Veronte Link

Release 6.14.28

Embention

2024-06-13

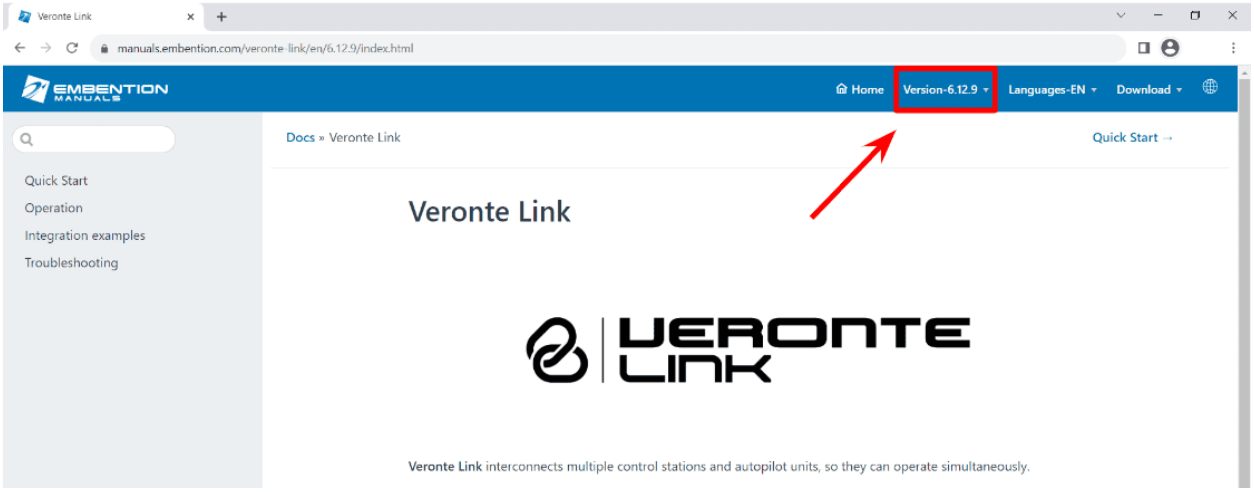
CONTENTS

Veronte Link interconnects multiple control stations and autopilot units, so they can operate simultaneously.

Important: This app is **backwards compatible** as long as it matches the "major" version (versions are composed as `major.minor.revision`, e.g. `6.12.22`), so users should always use the **last version** that is within the same "major" version.

Contact Embention to ensure having the latest version, please see Joint Collaboration Framework user manual or contact sales@embention.com.

The following image shows where to select a version from any Embention user manual.



CHAPTER

ONE

QUICK START

Veronte Link establishes **communication between a computer and any Veronte product** by creating a **VCP bridge**. It allows to use multiple control stations and autopilots to be interconnected, operating simultaneously.

Veronte Link also includes a **post-flight viewer**, to reproduce all recorded data from previous flights and generate plots and reports.

Veronte Link supports **Windows operating system**.

Note: **Windows 10** is recommended, but **Windows 11** is supported.

1.1 System Requirements

Before executing this software, users should check the following sections with the minimum and recommended PC hardware requirements.

Minimum requirements

- CPU: Intel Core i5-8365UE
- RAM: 8 GB DDR4
- STO: 256 GB SSD

Recommended requirements

- CPU: 12th Gen Intel(R) Core(TM) i7-12700H 14 cores up to 4,70 GHz
- RAM: 32,0 GB
- STO: 1TB SSD M.2 NVMe PCIe

1.2 Installation

Once a Veronte device has been purchased, a GitHub release should be created for the customer with the application.

To access to the release and download the software, read the Releases section of the **Joint Collaboration Framework** manual.

To install **Veronte Link** on Windows, execute Veronte Link.exe and follow the indications of the *Setup Wizard*.

Warning: In case of any issue during installation, please disable Windows Defender and Firewall. To disable Firewall, go to "Control Panel" and "Firewall of windows", then click on **Turn off**.

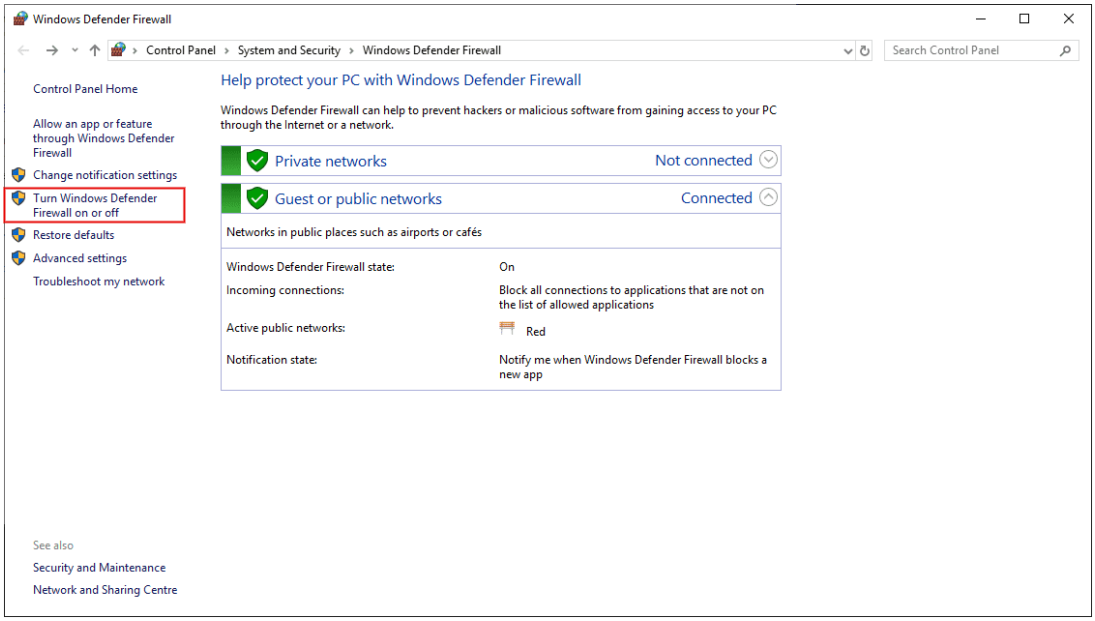


Fig. 1: **Windows Defender Firewall**

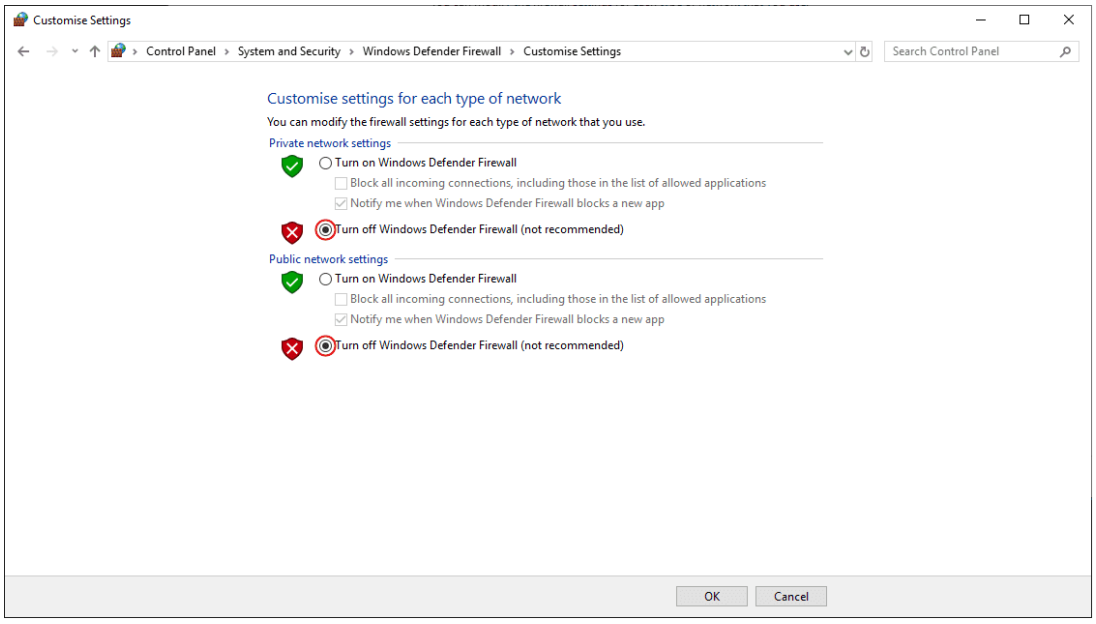


Fig. 2: **Windows Defender Firewall - Customize Settings**

CHAPTER

TWO

ADDITIONAL APPS

2.1 Veronte UDP Telemetry CLI

Veronte UDP Telemetry CLI is an additional command-line tool which allows **Veronte Link** to send Autopilot 1x telemetry over UDP.

2.1.1 Download

Once the **Veronte Autopilot 1x** has been purchased, a GitHub release should be created for the customer with the application.

To access to the release and download the software, read the Releases section of the **Joint Collaboration Framework** manual.

2.1.2 Installation

To install **Veronte UDP Telemetry CLI** on Windows just execute "veronte-udp-telemetry-cli.exe" and follow the indications of the *Setup Wizard*.

> **Warning:** If users have any problems with the installation, please disable the antivirus and the Windows firewall. Disabling the antivirus depends on the antivirus software.
>
> To disable the firewall, go to "Control Panel" → "System and Security" → "Windows Defender Firewall" and then, click on "Turn windows Defender Firewall on or off".

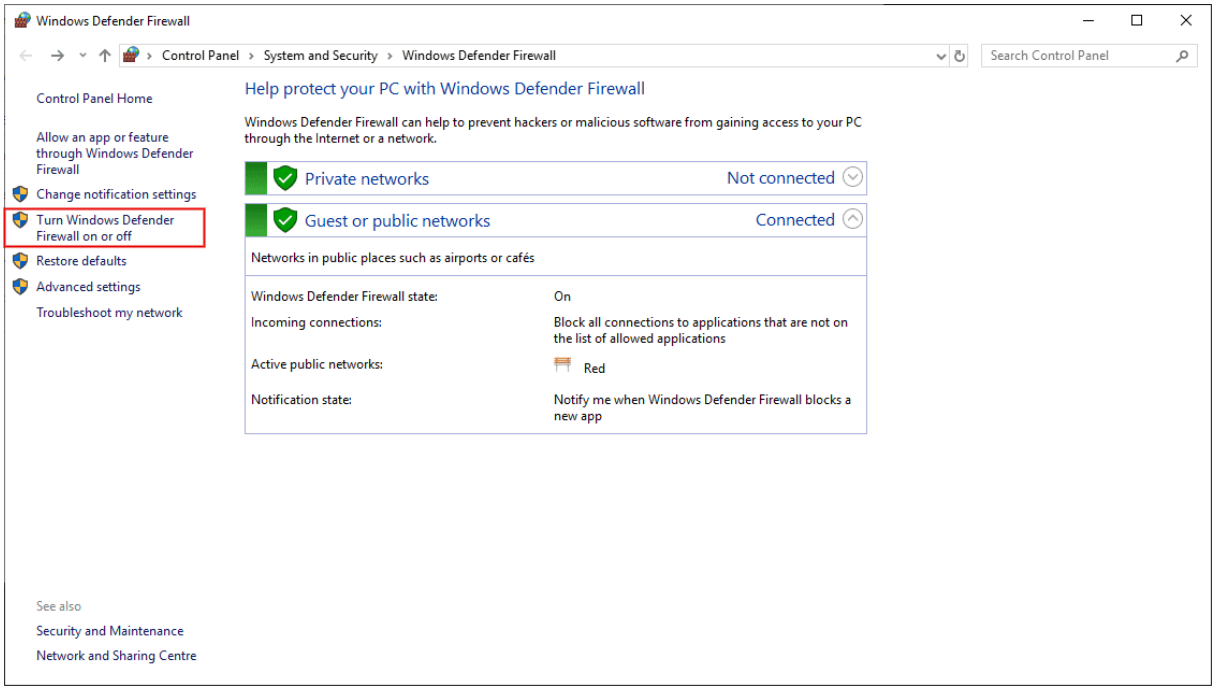


Fig. 1: **Windows Defender Firewall**

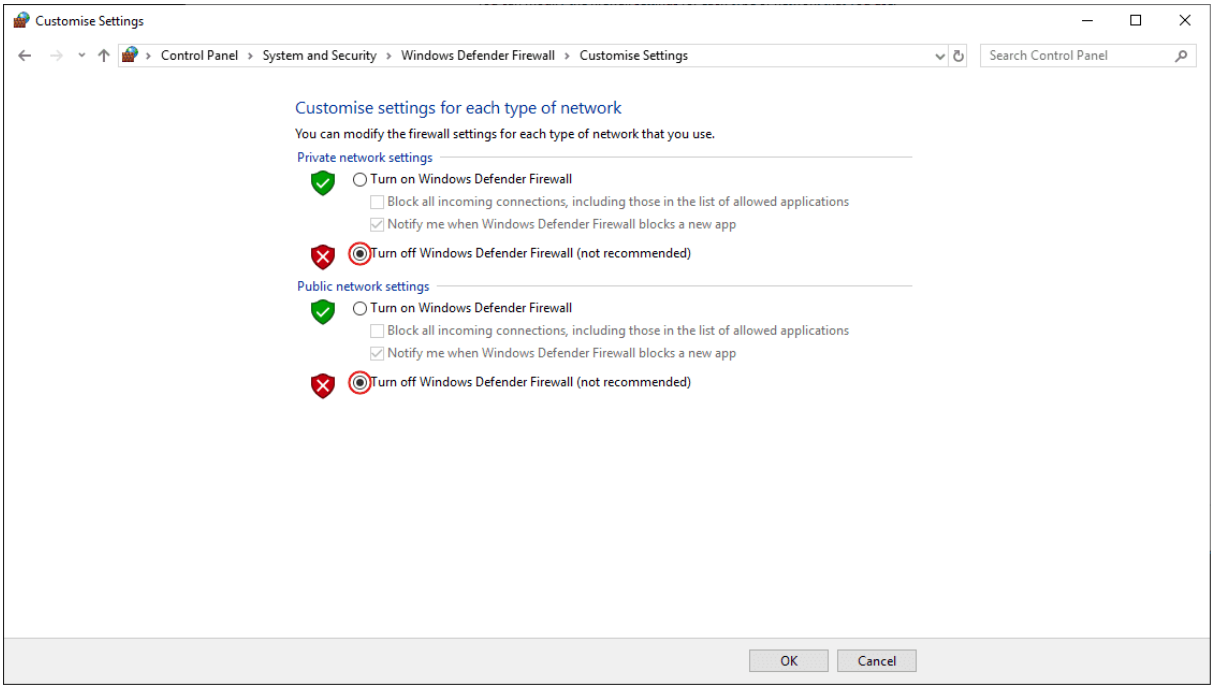


Fig. 2: **Windows Defender Firewall: Customize Settings**

2.1.3 Configuration

The following sections detail the steps to **configure** the Veronte system to transmit telemetry UDP messages through **Veronte UDP Telemetry CLI**, after it is installed.

2.1.3.1 1x PDI Builder

First, in **1x PDI Builder**, the intended variables to send must be added to the corresponding telemetry vector.

To do this:

1. Go to Telemetry menu → **Telemetry panel**.
2. By clicking the corresponding ➕ button, add the desired telemetry variables to one of the telemetry vectors *Data to VApp*.

