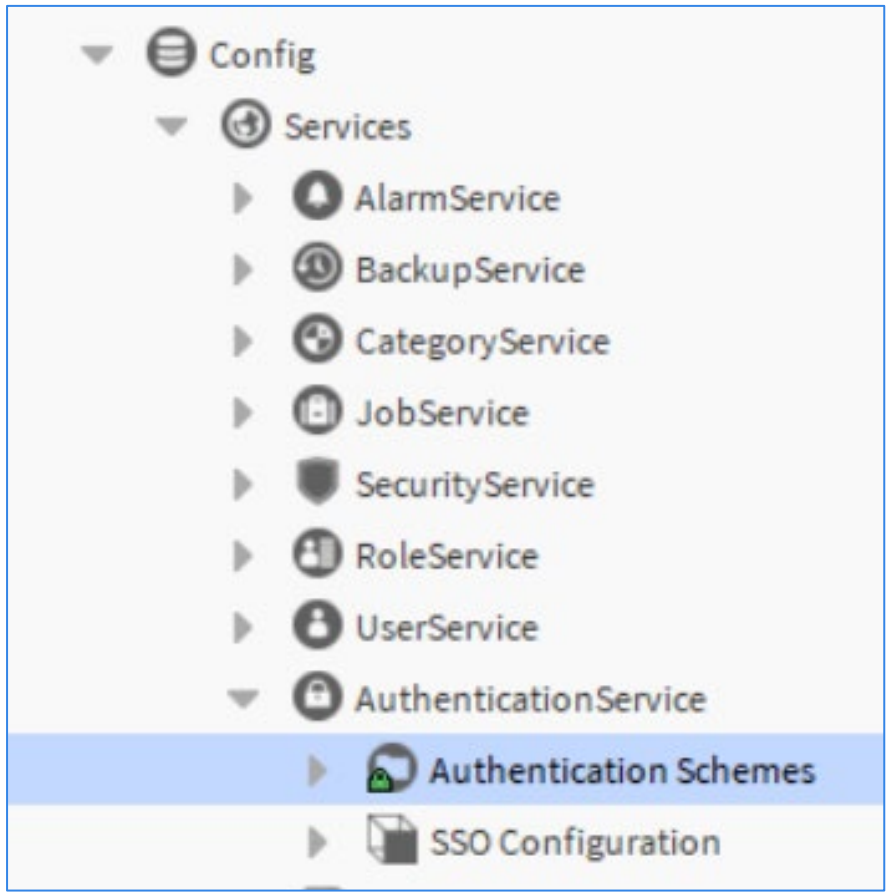
- When a station attempts a connection, it checks the user's login credentials

Authentication

The process depends on the **authentication scheme** and on the **type of connection**:

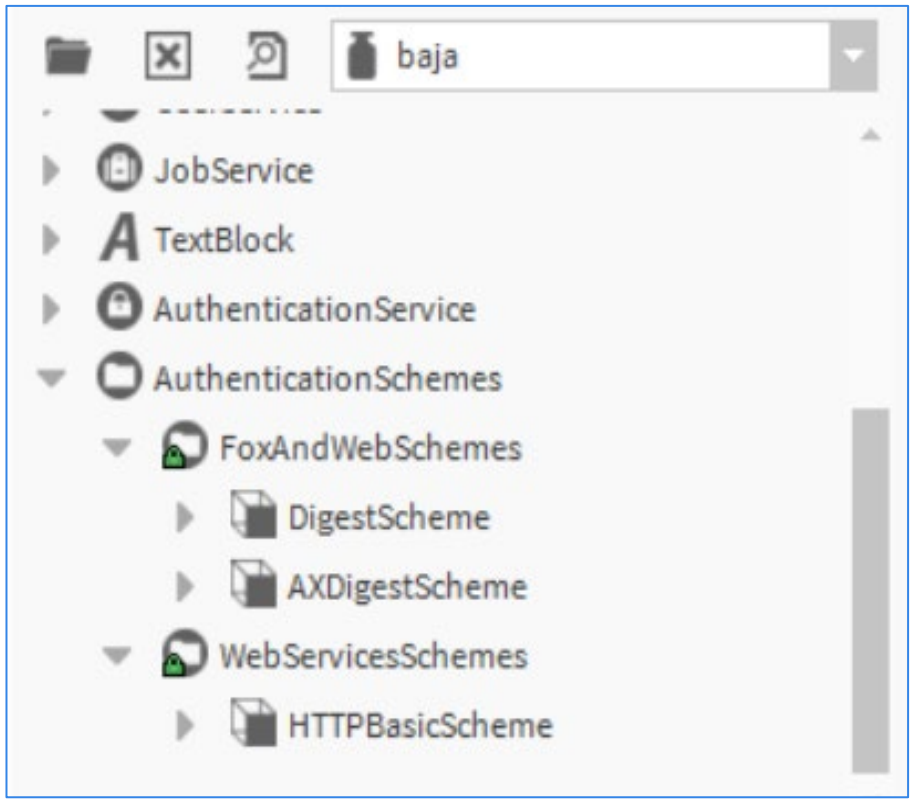
- Workbench-to-station (**FoxService**)
- HTTPs browser-to-station (**WebService**)
- Station-to-station (**FoxService**)

Niagara AuthenticationService



- Stations have a pluggable authentication system that support **many different authentication schemes** at once.
- Schemes determine:
  - **how** a client **talks** to the station
  - **how** the credentials are **transmitted**.

DigestScheme (default authentication scheme)



- **never sends** a user **password** directly to the station.
- **sends proof** that the user knows the password.

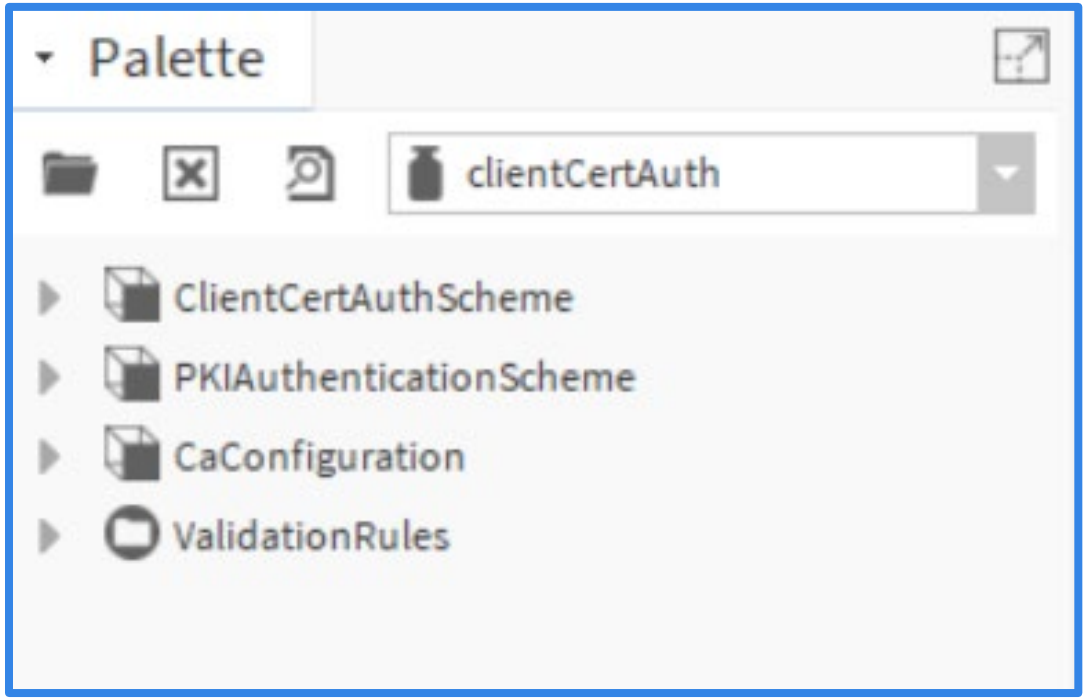
AxDigestScheme

- Niagara 4 station comes with the Digest and AXDigest authentication schemes installed by default
- Allows AX stations to connect to N4 station
- If your station will not have AX Connections, **you should remove it**

