GRASSMARLIN User Guide

Version 3.2

Table of Contents

List of Figures

1 Product Description

GRASSMARLIN is an open-source software tool that provides a method for discovering and cataloging Supervisory Control & Data Acquisition (**SCADA**) and Industrial Control System (**ICS**) hosts on IP-based networks. GRASSMARLIN uses a variety of sources to generate this data, including PCAP files, router and switch configuration files, CAM tables, and live network packet captures. The tool can automatically determine the available networks and generate the network topology as well as visualize the communication between hosts.

GRASSMARLIN is not an analysis tool. GRASSMARLIN exists to *facilitate* further analysis by a system administrator, auditor, or other individual. The focus is not on drawing conclusions from data, but on organizing large sums of data to allow people to quickly make informed decisions.

> Throughout this guide there will be notes from the developers, formatted like this. These notes are intended to contextualize various aspects of GRASSMARLIN, to explain the history behind a feature, the direction in which it is headed, or to explain limitations or exceptions.

1.1 Versioning Schema

Releases of GRASSMARLIN are labeled with a 3-part version number and a revision. This is formally written as **Major**.**Minor**.**Patch**r**Revision**. Most often, the shortened form of **Major**.**Minor** is used.

The **Major** version number indicates the set of features which are considered to be desirable—that is, the features that intended to be present. Version 3, for example, is striving to better accommodate data sources while enabling greater analytic potential.

The **Minor** version number indicates the set of features which are actually present—features may be absent because of errors in programming, a need to gather more data before implementation, or because the developers simply need more time to produce a feature.

> The full feature set of a Major version isn't actually known when we start work on that version. We know the general goal of the 3.X codebase is to accept more forms of data and allow more interaction and organization with that data, but we can't conclusively say yet exactly how that will end up. Feedback from the 3.0 release had a significant impact on the features that went into 3.2. Due to time constraints, some features didn't make it into 3.2 and have shaped the goals of 3.3. In time, when we feel we adequately meet the 3.X goals, we will pick a direction for version 4.X.

If a critical bug is noted in a particular Minor version, then a **Patch** will be issued with the same Major and Minor version and an incremented Patch version.

We don’t release Patches often because there is a certain amount of overhead associated with preparing a release, so they are reserved for only critical issues. We are working to reduce this overhead and, as we do so, we anticipate making Patches more readily available.

The **Revision** number corresponds to the exact set of files in the master source repository that was used to build the released version. This is the preferred way of identifying the version number when reporting issues, however for the builds posted to GitHub, the Major.Minor.Patch number is all that is required.

1.2 Supported Platforms

Supported platforms are those on which every effort has been made to ensure the GRASSMARLIN application installs and executes as intended. Failure for GRASSMARLIN to start on these platforms with an out-of-the-box configuration is regarded as a bug.

- Microsoft Windows (64-bit 7, 8, and 10)
- Fedora (23)
- Ubuntu (14.04, 15.10, and Security Onion)
- Kali 2.0
- CentOS (6, 7)
- Debian (8)

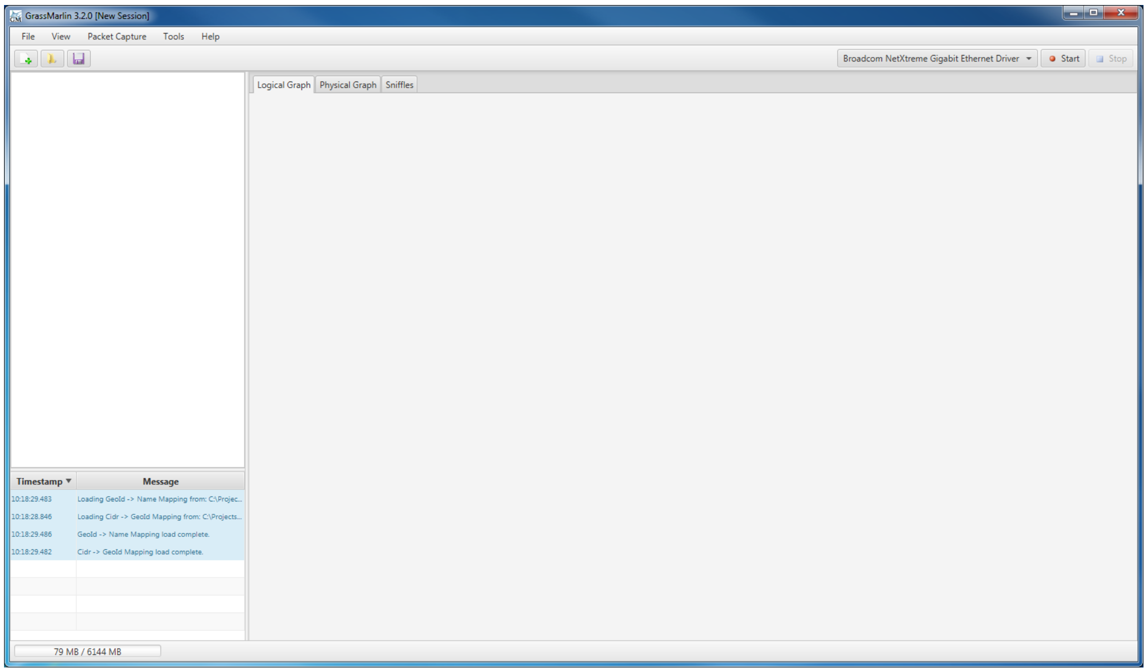
Additionally, installers are provided for additional platforms, for example 32-bit versions of Microsoft Windows. In these cases we have tested the application in these environments, however there are no assurances it will work without additional configuration. We do not formally provide support for these platforms, but we have observed working installations. Running GRASSMARLIN on these platforms may result in degraded performance and/or depend on post-install reconfiguration, but is possible.

The GRASSMARLIN code does work on Windows XP, and the Windows installers should run on Windows XP, however the typical Windows XP Java installation is not fully compatible with GRASSMARLIN, requiring some minor changes to the GRASSMARLIN batch file to compensate. The necessary changes vary by specific circumstance, but can generally be inferred from the error messages and an Internet search of Java errors. There is no support for this, but it has been made to work in specific environments.

2 GRASSMARLIN Structure

When GRASSMARLIN loads, a new **Session** is opened with no data (Figure 1). GRASSMARLIN will automatically load Plugins, perform an integrity check on the GeoIP database, load Fingerprints, and perform other initialization tasks.

In earlier versions of GRASSMARLIN, there was a progress bar along the bottom of the main window which displayed the progress for loading Fingerprints. If an Import was started before the Fingerprints finished loading, the results became unpredictable. Fingerprints are now loaded much faster; the main window will not be displayed until they have been loaded.



Timestamp	Message
10:18:29.483	Loading GeoId -> Name Mapping from: C:\Projec...
10:18:28.846	Loading Cidr -> GeoId Mapping from: C:\Projects...
10:18:29.486	GeoId -> Name Mapping load complete.
10:18:29.482	Cidr -> GeoId Mapping load complete.

Figure 1: New GRASSMARLIN Session

The main display on the right side of the window contains multiple tabs. Every **Session** contains a tab for, at a minimum, a **Logical Graph**, **Physical Graph**, and **Sniffles** (also known as the **Mesh Graph**). The content of each of these tabs is referred to as a **Visualization**.

The left side of the window is split into two sections. The lower section contains the **Message Log**, which displays status updates, warning, errors, and other messages about the activity of the GRASSMARLIN application. The upper portion contains a **Tree** that mirrors the content of the current Visualization. In Figure 1, above, the Visualization is blank, so the Tree is also blank. Changing tabs will change the content displayed in both the Tree and the Visualization. The different Visualizations are discussed in detail in Section 3.

Below the Message Log is the **Memory Indicator**. The Memory Indicator comprises two numbers and a percentage indicator. The number on the left is the current memory usage. The number to the right is the total available memory. The background behind those numbers is a percentage indicator that reports the average memory usage for the past 10 seconds.

Due to how Java manages memory and how the performance of GRASSMARLIN has been tuned, the maximum amount of memory is set when the application starts, but is not necessarily the total amount of memory available on the system. Section 7.2 contains details on how to adjust the amount of available memory.

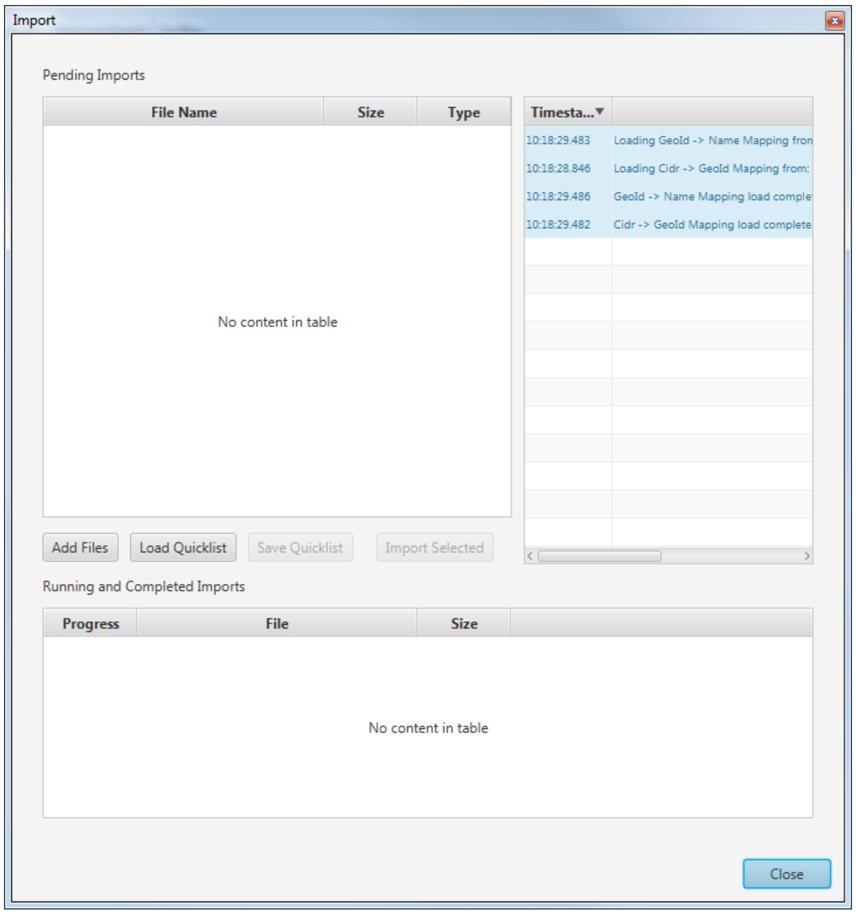
Above the main window are the Main Menu and the Toolbar. The three icons on the left side of the Toolbar are, from left to right, Import, Load Session, and Save Session. On the right side of the Toolbar is a combo box to select the Live PCAP device and to Start and Stop Live PCAP.

In addition to the main window, a **Console Window** is also opened in the background which will contain diagnostic and debugging data. This is described in more detail in Section 7.1.

2.1 Sessions

Data in GRASSMARLIN is stored in a **Session**. The Session contains imported files and visual state information.

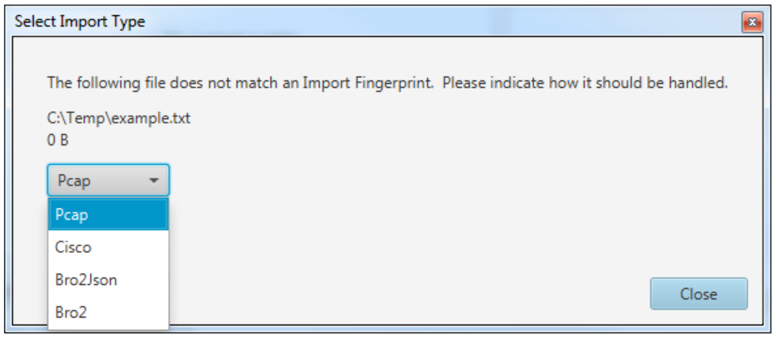
Clicking the **Import** button from the toolbar (or selecting the “Import Files…” item from the File menu) will open the **Import Dialog**.



Timesta...	
10:18:29.483	Loading GeoId -> Name Mapping fron
10:18:28.846	Loading Cidr -> GeoId Mapping from:
10:18:29.486	GeoId -> Name Mapping load comple
10:18:29.482	Cidr -> GeoId Mapping load complete

Figure 2: Import Dialog (blank)

To import a file, it must first be added to the list of **Pending Imports**. This associates the file to be imported with a particular data type. GRASSMARLIN will attempt to establish this association automatically by checking the file's extension against a list of known values. If no association can be made, the user will be prompted to specify the type.



The following file does not match an Import Fingerprint. Please indicate how it should be handled.

C:\Temp\example.txt
0 B

Figure 3: Select Import Type Dialog

The association of a Pending Import can be changed by right-clicking the row to be changed and selecting the correct type from the Context Menu.

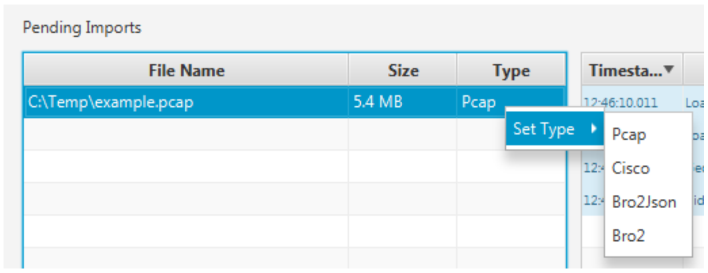


Figure 4: Change Import Type

> Most commands in GRASSMARLIN 3.2 are (or can be) accessed from a Context Menu. Context Menus are also known as right-click menus. A lot of people know what they are but aren't familiar with the name, so now you can pretend you knew what they were called right along, if you didn't already. In either case, we mention them a lot in this guide and want to clearly communicate the role they play. Also, the keyboard shortcut to display a context menu doesn't always work; that should be fixed in 3.3. Right-clicking should always work. We don't support single-button mice.

Files can be added to the Pending Imports list through the **Add Files** and **Load Quicklist** buttons. Quicklists are files which contain a set of files with their associated import types. To create a Quicklist, select one or more Pending Import items and use the **Save Quicklist** button. To remove a file from the Pending Imports list, select the row for that file and press the Delete key.

To import files into a session, select the corresponding Pending Import items and then click the **Import Selected** button.

The other component of a Session, the visual state information, is the current state of all the Visualizations, including the position of nodes, collapsed status of groups, color assignments, and more. When a Session is saved and later loaded, the restored session should look identical to how it was at the time it was saved.

> Saved sessions do not include the position and zoom level of the viewport, nor are window sizes saved. The position of nodes is saved, and the camera will be zoomed to fit the displayed content after loading.

2.1.1 Saved Sessions

The save format used in GRASSMARLIN 3.2 is incompatible with that used by earlier versions. A Session saved in 3.2 cannot be loaded in earlier versions, nor can a Session saved in 3.1 be opened in 3.2. Starting with 3.2, future versions should be able to load sessions from earlier versions.

> In GRASSMARLIN 3.0 and 3.1, Sessions were saved by exporting the data. As of this writing, work has begun, but not concluded, on a plugin to import 3.0 and 3.1 sessions into 3.2. It is anticipated that this plugin will be made available at the same time as version 3.3.

2.1.2 Import Types

Included with GRASSMARLIN are parsers for the following formats:

- PCAP (Including PCAPNG)
- Bro2Conn
- Bro2Conn (JSON)
- Cisco Config

This list may be expanded through the use of Plugins. Plugins are custom Java code which is designed to add additional functionality to GRASSMARLIN. A sample plugin which adds support for importing host data from a CSV format is available, however it is not intended for actual use; it is provided primarily as an example on which new plugins can be based. Pcap and PcapNg are supported through the iadgov.offlinepcap plugin, while the other supported formats are built-in to the GRASSMARLIN application.

> Plugins are new to 3.2. In 3.2, all plugins can do is add new Import Types. Moving forward, we expect to expand on the range of functionality plugins can provide, but this was one of the most frequently requested features. For more details on Plugins, see Section 8.1.

2.2 Live PCAP

GRASSMARLIN includes the capability to passively listen on a network. By design, GRASSMARLIN has no active network components.

> Plugins can change this, so be aware of the capabilities of any plugins you are using. Also, it is possible to access network shares from the Save and Load dialog boxes; these will generate network traffic appropriate to your OS and the type of share. While GRASSMARLIN is designed to only passively observe network communication, it is possible for a user to cause network traffic to be created. Exercise caution when operating in an environment where generating any network traffic is undesirable.

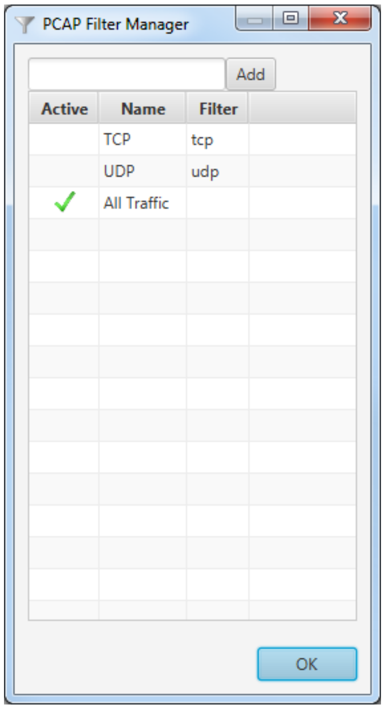
To begin **Live PCAP**, first identify the network device on which you desire to listen. All available devices are listed in the combo box on the right side of the Toolbar. When the desired device is selected, Live PCAP can be started by clicking either the Start button to the right of the combo box, or from the **Packet Capture -> Start Live PCAP** menu item.

