Technical Document

Niagara Lonworks Driver Guide

May 19, 2021

niagara4

Niagara Lonworks Driver Guide

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

Confidentiality

Trademark notice

Copyright and patent notice

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 applies to the Lonworks drivers for any release of Niagara 4. The audience is a knowledgeable Lonworks user who is Niagara 4 certified.

Document change log

Updates, changes and additions to this guide are listed by the date the document was released.

May 19, 2021

Update prerequisites.

February 2, 2021

Updated several topics related to Lon over IP in the "Network configuration" chapter. Improved graphics and descriptions in the component topics. Updated the "New (and Edit) device window" topic in the "Windows" chapter.

July 31, 2020

Reorganized the document for Niagara 4. Also updated for added support for 64-bit systems in Lonworks in Niagara 4.9 and later.

June 17, 2019

Added Niagara 4.8 security related properties to the "tunnel-TunnelService" component topic.

September 18, 2018

Added "Properties and elements' chapter to explain properties of Lonworks.

April 5, 2018

Major rewrite and reorganization on the occasion of bringing this document into the content management system, reorganized topics to emphasize tasks. Put all reference material (components, plugins, and windows) at the end of the document.

June 7, 2013

Added several security-related notes and cautions.

Related documentation

These documents contain related information.

- Multiple installation and startup guides, one for each remote host controller.
- *Niagara Drivers Guide*.
- *Niagara Platform Guide*.

Chapter 1 Getting started

Topics covered in this chapter

- ♦ Lonworks modules
- ♦ Prerequisites
- ♦ Installing the driver
- ♦ Lonworks driver

Lonworks (Local Operating Network) is a peer-to-peer networking protocol for control applications. It was designed to connect devices over twisted pair, power lines, fiber optics and RF (Radio Frequency) media.

A Lonworks network does not require a single master to multiple slave relationships. Control devices freely communicate directly with each other. This design avoids some of the problems that can occur when a master fails. The protocol manages device addresses by broadcasting them to all devices using a process known as binding, which occurs when the network is commissioned by a network management tool (explanation adapted from www.csimn.com).

Lonworks modules

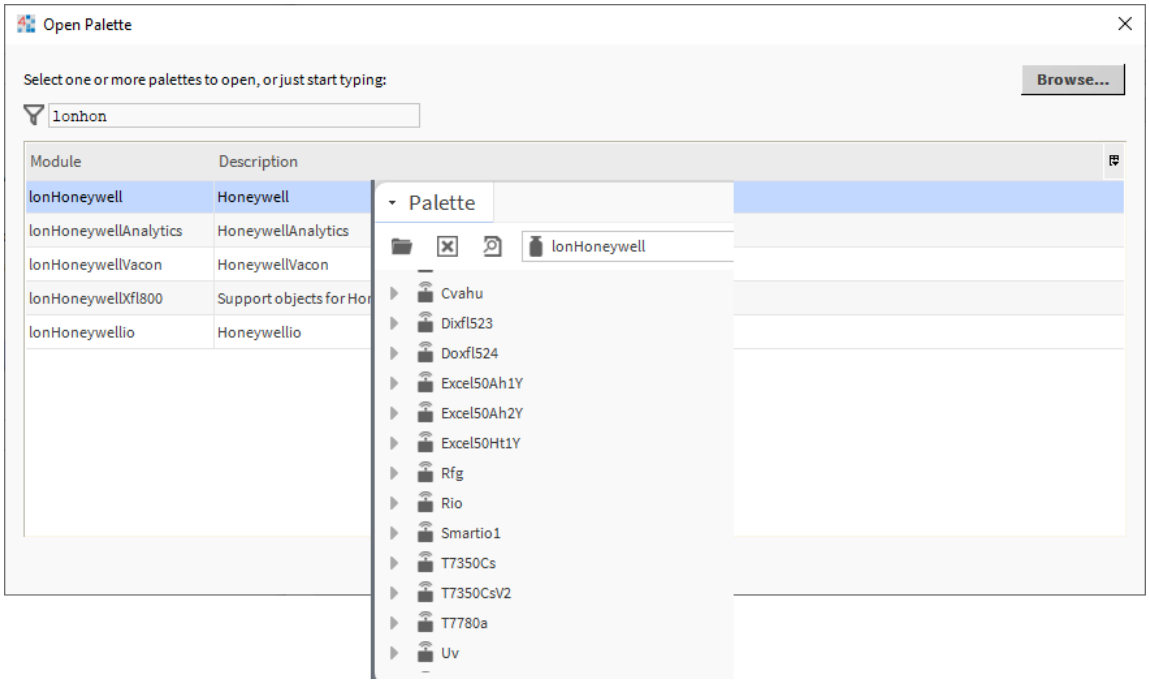
To use this driver, you must have a target host (remote controller) that is licensed for the feature **`lonworks`**. Your Supervisor station must also be licensed for **`lonworks`**).

The Lonworks driver provides these driver modules and many modules that support specific devices.

Palette name	Network component name in the Nav tree	.jar file names in the modules folder	Function
lonworks	LonNetwork	lonworks.jar	Provides Lon (Local Operating Network) networking functions; manages devices and proxy points.
lontunnel	TunnelService	lontunnel.jar	Supports remote PCs with a Niagara 4Tunnel Client installed to use a legacy or vendor-specific PC application to access devices connected to one or more driver networks. A tunnel connection allows the remote client application to operate as if it were directly attached to the driver network (via a virtual PC port).
kitLon	LonTime, LonTodEvent, LonEnumTodEvent, LonPoint, LonReplace, and BufferParams	kitLon.jar	Provides additional components that manage inputs and outputs and configure advanced messaging.
lonIP	LonIpNetwork	lonIp.jar	Communicates on an IP Lonworks channel as defined in CEA-852. The station presents itself as an IP channel node.
lon`Vendor`	depends on the vendor	lon`Vendor`.jar	`vendor` is replaced by the name or acronym for a building automation manufacturer, such as Aaon, FloridaHeatPumps, Hillrom, Mitsubishi, Trane, etc.

Each **`Lon vendor`** module includes its own palette, which you open in the palette side bar, JACE-8000 and manually drag or copy devices to a Lon network.

Figure 1 Opening a specific Lon vendor palette



Prerequisites

To successfully install and use the Lonworks driver your installation needs to meet these requirements.

- A Niagara 4 license for the **`lonworks`** feature for each host.

  Lonworks device and proxy point limits may exist in your license.
- A target host controller running the Niagara 4 framework or later
- Workbench running on a 64–bit PC (unless you are using Niagara 4.8 or an earlier version of Niagara that supports a 32–bit computer environment)
- When you install Workbench on your PC you enable it to be used as an installation tool or engineering workstation. Configuring Workbench as an installation tool installs the needed distribution files (.dist files) for commissioning various models of remote platforms. The .dist files are located under install directory in an `sw` subdirectory. For details, refer to the *Niagara Platform Guide*.
- A network of installed Lon devices

Installing the driver

Installation begins with the remote controller.

Prerequisites: Your installation meets the requirements outlined in Prerequisites. You are working in Workbench running on a PC.

Step 1 Open Workbench on your PC.

Step 2 Commission your remote host platform(s).

Step 3 Install the **`lonworks`** and **`kitlon`** modules, plus any custom Lonworks module (for example, **`lonHoneywell`**, **`lonSiebe`**, **`lonStaefa`**, and so on).

For the station running on the host to perform the Quik Learn routine, you must have the appropriate custom Lonworks vendor module installed in the remote host. This differs from a Discover and Add operation, which references modules in your client PC's software database.

Step 4 If you are using the Lon over IP driver (lonip), install the module in the controller also.

Step 5 If you need Lon tunneling, install the **lonTunnel** module in the controller.

Step 6 Upgrade any modules shown as out of date.

Lonworks driver

Lonworks network management requires a single owner to perform and maintain an accurate database of things, such as node address assignments, nv (Network Variable) bindings, message services used, and a raft of other entities. The Lonworks driver uses the standard framework architecture to supply this network management capability.

Station as network manager

You set up a Lon network with the station acting as the network manager for all connected Lonworks devices. In this application, Workbench models all Lon devices in the station's database (under the **LonNetwork**), and it models data in each device using Lon proxy points. The **lonworks** driver provides online learning capabilities to simplify this process.

The various manager views on the Lonworks network provide the interface to all network management operations performed by the station. This includes the engineering, maintenance, and troubleshooting of the Lonworks network as well as all bind operations. Using the wire sheet view of the network, you can establish links directly between Lon nodes.

Depending on the installation scenario, you may need to perform a learn of a network of Lon devices that are already configured (previously managed), or, on a previously- unmanaged (new) network. Both learn types are supported, however, engineering considerations apply.

Station as network node

In some scenarios you may wish to install a host as another Lonworks node, where another (external) Lonworks network management tool is used, for example, an LNS (L2TP Network Server). In this case, the **Local Lon Device** does not perform network management. Instead, you expose other station data as network variables (nvs) and ncis under the **Local Lon Device**.

When a station functions as a network node, you do not use the various views of the Network (**Lon Device Manager**, **Lon Link Manager**, and so on) as described in this document. Instead, you work only in (and under) the **Local Lon Device**.

Status and monitoring

Status of a network is `{Ok}` or perhaps `{fault}` (fault results if the Lonworks feature is not licensed). The **Fault Cause** property further explains any fault status. A `{down}` status occurs if you manually set the network's **Enabled** property to `false` (on the network Property Sheet). The **Health** slot contains historical timestamp properties that record the last network status transitions from `{Ok}` to any other status.

As in other driver networks, a Lonworks network has an available **Alarm Source Info** container slot that you can use to differentiate a Lonworks alarm from other component alarms in the station.

The network's **Monitor** component verifies the presence of networked devices. It sequentially pings all devices that are mapped in the station.

Chapter 2 Network configuration

Topics covered in this chapter

♦ Adding a network
♦ Changing the network's working domain
♦ Commissioning a router
♦ Configuring a station as a Lon node
♦ Preparing to work offline
♦ Lon over IP
♦ Station-less access
♦ Variable (Lon component) management

This chapter provides a collection of procedures to use the Lonworks driver in online scenarios. Like other drivers, you use the special manager views and property sheets to configure the network.

Managed vs. unmanaged networks

Lonworks networks may be previously managed or unmanaged.

- In a previously-managed network, all nodes are already installed and commissioned using a Lonworks network management tool. Each device has a unique subnet-node address, and usually coherent Lon bindings with other devices.You usually want to learn all existing network management. The driver's Quik Learn preserves this configuration.
- In unmanaged networks, nodes are installed using default subnet-node addresses, where there are often duplicate addresses or other address conflicts. Lon bindings between devices do not exist. There is no existing Lon network management to learn (only to establish for the first time).

Router wiring and configuration

If your network includes one or more routers, you must wire them according to the following rules, and assign **`Channel Ids`** and **`Subnet Ids`** that match the wired configuration:

- A channel is a network segment between routers. A particular subnet must be contained in a single channel. There can be multiple subnets on a single channel.
- The network must be configured such that there is only one path between any two specific nodes. There cannot be any loops.
- Routers have two Neuron chips (and IDs). Each Neuron is a member of a separate channel, and must be assigned to a unique subnet/node address.

  NOTE: Routers have a far and near interface relative to the network management device.

Commissioning

The term commissioning to the process that installs software from a central Supervisor PC and station to a remote host controller on a Niagara network.

Lonworks also requires the commissioning of networked devices including routers. The **Commission** button, which initiates this process can be found on the **Lon Device Manager**, and **Lon Router Manger** views.

Adding a network

Use the following procedure to add a **LonNetwork** or **LonIpNetwork** component under the station's **Drivers** container. If the host has multiple Lon ports, add one network for each physical port.

Prerequisites: You are connected to the remote station.

Step 1 Expand **Config** and double-click the station's **Drivers** container.

The **Driver Manager** view opens.

Step 2 To add a network, click **New**.

The **New** window opens.

Step 3 Select `Lon Network` or `Lon Ip Network`, fill in the number of networks to add, and click **OK**.

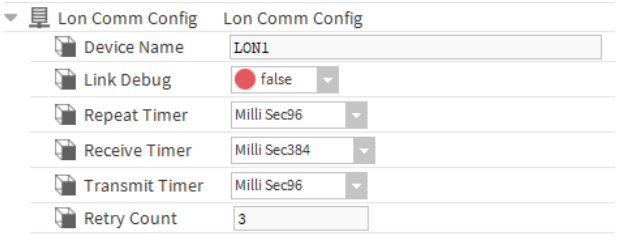
The second **New** window opens to name the network.

Step 4 Name the network and click **OK**.

The new network(s) reside under your **Drivers** container with a status of `{ok}`. If you create multiple networks at once, the driver configures each with a **Device Name** of `LON1`, `LON2`, and so on.

Step 5 To update the **Device Name**, right-click the network node in the Nav tree, click **Views→Property Sheet**, expand the **Lon Comm Config** property.

The **Lon Comm Config** properties open.



You leave all **Lon Comm Config** properties at default values. The exception is when you have multiple Lon networks (physical Lon ports), where each network must have a unique **Device Name** (LON1, LON2, and so on.).

Step 6 Update the **Device Name** (`LON2`, `LON3`, and so on.) and click **Save**.

Changing the network's working domain

If a host station is operating as the network manager (typical), its **Lon Netmgmt** always needs access to the zero-length domain to receive service pin messages. The driver always uses domain index Zero for its working domain (not to be confused with a zero-length domain). If you change the default working domain, the zero-length domain is retained, but moved to domain index one.

Step 1 Expand **Config→Drivers**, right-click the **Lon Network** and select **Views→Property Sheet**.

The **Property Sheet** opens.

Step 2 Click to expand **Lon Netmgmt** container.

Step 3 In **Domain Id** property, click the length drop-down and select the matching domain-length used by other Lonworks devices (either `1`, `3`, or `6`).

The adjacent **Id** property changes from empty to show a number of hexadecimal bytes. (for example, `00` or `00 00 00`). The ID range is from `00` to `FF`.

Step 4 In the **Id** property, enter the Domain Id used by other devices and click **Save** button.

The **Local Lon Device** is now configured with a non-default working domain.

NOTE: To place other (existing) Lon devices in this domain, you must recommission them.

Commissioning a router

Commissioning a router sets a router's internal tables to a functioning state. You perform this on any newly-added router except those created by a Quik Learn of a previously managed network, where the initial state of the router is already `Config Online`.

Prerequisites: The router is connected to the network and ready to communicate.

Step 1 To examine the router's internal tables, expand **Config→Drivers**, right-click the Lon network and click **Views→Lon Utilities Manager**.

The **Lon Utilities Manager** view opens.

Step 2 Select the router from the **`Device`** drop-down list.

The internal tables appear in the **Results** pane.

Step 3 To commission the router, click the drop-down menu at the top right of the view and click `Lon Router Manager`.

The **Lon Router Manager** view opens.

Step 4 Select the router in the table and click the **Commission** button.

A commission command performs the following steps:

1. If using a service pin, the process waits for a service pin message to obtain the router's **`Neuron Id`**, otherwise, it uses the **`Neuron Id`** already stored in the station database.
2. The commission job initializes the router's domain table, both far and near side, as follows:
   - It sets Domain Index zero to the Lon network's working domain.
   - If the device has two domains, it sets Domain Index one to either `not in use` if the working domain is the zero-length domain, or to `zero-length domain` if the working domain is not the zero-length domain.
   - Finally, the job configures the Netmgmt Authentication Key and the device's subnet/node address in all active domains.
3. Next, the job updates configuration device data as follows:
   - **`Channel Id`**: per the property as defined under **`LonNetmgmt`**.
   - **`Node Priority`**: per the Lon device property.
   - **`Location`**: per Lon the device property.
   - **`Authenticate`**: per the property under **`LonNetmgmt`**.
4. The job sets the router table per the current network configuration.
5. Finally, it sets the router state to `Configured, online`.

Configuring a station as a Lon node

To configure a station as a Lonworks node, you work only in (and under) the **Local Lon Device**.

Step 1 Expand **Config→Drivers→LonNetwork**, right-click **Local Lon Device** and click **Views→AX Property Sheet**.

The **AX Property Sheet** opens.

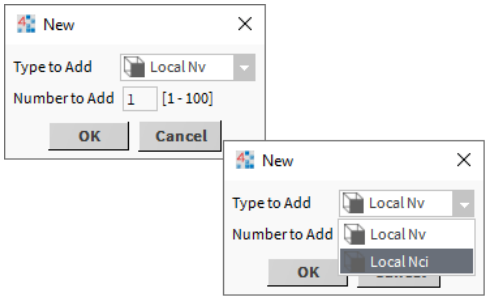
Step 2 Configure following properties:

a. Change the **`External Config`** property from `false` (the default) to `true`.

b. Make sure that the **`Self Doc`** property begins with the string `&3.0@0`.

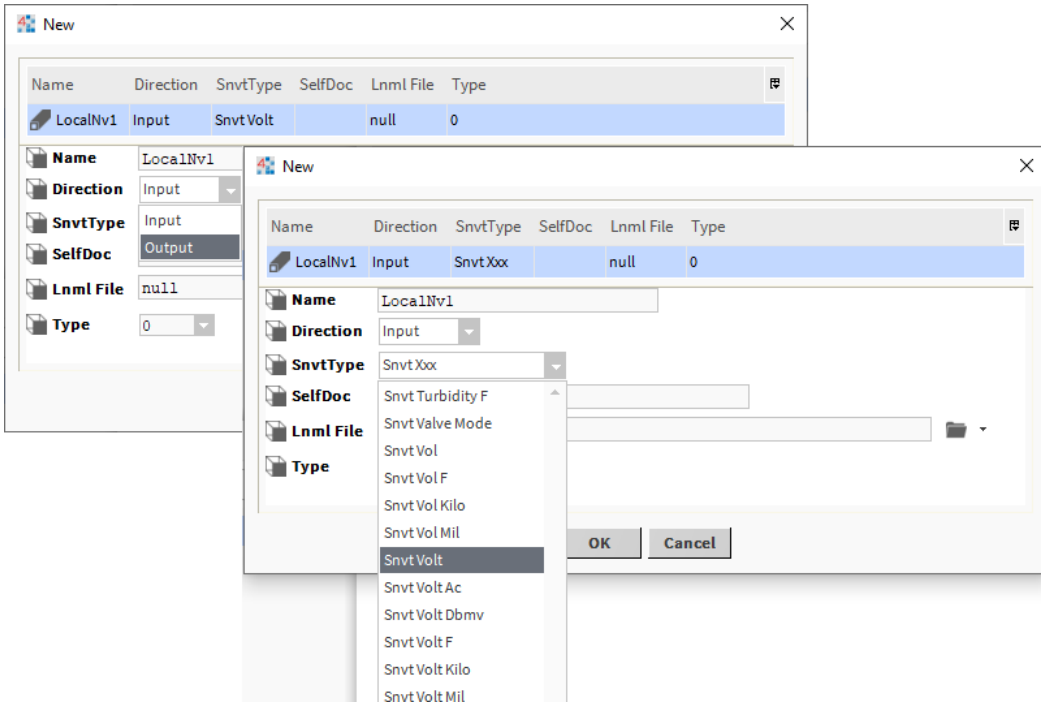
Step 3 To expose station data as Lonworks, double-click the **Local Lon Device** node in the Nav tree and click the **New** button.

A **New** window opens.



Step 4 Specify the type of data to add `Local Nv` (Network Variable) or `Local Nci` (Network Variable Input), how many to add, and click **OK**.

The second **New** window opens.



Step 5 Configure the data and click **OK**.

You repeat this some number of times, until you have defined the necessary collection of nvis, nvos, and ncis that will act as the Lonworks interface. If compatibility with Echelon LNS is required, you must create two specific nv entries:

- `nviObjRequest` (Direction = Input) (Snvt Type = Snvt Obj Request) [Self Doc = `@0|1;nviRequest`]
- `nvoObjStatus` (Direction = Output) (Snvt Type = Snvt Obj Status) [Self Doc - `@0|2;nvoStatus`]

If you select `Local Nci`, direction is always input, whereas if you select `Local Nv`, you can set each entry as an nvi (input) or nvo (output).

Preparing to work offline

Working offline makes it possible to configure a complicated Lon network before all devices are physically in place and connected to the network. This procedure creates an `.xif` file (external interface file) of a Lon network's **Local Lon Device**. This file describe the capabilities of the Lonworks device (host) to another Lonworks network management tool, enabling offline engineering.

Prerequisites: The station has been configured as a Lon node.

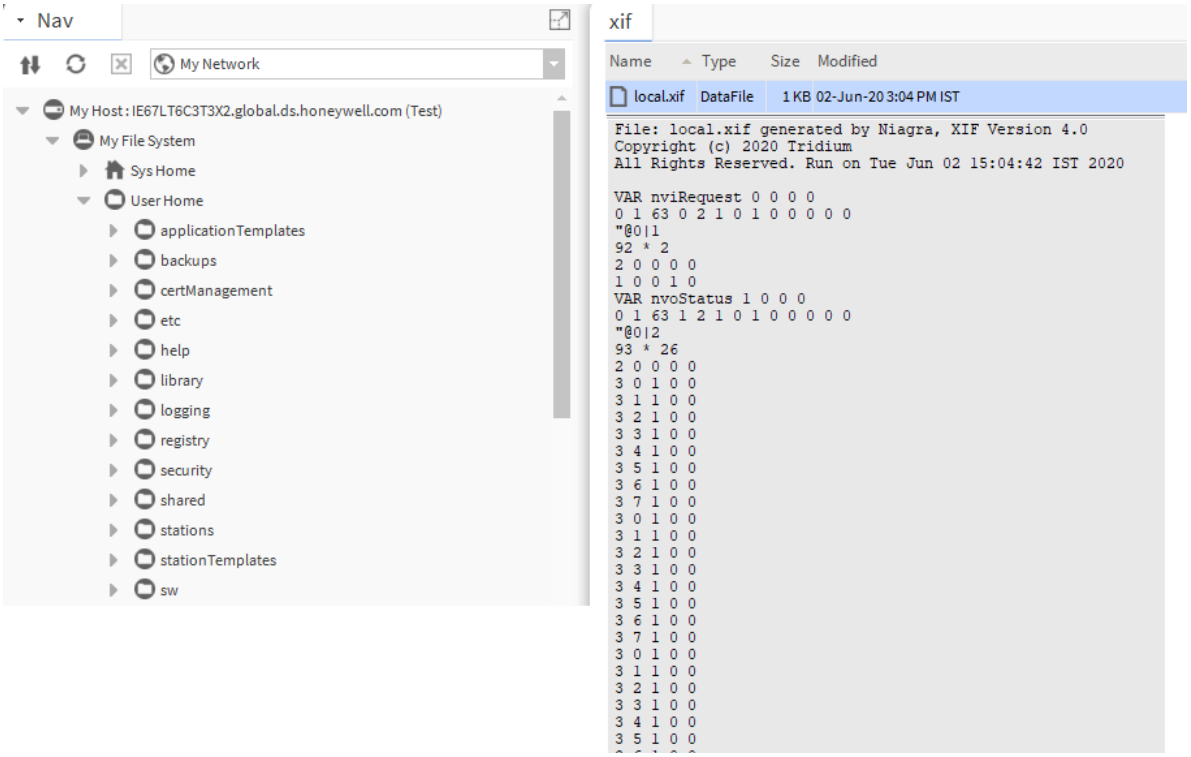
NOTE: This should be the last step when configuring the station as a Lon node, after finishing the other engineering tasks under the **Local Lon Device**.

Step 1 Right-click the **Local Lon Device** node in the Nav tree and click `ExtractXif`.

The **Extract Xif** window opens.

Step 2 Enter the name of the file and click **OK**.

The driver creates the .xif file under a `/xif` folder in the Workbench home directory (`!/xif`).



Lon over IP

The **LonIpNetwork** is a modified version of the standard **LonNetwork** that supports an Ethernet network. Lon over IP uses the **`lonIp`** module, which contains a palette. Compared to a standard **LonNetwork**, the **LonIpNetwork** has an additional frozen **ipChannel** container slot with other child components (and a view).

The modified Lon network communicates over an IP Lonworks channel as defined by the CEA-852 tunneling standard. In this document, this CEA-852 channel is referred to as the LonIp channel. When using the LonIp driver, the station presents itself as an IP channel mode member (as opposed to a Lon/IP router member).

License requirement

A host must be properly licensed for LonIp, including separate feature entries for both **`lonworks`** and **`lonIp`** in its license. Device limits or proxy point limits for either feature may exist in your license.

If the host is not licensed with the **`lonIp`** feature, a **LonIpNetwork** may remain in a fault condition.

Config Server requirement

CEA-852 specifies a configuration server (Config Server) that handles the distribution of routing information to all members of a LonIp channel. You can configure the host station to operate as a Config Server on the network of configured CEA-852 routers (Lon Ip channel), or you can use a pre-configured, third-party Config Server.

In either case, the LonIp channel must be properly configured, that is, the Config Server must recognize all channel members, and have up-to-date routing tables. Each channel member must be assigned a unique subnet/node address while using the same channel ID.

Channel setup is outside the scope of normal Lonworks station configuration, with the sole exception of the subnet/node address assigned to the station for use as an IP channel node. To set up Lon/IP router members in a proprietary, vendor-specific configuration, refer to each device's documentation, which provides details on how to set up its Lonworks subnet/node address, etc.

Once the IP channel is properly configured, and the station has been added as an IP channel node, all other Lonworks functions behave the same as if the host was connected directly to a Lon trunk.

NAT router port forwarding and other configuration requirements

If the host is behind a NAT (Network Address Translation) router, configure the router to forward any port used by the station, and any other LonIp channel members, as LonIp channel nodes. If the station is the Config Server, the Config Server port must also be forwarded. Refer to manufacturer instructions for setting up port forwarding.

The configuration of LonIp routers for use behind a NAT router requires setting the NAT router address, and possibly other configuration information that is not defined in CEA-852. Refer to each device manufacturer's instructions for such configuration.

Frequently-asked Lon over IP questions

These questions cover basic Lon driver features and behaviors.

Is a physical Lon adapter (FTT-=10, etc.) required by the host to use the LonIp driver?

No Lon adapter is required. The LonIp driver uses the host's Ethernet port—unlike the standard Lonworks driver, which binds any Lon network to a specific Lon adapter by device name (LON1 or LON2). In the LonIp driver, this device name is fixed, read-only as **`LonIp`**.

Find this in property sheet of a **LonIpNetwork**, by expanding its **Lon Comm Config** container.

Why cannot I add the LonIpNetwork from the Driver Manager, as with most other drivers?

You cannot use the **Driver Manager** view to add the **LonIpNetwork** component because the driver has no separate LonIpNetwork class. The **`LonIpNetwork`** palette entry is a BLonNetwork with the comm stack and the local device changed to support communications using IP.

What is the purpose of the Config Server?

The **Config Server** container collects address information from each of the members of the LonIp channel, and updates all the other members with the collected information. One central device must be responsible to collect and disseminate this member information because the protocol does not provide a mechanism for members to discover each other.

Why during a device discovery, do I get a lot of warning messages, similar to below?

```
WARNING [12:58:48 31-Jul-09 EDT][lonip] Received response message with no matching
transaction.16 00 01 19 0f 00 04 01 00 00 00 00 00 00 00 00 21 03 00 00 08 26 81 80 00 01 01 01
01 9a 21
```

This is normal behavior for the LonIp driver (**LonIpNetwork**), as the driver receives and processes all messages on the wire addressed to the device. Discovery involves broadcasting a request/response message, and then processing the first response, which closes out the transaction. Other responses may be received after the matching transaction has completed.

When using the standard Lonworks driver (**LonNetwork**), these warning messages do not appear because the Neuron processes the messages off the wire, and filters out all except the first response.

Setting up a LonIp channel with a third-party Config Server

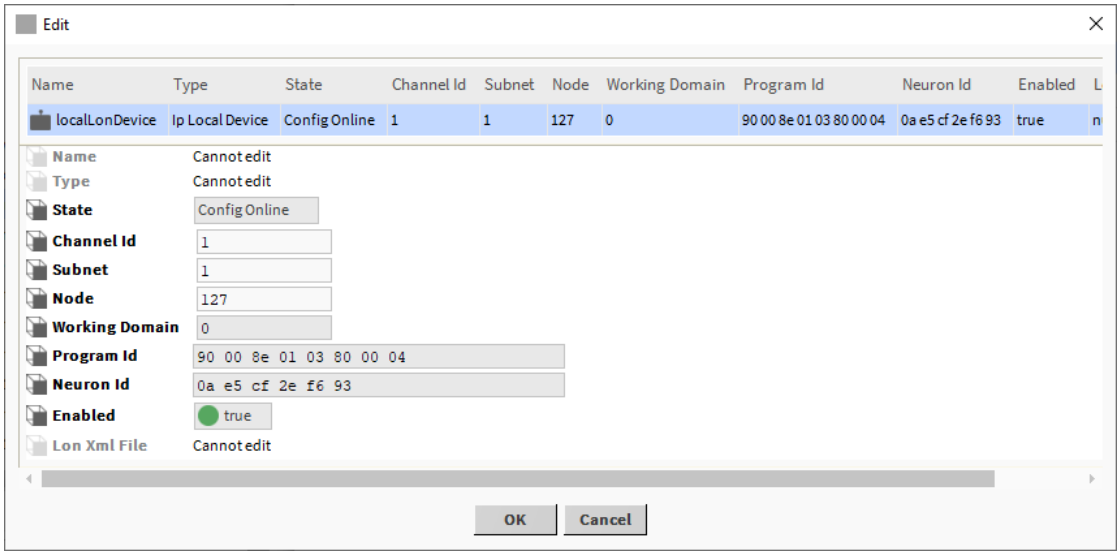
This setup method applies when the host is installed on a network that includes a configured third-party Config Server and CEA-852 routers.

Prerequisites: The station is open in Workbench.

Step 1 Open the **lonIp** palette in the palette side bar and drag or copy a **LonIpNetwork** component to the station's **Config→Drivers** container.

Step 2 Double–click the **LonIpNetwork** component (opens the **Device Manager**), then double-click the network's **Local Lon Device** (or select the **Local Lon Device** and click **Edit**).

The local Lon device **Edit** window opens.



Step 3 Configure the device's **Channel Id**, **Subnet** and **Node** to be consistent with the LonIp channel, which the station will join as a member, then click **OK**.

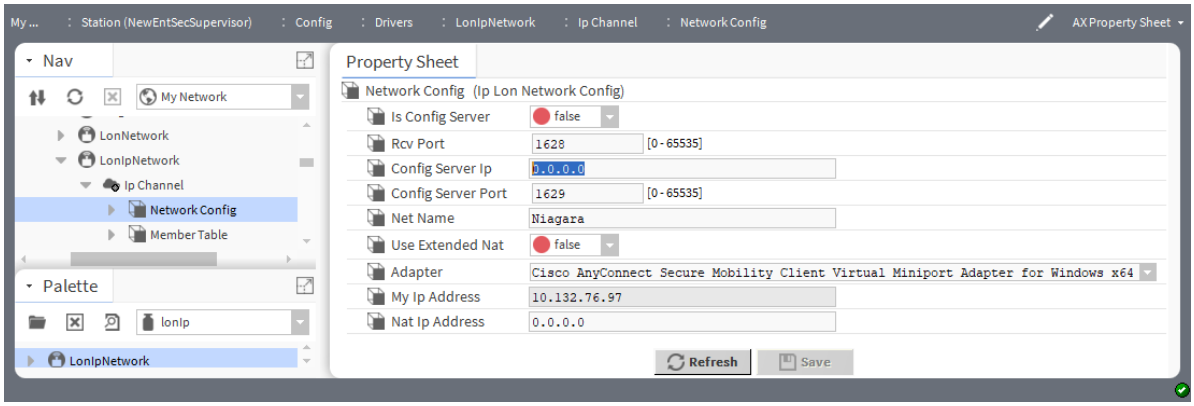
The third-party Config Server/CEA–852 router interface provides the **Channel Id** and **Subnet**. Choose a **Node Id** for the **LonIpNetowrk**'s **Local Lon Device** that is not already in use on the subnet.

Step 4 Expand the **LonIPNetwork→ipChannel** and double-click the **Network Config** container.

The Network Config **Property Sheet** opens.

Step 5 Set **Is Config Server** to `false`, click **OK** and click **Refresh**.

The **Config Server Ip** property appears in the **Property Sheet**.



Step 6 Configure these properties:

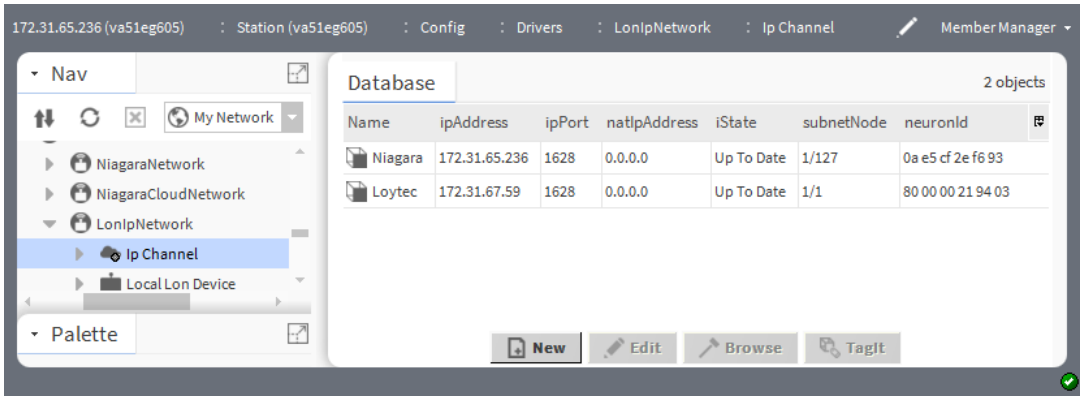
- For **`Config Server Ip`**, enter the server IP address of the third-party Config Server.
- If the host is behind a NAT router, set **`Use Extended Nat`** to `true`, and enter the **`Nat Ip Address`** (the Internet side of the NAT router).

Step 7 To continue, click **Save**.

The driver automatically forces **`User Extended Nat`** to `true` if it detects extended NAT messages.

Step 8 Double-click on **Ip Channel**.

The **Member Manager** view opens.



Step 9 Right-cick the channel member (row) that represents this station and click **Actions→Update**.

Step 10 At the third-party Config Server, add a member to the IP channel for the station. Refer to the vendor's documentation for the Config Server for this step.

Unlike the situation where the station functions as the Config Server, you do not need to edit or add any members in this view. Once the third-party Config Server is configured, it makes contact with the station. Then, the LonIp driver in the station dynamically populates the **Member Table** under the network's **Ip Channel** container. From this point forward, other Lonworks functions perform as if directly connected to a Lon trunk, that is, as if using a standard Lon network.

The next logical step may be to commission a Lon router. It is likely that at least one Lon router needs to be added.

Setting up a LonIp channel as the Config Server

Use this method when the host is installed on a network that has configured CEA-852 routers, but there is no Config Server.

Prerequisites: The station is open in Workbench.

Step 1 Open the **lonIp** palette in the palette side bar and drag or copy a **LonIpNetwork** to the station's **Drivers** container.

Step 2 Double-click the **LonIpNetwork** container, then double-click its **Local Lon Device** or select the **Local Lon Device** and click **Edit**.

The local Lon device **Edit** window opens.

Step 3 Edit the device's **Channel Id**, **Subnet**, and **Node** address to be consistent with the LonIp channel that the station will join as a member, then click **OK**.

The **Channel Id** and **Subnet** must match the other devices on the channel, and the **Node Address** must be unique for that subnet.

Step 4 Expand the **LonIpNetwork→ipChannel** and double-click the **Network Config** container.

Step 5 If **Is Config Server** is set to `false`, change it to `true`, click **Save** and click **Refresh**.

The default for **Is Config Server** is `true`. The **Config Server Ip** property is available only when **Is Config Server** is set to `false`.

Step 6 Double-click the network's **ipChannel**.

The **Member Manager** view opens.

Step 7 To add a member for each CEA-852 router and also one for the station, click **New**, select the number to add and click **OK**.

Step 8 For each member, edit the device's **ipAddress**, **ipPort**, and, if needed, **natIpAddress**.

- For the channel member that represents the station, configure **ipAddress** with the host's own IP address. If you are using defaults, for **Name** enter the channel name and leave the other properties (**ipPort** and **natIpAddress**) unchanged.
- For any member behind a NAT router, use the address on the local network for the member's **ipAddress** and the NAT router address on the Internet side for the **natIpAddress**.

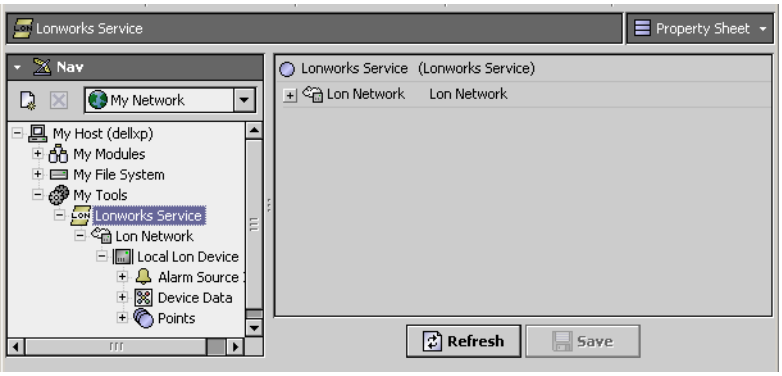
The driver configures all other properties (starting with **routerType**) using data obtained from each device.

Station-less access

If your Workbench PC has a Lonworks adapter, you can use the Lonworks Service tool to access the **Lon Network** as if you had a station running on a local PC.

Once started, you use the **Lon Device Manager** to discover, add, and upload Lon devices, examine nvs and ncis as Lon components, and even perform writes and perform other tasks (make bindings between devices, commission devices, etc.).

Figure 2 Lonworks Service tool in Workbench



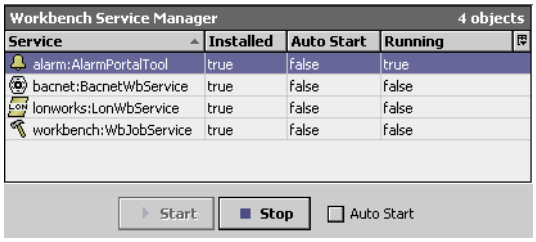
CAUTION: Any configuration you perform is not stored (persisted) in a station database, as this is station-less access (modeled completely in RAM). When you close Workbench, all modeling is lost—consider the Lonworks Service like a hand held device in this respect.
For this reason, do not use this tool to access a Lonworks network already managed by a station, but instead open that station, and access its**LonNetwork** component.

Starting and stopping Lonworks Service

The Lonworks Service, like a few other tools, is managed from the **Workbench Service Manager**. Using the manager, you can start and/or stop the Lonworks Service. You must start the Lonworks Service to use it.

Step 1 Click **Tools→Workbench Service Manager**.

The **Workbench Service Manager** view opens.



Step 2 To start Lonworks Service, select `Lonworks:LonWbService`, and click **Start**.

Step 3 To stop Lonworks Service, select `Lonworks:LonWbService`, and click **Stop**.

Variable (Lon component) management

Communication between nodes on a **Lon Network** uses network variables (inputs and outputs), which must be defined for each node (device). Multiple nodes share these variables. Some variables send information, while others receive. Lon networks link only devices whose network variables are of the same type.

Variables have manager views, including the **Local Nv Manager** view, which you open by double-clicking the **Local Lon Device** node in the Nav tree, and the **Nv Manager** and **Nc Manager**, which you open by right-clicking a **DynamicDevice** in the Nav tree, and clicking **Views→Nv Manager** or **Views→Nc Manager**.

Variables cannot be discovered. Instead, you add them as needed. In general, you should finalize the creation of the nvis, nvos, and ncis variables before you add associated proxy points. This means reviewing and, if necessary, editing the name, direction, and SNVT for each entry.

Following this initial configuration, if you change, add to, or delete from the definition of any network variable, you need to perform (external) network management on the host node to maintain proper network configuration. This includes maintenance of any bindings to the host.

Setting up variables for network components

As with other Lon devices, Lon network components communicate over the Lon network using variables. Variables are the entities that carry information from an output to an input.

Prerequisites: You are configuring the **Local Lon Device** under the **LonNetwork**.

Step 1 Double-click the **Local Lon Device** in the Nav tree.

The **Local Nv Manager** view opens.

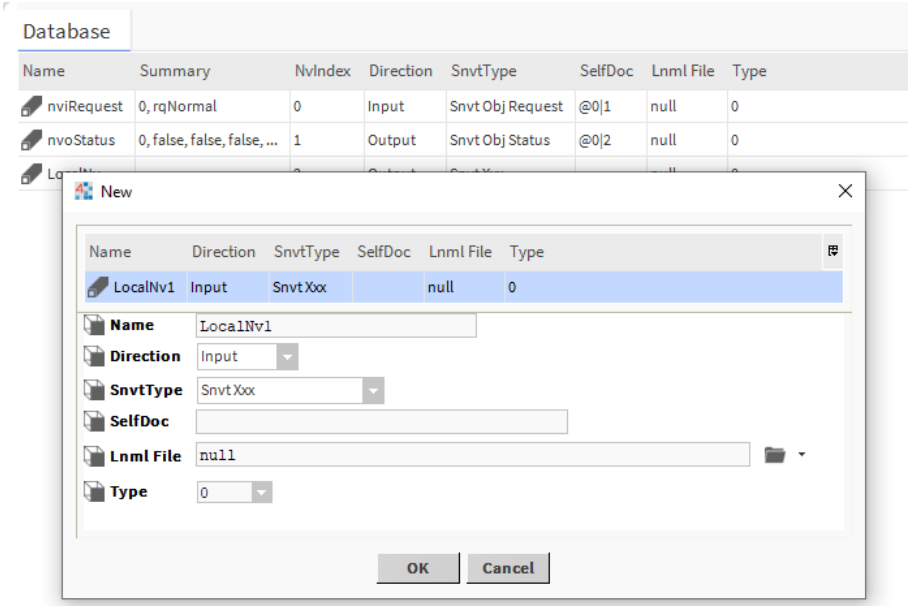
Step 2 To create a variable click the **New** button.

The **New** variable window opens.

Two variables are available for local Lon devices. Local Nv stands for Local Network Variable. Local Nci stands for Local Network configurable input.

Step 3 Select the type of variable from the drop-down list, the number to add, and click **OK**.

A second **New** window opens.



Step 4 Replace the default name (for example, `LocalNv`) with a unique name for each variable.

You change default names (`LocalNvn` or `LocalNcin`) to a unique name using a prefix, such as `nvi` (network variable input) and `nvo` (network variable output) followed by text, which identifies the type of information. For example: `nviSetpointA`, `nvoUnitStatusA`, `nciOffsetB`, and so on.

Step 5 Select a SNVT (Standard Network Variable Type) from the drop-down list.

An alternate way to configure a variable type is to define a UNVT (User Network Variable Type) in a Lon Xml file and associate it with the variable.

Step 6 Do one of the following:

- To omit self documentation, enter an asterisk (*) for **`Self Doc`**.
- To specify self documentation, enter a text string for **`Self Doc`** that adheres to this syntax: `@fbIndex|memberNum;text`, where:
  - `@` denotes the start of nv self documentation.
  - `fbIndex` is the functional block index.
  - `|` (pipe) is the functional profile separator.
  - `memberNum` is the member number within the functional block.
  - `text` is optional text for describing the network variable.

    By convention this syntax conforms to: nvi`Something` or nvo`Something`

  NOTE: A value other than asterisk (*) and that uses the syntax above is required in the **`Self Doc`** property of the two required network variables: `nviObjRequest` and `nvoObjStatus`.

Below are two common **`Self Doc`** property values for the two required node object nvs:

- nviObjRequest **`Self Doc`** = `@0|1;nviRequest`
- nvoObjStatus **`Self Doc`** = `@0|2;nvoStatus`

After creating the two required nvs, you can continue defining/adding other local nvs and ncis. Optionally, you can also add **Self Doc** strings to other local nvs and ncis, and also edit the **Self Doc** string for the **Local Lon Device**.

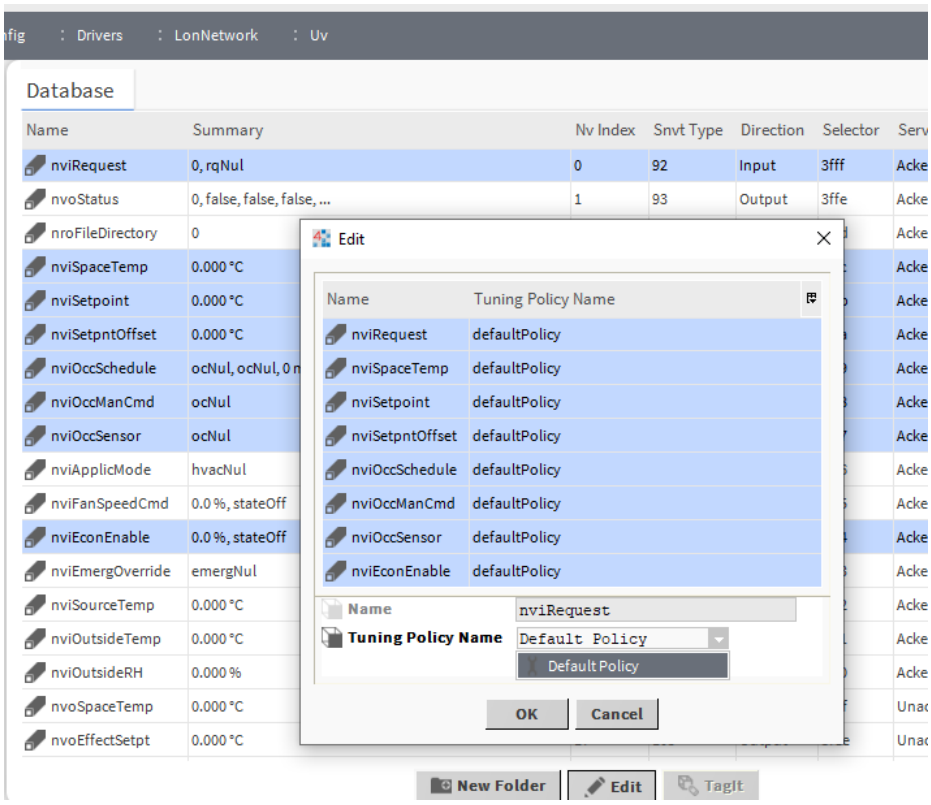
To omit self documentation (for any items except the two Node object 0 nvs described above), enter a single asterisk (*) in the **Self Doc** property.

Editing variable properties

The **Nv Manager** and **Nc Manager** views provide editing features. You can double-click any single row in these views to change the variable name or tuning policy. You use this gang **Edit** Window the tuning policy used by a group of network variables.

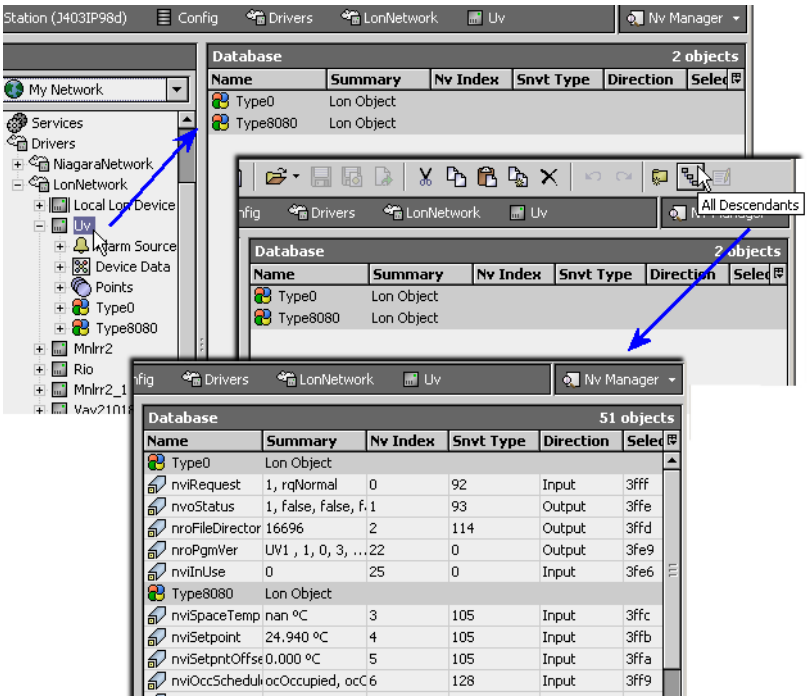
Step 1 Right-click a dynamic device in the Nav tree and click **Views→Nv Manager**.

The **Nv Manager** view opens.



