Canary Historian Connector

Canary Historian systems

- Name: **CanaryLabs**
- Version: 1.0.0.9
- Protocol: Proprietary
- Interface: TCP/IP
- Runtime: .NET 4.8 (Windows)
- Configuration:
    - Devices / Protocols

About Canary
Installation
Modules' Connections
Canary Historian Driver
Summary
System Requirements
Device Configuration
CanaryLabs Configuration Procedure
TagProvider
Troubleshooting
Driver Revision History

Overview

The platform includes enhancements specifically designed to leverage Canary Historian, a time-series database that provides high speed performance and friendly configuration tools. This integration includes:

- **Native Solution Historian**: Canary Historian embeds in the platform easily as if it were a native tool. Solutions can use it in conjunction with the existing built-in SQL Historian.
- **Create Canary Points directly at Designer**: No need to use or learn additional tools other than FrameworX Solution Designer.
- **Simplified License Management**: Request license upgrades directly from FrameworX.
- **Out of Box TrendChart Monitor**: Zero configuration is required; an out-of-the-box TrendChart monitor for Canary Historian tags.
- **Edge Data Collector**: Any of the 70+ communication protocols is now a Data Collector for Canary Historian, with Store and Forward support.
- **Canary Driver and TagProvider**: Connection with other Canary Historian instances, either mapping to Local Tags or using dynamic referencing, with tunneling features included.

About Canary

What is the Canary Historian?

Canary Historian is a time series database built for industrial automation. It is a high-performance historian designed for large-scale applications that handle substantial volumes of data writing. It provides scalability and enables efficient management of high-volume write operations. Its design supports extensive dataset analysis within time contexts for effective trend identification, avoiding the constraints of relational schemas. Canary's architecture balances speed, accuracy, and volume, making it an ideal choice for specialized time series data management needs.

When to use Canary Historian or the built-in SQL Historian?

Solutions can use both systems concurrently. Performance on SQL storage has increased over the years, but it cannot reach the speed and storage capabilities that a database specifically designed for time-series data, like Canary, can deliver. Use Canary for the tags that require high speed and/or long time retention. Keep those in SQL where the retrieval and storage requirements can be managed with SQL storage.

> **Using Multiple Historians**
>
> Trend Charts on Displays, Scripts, and all other modules use data abstraction layers, so there are no changes to the remaining configuration of the solution when you decide to modify the configuration of tags from using one historian to another.

Installation

The software platform should be installed and, when necessary, licensed.

Licensing Canary Historian

1. Go to SolutionCenter and navigate to 'Your License / License Information / Canary Historian'.
2. Request a license by clicking 'Request License'. Fill in the 'Name', 'Company', and 'Email' fields to generate the Serial Number, then click 'Request'.
3. Open the Canary Admin Tool, proceed to 'Home / Licenses', and select 'Local / Add license through the internet'. Enter the required details ('Name', 'Company', 'Serial Number', 'Email', 'Quantity') and check the 'Canary Historian' option. Click 'GET LICENSE'.
4. Upon receiving the License Key, confirm by clicking 'OK'. To verify the license, navigate to 'SolutionCenter / Your License' and refresh the information by clicking the 'Refresh' button. The message should change to "Canary Historian is licensed."

Canary Admin Tool configuration

Set up the Sender's EndPoints and access the Receiver's EndPoints in the Canary Admin Tool. see CanaryLabs Configuration Procedure

Creating **Solution and Enabling Canary Historian**

1. In SolutionCenter, create a solution (file .dbsln) and open it in the Solution Designer.
2. Access 'Solution / Settings / Target Platform and License', enable Canary Historian, and select the desired tag range (500, 1500, 5000, 15000, or Unlimited).

Done! The initial configuration is finished.

> This integration includes support specifically for Windows, as Canary Labs is a Windows-only integration.

Modules' Connections

Canary Historian can be used in different ways by the various modules.

Historian: Uses Canary as a Storage Location for archiving or retrieving HistorianTags.

Devices: Read and Write to external Canary Historian Server with the Device Module.

TagProvider: Map external Canary Historian Server to dynamic integration of its data and structures.

Displays and Scripts: No custom configuration is required, TrendCharts will use Canary Historian Data Automatically if available.

Historian Module

Configuring Historian using Canary

1. At Historian / StorageLocations, include Canary as a StorageLocation.
2. At Historian / HistorianTables, define virtual tables will control writing and retrieving data.
3. Map the tags at Historian / HistorianTags.

Read about the Historian Module.

Device Module

Configuring Devices using Canary.

1. Navigate to 'Devices / Channels' and select a protocol (Canary) to create a Channel.
2. Navigate to 'Devices / Nodes', create a Node linked to that Channel.
3. Map the tags at 'Devices / Points'.

Read about the Devices Module, or Read about the specific Canary Labs Communication driver.

TagProvider Module

Configuring TagProvider using Canary.

