Veronte Link

Release 6/1.1

Embention Sistemas Inteligentes, S.A.

2025-09-18

Contents

Scope of Changes

- Version 1.0
  - Added:
    - Buffer size parameter description in UDP, TCP-Server and TCP-Client connections
    - Warning about set up and operation with a Veronte Autopilot 1x via Cloud in the Cloud connection section
    - Software Changelog between version **6.14.28** and **6.14.60**
  - Changed:
    - Configuration status descriptions updated to v6.14.60 (Operation section)
    - Clarification on Error when replaying a session troubleshooting
- Version 1.1
  - Added:
    - Clarification regarding the deletion of session files
    - Software Changelog between version **6.14.60** and **6.14.64**
    - Note about the error that appears when users fail to log in to Cloud
    - Veronte Telemetry CSV explanation as Additional app
  - Removed:
    - "Planet" connection as connection option, as it has been removed in the v6.14.64 of the app

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.

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

Download and Installation

Veronte Link software is available in the **Veronte Toolbox** platform. From there, users can download and install the application. For more information, please refer to the Veronte Toolbox user manual.

A **personal account** is required to access **Veronte Toolbox**; create a Ticket in the user's **Joint Collaboration Framework** and the support team will create it for you.

Additional apps

Veronte UDP Telemetry CLI

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.

Download and Installation

Veronte UDP Telemetry CLI software is available in the **Veronte Toolbox** platform. From there, users can download and install the application. For more information, please refer to the Veronte Toolbox user manual.

A **personal account** is required to access **Veronte Toolbox**; create a Ticket in the user's **Joint Collaboration Framework** and the support team will create it for you.

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.

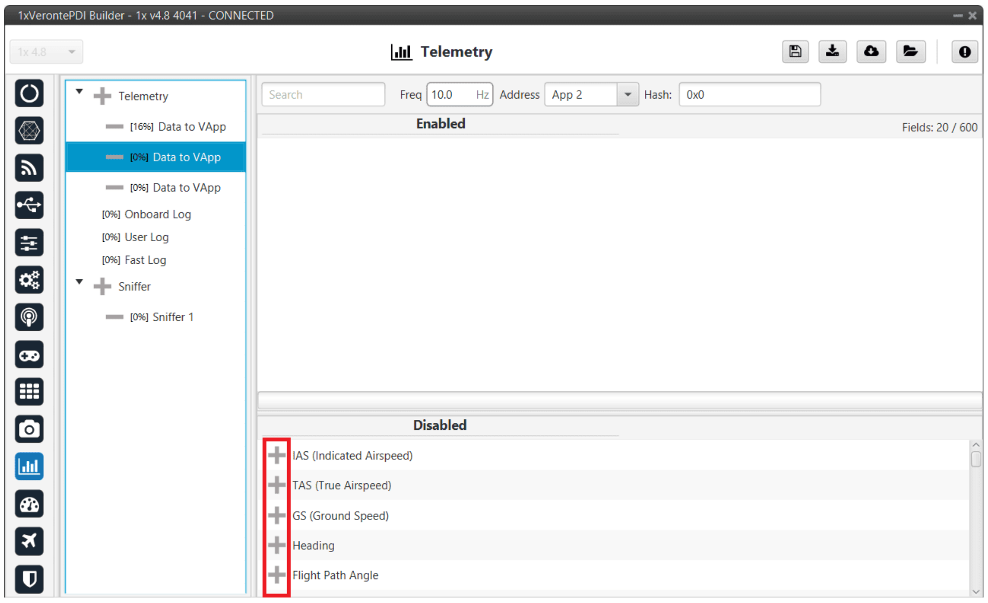
1x PDI Builder

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.

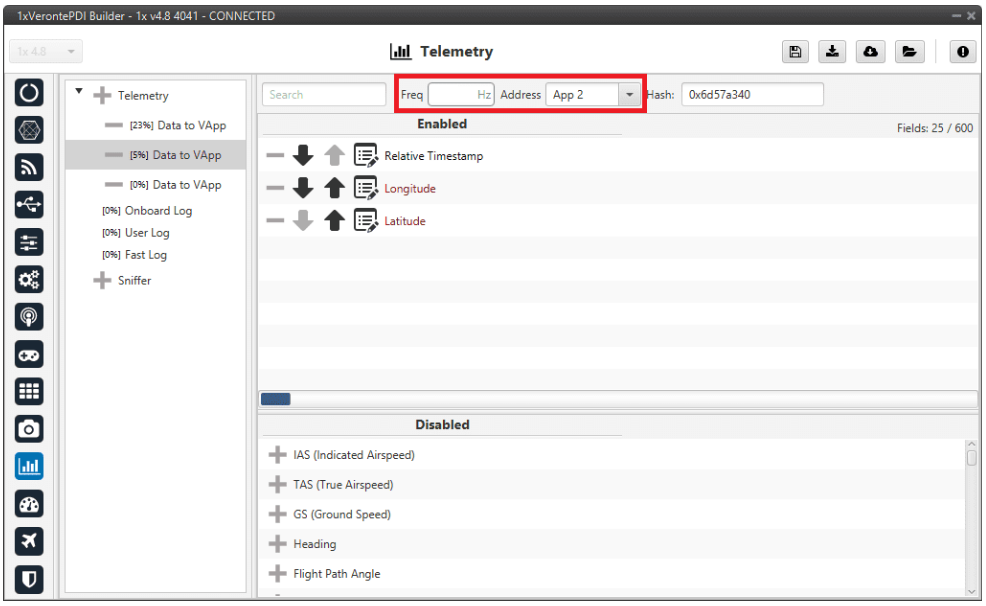


Add variables

> **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.



Data vector parameters

4. Save the changes by clicking button.

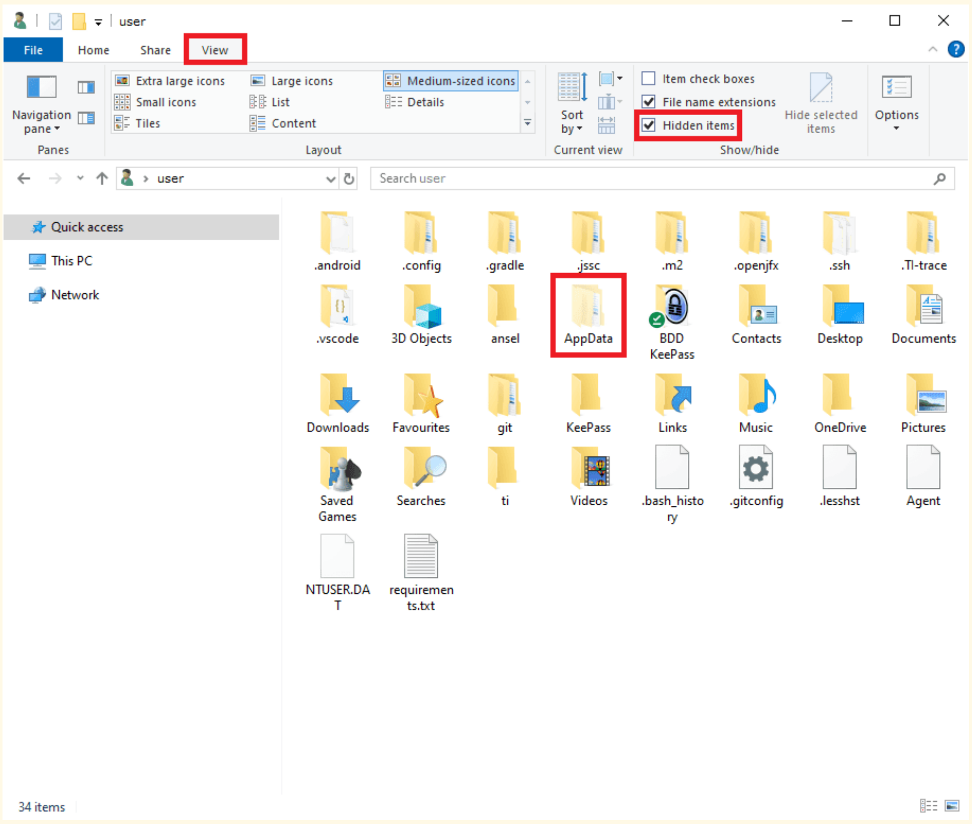
Veronte UDP Telemetry CLI

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`:

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:



Windows File Explorer

- The `tudp.config` file is automatically generated by the application. It will only appear in the folder after you have run Veronte UDP Telemetry CLI at least once.

This configuration file consists of 3 parts:

- `&HEAD hex`. This establishes the header of the UDP messsages sent.
- `&LVARS` ... `END_LVARS`. Defines the **LVARS**, which are complex variables defined by the user, as expressions in which autopilot variables may or may not be used. Each new line between `&LVARS` and `END_LVARS` is a **LVAR**. LVARs can be boolean or number type.

Each LVAR has the following structure: `L[id] = [default value] = [expression]`

- `id`: It is any integer to identify the LVAR and is used in the entries to indicate the LVAR to send.
- `default value`: The default value is the initial value that the LVAR will have the first time it is set.
- `expression`: For each UDP packet sent, the LVAR values are updated with the result by evaluating the expression.

> **Example**
>
> `L40 = false = (u1599_RVAR_1021 > (0.5)) && (u1599_BIT_1053 > (0))`
>
> LVAR L40, initialized as **false**, and for each UDP packet sent, the LVAR value is updated with the result of the expression (u1599_RVAR_1021 > (0.5)) && (u1599_BIT_1053 > (0)).
>
> Where, u1599_RVAR_1021 refers to RVAR 1021 of the autopilot with address 1599, and u1599_BIT_1053 refers to BIT 1053 of the same autopilot.

- **Offsets/Entries**. This is the information that is sent via UDP for each telemetry variable.
  Users must fill in for each entry (except bits) the following fields of the table in **this order**:
  1. **MULT** (float): Scaling factor by which the variable obtained from the autopilot is multiplied.

     > **Note**
     >
     > Only used for the following **VVARs** (VERVARs): L_EQ, RVAR, UVAR, CUSTOM and LVAR.
     >
     > This field does not affect the bits, but must be set nonetheless.

  2. **OFFSET** (float): Offset factor to be added to the variable value obtained from the autopilot, before being multiplied by the MULT value.

     > **Note**
     >
     > This field does not affect the bits, but must be set nonetheless.

3. **TVAR**: Type of variable representing the value sent via UDP. It can be:

> **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.

- **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 $1.175494 \cdot 10^{-38}$)

> **Note**
>
> Unlike the other TVARs, **bits** allows users to define several variables that are packed as only one within the UDP message. To do this, each new line is a variable that is included in the bits entry, until the `&END_BIT` line is read.

4. **UAV** (int): Serial Number of the Autopilot 1x where the variables **come from**.

> **Note**
>
> UAV address does not matter for LVARs, since it is either already indicated in the LVAR expression or it is a value that does not come from any autopilot.

5. **VERVAR/VVAR**: Type of variable in Veronte system.

- **RVAR**: Real variables obtained **directly from the autopilot**
- **UVAR**: Integer variables obtained **directly from the autopilot**
- **BIT**: Bit variables obtained **directly from the autopilot**
- **CUSTOM**
- **NONE**: Equivalent to 0

    - **L_EQ**: Linear equation. Similar to TVAR bits, it allows defining several variables in a single entry.

      The resulting value of this type of entry is the addition of all the consequent variables, multiplied by COEFFICIENT, to which the unit conversion (UNIT), addition (OFFSET) and multiplication (MULT) are finally applied.

      The linear equation continues to wait for more variables until the `&END_L_EQ` line is read.

      > **Note**
      >
      > As implemented, there is no use for the ID field when defining an L_EQ, since the IDs used are those of the following lines.

    - **LVAR**: It must be previously defined as explained above.

6. **ID** (int): 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.
7. **UNIT** (int): Index of the unit of measurement of the variable in case a conversion has to be made.

   Please, see the Index-Unit correspondence table for detailed information.
8. **LIMITS** (optional) (Only for BITs): It is optional and its format is `[min&max]`, both are of float type.
9. **COEFFICIENT** (Only for L_EQ): It is a coefficient of the linear equation.

Below are several examples of the configuration file depending on the type of variable to be sent.