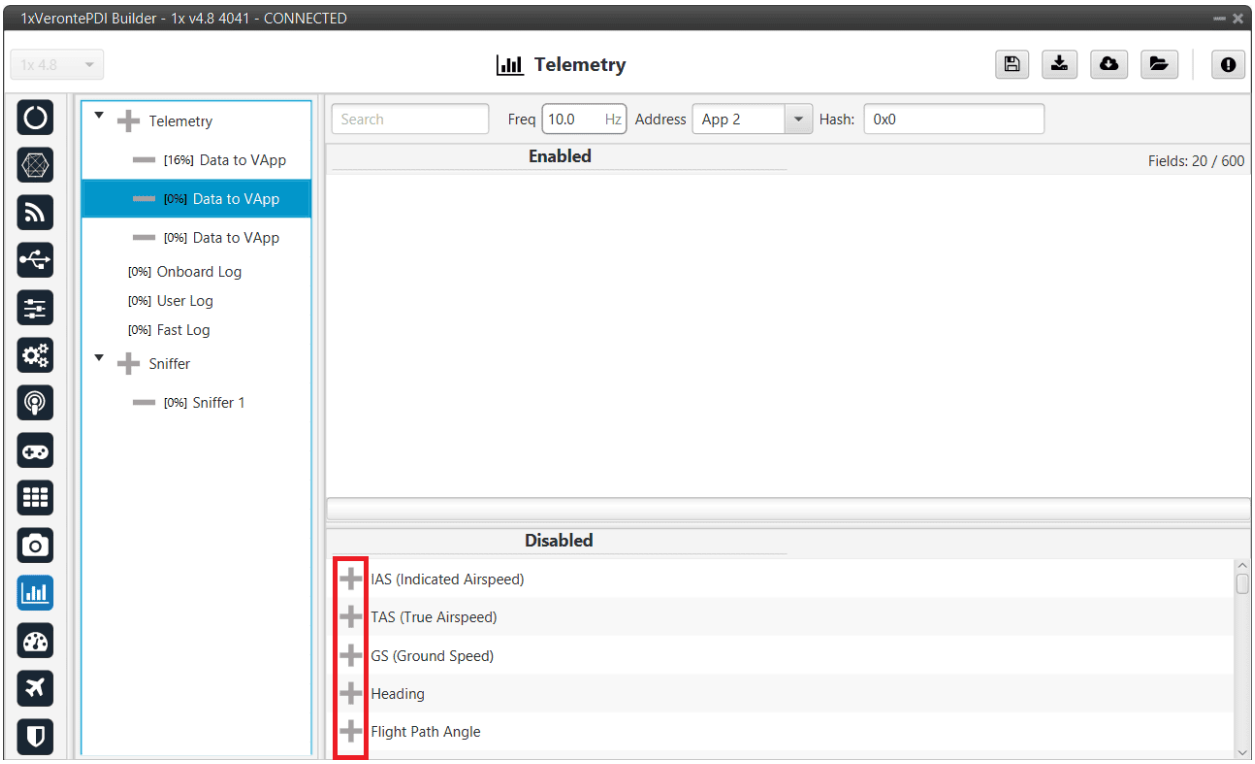


Fig. 3: **Add variables**

Note: For further information about this Telemetry menu, please refer to the Telemetry section of **1x PDI Builder** user manual.

---

3. Configure the *Data to VApp* vector where the variables have been added as follows:
   - **Frequency**: Desired frequency of data transmission
   - **Address**: App 2 (Veronte apps address)

Note: Hash parameter is not configurable, it is automatically calculated by the system based on the telemetry vector configured by the user. It is a hexadecimal representation of the CRC of the fieldset.

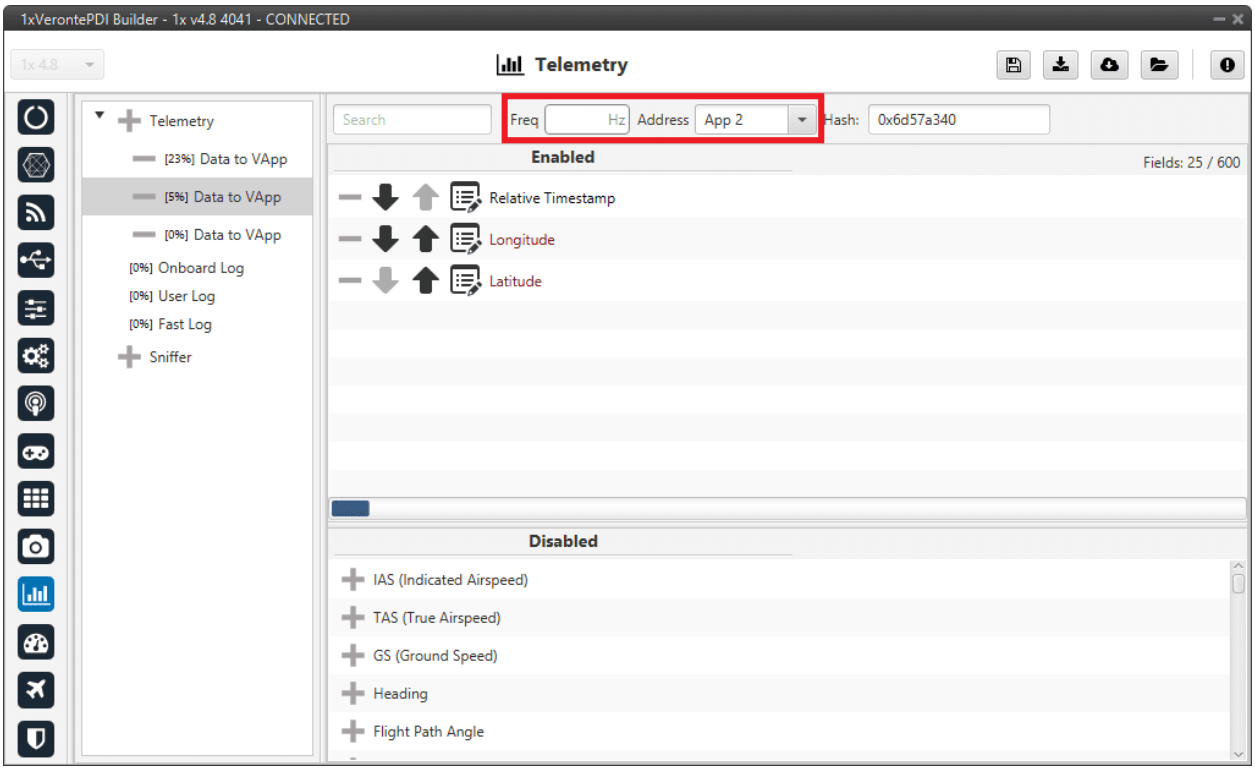


Fig. 4: **Data vector parameters**

4. Save the changes by clicking  button.

2.1.3.2 Veronte UDP Telemetry CLI

Veronte UDP Telemetry CLI has a configuration file (`tudp.config`) where users must specify which telemetry variables to send. Once the app is installed, this file can be found in `C:/Users/user/AppData/Roaming/VeronteUDPTelemetryCli`:

Fig. 5: **Configuration file**

Caution: On Windows, the AppData folder is hidden by default, if it is not visible in `C:\Users\user`, users can "show" it by checking the "Hidden Items" checkbox:

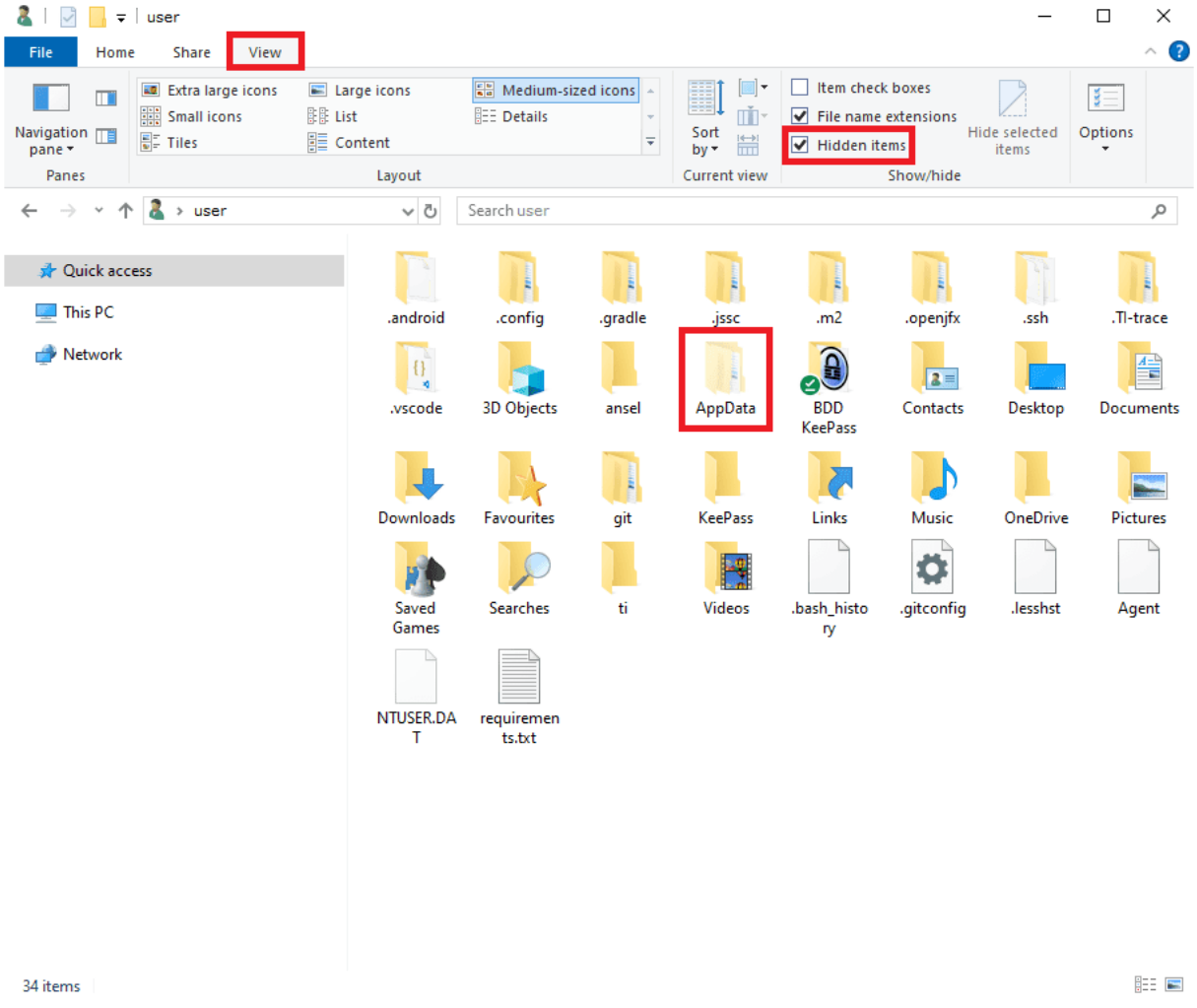


Fig. 6: **Windows File Explorer**

In `tudp.config`, there is a table where users must fill the following information for each telemetry variable to send:

Important: The variables configured in the `tudp.config` file must match the previous configuration from *1x PDI Builder configuration* section of this manual, so each variable is parsed according to the organization of the bits.

- **MULT**: Scale factor (the variable is multiplied by this number)
- **OFFSET**: Offset factor (this number is added to the variable)
- **TVAR**: Type of variable coding
  - **byte**: Unsigned byte (0 to 255)
  - **bit**: A desired number of bits
  - **UInt16**: Unsigned 16-bit integer (0 to 65.536)

  - **Int16**: Signed 16-bit integer (-32.768 to 32.768)
  - **UInt32**: Unsigned 32-bit integer (-2.147.483.648 to 2.147.483.648)
  - **Int32**: Signed 32-bit integer (0 to 4.294.967.295)
  - **Float**: 32 bit single-precision floating-point ($3.4028237 \cdot 10^{38}$ to $1.175494 \cdot 10^{-38}$)
- **UAV**: Serial Number of the Autopilot 1x where the variables come from.
- **VERVAR**: Type of variable in Veronte system.
  - **RVAR**: Real variables
  - **UVAR**: Integer variables
  - **BIT**: Bit variables
- **ID**: Identifier of the variable in Veronte. Refer to the Lists of Variables - Lists of interest section of **1x Software Manual** for Index-Variable correspondence or check it on the Variables panel of the UI menu of **1x PDI Builder** app.
- **UNIT**: Index of the unit of measure of the variable. Please, see the *Index-Unit correspondence table* for detailed information.

This a configuration example for *Relative Timestamp*, *Longitude* and *Latitude* variables:

```
#MULT     OFFSET  TVAR      UAV     VERVAR    ID      UNIT
1000      0       UInt32    4041    RVAR      300     NONE          //Time Since Hardware Start-Up (Milliseconds)
1         0       Float     4041    RVAR      500     NONE          //Longitude
1         0       Float     4041    RVAR      501     NONE          //Latitude
```

Fig. 7: **Relative Timestamp, Longitude and Latitude example**

2.1.3.3 Index-Unit correspondence table

Unit ID	Unit
0	m/s
1	kt
2	km/h
3	mph
4	ft/s
121	ft/m
321	mm/s
5	m
6	km
62	mm
63	cm
7	mi
8	NM
9	yd
10	ft
11	in
12	m/s^2
13	ft/s^2
14	in/s^2

continues on next page

Table 1 – continued from previous page

Unit ID	Unit
15	g (gravity)
202	rad
16	rad $[-\pi, \pi]$
203	rad [0, 2π]
205	º
17	º [-180,180]
101	º [0,360]
102	º ‘ ‘’
103	º ‘ ‘’ (N/S)
104	º ‘ ‘’ (E/W)
21	T
160	nT
23	G
22	mG
24	V
25	mV
26	A
27	mA
340	kA
28	Pa
29	kPa
30	bar
31	mbar
32	psi
33	mmHg
34	at
35	atm
147	Pa^2
36	K
37	ºC
38	ºF
39	s
120	Time
40	min
41	h
330	ns
108	μ s
109	ms
42	rad/s
117	º/s
43	rad/min
44	rad/h
45	rps
46	rpm
47	rph
57	m^3/s
58	gal/s
54	gal/h
59	l/s

continues on next page

Table 1 – continued from previous page

Unit ID	Unit
55	l/h
56	–
60	x1
64	‰
61	pkts/s
105	Hz
106	mHz
107	kHz
140	Bd
141	kBd
142	MBd
110	m^2
111	cm^2
112	mm^2
113	km^2
114	$mile^2$
115	ft^2
116	yd^2
118	bit
119	byte
131	KB
132	GB
122	kg
123	g
124	tonnes
125	lbs
126	oz
127	N
128	kN
129	lbf
130	pdl
134	rad/s^2
135	rad/min^2
136	rad/h^2
137	$º/s^2$
138	$º/m^2$
139	$º/h^2$
329	rpm/s
143	T^2
144	$(m/s)^2$
145	$(cm/s)^2$
146	$(mm/s)^2$
327	Ω
328	Henrios
322	watios
323	kW
324	Kgm/s
325	erg/s
326	cv

continues on next page

Table 1 – continued from previous page

Unit ID	Unit
331	m^3
332	dm^3
333	mm^3
334	L
335	mL

2.1.4 Operation

This section details the steps to **transmit telemetry UDP messages** through **Veronte UDP Telemetry CLI**.

2.1.4.1 Sending UDP messages

Veronte UDP Telemetry CLI connects to **Veronte Link** to send the previously configured Autopilot 1x telemetry via UDP messages. For this reason, the connection between the autopilot and **Veronte Link** must be properly established, and **Veronte Link** needs to be opened.

