Technical Document

Niagara Signing Service Guide

March 3, 2023

niagara4

Niagara Signing Service Guide

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

Contents

About this guide

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

Product Documentation

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

Document Content

This document describes how to use the Signing Service primarily on a Supervisor to enable components within remote stations to generate and renew secure Certificate Signing Requests (CSR).

Document change log

Changes to this document are listed in this topic.

- Initial release publication: March 3, 2023

Related Documents

Following is a list of related guides.

- Niagara AWS Utils Guide
- Abstract MQTT Driver Guide
- Niagara Station Security Guide

Chapter 1 Signing Service

Topics covered in this chapter

- ♦ Overview (Signing Service)
- ♦ Signing Service high level types
- ♦ Signing Service high level workflow

Overview (Signing Service)

As of Niagara 4.13, the Signing Service, which is intended primarily for use on the Niagara Supervisor, allows components within remote stations to make secure Certificate Signing Requests (CSR). This is performed to obtain signed X509 certificates and also send additional requests to renew these certificates before they expire. The Signing Service fulfills these requests by returning certificates signed by a designated Certificate Authority (CA).

Use:

Any component that requires a signed certificate to fulfill its functionality may be suitable to utilize the Signing Service. This could be a component that uses a certificate as way of authentication with an external service or protocol. It could also be a component that accepts connections from outside clients and uses a server certificate to prove identity.

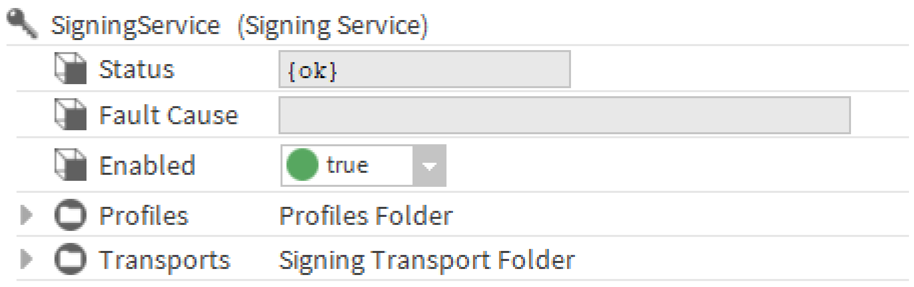
Benefits:

- Using the Signing Service removes the effort of manually generating individual X509 certificates specific for each component that requires one, and the additional task of manually signing that certificate with a CA.
- Once components are onboarded with the Signing Service, they will automatically request new certificates prior to renewal to save time on having to repeatedly recommission a fleet of controllers.
- It promotes good security practice of shorter certificate expiry periods.

This service is tailored to the need for stations to generate and renew one of many types of certificate on a rolling basis.

Signing Service high level types

The following section describes the key parts of the Signing Service.

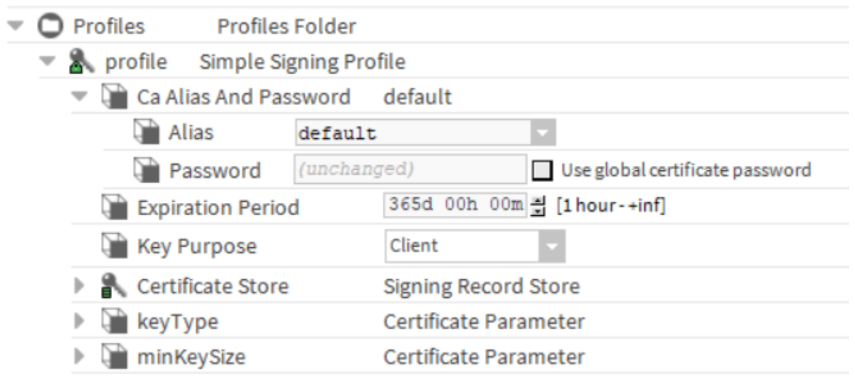


SigningService (Signing Service)

Status	{ok}
Fault Cause	
Enabled	● true
Profiles	Profiles Folder
Transports	Signing Transport Folder

Profiles

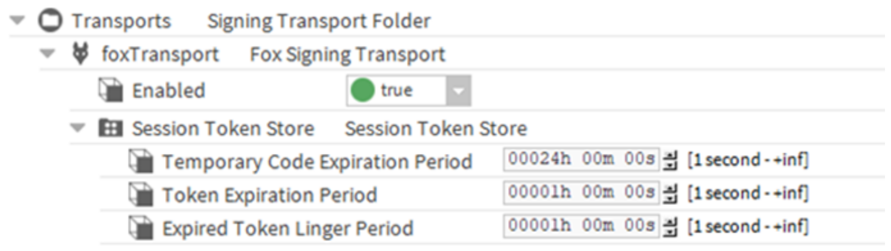
Profiles is where you define the alias and a unique or global certificate password of the CA certificate that will be used to sign all Certificate Signing Requests (CSR) associated with that profile. In addition, you may define values for various certificate fields that are applied to the signed certificate, and validate those fields within the CSR supplied by the remote station. Each profile also holds a `Certificate Store` where you may view the records of the CSR associated with that profile, including the signed certificate.



Profiles	Profiles Folder	
profile	Simple Signing Profile	
Ca Alias And Password	default	
Alias	default	
Password	(unchanged)	Use global certificate password
Expiration Period	365d 00h 00m	[1 hour - +inf]
Key Purpose	Client	
Certificate Store	Signing Record Store	
keyType	Certificate Parameter	
minKeySize	Certificate Parameter	

Transports

With **Transports**, certificate requester components can communicate and onboard with the Signing Service. This includes the mechanism by which clients authenticate to use the Signing Service, and how CSRs are associated with the relevant profile.



Transports	Signing Transport Folder	
foxTransport	Fox Signing Transport	
Enabled	true	
Session Token Store	Session Token Store	
Temporary Code Expiration Period	00024h 00m 00s	[1 second - +inf]
Token Expiration Period	00001h 00m 00s	[1 second - +inf]
Expired Token Linger Period	00001h 00m 00s	[1 second - +inf]

Fox Signing Transport

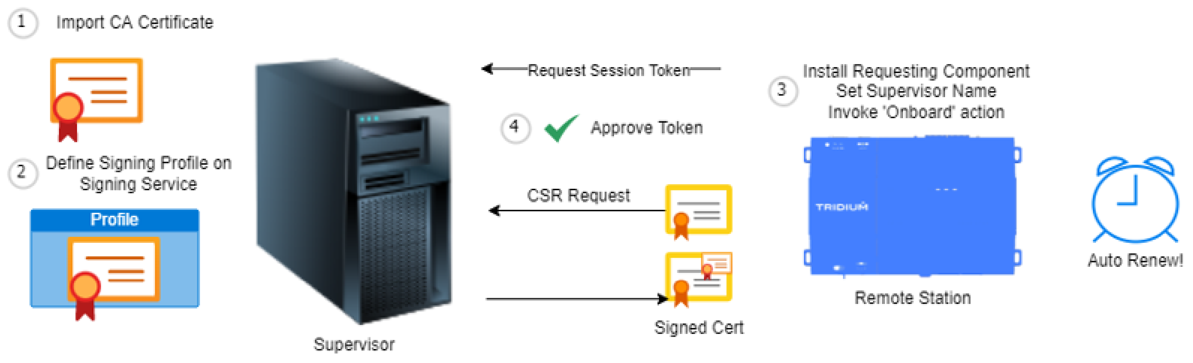
Currently, the **Fox Signing Transport** is the only existing transport. This requires a Fox connection up to the Supervisor on the remote station. Remote components must obtain a temporary session token for the purpose of onboarding, and each session token must be approved by an admin user who also designates a profile to service the request.

Signing Requester

A component on the remote station that will submit the CSR to the transport.

Signing Service high level workflow

This example assumes the usage of the **Fox Signing Transport** component.



Using Signing Service on Supervisor

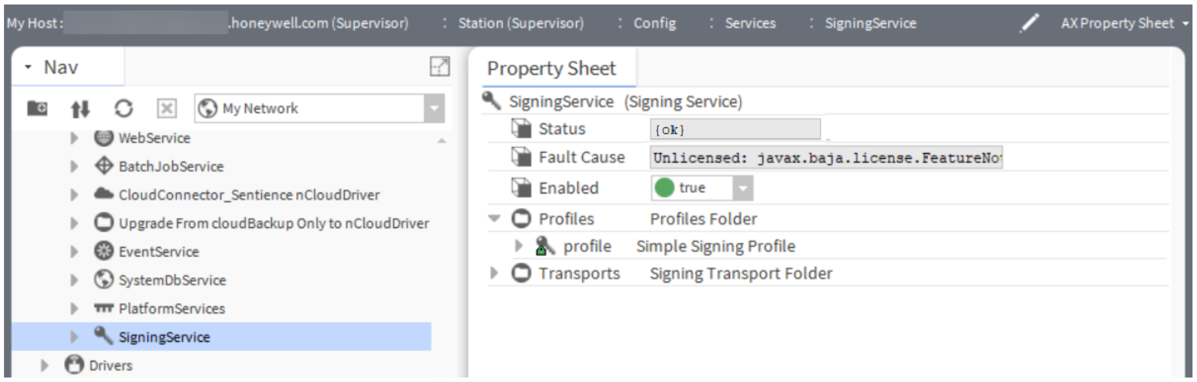
The following section describes the **Signing Service** high level workflow on a Supervisor.