If you are using the **Use Lon Objects** feature in the **Lon Netmgmt** container, the **Nv Manager** (or **Nc Manager**) view initially shows only the top-level LonObject(s), where each contains a set of Lon components.

Step 2 If you are seeing only the top-level LonObject(s), click the **All Descendants** tool to see the nvs or ncis under each object.

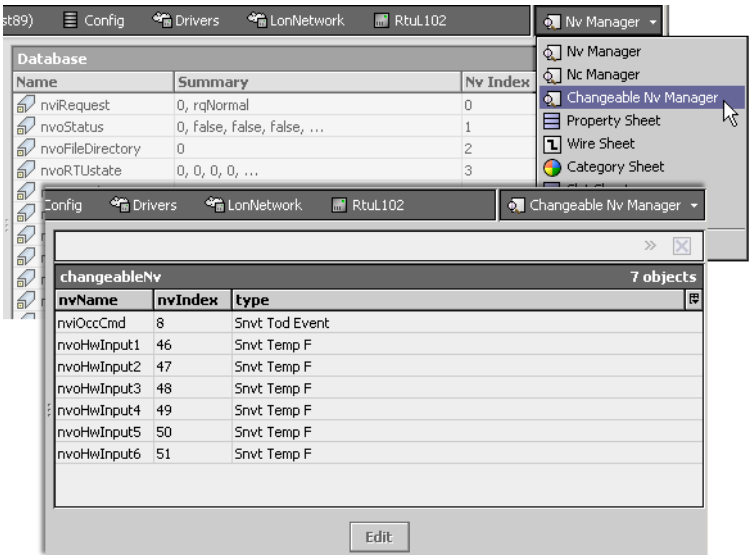


Editing changeable variable properties

This procedure documents how to change the **`Snvt type`** for changeable variables.

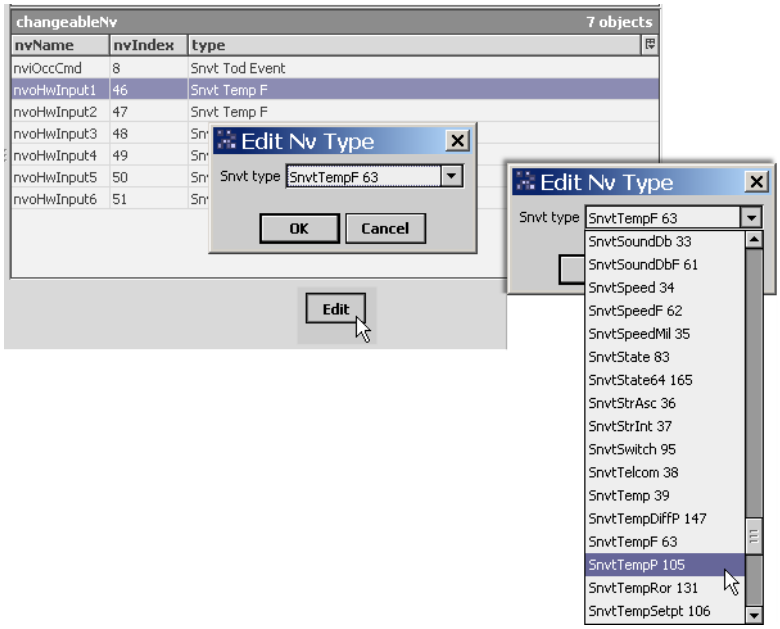
Step 1 Right-click a node in the Nav tree for a device that supports changeable network variables and click **Views→Changeable Nv Manager**.

The **Changeable Nv Manager** window opens.



Step 2 Select one or more of these variables and click the **Edit** button.

The **Edit Nv Type** window opens.



Step 3 Select another **Snvt type** from the drop-down list (ordered alphabetically, each type also includes the **Snvt type** index number), and click **OK**.

The system changes the network variables in the device to use the new **Snvt type**.

Chapter 3 Device network management

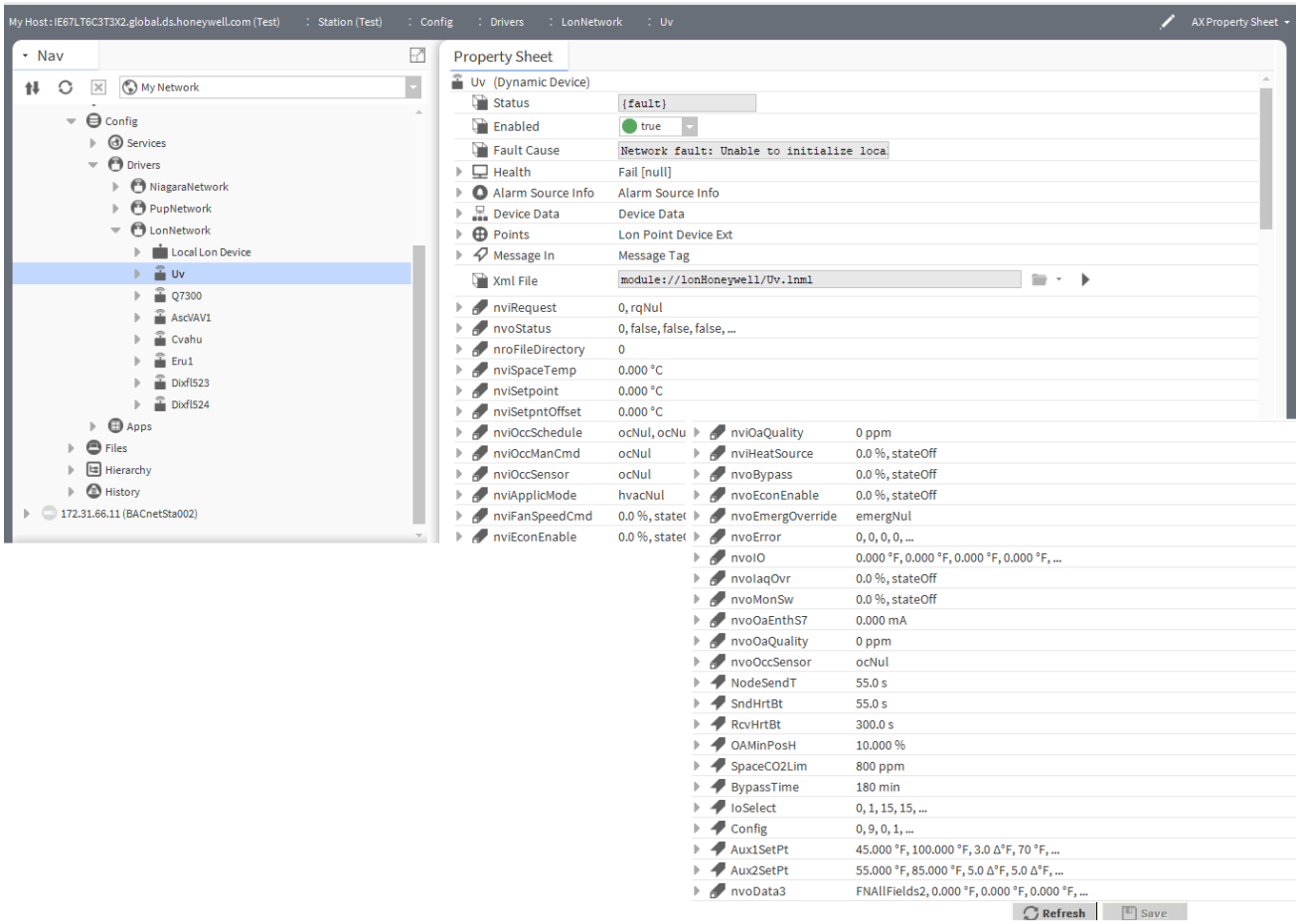
Topics covered in this chapter

- ♦ Identifying a device using its service pin
- ♦ Discovering Lon devices
- ♦ Viewing a node's current domain table
- ♦ Learning Lon devices
- ♦ Adding devices using a Lon palette
- ♦ Adding a dynamic device
- ♦ Assigning nvs using a Lon Xml file
- ♦ Matching devices
- ♦ Setting up device tables
- ♦ Replacing a device
- ♦ Installing an app in a device
- ♦ Configuring a device for LNS compatibility

Lon device components are similar to most driver device components. Among the standard slots are device status properties (including a **Health** container) and a device **Alarm Source Info** container slot. Each Lon device includes a **Points** extension for Lon proxy points. However, unlike a few other drivers, other device extensions, such as **Schedules**, **Histories**, and **Alarms** are not used. Instead, other components unique to a Lon device appear.

The driver supports two device components: the required **Local Lon Device**, and a **DynamicDevice** component.

Figure 3 Lon component (nvs, ncis, cps) access from LonDevice Property Sheet



In addition to standard device properties, each Lon device includes a **DeviceData** container, with a number of properties for status and configuration of that particular device. You do not need to edit anything under this container.

Unique to a Lon device is its set of Lon components that represent specific network variables (nvis and nvos), ncis, and cps for the device.

Main access to Lon components is via the Property Sheet of the Lon device, where Lon components are listed in nv index order below the Xml file property.

Also unique to a Lon device is a special Workbench command, special manager views, and right-click actions.

Ways to populate devices

The driver supports multiple ways to populate a Lon network with devices. You may discover devices, learn devices, add them to the offline database and later match them, and you can add them one-by-one when online.

Each Lon vendor module contains a palette with a device for each Lon Xml (`.lnml`) file in that module. You can drag devices from an open palette to a **Lon Network**.

Alternatively, you can select a **`DynamicDevice`** from the **`lonworks`** palette and specify a Lon Xml file (extension `.lnml`) that identifies the particular device type. A device is associated with a particular Lon Xml file using its Lonworks program ID.

Later, when online with the **Lon Network**, you can use the Discover and Match features in the **Lon Device Manager** view to map each device object with a particular discovered Lon node. A match is possible only if

the program IDs are the same. The match synchronizes the Lonworks **Neuron Id** and applies appropriate subnet-node addressing. The Lonworks match offers an address source toggle, which you select depending on whether the network is already managed or unmanaged.

Local Lon Device startup

Upon first execution (if Neuron Id is Zero), the following sequence occurs:

1. Address is set to default: channel Id 1, subnet node 1/127.
2. DeviceData is filled in from local neuron: Neuron Id, Address Count, Two Domains.

Upon station startup, the following occurs:

1. Updates to domain table, address table, and device state per station database.
2. Program Id is set.
3. Executes ping.

Identifying a device using its service pin

If you are unable to discover nodes that are physically connected to the station, they may be configured on a different domain. Use the following method to gain access to a device by use of its Lonworks Service pin.

Step 1 Right-click the network and click **Views→Lon Utilities Manager**.

The **Lon Utilities Manager** view opens.

Step 2 Near the bottom, click the lower-left drop-down control (command menu) and select **Identify**.

Step 3 Click the drop-down **Identify** control to the right (command submenu) and select **Service Pin**.

NOTE: For this command, the selected Lon device makes no difference (`Local Lon Device`, other).

Step 4 Click the **Execute** button ( ).

The view updates to show a timestamp with the message `Waiting on service pin` below.

Step 5 At any connected Lon device, press its service pin.

The **Lon Utilities Manager** view shows the device's domain table, including the domain length (**Domain Len**) and the ID for the device. If the device is configured on something other than the zero-length domain (**Domain Len** is not `0`), you must configure the network's working domain to match.

By default, when expressly listening for a service pin message from the **Lon Utilities Manager** view, you have 300 seconds of listen time for a service pin before the utilities manager cancels the wait period.

You configure both the **Domain Id** and the **Service Pin Wait** under the Lon network's **LonNetmgmt** container.

Discovering Lon devices

After adding the network, you can use online discovery to find and create Lon devices (also referred to as Lon nodes). For a new network, instead of adding devices, you use Quik Learn to populate your network.

Prerequisites: The host is connected to the Lon network and devices are ready to be added to the station database.

Step 1 Expand **Config→Drivers** and double-click the **LonNetwork** in the Nav tree.

The **Lon Device Manager** opens.

Step 2 To discover devices in the working domain (or, if different, the zero-length domain), click the **Discover** button.

A progress bar appears at the top of the view and updates as the discovery job progresses. If the node is not represented in the database, the discovery job adds it to the **Discovered** pane (learn mode). If the node is already in the database, the discovery job selects (highlights) it in the **Database** pane. Initially, this table is empty.

Any Lon node already represented in the station's Lon network does not appear in learn mode's **Discovered** pane. This differs from other drivers, where an existing discovered device appears ghosted in the **Discovered** pane.

If you do not see an expected node, it is likely configured on a different domain than either the network's working domain or the zero-length domain. To see such a node, press its Lonworks Service pin. It should now appear listed in the **Discovered** pane.

Viewing a node's current domain table

A device's current domain table includes the domain length (**`Domain Len`**) and **`ID`** for the device.

Prerequisites: The driver found the device from its service pin.

Step 1 Right-click the network and click **Views→Lon Utilities Manager**.

The **Lon Utilities Manager** view opens.

Step 2 Near the bottom, click the (device) drop-down and select a node found from a service pin.

Step 3 Click the lower-left drop-down control (command menu) and select **Data Structs**.

Step 4 To the right (command submenu), click the drop-down control and select **Domain Table** and click the Execute button ( ).

The domain table for the device opens. If the device is configured on something other than the zero-length domain (**`Domain Len`** not `0`), you must configure the network's working domain to match.

Learning Lon devices

An available Quik Learn feature (misspelling intended) populate a new Lon network with the proper device components. In a managed network, Quik Learn preserves the original Lon network management configuration. In an unmanaged network, Quik Learn applies unique and contiguous subnet-node addressing to the learned devices.

Prerequisites: For the best possible results, make sure that any needed Lon modules are installed on the host platform before running Quik Learn.

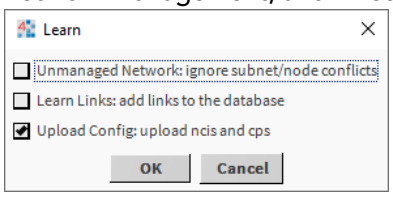
Step 1 Double-click the network node.

The **Lon Device Manager** view opens.

Step 2 If you see a split-pane (learn mode), click the Learn Mode tool ( ) to toggle out of learn mode.

Step 3 You may select devices before clicking the **Quik Learn** button or click without selecting devices.

If you clicked **Quik Learn** with no devices in the database selected, a window asks about previous network management, and whether to learn links, with config upload preselected.

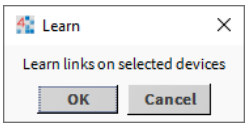


You use Quik Learn to populate a new Lon network with the proper device components. You can also use it to add new devices to an already configured Lon network, in which case, you would select the Unconfigured network option.

Quik Learn operation differs depending on if you have devices selected or not when you issue the command. If you click **Quik Learn** with no devices in the database selected, a popup window asks about previous network management, and whether to learn links, with config upload preselected.

For the best possible results, make sure that any needed Lon modules are installed on the host platform before running Quik Learn.

If you clicked **Quik Learn** with one or more devices in the **Database** pane selected, a window asks if you wish to learn links.



Step 4 Do one of the following:

- For a managed network with no devices selected, leave the top option cleared, and either check (or clear) the bottom options `Learn Links: add links to the database` and `Upload Config: upload ncis and cps`, then click **OK**.
- For an unmanaged network with no devices selected, check `Unmanaged Network: ignore subnet/node conflicts`, and `Upload Config: upload ncis and cps`, then click **OK**.
- For any network with devices selected, clicking **Cancel** aborts the job. Clicking **OK** initiates the learn job.

A progress bar appears at the top of the view, and updates as the Lon learn job runs.

For selected devices, the job verifies **`Subnet`**, **`Node Address`**, and **`Domain Id`**. If they do not match it aborts the job. Then it retrieves device data, including **`Program Id`**, **`Node State`**, **`Authenticated`**, **`Two Domains`**. **`Working domain`**, and **`Channel Id`**. Finally, it uploads **`Address`**, **`nvConfig`**, and **`Alias Tables`**.

When the learn job completes, the **Database** pane displays the learned Lon devices. All devices in a managed network should appear listed with unique subnet/node addresses.

Step 5 For unmanaged networks, click to select all devices, and click the **Commission** button.

A progress bar appears at the top of the view, and updates as the Lon Commission job runs. When the commission job completes, all devices should appear listed with a Config Online state, and with unique subnet/node addresses.

Step 6 To view links and learned bindings, right-click the network node and click **Views→Lon Link Manager**.

The **Lon Link Manager** view opens.

Step 7 To see a device's ncis values, right-click the device and select **Views→Nc Manager**.

The following lists in detail what happens during a Lon learn job when no devices are selected:

1. A discovery occurs for all nodes on the working domain (only). As each device is discovered:

   If you specified a managed network, the job ignores unconfigured nodes, and checks for duplicate subnet-node address conflicts. If it finds conflicts, it aborts the learn.

   Device data is retrieved for each node, including neuronId, programId, nodeState, authenticated, twoDomains, workingDomain, and channelId.

   The job attempts to match with the existing device (subnet, node, programId with neuronId=0). If it cannot match the device, it saves device data for the node until all devices are discovered.

2. Continuing, the Quik Learn job processes the remaining unmatched devices. For each unmatched device it:

   - Attempts to match the programId match (with neuronId=0) with existing device.

- If no match is possible, the job creates a new **DynamicDevice**, where it uses the node's **Program Is** to find the appropriate Lon Xml file among the host's locally installed **lon**vendor modules.

3. As the job matches or creates each device, it synchronizes the database's device data with the data read from the physical node.
4. If learn links were selected, the job configures these data:

   For each device, **Upload Address**, **nvConfig**, and **Alias Tables**.

   For each device it adds links.

   It configures **Location** per the Lon device property.

   It configures **Authenticate** per the property under **LonNetmgmt**.
5. If upload config was selected, for each device the job uploads the values of its ncis and cps.

NOTE: At this point forward, the station must become the only Lonworks network manager. Otherwise, proper network management cannot be successfully maintained.

Adding devices using a Lon palette

If you know the Lon device types you will have under your network, you may wish to add devices to the database offline, or even manually add them when online. When adding a new device, you can select a device component from a specific **lon<Vendor>** palette.

Prerequisites: You are connected to the remote host using Workbench.

Step 1 In Workbench, open the **Palette** side bar, if not already open.

Step 2 Open any of the **lon**<Vendor> palettes with devices of interest, for example, **lonHoneywell**.

Available devices are listed by default names matching the source .lnml file name.

Step 3 Open the target view of the target Lon network, or of the target **Lon Device Folder** under the Lon network.

This may be the **Device Manager** view or the wire sheet view.

Step 4 Drag a device from the palette to the target view or from the palette to the Lon network node in the Nav tree).

A popup **Name** window opens, in which you can enter a meaningful name.

Step 5 Enter a name or accept the default, and click **OK**.

The driver adds the device to the station database.

At the time of this document, you should also perform additional steps below

Step 6 If the station in a remote host does not have the same **lon**<Vendor> module already installed as the local module palette that you are copying from, right-click the newly-copied device, and on its popup menu select the ImportXml command.

The **ImportXml** window opens, showing module://lon<Vendor>/<device>.lnml, which corresponds to the device you just dragged from the palette.

Step 7 Click **OK**.

The driver reads the data from the local .lnml file into the device component in the remote station. The device is now ready for offline programming or if, online with the host and Lon network, it is ready for matching with an actual device.

Adding a dynamic device

A dynamic device is a Lon device that comes with self documentation or a Lon Xml (lnml) file that configures the device. Almost all Lon devices are dynamic device components.

Step 1 Double-click the network node in the Nav side bar.

The **Lon Device Manager** opens.

Step 2 Click the **New** button.

This procedure uses the New device wizard to create the device. You may also drag or copy a **DynamicDevice** from the **lonworks** palette to the Lon network container in the Nav tree.

The **New** device wizard window opens.

Step 3 In **Type To Add** property, select `Dynamic Device`, enter the number of devices to add (the default value is `1`), and click **OK**

Alternate types to add may appear beside `Dynamic Device`, for example, `Q7300`, `Xl10 Chc1` and `Xl10 Hyd2`. These are special subclasses of a Lon device. The fact that these devices appear as options indicates that their Lon Xml (`.lnml`) files already exist.

A second **New** window opens.

Step 4 Edit the name of the component as you wish to see it in the station, and click **OK**.

The driver adds the component to the station where it appears in the **Database** pane as type `DynamicDevice`.

Assigning nvs using a Lon Xml file

The Lon Xml file (extension `.lnml`) for each device specifies its Program Id and the types of messages and variables (nvs = network variables) it is capable of sending and receiving. This procedure associates a Lon Xml file with a dynamic device that is already in the station database.

Prerequisites: The dynamic device exists in the station database. The Lon Xml file exists.

Step 1 Right-click the dynamic device node in the Nav tree, and do one of the following:

- Click the `ImportXml` command, click the folder icon to the right of the **Xml File** property, and select `File Ord Chooser` from the drop-down list.
- Click **Views→Property Sheet**, click the folder icon to the right of the **Xml File** property, and click the `File Ord Chooser` from the drop-down list.

The **File Chooser** opens.

Step 2 Click My Modules in the File Spaces pane, scroll down to the files that begin with "lon," and double-click the Lon module for your device.

The list of available modules opens in the center pane. This list includes Lon Xml file previously made using the **Lon Xml Tool**.

Step 3 Double-click the module file.

The **File Chooser** opens a list of the available `.lnml` files for the selected device. For example, `lonSiebe` includes these `.lnml` files: `Mnlrh2.lnml`, `Mnlrh3.lnml`, and so on. Each file name corresponds to the known `xif` file name for a device.

Use the device manufacturer's documentation to determine which file to choose.

Step 4 Click an `.lnml` file, then click **Open**.

The **File Chooser** closes, and the **Import Xml** window shows the selected `.lnml` file.

Step 5 To import the `.lnml` file into the dynamic device component, click **OK**.

The property sheet lists the Network Variable inputs and outputs (nvis and nvos) that are associated with the device.

Now that the available data items are known, you can use the **Lon Point Manager** under that device to add (offline) proxy points.

Matching devices

If you added devices to a Lon network while offline, you can go online, discover devices, and match them with their existing database records. This copies discovered address data into the database record for the Lon device component. This procedure works for both managed nodes (network nodes already configured with unique identifying information by an external Lon tool) and unmanaged nodes (unconfigured network nodes, each of which may default to the same identifying information.)

Prerequisites: The device record(s) exists in the station database.

Step 1 Double-click the Lon network node in the Nav tree, and, if the **Discovered** pane is empty, click the **Discover** button.

The system populates the **Discovered** pane with discovered devices.

This window has an available toggle in which you select whether to use the subnet-node address of the discovered (network) node, or have address management performed automatically by the **Lon Device Manager**.

Step 2 Select (highlight) a discovered device in the **Discovered** pane and its database record in the **Database** pane, click the drop-down control on the **Match** button and do one of the following:

- For a managed Lon network, click `Use Net Subnet/Node`.
- For an unmanaged Lon network, click `Use Db Subnet/Node`.

This identifies which device to match with which database record. For managed networks it specifies to upload address data from the node. For unmanaged networks it specifies automatic assignment of subnet/node addresses.

Step 3 Click the **Match** button.

The system removes the device from the **Discovered** pane and leaves it selected in the **Database** pane.

Step 4 Right-click the device, select **Actions→Upload**, and check all (recursive, transient, persistent).

This configures the job to retrieve current configuration data from the device.

Step 5 Repeat this process for each device (previously-managed and previously-unmanaged).

Step 6 When all devices are matched, click the Learn Mode tool ( ).

This toggles out of learn mode.

Step 7 For managed networks, click to select (highlight) all the devices just matched, click the **Quik Learn** button, and click **OK**.

The system launches a learn links job, where it learns the Lon bindings for the devices from each device in the managed network. When complete, all devices should appear listed with a `Config Online` state, and with unique subnet/node addresses.

Setting up device tables

The **Commission** button in the **Lon Device Manager** sets up a device's internal tables to a functioning but unbound state. You perform this on any newly-added Lon device except those created by a Quik Learn of a previously managed network, where the initial state of the device is already set to `Config Online`.

Prerequisites: The network is an unmanaged network. The Lon Device Manager view is open and you have already discovered devices.

Step 1 Before clicking the **Commission** button, you can examine the device's (discovered node's) internal tables by right-clicking the **LonNetwork** node in the Nav tree and clicking **Views→Lon Utilities Manager.**

Step 2 Select the device from the drop-down list at the bottom of the view.

Step 3 To return to the **Lon Device Manager** view, double-click the **LonNetwork** node in the Nav tree.

Step 4 Click to select all unconfigured devices, and click the **Commission** button.

A progress bar appears at the top of the view, and updates as the Lon Commission job executes.

The job initializes the device domain table as follows:

1. For a DynamicDevice with which you are using a service pin, the job waits for a service pin message to obtain the node's **Neuron Id**, otherwise, it uses the **Neuron Id** already stored. Then it reads from the device (for example: hosted, two domains, address count, address of snvt self-doc table). These two actions do not occur when commissioning a **Local Lon Device**.
2. The job sets the domain index 0 to the Lon network's working domain.
3. If the device has two domains, it sets domain index 1 to not-in-use. If the network's **Always In Zero Length Domain** property is set to `true`, and the Lon network's working domain is not the zero-length domain, the job sets the second domain entry to the zero-length domain.
4. Next, it configures the **Netmgmt** **`Authentication Key`** and the device's subnet/node address in all active domains.
5. Continuing, the commission job sets all entries in the address table to not-in-use.
6. For DynamicDevices, it sets all entries in nvconfig to unbound selector (0x3FFF - nvIndex). If under **LonNetmgmt**, it sets **`VerifyNvDir`** and checks to ensure that device and database nv's have matching direction.
7. If there is a **`configSrv`** nv, the job sets it to external.
8. The commission job updates device data:

   `Channel ld`: per property under **LonNetmgmt**.

   `Node Priority`: per Lon device property.

   `Location`: per Lon device property.

   `Authenticate`: per property under **LonNetmgmt**.
9. Finally, the commission job sets Node to the state of: `Configured, online`.

Replacing a device

The Replace action in the **Lon Device Manager** downloads network management config data to a selected device. You use it when physically replacing a device (you have removed the old device, and replaced it with an identical type device). You may also use this procedure if running a verify report from the **Lon Utilities Manager** on a device that has generated errors.

Prerequisites: The old device has been removed and the replacement device is installed and ready. The kitLon palette is open.

Replacement can use the service pin method or the direct entry of the new Neuron Id. This component may be useful in appliance scenarios, where you access the system primarily using a browser and Px pages.

Step 1 Drag the **`LonReplace`** component from the **`kitLon`** palette to the Lon device (or devices).

Step 2 Double-click the **LonNetwork** node in the nav tree.

Step 3 Select the device to replace in the **Database** pane and click the **Replace** button.

The replace job performs these steps:

1. If you are using a service pin, the job waits for a service pin message to obtain the nodes **`Neuron Id`**, otherwise, it uses the **`Neuron ID`** already stored.
2. The job verifies that the new (replacement) device's **`Program Id`** matches the **`Program Id`** stored in the existing Lon device.

3. It initializes the device's domain table as follows:

   It sets Domain index 0 to the Lon network's working domain.

   If the device has two domains, it sets domain index 1 to not-in-use.

   If the network's **Always In Zero Length Domain**property is set to `true`, and the Lon network's working domain is not the zero-length domain, the job sets the second domain entry to the zero-length domain.

   It configures the **Netmgmt `Authentication Key`** and the device's subnet/node address in all active domains.
4. Continuing, it sets all entries in the address table to match the station database.
5. It sets all entries in nvConfig to match the station database.
6. It updates configuration device data as follows:

   `Channel ld`: per property under **LonNetmgmt**.

   `Node Priority`: per Lon device property.

   `Location`: per Lon device property.

   `Authenticate`: per property under **LonNetmgmt**.
7. Finally, the commission job sets Node to the state of: `Configured, online`.

   NOTE: You can examine a device's/discovered node's internal tables using the **Lon Utilities Manager**.

Installing an app in a device

The **AppDownload** function in the **Lon Device Manager** provides a means to download a vendor-supplied application file (of type: `*.nxe`) to one or more selected Lon devices.

An `*.nxe` file contains a binary application image for loading in a specific Lonworks device. If the download changes the external interface of the device, you must also supply either an `*.xif` or `*.lnml` file to correctly set the network variable data, which is represented in the station database separately from the device's application. Different external interfaces should be associated with unique program IDs. Some vendors support multiple device types in the same hardware platform.

Step 1 Select a device and click **AppDownload**.

A popup window asks if you to specify the application (*.nxe) file.

Step 2 Click the folder icon (for the **File Chooser**), and navigate to select the needed `.nxe` file.

Step 3 If the application download changes the device's external interface, select either a matching `.xif` or `.lnml` file.

If you supply both, the system uses the `.xif` and ignores the `.lnml`.

Step 4 To continue, click **OK**.

The job verifies that the supplied file(s) are for the same device type. If files they are for a device with a different programId, a prompt verifies that the change of device type is acceptable to you.

Step 5 If the change is acceptable click **OK**.

The download executes.

Configuring a device for LNS compatibility

An LNS (L2TP Network Server) device requires a unique configuration of the **`Self Doc`** property.

Prerequisites: You have already discovered or otherwise added the device to the Lon network.

Step 1 Expand the **LonNetwork** in the Nav tree, right-click **Local Lon Device** and click **Views→Property Sheet**.

The local Lon device **Property Sheet** view opens.

Step 2 For the **Self Doc** property either enter an asterisk (*), which disables self documentation, or enter a string that includes all LonMark objects in the station.

Use this syntax: `&3.0@0NodeObjectName,FunctionalBlock;selfDocText` where, from (left-to-right):

`&` denotes an interoperable device.

`3.0` identifies the major and minor version of Lonworks Interoperablility Guidelines used.

`@` is used as a separator.

`0` is the first of the indices of the corresponding functional block index (0, 1, 2, etc.). Node object must be the first of the indices (0).

`NodeObjectName` is the description of the Node object.

`FunctionalBlock` includes any other LonMark or non-LonMark functional profiles to add to the station database.

`Self DocText` text to describe the functional profiles.

For example: `&3.0@0NodeObject;NiagaraAX Server Node`

Step 3 To continue, double-click the **Local Lon Device** in the Nav tree.

The **Local Nv Manager** view opens.

For LNS compatibility, the Node object represented by the **Local Lon Device** also requires that you create two mandatory network variables: **SNVT_obj_request** and **SNVT_obj_status**.

Step 4 To create a variable, click the **New** button.

Chapter 4 Device variables and properties

Topics covered in this chapter

♦ Creating a Lon Xml file
♦ Differential temps in Lon Xml files
♦ Configuring a device offline by importing a Lon Xml file

A device's Lon Xml file defines each device's unique variables and properties.

Lon Xml files

Lon Xml files describe specific Lonworks device properties to the system. The file name extension for these files is `.lnml`. Each device requires its own Lon Xml file. Workbench provides many **lon**`Vendor` modules (`.jar` files), for example, **lonHoneywell**. Each module contains numerous Lon Xml files for specific devices. You may also create additional Lon Xml files for devices that are not included in the standard modules.

Each Lon Xml file includes data to describe the device itself (derived from an `.xif` file), as well as manufacturer-specific datatypes for nvs, ncis, and cps (derived from resource files). By convention, a device's Lon Xml file has the same name as its xif file (except for extension). For example, a device with a `T7350Cs.xif` file has a `T7350Cs.lnml` file.

During a **Lon Device Manager** discover, the driver automatically searches each Lon Xml file from which it displays the Program Id for each device. Or, when the needed **lon**`Vendor` modules are installed on the host platform, a Quik Learn automatically uses the matching Lon Xml file to create Lon devices.

Xml File Creation

You may need to make your own Lon Xml file if:

- The device includes one or more manufacturer-defined network variables, otherwise, you can use a **DynamicDevice** to represent the device, then perform a `Learn Nv` action.
- The device is newer or otherwise different from the device represented by the closest standard Lon Xml file, or a standard Lon Xml version for this device has yet to be released.

In general, using the `Lon Xml Tool` independently integrates Lonworks devices without depending on the framework manufacturer to issue updated **lon**`Vendor` modules.

For specific low-level details about Lon Xml, see to the Lon Markup Language section in the *Developer Guide*.

Lon Xml file source data

The device manufacturer provides the `.xif` and resource files for Lonworks devices. To create a Lon Xml file for any Lonworks device, you must have this `.xif` file. If the device contains manufacturer-defined datatypes, and you want this data for any Lon components, you must also get the related resource files from the manufacturer.

Resource files include the following types:

- `vendor`.ENU — Language resource file (in this case, English).
- `vendor`.FMT — Format selector file.
- `vendor`.FPT — Functional profile file.
- `vendor`.TYP —Type and enum file.

Before creating a Lon Xml file, place all related resource files in a single directory.

Creating a Lon Xml file

If no **lon**Vendor module contains the Lon Xml file you need, this procedure documents how to use the Lon Xml Tool to create a configuration file for the device.

Prerequisites: The source .xif file and (if necessary) other resource files, are available and ready to use. Workbench is open in a Supervisor station.

NOTE: Lon Xml supports default (init) values for a device ncis and cps, provided that the source .xif file is version 4.0 or later, and init data values other than all zeros exist.

Step 1 Click **Tools→Lon Xml Tool**.

The **Lon Xml Create** view opens.

Step 2 Browse to select the three source locations and define the complete path including the drive letter for each file or use the Sys Home (!) path.

- **Working Dir** identifies where to create the Lon Xml file (the **Directory Chooser** opens).
- **Xif File**defines the location of the manufacturer's .xif file for that device (the **File Chooser** opens).
- **Resource Dir** (if applicable) defines the directory that contains associated resource files (the **Directory Chooser** opens).

Step 3 To force the use of zero-based names for the unique name of each element in an arrayed nv, nci and cp for the Lon Xml file, click the **Zero Based Arrays** check box.

The tool creates names of arrayed elements by appending the array index to the array name. The default behavior (unchecked) uses one-based indices for arrayed nvs, ncis, and cps.

Step 4 To create the Lon Xml file, click **Compile**.

The tool parses the xif file for the device along with the resource files (if applicable), and creates a *.lnml file for the device in the working directory.

If you are using resource files, and when compiling, an exact match is not found, a popup **Engineering Unit Select** window opens before the tool creates the Lon Xml files. This window lets you review unit strings in the language resource file that may need association with engineering units.

In this window, the **Scope/Index** property shows the mapping within the language resource file to the unit string, and the **Original** property shows the found string. The original string is close to an approved unit in the list (possibly the best guess).

Step 5 Pick a new unit from the approved **New Text** list and click **OK**.

If the tool finds other unit strings that need specifying, the window remains open.

Step 6 Repeat selections until all unit strings are associated.

The tool creates the Lon Xml file for the device along with a datatypes Lon Xml file under the datatypes subfolder of the working directory.

NOTE: In a few cases, selection of a unit based on the Original text string may be difficult (does not suggest a normal engineering unit). In this case, leave the **New Text** property blank and click **OK** to go to the next unit. No units will be applied to elements using this string.

If you are creating Lon Xml lnml files for multiple devices by the same vendor, after doing this process for the first device, you might not be prompted again with an **Engineering Unit Select** popup, unless another device (xif) points to a new unit string.

The tool names each Lon Xml file based on the name of the .xif file, and gives its datatype Lon Xml file(s) names derived from the resource files.

IMPORTANT: If you copy the directories that contain Lon Xml files, such as to make a backup, make sure you copy the subdirectories as well as the primary directory and preserve the directory names.

Differential temps in Lon Xml files

In some cases, Lon components (nvs, ncis, cps) in Lonworks devices have temperature values that are intended as differential (delta, or offset) temperatures. This can be an issue when the system performs unit conversions, as absolute values use a conversion offset, and differential values do not.

Engineering unit support distinguishes between absolute temperatures (for example, Celsius and Fahrenheit) and differential temperatures (for example, degrees Celsius and degrees Fahrenheit). The Lon driver modules support special SNVT types for differential temperatures (for example, SNVT_temp_diff_p). Some controllers were designed before these units were introduced. As a consequence, these devices may have certain nvs defined as absolute temperatures, when in fact they are intended to be differential temperatures.

For unit conversion purposes, if you know which variables and properties (nvi, nvo, nci) in a Lonworks device should be treated as a differential temperature, you can edit its corresponding device Lon Xml file in a text editor. Using this edited Lon Xml file in Workbench to define Lonworks devices ensures that unit conversions are performed correctly.

Do this for any affected nv or nci by adding `+diff` to the `snvtType` attribute, as shown:

- from: `<snvtType v=TempP />`
- to: `<snvtType v=TempP+diff />`

As an alternative to editing device Lon Xml files, you can change the facets/units in the associated Lon proxy points under a device.

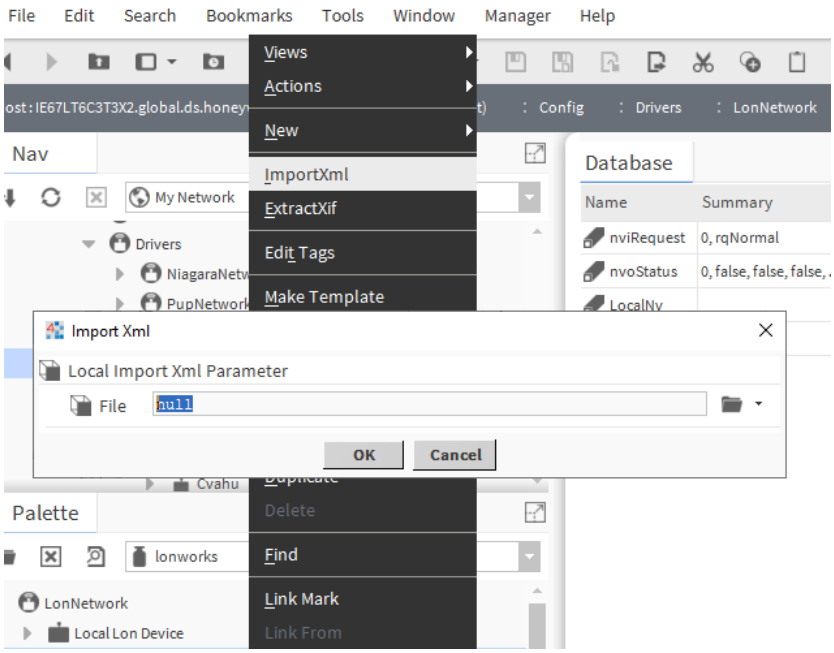
Configuring a device offline by importing a Lon Xml file

Lon Xml files configure Lon devices. When online, the system automatically learns (finds) each Lon Xml (extension `.lnml`) file to associate with individual devices. If you are configuring devices offline, this procedure explains how to import a Lon Xml , and associate it with a device. This same functionality is automatically provided when you Add discovered Lon devices, providing that a Lon Xml file was found for the device.

Prerequisites: You are working offline.

Step 1 Right-click a **DynamicDevice** in the Nav tree, and click `ImportXml`.

The Import Xml window opens.



Step 2 To change the location of the Xml File, click the folder icon to the right of the Xml File property and located the `.lnml` file.

Step 3 To create an extra folder organization of a device's Lon components grouped by LonMark objects, change **`Use Lon Objects`** property from `false` to `true` and click **OK**.

Workbench imports the specified `.lnml` file, and builds the various child Lon components (nvs, ncis, cps) under the device.

NOTE: If needed, you can build you own Lon Xml files for devices, using the **Lon Xml Tool** in Workbench. After building the file, you can specify it when adding Lon devices, or for devices already added, use the command to rebuild its Lon components.

Chapter 5 Data management

Topics covered in this chapter

- ♦ Unit conversion
- ♦ Discovering proxy points
- ♦ Creating proxy points
- ♦ Binding proxy points
- ♦ Linking and binding using a wire sheet
- ♦ Configuring point facets
- ♦ Polling rules
- ♦ Polled network variables
- ♦ Facet support for null output
- ♦ Building a utility command
- ♦ Viewing Lon components (variables)
- ♦ Directly editing Lon data
- ♦ Uploading data to the station database
- ♦ Downloading data to a device
- ♦ Px view usage of Lon data

This chapter includes messaging, collecting data from proxy points, and polling.

Unit conversion

Network variables use SNVTs (Standard Network Value Types, pronounced sniv-its) and SCPTs (Standard Configuration Property Types, pronounced skip-its) to define the units, scaling and structure of the data contained within the network variable. SNVTs and SCPTs use International System Units (SI), such as degree Celsius °C for temperature, l/s (liters per second) for volumetric flow, and so on. These are the native units in which numeric values are exchanged among devices.

Countries that use English units, such as degrees Fahrenheit °F for temperature and cfm (cubic feet per minute) for volumetric flow, must convert from metric to English values. It is important to understand the available unit conversion options.

Display unit conversions

In Workbench running on a PC, you may globally set unit conversion to automatically display Out values in English units (where the default is no conversion for display). Do this from the menu selection **Tools→Options→General**.

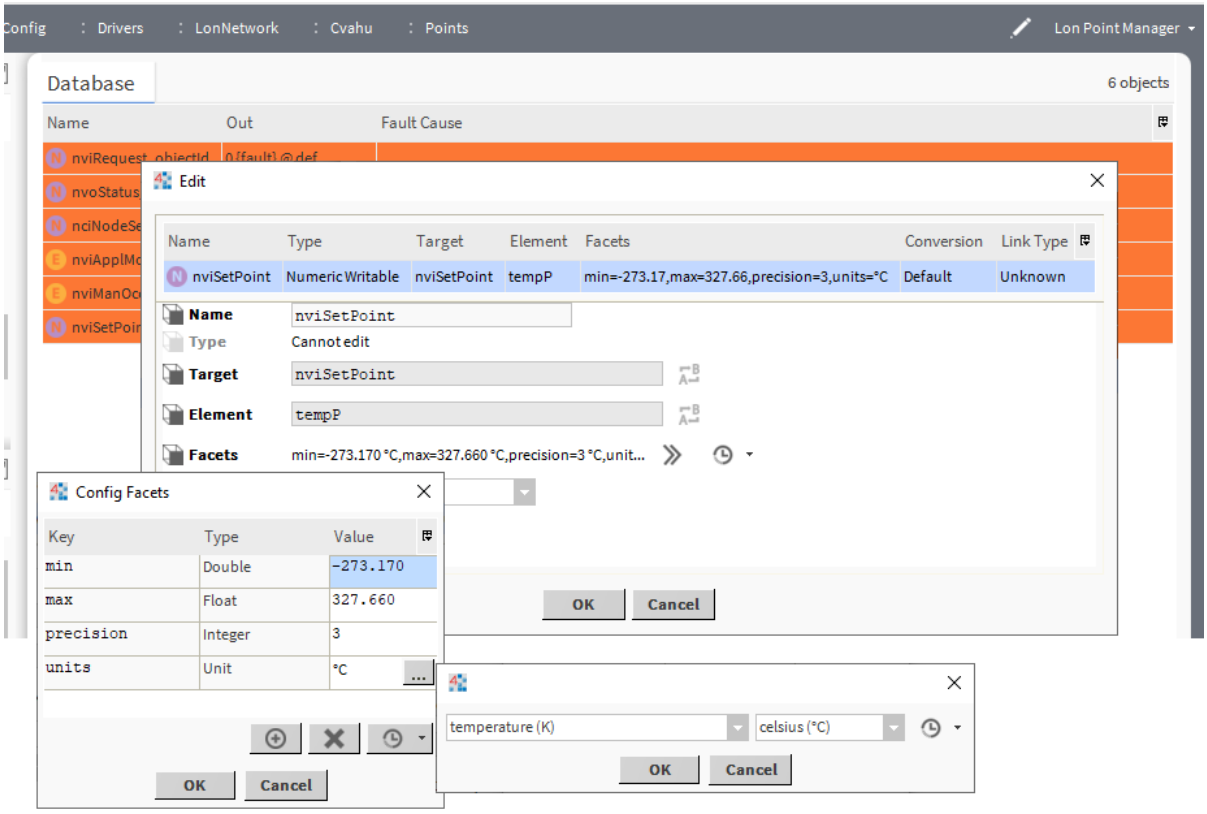
This setting is local to your Workbench display usage only If you provide other users browser access to Lon data (or any data using SI units), and need the same automatic display conversion, you must specify English unit conversion in each station user account (Facets, Unit Conversion).

This does not take the place of adjusting Facets for Lon proxy points.

Facet conversion for Lon proxy points

When you create Lon proxy points under a Lon device, by default each parent point's facets are the same as the device facets in its Lon ProxyExt (SI units). This is true regardless of your Workbench option settings for unit conversion display. For example, if you add Lon proxy points to a device's nci nciTempSetPts, for each data element (using SNVT_temp_setpt), each of the corresponding Lon proxy points defaults to facets of °C.

Figure 4 Default point facets reflect device facets



In U.S. installations, you edit each point's facets to English equivalents. Otherwise, without any further change (with your Workbench option set to English unit conversion), point Out values display in English units (such °F) in the **Lon Point Manager**. However, the actual (numeric) **Out** values remain in SI units (such as °C). If you link point outputs to other components, and/or add history extensions to these proxy points, the SI unit values are used.

Therefore, in a U.S. installation you should set the facets of Lon proxy points to the needed English equivalents, to avoid any control issues or other confusion. To do this, gang edit multiple (related) proxy points from the **Lon Point Manager**, or even in the initial **Add** window when you first create the proxy points.

Differential temperature

In cases where you add proxy points for nvs or ncis known to have a temperature differential (delta) application, yet the point is using absolute temperature facets/units (for example, Fahrenheit), change the proxy point's facets/units to the differential type, that is to degrees Fahrenheit. This prevents issues with unit conversions performed by the system. This also modifies the proxy units, device units, and the element qualifiers in the LonPrimitives in the associated Lon component.

You can also edit the Lon Xml (`.lnml`) file for a device, if needed, to accomplish the same conversion.

Discovering proxy points

In the **Lon Point Manager**, all available point data are already discovered when the device was initially learned. If you added a **DynamicDevice** offline, and then referenced a specific Lon Xml file, the available point data are then known to the driver.

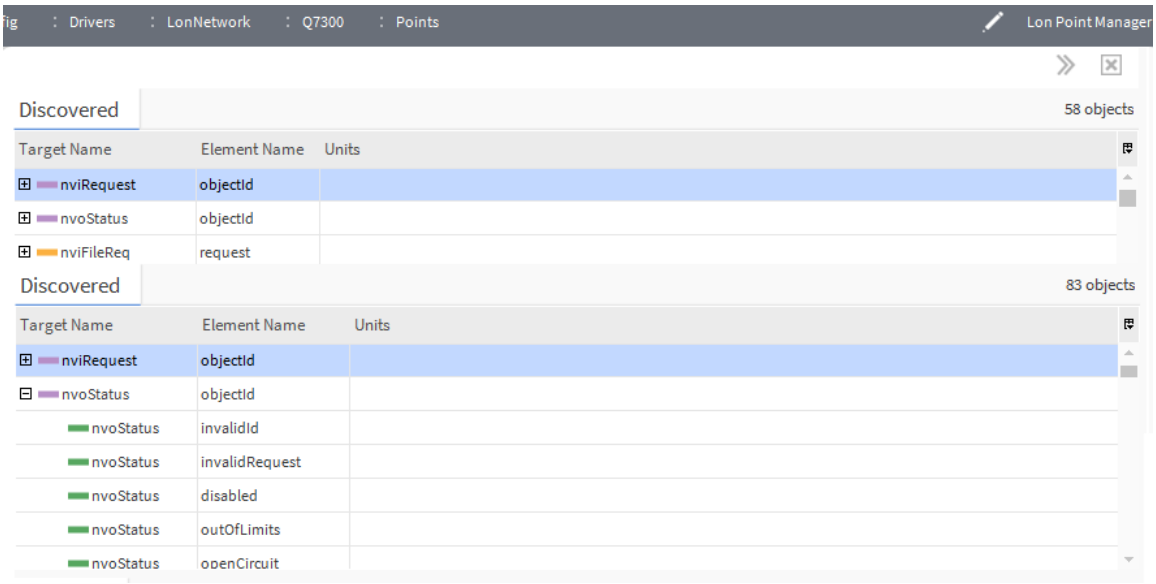
Prerequisites: You already discovered your devices.

This discovery procedure is basically the same for the local Lon device and dynamic devices.

Step 1 Double-click the Lon network followed by double-clicking the points icon ( ) under the Ext column for the device in question.

The **Lon Point Manager** opens.

Step 2 Click the **Discover** button or toggle learn mode on by clicking the Learn Mode tool ( ) on the toolbar.



Under Target Name the driver lists all points by the names of the Lon components that are associated with the device. For example: nvEmerg and nvoUnitStatus. By default, the driver displays nvs and ncis variables as structured SNVTs (Standard Network Variable Types). That is why you only see the first data element in the structure with a + to the left of the nv or nci name.

