UBIQUA 2.0 USER GUIDE

UPDATED APRIL 2017

ubiqua

Contents

UBIQUA 2.0

Document generated: April 28th 2017

Chapter 1: Overview

Welcome to Ubiqua

Ubiqua Protocol Analyzer is a tool designed to assist in the various phases of Wireless Sensor Network application development: debugging, testing, and deployment. Ubiqua integrates the top IEEE 802.15.4-based protocol decoder and a wide set of analysis features to provide a powerful, user-friendly, fast, and scalable debugging environment.

This document guides you through the installation process and allows you to learn the basic and advanced features of Ubiqua. If you have additional questions or comments we will be happy to assist you at support@ubilogix.com.

Chapter 2: Getting Started

To start following this guide you need to download and install Ubiqua on your computer. To download the latest version please visit the website at: http://www.ubilogix.com/ubiqua.

The website will ask you to sign in using your Ubilogix account. Remember to read the End User License Agreement before using Ubiqua and its related services. If you don't have an account you can create one at: http://www.ubilogix.com/register. The registration is free and it will take you no more than a couple of minutes.

Requirements

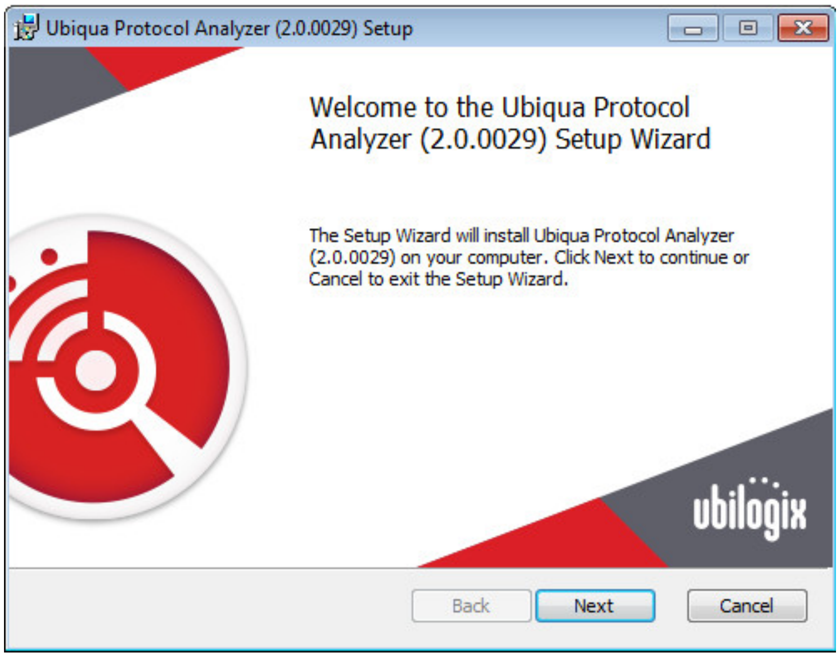
Ubiqua requires the following resources to work properly in your system. Make sure your system meets the minimum requirements before you install the software.

- Windows 7, Windows 8.1, Windows 10 or later.
- 1 GHz processor or higher with at least 1 GB RAM for 32-bit processors or 2 GB RAM for 64-bit.
- 300 MB available disk space.
- Microsoft .NET Framework 4. Available as a free download at: http://go.microsoft.com/fwlink/?LinkID=186913.
- Internet access.

Ubiqua requires `Microsoft .NET Framework 4.5` from version 1.3 (build 2142); previous versions do not support it.

Installation

After downloading the Ubiqua Installer, navigate to the location where it was saved and double-click on it. The installation process will begin with the Setup Wizard window.



Welcome to the Ubiqua Protocol Analyzer (2.0.0029) Setup Wizard

The Setup Wizard will install Ubiqua Protocol Analyzer (2.0.0029) on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Click on the Next button and then read the End-User License Agreement. You need to agree to the License Agreement by clicking the "I accept the terms in the License Agreement" check box in order to continue with the installation.

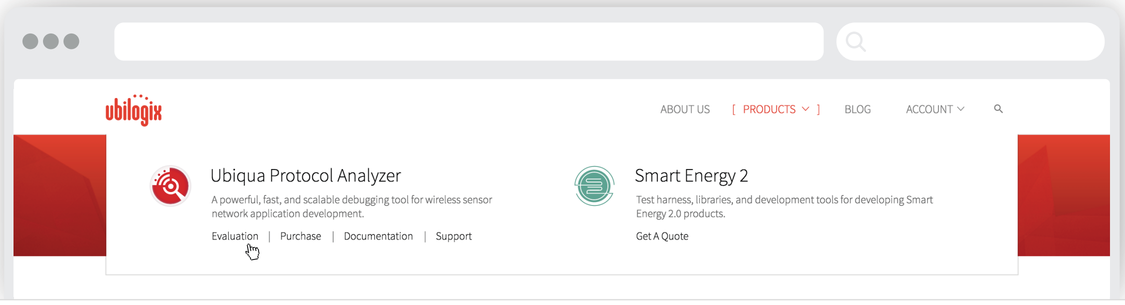
Next you have to choose a location on your hard disk drive where Ubiqua will be installed. By default it will create a folder named "Ubilogix" and a subfolder "Ubiqua Protocol Analyzer" within the Program Files folder and place all needed files for the software to operate in there. We recommend you don't change the default destination folder unless you have a better option that suits your needs. To continue installing Ubiqua, click on the Next button.

Now the wizard is ready to copy all the files. Click on the Install button. Depending on your user credentials the installer might ask you for permission to make changes, in such case please follow the instructions on screen. The installation process should not take more than a few seconds. On the final screen you will be informed that the installation process is completed, click the Finish button to close the setup wizard window.

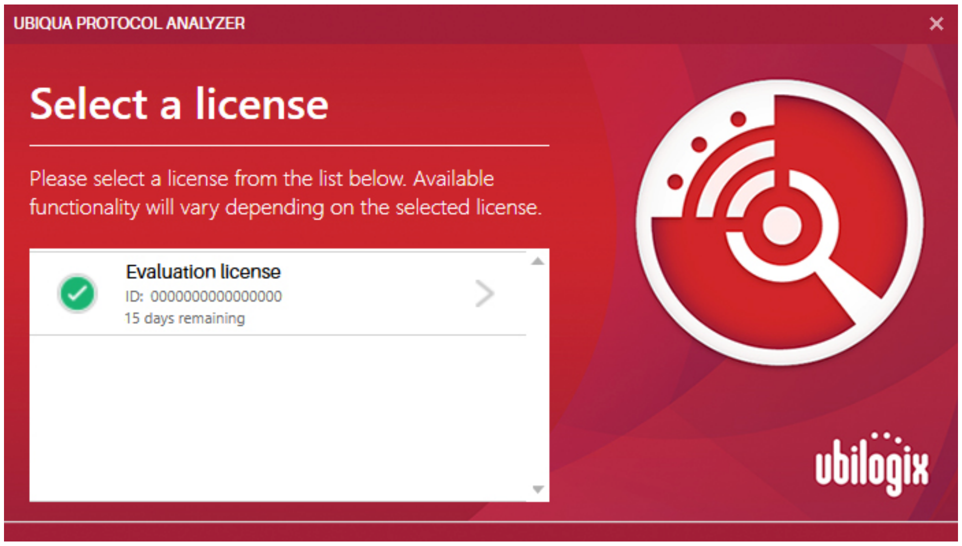
Congratulations, you now have Ubiqua installed on your system!

Starting a 21/days evaluation

To activate your free 21/days evaluation you need a Ubilogix account; if you don't have it, go to the Ubilogix website and from the top menu go to the "Products" option, select the "Evaluation" link and follow the instructions, if you decide to buy a license select the “Purchase” link, then click the “Create a subscription” button and configure your plan, for more details about our licensing methods you can read about it our FAQ Section.


ubilogix
ABOUT US
[PRODUCTS]
BLOG
ACCOUNT
Ubiqua Protocol Analyzer
A powerful, fast, and scalable debugging tool for wireless sensor network application development.
Evaluation | Purchase | Documentation | Support
Smart Energy 2
Test harness, libraries, and development tools for developing Smart Energy 2.0 products.
Get A Quote

You should get your activation code by email within a few minutes. To be able to use the evaluation you have to activate the access code; for this just follow the instructions in "Validating an access code" section. Please note that only one evaluation-license is provided per user account. The evaluation version of Ubiqua is the fully functional version, but with a 1000 packets limit when capturing, or opening capture files. Once the evaluation period ends, the Welcome window will only show the options to upgrade or purchase a license.



Select a license

Please select a license from the list below. Available functionality will vary depending on the selected license.

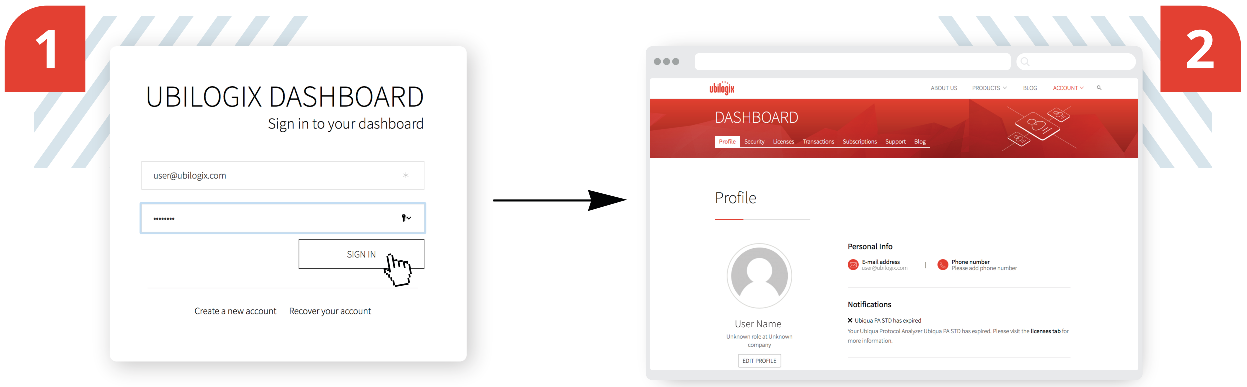
Evaluation license
ID: 0000000000000000
15 days remaining

Validating an access code

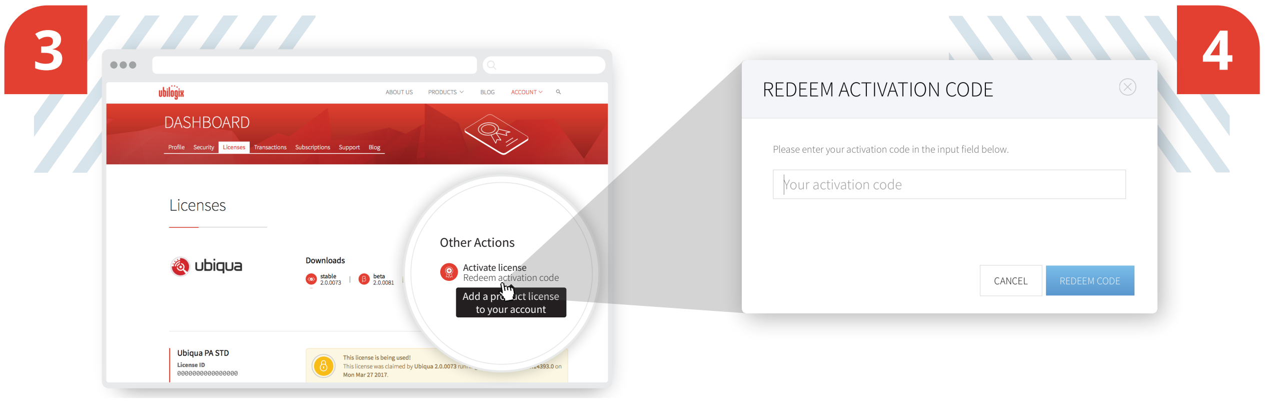
To activate your copy of Ubiqua, you need a valid Activation Code and a working Internet connection. The Activation Code is sent to you by e-mail after a successful Ubiqua purchase. If you have purchased Ubiqua but you haven't received your Activation Code, please check first the spam folder of your mailbox; then (if needed) write us to: support@ubilogix.com

To proceed with the activation process, login to your Ubilogix account pointing your web browser to www.ubilogix.com/login and type your access credentials (email and password)

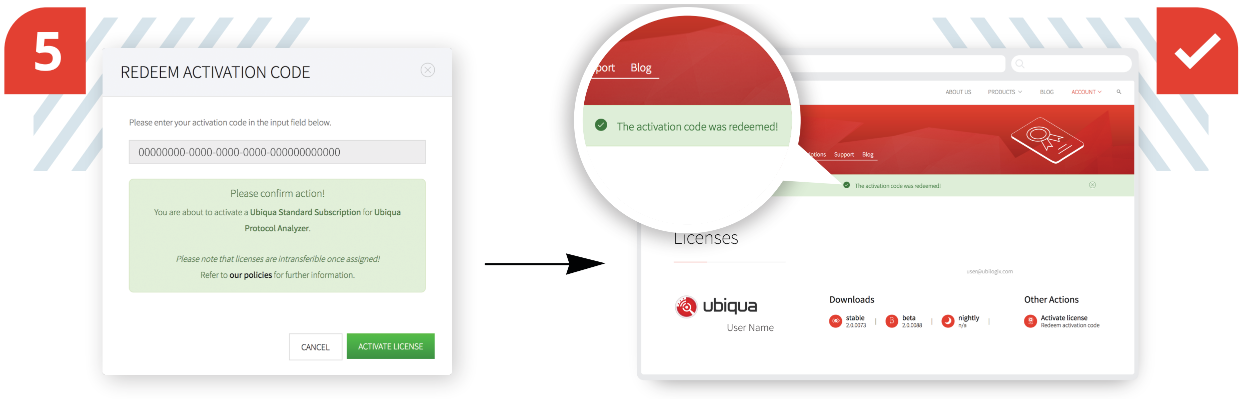
Click the Sign in button, by default you will be presented with your profile page.



From the Dashboard sub menu click the "Licenses" tab and then the "Redeem activation code" link. A dialog will appear where you can type or paste your activation code, following this action click the “Redeem Code" button.



Finally click the “Activate License” button, if the process succeeds your new Ubiqua License will be displayed, and a success message will be shown.


REDEEM ACTIVATION CODE

Please enter your activation code in the input field below.

00000000-0000-0000-0000-000000000000

Please confirm action!

You are about to activate a **Ubiqua Standard Subscription** for **Ubiqua Protocol Analyzer**.

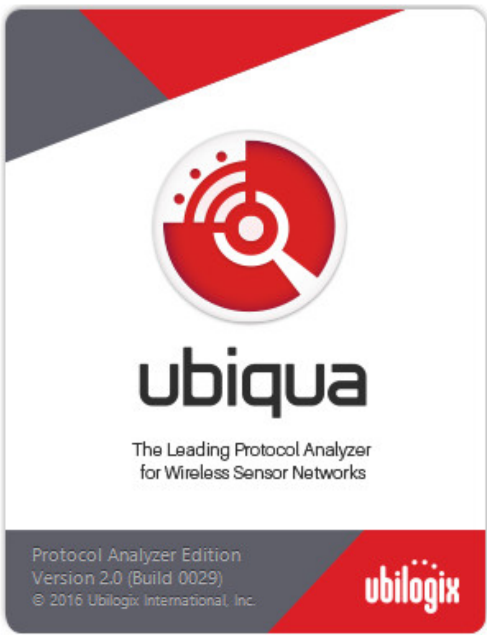
Please note that licenses are intransferible once assigned!

Refer to **our policies** for further information.

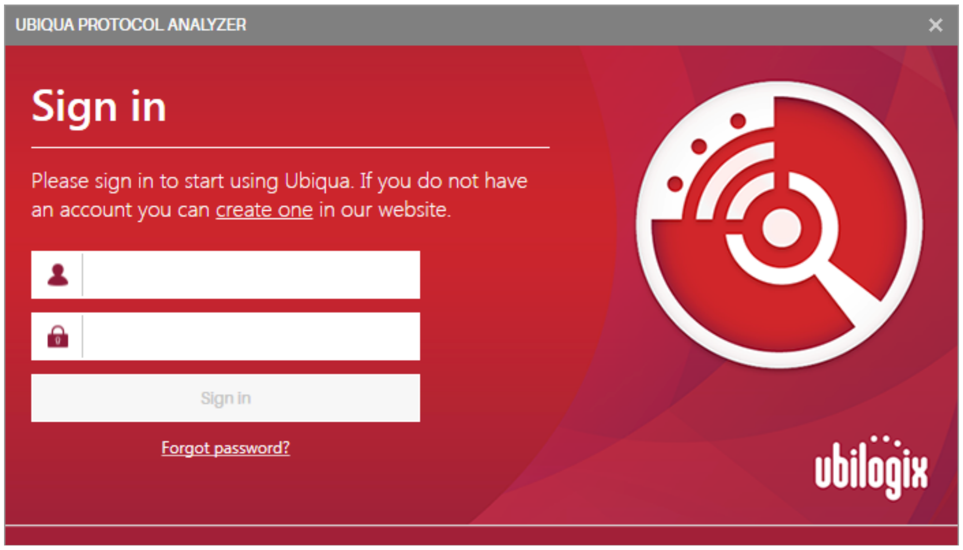
The activation code was redeemed!

Starting Ubiqua

To start Ubiqua either double-click on the Ubiqua Protocol Analyzer shortcut icon on your desktop or click on the Ubiqua Protocol Analyzer shortcut located in the Start > All Programs > Ubilogix group. A Splash screen featuring the Ubiqua logo will appear.

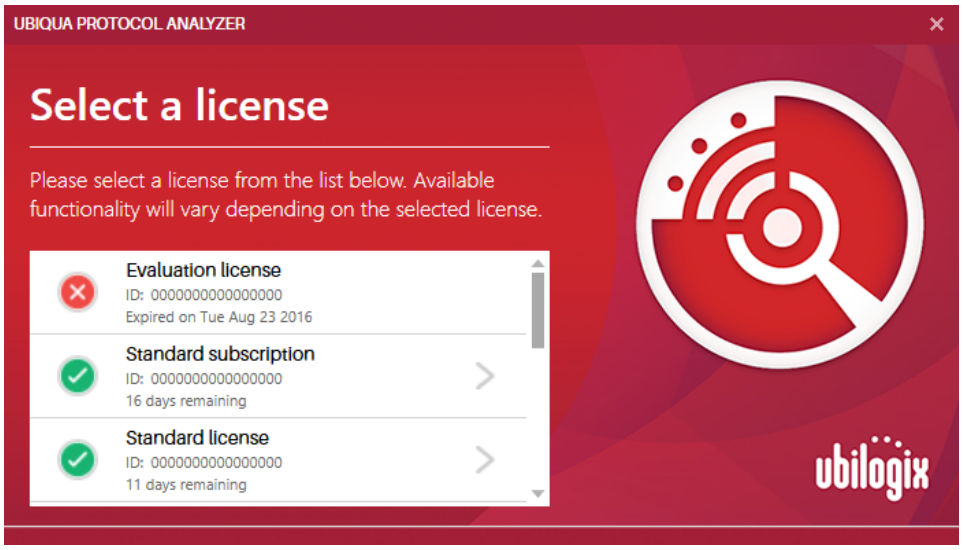

ubiqua
The Leading Protocol Analyzer for Wireless Sensor Networks
Protocol Analyzer Edition
Version 2.0 (Build 0029)
© 2016 Ubilogix International, Inc.
ubilogix

The first time you run Ubiqua the Ubiqua Authentication window and the Welcome window will appear. The Ubiqua Authentication window will ask for your Ubilogix account, write the e-mail and password of your Ubilogix account in the respective "E-mail" and "Password" text boxes. If you have an Ubilogix account but you can't remember your password, start the recover password process at: https://www.ubilogix.com/recover.



Please sign in to start using Ubiqua. If you do not have an account you can create one in our website.

From here you can select the available licenses.



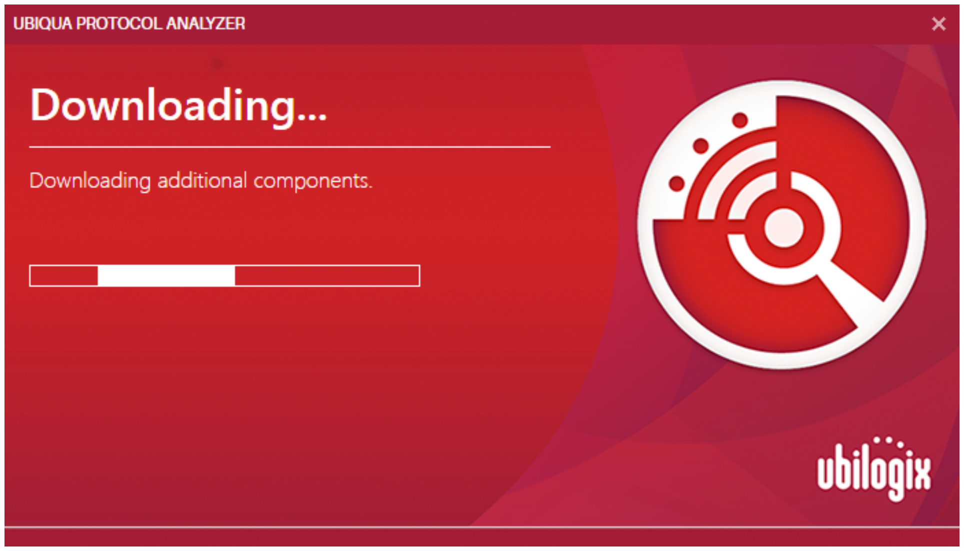
Please select a license from the list below. Available functionality will vary depending on the selected license.

Evaluation license
ID: 0000000000000000
Expired on Tue Aug 23 2016

Standard subscription
ID: 0000000000000000
16 days remaining

Standard license
ID: 0000000000000000
11 days remaining

If it is the first time you run Ubiqua it may need to connect to the Internet to download additional updated components; this process may take a few moments and then Ubiqua will be ready to run.



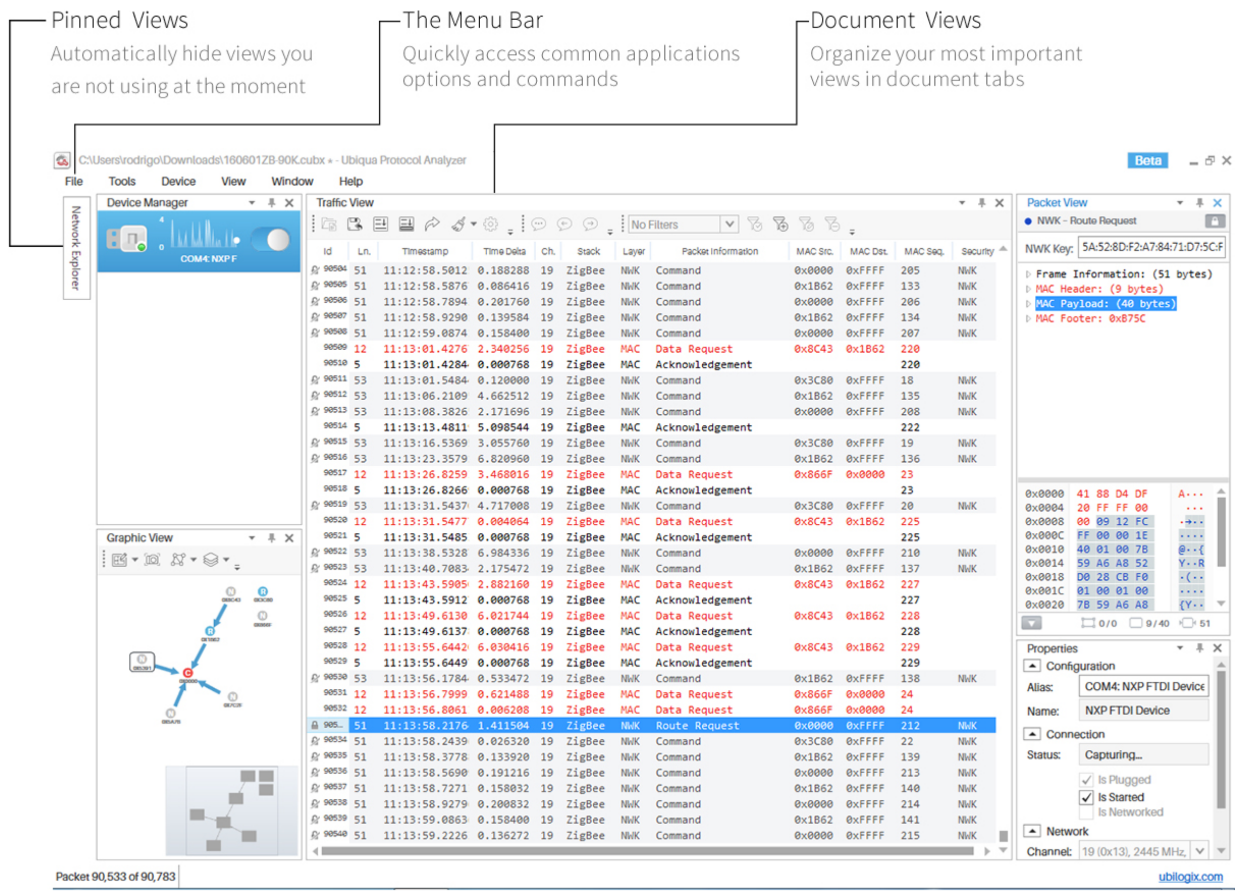
The User Interface

The Ubiqua user interface is organized in dockable panels. Each panel provides access to different sets of information and tools of the Ubiqua software. This section introduces you to this environment and will guide you through each of the visual components in the application. Further sections of this user guide discuss each component in more detail.

The main window of Ubiqua is composed of 3 sections: the menu bar, the workspace, and the status bar. The menu bar, as in other applications, offers you access to the most common options and commands available in the application. The status bar displays the current application status or shows information about the current selected packet. The most complex section is the workspace. As explained above, the many views of Ubiqua are actually dockable panels you can arrange to form the layout of your preference.