Note: For more information about this connection, please refer to *Connection - Operation* section of this manual.

These are the options to send the configured variables:

1. Launching **Veronte UDP Telemetry CLI** by double-clicking on the App shortcut or the `.exe` file:

Fig. 8: **Veronte UDP Telemetry CLI shorcut**

   This will send the UDP messages with the following default configuration:

   - **Host IP**: 127.0.0.1
   - **UDP port**: 3000
   - **Frequency**: 10 Hz

   Note: These installation files location will vary depending on the location selected during installation. Note that **Veronte UDP Telemetry CLI Installer** `.exe` is not the **Veronte UDP Telemetry CLI** `.exe` to launch.

   ---

2. Launching **Veronte UDP Telemetry CLI** `.exe` from terminal, where it is possible to specify the parameters of the trasmission using the following command-line options:

   - **- u**: IP address
   - **- p**: UDP port
   - **- f**: Desired frequency of data transmission (*Hz*)

This is an example:

Fig. 9: **Launching from terminal example**

The expected outcome is the following:

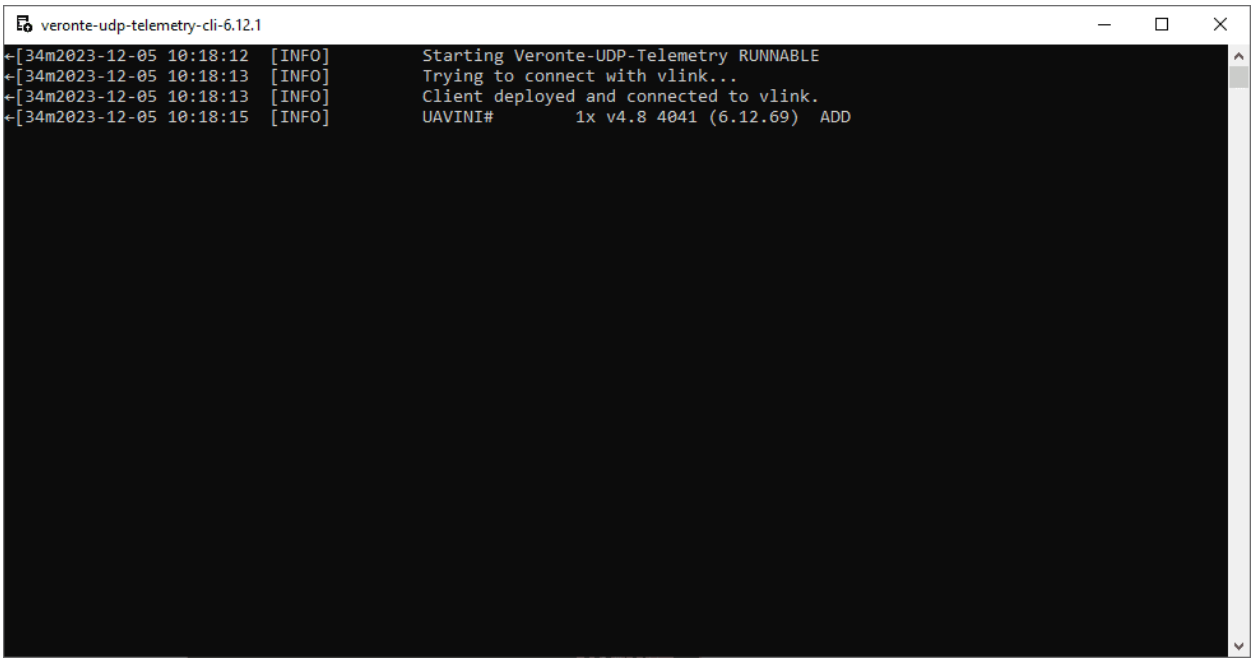


Fig. 10: **Expected outcome**

Note: **Veronte UDP Telemetry CLI** always adds the matcher *0x0A 0xA0* at the beginning of each sent UDP packet before the variable data.

Therefore the received UDP packet will be: *0x0A 0xA0* followed by the consecutive stream of data in the order and byte width configured in *tudp.config*.

If users have any doubts about the UDP packets that are generated, please refer to *Viewing UDP data - Troubleshooting* section of this manual.

CHAPTER

THREE

OPERATION

In order to establish a connection between a Veronte device and a PC with **Veronte Link**, follow the steps:

1. Connect the device to a PC via Serial (USB, RS232 or RS485) or UDP/TCP (Wifi or Ethernet).

Fig. 1: **PC-Veronte device connection**

> **Note:** Connecting the device to the PC is not necessary when communicating via Veronte Cloud. Please, see *Cloud connection* for further information.

2. Open **Veronte Link**, then a similar image to the following should be displayed:

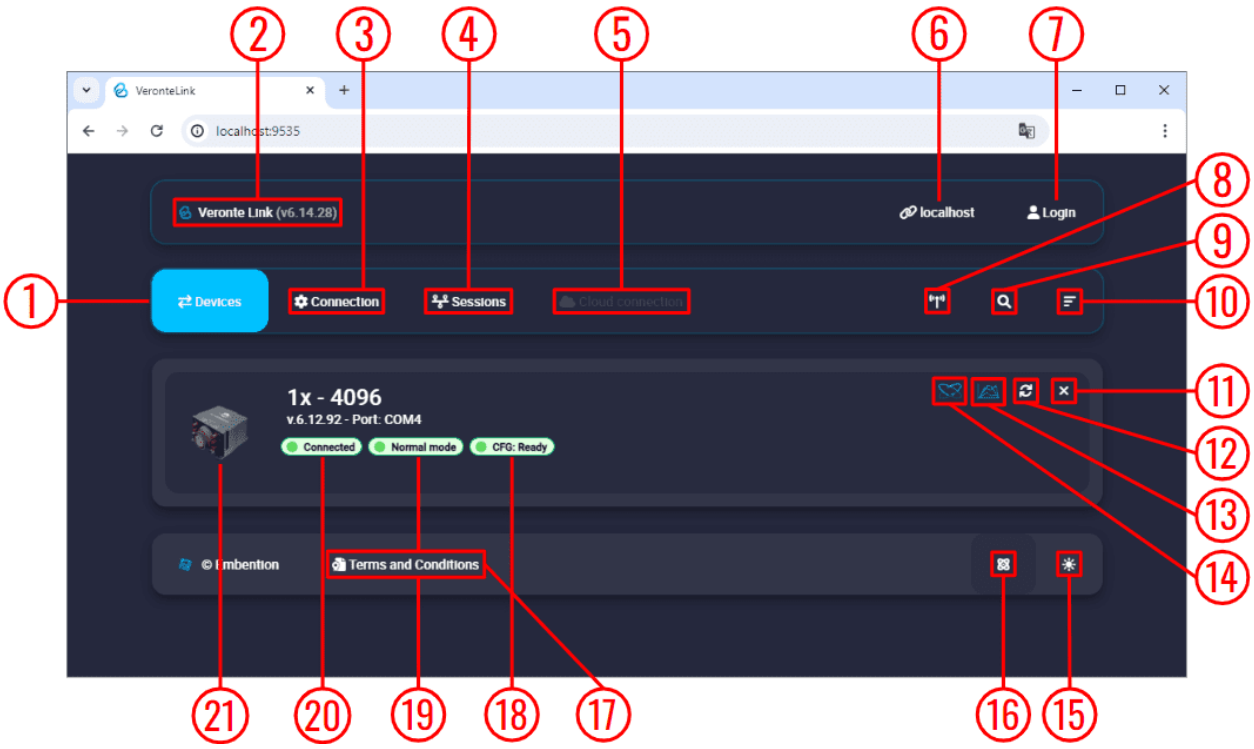


Fig. 2: **Veronte Link interface - Devices menu**

1. **Devices**: This is the currently displayed menu. It shows the devices connected to the PC.
2. **Veronte Link version**: Informs the user about the version of the software being used.
3. **Connection**: This menu allows the user to configure the connection between the PC and a Veronte device. See *Connection* section for more information.
4. **Sessions**: In this menu users can play back recorded logs and flights. See *Sessions* section for more information.
5. **Cloud connection**: This menu allows the user to configure the **internet** connection between the PC and the available Veronte Autopilots 1x. See *Cloud connection* section for more information.

   Note: Only available if the user has logged in.

6. **Host**: Allows connecting to the local IP address or to another desired IP address.
7. **Login**: Enables cloud connection through user logging.
8. **Find all**: Runs a discovery to all devices.
9. **Search from ID**: Searches for a specific device by its ID. Entering the ID **999** will search for all devices.
10. **Sort list**: Click on it to sort the list of devices.
11. **Remove device**: Only works after disconnecting the device.

    Note: Only available if a device is connected or has been connected.

12. **Refresh configurables**: It is recommended to use in case of any connection error.

Note: Only available if a device is connected or has been connected.

13. **Open Veronte FDR**: From here users can access Veronte FDR on the same version of the connected device.
14. **Open Veronte Ops**: From here users can access Veronte Ops on the same version of the connected device.
15. **Dark/light mode**: Switches to light/dark mode, changing the display mode of the interface.
16. **Switch particles**: Particles can be *on* or *off*, changing the application appearance.
17. **Terms and Conditions**: Users can consult the 'End User License Agreement (EULA)' by simply clicking on this button.
18. **Configuration status**: It can be:
    - *CFG: Waiting to read* (only for Veronte Autopilot 1x)
    - *CFG: Reading conf*
    - *CFG: Ready*
    - *CFG: Failed load conf*
    - *CFG: Not Downloaded* (for other products than Veronte Autopilot 1x)
    - *CFG: Not compatible*

    Note: Products are typically operational even if the configuration is not marked as "ready"
19. **Device status**: Can be in *Normal mode*, *Maintenance mode* or *Loaded with errors*.
20. **Connection status**: It can be *Connected* or *Disconnected*.
21. **Veronte device**: Here it is displayed an image of the Veronte device that is connected.

Important: Once **Veronte Link** is executed, an icon will appear in the taskbar and a browser window will open.

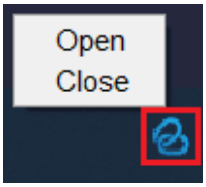


Fig. 3: **Veronte Link icon**

To **close** the application, it is not enough to close the browser window, it is necessary to **right-click** on the icon and select **Close**.

If the browser window is closed, it can be accessed again by pressing the **Open** button.

3.1 Connection

In this menu users must **configure the connection type** of the Veronte device.

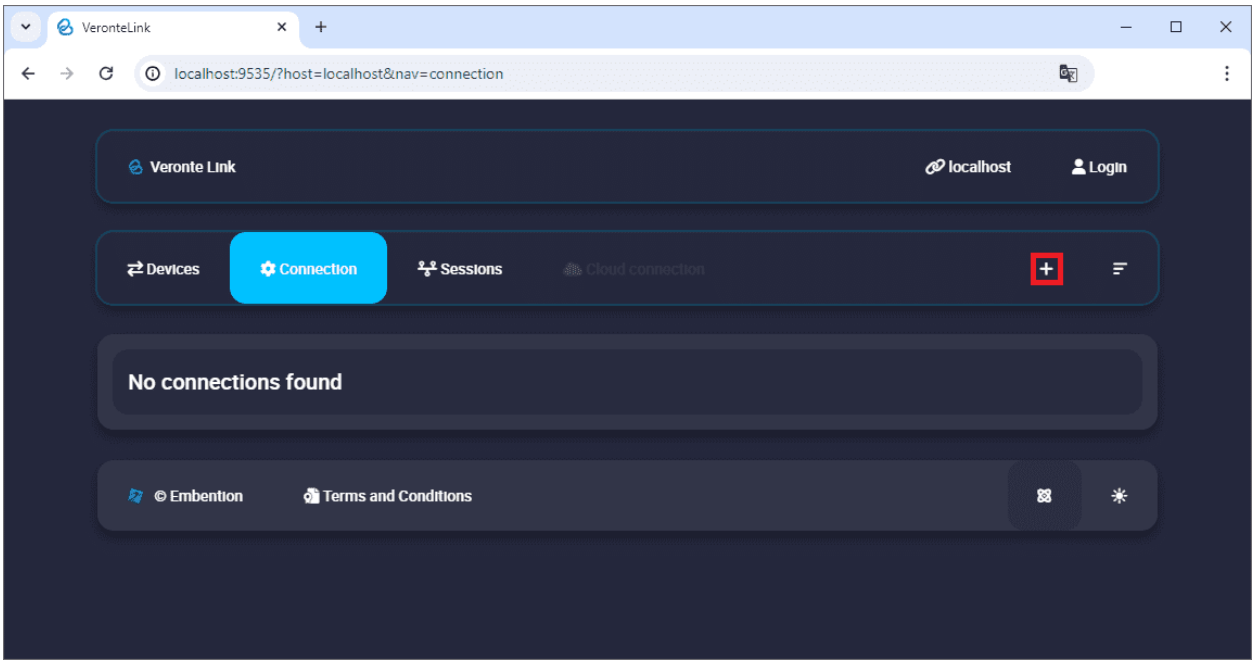


Fig. 4: **Connection menu**

Clicking on the '**+**' icon will display the **configuration** panel. The parameters to be configured depend on the type of connection selected:

> **Warning:** Apart from **Type** and **Port** parameters, it is not recommended to modify the default configuration, as the default parameters should work correctly.

- **Serial**: USB, RS232 or RS485 connections.

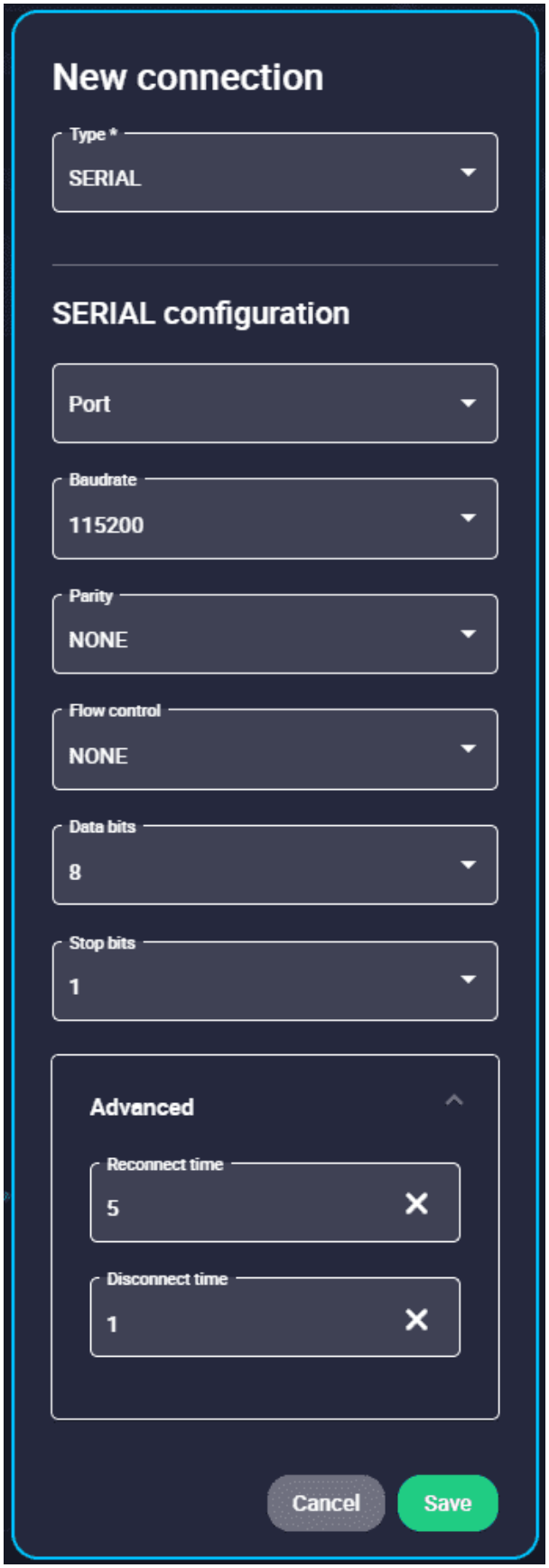


Fig. 5: **Serial connection configuration**

- **Port**: Select the port of the computer to which the device is connected. It does not have to be the same as the one in the example image (*Veronte Link interface* image).