While Live PCAP is running, the **Start** button will be disabled and the **Stop** button will be enabled (as will the corresponding menu items in the Packet Capture menu). Both the Start and Stop buttons will be disabled if there is a problem initializing the JNetPcap Library. JNetPcap is a third-party library that is used to gather the PCAP data. Consult the Troubleshooting section (Section 7) if there are any issues with Live PCAP.

Under the **Packet Capture** menu are three other items of note.

2.2.1 Pcap Filter Manager

The **Pcap Filter Manager** allows a filter to be applied to packets in Live PCAP streams before they are processed by GRASSMARLIN.



Active	Name	Filter
	TCP	tcp
	UDP	udp
✓	All Traffic	

Figure 5: PCAP Filter Manager Dialog

To set the active filter, double-click the desired row in the **Active** column. The active filter is denoted with a green check mark.

A Filter can be renamed or modified by clicking the corresponding column in the desired row, then editing the selected text.

To add a new Filter, enter the filter text and click Add or press Enter.

To delete a filter, select the row for the filter and press the Delete key.

When adding and editing a Filter, only valid TCPDUMP filters are permitted.

2.2.2 Open Capture Folder

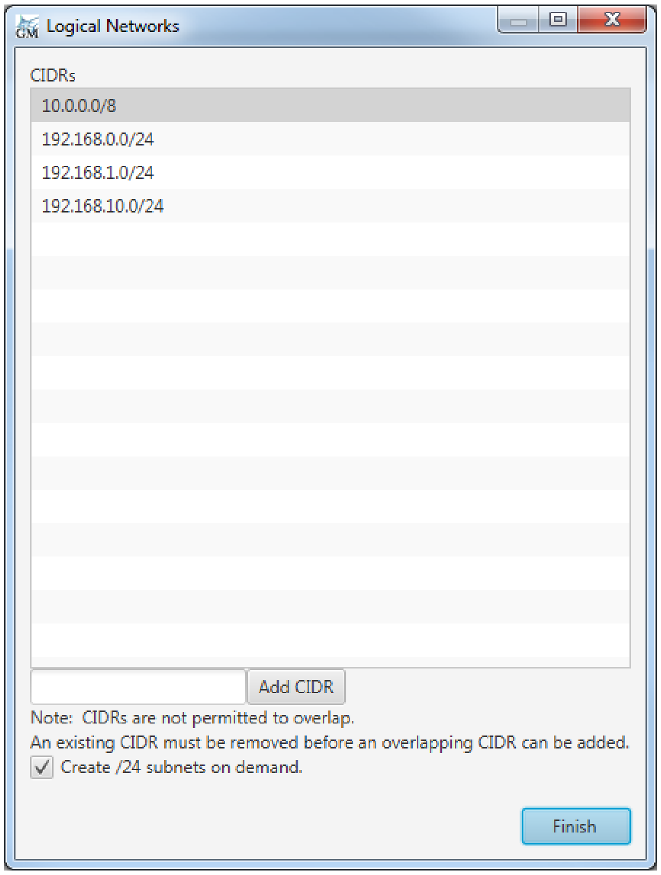
Whenever Live Pcap is gathered, it will be written to disk in a specific capture folder. The **Open Capture Folder** menu item will attempt to open that folder for browsing in your desktop environment.

When Live Pcap is performed, an import referencing this file is generated and associated with the current Session.

The capture file will contain only packets which passed the Pcap Filter.

2.2.3 Manage Networks

As packet metadata is Imported into GRASSMARLIN, new hosts will be identified. These hosts are identified by IP Addresses and will be assigned to Networks. Networks are groups that are defined by a CIDR. The **Manage Networks** menu item will open a dialog that allows for manual manipulation of the list of Networks, including the ability to enable or disable the automatic creation of new Networks as necessary.



10.0.0.0/8
192.168.0.0/24
192.168.1.0/24
192.168.10.0/24

Note: CIDRs are not permitted to overlap.
An existing CIDR must be removed before an overlapping CIDR can be added.
✓ Create /24 subnets on demand.

Figure 6: Logical Networks Dialog

By default, new networks will be created for each distinct /24 CIDR that is identified in packet metadata. The size of the CIDR range can be configured from the **Tools -> Preferences** menu item. The option to create Networks automatically and the size of the Network to create are application-level settings—that is they are saved automatically and loaded every time the application starts.

A CIDR will not be created if it overlaps an existing CIDR. To remove an existing CIDR select it from the list and press the delete key. A new CIDR can be added by entering the CIDR in the text field and pressing Enter or clicking Add CIDR. If the text is not a valid CIDR an error will be generated. If it overlaps an existing CIDR the field will be cleared but no error will be generated.

It is also possible to add or remove Networks from the Context Menu of the Logical Graph (See Section 3.3.1). When using the context menu, rather than the full Logical Networks Dialog, an abbreviated interface is used instead.

2.3 Cisco Config Files

The Physical Graph is populated from **Cisco Config Files**. Plugins may add support for additional formats, but this is the only natively supported format. Each file that is imported as a Cisco Config File is expected to contain the results of the following commands:

```
SHOW VERSION
SHOW RUNNING CONFIG
SHOW INTERFACES
```

And one of the following:

```
SHOW MAC
```

```
SHOW ARP
```

The choice between MAC and ARP depends on the type of device. The output of all four commands should be placed in a single text file and then Imported. It is inadvisable to place the results of commands from multiple devices in a single file; the exact behavior for that situation is undefined, but it will most likely confuse the association between certain commands and the device to which they correspond. The log may contain the echoed commands, but should not contain other text.

2.4 Graphs

Files that are Imported into GRASSMARLIN can contain several types of data. This data will be reflected on at least one of the three primary tabs. These tabs are described as graphs because they are graphs in the mathematical sense, meaning they are composed of a series of **Nodes** (or Vertices, depending on which reference you use) which are connected by **Edges**. Additionally, every Node has Properties. These Properties are a combination of Source, Name, Value, and Confidence.

> Although Node is the less common term, Vertex becomes ambiguous when working with polygons and other graphical elements. To avoid confusion, we favor Node over Vertex in this guide.

2.4.1 Logical Graph

The **Logical Graph** shows Nodes for distinct IP addresses with Edges representing packets sent between them. This graph is built from packet metadata, normally provided through Pcap or Bro2Conn files.

Additional detail for Nodes can be derived from **Fingerprints**, which operate on both packet metadata as well as packet contents. Plugins are also capable of providing additional detail.

> Most Fingerprints rely on packet contents, and so they will not match Bro2Conn or other formats that solely provide metadata.

Additional tabs can be created which display a subset of the Logical Graph. **Watch Tabs** will show a selected Node from the Logical Graph and all other Nodes that connect to that Node by traversing at most a given number of Edges (1-10, default 1). **Filter Views** will display a copy of the Logical Graph, but allow for individual Nodes (and Edges involving those Nodes) to be hidden.

2.4.2 Physical Graph

The **Physical Graph** contains Nodes for each distinct MAC address. These Nodes are grouped by the device which owns that MAC. **Routers** and **Switches** are the obvious devices, but MACs which do not belong to any device are associated with a **Workstation** device. "Workstation" is slightly misleading, as it simply refers to a port belonging to an otherwise unknown device. For simplicity, the term Device refers to Routers and Switches only; Workstations are excluded.

The Edges of the Physical Graph represent physical connections between MACs.

In addition to Workstations, Routers, and Switches, the Physical Graph also has Cloud Nodes. **Clouds** represent unknown regions of the network topology. Often they represent Routers and Switches for which the configuration files were not available, but they may also indicate a mis-configured device or other problem. Clouds are displayed as a pair of connected Nodes, one of which connects to one or more Devices, the other connecting to zero or more Workstations.

> Enhancing the mapping capabilities of GRASSMARLIN is a huge focus in present development. In testing practical situations, we have identified several scenarios where the data looks identical but the correct topology is known to be different. Manual control, deeper data, and more robust mapping capabilities are therefore on the list of features we want to include in GRASSMARLIN 3.3.

2.4.3 Sniffles

The **Sniffles** tab houses the **Mesh Graph** and is built from 802.15.4 data extracted from PCAP. Each node identifies a device on a Mesh Network and groups are drawn for each identified **Personal Area Network** (PAN).

> Gathering data for Mesh Networks is a lot more complicated than gathering data on wired IP-based networks. This data is often gathered by protocol-specific wireless sniffers, which led to the tab being affectionately called "Sniffles" in earlier releases. The more accurate title of "Mesh Graph" hasn't stuck, so it has remained Sniffles.

2.5 Properties

At the most basic, **Properties** are simply **Name-Value** pairs that are associated with a Node. Because it is possible for multiple **Sources** to each provide a Value for a given Name, every time a Property is added, it is associated with the Source providing that Name-Value pair. As many of the methods of building these associations are prone to error, a **Confidence** of 1 (high) to 5 (low) is also associated with each Property. Properties are then listed in the format of "Source.Name" having a value of "Value (Confidence)".

Additionally, aggregated Properties are computed, where the Value for a given Name is the set of distinct Values for that Name at the highest Confidence. For example, if a Fingerprint named "A" sets the “Model” property of a Node to a value of “Gruntmaster9000” with a confidence of 3 and another Fingerprint, “B”, also sets the “Model” property, but sets it to “Acorn” with a confidence of 2, then the details for that Node will show the properties of “A.Model”, “B.Model”, and “Model” with values of “Gruntmaster9000”, “Acorn” and “Acorn”, respectively. If both A and B reported the same confidence, then the “Model” property would contain both “Gruntmaster9000” and “Acorn”.

2.5.1 Intrinsic Properties

Some Properties have neither a Source nor a Confidence; these are referred to as **Intrinsic Properties**. These Properties are applied to all Nodes of a Graph that are of a certain type.

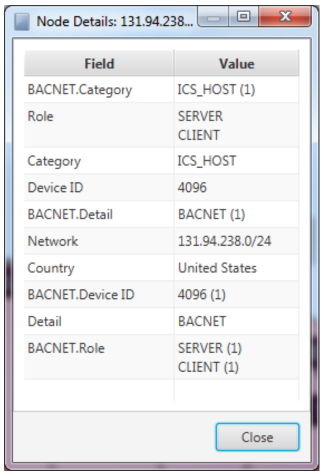
In the Logical Graph, every Node has both a **Network** and a **Country**. The Network will be a CIDR from the Logical Networks Dialog (Section 2.2.3). If the CIDR containing a Node’s IP is deleted (or automatic creation is not enabled) then there will be no Network associated with the Node, but the Node still has a Network property—it is just blank. The Country is derived from a GeoIp database that is included with GRASSMARLIN. This database resolves CIDR ranges to Country Codes which are in turn converted to Country Names. Not every IP is associated with a Country Name, but, as with Networks, the lack of a Country Name simply means the Country property contains a blank name—the Country Property is still present.

On the Physical Graph, every Node has an **Owner**, and every Node that is not part of a cloud also has a **MAC Address**. The Owner is the name of the device which contains the Node; generally a switch or router, but also possibly a Workstation. Clouds, as they are placeholders for unknown infrastructure, lack MAC Addresses, but every other Node of the Physical Graph will have one.

On the Sniffles Graph, every Node has a **PAN ID**.

2.5.2 Viewing Properties

To view the Properties associated with a Node, open the Context Menu of the Node and select the “View Details for (Name)…” menu item. This will open a dialog similar to Figure 7, below. The Fields that are present depend on the type of Node, which in turn depends (in part) on the Graph that is being viewed. Additionally, specific fields may be added as part of the Import process or as part of Fingerprinting, meaning that the fields displayed can vary greatly between Nodes; not all Nodes will have the same set of Fields.



Field	Value
BACNET.Category	ICS_HOST (1)
Role	SERVER CLIENT
Category	ICS_HOST
Device ID	4096
BACNET.Detail	BACNET (1)
Network	131.94.238.0/24
Country	United States
BACNET.Device ID	4096 (1)
Detail	BACNET
BACNET.Role	SERVER (1) CLIENT (1)

Figure 7: Node Details Dialog

3 Visualizations

The **Visualizations** on each tab have many commonalities, but there are also many features that are unique to specific types of data. All Visualizations have a context menu. The contents of this menu depend on the current Visualization, but also on what is near the cursor at the time the menu is displayed.

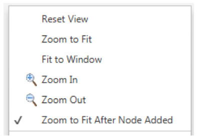


Figure 8: Common Context Menu Elements

All Visualizations start with the same six items:

Reset View

This resets the zoom level to 100% and positions the origin of the Visualization in the top-left corner. This is the default setting for a new Session.

Zoom to fit

This will adjust the zoom level and pan the Visualization so that the entire Graph is visible within the Window. It will not reposition individual Nodes.

Fit to Window

This will reposition the Nodes of the graph, ignoring the aspect ratio, so that the entire Graph fits in the current viewport.

The Fit to Window command is unavailable on the Physical Graph in 3.2. There was a bug with this command that could not be resolved prior to the 3.2 release, so it was removed from the interface and will be restored as soon as the bug can be remedied.

Zoom In

This will zoom in by a single step. Each step represents 10% magnification relative to the previous step. The maximum zoom level is step 48, which roughly corresponds to a 100x zoom.

Zoom Out

This will reduce the zoom by a single step. The minimum zoom is step -48, which roughly corresponds to a 0.01x zoom.

Zoom to Fit After Node Added

If this is checked, then whenever a new Node is added to the Graph, a Zoom to Fit will be executed.

Another common aspect to all Visualizations is the **Layout**. The Layout is an algorithm that determines where each Node should be placed. The algorithms used by the different Visualizations are themselves different, but the automatic positioning of new nodes is an aspect common to all.

3.1 Navigation

In addition to the options available in the Context Menu, each Visualization can be navigated using the mouse.

Pan the Visualization by dragging the mouse with the left-button pressed.

Zoom in and out using the scroll wheel.

Reposition Nodes by dragging them with the left mouse button pressed. Dragging a group will reposition the members of the dragged group.

Some types of Groups (Devices and Workstations on the Physical Graph) automatically position the Nodes they contain; dragging a Node in one of these Groups will drag the Group instead.

Groups that do not automatically position their elements can be expanded and contracted by holding the Ctrl button and scrolling the mouse wheel while the cursor is over the group. This will move the members of the group towards or away from the center of the group.

This only works if the mouse is "over the Group". It does not work if the mouse is over a Node within a Group. If the border of the Group is highlighted, then GRASSMARLIN considers the mouse to be over that Group.

3.2 Aggregates

Nodes are often drawn as belonging to a **Group**. The visualization of this Group is called an **Aggregate**. On the Logical Graph, Aggregates might represent, among other possibilities, a **Network**, **Country**, or **Model**, while on the Physical Graph an Aggregate represents a **Device** or **VLan**.

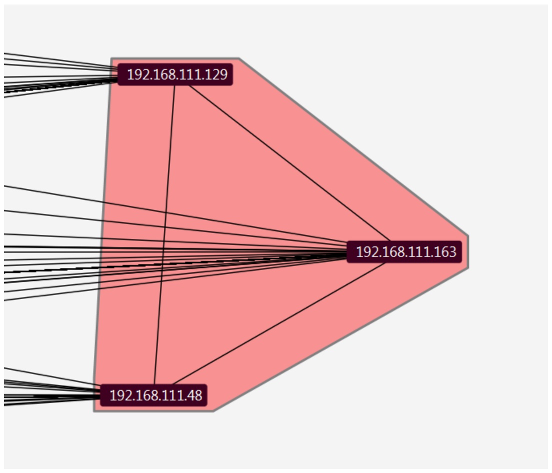


Figure 9: Logical Graph Aggregate containing 3 Nodes

On the different Visualizations, Aggregates provide different functionality, but the concept of Aggregates is universal across all Visualizations.

3.3 Logical Graph

As described in Section 2.4.1, the Logical Graph depicts the contents of an IP-based network and the data flowing between any given pair of Nodes. This flow of data does not represent the routing of data, merely the source/destination.

A Node is created for each distinct IP. At this time, no effort is made to distinguish between multiple endpoints that share an IP address. If automatic creation of Networks is enabled (See Section 2.2.3) then when a new IP is detected, it is checked for membership in an existing Network. If it does not match a defined Network, a new Network is created and added to the Network list. If the Network that would be created conflicts with existing Networks, then it is ignored instead.

3.3.1 Context Menu

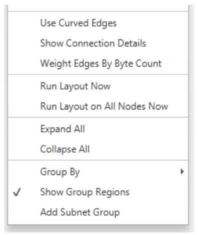


Figure 10: Logical Graph Context Menu Items

The Context Menu for the Logical Graph features several commands to modify the Visualization.

Use Curved Edges

While checked, instead of using straight lines to depict Edges, cubic Bezier curves will be used instead. Drawing curved edges is much more computationally intense than straight lines, so performance may suffer while this is enabled.

Show Connection Details

While checked, some text will be drawn near the midpoint of each edge. This text will indicate the number of bytes sent to each endpoint of the Edge and the list of Protocols used for that communication.

Weight Edges By Byte Count

While checked, the thickness of the lines drawn for Edges will be based on the amount of traffic between the endpoints of that Edge. A linear increase in thickness corresponds to an exponential increase in traffic.

Run Layout Now

This command will re-run the Layout algorithm on all Nodes that have not been manually positioned.

Run Layout on All Nodes Now

This will mark all Nodes as not being manually positioned, then re-run the Layout. The result is that every node will be subjected to this (and all subsequent) Layout operations.

Expand All

This will uncheck the “Collapse Group” menu item for each Group.

Collapse All

This will check the “Collapse Group” menu item for each Group.

Group By

This allows the Grouping criteria to be changed. By default, Groups are defined by the Network to which a Node belongs. The Grouping criteria can be changed to any Property that has been set on any Node.