HTTPBasicScheme

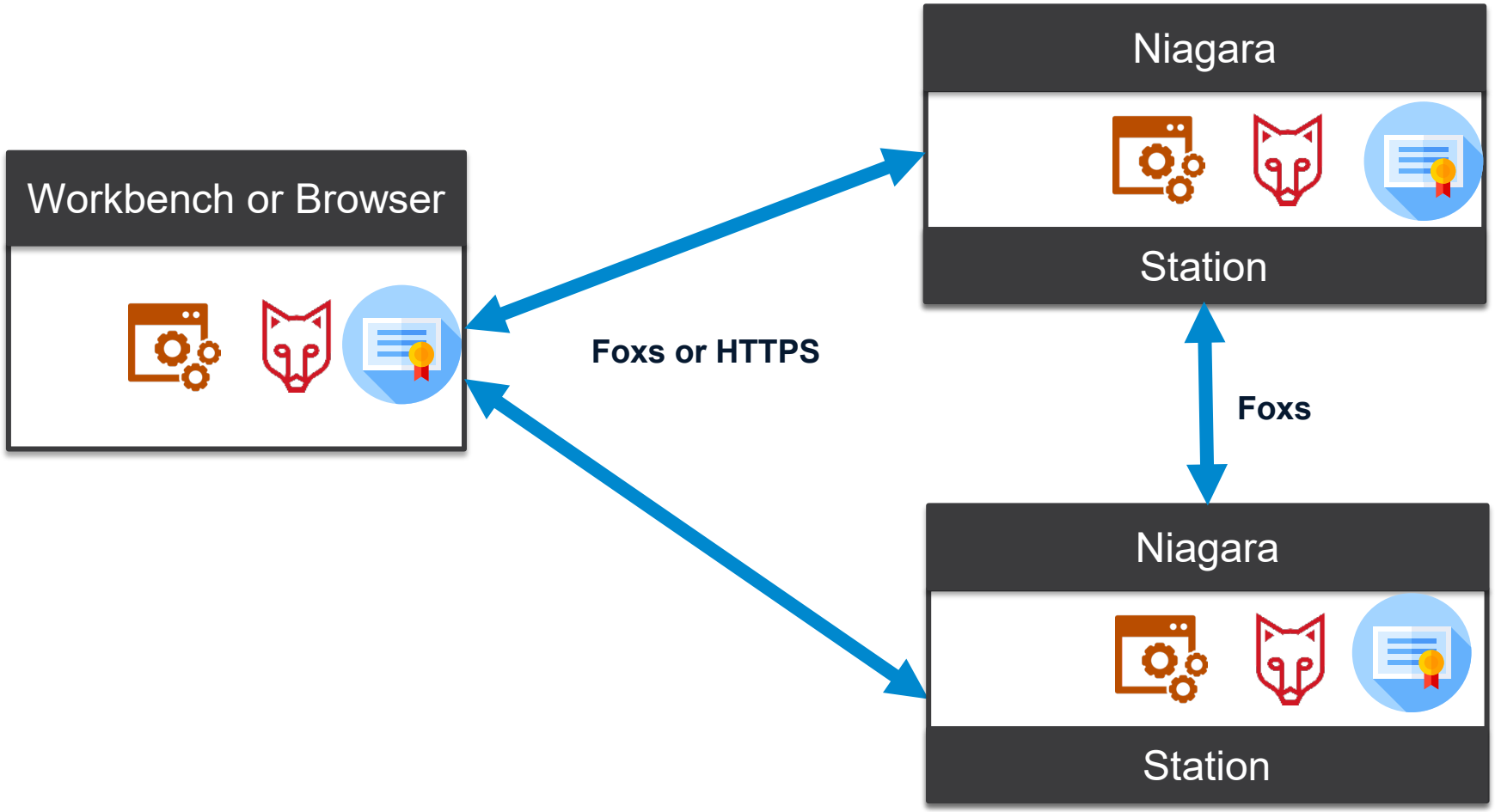
- It only works via the web
- Intended for clients that cannot use cookies
- Sends the username and password over the connection
- **Should use HTTPs**

Client Certificate Authentication

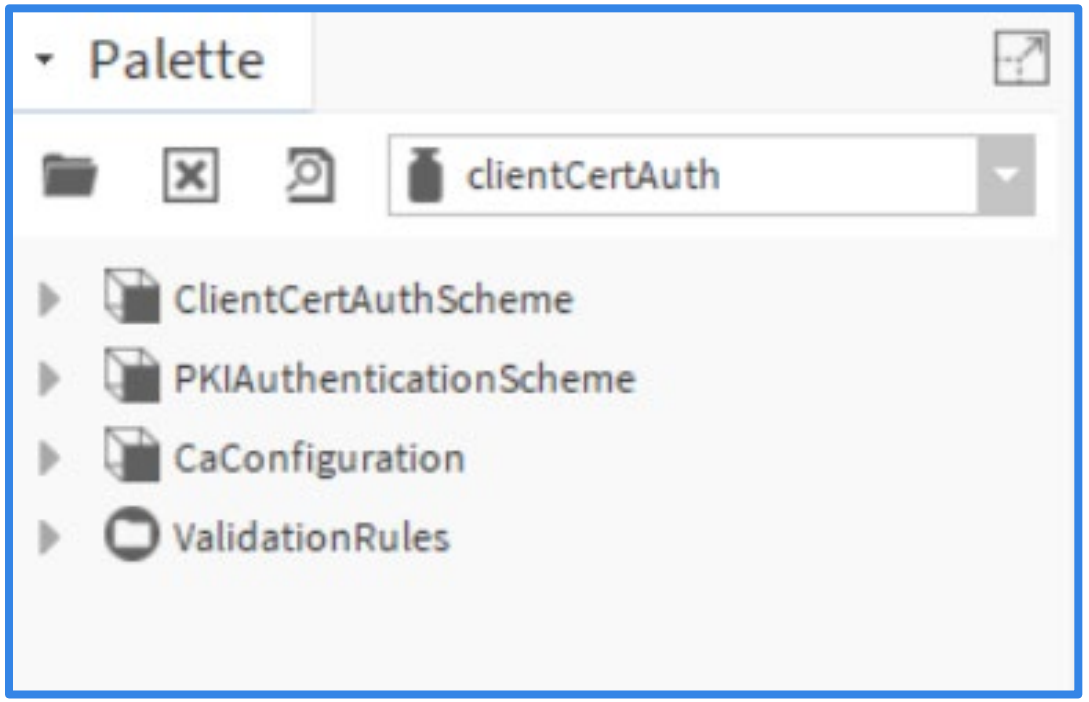


- Each user's client certificate with its public key is **directly bound** to the user
- Each certificate's public key **matches** its private key
- System prompts the user to upload his or her certificate with its private key
- Enables "Kiosk Mode"
- Also Niagara Network Service Accounts!

Client Certificate Architecture

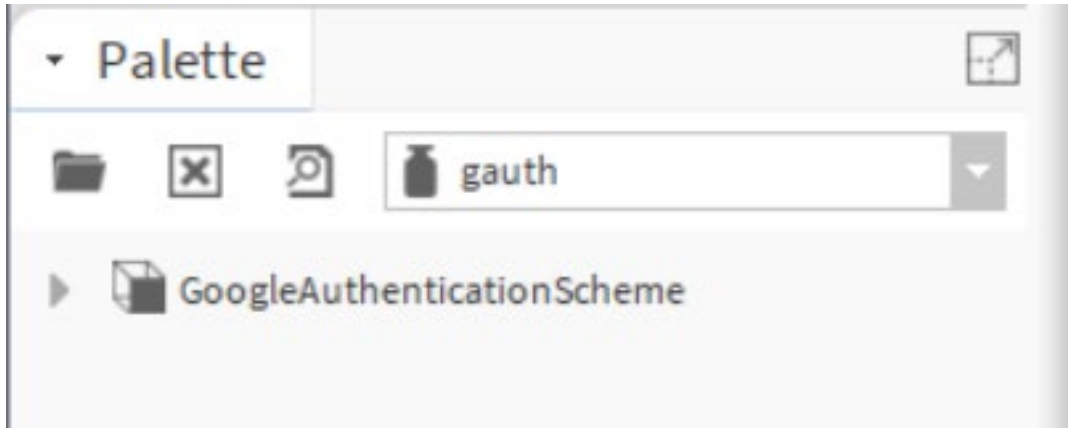


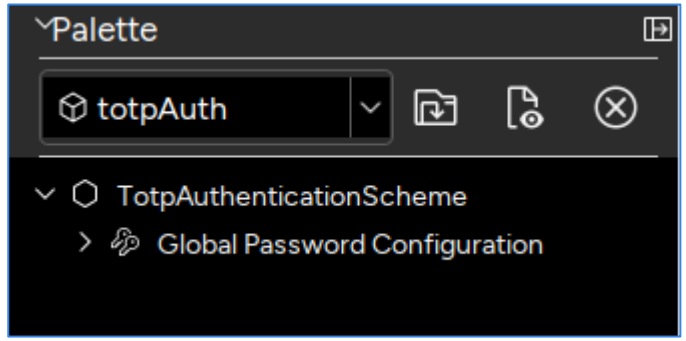
PKI Authentication (N4.15)



- The PKI Authentication (Public Key Infrastructure Authentication) allows you to log into the station using client certificate authentication (mTLS).
- In the **browser**, when a user accesses the login page, they are prompted to select their certificate from a list installed certificates in their browser
- The CA (Certificate Authority) is configured on the station.

TOTP Authentication (Gauth)