  More information about the port where the device is connected is explained in *Serial connection - Integration examples* section of the present manual.

- **Baudrate**: This field specifies how fast data is sent over a serial line.

- **Parity**: Is a method of detecting errors in transmission.

  When parity is used with a serial port, an extra data bit is sent with each data character, arranged so that the number of 1 bits in each character, including the parity bit.

  The available options are **EVEN**, **MARK**, **ODD**, **SPACE** and **NONE**.

- **Flow control**: RTS/CTS and XON/XOFF control can be configured if needed.

- **Data bits**: Defines the number of bits in the message. It can be configured from **5 to 8** bits.

- **Stop bits**: Number of stop bits sent at the end of every character. Can be **1**, **1.5** or **2**.

- **Advanced**:

  * **Reconnect time**: The time to consider a device reconnected. Default is set to 5 seconds.

  * **Disconnect time**: Time to consider a device disconnected is defined here. 1 second is configured by default.

Note: In case of not getting the device connected, make sure that the PC acquires a *communication port*.

- **UDP**: Ethernet or Wifi connections.

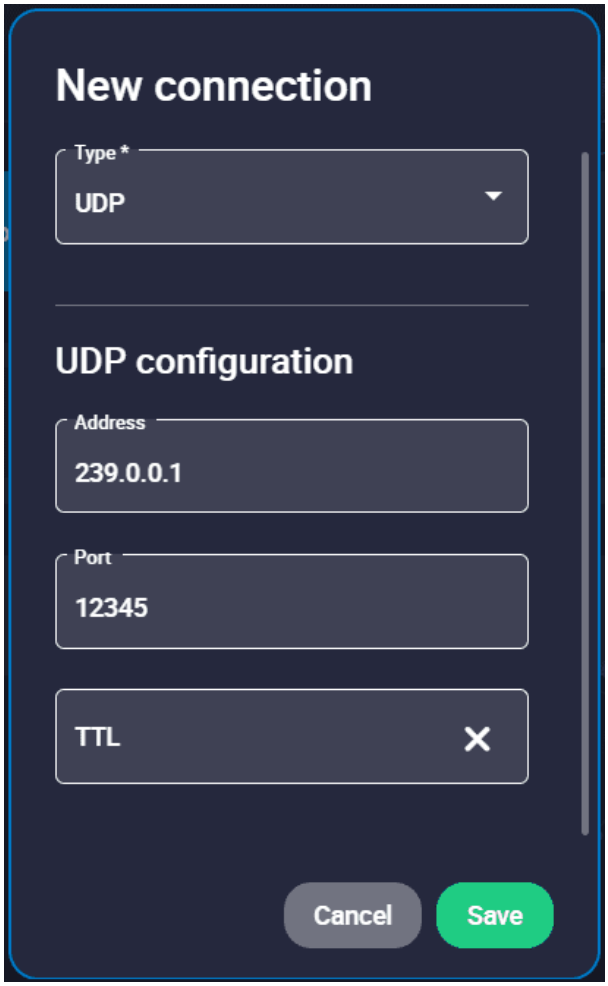


Fig. 6: **UDP connection configuration**

Important: Consider the maximum packet size supported by the Veronte Communication Protocol (VCP) when using serial data converters.

- **Address**: IP address, normally set to *239.0.0.1* (for broadcast) or *127.0.0.1* (for local).
- **Port**: IP Port must be set.
- **TTL**: Time To Live, it is the maximum amount of time or 'hops' that a UDP packet can exist inside a network before being discarded by a router.

  A default value should automatically be set.

Note: How to establish a connection via UDP is detailed in the *UDP connection - Integration examples* section of the present manual.

- **Planet**: Satellital connections, it requires internet connection.

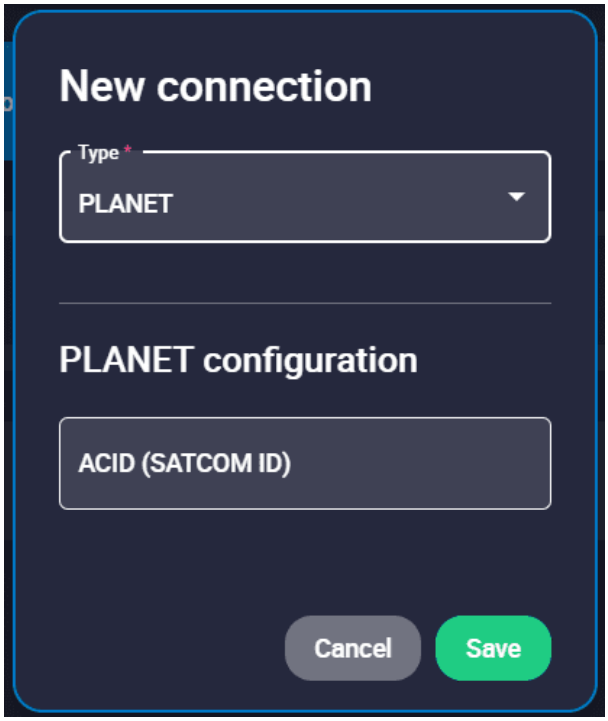


Fig. 7: **Planet connection configuration**

  - Satcom ID must be set.

- **TCP-SERVER**: Ethernet or Wifi connections.

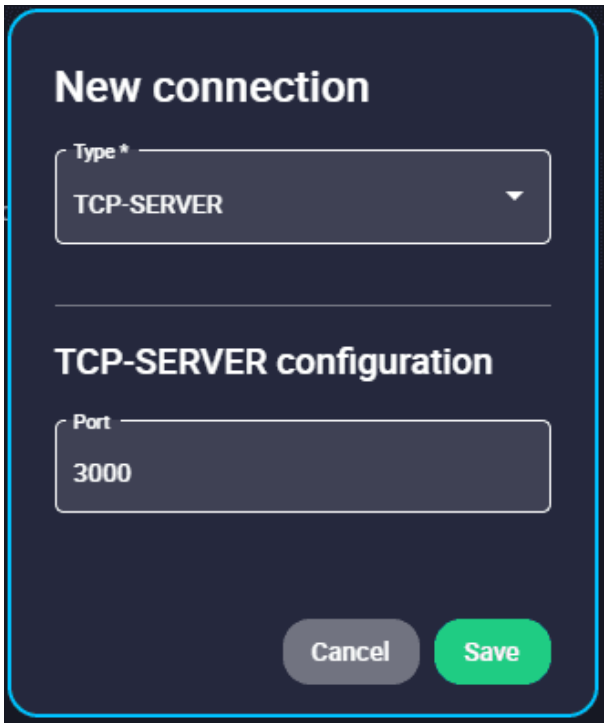


Fig. 8: **TCP-SERVER connection configuration**

  - **Port**: Set the TCP port from which the devices will get the information provided by Veronte Link.

Note: How to establish a TCP-SERVER connection is detailed in the *TCP-SERVER connection - Integration examples* section of this manual.

- **TCP-CLIENT**: Ethernet or Wifi connections.

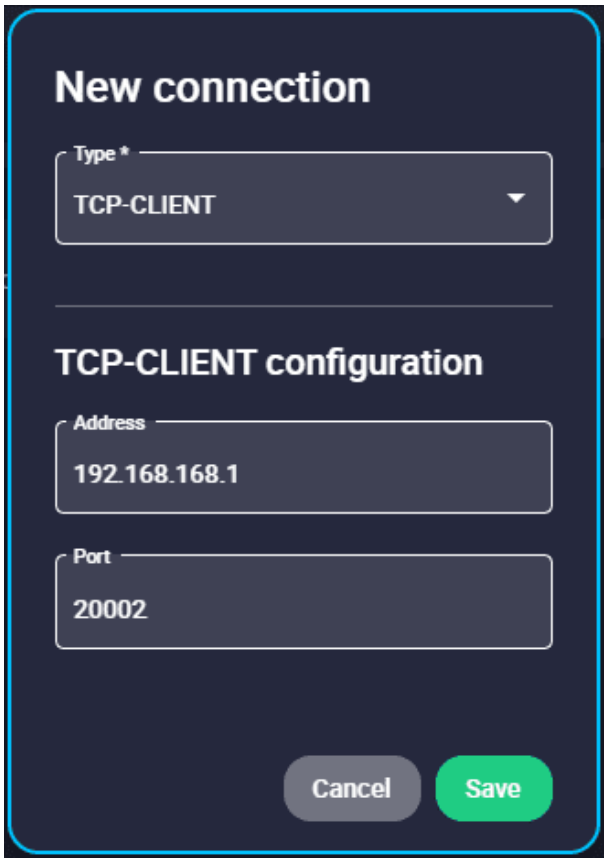


Fig. 9: **TCP-SERVER connection configuration**

  - **Address**: Enter the address of the device from which Veronte Link has to obtain the information.
  - **Port**: Enter the TCP port from which the information is obtained.

Note: How to establish a TCP-CLIENT connection is detailed in the *TCP-CLIENT connection - Integration examples* section of this manual.

Finally, click on **Save**.

3.2 Sessions

Sessions tab displays all finished device sessions.

Note: Sessions that are currently being recorded will not be displayed.

The following image and list describe each functionality.

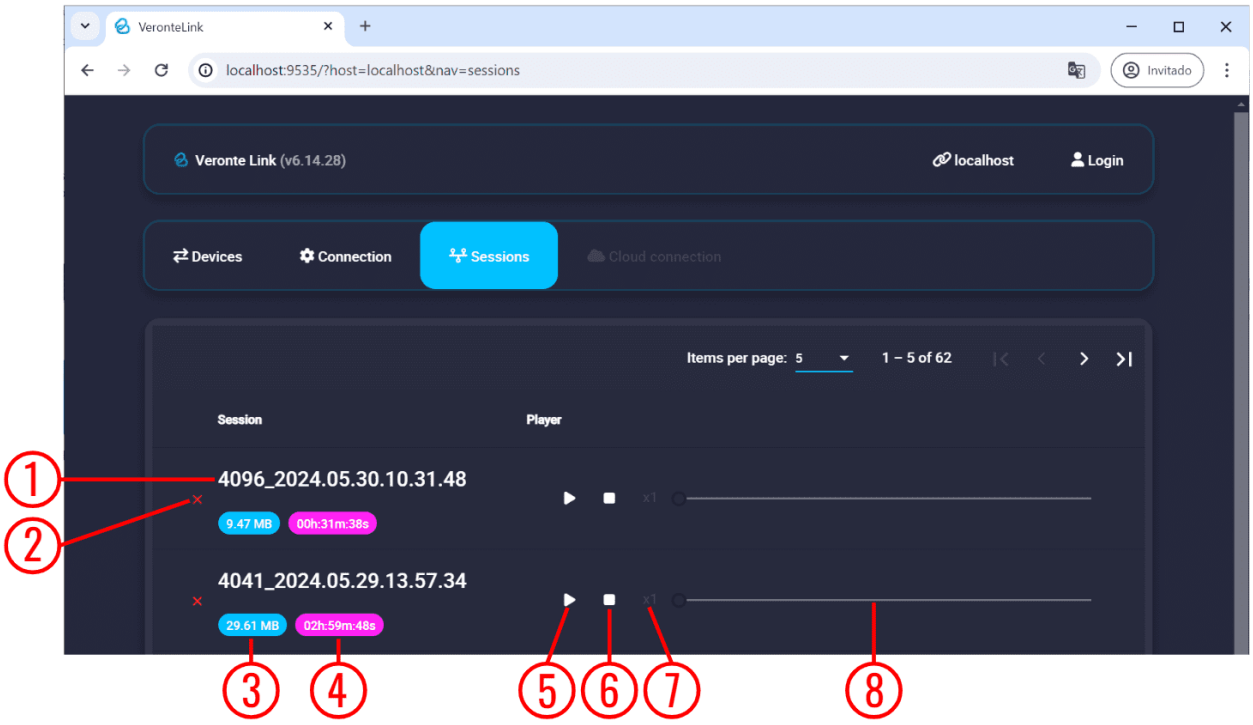


Fig. 10: **Sessions menu**

1. **Session name**: It is made with recording time (date and hour).
2. **Delete session**. If the user wishes to delete more than 1 session at a time, it is possible to delete them from the **Veronte Link sessions folder** located in the following path:

   `C:\Users\USER NAME\AppData\Roaming\VeronteLink\sessions`
3. **Files weight**.
4. **Duration**.
5. **Play/Pause**: Play button creates a **virtual device** in the "*session port*" similar to the following figure:

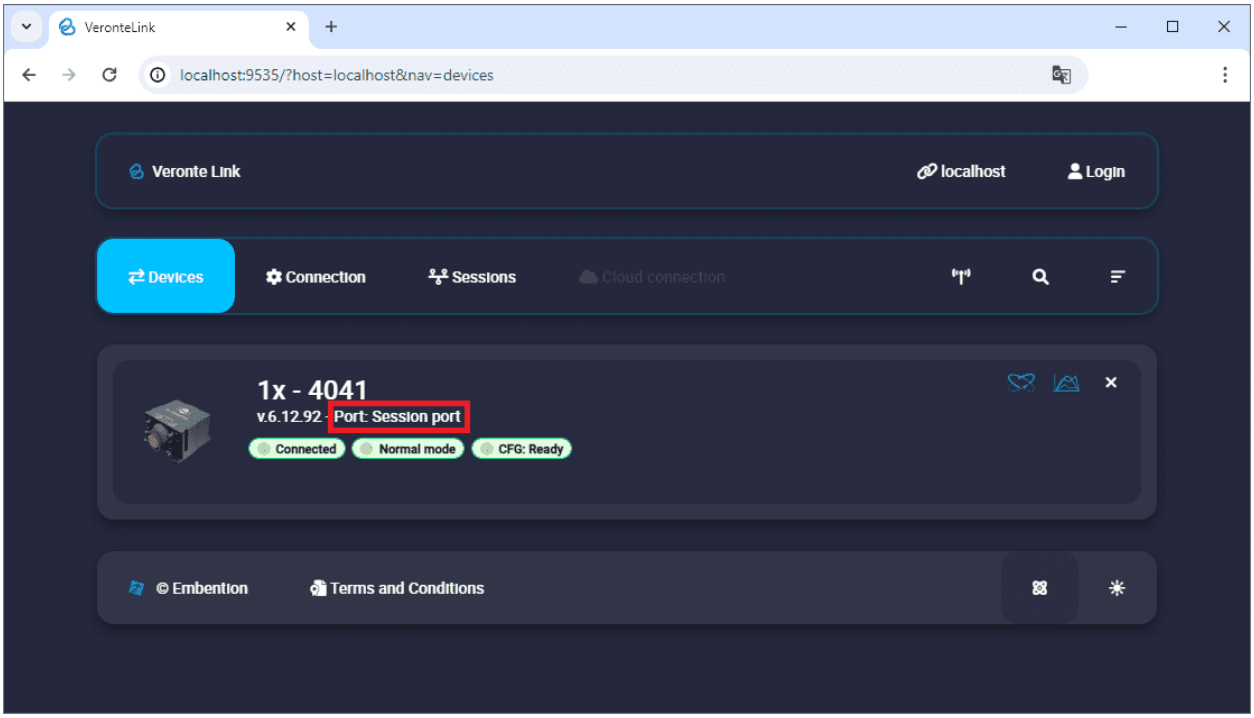


Fig. 11: **Virtual device**

It starts a simulation replaying everything that happened during the session recording. It will recreate all the ocurred events with detail and **Veronte Ops** will display the corresponding data and trajectories; read the Veronte Ops user manual for more information.