- **RVARs**. Example with Relative Timestamp, Longitude and Latitude variables:

```
#HEAD HEX
&HEAD   0AA0

#MULT       OFFSET      TVAR            UAV         VERVAR      ID      UNIT
1000        0           UInt32          1599        RVAR        300     NONE
1           0           Float           1599        RVAR        500     NONE
1           0           Float           1599        RVAR        501
NONE

# First row: Send Time Since Hardware Start-Up (Milliseconds) as an UInt32
(4 bytes)
# Second row: Send Longitude as a Float (4 bytes)
# Third row: Send Latitude as a Float (4 bytes)
```

- **LVARs**:

```
#HEAD HEX
&HEAD   1FB9

&LVARS
L1 = 0 = L1 + 1
L70 = 20 = L70 + L1
L45 = false = L70 % 2 == 0
L80 = false = (u1599_RVAR_1021 > (0.5)) && (u1599_BIT_1053 > (0))
&END_LVARS

# L80 initially has a value of false. Next values are obtained from the
expression
# (u1599_RVAR_1021 > (0.5)) && (u1599_BIT_1053 > (0))
# u1599_RVAR_1021 = value of RVAR 1021 (stky21, Stick Input y21)
# from autopilot with address 1599.
# The value of this variable is also updated every time

# Use of Lvars
#MULT       OFFSET      TVAR            UAV         VERVAR      ID      UNIT
1           0           Int16           0000        LVAR        1       NONE
1           0           Float           0000        LVAR        70      NONE
1           0           byte            0000        LVAR        45      NONE

# UAV address does not matter for these LVARs since they do not come from an
autopilot.

# First row: Send L1 as an Int16 (2 bytes)
# Second row: Send L70 as a Float (4 bytes)
# Third row: Send L45 as a byte (1 byte)
```

- **BITs**:

```
#HEAD HEX
&HEAD   0AA0

#MULT       OFFSET      TVAR            UAV         VERVAR      ID      UNIT
1           0           bits
#UAV        VERVAR      ID          UNIT        LIMITS(optional)
1599        RVAR        501         NONE
1599        RVAR        500         NONE        [-1000&1000]
&END_BIT

# Mult and Offset do not affect bits, but they must be set regardless.
# Any int value is valid and acts the same.
```

- **BITs with LVARs**. Taking the LVARs defined in the previous example:

```
#HEAD HEX
&HEAD   1FB9

#Definition of LVARs
&LVARS
L1 = 0 = L1 + 1
L70 = 20 = L70 + L1
L45 = false = L70 % 2 == 0
&END_LVARS

#Bits example with lvars
#MULT       OFFSET      TVAR            UAV         VERVAR      ID      UNIT
1           0           bits
#UAV        VERVAR      ID          UNIT        LIMITS(optional)
0000        LVAR        1           NONE        [0&10]
0000        LVAR        70          NONE        [100&500]
0000        LVAR        45          NONE
&END_BIT

# Mult and Offset do not affect bits, but they must be set regardless.
# Any int value is valid and acts the same.

# In this example, each one of the variables occupies one bit in the
resulting message.
# L1, which is incremented by one, is checked if it is within the set limit
0&10,
# i.e., for values strictly greater than 0 and strictly less than 10,
# the bit will be 1, and for all other values, it will be 0.

# The same applies to L70, when 100<L70>500, the bit is 1, and for the rest
it is 0.

# L45 on the other hand doesn't have a limit.
# When no limit is established, it compares it to 1.
# Since L45 is a boolean that checks that L70 is even,
# the bit will be one when the value is 1, and 0 when not.

# Concrete example: L1 = 16, L70 = 156, L45 = true (because L70 is even)
# The UDP packet will be:
# HEADER: 31 -71
# L1: 16 0
# L70: 0 0 28 67
# L45: 1
# bits: 6 = bits[1 1 0] because:
# L1 is not in the limits (0), L70 is (middle 1), and L45 is 1/true (left
1).
```

```
# The order is from least to most significant in the order indicated in the
bits list.
```

- **L_EQ**:

```
#HEAD HEX
&HEAD   0AA0

#MULT       OFFSET      TVAR            UAV         VERVAR      ID      UNIT
1           0          Int16           0000         L_EQ       NONE     NONE
#UAV        VERVAR      ID          UNIT        COEFFICIENT
1001        RVAR        1           NONE        2.3
1001        UVAR        1           NONE        2.3
&END_L_EQ

# LINEAR EQUATION:
# ((COEFFICIENT*RVAR(1) + COEFFICIENT*UVAR(1)) + Offset) * Mult
# ((2.3*RVAR(1) + 2.3*UVAR(1)) + 0) * 1
```

- **L_EQ with LVARs**. Taking the LVARs defined in the previous example:

```
#HEAD HEX
&HEAD   1FB9

#Definition of LVARs
&LVARS
L1 = 0 = L1 + 1
L70 = 20 = L70 + L1
&END_LVARS

#Linear equation example with lvars
#MULT       OFFSET      TVAR            UAV         VERVAR      ID      UNIT
2           13          Float           0000        L_EQ        0       NONE
#UAV        VERVAR      ID          UNIT        COEFFICIENT
0000        LVAR        1           NONE        50
0000        LVAR        70          NONE        25
&END_L_EQ

# Concrete example: L1 = 1, L70 = 21
# ((50*L1 + 25*L70) + Offset) * Mult = ((50*1 + 25*21) + 13) * 2 = 1176
# Sent as a Float, therefore in the udp packet it will be:
# 1176 = [0 0 -109 68]
```

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

Unit ID	Unit
6	km
62	mm
63	cm
7	mi
8	NM
9	yd
10	ft
11	in
12	m/s²
13	ft/s²
14	in/s²
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)

Unit ID	Unit
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

Unit ID	Unit
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

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

Unit ID	Unit
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

Unit ID	Unit
145	$(cm/s)^2$
146	$(mm/s)^2$
327	Ω
328	Henrios
322	watios
323	kW
324	Kgm/s
325	erg/s
326	cv
331	m^3
332	dm^3
333	mm^3
334	L
335	mL

Operation

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

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:

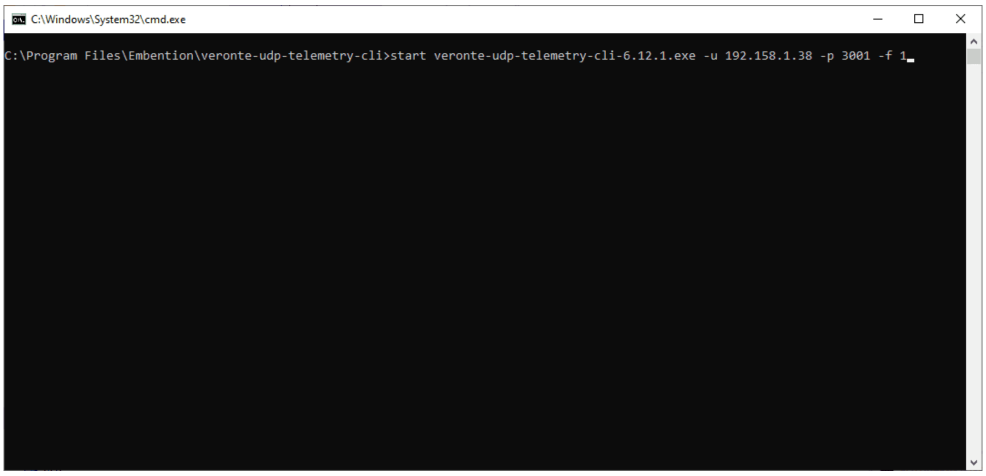
   Veronte UDP Telemetry CLI shorcut

   This will send the UDP messages with the following default configuration:
   - **Host url**: 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**: UDP address
   - **- p**: UDP port
   - **- f**: Desired frequency of data transmission (Hz)

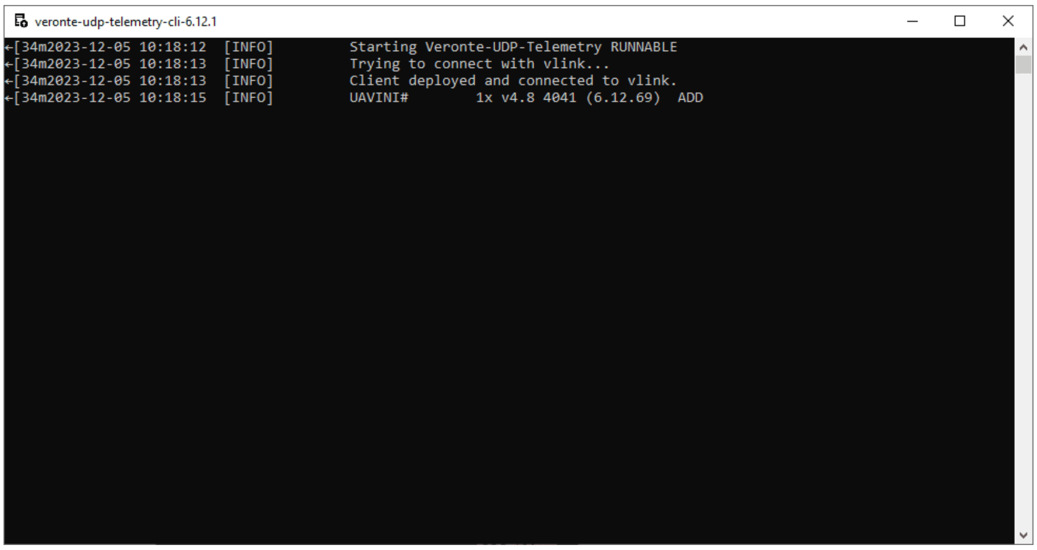
   This is an example:



```
C:\Program Files\Embention\veronte-udp-telemetry-cli>start veronte-udp-telemetry-cli-6.12.1.exe -u 192.158.1.38 -p 3001 -f 1
```

Launching from terminal example

The expected outcome is the following:



```
←[34m2023-12-05 10:18:12  [INFO]         Starting Veronte-UDP-Telemetry RUNNABLE
←[34m2023-12-05 10:18:13  [INFO]         Trying to connect with vlink...
←[34m2023-12-05 10:18:13  [INFO]         Client deployed and connected to vlink.
←[34m2023-12-05 10:18:15  [INFO]         UAVINI#        1x v4.8 4041 (6.12.69)  ADD
```

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.

Veronte Telemetry CSV

Veronte Telemetry CSV

Veronte Telemetry CSV is an additional command-line tool that processes Veronte Link sessions to export recorded telemetry data into CSV files.

Download and Installation

Veronte Telemetry CSV software is available in the **Veronte Toolbox** platform. From there, users can download and install the application. For more information, please refer to the Veronte Toolbox user manual.

A **personal account** is required to access **Veronte Toolbox**; create a Ticket in the user's **Joint Collaboration Framework** and the support team will create it for you.

Operation

This section details how to convert recorded session into CSV files.

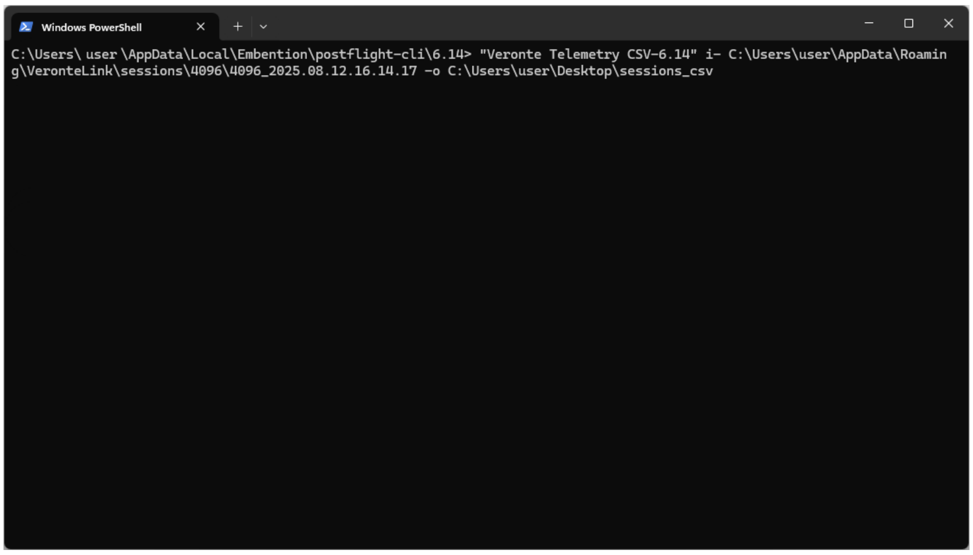
To properly operate the program, launch **Veronte Telemetry CSV** `.exe` from terminal with the following required parameters:

- **-i**: Path to the session folder that users want to convert.

  > **Note**
  > - Remember that the folder containing the session files is located in the following path `C:\Users\USER NAME\AppData\Roaming\VeronteLink\sessions`
  > - These session files are grouped by autopilot Serial Number and inside they are generated with the date and time.