1. At Unified Namespace / TagProvider Connections, create a connection to a Canary Historian.
2. At Unified Namespace / AssetTree, add Canary data in the AssetTree.
3. Use Canary data in any module using the syntax Asset("<AssetPath>")

Read about Unified Namespaces

Read about the specific Canary Labs TagProvider

Displays and Scripts

Displays and Scripts, with no additional configuration, will query Canary Historian when using tags that were previously connected with Canary using the Historian Module.

Examples of the automated usage:

In Displays TendCharts. See TrendChart Control.

In Scripts, using the Historian Tables method to query data. See Historian Namespace Reference.

Canary Historian Driver

This document has only the specific information about the device connection settings and its address syntax.

Refer to the User Guide for more information on the Device Module.

Summary

Communication Driver Information	
Driver name	CanaryLabs Historian
Assembly Name	T.ProtocolDriver.CanaryLabs.dll
Assembly Version	1.0.0.9
Manufacturer	CanaryLabs

System Requirements

The following requirements must be matched in order to use the CanaryLabs driver:

- Product version 9.*1*
- .NET Framework 7.1 or higher

Device Configuration

Channel

- **ProtocolOptions**: `TagPropertiesCommunication` (boolean, DrvParams[0]) — When true, the driver reads/writes tag properties (Description, Min, Max, Units) alongside values. Default: false.

Node Configuration

Station Configuration

The station parameters are:

- **ServerName**: The name or IP address of the machine running the CanaryLabs Historian. (If the service is not running in the default port, the syntax should be: **<ComputerName>:<PortNumber>** or **<IpAddress>:<PortNumber>**)

- **ClientID**: Indicates the client identification that is used to make calls in the Canary service
- **ConnectionType**: Specifies the ConnectionType with the service. The options are:

    - *Anonymous*: Connection using no credentials
    - *UserName*: Connection using a username/password that is defined in CanaryAdmin
    - *Windows*: Connection using Windows credentials

- **UserName**: The user name used for verification (Valid for the *Username* and *Windows* connection types)
- **Password**: The password used for verification (Valid for the *Username* and *Windows* connection types)
- **View**: A list of views found in the specified ServerName. If the view is virtual, the Node will be **Read-Only**
- **Dataset**: The name of the dataset you will connect to. (Not available when the selected view is Virtual)

- **Station field EnableBackfillInsert** (boolean, station index 8) — Controls Canary backfill insertion mode. Default: true.
- Station format is: `Receiver;ServerName;ClientId;ConnectionType;UserName;Password;View;Dataset;EnableBackfillInsert`

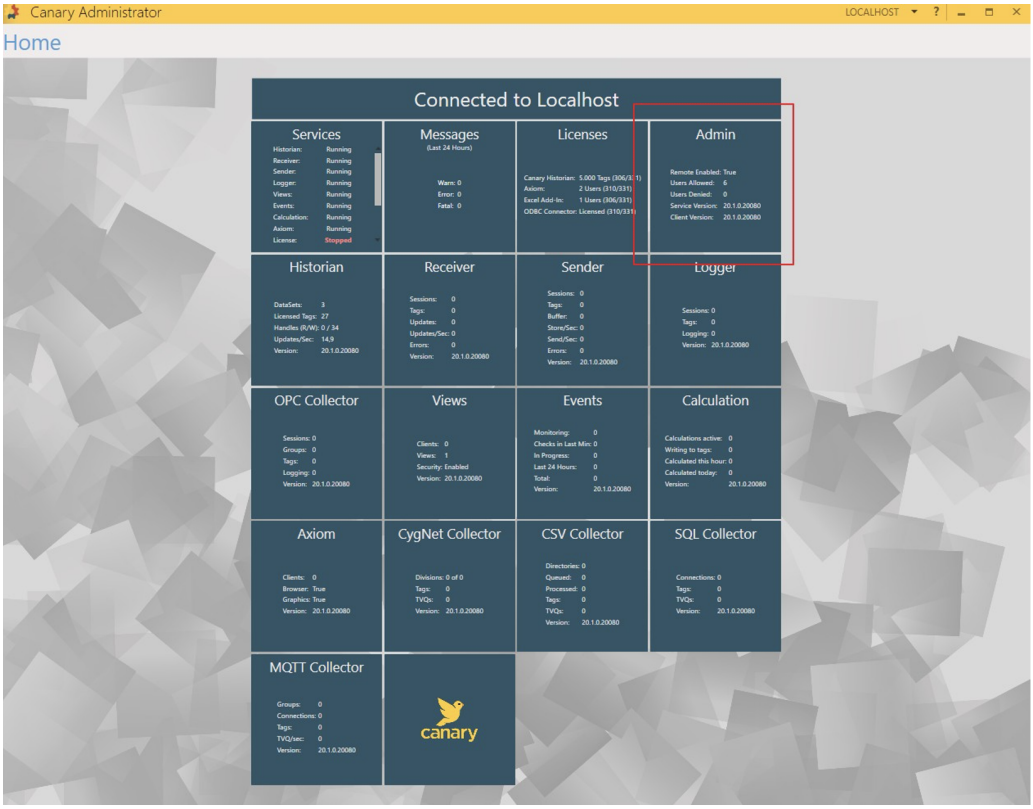
⚠ Use the Test Connection button to check the connection with the Server and Database.