Show Group Regions

While checked, the Aggregates for Groups will be drawn.

Add Subnet Group

Open a dialog which allows the user to enter a new Network to add to the Network List (See Section 2.2.3).

If the Context Menu is opened over an Aggregate (to include opening it over a Node belonging to an Aggregate), there will be additional commands. If there are multiple overlapping Aggregates, these items will be presented for each such Aggregate in a submenu bearing the name of the Group the Aggregate identifies.

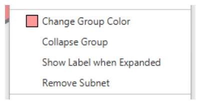


Figure 11: Logical Graph Group Context Menu Items

Change Group Color

Change the fill color of the Aggregate. Because the Aggregates are drawn with a translucent background the Aggregate will tend to look slightly lighter than the selected color.

> The default color is actually pure, bright red, but it appears very close to Salmon Pink when viewed over the white background at 40% opacity. Presently there is no way to adjust the opacity.

Collapse Group

When checked, the contents of this Group and the Aggregate will not be drawn, Edges to the members of the group will be redirected to the Label, and the Label will be visible.

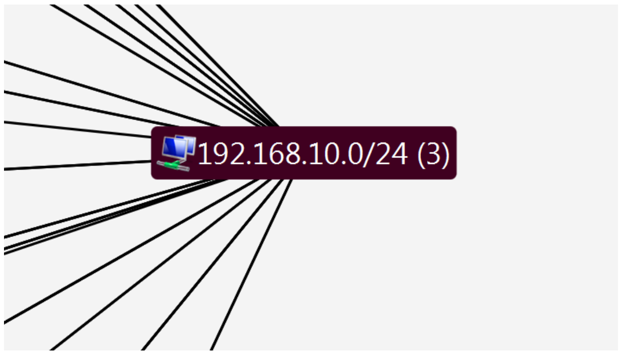


Figure 12: Collapsed Group

Show Label when Expanded

By default the Group Label is hidden unless the Group is Collapsed. If this is checked the Label will always be displayed at the computed center of the group (which is computed as the midpoint of the smallest rectangle that contains all the members of the group).

Remove Subnet

This will remove the Network associated with this Group from the Network List (See Section 2.2.3). This menu item will only be present when Group By is set to Network.

If the Context Menu is opened over a Node, the following commands will be present. As with Aggregates, overlapping Nodes will produce multiple menu items, each bearing the name of a Node and these items are repeated for each.

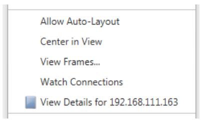


Figure 13: Logical Graph Node Context Menu Items

Allow Auto-Layout

While checked, this Node will be repositioned whenever the Layout is run. This is checked by default and automatically unchecked if the Node is repositioned, either directly or by repositioning a Group containing the Node.

Center in View

This command will center the Visualization on this Node, without adjusting Zoom or positioning of any Nodes.

View Frames…

This will open the Connections Dialog described in Section 3.3.4.

Watch Connections

This will create a new Watch Graph, as described in Section 3.3.6.

View Details For…

This will display the Node Details Dialog, as described in Section 3.3.5.

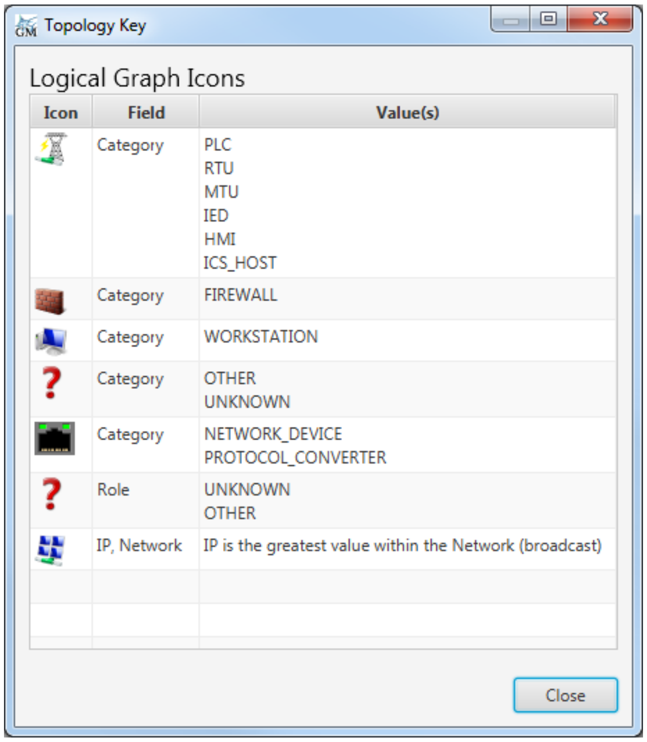
3.3.2 Layout

The Logical Graph organizes Nodes using a force-directed graph model. This treats each group as a single entity and calculates attractive and repulsive forces based on which Nodes are connected and their relative sizes. The Nodes of a Group are then organized around the edge of a circle centered on the calculated Group location. The Layout attempts to position each Node so that it is closest to the other Nodes to which it connects.

This Layout works best to display several Groups of varying sizes, which is a typical scenario for Logical Graph data.

3.3.3 Icons

The Properties of a Node on the Logical Graph can cause it to display one or more **Icons**. GRASSMARLIN includes a large set of country flags that will be displayed based on the value of the "Country" Property. In addition to the country flags, several other Properties can trigger the display of specific Icons. These can be viewed in the **Topology Key**, which is accessible from the **Help -> Topology Key** menu item.



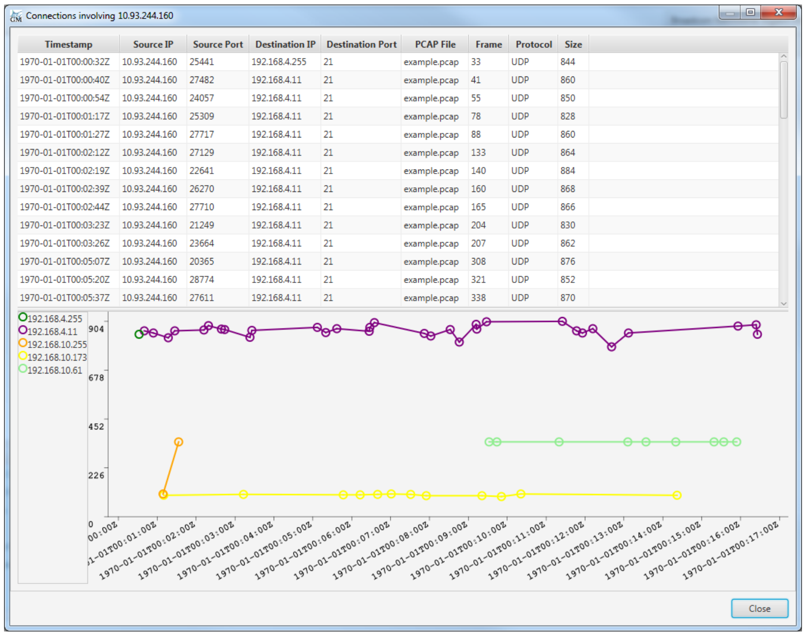
Logical Graph Icons

Icon	Field	Value(s)
	Category	PLC RTU MTU IED HMI ICS_HOST
	Category	FIREWALL
	Category	WORKSTATION
?	Category	OTHER UNKNOWN
	Category	NETWORK_DEVICE PROTOCOL_CONVERTER
?	Role	UNKNOWN OTHER
	IP, Network	IP is the greatest value within the Network (broadcast)

Figure 14: Topology Key Dialog

3.3.4 Connections Dialog

Selecting the View Frames… menu item on a Node will display the Connections Dialog. This dialog will display a list and chart of all the packets that have been sent or received by the selected Node.


Connections involving 10.93.244.160

Timestamp	Source IP	Source Port	Destination IP	Destination Port	PCAP File	Frame	Protocol	Size
1970-01-01T00:00:32Z	10.93.244.160	25441	192.168.4.255	21	example.pcap	33	UDP	844
1970-01-01T00:00:40Z	10.93.244.160	27482	192.168.4.11	21	example.pcap	41	UDP	860
1970-01-01T00:00:54Z	10.93.244.160	24057	192.168.4.11	21	example.pcap	55	UDP	850
1970-01-01T00:01:17Z	10.93.244.160	25309	192.168.4.11	21	example.pcap	78	UDP	828
1970-01-01T00:01:27Z	10.93.244.160	27717	192.168.4.11	21	example.pcap	88	UDP	860
1970-01-01T00:02:12Z	10.93.244.160	27129	192.168.4.11	21	example.pcap	133	UDP	864
1970-01-01T00:02:19Z	10.93.244.160	22641	192.168.4.11	21	example.pcap	140	UDP	884
1970-01-01T00:02:39Z	10.93.244.160	26270	192.168.4.11	21	example.pcap	160	UDP	868
1970-01-01T00:02:44Z	10.93.244.160	27710	192.168.4.11	21	example.pcap	165	UDP	866
1970-01-01T00:03:23Z	10.93.244.160	21249	192.168.4.11	21	example.pcap	204	UDP	830
1970-01-01T00:03:26Z	10.93.244.160	23664	192.168.4.11	21	example.pcap	207	UDP	862
1970-01-01T00:05:07Z	10.93.244.160	20365	192.168.4.11	21	example.pcap	308	UDP	876
1970-01-01T00:05:20Z	10.93.244.160	28774	192.168.4.11	21	example.pcap	321	UDP	852
1970-01-01T00:05:37Z	10.93.244.160	27611	192.168.4.11	21	example.pcap	338	UDP	870

Figure 15: Connections Dialog

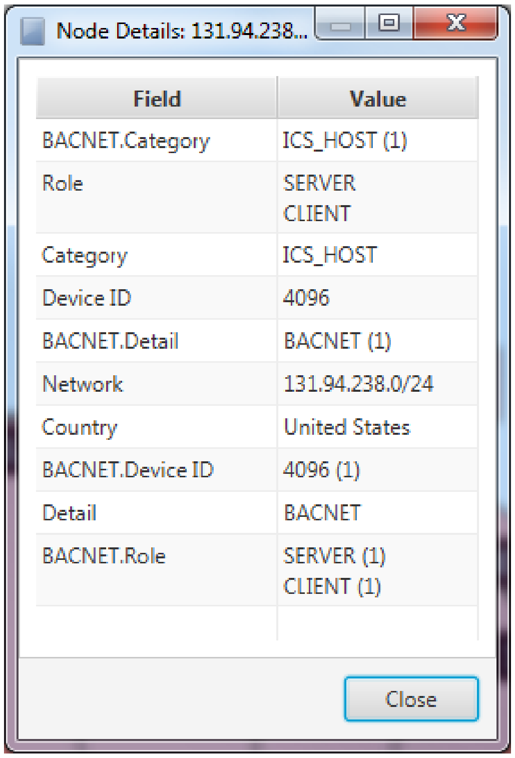
This dialog can use up a large amount of memory when displaying a Node involved in a large number of packets. Don't be surprised if it takes several seconds for the dialog to display for a busy Node. The most-recently viewed data is cached, so if you close the dialog and re-open the same Node, it will load much faster the second time, but if you open the Connections Dialog for a different Node, it will clear the cached data.

The table at the top displays the metadata for each packet to or from the selected Node. The table can be sorted by clicking on the column headers. The row for a packet that was imported from a PCAP file has a context menu with a single option to open Wireshark with that Import file and to browse to the selected packet.

The chart on the bottom depicts packet size by remote endpoint and time. Each series represents a remote endpoint, and the data for that series is all data sent or received between that Node and the Node for which the connections are being displayed. The direction of traffic is not reflected here. The context menu for the chart allows series to be hidden, or for opening specific packets in Wireshark. The context menu presents options to open Wireshark for packets which are near the cursor, with the closest packets listed first. To zoom in on a section of the chart, click and drag to highlight the section to view; when the mouse is released the chart will zoom to the highlighted region. To cancel a zoom or to restore the default zoom settings, use the corresponding commands in the chart's context menu.

3.3.5 Node Details Dialog

As described in Section 2.5.2, the Node Details Dialog displays the Properties associated with a Node. For the Logical Graph, this includes values reported from Fingerprinting. In Figure 16, the selected Node matched the BACNET Fingerprint, which reported a Category, Detail, Device ID, and two Roles. Clicking on the Field column header will cause the contents to be sorted by the Field name, which will group the Fields by Fingerprint.

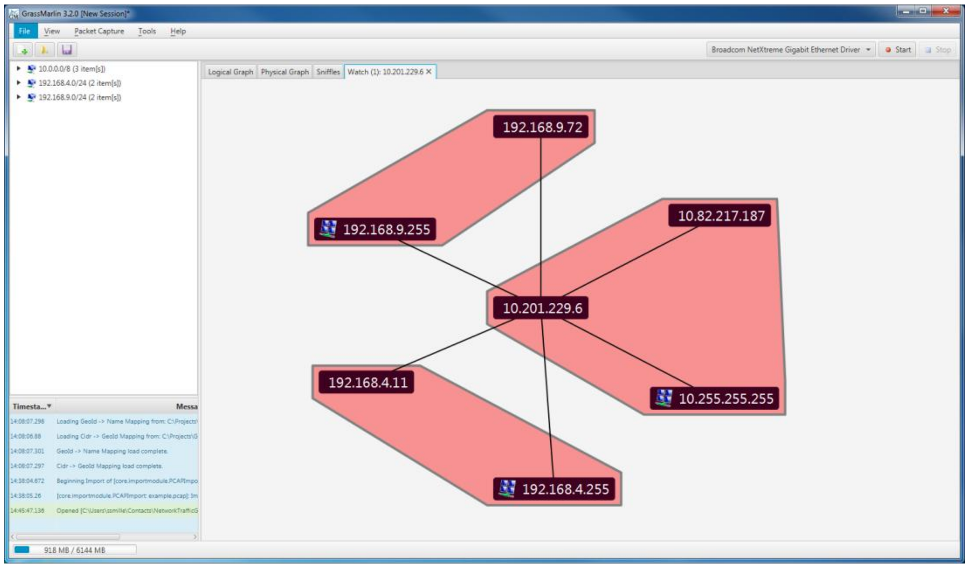

Node Details: 131.94.238...

Field	Value
BACNET.Category	ICS_HOST (1)
Role	SERVER CLIENT
Category	ICS_HOST
Device ID	4096
BACNET.Detail	BACNET (1)
Network	131.94.238.0/24
Country	United States
BACNET.Device ID	4096 (1)
Detail	BACNET
BACNET.Role	SERVER (1) CLIENT (1)

Figure 16: Node Details Dialog for Logical Graph

3.3.6 Watch Graphs

Watch Graphs are new tabs containing a specialized subset of the Logical Graph. A Watch Graph is created for a particular Node and initially contains only that Node and the Nodes that are connected to it, and is created using the **Watch Connections** context menu item on a Node in the Logical Graph.



```
Timesta...▼  Messa
14:08:07.298  Loading GeoId -> Name Mapping from: C:\Projects\
14:08:06.88   Loading Cidr -> GeoId Mapping from: C:\Projects\G
14:08:07.301  GeoId -> Name Mapping load complete.
14:08:07.297  Cidr -> GeoId Mapping load complete.
14:38:04.872  Beginning Import of [core.importmodule.PCAPImpo
14:38:05.26   [core.importmodule.PCAPImport: example.pcap]: Im
14:45:47.136  Opened [C:\Users\ssmille\Contacts\NetworkTrafficG
```

Figure 17: Watch Window

The context menu for a Watch Graph is almost identical to that for the Logical Graph—it has all the same items (except the ability to create a new Watch Graph) plus one extra. The **Set Watch Degrees** item changes the filter to permit Nodes up to the given number of steps away from the initial Node. Setting the Watch Degrees to 2, for example, will display the Node being watched, all nodes which connect to the watched Node, and all Nodes which connect to those Nodes. The Watched Node is positioned in the center and the other Nodes form concentric circles around it, with the radius of the circle proportional to the number of degrees of separation.

The Layout doesn't do a particularly good job of placing members of the same Group near each other, especially across different rings. The Show Group Regions menu item was added to allow Groups to be hidden on Watch Windows—it works on other Visualizations, but this was the feature that really called for it.

3.3.7 Filter Views

Filter Views can be created with the **View -> New Filtered Logical View** menu item. These Visualizations are copies of the Logical View where individual Nodes can be hidden. This is accomplished with the Hide Node context menu item. Once a Node has been Hidden, it can be restored using the corresponding sub-item of the Unhide Nodes context menu item.