Step 3 To expand and see all data items in the SNVT structure, click the **+**.

Each item is a potential candidate for a Lon proxy point.

Creating proxy points

As with device objects in other drivers, each Lon device has a **Points** extension that serves as the container for proxy points. The default view for Lon points is the **Lon Point Manager**. You use it to add Lon proxy points under any Lon device.

The **Lon Point Manager** works differently than other driver **Point Manager** views. When toggled out of learn mode (**Database** pane only), the **Add** button is unavailable for manually adding proxy points.

The set of network variables (nvs, ncis) associated with each device type determine the Lon proxy points that are available for the device. Once you learn the device, this information is known to the system, and available in learn mode. If you add proxy points offline, this information is also known once you associate a Lon Xml File (extension `.lnml`) with the parent Lon device. No point discovery job is available for specific data items in any Lon device.

Once the station database contains the information for a specific device, you can add input and output network variables (proxy points) to the device.

This procedure is basically the same for the local Lon device and dynamic devices. Where the local Lon device proxy points differ has to do with how you can use them. Depending on the item selected, the default point type is either writable or read-only, as follows:

- nvi (read-only, that is NumericPoint, EnumPoint, BooleanPoint, or StringPoint)

- nvo (writable, that is NumericWritable, EnumWritable, BooleanWritable, or StringWritable)
- nci (read-only, that is NumericPoint, EnumPoint, BooleanPoint, or StringPoint)

  NOTE: Point type selections for local nvis and ncis are limited to read-only types (meant to be written externally, that is, outside the station). In the case of local nvos, only writable point types have any practical application, as only the station is capable of writing to its own nvos. This is the reverse of the default point type selections when creating other Lon proxy points.

Step 1 Double-click the Lon network node in the Nav tree.

The **Lon Device Manager** view opens.

Step 2 In the row of the **Database** pane representing the device, under the `Exts` column, double-click the points icon ( ).

The **Lon Point Manager** view opens.

Step 3 To make sure the view is in split-pane (learn mode), click the **Discover** button.

The driver lists device network variables (nvis, nvos, and ncis) in the top **Discovered** pane. Each row discovery job runs and the table represents one Lon proxy point candidate with each network variable (nv) occupying one row.

A plus (+) to the left of a variable identifies a structured SNVT (Standard Network Variable Type) with multiple data properties. By default, the first data property in the SNVT structure appears on top.

Step 4 Click the plus (+) to view the additional data properties.

Step 5 Click to select the data items you wish to proxy.

For an nv implemented as a structure, you may need one or more secondary data properties instead of (or in addition to) the first top data property.

Step 6 To add the selected items in the station as proxy points, choose one of these methods:

- Drag the selected points from the **Discovered** pane to **Database** pane (brings up an **Add** window).
- Double-click an item in the **Discovered** pane (also brings up an **Add** window).
- Click the Add tool ( ) in the toolbar.
- Click to select in **Discovered** point , then press the letter **a** (quick add).

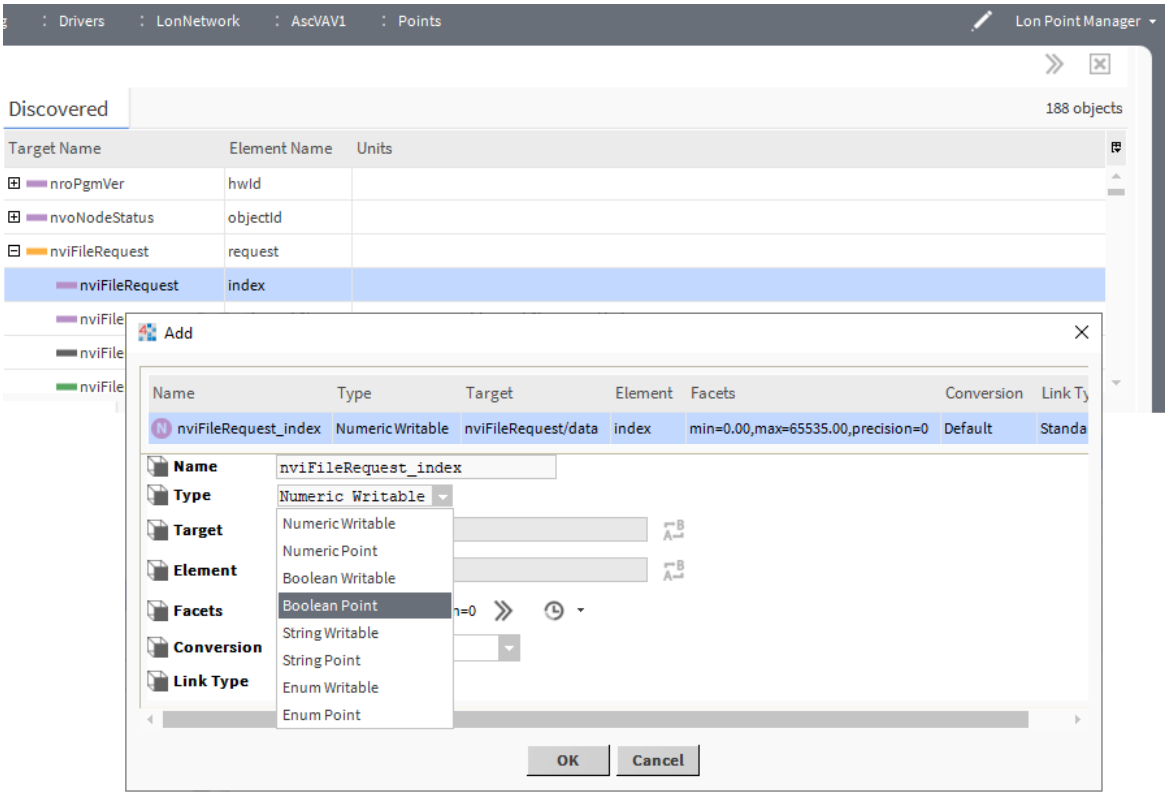
If you either dragged or double-clicked the selected points, the **Add** window opens. The quick add option (select the row and click **a**) bypasses the **Add** window. This option is for points that do not require further configuration.

Step 7 If the **Add** window opens, edit the point configuration, and click **OK**.

In some cases, you may need to edit a point's facets or even conversion type. When you add a Lon proxy point, the driver pre-selects a default point type. Often, this data type (category) is either numeric or enum, and less often, Boolean or string. A Lon proxy point supports default conversion between data types. This means that when creating a proxy point, you can select another data type, if the selection would be useful for your station's application. The driver supports changing point type (category) for local Lon device points as well as for dynamic device points.

For example, if when adding a proxy point for an nvo implemented with `SNVT_Switch`, you are interested only in the state element as either false (off) or true (on), and you do not anticipate a null value, you can select a `Boolean Point` instead of the pre-selected `Enum Point`.

Figure 5 Changing point type when adding Lon proxy point



This would allow you to link the Out of the proxy Boolean Point directly to a Logic-type kitControl object without using an intermediate conversion object (StatusEnumToStatusBoolean). The updated proxy point would convert enum ordinals 0 to false, and 1 to true.

The table below shows how default Lon proxy point conversions are handled between data types.

Table 1 LonData element type to proxy point data type, default conversions

LonData element type	Proxy point type	Conversion method	Setup notes
Numeric	Boolean	False = 0, True = 1value <=0 = False, >0 = True	—
	String	Converts between string representation and numeric values	—
	Enum	Numeric is enum ordinal	Add facet EnumRange
Boolean	String	True = true, False = false	—
	Numeric	False = 0, True = 1value <=0 = False, >0 = True	—
Enum	Numeric	Numeric is enum ordinal	—
	Boolean	True to ordinal 1, false to ordinal 0.	—
	String	Read/Write enum tag	—
String	Numeric	Converts between string representation and numeric values	—
	Enum	String is enum tag	Add facet EnumRange

LonData element type	Proxy point type	Conversion method	Setup notes
	Boolean	True = true, False = false	—

Also, when you add a proxy point under a Lon device for an nvi, nci, or cp, the **Add** window defaults to a writable type, for example NumericWritable or EnumWritable. Before creating the proxy point, you may wish to change it to a read-only point type (if you wish to monitor only).

Step 8 When you have Lon proxy point(s) configured properly for your usage, click **OK**.

The system adds the proxy points the station, and displays them in the **Database** pane. If you are online, points poll for current values. If you are working offline, all proxy points appear down (yellow).

Binding proxy points

By default, when you add nvs (nvi or nvo) proxy points, the station polls each point to update its value. For may proxy points, particularly network variable outputs (nvos), polling is less efficient than binding them from the source device (sending) to the target local Lon device (receiving). Then, the station updates them using the native Lonworks nv update mechanisms, reducing the polling queue. Binding output to input proxy points improves communication efficiency.

The station does not poll ncis and cps (SCPTs) proxy points. It updates these values when you upload from the Lon device or issue a `Force Read` action for the corresponding Lon components.

You bind nvs used for Lon proxy points unless there is a specific reason not to do so. One possible scenario may include devices with nvos that send updates too frequently, and there is no way (in such devices) to adjust the minimum write time upwards. In this example, even after a global bind, you can specifically pick nvs (in srcDevices) to change to `Poll Only`.

Step 1 In the Nav side bar, right-click the Lon network node and click **Views→Lon Link Manager**.

The **Lon Link Manager** opens. Each newly-added proxy point appears as a row in the **NetworkVariableLinks** tab, with a **`linkStatus`** of `NewLink`. By default, each point uses a **`linkType`** of `Standard`.

Step 2 If needed, click on the host column header to sort the table by devices.

This groups nvo sources together by Lon device name.

Step 3 Do one of the following:

- To bind all proxy points, click the **Bind** button.
- To bind a subset of proxy points, select the points and click the **Selective Bind** button.

A progress bar displays at the top of the manager. When the bind job finishes, the **`linkStatus`** of entries for proxy points or selected points should show as `Bound`.

Step 4 To verify the bind, right-click the Lon network node in the Nav tree, click **Views→Lon Utilities Manager**, configure **`Command`**: Reports and **`SubCommand`**: Verify, and click **Execute**.

Step 5 Check the report to ensure that each device's nv bind configuration matches the network management configuration stored in the station.

Linking and binding using a wire sheet

Although not a manager view, whenever the station is configured as the Lonworks network manager (typical), the wire sheet view of the Lon network (and any child **LonDeviceFolder** containing Lon devices) is an important view. Here, bindings between Lon devices appear as graphical links (wires), and you can add or delete bindings between devices.

Using the wire sheet is not a strict requirement. You can perform the same operations using right-click commands (on Lon devices) in the Nav tree, such as Link Mark and Link from <LonDeviceName>. However, the wire sheet graphically shows you existing bindings as wires or nubs.

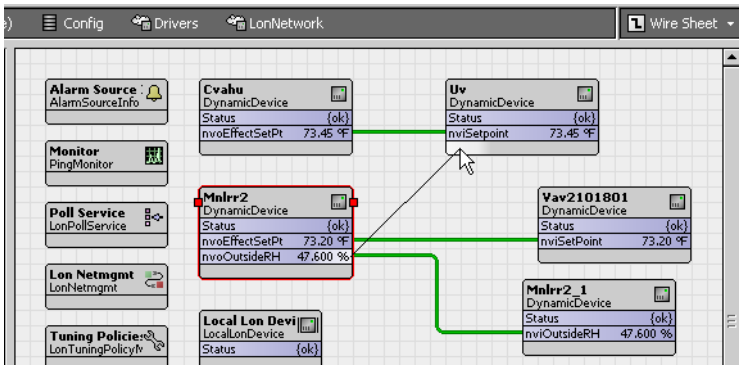
CAUTION: Lon proxy points provide station access to nvs, ncis, and cps for usage in control, display, and various alarm and history schemes (via point extensions). However, for the purpose of sharing nv data among devices, do not link the proxy points in one device directly to the proxy points in another device on the same network. This is a linking misapplication! Instead, link the devices as described in this procedure. Linking the devices also links the network variables associated with each device.

Step 1 Right-click the Lon network node in the Nav tree and click **Views→Wire Sheet**

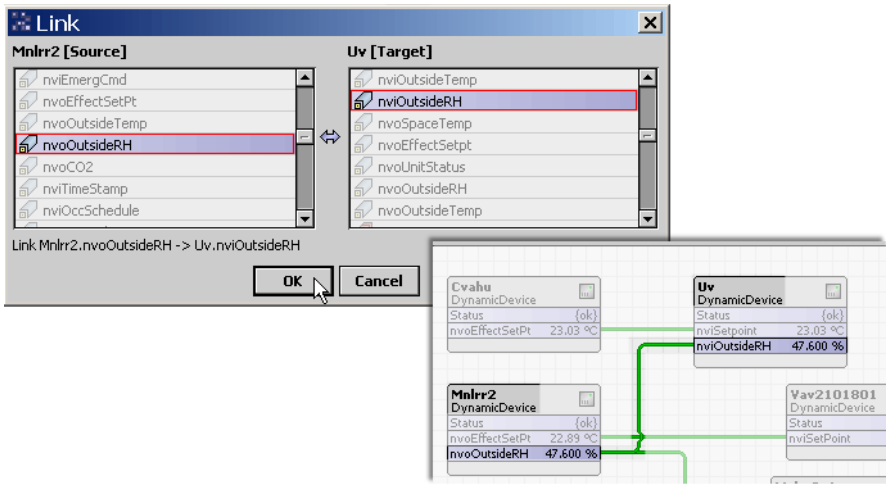
The wire sheet grid opens.

Step 2 To add a LonPoint to the configuration, open the **`kitLon`** palette and drag a LonPoint to the wire sheet, and select the type of SNVT.

Step 3 Link the out from a control point (elsewhere in the station) to the dynamic slot of the LonPoint, then link the nvo (network variable out) slot of the LonPoint to the nvi (network variable in) on one or more Lon device(s).



The screen capture shows a link being pulled from an nvo of one device to another device.



However you specify a link between Lon devices, the popup Link editor provides nv (for example, SNVT) verification to allow only legitimate links.

This link is still a station-dependent link. In other words, until you perform a Bind or Selective Bind operation from the **Lon Link Manager**, the station is required for data exchange.

Step 4 Right-click the Lon network node in the Nav tree and click **Views→Lon Link Manager.**

Step 5 Select the link you just created using the wire sheet and click the **Bind** or **Selective Bind** buttons.

NOTE: In the **Lon Link Manager**, a link/binding between two Lon devices appears as an entry row without LocalDev as the value of either srcDevice or targetDevice (whereas, any entry row with a LocalDev value represents a Lon proxy point in the station).

Configuring point facets

You can configure point facets when you create the point or edit them later.

Step 1 Double-click the **Points** folder in the Nav tree.

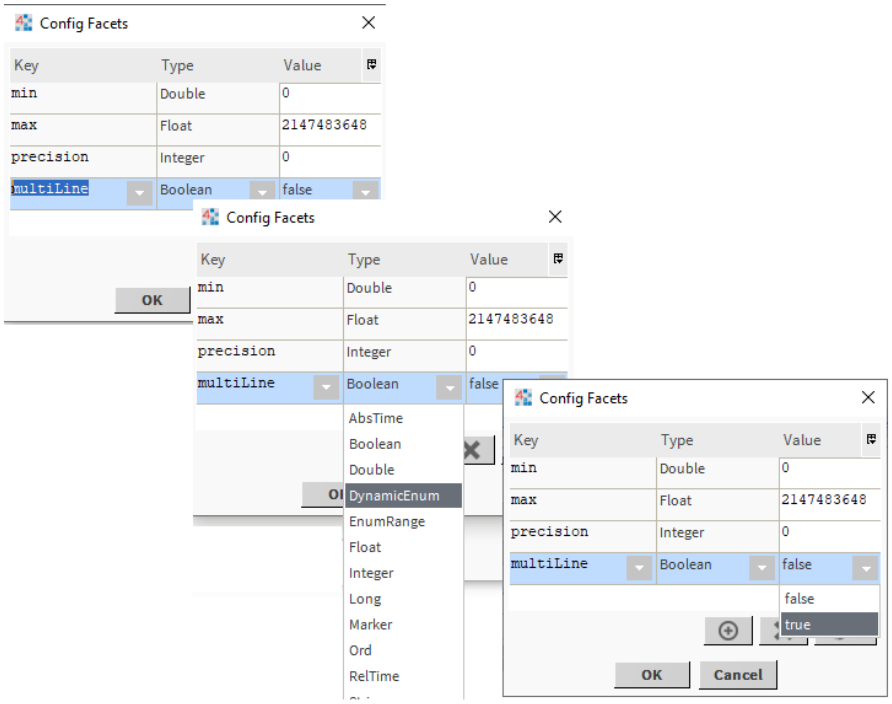
The **Lon Point Manager** view opens. This view lists the discovered points and those in the station database.

Step 2 Select the point whose facets you wish to edit and click the **Edit** button.

The **Edit** view opens.

Step 3 Click the chevron to the right of the **`Facets`** property.

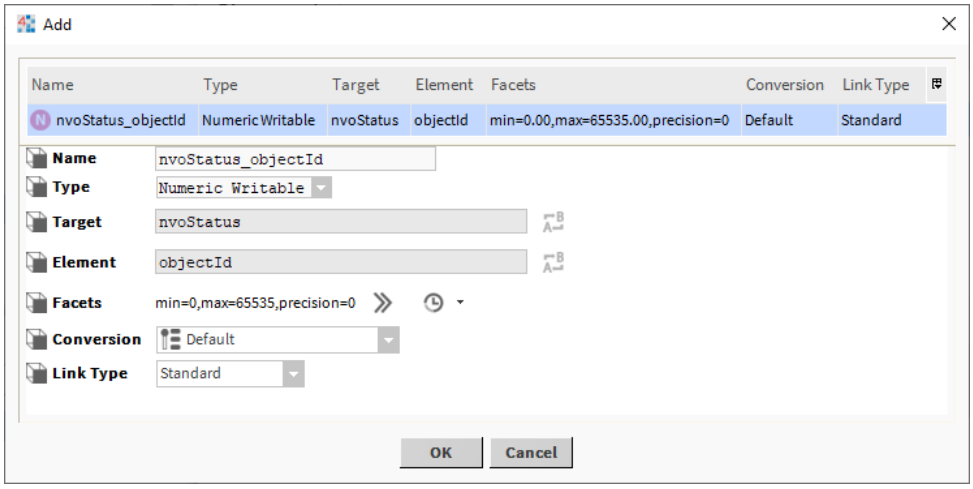
The **Config Facets** window opens.



The screen captures illustrate configuring the `nanIsNull` facet in a legacy system so that the proxy point supports null data.

Step 4 To continue, click **OK**.

The changes display to the right of the **`Facets`** property in the **Edit** window.



Polling rules

General rules apply for polling Network Variable (nvi and nvo) values.

1. A Network Variable (nv) is subscribed if it is being viewed or has a proxy point which is subscribed. A proxy point that is subscribed if it is viewed, if it is linked to control logic if it has an alarm or history extension.
2. When a nv is first subscribed, the driver reads its value (if the device is in a valid state). The poll thread handles further reads. The Poll Service must be enabled for polling to function normally.
3. Adding a proxy point creates a local connection to the associated nv. each proxy point for a Network Variable Output (nvo) or a writable proxy point for a Network Variable Input (nvi), appears as one entry (row) in the **Lon Link Manager**.
4. The **`Bound To Local`** property on **nv Props** container is set when an output nv with a proxy is bound. An output nv is not polled when it is bound locally. Instead, nv updates are received from the device.
5. The **`Poll Enable`** property is set to `true` if there are unbound links and the nv is not bound locally. To prevent polling on an nv, set its **`Poll Enable`** property to `false`. The driver still performs the initial read when the nv is first subscribed, but subsequent polling by the poll service does not occur.

This table provides polling rules for nvs with specific connection scenarios

Table 2 Polling rules

Type of nv, connection	Update rules
nvi with writable proxy	Poll when nv or proxy point subscribes. If proxy value changes update nv.
nvi with read-only proxy	Poll when nv or proxy point subscribes.
nvo with writable proxy	<Not allowed>
nvo with read-only proxy (unbound)	Poll when nv or proxy point subscribes.
nvo with read-only proxy (bound)	Receive nv updates (No polling).
polled nvo with read-only proxy bound	Poll when nv or proxy subscribed.
nvi linked	Poll when subscribed.

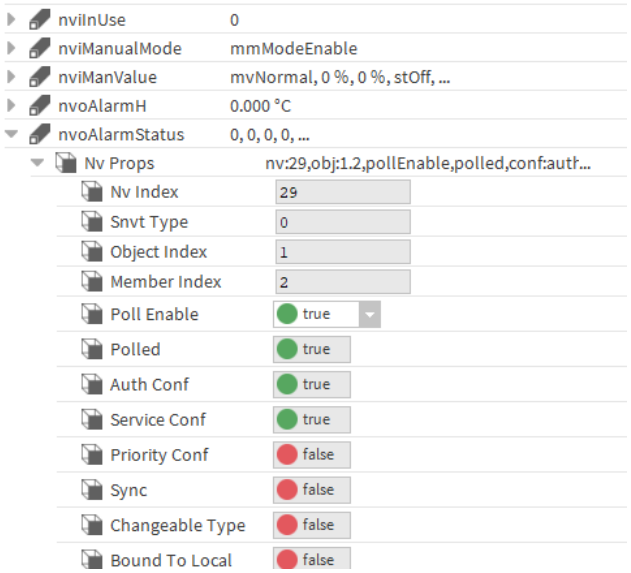
Type of nv, connection	Update rules
nvo linked unbound	Poll when subscribed. If nvo changes push value to linked nvi. Write nvi value to device.
nvo linked bound	Poll when subscribed.

NOTE: Input nvs (nvi) with read-only proxy points are never bound, and do not show up in the **Lon Link Manager**.

Polled network variables

Some Lonworks devices include one or more nvos that are polled types, meaning that the nvo does not send nv updates upon a value changes. Such an nvo must be polled. Any such nv represents a data item that does not frequently change.

Figure 6 Lon component example of polled only nvo



You access these properties by expanding the Lon network, followed by expanding a device container, a Lon point, the **Nvo** container,**Nv Props** container.

In the Lon device object, any such nvo Lon component appears with its read-only **`Polled`** property set to `true` (as found under its **Nv Props** container).

If you create a proxy point for such an nvo and bind to it in the **Lon Link Manager**, it still polls to receive value updates. Also, when creating a link (bind) between two Lon devices, the Link Editor only allows polled outputs to be linked to polled inputs.

Facet support for null output

Rather than report `nan` (not a number) when nodes receive invalid values from numeric and enumerated Lon data elements, the driver reports `null`.

If the point's out slot is linked to the input of a math component (an Average object, for example) or to the priority array input of a writable control point, a value of `nan` causes the calculation to return an invalid result. The `null` value, however, causes the calculation to ignore the erroneous input. In the linked Average example, the result reflects the average of the other non-null input values, instead of displaying `nan`. The linked writable point drops down to the next-highest priority arrayed input (or fallback) for control, instead of displaying `nan`.

For numeric points, the driver provides these facets:

- **Key**: `nanIsNull`, **Type**: `Boolean` - if Value = `true`, invalid values cause the driver to return a null value.
- **Key**: `isNull`, **Type**: `Double` - if the specified Value is present, the driver returns a null value.

For enum points: **Key**: `isNull` - **Type**: `Integer` - if the specified value is present, the driver returns a null value.

Building a utility command

Using a utility command and sub-command you can identify devices from a service pin message, determine the working domain on an existing managed network, and verify that network variables and links are bound correctly.

Step 1 Right-click the Lon network node in the Nav tree and click **Views→Lon Utilities Manager**.

The **Lon Utilities Manager** view opens.

Step 2 Select the device in question.

The list includes both devices in the station database as well as discovered nodes.

Step 3 Select a main command.

Common main commands include `Identify` and `Reports`.

Step 4 Select a sub-command.

The list varies according to the selected command.

Step 5 After you build the command, click **Execute**.

The results display in the central pane of the view

Viewing Lon components (variables)

Under any Lon device, all of its available network variables (nvis and nvos, ncis, and cps (SCPTs) are represented in the station by Lon components, which are viewable on the device's Property sSheet. In turn, each Lon component contains LonData plus one or more container slots, including a Props structure needed by the station's network management to access or create links to variables.

Prerequisites: Variables (Lon components) are available for viewing.

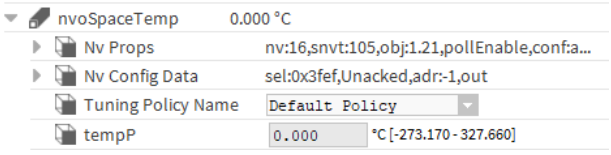
Step 1 Right-click any device and click **Views→Property Sheet**.

The device **Property Sheet** opens.

The Lon Xml file (extension `.lnml`) associated with the device determines the device's collection of Lon components. A Quik Learn or discovery job links the component with an existing Lon Xml file. You can manually specify the Lon Xml file in the **Add** or **Edit** window for a Lon device, then use the right-click device command **Import Xml** to associate the file with a device.

If a dynamic device does not have a referenced Lon Xml file the system can identify its network variables using a `Learn Nv` action when the device is online with the station.

Step 2 Expand the individual variables.



The screen capture shows a space temperature variable.

When learning or adding a Lon device, you can optionally specify for the device's Lon components to be grouped under LonObject containers. Each LonObject represents a LonMark object, and its child Lon components are the set of nvs, ncis, and cps that comprise that object.

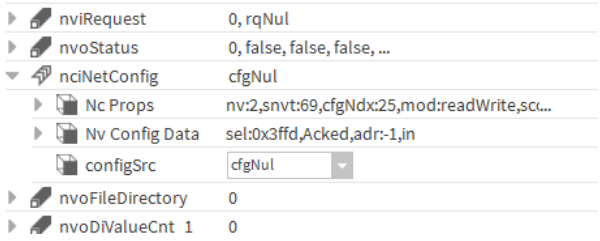
Step 3 Expand a device in the Nav tree.

LonObjects are visible in the Nav tree when you expand the Lon device. This extra level of component hierarchy may be useful in devices that have a large number of LonMark objects.

Directly editing Lon data

With an online Lon device, you can access and use Lon data values directly without making proxy points. This is because data are communicated among devices using variables, which appear on your device Property Sheets.

Step 1 Expand the Lon device in the Nav tree and expand a nci or cp variable.



Step 2 Review the current values of the variables, and, if necessary, make a value change.

Step 3 To continue, click **Save**.

The driver writes the change to the device.

You can change nvi Lon data values in the same way, however, nvi values are overwritten again. You cannot directly change nvo Lon data values (these are read-only), but for any Lon components you can issue a `Force Read` action.

Without requiring Lon proxy points, you can also make Px view usage of Lon data.

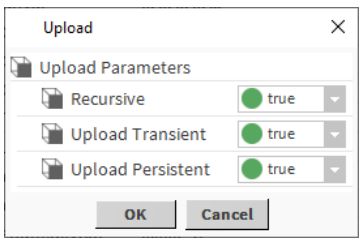
Uploading data to the station database

After using **Discover** and **Add** to create a new Lon device (and assigning the device a Lon Xml file), you can use an **Upload** action to populate current LonData in the device's Lon components (nvs, ncis, cps).

NOTE: If you are using Quik Learn, an **`Upload Config`** property is already included. This uploads all current LonData into the device's Lon components, making another upload unnecessary. An `Upload` action is also available at the **LonNetwork** level—with the same **Upload** window selections. This provides a global upload from all Lon devices.

Step 1 Right-click the device or the Lon network in the Nav tree, and click **Actions→Upload**.

The **Upload** Lon device window opens.



You leave window properties at defaults (`true`)—**`Recursive`** is always recommended.

Step 2 To execute the upload, click **OK**.

The driver uploads current LonData in the device to the Lon components (network variables) for the device stored in the station database.

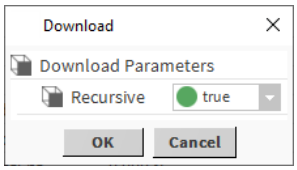
Downloading data to a device

This procedure documents how to restore nci and cp values to known good values, as previously saved in the station.

NOTE: A `Download` action is also available at the **LonNetwork** level—with the same **Download** window selection. This provides a global download to all Lon devices.

Step 1 Right-click the device or the Lon network in the Nav tree, and click **Actions→Download**.

The **Download** Lon device window opens.



You leave window properties at defaults (`true`)—**`Recursive`** is always recommended.

Step 2 To start the download, click **OK**.

Px view usage of Lon data

For simple monitoring from Px views, data bind Px widgets to Lon component ords to retrieve LonData values, without requiring proxy point creation. However, note the following about this LonData usage:

- Polling fetches Lon data values, whereas, if you create proxy points for items, you can use the Lon Link Manager to create Lon binds to the local device to receive values via nv updates. Overuse of polling may adversely affect Lon network traffic.
- Data logging or alarming (on value) requires a proxy point, where you add and configure history and/or alarm extensions.
- Using Lon data values in station control schemes requires proxy points.
- An nvi, nci, or cp represented with writable proxy points offers a Px view, user right-click command access to set and override values. However, if Px view access to these items is via Lon components (LonData), available actions are limited to Force Read (poll) and Force Write (resend value in station).

Chapter 6 Troubleshooting

Topics covered in this chapter

♦ Debugging messages
♦ Adjusting message buffers in a Neuron chip

This topic provides actions you can take to resolve error conditions.

Discovery is not working even though all my devices are physically connected and configured in the station.

You may need to configure the working domain before can communicate with your devices. If you are unable to discover nodes that are physically connected to the station, they may be configured on a different domain. This involves gaining access to a device by use of its Lonworks Service pin.

Discovery is unable to find the nodes in my network.

The nodes are likely configured on a different domain. If the network was previously managed and you wish to maintain the current network management configuration (addresses, bindings) using Quik Learn. You must change the working domain of the **LonNetwork** to match.

When commissioning and binding I am getting errors. The commission or bind reports that it failed.

Sometimes, certain Lon devices may process Lon messages more slowly than normal, resulting in errors when you do commissioning or binding operations. To confirm such problems use the **Lon Utilities Manager** (afterwards) and run a verify report. This report lists inconsistencies between the bindings on the devices and the bindings that the Lon network management determines should exist.

In an example with a Lon network containing such slow responding devices, operation may improve with Lon Comm properties adjusted upwards to:

- **`Repeat Timer`**: mSec128
- **`Receive Timer`**: mSec1024
- **`Transmit Timer`**: mSec128
- **`Retries`**: 4

In this case, it may also be necessary to adjust link descriptors under the **`Lon Netmgmt`** container from default values.

No matter what I try, my LonIpNetwork remains in fault.

Your **LonIpNetwork** may not be licensed.

Debugging messages

Link debugging shows the raw byte data for messages sent from and to the Neuron chip in the device.

Step 1 Open the **`lonworks`** palette and drag a **LinkFilter** component to the Lon network node in the Nav tree.

Step 2 Right-click the **LinkFilter** node in the Nav tree, click **Views→Property Sheet**, then set at least one of the three enable properties to `true`.

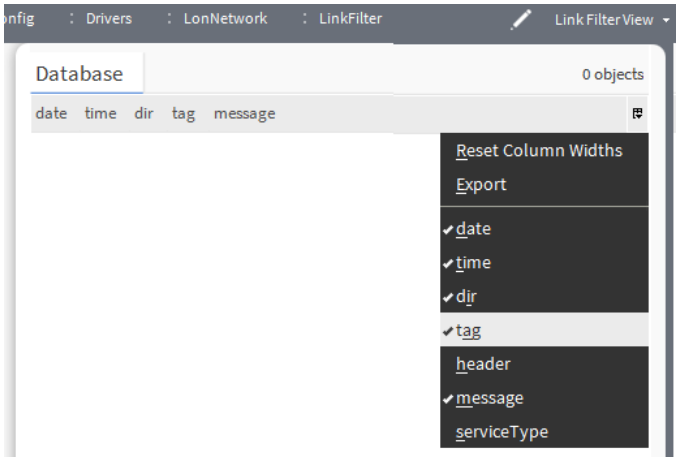
Step 3 Expand the **Lon Comm Config** container and change **Link Debug** from `false` to `true`.

Step 4 Right-click station and click **Spy→logSetup**.

Step 5 Enable `Trace` on the port.

Step 6 Double-click the **LinkFilter** node in the Nav tree.

The **Link Filter View** opens and populates with link events.



Step 7 To view details for any specific event, double-click it.

The **Application Buffer** window opens for the selected link event.

Refer to the *LonWorks Host Application Programmer's Guide* (Appendix C), for message header meaning (first 16 bytes). Refer also to the *Neuron C Data Book* (Appendixes A and B), for definition of LonTalk messages.

Adjusting message buffers in a Neuron chip

This procedure configures message buffer counts and sizes in the Neuron chip of a Local Lon Device.

Prerequisites: The kitLon palette is open.

CAUTION: Do not change message buffer counts and sizes unless you fully understand the implications. An improper configuration can significantly degrade operation and/or result in loss of data.

Step 1 Drag or copy the BufferParams component to your **Local Lon Device** in the Nav tree.

Step 2 Double-click the BufferParams component and take a screen capture or otherwise record the default buffer properties.

Step 3 To verify the Neuron's current buffer configuration, right-click the Lon network node in the Nav tree, click **Views→Lon Utilities Manager**, configure **`Command`**: Data Structs, **`Subcommand`**: Read Only Structure, and click **Execute**.

Step 4 Double-click BufferParams and configure a different set of message buffers.

The component's **`Current Size`** property reflects the changes to buffer count or size. To be applicable, modifications cannot exceed the size shown in the **`Original Size`** property—that is, **`Current Size`** must be less than or equal to **`Original Size`**. For example, to increase a buffer size, you may need to decrease its count.

The system does not automatically write message buffer changes saved on the BufferParam's **Property Sheet** to the Neuron chip in the **Local Lon Device**.

Step 5 Before continuing, ensure that **`Current Size`** less than or equal to **`Original Size`**, and click **Save**.

Step 6 To write the message buffer changes to the Neuron chip, right-click the BufferParams node in the Nav tree and click **Actions→Update Buffers**.

If **`Current Size`** is greater than **`Original Size`**, the job throws an exception, and writes no changes to the Neuron chip.

Chapter 7 Components

Topics covered in this chapter

- lonworks-LonNetwork
- lonworks-DynamicDevice
- lonworks-LonRouter
- lonworks-LonDeviceFolder
- lonworks-LonObjectFolder
- lonworks-LonPointFolder
- lonworks-LonTuningPolicyMap
- lonworks-LinkFilter
- lonworks-LonTimeZone
- lonworks-LonWbService
- lonworks-LonAppDownloadJob
- lonworks-LonBindJob
- lonworks-LonChangeNvTypeJob
- lonworks-LonCommissioJob
- lonworks-LonCommissionRouterJob
- lonworks-LonDiscoverJob
- lonworks-LonLearnJob
- lonworks-LonLearnLinksJob
- lonworks-LonReplaceJob
- lonworks-LonXmlTool
- lonIp-LonIpNetwork
- tunnel-TunnelService
- lontunnel-LonTunnel
- kitLon-LonTime
- kitLon-LonTodEvent (Lon Tod Event)
- kitLon-LonEnumTodEvent (Lon Enum Tod Event)
- kitLon-LonPoint (Lon Point)
- kitLon-LonReplace (Lon Replace)
- kitLon-BufferParams (Buffer Params)
- lonworks-PseudoNV

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**

lonworks-LonNetwork

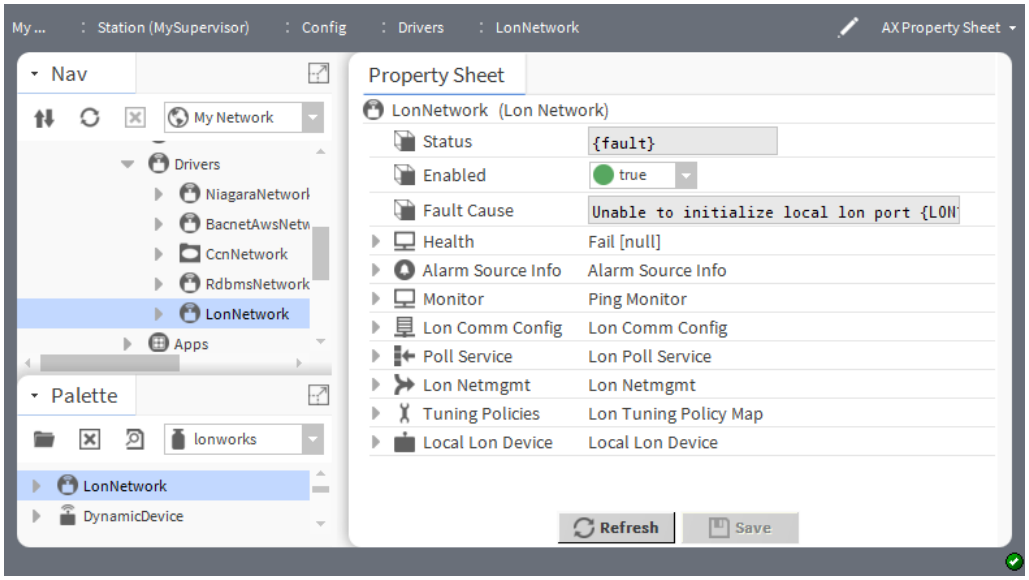
This component is the base container for all Lonworks (Lon) components in the station. As with other driver networks, it resides under the station's **Drivers** container.

If the host has multiple Lon ports, you need a separate **LonNetwork** for each physical port.

The default view of the **LonNetwork** is the **Lon Device Manager**.

In addition to being the network container for **LonDevice**s and their child data objects (Lon proxy points), the **LonNetwork** includes a **LocalLonDevice**, which configures the station's representation as a Lonworks device.

Figure 7 LonNetwork properties



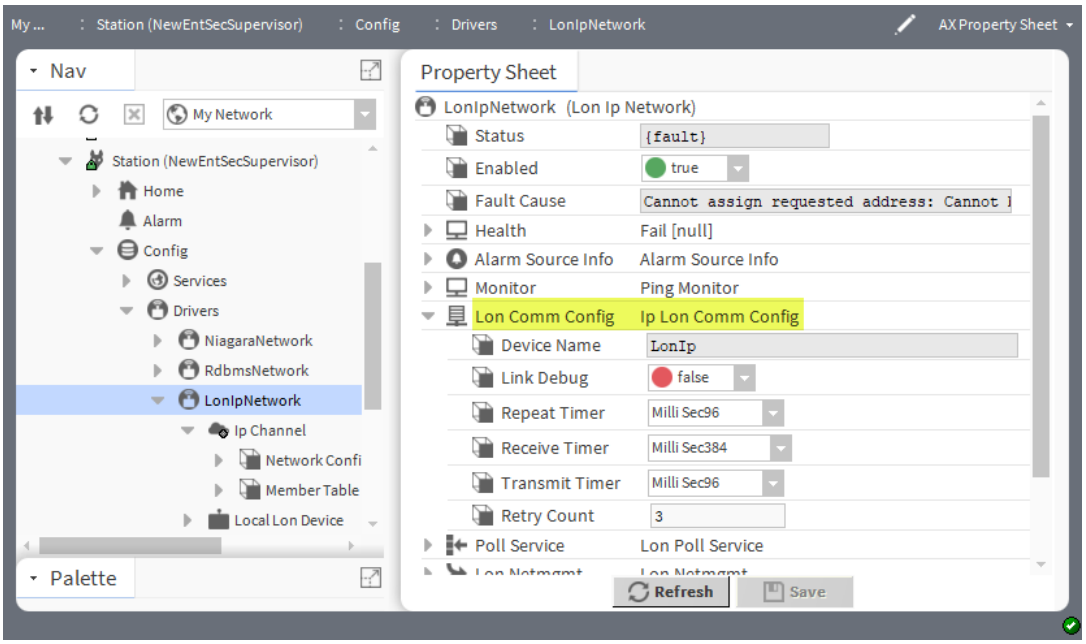
To access this view, expand **Config→Drivers**, right-click the **LonNetwork** component in the Nav tree, and click **Views→AX Property Sheet**.

Property	Value	Description
Status	read-only	Indicates the condition of the network, device or component at the last check. `{ok}` indicates that the component is licensed and polling successfully. `{down}` indicates that the last check was unsuccessful, perhaps because of an incorrect property, or possibly loss of network connection. `{disabled}` indicates that the **`Enable`** property is set to `false`. `{fault}` indicates another problem. Refer to **`Fault Cause`** for more information.
Enabled	`true` or `false`	Activates (`true`) and deactivates (`false`) the object (network, device, point, component, table, schedule, descriptor, etc.).
Fault Cause	read-only	Indicates the reason why a system object (network, device, component, extension, etc.) is not working properly (in fault). This property is empty unless a fault exists.
Health	read-only	Reports the status of the network, device or component. This advisory information, including a time stamp, can help you recognize and troubleshoot problems but it provides no direct management controls.
Alarm Source Info	additional properties	Contains a set of properties for configuring and routing alarms when this component is the alarm source.
Monitor	Ping Monitor	Configures a network's ping mechanism, which verifies network health. This includes verifying the health of all connected objects (typically, devices) by pinging each device at a repeated interval.

Property	Value	Description
Lon Comm Config	additional properties	Configures the communication stack for a single Lonworks connection including specifying the port's **Device Name** For specific properties, refer to a separate topic in this guide.
Poll Service	additional properties	Configures the frequency with which the driver polls points and devices. For information about polling properties, refer to the *Niagara Drivers Guide*
Tuning Policies	additional properties	Configures network rules for evaluating both write requests to writable proxy points as well as the acceptable freshness of read requests. For information about tuning policy properties, refer to the *Niagara Drivers Guide*
Lon Netmgmt	additional properties	Serves as the container for Lon network management functions provided by the station. For specific properties, refer to a separate topic in this guide.
Local Lon Device	additional properties	Configures the local interface to the Lonworks fieldbus. For specific properties, refer to a separate topic in this guide.

Lon Comm Config

Figure 8 Lon Comm Config properties

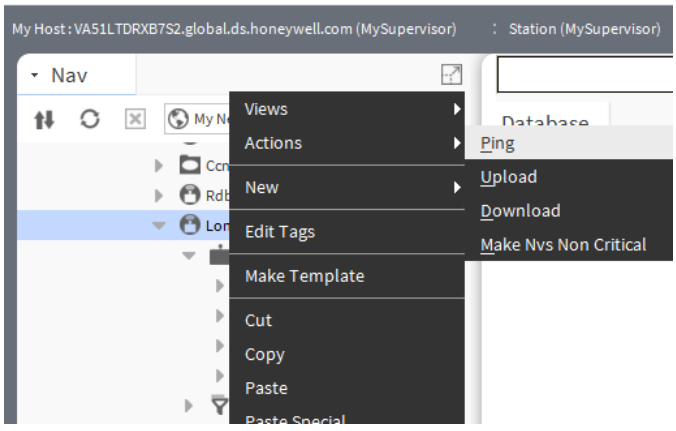


In addition to the standard polling properties documented in the *Niagara Drivers Guide*, these properties are unique to Lon Comm Config.

Property	Value	Description
Device Name	string	Identifies the host's `LONn` port used in a Lon network. For example, Device Name = `Mnlrr2`. For a **LonTunnel** copied from the palette, this defaults to `LON1`. Defice names must match the name string in the **Lon-Network Lon Comm Config** container.
Link Debug	`true` or `false` (default) [used once]	Controls the display of messages from and to the Neuron. `false` limits the message display. `true` shows the raw byte data for messages from and to the Neuron. To use this feature you must also enable link debug using the spy page in the station. Refer to the *Troubleshooting* chapter.
Repeat Timer	drop-down list of millisecond values (defaults to 384 milliseconds) [used once]	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.
Receive Timer	drop-down list of millisecond values (defaults to 96 milliseconds) [used once]	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.
Transmit Timer	drop-down list of millisecond values (defaults to 96 milliseconds)	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.
Retry Count	number (defaults to 3)	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.

Actions

Figure 9 Lon Network actions menu



To access these actions, right-click the **LonNetwork** and click **Actions**.

Action	Description
Ping	Attempts communication with the device. If successful, the device status reports `{ok}`. If this fails, the system sets device status to `{down}`
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.
Make Nvs Non Critical	Changes the network variable.

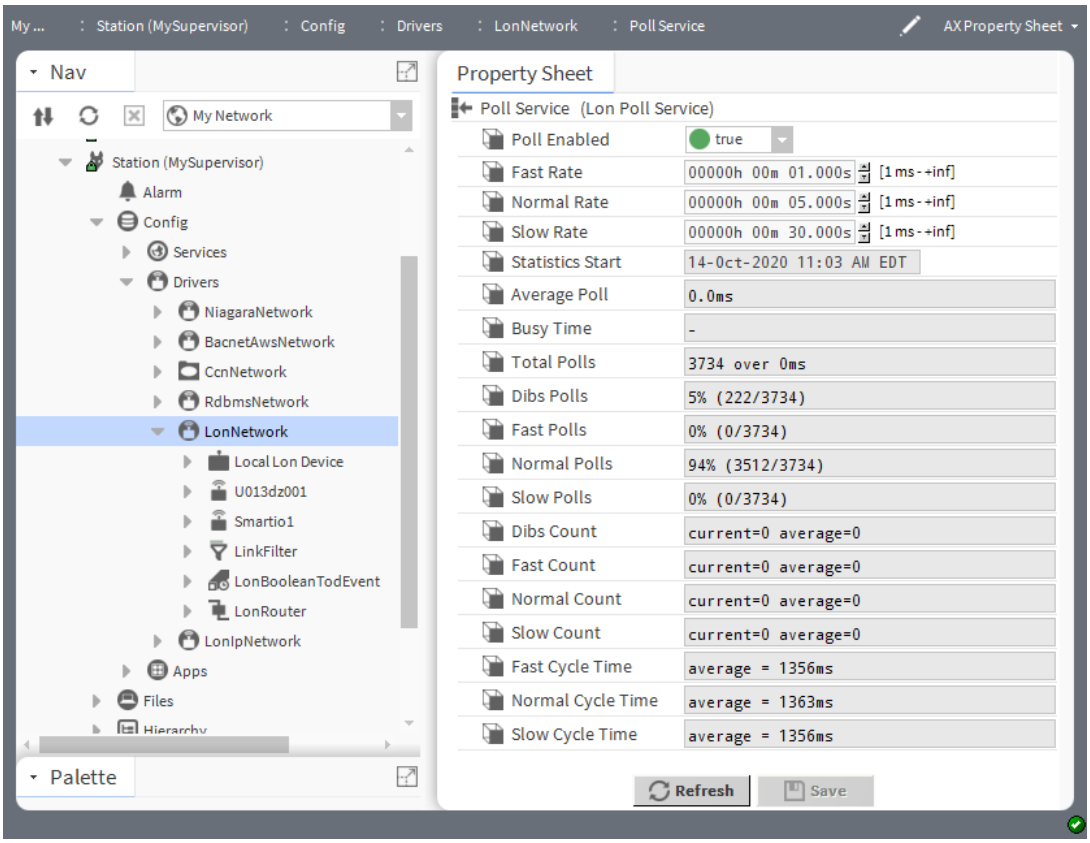
lonworks-LonPollService

This component controls the polling of Lon devices.

The Lon Poll Service sequentially poles all the devices on the network, and reads nv data per the nv update rules.

The driver polls proxy points when you first create the points and before you bind Network Values (nvs) to the station (local device) using the **Lon Link Manager**. You bind nvs so that most updates occur using nv updates, vs. polling requests. As necessary, polling also occurs to update nv values of Lon components (visible as nvis and nvos in the property sheet of a Lon device). These exist even if proxy points are not created.

Figure 10 Poll Service properties

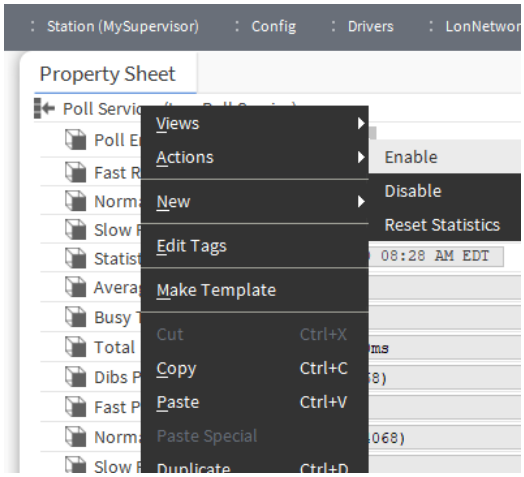


To view these properties, expand **Config→Services** right-click **LonNetwork**, click **Views→AX Property sheet** and expand or click **Poll Service**.

This is a standard network service. For descriptions of each property, refer to the *Niagara Drivers Guide*.