Prerequisites:

- The Niagara 4.13 module versions: **`signingService-rt`**, **`signingService-ux`**, **`signingService-wb`**, and **`certSigningService`**
- At the time of the Niagara 4.13 release, the only transport mechanism available to communicate with the **Signing Service** is via the FOX transport. This requires each remote station intending to use the **Signing Service** to have a NiagaraNetwork connection to the Supervisor station.

Step 1　Import the CA certificate into the Supervisor via the **Platforms Certificate Management** view, or via the station at `slot:/Services/PlatformServices/CertManagerService`.

Step 2　Add a **Signing Service** and a **Signing Profile** to the **Services** folder.


My Host: [redacted].honeywell.com (Supervisor) : Station (Supervisor) : Config : Services : SigningService
AX Property Sheet
Nav
My Network
WebService
BatchJobService
CloudConnector_Sentience nCloudDriver
Upgrade From cloudBackup Only to nCloudDriver
EventService
SystemDbService
PlatformServices
SigningService
Drivers
Property Sheet

SigningService	(Signing Service)
Status	{ok}
Fault Cause	Unlicensed: javax.baja.license.FeatureNo
Enabled	true
Profiles	Profiles Folder
profile	Simple Signing Profile
Transports	Signing Transport Folder

Step 3　Select the alias of the imported CA, and define any certificate parameters required.

The fixed parameters allow certificate expiration and key purpose to be enforced. Other optional parameters in the palette allow enforcement of key size and key type, and certificate parameters such as the **`Distinguished Name`** fields.

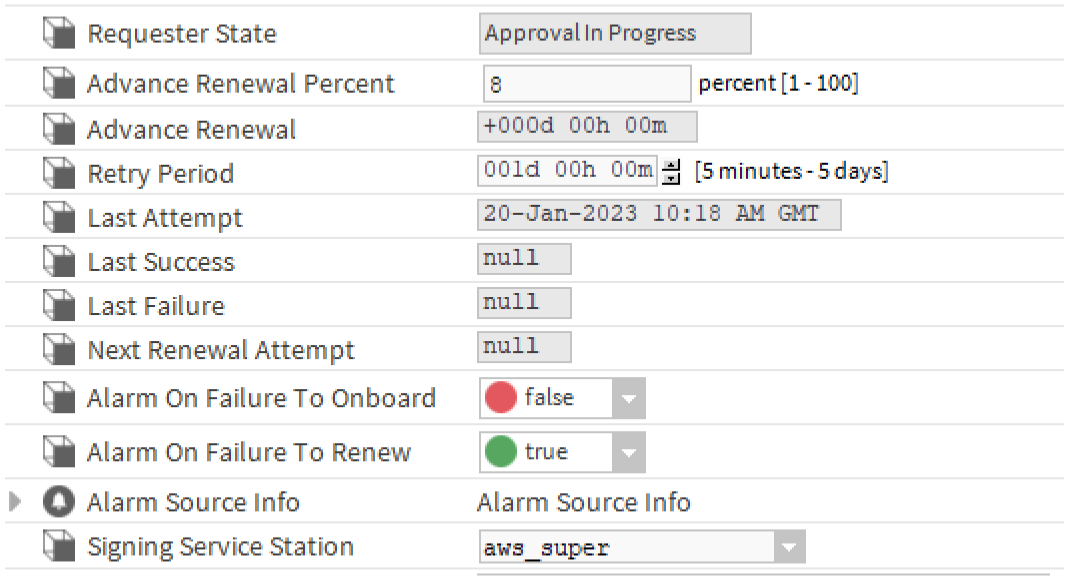
Using Signing Service on remote station

The following section describes the **Signing Service** high level workflow on a remote station.

Prerequisites:

- The Niagara 4.13 module version: **`signingService-rt`**
- At the time of the Niagara 4.13 release, the only transport mechanism available to communicate with the **Signing Service** is via the FOX transport. This requires each remote station intending to use the **Signing Service** to have a NiagaraNetwork connection to the Supervisor station.

Step 1　Install a requesting component on the station and select the name of the Supervisor that contains the service in the Signing Service station property.



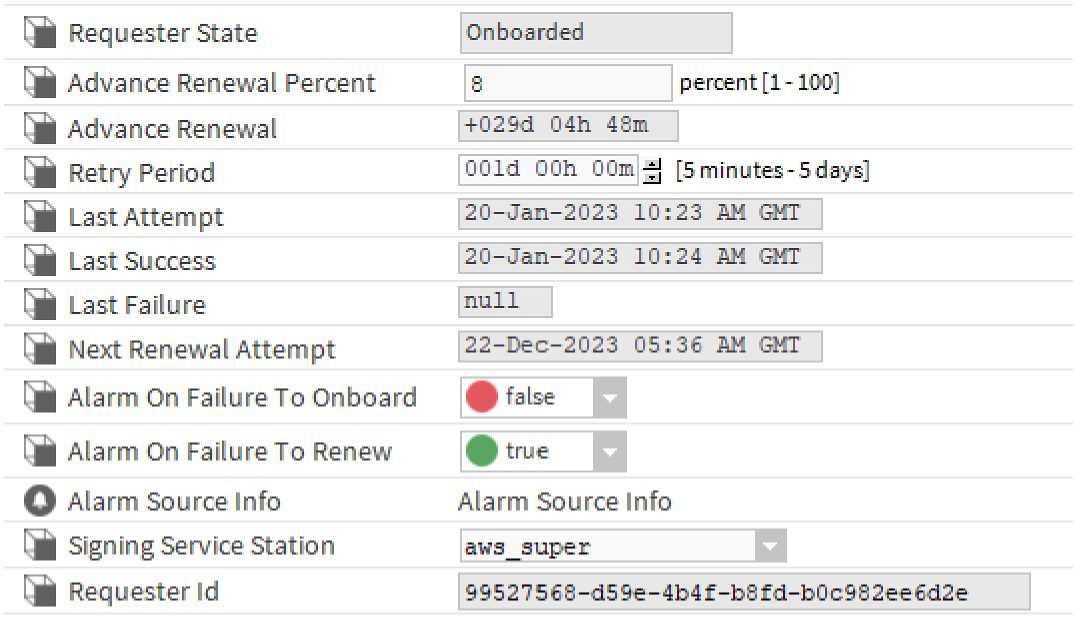
Requester State	Approval In Progress
Advance Renewal Percent	8 percent [1 - 100]
Advance Renewal	+000d 00h 00m
Retry Period	001d 00h 00m [5 minutes - 5 days]
Last Attempt	20-Jan-2023 10:18 AM GMT
Last Success	null
Last Failure	null
Next Renewal Attempt	null
Alarm On Failure To Onboard	🔴 false
Alarm On Failure To Renew	🟢 true
Alarm Source Info	Alarm Source Info
Signing Service Station	aws_super

Step 2 You can now invoke the **Onboard** action and enter a comment to help the admin user on the Supervisor approve your request.

If successful, the status will change to `Approval In Progress`. The **Signing Requester** will now communicate with the **Signing Service Transport** and generate a session token.

Step 3 On the Supervisor, an admin user navigates to the **Session Token Store** for the **Fox Signing Transport** and inspects the request metadata and comment, then decides to approve or reject the request.

Once an admin user of the Signing Service has approved your session token, the requesting component will generate and store a CSR locally, and submit this to the Signing Service. It will receive back a signed certificate, which is then stored in the main station's **Certificate Management→User Key Store**. The system will attempt to automatically renew prior to expiry.



Requester State	Onboarded
Advance Renewal Percent	8 percent [1 - 100]
Advance Renewal	+029d 04h 48m
Retry Period	001d 00h 00m [5 minutes - 5 days]
Last Attempt	20-Jan-2023 10:23 AM GMT
Last Success	20-Jan-2023 10:24 AM GMT
Last Failure	null
Next Renewal Attempt	22-Dec-2023 05:36 AM GMT
Alarm On Failure To Onboard	🔴 false
Alarm On Failure To Renew	🟢 true
Alarm Source Info	Alarm Source Info
Signing Service Station	aws_super
Requester Id	99527568-d59e-4b4f-b8fd-b0c982ee6d2e

NOTE: The exact steps may differ with future transport implementations. See specific component documentation for these.