Note: In addition, when the virtual device is in a **ready** state, users can **open the 1x PDI Builder software and download the configuration (PDI files)**.

---

6. **Stop**: It stops playing the session. It **does not delete the session**.
7. **Speed**: Playing speed can be selected as x0.5, x1, x2, x4 and x8.

Note: This button is only available when reproducing a session.

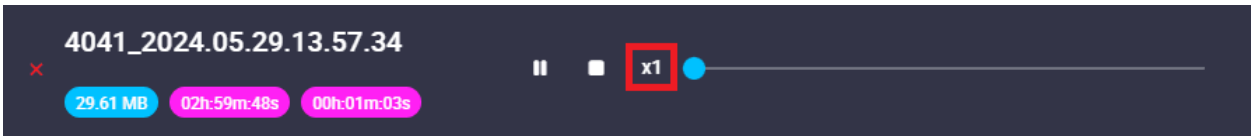


Fig. 12: **Speed button enabled**

---

8. **Display bar**: Click and drag to replay any moment.

3.3 Cloud connection

Cloud connection tab allows the user to connect to a Veronte Autopilot 1x through **LTE network**. This functionality is enabled thanks to the **HSPA+** module (internal or external) embedded in Veronte autopilots.

Note: To activate the internal card or Veronte Cloud data traffic through internet, please contact sales@embention.com. Remember that there is **no internet connection** when **HSPA+ module** is deactivated.

To configure this type of connection, these steps must be followed:

1. **Login**: After clicking the *Login* button, users must introduce their associated username and password.

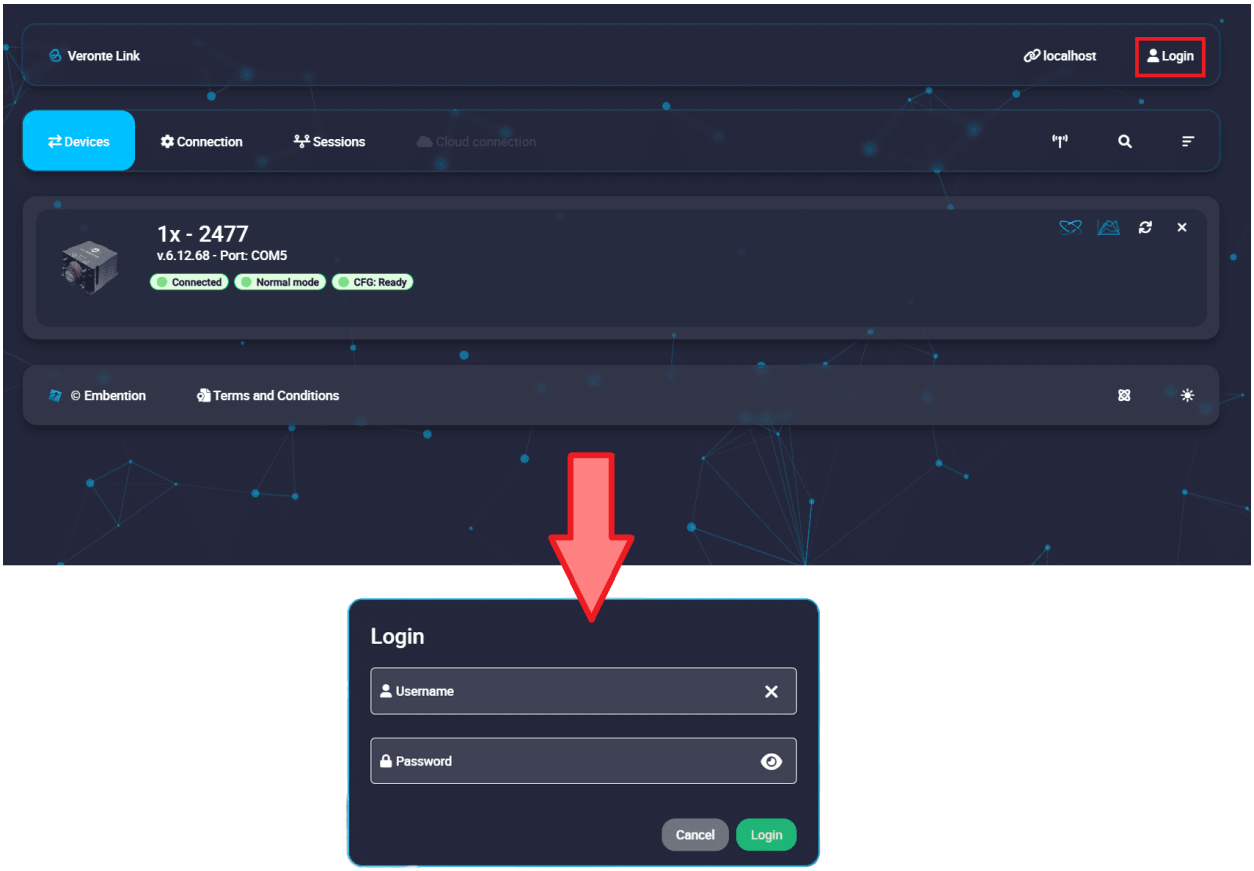


Fig. 13: **Cloud Connection: Login**

Note: Login credentials are automatically assigned. In case they have not been provided to you, please contact the support team by creating a ticket in the customer's Joint Collaboration Framework; for more information, see Tickets section of the JCF manual or contact sales@embention.com.

2. **Open Cloud connection tab**. Veronte Autopilots 1x linked to user's account should be displayed.

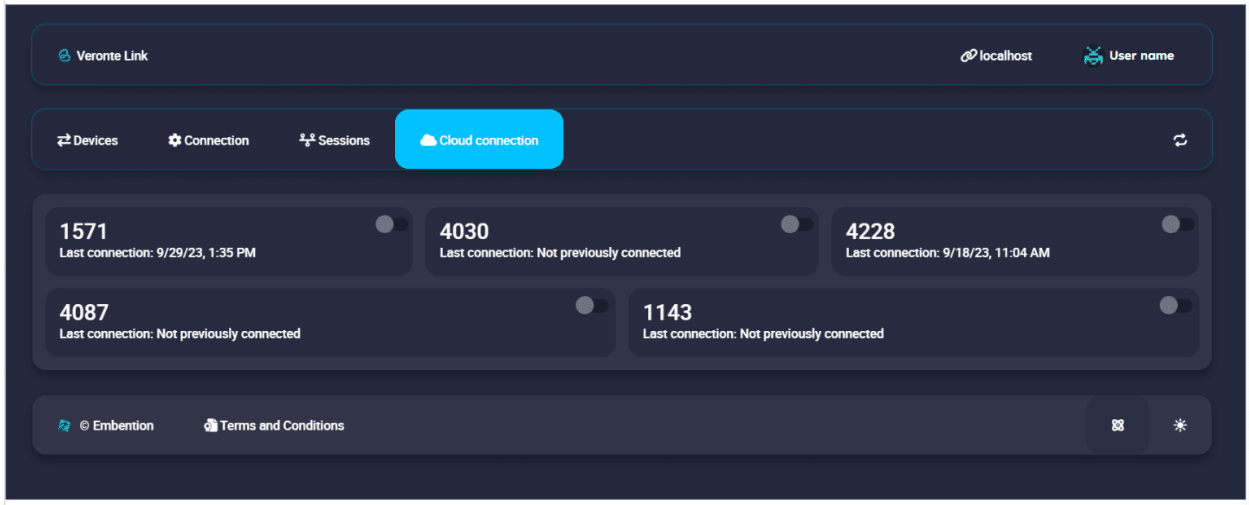


Fig. 14: **Cloud Connection: Available devices**

The following information is displayed for each autopilot:

- **ID**: Identification number of the autopilot (Serial Number).
- **Last connection**: Indicates the date on which the last connection to that device was established.

3. **Activate the connection** with the desired Autopilot 1x by turning on **the toggle button** displayed next to it.

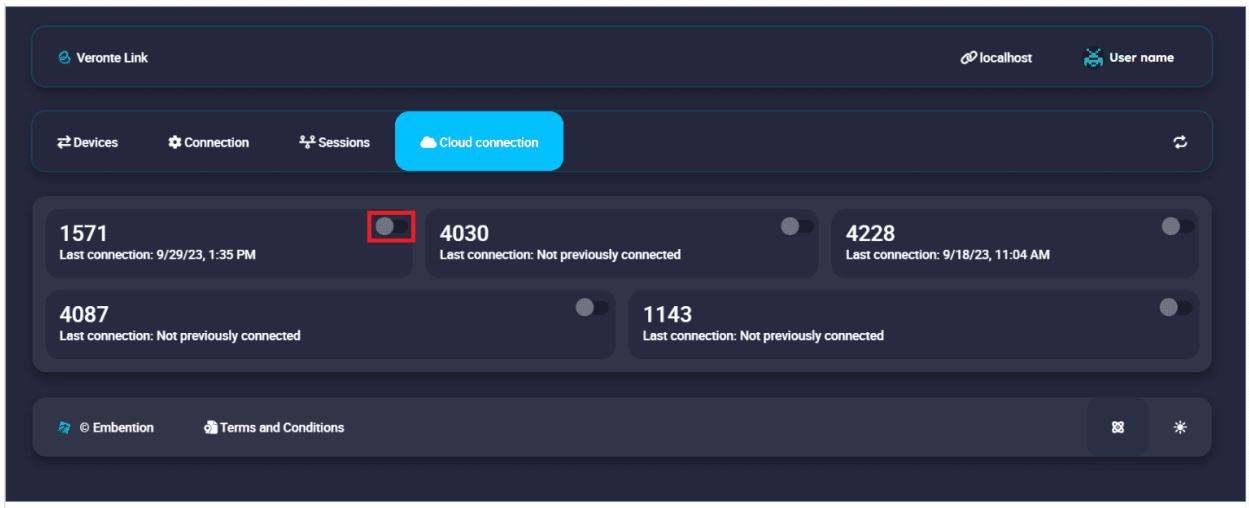


Fig. 15: **Cloud Connection: Connect to an Autopilot 1x**

Note: Since Cloud connections are based on **LTE communication**, this connection may not be immediate. The selected autopilot will only be displayed in the '*Devices*' tab when it is succesfully connected.

---

4. At this point, **Veronte Link** must have established the connection with the selected Autopilot 1x. Consequently, the autopilot must be displayed in the *Devices* tab.

Note: Since Cloud connections are based on **LTE communication**, **connection may be lost** even when the

toggle button is on. In this case, the autopilot will disappear from the '*Devices*' tab, appearing again when the connection is retrieved.

5. **Log out**: Click on the username to enable the log out button, and then press it.

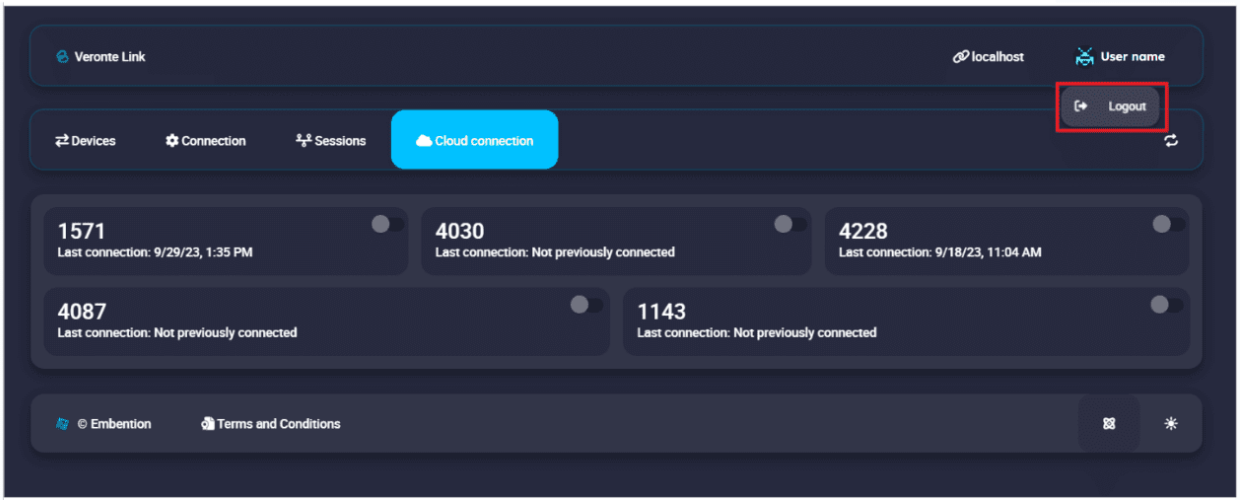


Fig. 16: **Cloud Connection: Log out**

CHAPTER

FOUR

INTEGRATION EXAMPLES

4.1 Serial connection

As the com port configuration is common to all devices, the following steps are applied to MC24 and MC110 controllers as an example.

1. Once **Veronte Link** is installed, the first step that must be done is to set the connection that your MC unit is currently using. By default, every MC is capable to comunicate through USB, RS232 and RS485 so any of these can be used (properly adapted to USB/serial).

   First, click on "**+**":

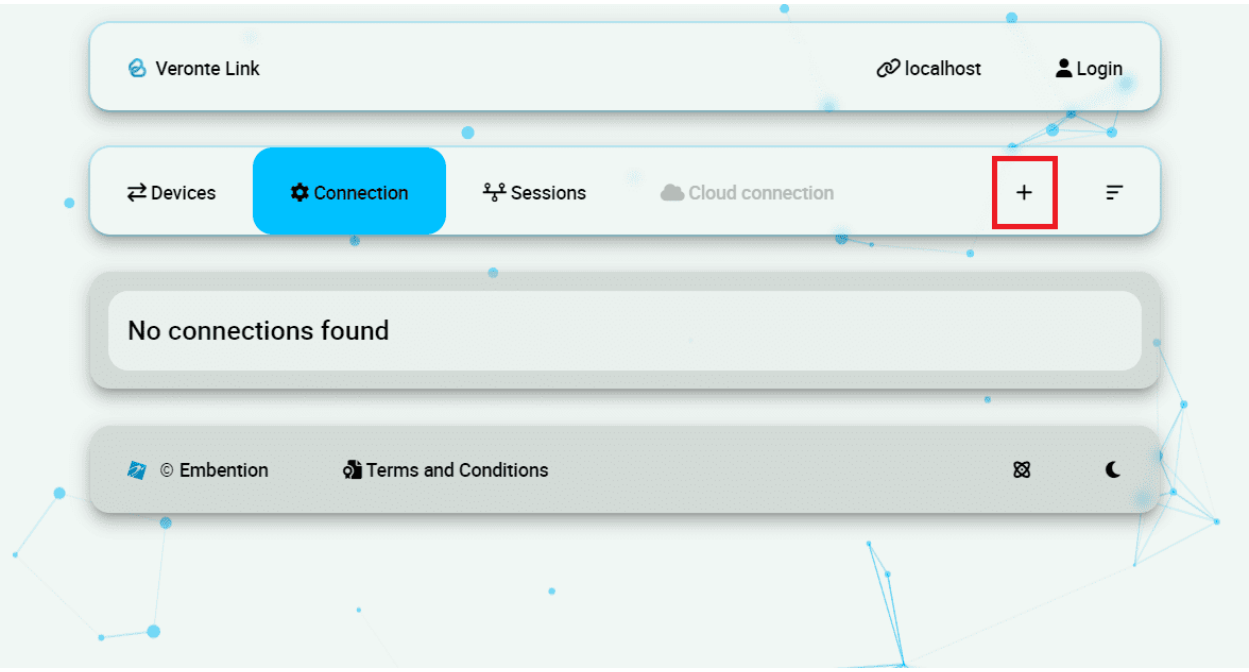


Fig. 1: **Add new connection**

2. Besides, it is required to find out which port is employing the MC unit. Windows allows to do that with the **Device Manager** from the **Control Panel**.