- **-o**: Path where session files will be stored in CSV format.

This is an example:

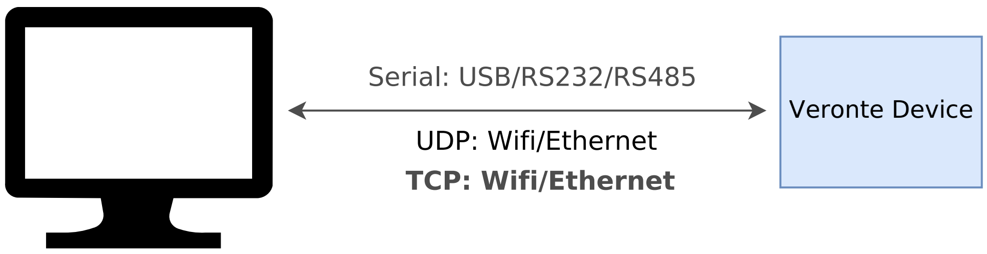


```
C:\Users\ user \AppData\Local\Embention\postflight-cli\6.14> "Veronte Telemetry CSV-6.14" i- C:\Users\user\AppData\Roamin
g\VeronteLink\sessions\4096\4096_2025.08.12.16.14.17 -o C:\Users\user\Desktop\sessions_csv
```

Operation

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

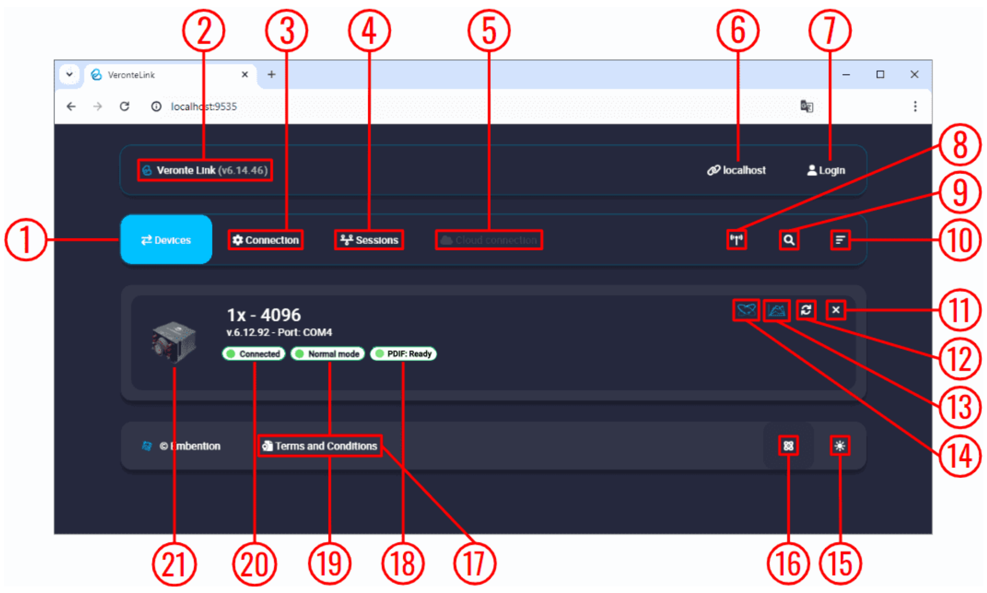
1. Connect the device to a PC via Serial (USB, RS232 or RS485) or UDP/TCP (Wifi or Ethernet).



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:



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. **Connections**: This menu allows the user to configure the connection between the PC and a Veronte device. See Connections 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.

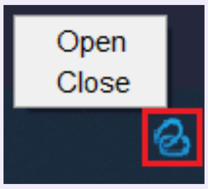
14. **Open Veronte Ops**: From here users can access Veronte Ops on the same version of the connected device.
15. **Dark/light mode**: Switches to light/dark mode, changing the display mode of the interface.
16. **Switch particles**: Particles can be on or off, changing the application appearance.
17.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



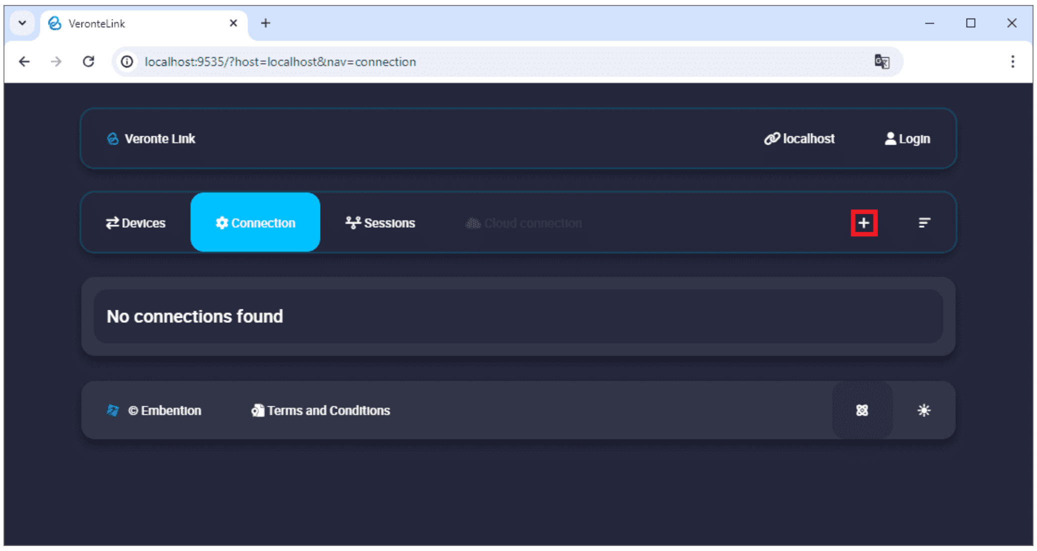
Veronte Link icon

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.

Connections

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



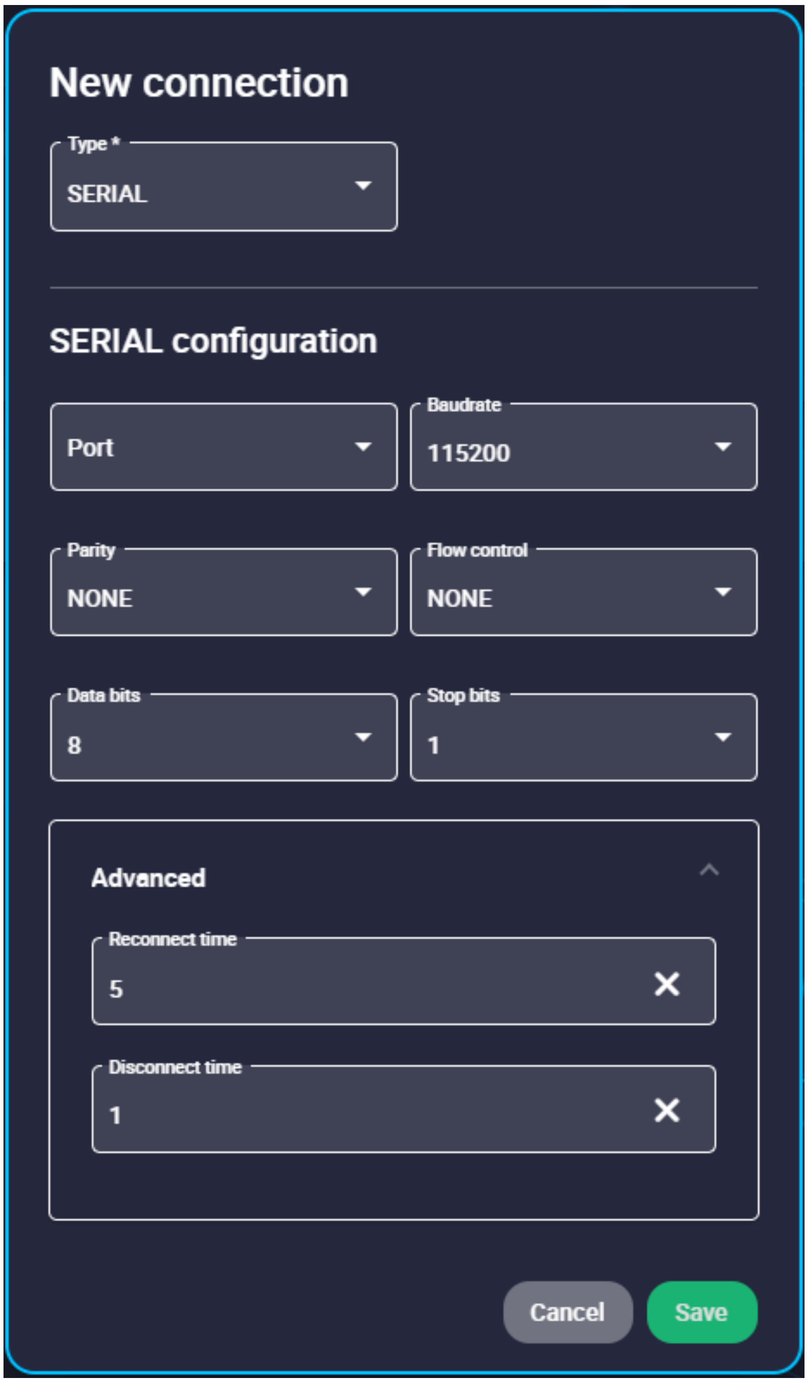
Connection menu

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.


New connection

Type *: SERIAL

SERIAL configuration

Port

Baudrate: 115200

Parity: NONE

Flow control: NONE

Data bits: 8

Stop bits: 1

Advanced

Reconnect time: 5

Disconnect time: 1

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).

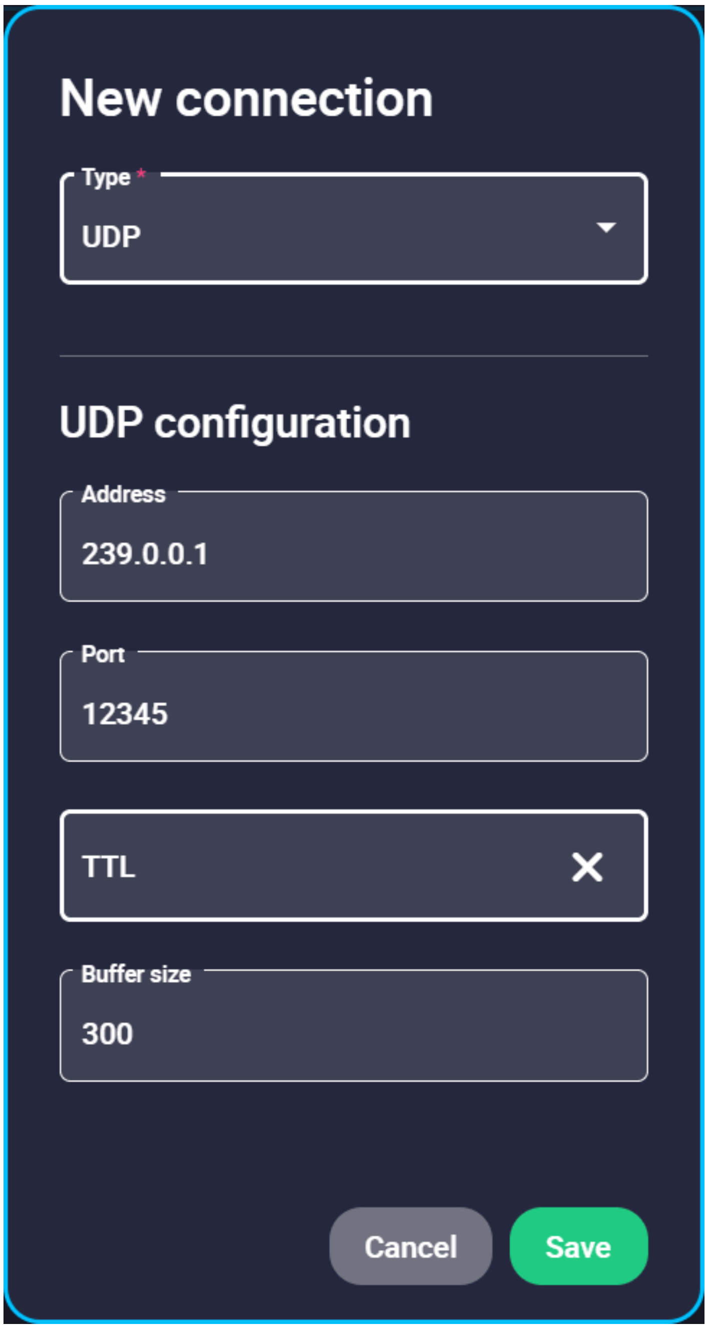
  More information about the port where the device is connected is explained in Serial connection - Integration examples section of the present manual.
- **Baudrate**: This field specifies how fast data is sent over a serial line.
- **Parity**: Is a method of detecting errors in transmission.