Renewal will be automatically attempted prior to the certificate's expiration according to the **`Advance Re-newal Percent`** value. This defaults to 8% so that an annual certificate will automatically attempt to renew approximately one month in advance of expiration. The existing certificate will now be replaced in the **User Key Store**. Approval is not required for renewal as the existing certificate is used as authentication with the service. However, if renewal fails before the existing certificate expires, you will need to manually repeat the **Onboard** action for the component. You can also manually invoke the **Renew** action to force an attempt.

An alarm will be generated on the local station for certificates that have failed a renewal attempt. The component will continue to schedule new renewal attempts in the case of a failure according to the **`Retry Period`**.

Some possibly causes for onboarding failure include:

- The request was rejected by the admin user on the Signing Service.
- The request was not approved in time.
- The CSR failed validation due to an incorrect value.
- A configuration error in the Signing Profile, for example, an incorrect CA password.
- **`Signing Service Station`** property was not populated.
- Communications failure with the Supervisor. Is the Niagara network connection functioning?
- An active certificate with matching alias exists in the local key store during onboarding. In this instance, the certificate will be overwritten only if it has expired, and the password and serial number for the certificate matches.

Chapter 2 Components, views and windows

Topics covered in this chapter

The user interface includes components, views and windows, which provide the means for communicating with the system.

The Help topics include context sensitive information about each component and view, as well as information about individual windows.

Components

Components include services, folders and other model building blocks associated with a module. You may drag them to a property or wire sheet from a palette.

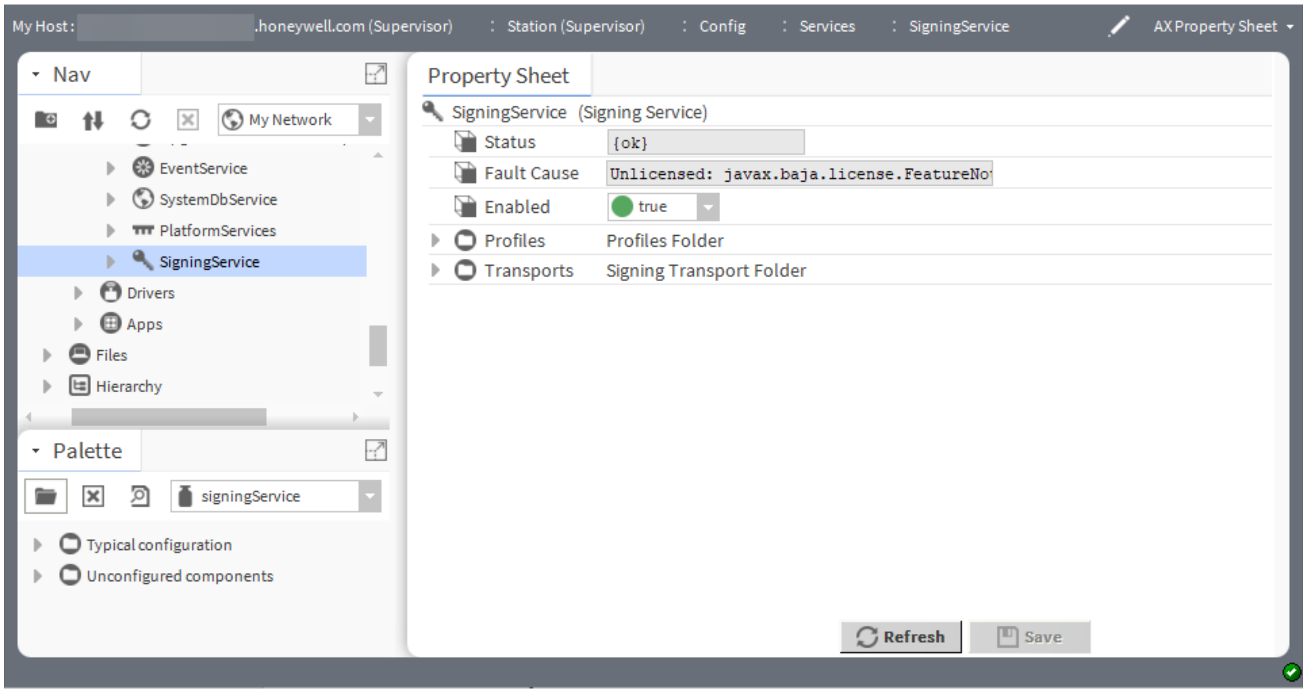
Descriptions included in the following topics appear as context-sensitive help topics when accessed by:

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

Signing Service (signingService-SigningService)

As of Niagara 4.13, the **Signing Service**, intended primarily for use on a Niagara Supervisor, allows components within remote stations to make secure Certificate Signing Requests (CSR) to obtain signed X509 certificates, then additional requests to renew these certificates before they expire. The **Signing Service** fulfills these requests by returning certificates signed by a designated Certificate Authority (CA).

This component is used to fulfill the requests of queuing incoming CSR requests from signing transports, and delegating the request to the appropriate signing profile.

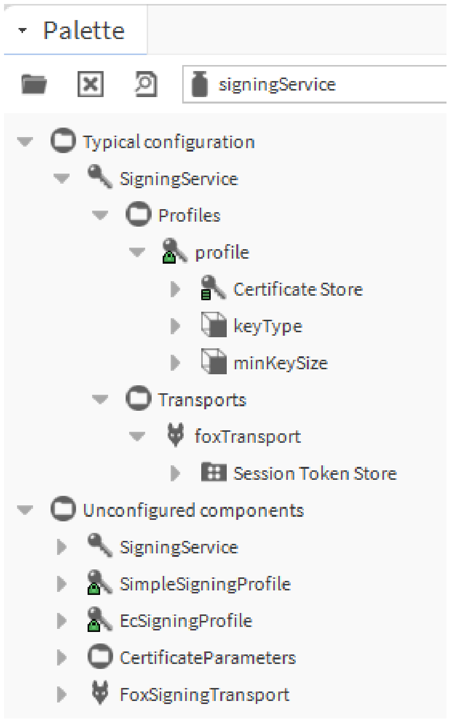


SigningService (Signing Service)

Status	{ok}
Fault Cause	Unlicensed: javax.baja.license.FeatureNo
Enabled	true
Profiles	Profiles Folder
Transports	Signing Transport Folder

To use the **Signing Service** component, drag it from the `signingService` palette to the **Config→Services** folder in the Nav tree.

You also find an example of a **Signing Service** pre-configured with an example signing profile and transport in the **Typical configuration** folder. There is also a plain unconfigured service in the **Unconfigured components** folder.



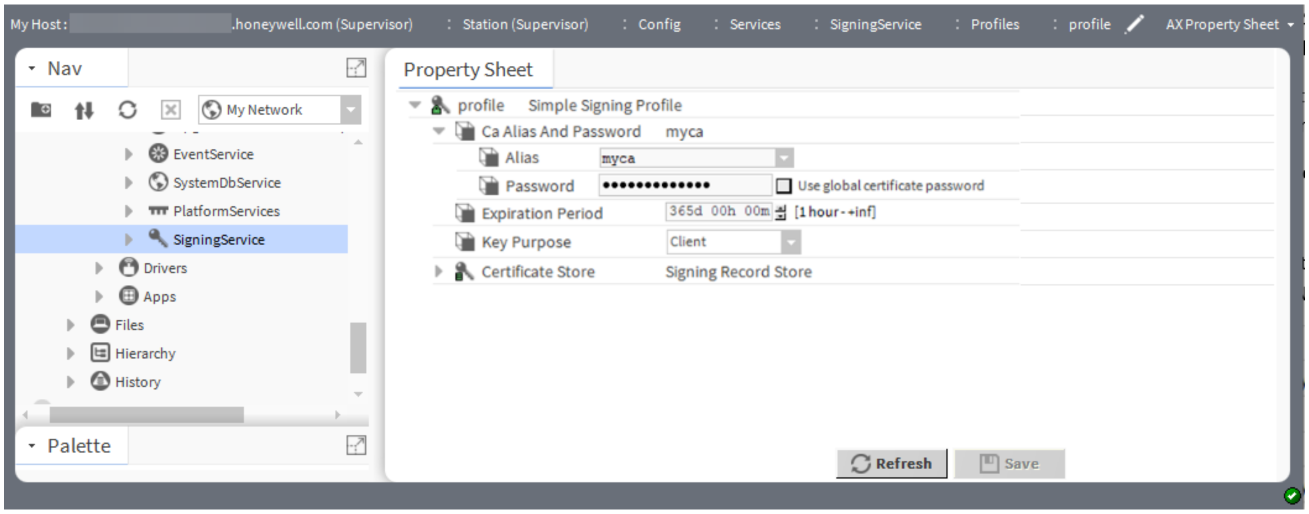
To access the properties, expand **Config→Services** and click **SigningService**

Property	Value	Description
Enabled	`true` or `false` (defaults to `true`)	When `true`, all **Signing Service** functionality is enabled. While `false`, all submitted CSR will fail to process, and any queued CSR will be rejected.
Profiles	folder	Holds signing profile instances. See "*Signing Service High Level Types*" for more details.
Transports	folder	Holds signing profile instances. See "*Signing Service High Level Types*" for more details.