Actions

Figure 11 Lon Poll Service actions menu



To access these actions, right-click the **Poll Service** and click **Actions**.

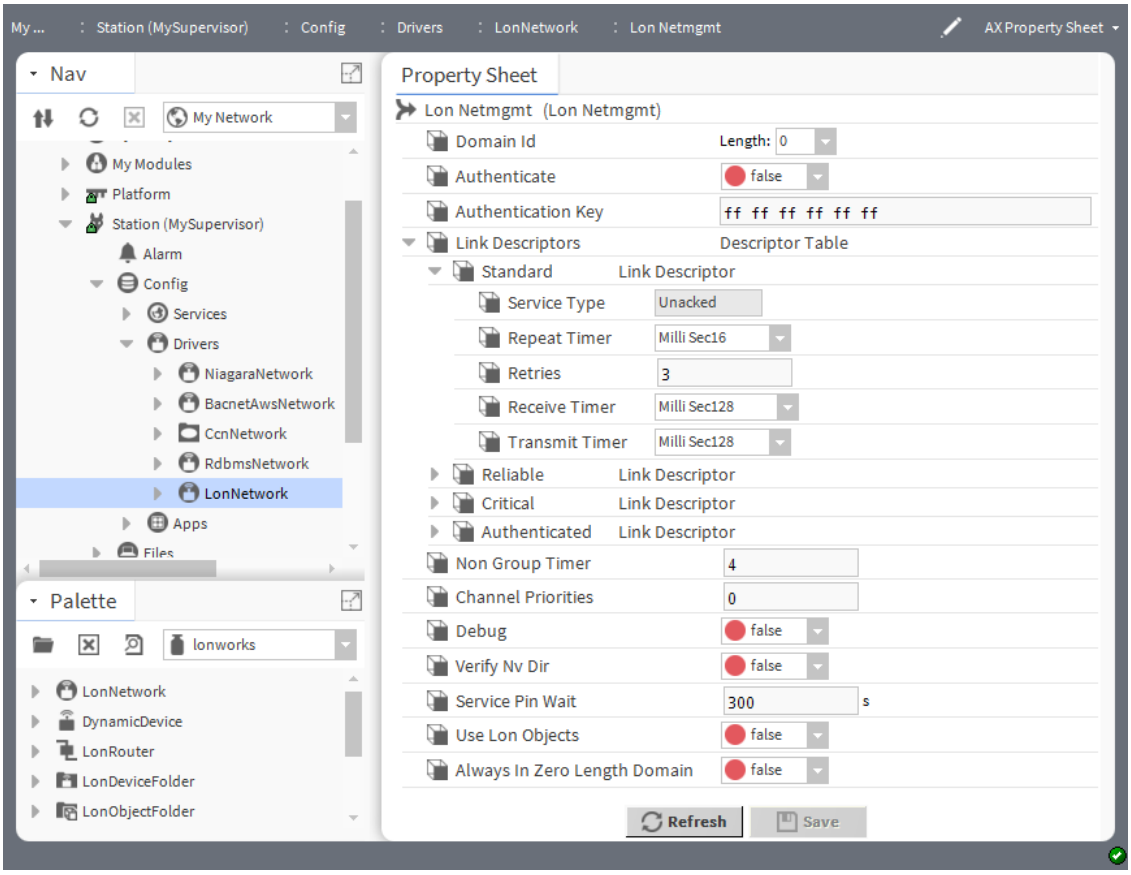
Action	Description
Enable	Turns the function on.
Disable	Turns the function off.
Reset Statistics	Clears all read-only properties.

lonworks-LonNetmgmt

This component serves as the base component for all Lonworks network management functions. It is a frozen container slot of a **LonNetwork**.

This container is the base component for all Lonworks network management functions provided by the station. Leave the **Lon Netmgmt** properties at their defaults, except for perhaps **`Use Lon Objects`**.

Figure 12 Lon Netmgmt properties



To view these properties, right-click the **LonNetwork** node in the Nav tree, click **Views→AX Property Sheet**, click **Lon Netmgmt** and expand one of the **Link Descriptors**.

This **Property Sheet** includes one container, **Link Descriptors** with four descriptors.

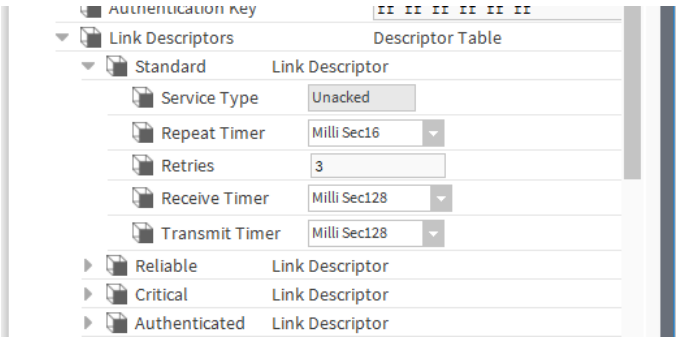
Property	Value	Description
Domain Id	drop-down list (defaults to a zero-length field, that is to a field with nothing in it.)	Defines the default working domain ID for the network. Without further configuration, you can discover all Lon nodes that also belong to the zero-length domain. However, if you cannot discover nodes, they are likely configured on a different domain. Options include 0, 1, 3, 6.
Authenticate	`true` or `false` (default)	Turns authentication on and off.
Authentication Key	text	Configures the key used to secure the request.
Link Descriptors	additional properties	Refer to Link Descriptors, page 68.
Non Group Timer	number (defaults to 4)	Sets up a timer.
Channel Priorities	number (defaults to zero (0))	Sets up the processing priority for the current channel.

Property	Value	Description
Debug	`true` or `false` (default)	Enables (`true`) and disables (`false`) debugging.
Verify Nv Dir	`true` or `false` (default)	Enables (`true`) and disables (`false`) the network variable directory.
Service Pin Wait	seconds (defaults to 300 seconds)	Configures how long the system listens for a service pin message from the **Lon Utilities Manager** view. The framework does not require express listening for a service pin message. It handles an unsolicited service pin message by either adding the node to the station database or selecting the node in the **Database** pane.
Use Lon Objects	`true` or `false` (default)	Configures the driver to use LonObject containers in a LonMark-complient device. `false` places all learned components directly under the Lon device's device object. `true` adds a Lon object container group for the device's Lon components. Each LonMark container represents a LonMark object. This affects the default **Nv Manager** view on a Lon device.
Always in Zero Length Domain	`true` or `false` (default)	Determines which domain to use when commissioning or replacing a Lon device that has two domain entries (domain indexes 0 and 1). By default (`false`) the driver sets the second domain (domain index 1) to not-in-use. In the case where the **`LonNetwork`**'s working domain is not the zero-length domain, the device will not be a zero-length domain member. `true` reverses the default and commissions all devices with two domain entries as members of the zero-length domain.

Link Descriptors

Descriptors specify timers and a retry count applied to the address table of nodes during the bind process. All implicit addressing uses them. Usually, you do not need to change link descriptor properties from their defaults.

Figure 13 Link Descriptors and properties



If Lon network devices respond slowly, you may need to adjust these Link Descriptor properties upwards, often in combination with adjusting **Lon Comm Config** properties.

There are four descriptors or types of binding service: **`Standard`**, **`Reliable`**, **`Critical`** and **`Authenticated`**.

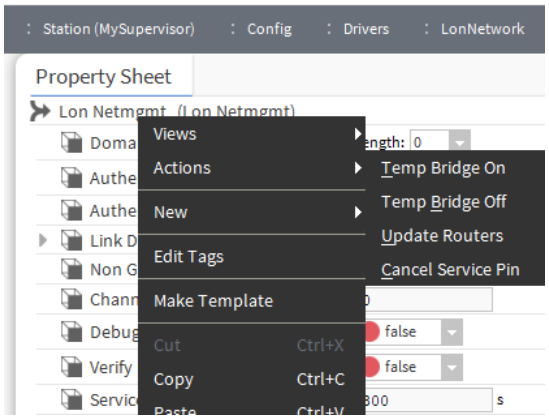
- **`Standard`** (unacknowledged) is the least reliable, yet often most widely used. This is the default service type for a binding. With this service, the message is sent one time and no response is expected. This service is typically used when the highest level of network performance is needed, network bandwidth is limited and the application is not sensitive to the loss of messages.
- **`Reliable`** (unacknowledged/repeated) is recommended for messaging that has been classified as important—if there is a specific need to know that a status has changed. Reliable messaging sends a message three times. The srcDevice sends each message three times and, if the targetDevice receives the message, it discards subsequent messages.
- **`Critical`** (acknowledged) means that the targetDevice must acknowledge the fact that it received a message. The srcDevice continues to send the message until the targetDevice acknowledges receipt of that message. You are limited to the number of bindings that can be defined as critical, because it taxes the system, and often causes communication issues (by loading down the network with message verification traffic).
- **`Authenticated`** is a security service reserved for secure messaging. When using authentication, the srcDevice must identify itself to the targetDevice and vice-versa. This service is rarely used, and only in instances where messaging must be secure.

Each descriptor has the same set of properties.

Property	Value	Description
Service Type	read-only	Identifies the type of connection, `Unacked` (unacknowledged) or `Acked` (acknowledged).
Repeat Timer	drop-down list of millisecond values (defaults to 384 milliseconds)	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.
Retries	number (defaults to 3)	Configures how many times to attempt the function.
Receive Timer	drop-down list of millisecond values (defaults to 96 milliseconds)	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.
All Link Descriptors, Transmit Timer	drop-down list of millisecond values (defaults to 96 milliseconds)	Applies to all outgoing network management messages, all poll messages, and unbound nv update messages from the station.

Actions

Figure 14 Newmgmt actions



To access these actions, right-click the **Lon Netmgmt** and click **Actions**.

Action	Description
Temp Bridge On	Establishes a temporary connection to the server.
Temp Bridge Off	Disconnects a temporary connection from the server.
Update Router	Updates the router records.
Cancel Service Pin	Cancels a Lonworks service pin message for the host node. This action is unique to **Local Lon Device**, and is useful if installing the host using an external Lonworks network management tool (for a station as a Lon node only).

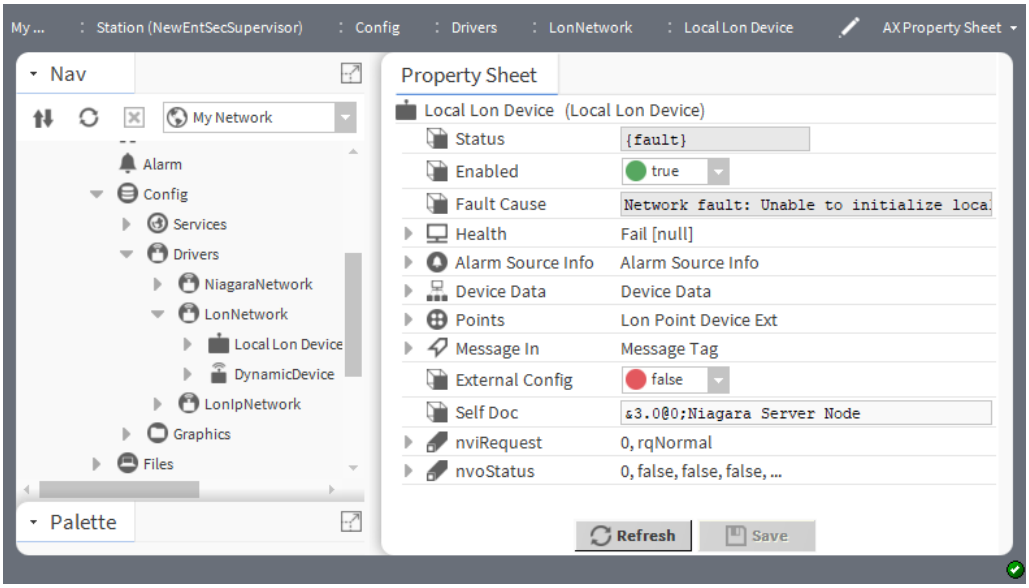
lonworks-LonSetServiceTypeJob

This component implements the `SetServiceType` action for **LonNetmgmt**.

lonworks-LocalLonDevice

This component represents the local interface to the Lonworks fieldbus. Component-wise, it is modeled similar to any other Lon device. As a frozen slot under the **LonNetwork**, it always appears in the network. There is only one **Local Lon Device**. You cannot delete it or duplicate it.

Figure 15 Local Lon device properties



You access these properties by expanding the **LonNetwork**, right-clicking the **Local Lon Device** node in the Nav tree, and clicking **Views→AX Property Sheet**.

With a station as the network manager, you can leave **Local Lon Device** properties at their default settings, and do nothing else at this level in the network.

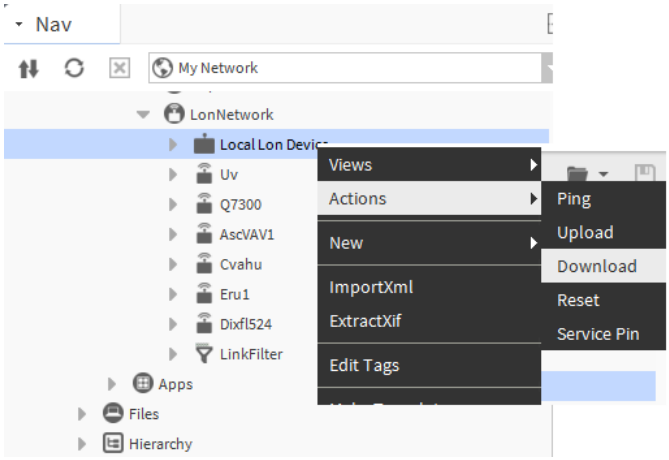
In addition to the common properties (Status, Enabled, Fault Cause, Health and Alarm Source Info), these properties are unique to this component.

Container	Value	Description
Device Data	additional properties	Identifies the Neuron chip and other information about the Local Lon device.
Points (Lon Point Device Extension)	container	Contains the points associated with this device.
Message In	additional properties	Provides a message ID. Represents a single network variable in a Lon device.
External Config	`true` or `false` (default)	Determines where network management takes place. `true` specifies that all Lonworks network management is performed externally, meaning that you do not use **`LonNetwork`** views like the **Lon Device Manager**, **Lon Link Manager**, and so on.
Self Doc	string of text (defaults to: `&3.0@;Niagara Server Node`) This property requires **`External Config`** to be set to `true`	Defines a text string for the host node's self documentation up to 1024 bytes. A single asterisk (*) omits self documentation. **NOTE:** When using self documentation, always retain the leading header portion (`&3.0@`) along with an additional zero (`0`), plus other characters.

Container	Value	Description
nviRequest	additional properties	Configure an network variable input request. For property descriptions, refer to *lonworks-LocalNv*.
nvoStatus	additional properties	Report network variable output status. For property descriptions, refer to *lonworks-LocalNv*.

Actions

Figure 16 Local Lon device actions menu



To access these actions, right-click the **Local Lon Device** and click **Actions**.

Action	Description
Ping	Attempts communication with the device. If successful, the device status reports `{ok}`. If this fails, the system sets device status to `{down}`
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.
Reset	Issues a reset command to the device.
Service Pin	Issues a Lonworks service pin message for the host node. This action is unique to **Local Lon Device**, and is useful if installing the host using an external Lonworks network management tool (station as a Lon node only).

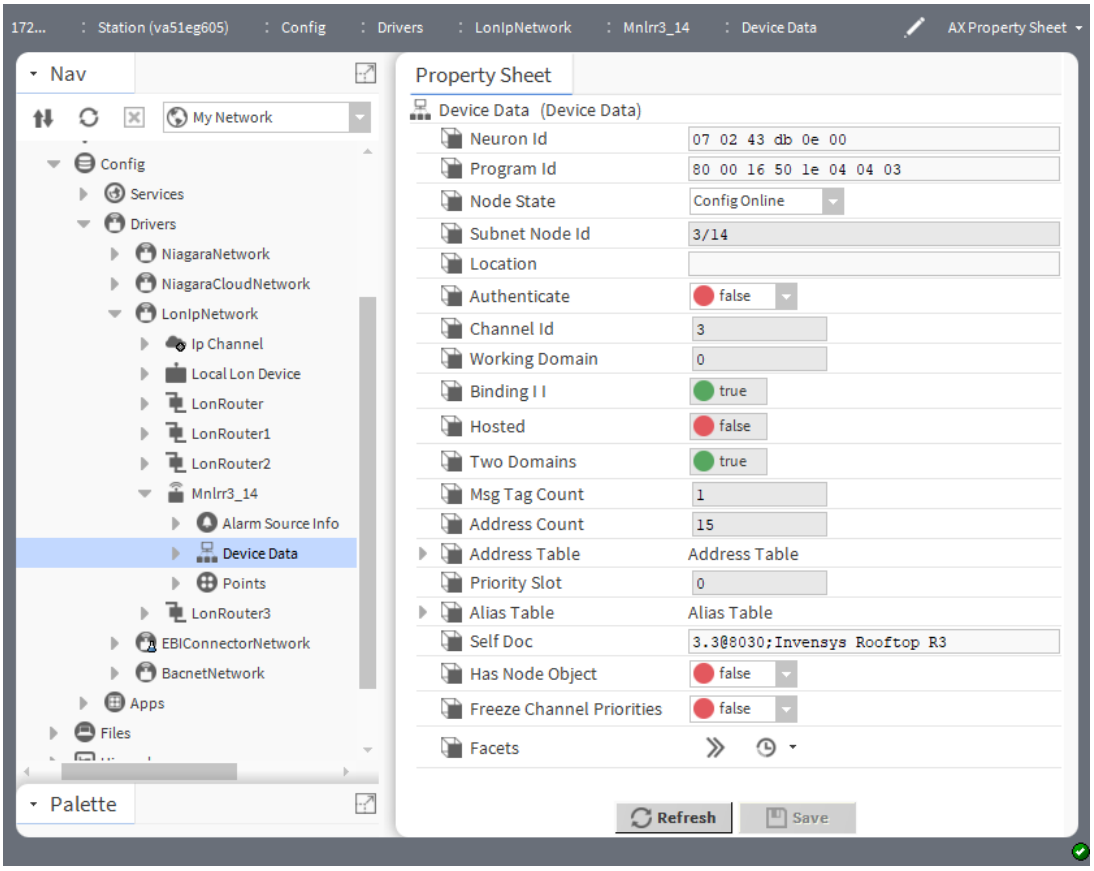
NOTE: In general, you should not invoke actions on the **Local Lon Device**, except a `Service Pin`

lonworks-DeviceData

This component is a frozen container slot under most Lon device objects. It contains the data needed to represent a specific Neuron chip including state information and data that can be configured in the Neuron tables. This component is otherwise referred to as the ExtDeviceData slot.

In general, it is recommended that you leave slot values under a device's **DeviceData** container at default settings. Largely, these are data structures that contain address and state information administered by the network management functions of the station.

Figure 17 Device Data property sheet



To access this view, expand the **Local lon Device** component in the Nav tree, and right-click **Device Data→-Views→Property Sheet**.

Property	Value	Description
Neuron Id	hexadecimal number	Maps the device in the station database to a different physical device (or no device, if the address is not available). While most **DeviceData** properties should be left at default values, this is one property you may routinely edit.
Program Id	hexadecimal number	Identifies the interface to a dynamic device (network node). Different external interfaces should be associated with unique Program Ids. Some vendors support multiple device types in the same hardware platform. Refer to AppDownload.
Node State	drop-down list	**CAUTION:** Do not set this property for any hosted Lon node (or the **`Local Lon Device`**) to `Applicationless` or the device will become irretrievably non-functional. Instead, use the **Lon Device Manager** view of the Lon network to make changes to Lon devices.
Subnet Node Id	read-only	Reports the assigned Lonworks subnet/node address. Node must be unique for this device on the channel.
Location	text	Provides the location of the device.

Property	Value	Description
Authenticate	`true` or `false` (default) [used once]	Turns authentication on and off.
Channel Id	read-only or a number (defaults to zero (0))	Reports or defines the ID of the channel to which the device is connected.
Binding I I	read-only	Reports if this type of binding is enabled (`true`) or disabled (`false`).
Hosted	read-only	
Two Domains	read-only	Reports is the device belongs to two domains.
Msg Tag Count	read-only	Reports the number of tag messages associated with the device.
Address Count	read-only	Reports the number of addresses associated with the device.
Address Table	list	All properties are marked as "Not In Use."
Priority Slot	read-only	Reports the device's priority setting.
Alias Table	additional property	Associates a network variable selector with a network variable index. These identifiers associate a network variable update message with a network variable within the receiving application. Refer to a subtopic in this guide.
Self Doc	string of text (defaults to: `&3.0@; Niagara Server Node`) This property requires **`External Config`** to be set to `true`	Defines a text string for the host node's self documentation up to 1024 bytes. A single asterisk (*) omits self documentation. **NOTE:** When using self documentation, always retain the leading header portion (`&3.0@`) along with an additional zero (`0`), plus other characters.
Has Node Object	`true` or `false` (default)	Determines if the node includes an object (`true`) or not (`false`).
Freeze Channel Priorities	`true` or `false` (default)	Enables (`true`) and disables (`false`) the ability to change channel priorities.
Facets	additional properties	Determine how values are formatted for display depending on the context and the type of data. For example, instead of the Boolean facets `trueText` and `falseText` you may want to display `ON` and `OFF`, `Access Granted` and `Access Denied` or `Locked` and `Unlocked`. You access facets by clicking an **Edit** button or a chevron >>. Both open an **Edit Facets** window.

lonworks-AliasTable

This component contains the data in a Lonworks device's alias table. It appears as a child of the device's **Device Data** container or **ExtDeviceData** slot.

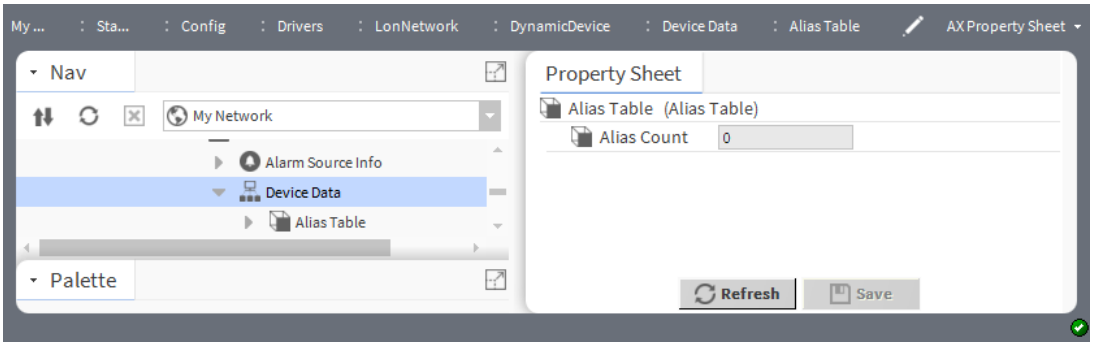
To support network variable selection at layer 6, each network variable on a device has a unique network variable index for that device. Network variable indexes are assigned sequentially for each network variable on

the device, starting with index 0. For example, a device with a switch output and a switch feedback input may use index 0 for the output and index 1 for the input.

When sending a network variable update, the layer-6 implementation uses this table to translate the network variable index on the sending device to a network variable selector. This action translates the network variable selector to a network variable index on the receiving device.

(Description adapted from the Echelon *Introduction to the LonWorks® Platform*, Revision 2.)

Figure 18 Alias Table property



To access this table, expand **Config→Drivers→LonNetwork**, right-click a dynamic device node, click **Views→AX Property Sheet**, expand **Device Data** and click **Alias Table**.

Property	Value	Description
Alias Count	read-only	Reports the number of alias entities.

lonworks-ExtAddressTable

This component applies only to a device that uses Extended Network Management as defined in CEA-709.1-B, and contains the data in its extended address table. It appears as a child of the Lon device's **Device Data** container (**`ExtDeviceData`** slot).

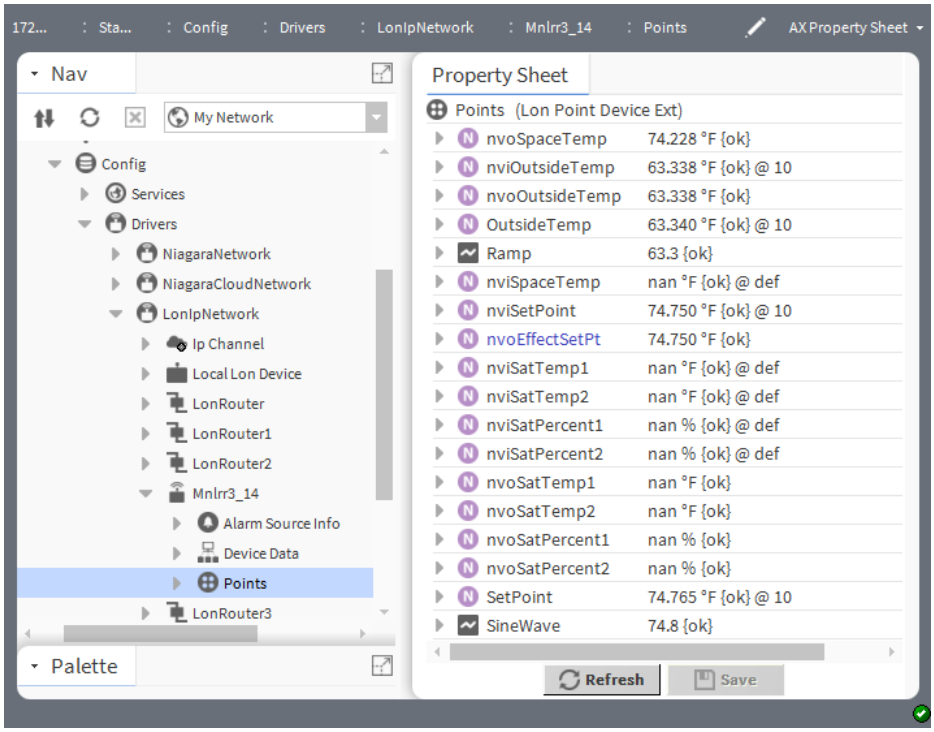
lonworks-ExtDeviceData

This component is a frozen container slot under any Lon device that uses Extended Network Management, as defined in CEA-709.1-B (replacing the more typical **`DeviceData`** slot). Like DeviceData, it represents the device's state and configuration data, including the larger number of possible address tables in its child **ExtAddressTable** slot (unique to ExtDeviceData).

lonworks-LonPointDeviceExt

This component is the Lonworks implementation of the framework's **PointDeviceExt**. Its primary view is the **Lon Point Manager**. It is the only device extension for any Lon device. In the **`lonworks`** palette, it is available under the **`DynamicDevice`** as **`Points`**.

Figure 19 Points



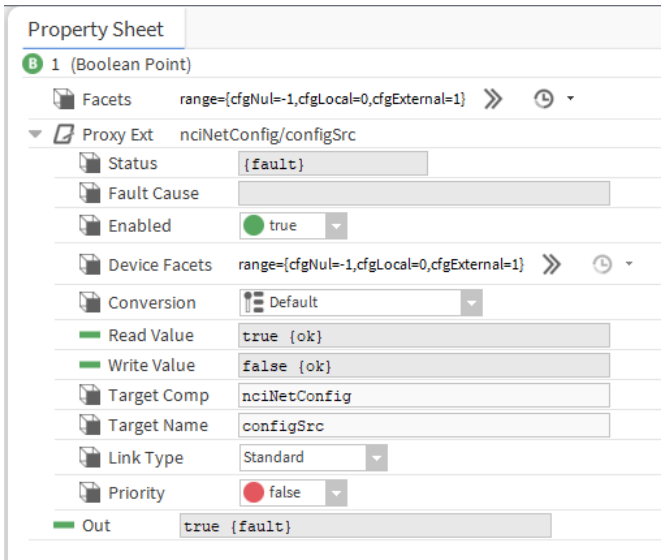
To view these properties, expand the **LonNetwork**, expand a device folder right-click **Points** and click **View→AX Property Sheet**.

Point names depend on the configuration. In the screen capture the points are named for the direction of communication: nvi for Network Variable Input and nvo for Network Variable Output.

lonworks-LonBooleanProxyExt

This component is the Lon proxy extension for a BooleanPoint. It will link a single Boolean point to a lonworks LonPrimitive. The appropriate conversions will be performed.

Figure 20 LonBooleanProxyExt properties



To access this view, expand **LonNetwork**, expand a device, expand the **Points** folder, right click **Boolean Point** and click **Views→AX Property Sheet**

In addition to the common properties (Status, Fault Cause and Enabled), these properties are unique to this component.

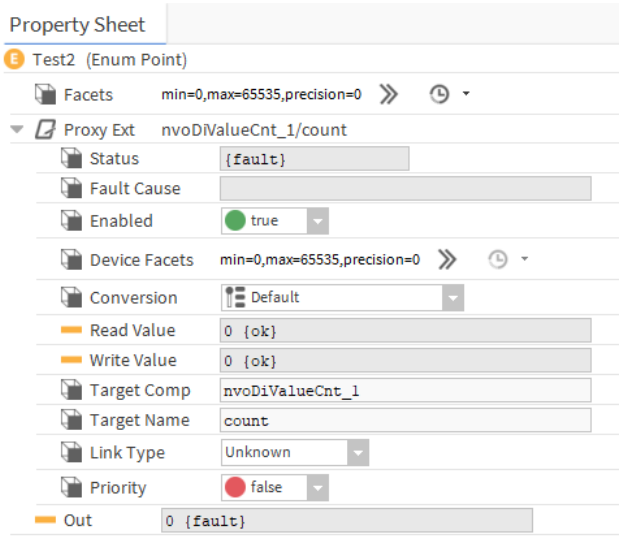
Property	Value	Description
Device Facets	additional properties	Determine how values are formatted for display depending on the context and the type of data. For example, instead of the Boolean facets `trueText` and `falseText` you may want to display `ON` and `OFF`, `Access Granted` and `Access Denied` or `Locked` and `Unlocked`. You access facets by clicking an **Edit** button or a chevron **>>**. Both open an **Edit Facets** window.
Conversion	drop-down list (defaults to Default)	Defines how the system converts proxy extension units to parent point units. `Default` automatically converts similar units (such as Fahrenheit to Celsius) within the proxy point. **NOTE:** In most cases, the standard `Default` conversion is best. `Linear` applies to voltage input, resistive input and voltage output writable points. Works with linear-acting devices. You use the Scale and Offset properties to convert the output value to a unit other than that defined by device facets. `Linear With Unit` is an extension to the existing linear conversion property. This specifies whether the unit conversion should occur on "Device Value" or "Proxy Value". The new linear with unit convertor, will have a property to indicate whether the unit conversion should take place before or after the scale/offset conversion. `Reverse Polarity` applies only to Boolean input and relay output writable points. Reverses the logic of the hardware binary input or output. `500 Ohm Shunt` applies to voltage input points only. It reads a 4-to-20mA sensor, where the Ui input requires a 500 ohm resistor wired across (shunting) the input terminals. `Tabular Thermistor` applies to only a Thermistor input point and involves a custom resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Thermistor Type 3` applies to an Thermistor Input point, where this selection provides a "built-in" input resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Generic Tabular` applies to non-linear support for devices other than for thermistor temperature sensors with units in temperature. Generic Tabular uses a lookup table method similar to the "Thermistor Tabular" conversion, but without predefined output units.
Read Value	read-only	Displays the last value read from the device, expressed in device facets.

Property	Value	Description
Write Value	read-only	Displays the last value written, using device facets.
Target Comp	read-only	Represents a target Lon component, for example nvoUnitStatus.
Target Name	read-only	Provides a name for the LonData element in the Lon component. For example: heatOutputPrimary.
Link Type	drop-down list	Configures the type of link between nodes and devices. Options are: `Unknown`, `Standard`, `Reliable`, `Critical`, `Authenticated` and `Poll Only`.
Priority	`true` or `false` (default)	Configures processing sequence.

lonworks-LonEnumProxyExt

This component defines a Lon proxy point with a StatusEnum output. The driver performs the appropriate conversions.

Figure 21 LonEnumProxyExt Property Sheet



To access this view, expand **LonNetwork**, expand a device, expand the **Points** folder, right click **Enum Point** and click **Views→AX Property Sheet**.

In addition to the common properties (Status, Fault Cause and Enabled), these properties are unique to this component.

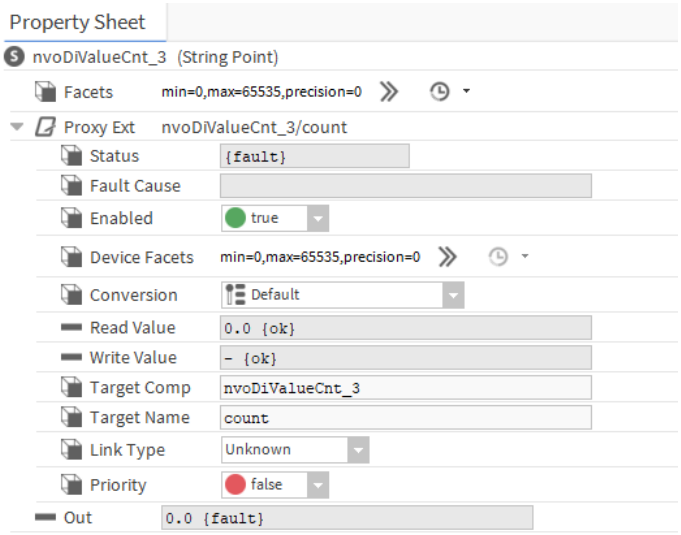
Property	Value	Description
Device Facets	additional properties	Determine how values are formatted for display depending on the context and the type of data. For example, instead of the Boolean facets `trueText` and `falseText` you may want to display `ON` and `OFF`, `Access Granted` and `Access Denied` or `Locked` and `Unlocked`. You access facets by clicking an **Edit** button or a chevron >>. Both open an **Edit Facets** window.
Conversion	drop-down list (defaults to Default)	Defines how the system converts proxy extension units to parent point units. `Default` automatically converts similar units (such as Fahrenheit to Celsius) within the proxy point. **NOTE:** In most cases, the standard `Default` conversion is best. `Linear` applies to voltage input, resistive input and voltage output writable points. Works with linear-acting devices. You use the Scale and Offset properties to convert the output value to a unit other than that defined by device facets. `Linear With Unit` is an extension to the existing linear conversion property. This specifies whether the unit conversion should occur on "Device Value" or "Proxy Value". The new linear with unit convertor, will have a property to indicate whether the unit conversion should take place before or after the scale/offset conversion. `Reverse Polarity` applies only to Boolean input and relay output writable points. Reverses the logic of the hardware binary input or output. `500 Ohm Shunt` applies to voltage input points only. It reads a 4-to-20mA sensor, where the Ui input requires a 500 ohm resistor wired across (shunting) the input terminals. `Tabular Thermistor` applies to only a Thermistor input point and involves a custom resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Thermistor Type 3` applies to an Thermistor Input point, where this selection provides a "built-in" input resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Generic Tabular` applies to non-linear support for devices other than for thermistor temperature sensors with units in temperature. Generic Tabular uses a lookup table method similar to the "Thermistor Tabular" conversion, but without predefined output units.
Read Value	read-only	Displays the last value read from the device, expressed in device facets.
Write Value	read-only	Displays the last value written, using device facets.
Target Comp	read-only	Represents a target Lon component, for example nvoUnitStatus.

Property	Value	Description
Target Name	read-only	Provides a name for the LonData element in the Lon component. For example: heatOutputPrimary.
Link Type	drop-down list	Configures the type of link between nodes and devices. Options are: `Unknown`, `Standard`, `Reliable`, `Critical`, `Authenticated` and `Poll Only`.
Priority	`true` or `false` (default)	Configures processing sequence.

lonworks-LonStringProxyExt

This component is the Lon proxy extension for a StringPoint. It links a single string point to a Lonworks LonPrimitive and performs the appropriate conversions.

Figure 22 LonStringProxyExt properties



To access this view, expand **LonNetwork**, expand a device, expand the **Points** folder, right click **String Point** and click **Views→AX Property Sheet**.

In addition to the common properties (Status, Fault Cause and Enabled), these properties are unique to this component.

Property	Value	Description
Device Facets	additional properties	Determine how values are formatted for display depending on the context and the type of data. For example, instead of the Boolean facets `trueText` and `falseText` you may want to display `ON` and `OFF`, `Access Granted` and `Access Denied` or `Locked` and `Unlocked`. You access facets by clicking an **Edit** button or a chevron >>. Both open an **Edit Facets** window.
Conversion	drop-down list (defaults to Default)	Defines how the system converts proxy extension units to parent point units. `Default` automatically converts similar units (such as Fahrenheit to Celsius) within the proxy point.

Property	Value	Description
		NOTE: In most cases, the standard `Default` conversion is best. `Linear` applies to voltage input, resistive input and voltage output writable points. Works with linear-acting devices. You use the Scale and Offset properties to convert the output value to a unit other than that defined by device facets. `Linear With Unit` is an extension to the existing linear conversion property. This specifies whether the unit conversion should occur on "Device Value" or "Proxy Value". The new linear with unit convertor, will have a property to indicate whether the unit conversion should take place before or after the scale/offset conversion. `Reverse Polarity` applies only to Boolean input and relay output writable points. Reverses the logic of the hardware binary input or output. `500 Ohm Shunt` applies to voltage input points only. It reads a 4-to-20mA sensor, where the Ui input requires a 500 ohm resistor wired across (shunting) the input terminals. `Tabular Thermistor` applies to only a Thermistor input point and involves a custom resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Thermistor Type 3` applies to an Thermistor Input point, where this selection provides a "built-in" input resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Generic Tabular` applies to non-linear support for devices other than for thermistor temperature sensors with units in temperature. Generic Tabular uses a lookup table method similar to the "Thermistor Tabular" conversion, but without predefined output units.
Read Value	read-only	Displays the last value read from the device, expressed in device facets.
Write Value	read-only	Displays the last value written, using device facets.
Target Comp	read-only	Represents a target Lon component, for example nvoUnitStatus.
Target Name	read-only	Provides a name for the LonData element in the Lon component. For example: heatOutputPrimary.
Link Type	drop-down list	Configures the type of link between nodes and devices. Options are: `Unknown`, `Standard`, `Reliable`, `Critical`, `Authenticated` and `Poll Only`.
Priority	`true` or `false` (default)	Configures processing sequence.

lonworks-LonFilePos

This component represents a SNVT_file_pos.

lonworks-LonFileReq

This component represents a SNVT_file_req.

lonworks-LonFileStatus

This component represents a SNVT_file_status.

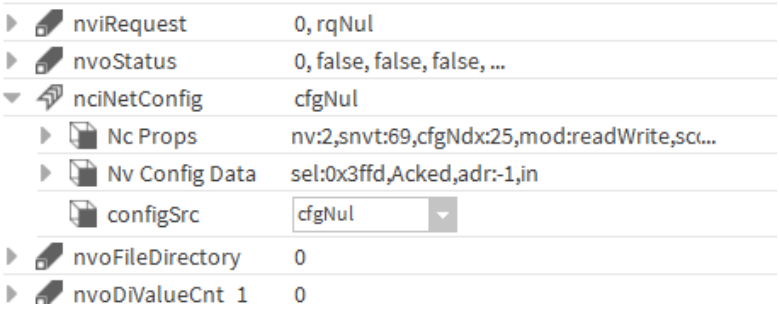
lonworks-LonFloatProxyExt

This component defines a Lon proxy point with a StatusFloat output. The driver performs the appropriate conversions.

lonworks-NetworkConfig

This component represents a single nci (Network Configurable Input) in a Lon device. It provides specific support for runtime updates and contains data needed to support network management.

Figure 23 Nci properties

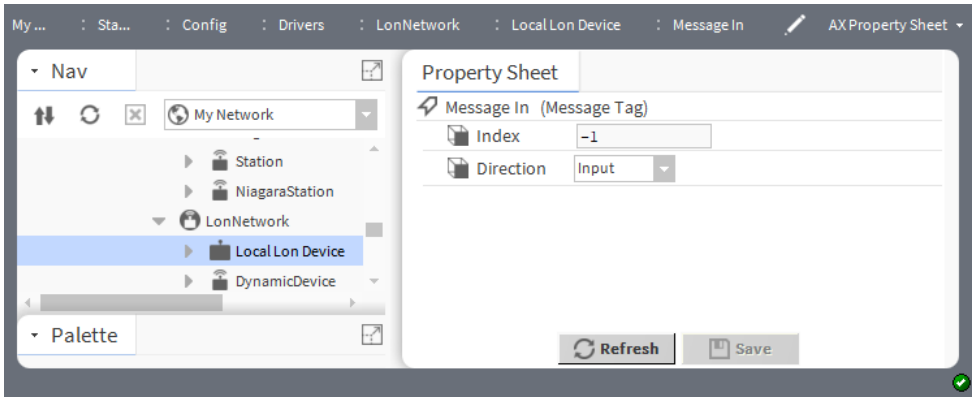


Property	Value	Description
configSrc	drop-down list	Configures a network source.

lonworks-MessageTag

This component represents a single network variable in a Lon device. It provides specific support for runtime updates and contains data needed to support network management. The MessageTag is available in the **`lonworks`** module.

Figure 24 Message Tag properties



To view these properties, expand **LonNetwork** node in the Nav tree, right-click **Local Lon Device**, click **Views→AX Property Sheet** and click**Message In**.

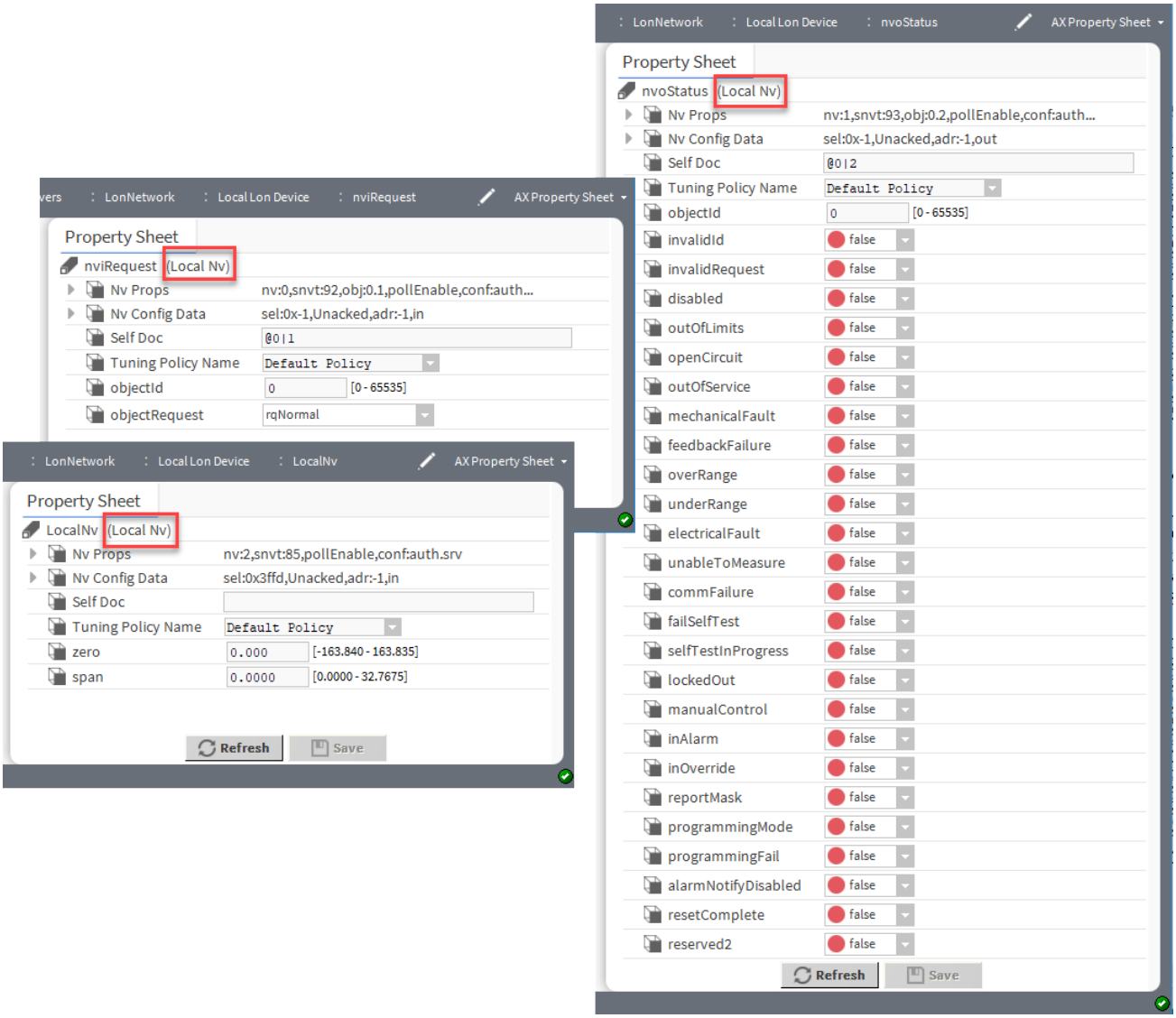
Property	Value	Description
Index	number (defaults to –1, meaning disabled)	Represents a single network variable in a Lon device.
Direction	drop-down list (defaults to Input)	Configures or indicates the direction of travel of the message: to the station (`Input`) or from the station `Output`.

lonworks-LocalNv

This component is a single network variable in a **Local Lon Device**. The external interface of the local device represents it to other devices on the Lonworks trunk.

A network variable (nv) may configure input (nvi), output (nvo).

Figure 25 Local Nv properties



To access these views, double-click the **Local Lon Device** component in the Nav tree, and right-click **nvoStatus→Views→AX Property Sheet**.

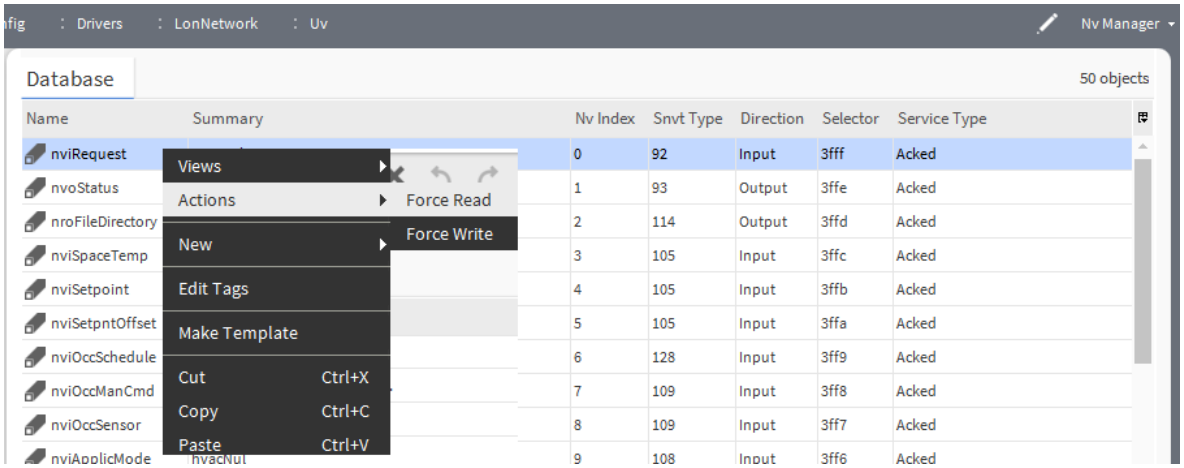
Property	Value	Description
Nv Props	additional properties	Reports and configures NV (Network Variable) input properties. For property descriptions, refer to *lonworks-LocalNv*. Refer to Nv Props, page 86.
Nv Config Data	additional properties	Maps directly to a device's network variable and reports NV (Network Variable) output configuration data. For property descriptions, refer to *lonworks-LocalNv*. Refer to Nv Config Data, page 88.
objectid	number from 0–65535 (defaults to zero (0))	Identifies the LonMark object for the LonObject. It must be unique in the device among all objects.
objectrequest	drop-down list (defaults to `rqNormal`)	Selects an action to perform on the selected object.
zero	number	
span	number to four decimal places—from 0.0000 to 32.7675	
Self Doc	string of text (defaults to: `&3.0@;Niagara Server Node`) This property requires **`External Config`** to be set to `true`	Defines a text string for the host node's self documentation up to 1024 bytes. A single asterisk (*) omits self documentation. **NOTE:** When using self documentation, always retain the leading header portion (`&3.0@`) along with an additional zero (`0`), plus other characters.
Tuning Policy Name	drop-down list (defaults to Default Policy)	Selects the appropriate tuning policy by name.
objectId	number from 0–65535 (defaults to zero (0))	Identifies the LonMark object for the LonObject. It must be unique in the device among all objects.
objectRequest	drop-down list	Selects the type of request. The explanation of each request is beyond the scope of this document.
invalidId	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
invalidRequest	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
disabled	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
outOfLimits	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
openCircuit	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.