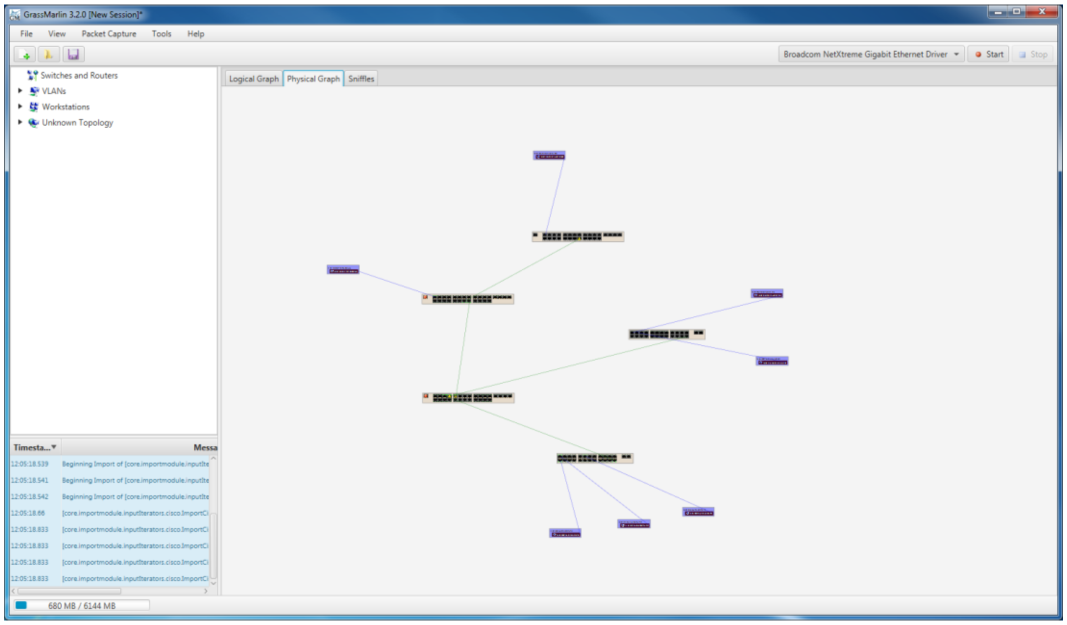
The Layout for a Filter View copies the location of the corresponding Node on the Logical Graph. As long as the Auto-Layout option is selected for a Filter View Node, any change to the corresponding Node's position on the Logical Graph Visualization will be copied to the Filter View. Changes to Groups (Collapsed, Color) are not mirrored, although dragging a collapsed group on the Logical Graph will move the Group's members, and this movement will be reflected on the Filter View.

3.4 Physical Graph

The Physical Graph depicts the physical network infrastructure, including managed Switches and Routers, the connections between them, and the physical workstations. This data is derived from the configuration data for managed devices, as described in Section 2.3.

On the Physical Graph, a Node is created for each MAC Address. Nodes are then organized by the device to which they belong. A MAC Address belonging to a managed device will be represented by a Port of that device, whereas a MAC which does not belong to a managed device will be represented as a NIC of unknown design, referred to as a Workstation. Edges represent direct, physical connections. Due to the nature of configuration files, the connections between Nodes are not known for certain, but inferred. Often data will suggest that a port connects to multiple devices, which would be a physical impossibility. To resolve this, Nodes belonging to a Cloud are created, where the Nodes represent not a MAC address but unknown infrastructure. A Cloud always contains two Nodes—each connected to the other, and one connected to all managed devices while the other is connected to all workstations.

The Physical Graph can only be defined from imported configuration files which contain descriptions for managed devices. The properties of these devices are then used to infer the network topology. There is no way to directly define the topology. In future versions, in addition to supporting more data formats, it is our goal to allow the calculated graph to be manipulated—adding, modifying, and removing connections and devices to allow the displayed model to be an accurate reflection of the physical network. Presently, to manipulate the graph in this manner it must be exported to SVG and then modified as an image. Modifications cannot be re-imported to GRASSMARLIN. Alternatively, a Saved Session can be manually modified, which would allow GRASSMARLIN to work with the modified data. This is a shortcoming in 3.2 and more advanced Physical Graph features should be coming in future versions.



```
Timesta...▼    Messa
12:05:18.539   Beginning Import of [core.importmodule.inputIte
12:05:18.541   Beginning Import of [core.importmodule.inputIte
12:05:18.542   Beginning Import of [core.importmodule.inputIte
12:05:18.66    [core.importmodule.inputIterators.cisco.ImportCi
12:05:18.833   [core.importmodule.inputIterators.cisco.ImportCi
12:05:18.833   [core.importmodule.inputIterators.cisco.ImportCi
12:05:18.833   [core.importmodule.inputIterators.cisco.ImportCi
12:05:18.833   [core.importmodule.inputIterators.cisco.ImportCi
680 MB / 6144 MB
```

Figure 18: Physical Graph Visualization

3.4.1 Context Menu

These Menu Items are available in the Context Menu of the Physical Graph regardless of the content below the mouse:

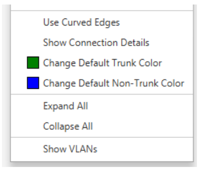


Figure 19: Physical Graph Context Menu

Use Curved Edges

Similar to the same menu item on the Logical Graph, this toggles between straight and curved lines. The curves on the Physical Graph are designed to more closely resemble cables with regards to how the curve is plotted.

Show Connection Details

For the Physical Graph, the Connection Details are the list of VLANs associated with a particular connection. Aside from that, this behaves similarly to the corresponding menu item on the Logical Graph.

Change Default Trunk Color

A connection between ports which are marked as Trunk Ports is a Trunk Connection, and the default color for these connections is defined by this item (default Dark Green). Changing the Default will update existing connections that have not been individually customized.

Change Default Non-Trunk Color

This behaves the same as the previous item, but for connections that are not between Trunk Ports.

Expand All

This functions identically to the corresponding Menu Item in the Logical Graph; it expands all Devices to display all Ports.

Collapse All

This functions identically to the corresponding Menu Item in the Logical Graph; it collapses all Devices, hiding all Ports.

Show VLANs

Toggle the display of aggregates for VLANs.

When the mouse is over a Device, to include over a Port belonging to a Device, the following options are present.

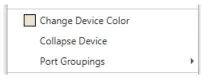


Figure 20: Physical Graph Device Context Menu

Change Device Color

This option allows the background color of the device to be changed. As is done for other Aggregates, the selected color will be applied with a degree of translucency applied.

Collapse Device

This reduces the Device to a single label, much as collapsing a Group on the Logical Graph does.

There is a bug where the edges are drawn over the collapsed label; this should be fixed in GRASSMARLIN 3.3.

Port Groupings

A Device contains one or more Port Groups. A Port Group is a collection of one or more Ports which belong to a specific interface. Each Port Group can be visually represented using one of several different patterns, and the pattern to use for each group can be customized here.

The rendering options for Port Groups were intended to mimic the physical arrangements of Ports. In many cases the end result is similar but not-quite-right. Plugin support for new types of Port Groupings is in development, but it is unknown with which version it will be released.

When the Context Menu is opened over a Port, these items will be present.

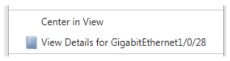


Figure 21: Physical Graph Port Context Menu

Center in View

This pans the display to center the selected Port in the Viewport. It does not affect the zoom level or positioning of any Nodes.

View Details for Port

This command displays the Node Details Dialog for the selected Port. See Section 3.4.4 for details regarding this dialog and its contents.

In addition to the above options that are available for Ports, if the cursor is over a Node that has one or more connections, the following options will be present for each connection.

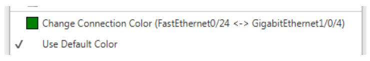

Change Connection Color (FastEthernet0/24 <-> GigabitEthernet1/0/4)
Use Default Color

Figure 22: Physical Graph Edge Context Menu

Change Connection Color

This command is used to assign a non-default color to this Connection. When set through this command, the Use Default Color option will automatically be unchecked.

Use Default Color

When checked, this assigns the configured default color to this Connection. If this is a Trunk Connection, it will use the Default Trunk Connection Color, otherwise it will use the default Non-Trunk Connection Color.

A NIC is any MAC address which appears in the routing information but is not associated with a managed network device. Each NIC presents the following context menu options:

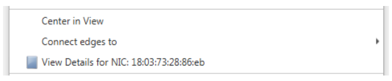


Figure 23: Physical Graph NIC Context Menu

Center in View

Center the display on this NIC without adjusting the position of elements or the zoom level.

Connect edges to

The Physical Graph is expected to be used to diagram the physical components of a network. To this end, the edges that connect to a NIC can connect to the midpoint of any side of the NIC label, or to the center of the label, to better present a clean, descriptive representation. When curved edges are used, this also influences the control points for the curve.

View Details for NIC

Display the details for the selected NIC.

Each NIC initially belongs to a workstation. A NIC may only belong to a single workstation, but a workstation may contain multiple NICs. The following context menu items are available when the context menu is opened over a workstation, including when it is opened over a NIC (since each NIC must belong to a workstation).

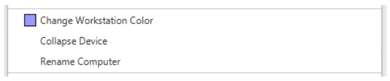


Figure 24: Physical Graph Workstation Context Menu

Change Workstation Color

Change the color of the aggregate drawn for this Workstation. The rendered color will be the selected color at 40% opacity, as is done for other aggregates.

Collapse Device

Toggle hiding the individual NIC(s) that belong to this Workstation, rendering a single label for the entire Workstation.

Rename Computer

Change the name of the Workstation. If two Workstations have the same name, they are merged into a single Workstation.

When a Workstation is renamed, the old Workstation is deleted and a new one created. A custom color will be lost and the new Workstation positioned at a default location. If the new name matches an existing group, then the members of the old Workstation will be merged into the existing group.

3.4.2 Layout

The Physical Graph plots all Devices and Clouds equidistant around the perimeter of a circle. Workstations connected to each of these entities are drawn in an arc, further from the center of the circle.

3.4.3 Icons

On the Physical Graph, Ports are rendered with several overlapping icons. Each icon indicates specific properties of the Port.

Connected

If a managed Port is known to be connected, then the two green lights on the Port icon will be lit. The lights are on either the top or bottom of the icon, depending on the port orientation.

Disabled

If a managed Port is known to be disabled, it will have an icon depicting a red circle with a horizontal white line in it. It is still possible for these ports to be Connected.

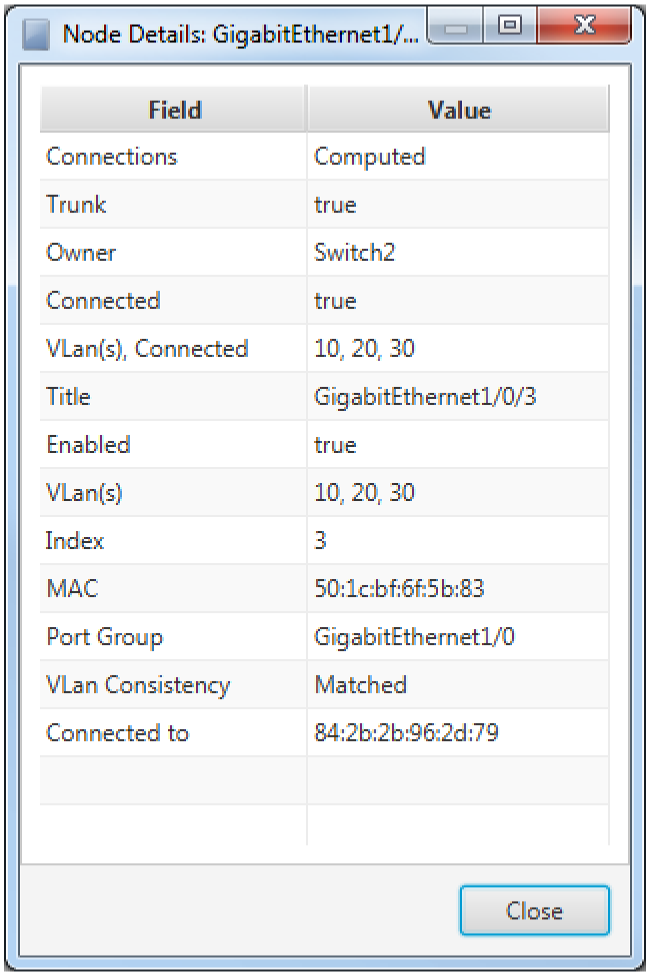
Inconsistent

A managed Port that has an icon on it depicting a yellow triangle with a black exclamation point in it is Inconsistent. There are several possible causes for this state, including:

- The VLANs assigned to this Port do not match the VLAN assignments of a Port that is directly connected to it.
- A Port is marked as a Trunk but the Port to which it is connected is not (or vice versa).
- There is an inconsistency in the topology data and this Port connects to an unknown device.

If a Port is functioning as a Trunk Port but is not actually marked as such (or vice versa), then this sort of "unknown device" error can happen.

3.4.4 Node Details



Field	Value
Connections	Computed
Trunk	true
Owner	Switch2
Connected	true
VLan(s), Connected	10, 20, 30
Title	GigabitEthernet1/0/3
Enabled	true
VLan(s)	10, 20, 30
Index	3
MAC	50:1c:bf:6f:5b:83
Port Group	GigabitEthernet1/0
VLan Consistency	Matched
Connected to	84:2b:2b:96:2d:79

Figure 25: Node Details Dialog for a Physical Graph Port

The Properties reported for a Port are defined as follows:

Connections

The Connections Property will have a value of "Unknown" if it connects to a device which was not defined. This tends to happen when the Trunk field is not properly set, but it can arise from other forms of data corruption as well.

Trunk

The Trunk Property will be set to "true" or "false"; by default it is "false", being assigned a value of "true" if the configuration file indicates this is a Trunk Port.

Owner

The Value of this Property is the name of the Device to which this Port belongs.

Connected

The Connected Property is assigned a value of "true" if the Port has a physical connection, "false" otherwise.

VLan(s), Connected

This Property contains a comma-separated list of VLans which can send traffic to this Port. If no VLans have explicitly been defined, then the default VLan (1) is set.

Title

This Property contains the full name assigned to this Port in the configuration file.

Enabled

This Property is assigned a value of "true" by default and, if the port is flagged as disabled, it will be assigned a value of "false".

> There is a lot of variation in configuration files and they don't always parse correctly. The default-true behavior is called out here because if there is a parse error (to include new file formats or simply a bug in the parsing logic), the default value of "true" is likely to prevail.

VLan(s)

This Property contains a comma-separated list of VLans assigned to this Port.

Index

This Property contains the numeric identifier for this Port within its Port Group.

MAC

This Property contains the MAC Address associated with this Port.

> MAC Spoofing will almost certainly cause significant problems when computing the network topology.

Port Group

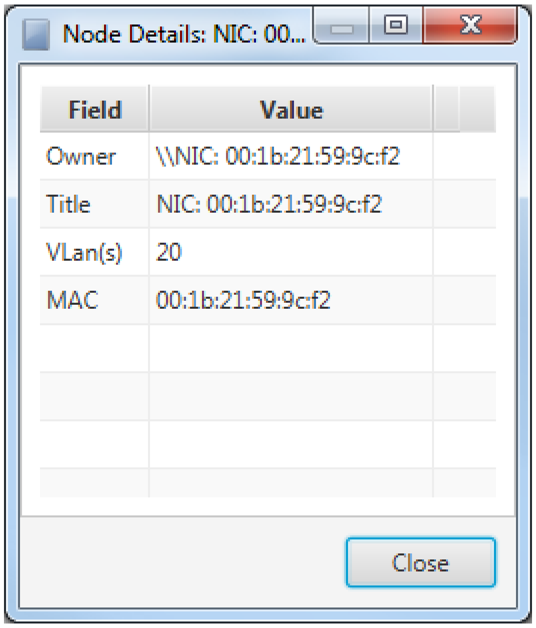
This Property contains the name of the Port Group to which this Port belongs.

VLan Consistency

If the "VLan(s)" and "VLan(s), Connected" Properties differ, then this will be set to "Mismatched", otherwise it is set to "Matched". If this is "Mismatched", then the Port will be marked as Inconsistent.

Connected to

This Property contains a list of MAC Addresses to which this Port is connected. If the Port is flagged as a Trunk Port, then only Ports belonging to other managed network devices will be included. If it is not a Trunk Port, then MAC Addresses belonging to other managed network devices are excluded. When the topology mapper is unable to resolve a section of topology, this Property will contain a best-guess and may reference MAC Addresses belonging to other managed devices as well as endpoints of unknown/unmanaged devices.



Field	Value
Owner	\\NIC: 00:1b:21:59:9c:f2
Title	NIC: 00:1b:21:59:9c:f2
VLan(s)	20
MAC	00:1b:21:59:9c:f2

Figure 26: Node Details Dialog for a Physical Graph NIC

The Node Details for a NIC are considerably simpler than those for a Port.

Owner

This Property contains the name of the Workstation that owns this NIC. By default, it is "\\" followed by the NIC's Title Property. This can be changed through the Workstation's context menu (Section 3.4.1).

Title

This Property contains the name assigned to this NIC; presently it is hard coded to "NIC: " followed by the MAC Address.

VLan(s)

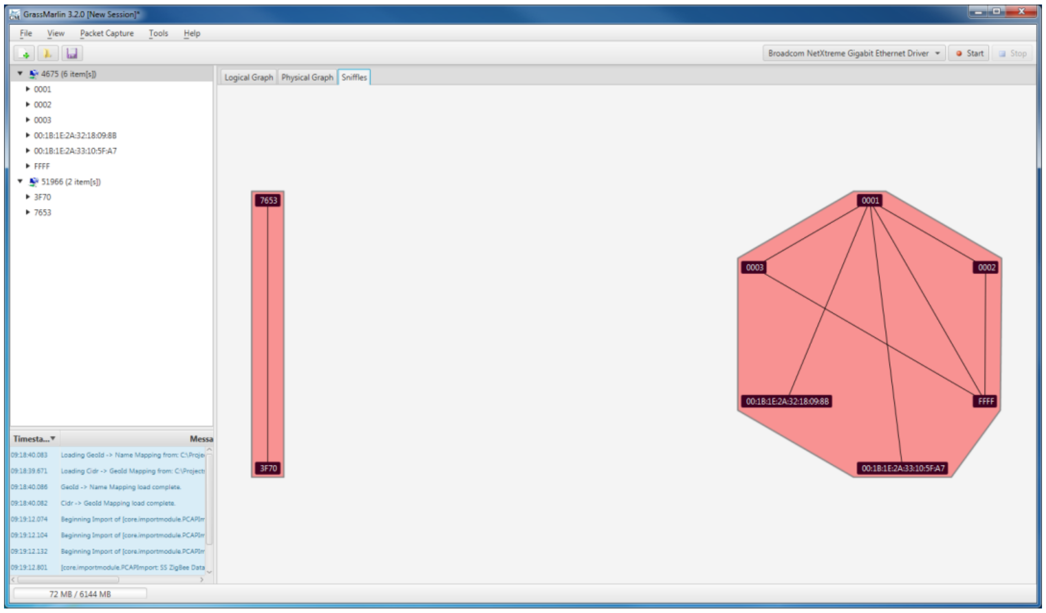
This Property contains a comma-separated list of VLANs associated with the Port(s) to which this NIC is connected.