  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.



New connection

Type * UDP

UDP configuration

Address 239.0.0.1

Port 12345

TTL

Buffer size 300

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.

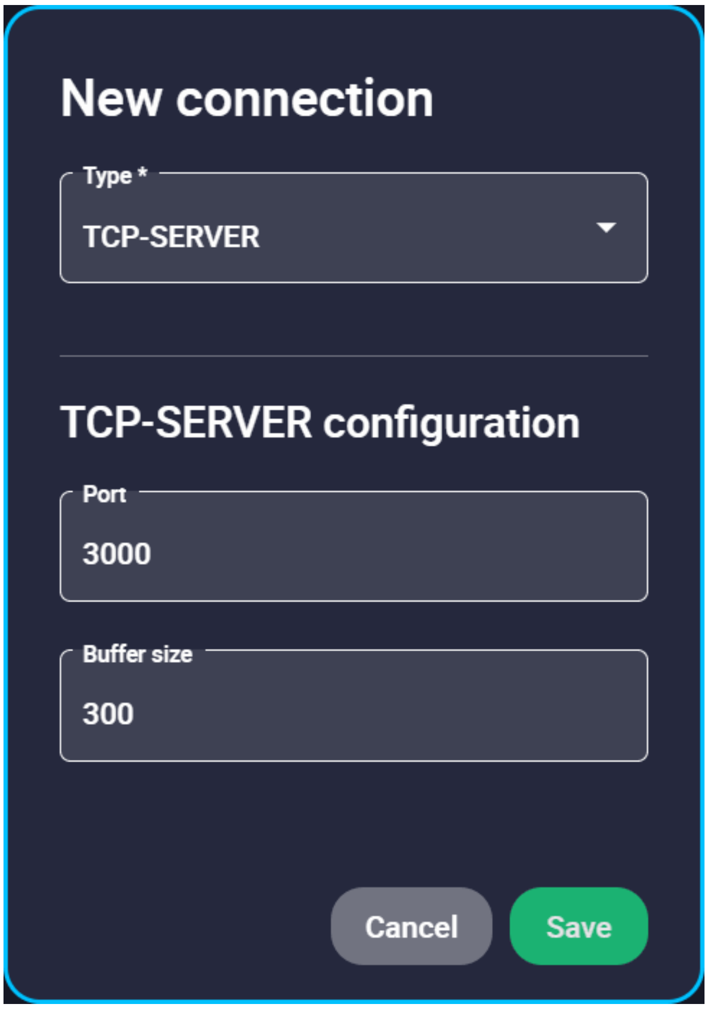
    A default value should automatically be set.
  - **Buffer size**: Users would have to increase or decrease this value depending on the number of devices sending information through this channel.

    By default this parameter has a value of **300**, which is the maximum value of a VCP message.

    > **Explanation**
    >
    > For example, if a PCS is connected by radio to an Autopilot 1x, the buffer size should be increased because more consecutive messages arrive and can be mixed between them, generating invalid messages that cause messages to be discarded.

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

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


New connection

Type *: TCP-SERVER

TCP-SERVER configuration

Port: 3000

Buffer size: 300

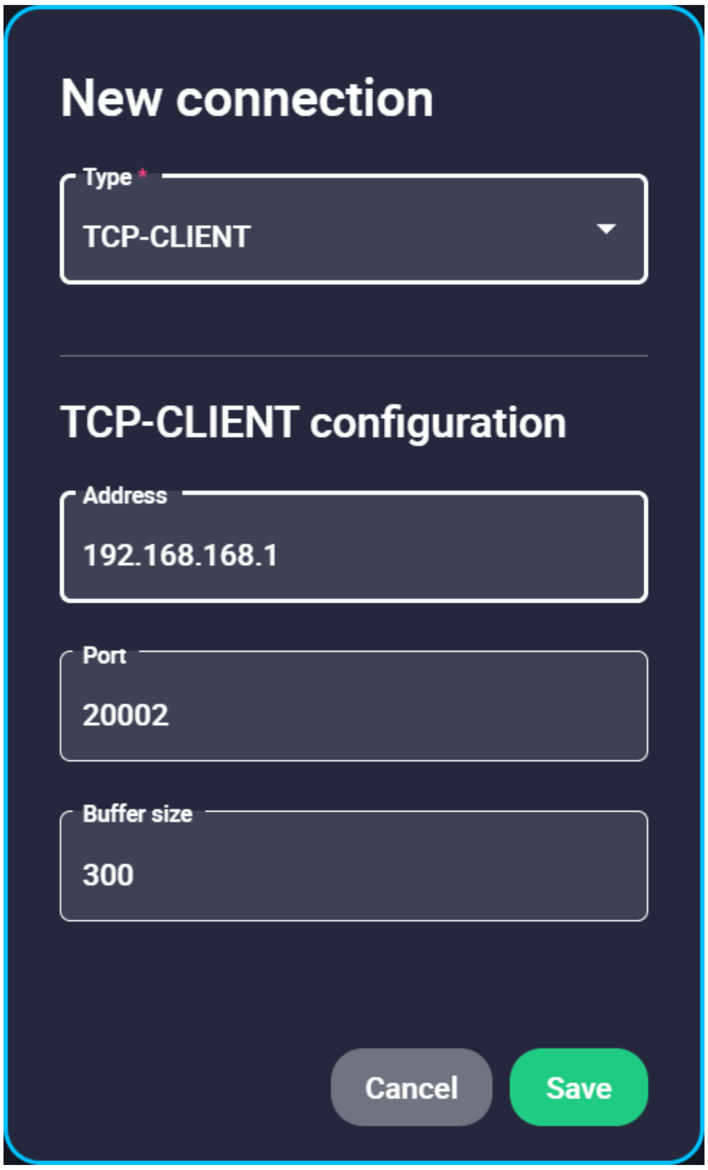
TCP-SERVER connection configuration

- **Port**: Set the TCP port from which the devices will get the information provided by Veronte Link.
- **Buffer size**: Users would have to increase or decrease this value depending on the number of devices sending information through this channel.
  By default this parameter has a value of **300**, which is the maximum value of a VCP message.

> **Explanation**
>
> For example, if a PCS is connected by radio to an Autopilot 1x, the buffer size should be increased because more consecutive messages arrive and can be mixed between them, generating invalid messages that cause messages to be discarded.
>
> Otherwise, if a very high buffer size is set, and only one device sends messages, the buffer will take longer to fill up, thus generating a delay in the reception of messages.

> **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.


New connection

Type *: TCP-CLIENT

TCP-CLIENT configuration

Address: 192.168.168.1

Port: 20002

Buffer size: 300

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.
- **Buffer size**: Users would have to increase or decrease this value depending on the number of devices sending information through this channel.

  By default this parameter has a value of **300**, which is the maximum value of a VCP message.

> **Explanation**
>
> For example, if a PCS is connected by radio to an Autopilot 1x, the buffer size should be increased because more consecutive messages arrive and can be mixed between them, generating invalid messages that cause messages to be discarded.

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

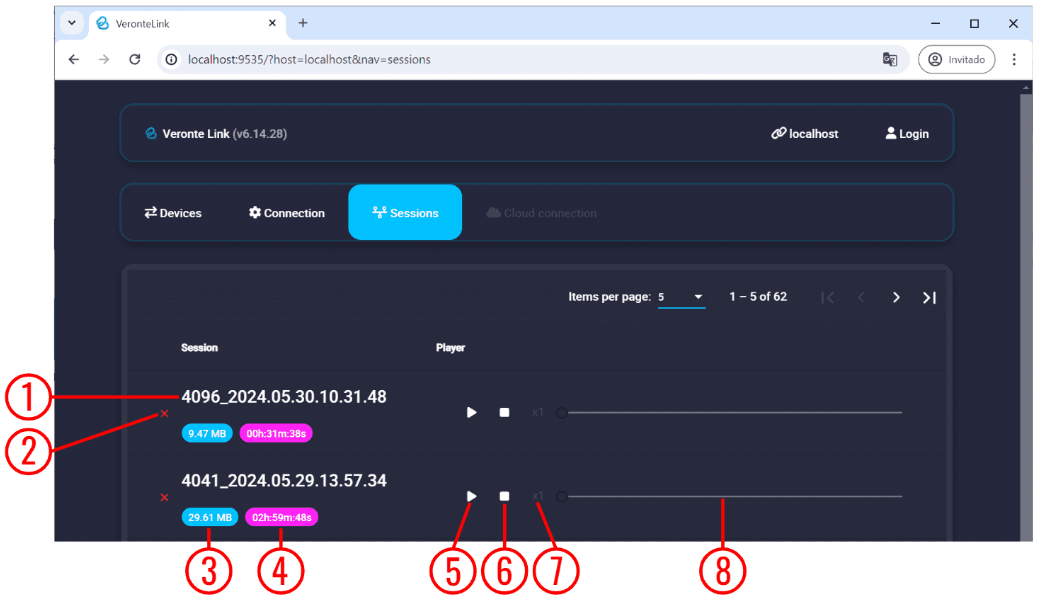
Finally, click on **Save**.

Sessions

Sessions tab displays all **finished** device sessions.

> **Important**
>
> - Sessions that are currently **being recorded** will not be displayed.
> - A session from the currently **connected device** cannot be replayed.
> - If users experience problems when attempting to replay a session, please check the Error when replaying a session - Troubleshooting section of this manual.

The following image and list describe each functionality.

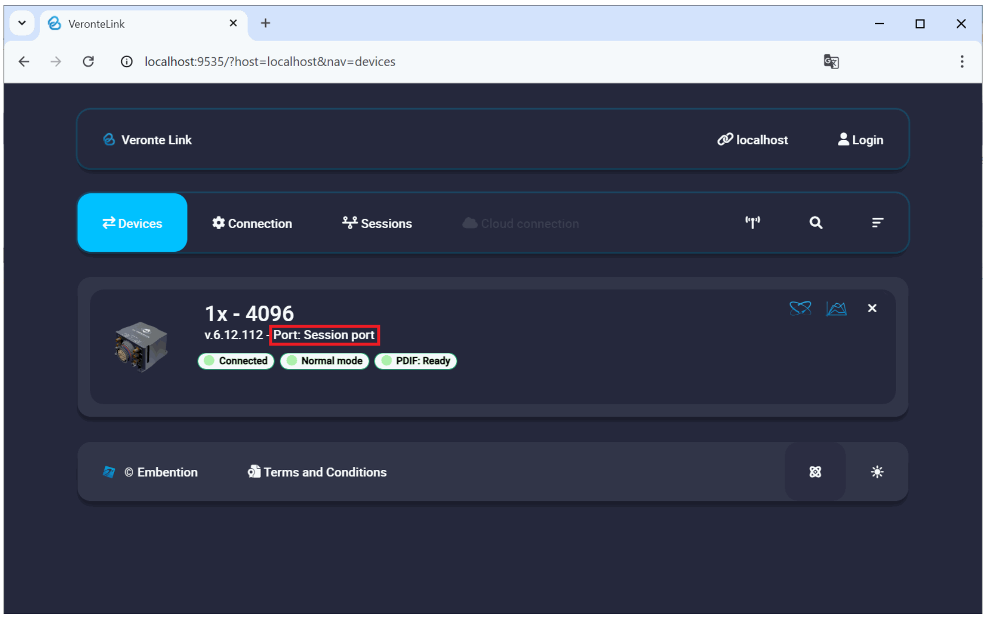


Sessions menu

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`

   > **Note**
   >
   > If users delete a session directly in the local Veronte Link session folder, it may still appear in the application. To fix this, simply refresh the Veronte Link application.

3. **Files weight**.
4. **Duration**.
5. **Play/Pause**: Play button creates a **virtual device** in the "session port" similar to the following figure:

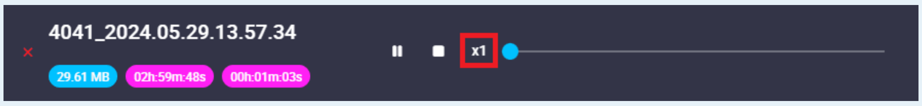


Virtual device

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.
>
> 
>
>
> **Speed button enabled**

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

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.