Property	Value	Description
outOfService	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
mechanicalFault	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
FeedbackFailure	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
overRange	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
underRange	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
electricalFault	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
UnableToMeasure	`true` or `false` (default)	Enables and disables this feature in nvoStatus.
commFailure	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
failSelfTest	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
selfTestInProgress	`true` or `false` (default)	Enables and disables this feature in nvoStatus.
lockedOut	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
manualControl	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
inAlarm	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
inOverride	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
reportMask	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
programming-Mode	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
programmingFail	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
alarmNotifyDisa-bled	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
resetComplete	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.
reserved2	`true` or `false` (default)	Enables (`true`) and disables (`false`) this feature in nvoStatus.

Actions

By default, Lon components have two available actions, with right-click menu access from the Lon device's **Property Sheet**, **Nc Manager** and **Nv Manager** views.

Figure 26 Lon nvirequest actions

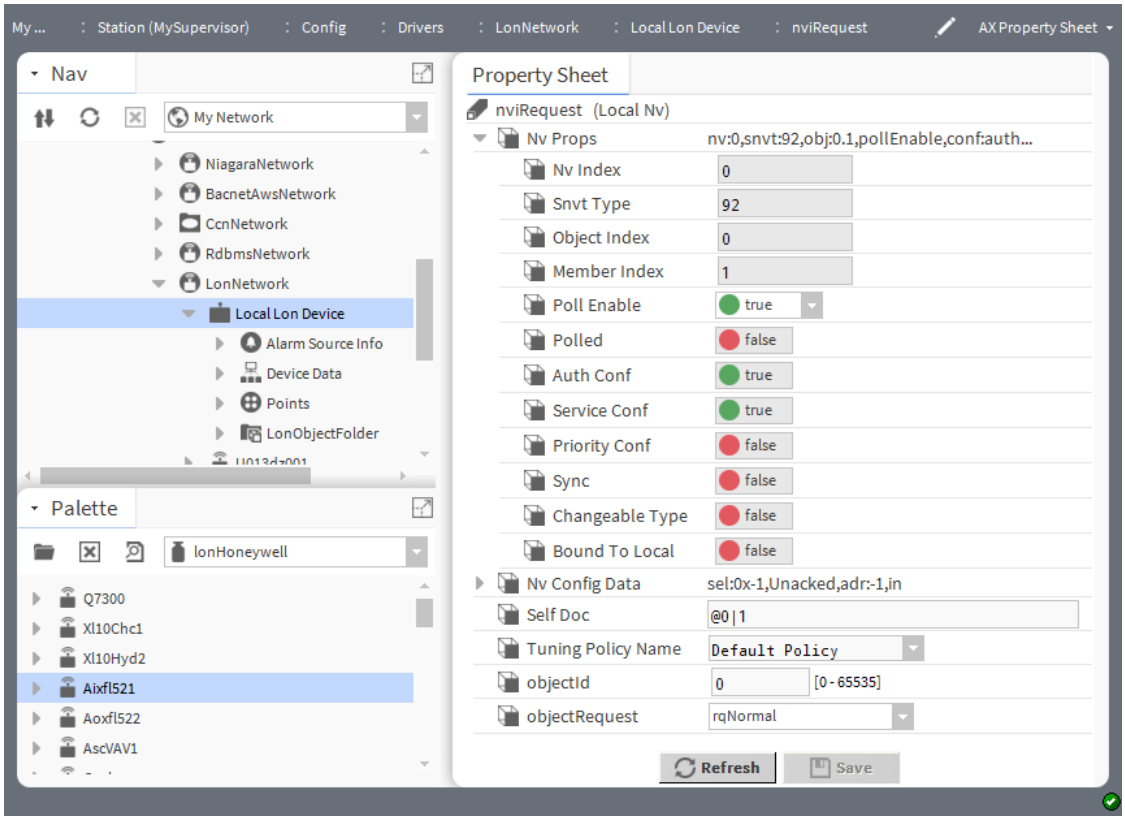


To access these actions, right-click the **nviRequest** and click **Actions**.

Action	Description
Force Read	Causes a read of the request's Lon data from the device.
Force Write	Sends the Lon request's data values to the device again. This does not apply to nvos.

Nv Props

Figure 27 Nv Props properties (under nviRequest)

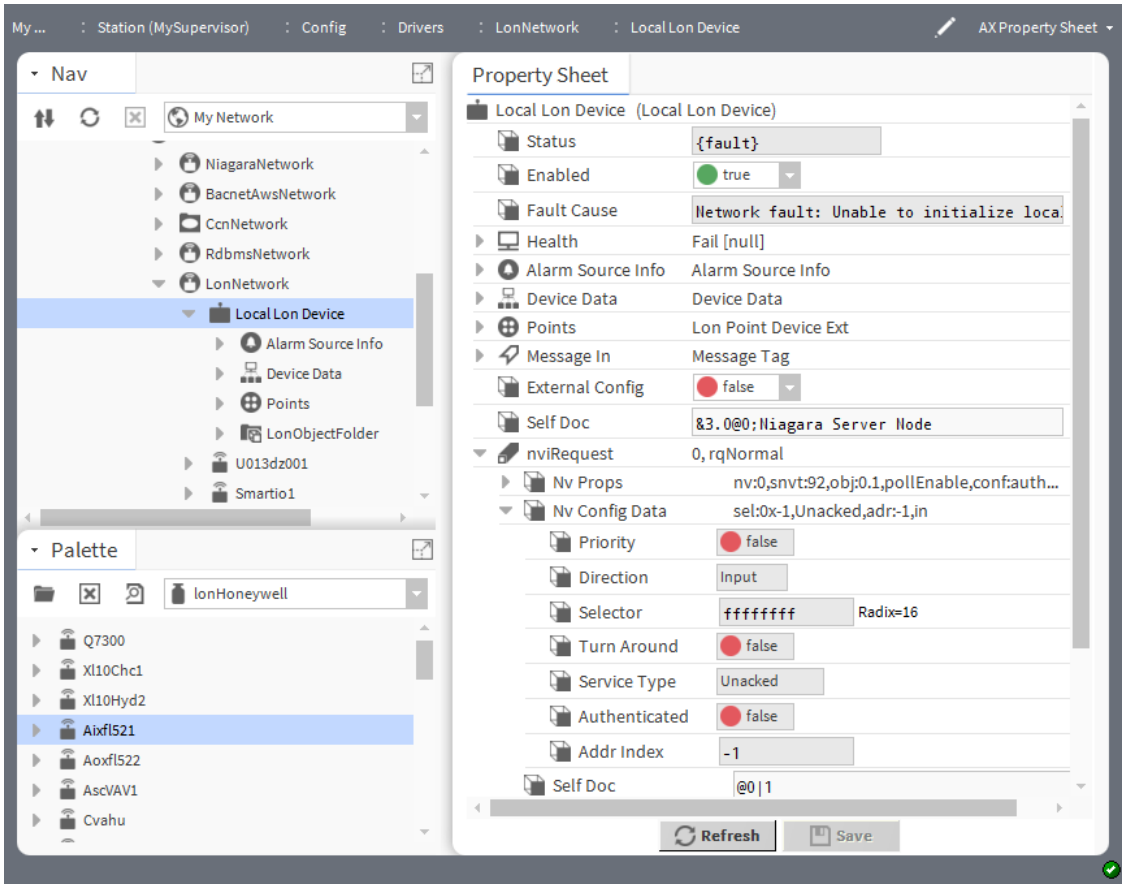


To access these properties, right-click the **Local Lon Device** component in the Nav tree, click **Views→AX Property Sheet**, expand **nviRequest** or **nviStatus** and expand **Nv Props**.

Property	Value	Description
Nv Index	read-only	Reports the number associated with this request's standard network variable type.
Snvt Type	read-only	Reports a number for the Standard Network Variable Type (Snvt).
Object Index	read-only	Reports the object number.
Member Index	read only	Reports the member number.
Poll Enable	`true` (default) or `false`	Turns polling on (`true`) and off (`false`).
Polled	read-only	Reports if this device was polled (`true`) or not (`false`).
Auth Conf	read-only	Reports if authentication is enabled.
Service Conf	read-only	Reports additional information.
Priority Conf	read-only	Reports if request priority has been set.
Sync	read only	Reports if the system is configured to synchronize device data.
Changeable Type	read-only	Reports if the system is configured to change the type.
Bound To Local	read-only	Reports if the point is bound locally. The system sets this property when an output nv with a proxy is bound. An output nv is not polled when it is bound locally. Instead, nv updates are received from the device.

Nv Config Data

Figure 28 NV Config Data properties (under nviRequest)



To access these properties, right-click the **Local Lon Device** component in the Nav tree, click **Views→AX Property Sheet**, expand **nviRequest** or **nviStatus** and expand **Nv Config Data**.

Property	Value	Description
Priority	`true` or `false` (default)	Configures processing sequence.
Direction	drop-down list (defaults to Input)	Configures or indicates the direction of travel of the message: to the station (`Input`) or from the station `Output`.
Selector	number between 0 and 16383 or read-only	Configures or reports the selector to exclude. The **Lon Link Manager** uses decimal notation for selector numbers, but a Lon device's **Nv Manager** lists selectors for nvs using hexadecimal numbers. For example: Selector = `16`.
Turn Around	read-only	An amount of time in µsec.
Service Type	read-only	Identifies the type of connection, `Unacked` (unacknowledged) or `Acked` (acknowledged).
Authenticated	read-only	Reports the state of the request.
Addr Index	read-only	Reports a number associated with a request.

lonworks-NetworkVariable

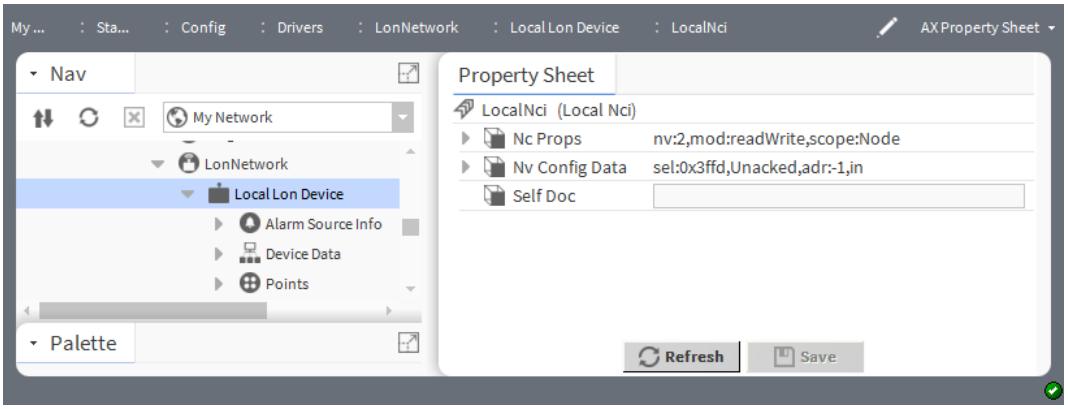
This component represents a single point in a Lon device, as one type of Lon component. It provides specific support for runtime updates and contains data needed to support network management.

lonworks-LocalNci

This component is a single network variable in a **Local Lon Device** that represents the external interface of the local device as presented to other devices on the Lonworks trunk.

To add this component to a **LonNetwork** local Lon device, expand **Config→Drivers→LonNetwork** and click **New**.

Figure 29 LocalNci properties



To access this view, right-click the **Local Lon Device** component in the Nav tree, and click **Views→AX Property Sheet**.

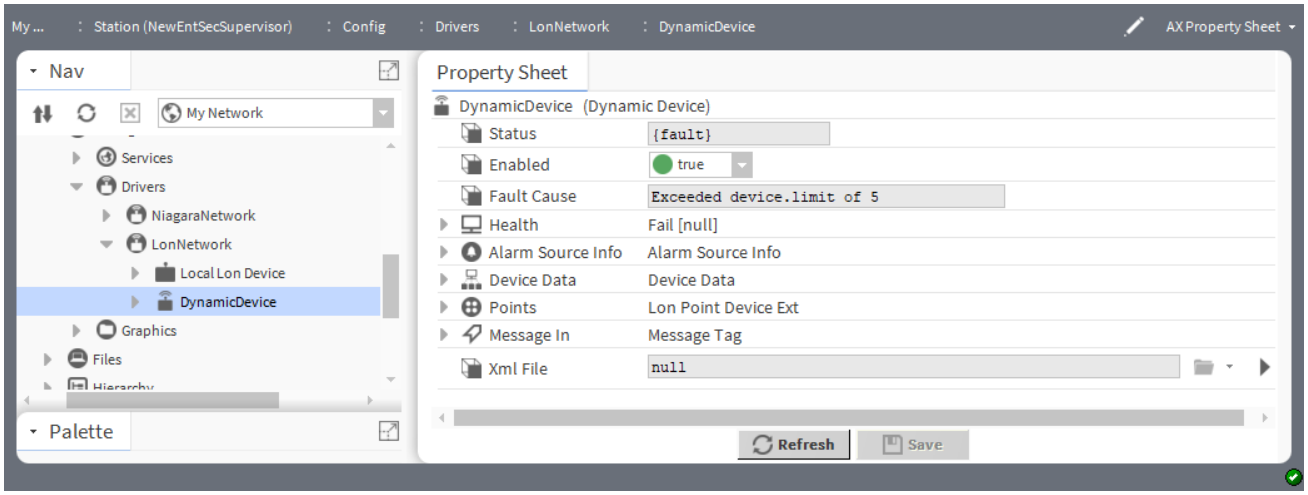
Container	Value	Description
Nv Props	additional properties	Reports and configures NV (Network Variable) input properties. For property descriptions, refer to *lonworks-LocalNv*.
Nv Config Data	additional properties	Maps directly to a device's network variable and reports NV (Network Variable) output configuration data. For property descriptions, refer to *lonworks-LocalNv*.
Self Doc	text string	Defines a text string for the host node's self documentation up to 1024 bytes. A single asterisk (*) omits self documentation. **NOTE:** When using self documentation, always retain the leading header portion (`&3.0@`) along with an additional zero (`0`), plus other characters.

lonworks-DynamicDevice

This component represents a Lonworks node. It adds support to the **LonDevice** with dynamically created components from a device's self documentation or a Lon Xml (`.lnml`) file. Almost all Lon devices are implemented as dynamic device components.

In addition to the standard component views (Property Sheet, Wire Sheet, Category Sheet, Slot Sheet, and Link sheet, this component has two special views: **Nv Manager** and the **Nc Manager**.

Figure 30 Dynamic device properties



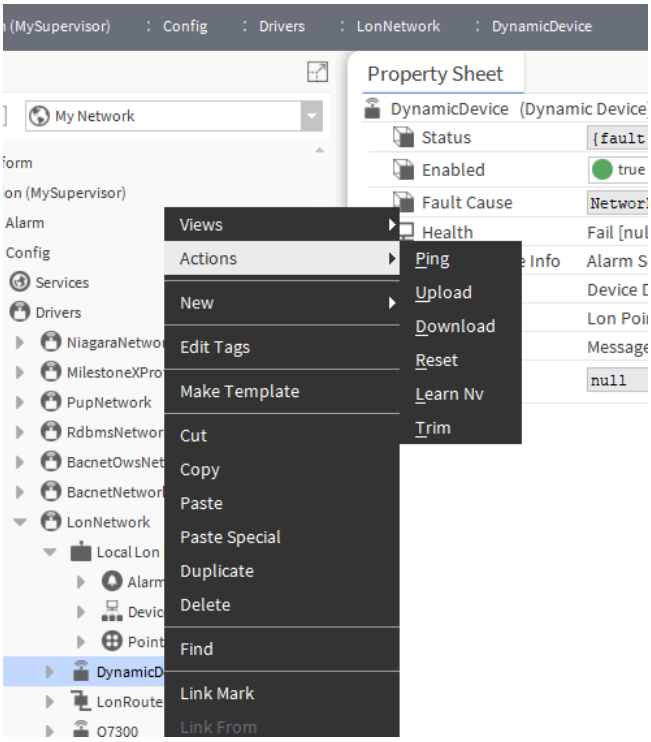
To access these properties, expand **Config→Drivers→LonNetwork**, right-click a dynamic device node and click **Views→AX Property Sheet**. (Your dynamic device node is probably named for the device rather than named "Dynamic Device.")

In addition to the standard properties (Status, Enaled, Fault Cause, Health and Alarm Source Info), these properties are unique to this component.

Property container	Example properties	Description
Device Data	additional properties	Identifies the Neuron chip and other information about the Local Lon device.
Points	contained	Holds individual proxy points.
Message In	additional properties	Provides a message ID. Represents a single network variable in a Lon device.
Xml File	text	Configures the path to the xml file for a specific function.

Actions

Figure 31 Lon device actions



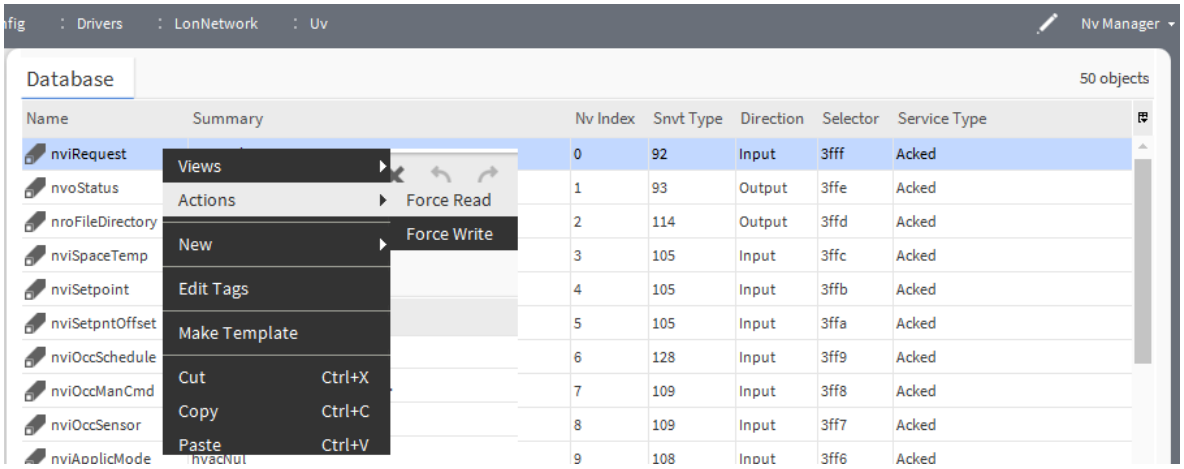
To access these actions, expand **LonNetwork**, right-click a dynamicdevice and click **Actions**.

NOTE: Similar actions exist for the **Local Lon Device**, with exception of Learn Nv and Trim.

Action	Description
Ping	Attempts communication with the device. If successful, the device status reports `{ok}`. If this fails, the system sets device status to `{down}`
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.
Reset	Issues a reset command to the device.
Learn Nv	Builds the Lon device's child Lon components as a collection of nvs, ncis, and (if available) cps variables based on the Lonworks self documentation that resides in the device.
Service Pin	Issues a Lonworks service pin message for the host node. This action is unique to **Local Lon Device**, and is useful if installing the host using an external Lonworks network management tool (station as a Lon node only).

Actions

Figure 32 nviRequest/nvoStatus actions



By default, Lon components have two available actions, with right-click menu access from the Lon device's **Property Sheet**, **Nc Manager** and **Nv Manager** views.

Action	Description
Force Read	Causes a read of the item's Lon data from the device.
Force Write	Sends the Lon component's Lon data values to the device again. This does not apply to nvos.

lonworks-AliasTable

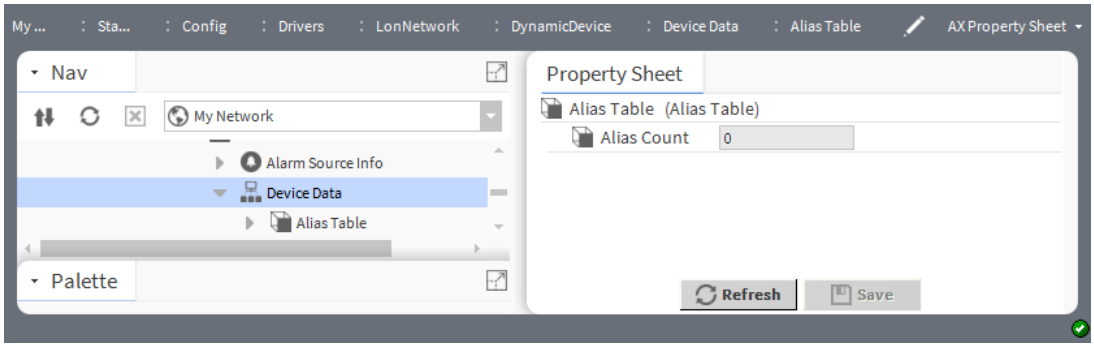
This component contains the data in a Lonworks device's alias table. It appears as a child of the device's **Device Data** container or **ExtDeviceData** slot.

To support network variable selection at layer 6, each network variable on a device has a unique network variable index for that device. Network variable indexes are assigned sequentially for each network variable on the device, starting with index 0. For example, a device with a switch output and a switch feedback input may use index 0 for the output and index 1 for the input.

When sending a network variable update, the layer-6 implementation uses this table to translate the network variable index on the sending device to a network variable selector. This action translates the network variable selector to a network variable index on the receiving device.

(Description adapted from the Echelon *Introduction to the LonWorks® Platform*, Revision 2.)

Figure 33 Alias Table property



To access this table, expand **Config→Drivers→LonNetwork**, right-click a dynamic device node, click **Views→AX Property Sheet**, expand **Device Data** and click **Alias Table**.

Property	Value	Description
Alias Count	read-only	Reports the number of alias entities.

lonworks-ExtAddressTable

This component applies only to a device that uses Extended Network Management as defined in CEA-709.1-B, and contains the data in its extended address table. It appears as a child of the Lon device's **Device Data** container (**`ExtDeviceData`** slot).

lonworks-ExtDeviceData

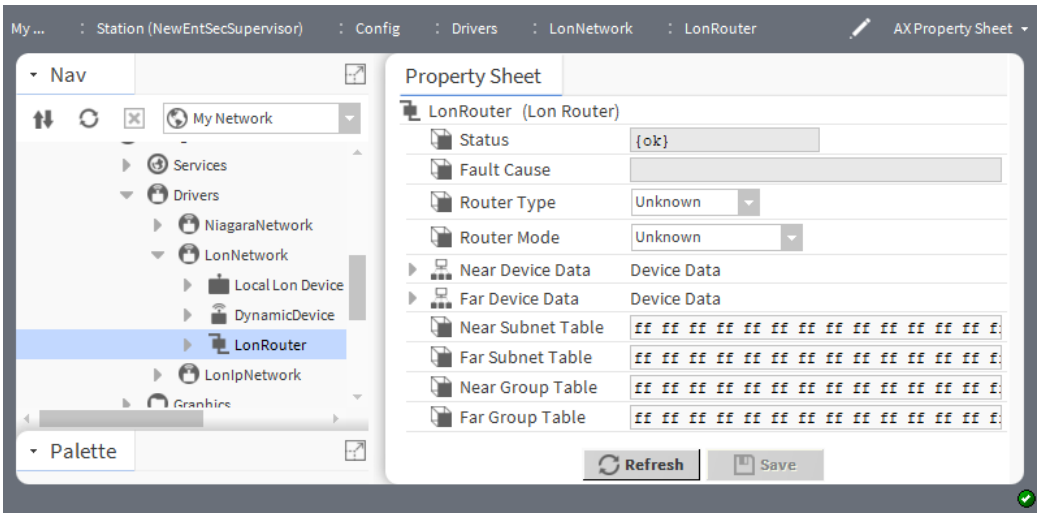
This component is a frozen container slot under any Lon device that uses Extended Network Management, as defined in CEA-709.1-B (replacing the more typical **`DeviceData`** slot). Like DeviceData, it represents the device's state and configuration data, including the larger number of possible address tables in its child **ExtAddressTable** slot (unique to ExtDeviceData).

lonworks-LonRouter

This component represents a Lonworks network router with two **Device Data** container slots: **Near Device Data** and **Far Device Data**. These containers represent both of the router's neuron chips (relative to the Local Lon Device).

You create and manage Lon routers from the **Lon Router Manager** view of the Lon network. The **`LonRouter`** is also available in the **`lonworks`** palette.

Figure 34 Lon Router properties



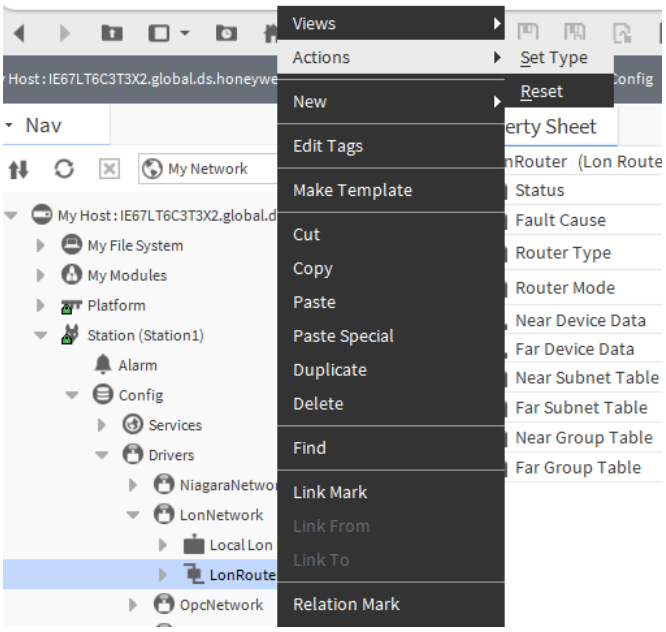
To view these properties, expand **Config→Drivers→LonNetwork** and double-click **LonRouter**.

In addition to the standard properties (Status and Fault Cause), these properties are unique to this component.

Property	Value	Description
Router Type	drop-down list (defaults to `Unknown`)	Selects the way the router handles messages. `Configured` determines which packets to forward based on internal routing tables. `Learning` forwards messages that meet forwarding rules. `Bridge` forwards all messages received regardless of the destination. `Repeater` forwards all messages in both directions regardless of the message's destination or domain. `Unknown`
Router Mode	drop-down list (defaults to `Unknown`)	Selects the mode of the router. `Normal` returns the router from the Temporary Bridge mode. `Init Router Table` copies all forwarding tables from EEPROM to the RAM tables for a configured router or sets all RAM tables to flood for a learning router (same action as node reset). `Temporary Bridge` temporarily forwards all messages to the domain until the next reset or return to Normal. `Unknown`
Near Device Data	additional property [Lon Router, Ip Channel]	Configure the devices in the device data container slot connected to the network's neuron chips that are near relative to the **Local Lon Device**. For information about these properties, refer to *lonworks-DeviceData*.
Far Device Data	additional property [Lon Router, Ip Channel]	Configure the devices in the device data container slot connected to the network's neuron chips that are near relative to the **Local Lon Device**. For information about these properties, refer to *lonworks-DeviceData*.
Near Subnet Table	hexadecimal number	EEPROM or RAM table for the specified domain and single router at its far side.
Far Subnet Table	string	Configures the EEPROM or RAM table for the specified domain and single router at its far side.
Near Group Table	hexadecimal number	EEPROM or RAM table for the specified domain and single router at its far side.
Far Group Table	string	Configures the EEPROM or RAM table for the specified domain and single router at its far side.

Actions

Figure 35 Lon Router actions menu



To access the **Action** menu, expand**LonNetwork**, right click**LonRouter** and click **Actions**.

Action	Description
Set Type	Set the enumerated descriptor for the router type.(Configured, Learning, Bridge, and Repeater).
Reset	Issues a reset command to the device.

lonworks-RouterEntryTable

This component is a table in a Lonworks router.

lonworks-LonDeviceFolder

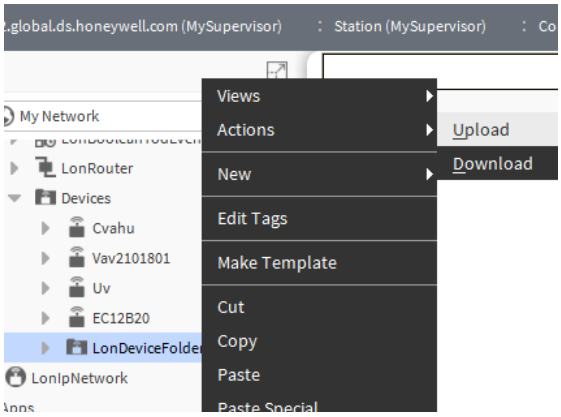
This component is the Lon implementation of a folder under a **LonNetwork**. You use these folders to organize Lon devices under the network.

You add such folders using the **New Folder** button in the **Lon Device Manager** view of the **LonNetwork**. The **LonDeviceFolder** is also available in the **`lonworks`** palette.

Each **LonDeviceFolder** has its own **Lon Device Manager** view.

Actions

Figure 36 Lon Device Folder actions



To access the **Action** menu, expand**LonNetwork→Devices** and right-click**LonDeviceFolder** and click **Actions**.

Action	Description
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.

lonworks-LonDevice

This component is the base class that represents a physical Lonworks device. It adds polling, lonLinks and file access for the support of configuration properties. Most Lon devices are represented by a dynamic device, a component that is subclassed from the Lon device.

lonworks-ConfigParameter

This component represents a single config property (CP) in a Lon device. It provides specific support for run-time updates. A node's CPs (if any) appear in the **Property Sheet** of the **LonDevice**.

lonworks-LonData

This component is the superclass for all classes which can be used as the data component of Lon components. These classes can convert their data from and to the byte format used by physical Lonworks devices. This component contains one or more LonPrimitives.

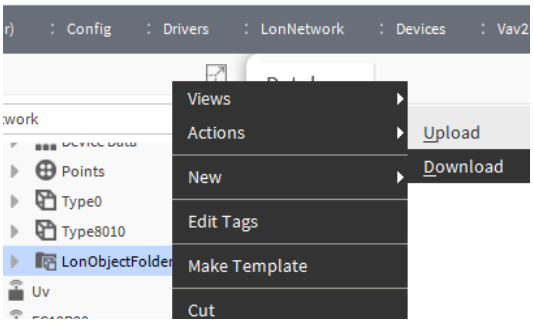
lonworks-LonObjectFolder

This component is a folder organizes Lon objects when adding (learning) new Lon devices. This folder is also used under a programmable Lon device, such as an XL15C.

To organize Lon objects, you copy this folder from the **`lonworks`** palette and place it under the device node in the Nav tree.

Actions

Figure 37 Lon Object Folder actions



To access the **Action** menu, expand**LonNetwork→Dynamic Device** and right-click**LonObjectFolder** and click **Actions**.

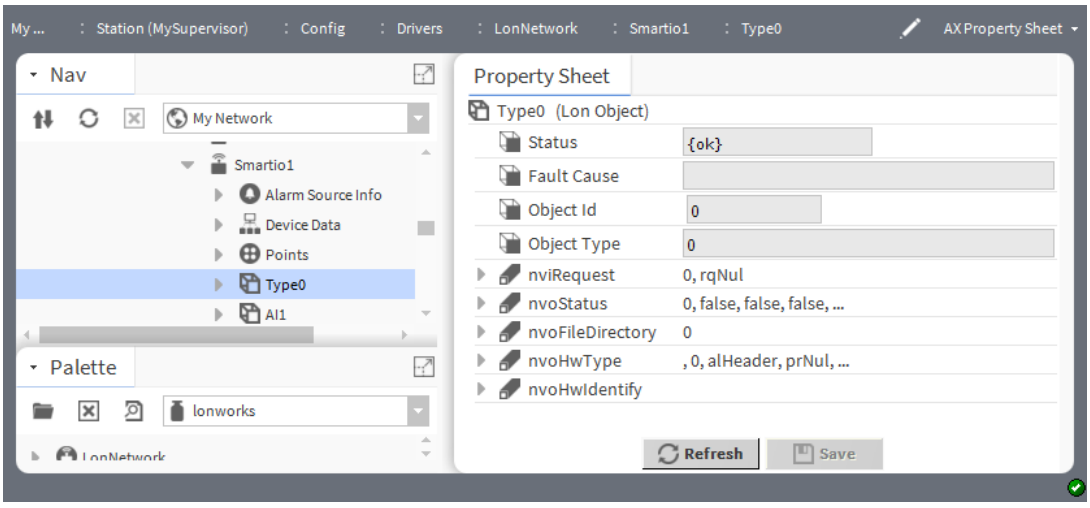
Action	Description
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.

lonworks-LonObject

This component is a container for grouping Lon components in a LonMark compliant device. Each Lon object represents a LonMark object, where its child Lon components represent the set of nvs, ncis and cps in that LonMark object.

LonObjects are visible in the Nav tree when you expand the Lon device. This extra level of component hierarchy may be useful in devices that have a large number of LonMark objects. The default name for a LonObject is `Typen`, where `n` is the LonMark object type number.

Figure 38 Example of LonObject properties



You configure Lon objects in the network's **Lon Netmgmt** container under the **LonNetwork**, which applies to any Quik Learn or add device operation. The **`Use Lon Objects`** property is available selectively, such as when issuing an `ImportXml` command or `Learn Nv` on a **DynamicDevice**.

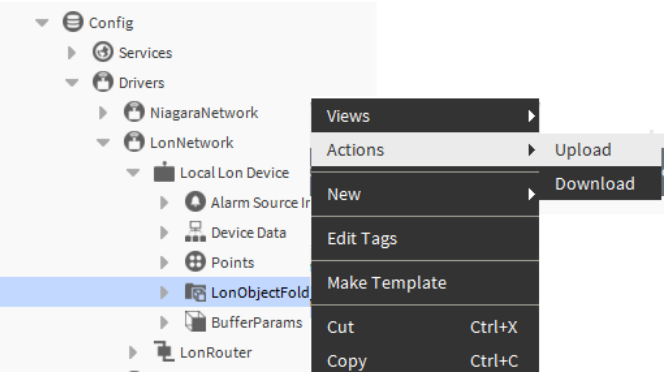
In addition to the standard properties (Status and Fault Cause), two properties may be common to Lon objects. Other properties depend on the device.

Property	Value	Description
Object Id	number from 0–65535 (defaults to zero (0))	Identifies the LonMark object for the LonObject. It must be unique in the device among all objects.
Object Type	number	Identifies a numerical LonMark object type for the LonObject, for example: 0 for the Node object and 8080 for a Unit Ventilator Controller functional profile.

Actions

To access the **Actions** menu access expand**LonNetwork→Local Lon Device→ Lon Object Folder** right click **Lon Object**.

Figure 39 Local object actions menu



Action	Description
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.

lonworks-LonPointFolder

This component is the Lonworks implementation of a folder under a Lon device's Points container (**LonPointDeviceExt**). You add such folders using the **New Folder** button in the **Lon Point Manager** view of the **Points** component.

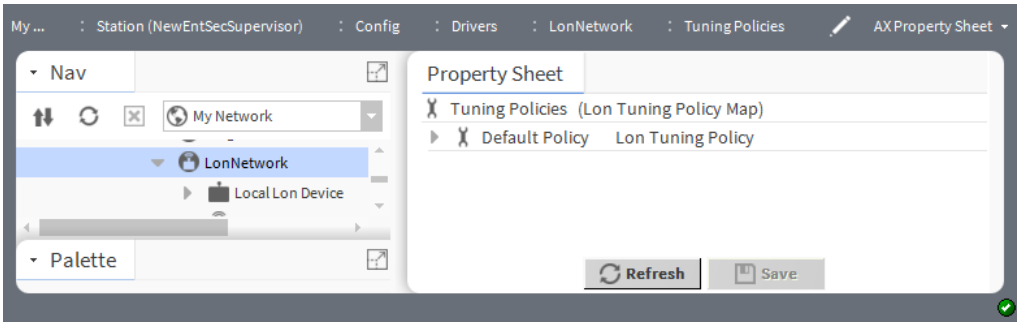
Each **LonPointFolder** has its own **Lon Point Manager** view. The **LonPointFolder** is also available in the **`lonworks`** palette.

lonworks-LonTuningPolicyMap

This component is the Lon network container slot for one or more Lon tuning policies.

This component is a sub-component under the **LonNetwork**.

Figure 40 LonTuningPolicyMap container



To open this view, right-click **LonNetwork**, click **Views→AX Property Sheet** and click **Tuning Policies**.

This view contains no configurable properties.

lonworks-LonTuningPolicy

This component provides a tuning policy for the Lon network, with standard framework tuning policy properties.

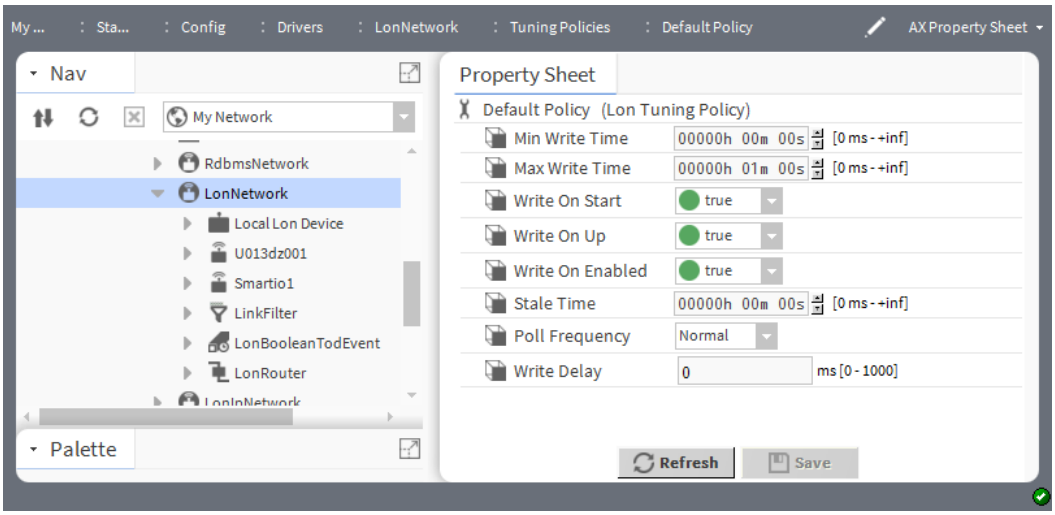
Tuning policy effects on Lon component behavior

The following table indicates the way the network tuning policy affects Lon component behavior.

Table 3 Effects of tuning policy on Lon component behavior

Lon component	Behavior
nci or cp	No effect
nvo	No effect
nvi with proxy (unbound)	Min/max times and start/on/enable flags apply
nvi with proxy (bound)	Min/max times and start/on/enable flags apply
nvi linked to nvo (unbound)	Min/max times apply to update
nvi linked to nvo (bound)	No effect

Figure 41 Tuning policy properties



To view these properties, expand **Config→Drivers**, right-click **LonNetwork**, click **Views→AX Property Sheet** and expand **Tuning Policies** and click **Default Policy**.

Tuning policy properties are defined in the *Niagara Drivers Guide*.

Lon-specific information about Max Write Time

If you link two devices, for example LON1 and LON2, the LON2 device does not receive updates per **`Max Write Time`**. However, when the LON1 value changes, updates go out. The updated value does not depend on **`Max Write Time`** of LON2 device. These devices are linked between each other but not through proxy points.

NOTE: The **`Max Write Time`** is observed when the devices are linked through proxy points.
The LON Device accepts the default values if it does not receive any updates within five minutes of linking. The updates are based on its receive heartbeat setting which is pre-programmed.

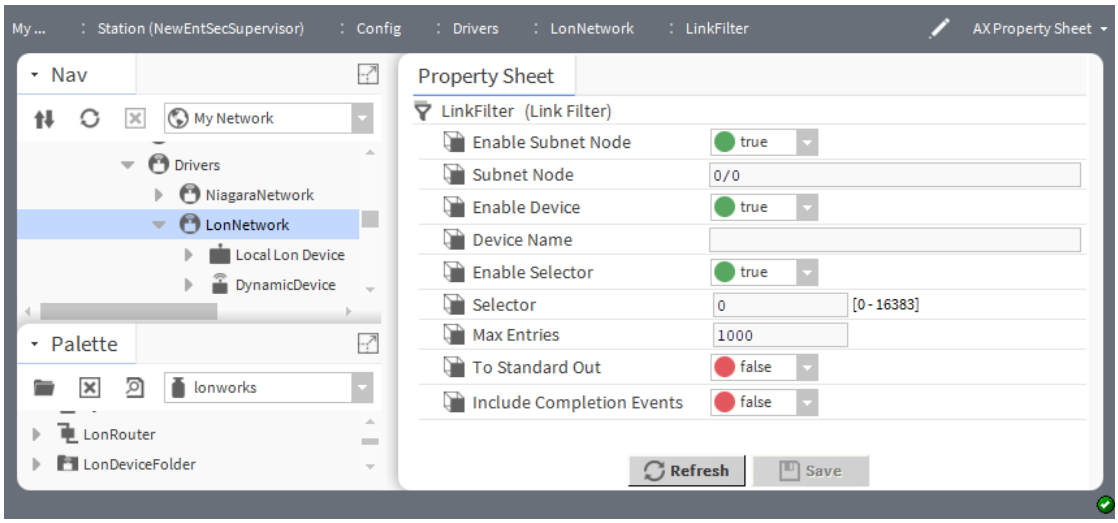
The **`MAX Write Time`** is one minute, which is sufficient enough to update before the received heartbeat expires.

lonworks-LinkFilter

This component is for troubleshooting. Its default **Link Filter** view provides debug-level messages related to Lonworks links.

To use this component you drag or copy it from the **`lonworks`** palette to a Lon network node in the Nav tree.

Figure 42 LinkFilter properties



To access this view, right-click **LonNetwork** and click **Views→AX Property Sheet**, then expand **Link Filter**.

Property	Value	Description
Enable Subnet Node	`true` or `false` (default)	Turns the subnet node on and off. `true` applies the filter to all but the device identified by its **`Subnet Node`**.
Subnet Node	`s/n` syntax	Identifies the node to exclude, where `s` defines the subnet, and `n` defines the node. For example, Subnet Node = `1/5`
Enable Device	`true` or `false` (default)	Turns the device on and off.

Property	Value	Description
		`true` applies the filter to all but the device named in the **`Device Name`** property.
Device Name	string	Identifies the host's `LONn` port used in a Lon network. For example, Device Name = `Mnlrr2`. For a **LonTunnel** copied from the palette, this defaults to `LON1`. Defice names must match the name string in the **Lon-Network Lon Comm Config** container.
Enable Selector	`true` or `false` (default)	Turns use of the selector on and off. `true` applies the filter to all but the device(s) specified by the **`Selector`** property.
Selector	number between 0 and 16383]	Identifies a selection to exclude. The **Lon Link Manager** uses decimal notation for selector numbers, but a Lon device's **Nv Manager** lists selectors for nvs using hexadecimal numbers. For example: Selector = `16`.
Max Entries	number (defaults to 1,000)	Defines the number of records to display in the **Link Filter View**. When collection reaches this number it stops.
To Standard Out	`true` or `false` (default)	Enables and disables the printing of Boolean properties to the station. `true` prints the results of Boolean properties to the station.
Include Completion Events	`true` or `false` (default)	Enables and disables the inclusion of completion events. `true` identifies link events, which require notification upon completion.

Action

A `Clear Table` action removes collected records. This enables continued collection if the buffer previously reached the **`Max Entries`** limit. You can drag or copy multiple **`LinkFilter`** components to the Lon network node.

lonworks-LonTimeZone

This component represents SNVT_Time_Zone.

lonworks-LonWbService

This component provides access to a Lonworks network for client-side applications.

lonworks-LonAppDownloadJob

This component loads a new application image to a dev table.

lonworks-LonBindJob

This component command the binding of links between Lonworks devices.

lonworks-LonChangeNvTypeJob

This component provides feedback on a change nv operation.

lonworks-LonCommissioJob

This component commands that a specific node be commissioned. It configures the domain table and initializes the address and nv config tables.

lonworks-LonCommissionRouterJob

This component commands that the addNode service be performed on the specified node. This service configures the domain table and initializes the address and nv config tables.

lonworks-LonDiscoverJob

This component queries the connected network for undiscovered devices.

lonworks-LonLearnJob

This component commands the learnNode service to be performed on the specified node. This service reads the domain table, address table and nv config table. It makes no selections, but assumes the requirement is to learn a previously-managed network.

The learn job performs these steps:

1. If the job finds a duplicate subnet/node it aborts the process.
2. Matches discovered devices to database devices with a zero Neuron Id.
3. Adds dynamic devices, if there is an available Lon Xml file to import for the device. Otherwise, it learns the nvs from device.
4. For unmanaged devices it performs the same functions as when no devices are selected. It does not check for duplicate nodes.

lonworks-LonLearnLinksJob

This component commands the learnNode service to be performed on the specified node. This service reads the domain table, address table and nv config table.

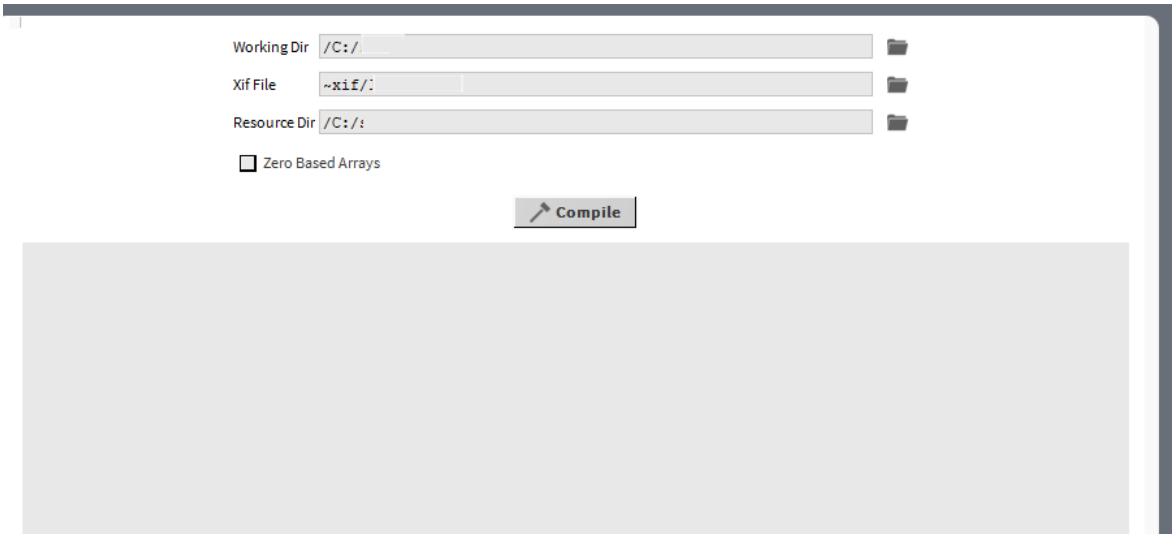
lonworks-LonReplaceJob

This component commands that a specific node be replaced. This sets the domain table and address and nv config tables to match the configuration of the original device.

lonworks-LonXmlTool

This component is a Workbench utility used to make Lon Xml (extension `.lnml`) files from .xif files for a Lon device.

Figure 43 Lon Xml Tool properties



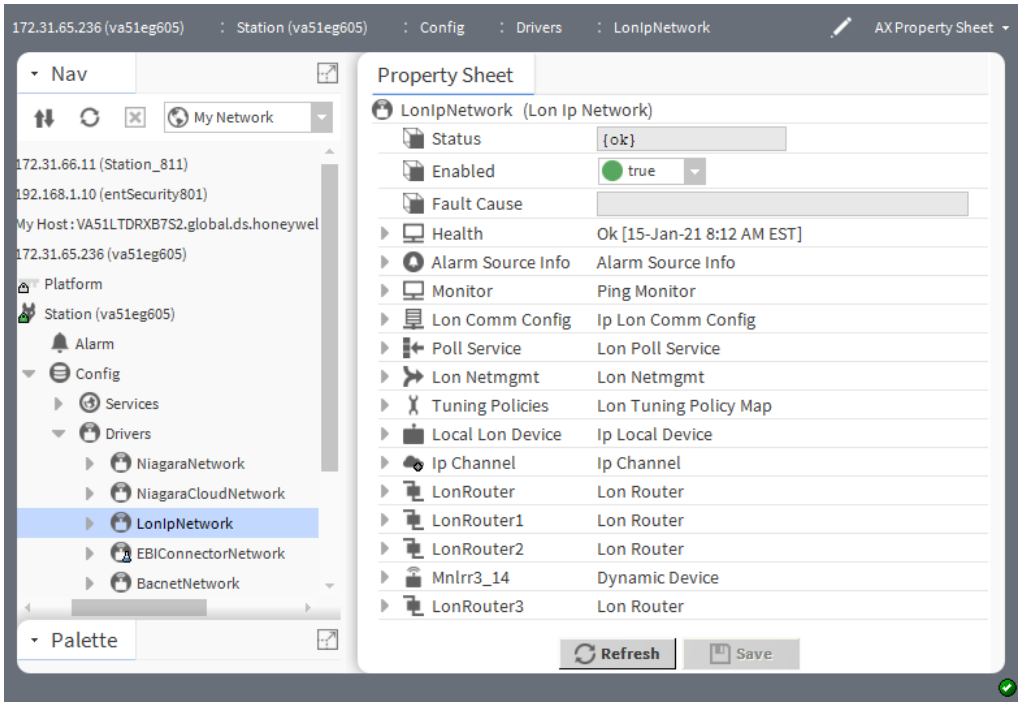
You access this view by clicking the menu item **Tools→Lon Xml Tool**.