MAC

Each NIC on the Physical Graph is created because something referenced a MAC Address; they are not explicitly declared. Every unique MAC calls for a new NIC Node, and this is the MAC Address for which this Node was created.

3.5 Sniffles

The Sniffles tab contains data for Mesh Networks, depicting wireless devices grouped by PAN, as determined from IEEE802.15.4 packet data. This packet data is wrapped in a ZigBee UDP header to allow GRASSMARLIN to parse the data.



```
4675 (6 item[s])
  0001
  0002
  0003
  00:1B:1E:2A:32:18:09:8B
  00:1B:1E:2A:33:10:5F:A7
  FFFF
51966 (2 item[s])
  3F70
  7653
```

```
Timesta...    Messa
09:18:40.083  Loading GeoId -> Name Mapping from: C:\Proje
09:18:39.671  Loading Cidr -> GeoId Mapping from: C:\Project
09:18:40.086  GeoId -> Name Mapping load complete.
09:18:40.082  Cidr -> GeoId Mapping load complete.
09:19:12.074  Beginning Import of [core.importmodule.PCAPIm
09:19:12.104  Beginning Import of [core.importmodule.PCAPIm
09:19:12.132  Beginning Import of [core.importmodule.PCAPIm
09:19:12.801  [core.importmodule.PCAPImport: SS ZigBee Data
```

Figure 27: Sniffles Visualization

In GRASSMARLIN 3.3, we anticipate the ability to parse the 802.15.4 data natively, removing the need for the ZigBee headers. We also hope to incorporate this data directly into the Logical Graph.

With respect to context menus and other controls, the Mesh Graph behaves almost identically to the Logical Graph. See Section 3.3.1 for details on the context menu commands. The Mesh Graph does not make use of Icons, Watch Windows, or Filter Views.

4 Fingerprint Manager

The fingerprint manager shows all the fingerprints that are loaded and provides the ability to disable and enable them (All fingerprints are enabled by default). The fingerprint manager also provides the ability to edit and create new fingerprints. To open the fingerprint manager click on “Tools” then “Fingerprint Manager”.

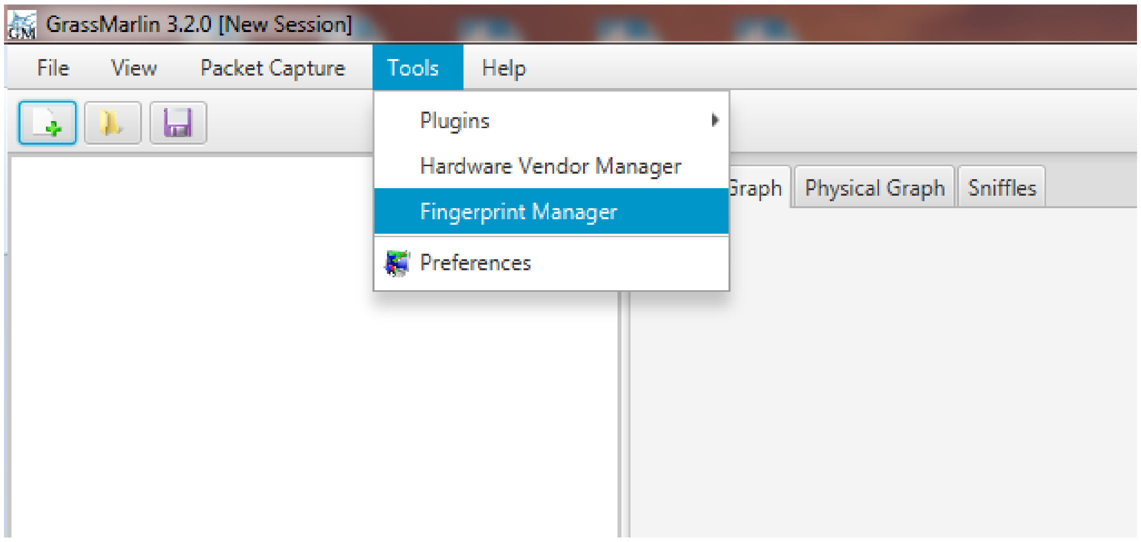
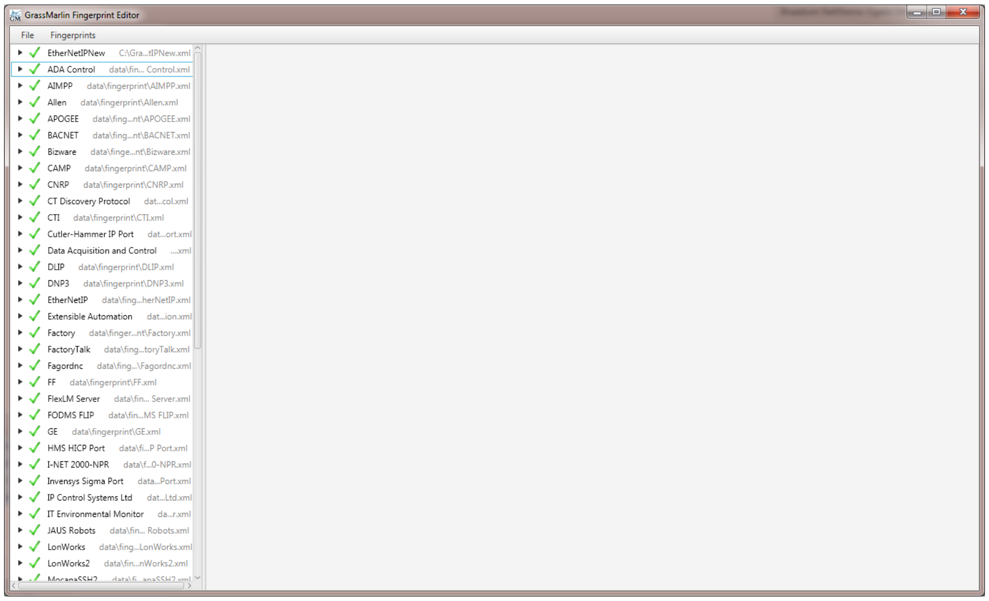


Figure 28: Opening Fingerprint Manager

4.1 Fingerprint Basics

When you first open the Fingerprint Manager, you will see a list of the loaded fingerprints along with a green check next to those that are enabled.



✓	EtherNetIPNew	C:\Gra...tIPNew.xml
✓	ADA Control	data\fin... Control.xml
✓	AIMPP	data\fingerprint\AIMPP.xml
✓	Allen	data\fingerprint\Allen.xml
✓	APOGEE	data\fing...nt\APOGEE.xml
✓	BACNET	data\fing...nt\BACNET.xml
✓	Bizware	data\finge...nt\Bizware.xml
✓	CAMP	data\fingerprint\CAMP.xml
✓	CNRP	data\fingerprint\CNRP.xml
✓	CT Discovery Protocol	dat...col.xml
✓	CTI	data\fingerprint\CTI.xml
✓	Cutler-Hammer IP Port	dat...ort.xml
✓	Data Acquisition and Control	....xml
✓	DLIP	data\fingerprint\DLIP.xml
✓	DNP3	data\fingerprint\DNP3.xml
✓	EtherNetIP	data\fing...herNetIP.xml
✓	Extensible Automation	dat...ion.xml
✓	Factory	data\finger...nt\Factory.xml
✓	FactoryTalk	data\fing...toryTalk.xml
✓	Fagordnc	data\fing...\Fagordnc.xml
✓	FF	data\fingerprint\FF.xml
✓	FlexLM Server	data\fin... Server.xml
✓	FODMS FLIP	data\fin...MS FLIP.xml
✓	GE	data\fingerprint\GE.xml
✓	HMS HICP Port	data\fi...P Port.xml
✓	I-NET 2000-NPR	data\f...0-NPR.xml
✓	Invensys Sigma Port	data...Port.xml
✓	IP Control Systems Ltd	dat...Ltd.xml
✓	IT Environmental Monitor	da...r.xml
✓	JAUS Robots	data\fin... Robots.xml
✓	LonWorks	data\fing...LonWorks.xml
✓	LonWorks2	data\fin...nWorks2.xml
✓	MocanaSSH2	data\fi...anaSSH2.xml

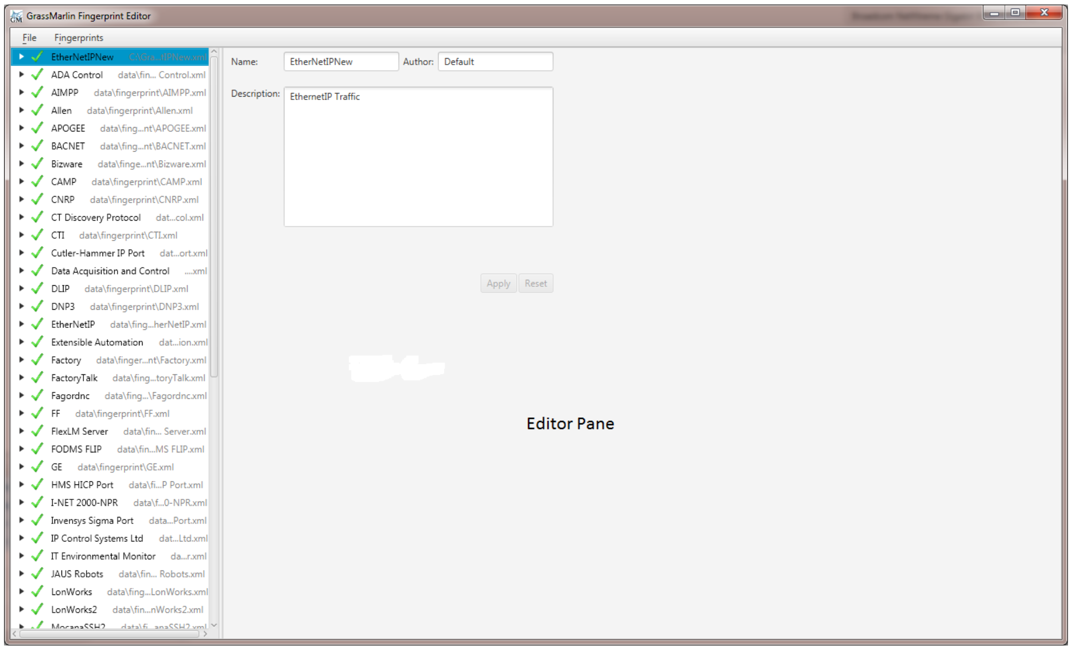
Figure 29: Fingerprint Manager

4.2 Using the Manager

The "Fingerprints" menu provides the options for enabling and disabling fingerprints along with their respective keyboard shortcuts. When a new import is run, only those fingerprints that are enabled are used to process the data. In order for newly enabled fingerprints to be used, the import must be run again.

4.3 Editing Fingerprints

Selecting a fingerprint opens it in the Editor Pane.



✓	EtherNetIPNew	C:\Gra...tIPNew.xml
✓	ADA Control	data\fin... Control.xml
✓	AIMPP	data\fingerprint\AIMPP.xml
✓	Allen	data\fingerprint\Allen.xml
✓	APOGEE	data\fing...nt\APOGEE.xml
✓	BACNET	data\fing...nt\BACNET.xml
✓	Bizware	data\finge...nt\Bizware.xml
✓	CAMP	data\fingerprint\CAMP.xml
✓	CNRP	data\fingerprint\CNRP.xml
✓	CT Discovery Protocol	dat...col.xml
✓	CTI	data\fingerprint\CTI.xml
✓	Cutler-Hammer IP Port	dat...ort.xml
✓	Data Acquisition and Control	....xml
✓	DLIP	data\fingerprint\DLIP.xml
✓	DNP3	data\fingerprint\DNP3.xml
✓	EtherNetIP	data\fing...herNetIP.xml
✓	Extensible Automation	dat...ion.xml
✓	Factory	data\finger...nt\Factory.xml
✓	FactoryTalk	data\fing...toryTalk.xml
✓	Fagordnc	data\fing...\Fagordnc.xml
✓	FF	data\fingerprint\FF.xml
✓	FlexLM Server	data\fin... Server.xml
✓	FODMS FLIP	data\fin...MS FLIP.xml
✓	GE	data\fingerprint\GE.xml
✓	HMS HICP Port	data\fi...P Port.xml
✓	I-NET 2000-NPR	data\f...0-NPR.xml
✓	Invensys Sigma Port	data...Port.xml
✓	IP Control Systems Ltd	dat...Ltd.xml
✓	IT Environmental Monitor	da...r.xml
✓	JAUS Robots	data\fin... Robots.xml
✓	LonWorks	data\fing...LonWorks.xml
✓	LonWorks2	data\fin...nWorks2.xml
✓	MocanaSSH2	data\fi...anaSSH2.xml

Name: EtherNetIPNew
Author: Default
Description: EthernetIP Traffic

Figure 30: Fingerprint open in Editor

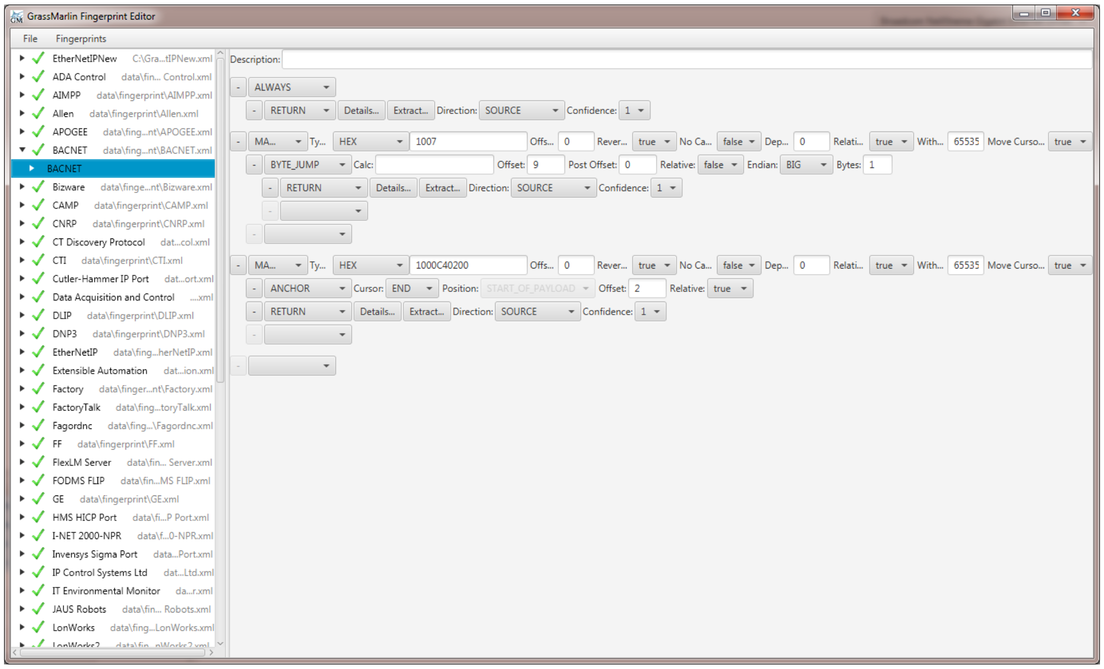
Clicking on the currently selected item allows for the editing of its name. All other changes are made through the Editor Pane. A change to a fingerprint is not "live" until the fingerprint has been saved.

4.3.1 Fingerprint Info

When a fingerprint is selected the basic fingerprint information is displayed in the Editor Pane. This is the fingerprint's name, author, and description.

4.3.2 Payload Processors

If you expand the fingerprint by clicking on the little black arrow next to its name, the next level down is the payload processors associated with that fingerprint. Clicking on a payload processor opens it for editing.



```
GrassMarlin Fingerprint Editor
File   Fingerprints

EtherNetIPNew                 C:\Gra...tIPNew.xml
ADA Control                   data\fin... Control.xml
AIMPP                         data\fingerprint\AIMPP.xml
Allen                         data\fingerprint\Allen.xml
APOGEE                        data\fing...nt\APOGEE.xml
BACNET                        data\fing...nt\BACNET.xml
  BACNET
Bizware                       data\finge...nt\Bizware.xml
CAMP                          data\fingerprint\CAMP.xml
CNRP                          data\fingerprint\CNRP.xml
CT Discovery Protocol         dat...col.xml
CTI                           data\fingerprint\CTI.xml
Cutler-Hammer IP Port         dat...ort.xml
Data Acquisition and Control  ....xml
DLIP                          data\fingerprint\DLIP.xml
DNP3                          data\fingerprint\DNP3.xml
EtherNetIP                    data\fing...herNetIP.xml
Extensible Automation         dat...ion.xml
Factory                       data\finger...nt\Factory.xml
FactoryTalk                   data\fing...toryTalk.xml
Fagordnc                      data\fing...\Fagordnc.xml
FF                            data\fingerprint\FF.xml
FlexLM Server                 data\fin... Server.xml
FODMS FLIP                    data\fin...MS FLIP.xml
GE                            data\fingerprint\GE.xml
HMS HICP Port                 data\fi...P Port.xml
I-NET 2000-NPR                data\f...0-NPR.xml
Invensys Sigma Port           data...Port.xml
IP Control Systems Ltd        dat...Ltd.xml
IT Environmental Monitor      da...r.xml
JAUS Robots                   data\fin... Robots.xml
LonWorks                      data\fing...LonWorks.xml

Description:
- ALWAYS
  - RETURN  Details...  Extract...  Direction: SOURCE  Confidence: 1
- MA...  Ty... HEX  1007  Offs... 0  Rever... true  No Ca... false  Dep... 0  Relati... true  With... 65535  Move Curso... true
  - BYTE_JUMP  Calc:   Offset: 9  Post Offset: 0  Relative: false  Endian: BIG  Bytes: 1
    - RETURN  Details...  Extract...  Direction: SOURCE  Confidence: 1
- MA...  Ty... HEX  1000C40200  Offs... 0  Rever... true  No Ca... false  Dep... 0  Relati... true  With... 65535  Move Curso... true
  - ANCHOR  Cursor: END  Position: START_OF_PAYLOAD  Offset: 2  Relative: true
  - RETURN  Details...  Extract...  Direction: SOURCE  Confidence: 1
```

Figure 31: Payload Processor open in Editor

Payload processors are made up of several functions which provide a way of searching and testing a packet's payload. There are two other functions that can only be used within, and are usually used to end, a function chain. These are the Anchor and Return functions. These two functions are what allow you to do something with the information within the packet's payload. There is a cursor that belongs to the payload processor that can be moved by the Match and/or Anchor functions to mark certain locations within the payload to be used in finding/extracting data.