Simple Signing Profile (signingService-SimpleSigningProfile)

A signing service profile is where you define the alias and password of the CA certificate that will be used to sign all Certificate Signing Requests (CSRs) associated with that profile. It is also where you can define values for various certificate fields that are applied to the signed certificate, and to validate those fields within the CSR supplied by the remote station.

Each profile also holds a **Certificate Store** where you can view the records of a CSR associated with that profile, including the signed certificate.



profile	Simple Signing Profile
Ca Alias And Password	myca
Alias	myca
Password	••••••••••••• ☐ Use global certificate password
Expiration Period	365d 00h 00m [1 hour-+inf]
Key Purpose	Client
Certificate Store	Signing Record Store

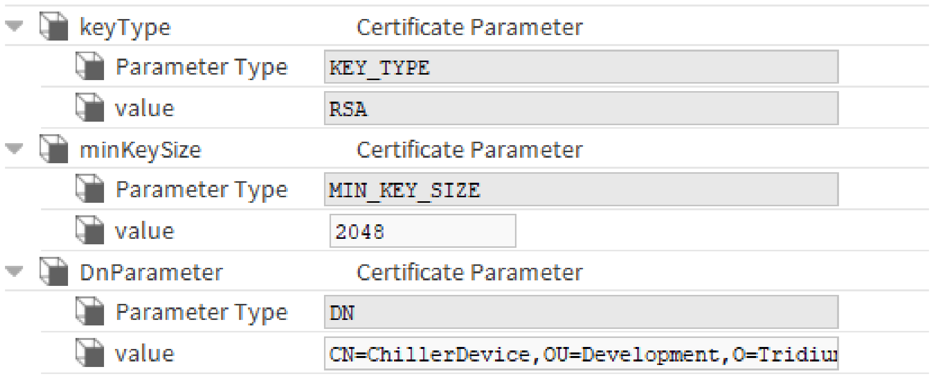
The **Simple Signing Profile** component is available in the **SigningService** palette. The **Typical configuration** folder contains a signing profile predefined for signing RSA certificates with a minimum key size. There is also a plain unconfigured profile without any parameters in the **Unconfigured components** folder.

For a signing profile to fulfill requests, the following prerequisites must be met:

- The CA certificate has been imported into the station or platform certificate manager.
- The alias and password of the CA have been set in the Signing Profile's **Ca Alias And Password** property. A password is mandatory. You can choose to define a unique password or optionally you can select the **Use global certificate password** option.

You may set fixed certificate values such as the expiration period of the signed certificates and the key purpose. These values override those set in the incoming CSR from the requesting component.

You have also the option to drop one to many certificate parameter objects underneath the signing profile. These allow you to define default values for any fields that may be unspecified in the incoming CSR. A validation error will be thrown if the incoming CSR defines a different value, causing the signing request to be rejected. Parameters include key type and minimum key size, and certificate parameters such as the **Distinguished Name** fields.


keyType	Certificate Parameter
Parameter Type	KEY_TYPE
value	RSA
minKeySize	Certificate Parameter
Parameter Type	MIN_KEY_SIZE
value	2048
DnParameter	Certificate Parameter
Parameter Type	DN
value	CN=ChillerDevice,OU=Development,O=Tridiu

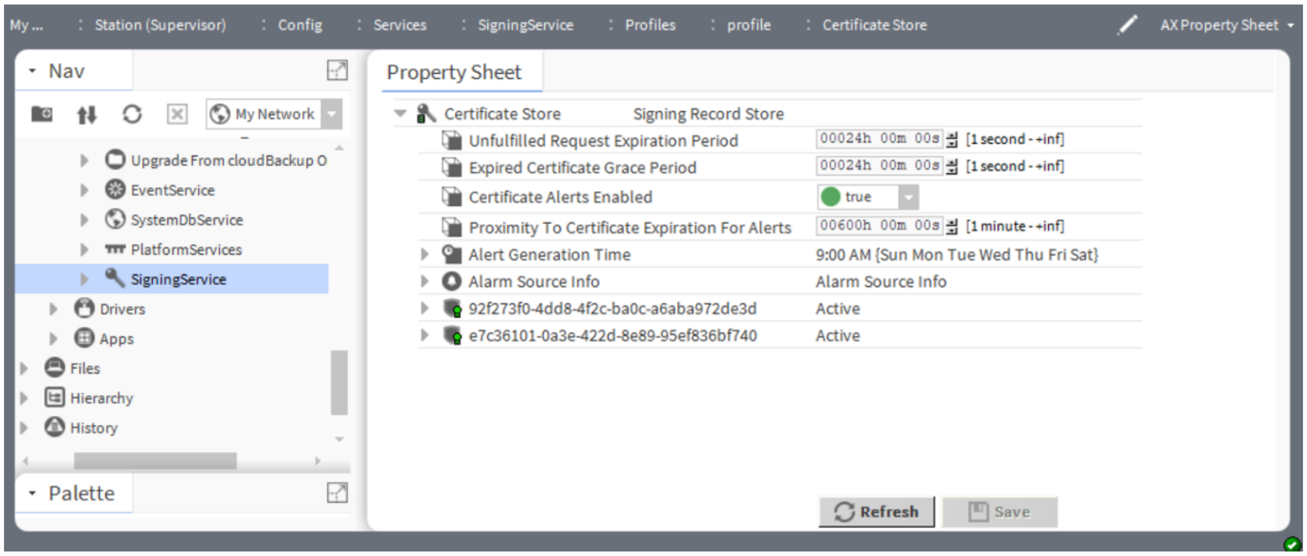
Property	Value	Description
Ca Alias And Password	additional mandatory properties	Specifies the alias and password of the CA certificate that will be used to sign incoming CSR for this profile.
Expiration Period	days, hours, minutes	Defines the expiration period that will be applied to the signed certificate, overriding any specified in the CSR. The period is applied to the time at which the CSR is processed by the service.
Key Purpose	drop-down menu	Specifies the purpose certificate extensions that will be applied to the signed certificate, overriding any specified in the CSR. Values are `Client`, `Server`, `Cert Authority` and `Code Signing`.
Certificate Store	additional properties	Holds the records of CSR associated with that profile, including the signed certificate.

Signing Record Store (signingService-SigningRecordStore)

The certificate **Signing Record Store** holds and manages the records of outstanding and fulfilled CSR requests for the parent signing profile.

When CSRs are submitted and processed, they will appear underneath the store as **Certificate Signing Record** instances specific to the requester Id of the requesting component.

The **Signing Record Store** is available in the **`signingService`** palette. To access the properties, expand **Config→Services→SigningService→Profiles→Simple Signing Profile→Certficate Store**.


My ... : Station (Supervisor) : Config : Services : SigningService : Profiles : profile : Certificate Store

AX Property Sheet

Nav

My Network

- Upgrade From cloudBackup O
- EventService
- SystemDbService
- PlatformServices
- SigningService
- Drivers
- Apps
- Files
- Hierarchy
- History

Palette

Property Sheet

Certificate Store	Signing Record Store
Unfulfilled Request Expiration Period	00024h 00m 00s [1 second - +inf]
Expired Certificate Grace Period	00024h 00m 00s [1 second - +inf]
Certificate Alerts Enabled	true
Proximity To Certificate Expiration For Alerts	00600h 00m 00s [1 minute - +inf]
Alert Generation Time	9:00 AM {Sun Mon Tue Wed Thu Fri Sat}
Alarm Source Info	Alarm Source Info
92f273f0-4dd8-4f2c-ba0c-a6aba972de3d	Active
e7c36101-0a3e-422d-8e89-95ef836bf740	Active

Property	Value	Description
Unfulfilled Request Expiration Period	hours, minutes, seconds	Defines the period after which a `Pending` record will be deleted.
Expired Certificate Grace Period	hours, minutes, seconds	Defines the period after which an `Expired` certificate will continue to service renewal requests.
Certificate Alerts Enabled	`true` or `false`	Set to `true` enables alerts for certificates that are due or have already expired.
Proximity To Certificate Expiration For Alerts	hours, minutes, seconds	Defines the period in which alerts shall be generated for certificates that are due to expire.
Alert Generation Time	additional properties	Contains time trigger settings for timings of alert checks.
Alarm Source Info	additional properties	Contains properties for configuring certificate alerts.

Actions

Update Record States: Forces the update of the status of each record in the store.

Generate Alerts: Forces an instant generation of new certificate alerts.