Property	Value	Description
Working Dir	file path	Defines where to store the Lon Xml file.
Xml File	file path	Identifies the Lon Xml file.
Resource Dir	file path	Defines the path to any additional resource files from the manufacturer. The tool uses this information to configure nvs in the Lon Xml file.
Zero Based Arrays	check box, defaults to unchecked	Determines the array index to append to the array name when assigning a name to a nvs, ncis, and cps. The default (empty check box) appends a one-based index. Enabling the check box appends a zero-based index.

lonIp-LonIpNetwork

This component is the base container for all Lon IP components in the station.

Figure 44 LonIPNetwork Property Sheet



To access this view, drag the **LonIPNetwork** component from the palette to the Nav tree, and right ckick the network and click **Views→Property Sheet**.

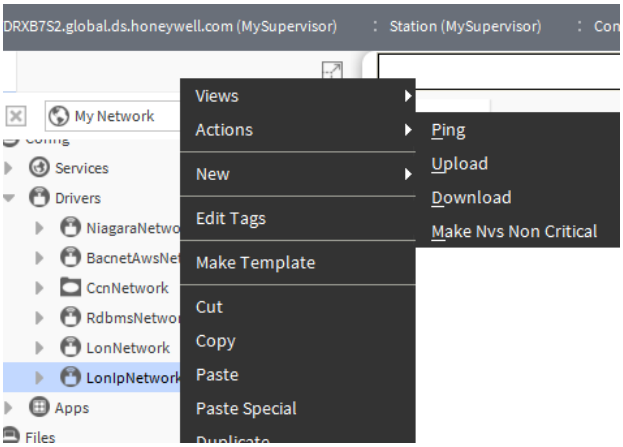
In addition to the standard properties (Status, Enabled, Fault Cause, Health, Alarm Source Info, Monitor, Poll Service and Tuning Policies), these properties are unique to Lonworks.

Property	Value	Description
Lon Comm Config	additional properties	Configures the communication stack for a single Lonworks connection including specifying the port's **`Device Name`** For specific properties, refer to a separate topic in this guide.
Lon Netmgmt	additional properties	Serves as the container for Lon network management functions provided by the station. For specific properties, refer to a separate topic in this guide.
Local Lon Device	additional properties	Configures the local interface to the Lonworks fieldbus. For specific properties, refer to a separate topic in this guide.
IP Channel	additional properties	Configures network and server properties. For specific properties, refer to separate topics in this guide. Double-clicking this slot opens the **Member Manager**.

Actions

To access the **Action** menu, right click **LonIPNetwork** in the nav tree and click **Actions**.

Figure 45 LonIpNetwork actions



Action	Description
Ping	Attempts communication with the device. If successful, the device status reports `{ok}`. If this fails, the system sets device status to `{down}`
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.
Make Nvs Non Critical	Changes the network variable.

lonIp-IpChannel

This component is a child of the **LonIpNetwork**. It contains two frozen container slots: **Network Config** and **MemberTable**.

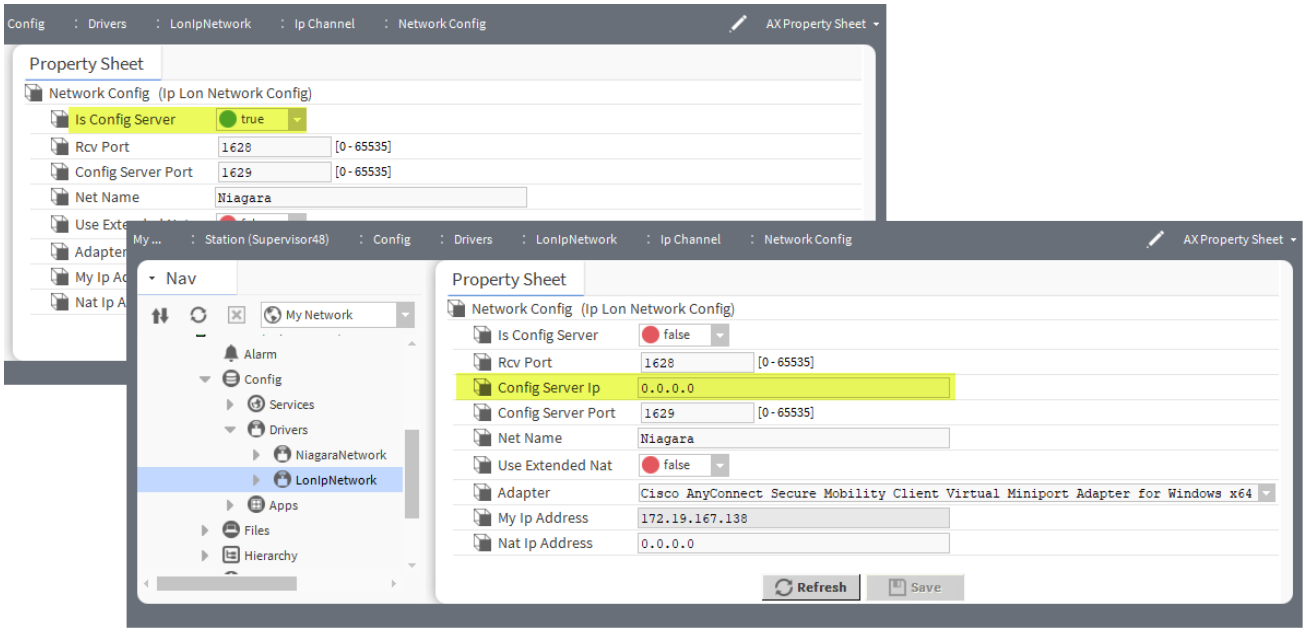
You drag or copy this component from the **`lonIp`** palette to the **LonIpNetwork**. Its default view is the **Member Manager** view. This component has no properties of its own.

lonIp-IpLonNetworkConfig

This component is one of two frozen containers under an **Ip Channel** component of a **LonIpNetwork**. It contains properties that specify the CEA-852 IP Configuration Server (Config Server) and the software ports used.

NOTE: The **`Config Server`** property defaults to `false`. To configure the controller station to function as a configuration server, set this property to `true`.

Figure 46 Ip Lon Network Config properties



The screen capture shows two versions of this **AX Property Sheet**. When **`Is Config Server`** is `true`, there is no need to define the **`Config Server Ip`**, which is only available when **`Is Config Server`** is set to `false`.

To access this view expand, **LonIPNetwork→IP Channel** and double-click **Network Config**.

Property	Value	Description
Is Config Server	`true` (default) or `false`	Determines how to populate the Ip Channel table after you add a device to the Config Server and the station contacts the server (the station becomes a member of the CEA-852 channel). `false` dynamically populates this table with the other node members of the channel. `true` uses the **Member Manager** view on the parent **Ip Channel** component for you to manually add a channel member for each Lon/IP router device, specifying the IP address of the controller.
Rcv Port	number between 0 and 65535 (defaults to 1628)	Configures the port to receive data.
Config Server Ip (appears when **`Is Config Server`** is `false`)	IP address	Defines the IP address of the third-party Config Server.
Config Server Port	number between 0 and 65535 (defaults to 1629)	Identifies the port used by the server.
Net Name	text	Configures the network name.
Use Extended Nat	`true` or `false` (default)	Enables NAT (Network Address Translation).

Property	Value	Description
Adapter	drop-down list	Provides the name for the network adapter.
My Ip Address	read-only	Reports the IP address of the platform.
Nat Ip Address	IP address (defaults to 0.0.0.0)	Defines the device's assigned external (Internet) address if the device is behind a NAT (Network Address Translation) router. If not, leave this property at its default value.

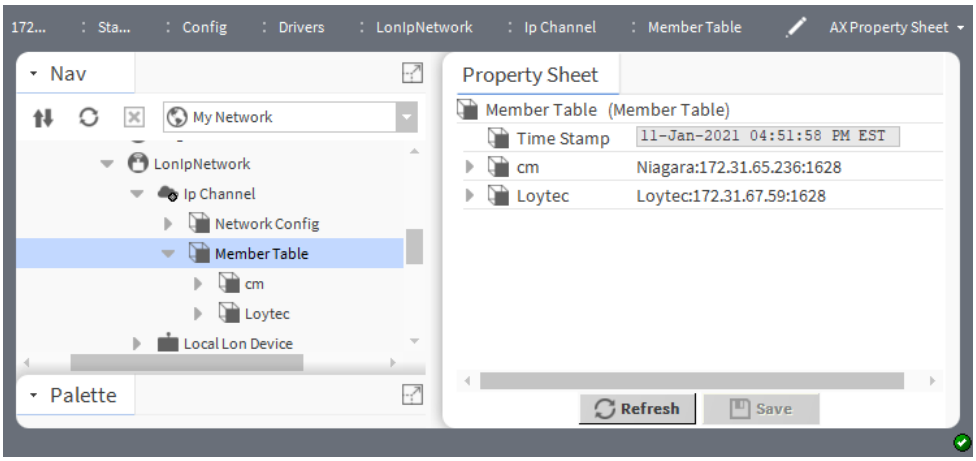
lonIp-MemberTable

This component is a feature of the LonIP driver and a child of the **Ip Channel** component under the **LonIp-Network**. It lists the channels under the network.

Depending on how the network is configured, the table is populated in one of two ways:

- The **Ip Channel** has a **`Is Config Server`** property, which defaults to `false`. After you add a device to the Config Server, and contact is made with the Config Server (the station becomes a member of the CEA-852 channel), the driver dynamically populates this table with the other node members of the channel.
- If the **Ip Channel** property **`Is Config Server`** is set to `true`: you use the (default) **Member Manager** view on the parent **Ip Channel** component to add a new channel member, specifying the IP address of the controller. You also add a channel member for each Lon/IP router device.

Figure 47 Member table Property Sheet



To access this view expand, **LonIPNetwork→IP Channel**, and double-click and click **Member Table**.

This **Property Sheet** contains one container.

Property	Value	Description
Time Stamp	read-only	Reports when the event occurred.
Channel Member	additional properties	Lists the channels. Channel properties are documented in a separate topic.

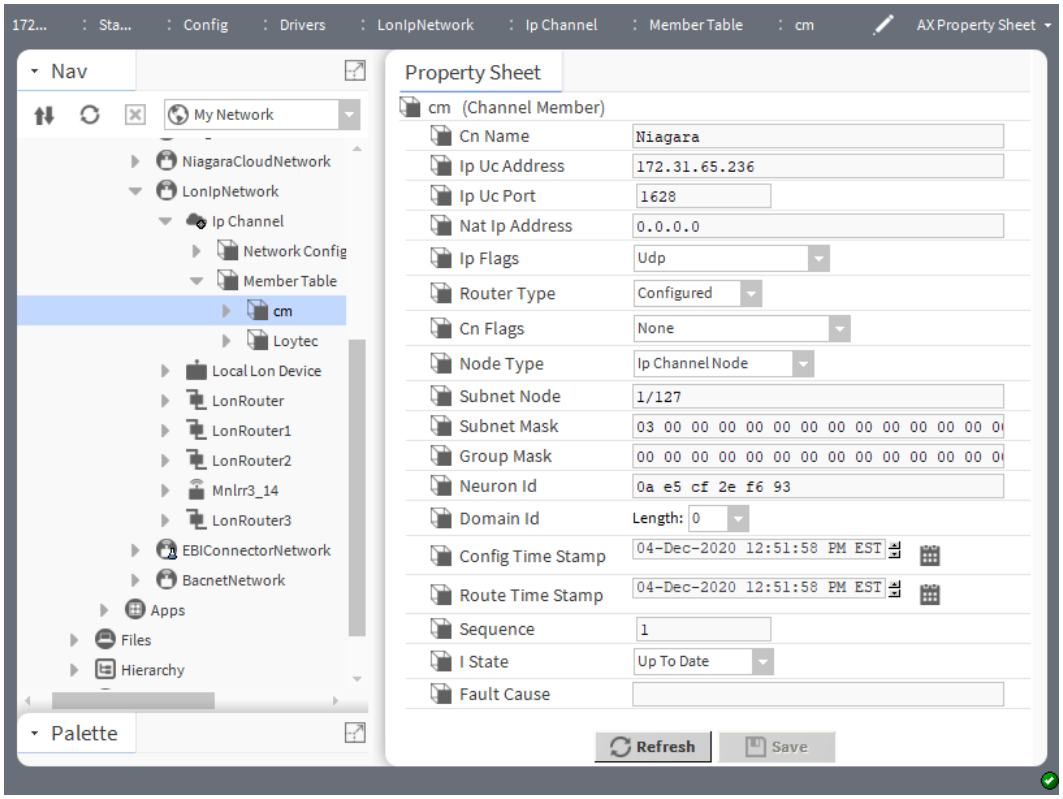
lonIp-ChannelMember

This component is a child of the **MemberTable** container under the **IP Channel** component of a **LonIpNetwork**. A ChannelMember represents a Lon/IP router or an IP channel node.

ChannelMember components represent Lon/IP 852 routers, as well as one required entry for the host station itself (as a channel node). If using a third-party (non-framework) Config Server, the driver automatically adds all ChannelMembers once communications to the CS have been established.

If configuring the station as the CEA-852 Config Server, you must manually add ChannelMembers. Once added, you can monitor these component's status in the **Member Manager** view, and if necessary, also edit and open a browser connection to any selected one (for possible configuration, if the device provides a built-in web server). ChannelMembers also provide a number of right-click actions.

Figure 48 Ip Channel Member properties



To access this view expand, **LonIPNetwork→IP Channel**, and double-click and click **Member Table→ChannelMember**.

In addition to the standard property (Fault Cause), these properties are unique to this component.

Property	Value	Description
Cn Name	name	Identifies the channel by name.
Ip Uc Address	IP address (defaults to 0.0.0.0)	Configures the IP UC (united communications) address. UC is a set of products that provide a consistent, unified user interface and experience across multiple devices and media types. (Wikipedia)
Ip Uc Port	number	Configures the IP UC port number.
Nat Ip Address	Ip address (defaults to 0.0.0.0)	Defines the device's assigned external (Internet) address if the device is behind a NAT (Network Address Translation) router. If not, leave this property at its default value.
Ip Flags	drop-down list	Selects the type of flag.

Property	Value	Description
		`Udp` (User Datagram Protocol) `Tcp` (Transmission Control Protocol) `Udp_Tcp` chooses both. `Multi Cast` addresses communication to a group. `Udp_Multi Cast` combines the two types. `Tcp_Multi Cast` combines the two types. `Udp_Tcp_Multi Cast` combines the two types.
Router Type	drop-down list (defaults to `Unknown`)	Selects the way the router handles messages. `Configured` determines which packets to forward based on internal routing tables. `Learning` forwards messages that meet forwarding rules. `Bridge` forwards all messages received regardless of the destination. `Repeater` forwards all messages in both directions regardless of the message's destination or domain. `Unknown`

Cn Flags	drop-down list	Configures when to include the enumerated descriptor for the channel flag:
		`None`
		`All Broad Casts`
		`Security`
		`All Broad Casts_Security`

Node Type	drop-down list	Identifies the type of node:
		`Unintialized`
		`Non Ip To Ip Router`
		`Ip Channel Node`
		`Ip channel Proxy`
		`Ip To Ip Router`

Subnet Node	subnet (number) and node (number)	Reports the assigned Lonworks subnet/node address. Node must be unique for this device on the channel.
Subnet Mask	32–bit number	Defines a bitmask that when applied by a bitwise AND operation to any IP address in the network, yields the routing prefix. Subnet masks are also expressed in dot-decimal notation like an IP address. For example, 255.255.255.0 is the subnet mask for the prefix 198.51.100.0/24. (Wikipedia)
Group Mask	32–bit number	Identifies a group of devices.
Neuron Id	hexadecimal number	Maps the device in the station database to a different physical device (or no device, if the address is not available). While

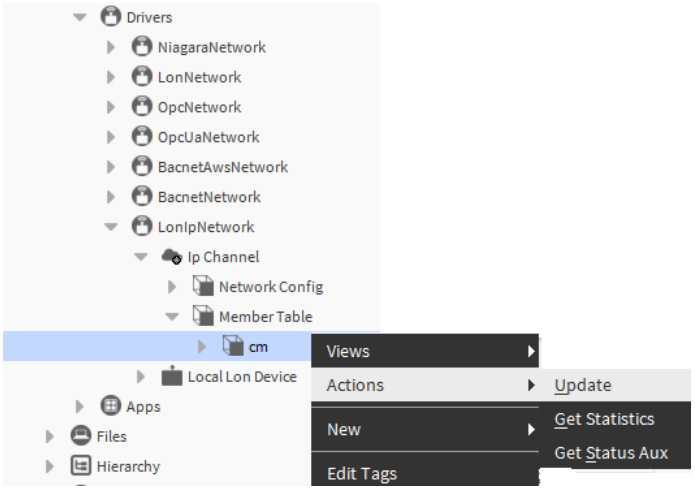
Property	Value	Description
		most **DeviceData** properties should be left at default values, this is one property you may routinely edit.
Domain Id	text	Defines a Lonworks working domain.
Config Time Stamp	date and time	Configures the contents of system time stamps.
Route Time Stamp	date and time	Configures a separate route time stamp.
Sequence	number (defaults to one (1))	Defines the position in a routing sequence.

I State	drop-down list	Selects the stage in the process of setting up the member.
		`New Member`
		`Sent D R Req`
		`Sent D R`
		`Sent C R Req`
		`Up To Date`
		`Error`

Actions

To access **Actions** menu expand, **LonIPNetwork→IP Channel→Member Table**, right click **ChannelMember**.

Figure 49 Channel Member actions



Action	Description
Update	Forces an update of LonIp channel information to the device. An update occurs automatically upon station startup, and when LonIp channel information changes.
Get Statistics	Opens a popup window with various statistics for the selected channel member.
Get Status Aux	Opens the **Get Statu Aux** window for the selected channel member.

lonIp-IpLocalDevice

This component is similar to the **LocalLonDevice** component under a regular **LonNetwork**, and has the same default name. It is a child of the **LonIpNetwork** component as copied from the **lonIp** palette.

You use this component to specify the station's channel and subnet/node address (it must be consistent with the IP Lon channel that the station is to become a member of). The station presents itself as an IP channel node using this address.

Ip Local Lon Device properties and actions are the same as those for the Local Lon Device.

tunnel-TunnelService

This component provides a station server for application tunneling, where remote PCs with a tunnel client installed can use a legacy or vendor-specific PC application to access devices connected to one or more driver networks. A tunnel connection allows the remote client application to operate as it were directly attached to the driver network (via a virtual PC port).

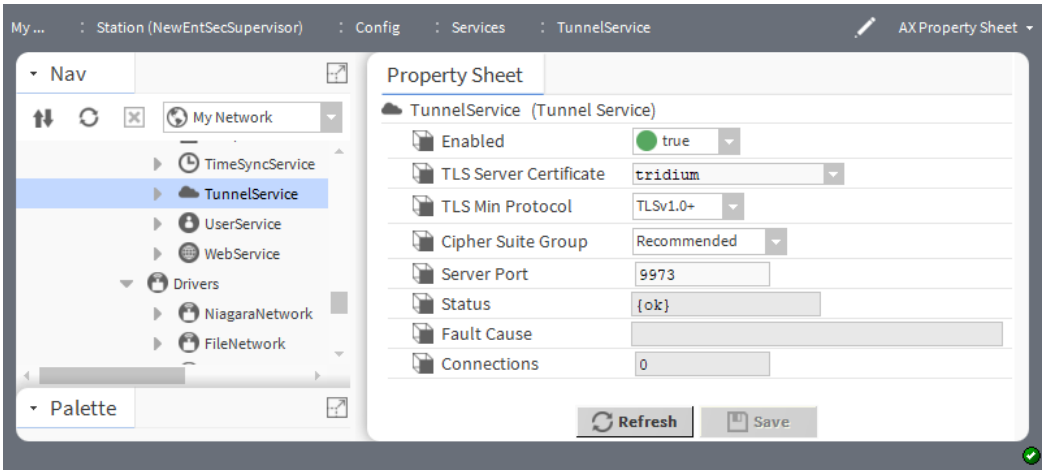
WARNING: While this service is available, you are strongly encouraged not to provide tunneling access to Lonworks devices. Tunneling is not a secure form of communication. If you use it, you provide an opening for malicious activity within your network.

A client PC tunnels using an IP (LAN/WAN) connection, which is granted only after authentication as a station user (with admin write permissions for the particular child tunnel component accessed).

The **LonTunnel** child component provides support for tunneling Windows Lon-based applications. Other serial-based drivers may support a **SerialTunnel** child component.

In any station only one **TunnelService** is recommended. It can hold the required number (and types) of child tunnels.

Figure 50 TunnelService properties



To view these properties, expand **Config→Services** and double-click **TunnelService**.

In addition to the standard properties (Enabled, Status and Fault Cause), these properties are unique to this component.

Property	Value	Description
TLS Server Certificate	tridium (default),	Defines the server certificate from the **User Key Store**. **NOTE:** This certificate should be valid and signed.
TLS Min Protocol	TLSv1.0+ (default), TLS 1.1+, TLS 1.2	Selects the minimum accepted TLS (Transport Layer Security) version.

Property	Value	Description
Cipher Suite Group	drop-down list	Controls which cipher suites can be used during TLS negotiation. `recommended` is more secure than `supported`. Use it unless it causes compatibility issues with the client.
Server Port	read-only	Reports the port used to monitor incoming client tunnel connections.
Connections	read-only	Shows the number of active tunnel connections ranging from 0 (no active connection) to the number of child tunnel components. **WARNING:** While tunneling is available, you are strongly encouraged not to use it. Tunneling is not a secure form of communication.

lontunnel-LonTunnel

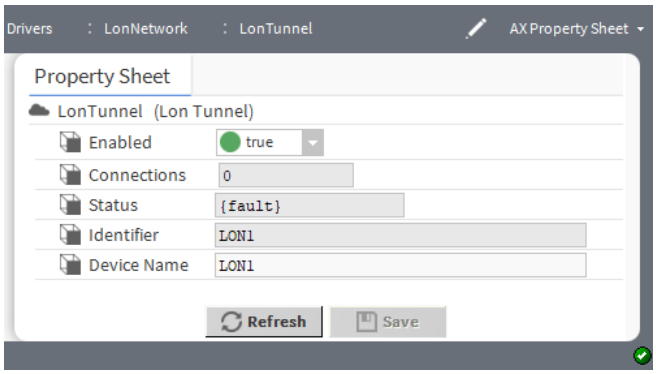
This is the server-side component required to support application tunneling of Windows Lon-based applications to Lon devices reachable in a station's Lon network. It is found in the **`lontunnel`** module. A station running the Lonworks driver can provide tunneling access to its connected Lonworks devices. This uses a vendor's Windows Lon-based application (via the Lon tunnel client) to perform device-specific operations. Examples include application downloads or other device configuration.

WARNING: While this module is available, you are strongly encouraged not to provide tunneling access to Lonworks devices. Tunneling is not a secure form of communication. If you use it, you provide an opening for malicious activity within your network.

The tunneling client is separate from Workbench—meaning that you can install it on various PCs. The key advantage is that Lon tunneling requires only a standard IP connection (to the station), yet provides access as if the client PC was attached to the target Lonworks network via a physical FTT-10 adapter.

NOTE: No special licensing is required to use tunneling features.

Figure 51 LonTunnel properties



Unless you have multiple Lon tunnels (one for each Lon port on the host), you do not need to do any further configuration, apart from defaults.

To view these properties, expand **LonNetwork** and double-click **LonTunnel**.

In addition to the standard properties (Enabled and Status) these properties are unique to this component.

Property	Value	Description
Connections	read-only	Shows the number of active tunnel connections ranging from 0 (no active connection) to the number of child tunnel components. **WARNING:** While tunneling is available, you are strongly encouraged not to use it. Tunneling is not a secure form of communication.
Identifier	read-only	Reflects the entered **Device Name** (below), used as the **Tunnel Name** when configuring the client-side **Lon Tunneling** window.
Device Name	`LONn`, where `n` changes based on the network	Identifies the device name (identifier) of the host's Lon network to access. This is LON1 in most cases (any host with only a single Lon port). However, if the target host has multiple Lon ports (supports multiple Lon networks) you could enter LON2, for example.

Action

In addition, a **LonTunnel** has an available (right-click) action: `Disconnect All` disconnects any active connection through this **LonTunnel** (maximum of 1), causing removal of the **TunnelConnection** below it. On the remote (Lon tunnel client) side, the driver reports a popup message: `Connection closed by remote host`.

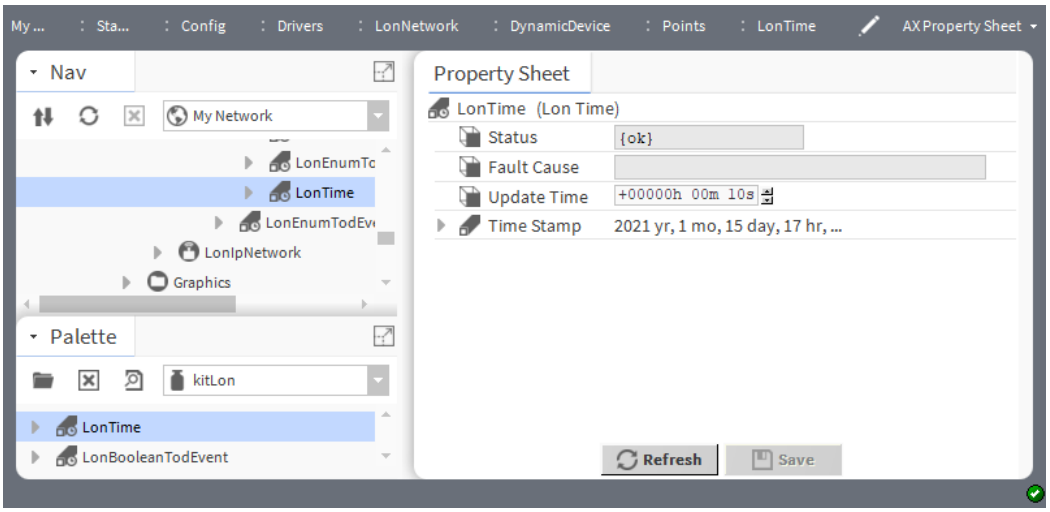
NOTE: Any **TunnelConnection** component also has its own `Disconnect` action, which effectively performs the same function.

kitLon-LonTime

This component provides the ability to link system time from the station to any Lon device that has an nvi using SnvtTimeStamp.

You drag or copy this component from the **kitLon** palette to the Lon network, then link it on the wire sheet to one or more Lon device(s).

Figure 52 Lon Time Property Sheet



To access these properties, right-click **LonNetwork** and click **Views→AX Property Sheet**, then expand **LonTime**.

In addition to the standard properties (Status and Fault Cause) there properties are unique to this component.

Property	Value	Description
Update Time	hours minutes seconds (defaults to 10 seconds)	Sets the frequency with which the driver updates the time.
Time Stamp	additional properties: year, month, day, hour	Configures the contents of system time stamps. Refer to Time Stamp, page 114.

Time Stamp

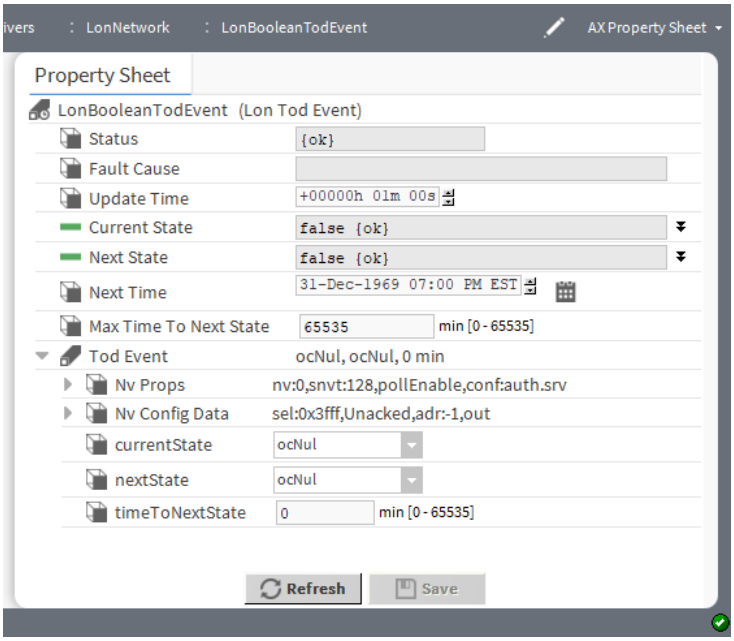
Property	Value	Description
Nv Props	additional properties	Reports and configures NV (Network Variable) input properties. For property descriptions, refer to *lonworks-LocalNv*.
Nv Config Data	additional properties	Maps directly to a device's network variable and reports NV (Network Variable) output configuration data. For property descriptions, refer to *lonworks-LocalNv*.
year	defaults to current computer's year	Sets up the year.
month	defaults to current computer's month	Sets up the month.
day	defaults to current computer's day	Sets up the day.
hour	defaults to current computer's hour	Sets up the hour.
minute	defaults to current computer's minute	Sets up the minute.
second	defaults to current computer's second	Sets up the second.

kitLon-LonTodEvent (Lon Tod Event)

This component can take inputs from BooleanSchedule outputs (Out, Next Value), and link them to a Lon device's nvi that uses an SnvtTodEvent.

To use, copy this component from the **`kitLon`** palette into the Lon network, and link it between BooleanSchedules and Lon devices.

Figure 53 Lon Tod Event Property Sheet



To access these properties, right-click **LonNetwork** and click **Views→AX Property Sheet**, then expand **LonBooleanTodEvent**.

In addition to the standard properties (Status and Fault Cause) there properties are unique to this component.

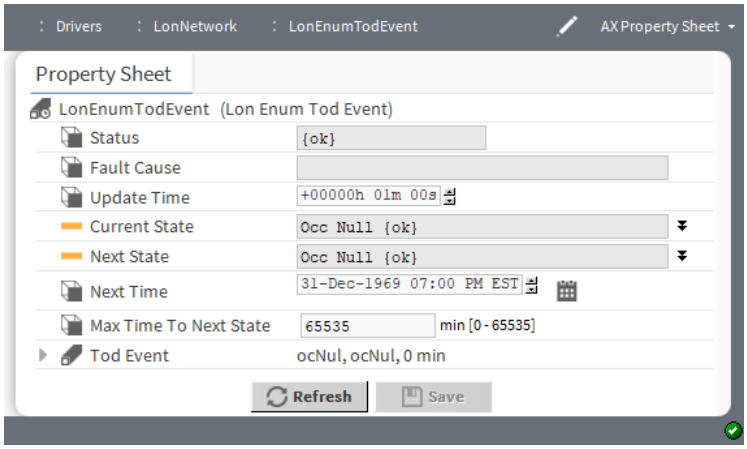
Property	Value	Description
Update Time	hours minutes seconds (defaults to 10 seconds)	Sets the frequency with which the driver updates the time.
Current State	read-only	Reports the current state of the device or component.
Next State	read-only	Reports the next state of the device or component.
Next Time	date	Configures the time of the next event.
Max Time To Next State	date and time with calendar to make selection easy	Specifies the maximum time to wait before rewriting a proxy point value in case nothing else has triggered a write. Any write action resets this timer. If left set to the default, the driver disables this rule resulting in no timed rewrites.
Tod Event	additional properties	Configures the event state. Refer to a separate topic.

kitLon-LonEnumTodEvent (Lon Enum Tod Event)

This component can take inputs from EnumSchedule outputs (Out, Next Value), and link them to a Lon device's nvi that uses SnvtEnumTodEvent.

To use, copy this component from the **kitLon** palette into the **LonNetwork**, and link it between EnumSchedules and Lon devices.

Figure 54 LonEnumTodEvent properties



To access these properties, right-click **LonNetwork** and click **Views→AX Property Sheet**, then expand **LonEnumTodEvent**.

In addition to the standard properties (Status and Fault Cause) these properties are unique to this component.

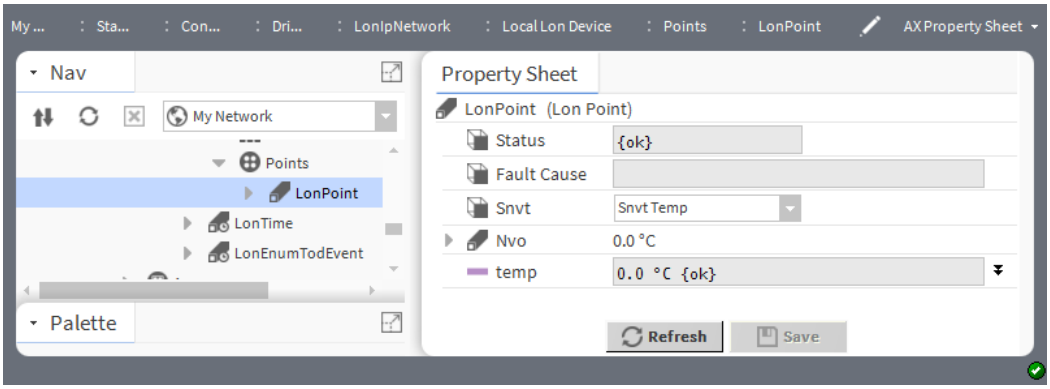
Property	Value	Description
Update Time	hours minutes seconds (defaults to 10 seconds)	Sets the frequency with which the driver updates the time.
Current State	read-only	Reports the current state of the device or component.
Next State	read-only	Reports the next state of the device or component.
Next Time	date	Configures the time of the next event.
Max Time To Next State	date and time with calendar to make selection easy	Specifies the maximum time to wait before rewriting a proxy point value in case nothing else has triggered a write. Any write action resets this timer. If left set to the default, the driver disables this rule resulting in no timed rewrites.
Tod Event	additional properties	Configures the event state. Refer to a separate topic.

kitLon-LonPoint (Lon Point)

This component creates a control object that has an nvo with a selectable SNVT, which can be used to link to an nvi on one or more Lon devices.

You drag or copy objects from the **kitLon** palette into the LonNetwork, for use in special applications.

Figure 55 Example of LonPoint properties



To access these properties, expand **Config→Drivers→LonNetwork→DynamicDevice→Points** and double-click **LonPoint**. The example shows the properties used to configure a SNVT (standard network variable type) of Temp (temperature). Each SNVT (Snvt) provides different properties.

In addition to the standard properties (Status and Fault Cause) this component's unique properties depend on the type of point. The example is a temperature point.

Property	Value	Description
Snvt	drop-down list (defaults to Snvt Temp)	Configures the type of point. Based on this value, many additional properties are available, which are not documented here.
Nvo	container with additional properties	
Nvo, Nv Props	additional properties	Reports and configures NV (Network Variable) input properties. For property descriptions, refer to *lonworks-LocalNv*.
Nvo, Nv Config Data	additional properties	Maps directly to a device's network variable and reports NV (Network Variable) output configuration data. For property descriptions, refer to *lonworks-LocalNv*.

kitLon-LonReplace (Lon Replace)

This component provides a right-click action, which opens a **Replace** window, allowing you to replace a parent device without using Workbench and the **Lon Device Manager** view.

You add this component to a **DynamicDevice** from the **`kitLon`** palette.

Replacement can use the service pin method or the direct entry of the new **`Neuron Id`**. This component may be useful in appliance scenarios, where you access the system primarily using a browser and Px pages.

This component has no properties.

Action

This component provides a single action, **Replace**.

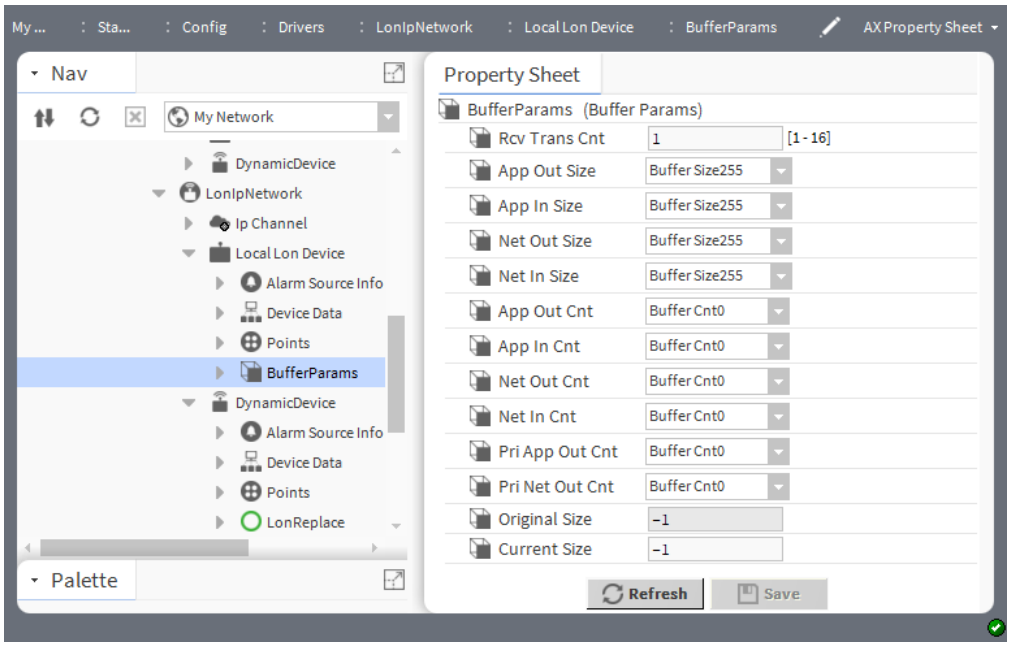
kitLon-BufferParams (Buffer Params)

This component in a **Local Lon Device** container configures the Neuron chip message buffers in the Lon adapter of a controller. It works only as a child of the **Local Lon Device** (the only valid parent). Properties change the **`Application`** and **`Network`** buffer counts and sizes.

CAUTION: This component is only for advanced users who understand the implications of changing these buffer settings. Improper use can degrade operation and/or result in loss of data!

You add this component to the **Local Lon Device** from the **`kitLon`** palette.

Figure 56 Buffer parameter properties



To access this Property Sheet, expand **LonNetwork→Local Lon Device** and double-click **BufferParams**

Property	Value	Description
Rcv Trans Cnt	number from 1–16 (defaults to 1)	Configures the receive transfer count.
App Out Size	drop-down list (defaults to Buffer Size 255)	Configures the size of the buffer for outgoing data.
App In Size	drop-down list (defaults to Buffer Size 255)	Configures the size of the buffer for incoming data.
Net Out Size	drop-down list (defaults to Buffer Size 255)	Configures the out buffer size.
Net In Size	drop-down list (defaults to Buffer Size 255)	Configures the in buffer size.
App Out Cnt	drop-down list (defaults to Buffer Cnt0)	Configures the application out count.

Property	Value	Description
App In Cnt	drop-down list (defaults to Buffer Cnt0)	Configures the application in count
Net Out Cnt	drop-down list (defaults to Buffer Cnt0)	Configures the out buffer count.
Net In Cnt	drop-down list (defaults to Buffer Cnt0)	Configures the in buffer count.
Pri App Out Cnt	drop-down list (defaults to Buffer Cnt0)	Configures the primary application out count.
Pri Net Out Cnt	drop-down list (defaults to Buffer Cnt0)	Configures the primary out buffer count.
Original Size	read-only	Reports the original size of the buffer. A –1 indicates that this component is not configured.
Current Size	read-only	Reports the original size of the buffer. A –1 indicates that this component is not configured.

Actions

To access the **Actions** menu expand **LonNetwork**, **Local Lon Device** and double click **Buffer Params**.

This component provides a single action, **Update Buffers**. This action writes buffer message changes to the Neuron chip.

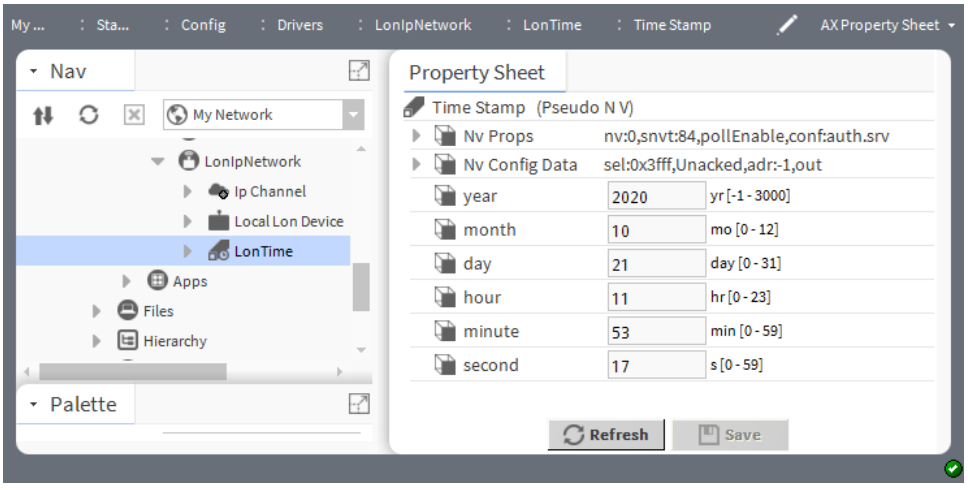
lonworks-PseudoNV

These components take several forms depending on the parent component, which is a kitLon component.

Time Stamp

This component is related to the **`kitLon`** component **LonTime**.

Figure 57 Example of a Pseudo N V for Time Stamp



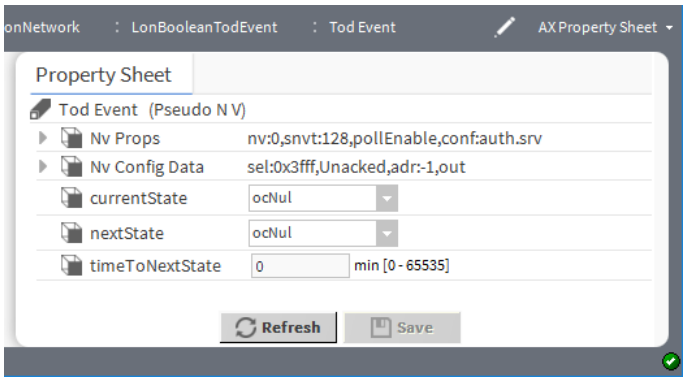
To view these properties, locate double-click a **LonTime** component in the Nav tree and expand **Time Stamp**, or click **TimeStamp**.

Property	Value	Description
Nv Props	additional properties	Reports and configures NV (Network Variable) input properties. For property descriptions, refer to *lonworks-LocalNv*.
Nv Config Data	additional properties	Maps directly to a device's network variable and reports NV (Network Variable) output configuration data. For property descriptions, refer to *lonworks-LocalNv*.
year	defaults to current computer's year	Sets up the year.
month	defaults to current computer's month	Sets up the month.
day	defaults to current computer's day	Sets up the day.
hour	defaults to current computer's hour	Sets up the hour.
minute	defaults to current computer's minute	Sets up the minute.
second	defaults to current computer's second	Sets up the second.

Tod Event

This component is related to two **`kitLon`** components: **LonBooleanTodEvent** and **LonEnumTodEvent**.

Figure 58 Tod Event properties



To access these properties, expand **LonNetwork→LonBooleanTodEvent→Tod Event**.

Property	Value	Description
Nv Props	additional properties	Reports and configures NV (Network Variable) input properties. For property descriptions, refer to *lonworks-LocalNv*.
Nv Config Data	additional properties	Maps directly to a device's network variable and reports NV (Network Variable) output configuration data. For property descriptions, refer to *lonworks-LocalNv*.
currentState	drop-down list (defaults to `ocNul`)	Configures a current state for the event.
nextState	drop-down list (defaults to `ocNul`)	Configures the next state of the device or component.
timeToNextState	minutes (defaults to zero (0))	Specifies the amount of time to wait before rewriting a proxy point value in case nothing else has triggered a write. Any write action resets this timer. If left set to the default, the driver disables this rule resulting in no timed rewrites.

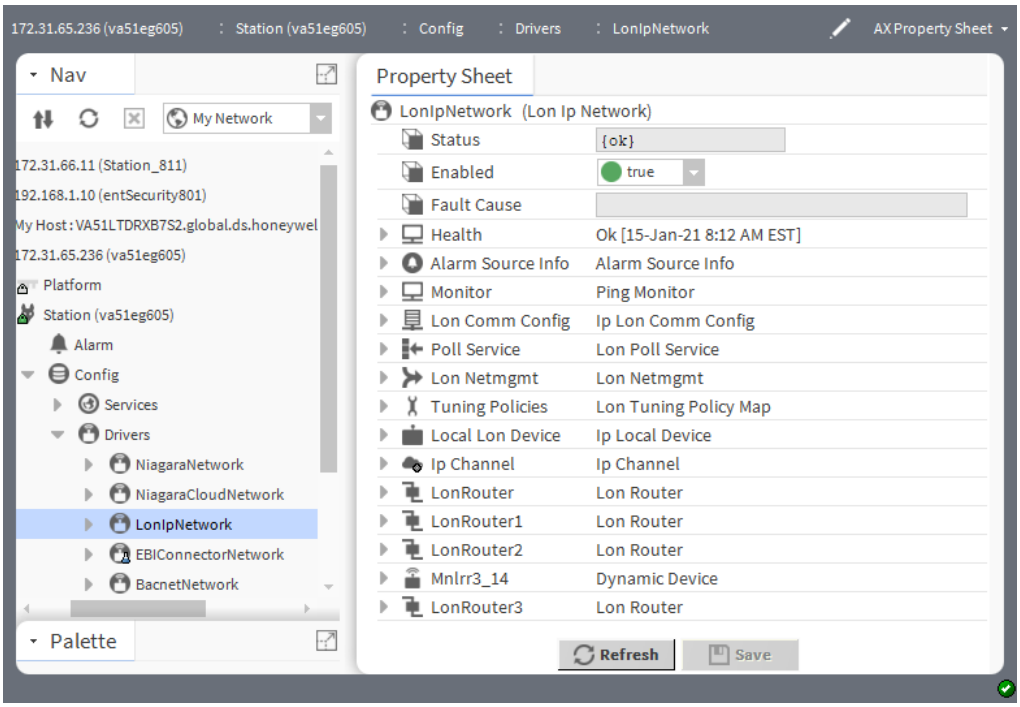
Chapter 8 lonIp-LonIpNetwork

Topics covered in this chapter

♦ lonIp-IpChannel
♦ lonIp-IpLocalDevice

This component is the base container for all Lon IP components in the station.

Figure 59 LonIPNetwork Property Sheet



To access this view, drag the **LonIPNetwork** component from the palette to the Nav tree, and right ckick the network and click **Views→Property Sheet**.

In addition to the standard properties (Status, Enabled, Fault Cause, Health, Alarm Source Info, Monitor, Poll Service and Tuning Policies), these properties are unique to Lonworks.

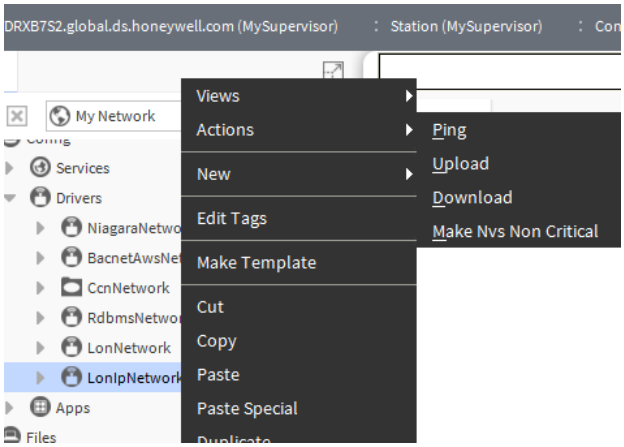
Property	Value	Description
Lon Comm Config	additional properties	Configures the communication stack for a single Lonworks connection including specifying the port's **`Device Name`** For specific properties, refer to a separate topic in this guide.
Lon Netmgmt	additional properties	Serves as the container for Lon network management functions provided by the station. For specific properties, refer to a separate topic in this guide.