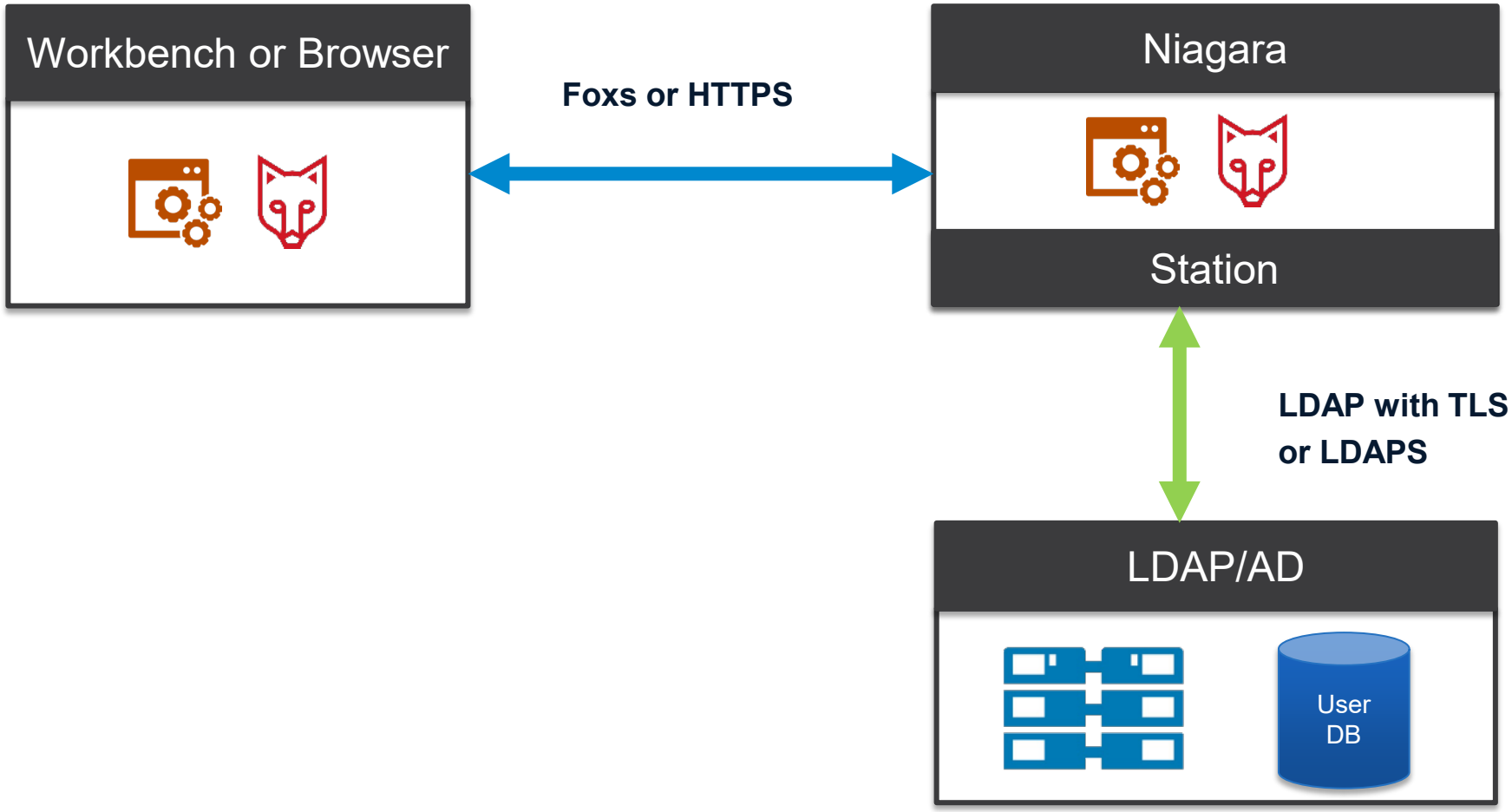


- The GoogleAuthenticationScheme provides two-factor authentication using a password and single-use token sent to the user's mobile device.
- The authenticator app is time based and automatically updates the tokens every 30 seconds.
- This scheme requires a TOTP (**time-based one-time passwords)** Authenticator app be installed on the user's phone.

LDAP and AD

- **Lightweight Directory Access Protocol (LDAP)** is an application protocol for accessing and maintaining distributed directory information services over an IP network.
- **Active Directory (AD)** is a Microsoft specific implementation of an LDAP server.
- **LDAP** is commonly used in corporate networks for **managing domain user accounts** and the user's access to applications and network resources.
- Allows the **customer's IT group** to manage **access to the Niagara station** using their standard tools.
- Provides **single login credentials** but not SSO

LDAP/AD Architecture



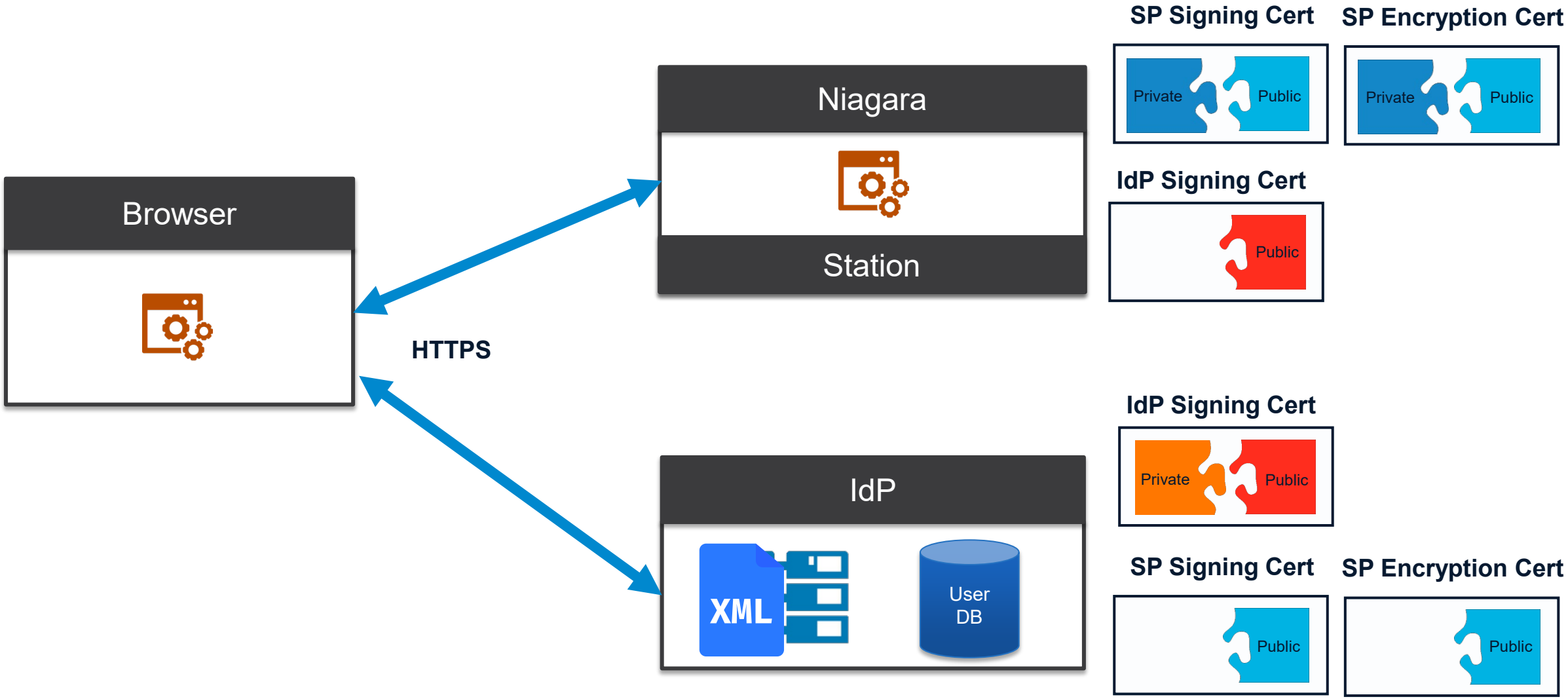
SAML

- An open standard for exchanging **authentication and authorization data** in the form of **messages** passed between security domains.
- Messages **may be encrypted** and are **typically signed** using a PKI certificate.
- Since **Niagara 4.4 version SAML 2.0** is supported.
- Works with popular third party on **premise and cloud based** SAML IdPs

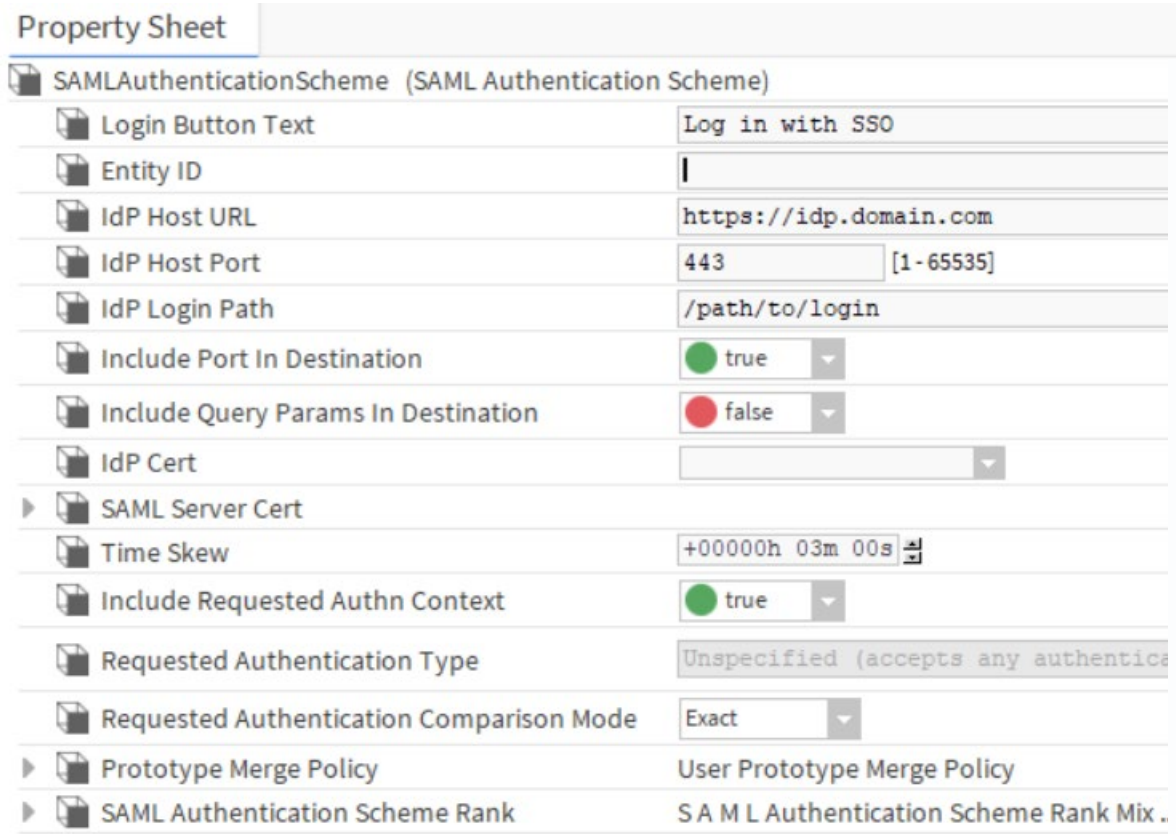
SAML

- **Assertion** – a package of information that supplies statements made by a SAML authority.
- **Attribute** – a piece of information which determines the properties of a field or tag in a database.
- **Identity Provider (IdP)** – a system entity that issues authentication assertions in conjunction with SSO.
- **Service Provider (SP)** – a system entity that receives and accepts authentication assertions in conjunction with SSO.