CanaryLabs Configuration Procedure

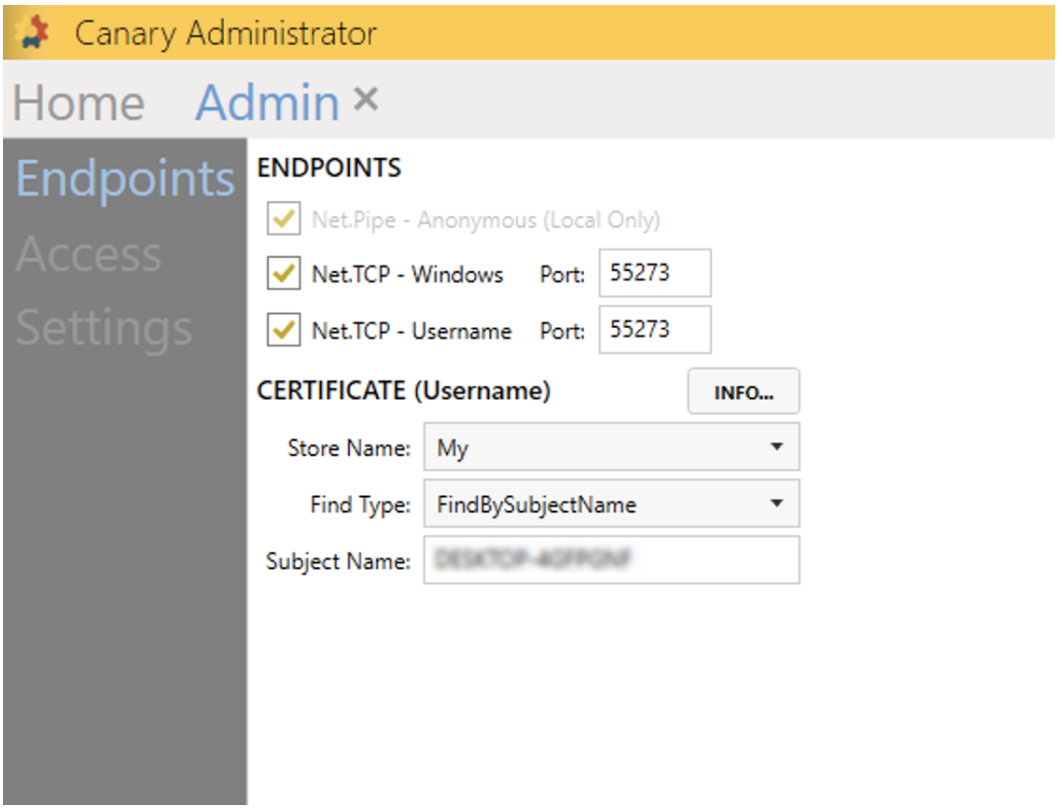
Once you have the CanaryLabs Historian software and the Canary Admin application installed, you will need to configure some permissions for the connection to work.

Admin Configuration

Open the **Admin** Tab.

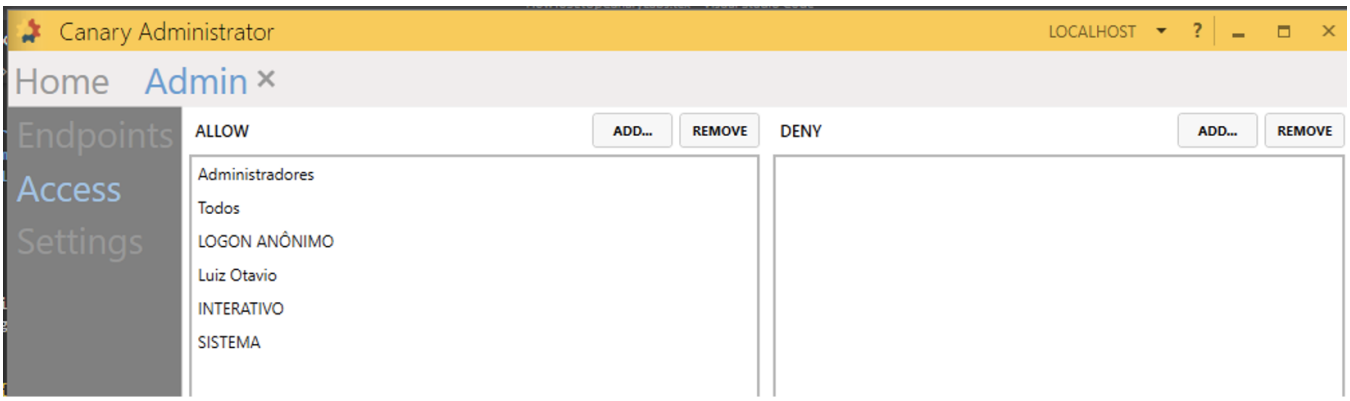


Under **Endpoints,** select every checkbox, but do not change the port numbers.