Requester certificate rules

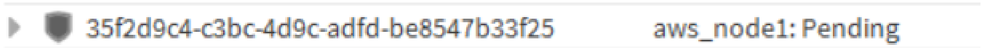
The rules governing requester certificates are as follows:

- The requester Id is unique to each requesting component. There may be several on the same station, or from different stations.
- Each requester can be associated with only a single signing record.
- The certificate record is `Pending` while the CSR is queued until the request is processed. Then, it becomes `Rejected` or `Active` as the certificate is signed and stored.
- Any pending request that is unfulfilled for the **`Unfulfilled Request Expiration Period`** is automatically removed. The requester must **Onboard** to the signing service once again.
- An active certificate record is required for the requesting component to renew its current certificate. This should be attempted automatically in advance of expiry but can also be requested manually.

- The certificate record will continue to service renewal attempts up to the expiry time of the certificate, plus the **`Expired Certificate Grace Period`**.
- After the certificate expiry plus grace period has passed, the expired certificate can no longer service renewal requests and is automatically removed. The requester must **Onboard** to the signing service once again.

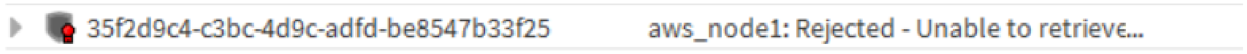
The status for each record in the store is indicated by the icon and status text. The possible states are:

- **Pending** — The CSR is queued and waiting to be fulfilled.



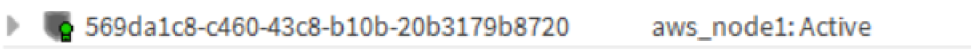
35f2d9c4-c3bc-4d9c-adfd-be8547b33f25 aws_node1: Pending

- **Rejected** — The CSR failed to be successfully fulfilled. Check the records message property for cause.

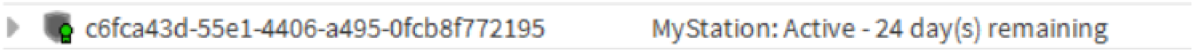


35f2d9c4-c3bc-4d9c-adfd-be8547b33f25 aws_node1: Rejected - Unable to retrieve...

- **Complete/Active** — The CSR was fulfilled, and the certificate is valid.

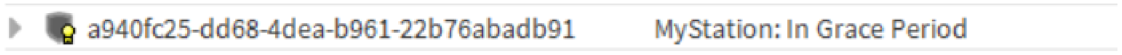


569da1c8-c460-43c8-b10b-20b3179b8720 aws_node1: Active



c6fca43d-55e1-4406-a495-0fcb8f772195 MyStation: Active - 24 day(s) remaining

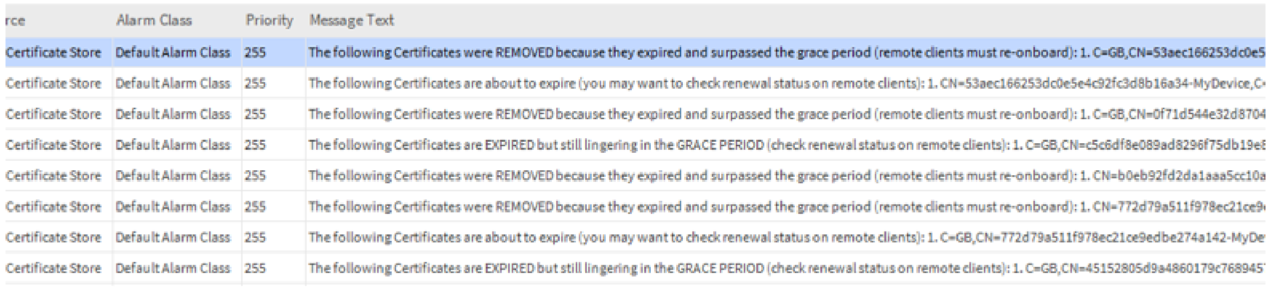
- **Complete/In Grace Period** — The CSR was previously fulfilled, the certificate has passed its expiry period, however is in the grace period, so it will continue to be used to service renewal requests until the **`Expired Certificate Grace Period`** has passed. The grace period is applied from the time of certificate expiry.



a940fc25-dd68-4dea-b961-22b76abadb91 MyStation: In Grace Period

The **Certificate Store** also has the ability to generate alerts for various certificate states. Certificates are grouped together into single alerts for each of the following states. They are grouped with those that have also entered that state since the last alert check:

- Those near their expiration; the proximity is configurable.
- Those expired but still in the grace period; this may indicate that auto-renew on the requesting component is failing.
- Those that have expired and been removed since the last alert check.

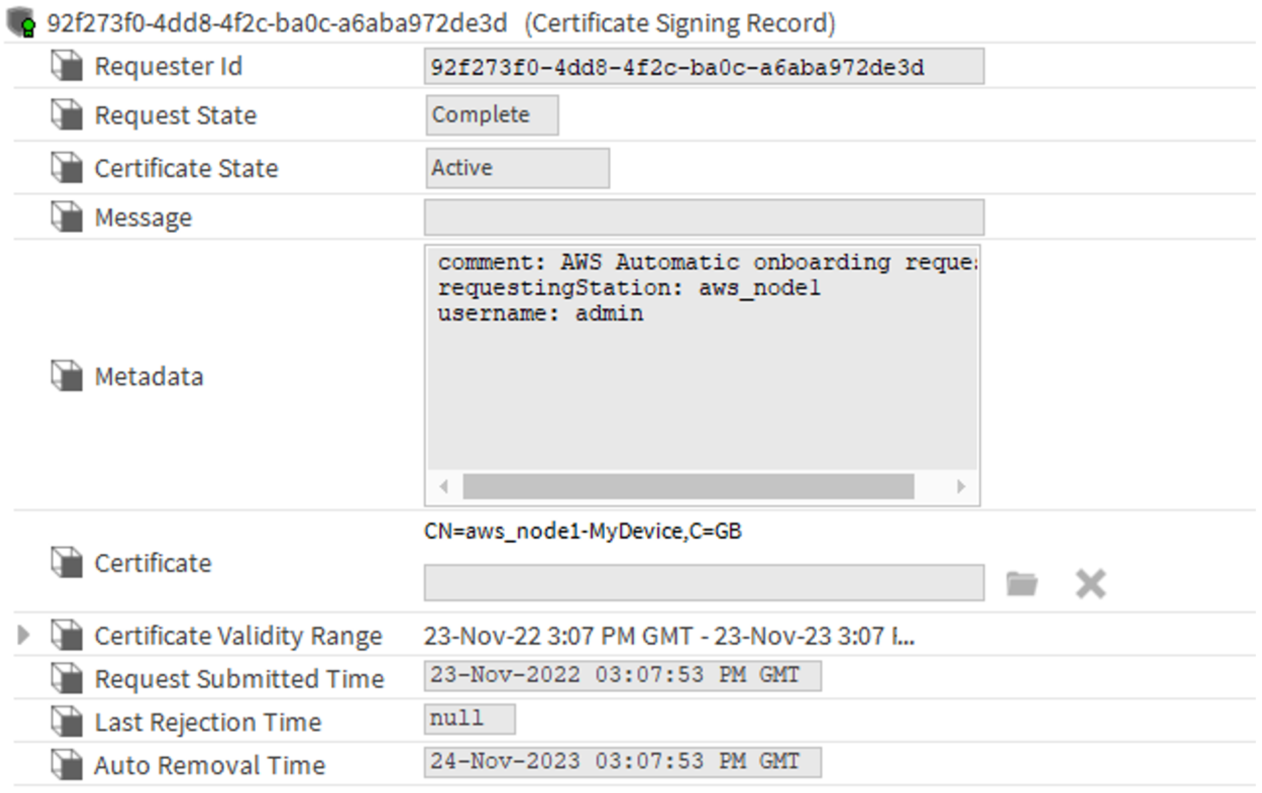


rce	Alarm Class	Priority	Message Text
Certificate Store	Default Alarm Class	255	The following Certificates were REMOVED because they expired and surpassed the grace period (remote clients must re-onboard): 1. C=GB,CN=53aec166253dc0e5
Certificate Store	Default Alarm Class	255	The following Certificates are about to expire (you may want to check renewal status on remote clients): 1. CN=53aec166253dc0e5e4c92fc3d8b16a34-MyDevice,C
Certificate Store	Default Alarm Class	255	The following Certificates were REMOVED because they expired and surpassed the grace period (remote clients must re-onboard): 1. C=GB,CN=0f71d544e32d8704
Certificate Store	Default Alarm Class	255	The following Certificates are EXPIRED but still lingering in the GRACE PERIOD (check renewal status on remote clients): 1. C=GB,CN=c5c6df8e089ad8296f75db19e
Certificate Store	Default Alarm Class	255	The following Certificates were REMOVED because they expired and surpassed the grace period (remote clients must re-onboard): 1. CN=b0eb92fd2da1aaa5cc10a
Certificate Store	Default Alarm Class	255	The following Certificates were REMOVED because they expired and surpassed the grace period (remote clients must re-onboard): 1. CN=772d79a511f978ec21ce9
Certificate Store	Default Alarm Class	255	The following Certificates are about to expire (you may want to check renewal status on remote clients): 1. C=GB,CN=772d79a511f978ec21ce9edbe274a142-MyDe
Certificate Store	Default Alarm Class	255	The following Certificates are EXPIRED but still lingering in the GRACE PERIOD (check renewal status on remote clients): 1. C=GB,CN=45152805d9a4860179c768945