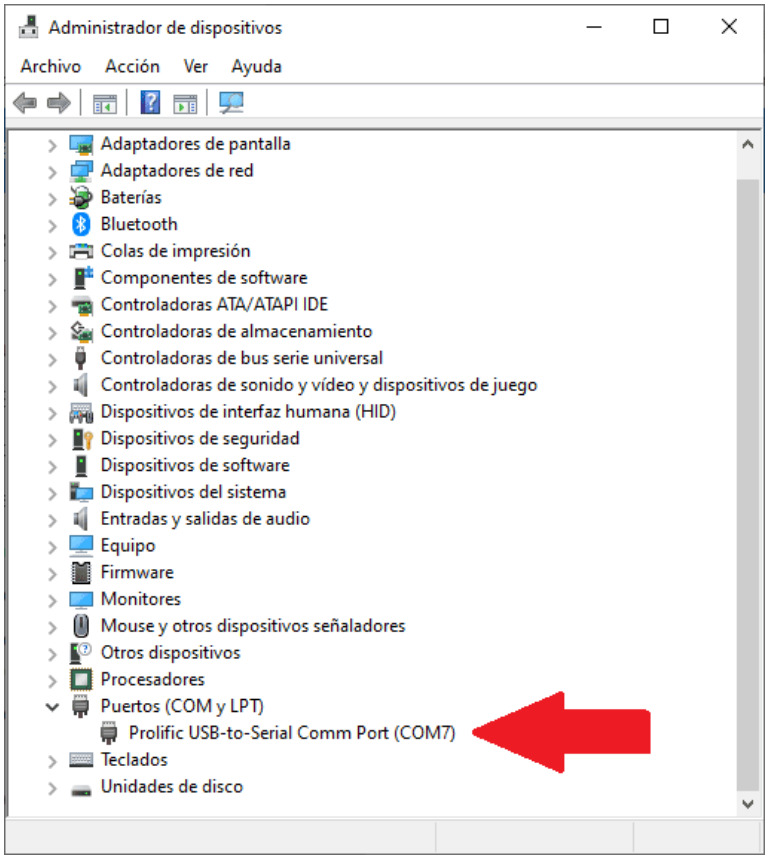


Fig. 2: **Windows Device Manager**

3. Select your COM settings by entering the **Comm Port** previously found. Normally, the other default parameters should not be changed.

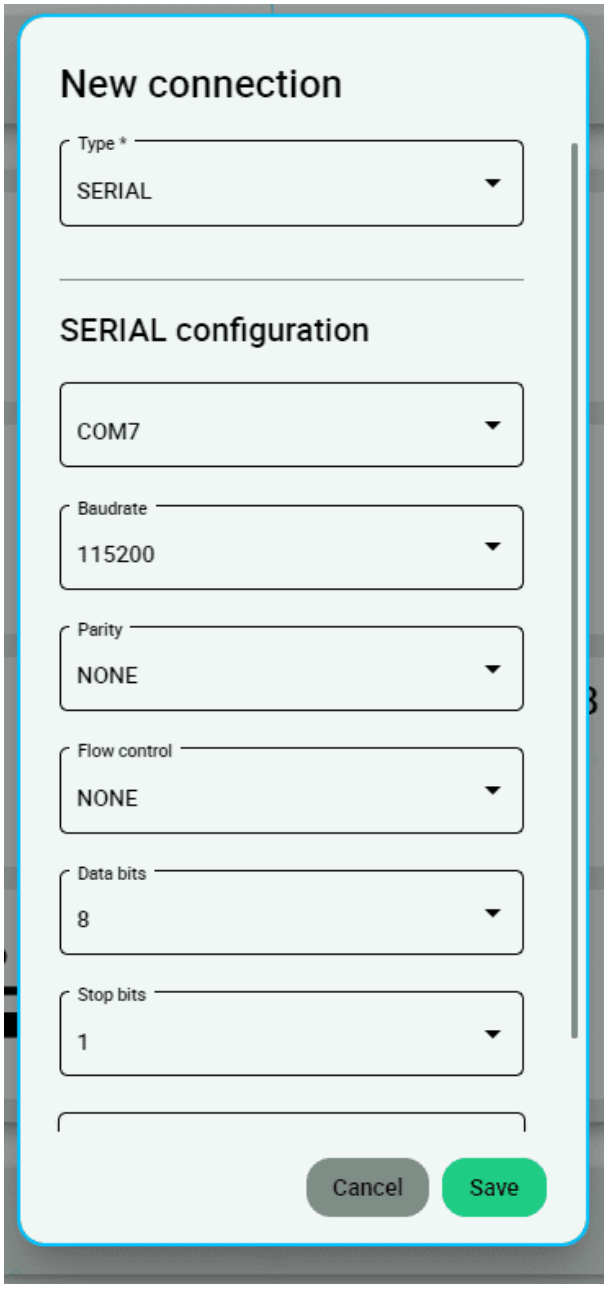


Fig. 3: **New connection configuration**

4. If the selected port is correct and everything went well, a new MC will be displayed in the devices list. However, the *device status* will remain as **CFG: Waiting to read**.

   The user is ready now to start configuring the motor controller using **MC PDI Builder**.

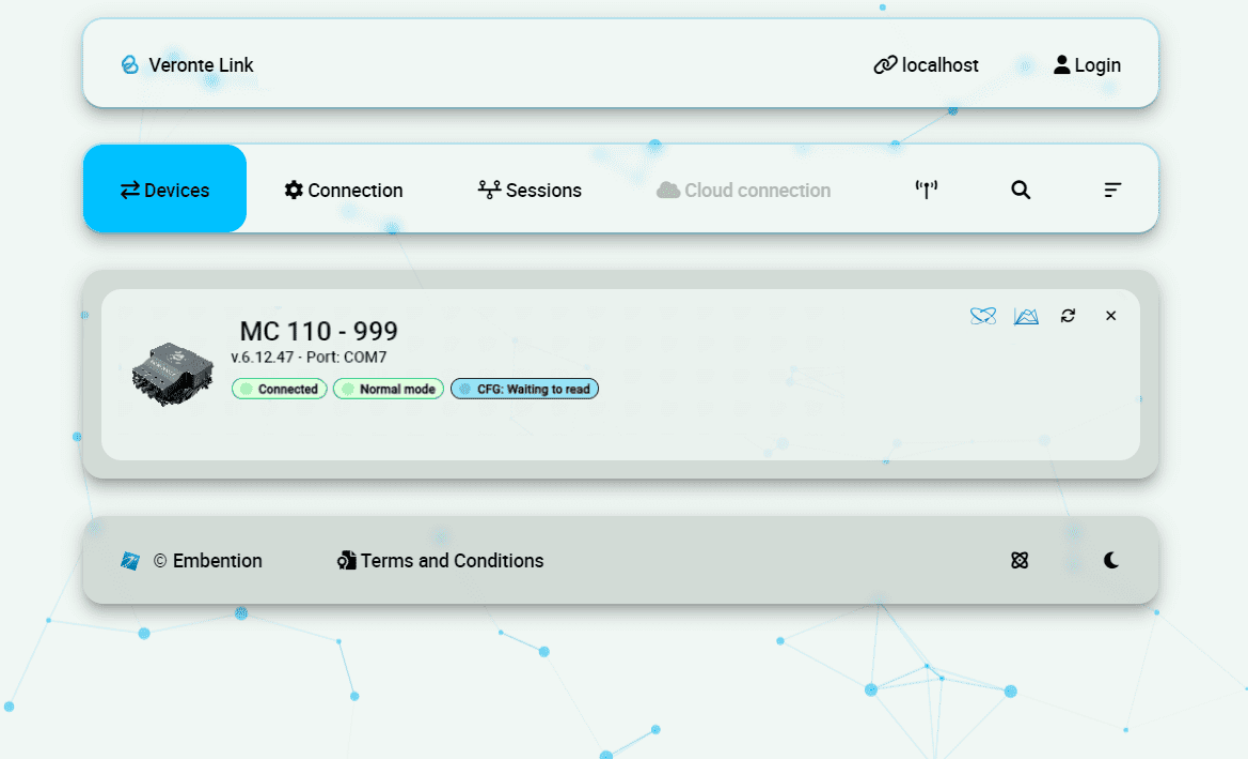


Fig. 4: **MC unit correctly connected**

More Veronte devices (*MC units, Veronte Autopilots, etc.*) could be added following these instructions.

Note: In case of connecting a Veronte Autopilot 1x, after a few seconds, the *device status* should replace **CFG: Waiting to read** by **CFG: Ready**, since **only Autopilot 1x is able to change or load configuration in normal mode**.

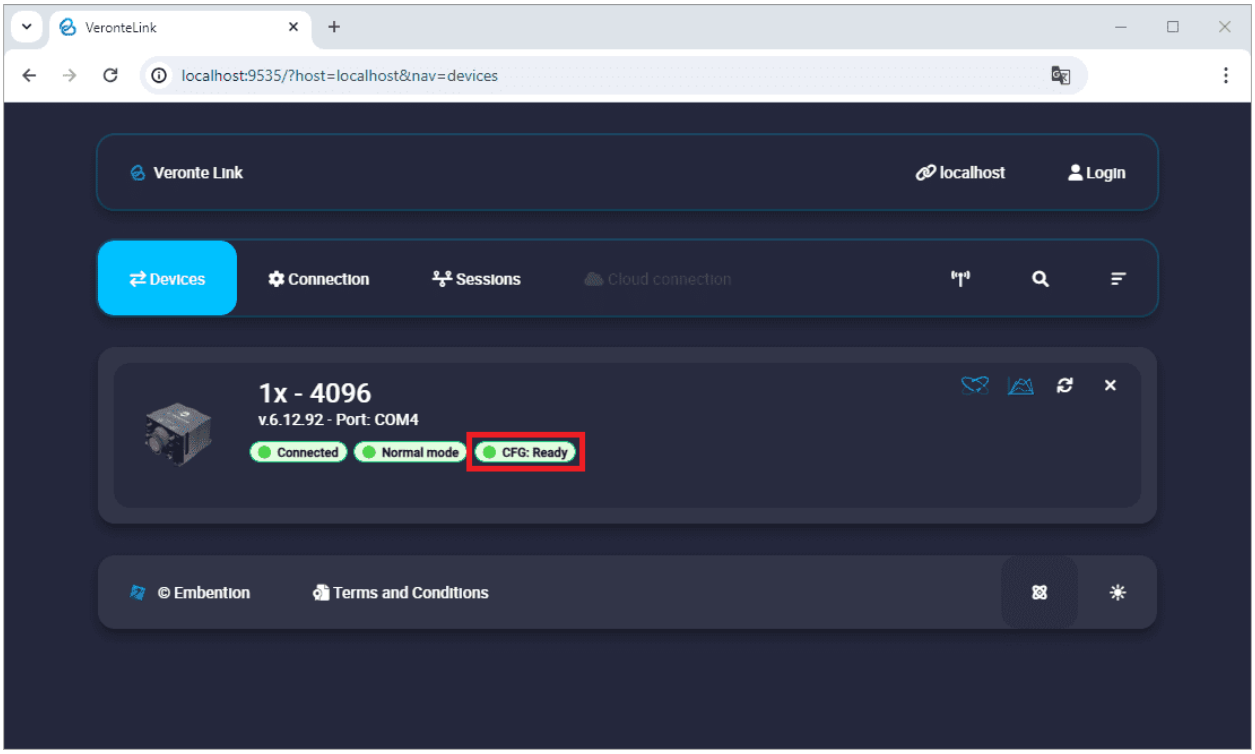


Fig. 5: **Veronte Autopilot 1x connected and ready**

For other Veronte devices than 1x, **CFG: Not Downloaded** is equivalent to **CFG: Waiting to read**. Hence, **CFG: Ready** should replace the status **CFG: Not Downloaded**.

4.2 UDP connection

Wi-Fi/Ethernet configuration

The following steps are applied to a **PCS** unit as an example.

Important: If connecting through **Ethernet, step 1 does not apply**.

1. The first step is to look under *available networks* for the PCS unit and connect to it.
2. Once the connection is made, enter **Veronte Link** and configure the UDP connection in the **Connection menu**. First, click on "**+**":

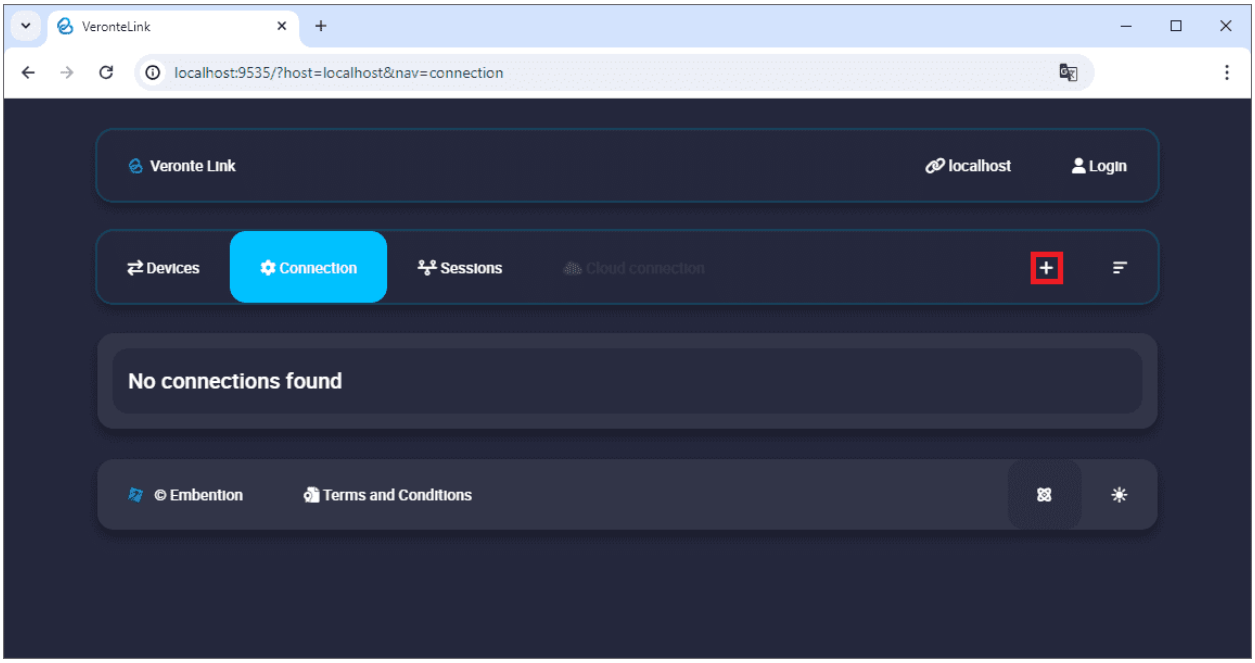


Fig. 6: **Add new connection**

3. Then, the configurable parameters must be entered.

Fig. 7: **New UDP connection configuration**

Important: This address and port are configured for this PCS unit, they do not have to be the same for another device.