Docking Panels

Each panel has 4 dock modes: floating, dockable, auto hide, and hide. The layout of the workspace is changed by dragging and dropping the individual panels.


Pinned Views
Automatically hide views you are not using at the moment
The Menu Bar
Quickly access common applications options and commands
Document Views
Organize your most important views in document tabs
C:\Users\rodrigo\Downloads\160601ZB-90K.cubx * - Ubiqua Protocol Analyzer
Beta
File
Tools
Device
View
Window
Help
Network Explorer
Device Manager
COM4: NXP F
Graphic View
Traffic View
No Filters
Packet View
ubilogix.com

Id	Ln.	Timestamp	Time Delta	Ch.	Stack	Layer	Packet Information	MAC Src.	MAC Dst.	MAC Seq.	Security
90504	51	11:12:58.5012	0.188288	19	ZigBee	NWK	Command	0x0000	0xFFFF	205	NWK
90505	51	11:12:58.5876	0.086416	19	ZigBee	NWK	Command	0x1B62	0xFFFF	133	NWK
90506	51	11:12:58.7894	0.201760	19	ZigBee	NWK	Command	0x0000	0xFFFF	206	NWK
90507	51	11:12:58.9290	0.139584	19	ZigBee	NWK	Command	0x1B62	0xFFFF	134	NWK
90508	51	11:12:59.0874	0.158400	19	ZigBee	NWK	Command	0x0000	0xFFFF	207	NWK
90509	12	11:13:01.4276	2.340256	19	ZigBee	MAC	Data Request	0x8C43	0x1B62	220	
90510	5	11:13:01.4284	0.000768	19	ZigBee	MAC	Acknowledgement			220	
90511	53	11:13:01.5484	0.120000	19	ZigBee	NWK	Command	0x3C80	0xFFFF	18	NWK
90512	53	11:13:06.2109	4.662512	19	ZigBee	NWK	Command	0x1B62	0xFFFF	135	NWK
90513	53	11:13:08.3826	2.171696	19	ZigBee	NWK	Command	0x0000	0xFFFF	208	NWK
90514	5	11:13:13.4811	5.098544	19	ZigBee	MAC	Acknowledgement			222	
90515	53	11:13:16.5369	3.055760	19	ZigBee	NWK	Command	0x3C80	0xFFFF	19	NWK
90516	53	11:13:23.3579	6.820960	19	ZigBee	NWK	Command	0x1B62	0xFFFF	136	NWK
90517	12	11:13:26.8259	3.468016	19	ZigBee	MAC	Data Request	0x866F	0x0000	23	
90518	5	11:13:26.8266	0.000768	19	ZigBee	MAC	Acknowledgement			23	
90519	53	11:13:31.5437	4.717008	19	ZigBee	NWK	Command	0x3C80	0xFFFF	20	NWK
90520	12	11:13:31.5477	0.004064	19	ZigBee	MAC	Data Request	0x8C43	0x1B62	225	
90521	5	11:13:31.5485	0.000768	19	ZigBee	MAC	Acknowledgement			225	
90522	53	11:13:38.5328	6.984336	19	ZigBee	NWK	Command	0x0000	0xFFFF	210	NWK
90523	53	11:13:40.7083	2.175472	19	ZigBee	NWK	Command	0x1B62	0xFFFF	137	NWK
90524	12	11:13:43.5905	2.882160	19	ZigBee	MAC	Data Request	0x8C43	0x1B62	227	
90525	5	11:13:43.5912	0.000768	19	ZigBee	MAC	Acknowledgement			227	
90526	12	11:13:49.6130	6.021744	19	ZigBee	MAC	Data Request	0x8C43	0x1B62	228	
90527	5	11:13:49.6137	0.000768	19	ZigBee	MAC	Acknowledgement			228	
90528	12	11:13:55.6442	6.030416	19	ZigBee	MAC	Data Request	0x8C43	0x1B62	229	
90529	5	11:13:55.6449	0.000768	19	ZigBee	MAC	Acknowledgement			229	
90530	53	11:13:56.1784	0.533472	19	ZigBee	NWK	Command	0x1B62	0xFFFF	138	NWK
90531	12	11:13:56.7999	0.621488	19	ZigBee	MAC	Data Request	0x866F	0x0000	24	
90532	12	11:13:56.8061	0.006208	19	ZigBee	MAC	Data Request	0x866F	0x0000	24	
905...	51	11:13:58.2176	1.411504	19	ZigBee	NWK	Route Request	0x0000	0xFFFF	212	NWK
90534	51	11:13:58.2439	0.026320	19	ZigBee	NWK	Command	0x3C80	0xFFFF	22	NWK
90535	51	11:13:58.3778	0.133920	19	ZigBee	NWK	Command	0x1B62	0xFFFF	139	NWK
90536	51	11:13:58.5690	0.191216	19	ZigBee	NWK	Command	0x0000	0xFFFF	213	NWK
90537	51	11:13:58.7271	0.158032	19	ZigBee	NWK	Command	0x1B62	0xFFFF	140	NWK
90538	51	11:13:58.9279	0.200832	19	ZigBee	NWK	Command	0x0000	0xFFFF	214	NWK
90539	51	11:13:59.0863	0.158400	19	ZigBee	NWK	Command	0x1B62	0xFFFF	141	NWK
90540	51	11:13:59.2226	0.136272	19	ZigBee	NWK	Command	0x0000	0xFFFF	215	NWK

● NWK - Route Request

NWK Key: 5A:52:8D:F2:A7:84:71:D7:5C:F

```
▷ Frame Information: (51 bytes)
▷ MAC Header: (9 bytes)
▷ MAC Payload: (40 bytes)
▷ MAC Footer: 0xB75C
```

```
0x0000  41 88 D4 DF   A···
0x0004  20 FF FF 00    ···
0x0008  00 09 12 FC   ·→··
0x000C  FF 00 00 1E   ····
0x0010  40 01 00 7B   @··{
0x0014  59 A6 A8 52   Y··R
0x0018  D0 28 CB F0   ·(··
0x001C  01 00 01 00   ····
0x0020  7B 59 A6 A8   {Y··
```

0/0 9/40 51

Properties

Configuration

Alias: COM4: NXP FTDI Device

Name: NXP FTDI Device

Connection

Status: Capturing...

✓ Is Plugged
✓ Is Started
Is Networked

Network

Channel: 19 (0x13), 2445 MHz,

Packet 90,533 of 90,783

For instance, while dragging the panel over the workspace, icons will appear over the existing panels giving you the option to put the dragged panel on top, on the right, on the bottom, on the left, or as a tab of the target panel. If you drag the panel out the main window, the mode will change to "floating". To auto hide a panel in dockable mode, just click the button with the pin icon on the top right area of the panel (to the left of the close button).

Auto hidden panels will show up when the mouse cursor is over the panel tab (which is located at the far left side of the main window), and will automatically hide when you move the mouse out of the panel. In the above figure, the Watch View is in auto hide mode. To change the mode of an auto hidden panel, you must click the button with the pin icon again (it will show a rotated pin icon).

Once you have set the layout to your convenience, Ubiqua will keep the views arranged in the way you specified. You can always reset the layout to the default by using the Window > Reset Window Layout menu item. You have also the option to save the layout on a file and recover it later, to do so use the File > Save Environment... and File > Load Environment... menu items. The environment file does not only contain the layout but all application settings (for details see Setting Preferences).

The Menu Bar

The menu bar offers access to the most common actions in Ubiqua. The following is a brief description of what each menu item does and where you can find more information.

File

- **Open Capture... –** Opens a capture file and loads all the stored data which includes raw packets, security keys, addresses, graph layouts, and others (details in Capture Files).
- **Open Partial Capture** – Obtains a consecutive selection of packets of a CUBX capture file. To open partial capture, click on the Open Partial Capture option of the File menu. Select the capture file to open and click the Open button. The Select Capture Packets dialog opens and the user can select a packets range . The date and time are visible when the user hovers over the slider of the selected packet.
- **Save Capture As...** – Saves a new capture file with all the present data in the application.
- **Save Filtered Capture...** – Saves a new capture file containing all the present data in the application but only the packets available through the currently enabled filter (details in Filtering Packets).
- **Export Packets...** – Exports only the packets in the capture; excluding the Security Keys and collected Network Addresses (details in Exporting Packets).
- **Open Environment...** – Opens an environment file and loads all application settings such as views layouts, Traffic View column customizations, and others (details in Setting Preferences).
- **Save Environment...** – Saves a new environment file and stores all application settings.
- **Recent Captures** – Allows you to access the recently opened capture files; the list is limited to the last 32 files. Unexisting files will be dimmed and the "(not found)" text added at the end. To reset the list click the Clear menu item you will find at the end of this submenu.
- **Exit –** Closes the application.

Tools

- **Change Protocol Stack... –** This menu item allows you to change the protocol stack of captured packets by channel and PAN ID combinations (details in Changing Protocol Stacks).

- **Go To Packet... –** Selects the packet with the specified ID in the Traffic View (details in Go To Packet).

- **Options... –** Shows the options dialog where the application settings are set (see Setting Preferences).

Device

- **Start Device –** Starts the currently selected device on the Device Manager.

- **Protocol Stack –** Allows you to set the protocol stack used to decode incoming packets from the selected device on the Device Manager (see Configuring Devices for details).

- **Channel –** Allows you to set the channel of the selected device on the Device Manager.

- **Add Device... –** Opens the dialog used to add a new device in the manager (see Adding Devices).

- **Remove Device –** Removes the currently selected device on the Device Manager (see Removing Devices).

View

- **Traffic View –** Shows a view containing a grid with the available packets (see Traffic View).

- **Packet View –** Shows a view containing the details of the currently selected packet (see Packet View).

- **Graphic View –** Shows a view containing a graphical representation of the available packets. See Graphic View for more details about the functionality of this view.

- **Network Explorer** – Shows a view containing a tree of the identified network nodes (see Network Explorer).
- **Device Manager** – Shows a view containing a list of devices (see Device Manager).
- **Properties** – Shows a view containing the details and options for the current selection (see Configuring Devices and Customizing Nodes for more details).
- **Watch View** – Shows a view containing the current values of fields and variables (see Watch View for more details about the functionality of this view).

Window

- **Reset Window Layout** – Resets the workspace layout to its default as explained in Docking Panels.
- **Minimize To Tray** – Hides the Ubiqua window leaving only an icon in the system taskbar where you can show the window again. This option is useful for capturing packets in background mode or to control Ubiqua by the means of the services it provides (see Ubiqua Services for details).

Help

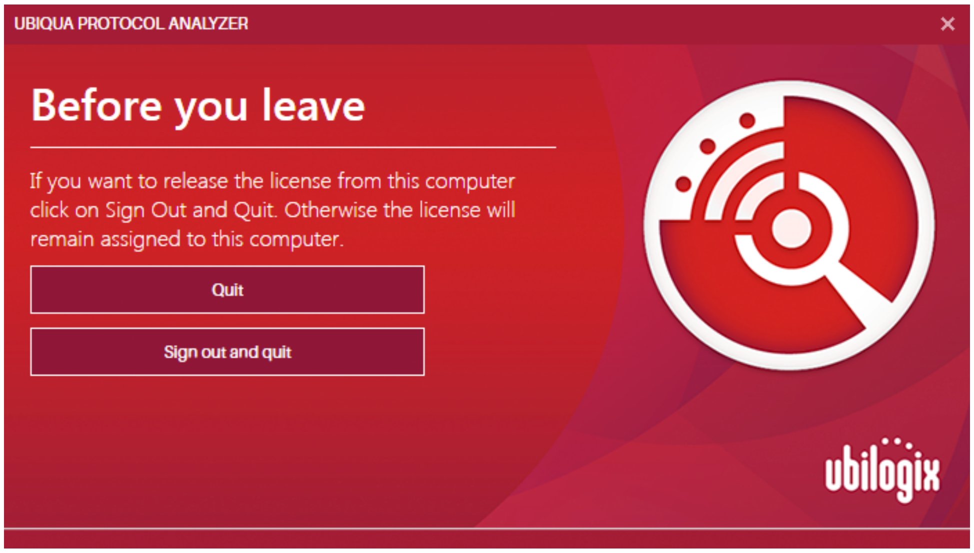
- **Documentation** – Allows you to open documentation files such as this guide.
- **Ubiqua Support Page** – Opens the Ubilogix support webpage.
- **Check for Updates...** – Checks if a new version of Ubiqua is available for download and opens a dialog showing the latest release notes and if you are eligible for the update according to your license status (see Check for Updates).
- **About Ubiqua** – Opens the application about window.

Exiting Ubiqua PA

When you have a Standard Subscription, Ubiqua PA will present a dialog with the options avaiable before clossing the application.

1. **Quit:** this option will close Ubiqua PA leaving your user authenticated so you don't have to sign in next time you start the application.

2. **Sign out and quit:** this option will release your license from the current system and let you login to another computer with the same license.


UBIQUA PROTOCOL ANALYZER

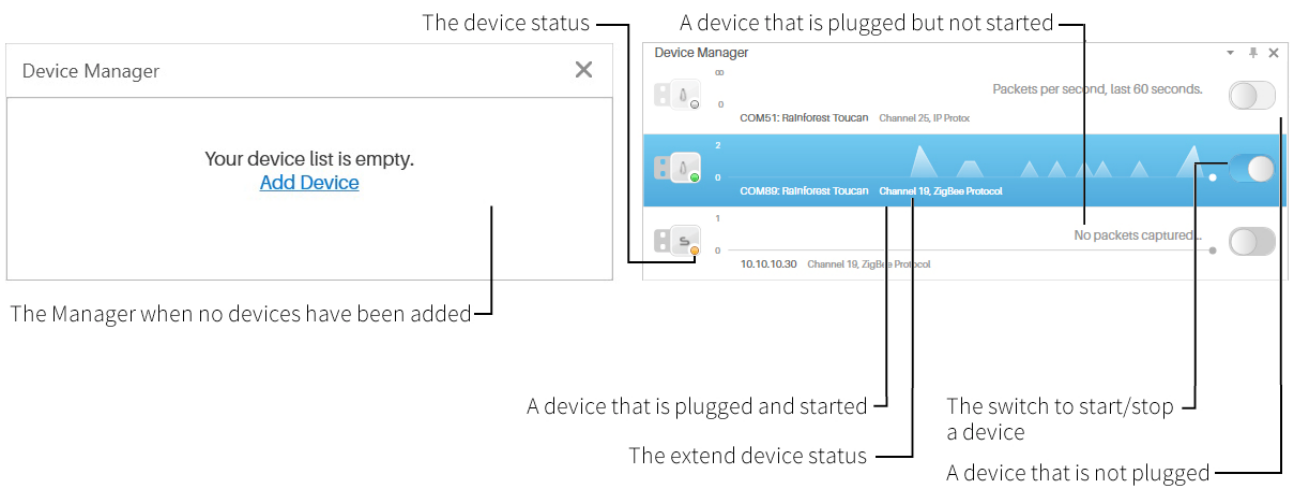
Before you leave

If you want to release the license from this computer click on Sign Out and Quit. Otherwise the license will remain assigned to this computer.

Chapter 3: Device Manager

The Device Manager lists and controls the devices used as a source of packets in the application. If the view is not visible, show it by selecting the Views > Device Manager menu item. The devices list will be empty the first time you open the manager. You need to populate this list by manually adding each device as needed (see the "Adding Devices" section for instructions).

Once the list has been populated, each device will be displayed in an individual control containing the following information: vendor icon, alias, status, extended status, and a switch to change the device status (see the figure below). The status will display "Offline" if the device is not plugged to your computer, "Idle" if it is plugged but not started, and "Capturing..." when it is plugged and started. The extended status will display additional information related to the role of the device in the application. For instance, the extended status of a sniffer device capturing packets will show the channel and protocol stack used for decoding.



The device status
A device that is plugged but not started
Device Manager
Your device list is empty.
Add Device
The Manager when no devices have been added
Device Manager
Packets per second, last 60 seconds.
COM51: Rainforest Toucan
Channel 25, IP Protoc
COM89: Rainforest Toucan
Channel 19, ZigBee Protocol
No packets captured...
10.10.10.30
Channel 19, ZigBee Protocol
A device that is plugged and started
The extend device status
The switch to start/stop a device
A device that is not plugged

Adding Devices

To add a new device to the list, right click inside the Device Manager view and select the Add Device... menu item, or click the Add Device link on the "Your device list is empty" message (if that is your case). Before opening the Add Device dialog make sure your device is compatible with Ubiqua (see Supported Sniffer Hardware for details), then open the dialog (see the figure below), and follow the next steps:

1. Connect the device to your computer.

2. The Found New Hardware wizard might appear. In such case, follow the instructions on the screen to install the device drivers. If you need to manually specify the location of drivers, you will find them in the **Program Files\Ubilogix\Drivers** folder.

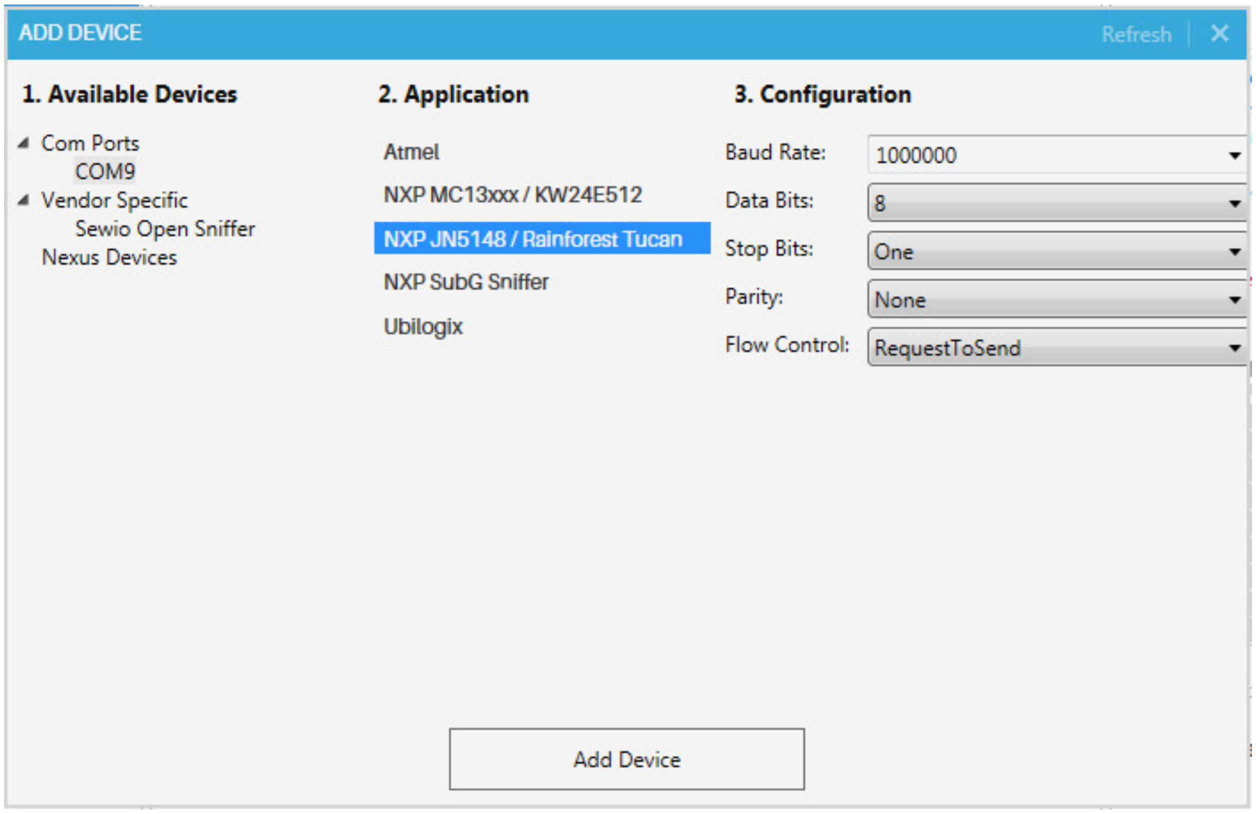
3. Wait a couple of seconds while the driver is installed and the device recognized.

4. Available Devices section, select your device. After the selection is made, Ubiqua will fill in the Application section with specific vendors.

5. Select the device you want to add. If no devices are found try searching again by clicking the Refresh button at the top right corner of the Add Device window.

6. From the Application combo box select the appropriate device application.

7. Click the Add Device button.



Baud Rate: 1000000
Data Bits: 8
Stop Bits: One
Parity: None
Flow Control: RequestToSend

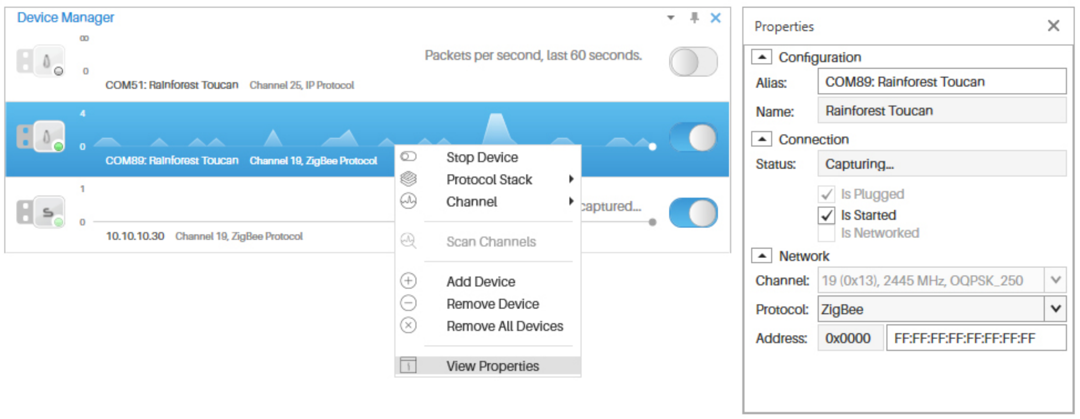
It is also possible to add devices by specifying custom values for properties such as baud rate, data bits, stop bits and others.

When setting up Sewio Open sniffer for the first time, we must configure the sniffer via its web interface, aditional information can be found in http://www.ubilogix.com/ubiqua/documentation/guide/sewio-hardware/.

Configuring Devices

Once you have installed the drivers and added the device to the list, you can configure its properties by right-clicking on the device and selecting the View Properties menu item (see the figure below). This will open the Properties view, where the properties can be edited. As long as the Properties view is open, it will show the properties of the selected device in the Device Manager.

Please keep in mind that if the protocol or channel is changed, and the device is running, you must stop and start it again. Ubiqua provides a shortcut for this, the Protocol Stack and Channel menu items of the device contextual menu. Right click on a device and make your selection, Ubiqua will automatically restart the device if it was running.items of the device contextual menu. Right click on a device and make your selection, Ubiqua will automatically restart the device if it was running.



Alias: COM89: Rainforest Toucan
Name: Rainforest Toucan
Status: Capturing...
Is Plugged
[✓] Is Started
Is Networked
Channel: 19 (0x13), 2445 MHz, OQPSK_250
Protocol: ZigBee
Address: 0x0000 FF:FF:FF:FF:FF:FF:FF:FF

Capturing Packets

To start capturing packets just click on the device switch button or select the Start/Stop Device on the contextual menu. If a problem was found starting the device and you can't stop or start it anymore, proceed to unplug the device and plug it back to your system; usually, this will resolve the problem. If you keep experiencing problems, make sure the port is not being used by another program in your system and try again. Also, please note that according to your license you could be limited to a certain number of devices started at the same time. Whatever the cause of the problem is, the extended status of the device in question will display a possible solution.

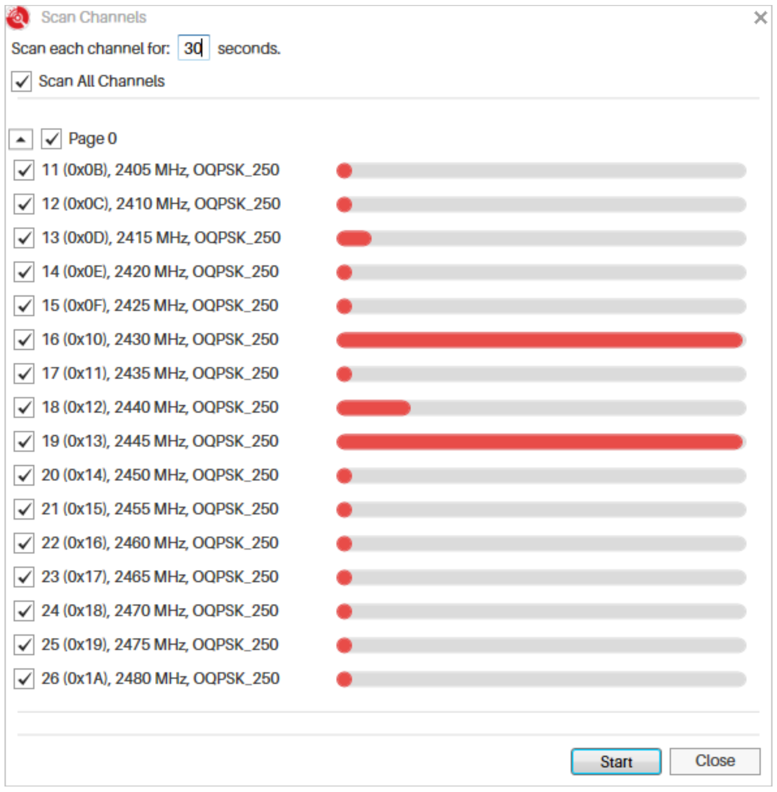
Removing Devices

Devices on the list are maintained by the application and stored in environment files. To remove a device right click on it and select the Remove Device menu item. If you want to remove all devices right click on the Device Manager and select the Remove All Devices menu item. Please note that this action can't be undone.

Scanning Channels

Scan channels allows capturing through all channels or the selected channels. Each selected channel will capture during a specific time, at the end, Ubiqua will show the number of captured packets per channel on a graph.