SAML Architecture



SAMLAuthenticationScheme

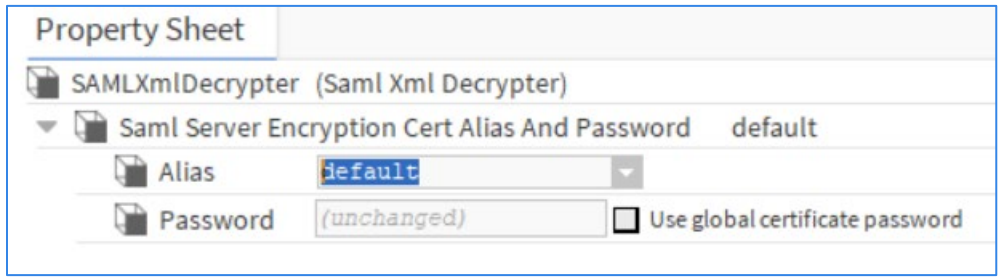


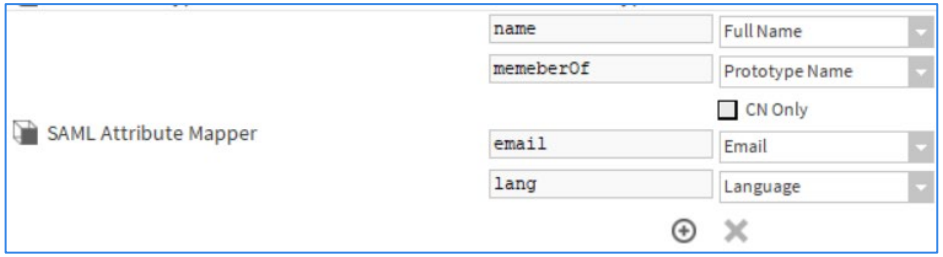
- Entity ID – URL to identify the station (SP) SAML services.
- IdP Host URL – redirect URL to the IdP server.
- IdP Login Path – appended to the IdP Host URL to specify the IdP login page URL.
- IdP Cert – provided by the IdP, must be in the station's trust store. Validates messages signed by the IdP.
- SAML Server Cert – in the station's key store, must be provided to the IdP. Used to sign messages sent to the IdP.

SAML User Prototypes

- The **defaultPrototype** is a **baja:User** component used with Niagara user synchronization and legacy LDAP/AD authentication.
- **LDAP, AD** and **SAML** authentication utilize newer **baja:UserPrototype** component found in baja and ldap palettes.
- **Alternate Default Prototype** should be configured to select a baja:UserPrototype and is used if no matching prototype is detected.
- **Prototype Merge Policy** default merge mode settings are the most restrictive, can optionally be set to ‘Use First’ which is same as disabled or pre-4.12 behavior.

Saml Encryption & Attribute Mapper





- Optional for IdP to encrypt assertions sent to SP
- Must add SAML Xml Decrypter to SAML Authentication Scheme.

- Defines attributes by name from the SAML claim sent by IdP and maps the attribute values to properties on the Niagara user account.

SAML Metadata URL (4.8)

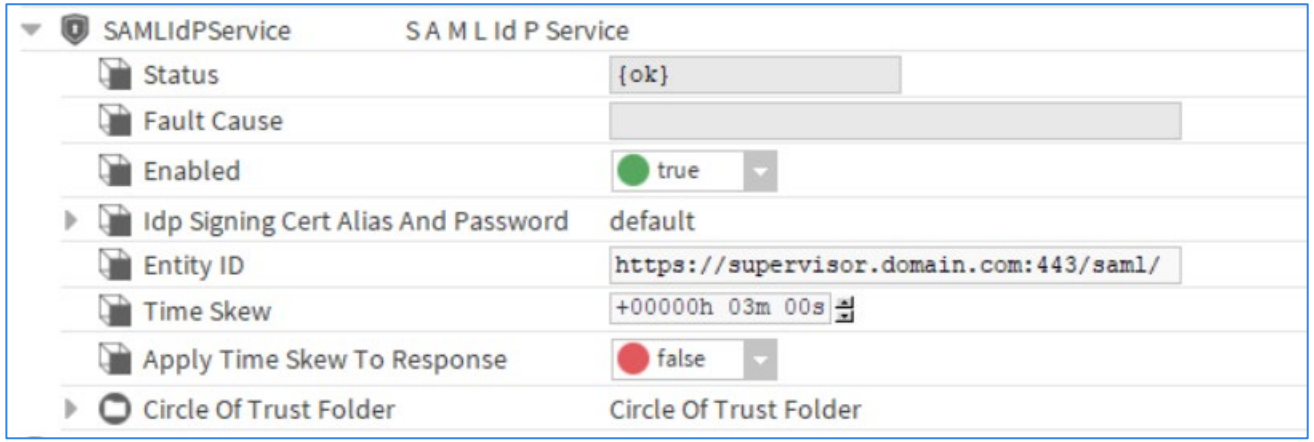
- Simplifies IdP configuration by providing metadata via XML.