4.3.3 Payload Processor Functions

The functions available to create a payload processor are:

Always

The Always function, as its name implies, is used to always return information from a matched fingerprint without doing any packet processing

Match

The Match function allows you to do further processing conditional on finding a certain value within a packet's payload.

Byte Test

The Byte Test function allows for the testing of the integer value of a location in the packet's payload. Like the Match function the processor will do further processing if the test passes.

Is Data At

This is just a simple test to see if there is data at a given location within the packet's payload.

Byte Jump

This function is used to move the cursor based on an integer value from the packet's payload. The calc field allows you to calculate the new position using that value. Any basic arithmetic operation is allowed using "x" as the stand in variable for the value pulled from the packet (e.g. "x + 4" would take the value pulled from the packet and add four which would then be used to set the new position of the cursor).

Anchor

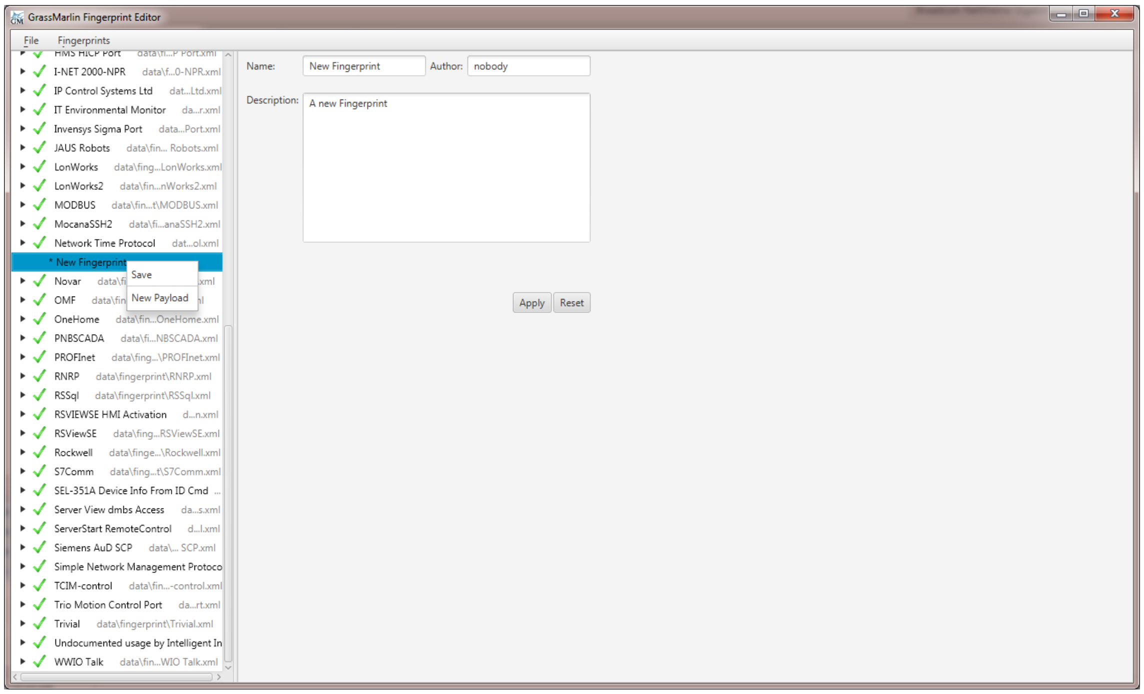
This function allows you to set a cursor position.

Return

This function is what allows you to return information from the fingerprint. There are two main ways that you can define that information: Details and Extract. Details allows you to define name/value pairs that are reported back. Extract is what you use to pull information from the payload, interpret it in some way, and report it. Any information returned is displayed in the host details window of either the source or destination of this connection, whichever is designated by the Return.

Creating a Payload Processor

To create a new processor, right click on the fingerprint and select "New Payload"



```
HMIS HICP Port                  data\fi...P Port.xml
I-NET 2000-NPR                  data\f...0-NPR.xml
IP Control Systems Ltd          dat...Ltd.xml
IT Environmental Monitor        da...r.xml
Invensys Sigma Port             data...Port.xml
JAUS Robots                     data\fin... Robots.xml
LonWorks                        data\fing...LonWorks.xml
LonWorks2                       data\fin...nWorks2.xml
MODBUS                          data\fin...t\MODBUS.xml
MocanaSSH2                      data\fi...anaSSH2.xml
Network Time Protocol           dat...ol.xml
* New Fingerprint
Novar                           data\fi ... .xml
OMF                             data\fin ... l
OneHome                         data\fin...OneHome.xml
PNBSCADA                        data\fi...NBSCADA.xml
PROFInet                        data\fing...\PROFInet.xml
RNRP                            data\fingerprint\RNRP.xml
RSSql                           data\fingerprint\RSSql.xml
RSVIEWSE HMI Activation         d...n.xml
RSViewSE                        data\fing...RSViewSE.xml
Rockwell                        data\finge...\Rockwell.xml
S7Comm                          data\fing...t\S7Comm.xml
SEL-351A Device Info From ID Cmd  ...
Server View dmbs Access         da...s.xml
ServerStart RemoteControl       d...l.xml
Siemens AuD SCP                 data\... SCP.xml
Simple Network Management Protoco
TCIM-control                    data\fin...-control.xml
Trio Motion Control Port        da...rt.xml
Trivial                         data\fingerprint\Trivial.xml
Undocumented usage by Intelligent In
WWIO Talk                       data\fin...WIO Talk.xml
```

Name: New Fingerprint
Author: nobody
Description: A new Fingerprint

Figure 32: Creating a new Payload Processor

This creates a new, empty payload processor for that fingerprint.

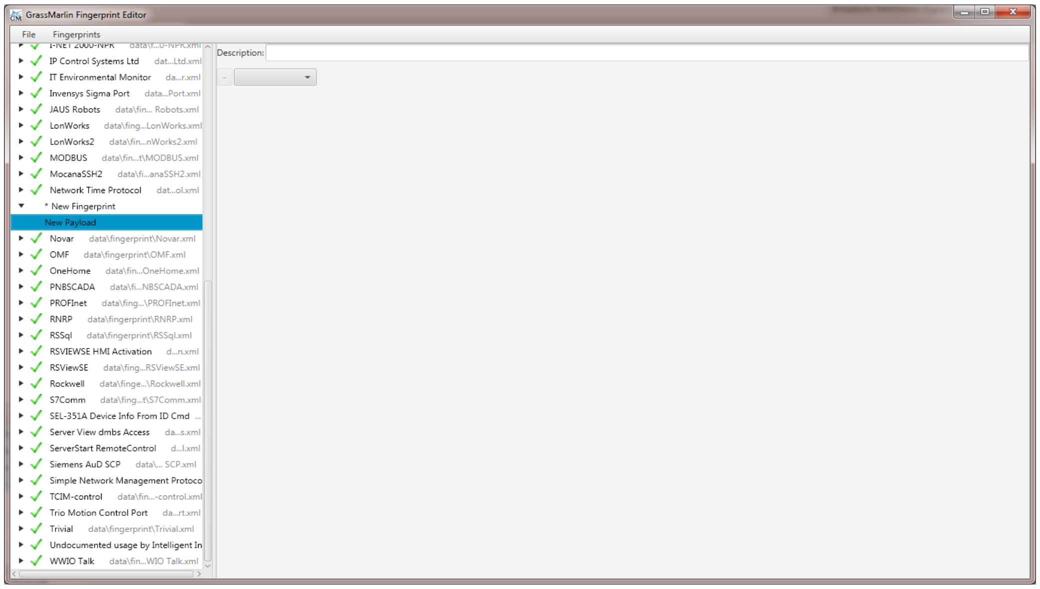

GrassMarlin Fingerprint Editor
File
Fingerprints
I-NET 2000-NPR data\f...0-NPR.xml
IP Control Systems Ltd dat...Ltd.xml
IT Environmental Monitor da...r.xml
Invensys Sigma Port data...Port.xml
JAUS Robots data\fin... Robots.xml
LonWorks data\fing...LonWorks.xml
LonWorks2 data\fin...nWorks2.xml
MODBUS data\fin...t\MODBUS.xml
MocanaSSH2 data\fi...anaSSH2.xml
Network Time Protocol dat...ol.xml
* New Fingerprint
New Payload
Novar data\fingerprint\Novar.xml
OMF data\fingerprint\OMF.xml
OneHome data\fin...OneHome.xml
PNBSCADA data\fi...NBSCADA.xml
PROFInet data\fing...\PROFInet.xml
RNRP data\fingerprint\RNRP.xml
RSSql data\fingerprint\RSSql.xml
RSVIEWSE HMI Activation d...n.xml
RSViewSE data\fing...RSViewSE.xml
Rockwell data\finge...\Rockwell.xml
S7Comm data\fing...t\S7Comm.xml
SEL-351A Device Info From ID Cmd ...
Server View dmbs Access da...s.xml
ServerStart RemoteControl d...l.xml
Siemens AuD SCP data\... SCP.xml
Simple Network Management Protoco
TCIM-control data\fin...-control.xml
Trio Motion Control Port da...rt.xml
Trivial data\fingerprint\Trivial.xml
Undocumented usage by Intelligent In
WWIO Talk data\fin...WIO Talk.xml
Description:

Figure 33: New Payload Processor

To add a function, just select it from the dropdown box.

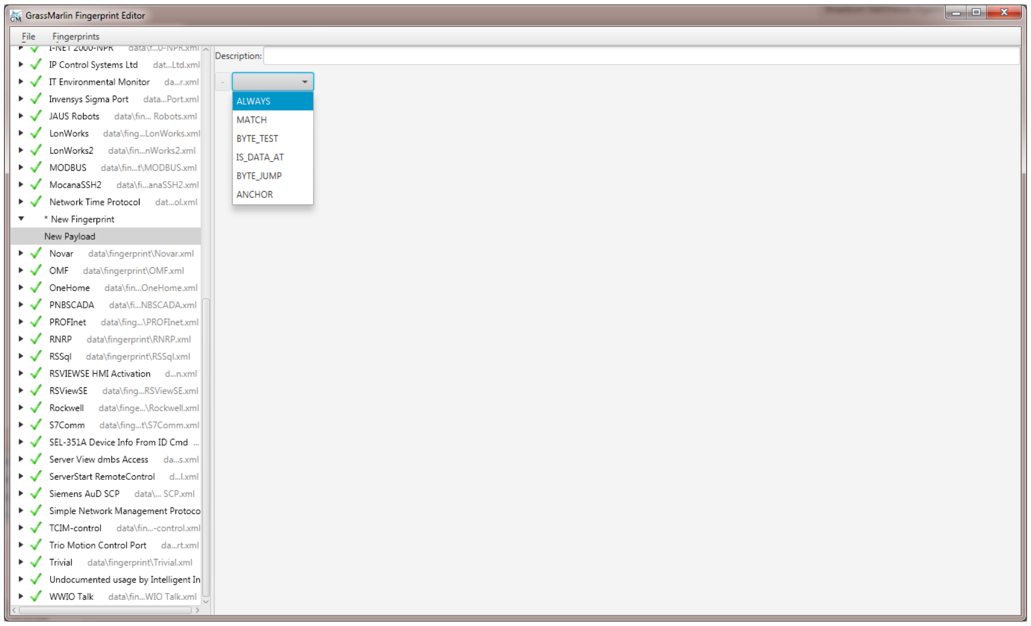
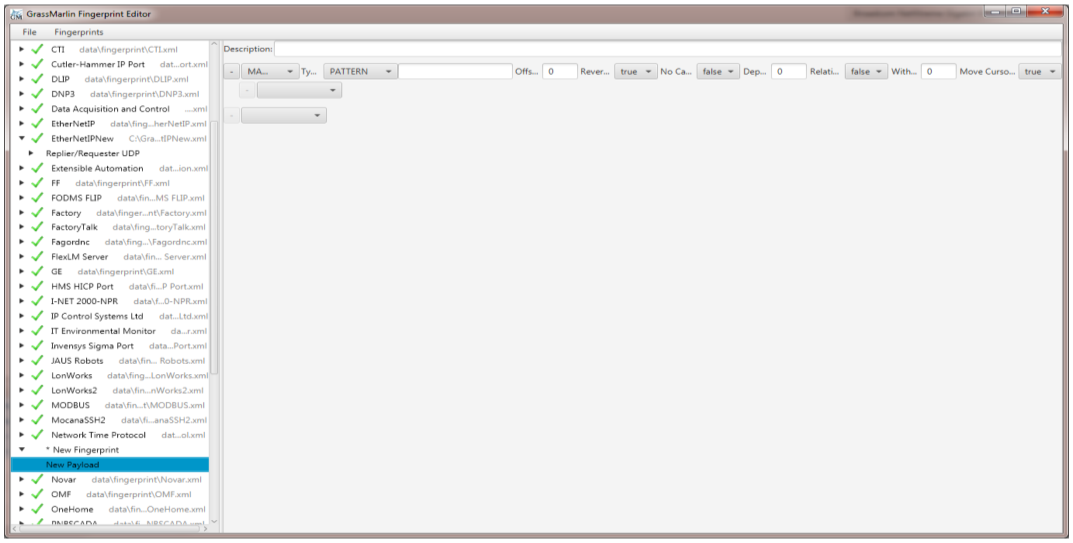


Figure 34: Adding a Function

This will add a line containing all the configuration settings for that function. If the function allows for follow on functions, an indented drop down box will also be added.



```
CTI  data\fingerprint\CTI.xml
Cutler-Hammer IP Port  dat...ort.xml
DLIP  data\fingerprint\DLIP.xml
DNP3  data\fingerprint\DNP3.xml
Data Acquisition and Control  ....xml
EtherNetIP  data\fing...herNetIP.xml
EtherNetIPNew  C:\Gra...tIPNew.xml
  Replier/Requester UDP
Extensible Automation  dat...ion.xml
FF  data\fingerprint\FF.xml
FODMS FLIP  data\fin...MS FLIP.xml
Factory  data\finger...nt\Factory.xml
FactoryTalk  data\fing...toryTalk.xml
Fagordnc  data\fing...\Fagordnc.xml
FlexLM Server  data\fin... Server.xml
GE  data\fingerprint\GE.xml
HMS HICP Port  data\fi...P Port.xml
I-NET 2000-NPR  data\f...0-NPR.xml
IP Control Systems Ltd  dat...Ltd.xml
IT Environmental Monitor  da...r.xml
Invensys Sigma Port  data...Port.xml
JAUS Robots  data\fin... Robots.xml
LonWorks  data\fing...LonWorks.xml
LonWorks2  data\fin...nWorks2.xml
MODBUS  data\fin...t\MODBUS.xml
MocanaSSH2  data\fi...anaSSH2.xml
Network Time Protocol  dat...ol.xml
* New Fingerprint
  New Payload
Novar  data\fingerprint\Novar.xml
OMF  data\fingerprint\OMF.xml
OneHome  data\fin...OneHome.xml
```

```
Description:
- MA... Ty... PATTERN  Offs... 0  Rever... true  No Ca... false  Dep... 0  Relati... false  With... 0  Move Curso... true
```

Figure 35: New Function Added

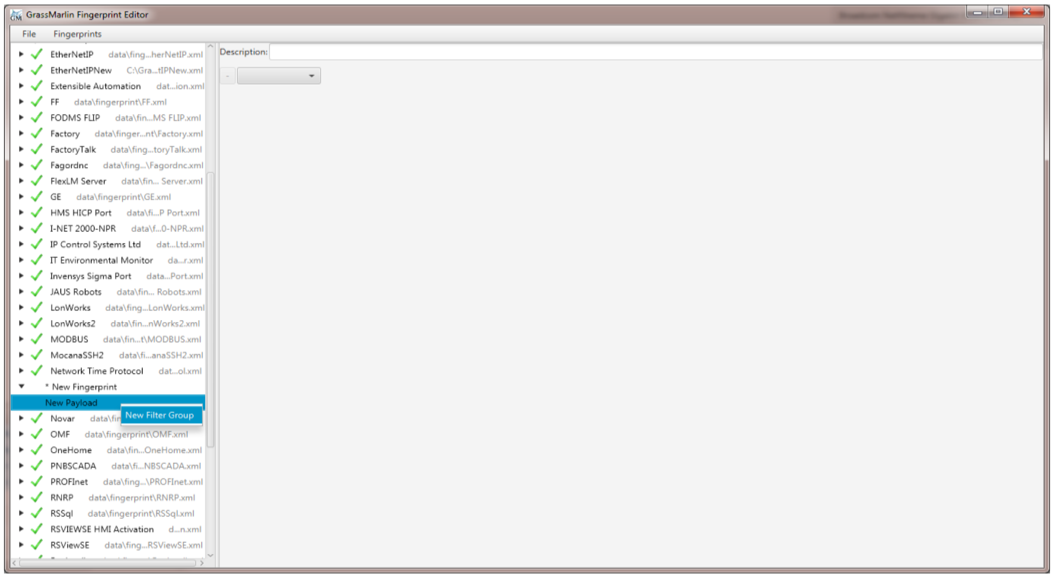
To remove a function, click on the "-" button to the left of the function to be removed. Removing a function also removes all follow on functions for that function.