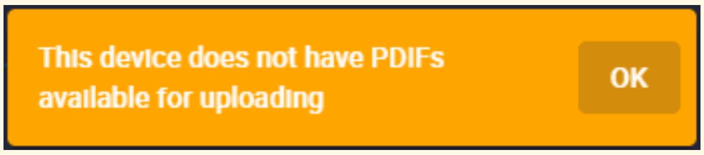
To configure this type of connection, these steps must be followed:

⚠ **Warning**

In order to set up and operate a Veronte Autopilot 1x via Cloud connection, users must **first**:

1. Log in Cloud.
2. Establish a connection to Autopilot 1x that is not through Cloud (Serial, UDP, TCP-Server or TCP-Client).
3. **Upload** PDIF (configuration) to 1x with the **1x PDI builder** app or with the **Upload PDIFs to cloud** button .

   This button will work as long as the cloud device is connected and the PDIFs are in the Ready state otherwise it will show errors:

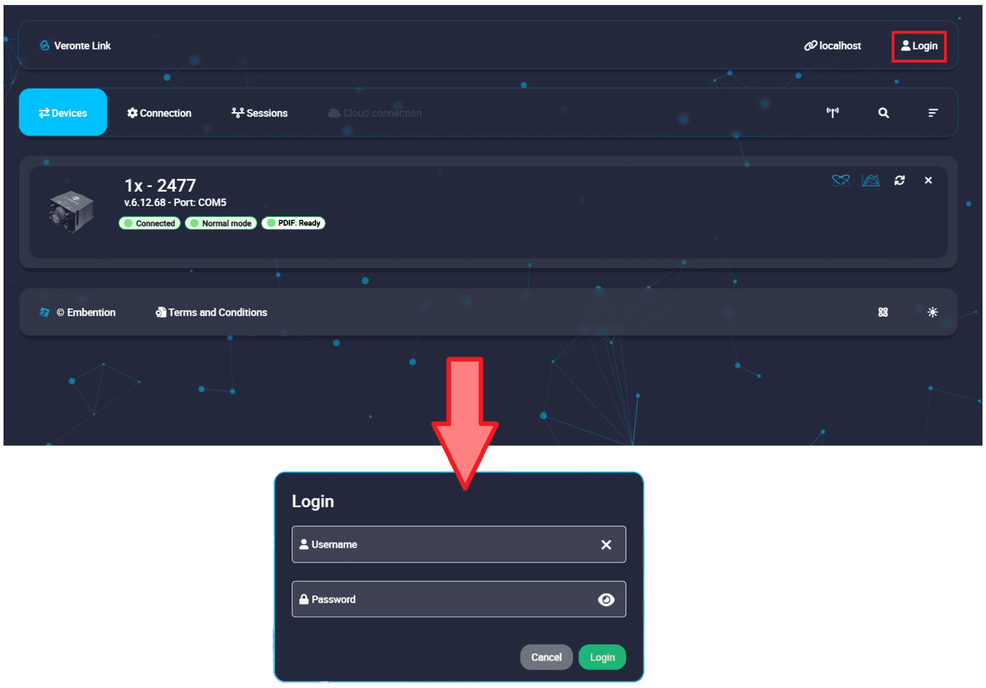
   This device does not have PDIFs available for uploading

Now the users are ready to establish the connection via Cloud and work. If these steps are not followed, Autopilot 1x will be in **PDIF: Failed load** status.

ⓘ **Note**

This only needs to be done once per Veronte Autopilot 1x and per configuration.

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

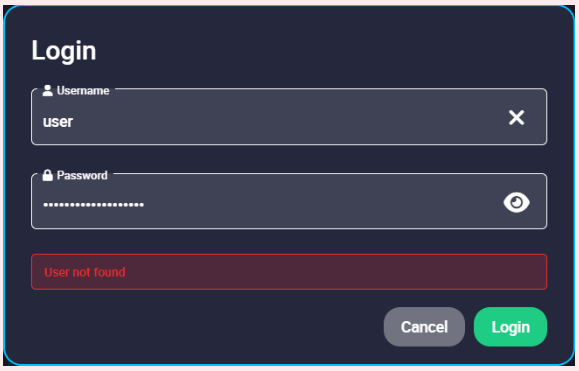


Cloud Connection: Login

Error

If incorrect credentials are entered, the system will display a specific error message:

- "**User not found**": This message appears if the username entered does not exist in the system.

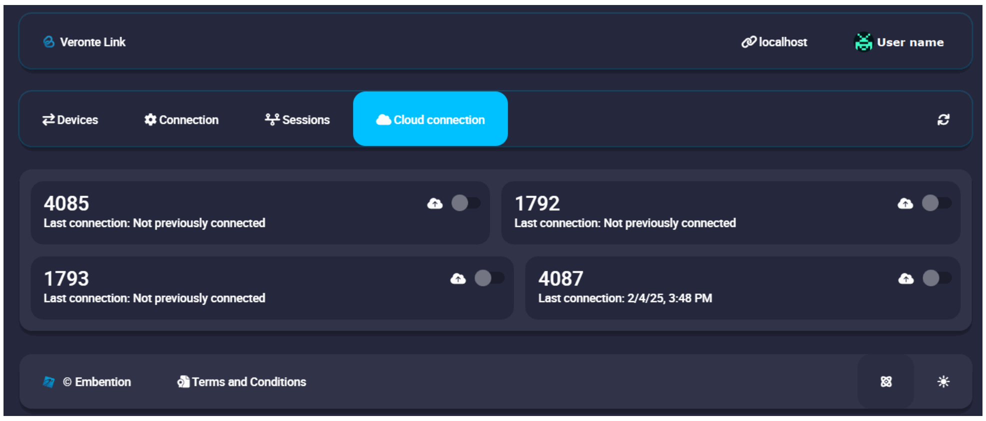


User not found

- "**Incorrect username or password**": This message appears if the password is incorrect or the user does not have the required permissions to log in.

> ⓘ **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.

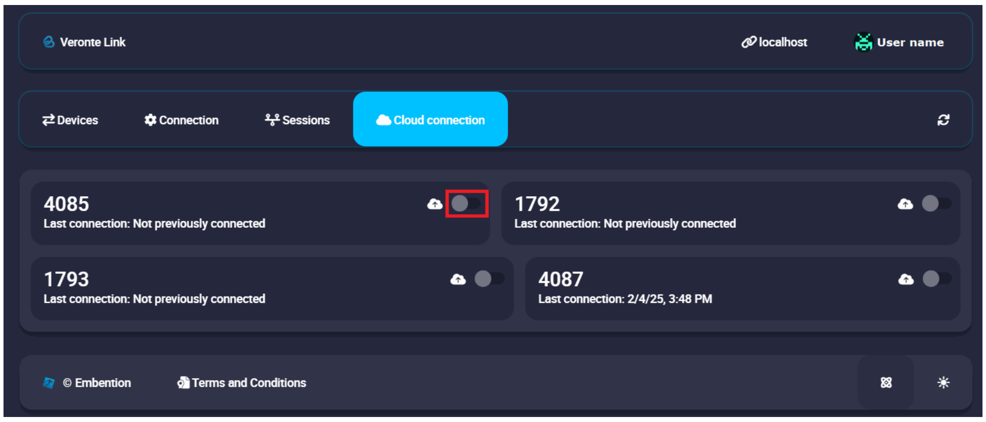

Veronte Link
localhost
User name
Devices
Connection
Sessions
Cloud connection
4085
Last connection: Not previously connected
1792
Last connection: Not previously connected
1793
Last connection: Not previously connected
4087
Last connection: 2/4/25, 3:48 PM
© Embention
Terms and Conditions

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.


Veronte Link
localhost
User name
Devices
Connection
Sessions
Cloud connection
4085
Last connection: Not previously connected
1792
Last connection: Not previously connected
1793
Last connection: Not previously connected
4087
Last connection: 2/4/25, 3:48 PM
© Embention
Terms and Conditions

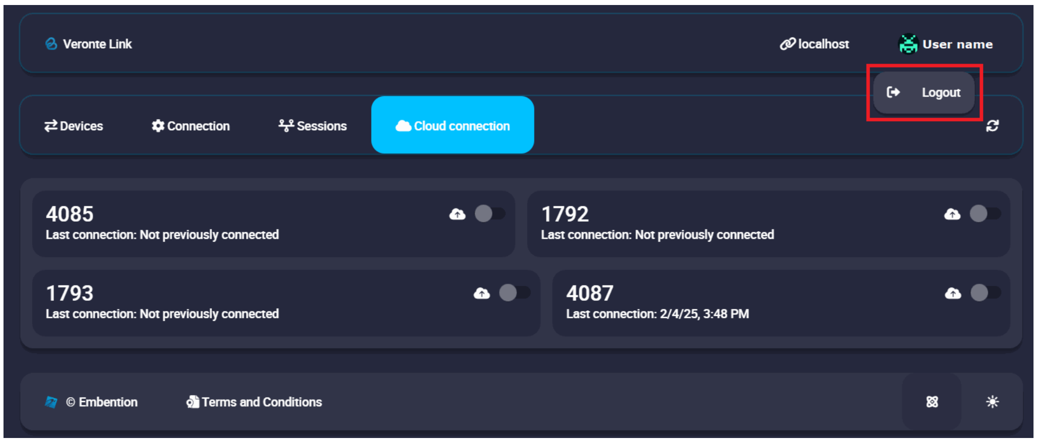
Cloud Connection: Connect to an Autopilot 1x

> **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.

To **log out**, click on the username to enable the log out button, and then press it.



4085
Last connection: Not previously connected

1792
Last connection: Not previously connected

1793
Last connection: Not previously connected

4087
Last connection: 2/4/25, 3:48 PM

Cloud Connection: Log out

Integration examples

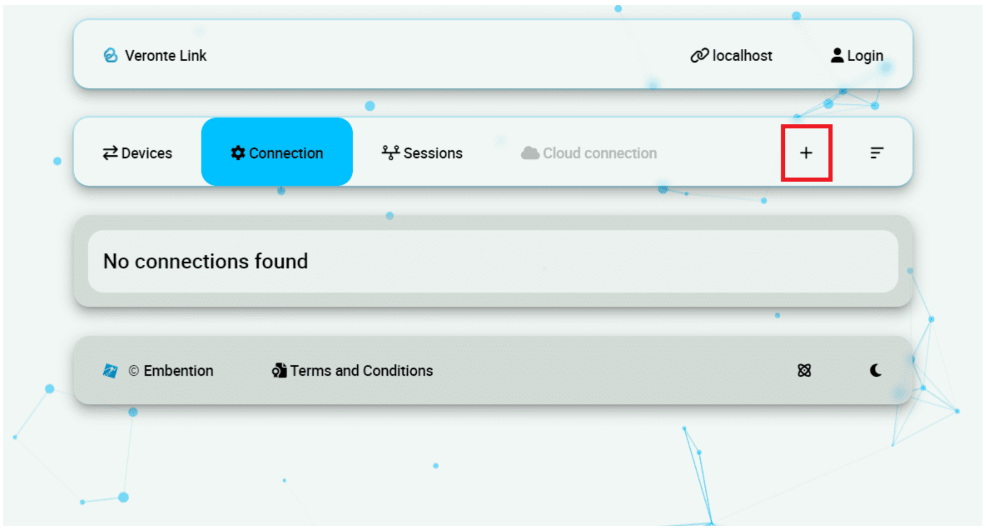
Serial connection

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.