Property	Value	Description
Local Lon Device	additional properties	Configures the local interface to the Lonworks fieldbus. For specific properties, refer to a separate topic in this guide.
IP Channel	additional properties	Configures network and server properties. For specific properties, refer to separate topics in this guide. Double-clicking this slot opens the **Member Manager**.

Actions

To access the **Action** menu, right click **LonIPNetwork** in the nav tree and click **Actions**.

Figure 60 LonIpNetwork actions



Action	Description
Ping	Attempts communication with the device. If successful, the device status reports `{ok}`. If this fails, the system sets device status to `{down}`
Upload	Reads transient (nvs) and persistent (ncis and cps) data from the device and writes them to the station database (Lon device). An **Upload** window selects the type of data.
Download	Writes persistent data (ncis and cps) to the device from values in the station database (Lon device). A **Download** window selects recursive writes.
Make Nvs Non Critical	Changes the network variable.

lonIp-IpChannel

This component is a child of the **LonIpNetwork**. It contains two frozen container slots: **Network Config** and **MemberTable**.

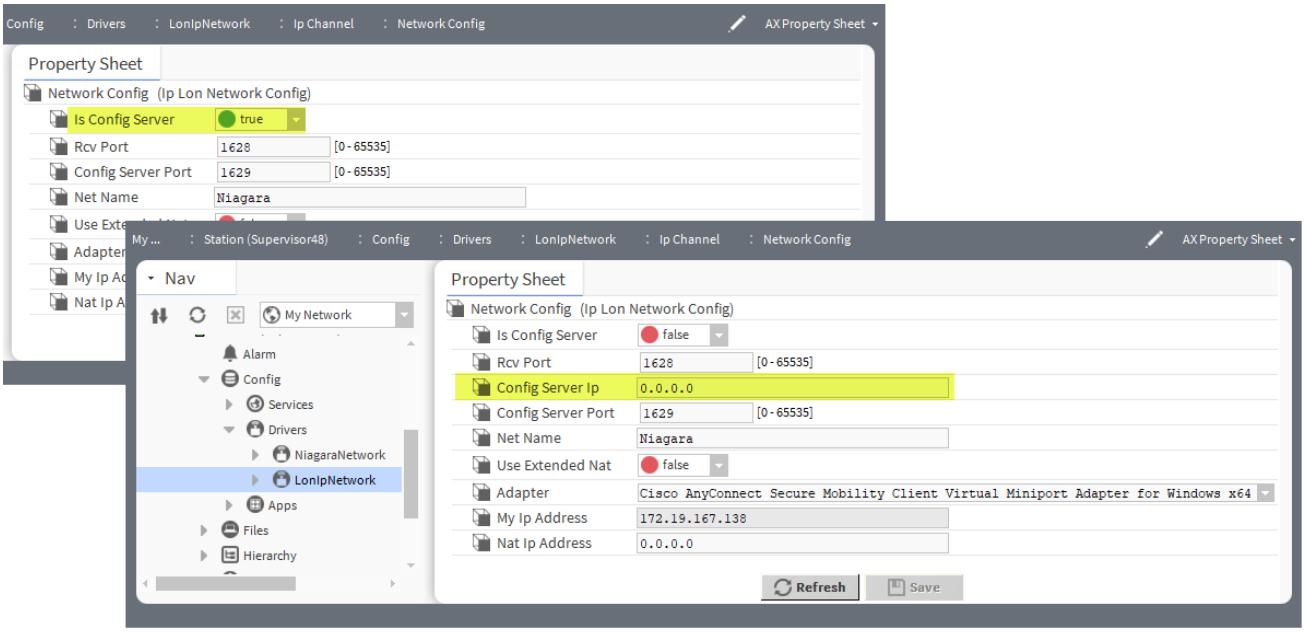
You drag or copy this component from the **lonIp** palette to the **LonIpNetwork**. Its default view is the **Member Manager** view. This component has no properties of its own.

lonIp-IpLonNetworkConfig

This component is one of two frozen containers under an **Ip Channel** component of a **LonIpNetwork**. It contains properties that specify the CEA-852 IP Configuration Server (Config Server) and the software ports used.

NOTE: The **Config Server** property defaults to `false`. To configure the controller station to function as a configuration server, set this property to `true`.

Figure 61 Ip Lon Network Config properties



The screen capture shows two versions of this **AX Property Sheet**. When **`Is Config Server`** is `true`, there is no need to define the **`Config Server Ip`**, which is only available when **`Is Config Server`** is set to `false`.

To access this view expand, **LonIPNetwork→IP Channel** and double-click **Network Config**.

Property	Value	Description
Is Config Server	`true` (default) or `false`	Determines how to populate the Ip Channel table after you add a device to the Config Server and the station contacts the server (the station becomes a member of the CEA-852 channel). `false` dynamically populates this table with the other node members of the channel. `true` uses the **Member Manager** view on the parent **Ip Channel** component for you to manually add a channel member for each Lon/IP router device, specifying the IP address of the controller.
Rcv Port	number between 0 and 65535 (defaults to 1628)	Configures the port to receive data.
Config Server Ip (appears when **`Is Config Server`** is `false`)	IP address	Defines the IP address of the third-party Config Server.
Config Server Port	number between 0 and 65535 (defaults to 1629)	Identifies the port used by the server.
Net Name	text	Configures the network name.
Use Extended Nat	`true` or `false` (default)	Enables NAT (Network Address Translation).

Property	Value	Description
Adapter	drop-down list	Provides the name for the network adapter.
My Ip Address	read-only	Reports the IP address of the platform.
Nat Ip Address	IP address (defaults to 0.0.0.0)	Defines the device's assigned external (Internet) address if the device is behind a NAT (Network Address Translation) router. If not, leave this property at its default value.

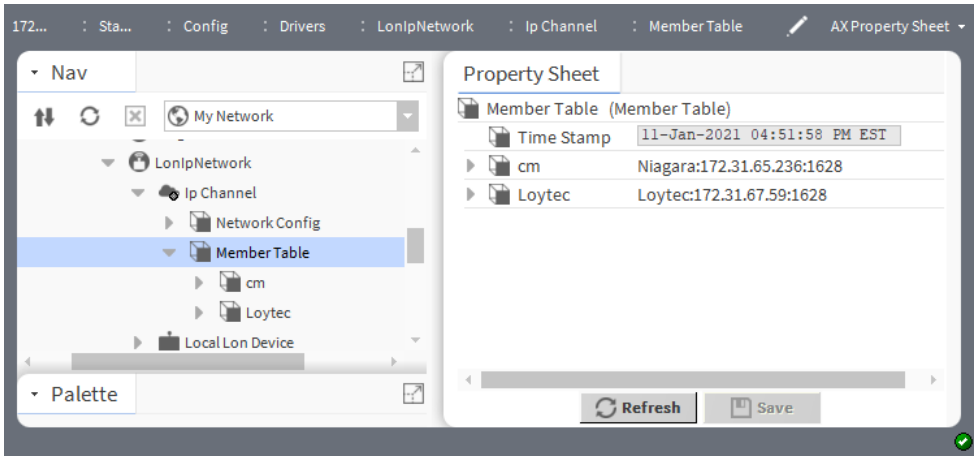
lonIp-MemberTable

This component is a feature of the LonIP driver and a child of the **Ip Channel** component under the **LonIp-Network**. It lists the channels under the network.

Depending on how the network is configured, the table is populated in one of two ways:

- The **Ip Channel** has a **`Is Config Server`** property, which defaults to `false`. After you add a device to the Config Server, and contact is made with the Config Server (the station becomes a member of the CEA-852 channel), the driver dynamically populates this table with the other node members of the channel.
- If the **Ip Channel** property **`Is Config Server`** is set to `true`: you use the (default) **Member Manager** view on the parent **Ip Channel** component to add a new channel member, specifying the IP address of the controller. You also add a channel member for each Lon/IP router device.

Figure 62 Member table Property Sheet



To access this view expand, **LonIPNetwork→IP Channel**, and double-click and click **Member Table**.

This **Property Sheet** contains one container.

Property	Value	Description
Time Stamp	read-only	Reports when the event occurred.
Channel Member	additional properties	Lists the channels. Channel properties are documented in a separate topic.

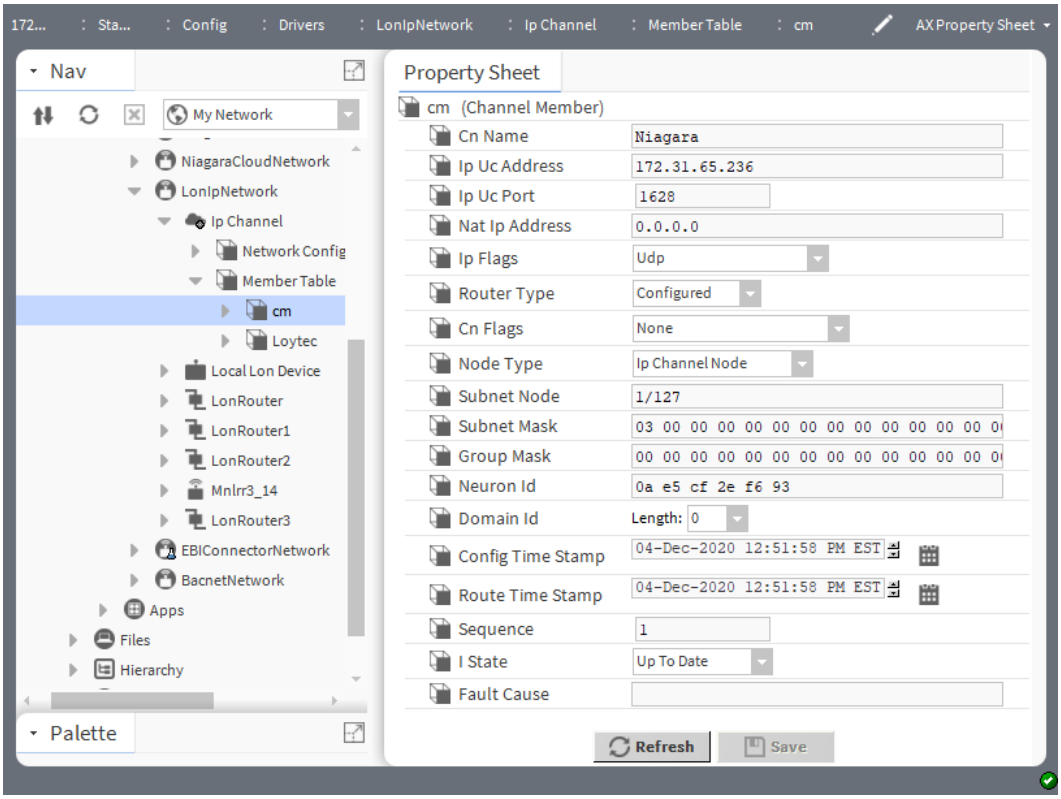
lonIp-ChannelMember

This component is a child of the **MemberTable** container under the **IP Channel** component of a **LonIpNetwork**. A ChannelMember represents a Lon/IP router or an IP channel node.

ChannelMember components represent Lon/IP 852 routers, as well as one required entry for the host station itself (as a channel node). If using a third-party (non-framework) Config Server, the driver automatically adds all ChannelMembers once communications to the CS have been established.

If configuring the station as the CEA-852 Config Server, you must manually add ChannelMembers. Once added, you can monitor these component's status in the **Member Manager** view, and if necessary, also edit and open a browser connection to any selected one (for possible configuration, if the device provides a built-in web server). ChannelMembers also provide a number of right-click actions.

Figure 63 Ip Channel Member properties



To access this view expand, **LonIPNetwork→IP Channel**, and double-click and click **Member Table→ChannelMember**.

In addition to the standard property (Fault Cause), these properties are unique to this component.

Property	Value	Description
Cn Name	name	Identifies the channel by name.
Ip Uc Address	IP address (defaults to 0.0.0.0)	Configures the IP UC (united communications) address. UC is a set of products that provide a consistent, unified user interface and experience across multiple devices and media types. (Wikipedia)
Ip Uc Port	number	Configures the IP UC port number.
Nat Ip Address	Ip address (defaults to 0.0.0.0)	Defines the device's assigned external (Internet) address if the device is behind a NAT (Network Address Translation) router. If not, leave this property at its default value.
Ip Flags	drop-down list	Selects the type of flag.

Property	Value	Description
		`Udp` (User Datagram Protocol) `Tcp` (Transmission Control Protocol) `Udp_Tcp` chooses both. `Multi Cast` addresses communication to a group. `Udp_Multi Cast` combines the two types. `Tcp_Multi Cast` combines the two types. `Udp_Tcp_Multi Cast` combines the two types.
Router Type	drop-down list (defaults to `Unknown`)	Selects the way the router handles messages. `Configured` determines which packets to forward based on internal routing tables. `Learning` forwards messages that meet forwarding rules. `Bridge` forwards all messages received regardless of the destination. `Repeater` forwards all messages in both directions regardless of the message's destination or domain. `Unknown`

Cn Flags	drop-down list	Configures when to include the enumerated descriptor for the channel flag:
		`None`
		`All Broad Casts`
		`Security`
		`All Broad Casts_Security`

Node Type	drop-down list	Identifies the type of node:
		`Unintialized`
		`Non Ip To Ip Router`
		`Ip Channel Node`
		`Ip channel Proxy`
		`Ip To Ip Router`

Subnet Node	subnet (number) and node (number)	Reports the assigned Lonworks subnet/node address. Node must be unique for this device on the channel.
Subnet Mask	32–bit number	Defines a bitmask that when applied by a bitwise AND operation to any IP address in the network, yields the routing prefix. Subnet masks are also expressed in dot-decimal notation like an IP address. For example, 255.255.255.0 is the subnet mask for the prefix 198.51.100.0/24. (Wikipedia)
Group Mask	32–bit number	Identifies a group of devices.
Neuron Id	hexadecimal number	Maps the device in the station database to a different physical device (or no device, if the address is not available). While

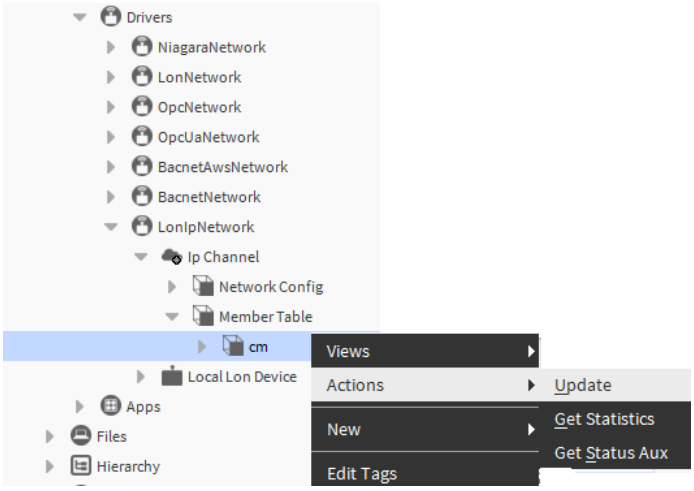
Property	Value	Description
		most **DeviceData** properties should be left at default values, this is one property you may routinely edit.
Domain Id	text	Defines a Lonworks working domain.
Config Time Stamp	date and time	Configures the contents of system time stamps.
Route Time Stamp	date and time	Configures a separate route time stamp.
Sequence	number (defaults to one (1))	Defines the position in a routing sequence.

I State	drop-down list	Selects the stage in the process of setting up the member.
		`New Member`
		`Sent D R Req`
		`Sent D R`
		`Sent C R Req`
		`Up To Date`
		`Error`

Actions

To access **Actions** menu expand, **LonIPNetwork→IP Channel→Member Table**, right click **ChannelMember**.

Figure 64 Channel Member actions



Action	Description
Update	Forces an update of LonIp channel information to the device. An update occurs automatically upon station startup, and when LonIp channel information changes.
Get Statistics	Opens a popup window with various statistics for the selected channel member.
Get Status Aux	Opens the **Get Statu Aux** window for the selected channel member.

lonIp-IpLocalDevice

This component is similar to the **LocalLonDevice** component under a regular **LonNetwork**, and has the same default name. It is a child of the **LonIpNetwork** component as copied from the **`lonIp`** palette.

You use this component to specify the station's channel and subnet/node address (it must be consistent with the IP Lon channel that the station is to become a member of). The station presents itself as an IP channel node using this address.

Ip Local Lon Device properties and actions are the same as those for the Local Lon Device.

Chapter 9 Plugins (views)

Topics covered in this chapter

- Lon Device Manager
- Lon Router Manager
- Lon Link Manager
- Lon Utilities Manager
- Local Nv Manager
- Nv Manager
- Nc Manager
- Changeable Nv Manager
- Lon Point Manager
- Link Filter View
- Member Manager

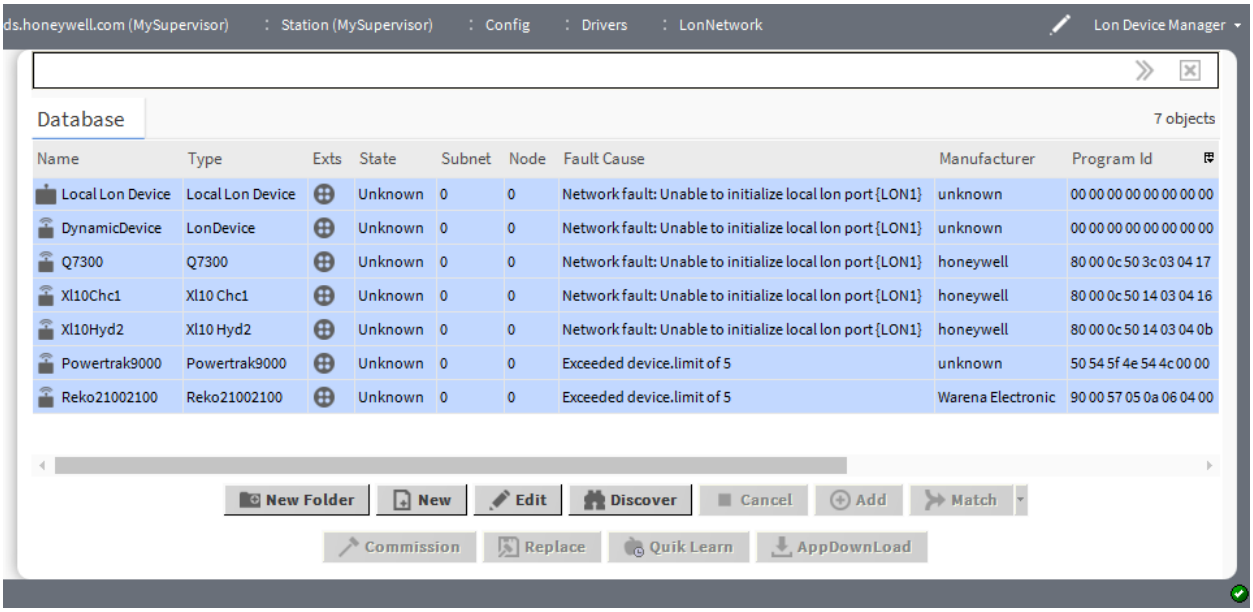
Plugins provide views of components and can be accessed in many ways. For example, double-click a component in the Nav tree to see its default view. In addition, you can right-click on a component and select from its **Views** menu.

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

Lon Device Manager

This is the default view for the Lon network. You use it to discover, add, commission, configure, and replace the network devices.

Figure 65 Lon Device Manager (not learn mode)



To access this view, double-click the **LonNetwork** node in the Nav tree or right-click the same node followed by clicking **Views→Lon Device Manager**.

This view provides support for learning (or discovering/adding) Lonworks devices to the database, for managing device addresses, and for downloading standard applications to devices.

Columns

Column	Description
State	Indicates the current condition of the device.
Channel Id	Identifies the Neuron chip and other information about the local Lon device.
Subnet	Identifies the subnet. A particular subnet must be contained in a single channel. There can be multiple subnets on a single channel.
Node	Reports the assigned Lonworks subnet/node address, unique on site for this device.
Manufacturer	The device manufacturer provides the .xif and resource files for Lonworks devices.
Program Id	Identifies the interface to a dynamic device (network node). Different external interfaces should be associated with unique Program Ids. Some vendors support multiple device types in the same hardware platform. Refer to AppDownload.
Neuron Id	Maps the device in the station database to a different physical device (or no device, if the address is not available). While most DeviceData properties should be left at default values, this is one property you may routinely edit.
Path	Facilitates sorting of devices by parent **LonDevicefolder** (reflected in the path. This technique is often used on a routed Lon network, where devices may be on different network channels (channel Id and subnet).

Buttons

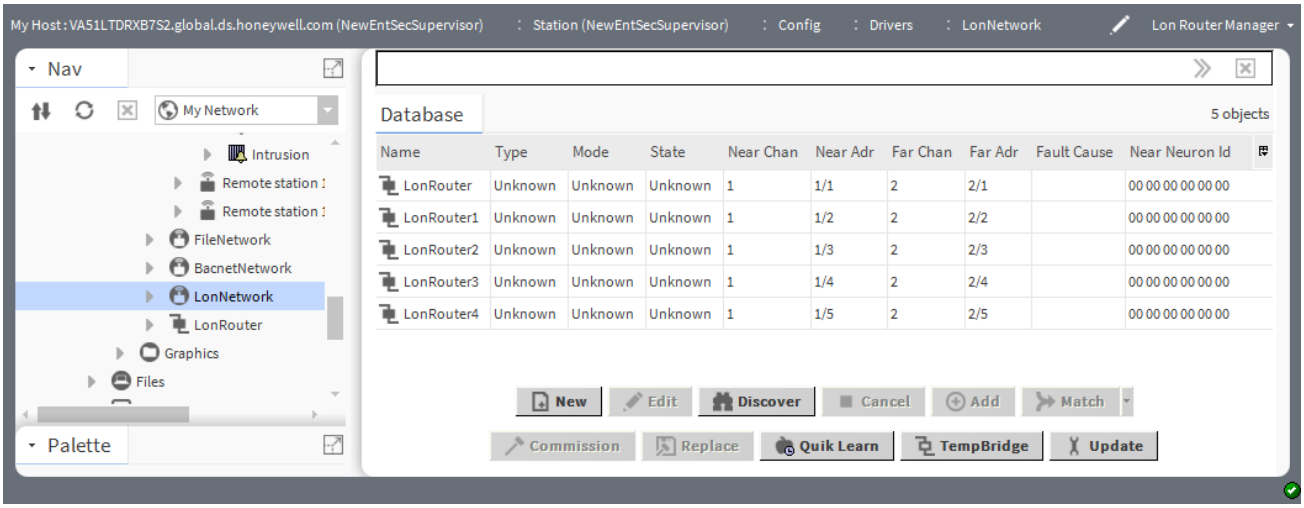
In many ways, the **Lon Device Manager** works similar to other device managers that support online device discovery. A second row of buttons exists below ones common to most device managers. These buttons are only available when not in learn mode (split panes, **Discovered** and **Database**).

- **New Folder** creates a new folder for devices.
- **New** creates a new device record in the database.
- **Edit** opens the device's database record for updating.
- **Discover** runs a discover job to locate installed devices, which appear in the **Discovered** pane. This view has a standard appearance that is similar to all Workbench **Device Manager** views.
- **Cancel** ends the current discovery job.
- **Add** inserts a record for the discovered and selected device in the database.
- **Match** associates a discovered device with a record that is already in the database.
- **Commission** sets a selected device's internal tables (including address-related) to a functioning but unbound state. You perform this on any newly-added Lon device except those created by a Quik Learn of a previously managed network, where the initial state of the device is already `Config Online`.
- **Replace** Downloads network management data to a selected device it is used when replacing a device, or when re-synchronizing nv bind information.
- **Quik Learn** combines online node discovery and Lon device (database) creation in one operation. Specifically, it searches locally-installed **lon**`vendor` modules to find possible Lon Xml files. This differs from a **Discover** and **Add**, where the system searches all modules used by Workbench.
- **AppDownload** Downloads a vendor-supplied `.nxe` file, such as may be used for a firmware update or application update.

Lon Router Manager

This view is one of four views of the **LonNetwork** and provides support for discovering and adding Lonworks routers to the station database, and for managing router addresses. If no Lonworks routers are installed on the physical network, you can safely ignore this view.

Figure 66 Lon Router Manager view of a LonNetwork



You access this view by right-clicking the Lon network node in the Nav tree followed by clicking **Views→Lon Router Manager**.

In many ways, this view works like the **Lon Device Manager**. However, the **Lon Router Manager** is unique in that it is for management of Lon routers only, meaning that it does not include other (non-router) Lon devices.

Columns

Column	Description
Name	Reports the name of the Lonworks router.
Status	Indicates the condition of the network, device or component at the last check. `{ok}` indicates that the component is licensed and polling successfully. `{down}` indicates that the last check was unsuccessful, perhaps because of an incorrect property, or possibly loss of network connection. `{disabled}` indicates that the **`Enable`** property is set to `false`. `{fault}` indicates another problem. Refer to **`Fault Cause`** for more information.
Type	Identifies the type of device: Dynamic Device, Ip Local Device, etc.
Mode	Reports type of function of the router: `Normal`, `Init Router Table`, `Temporary Bridge` or `Unknown`.
State	Reports the current condition of the router.
Near Chan	Identifies one side (half) of the router. A router typically does not perform at the wire-rate because of latency, including the time to receive and buffer the incoming packet at the near side, the time to forward the packet between the halves, and the time to buffer and transmit the packet at the far side. (from *Echelon Lonworks Router User Guide*)
Nare Adr	Reports the address of one side of the router.
Far Chan	Identifies the other side of the router.
Far Adr	Reports the address of the other side of the router.
Fault Cause	Indicates the reason why a system object (network, device, component, extension, etc.) is not working properly (in fault). This property is empty unless a fault exists.
Near Neuron Id	Displays the unique 48–bit ID for the near side of the router.
Far Neuron Id	Displays the unique 48–bit ID for the far side of the router.

Buttons

As in the **Lon Device Manager**, there is a second row of buttons below ones common to most device managers. These buttons are available when not in learn mode (split panes, Discovered and Database). These buttons are:

- **New** creates new router records.
- **Edit** changes existing router records.
- **Discover** runs a discover job to locate installed devices, which appear in the **Discovered** pane. This view has a standard appearance that is similar to all Workbench **Device Manager** views.
- **Cancel** ends the current discovery job.
- **Add** inserts a record for the discovered and selected device in the database.
- **Match** associates a discovered device with a record that is already in the database.
- **Commission** sets a selected device's internal tables (including address-related) to a functioning but unbound state. You perform this on any newly-added Lon device except those created by a Quik Learn of a previously managed network, where the initial state of the device is already `Config Online`.
- **Replace** Downloads network management data to a selected device it is used when replacing a device, or when re-synchronizing nv bind information.
- **Quik Learn** combines online node discovery and Lon device (database) creation in one operation. Specifically, it searches locally-installed **lon**`vendor` modules to find possible Lon Xml files. This differs from a **Discover** and **Add**, where the system searches all modules used by Workbench.
- **TempBridge** creates a temporary bridge. If the packet is sent on a domain to which a bridge belongs, this type of router forwards packets received on one channel to the other channel. You would use a bridge to span domains. In a single domain network, a bridge functions essentially the same as a repeater. (from *Echelon Lonworks Router User Guide*
- **Update** Refreshes the view.

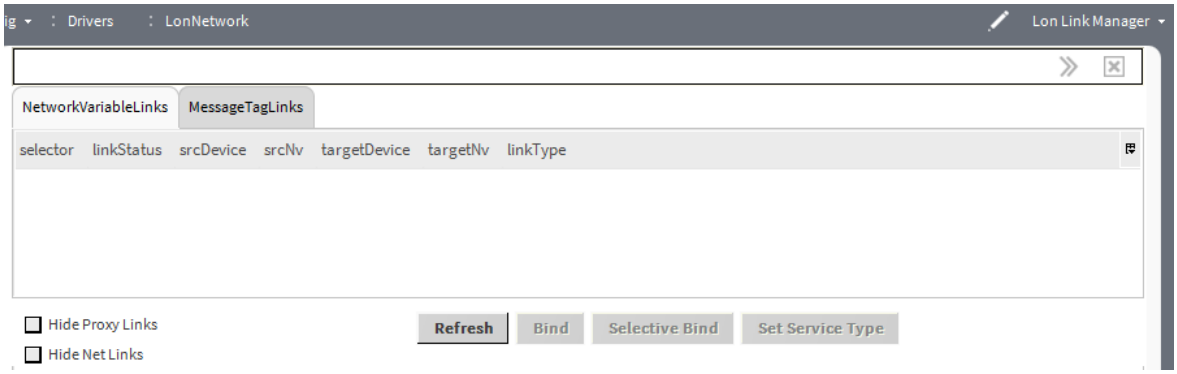
Lon Link Manager

This view is one of four views on the **LonNetwork**. You use it to review and manage the bindings of network variables. When the station serves as the network manager of the Lonworks network, you often use this view.

NetworkVariableLinks tab

This is the default tab. It displays links and bindings between network variables. Often, this is the only tab you use in this view.

Figure 67 NetworkVariableLinks tab in Lon Link Manager



You access this view by right-clicking the **LonNetwork** node in the Nav tree followed by clicking **Views→Lon Link Manager**.

Each row in this tab represents either a connection between the station (LocalDev) and a specific network variable (nvi or nvo) in a Lon device, or a Lon network variable binding directly between two other Lon devices.

- For local device connections, each row corresponds to a particular Lon proxy point. Proxy points for ncis, cps, and read-only nvi proxy points are not included.
- For a binding directly between two Lon devices, the software models the binding as a graphical connection (link) that is visible on each device's glyph (shape) as it appears on the station's wire sheet.

Columns

Data columns provide the current information about each link and binding. By default, this table is sorted by selector, ascending (1, 2, 3, so on). You can click on any table column header to resort the table, and click and drag to resize column widths.

Column	Description
Selector	This number references an individual network variable, which is stored in a table within the local Neuron chip.
linkStatus	Reports the current health of each link: • `NewLink` indicates an added proxy point or link between nodes that is not yet bound. Until bound, entries do not exist in the address tables of the nodes, and the system uses polling exclusively. • `Bound` indicates that a binding has been established between two network variables or nodes. If you used Quik Learn to learn links in an existing (managed) network, learned links appear as bound. In addition, following a bind command, all entries for Lon proxy points list as bound. • `Poll Only` indicates that **`Set Service Type`** is configured for `Poll Only`. Upon the next bind command, the system sets network variables to the unbound state. • `Obsolete` indicates that a formerly bound proxy point was deleted, or that a bound link between two nodes was deleted. In either case, the address tables in both nodes have not yet been updated—entries still exist. Use the **Bind** command to change these obsolete entries to `Unbound`. • `Unbound` indicates an obsolete link following a a **Bind** command, just to verify that the link has been deleted. To remove these entries (rows) entirely, click the **Refresh** button. In some cases, a link/binding may appear in the **Lon Link Manager** showing an error-type **`linkStatus`**, as one of the following types: • `AliasError`—One or more connections have been made which require the use of alias nvs when none or insufficient aliases are available. Delete links and bind to remove error. • `AutheicateError`—Authentication error. Check the messaging service type for the network variable linkage. • `ComError`—Communications error occurred during the bind operation (for example, timeouts). Try binding the network variables again. If errors persist, check communications. • `DeviceError`—Device error. The state of the node will not support a network variable binding. • `DirtyGroup`— Means an address table problem. Usually, a second pass at a bind will clear things up. • `DirtyPoll`—An existing group has been disrupted by commissioning or removal of one of its member devices. Bind to reassign groups. • `Error`—An error apart from one of the other specified error types. • `GroupError`—Unable to assign the node to a group. A node can belong to 15 different groups (up to 15 address table entries) when acknowledged messaging service is used. To correct, either reduce the number of address table entries, or adopt unacknowledged/repeated messaging service. • `GroupExcludeError`— A set of connections has been created for which no set of groups can be assigned that meet all group assignment rules. Delete links and bind to remove error.

Column	Description
	• `NvTypeError`—SNVT types do not match. Either the Lon xml (lnml) file is not appropriate for the node, or (if Learn Nv was used) the node's self-documentation does not properly represent the actual data stored in that device. Obtain an up-to-date lnml file for the device, or use Learn Nv. • `MaxCritError`—Too many targets (bindings or linkages) are made to use the critical messaging service. Reduce the number of bindings or adopt a non-critical messaging service. • `PriorityError`—The priority setting of the link cannot be legally accommodated (that is, priority is selected but nv is not priority, and priority is not configurable). • `ServiceTypeError`—Message service type mismatch. The assigned message service type (linkType) does not match the actual service type found in the node.
srcDevice	Identifies the source device.
srcNv	Identifies the source network variable.
targetDevice	Identifies the target device.
targetNv	Identifies the target network variable.
linkType	Identifies the type of link.

Check boxes

The two check boxes in the lower left of the view filter (hide) links from the view as follows:

- **`Hide Proxy Links`** filters out links from or to proxy points (LocalDev) from the view.
- **`Hide Net Links`** filters out links between other Lon devices/controllers from the view.

The hide filters only affect the display of links. A bind operation still binds all links, even though they might not be visible.

Buttons

The buttons serve these functions:

- **Refresh** globally polls the status of the bindings on the network and displays that information in the table. If unbound (deleted) links were previously listed, **Refresh** removes them from the table.
- **Bind** globally establishes nv bindings for all entries per the linkType (**`Service Type`**) specified for each entry. This button remains available regardless of any selected items. When you click it, the driver performs a global Lon-bind job with standard station job visibility. To bind only selected items, use **Selective Bind** instead.
- **Selective Bind** globally establishes nv bindings for only the selected entries per the linkType (**`Service Type`**) specified in each entry. When you click it, the driver performs a Lon-bind job with standard station job visibility. To globally bind all entries, use the **Bind** button instead.

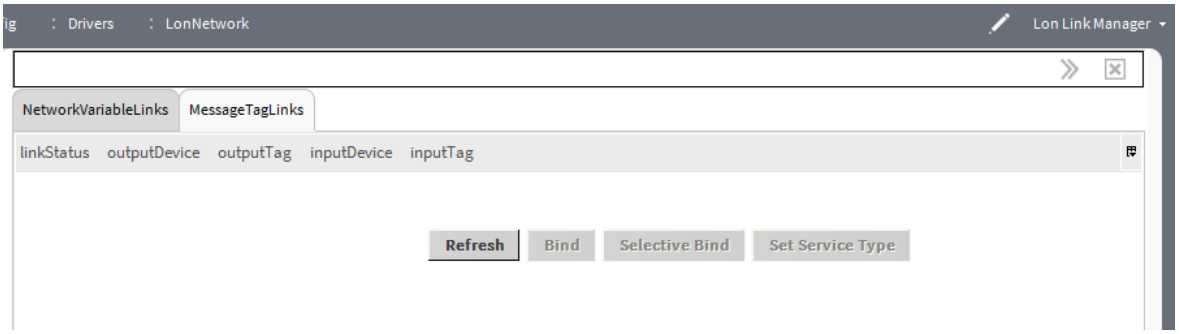
  If a link is already bound and you change it to another message **`Service Type`**, its linkStatus changes to `NewLink` until you bind (or selective bind) it again.
- **Set Service Type** sets the LonTalk message **`Service Type`** that controls the properties for the next bind. When you click it, a popup window provides a drop-down selection of message service types to use. Or, you can select **`Poll Only`**, which sets nvs to an unbound state.

MessageTagLinks tab

This tab manages the bindings between messages in and message-out tags of Lon devices (if any devices are used this way).

In some cases, applications require a different data interpretation model than that provided by network variables. In these cases, nodes construct individual messages and assign them an address. These are referred to as explicit messages, and nodes can use logical input and output ports (message tags) to send and receive these messages. While the driver provides this feature, it is not used. Generally, Lonworks nodes use network variables to exchange data.

Figure 68 MessageTagLinks tab in Lon Link Manager



All LonMark-certified nodes contain a message-in tag, which can be used to receive messages. In addition, nodes can declare bi-directional message tags that can be used to both send and receive messages. If message tags bindings are used, the **Lon Link Manager** opens their status in a fashion similar to that used to display network variable bindings.

Each row represents a link between devices for the purpose of exchanging messages.

Columns

Data columns in message tag links table provide the current information about each message tag link.

Column	Description
linkStatus	Reports the health of the link.
outputDevice	Identifies the sending device.
outputTag	Identifies the type of sending binding.
inputDevice	Identifies the receiving device.
inputTag	Identifies the type of receiving binding.

Buttons

- **Refresh** globally polls the status of the bindings on the network and displays that information in the table. If unbound (deleted) links were previously listed, **Refresh** removes them from the table.
- **Bind** globally establishes nv bindings for all entries per the linkType (**`Service Type`**) specified for each entry. This button remains available regardless of any selected items. When you click it, the driver performs a global Lon-bind job with standard station job visibility. To bind only selected items, use **Selective Bind** instead.
- **Selective Bind** globally establishes nv bindings for only the selected entries per the linkType (**`Service Type`**) specified in each entry. When you click it, the driver performs a Lon-bind job with standard station job visibility. To globally bind all entries, use the **Bind** button instead.

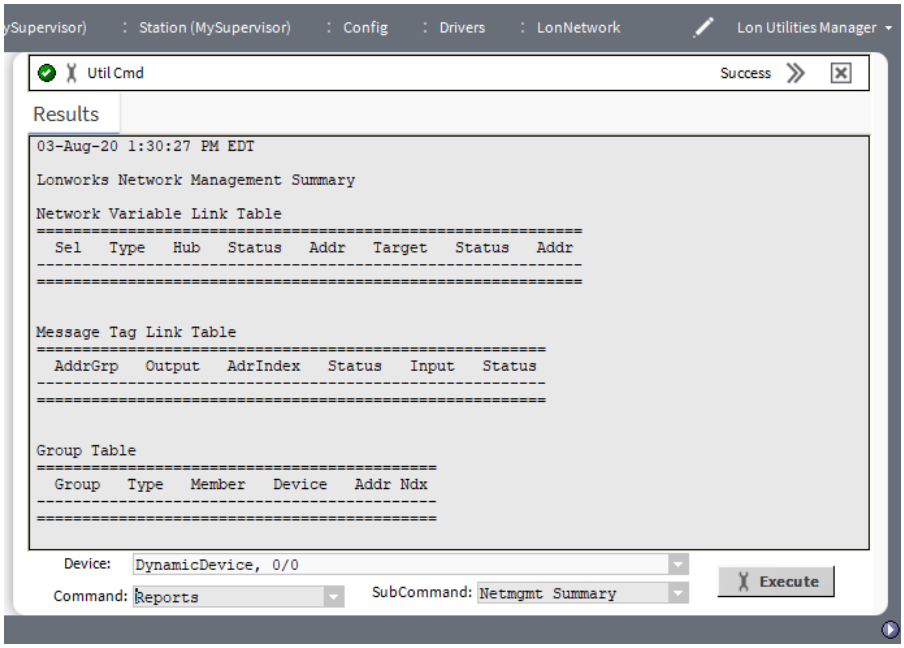
  If a link is already bound and you change it to another message **`Service Type`**, its linkStatus changes to `NewLink` until you bind (or selective bind) it again.
- **Set Service Type** sets the LonTalk message **`Service Type`** that controls the properties for the next bind. When you click it, a popup window provides a drop-down selection of message service types to use. Or, you can select **`Poll Only`**, which sets nvs to an unbound state.

Lon Utilities Manager

This view one of four views of a Lon network. It provides access to Lonworks diagnostic utilities to apply to selected Lonworks nodes, such as those included in the former NODEUTIL (command line) program. This collection of utilities allow you to query the status of a Lonworks node, find nodes on the network, identify

nodes, display a node's file structures and data structures, discover more about communications errors, verify binding information, and more. The queries require a command. Some queries also include a sub-command. The results provide information about the current state of the Lon network.

Figure 69 Lon Utilities Manager view of a LonNetwork



You access this view by right-clicking the **LonNetwork** node in the Nav tree followed by clicking **Views→Lon Utilities Manager**.

The **Execute** button is the run button, which performs the search.

The **`Device`** drop-down list depends upon the current list of devices.

When to use Lon utilities

Reason	Command and sub-command
To identify a device from a service pin message:	Main command: `Identify`, sub-command: `Service Pin`.
To determine the working domain of an existing managed network:	Select <a Lon device>, main command: `Reports`, sub-command: `Domain Table`.
To verify that binds issued from the **Lon Link Manager** executed correctly, meaning that the actual devices and the station database are both synchronized:	If selective binds made to just one Lon device: Select <that device>, main command: `Reports`, sub-command: `Verify`. If binds were made to multiple Lon devices: Select `Local Lon Device`, main command: `Reports`, sub-command: `Verify`. If the report shows discrepancies, take the appropriate actions from the **Lon Device Manager**: For any device shown in error, select it, then do a Replace. If the database shows errors, select the affected device(s), then do a Quik Learn. Following either action, you perform another `Bind` from the **Lon Link Manager**, then re-run the verify report from the **Lon Utilities Manager**.

Command

This drop-down list selects from among utilities that you can run:

`Status` displays real-time status for a node, or performs some other network management operation that affects device status.

`Display` issues a query status message to the node. This retrieves the network error statistic accumulators, which identify the cause of the last reset, the current state of node, and the last runtime error logged.

`Clear` sends a message to the node to delete its error flag and error counts. After this is processed, the status display for the node updates. Following a clear, you can use the display sub-command to gauge the effects of network traffic on the node.

`Set Unconfigured` sets a node to an online state, and is provided as a convenience for a device not represented in the station database. From a Lon device's property sheet, under the **Device Data** slot, you can directly access and write to a device's **`Node State`**.

`Reset` issues a software reset to the node, and displays new status.

`Data Structs` displays various internal device tables and structures.

`File` displays the internal files of any selected Lonworks device, providing that the device supports such files.

`Identify` defines a node in the property. Basically, commands are either wink (send to node) or service pin (send from node).

`Reports` displays various information for networked nodes, including network management data, transmit errors data, and verifies against what is actually stored inside nodes that are online any inconsistencies in the station database.

`Read Mem` displays the contents of memory.

`Find` discovers nodes on the network, which are not already represented in the station by a Lon device. It is equivalent to the **Discover** command in the **Lon Device Manager**. There are no Find sub-commands, and selected device is not important.

Status sub-commands

These commands are available when you select `Status` for **`Command`**.

- `Display` issues a query status message to the node. This retrieves the network error statistic accumulators, cause of the last reset, current state of node, and the last runtime error logged.
- `Clear` sends a message to the node to clear its error flag and error counts. After this is processed, the status display for the node updates. Following a clear, you can use the `Display` sub-command to gauge the effects of network traffic on the node.
- `Set Unconfigured` sets the node's internal and database state to unconfigured, and displays the new node status. In an unconfigured state, the driver loads a node's application, but configuration data are either not loaded or deemed to be corrupted due to a configuration checksum error.

  When unconfigured, a node responds to status query messages with its Neuron ID rather than subnet/node addressing. Network variables do not update, and its service LED blinks at a once-per-second rate.
- `Set Online` sets a node to an online state, and is provided as a convenience for a device not represented in the station database. From a Lon device's property sheet, under the Device Data slot, you can directly access and write to a device's Node State.
- `Reset` issues a software reset to the node and displays the new status.

Data Structs sub-commands

Within the general commands, these are the additional refinements you can configure. These commands are available when you select `Data Structs` for **`Command`**.

These sub-commands are available for local and remote devices:

- `Address Table` defines the network addresses to which the node can send implicit messages and network variable updates, as well as the groups to which the node belongs. Up to 15 address entries are possible.
- `Domain Table` defines the domain(s) to which the node belongs. Up to two domains can be assigned.

- `Read Only Structure` defines the node's identification as well as some of the application image properties of the node.
- `Config Structure` resides in its EEPROM from where it defines the hardware and transceiver properties of the node. The manufacturer writes some portions of this structure, while the system writes other properties when you install the node.

These additional sub-commands are available when you select a remote device:

- `Nv Alias Table` defines the attributes of the alias network variables in the node. Aliases allow a network variable in one controller to be linked to multiple network variables in another (single) controller.
- `Nv Config` defines the attributes of the network variables that can be configured in the node. Up to 62 entries are possible.
- `Nv Value` displays raw (hex data) values for network variables in the node without any conversion (as provided for the nv Lon components or nv proxy points). You can use this value to verify against values seen in the framework.
- `Self Documentation` displays the self-documentation available in the node. You must provide this capability for a device to have the LonMark logo.
- `Read Mem` provides a (starting) **`Address`** and **`Len`**(gth) properties in which you can enter values, using hexadecimal notation, to read raw memory contents from the selected device. The results display in hex bytes as well as ASCII characters, using 16–byte rows listed by address location.

For example, if you enter an **`Address`** of `01FC` and a **`Len`** of `FF`, the system returns 16 rows of data beginning from `01FC` through `02EC`.

File sub-commands

`File Directory` shows list of available files, including type and size.

`Config Template File` shows template file entries.

`Config Value File` shows config value file entries, and bytes.

`Other` shows files other than template and value files.

Identify sub-commands

`Wink` sends a wink message to a selected node, whereby, depending on the device, it visually (or audibly) indicates receiving the message—for example, flashing an LED in a pattern.

NOTE: Before executing this command, select the specific device from the drop-down list.

`Service Pin` causes the **Lon Utilities Manager** to listen on the network for a node to identify itself. In this mode, the utilities manager displays Waiting on a service pin. This command does not require you to select a device.

When you push the service pin of a Lonworks device, the node sends its domain table to the utilities manager, where it displays in the view.

`Clear Service Pin` cancels a pending Service Pin command.

Reports sub-commands

The following sub-commands are available:

`Netmgmt Summary` summarizes network links, address table entries, and group assignments.

`Program Ids` lists the current Program Ids with associated module and .lnml file or class.

`Transmit Errors` creates and displays a table of various transmission error counts for all nodes on the network. This command results in a status clear sent to all nodes, such that error counts are reset—the next report shows counts since this report was created.

`Transmit Errors No Clear` creates and displays a table of various transmission error counts for all nodes on the network. Does not clear device status in nodes (unlike Transmit Errors command).

`Verify` compares the networked devices' nv configuration and address table entries against the station's database (Lon device's properties) and reports discrepancies. If the **Local Lon Device** is selected, the report verifies all networked Lon devices. If any other device is selected, the report verifies only that device.

`Verify Channels` confirms channels.

`Network Summary` creates a table of devices showing **`Channel Id`**, **`subnet/node address`**, and **`Neuron Id`** as well as routers (if any).

Read Mem

This command provides two properties:

`address` specifies a specific address in memory.

`len` specifies the length to display.

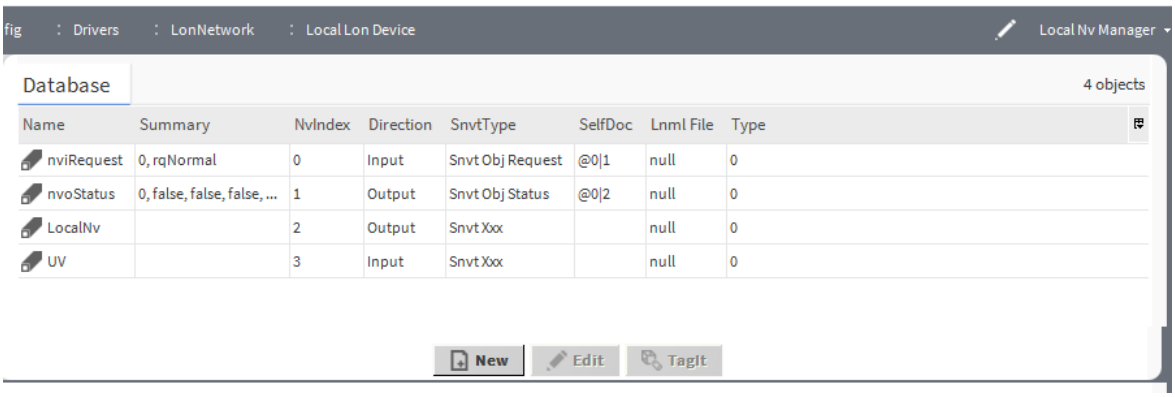
Find

This command provides a search capability. The SubCommand is the search argument.

Local Nv Manager

This manager is the default view of the **Local Lon Device**. It is unique to the **Local Lon Device**. The standard Lon device views are not available for this device. You use this view to add, edit, and access items locally exposed as SNVTs (input or output), which are applicable only when the station is not acting as the Lonworks network manager.

Figure 70 Default sort of local nvs, ncis, is by NvIndex



To access this view right-click **LonNetwork→Local Lon Device** node in the Nav tree, and select **Views→Local Nv Manager**.