4.3.4 Filters and Filter Groups

Filters are used to do an initial filter of the traffic using packet metadata before being passed to the payload processors. The available filters are:

- Ack
- Dsize
- DsizeWithin
- DstPort
- Ethertype
- Flags
- MSS
- Seq
- SrcPort
- TTL
- TTLWithin
- TransportProtocol
- Window

To add a filter group to a payload processor, right click on the processor and select "New Filter Group".

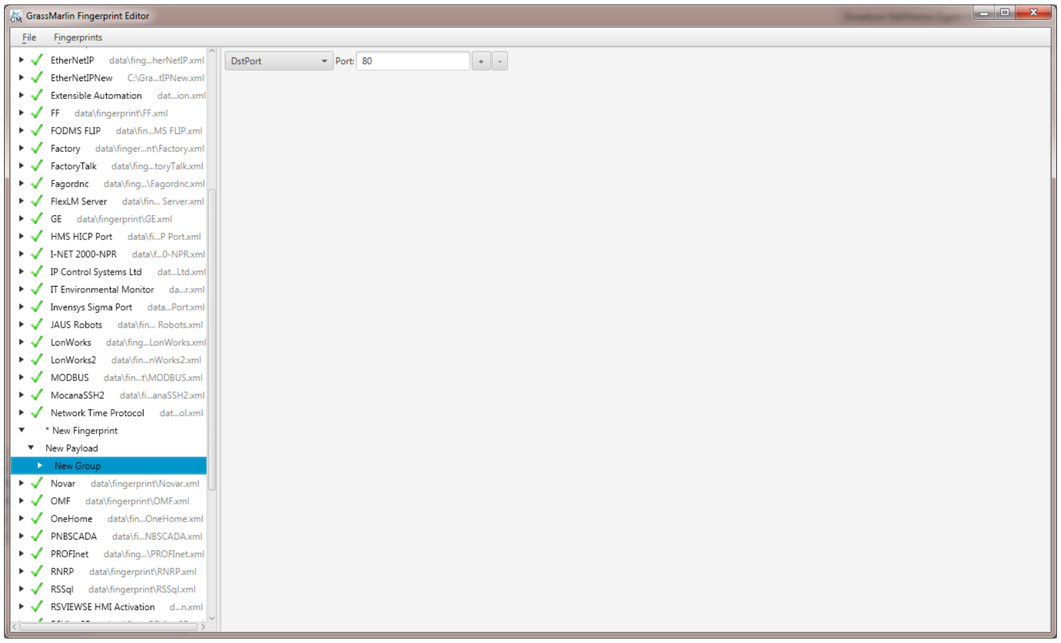


EtherNetIP	data\fing...herNetIP.xml
EtherNetIPNew	C:\Gra...tIPNew.xml
Extensible Automation	dat...ion.xml
FF	data\fingerprint\FF.xml
FODMS FLIP	data\fin...MS FLIP.xml
Factory	data\finger...nt\Factory.xml
FactoryTalk	data\fing...toryTalk.xml
Fagordnc	data\fing...\Fagordnc.xml
FlexLM Server	data\fin... Server.xml
GE	data\fingerprint\GE.xml
HMS HICP Port	data\fi...P Port.xml
I-NET 2000-NPR	data\f...0-NPR.xml
IP Control Systems Ltd	dat...Ltd.xml
IT Environmental Monitor	da...r.xml
Invensys Sigma Port	data...Port.xml
JAUS Robots	data\fin... Robots.xml
LonWorks	data\fing...LonWorks.xml
LonWorks2	data\fin...nWorks2.xml
MODBUS	data\fin...t\MODBUS.xml
MocanaSSH2	data\fi...anaSSH2.xml
Network Time Protocol	dat...ol.xml
* New Fingerprint	
New Payload	
Novar	data\fin
OMF	data\fingerprint\OMF.xml
OneHome	data\fin...OneHome.xml
PNBSCADA	data\fi...NBSCADA.xml
PROFInet	data\fing...\PROFInet.xml
RNRP	data\fingerprint\RNRP.xml
RSSql	data\fingerprint\RSSql.xml
RSVIEWSE HMI Activation	d...n.xml
RSViewSE	data\fing...RSViewSE.xml

Description:

Figure 36: Adding a new Filter Group

The new filter group will then be opened in the editor pane.

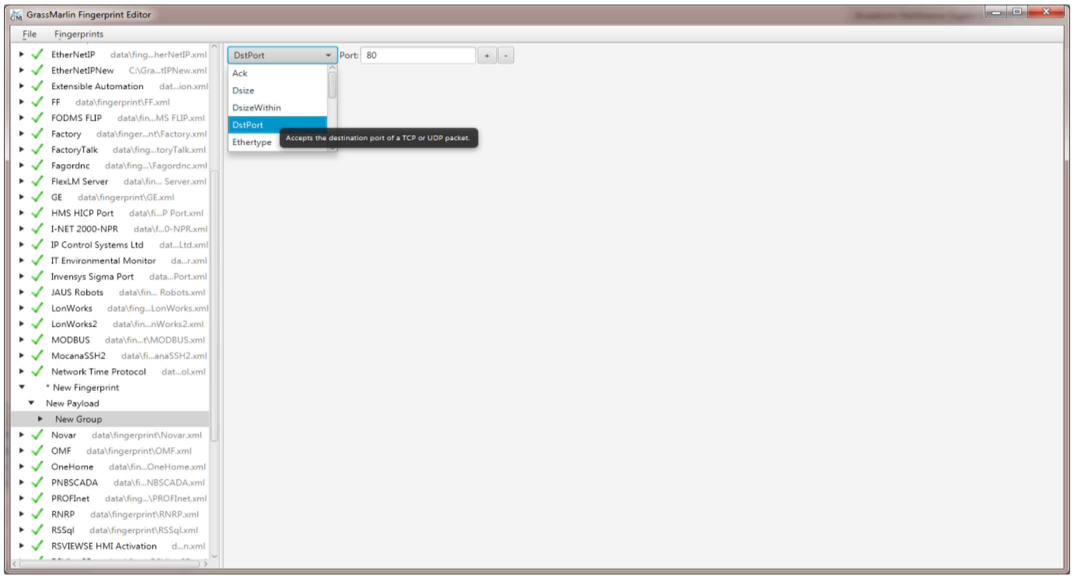


EtherNetIP	data\fing...herNetIP.xml
EtherNetIPNew	C:\Gra...tIPNew.xml
Extensible Automation	dat...ion.xml
FF	data\fingerprint\FF.xml
FODMS FLIP	data\fin...MS FLIP.xml
Factory	data\finger...nt\Factory.xml
FactoryTalk	data\fing...toryTalk.xml
Fagordnc	data\fing...\Fagordnc.xml
FlexLM Server	data\fin... Server.xml
GE	data\fingerprint\GE.xml
HMS HICP Port	data\fi...P Port.xml
I-NET 2000-NPR	data\f...0-NPR.xml
IP Control Systems Ltd	dat...Ltd.xml
IT Environmental Monitor	da...r.xml
Invensys Sigma Port	data...Port.xml
JAUS Robots	data\fin... Robots.xml
LonWorks	data\fing...LonWorks.xml
LonWorks2	data\fin...nWorks2.xml
MODBUS	data\fin...t\MODBUS.xml
MocanaSSH2	data\fi...anaSSH2.xml
Network Time Protocol	dat...ol.xml
* New Fingerprint	
New Payload	
New Group	
Novar	data\fingerprint\Novar.xml
OMF	data\fingerprint\OMF.xml
OneHome	data\fin...OneHome.xml
PNBSCADA	data\fi...NBSCADA.xml
PROFInet	data\fing...\PROFInet.xml
RNRP	data\fingerprint\RNRP.xml
RSSql	data\fingerprint\RSSql.xml
RSVIEWSE HMI Activation	d...n.xml

DstPort Port: 80

Figure 37: New Filter Group Added

To change a filter, use the drop down menu on the left.


GrassMarlin Fingerprint Editor
File
Fingerprints
EtherNetIP data\fing...herNetIP.xml
EtherNetIPNew C:\Gra...tIPNew.xml
Extensible Automation dat...ion.xml
FF data\fingerprint\FF.xml
FODMS FLIP data\fin...MS FLIP.xml
Factory data\finger...nt\Factory.xml
FactoryTalk data\fing...toryTalk.xml
Fagordnc data\fing...\Fagordnc.xml
FlexLM Server data\fin... Server.xml
GE data\fingerprint\GE.xml
HMS HICP Port data\fi...P Port.xml
I-NET 2000-NPR data\f...0-NPR.xml
IP Control Systems Ltd dat...Ltd.xml
IT Environmental Monitor da...r.xml
Invensys Sigma Port data...Port.xml
JAUS Robots data\fin... Robots.xml
LonWorks data\fing...LonWorks.xml
LonWorks2 data\fin...nWorks2.xml
MODBUS data\fin...t\MODBUS.xml
MocanaSSH2 data\fi...anaSSH2.xml
Network Time Protocol dat...ol.xml
* New Fingerprint
New Payload
New Group
Novar data\fingerprint\Novar.xml
OMF data\fingerprint\OMF.xml
OneHome data\fin...OneHome.xml
PNBSCADA data\fi...NBSCADA.xml
PROFInet data\fing...\PROFInet.xml
RNRP data\fingerprint\RNRP.xml
RSSql data\fingerprint\RSSql.xml
RSVIEWSE HMI Activation d...n.xml
Ack
Dsize
DsizeWithin
DstPort
Ethertype
+
-

DstPort Port: 80

Accepts the destination port of a TCP or UDP packet.

Figure 38: Changing a Filter

To add or remove a filter, use the buttons on the right. Adding a filter that requires information only contained in TCP packets will also add a Transport Protocol filter configured to filter for TCP packets if there isn't one already. A payload processor may have more than one filter group. Filters within a filter group are applied as a single filter, meaning that a packet must pass all the filters contained within the group to pass. If a packet passes any one filter group for a payload processor, it will be passed to that processor.

5 Reports

Reports in GRASSMARLIN 3.2 are exportable tables of data derived from Imported content. These Reports are available from the **View** menu. Each Report menu item will open a dialog allowing the Report to be viewed and, in some cases, customized, with an option to export the Report to a CSV file.

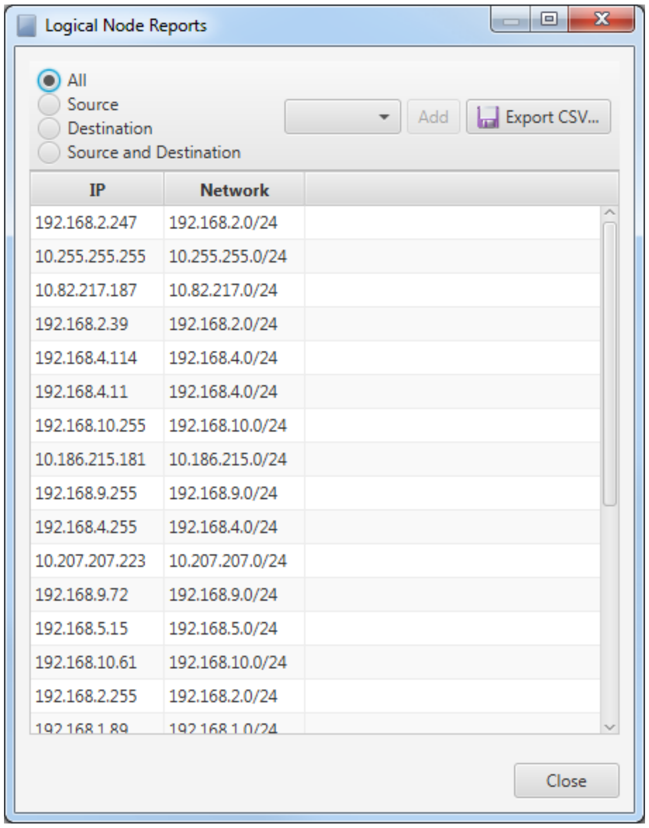
If a Report is viewed while data is being gathered (Imports are running or Live PCAP is executing) the contents of the Reports may be inaccurate. It is not advisable to view Reports while actively changing the data.

5.1 Logical Nodes

The Logical Nodes provides a list of Nodes contained in the Logical Graph. By default, only a single column (IP) is present, although additional columns can be added with any Property present in the set of Nodes.

The list of Nodes can be filtered to contain only Nodes that are a Source for traffic, a Destination for traffic, or both.

To add a column select the Property Name from the drop-down and click the Add button. To remove a column, select the "Remove Column" menu item from the column header's context menu.



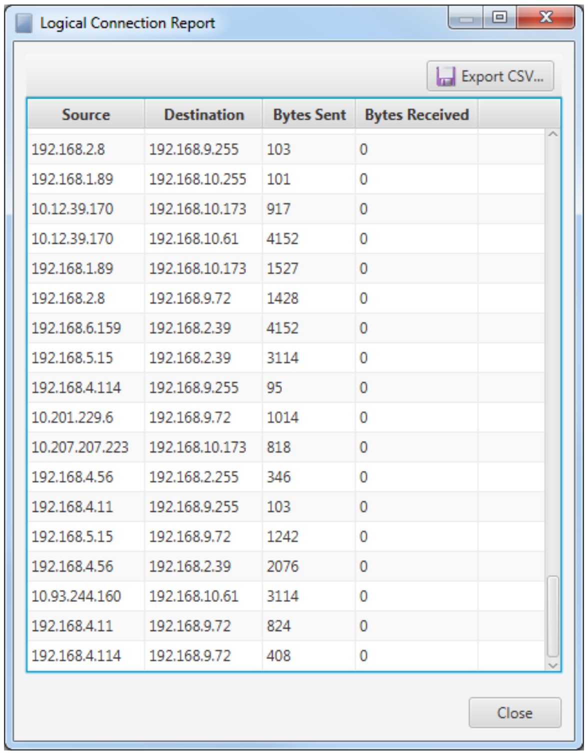
[✓ All] Source Destination Source and Destination

IP	Network
192.168.2.247	192.168.2.0/24
10.255.255.255	10.255.255.0/24
10.82.217.187	10.82.217.0/24
192.168.2.39	192.168.2.0/24
192.168.4.114	192.168.4.0/24
192.168.4.11	192.168.4.0/24
192.168.10.255	192.168.10.0/24
10.186.215.181	10.186.215.0/24
192.168.9.255	192.168.9.0/24
192.168.4.255	192.168.4.0/24
10.207.207.223	10.207.207.0/24
192.168.9.72	192.168.9.0/24
192.168.5.15	192.168.5.0/24
192.168.10.61	192.168.10.0/24
192.168.2.255	192.168.2.0/24
192.168.1.89	192.168.1.0/24

Figure 39: Logical Node Report Dialog

5.2 Logical Connections

The Logical Connections Report identifies the amount of traffic sent and received between every pair of Nodes. Presently, this Report cannot be customized.



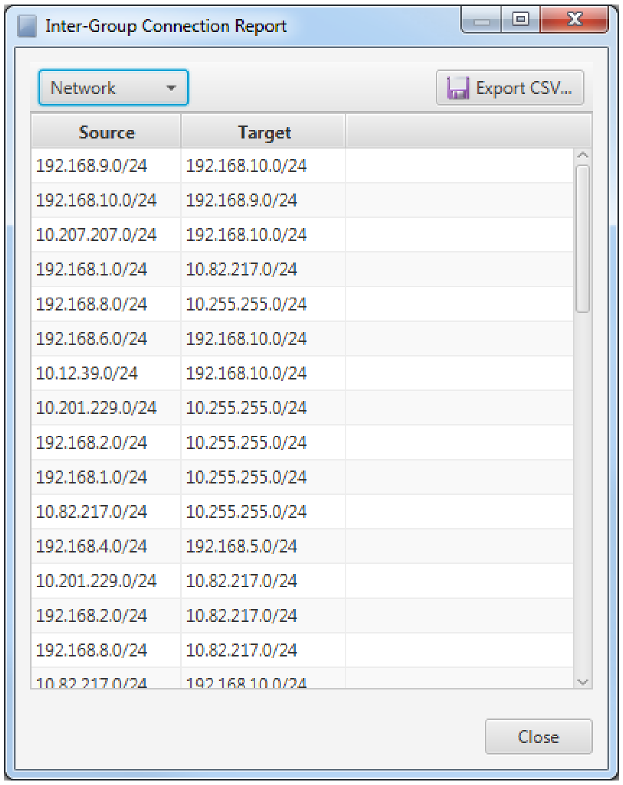
Source	Destination	Bytes Sent	Bytes Received
192.168.2.8	192.168.9.255	103	0
192.168.1.89	192.168.10.255	101	0
10.12.39.170	192.168.10.173	917	0
10.12.39.170	192.168.10.61	4152	0
192.168.1.89	192.168.10.173	1527	0
192.168.2.8	192.168.9.72	1428	0
192.168.6.159	192.168.2.39	4152	0
192.168.5.15	192.168.2.39	3114	0
192.168.4.114	192.168.9.255	95	0
10.201.229.6	192.168.9.72	1014	0
10.207.207.223	192.168.10.173	818	0
192.168.4.56	192.168.2.255	346	0
192.168.4.11	192.168.9.255	103	0
192.168.5.15	192.168.9.72	1242	0
192.168.4.56	192.168.2.39	2076	0
10.93.244.160	192.168.10.61	3114	0
192.168.4.11	192.168.9.72	824	0
192.168.4.114	192.168.9.72	408	0

Figure 40: Logical Connections Report Dialog

5.3 Inter-Group Connections

The Inter-Group Connections Report displays a list of connections that exist between different Groups. The only option for this report is the Property on which to Group. This is selected from the drop-down box in the top left corner of the dialog.