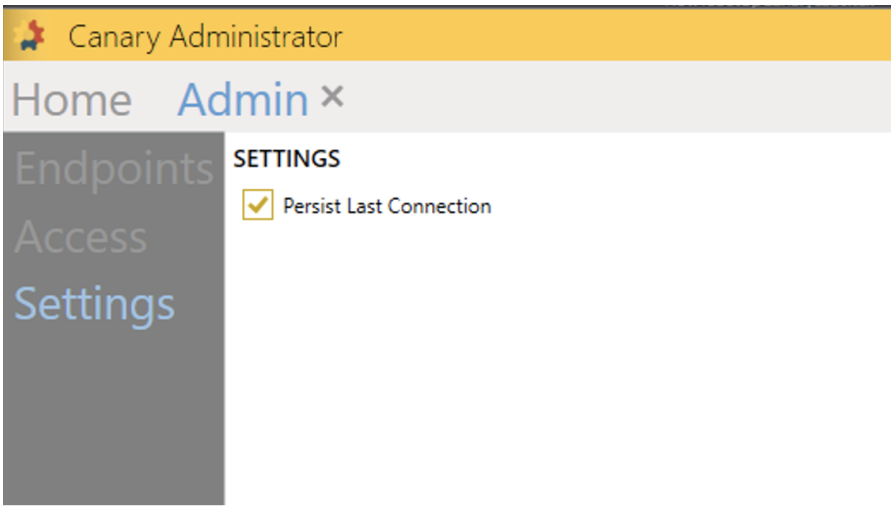


Under **Access,** add the users that will be allowed to access the CanaryAdmin programs.

- *Anonymous*
- *System*
- *Administrator*
- *Interactive*
- *Everyone*
- ?*Your User*?.

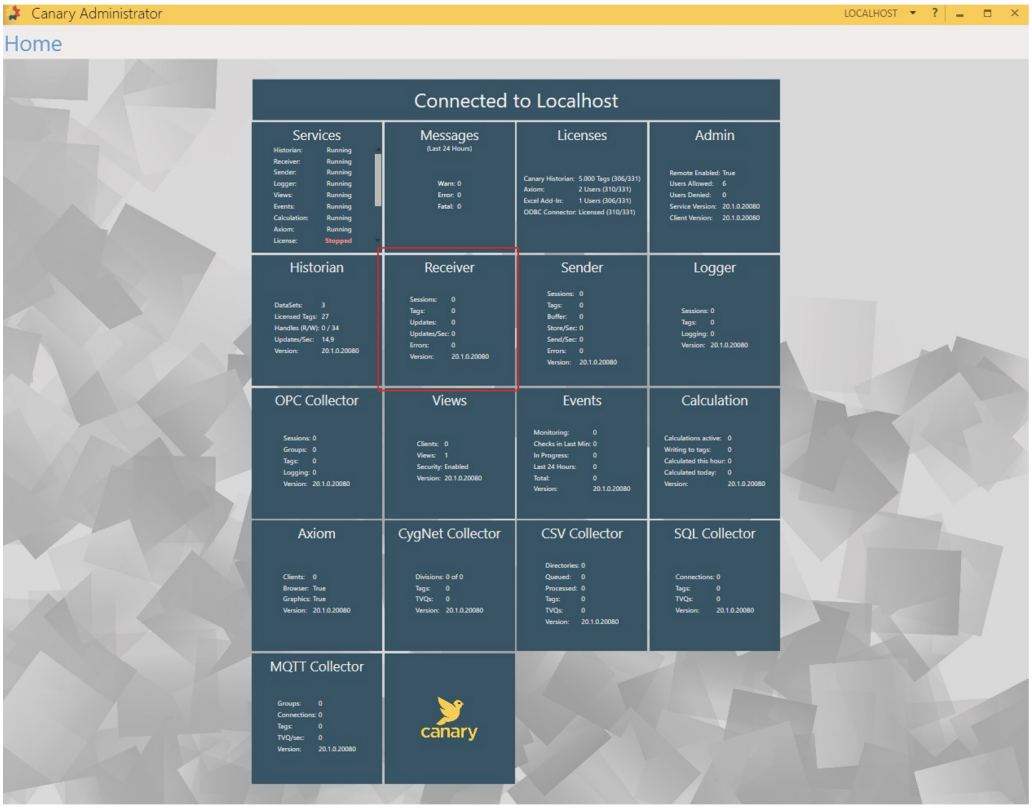


Under **Settings,** make sure the checkbox for the *Persist Last Connection* is selected.