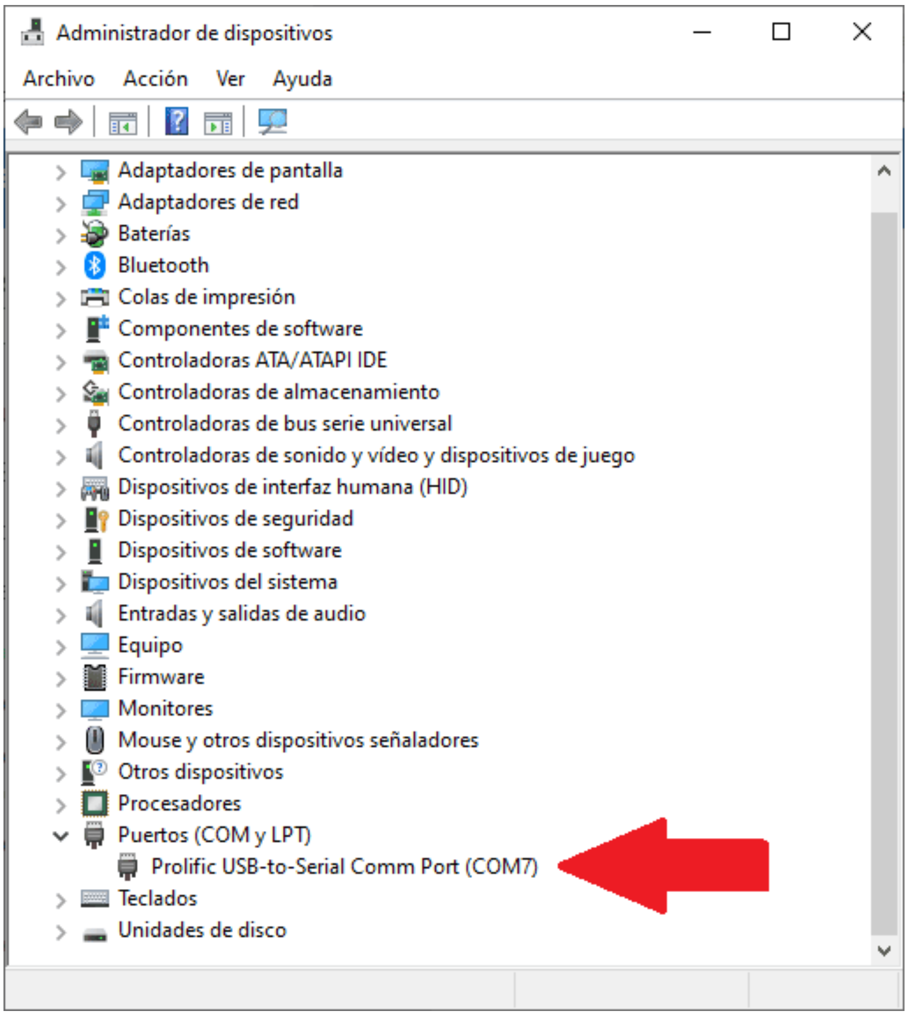
   By default, every MC is capable to comunicate through USB, RS232 and RS485 so any of these can be used (properly adapted to USB/serial).

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

   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**.



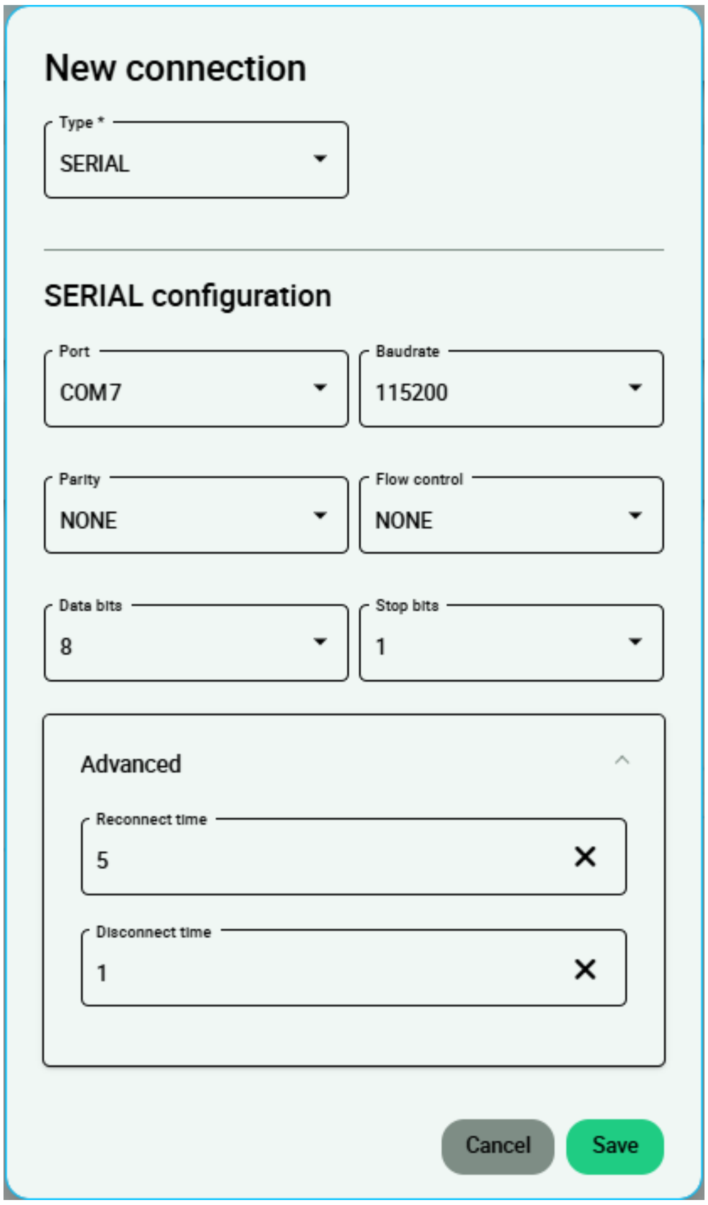
Administrador de dispositivos

Archivo Acción Ver Ayuda

- Adaptadores de pantalla
- Adaptadores de red
- Baterías
- Bluetooth
- Colas de impresión
- Componentes de software
- Controladoras ATA/ATAPI IDE
- Controladoras de almacenamiento
- Controladoras de bus serie universal
- Controladoras de sonido y vídeo y dispositivos de juego
- Dispositivos de interfaz humana (HID)
- Dispositivos de seguridad
- Dispositivos de software
- Dispositivos del sistema
- Entradas y salidas de audio
- Equipo
- Firmware
- Monitores
- Mouse y otros dispositivos señaladores
- Otros dispositivos
- Procesadores
- Puertos (COM y LPT)
  - Prolific USB-to-Serial Comm Port (COM7)
- Teclados
- Unidades de disco

Windows Device Manager

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


New connection

Type *: SERIAL

SERIAL configuration

Port: COM7

Baudrate: 115200

Parity: NONE

Flow control: NONE

Data bits: 8

Stop bits: 1

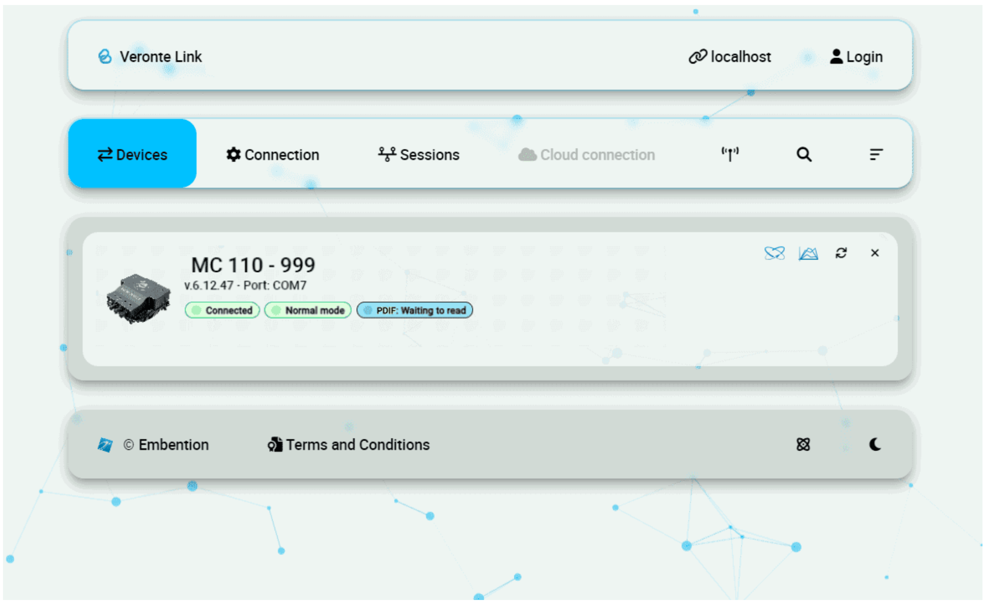
Advanced

Reconnect time: 5

Disconnect time: 1

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

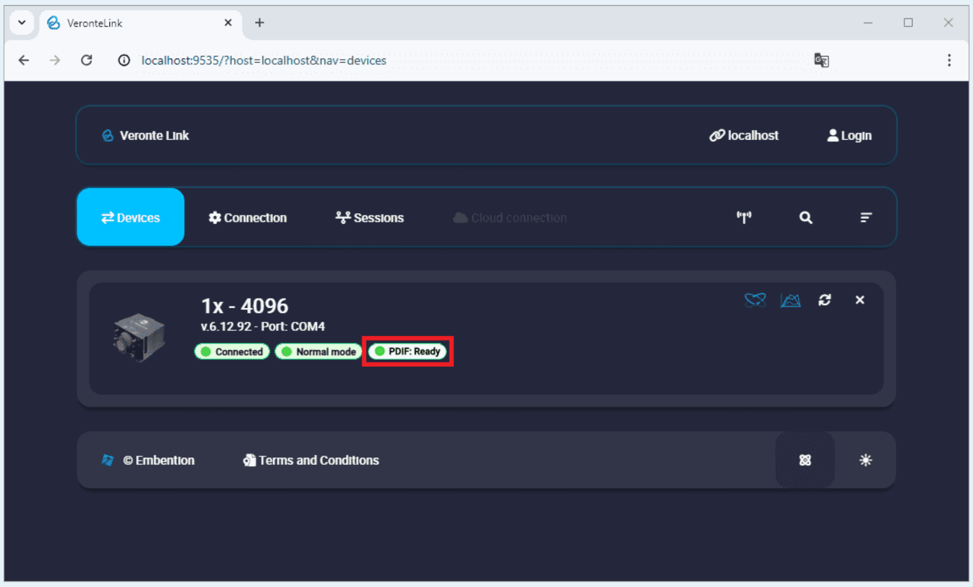


MC unit correctly connected

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 **PDIF: Waiting to read** by **PDIF: Ready**, since **only Autopilot 1x is able to change or load configuration in normal mode**.



Veronte Autopilot 1x connected and ready

For Veronte devices other than 1x, **PDIF: Waiting to read** should be replaced by the status **PDIF: Not Downloaded**.

UDP connection

Wi-Fi/Ethernet configuration

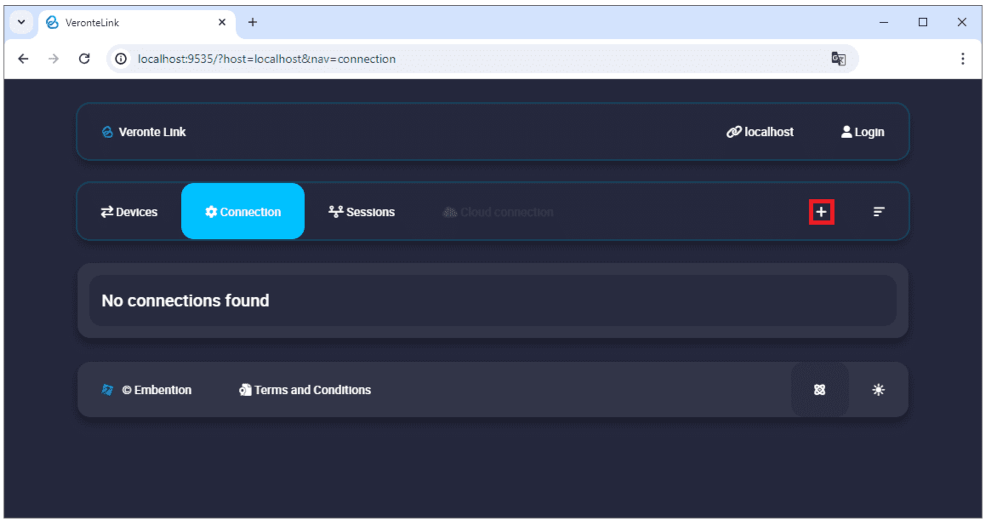
The following steps are applied to a **PCS** unit as an example.