These pairings are based on directional traffic; if a Node in Group A sent a packet to a Node in Group B, then a record showing A to B will be present. A second entry containing Communication from B to A will be present only if there is at least one packet originating with a Node in Group B set to a Node in Group A.


Inter-Group Connection Report

Source	Target
192.168.9.0/24	192.168.10.0/24
192.168.10.0/24	192.168.9.0/24
10.207.207.0/24	192.168.10.0/24
192.168.1.0/24	10.82.217.0/24
192.168.8.0/24	10.255.255.0/24
192.168.6.0/24	192.168.10.0/24
10.12.39.0/24	192.168.10.0/24
10.201.229.0/24	10.255.255.0/24
192.168.2.0/24	10.255.255.0/24
192.168.1.0/24	10.255.255.0/24
10.82.217.0/24	10.255.255.0/24
192.168.4.0/24	192.168.5.0/24
10.201.229.0/24	10.82.217.0/24
192.168.2.0/24	10.82.217.0/24
192.168.8.0/24	10.82.217.0/24
10.82.217.0/24	192.168.10.0/24

Figure 41: Inter-Group connections Report Dialog

6 Tools

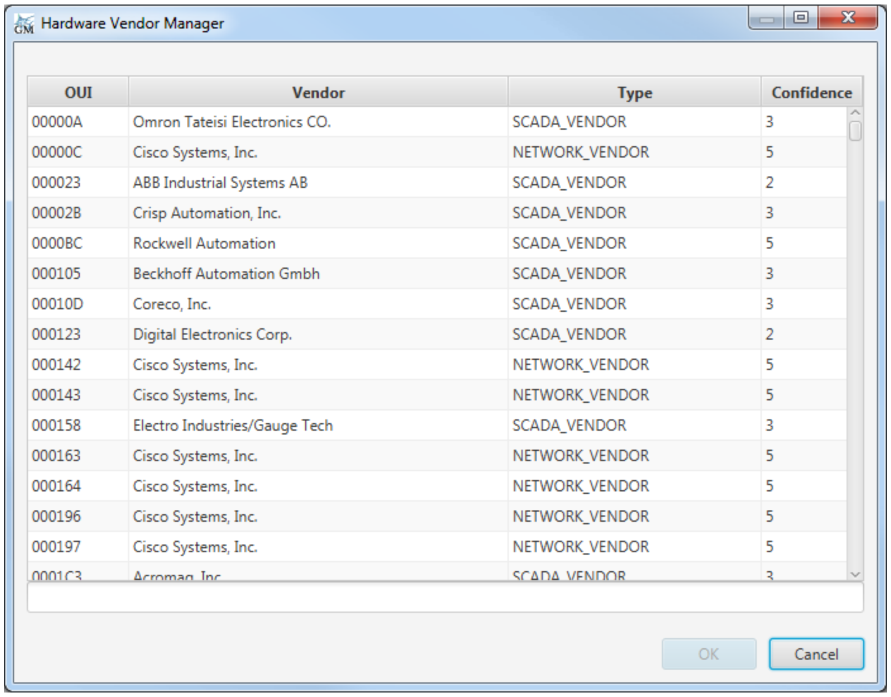
The Tools menu contains several useful features that don't deal directly with data in a Session. In addition to the built-in items described here, this is where the interfaces to Plugins are made available.

The Fingerprint Manager, described in Section 4, is also available from the Tools menu.

6.1 Hardware Vendor Manager

The Hardware Vendor Manager displays a list of OUIs and the venders with which those OUIs are associated. The text field at the bottom of the dialog can be used to search the OUI, Vendor, and Type columns; if the data in any of those 3 columns contains the entered text, the row will be included in the search results.

The contents of the Hardware Vendor Manager can be modified. If the data is modified, clicking the Ok button will save the data to disk and close the dialog, whereas the Cancel button will close the dialog without saving the changes. To edit a value, select a row then click on the contents of the column to edit for that row. Selecting a row then pressing the Delete key will delete that row.



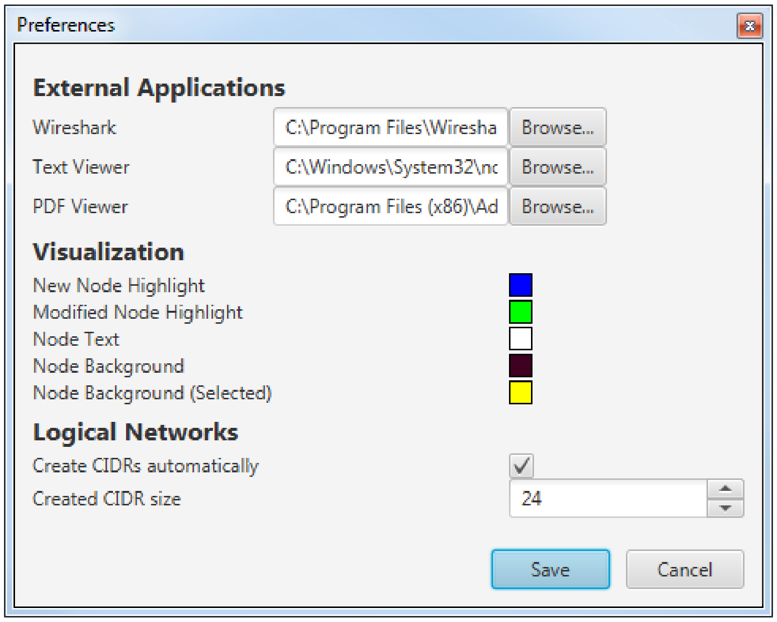
OUI	Vendor	Type	Confidence
00000A	Omron Tateisi Electronics CO.	SCADA_VENDOR	3
00000C	Cisco Systems, Inc.	NETWORK_VENDOR	5
000023	ABB Industrial Systems AB	SCADA_VENDOR	2
00002B	Crisp Automation, Inc.	SCADA_VENDOR	3
0000BC	Rockwell Automation	SCADA_VENDOR	5
000105	Beckhoff Automation Gmbh	SCADA_VENDOR	3
00010D	Coreco, Inc.	SCADA_VENDOR	3
000123	Digital Electronics Corp.	SCADA_VENDOR	2
000142	Cisco Systems, Inc.	NETWORK_VENDOR	5
000143	Cisco Systems, Inc.	NETWORK_VENDOR	5
000158	Electro Industries/Gauge Tech	SCADA_VENDOR	3
000163	Cisco Systems, Inc.	NETWORK_VENDOR	5
000164	Cisco Systems, Inc.	NETWORK_VENDOR	5
000196	Cisco Systems, Inc.	NETWORK_VENDOR	5
000197	Cisco Systems, Inc.	NETWORK_VENDOR	5
0001C3	Acromag, Inc.	SCADA_VENDOR	3

Figure 42: Hardware Vendor Manager Dialog

In GRASSMARLIN 3.3, we anticipate the automatic creation of Properties based on the OUI data in a packet and the contents of this table.

6.2 Preferences

Preferences are application-wide settings that are automatically saved and loaded. These include what applications should be used to display certain file types, color of certain visual elements, and the size of automatically-created CIDRs. Changes made on the Preferences Dialog are only used if the dialog is closed with the Save button; all changes will be discarded without effect if the Cancel button is used.


Preferences

External Applications

Wireshark	C:\Program Files\Wiresha	Browse...
Text Viewer	C:\Windows\System32\nc	Browse...
PDF Viewer	C:\Program Files (x86)\Ad	Browse...

Visualization

New Node Highlight	🟦
Modified Node Highlight	🟩
Node Text	⬜
Node Background	⬛
Node Background (Selected)	🟨

Logical Networks

Create CIDRs automatically	✓
Created CIDR size	24

Figure 43: Preferences Dialog

7 Troubleshooting

GRASSMARLIN was designed to be robust when it encounters errors. This does not mean that errors can't or don't happen, but when they do the application handles the situation as best it can, notifies the user, and recovers to a known, stable point. Some errors that may arise include files that are corrupted and/or unparsable, incorrect Import Type assignments, and memory errors.

7.1 Console

The Console contains detailed information that can useful but is not necessarily critical information. In addition to details made available as part of various operations, there are three sections of data that are written to the console every time GRASSMARLIN starts (assuming no fatal errors prevent one or more of these from executing). The first piece of information to be written is the full version number. This includes both Major-Minor-Version and the Revision number. Next, it performs an integrity check on the GeoIP database. If there are any country IDs that lack a name, this fact is reported to the console. Additionally, if there is a defined country name but the flag file for that name is not present, a message is generated in the Console for every such country as well.

In GRASSMARLIN 3.2, it is expected that there will be 2 undefined Country IDs and 53 missing flag files.

After the GeoIP check, all of the configuration values are written to the Console. Among the values are those that identify the paths for various components. Most of these paths are relative to the current working directory, however some are absolute paths. If GRASSMARLIN resource files are moved or otherwise relocated, these values are likely to be wrong. There is no supported way to explicitly change many of these, and the unsupported methods are platform-specific. The only supported methods for changing these values is to modify them in the Preferences Dialog, or by running GRASSMARLIN with the "–config" parameter.

Most messages that are written to the Message Log are also written to the Console.

As Plugins are being loaded, information about those plugins is written to the Console.

Details for issues reading a line in a Bro2Json file are written to the Console.

Some issues with converting a Visualization to a SVG file are written, with detail, to the Console.

During the Layout of the Physical Graph, details are written to the Console.

Java is also known to write error messages to the Console. These error messages tend to look like the following and refer to places in the code where errors were not properly handled. There are a few known cases where an error message is generated for the Console, despite the error being properly handled.

```
Exception in thread "main" util.Launcher$ThisIsAnExampleException
    at util.Launcher.GenerateSampleError(Launcher.java:56)
    at util.Launcher.main(Launcher.java:60)
```

When messages such as the previous example are generated, this is known as a stack trace, and it is very instrumental in fixing the issue that led to the error. When submitting a bug report, providing the full text of the stack trace to the developers is of the utmost importance.

7.2 Command-Line

GRASSMARLIN is a Java application and by necessity the command to execute GRASSMARLIN is a call to the Java executable code with GRASSMARLIN as an argument. The installers provide a script which will execute Java with the appropriate parameters. By platform, these scripts vary. The Windows script is included here as an example, but while specifics may differ slightly, the same logic is applied to all platforms.

The Java command line features several arguments. Those before the name of the JAR file to execute are arguments to Java, those after are arguments to GRASSMARLIN. The selected options are the result of performance testing and tuning to improve performance and responsiveness of GRASSMARLIN under typical loads and configurations.

```
java.exe -Dnio.mx=%quartermem%mb -Dnio.ms=512mb -server -d64 -Xms%halfmem%m -
Xmx%halfmem%m  -XX:+UseG1GC -XX:+DisableExplicitGC -XX:NewSize=%quartermem%m -
XX:MaxGCPauseMillis=2000 -jar "%~dp0\GrassMarlin.jar"
```

In the Windows script, %quartermem% and %halfmem% are variables set to a quarter and half of the total system memory, as detected by the installer. These arguments have the following effects:

-Dnio.mx	Maximum memory available to JNetPcap.
-Dnio.ms	Minimum memory available to JNetPcap.
-server	Instructs Java to use the server JVM. The Server JVM is for use on systems with at least 2 cores and 2 GB of RAM. The alternative, -client, should not be used unless necessary.
-d64	Encourages use of the 64-bit JVM. –d32 does the same for 32-bit JVM. This isn't supposed to force the 64-bit build, but does on some platforms, causing issues on 32-bit systems.
-Xmx	Maximum memory available to Java.
-Xms	Minimum memory available to Java.
-XX:+UseG1GC	Instruct Java to use the G1 Garbage Collector, which is optimized for larger memory sizes.
-XX:+DisableExplicitGC	Reduces the frequency with which the Garbage Collector is called. This is done to address a performance bug.
-XX:+NewSize	Sets the size of the buffer used for short-lived objects.
-XX:MaxGCPauseMillis	Instruct Java to attempt to pause for no longer than this duration when running the Garbage Collector.

In addition to the arguments to Java, GRASSMARLIN supports two arguments to modify the default behavior.

An argument that does not start with a hyphen will probably be interpreted as a file to load or import in future versions.

-nopcap	Disables Live PCAP functionality.

-noplugins	Suppresses the enumeration and loading of plugins.

7.3 JNetPcap

GRASSMARLIN relies on the JNetPcap library for Live PCAP. If JNetPcap is not functioning correctly, this may indicate issues with the system configuration. Offline Pcap operations do not use JNetPcap, so issues with JNetPcap should have no effect on loading Pcap files. If there are issues initializaing the JNetPcap library, then the Live Pcap interface will be disabled and relevant information should be written to the Console and Message Log.

8 Interoperating with GRASSMARLIN

GRASSMARLIN was developed not to perform analysis, but to facilitate it. To this end, it is also a goal of the GRASSMARLIN team to facilitate development of tools that interoperate with GRASSMARLIN; either tools that perform analysis or that facilitate it further. As an open source project, you're free to fork the code and build upon what we've started, but that has the innate risk of breaking compatibility with future releases of GRASSMARLIN. Also, as developers ourselves, we're quite familiar with the burden of looking through thirty-something-thousand lines of somebody else's code. We tried to keep the code clean and orderly, but we have deadlines and limited manpower, to say nothing of code written on the Friday before a holiday weekend, or in the days leading up to a vacation. To that end, we would like to describe in a little detail how to go about writing plugins and interoperating with saved Sessions. There are no promises that there will not be breaking changes to these features in the future, but both should serve as an easy starting point for modifying GRASSMARLIN to better serve your needs.

8.1 Plugins

Plugins are binary Java modules that will be loaded at runtime to modify the capabilities of GRASSMARLIN. Plugins are loaded during initialization by looking at all the ".jar" files in the "plugins" directory, which is located in the current working directory. For each file it will attempt to load a class from that file where the name of the class is "Plugin" and it exists in a package with the same name as the file (sans the ".jar" extension). For example, the file "iadgov.csvimport.jar" will be searched for the class "iadgov.csvimport.Plugin". This class must implement the "util.Plugin" class as defined in the GRASSMARLIN code and have a public, zero-argument constructor. An instance of the Plugin class will be created using that constructor.

The plugin constructor can do anything, really. There is no reason for it to do so, but you can do anything you want there. You can spawn threads, use reflection to change core classes, or do anything else you're allowed to do in Java—which is pretty much anything. We don't recommend this, but you can do it. You probably don't want to do anything here, but it is important to know that you *can*—which means plugins can be written for malicious tasks, which means you need to be very aware of which plugins you load and ensure they do what you think they do. There is a command-line argument (-noplugins) to suppress the loading of plugins.

Plugins, without any other logic, provide a name and (optionally) a menu item. The menu item can do anything, but it is expected that it will launch a configuration or about dialog, or be a Menu with child MenuItems that allow context menu-style configuration.

The core.Plugin class definition contains interfaces which can be implemented by the Plugin classes to add additional functionality into GRASSMARLIN. Presently, there is only a single interface, ImportProcessorV1, which can be used to add new import types.

Remember that part about no promises regarding compatibility in future versions? The V1 was added to the name because we quickly realized we need to change how this works in future versions. It isn't going to be a huge change, but the Iterators are all going to be handled through a single queue, which changes the ImportItem class. Since this will most likely be a breaking change with any classes derived from ImportItem, the V2 interface, which will look very similar to the V1 interface, will be used to identify plugins built for the new class. The actual changes to code to correct for this will be minimal.

8.2 Save Files

Detailing the exact format of the GRASSMARLIN 3.2 Session files is beyond the scope of this document, but here are a few details to facilitate working with them in your own projects.

- The files, which are saved with a ".gm3" extension, are ZIP archives containing multiple XML documents.
- Most of the XML documents fall into one of 3 categories:
    - Graph data, which has a <graph/> root-level tag and contains a list of nodes and edges.
    - Visualization data, which has a variable root tag and contains information about colors, positioning of cells, display options, etc.
    - Session data, which has a <session/> root-level tag and contains information about imported files and other session-level data.
- There is also a manifest which contains the version.
- The "ref" attribute appears frequently in the Visualization data and refers to a node or edge in the corresponding Graph data. Whether it refers to a node or edge is established by context and the specific node or edge is found by finding the Nth node or edge in the (0-based) list in the Graph data, where N is the value of the attribute.
- The "from" and "to" attributes of edges in the Graph data behave similarly to the "ref" attribute above, acting as lookups in the node list.
- The session contains only 3 graphs, so the names of the files corresponding to these graphs is well-defined; logical.xml, physical.xml, and mesh.xml.
- The code to load or save a session starts in the core.document.serialization.Grassmarlin class.
- The code to load a session looks at the manifest and reads the version. The only supported version is 3.2. If this is the version in the file, then it will be handed to the core.document.serialization.Grassmarlin_3_2 class, which performs all the heavy lifting of building the session.
- It is the intention of the developers to maintain the same basic framework and to implement changes for future versions by subclassing Grassmarlin_3_2 and overriding the relevant methods.
- DOM methods would make much cleaner code, but we test with datasets involving around a billion packets. If your data is on a much smaller scale, reading the XML files using DOM libraries is much simpler than what we did. It doesn't scale well, though, which is why we used stream processing.