Certificate Signing Record (signingService-CertificateSigningRecord)

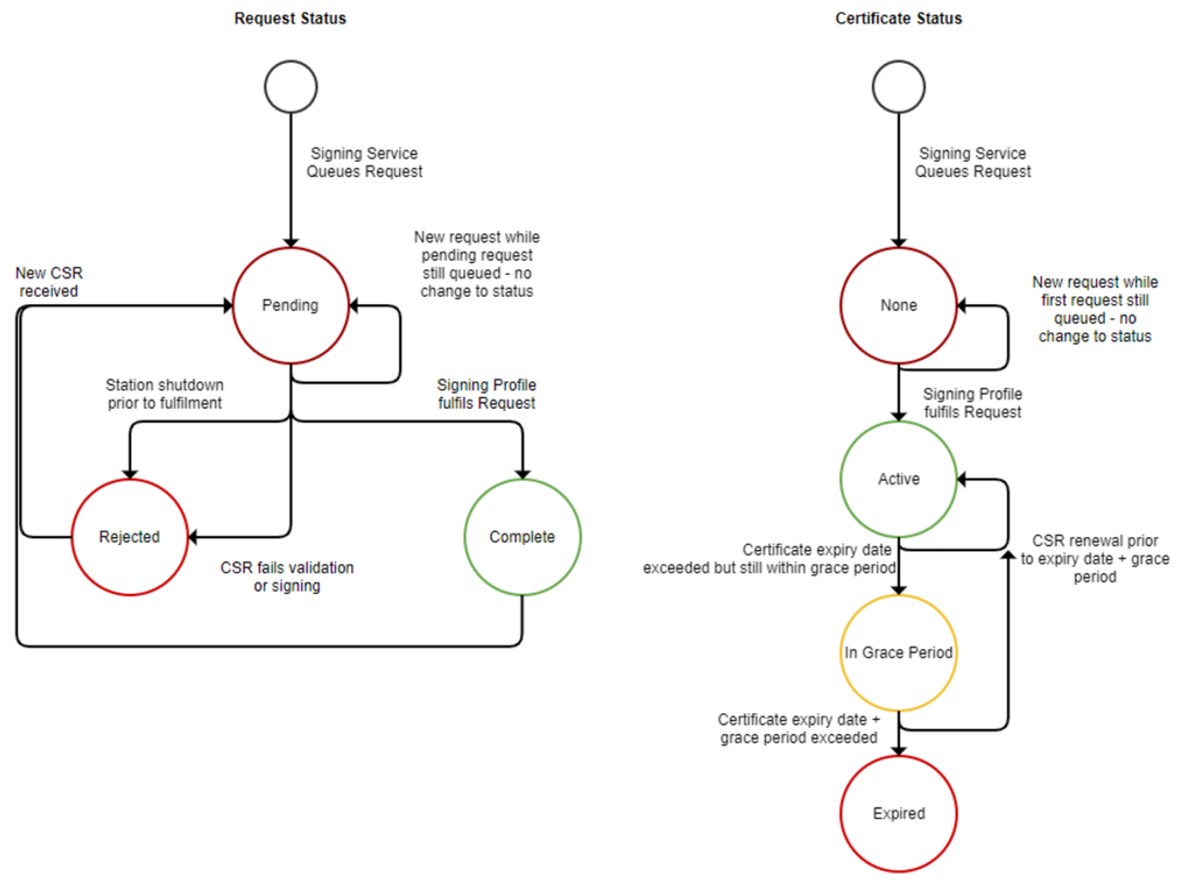
This component represents an individual signing record associated with a requesting component. It encapsulates the status of the signing request and the status of the certificate.

Each record displays the certificate itself, the metadata captured during onboarding to the Signing Service for identification, and various date/times key to the certificate's lifecycle.


92f273f0-4dd8-4f2c-ba0c-a6aba972de3d (Certificate Signing Record)
Requester Id: 92f273f0-4dd8-4f2c-ba0c-a6aba972de3d
Request State: Complete
Certificate State: Active
Message:
Metadata:
```
comment: AWS Automatic onboarding reque
requestingStation: aws_node1
username: admin
```
Certificate: CN=aws_node1-MyDevice,C=GB
Certificate Validity Range: 23-Nov-22 3:07 PM GMT - 23-Nov-23 3:07 I...
Request Submitted Time: 23-Nov-2022 03:07:53 PM GMT
Last Rejection Time: null
Auto Removal Time: 24-Nov-2023 03:07:53 PM GMT

Property	**Value**	**Description**
Requester Id	read-only	Displays a unique Id that identifies the requesting component.
Request State	read-only	Displays the status of the request (`Pending`, `Rejected`, `Complete`)
Certificate State	read-only	Displays the status of the certificate (`None`, `Active`, `In Grace Period`, `Expired`)
Message	read-only	Displays message to identify reason for rejection or failure to onboard.
Metadata	read-only	Displays extra details regarding the requester from the on-boarding process, including station name and name of the user who submitted the request.
Certificate	read-only	Displays the certificate distinguished name.
Certificate Validity Range	read-only	Specifies the not-before and not-after times for the signed certificate.
Request Submitted Time	read-only	Specifies the time at which the CSR was submitted to the sign-ing service.
Last Rejection Time	read-only	Specifies the time at which the CSR last failed to be fulfilled.
Auto Removal Time	read-only	Specifies the time at which the record will automatically be removed.

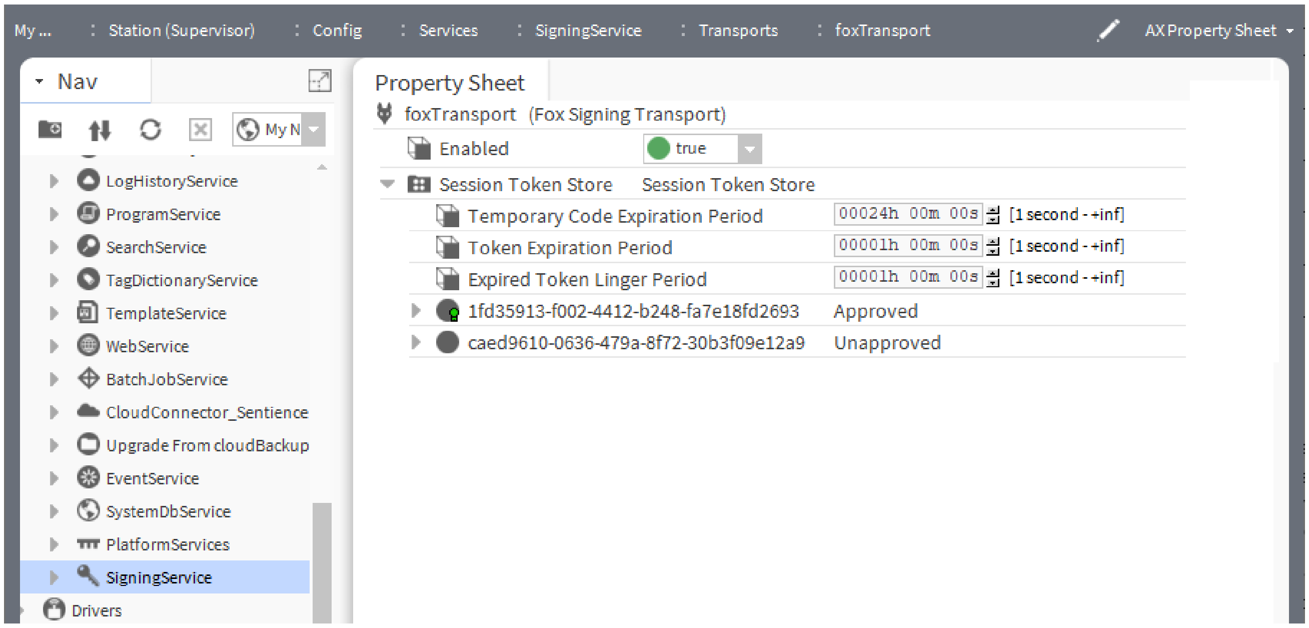
The status workflow is as follows:


Request Status
Signing Service Queues Request
New request while pending request still queued - no change to status
New CSR received
Pending
Station shutdown prior to fulfilment
Signing Profile fulfils Request
Rejected
Complete
CSR fails validation or signing
Certificate Status
Signing Service Queues Request
None
New request while first request still queued - no change to status
Signing Profile fulfils Request
Active
Certificate expiry date exceeded but still within grace period
CSR renewal prior to expiry date + grace period
In Grace Period
Certificate expiry date + grace period exceeded
Expired

Fox Signing Service (signingService-FoxSigningService)

The **Fox Signing Transport** is a signing transport that allows remote stations to utilize the Signing Service via their Fox Niagara network connection. A dedicated Fox Signing message channel provides a secure and trusted communication mechanism between the Signing Service and the remote requesting component for exchange of CSR and signed certificates.