To scan channels, right click on the device and select the Scan Channels... menu item. After the Scan Channels dialog opens, select the channels to scan and set the scanning time per channel; to start the scanning process press the Start button. The graph will display the packets captured per channel when the you place the mouse over any channel. Note that the packets captured during the scanning process may appear in the Traffic View and that the channel list may be different, depending on the capabilities of the device used for scanning.



Scan Channels

Scan each channel for: 30 seconds.

✓ Scan All Channels

✓ Page 0

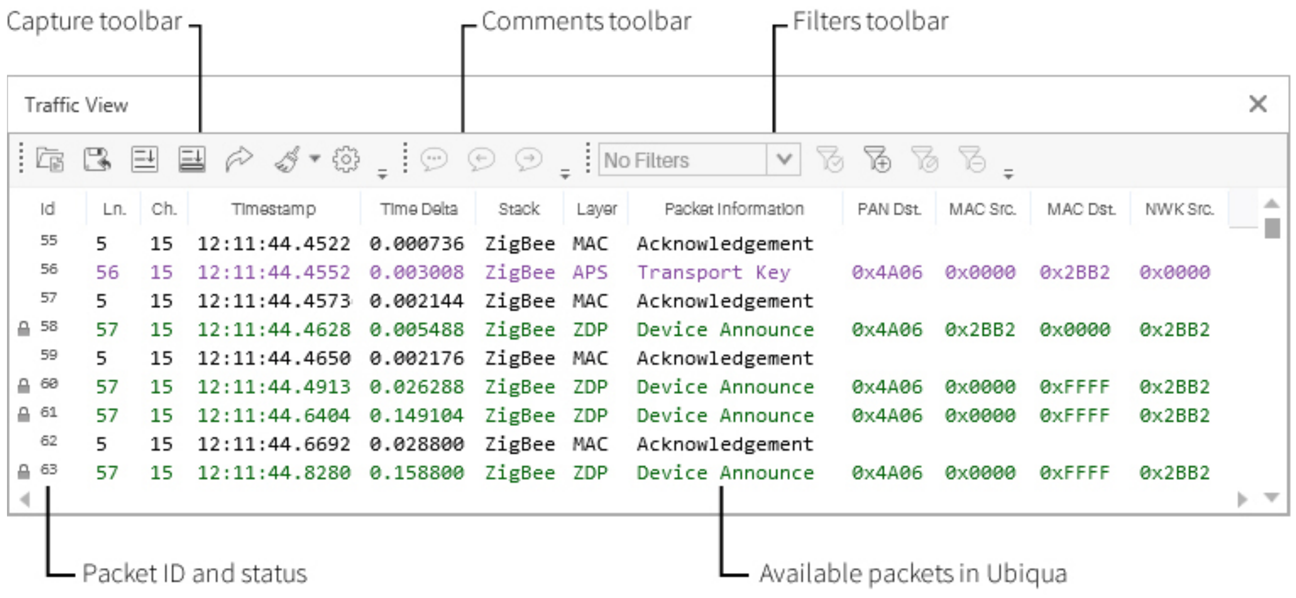
✓ 11 (0x0B), 2405 MHz, OQPSK_250
✓ 12 (0x0C), 2410 MHz, OQPSK_250
✓ 13 (0x0D), 2415 MHz, OQPSK_250
✓ 14 (0x0E), 2420 MHz, OQPSK_250
✓ 15 (0x0F), 2425 MHz, OQPSK_250
✓ 16 (0x10), 2430 MHz, OQPSK_250
✓ 17 (0x11), 2435 MHz, OQPSK_250
✓ 18 (0x12), 2440 MHz, OQPSK_250
✓ 19 (0x13), 2445 MHz, OQPSK_250
✓ 20 (0x14), 2450 MHz, OQPSK_250
✓ 21 (0x15), 2455 MHz, OQPSK_250
✓ 22 (0x16), 2460 MHz, OQPSK_250
✓ 23 (0x17), 2465 MHz, OQPSK_250
✓ 24 (0x18), 2470 MHz, OQPSK_250
✓ 25 (0x19), 2475 MHz, OQPSK_250
✓ 26 (0x1A), 2480 MHz, OQPSK_250

Chapter 4: Traffic View

The Traffic View is the most feature-rich component in Ubiqua. This chapter describes the full functionality of the Traffic View including instructions on topics such as: how to manage capture files, the actions that can be performed with packets selected on the grid, how to filter packets, and other related features.

The Traffic View is mainly composed of 2 components: a grid, and a set of toolbars (see the figure below). The grid shows all the packets captured with devices or loaded from capture files. Toolbars provide access to most of the functionality available throughout the system and because of this, the Traffic View can be seen as the central point of Ubiqua. Many of the actions performed on it produce changes or updates in other views. For instance, selecting a packet on the grid changes the Packet View contents. Also, actions in other views could cause the Traffic View to update its contents. For instance, starting a sniffer in the Device Manager will cause the grid to update itself to show incoming packets.

The following image depicts a traffic view with some packets captured. Also, note that the first column shows an icon that depicts additional information about the packet on its corresponding row, for example whether there's a comment for the packet (call out icon), if there was an error when decoding (cross mark icon) or if the packet was encrypted (closed lock) and the decryption was successful (open lock).



Id	Ln.	Ch.	Timestamp	Time Delta	Stack	Layer	Packet Information	PAN Dst.	MAC Src.	MAC Dst.	NWK Src.
55	5	15	12:11:44.4522	0.000736	ZigBee	MAC	Acknowledgement				
56	56	15	12:11:44.4552	0.003008	ZigBee	APS	Transport Key	0x4A06	0x0000	0x2BB2	0x0000
57	5	15	12:11:44.4573	0.002144	ZigBee	MAC	Acknowledgement				
🔒 58	57	15	12:11:44.4628	0.005488	ZigBee	ZDP	Device Announce	0x4A06	0x2BB2	0x0000	0x2BB2
59	5	15	12:11:44.4650	0.002176	ZigBee	MAC	Acknowledgement				
🔒 60	57	15	12:11:44.4913	0.026288	ZigBee	ZDP	Device Announce	0x4A06	0x0000	0xFFFF	0x2BB2
🔒 61	57	15	12:11:44.6404	0.149104	ZigBee	ZDP	Device Announce	0x4A06	0x0000	0xFFFF	0x2BB2
62	5	15	12:11:44.6692	0.028800	ZigBee	MAC	Acknowledgement				
🔒 63	57	15	12:11:44.8280	0.158800	ZigBee	ZDP	Device Announce	0x4A06	0x0000	0xFFFF	0x2BB2

Color Codes

The Traffic View provides a helpful coloring scheme to easily identify the layer and protocol for each packet captured. The following table lists the available options:

Layer	Color Name	Protocol
Dark Green	TCP	Thread, ZigBee IP, IP
Dark Green	UDP	Thread, ZigBee IP, IP
Dark Green	ICMPv6	Thread, ZigBee IP, IP
Dark Green	ICMPv6	Thread, ZigBee IP, IP
Dark Green	PANA	Thread, ZigBee IP, IP
Dark Green	mDNS	Thread, ZigBee IP, IP
Dark Green	MLE	ZigBee IP, IP
Orange	MLE	Thread
Light Green	CoaP	Thread, ZigBee IP, IP
Dark Blue	Ethernet	Thread, ZigBee IP, IP
Purple	HTTP	Thread, ZigBee IP, IP
Orange	SE2	ZigBee IP, IP
Orange	JenNet-IP	JenNet-IP
Gray	MAC-Beacon	ZigBee IP, Thread, ZigBee
Brown	Mac-Data	ZigBee IP, Thread, ZigBee
Black	MAC-Acknowledgement	ZigBee IP, Thread, ZigBee
Red	MAC-Command	ZigBee IP, Thread, ZigBee
Red	NetBios	ZigBee IP, Thread, ZigBee
Gray	PopNet- Beacon	PopNet
Brown	PopNet- Mac-Data	PopNet
Black	PopNet- MAC-Acknowledgement	PopNet
Red	PopNet- MAC-Command	PopNet
Light Green	PopNet-APP	PopNet
LightBlue	PopNet-NWK	PopNet

Layer	Color Name	Protocol
Black	Pop-Nwk_Acknowledgement	PopNet
Purple	DHCPv6	Thread, ZigBee IP, IP
Purple	DTLS	Thread, ZigBee IP, IP
DarkGreen	ZDP	ZigBee
Light Green	ZCL	ZigBee
Purple	APS	ZigBee
Dark Blue	NWK	ZigBee
Dark Blue	NWK-GP	ZigBee
LightBlue	6LowPAN	ZigBee IP, Thread
DarkGreen	EAP	ZigBee IP
LightBlue	IPv4	Thread, ZigBee IP, IP
Red	IPv6	ZigBee IP, Thread, ZigBee, IP

Capture Files

Ubiqua uses capture files not only to store the packets you see in the Traffic View, but also to store other data such as the layout and settings of the nodes in the Graphic View, or the security keys used for decoding. When saving a capture, the data available in all views is retrieved and stored into a new file. Note that this process does not store decoded data, so when you open a capture file all the stored packets will be decoded again to populate data in all views.

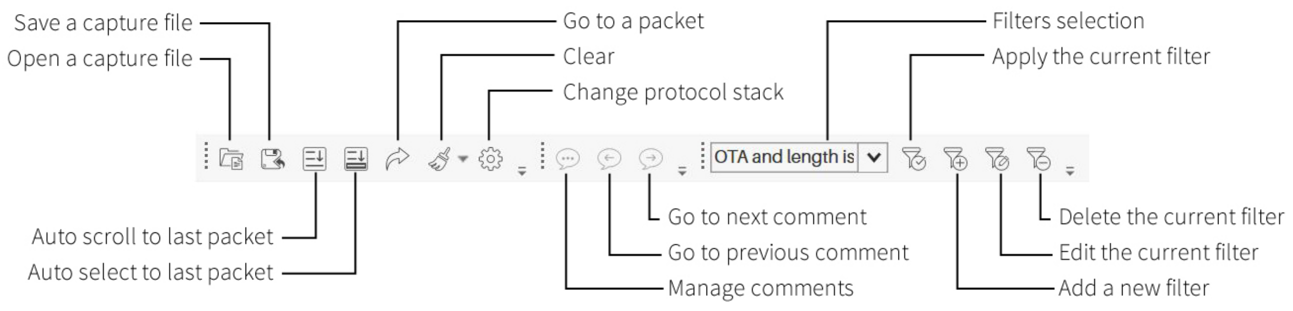
Saving Capture Files

To save the available capture data into a new file, follow the next steps:

1. Start the Save As dialog by either selecting the File > Save Capture As... menu item, clicking the Save Capture toolbar button (see the figure below), or pressing Ctrl+S on your keyboard.

2. Select the location where you want to store the new capture file, specify the file name, and press the Save button. Ubiqua capture files have the `.cubx` file extension but you can also save the capture in the `.subx` and `.pcap` file formats.

3. A progress dialog will appear showing the status of the save process.



Opening Capture Files

The process to open a capture file is very similar to saving:

1. Start the Open dialog by either selecting the File > Open Capture menu item, clicking the Open Capture toolbar button (see the above figure), or pressing Ctrl+O on your keyboard.

2. Select or specify the capture file and press the Open button. Additionally to its own `.cubx` format, Ubiqua also supports opening captures in a number of other file formats such as `.dcf` and `.pcap`. If the capture file does not hold Protocol information at the time a dialog opens with the available protocols and the user can select one of them to decode the capture.

3. A progress dialog will appear showing the status of the opening process. Note that depending on the file size (directly related to the number of packets stored), this process may take some time as the contained packets are being decoded on the fly to populate data in the corresponding views.

Auto Scroll and Selection

The Traffic View features two options that are useful to track the latest packet on the grid, this options are Auto Scroll and Auto Select. The Auto Scroll option moves the scroll position to bring into view the latest packet captured, while the Auto Select option selects the latest captured package (which also brings its contents into the Packet View). To activate or deactivate this options, use its corresponding toolbar toggle buttons (see the above figure).

Clear

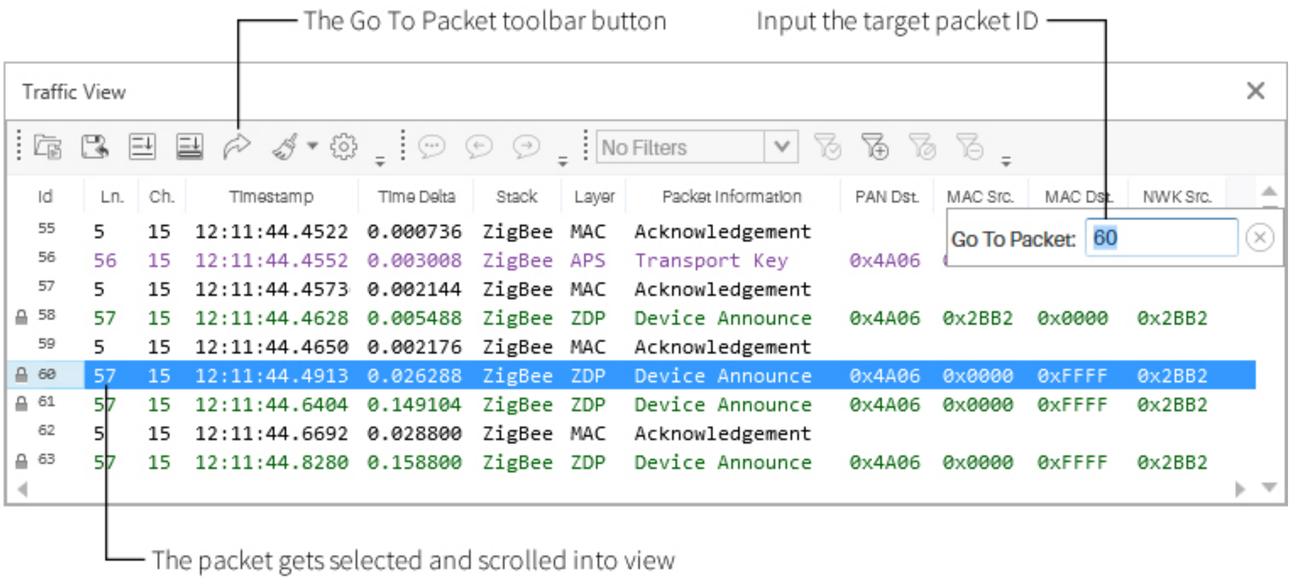
The clear functionality resets all the information available for 4 different types of data (see below). You can choose any or all types of data to clear by clicking the down arrow next to the Clear button.

This functionality is accessible from the Clear toolbar button, and you can choose among what types of data to clear by clicking the down arrow next to the Clear button (see the figure above).

The types of data to clear are:

- **Clear Packets** – Removes all the packets in the Traffic View.
- **Clear Nodes** – Clears the nodes information in the Graphic View, the Network Explorer and the properties window.
- **Clear Security Keys** – Removes the security keys stored in the keychain. (For more info see security keys).
- **Clear Addresses** – Removes the network addresses listed in the addresses table in the Security tab of the Options window.

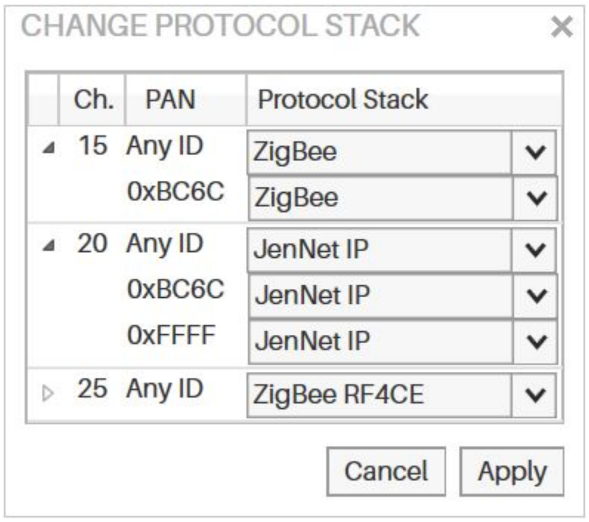
Go To Packet


The Go To Packet toolbar button
Input the target packet ID
Traffic View
No Filters
Go To Packet: 60
The packet gets selected and scrolled into view

Id	Ln.	Ch.	Timestamp	Time Delta	Stack	Layer	Packet Information	PAN Dst.	MAC Src.	MAC Dst.	NWK Src.
55	5	15	12:11:44.4522	0.000736	ZigBee	MAC	Acknowledgement				
56	56	15	12:11:44.4552	0.003008	ZigBee	APS	Transport Key	0x4A06			
57	5	15	12:11:44.4573	0.002144	ZigBee	MAC	Acknowledgement				
58	57	15	12:11:44.4628	0.005488	ZigBee	ZDP	Device Announce	0x4A06	0x2BB2	0x0000	0x2BB2
59	5	15	12:11:44.4650	0.002176	ZigBee	MAC	Acknowledgement				
60	57	15	12:11:44.4913	0.026288	ZigBee	ZDP	Device Announce	0x4A06	0x0000	0xFFFF	0x2BB2
61	57	15	12:11:44.6404	0.149104	ZigBee	ZDP	Device Announce	0x4A06	0x0000	0xFFFF	0x2BB2
62	5	15	12:11:44.6692	0.028800	ZigBee	MAC	Acknowledgement				
63	57	15	12:11:44.8280	0.158800	ZigBee	ZDP	Device Announce	0x4A06	0x0000	0xFFFF	0x2BB2

This feature is similar to the Auto Scroll and Auto Select options. The difference is that this option will scroll and then select, not the latest packet, but the packet with the ID you specify. To start using this option, click the Go To Packet toolbar button (see the figure above), or press the Ctrl+G keys on your keyboard. A pop-up box will appear at the top right area of the Traffic View. Type the ID of the packet you want to look and then press Enter. The grid will scroll and the packet will be selected. If the ID you specified does not correspond to any of the available packets you will hear an exclamation sound. Close the pop-up by clicking the × button or pressing the Esc key.

Changing Protocol Stacks



Ch.	PAN	Protocol Stack
15	Any ID	ZigBee
	0xBC6C	ZigBee
20	Any ID	JenNet IP
	0xBC6C	JenNet IP
	0xFFFF	JenNet IP
25	Any ID	ZigBee RF4CE

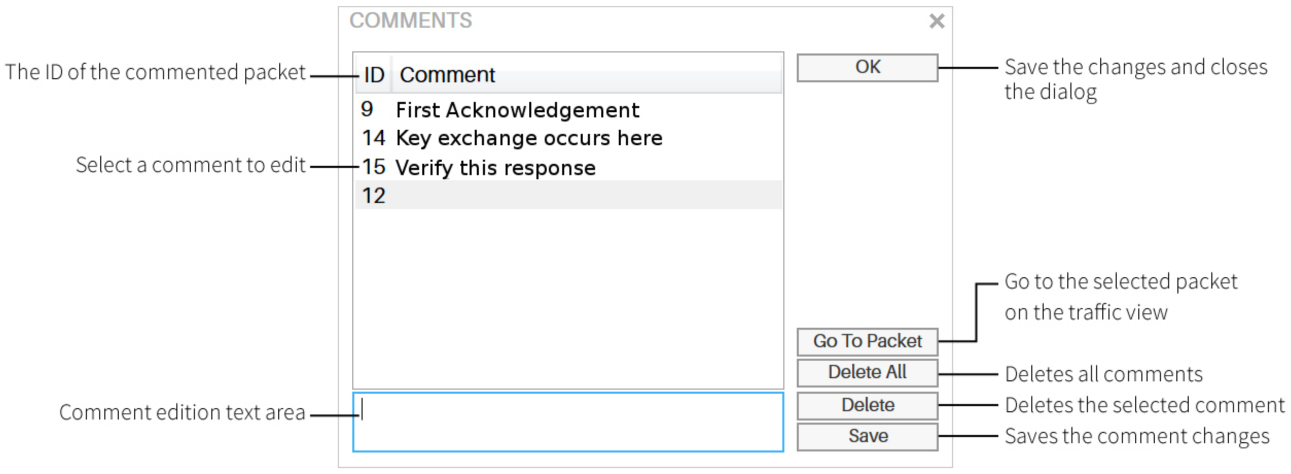
This feature allows you to change the protocol stack of the packets already available in Ubiqua. The change protocol process will clear all the current capture data, apply the changes you selected, and then re-decode all the packets, producing new capture data. To start the process you must stop all capturing devices, then click the Change Protocol Stack tool bar button or press the Ctrl+H keys on your keyboard. The Change Protocol Stack dialog will appear (see the figure above). On it, you will have the choice to set what protocol stack will be used to re-decode the packets on a given channel (for a list of the available stacks see Supported Protocols). There is also the option to set a new protocol stack in combinations of target channels and PAN IDs, giving you a fine-grained control over the changes. Once you have selected the new stacks, press the Apply button. After this point, all capture data will be cleared and the packets will be re-decoded using your settings (a progress bar will show you the status).

Please note that devices retain their own protocol setting. The change protocol stacks feature only affects previously captured packets.

Commenting Packets

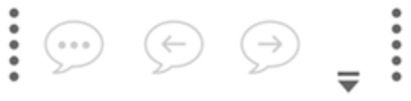
Ubiqua features the ability to annotate the packets shown in the Traffic View. Capture files store this information so you will be able to share insights about packets with other Ubiqua users, or just give you the opportunity to store notes about your capture analysis.

To start using comments, select a packet on the grid. Right click on it to open the contextual menu and select the Add Comment menu item. The Comments dialog will appear (see the figure below), write your notes and press the OK button. To see the comment, just move the mouse cursor over the row header and it will display the comment on a tooltip. To delete a packet's comment from the grid, right click on it, and select the Delete Comment menu item.


COMMENTS
The ID of the commented packet
Select a comment to edit
Comment edition text area
OK
Go To Packet
Delete All
Delete
Save
Save the changes and closes the dialog
Go to the selected packet on the traffic view
Deletes all comments
Deletes the selected comment
Saves the comment changes

ID	Comment
9	First Acknowledgement
14	Key exchange occurs here
15	Verify this response
12	

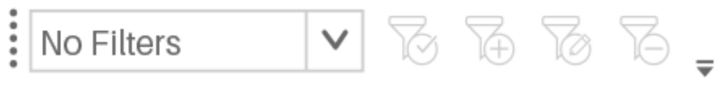
The comments toolbar has 3 buttons (see image below), the first one opens the Comments dialog (where you can edit or delete the comments) and the Go To Previous/Next Comment buttons that you can use to move the scroll of the grid and select the previous or next commented packet.



Filtering Packets

There are certain times in which you don't want to see all the captured packets but a subset of them. To help with overload, Ubiqua features the ability to filter packets and show only the ones that fulfill a certain logical expression.

Filters are managed through the filters toolbar at the top of the Traffic View (see below).

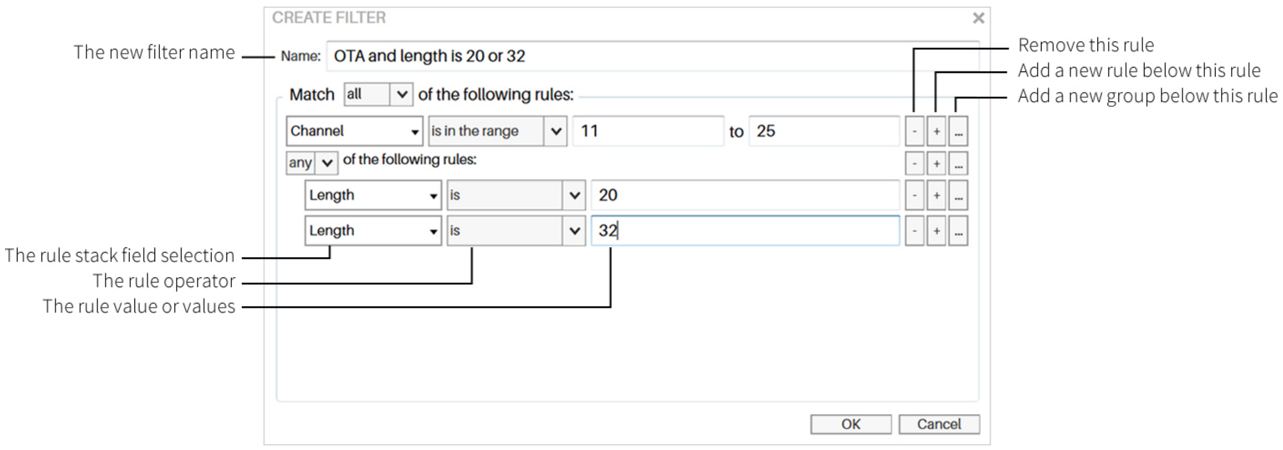


To create a new filter click the Add Filter toolbar button. The Add Filter dialog will appear (see the figure below). You will find it very similar to the ones used to create playlists on music players. The dialog allows you to define filters by combining a set of rules grouped by a "match all" or "match any" operations. Each rule is composed of a field, an operator, and a set of values. Also, you have the possibility to specify subsets of rules for complex scenarios. A field is an element whose value can be obtained by decoding the packet, the operator defines the condition the field value must comply, and the set of values (which can vary depending on the used operator and the data type of the field) specify what are the expression values that will be used to evaluate the rule.

The figure below shows an example. The "OTA and length is 20 or 32" filter is composed of a simple rule and a rule group. Both rules must be matched. The first one asks to filter packets whose Channel is in the range of 11 to 25. The rule group asks to match any of 2 sub-rules, the first is that Length is equal to 20, and the second is that Length is equal to 32.

Once that filters have been created, select one from the Filters selection at the Filters toolbar and click the Apply Filter toolbar button. If there is not enough decoded information available to compute the filter, packets will be decoded again to retrieve any additional data needed to finally apply the filter. Once that filters have been applied, Ubiqua will only show the packets that the filter allows even when capturing new packets from devices. To edit the selected filter click the Edit Filter toolbar button, to remove it, click the Delete Filter toolbar button.

Filters are not stored on capture files. They will be stored on the environment and they will be available as long as you don't delete them. As with the layout of the views, they can be stored on an Environment File (more details on this in Setting Preferences).