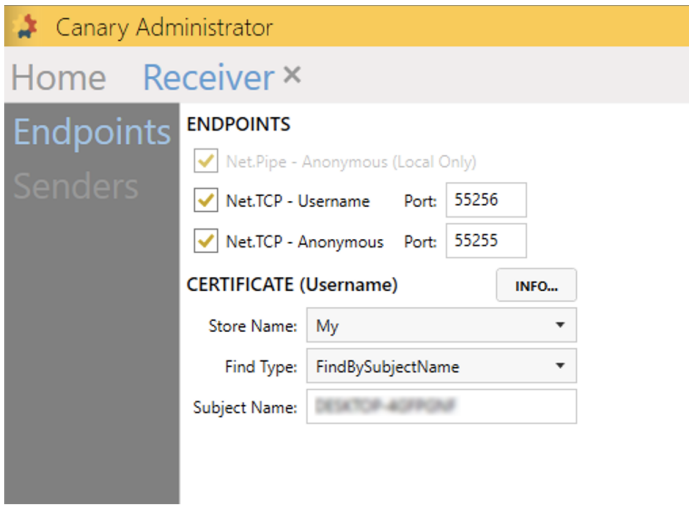


Receiver Configuration

Open the **Receiver** Tab.

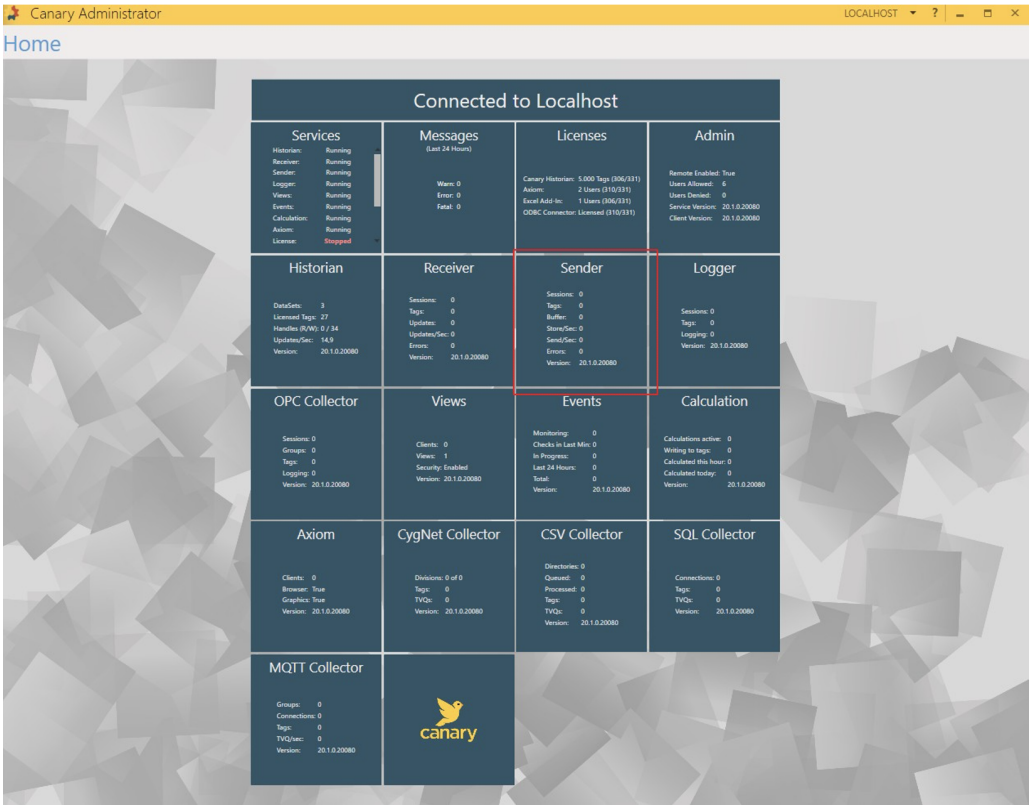


Under **Endpoints,** select every checkbox, but do not change the port numbers.

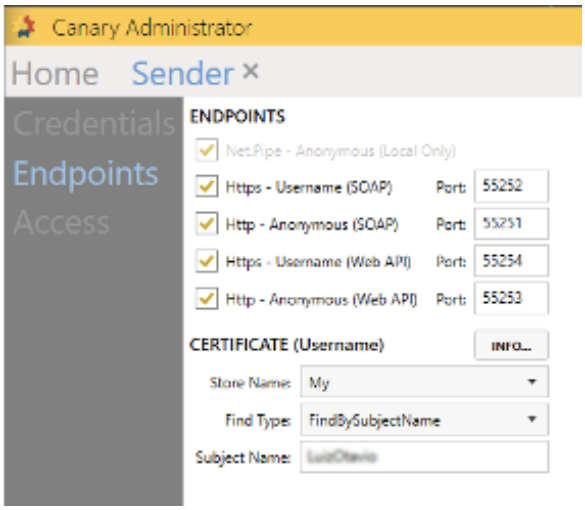


Sender Configuration

Open the **Sender** Tab.