Each signing transport is responsible for onboarding remote components into the service. The **Fox Signing Transport** requires the remote component to request a temporary session token, which must be approved by an admin user before the component can submit a CSR to the service. These tokens are stored and approved in the **Session Token Store**. Session tokens are not required for certificate renewal, only for initial **Onboarding** with the service.



foxTransport (Fox Signing Transport)

Enabled	true	
Session Token Store	Session Token Store	
Temporary Code Expiration Period	00024h 00m 00s	[1 second - +inf]
Token Expiration Period	00001h 00m 00s	[1 second - +inf]
Expired Token Linger Period	00001h 00m 00s	[1 second - +inf]
1fd35913-f002-4412-b248-fa7e18fd2693	Approved	
caed9610-0636-479a-8f72-30b3f09e12a9	Unapproved	

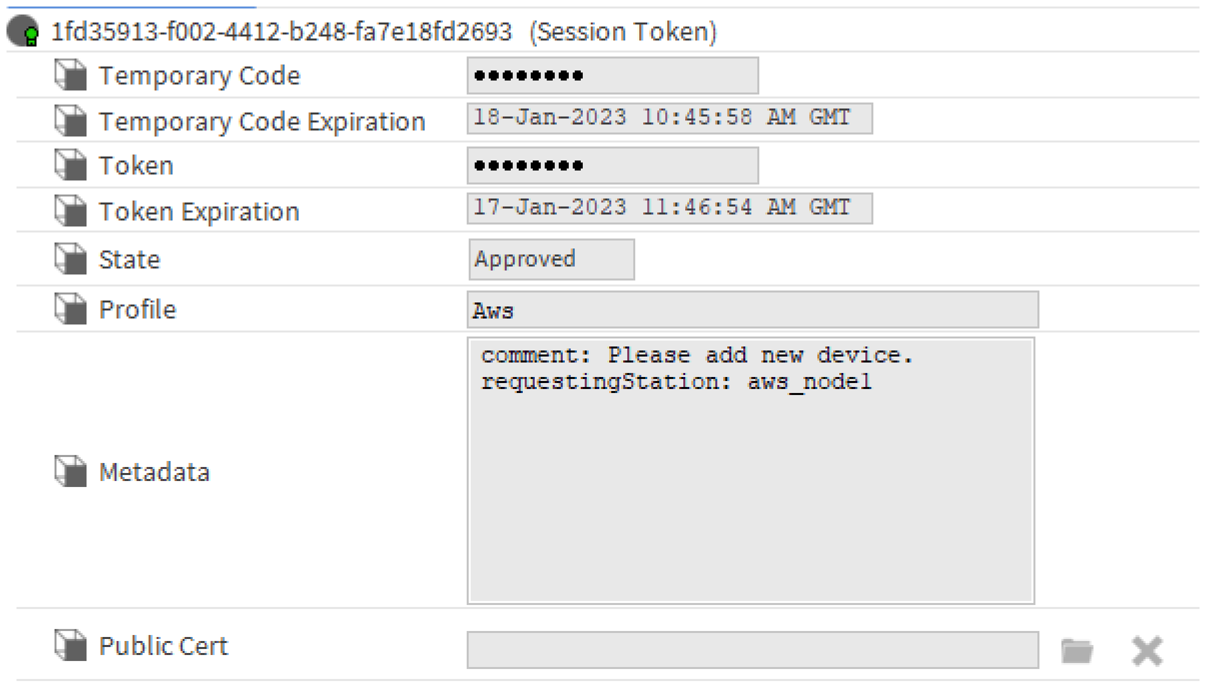
The **Fox Signing Transport** component is available in the `signingService` palette. To access the properties, expand **Config→Services→SigningService→Transports→foxTransport (Fox Signing Transport)**.

Property	Value	Description
Enabled	`true` or `false` (defaults to `true`)	Enables the Fox signing message channel communications.
Session Token Store	additional properties	Contains session tokens, which permit a requesting component to onboard with the Signing Service.

Session Token (signingService-SessionToken)

Each **Session Token** represents a requesting component which has attempted to onboard with the Signing Service to submit a CSR.

The name of the token is the unique requested Id. Once the CSR has been fulfilled, the token's certificate property will be populated. Session tokens will be automatically removed some time after the CSR has been fulfilled.


1fd35913-f002-4412-b248-fa7e18fd2693 (Session Token)

Temporary Code	●●●●●●●●●
Temporary Code Expiration	18-Jan-2023 10:45:58 AM GMT
Token	●●●●●●●●●
Token Expiration	17-Jan-2023 11:46:54 AM GMT
State	Approved
Profile	Aws
Metadata	comment: Please add new device. requestingStation: aws_node1
Public Cert	

Property	Value	Description
Temporary Code	read-only	Displays the secret temporary code value. This enables the requesting component to check on the status of its token.
Temporary Code Expiration	read-only	Displays the date and time at which the code expires.
Token	read-only	Displays the secret token value which enables the requesting component to submit a CSR.
Token Expiration	read-only	Displays the date and time at which the token expires.
State	read-only	Displays the status of the token: `Unapproved/Approved/Rejected`
Profile	read-only	Displays the name of the profile that was associated with the approved token.
Metadata	read-only	Lists metadata values submitted by the requesting component, which should help inform the admin user on the decision to approve or reject the token.
Public Cert	read-only	Displays the distinguished name of the certificate. Appears when the CSR has been fulfilled.

Actions

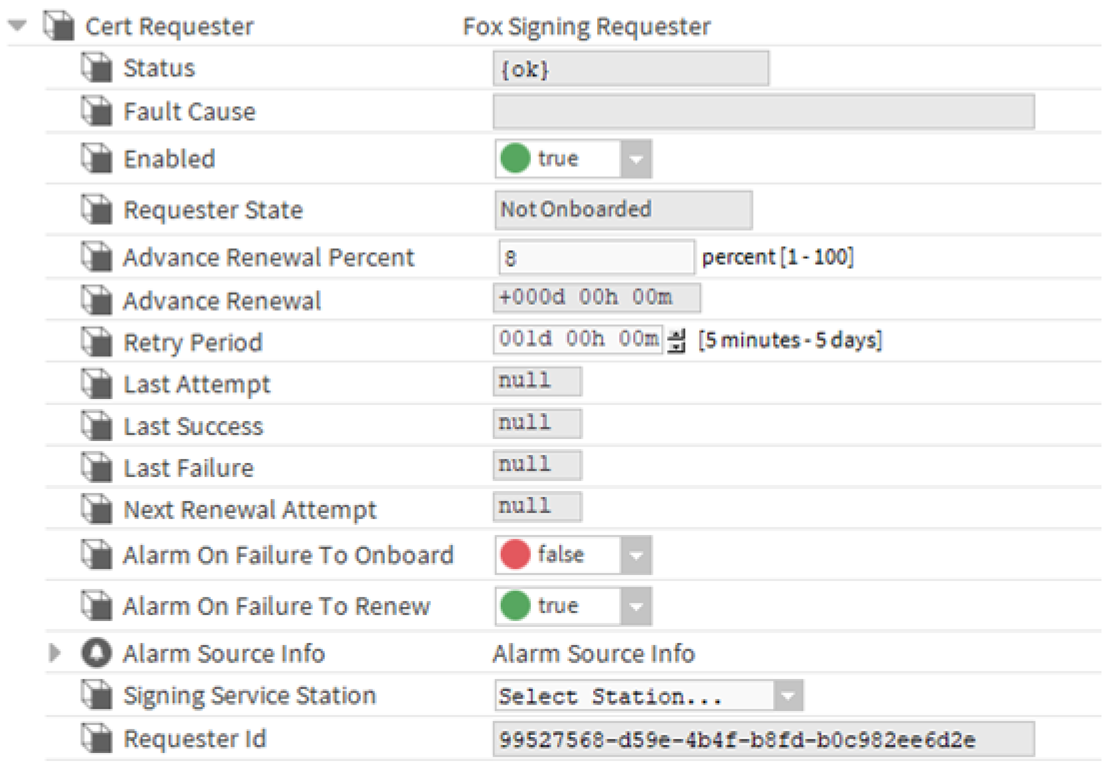
Approve: Approves the selected token to allow the requesting component to use the signing service.