Name: OTA and length is 20 or 32

Match all of the following rules:

Channel is in the range 11 to 25

any of the following rules:

Length is 20

Length is 32

Exporting Packets

f you need to extract packets of your capture and you need them on a different file format (not just a capture file), you can export the selected packets on the Traffic View. To start the export process, select the packets you want to export, Go to the File menu and choose the Export Packets item. Note that in order to select packets in the Traffic View, you must disable both the "Auto Select Last Packet" and the "Auto Scroll" options.

You can export either all the packets or only the selected packets in the following file formats: `.xls` (Microsoft Excel spreadsheets), `.csv` (Comma Separated Values), `.opml` (Outline Processor Markup Language), `.txt` (simple text files) and `.cubx` (Ubilogix capture files). The `.opml` and `.txt` file formats will include all the decoded data of the packets while the `.xls` and `.csv` file formats will only include the data shown in the columns of the Traffic View. The `.cubx` file format will include the packets with the exception of the Security Keys and the Network Addresses.

Copying a Packet

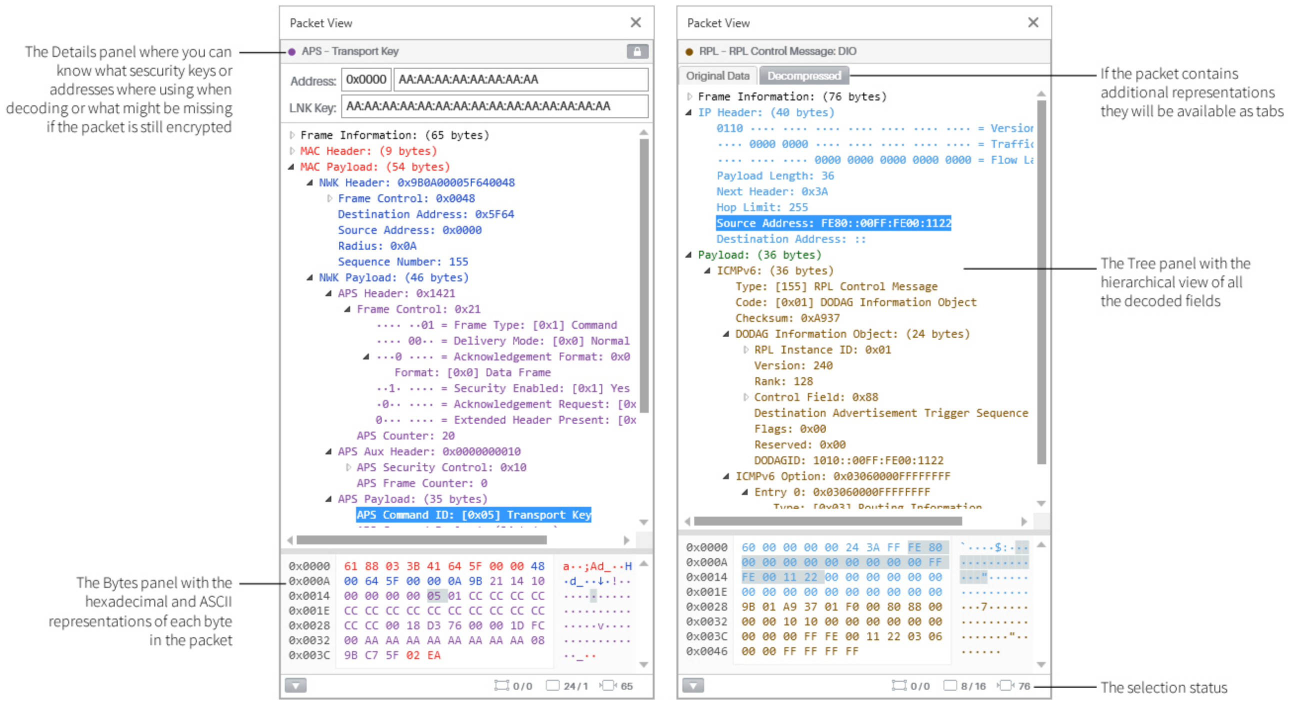
You can copy a packet as it appears in the Traffic View or the Packet View's tree by selecting the packet you want to copy; right click on the selection and choose the Copy menu item. The clipboard will store the selected packet data and it can be pasted in any text editor. When traffic is being captured in fast rates, it is recommended to disable the "Auto Scroll" and "Auto Select Last" options to easily select the packet from the Traffic View.

Chapter 5: Packet View

The Packet View displays the decoded packet data along with its raw contents. The view is composed of two panels, a status bar and an optional Description panel for when the packet contents are still encrypted. (see figure below). At the top of the Packet View, the Title bar shows details such as the security keys and network addresses used to decrypt the packet. Below the Title bar, the First panel contains a tree; the tree is a hierarchical representation of all the decoded fields of the packet currently selected on the Traffic View. Finally, the second panel is the Bytes panel; this panel displays the hexadecimal and ASCII representations for each byte of the packet.

The panels are linked together so if you select a tree node, the corresponding bytes for the field will be highlighted on the Bytes panel and the other way around. Finally, the status bar of the view indicates the offset and length values relative to the full packet, to the selected fields of the tree node, or to the selected panel cells.

In the Title bar there's a "lock" icon (see the figure below), when clicked it shows the optional panel that shows the key for decrypting the packed data or if not accessible allows the user to type the decryption key.



The Details panel where you can know what sescurity keys or addresses where using when decoding or what might be missing if the packet is still encrypted
The Bytes panel with the hexadecimal and ASCII representations of each byte in the packet
If the packet contains additional representations they will be available as tabs
The Tree panel with the hierarchical view of all the decoded fields
The selection status
Packet View
Packet View
Original Data
Decompressed

```
● APS – Transport Key
Address: 0x0000  AA:AA:AA:AA:AA:AA:AA:AA
LNK Key: AA:AA:AA:AA:AA:AA:AA:AA:AA:AA:AA:AA:AA:AA:AA:AA

▷ Frame Information: (65 bytes)
▷ MAC Header: (9 bytes)
◢ MAC Payload: (54 bytes)
   ◢ NWK Header: 0x9B0A00005F640048
      ▷ Frame Control: 0x0048
        Destination Address: 0x5F64
        Source Address: 0x0000
        Radius: 0x0A
        Sequence Number: 155
   ◢ NWK Payload: (46 bytes)
      ◢ APS Header: 0x1421
         ◢ Frame Control: 0x21
              .... ..01 = Frame Type: [0x1] Command
              .... 00.. = Delivery Mode: [0x0] Normal
            ◢ ...0 .... = Acknowledgement Format: 0x0
                Format: [0x0] Data Frame
              ..1. .... = Security Enabled: [0x1] Yes
              .0.. .... = Acknowledgement Request: [0x
              0... .... = Extended Header Present: [0x
           APS Counter: 20
      ◢ APS Aux Header: 0x0000000010
         ▷ APS Security Control: 0x10
           APS Frame Counter: 0
      ◢ APS Payload: (35 bytes)
           APS Command ID: [0x05] Transport Key

0x0000  61 88 03 3B 41 64 5F 00 00 48   a..;Ad_..H
0x000A  00 64 5F 00 00 0A 9B 21 14 10   .d_..↓.!..
0x0014  00 00 00 00 05 01 CC CC CC CC   ..........
0x001E  CC CC CC CC CC CC CC CC CC CC   ..........
0x0028  CC CC 00 18 D3 76 00 00 1D FC   .....v....
0x0032  00 AA AA AA AA AA AA AA AA 08   ..........
0x003C  9B C7 5F 02 EA                  .._..

0/0   24/1   65
```

```
● RPL – RPL Control Message: DIO

▷ Frame Information: (76 bytes)
◢ IP Header: (40 bytes)
    0110 .... .... .... .... .... .... .... = Versior
    .... 0000 0000 .... .... .... .... .... = Traffic
    .... .... .... 0000 0000 0000 0000 0000 = Flow La
    Payload Length: 36
    Next Header: 0x3A
    Hop Limit: 255
    Source Address: FE80::00FF:FE00:1122
    Destination Address: ::
◢ Payload: (36 bytes)
   ◢ ICMPv6: (36 bytes)
       Type: [155] RPL Control Message
       Code: [0x01] DODAG Information Object
       Checksum: 0xA937
     ◢ DODAG Information Object: (24 bytes)
        ▷ RPL Instance ID: 0x01
          Version: 240
          Rank: 128
        ▷ Control Field: 0x88
          Destination Advertisement Trigger Sequence
          Flags: 0x00
          Reserved: 0x00
          DODAGID: 1010::00FF:FE00:1122
     ◢ ICMPv6 Option: 0x03060000FFFFFFFF
        ◢ Entry 0: 0x03060000FFFFFFFF

0x0000  60 00 00 00 00 24 3A FF FE 80   `....$:...
0x000A  00 00 00 00 00 00 00 00 00 FF   ..........
0x0014  FE 00 11 22 00 00 00 00 00 00   ..."......
0x001E  00 00 00 00 00 00 00 00 00 00   ..........
0x0028  9B 01 A9 37 01 F0 00 80 88 00   ...7......
0x0032  00 00 10 10 00 00 00 00 00 00   ..........
0x003C  00 00 00 FF FE 00 11 22 03 06   ......."..
0x0046  00 00 FF FF FF FF               ......

0/0   8/16   76
```

Each panel in the Packet View has its own context menu. Through the tree context menu you can copy (Ctrl+C) the selected field in the default format or in additional formats by using the Copy As options, also, you can create or edit filters (see Creating Quick Filters). The context menu in the Bytes panel allows copying the selected bytes and also you can navigate to the parent field of the selection by pressing the Esc key.

You are not restricted to only one Packet View instance at a time, you can open as many as you need. Just double click on a packet in the Traffic View to launch a Packet View of the same.

Testing Decryption

The Title bar will always display the row color in the form of a colored circle on the top left and the packet information as it is displayed in the Traffic View (see Traffic View for more information). Whenever a packet is encrypted, regardless of whether they were decrypted or not, an expansion button will appear on the top right. If the packet was decrypted the security keys used for decryption and the related addresses will be listed, if the packet could not be decrypted and the missing item is known, you will be provided with auto-complete text boxes to provide either the missing security key or address relationship. After providing the missing items, a Redecode button will appear. Click the button to decode the packet again, if the decryption if successful, you will be asked if the complete capture must be re-decoded or not.

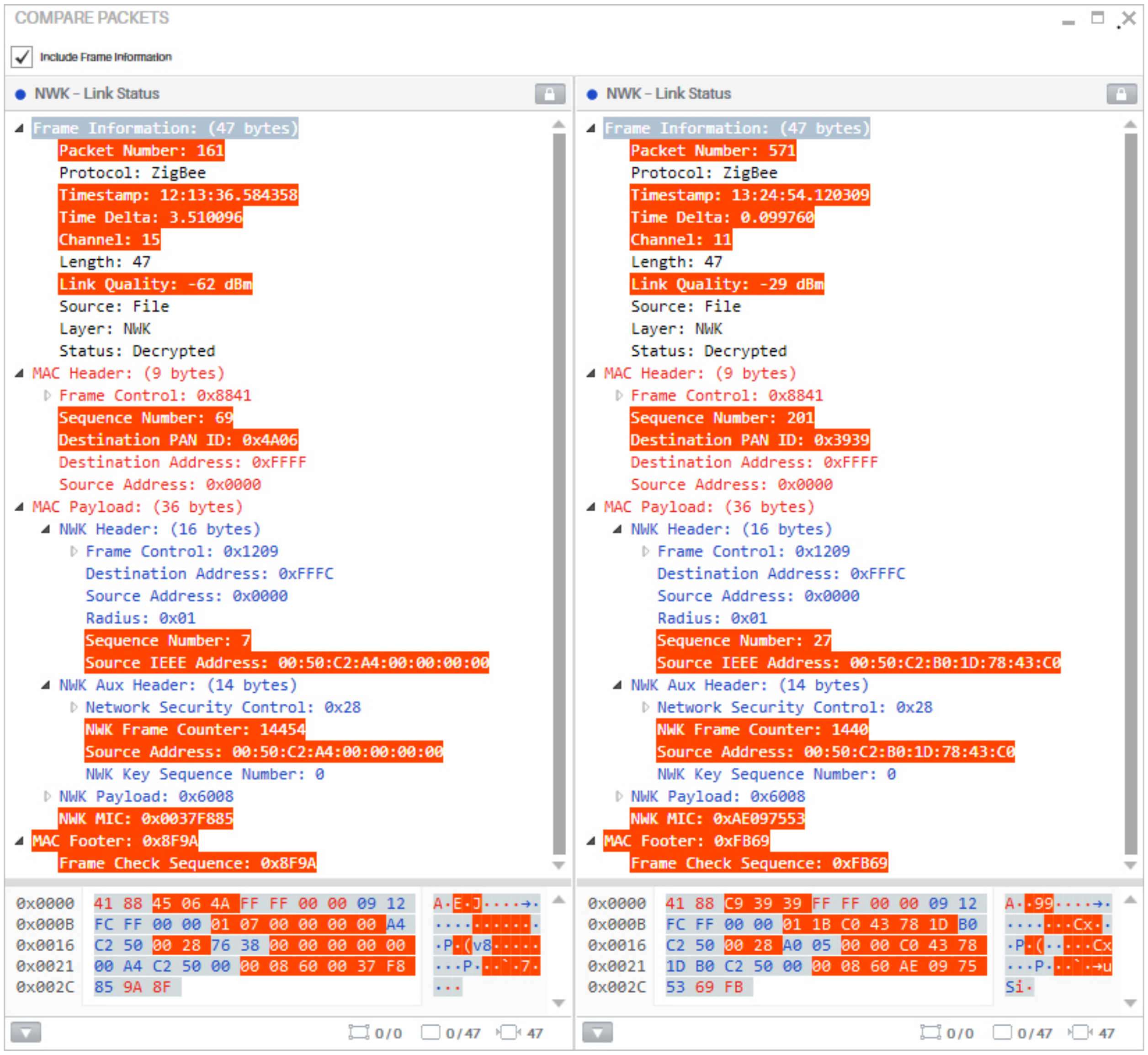
Creating Quick Filters

From this view you can create filters using the values of the displayed field values on the tree nodes. Just right click on a tree node and move your mouse cursor over the Create Filter menu item. Depending on the data type of the packet field you selected, different options will appear. Select one of the options and then the filter will be created and applied. Use the filter toolbar to edit or remove you newly created filter (see Filtering Packets for details).

Comparing Packets

The Packet View can also be used to compare packets and highlight the differences between them. To do so, click on the first packet on the Traffic View. Press and hold the Ctrl key on your keyboard. Click on the packet to compare to. Right click on one of the selected packets. Select the Compare Packets menu item. The Compare Packets window will appear highlighting the differences between both packets in red (see the figure below). The fields with different values will be already expanded on the tree representation although, only

OTA fields will be compared. Select the checkbox at the top of the window to also include the Frame Information (non OTA fields) in the comparison.



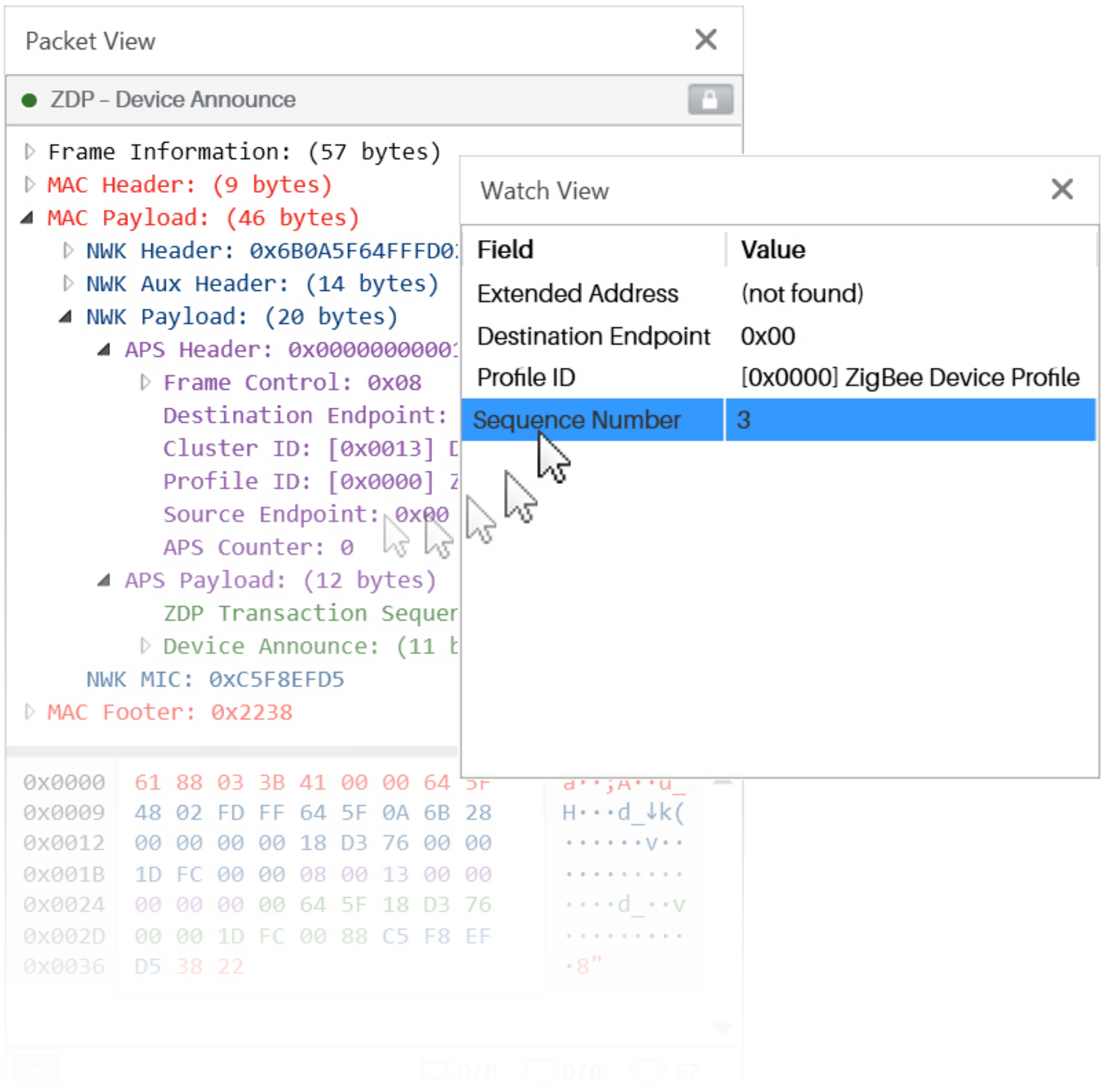
```
Frame Information: (47 bytes)
    Packet Number: 161
    Protocol: ZigBee
    Timestamp: 12:13:36.584358
    Time Delta: 3.510096
    Channel: 15
    Length: 47
    Link Quality: -62 dBm
    Source: File
    Layer: NWK
    Status: Decrypted
MAC Header: (9 bytes)
    Frame Control: 0x8841
    Sequence Number: 69
    Destination PAN ID: 0x4A06
    Destination Address: 0xFFFF
    Source Address: 0x0000
MAC Payload: (36 bytes)
    NWK Header: (16 bytes)
        Frame Control: 0x1209
        Destination Address: 0xFFFC
        Source Address: 0x0000
        Radius: 0x01
        Sequence Number: 7
        Source IEEE Address: 00:50:C2:A4:00:00:00:00
    NWK Aux Header: (14 bytes)
        Network Security Control: 0x28
        NWK Frame Counter: 14454
        Source Address: 00:50:C2:A4:00:00:00:00
        NWK Key Sequence Number: 0
    NWK Payload: 0x6008
    NWK MIC: 0x0037F885
MAC Footer: 0x8F9A
    Frame Check Sequence: 0x8F9A

0x0000  41 88 45 06 4A FF FF 00 00 09 12  A·E·J····→·
0x000B  FC FF 00 00 01 07 00 00 00 00 A4  ···········
0x0016  C2 50 00 28 76 38 00 00 00 00 00  ·P·(v8·····
0x0021  00 A4 C2 50 00 00 08 60 00 37 F8  ···P···`·7·
0x002C  85 9A 8F                          ···

0/0   0/47   47
```

```
Frame Information: (47 bytes)
    Packet Number: 571
    Protocol: ZigBee
    Timestamp: 13:24:54.120309
    Time Delta: 0.099760
    Channel: 11
    Length: 47
    Link Quality: -29 dBm
    Source: File
    Layer: NWK
    Status: Decrypted
MAC Header: (9 bytes)
    Frame Control: 0x8841
    Sequence Number: 201
    Destination PAN ID: 0x3939
    Destination Address: 0xFFFF
    Source Address: 0x0000
MAC Payload: (36 bytes)
    NWK Header: (16 bytes)
        Frame Control: 0x1209
        Destination Address: 0xFFFC
        Source Address: 0x0000
        Radius: 0x01
        Sequence Number: 27
        Source IEEE Address: 00:50:C2:B0:1D:78:43:C0
    NWK Aux Header: (14 bytes)
        Network Security Control: 0x28
        NWK Frame Counter: 1440
        Source Address: 00:50:C2:B0:1D:78:43:C0
        NWK Key Sequence Number: 0
    NWK Payload: 0x6008
    NWK MIC: 0xAE097553
MAC Footer: 0xFB69
    Frame Check Sequence: 0xFB69

0x0000  41 88 C9 39 39 FF FF 00 00 09 12  A··99····→·
0x000B  FC FF 00 00 01 1B C0 43 78 1D B0  ········Cx··
0x0016  C2 50 00 28 A0 05 00 00 C0 43 78  ·P·(·····Cx
0x0021  1D B0 C2 50 00 00 08 60 AE 09 75  ···P···`·→u
0x002C  53 69 FB                          Si·

0/0   0/47   47
```

Note that is recommended to disable the "Auto Scroll" and "Auto Select Packet" options before selecting the packets to be compared from the Traffic View.

Chapter 6: Watch View

This view has the objective to assist you tracking the value of a field shown in the Packet View tree. This is useful when capturing as it allows you to see how the value is changing as more packets arrive. To use this feature, just select the field from the Packet View and drag it to the Watch View (see the figure below), the value will be updated each time you select a different packet on the Traffic View. Combine this feature with the Auto Select option to see the value changing with incoming packets. To delete a field from the Watch View, select it and press the Delete key on your keyboard or right click on it and select the Delete menu item. The contents on the Watch View are stored on the Environment, so they will be the same each time you open Ubiqua (for more information see Setting Preferences).



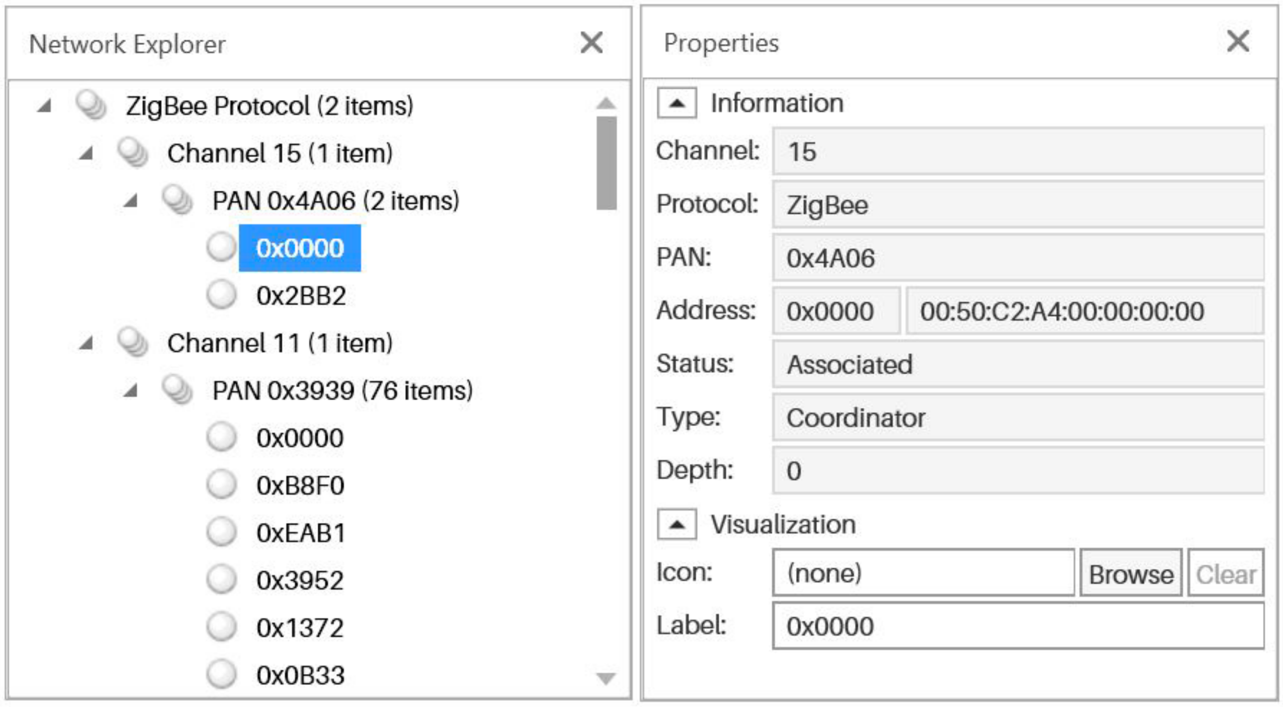
```
▷ Frame Information: (57 bytes)
▷ MAC Header: (9 bytes)
◢ MAC Payload: (46 bytes)
   ▷ NWK Header: 0x6B0A5F64FFFD0
   ▷ NWK Aux Header: (14 bytes)
   ◢ NWK Payload: (20 bytes)
      ◢ APS Header: 0x0000000000
         ▷ Frame Control: 0x08
           Destination Endpoint:
           Cluster ID: [0x0013] D
           Profile ID: [0x0000] Z
           Source Endpoint: 0x00
           APS Counter: 0
      ◢ APS Payload: (12 bytes)
           ZDP Transaction Sequer
         ▷ Device Announce: (11 b
     NWK MIC: 0xC5F8EFD5
▷ MAC Footer: 0x2238
```

Field	Value
Extended Address	(not found)
Destination Endpoint	0x00
Profile ID	[0x0000] ZigBee Device Profile
Sequence Number	3

```
0x0000  61 88 03 3B 41 00 00 64 5F   a··;A··d_
0x0009  48 02 FD FF 64 5F 0A 6B 28   H···d_↓k(
0x0012  00 00 00 00 18 D3 76 00 00   ······v··
0x001B  1D FC 00 00 08 00 13 00 00   ·········
0x0024  00 00 00 00 64 5F 18 D3 76   ····d_··v
0x002D  00 00 1D FC 00 88 C5 F8 EF   ·········
0x0036  D5 38 22                     ·8"
```

Chapter 7: Network Explorer

The Network Explorer view shows a tree representation of the network nodes found on the capture data (see the figure below). The hierarchy in the tree is organized by protocol, PAN ID, and channel. The Network Explorer is directly connected to the Graphic and Properties views. Select a node on the explorer and the same node will be selected on the Graphic View. Double click on a node and the Properties view will open showing the node information. If the Properties view is already open, just select a different node to see its properties.