Important

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

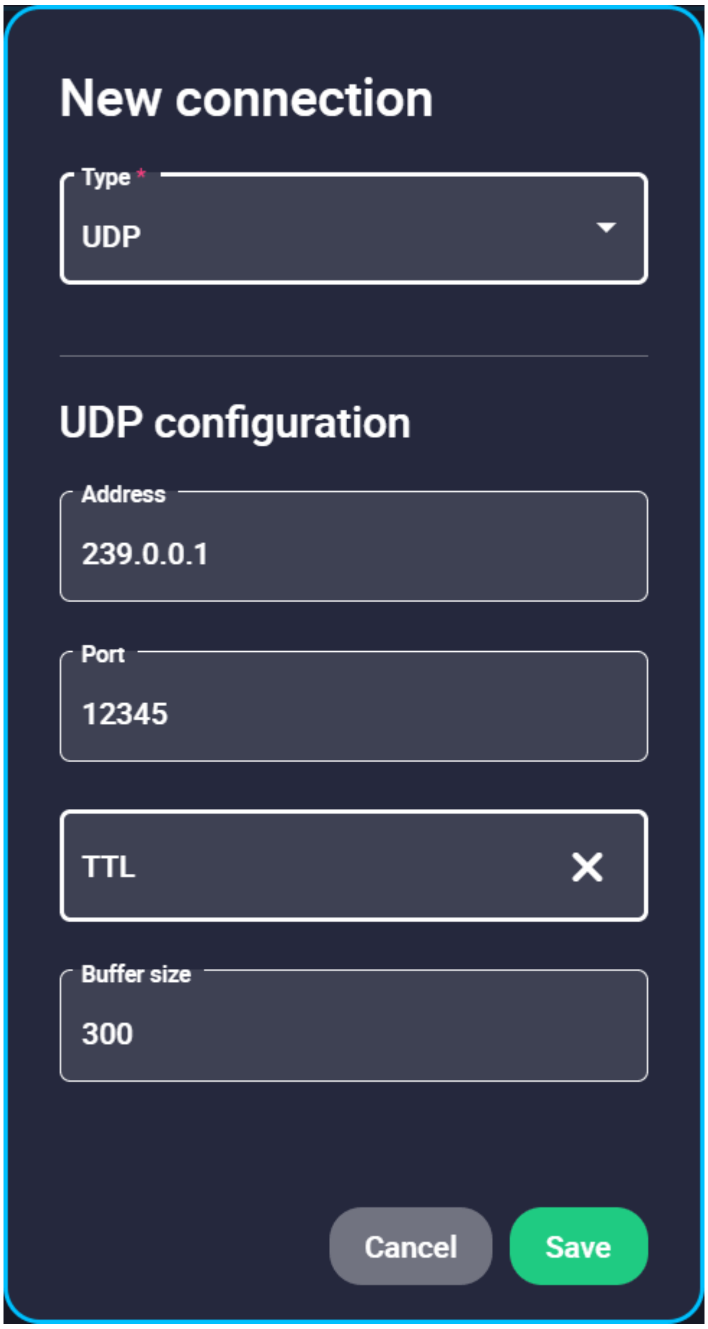
1. The first step is to look under available networks for the PCS unit and connect to it.
2. Once the connection is made, enter **Veronte Link** and configure the UDP connection in the **Connection menu**.

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



Add new connection

3. Then, the configurable parameters must be entered.


New connection

Type *: UDP

UDP configuration

Address: 239.0.0.1

Port: 12345

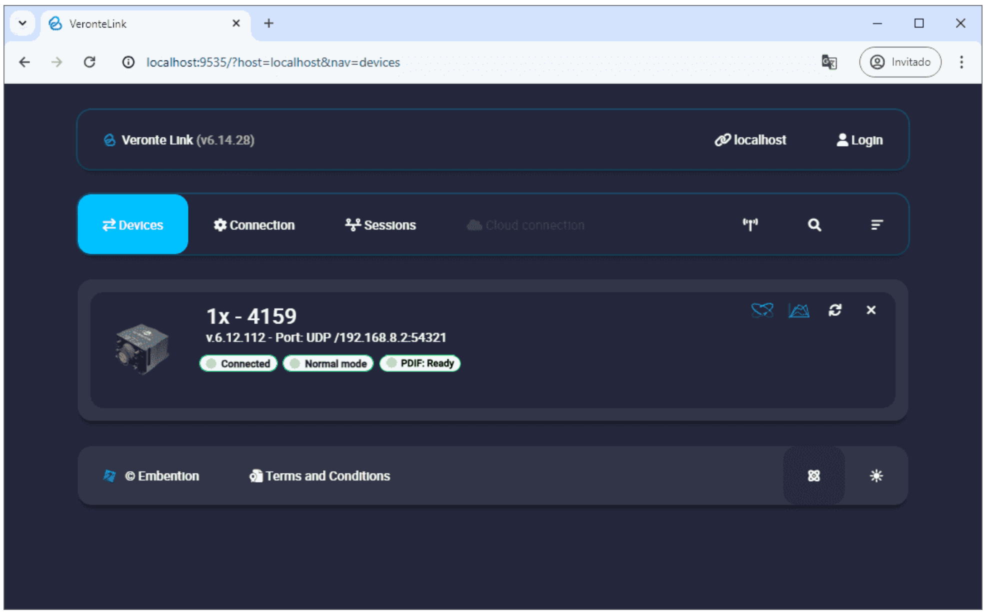
TTL

Buffer size: 300

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



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.

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

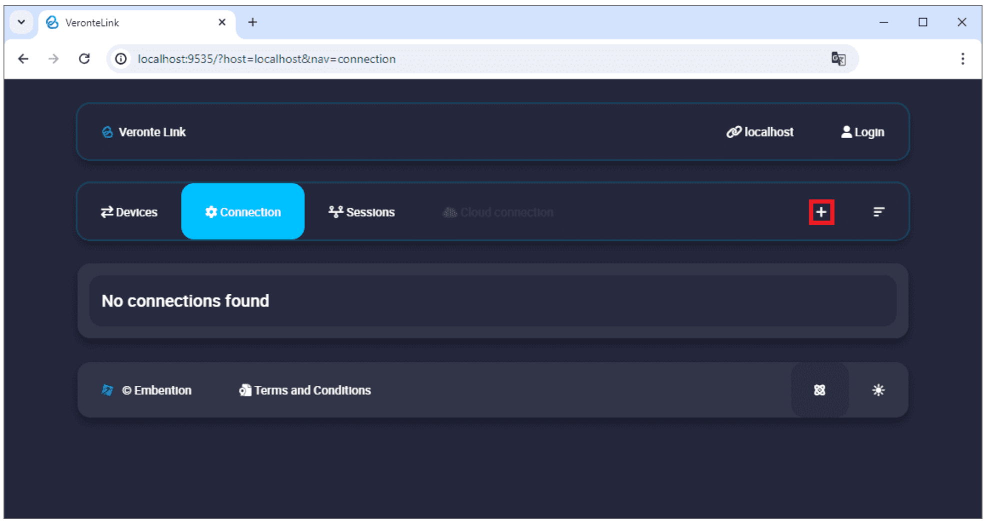
In this connection, the **radio** acts as "**Client**" and **Veronte Link** as "**Server**".

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.

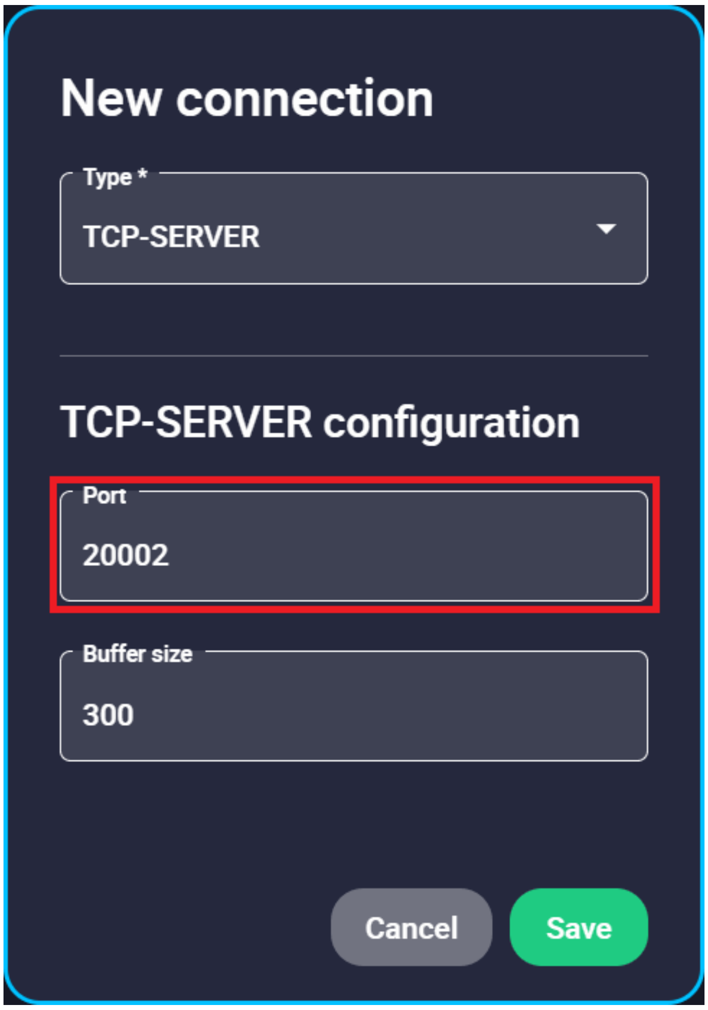
2. Connect **Veronte Autopilot 1x** to the Microhard radio via **RS232** as detailed in the Microhard pDDL900-ENC external - Integration examples section of the **1x Hardware Manual**.
3. Once the configuration and connection is done, open **Veronte Link** and configure the **TCP-SERVER** connection in the **Connection menu**.

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



Add new connection

4. Then, the configurable parameters must be entered.



New connection

Type *: TCP-SERVER

TCP-SERVER configuration

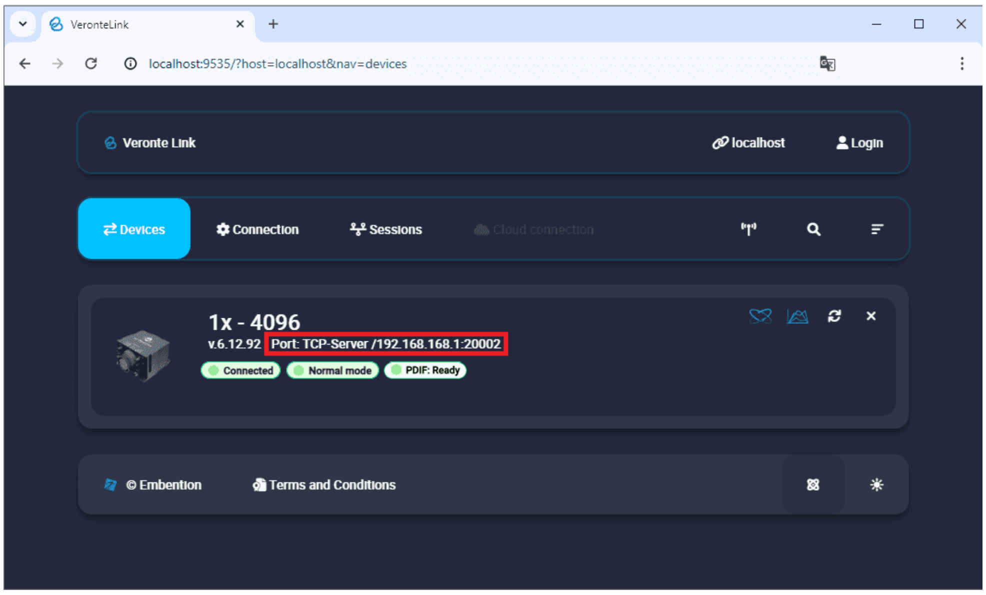
Port: 20002

Buffer size: 300

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:



1x unit correctly connected

TCP-CLIENT connection

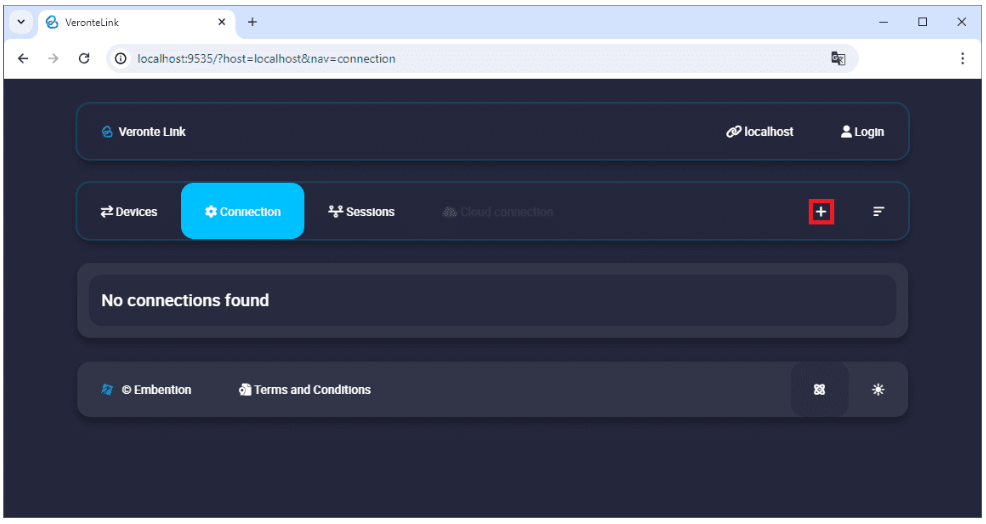
Ethernet configuration

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 "**+**":