https://<host>/saml/samlrp/metadata?scheme=<schemeName>

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" validUntil="2028-03-20T08:05:56Z" cacheDuration="PT604800S" entityID="ip-66-111.va51.tridium.com">
  <md:SPSSODescriptor AuthnRequestsSigned="true" WantAssertionsSigned="true" protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
    <md:KeyDescriptor use="signing">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>MIIDmzCCAoOgAwIBAgIMRxoFhhLw3ronolS5MA0GCSqGSIb3DQEBCwUAMD4xETAPBgNVBAMMCE5pYWdhcmE0MRwwGgYDVQQKDBNGb3JSZWNvdmVyeVB1cnBvc2VzMQswCQYDVQQGEwJVUzAe
        </ds:X509Data>
      </ds:KeyInfo>
    </md:KeyDescriptor>
    <md:KeyDescriptor use="encryption">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:X509Data>
          <ds:X509Certificate>MIIDmzCCAoOgAwIBAgIMRxoFhhLw3ronolS5MA0GCSqGSIb3DQEBCwUAMD4xETAPBgNVBAMMCE5pYWdhcmE0MRwwGgYDVQQKDBNGb3JSZWNvdmVyeVB1cnBvc2VzMQswCQYDVQQGEwJVUzAe
        </ds:X509Data>
      </ds:KeyInfo>
    </md:KeyDescriptor>
    <md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</md:NameIDFormat>
    <md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="ip-66-111.va51.tridium.com/assertionConsumerService" index="1"/>
  </md:SPSSODescriptor>
</md:EntityDescriptor>
```

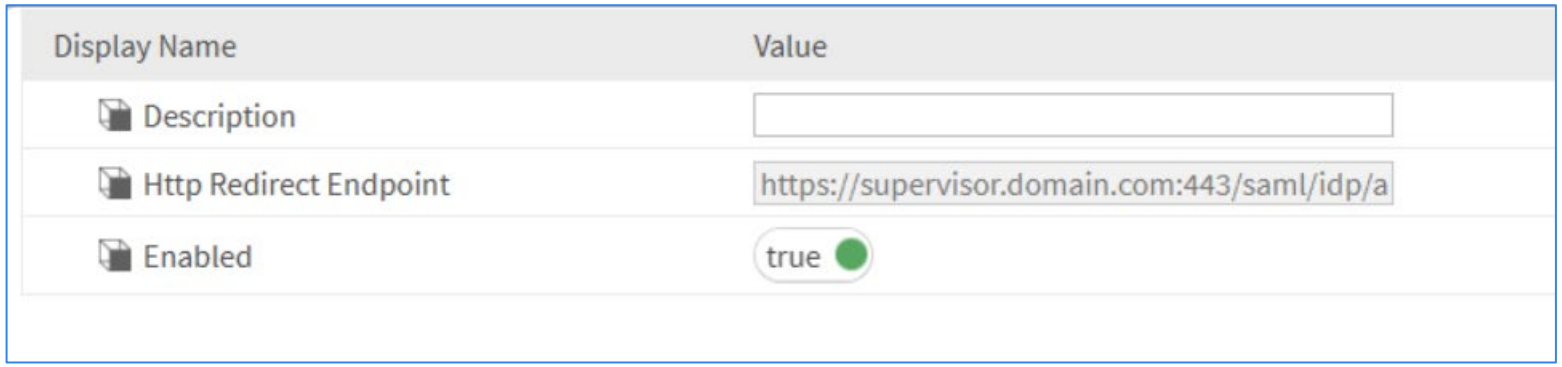
Niagara Supervisor as SAML IDP



- Native Niagara based Identity Provider (IdP).
- Typically setup in supervisor station.
- Requires samlDP feature in license.
- Requires Niagara 4.9+

Circle Of Trust

- Component which defines a group of stations to which designated users have access via SAML authentication.
- Each COT has its own HTTP Redirect Endpoint URL.
- Can define multiple COT components under the SAML IdP Service.



Circle Of Trust - Editor

- **Stations** – configures which stations are included.
- **User** – configures which station users are included.
- **Auth Schemes** – configures authentication schemes such as LDAP where a local user may not exist to be assigned. Enabling an authentication scheme allows all users who log in with that scheme to utilize SAML SSO.
- **Prototypes** – defines names for user prototypes used in the remote station to assign role, nav file and other properties to a user created via SAML authentication.

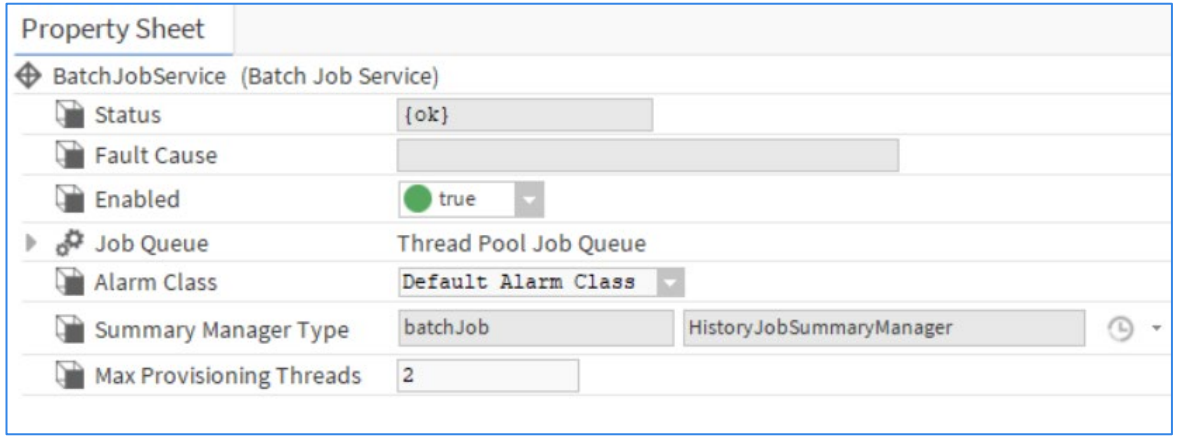
COT – SAML Prototypes

- Configures the user prototype for each COT.
- Only lists COT components which have the user enabled.
- Configured on user prototype for other authentication schemes such as LDAP.

Provisioning Tools for SAML SSO

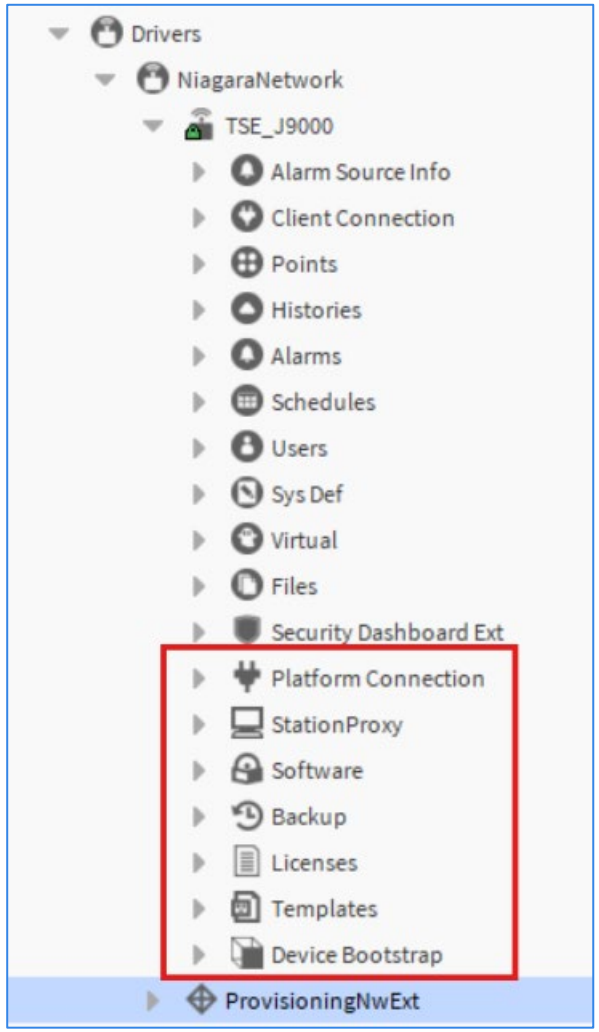
- First, add/configure Supervisor's **SAML IdP Service** including COT and user's SAML prototypes.
- Either manually or using set property job step, setup user prototypes in remote stations.
- Use the provisioning job step to:
  - Add and configure the SAML Authentication Scheme to remote stations.
  - Import the public signing certificate from the supervisor to the trust store of each remote station.
  - Generate a unique SAML signing certificate in the remote station's user key store to be used for signing SAML messages.
  - A copy of the remote stations SAML signing certificate's public key is assigned to the station Service Provider (SP) under the COT in the supervisor.

Provisioning



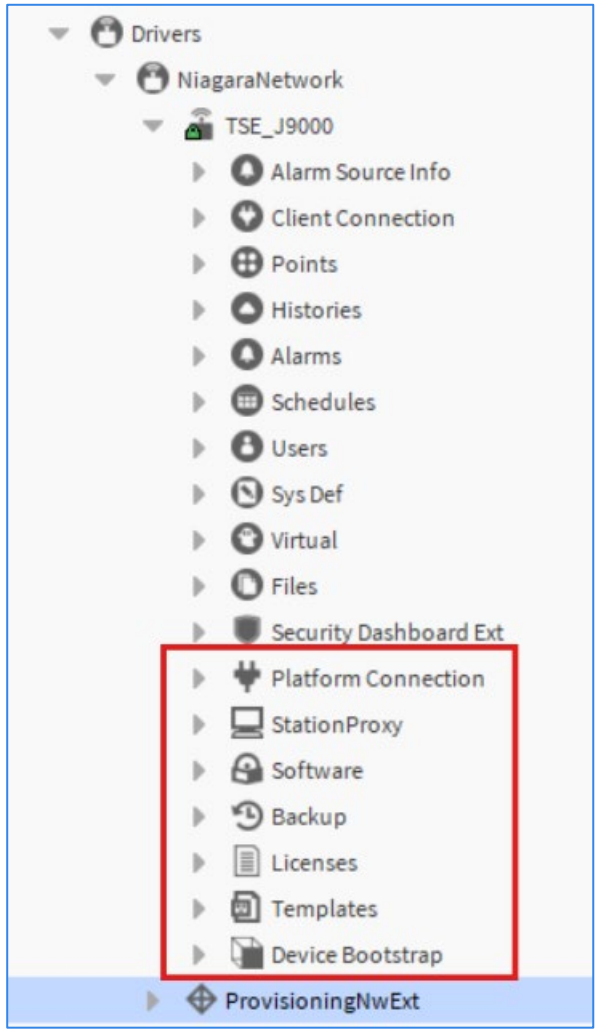
- Requires the **Batch Job Service** to be in the services container of the Supervisor station.
- The **Job Queue Max Threads** defaults to 1 thread but can be adjusted to allow multiple jobs to run concurrently.