4. Finally, if the configured connection is correct and everything went well, a new PCS will appear in the device list and the *device status* will change to **CFG: Ready**.

   The user is ready now to start configuring the PCS using **1x PDI Builder**.

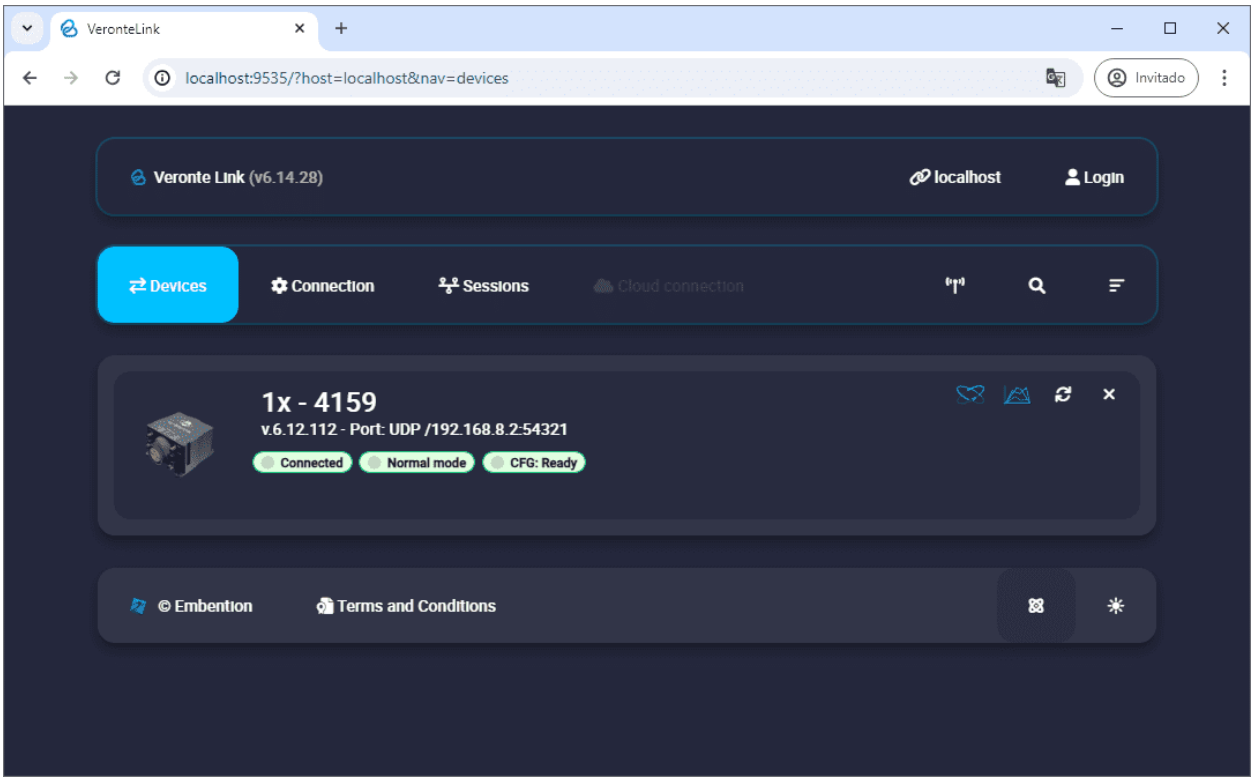


Fig. 8: **PCS unit correctly connected**

Note: The image of a Veronte Autopilot 1x is displayed and not a PCS as the device that is actually connected is the Autopilot 1x inside the PCS.

4.3 TCP-SERVER connection

Ethernet configuration

The following steps detail how to connect Veronte Link to an Autopilot 1x via a TCP connection to a **Microhard** radio.

Note: In this connection, the **radio** acts as "**Client**" and **Veronte Link** as "**Server**".

1. Configure, in the Microhard WebUI, the radio as "**TCP Client**" and enter the following parameters:
   - **Remote Server IP Address**: IP address of the PC.
   - **Remote Server port**: TCP port to which the radio has to connect. It must be the same as the one configured in Veronte Link.

   For more information on the radio configuration, users can refer to the Microhard radio configuration - Integration examples section of the **1x Hardware Manual** or directly to the Microhard radio documentation.
2. Connect **Veronte Autopilot 1x** to the Microhard radio via **RS232** as detailed in the Microhard pDDL900-ENC external - Integration examples section of the **1x Hardware Manual**.

3. Once the configuration and connection is done, open **Veronte Link** and configure the **TCP-SERVER** connection in the **Connection menu**.

   First, click on "**+**":

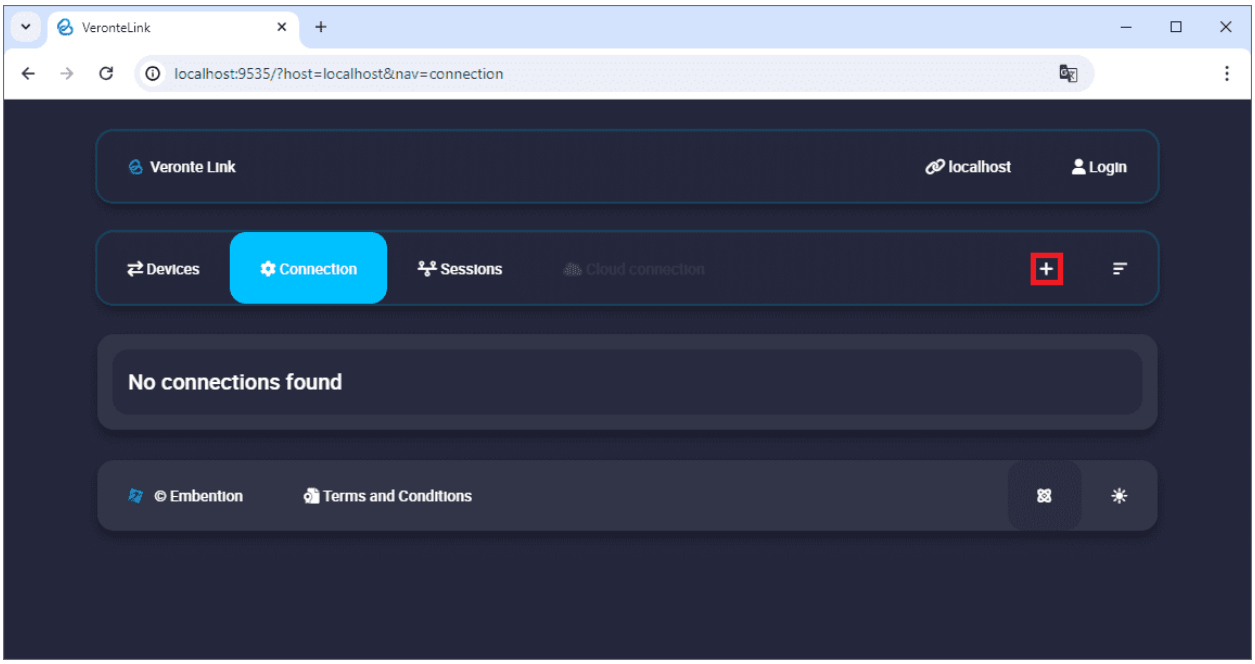


Fig. 9: **Add new connection**

4. Then, the configurable parameters must be entered.

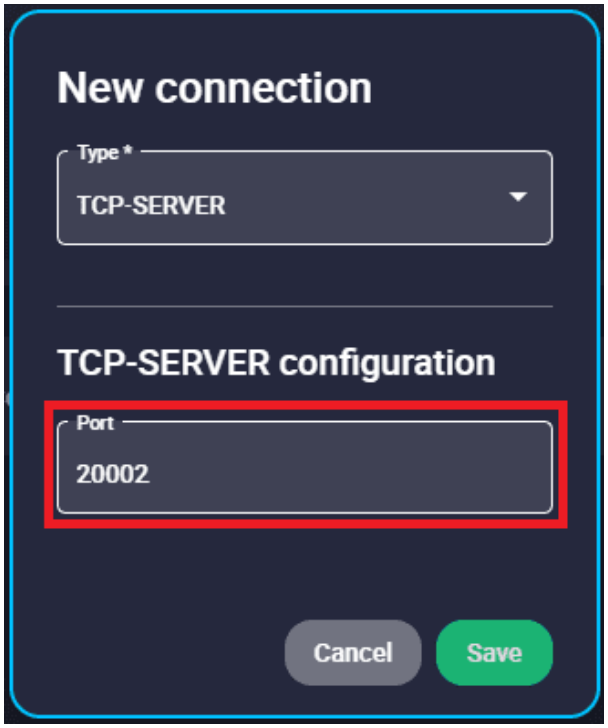


Fig. 10: **New TCP-SERVER connection configuration**

- **Port**: Enter a TCP port to which the radio will be connected, the same as the one previously configured as "Remote Server port" in the radio configuration.

5. Finally, if the configured connection is correct and everything went well, a new Autopilot 1x will appear in the Devices list. It should look like this:

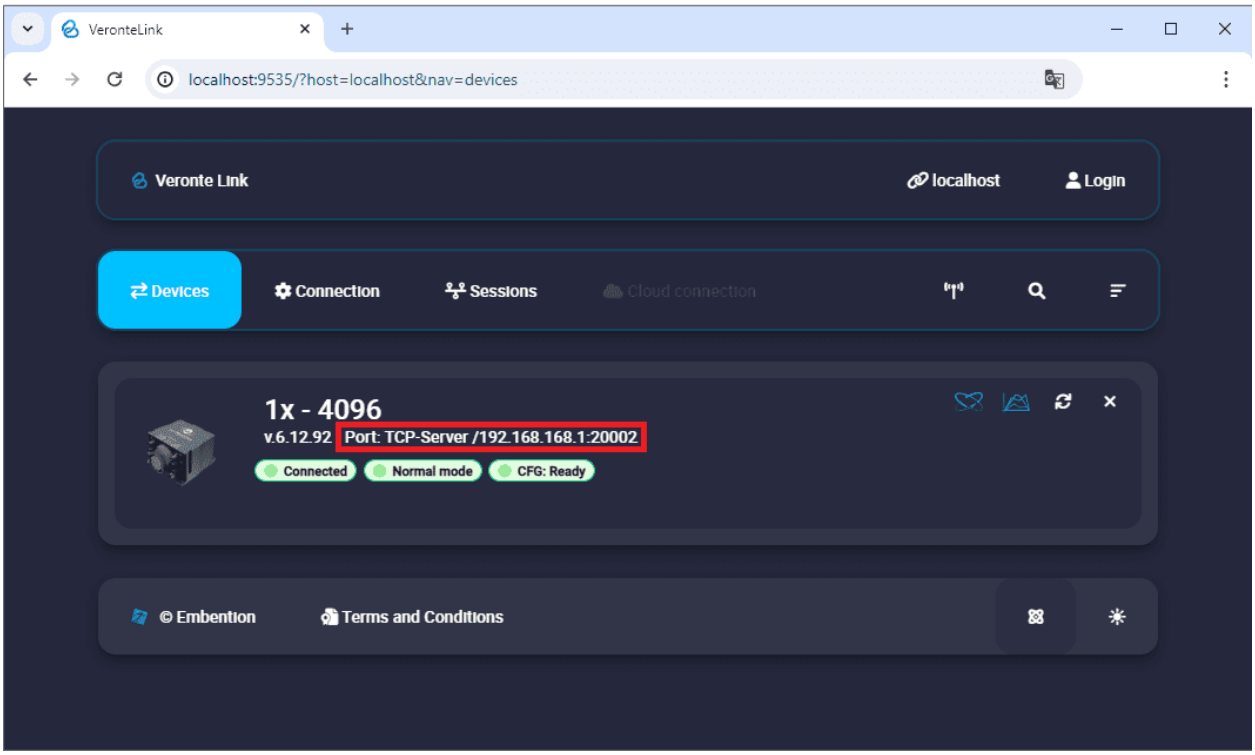


Fig. 11: **1x unit correctly connected**

4.4 TCP-CLIENT connection

Ethernet configuration

The following steps detail how to connect Veronte Link to an Autopilot 1x via a TCP connection to a **Microhard** radio.

Note: In this connection, the **radio** acts as "**Server**" and **Veronte Link** as "**Client**".

1. Configure, in the Microhard WebUI, the radio as "**TCP Server**" and enter a "**Local Listening Port**" to which Veronte Link will have to connect (usually the default one is used).

   For more information on the radio configuration, users can refer to the Microhard radio configuration - Integration examples section of the **1x Hardware Manual** or directly to the Microhard radio documentation.

2. Connect **Veronte Autopilot 1x** to the Microhard radio via **RS232** as detailed in the Microhard pDDL900-ENC external - Integration examples section of the **1x Hardware Manual**.

3. Once the configuration and connection is done, open **Veronte Link** and configure the **TCP-CLIENT** connection in the **Connection menu**.

   First, click on "**+**":

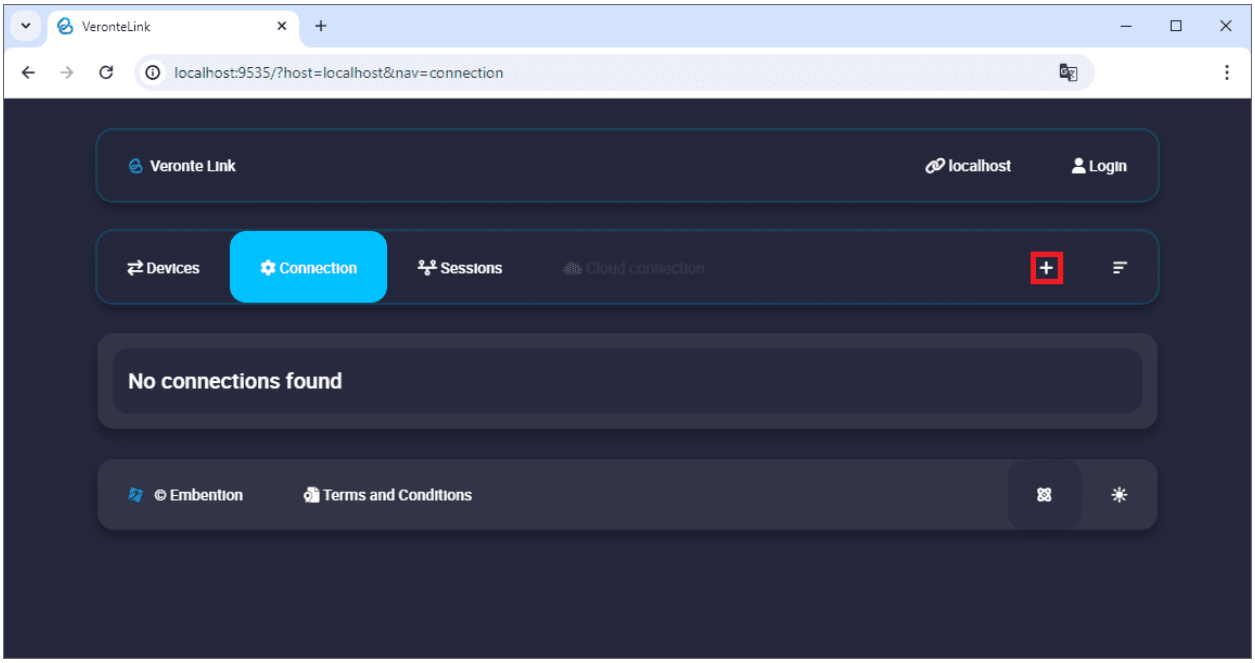


Fig. 12: **Add new connection**

4. Then, the configurable parameters must be entered.

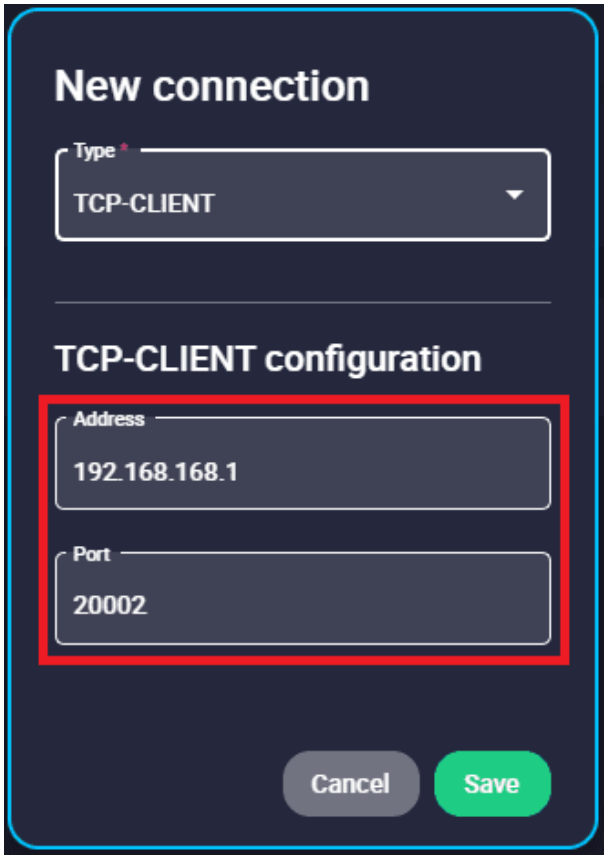


Fig. 13: **New TCP-CLIENT connection configuration**

- **Address**: IP address of the radio.
- **Port**: Enter as TCP port the "Local Listening Port" previously set in the radio configuration.