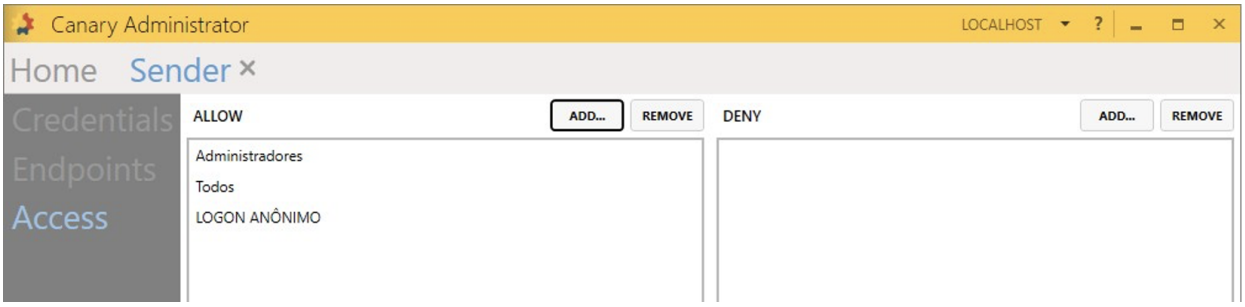


Under **Configuration** > **Endpoints,** select every checkbox, but do not change the port numbers.



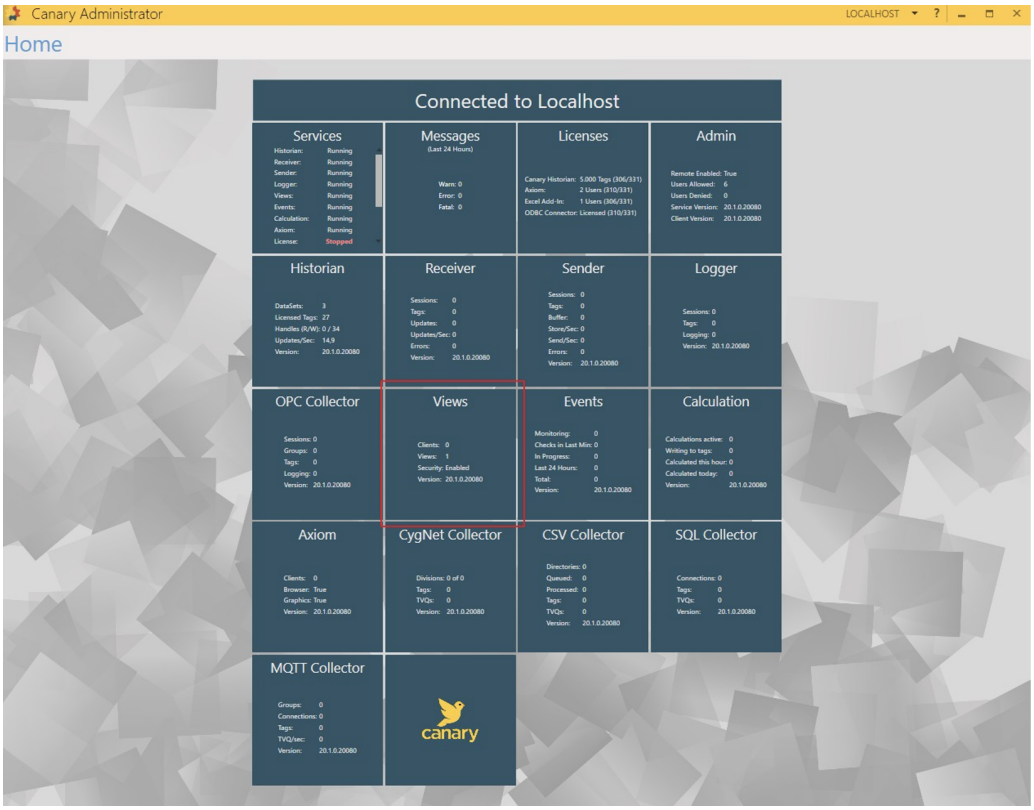
Under **Configuration** > **Access,** add all the users that will be allowed to write in the Canary Historian. E.g.:

- *Anonymous*
- *System*
- *Administrator*
- *Interactive*
- *Everyone*
- ?*Your User*?.

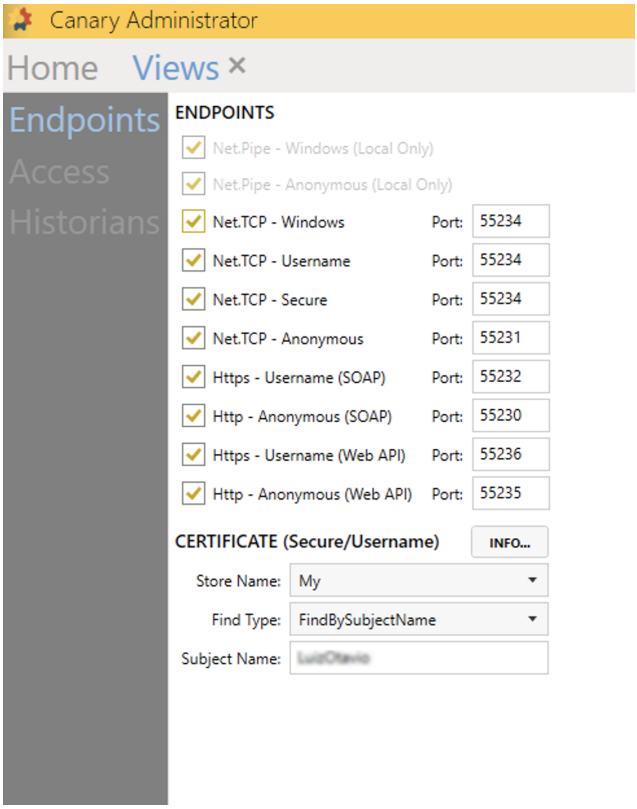


Views Configuration

Open the **Views** Tab.