- ZigBee Protocol (2 items)
  - Channel 15 (1 item)
    - PAN 0x4A06 (2 items)
      - 0x0000
      - 0x2BB2
  - Channel 11 (1 item)
    - PAN 0x3939 (76 items)
      - 0x0000
      - 0xB8F0
      - 0xEAB1
      - 0x3952
      - 0x1372
      - 0x0B33

Information

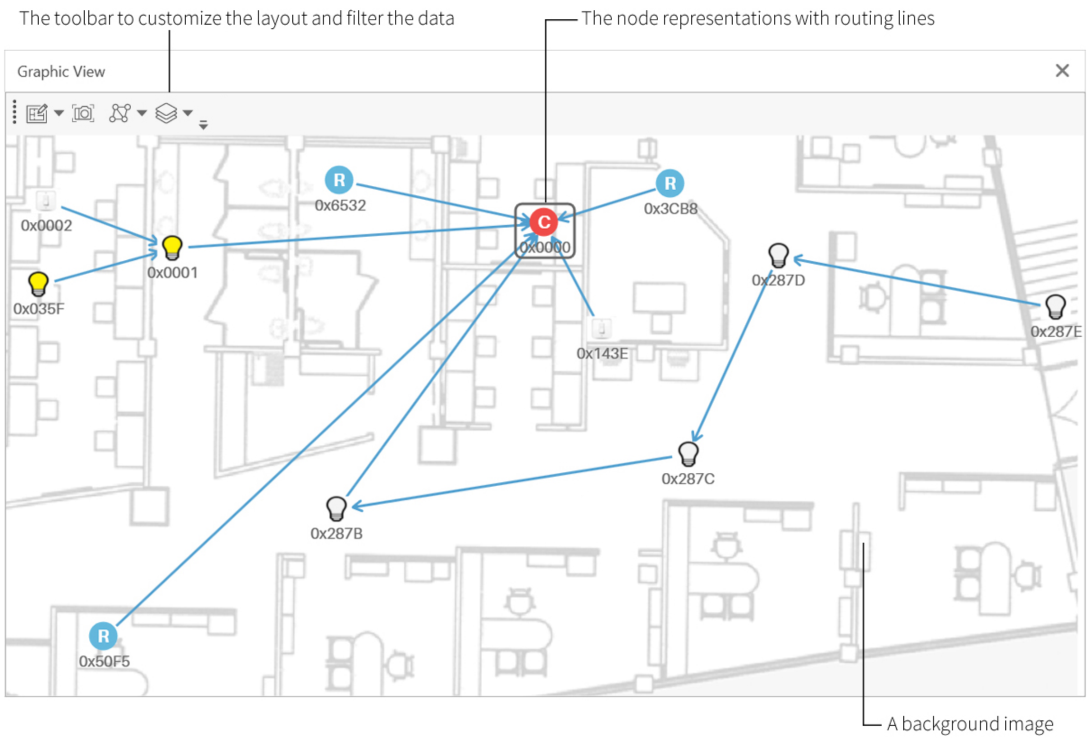
Channel:	15
Protocol:	ZigBee
PAN:	0x4A06
Address:	0x0000 00:50:C2:A4:00:00:00:00
Status:	Associated
Type:	Coordinator
Depth:	0

Visualization

Icon:	(none)
Label:	0x0000

Chapter 8: Graphic View

This view is a graphical representation of the network topology as interpreted by the current capture data. Each time a new node and address is detected on an incoming packet, such node is included into the representation. The Graphic View is composed of a canvas and a toolbar (see the figure below). The canvas contains a network graph showing the current network nodes. The toolbar offers access to options such as adding a background, customizing the network graph layout, selecting what data is shown, and controlling the zoom. This chapter details each of those options.



Changing The Background

To change the background of the canvas, click on the Load Background Image toolbar button (see the figure below). The Open dialog will appear. Select the new background

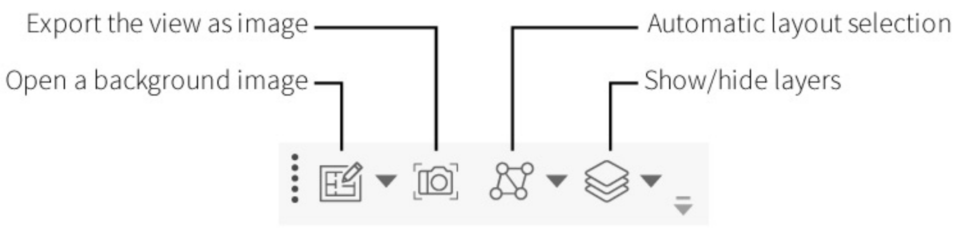
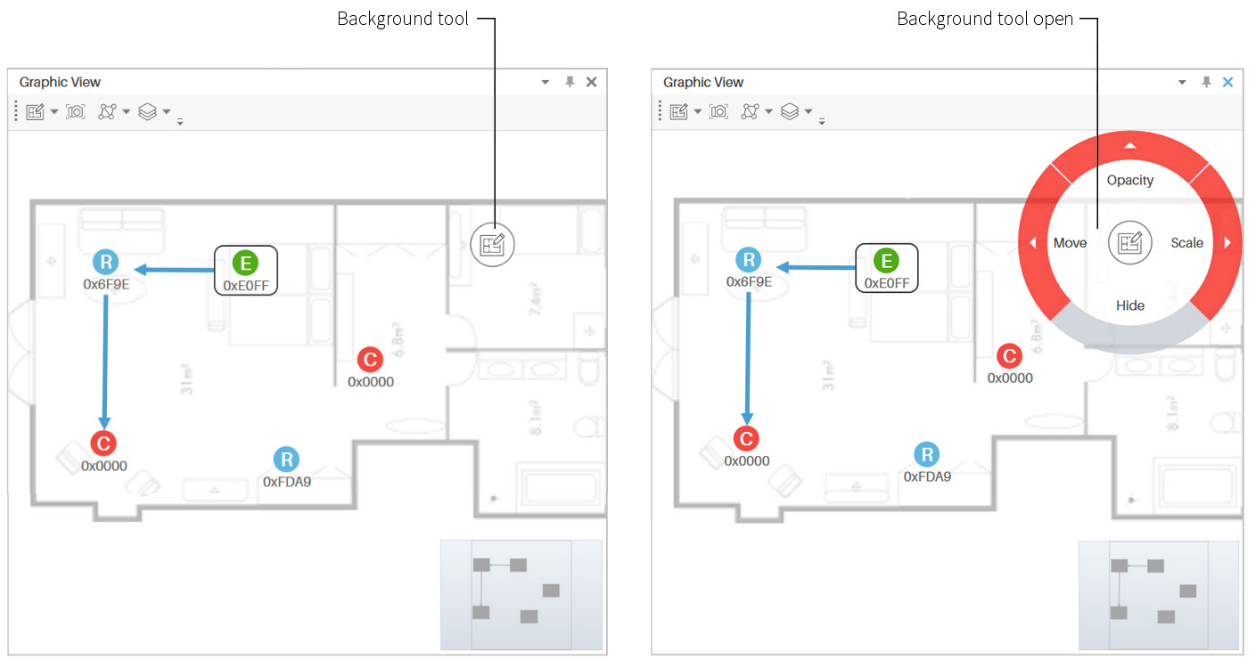


image and click the Open button. You can use images in the .bmp, .gif, .jpg, or .png file formats.

After adding a background to the canvas a Background Tool button appears in the top right section of the canvas area, this tool allows the user to move, zoom and change the transparency of the background image. This tool button can be hidden after use by clicking on the Hide button in the control or in the Show Tool menu item in the Background Tool button in the Graphic View Toolbar.

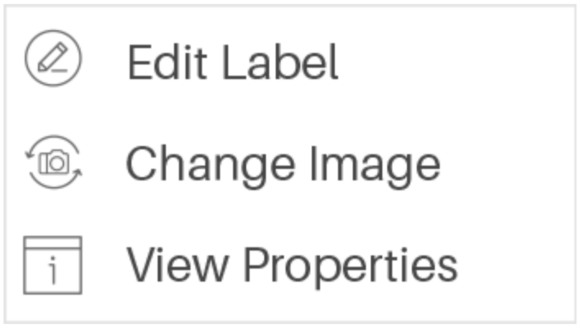


If the Background tool is not visible, you can show it again by clicking the Background button in the Toolbar and selecting the Show Tool Option.

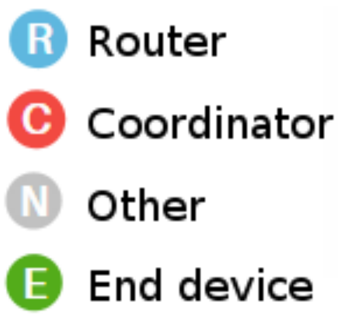
Customizing Nodes

You can change the position of the nodes by clicking and dragging each one within the Graphic View. The connecting arrows will move as the node does for persistence. Also, note that for this action is recommended that the Graphic and the Network Explorer Views are not hidden or auto-hidden.

The graph nodes can be customized by assigning them an alias and changing the icon used to represent them. Just right click on the node you want to customize (see figure below) and select the View Properties menu item. The Properties view will appear. Change the Alias and Icon properties and the Graphic View will be updated with your new choices.



The default node icon shows a colored circle to represent different node statuses. A blue icon represents a router node. A green one represents an end device. A red icon represents a coordinator node. And a gray one represents any other case.



Changing the Network Layers

Ubiqua shows two kind of relationships between network nodes: Topology and Routing. Each one of these are represented in the Graphic view as layers. The Topology layer for ZigBee captures is constructed by examining the information of the MAC 2003 and 2006 association request/response message and are represented by blue color edges. The Routing layer for ZigBee captures is constructed by examining the information contain in the ZigBee Network Link Status messages and are represented by green color edges.

Spanning and Zooming

To span the Graphic View, click on any free space on the canvas and drag the mouse cursor. For zooming use the Overview map control or use the mouse wheel button.

Exporting Images

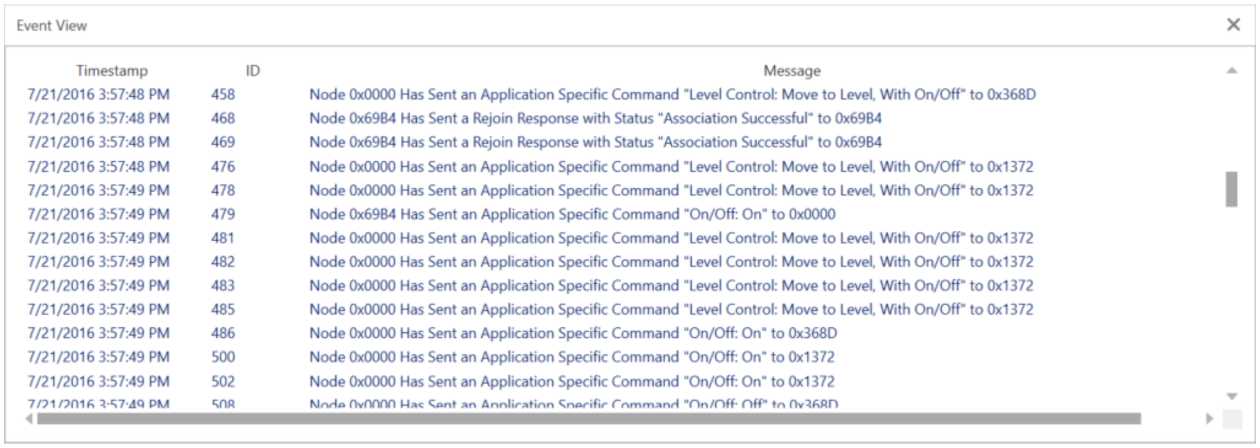
The Graphic View has a feature useful to share the visual representation of the network. You can export the contents of the view as an image. To do so, click on the Save Image toolbar button. The Save dialog will appear, select the location and the file format and click the Save button. You can save images in the .bmp, .gif, .jpg, .png, or .tif file formats.

Filtering Nodes

The Graphic View has the ability to hide and show nodes based on their PAN id or corresponding communication channel. To do this select the "Network Nodes" option from the "Layers" toolbar in the Graphic view, and select the way you want to Show/Hide the nodes, either by channel or Pan ID; next a list of the selected items will appear. To show an item just click to show a check mark, to hide it click the item for uncheck.

Chapter 9: Event View

The Event View highlights notable traffic events in a dedicated listing. By clicking on an entry in the Event View, the corresponding packet will be selected in the Packet View.



Event View

Timestamp	ID	Message
7/21/2016 3:57:48 PM	458	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x368D
7/21/2016 3:57:48 PM	468	Node 0x69B4 Has Sent a Rejoin Response with Status "Association Successful" to 0x69B4
7/21/2016 3:57:48 PM	469	Node 0x69B4 Has Sent a Rejoin Response with Status "Association Successful" to 0x69B4
7/21/2016 3:57:48 PM	476	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x1372
7/21/2016 3:57:49 PM	478	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x1372
7/21/2016 3:57:49 PM	479	Node 0x69B4 Has Sent an Application Specific Command "On/Off: On" to 0x0000
7/21/2016 3:57:49 PM	481	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x1372
7/21/2016 3:57:49 PM	482	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x1372
7/21/2016 3:57:49 PM	483	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x1372
7/21/2016 3:57:49 PM	485	Node 0x0000 Has Sent an Application Specific Command "Level Control: Move to Level, With On/Off" to 0x1372
7/21/2016 3:57:49 PM	486	Node 0x0000 Has Sent an Application Specific Command "On/Off: On" to 0x368D
7/21/2016 3:57:49 PM	500	Node 0x0000 Has Sent an Application Specific Command "On/Off: On" to 0x1372
7/21/2016 3:57:49 PM	502	Node 0x0000 Has Sent an Application Specific Command "On/Off: On" to 0x1372
7/21/2016 3:57:49 PM	508	Node 0x0000 Has Sent an Application Specific Command "On/Off: Off" to 0x368D

Supported ZigBee events

Node Join
Triggered when a new node joins the network.

Touchlink Node Join
Triggered when a node joins the network using the touchlink mechanism.

Node Leave
Triggered when a node leaves the network.

Node Rejoin
Triggered when a node rejoins the network.

Application Command
Generic log of an application-layer command.

APS Transport Key
Triggered when Ubiqua registers a new ZigBee key.

Supported Thread events

Child Attach
Triggered when a node joins the network.

Network Data
Triggered when Ubiqua receives a Type Length Value payload.

Child Becomes Router
Triggered when a network node assumes the Router role.

Chapter 10: Ubiqua Services

Ubiqua Protocol Analyzer provides a number of services exposing a limited set of the functionality available throughout the application as Web resources. The Remote Access Service is one of such services. In this chapter you will find the specification of its interface containing the provided resources, what HTTP methods are allowed, what the data representation formats are, and some examples of what you can expect as service outputs.

Remote Access Service

Enabling the Service

To enable the service, go to Tools > Options... in the main menu (see the figure below) and check the "Enable Remote Access Service" box; this box does not denote the service status, it only enables the service. A label showing the service status is next to the right of the "Listening port:" box. The default port is 19501, to specify another one use the "Listening port:" option. The service will start after you click on the OK button. If you keep the enable option checked, the service will start as soon as the application starts.

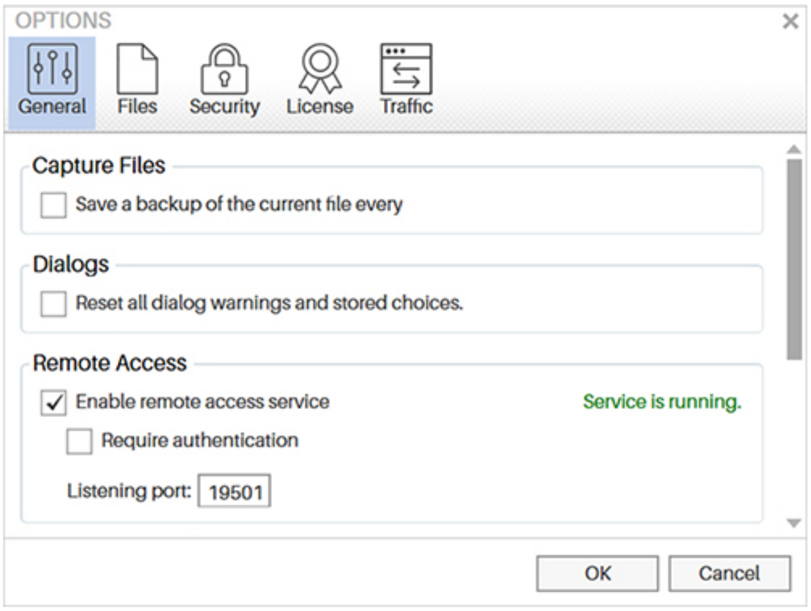
Service
As of Ubiqua 2.0, for better security and user protection, the Remote Services functionality requires an administrator to authorize the use of the network port assigned to Ubiqua. You can authorize the port by running the following command with a privileged account:

```
netsh http add urlacl url=http://*:19501/ user=Everyone listen=yes
```

If you are having problems connecting to the Remote Access Service, your Operating System may be blocking access to the configured port, refer to this link for more information.

If you want to secure the access, the service supports HTTP Basic Authentication. Just check the "Require authentication" box and provide a username and a password.

The service is still enabled if an error occurs and the status label will indicate the current status of the service.


OPTIONS

Capture Files

☐ Save a backup of the current file every

Dialogs

☐ Reset all dialog warnings and stored choices.

Remote Access

☑ Enable remote access service — Service is running.

☐ Require authentication

Listening port: 19501

Available Resources

The Remote Access Service provides 5 resources: `/capture`, `/sniffers`, `/filters`, `/keys` and `/addresses`. The `/capture` resource represents the current capture in Ubiqua, and it can be used to retrieve the current number of packets and a specific range of packets, to export the capture file to the local file system, or to clear all the packets. The `/sniffers` resource represents the sniffer devices currently attached and available in Ubiqua, it can be used to retrieve the list of all devices, the status of a specific device, and to start or stop a sniffer. The `/filters` resource represents the current filters in Ubiqua, and it can be used to retrieve the list of filters, to enable or disable a filter. The `/keys` resource represents the current security keys in Ubiqua, and it can be used to retrieve the list of security keys, and to insert a new security key. The `/addresses` resource represents the current addresses in Ubiqua, and it can be used to retrieve the list of addresses, and to insert a new relationship of long address and short address.

The schema document for this service responses is located at

https://www.ubilogix.com/resources/ubiqua/services.xsd

Current Capture

GET / `capture`

Returns information about the current capture.

Request Parameters

- **offset** – An integer accepting positive numbers and zero `0`. The offset of the packets to retrieve.

- **limit** – An integer accepting positive numbers and zero `0`. The number of packets to retrieve.

Response Code 200 (OK)

An XML document with the information about the current capture.

Sample Request

A GET request to the URL `http://localhost:19501/capture?offset=10&limit=2`, produces a response with code 200, and the following document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Capture xmlns="urn:ubilogix:services">
  <Packets Count="48454" Offset="10" Limit="2">
   <Packet Id="11">
   <Info>Beacon</Info>
   <Source>File</Source>
     <Status>Normal</Status>
   <Raw>008038AA1A0000FFCF8000002284111111111111111FFFFFF00FFFF</Raw>
  </Packet>
  <Packet Id="12">
   <Info>Beacon Request</Info>
   <Source>File</Source>
     <Status>Normal</Status>
   <Raw>03087DFFFFFFFF07FFFF</Raw>
  </Packet>
 </Packets>
</Capture>
```

PUT / `capture`

Performs the specified action on the current capture.

Request Parameters

- **action** – A string with one of the following values:
  - ○ **clear**, to reset the capture.
  - ○ **save**, to store the capture log in Ubiqua file format on the specified file location.
  - ○ **export**, to store the capture log in other file format on the specified file location. Use one of the valid file extensions: `.txt`, `.xls`, `.csv`, `.opml`, and `.cubx`.
  - ○ **load**, to open a file capture with any supported file format of Ubiqua.
- **filename** – A string with the file name to export the capture to or, the file name to open capture.

Response Code 200 (Created)

○ The capture was successfully exported.

○ The file is created with no packet information if there is no data to export.

Response Code 202 (Accepted)

○ The capture was successfully cleared or, it is being exported but the process is not yet completed.

Response Code 400 (Bad Request)

○ The file type of the file specified in the `filename` parameter is not supported by Ubiqua.

Response Code 500 (Internal Server Error)

○ The export process failed.

Sample Requests

Save a capture by sending a PUT request to the URL `http://localhost:19501/capture`, with the following request body: `action=save&filename=C:\test.subx` and a content-type request header containing `application/x-www-form-urlencoded`, produces a response with code 200 and an empty body.

Export a capture by sending a PUT request to the URL `http://localhost:19501/capture`, with the following request body: `action=export&filename=C:\test.txt` and a content-type request header containing `application/x-www-form-urlencoded`, produces a response with code 200 and an empty body.

Sniffer Devices

GET / `sniffers`

Returns a list of the sniffer devices connected to Ubiqua.

Response Code 200 (OK)

An XML document with the list of sniffer devices.

Sample Request

A GET request to the URL `http://localhost:19501/sniffers`, produces a response with code 200, and the following document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Devices xmlns="urn:ubilogix:services">
  <Sniffers>
    <Sniffer Id="USB5255" Link="http://10.0.1.9:19501/sniffers/USB5255">
      <IsStarted>false</IsStarted>
      <IsPlugged>true</IsPlugged>
      <Channel>25</Channel>
      <Protocol>6</Protocol>
    </Sniffer>
    <Sniffer Id="COM35" Link="http://10.0.1.9:19501/sniffers/COM35">
      <IsStarted>false</IsStarted>
      <IsPlugged>true</IsPlugged>
      <Channel>25</Channel>
      <Protocol>11</Protocol>
    </Sniffer>
    <Sniffer Id="01112004" Link="http://10.0.1.9:19501/sniffers/01112004">
      <IsStarted>false</IsStarted>
      <IsPlugged>false</IsPlugged>
      <Channel>11</Channel>
      <Protocol>0</Protocol>
    </Sniffer>
  </Sniffers>
</Devices>
```

GET / `sniffers/{id}`

Returns the information of the sniffer device with the specified ID.

Response Code 200 (OK)

Gets an XML document with the status of the sniffer device with the specified ID.

Response Code 404 (Not Found)

The sniffer with the specified ID was not found on the list of available sniffer devices.

Sample Request

A GET request to the URL `http://127.0.0.1:19501/sniffers/USB5255`, produces a response with code 200, and the following document:

```
<?xml version="1.0" encoding="UTF-8"?>
<Sniffer        Id="Texas        Instruments        CC2531        [USB5255]"
xmlns="urn:ubilogix:services">
  <IsStarted>false</IsStarted>
  <IsPlugged>true</IsPlugged>
  <Channel>25</Channel>
  <Protocol>0</Protocol>
</Sniffer>
```

PUT /`sniffers/{id}`

Starts or stops the sniffer device of the specified ID.

Request Parameters

- **action** – A string with one of the following values:
  - **start**, to start the sniffer.
  - **stop**, to stop the device.
- **channel** – A value between 11 and 26 representing the physical channel (optional).
- **protocol** – A string with one of the following supported values (optional): **IEEE 802.15.4-2003, IEEE 802.15.4-2006, ZigBee, IP, Synkro RF, Raw Data, PopNet, ZigBee RF4CE, ZigBee IP, JenNet IP, IETF 6LoWPAN.**

Response Code 200 (OK)

Starts or stops a sniffer device as instructed by the `action` parameter.

Response Code 400 (Bad Request)

The sniffer can not start: channel out of range or protocol not supported.

Response Code 404 (Not Found)

The sniffer with the specified ID was not found on the list of available sniffer devices.

Response Code 503 (Service Unavailable)

The sniffer with the specified ID is already in the requested state.

Sample Request

A PUT request to the URL `http://localhost:19501/sniffers/USB5255`, with the following request body: `action=start`, `channel=16`, and `modulation=OQPSK_250` of content-type `application/x-www-form-urlencoded`, produces a response with code 200 and an empty body. The device was started.

Filters

GET /`filters`

Returns a list of the filters in Ubiqua.

Response Code 200 (OK)

An XML document with the list of filters.

Sample Request

A GET request to the URL `http://localhost:19501/filters`, produces a response with code 200, and the following document:

```
<?xml version="1.0" encoding="UTF-8"?>
<Filters xmlns="urn:ubilogix:services">
    <Filter Name="Time Delta is 0.000000" IsEnabled="true" Link="http://
10.0.1.8:19501/filters/0" />
   <Filter Name="channel 21" IsEnabled="false" Link="http://10.0.1.8:19501/
filters/1" />
</Filters>
```

GET /`filters/{id}`

Returns the status and conditions of the filter with the specified ID.

Response Code 200 (OK)

Gets an XML document with the status and conditions of the filter with the specified ID.

Response Code 404 (Not Found)

The filter with the specified ID was not found on the list of filters.