Important: This address and port are configured for this radio unit, they do not have to be the same for another device.

---

5. Finally, if the configured connection is correct and everything went well, a new Autopilot 1x will appear in the Devices list. It should look something like this:

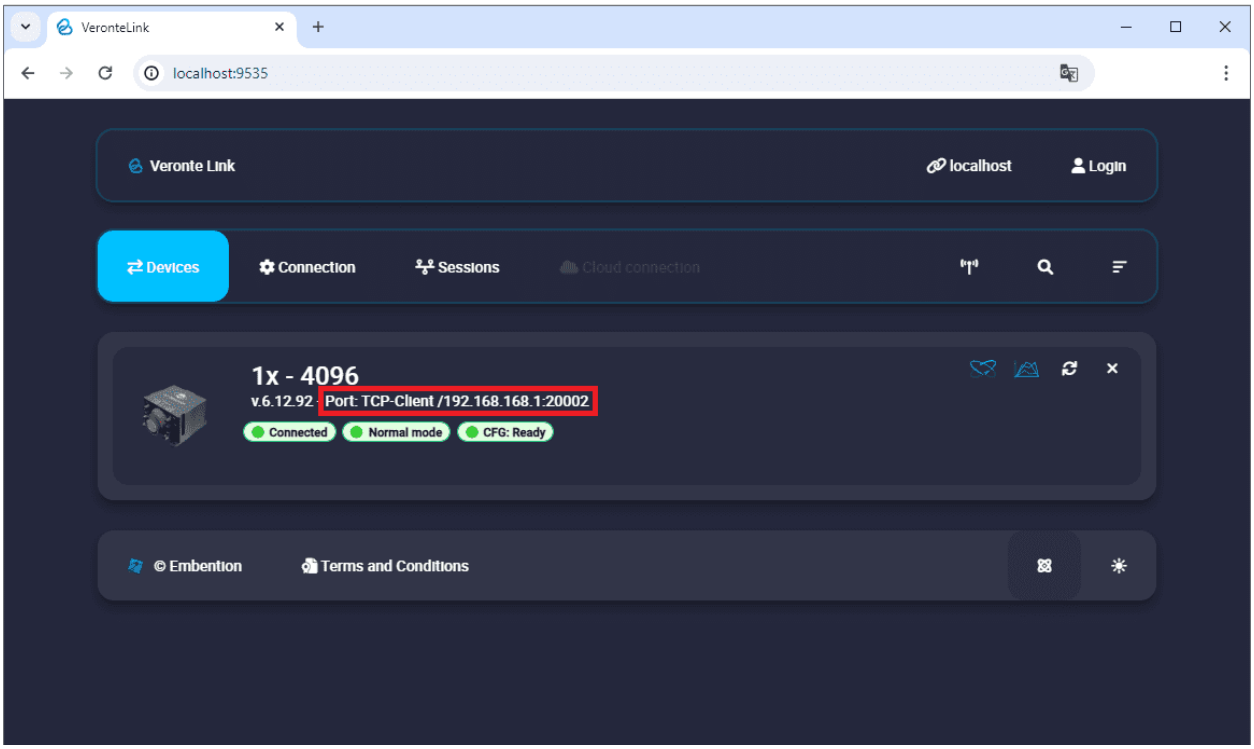


Fig. 14: **1x unit correctly connected**

CHAPTER

FIVE

TROUBLESHOOTING

In case of any software error, it is possible to extract and analyze files from session folder.

> **Warning:** Do not modify or delete manually any **Veronte Link** file. Copy them to a different path to send or analyze.

Veronte Link files are placed on the following paths:

- `C:\Users\USER NAME\AppData\Roaming\VeronteLink\configurables` Device configurations.
- `C:\Users\USER NAME\AppData\Roaming\VeronteLink\sessions` Session files, it includes flights information.
- `C:\Users\USER NAME\AppData\Roaming\VeronteLink\tracelogs` Event logs, it includes flights information.
- `C:\Users\USER NAME\AppData\Roaming\VeronteLink` ⇒ `cfg.son` **Veronte Link** connections configuration file. **If deleted, the configuration will be lost**.
- `C:\Users\USER NAME\AppData\Roaming\VeronteLink` ⇒ `vlink.lock` Internal file that only appears if any instance of **Veronte Link** is open. **If deleted, there will be instability in the system**.

5.1 Comm Port error in Windows Device Manager

If the following Windows Comm Port error occurs:

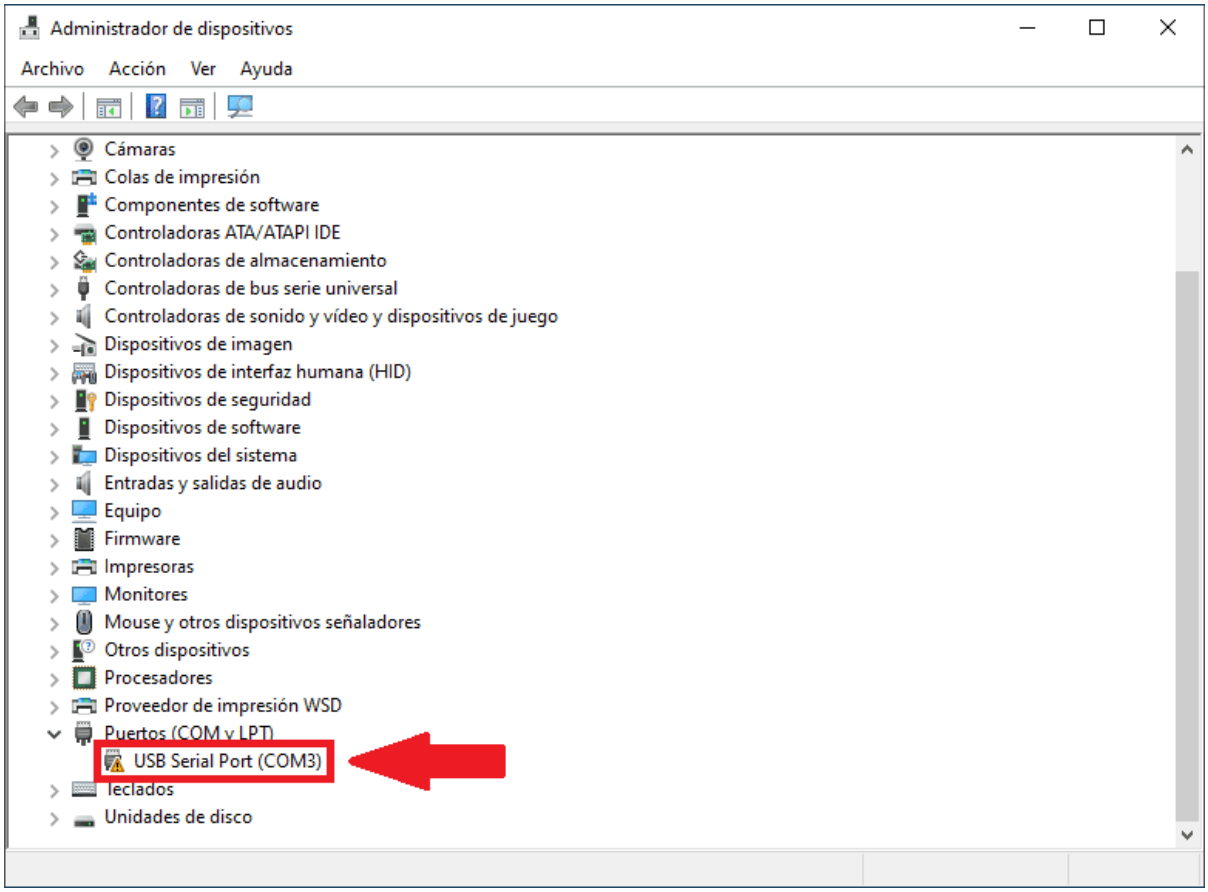


Fig. 1: **Windows Device Manager - Comm Port error**

Users must extend the disconnection time to 5 seconds to fix it. To do this:

1. Go to the **Connection** menu → click on the ⚙ icon to open the COM configuration.
2. Open the **Advanced** parameters drop down menu → modify the **Disconnect time** to **5 seconds**.

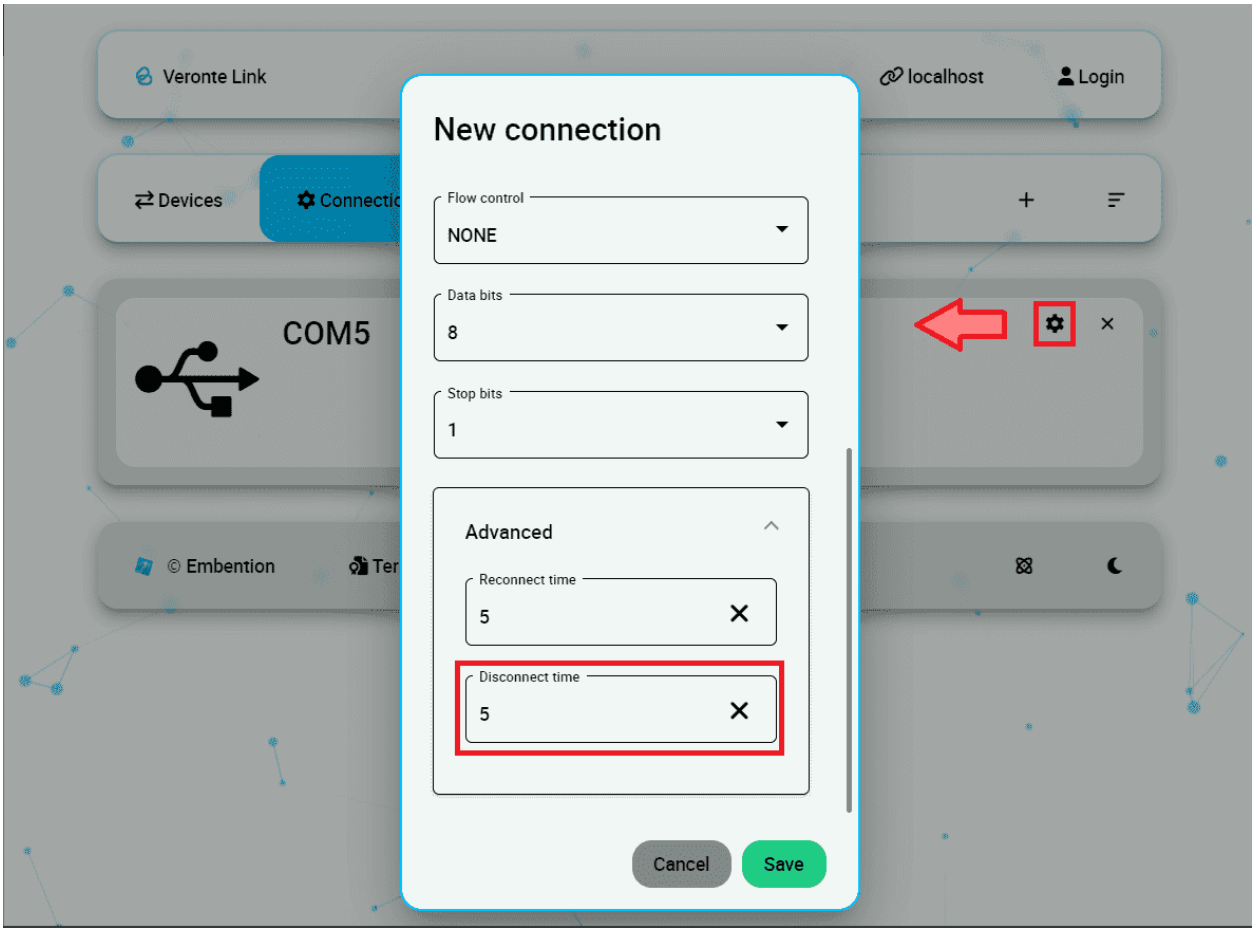


Fig. 2: **Connection configuration - Disconnect time**

If the user is still having problems with this, please contact the support team by creating a ticket in the customer's **Joint Collaboration Framework**; for more information, see Tickets section of the JCF manual.

5.2 Viewing UDP data

An application such as Wireshark can be used to visualize raw data sent from Autopilot 1x. Nonetheless, at the beginning, it may show characaters that do not come from 1x, because Wireshark reads all data from the connected port, including protocol information.

To distinguish 1x messages, the user has to search the matcher *0x0A 0xA0* for **Veronte UDP Telemetry CLI**. The matcher indicates the beginning of the data. In the following example, characters marked with blue correspond to 1x, while yellow characters are the UDP protocol structure.

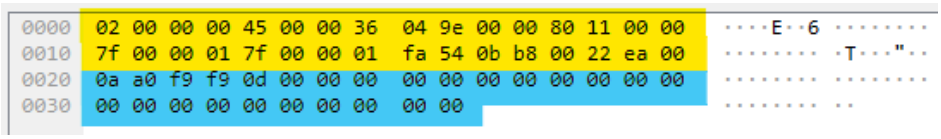


Fig. 3: **Distinguished data on Wireshark**

CHAPTER

SIX

SOFTWARE CHANGELOG

This section presents the changes between the previous software version (**v.6.12.22**) and the current (**v.6.14.28**).

Added

- TCP server connection support
- Support Autopilot multiconnection
- Calculate discovery CRC Configuration after upload and download configuration
- Telemetry, status messages and discovery response to check arbitration Ports functionality