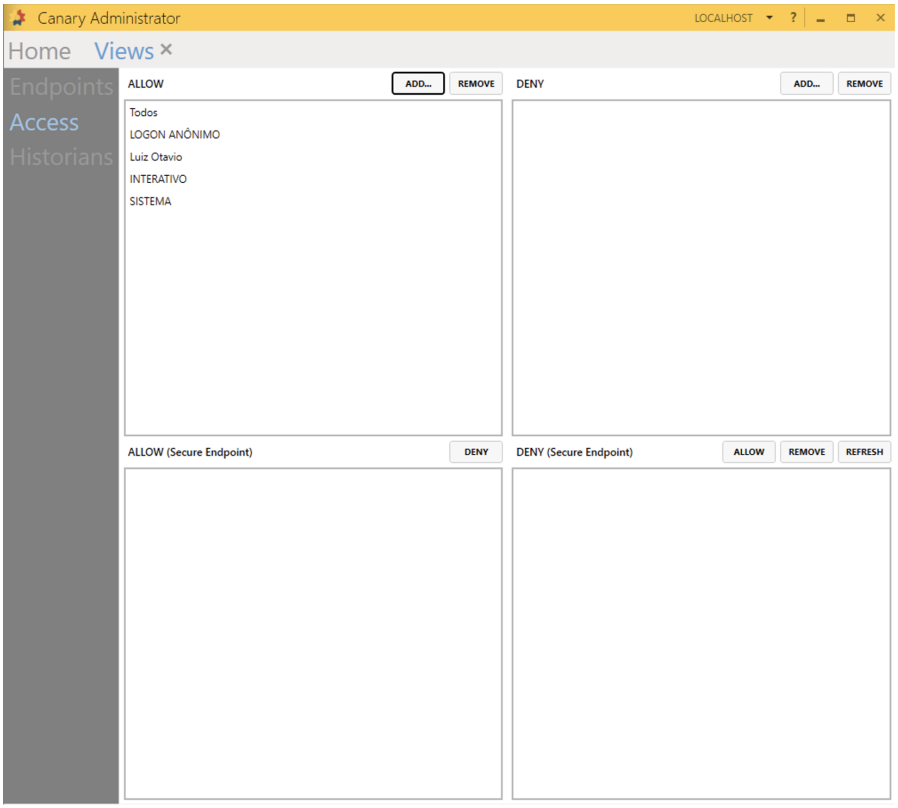


Under **Configuration** > **Endpoints,** select every checkbox, but do not change the port numbers.

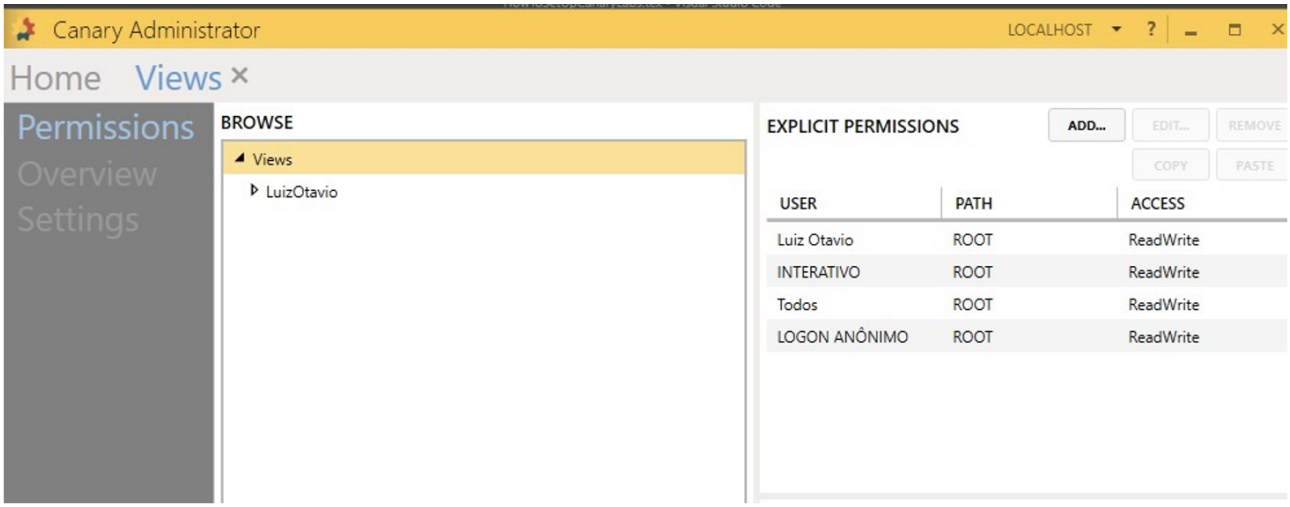


Under **Configuration** > **Access,** add all the users that will be allowed to write in Canary Historian. E.g.:

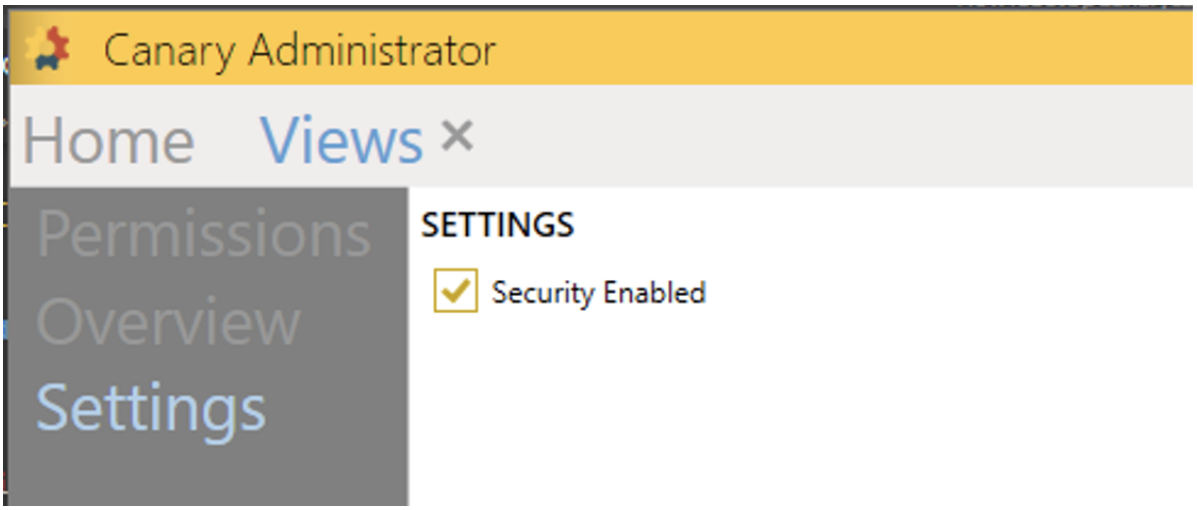
- *Anonymous*
- *System*
- *Administrator*
- *Interactive*
- *Everyone*
- ?*Your User*?.



Under **Security > Permissions**, add the users for the Root path with the *ReadWrite* AccessType.



Under **Security > Settings**, make sure the *Security Enabled* checkbox is selected.



With an **OK!** status message, 3 features are enabled for this Driver:

1. **Import Tool**
2. **Unified Namespace**
3. **Asset Modeling**

Import Tool

You can automatically import the existing Canary Variables into your Project by clicking the **Import** button.

This tool allows you to choose variables, from the list, that will be imported into your project; automatically creating the Tags and Communication Points.

Point Configuration

Address

You can use the Browse button to see the available data in the CanaryHistorian database, or you can write a Tag address directly in the Item field.

For a simple Tag, use the syntax: ***<DatabaseName>.<VariableName***>. E.g.:

TagName	**Address**
Motor1 Temperature	Motor1.Temperature
Motor1 RPM	Motor1.RPM
Motor2 Temperature	Motor2.Temperature
Motor2 RPM	Motor2.RPM

Or, you can create a Template (ex. Motor1 and Motor2), and input it into the Points list. Its members (e.g.: Temperature and RPM) will be mapped automatically. E.g.:

Engineering Environment	
TagName	**Address**
Motor1	Motor1
Motor2	Motor2
Runtime	
TagName	**Address**
Motor1.Temperature	Motor1.Temperature1
Motor1.RPM	Motor1.RPM
Motor2.Temperature	Motor2.Temperature1

Motor2.RPM	Motor2.RPM

⚠ Array elements are also automatically expanded in runtime.

Use the **Verify** button to check if a name is valid and get the current value and quality.

TagProvider

This Communication Protocol supports the TagProvider feature, which is a tool that allows you to access your Communication Device Data Model without creating project Tags.

For more information, please refer to the TagProvider document.

Configuration

To configure the CanaryLabs protocol as a TagProvider, go to **Edit / Tags / Providers**, and create a new provider for the CanaryLabs protocol.

Configure the items under the **PrimaryStation** column the same way that was described in the Node Configuration.

Troubleshooting

The status of the driver execution can be observed through the diagnostic tools, which are:

- Trace window
- Property Watch
- Module Information

A status value of 0 (zero) means the communication was successful. Negative values indicate an internal driver error, and positive values are the protocol's error codes.

Driver Revision History

CanaryLabs Revision History	
Version	**Notes**
1.0.0.9	The station editor now shows the saved View and Dataset when it is reopened; in remote engineering, Test Connection states that the test ran on the Designer and does not validate the runtime.

In this section...