Sample Request

A GET request to the URL `http://127.0.0.1:19501/filters/2`, produces a response with code 200, and the following document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Filter>
  <Name>Sequence Number is 10</Name>
  <IsEnabled>true</IsEnabled>
  <Conditions Match="All">
    <Condition>
      <TargetName>SequenceNumber</TargetName>
      <TargetDisplayName>Sequence Number</TargetDisplayName>
      <Operator>Is</Operator>
      <TargetType>Integer</TargetType>
      <TargetLength>8</TargetLength>
      <TargetValue>10</TargetValue>
      <TargetAdditionalValue />
    </Condition>
  </Conditions>
</Filter>
```

PUT / `filters`

Enables or disables the filter.

Request Parameters

- **action** – A string with one of the following values:
  - **enable**, to enable the filter.
  - **disable**, to disable the filter.
- **filter** – A string with the filter name.

Response Code 200 (OK)

Enables or disables the filter as instructed by the exttt{action} parameter.

Response Code 400 (Bad Request)

The filter does not exist, or the action is unknown.

Response Code 503 (Service Unavailable)

The filter cannot be enabled or disabled when a task is running.

Sample Request

A PUT request to the URL `http://localhost:19501/filters`, with the following request body: `action=enable&filter=Frame%20Type%20is%20Command` of content-type `application/x-www-form-urlencoded`, produces a response with code 200 and an empty body. The filter was enabled.

Security Keys

GET / `keys`

Returns a list of the Security keys in Ubiqua.

Response Code 200 (OK)

An XML document with the list of security keys.

Sample Request

A GET request to the URL `http://localhost:19501/keys,` produces a response with code 200, and the following document:

```
<?xml version="1.0" encoding="UTF-8"?>
<Keys xmlns="urn:ubilogix:services">
 <Key Type="NetworkKey">76:D9:74:54:AE:A1:C6:1B:24:A1:90:96:0E:D1:D2:39</Key>
 <Key Type="LinkKey">A8:53:6D:BC:99:11:5B:E1:31:B5:F4:FD:A5:35:60:42</Key>
 <Key Type="LinkKey">E2:D6:45:F2:8C:A8:B0:E8:EF:7E:CC:E0:4C:BF:8D:BE</Key>
</Keys>
```

POST / `keys`

Inserts a new security key in Ubiqua.

Request Parameters

- **type** – A string with one of the following supported values: **PopNetKey, RF4CEKey, NetworkKey, LinkKey, Trust-CenterMasterKey, IEEE-802.15.4-2006Key.**
- **key** – A string with the new security key to add. The security key length must be 128 bits.

Response Code 202 (Accepted)

New security key was added with the specified key type.

Response Code 400 (Bad Request)

The key type is invalid, or the key length is invalid.

Sample Request

A POST request to the URL `http://localhost:19501/keys`, with the following request body: `type=NetworkKey&key=64:B9:99:95:4D:B1:F9:0F:20:F6:80:02:FB:FE:6F:C4` of content-type `application/x-www-form-urlencoded`, produces a response with code 202 and an empty body. The security key was added.

Network Addresses

GET / `addresses`

Returns a list of the addresses in Ubiqua.

Response Code 200 (OK)

An XML document with the list of addresses.

Sample Request

A GET request to the URL `http://localhost:19501/addresses`, produces a response with code 200, and the following document:

```
<?xml version="1.0" encoding="UTF-8"?>
<Addresses>
 <AddressRelation>
  <LongAddress>0000000000000000</LongAddress>
  <ShortAddressList>
   <ShortAddress>0000</ShortAddress>
   <ShortAddress>7C89</ShortAddress>
   <ShortAddress>ECAA</ShortAddress>
  </ShortAddressList>
 </AddressRelation>
 <AddressRelation>
  <LongAddress>001DB70000033D6D</LongAddress>
  <ShortAddressList>
   <ShortAddress>F02D</ShortAddress>
  </ShortAddressList>
 </AddressRelation>
</Addresses>
```

POST / `addresses`

Inserts a new relationship of long address and short address.

Request Parameters

- **longAddress** – The long address to add, with a length of 8 bytes.
- **shortAddress** – The short address to relate with the specified long address, with a length of 2 bytes.

Response Code 202 (Accepted)

New addresses relationship was added.

Response Code 400 (Bad Request)

Either the longAddress or shortAddress value is invalid.

Sample Request

A POST request to the URL `http://localhost:19501/addresses`, with the following request body: `longAddress=001DB70000033D6D&shortAddress=40EA` of content-type `application/x-www-form-urlencoded`, produces a response with code 202 and an empty body. The addresses relationship was added.

Using The Command Line

If you prefer to interact with Ubiqua services from the command line you can use cURL, a free command line tool to send HTTP requests. Download cURL from http://curl.haxx.se/. This section shows some examples on how to use cURL with Ubiqua.

Getting the Sniffers List

```
C:\>curl -v "http://localhost:19501/sniffers"
* About to connect() to localhost port 19501 (#0)
*   Trying 127.0.0.1... connected
* Connected to localhost (127.0.0.1) port 19501 (#0)
> GET /sniffers HTTP/1.1
> User-Agent: curl/7.21.6 (i386-pc-win32) libcurl/7.21.6 OpenSSL/0.9.8r zlib/
1.2.5
> Host: localhost:19501
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Length: 341
< Content-Type: application/xml
< Server: Ubiqua-RemoteControl/1.3.2183 Microsoft-HTTPAPI/2.0
< Date: Wed, 25 Sep 2013 00:18:04 GMT
<
<?xml version="1.0" encoding="UTF-8"?>
<Devices xmlns="urn:ubilogix:services">
  <Sniffers>
    <Sniffer Id="01TP2RLY" Link="http://10.0.1.8:19501/sniffers/01TP2RLY">
      <IsStarted>false</IsStarted>
      <IsPlugged>true</IsPlugged>
      <Channel>25</Channel>
      <Protocol>0</Protocol>
    </Sniffer>
  </Sniffers>
</Devices>
* Connection #0 to host localhost left intact
* Closing connection #0
```

Starting a Sniffer

```
C:\>curl  -v  -X  PUT  -d  "action=start"  "http://localhost:19501/sniffers/
01TP2RLY"
* About to connect() to localhost port 19501 (#0)
*   Trying 127.0.0.1... connected
* Connected to localhost (127.0.0.1) port 19501 (#0)
> PUT /sniffers/01TP2RLY HTTP/1.1
> User-Agent: curl/7.21.6 (i386-pc-win32) libcurl/7.21.6 OpenSSL/0.9.8r zlib/
1.2.5
> Host: localhost:19501
> Accept: */*
> Content-Length: 12
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 200 OK
< Content-Length: 0
< Server: Ubiqua-RemoteControl/1.3.2183 Microsoft-HTTPAPI/2.0
< Date: Wed, 25 Sep 2013 00:12:34 GMT
<
* Connection #0 to host localhost left intact
* Closing connection #0
```

Saving the Capture Log

```
C:\>curl -v -X PUT -d "action=save&filename=c:\data.pcap" http://localhost:
19501/capture
* About to connect() to localhost port 19501 (#0)
*   Trying 127.0.0.1... connected
* Connected to localhost (127.0.0.1) port 19501 (#0)
> PUT /capture HTTP/1.1
> User-Agent: curl/7.21.6 (i386-pc-win32) libcurl/7.21.6 OpenSSL/0.9.8r zlib/
1.2.5
> Host: localhost:19501
> Accept: */*
> Content-Length: 35
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 200 OK
< Content-Length: 0
< Server: Ubiqua-RemoteControl/1.3 Microsoft-HTTPAPI/2.0
< Date: Wed, 25 Apr 2012 01:20:03 GMT
<
* Connection #0 to host localhost left intact
* Closing connection #0
```

Sample Source Code

The code snippets listed in this chapter are written in the C# programming language and are intended to be used with the .NET Framework.

Getting the Sniffers List

```
using System;
using System.IO;
using System.Net;
using System.Text;

// Create the web request
HttpWebRequest request = WebRequest.Create(
    "http://localhost:19501/sniffers") as HttpWebRequest;

// Get response
using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
{
    // Get the response stream
    StreamReader reader = new StreamReader(response.GetResponseStream());

    // Console application output
    Console.WriteLine(reader.ReadToEnd());
}
```

Starting a Sniffer

```
using System.Web;

Uri address = new Uri("http://localhost:19501/sniffers/USB5255");

// Create the web request
HttpWebRequest request = WebRequest.Create(address) as HttpWebRequest;

// Set type to PUT
request.Method = "PUT";
request.ContentType = "application/x-www-form-urlencoded";

// Create the data we want to send
string action = "start";

StringBuilder data = new StringBuilder();
data.Append("action=" + HttpUtility.UrlEncode(action));

// Create a byte array of the data we want to send
byte[] byteData = UTF8Encoding.UTF8.GetBytes(data.ToString());

// Set the content length in the request headers
request.ContentLength = byteData.Length;
```

```
// Write data
using (Stream putStream = request.GetRequestStream())
{
    putStream.Write(byteData, 0, byteData.Length);
}

// Get response
using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
{
    // Get the response stream
    StreamReader reader = new StreamReader(response.GetResponseStream());

    // Console application output
        Console.WriteLine(reader.ReadToEnd());
}
```

Getting the Capture Status With Authentication

```
// Create the web request
HttpWebRequest request
    = WebRequest.Create("http://localhost:19501/capture") as HttpWebRequest;

// Add authentication to request
request.Credentials = new NetworkCredential("test", "test");

// Get response
using (HttpWebResponse response = request.GetResponse() as HttpWebResponse)
{
    // Get the response stream
    StreamReader reader = new StreamReader(response.GetResponseStream());

    // Console application output
    Console.WriteLine(reader.ReadToEnd());
}
```

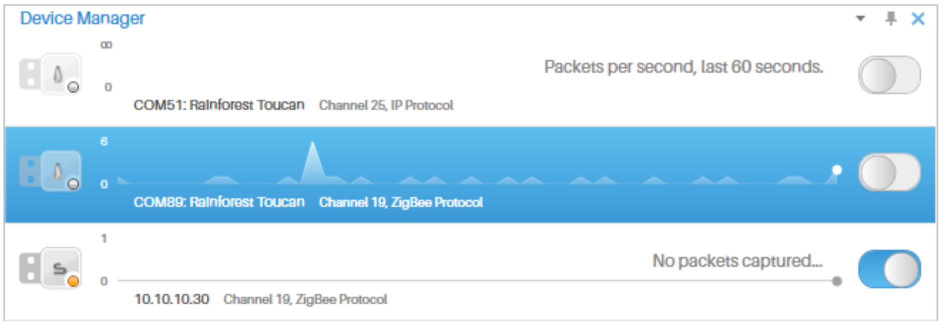
Running Ubiqua As Server

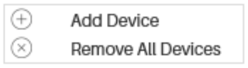
To run Ubiqua as a server just add the `/server` argument to your Ubiqua shortcut. Or run Ubiqua from the command line as `Ubiqua.exe  /server`. Alternatively, if Ubiqua is already running, select the Window > Minimize To Tray menu item.

Chapter 11: Sewio Hardware

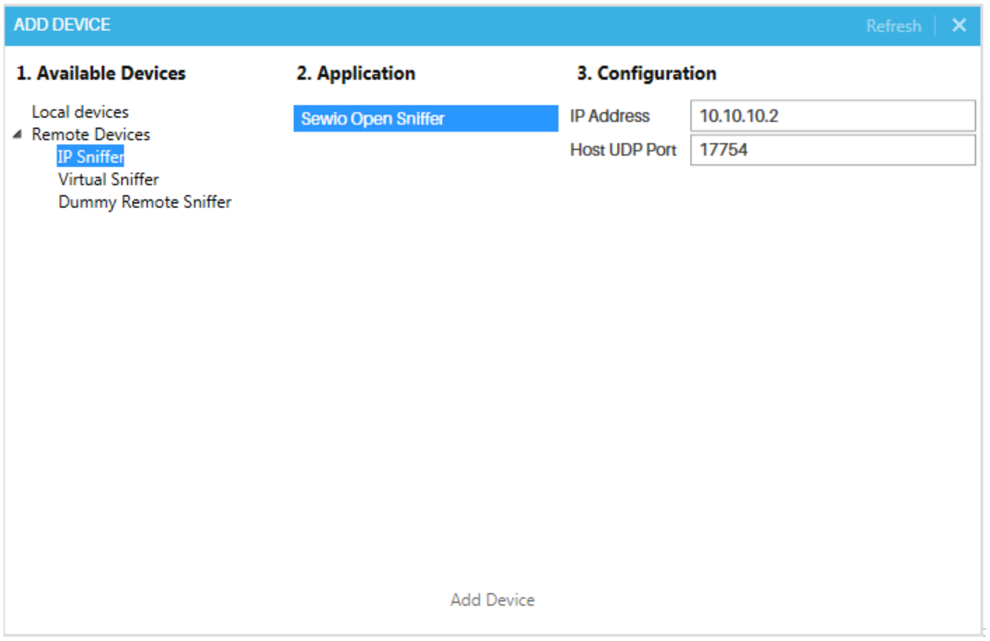
In order to use the Sewio Open Sniffer in Ubiqua, it must be connected and configured in the same subnetwork as the computer running Ubiqua. Please refer to "Part 1" of the sniffer installation http://www.sewio.net/open-sniffer/sniffer-installation/.

The process of configuring the sniffer in Ubiqua is simple. First, we add the device by right clicking on the Device Manager View or via the Device main menu.





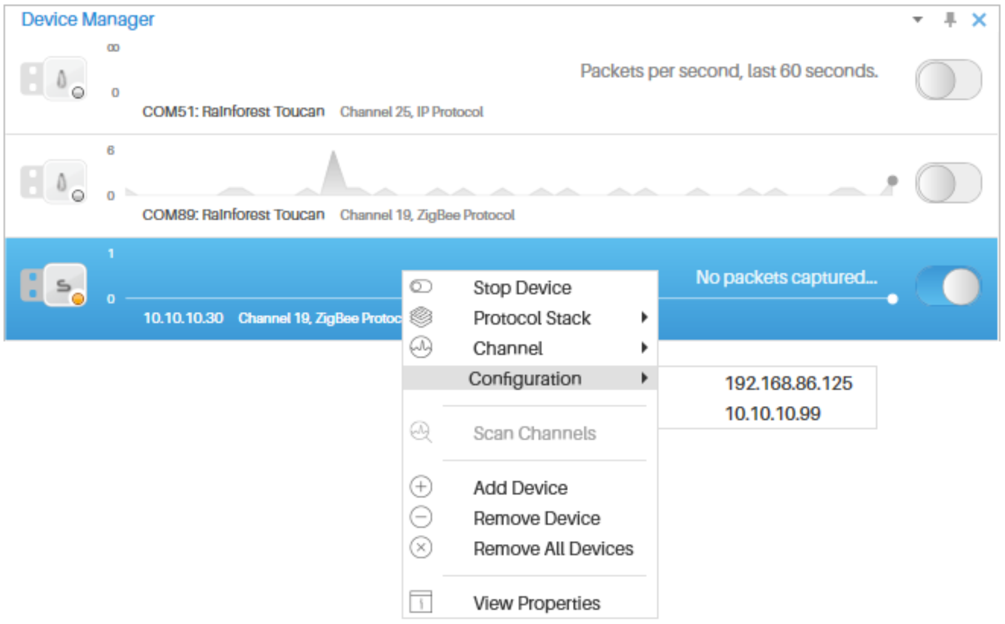
In the Add Device window select Sewio as the Vendor.



IP Address 10.10.10.2
Host UDP Port 17754

Then type the configuration parameters (IP Address and Port number) of the Sewio device, set the IP Address of the Sniffer and the UDP port that you want Ubiqua to listen for the packets. If you're configuring multiple sniffers, it is recommended to use a different UDP port for each sniffer.

At this point the Sewio device is added to Ubiqua. Before starting the device it's important to set the host, channel and protocol stack of the capture data using the context menu that appears by right clicking on top of the device. Usually the device is configured in the "10.10.10.XXX" IP address range. Lastly, select Start Device to begin capturing packets.



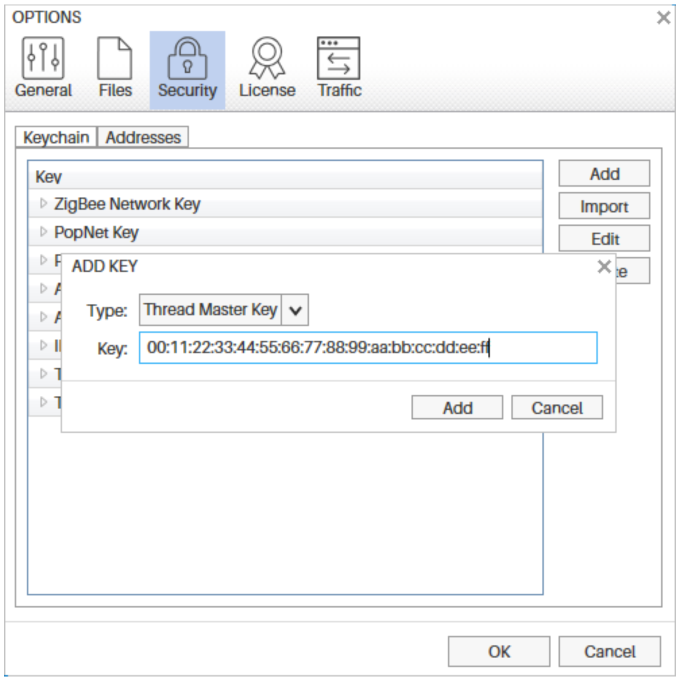
Note that if you change the device configuration from the Sewio web interface, the sniffer may stop sending packages to Ubiqua, just stop and restart the device from the Ubiqua Device Manager to correct the situation.

Chapter 12: Thread Support

Ubiqua decodes all protocols used by Thread such as IEEE 802.15.4-2006, 6LowPAN, UDP, DTLS, CoAP, DHCPv6 and MLE. To easily distinguish between protocols, each of them has a different color (see Traffic View).

Ubiqua can decrypt Thread packets that use IEEE 802.15.4 2006 security (MAC Layer) and MLE encryption.

There are three types of keys used by Ubiqua, Thread Master Key, MLE key and MAC key to decode Thread packets. With the keychain used to storage all keys, a Thread Master Key can be used to derive a MLE or MAC key, and if the decrypting process is successful the MLE and MAC keys will also be stored on the keychain.

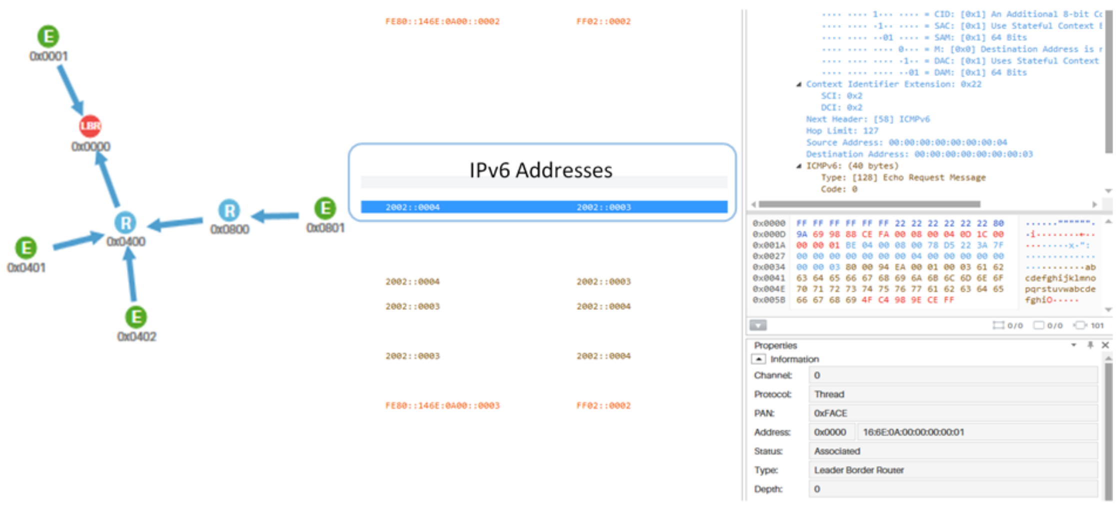


Type: Thread Master Key

Key: 00:11:22:33:44:55:66:77:88:99:aa:bb:cc:dd:ee:ff

Network Data

Thread uses a lot of network data information to manage the generation of the ipv6 addresses and to identify the devices into the networks that work as border routers, commissioners or collapsed devices. Ubiqua collects this network data to generate IPv6 networks addresses (stateful using 6lowpan context id and stateless), etc. this information is displayed in the Graphic View.



FE80::146E:0A00::0002	FF02::0002
2002::0004	2002::0003
2002::0004	2002::0003
2002::0003	2002::0004
2002::0003	2002::0004
FE80::146E:0A00::0003	FF02::0002

```
          .... .... 1... .... = CID: [0x1] An Additional 8-bit Cc
          .... .... .1.. .... = SAC: [0x1] Use Stateful Context E
          .... .... ..01 .... = SAM: [0x1] 64 Bits
          .... .... .... 0... = M: [0x0] Destination Address is r
          .... .... .... .1.. = DAC: [0x1] Uses Stateful Context
          .... .... .... ..01 = DAM: [0x1] 64 Bits
      ◢ Context Identifier Extension: 0x22
          SCI: 0x2
          DCI: 0x2
        Next Header: [58] ICMPv6
        Hop Limit: 127
        Source Address: 00:00:00:00:00:00:00:04
        Destination Address: 00:00:00:00:00:00:00:03
      ◢ ICMPv6: (40 bytes)
          Type: [128] Echo Request Message
          Code: 0
```

```
0x0000  FF FF FF FF FF FF 22 22 22 22 22 22 80   ......"""""".
0x000D  9A 69 98 88 CE FA 00 08 00 04 0D 1C 00   .i........+..
0x001A  00 00 01 BE 04 00 08 00 78 D5 22 3A 7F   .........x.":
0x0027  00 00 00 00 00 00 00 04 00 00 00 00 00   .............
0x0034  00 00 03 80 00 94 EA 00 01 00 03 61 62   ...........ab
0x0041  63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F   cdefghijklmno
0x004E  70 71 72 73 74 75 76 77 61 62 63 64 65   pqrstuvwabcde
0x005B  66 67 68 69 4F C4 98 9E CE FF            fghiO.....
```

0/0   0/0   101

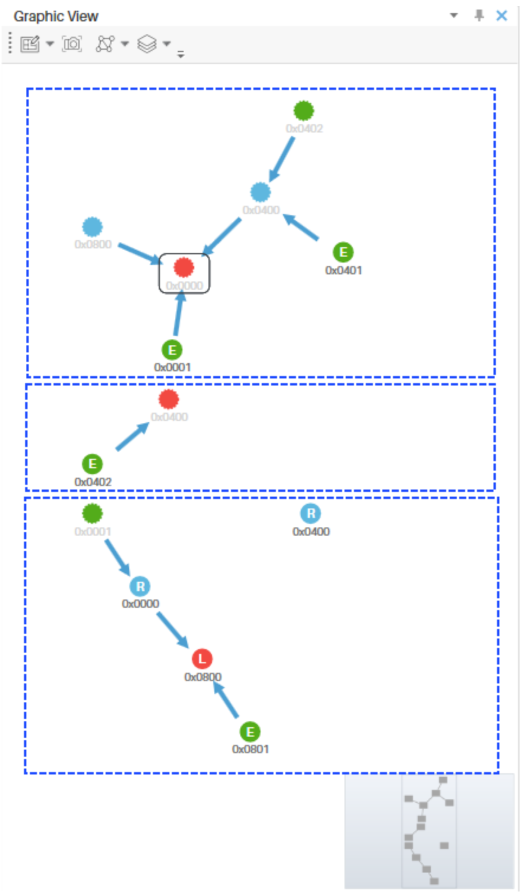
Properties

Information

Channel:	0	
Protocol:	Thread	
PAN:	0xFACE	
Address:	0x0000	16:6E:0A:00:00:00:00:01
Status:	Associated	
Type:	Leader Border Router	
Depth:	0	

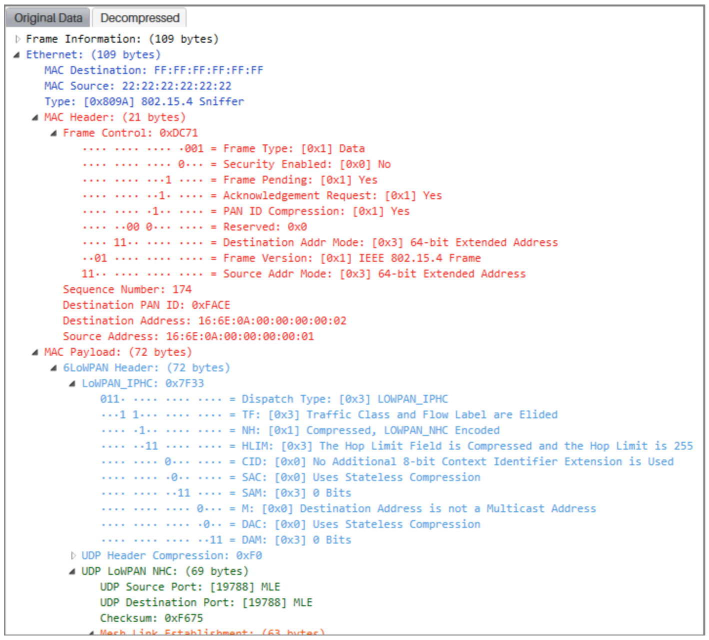
Graphic View

Ubiqua is capable to show the topology by partition ID used in Thread. If a capture has several partitions into the same PAN ID and channel, Ubiqua arranges the partitions by order of creation. The image below shows a topology formed by 3 different partitions.

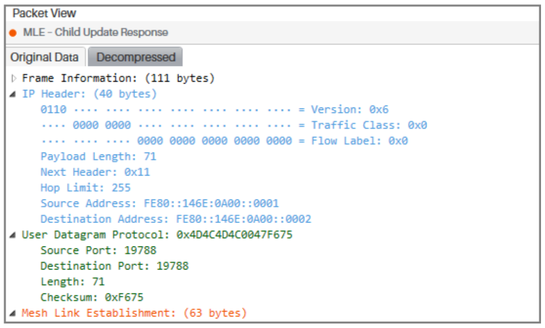


Packet View

Ubiqua's Thread support adds a tab in the Packet View called "Decompressed". This tab shows the decompressed IP Header (from 6lowpan layer).