- The **Alarm Class** property should be set to a specific alarm class to facilitate routing provisioning related alarms.

ProvisioningNwExt



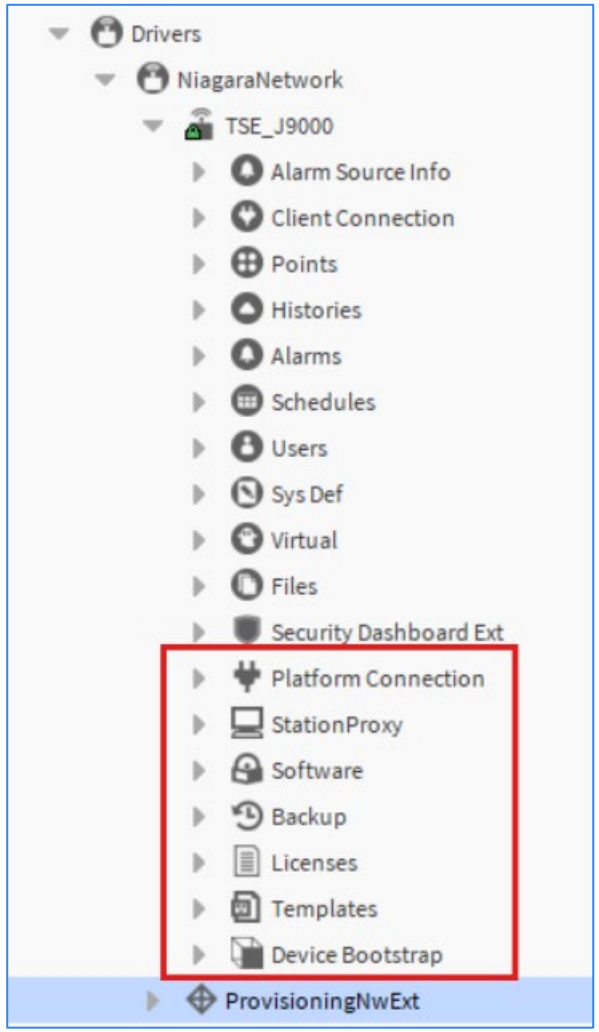
- **ProvisioningNwExt** must be added to the Niagara Network in the Supervisor station.
- Results in additional extensions being automatically added to Niagara station device.
- Use the **Provisioning Manager**, alternate view on the Niagara Network, to configure the platform credentials to be used by the station to connect to the platform of each JACE.

Provisioning Device Extentions

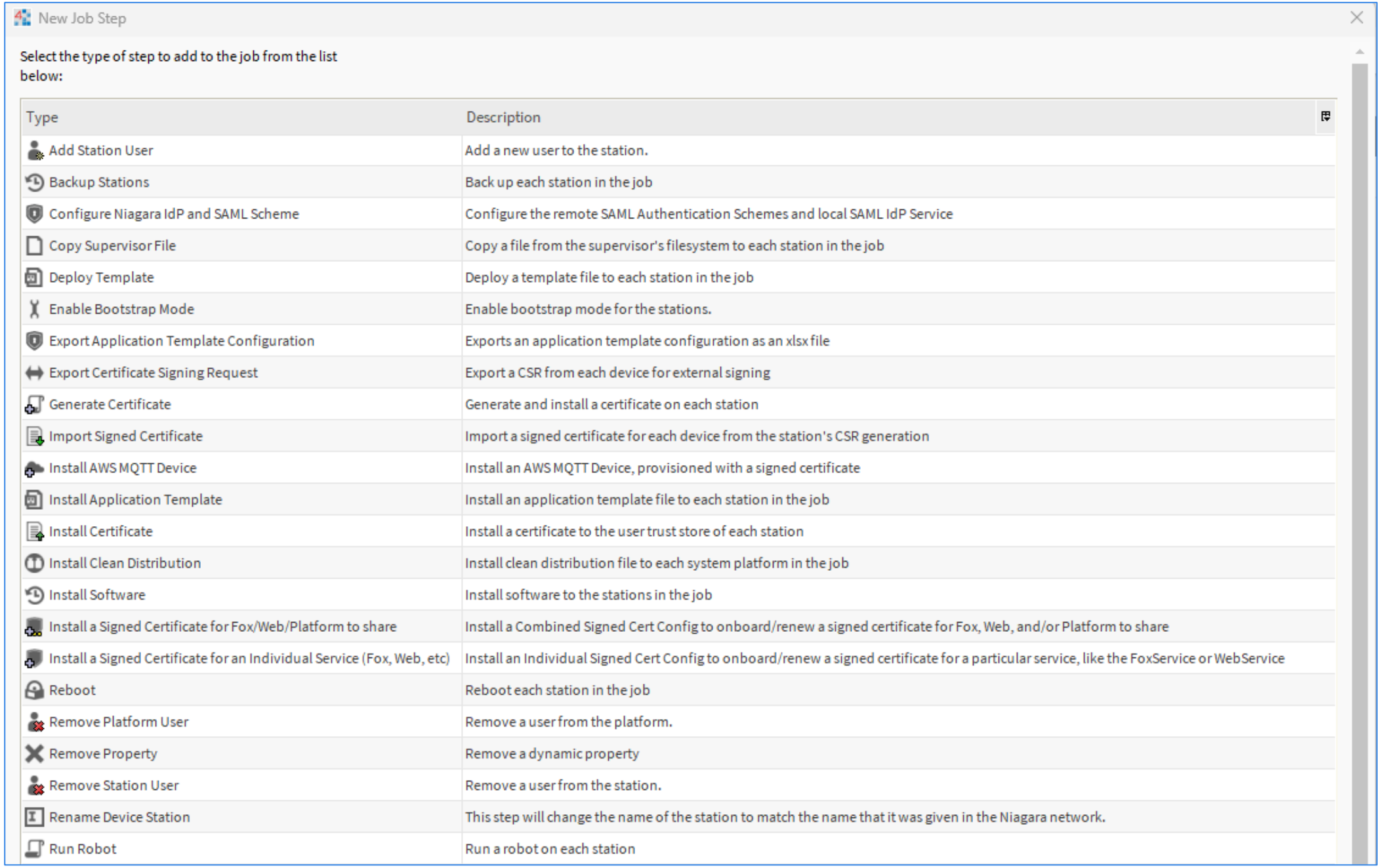


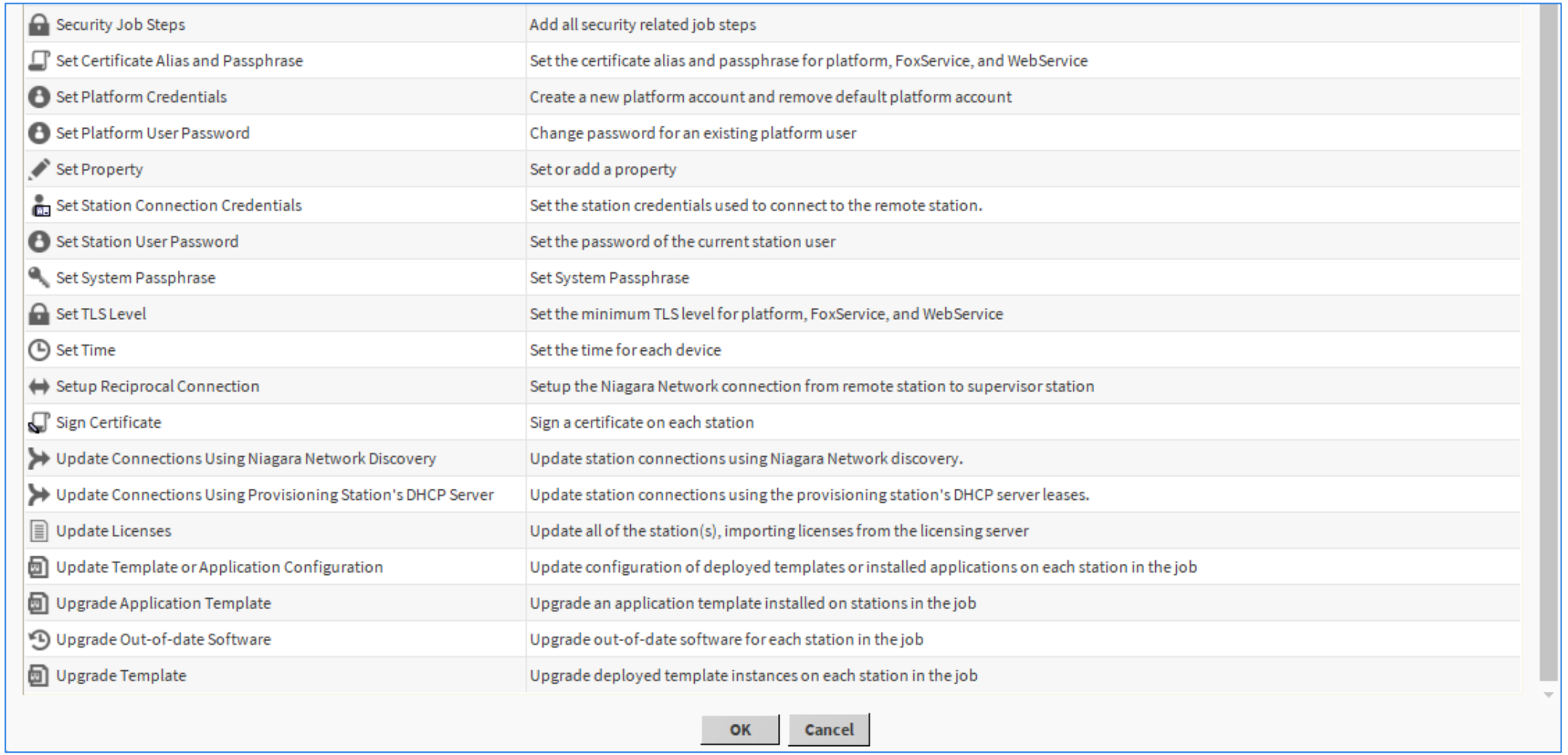
- **Platform Connection** – properties used to configure credentials for the Supervisor to make automated platform connections to the remote hosts.
- **Station Proxy** – provides access to the application director for the remote host via the Supervisor station.
- **Software** – provides a Software Manager view of the remote host via the Supervisor station.
- **Backup** – displays information regarding the last backup of the remote host.

Provisioning Device Extentions

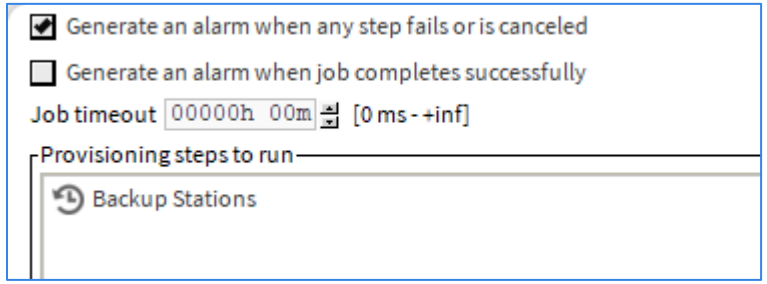


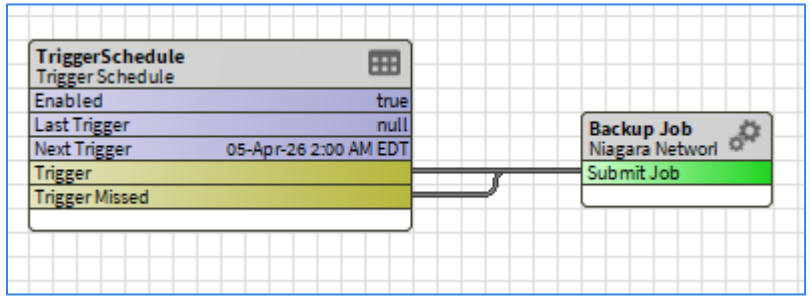
- **Licenses** – provides basic information about the remote hosts's licenses and certificates such as the host ID, what licenses and certificates are installed and their expiration.
- **Templates** – provides functionality to deploy templates to remote station.
- **Device Bootstrap** – enables provisioning remote devices with default platform credentials and passphrase.


New Job Step
Select the type of step to add to the job from the list below:
Type
Description
Add Station User
Add a new user to the station.
Backup Stations
Back up each station in the job
Configure Niagara IdP and SAML Scheme
Configure the remote SAML Authentication Schemes and local SAML IdP Service
Copy Supervisor File
Copy a file from the supervisor's filesystem to each station in the job
Deploy Template
Deploy a template file to each station in the job
Enable Bootstrap Mode
Enable bootstrap mode for the stations.
Export Application Template Configuration
Exports an application template configuration as an xlsx file
Export Certificate Signing Request
Export a CSR from each device for external signing
Generate Certificate
Generate and install a certificate on each station
Import Signed Certificate
Import a signed certificate for each device from the station's CSR generation
Install AWS MQTT Device
Install an AWS MQTT Device, provisioned with a signed certificate
Install Application Template
Install an application template file to each station in the job
Install Certificate
Install a certificate to the user trust store of each station
Install Clean Distribution
Install clean distribution file to each system platform in the job
Install Software
Install software to the stations in the job
Install a Signed Certificate for Fox/Web/Platform to share
Install a Combined Signed Cert Config to onboard/renew a signed certificate for Fox, Web, and/or Platform to share
Install a Signed Certificate for an Individual Service (Fox, Web, etc)
Install an Individual Signed Cert Config to onboard/renew a signed certificate for a particular service, like the FoxService or WebService
Reboot
Reboot each station in the job
Remove Platform User
Remove a user from the platform.
Remove Property
Remove a dynamic property
Remove Station User