Reject: Rejects the selected token. The requesting component will not submit CSR.

Fox Signing Requester (signingService-FoxSigningRequester)

This component is a client to the **Fox Signing Transport** and provides a parent component with the ability to make requests to onboard with the Signing Service and to renew certificates.

It is not possible to drop an instance of the **Fox Signing Requester** onto any arbitrary component as some code is required to support the interaction between the component and the requester.



Cert Requester	Fox Signing Requester
Status	{ok}
Fault Cause	
Enabled	true
Requester State	Not Onboarded
Advance Renewal Percent	8 percent [1 - 100]
Advance Renewal	+000d 00h 00m
Retry Period	001d 00h 00m [5 minutes - 5 days]
Last Attempt	null
Last Success	null
Last Failure	null
Next Renewal Attempt	null
Alarm On Failure To Onboard	false
Alarm On Failure To Renew	true
Alarm Source Info	Alarm Source Info
Signing Service Station	Select Station...
Requester Id	99527568-d59e-4b4f-b8fd-b0c982ee6d2e

Property	Value	Description
Status	read-only	Displays the status of the requester component. It indicates if in alarm state when onboard/renew has failed.
Fault Cause	read-only	Displays the most recent fault message.
Enabled	`true` or `false` (defaults to `true`)	If `false`, actions are disabled.
Requester State	read-only	Displays one of the following states: `Not Onboarded`, `Approval In Progress`, `CSR Submitted`, `Renewal CSR Submitted`, `Renewal Failed`, `Onboarded`
Advance Renewal Percentage	read-only	Displays the percent of the signed certificates validity period in which a renewal will be automatically attempted. With a 100–day certificate and a value of 10%, renewal will be first tried with 10 days remaining.
Advance Renewal	read-only	Displays the period prior to certificate expiry in which a renewal will be attempted.
Retry Period	read-only	Displays the delay between automatic attempts to renew.
Last Attempt	read-only	Reports the last time an onboard/renew was attempted.

Property	Value	Description
Last Success	read-only	Reports the last time a certificate was successfully retrieved.
Last Failure	read-only	Reports the last time a request failed.
Next Renewal Attempt	read-only	Reports the last time at which a renewal will next be attempted.
Alarm on Failure to Onboard	`True` or `false` (defaults to `false`)	If `true`, an alarm is generated when an onboard attempt fails.
Alarm on Failure to Renew	`True` or `false` (defaults to `true`)	If `true`, an alarm is generated when a renewal attempt fails.
Alarm Source Info	additional properties	Provides configuration for alarms.
Signing Service Station	drop-down	Selects from a list of Niagara network stations the one containing the Signing Service.
Requester Id	read-only	Displays the unique Id for this requester as registered with the Signing Service. It is automatically populated.

Actions

Onboard: Registers this component with the Signing Service. The component will automatically check for approval and perform the CSR submission and retrieval.

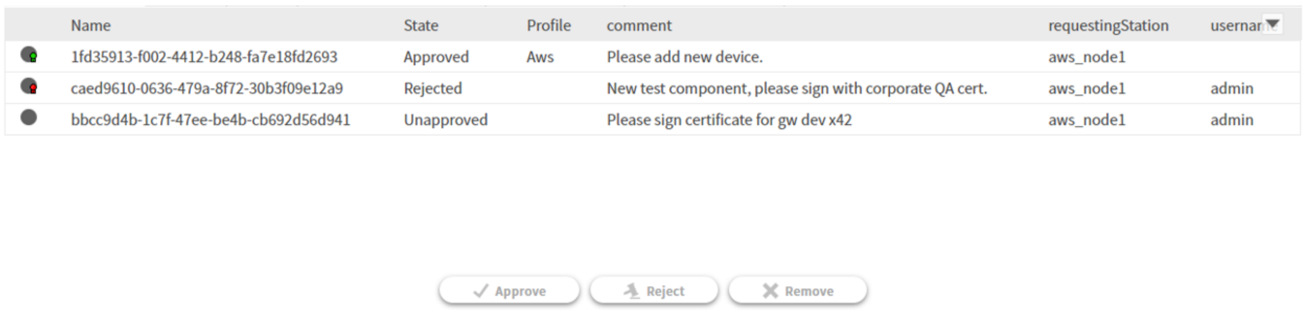
Renew: Forces a certificate renewal attempt.

Plugins (views)

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

Session Token Manager View (signingService-SessionTokenUxManager)

You can approve or reject tokens using actions on individual session tokens. Alternatively, you can use the **Session Token Manager** view, which is the default view for the store.

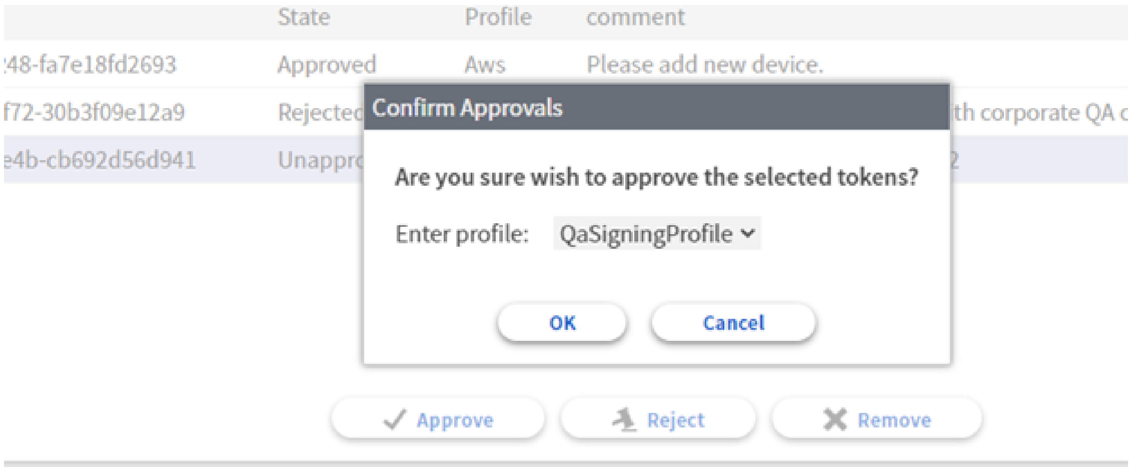


	Name	State	Profile	comment	requestingStation	usernar
	1fd35913-f002-4412-b248-fa7e18fd2693	Approved	Aws	Please add new device.	aws_node1	
	caed9610-0636-479a-8f72-30b3f09e12a9	Rejected		New test component, please sign with corporate QA cert.	aws_node1	admin
	bbcc9d4b-1c7f-47ee-be4b-cb692d56d941	Unapproved		Please sign certificate for gw dev x42	aws_node1	admin

To access it, expand **Config→Services→SigningService→Transports→foxTransport** and double-click **Session Token Store**. The view displays each session token as a row that includes the status of the token plus all the metadata values submitted by the requesting component, which should help you on your decision to approve or reject the token.

NOTE: Only admin users are permitted to approve or reject tokens. They may batch approve, reject or delete session tokens by selecting multiple rows.

When approving a token, you must select the **Signing Profile** to which this request should be associated. The CSR will be signed by the CA certificate defined in that profile.



	State	Profile	comment
:48-fa7e18fd2693	Approved	Aws	Please add new device.
f72-30b3f09e12a9	Rejected		th corporate QA c
e4b-cb692d56d941	Unappr		2

Confirm Approvals

Are you sure wish to approve the selected tokens?

Enter profile: QaSigningProfile

You can approve a rejected token at a later date provided the token has not automatically been removed before.

Glossary

Certificate	A PKI (Public Key Certificate) or digital certificate is an electronic document used to prove ownership of a public key. The certificate includes information about the key, the identity of its owner, and the digital signature of an entity that verified the validity of the certificate's contents. If the signature is valid, and the client can trust the signer, the client can be confident that it can use the public key contained in the certificate to communicate with the server.
Certificate Authority (CA)	An entity that issues the digital certificates used to certify the ownership of a public key by the named subject of the certificate. This allows system users to rely upon signatures or assertions made by the private key that corresponds to the certified public key. In this relationship model, the party that relies on the certificate trusts that the subject (owner) of the certificate is authentic because of the relationship of both parties to the CA.
Certificate Signing Request (CSR)	A block of encoded text that is given to a Certificate Authority (CA) when applying for an SSL certificate.
Distinguished Name (DN)	A string that uniquely identifies a certificate.
MQ Telemetry Transport (MQTT)	A protocol for exchanging queued messages between disparate systems.
Onboarding	Process of registering a requester with a service.
Requester/Requesting Component	A component that submits the Certificate Signing Request (CSR) to the service.
X509	A standard format of public key certificates.

Index