While defining local nvs and ncis, the **Local Nv Manager** lists items with a sort order of NvIndex. However, you can click on any column header needed to resort entries.

In a typical operation, the station acts as the **LonNetwork** manager—you do not use this view (or method) to share data among other Lon devices. Instead, you create Lon proxy points under Lon devices, and then link the proxy points into station logic.

However, if you configure the station as only another Lon node, this view lets you create custom Lonworks network variables (nvis, nvos, ncis) available externally to other Lonworks devices. The station appears as a peer Lonworks device. In this scenario, Lonworks network management is not handled by the station.

Columns

Column	Description
Name	Displays the name of the component that represents this device.
Summary	Summarizes the component configuration.
NvIndex	Displays the index value for the network variable.
Direction	Indicates the direction the message traveled.
Send Type	Data columns provide the current information about each link and binding. By default, this table is sorted by selector.
SnvtType	Displays the standard network variable type.
Selfdoc	Displays information for the component. Lon components are self-documenting.
Lnml File	Supports use of manufacturer-defined user nvs, or UNVTs (vs. SNVTs) by specifying an lnml file for a device that implements
Type	Works in conjunction with the Lnml File property, where once you specify an lnml file, Type provides a list of UNVTs from which you can choose for this nv.

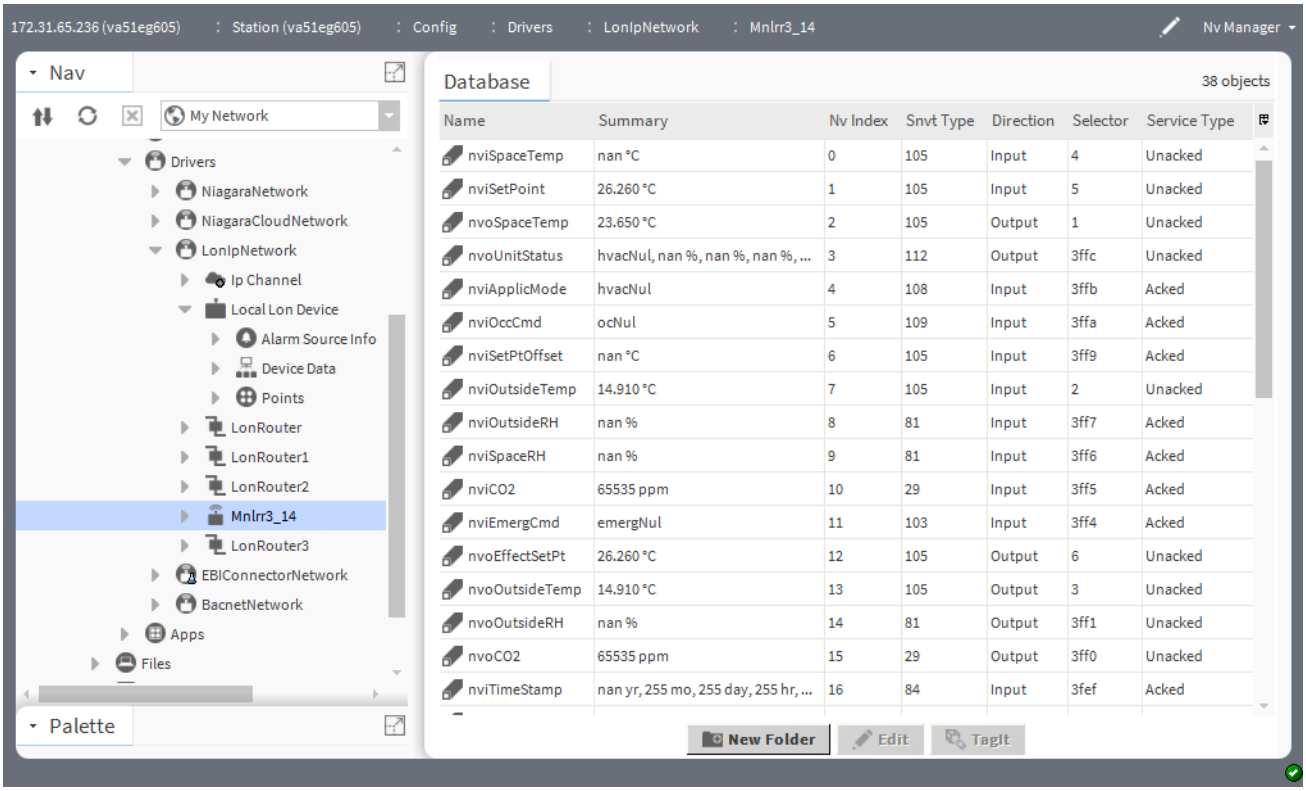
Button

- **New** creates a new device record in the database.
- **Edit** opens the device's database record for updating.
- **TagIt** associates metadata, such as location or unique configuration with the object.

Nv Manager

This Network Variable (Nv) Manager view is the default view of a Lon device (except **Local Lon Device**). Use it to browse through a table of the device's network variable: Lon components nvi and nvo. You can use the standard table edit controls to change the number and types of data displayed in this view. The system updates summary data dynamically through polling.

Figure 71 Empty Nv Manager with column options



To access this view, right-click a **LonDevice** in the Nav tree and click **Views→Nv Manager** or double-click the **LonDevice**.

Columns

Data columns provide the current information about each link and binding. By default, this table is sorted by selector.

Column	Description
Name	Displays the network variable name.
Summary	Displays the value of one or more related properties.
Nv Index	Displays the location of the variable in the record.
Snvt Type	Displays the type of standard network variable for this binding.
Object Index	Displays information related to the object.
Member Index	Displays information related to the member.
Poll Enable	Indicates of polling is on or off. If this column displays true, there are unbound links and the nv is not bound locally. If false, polling is prevented on an nv. The driver still performs the initial read when the nv is first subscribed, but subsequent polling by the poll service does not occur.
Polled	Some Lonworks devices include one or more nvos that are polled types, meaning that the nvo does not send nv updates upon a value change. Such an nvo must be polled. Typically, any such nv represents a data item that does not frequently change.
Auth Conf	
Service Conf	
Priority Conf	Reports if the priority flag is set.

Column	Description
Modify Offline	
Sync	
Changeable Type	Sists all network variables in the device
Bound to Local	Displays a value when an output nv with a proxy is bound.
Priority	Displays processing priority.
Direction	Indicates the direction (input or output) for the Local Nv.
Selector	References an individual network variable, which is stored in a table within the local Neuron chip.
Turn Around	
Service type	Displays `Unacked` or `Acked`.
Authenticated	Represents the Authenticated bind service type.
Addr Index	Displays the location of the address using a number.
Tuning Policy Name	Displays the tuning policy being used.

Buttons

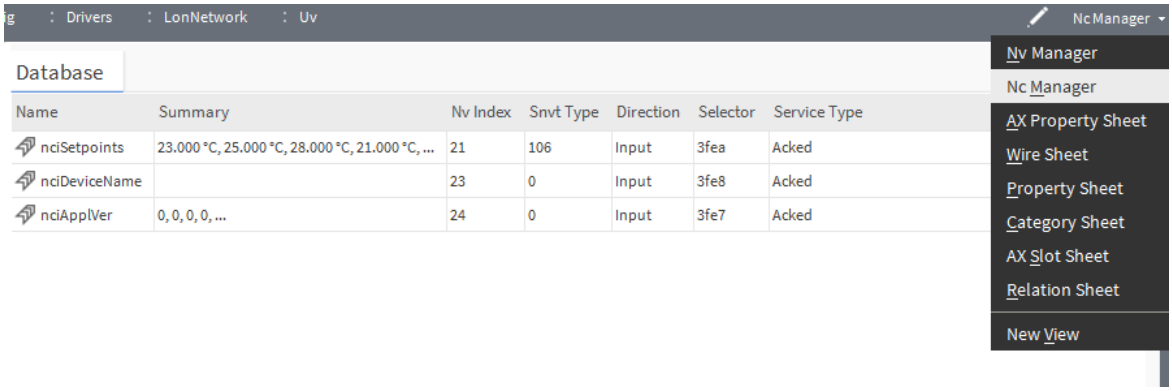
- **New Folder** creates a new folder for devices.
- **Edit** opens the device's database record for updating.
- **TagIt** associates metadata, such as location or unique configuration with the object.

Nc Manager

This view provides access to the ncis. The **Property Sheet** that supports both the **Nc** and **Nv Managers** provides access to these properties.

This view is a table view similar to the **Nv Manager**, but it provides access only to Network Configurable Inputs (ncis) in the device. Table columns can be sorted. You can change the data columns for display using the standard table controls. Edit access to an nci using the **Nc Manager** is limited to changing the name of the associated Lon component.

Figure 72 Nc Manager view on Lon device shows only ncis



To open this view, right-click a **LonDevice** and click**Views→Nc Manager**.

If Lon objects appear in the **Nc Manager**, click the **All Descendants** tool ( ) to flatten the hierarchy in the view and see the ncis in each Lon object.

Columns

Data columns provide the current information about each link and binding. By default, this table is sorted by selector.

Column	Description
Name	Displays the network variable name.
Summary	Displays the value of one or more related properties.
Nv Index	Displays a number.
Snvt Type	Displays the type of standard network variable for this binding.
Config Index	Displays a number.
Mfg Defined	Displays the manufacturer.
Modify Flag	Displays an indicator.
Scope	
Select	
Priority	Indicates a processing priority.
Direction	Reports if the transmission was an input or output.
Selector	References an individual network variable, which is stored in a table within the local Neuron chip.
Turn Around	
Service Type	Displays the type of service.
Authenticated	Represents the Authenticated bind service type.
Addr Index	Displays the address as an indexed number.

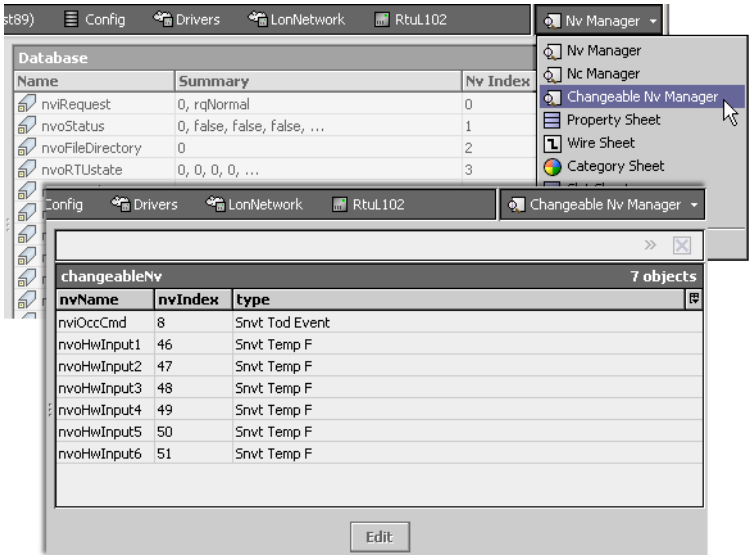
Buttons

- **New Folder** creates a new folder for devices.
- **Edit** opens the device's database record for updating.
- **TagIt** associates metadata, such as location or unique configuration with the object.

Changeable Nv Manager

If a Lonworks device supports changeable network variables, the Lon device component provides another this view.

Figure 73 Changeable Nv Manager view of a Lon device



To open this view, right-click a **LonDevice** and click**Views→Changeable Nv Manager**.

This table-based view lists all network variables in the device that are designated as changeable types, along with the current SNVT type used for each.

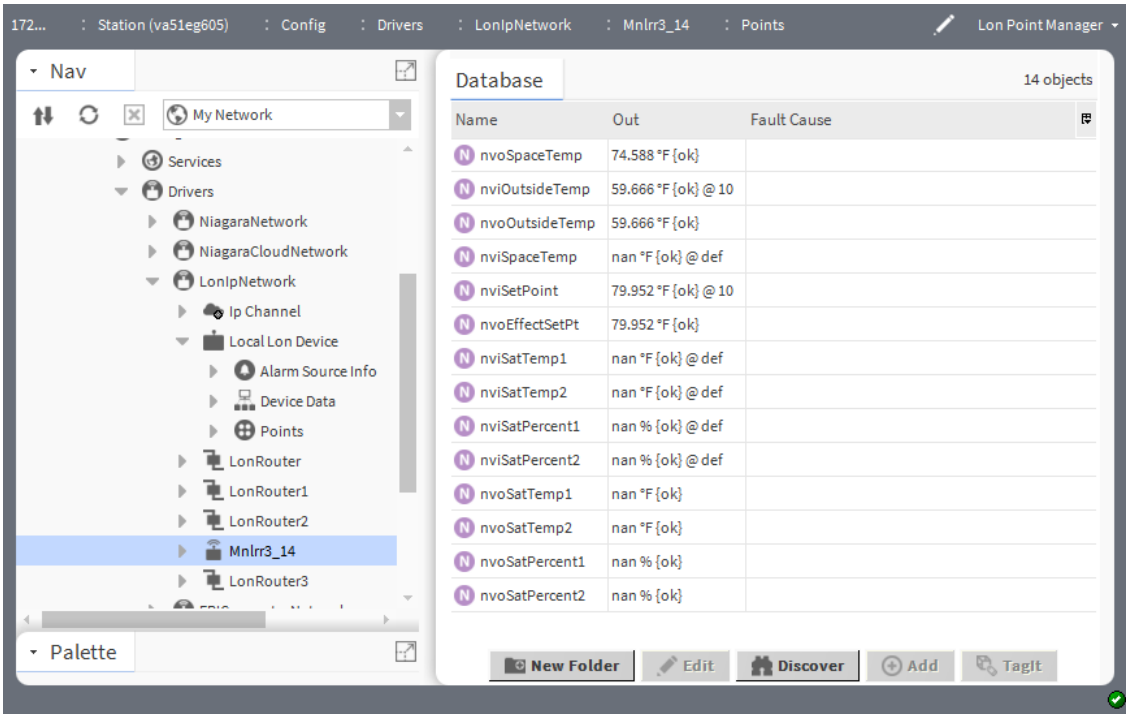
Columns

Column	Description
nvName	Displays the name of the network variable.
nvIndex	Displays a number for the network variable.
type	Displays the type of network variable.

Lon Point Manager

This view creates, edits, accesses, and deletes Lonworks proxy points. It is the default view for the **`LonPointDeviceExt`** (**Points** container) under a **LonDevice**. It is also the default view for any **LonPointFolder**- under the **Points** container.

Figure 74 Lon Point Manager view



To access this view, right-click the **Points** container or **LonPointFolder** and click **Views→Lon Point Manager**.

Discovered columns

Column	Description
Target Name	Under Target Name the driver lists all points by the names of the Lon components that are associated with the device. For example: nvEmerg and nvoUnitStatus.
Element Name	Displays the name of the element.
Units	Indicates the unit of measure.

Database columns

Column	Description
Name	Provides a name for the nvi, nvo, or nci variable. If part of a data structure, it provides a trailing _elementName, which ensures a unique point name. For example, nciTempSetPts_ occupiedCool This is the name of the point and does not affect the Lonworks node.
Out	Displays the calculated or reported value for the property.
Fault Cause	Indicates the reason why a system object (network, device, component, extension, etc.) is not working properly (in fault). This property is empty unless a fault exists.
Type	Identifies the type of point: Numeric, Boolean, Enum, etc.
Target	Reports the Lon component represented by the proxy point, for example: nvoStatus.
Element	If the point is structured, reports the element in the structure. If the point is unstructured, reports the SNVT.
Facets	Represents the parent proxy point's facets, which determine how to display the value in the driver. These depend on the type of device.

Column	Description
Conversion	Specifies how to convert a value read from the device into the parent point's facets.
Enabled	Indicates if the variable is being used.
Read Value	Reports the last value read from the device.
Write Value	Reports the last value written to the device.
Link Type	Specifies the Lon service type to use when binding to this item.

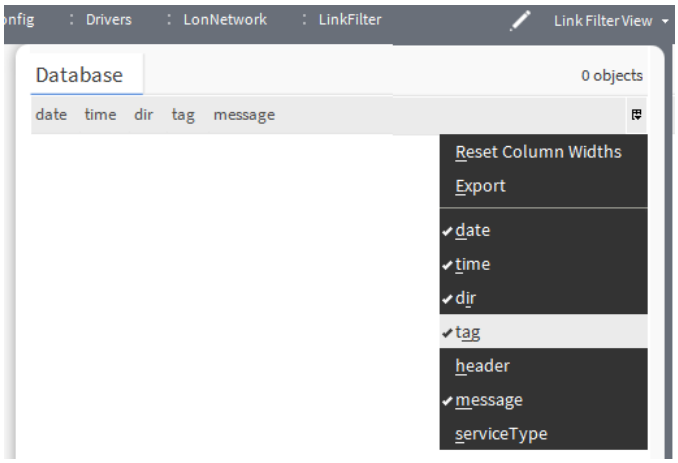
Buttons

- **New Folder** creates a new folder for devices.
- **Edit** opens the device's database record for updating.
- **Discover** runs a discover job to locate installed devices, which appear in the **Discovered** pane. This view has a standard appearance that is similar to all Workbench **Device Manager** views.
- **Add** inserts a record for the discovered and selected device in the database.
- **TagIt** associates metadata, such as location or unique configuration with the object.

Link Filter View

This is the default view on a **LinkFilter** component, a debug-type object found in the **`lonworks`** palette. This table-based view provides low-level messaging data related to Lonworks links.

Figure 75 Link Filter View



To access this view, expand LonNetwork and double-click LinkFilter or right-click LinkFilter and click **Views→Link Filter View**.

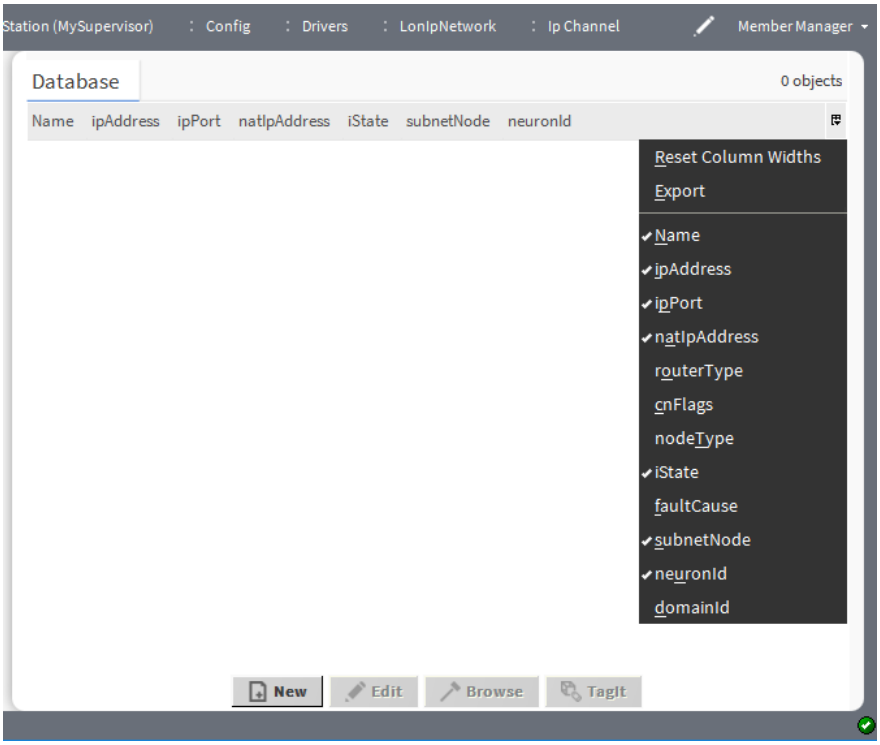
Column	Description
date	Indicates the date the record was captured and displayed.
time	Indicates the time the record was captured and displayed.
dir	Indicates the direction of the message: send or receive.
tag	Displays an associated tag.
header	Facilitate the need to resort entries.

Column	Description
message	Displays additional information.
service Type	Displays the type of service.

Member Manager

This component provides the default view on the **IpChannel** component under a **LonIpNetwork**. It is used to manage channel member children of the **IpChannel**'s **MemberTable** component.

Figure 76 Empty Member Manager view with column menu



To access this view, expand **LonIpNetwork** and double-click the **Ip Channel** container or right-click **Ip Channel** and click **Views→Member Manager**..

If you set up this network with a third-party (non-framework) Config Server, use this table for information. This table is composed of automatically-created child ChannelMember components. No additions or edits using this view are necessary.

If you set up **LonIpNetwork** with this station acting as the Config Server, use this view to add and edit the necessary **ChannelMember** components, via the **New** and **Edit** buttons.

Column	Description
Name	Provides the name of the Channel Member Components
ipAddress	Defines the IPv4 address of the device on the LAN. Enter the unique IP address of the device. For the component that represents this station, enter the host platform's IP address.
ipPort	Defines the software port monitored for messages from the Config Server.
natIpAddress	If behind a NAT router, this property defines the device's assigned external (Internet) address. If not behind a NAT router, leave at the default value.

Column	Description
routerType	Indicates the enumerated descriptor for the router type. (Not shown as default column in the view). • Configured • Learning • Bridge • Repeater
cnFlags	Indicates the enumerated descriptor for the channel flag. (Not shown as default column in the view). • None • All Broad Casts • Security • All Broad Casts _Security
nodeType	Indicates the enumerated descriptor for the channel member type. (Not shown as default column in the view). • Uninitialized • Non Ip to Ip Router • Ip Channel Node • Ip Channel Proxy • Ip to Ip Router
iState	Indicates the enumerated descriptor for member status, as one of the following: • New Member • Sent D R Req • Sent D R • Sent C R Req• Up to Date • Error
faultCause	Displays any possible issue, such as, "Device will not ack DEVICE_ CONFIGURATION."
subnetNode	Reports the assigned Lonworks subnet/node address, unique on site for this device.
neuronId	Reports the device's unique Lonworks Neuron ID.
domainId	Reports the assigned Lonworks working domain. (Not shown as default column in the view)

Buttons

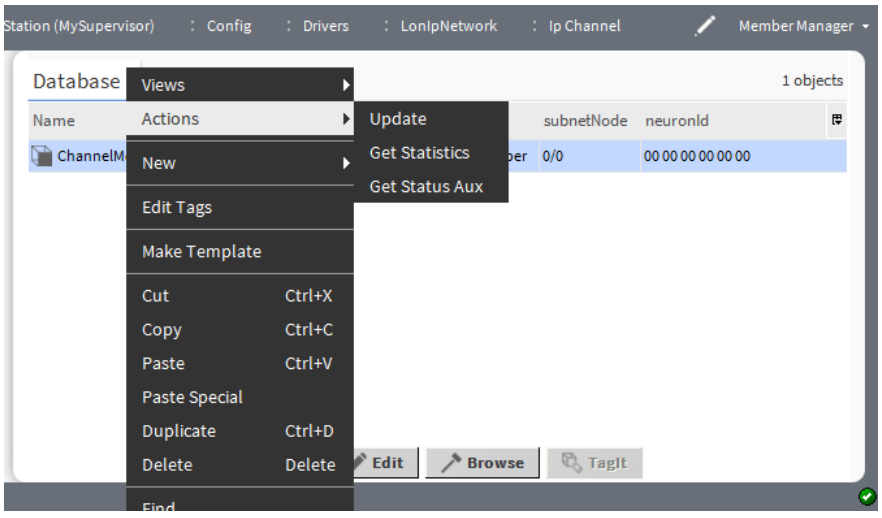
These buttons on the **Member Manager** apply if the station is set up as the CEA-852 Config Server.

- **New** creates a new device record in the database.
- **Edit** opens the device's database record for updating.
- **Browse** opens a file selector. This number references an individual network variable, which is stored in a table within the local Neuron chip.
- **TagIt** associates metadata, such as location or unique configuration with the object.

Actions

Every selected ChannelMember has an action menu.

Figure 77 Action menu for any selected IpLon ChannelMember



Action	Description
Update	Forces an update of Lon **Ip Channel** information to that device. This also occurs automatically upon station startup, and whenever Lon **Ip Channel** information changes.
Get Statistics	Produces a popup window with various statistics for the selected channel member.
Get Status Aux	Produces a popup action submenu for the selected channel member.

Chapter 10 Windows

Topics covered in this chapter

♦ New (and Edit) device window
♦ New (and Edit) local variable window
♦ Add (and Edit) points windows
♦ Lon Tunnel client window
♦ Add (and Edit) ChannelMember window
♦ Status for iLON windows
♦ Get Status Aux

Windows create and edit database records or collect information when accessing a component. You access them by dragging a component from a palette to a Nav tree node or by clicking a button.

Windows do not support **On View (F1)** and **Guide on Target** help. To learn about the information each contains, search the help system for key words.

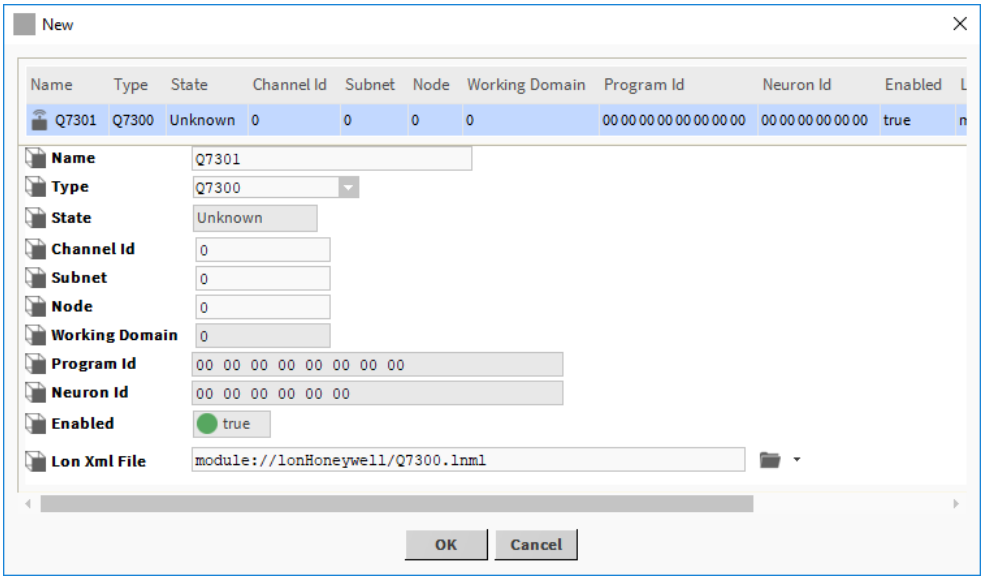
New (and Edit) device window

This window adds and edits discovered devices to the station database and configures their properties.

The driver includes components for a variety of Lon devices including:

- Dynamic
- Q7300
- Xl10Chc
- Xl10Hyd
- Powertrak9000
- Reko21002100

Figure 78 Dynamic device New properties



To open a device window, expand **Config→Drivers**, double-click **LonNetwork**, select a device row under the **Database** pane and click **Edit**.

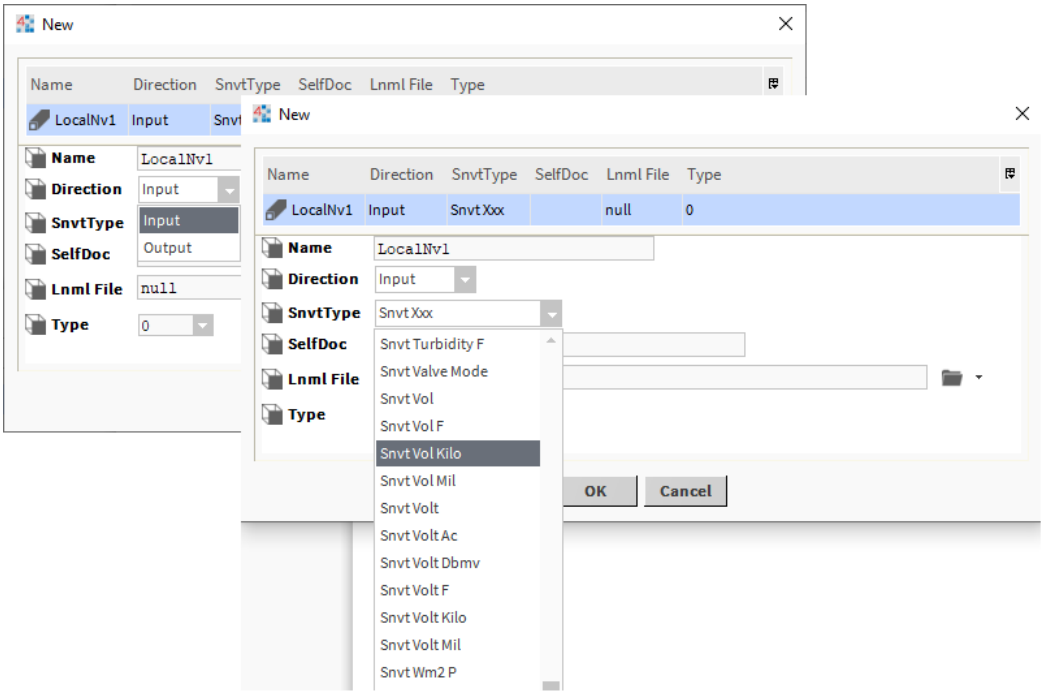
In addition to the standard property, Enabled, these properties are unique to dynamic devices.

Property	Value	Description
Name (available for dynamic devices only)	text	Provides a name for the device. This value does not affect the Lonworks node. You may change it as needed.
Type (available only when creating a new device by clicking the **New** button)	for devices: drop-down list	Identifies the type of device: Dynamic Device, Ip Local Device, Q7300, Xl10 Chc or Xl10Hyd.
State	read-only	Reports the current condition of the device.
Channel Id	number (defaults to zero (0))	Configures the Lonworks channel number. If you are configuring a third-party Config Server, use the channel number provided by the router's interface.
Subnet	number (defaults to zero (0))	Identifies the network under a channel to which the device belongs. If you are configuring a third-party Config Server, use the subnet number provided by the router's interface.
Node	number (defaults to zero (0))	Defines the node within the subnet and channel for a newly-added device. You assign this number. It should be unique for this device witin the subnet.
Working Domain	read-only (defaults to 0)	Reports the domain to which the device is connected. If a host station is operating as the network manager (typical), its Lon Netmgmt always needs access to the zero-length domain to receive service pin messages.
Program Id	read-only	
Neuron Id	read-only (hexadecimal number)	Reports a physical device number.
Lon Xml File (available for dynamic devices only)	filepath	Supports the use of the manufacturer-defined user nvs, or UNVTs (vs. SNVTs) by specifying a `.lnml` file for a device that implements the UNVTs needed. This file contains device, point and variable information. A Quik Learn automatically uses it to create Lon devices. The file extension is `.lnml`

New (and Edit) local variable window

This window adds discovered points to the station database and configures them with direction (input or output), SNVT type, and a name for each variable.

Figure 79 New variable window



To access this window, double-click the **LonNetwork** node in the Nav tree, double-click the **Local Lon Device** followed select a network variable and click the **Edit** button.

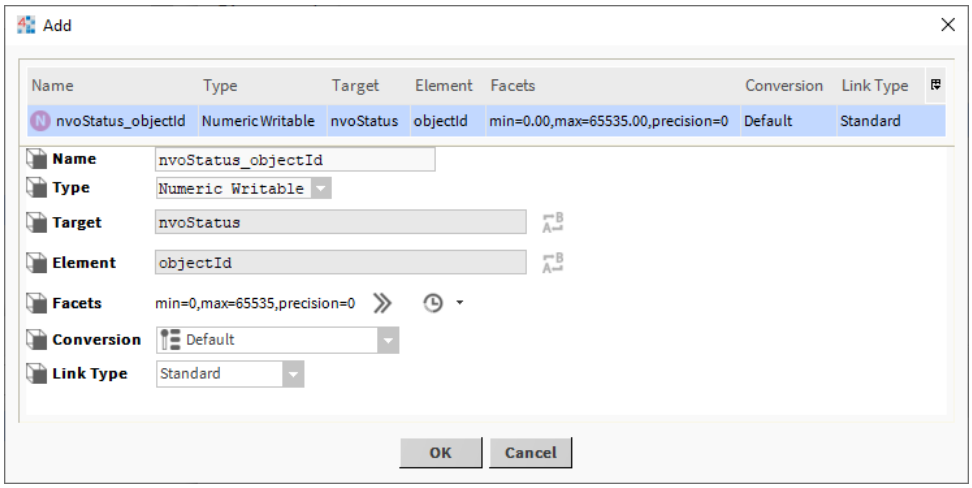
Property	Value	Description
Name	text	Provides a name for the nvi, nvo, or nci variable. If part of a data structure, it provides a trailing _elementName, which ensures a unique point name. For example, `nciTempSetPts_occupiedCool` This value does not affect the Lonworks node. You may change it as needed.
Direction	drop-down list (for a new variable defaults to Input and for a existing variable, this value is read-only)	Configures or indicates the direction of travel of the message: to the station (`Input`) or from the station `Output`.
SnvtType	drop-down list	Selects the Standard Network Variable Type (Snvt). If you are using an `.lnml` file, leave this property at its default `SNVT Xxx` value.
SelfDoc	string of text (defaults to: `&3.0@; Niagara Server Node`) This property requires **`External Config`** to be set to `true`	Defines a text string for the host node's self documentation up to 1024 bytes. A single asterisk (*) omits self documentation. **NOTE:** When using self documentation, always retain the leading header portion (`&3.0@`) along with an additional zero (`0`), plus other characters.

Property	Value	Description
Lnml File	file name	Defines the file for a device that implements the UNVT (User–defined Network Variable Type) needed. This file contains device, point and variable information. A Quik Learn automatically uses it to create Lon devices. The file extension is `.lnml`
Type	for variables: drop-down list	Selects a number for network variable type. This value works in conjunction with the **`Lnml File`** property, where, once you specify an `.lnml` file, **`Type`** provides a list of UNVTs from which you can choose this nv.

Add (and Edit) points windows

This window adds device points to the station database.

Figure 80 Add points window



You open this window after double-clicking a Lon network node in the Nav tree, double-clicking the points icon under the Exts column in a device **Database** pane, discovering points, selecting them in the points **Discovered** pane and either dragging them to the points **Database** pane or clicking the **Add** button.

In addition to the standard property (Facets) this window provides these properties.

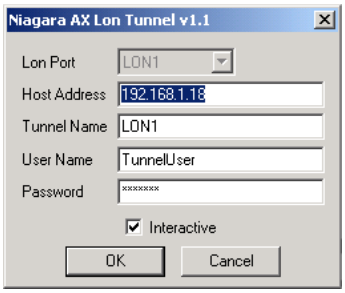
Property	Value	Description
Name	text	Provides a name for the point. This value does not affect the Lonworks node. You may change it as needed.
Type	for points: drop-down list (defaults to Numeric Writable for new points)	For new points, identifies the type of point: Numeric, Boolean, Enum, etc. Once created this value cannot be edited.
Target	read-only	Reports the Lon component represented by the proxy point, for example: nvoStatus.

Property	Value	Description
Element	read-only	If the point is structured, reports the element in the structure. If the point is unstructured, reports the SNVT.
Conversion	drop-down list (defaults to Default)	Defines how the system converts proxy extension units to parent point units. `Default` automatically converts similar units (such as Fahrenheit to Celsius) within the proxy point. **NOTE:** In most cases, the standard `Default` conversion is best. `Linear` applies to voltage input, resistive input and voltage output writable points. Works with linear-acting devices. You use the Scale and Offset properties to convert the output value to a unit other than that defined by device facets. `Linear With Unit` is an extension to the existing linear conversion property. This specifies whether the unit conversion should occur on "Device Value" or "Proxy Value". The new linear with unit convertor, will have a property to indicate whether the unit conversion should take place before or after the scale/offset conversion. `Reverse Polarity` applies only to Boolean input and relay output writable points. Reverses the logic of the hardware binary input or output. `500 Ohm Shunt` applies to voltage input points only. It reads a 4-to-20mA sensor, where the Ui input requires a 500 ohm resistor wired across (shunting) the input terminals. `Tabular Thermistor` applies to only a Thermistor input point and involves a custom resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Thermistor Type 3` applies to an Thermistor Input point, where this selection provides a "built-in" input resistance-to-temperature value response curve for Type 3 Thermistor temperature sensors. `Generic Tabular` applies to non-linear support for devices other than for thermistor temperature sensors with units in temperature. Generic Tabular uses a lookup table method similar to the "Thermistor Tabular" conversion, but without predefined output units.
Link Type	drop-down list	Configures the type of link between nodes and devices. Options are: `Unknown`, `Standard`, `Reliable`, `Critical`, `Authenticated` and `Poll Only`.

Lon Tunnel client window

After installation, access the Lon tunnel configuration window from the Windows Control Panel. All properties require a valid entry.

Figure 81 Lon Tunnel client window

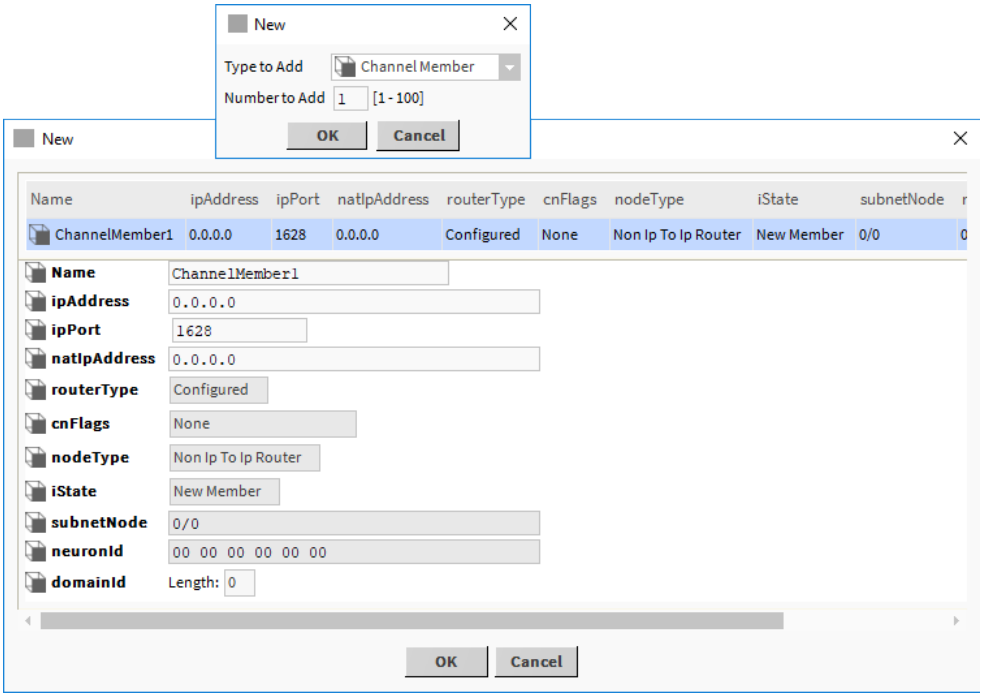


Property	Value	Description
Lon Port	drop-down list	Identifies the virtual Lon port provided by this tunnel client. This should not conflict with any existing Lon port assignment, as known to Windows, for a physical Lon adapter (for example, LON1). When you tunnel from a Lon-based Windows application, you specify this virtual Lon port.
Host Address	IP address	Identifies the IP address (or hostname) of the tunnel server. This is the target host running a station with a **LonNetwork**, **TunnelService**, and **LonTunnel**.
User Name	text	Identifies the user in the target host station, where this station user must have admin write permissions for the station's TunnelService and child LonTunnel(s). **CAUTION:** The TunnelService uses basic authentication to secure the login on any client connection (Lon tunnel or serial tunnel). To ensure a secure connection, create a special user in the station to use (only) for all Lon or serial tunnel access.
Password	text	Secures the station user. **NOTE:** The password is not encrypted then passed to the station.
Interactive	check box	Controls when this window opens. If checked, this window reopens each time a Lon-based application first opens this virtual Lon port. If cleared, this window opens only if an open fails to establish a connection to the tunnel server (as stored from last entry). You leave Interactive checked. **NOTE:** When this window opens interactively, the Lon Port setting is read-only. To change that, you must access the Lon tunneling applet from the Windows Control Panel.

Add (and Edit) ChannelMember window

This window configures new Lon **Ip Channel** members and reports on the configuration and status of the channel member.

Figure 82 New ChannelMember windows



To access this window, expand **LonIpNetwork**, double-click **Ip Channel** and click **New** or **Edit**. The read-only items in this table are available for data columns in the **Member Manager** view (only default items shown selected).

Property	Value	Description
Name (channel)	text (defaults to `ChannelMember`)	Provides the channel name. For the component that represents this station, you enter a name to match the default value of the **`Net Name`** property of the ipChannel's **Network Config** component.
ipAddress	IP address	Defines the IPv4 address of the device on the LAN. Enter the unique IP address of the device. For the component that represents this station, enter the host platform's IP address.
ipPort	number (defaults to `1628`)	Defines the software port monitored for messages from the Config Server.
natIpAddress	IP address (defaults to 0.0.0.0)	Defines the device's assigned external (Internet) address if the device is behind a NAT (Network Address Translation) router. If not, leave this property at its default value.
routerType	drop-down list (defaults to `Unknown`)	Selects the way the router handles messages. `Configured` determines which packets to forward based on internal routing tables. `Learning` forwards messages that meet forwarding rules. `Bridge` forwards all messages received regardless of the destination.

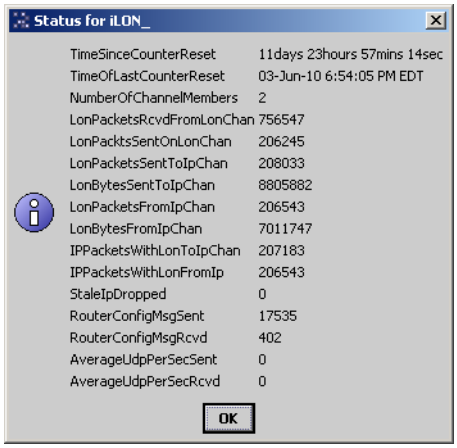
Property	Value	Description
		`Repeater` forwards all messages in both directions regardless of the message's destination or domain. `Unknown`
cnFlags	read-only	Indicates the enumerated descriptor for the channel flag. (Not shown as default column in the view): `None`, `All Broad Casts`, `Security`, `All Broad Casts_Security`
nodeType	read-only	Indicates the enumerated descriptor for the channel member type. (Not shown as default column in the view): `Uninitial-ized`, `Non Ip to Ip Router`, `Ip Channel Node`, `Ip Channel Proxy` or `Ip to Ip Router`
iState	read-only	Indicates the enumerated descriptor for member status: `New Member`, `Sent D R Req`, `Sent D R`, `Sent C R Req`, `Up to Date` or `Error`.
subnetNode	read-only	Reports the assigned Lonworks subnet/node address. Node must be unique for this device on the channel.
neuronID	read-only	Reports the device's unique Lonworks Neuron ID.
domainID	read-only	Reports the assigned Lonworks working domain. (Not shown as default column in the view).

Status for iLON windows

These windows report Lon information in response to actions.

Actions→Get Statistics status window

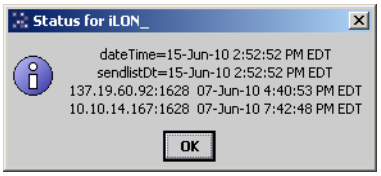
Figure 83 Status window from ChannelMember's action Get Statistics



You view this window by right-clicking a ChannelMember followed by clicking **Actions→Get Statistics**. This information may be particularly useful when troubleshooting.

Actions→Get Status Aux status window

Figure 84 Popup submenu action window from ChannelMember's action Get Status Aux

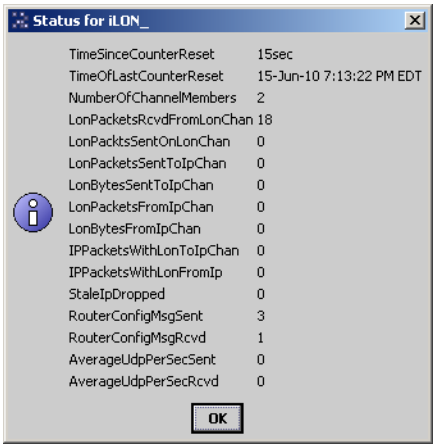


The driver pops this window up when you right-click a ChannelMember and click **Actions→Get Status Aux**. The **Get Status Aux** submenu actions include the following:

Action	Description
Status No Clear	Reads current status without resetting the statistics counters. This is equivalent to the `Get Statistics` action, with resultant popup previously.
Status With Clear	Reads current status and resets (clears) the statistics counters. The resultant popup shows many 0 entries.
Registration	Reports this member's registration information in a popup.
Membership	Reports this member's internal membership list in a popup.
Route	Reports this member's routing information in a popup.

Actions→Get Status Aux→Status With Clear status window

Figure 85 Status popup window after Status With Clear action (from Get Status Aux)

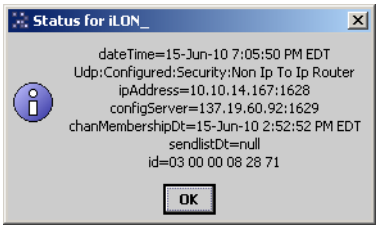


You view this window by right-clicking a ChannelMember followed by clicking **Actions→Get Status Aux→-Status With Clear**.

For information about each data item, refer to the populated version of this window earlier in this topic.

Actions→Get Status Aux→Registration window

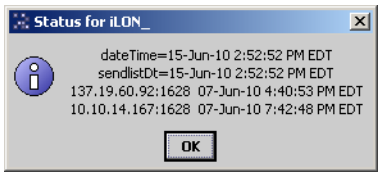
Figure 86 Status popup window from ChannelMember action Get Status Aux, Registration



You view this window by right-clicking a ChannelMember followed by clicking **Actions→Get Status Aux→Registration**.

Actions→Get Status Aux→Membership window

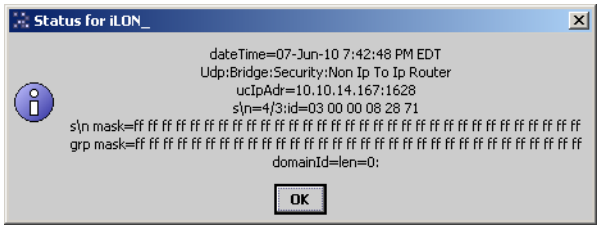
Figure 87 Status popup window from ChannelMember action Get Status Aux, Membership



You view this window by right-clicking a ChannelMember followed by clicking **Actions→Get Status Aux→Membership**.

Actions→Get Status Aux→Route window

Figure 88 Status popup window from ChannelMember action Get Status Aux, Route

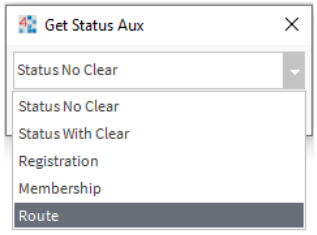


You view this window by right-clicking a ChannelMember followed by clicking **Actions→Get Status Aux→Route**.

Get Status Aux

This window configures a LonIPNetwork's channel member.

Figure 89 Get Status Aux window with options



To open this window expand **Config→Drivers→LonIpNetwork→Ip Channel→Member Table**, right-click a ChannelMember and click **Actions→Get Status Aux**.

The app provides these options:

Option	Description
Status No clear	
Status With Clear	
Registration	
Membership	
Route	

Glossary

cps	Configuration Properties are defined for each device in the device's Lon Xml file.
DIBS	Distributed Internet Backup System
Engineering workstation	An installation of Niagara on a PC, which is used to commission JACE hardware and perform application engineering on both offline and online stations. In some cases the "engineering workstation" may also be licensed to run a station to facilitate application development and testing.
LNS	L2TP Network Server. L2TP stands for Lay 2 Tunneling Protocol.
Lon component	A Lon network variable. These entities provide the basic functionality for communication in a Lon network.
LonMark	A Lon technology standard for building safe, sustainable buildings. This standard guides developers and system integrators to build robust products, design and implement successful networks to support the IoT (Internet of Things).
LonPoint	A control object with an nvo (network variable output) and a selectable SNVT type, which can be used to link to an nvi (network variable input) on one or more Lon devices.
nc, ncis	Network Configurable Inputs are typically setpoints. An nci may be part of a Px graphic.
nvis	Input Network Variables are defined for each device in the device's Lon Xml file.
nvos	Output Network Variables are defined for each device by the device's Lon Xml file.
nv, nvs	Network Variables refer, in general, to input and output pieces of information associated with each Lon device. The Lon Xml file for the device defines these unique device properties.
SNVT	Standard Network Variable Type, pronounced: "snivet" defines the Network Variable types that can be bound together. For example, a temperature variable cannot be bound to a pressure variable.
vcm	Ventilation Control Module

Index

M

N

O

P

Q

R

S

T

U

V

W

X