Remove a user from the station.
Rename Device Station
This step will change the name of the station to match the name that it was given in the Niagara network.
Run Robot
Run a robot on each station

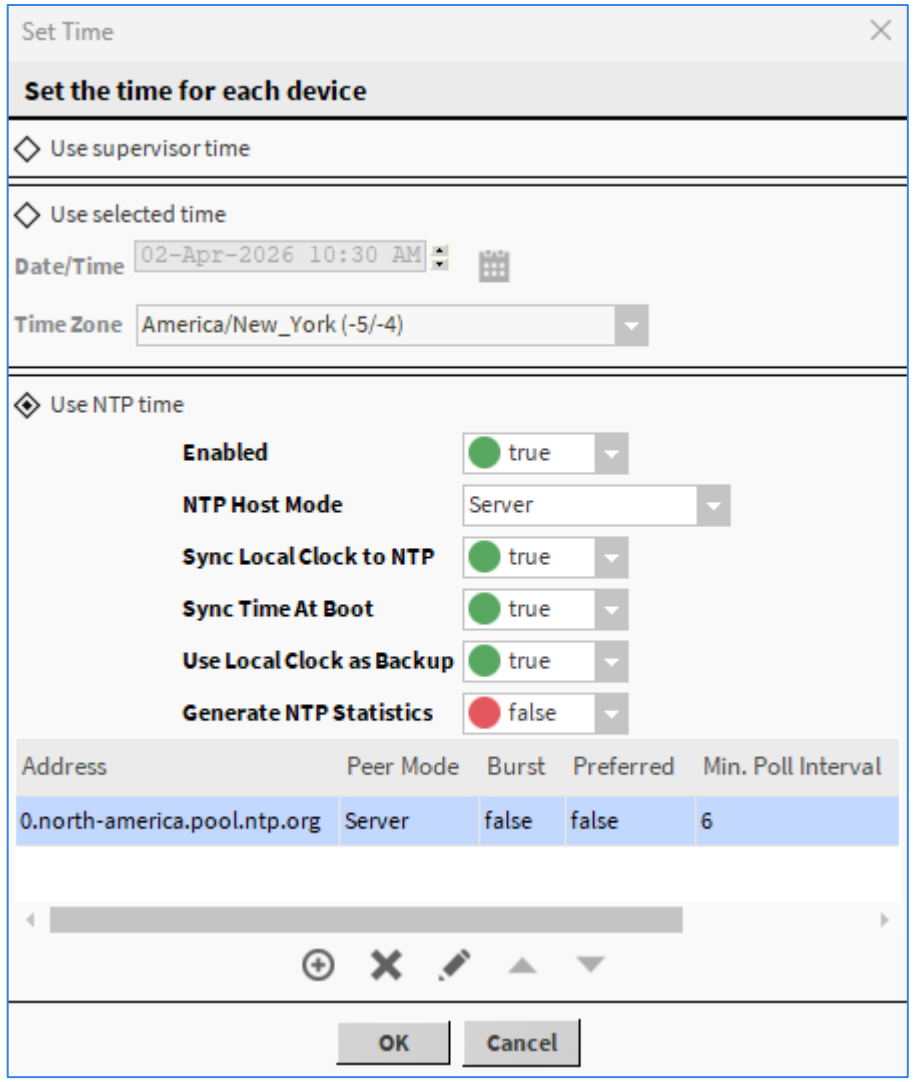

Security Job Steps
Add all security related job steps
Set Certificate Alias and Passphrase
Set the certificate alias and passphrase for platform, FoxService, and WebService
Set Platform Credentials
Create a new platform account and remove default platform account
Set Platform User Password
Change password for an existing platform user
Set Property
Set or add a property
Set Station Connection Credentials
Set the station credentials used to connect to the remote station.
Set Station User Password
Set the password of the current station user
Set System Passphrase
Set System Passphrase
Set TLS Level
Set the minimum TLS level for platform, FoxService, and WebService
Set Time
Set the time for each device
Setup Reciprocal Connection
Setup the Niagara Network connection from remote station to supervisor station
Sign Certificate
Sign a certificate on each station
Update Connections Using Niagara Network Discovery
Update station connections using Niagara Network discovery.
Update Connections Using Provisioning Station's DHCP Server
Update station connections using the provisioning station's DHCP server leases.
Update Licenses
Update all of the station(s), importing licenses from the licensing server
Update Template or Application Configuration
Update configuration of deployed templates or installed applications on each station in the job
Upgrade Application Template
Upgrade an application template installed on stations in the job
Upgrade Out-of-date Software
Upgrade out-of-date software for each station in the job
Upgrade Template
Upgrade deployed template instances on each station in the job
OK
Cancel

Backup Job

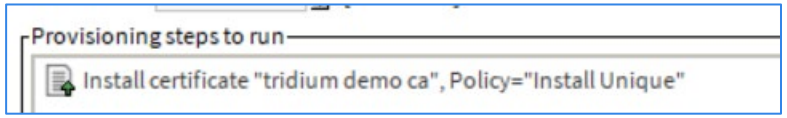


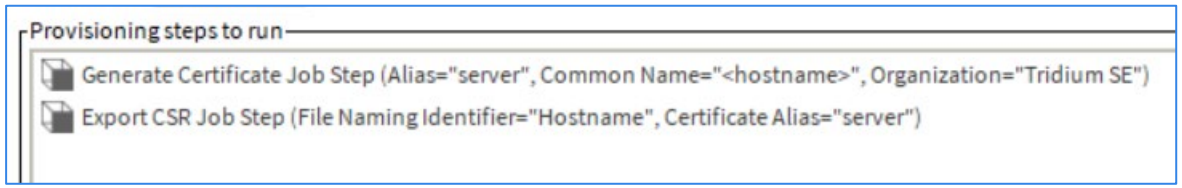


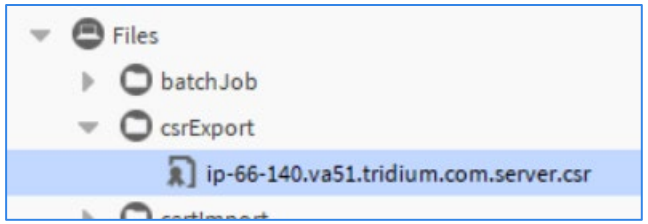
Set Time Job



Provisioning Secure Comms

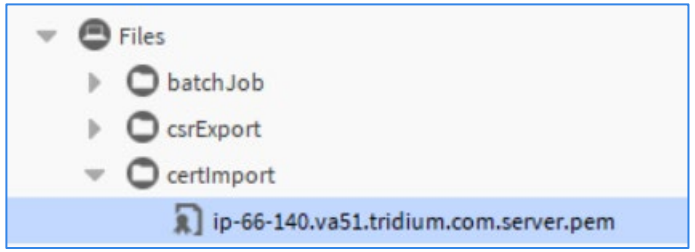


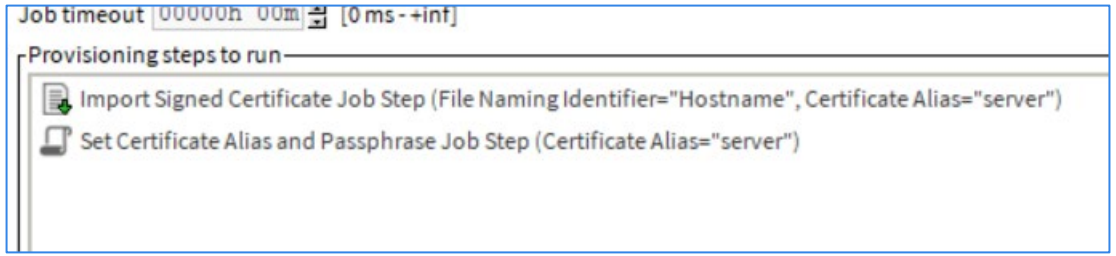




Send .csr to Certificate Authority

Receive signed certificate and put in *Files>certImport* of Provisioning station



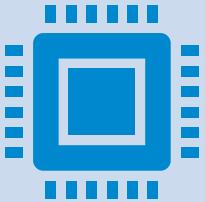


NS26

TRIDIUM

Virtual Components

Transient, on-demand components in a station that only exist when needed



System scalability is increased by using virtual components because they only exist when needed, which results in fewer persistent components and requires less heap memory.

Virtual Components

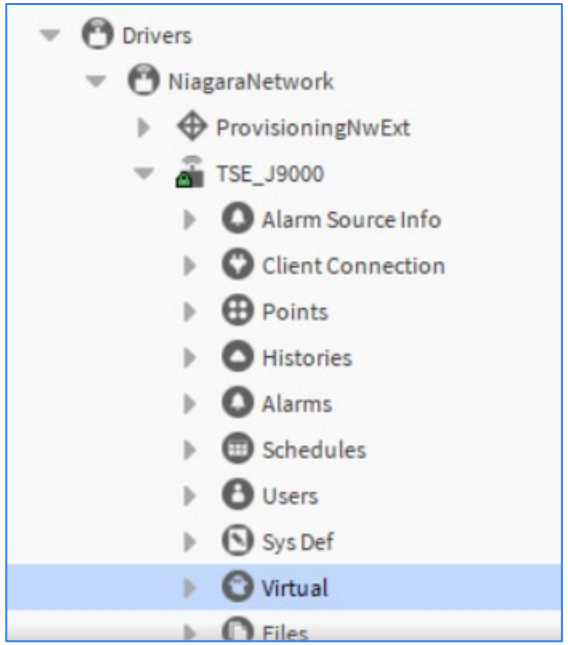
You cannot add point extensions such as history or alarm extension to virtual components.

You cannot link to virtual components, nor link the virtual components to other components.

You cannot assign Px views to virtual components.

TRIDIUM

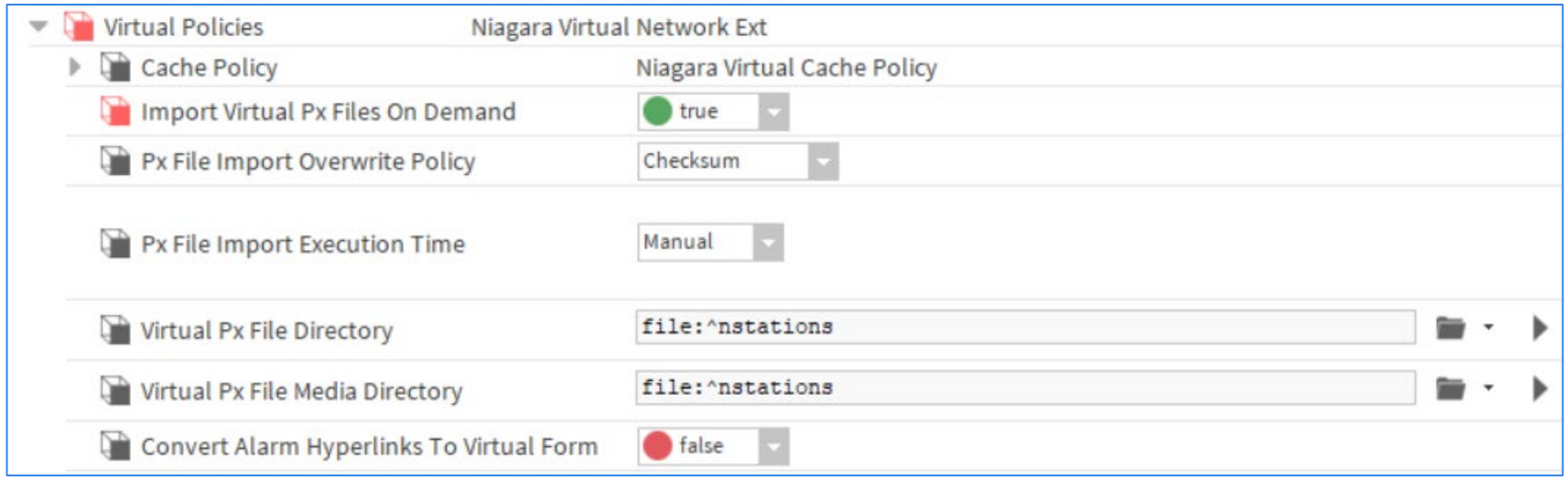
Niagara Network Virtual Components



- Must set the **Virtuals Enabled** property to **true** on the Niagara station device.
- The **Virtual Gateway** is represented by the **ghost icon** on the **Virtual** extension under the Niagara station component in the nav tree.

Niagara Virtual Policies

- For Supervisor stations that uses virtuals, this component configures cache properties and Px file policies.
- Remote stations do not require this component.



Nspace ORD Scheme

- May be used in Px view hyperlinks.
- Local station examples – optional to specify station name