```
Frame Information: (109 bytes)
Ethernet: (109 bytes)
   MAC Destination: FF:FF:FF:FF:FF:FF
   MAC Source: 22:22:22:22:22:22
   Type: [0x809A] 802.15.4 Sniffer
  MAC Header: (21 bytes)
    Frame Control: 0xDC71
       .... .... .... .001 = Frame Type: [0x1] Data
       .... .... .... 0... = Security Enabled: [0x0] No
       .... .... ...1 .... = Frame Pending: [0x1] Yes
       .... .... ..1. .... = Acknowledgement Request: [0x1] Yes
       .... .... .1.. .... = PAN ID Compression: [0x1] Yes
       .... ..00 0... .... = Reserved: 0x0
       .... 11.. .... .... = Destination Addr Mode: [0x3] 64-bit Extended Address
       ..01 .... .... .... = Frame Version: [0x1] IEEE 802.15.4 Frame
       11.. .... .... .... = Source Addr Mode: [0x3] 64-bit Extended Address
    Sequence Number: 174
    Destination PAN ID: 0xFACE
    Destination Address: 16:6E:0A:00:00:00:00:02
    Source Address: 16:6E:0A:00:00:00:00:01
  MAC Payload: (72 bytes)
    6LoWPAN Header: (72 bytes)
      LoWPAN_IPHC: 0x7F33
         011. .... .... .... = Dispatch Type: [0x3] LOWPAN_IPHC
         ...1 1... .... .... = TF: [0x3] Traffic Class and Flow Label are Elided
         .... .1.. .... .... = NH: [0x1] Compressed, LOWPAN_NHC Encoded
         .... ..11 .... .... = HLIM: [0x3] The Hop Limit Field is Compressed and the Hop Limit is 255
         .... .... 0... .... = CID: [0x0] No Additional 8-bit Context Identifier Extension is Used
         .... .... .0.. .... = SAC: [0x0] Uses Stateless Compression
         .... .... ..11 .... = SAM: [0x3] 0 Bits
         .... .... .... 0... = M: [0x0] Destination Address is not a Multicast Address
         .... .... .... .0.. = DAC: [0x0] Uses Stateless Compression
         .... .... .... ..11 = DAM: [0x3] 0 Bits
      UDP Header Compression: 0xF0
      UDP LoWPAN NHC: (69 bytes)
         UDP Source Port: [19788] MLE
         UDP Destination Port: [19788] MLE
         Checksum: 0xF675
        Mesh Link Establishment: (63 bytes)
```

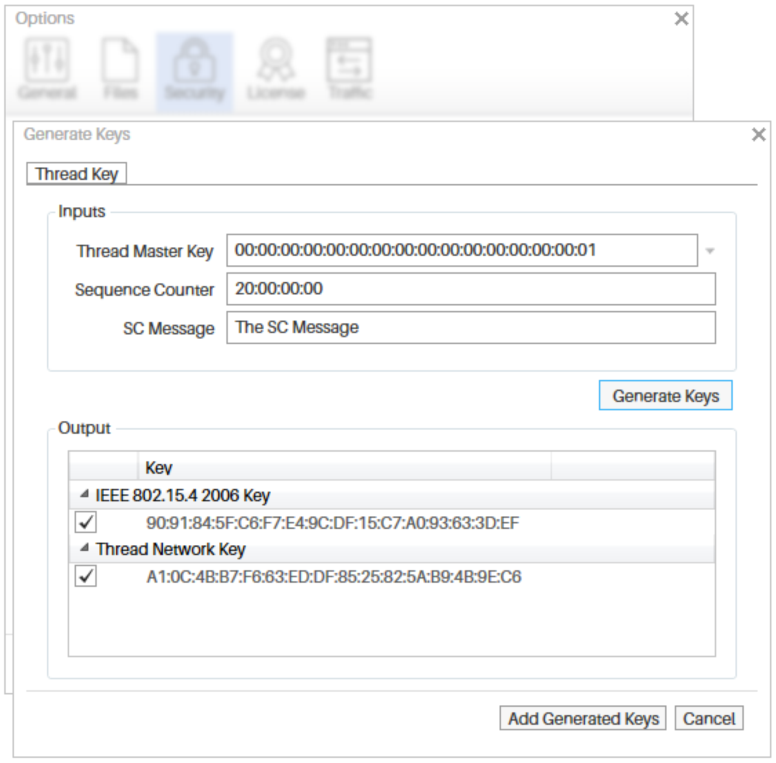


```
Frame Information: (111 bytes)
IP Header: (40 bytes)
   0110 .... .... .... .... .... .... .... = Version: 0x6
   .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x0
   .... .... .... 0000 0000 0000 0000 0000 = Flow Label: 0x0
   Payload Length: 71
   Next Header: 0x11
   Hop Limit: 255
   Source Address: FE80::146E:0A00::0001
   Destination Address: FE80::146E:0A00::0002
User Datagram Protocol: 0x4D4C4D4C0047F675
   Source Port: 19788
   Destination Port: 19788
   Length: 71
   Checksum: 0xF675
Mesh Link Establishment: (63 bytes)
```

In Thread a specific node's behavior is very dynamic and the data of several packets needs to be analyzed to show the topology, addressing, security and behavior context. Ubiqua's Thread Packet View implements more than just a simple decoder by also showing information derived from previous packets to help engineers to easily analyze the capture.

Generate Key

Some times you start a capture after the encryption Keys have been exchanged, and although Ubiqua performs a fair job in trying to get them for you, some times the traffic does no allow for it. In these cases you can make use of the "Generate" feature as follows: First click theTools-> Options Menu, to show then Options window , select the "Security" Icon and then the "Keychain" tab. From the right side of the Keychain tab a column with 5 buttons will appear, click the "Generate" button to show the "Generate Keys" window. Input the known values for the "Thread Master Key", "Sequence Counter" and "SC Message" fields and finally click the "Generate Keys" button. You will see the 2 new keys in the output section ("IEEE 802.15.4 2006 Key" and the "Thread Network Key"). To save the Keys in the Keychain click the "Add Generated Keys" button or click "Cancel" to close the Generate Keys Window.



Inputs

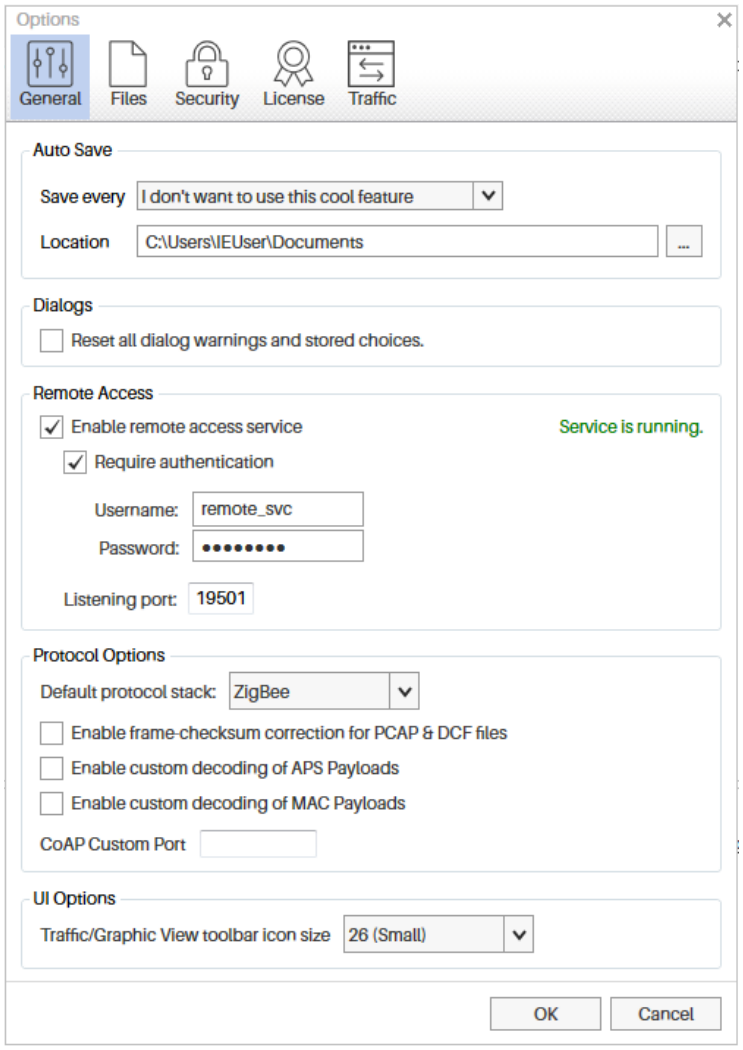
Thread Master Key: 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:01
Sequence Counter: 20:00:00:00
SC Message: The SC Message

Output

	Key
◢ IEEE 802.15.4 2006 Key	
✓	90:91:84:5F:C6:F7:E4:9C:DF:15:C7:A0:93:63:3D:EF
◢ Thread Network Key	
✓	A1:0C:4B:B7:F6:63:ED:DF:85:25:82:5A:B9:4B:9E:C6

Chapter 13: Setting Preferences

To start setting your preferences open the Options dialog by selecting the Tools > Options... menu item on the main window of Ubiqua. You will find more instructions on the following sections.



Auto Save

Save every: I don't want to use this cool feature

Location: C:\Users\IEUser\Documents

Dialogs

☐ Reset all dialog warnings and stored choices.

Remote Access

✓ Enable remote access service — Service is running.

✓ Require authentication

Username: remote_svc

Password: ●●●●●●●●

Listening port: 19501

Protocol Options

Default protocol stack: ZigBee

☐ Enable frame-checksum correction for PCAP & DCF files

☐ Enable custom decoding of APS Payloads

☐ Enable custom decoding of MAC Payloads

CoAP Custom Port

UI Options

Traffic/Graphic View toolbar icon size: 26 (Small)

Auto Recovery

Ubiqua features an option that can help you recover the last capture data in case of a problem. Once this "Auto recovery" option has been enabled and in case that Ubiqua does not close properly, you will be presented with a "Recover Capture" dialog the next time Ubiqua starts. To enable auto recovery first open the Options dialog using the Tools > Options... menu. Go to the "General" tab, and find the checkbox next to "Save a backup of the current file..." in the section called "Capture Files". When checked, a select box called "minutes" will appear for you to specify the time interval between each auto save of the recovery file.

As mentioned before, if Ubiqua is not properly closed a dialog will be displayed when Ubiqua is started; this dialog will display a list containing the capture files that can be recovered along with an approximate number of packets that each file contains. Press the Recover Capture button to start the recovery process.

Dialogs

Ubiqua implements dialogs to remember the user's choices within the system. This option is used to remember the user preference and to suppress the same dialog in the future. The dialog appears when a capture file that does not contain protocol information is opened. The user can choose one of the supported protocols to decode it and selecting 'remember my choice' set the selected protocol as default to open the following captures. A dialog opens if the Frame Check-Sum validation fails, and the user can choose among to append 2 more bytes with the correct checksum, to replace the last 2 bytes with the correct checksum, or do nothing; the "remember my choice" will suppress the dialog when opening following captures. Finally, a dialog will open each time a new filter is created or the selected filter is edited; the "remember my choice" will suppress the dialog and it will apply the last user's choice. The stored dialog selections can be reset using the "Reset all dialog warnings and stored choices" box on the General tab.

Remote Access

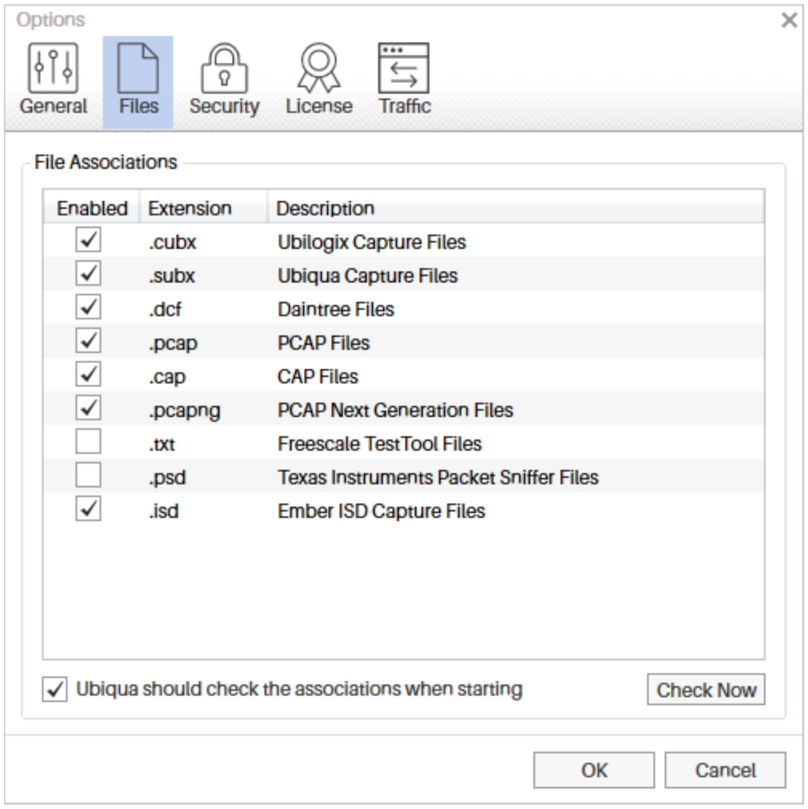
This option toggles on or off the Ubiqua Services which are web resources that expose certain Ubiqua functions for convenience. See here for more info.

Protocol Options

From this section you can change default protocol settings for unrecognized loaded capture files and customize other decoding options, such as the CoAP Custom Port that allows Ubiqua to listen on a designated UDP port and decode incoming traffic as CoAP.

File Associations

You can set up Ubiqua to open capture files by double clicking on them in your system. To do so, open Options dialog, select the Files tab, and you will see a list of the supported capture files and extensions. Click the checkboxes on the Active column to associate Ubiqua to that file format. Use the checkbox at the bottom to make Ubiqua check if the associations are correctly set on your system each time it starts.

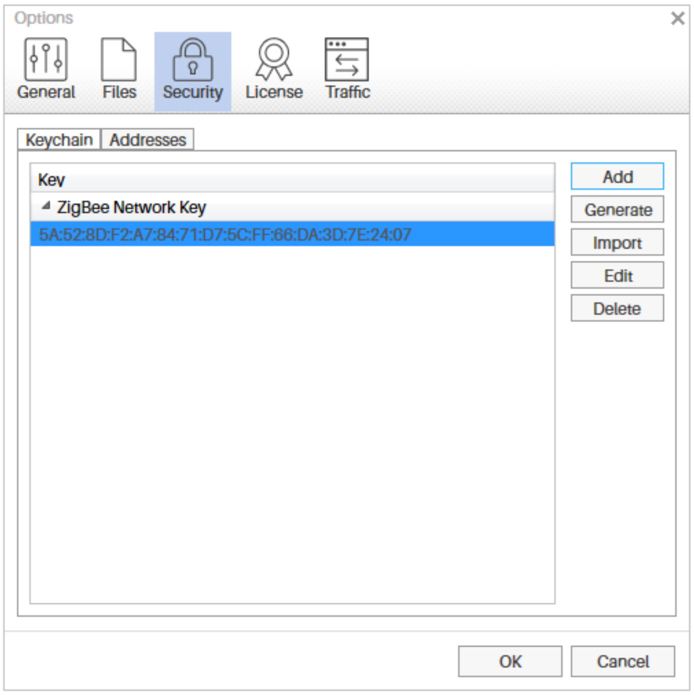


Enabled	Extension	Description
✓	.cubx	Ubilogix Capture Files
✓	.subx	Ubiqua Capture Files
✓	.dcf	Daintree Files
✓	.pcap	PCAP Files
✓	.cap	CAP Files
✓	.pcapng	PCAP Next Generation Files
	.txt	Freescale TestTool Files
	.psd	Texas Instruments Packet Sniffer Files
✓	.isd	Ember ISD Capture Files

✓ Ubiqua should check the associations when starting

Security Keys

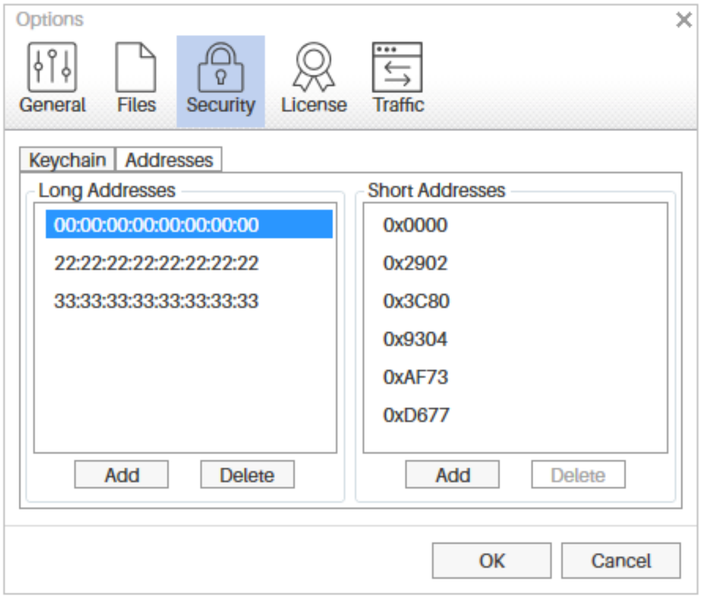
Security keys for decoding purposes are managed on the Security > Keychain tabs of the Options dialog. In this tab, you are presented with a list of all the security keys available in Ubiqua. This keys could have been discovered in the capture data or they could have been added manually. Use the buttons on the right to add, edit, delete, or import keys.



5A:52:8D:F2:A7:84:71:D7:5C:FF:66:DA:3D:7E:24:07

Addresses

Addresses are the relationships between long address and short address used when decoding packets. Their management is similar to the security keys case. The Options dialog includes a list of all the available addresses on the Security > Addresses tab. Use the buttons on the right to add, or delete one of the addresses.



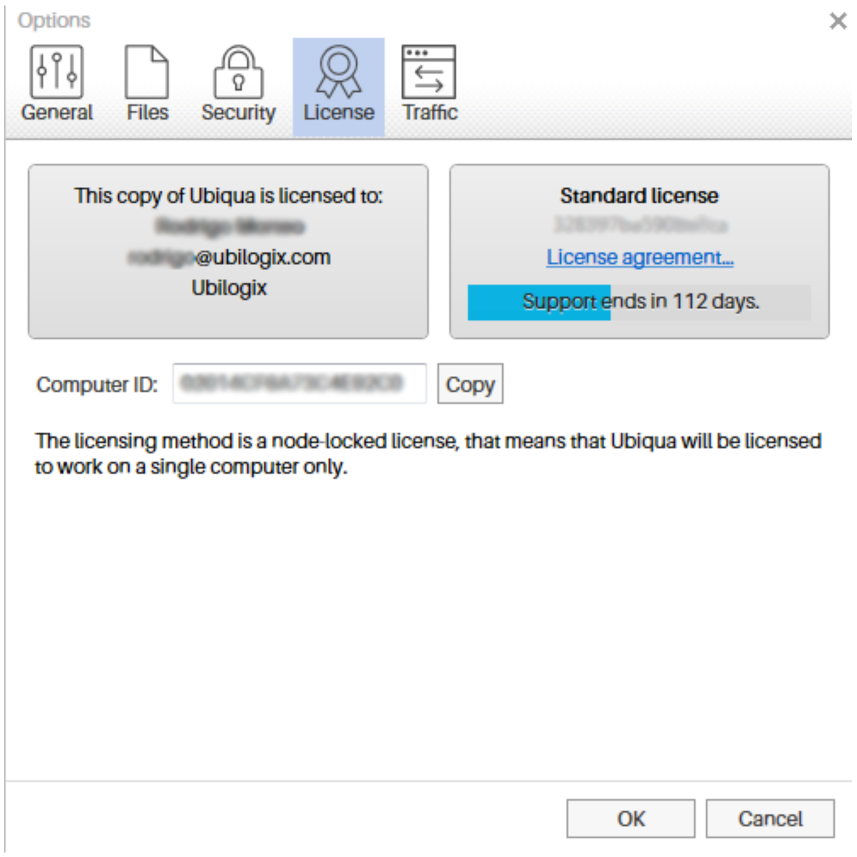
Long Addresses	Short Addresses
00:00:00:00:00:00:00:00	0x0000
22:22:22:22:22:22:22:22	0x2902
33:33:33:33:33:33:33:33	0x3C80
	0x9304
	0xAF73
	0xD677

License Management

The License Management option is the most important part of the Options dialog. You will find it in the License tab. Here, you can see who the owner of the license is (name, email, and business). At the top right section, the status of the license is shown in terms of how many remaining days are left for the license to be valid (in case of the 21/days evaluation) or the number of days left of support (in case of a full license).

Your Computer ID is also displayed on this section. Ubilogix uses this ID to match your license with your computer. Many of the actions of your license (such as transferring), are linked to this number so be sure to include it when requesting changes to your license to support@ubilogix.com.

Use the buttons at the bottom to upgrade your current license to a more inclusive one, to start the license transfer process, or to regenerate your certificate file. Note that an active Internet connection is required for all these processes.



This copy of Ubiqua is licensed to:
@ubilogix.com
Ubilogix

Standard license
License agreement...
Support ends in 112 days.

Computer ID:

The licensing method is a node-locked license, that means that Ubiqua will be licensed to work on a single computer only.

Check for Updates

The Check for Updates option is not included on the Options dialog. You will find it on the Help > Check for Updates... menu item. Use this option to know if a new version of Ubiqua is available, and if you are entitled to it (according to your license).

Environment Files

Environment files are loaded and saved from the File > Load/Save Environment menu items. These files include all the current selections and values of the Options dialog including address relationships, security keys, and others. Environment files do not store any license information.

Chapter 14: Supported Hardware

Ubiqua Protocol Analizer supports different sniffer hardware devices as specified in the table below. All the supported hardware drivers are located in `Program Files\Ubilogix\Drivers` folder in the drive where you installed the software. As provided from the vendor each USB dongle is already programmed with a sniffer firmware application. If you need more information about the sniffer firmware application or the sniffer drivers please refer to the Ubilogix support webpage.

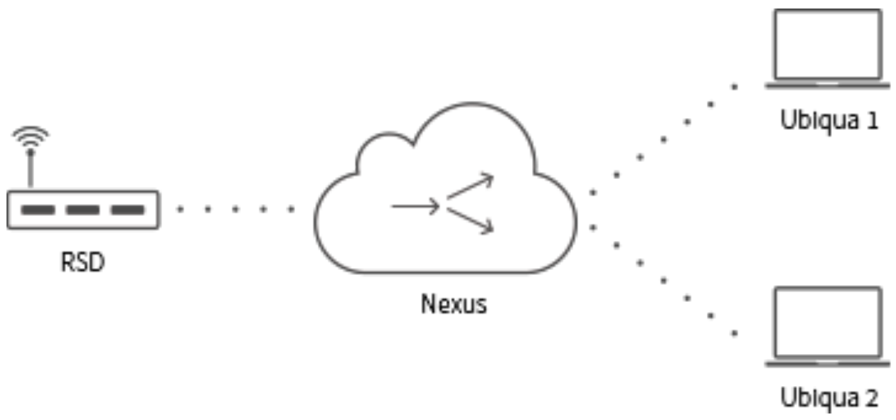
	Vendor	Name	Part Number	Drivers Folder
	Atmel	RZ600 Evaluation Kit	AT86RF212 and the AT86RF23x families	`\Drivers\ATMEL`
	NXP	MC1322x USB	1322X-USB	`\Drivers\FTDI`
	NXP	JN5168 USB Dongle	DR1198	`\Drivers\FTDI`
	NXP	JN5169 USB Dongle	OM15020	`\Drivers\FTDI`
	NXP	JN5179 USB Dongle	OM15021	`\Drivers\FTDI`
	NXP	KW2x: Kinetis KW2x Family	TWR-KW24D512 and KW24D512-USB	`\Drivers\FS KW2x`
	NXP	Kinetis® KW41Z-2.4 GHz	KW41Z	
	Rainforest Automation	TOUCAN Wireless Sniffer	RFA-Z106-SN TOUCAn	`\Drivers\FTDI`
	Texas Instruments	CC2531 Ev. Module Kit	CC2531EMK	`\Drivers\TI`
	Sewio	Open Sniffer	Open Sniffer for 802.15.4, Zigbee, 6LoWPAN	
	Ubisys	ZigBee USB Stick	ZigBee USB Stick U1	`http://www.ubisys.de/en/smarthome/download-software.html`

Chapter 15: Remote Sniffers

The Remote Sniffer Device system enables Ubiqua users to collect traffic data from stand-alone sniffer devices deployed in distant locations, with minimal effort required in terms of configuration and maintenance. Furthermore, it allows for the real-time sharing of captured data with multiple collaborators.

A supported Remote Sniffer Device (RSD) is configured to establish a connection with either our cloud-based Nexus service, or a private Nexus instance running on a controlled network. This service in turn proxies traffic and commands to and from authorized Ubiqua clients. The RSD owner may choose to share access to the device with other Ubiqua users.

Due to initiating rather than accepting connections, the RSD will normally not require manual configuration of firewalls in order to function.



Security and privacy

Traffic between the RSD and the Nexus service, as well as traffic between the Nexus service and Ubiqua, is sent over a TLS-encrypted websocket connection. The Nexus service does not store or process the transmitted information in any way, other than to forward it to the end-user client applications.

Support for end-to-end encryption between RSDs and Ubiqua clients is currently in development.

Chapter 16: Supported Protocols

IEEE 802.15.4

The 802.15.4 is a standard developed for Wireless Personal Area Networks (WPANs). WPANs convey information over short distances among the participants in the network. They enable small, power efficient, inexpensive solutions to be implemented for a wide range of applications and types of devices. Some key characteristics of an 802.15.4 network are:

- An over the air data rate of 250 Kbit/s in the 2.4 GHz band.
- 16 independent communication channels in the 2.4 GHz band.
- Large networks (up to 65534 devices).
- Devices use carrier sense multiple access with collision avoidance (CSMA-CA) to access the medium.
- Devices use Energy Detection (ED) for channel selection.
- Devices inform the application about the quality of the wireless link (Link Quality Indication).

The 802.15.4 Standard defines two network topologies, both using one and only one central device (the PAN coordinator). The PAN coordinator is the principal controller of the network.

- **Star Network Topology –** In a star network, all communication in the network is either to the PAN coordinator or from the PAN coordinator. That is, communication between non-PAN coordinator devices is not possible.
- **Peer-to-Peer Network Topology –** In a peer-to-peer network, communication can occur between any two devices in the network as long as they are within range of one another.

For more information visit the IEEE 802.15 Working Groups web sites for the latest specifications and other information: http://www.ieee802.org/15/pub/TG4.html

ZigBee

ZigBee is a specification for a suite of high-level communication protocols using small, low-power digital radios based on the IEEE 802.15.4 standard for wireless personal area networks (WPANs). ZigBee is targeted at RF applications that require a low data rate, long battery life, and secure networking. ZigBee and IEEE 802.15.4 based networks consist of many devices working together supporting sensing and control applications. Ubiqua PA has support for ZigBee 2007 stack profile 1 and 2, application ZigBee profile HA, SE, HC, CBA and TP2:

- ZigBee Stack Profile 1 (also known as ZigBee).
- ZigBee Stack Profile 2 (also known as ZigBee Pro).
- ZigBee Device Profile: ZigBee Device Object Profile, Profile ID: `0x0000`.
- ZigBee Application Profile HA: ZigBee Home Automation Application Profile, Profile ID: `0x0104`
- ZigBee Application Profile HA: ZigBee Home Automation Application Profile, Profile ID: `0x0109`
- ZigBee Application Profile HC: ZigBee Health Care Profile, Profile ID: `0x0108`.
- ZigBee Application Profile CBA: ZigBee Commercial Building Automation, Profile ID: `0x0105`.
- ZigBee Application Profile TP2: ZigBee Profile Test Profile #2, Profile ID: 0x7F01.

For more information visit the ZigBee Alliance web site for the latest specifications and other information: http://www.zigbee.org

Thread

The Thread stack is an open standard for reliable, cost-effective, low-power, wireless D2D (device-to-device) communication. It is designed specifically for Connected Home applications where IP-based networking is desired and a variety of application layers can be used on the stack. General characteristics of the Thread stack and network:

- Simple network installation, start up and operation.

- Secure.
- Small and large networks.
- Better range.
- No single point of failure.
- Low power.

For more information visit http://threadgroup.org/.

PopNet™

PopNet is a full mesh networking protocol and operating environment designed to work with inexpensive IEEE 802.15.4 radios and microcontrollers for low-power sensor and control applications.

For more information visit the San Juan Software web site for the latest specifications and other information: http://www.sanjuansw.com/

ZigBee RF4CE

ZigBee RF4CE is a radio protocol standard for remote control of consumer electronics devices based on 2.4GHz PHY/MAC IEEE 802.15.4 radios. Some characteristics:

- It supports multiple STAR topology with inter-PAN communication.
- Has a simple security (uses AES 128 Engine) set-up configuration for all devices.
- Provides simple pairing mechanism between devices.
- Provides power saving mechanisms for all device classes.
- Support profile for Consumer Electronics products.

IETF 6LowPAN

6LowPAN is not a protocol, is the name of the working group in the Internet area of the IETF. The 6LowPAN group has defined encapsulation and header compression mechanisms to allow IPv6 packets to be sent and received over the IEEE 802.15.4 based networks. The base specification developed by the 6Low PAN IETF group is RFC 4944, the problem statement document is RFC 4919. Several functionalities have been obtained from this work:

- Adapting the packet sizes of the two networks IPv6 and IEEE 802.15.4.
- Address resolution.
- Differing devices designs.
- Differing focus on parameter optimization.
- Adaptation layer for interoperability and packet format.
- Addressing management mechanisms.
- Routing considerations and protocol for mesh topologies in 6LowPANs
- Device and service discovery.

ZigBee Light Link

ZigBee Light Link gives the lighting industry a global standard for interoperable and very easy-to-use consumer lighting and control products. It allows consumers to gain wireless control over all their LED fixtures, light bulbs, timers, remotes and switches. Products using this standard will let consumers change lighting remotely to reflect ambiance, task or season, all while managing energy use and making their homes greener. Some features:

- Long battery life mesh technology.
- AES 128 encryption used to protect lighting network against unauthorized use.
- Interoperable with other ZigBee profiles.

- Control lighting products via remote, sensors or smart phones, tablets and computers.
- Simple Touchlink mechanism to add components and manage your lighting network.
- Ensures power-efficient control solutions and low maintenance cost.

For more information visit the ZigBee Alliance web site for the latest specifications and other information: http://www.zigbee.org/zigbee-for-developers/applicationstandards/zigbee-light-link

ZigBee Green Power

ZigBee Green Power enables new capabilities available to the ZigBee and ZigBee PRO networks. When the ZigBee Green Power standard is made available to Alliance members at the end of 2009, only ZigBee will offer an established, competitive marketplace for deploying switches, sensors and controllers using harvested energy in residential, commercial and industrial environments. Its energy harvesting capabilities will give manufacturers greater flexibility when designing innovative ZigBee products and solutions. Because ZigBee Green Power will work seamlessly with ZigBee and ZigBee PRO networks, it will enjoy all of ZigBee's numerous strengths.

For more information visit the ZigBee Alliance web site for the latest specifications and other information: http://www.zigbee.org

JenNet-IP

JenNet-IP is an IPv6 based low power wireless networking solution enabling the "Internet of Things". It provides a wireless command and control solution optimised for building automation with the internet connectivity, fast response and ability to control groups and set scenes demanded by these applications.

JenNet IP uses the IETF 6LoWPAN standard with a "mesh-under" networking approach provided by NXP's industry proven JenNet network layer. It provides a self-healing, highly robust and scalable tree network solution, for networks of up to 500 nodes.

JenNet IP features an easy to use and powerful Management Information Base (MIB) API, "JIP" that provides a powerful application layer for interoperable device management and control, enabling developers to develop products to suit all applications. Some features are:

- Wireless IPv6 networking enabling the "Internet of Things".
- Gateway or non-gateway options.
- Optimised for lighting and building automation.
- Self-healing and re-shaping tree network.
- JIP" API based on SNMP.
- “Mesh-under".
- Highly secure.
- Over-Network Upgrade future proofs.
- Low memory footprint.
- Low cost of ownership.

For more information, the latest specifications and other information see http://www.nxp.com/products/wireless-connectivity/2.4-ghz-wireless-solutions/jennet-ip:JENNET-IP

Chapter 17: Custom Payloads

Starting Ubiqua version 1.4, users are now able to use custom decoders for the Mac payload and Application ZigBee layers. The custom decoding is described by the user in a XML file.

To integrate the custom payload and start enjoying this functionality follow these instructions:

1. Create the custom payload XML file. The XML file has to be named with a specific name as it is required for the proper functionality with Ubiqua; the file depending of the case may be named as APS-Custom.xml or MAC-Custom-Payload.xml.

2. With Ubiqua closed, add the APS-Custom.xml file and/or MAC-Custom-Payload.xml file in the Ubiqua Data Folder. Depending of your operating system the Ubiqua Data Folder can be located at,

   ○ Windows XP: `C:\Documents and Settings\All Users\Application Data\Ubilogix\Ubiqua\`

   ○ Windows Vista or later: `C:\ProgramData\Ubilogix\Ubiqua`.

3. Run Ubiqua and enable the custom decoding, you can access this option through the Tools option of the main menu, then choose Options, and the General tab will open, at the bottom of this window you can find both options, custom decoding of APS or MAC payloads.

Creation of the custom decoder

The custom decoders are constructed by following the XML schema `Decoders.xsd`. Next section briefly explains the different elements and attributes used by the XML custom decoder.

The "Field" element

The "Field" element, reads an amount of bits from the frame. The Field element includes important xml attributes used for identifying and assigning meta information to the field. Some of the important attributes of the Field element are the following:

- **Name** – It's an optional attribute that assigns an ID to the value that was read into the Field element.

- **IsGroup** – Hides a Field element from the Packet View.

- **Label** – This attribute assigns a label to display into the PacketView, if the Label is not present, the default Label is “reserved".

- **Offset** – It's an Attribute with an integer value that defines the amount of offset bits from where the reading begins.

- **Type** – Defines the data type that will be interpreted on the PacketView.

An example of Field:

```
<Field   Name="CommandType"   Label="Command   Type"   Offset="8"   Length="16"
Type="Integer" />
```

In the example above, the field Contains an ID "CommandType" with the value that was read, has an offset of 8 bits, a length of 16 bits and the data type is integer. The label displayed into the PacketView is "Command Type”.

The "Binding" element

The "binding" element creates a link with a variable. Depending of the value of the element, the length of the field element will be determined.

Note: A variable is defined with the "name" attribute in the Field element.

Example:

```
<Field Label="Command Type" Name="CommandType" Length="8">
      <Options>
        <Option Value="0x00" Label="Start" />
        <Option Value="0x01" Label="Stop" />
        <Option Value="0x02" Label="Pause" />
        <Option Value="0x03" Label="Configure"/>
      </Options>
</Field>
 <!-- If the "CommandType" is 0x00 then the length of  "Command Payload"
field is 8 bits (0x01 -> 16,  0x02-> 24, 0x03 -> 32, any other value is 0
bits)  -->
```

```
<Field Label="Command Payload">
  <Field.Length>
    <Binding Source="CommandType">
      <Case Value="0x00" Result="8" />
      <Case Value="0x01" Result="16" />
      <Case Value="0x02" Result="24" />
      <Case Value="0x03" Result="32" />
      <Case Value="0x04-0xFF" Result="0" />
    </Binding>
  </Field.Length>
  <!--Payload-->
</Field>
```

Custom decoder Field data Types

Each Field element has a Type attribute which is used to indicate how to parse and display the field value. The possible data types of the Type attribute are the following:

- **Hexadecimal** – Value should be parsed as an unsigned integer and displayed as an hexadecimal string (default).
- **Bits** – Value should be parsed as an unsigned integer and displayed as an element of a bitmap.
- **Integer** – Value should be parsed and displayed as an unsigned integer.
- **SignedInteger** – Value should be parsed and displayed as a signed integer.
- **String** – Value should be parsed and displayed as an ASCII string.
- **Boolean** – Value should be parsed and displayed as a boolean.
- **DateTime** – Value should be parsed and displayed as a date time. The format must be "yyyy-MM-dd HH:mm:ss.SSSSSS", as defined by the LDML standard.
- **Time** - Value should be parsed and displayed as a time. The format must be "HH:mm:ss.SSSSSS", as defined by the LDML standard.
- **Seconds** – Value should be parsed and displayed as seconds. The format must be "ss.SSSSSS", as defined by the LDML standard.
- **SignalPower** – Value should be parsed as an integer and displayed with the "dBm" postfix.

- **Address** – Value should be parsed as an unsigned integer and displayed as an hexadecimal string with digits separated by a colon only if the length is 6 or 8 bytes.
- **Key** – Value should be parsed as an hexadecimal string and displayed as a security key.

Loop functionality

The "isRepeteable" attribute of the Field element allows creating a Field with a loop functionality. If the "isRepeteable" is sets to true, the field repeats the content until the length is equals to 0.

Example:

```
<Field Label="Commands" Length="80">
      <Field Label="Command" Length="8" IsRepeatable="true">
        <Field Label="Field 1" Length="2" />
        <Field Label="Field 2" Length="6" />
      </Field>
 </Field>
```

In the above example, the field with the label "Command" will be repeated 10 times because its length is 8 bytes and the length of the Commands field is 80 bytes.

Hiding unnecessary fields in the Ubiqua Packet View.

If there is a "Field" element that is not required to be visible in the Packet View, add the "isGroup" attribute.

Example:

```
<Field Label="Values" IsGroup="true">
 <Field Label="Value" Length="8" />
</Field>
```

In the above example only the Field with the label "Value" is shown in the Packet View.

Examples

Custom decoder complete examples

Custom MAC payload example

The MAC Custom Payload only works using the IEEE 802.15.4-2003 and IEEE 802.15.4-2006 protocol stacks. MAC-Custom-Payload.xml contains the MAC header variables needed to decode the custom payload.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    File: MAC-Custom.xml
    Abstract: The layer definition of MAC Custom Payload

    Disclaimer: IMPORTANT: This Ubilogix software is supplied to you by Ubilogix
    International Inc. ("Ubilogix") in consideration of your agreement to the
    following terms, and your use or modification of this Ubilogix software
    constitutes acceptance of these terms. If you do not agree to these terms,
    please do not use or modify this Ubilogix software.

    In consideration of your agreement to abide by the following terms, and subject
    to these terms, Ubilogix grants you a personal, non-exclusive license, under
    Ubilogix's copyrights in this original Ubilogix software ("the Ubilogix
    Software"), to use and modify the Ubilogix Software provided that you must
    retain this notice and the following text and disclaimers. Except as expressly
    stated in this notice, no other rights or licenses, express or implied, are
    granted by Ubilogix herein, including but not limited to any patent rights that
    my be infringed by your derivative works or by other works in which the Ubilogix
    Software may be incorporated.

    THE UBILOGIX SOFTWARE IS PROVIDED BY UBILOGIX ON AS "AS IS" BASIS. UBILOGIX
    MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE
    IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A
    PARTICULAR PURPOSE, REGARDING THE UBILOGIX SOFTWARE OR ITS USE AND OPERATION
    ALONE OR IN COMBINATION WITH YOUR PRODUCTS.

    IN NO EVENT SHALL UBILOGIX BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL OR
    CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE
    GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
    ARISING IN ANY WAY OUT OF THE USE, REPRODUCTION, MODIFICATION AND/OR DISTRIBUTION
    OF THE UBILOGIX SOFTWARE, HOWEVER CAUSED AND WHETHER UNDER THEORY OF CONTRACT,
    TORT (INCLUDING NEGLICENCE), STRICT LIABILITY OR OTHERWISE, EVEN IF UBILOGIX HAS
    BEEN ADVICED OF THE POSSIBLITY OF SUCH DAMAGE.

    Copyright (C) 2015 Ubilogix International, Inc. All rights reserved.
-->
<Layer xmlns="urn:ubilogix:decoders"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ubilogix:decoders ../../Schemas/Decoders.xsd"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    Name="MAC Custom Payload" ShortName="Custom" Language="en-US" Version="1.0.1">

  <Colors>
```

```xml
    <Color Name="NC001" Value="Black" />
    <Color Name="NC002" Value="Red" />
    <Color Name="RC001" Value="Pink" />
    <Color Name="RC002" Value="LightGreen" />
  </Colors>
  <Variables>
    <!--The Frame Type from MAC header-->
    <Variable Name="FrameType" Value="0xF" />
    <!--MAC security bit from MAC header -->
    <Variable Name="MacSecurityEnabled" Value="0xF" />
    <!--The destination PAN ID-->
    <Variable Name="DstPanId" Value="0xF" />
    <!--The destination Address-->
    <Variable Name="DstAddr" Value="0xF" />
    <!--The Source PAN ID-->
    <Variable Name="SrcPanId" Value="0xF" />
    <!--The Source Address-->
    <Variable Name="SrcAddr" Value="0xF" />
  </Variables>

  <Fields>
    <Field Name="DataPayload" Label="Data Payload">
      <Field Label="Payload" Length="Stretch">
        <!--Payload Here-->
      </Field>
    </Field>
  </Fields>
</Layer>
```

Custom APS decoder example

The APS Custom Layer allows decoding the payload for APS when the Profile ID is private or to decode a specific command from ZCL when the Cluster ID is private.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<!--
    File: APS-Custom.xml
    Abstract: The layer definition of APS Custom Payload

    Disclaimer: IMPORTANT: This Ubilogix software is supplied to you by Ubilogix
    International Inc. ("Ubilogix") in consideration of your agreement to the
    following terms, and your use or modification of this Ubilogix software
    constitutes acceptance of these terms. If you do not agree to these terms,
    please do not use or modify this Ubilogix software.

    In consideration of your agreement to abide by the following terms, and subject
    to these terms, Ubilogix grants you a personal, non-exclusive license, under
    Ubilogix's copyrights in this original Ubilogix software ("the Ubilogix
    Software"), to use and modify the Ubilogix Software provided that you must
    retain this notice and the following text and disclaimers. Except as expressly
    stated in this notice, no other rights or licenses, express or implied, are
    granted by Ubilogix herein, including but not limited to any patent rights that
    my be infringed by your derivative works or by other works in which the Ubilogix
    Software may be incorporated.

    THE UBILOGIX SOFTWARE IS PROVIDED BY UBILOGIX ON AS "AS IS" BASIS. UBILOGIX
    MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE
    IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A
    PARTICULAR PURPOSE, REGARDING THE UBILOGIX SOFTWARE OR ITS USE AND OPERATION
    ALONE OR IN COMBINATION WITH YOUR PRODUCTS.

    IN NO EVENT SHALL UBILOGIX BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL OR
    CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE
    GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
    ARISING IN ANY WAY OUT OF THE USE, REPRODUCTION, MODIFICATION AND/OR DISTRIBUTION
    OF THE UBILOGIX SOFTWARE, HOWEVER CAUSED AND WHETHER UNDER THEORY OF CONTRACT,
    TORT (INCLUDING NEGLICENCE), STRICT LIABILITY OR OTHERWISE, EVEN IF UBILOGIX HAS
    BEEN ADVICED OF THE POSSIBLITY OF SUCH DAMAGE.

    Copyright (C) 2015 Ubilogix International, Inc. All rights reserved.

-->
<Layer xmlns="urn:ubilogix:decoders"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ubilogix:decoders ../../Schemas/Decoders.xsd"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    Name="APS Custom" ShortName="Custom" Language="en-US" Version="1.0.0">

  <Colors>
    <Color Name="NC001" Value="Black" />
    <Color Name="NC002" Value="Red" />
    <Color Name="RC001" Value="Pink" />
    <Color Name="RC002" Value="LightGreen" />
  </Colors>
  <Variables>
    <!--APS Header Variables-->
    <Variable Name="ApsZclClusterId" Value="0xFFFF" />
    <!--Cluster ID value from APS Header-->
    <Variable Name="ApsFrameType" Value="0xF" />
    <!--Frame Type value from APS Header-->
    <Variable Name="DeliveryMode" Value="0xF" />
    <!--Delivery Mode value from APS Header-->
    <Variable Name="AckFormat" Value="0xF" />
    <!--Acknowledgement Format value from APS Header-->
    <Variable Name="SecurityEnabled" Value="0xF" />
    <!--Security Enabled value from APS Header-->
    <Variable Name="DstEndpoint" Value="0xFF" />
    <!--Destination Endpoint value from APS Header-->
    <Variable Name="ProfileId" Value="0xFF" />
    <!--Profile ID value from APS Header-->
    <!--ZCL Header Variables-->
    <Variable Name="ZclGeneralCmdFrame" Value="0xFFF" />
    <!--General Command Frame value from ZCL Header-->
    <Variable Name="ZclDirection" Value="0xF" />
    <!--Direction value from ZCL Header-->
    <Variable Name="ZclFrameType" Value="0xF" />
    <!--Frame Type value from ZCL Header-->
    <Variable Name="ManufacturerSpecific" Value="0xF" />
    <!--Manufacturer Specific value from ZCL Header-->
    <Variable Name="ZclSpecificPrivatedCmd" Value="0xFF" />
    <!--Specific Privated Command value from ZCL Header-->
    <Variable Name="ZclTransSequenceNumber" Value="0xFF" />
    <!--Transaction Sequence Number value from ZCL Header-->
    <Variable Name="ZclSpecificCommand" Value="0xFF" />
  </Variables>
  <Fields>

    <!--For APS Custom Payload-->
    <Field Name="DataPayload" Label="DataPayload">
      <Field Label="APS Payload">
        <!--Payload Here-->
      </Field>
    </Field>

    <!--For ZCL Custom Specific Command-->
    <Field Name="SpecificCommandPayload" Label="SpecificCommandPayload">
      <Field Label="Specific Command Payload" Length="Stretch">
        <!--Payload  Here-->
      </Field>
    </Field>
  </Fields>
</Layer>
```

Specific functionality examples

The following examples shows some of the functionality used for defining your own custom decoder.

Example 1

In this example, depending of the value of the "Command Type" Field the payload will be decoded accordingly. The example uses a Field named "Payload Field", with this field the length of the payload is calculated.

```
  <Field Label="Custom Payload Enabled" Length="Stretch">
        <!--Reads a field with 8 bits of length , saves the value of the field in
"CommandType" and display the "Type" of command depending of the value-->
        <Field Name="CommandType" Label="Command Type" Length="8">
          <Options DefaultLabel="Unknown Command">
            <Option Value="0x00" Label="Type 1"/>
            <Option Value="0x01" Label="Type 2"/>
            <Option Value="0x02" Label="Type 3"/>
            <Option Value="0x03" Label="Type 4"/>
          </Options>
        </Field>
        <!--Reads a field with 16 bits of length, saves the value in "PayloadLength"--
>
        <Field Name="PayloadLength" Label="Payload Length" Length="16" Type="Integer">
          <Options DefaultLablel="Bits" />
        </Field>
        <!--Depending the value of "CommandType" appears the Command Type N-->
        <Field Label="Payload" GroupsSource="CommandType">
          <DefaultFieldsGroup>
            <Field Label="Unknown Command Payload" Length="Stretch" />
          </DefaultFieldsGroup>
          <!-- If the Command Type is 0x00-->
          <FieldsGroup Group="0x00" Label="Payload Command Type 1">
            <Field Label="Payload Command Type 1" NodeColor="NC001">
              <Field.Length>
                   <!-- The length of this field is based on the value of the field
PayloadLength -->
                <Binding Source="PayloadLength" DefaultResultMultiplier="1" />
              </Field.Length>
              <Field Label="Payload 1" Length="Auto">
                <Field Label="Number of Company" Length="8" Type="Integer" />
                <Field Label="Name of Company" Length="80" Type="String" />
                <Field Name="Status" Label="Status" Length="8">
                  <Options>
                    <Option Value="0x00-0xF0" Label="OK" />
                    <Option Value="0xF1-0xFF" Label="Bad"/>
                  </Options>
                </Field>

                <!-- Field "Condition" appear if Status is 0x00 to 0xF0-->
                <Field Label="Condition">
                  <Field.Length>
                    <Binding Source="Status">
                      <Case Value="0x00-0xF0" Result="Auto"/>
                    </Binding>
                  </Field.Length>
                  <Field Label="Payload" Length="Stretch" />
                </Field>

              </Field>
            </Field>
          </FieldsGroup>

          <!-- If the Command Type is 0x01-->
          <FieldsGroup Group="0x01" Label="Payload Command Type 2">
            <Field Label="Payload Command Type 2" NodeColor="NC001">
              <Field.Length>
                   <!-- The length of this field is based on the value of the field
PayloadLength -->
                <Binding Source="PayloadLength" DefaultResultMultiplier="1" />
              </Field.Length>
              <Field Label="Payload 2" Length="Stretch" />
            </Field>
          </FieldsGroup>

          <!-- If the Command Type is 0x02-->
          <FieldsGroup Group="0x02" Label="Payload Command Type 3">
            <Field Label="Payload Command Type 1" NodeColor="NC002">
              <Field.Length>
                   <!-- The length of this field is based on the value of the field
PayloadLength -->
                <Binding Source="PayloadLength" DefaultResultMultiplier="1" />
              </Field.Length>
              <Field Label="Payload 2" Length="Stretch" />
            </Field>
          </FieldsGroup>
         <!-- If the Command Type is 0x03-->
          <FieldsGroup Group="0x03" Label="Payload Command Type 4">
            <Field Label="Payload Command Type 1" NodeColor="NC002">
              <Field.Length>
                   <!-- The length of this field is based on the value of the field
PayloadLength -->
```

```
            <Binding Source="PayloadLength" DefaultResultMultiplier="1" />
          </Field.Length>
          <Field Label="Payload 2" Length="Stretch" />
        </Field>
      </FieldsGroup>
    </Field>
  </Field>
```

Example 2

This example illustrates the use of the ZCL Header.

```
<Field Label="Payload Example 2" Length="Stretch">

         <!--To include the ZCL Header decoding, is needed add the next line, if the
APS Payload custom do not include decoding for ZCL Header just omits the next line-->
        <Field Label="ZCL Header">
          <Break Target="Layer" LayerName="ZigBee ZCL" FieldName="DataPayload" />
        </Field>

        <Field Label="Command Type" IsGroup="true" GroupsSource="ZclFrameType">
          <FieldsGroup Group="0x1" Label="Specific Command">
                                        <Field    Label="Specific    Command    Payload"
GroupsSource="ZclSpecificPrivatedCmd">
              <FieldsGroup Group="0x00" Label="Command Type 0">
                <Field Label="Command Type 0" Length="Stretch" />
              </FieldsGroup>
              <FieldsGroup Group="0x01" Label="Command Type 1">
                <Field Label="Command Type 1" Length="Stretch" />
              </FieldsGroup>
              <!--More FieldsGroup with different "Group" value-->
            </Field>
          </FieldsGroup>
        </Field>
      </Field>
```

Chapter 18: Troubleshooting

Cannot Start a Device

Please check the port is not being used by another program in your system. If this does not solve the issue, unplug and then plug the device back in again.

Getting Further Help

If you need further help, please open a support ticket via your Dashboard and we will be happy to assist you.

To help us to diagnose your issue, please include the `DebugLog.txt` file located in `C:\Users\USERNAME\AppData\Roaming\Ubilogix\Ubiqua`. You can reach this folder quickly by typing `%AppData%` in your File Explorer address bar.