  nspace:|slot:/Home → station:|slot:/Home
  nspace:<*stationName*>|slot:/Home → station:|slot:/Home

- Remote station example – nspace resolves to virtual in supervisor

  nspace:<*stationName*>|slot:/Home

  station:|slot:Drivers/NiagaraNetwork/<*stationName*>/virtual|virtual:/Home

- Used in place of station in scheme for ORDs in SystemDb

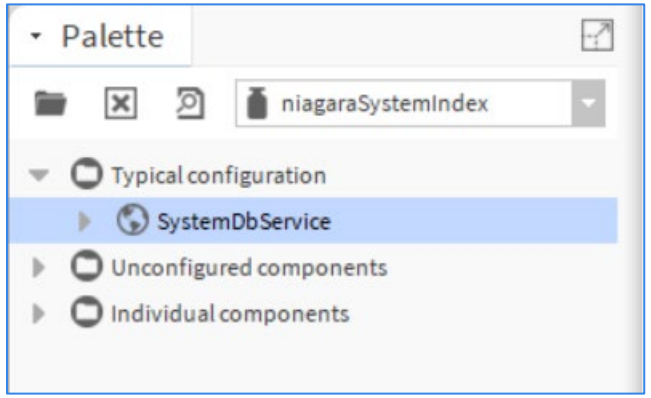
System Db

- Graph databases more easily accommodate unstructured data than relational databases.
- The System Database is a graph database running on the Niagara Supervisor station which indexes components in subordinate stations based on tags and relations.
- System Database indexing may be performed manually or on a schedule.
- The search service and hierarchies may submit NEQL queries against the System Database using the **sys:** scheme.
- 4.13+ version supports multi-tier System Database.

SystemDb Indexing

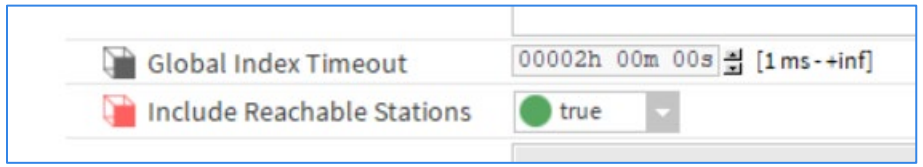
- Export – Remote stations running 4.6 or newer build may be configured to push indexing to the Supervisor.
- Import – Supervisor may be configured to pull indexing from the remote stations running 4.4 or newer build.
  - Individual – Remote stations may be configured with unique indexing rules on a per station basis.
  - Global – Uses common indexing rules for all remote stations.
- May also need to index the local Supervisor station.
- Indexing may need to be performed on a periodic basis.

System Db - Setup



- **Unconfigured services** and **components** are found in **systemDb** and **systemIndex** palettes.
- The **niagaraSystemIndex** palette contains services and components with **typical configurations**.
- Must configure the System Database Type Selection to **orientSystemDb**
- Optionally configure **Niagara Network** and **Local System Indexers**.

Multi-tier SystemDb Indexing

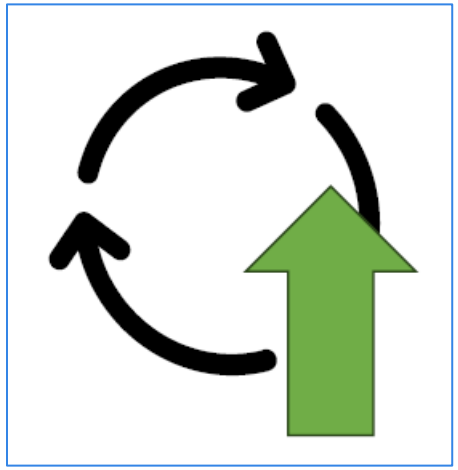


- Top level and intermediate stations must be at 4.13+ version, leaf node stations must be at 4.4+ version.
- Set Global Niagara Network Indexer's **Include Reachable Stations** property to true.

Proxy Points

- use of proxy points has historically been in a Supervisor station to display real-time values centrally in Px views
- enhanced writable virtual points has diminished the need for proxy points in a Supervisor stations.
- A proxy point is required when station control logic requires its data value as the source of a link.
- Virtual components cannot be used because they do not persist. They exist only during active subscriptions (typically from users accessing Px pages).

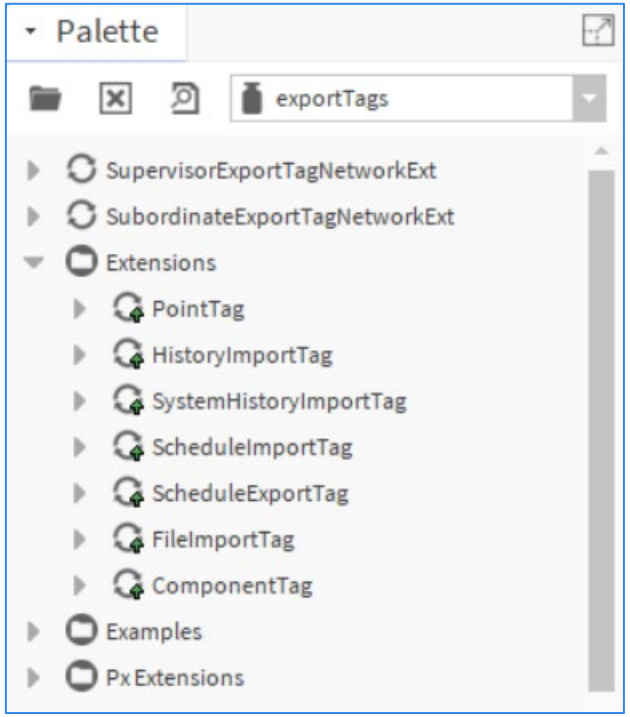
Export Tags



- When added, determine which station objects (components, histories, files) should be represented in a particular Supervisor, under its NiagaraStation component for that subordinate station
- When the export tag “Join” command is given, station components are replicated to the Supervisor
- Components accessible in Supervisor’s Niagara Network

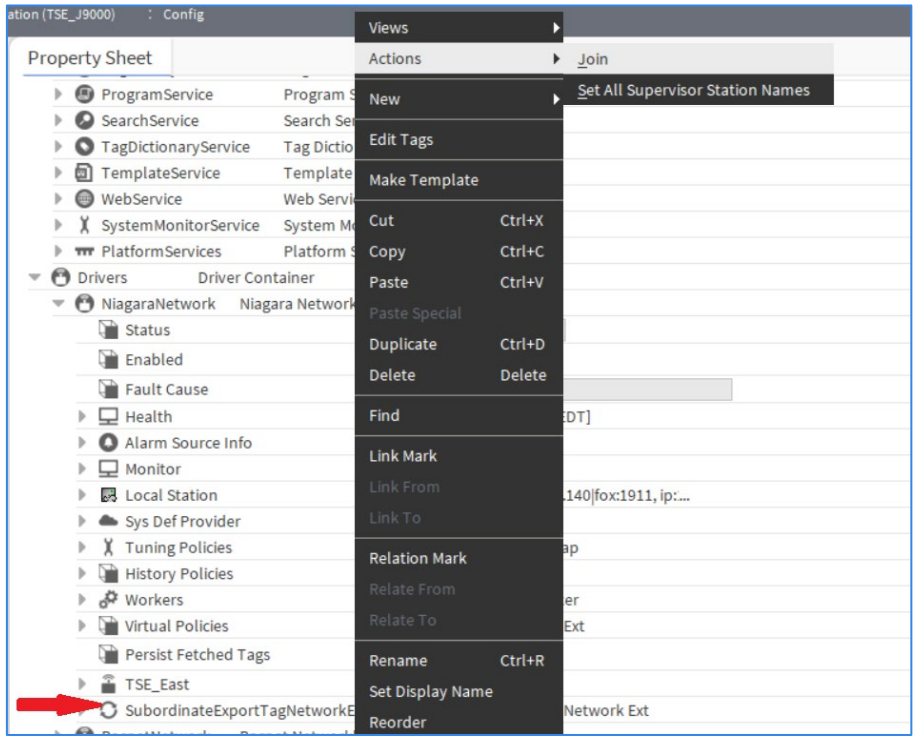
Export Tags

- SupervisorExportTagNetworkExt
- SubordinateExportTagNetworkExt



- PointTag
- PxViewTag
- HistoryImportTag
- SystemHistoryImportTag
- ScheduleImportTag
- ScheduleExportTag
- FileImportTag
- ComponentTag
- ExportTagProgram

Joining Stations



- Execute a Join action in a Controller Station
- Right-click SubordinateExportTagNetworkExt > Actions > Join
- Replicates any component with an Export Tag to the Supervisor's Niagara Network node

Questions