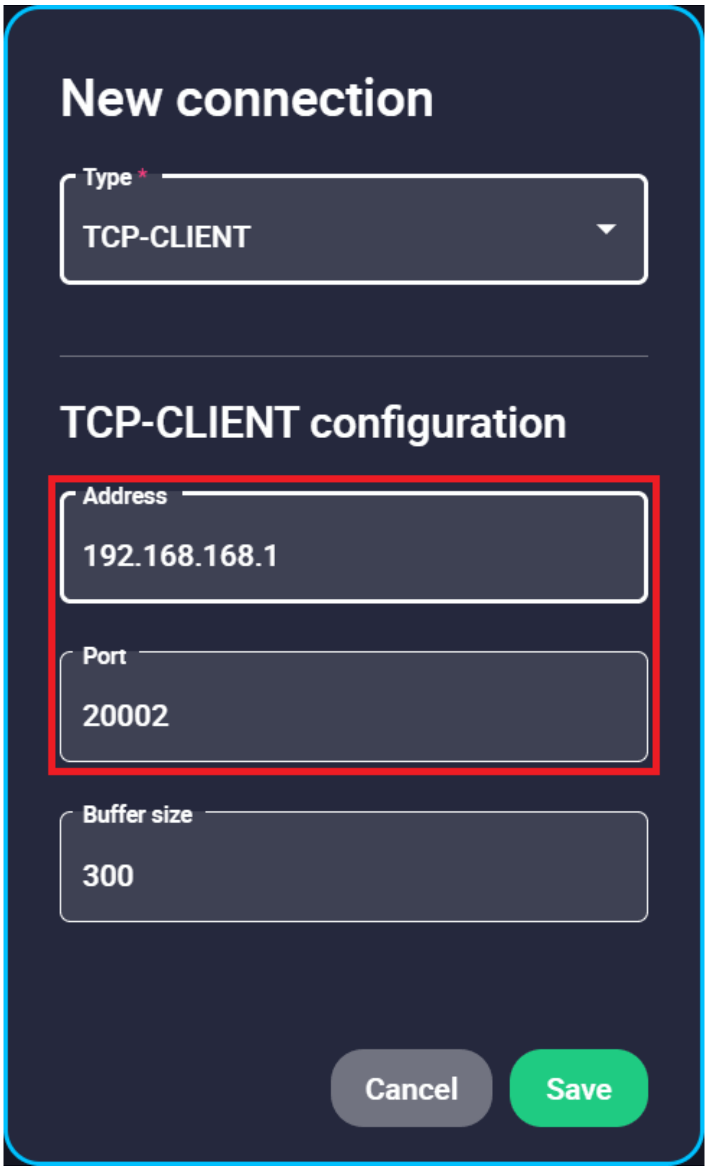


localhost:9535/?host=localhost&nav=connection

No connections found

Add new connection

4. Then, the configurable parameters must be entered.



New connection

Type *: TCP-CLIENT

TCP-CLIENT configuration

Address: 192.168.168.1

Port: 20002

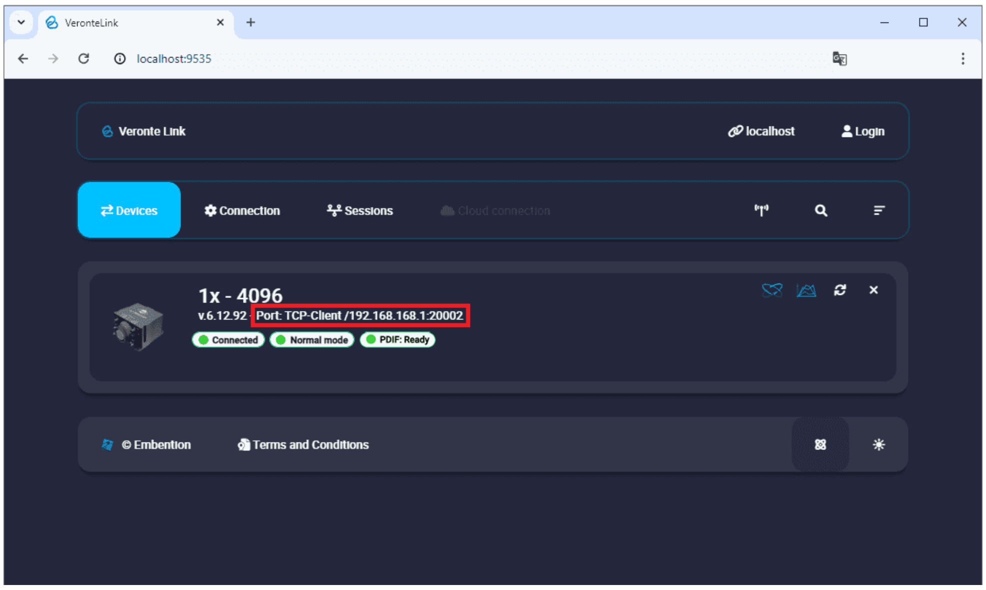
Buffer size: 300

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:



1x unit correctly connected

Troubleshooting

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

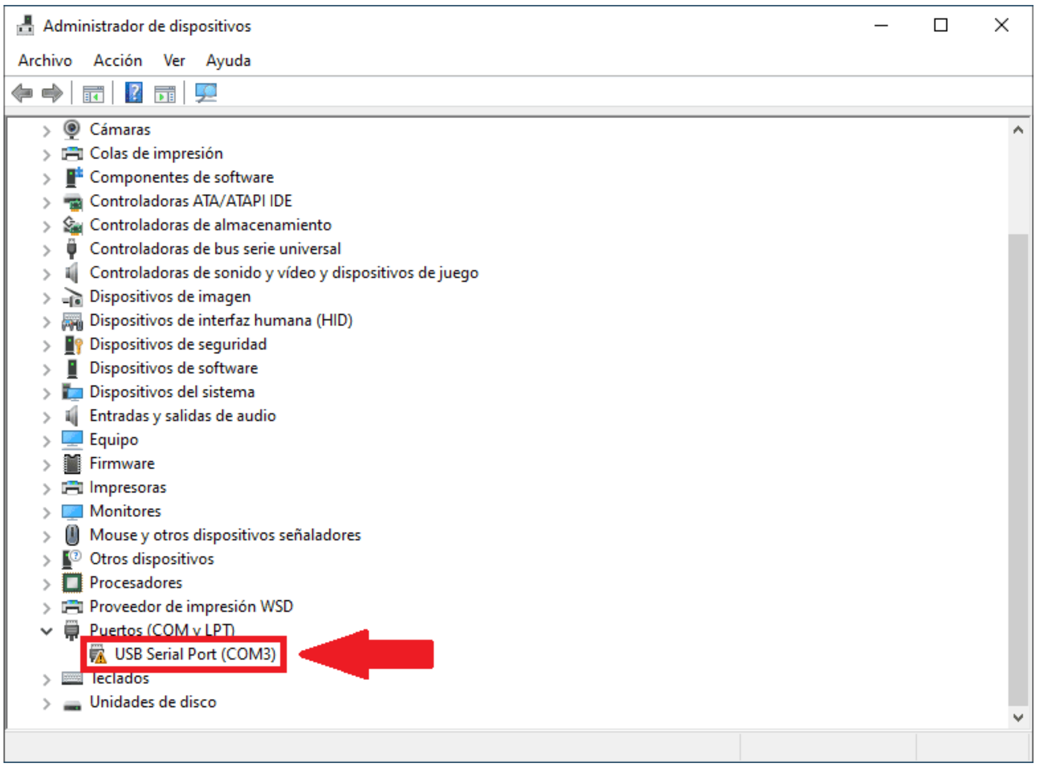
> ⚠ **Warning**
>
> Do not modify or delete manually any **Veronte Link** file. Copy them to a different path to send or analyze.

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**.

Comm Port error in Windows Device Manager

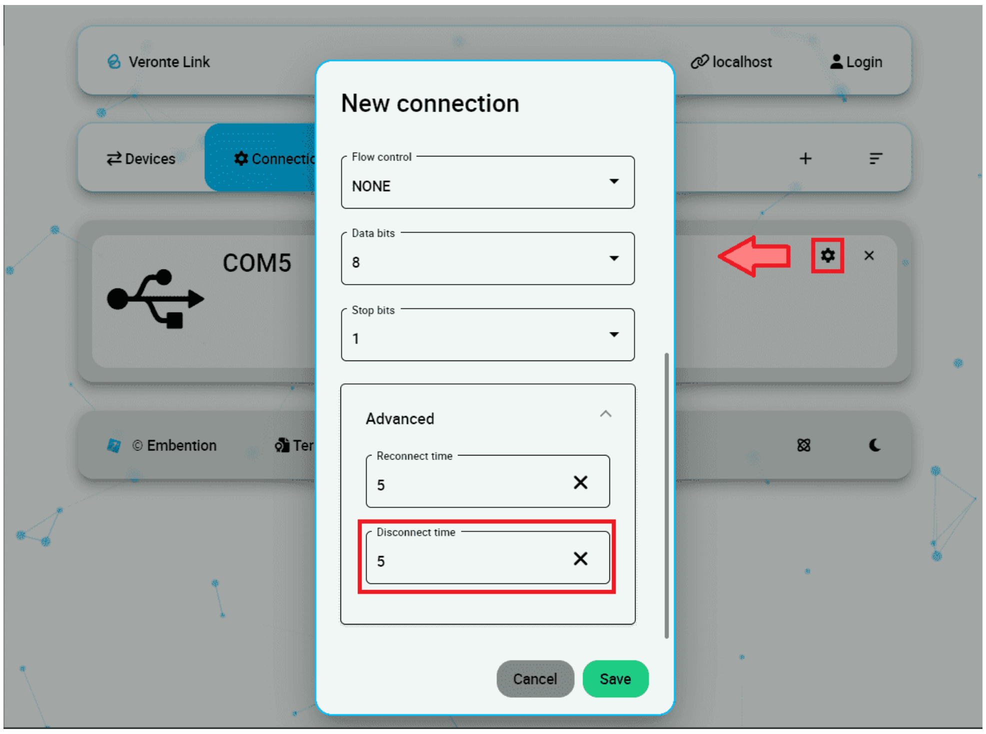
If the following Windows Comm Port error occurs:


Administrador de dispositivos
Archivo Acción Ver Ayuda
Cámaras
Colas de impresión
Componentes de software
Controladoras ATA/ATAPI IDE
Controladoras de almacenamiento
Controladoras de bus serie universal
Controladoras de sonido y vídeo y dispositivos de juego
Dispositivos de imagen
Dispositivos de interfaz humana (HID)
Dispositivos de seguridad
Dispositivos de software
Dispositivos del sistema
Entradas y salidas de audio
Equipo
Firmware
Impresoras
Monitores
Mouse y otros dispositivos señaladores
Otros dispositivos
Procesadores
Proveedor de impresión WSD
Puertos (COM y LPT)
USB Serial Port (COM3)
Teclados
Unidades de disco

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**.



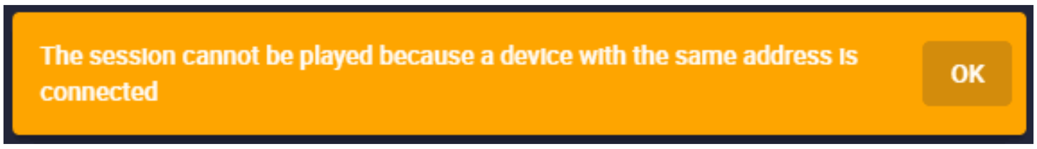
Flow control: NONE
Data bits: 8
Stop bits: 1
Advanced
Reconnect time: 5
Disconnect time: 5

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.

Error when replaying a session

If the following error message appears when attempting to replay a session:



The session cannot be played because a device with the same address is connected

OK

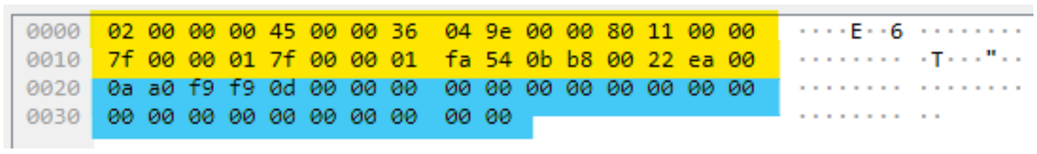
Error message

It is often due to users trying to replay a previous session from the device that is **currently connected**, which is not possible as indicated in the Sessions section.

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.



```
0000  02 00 00 00 45 00 00 36  04 9e 00 00 80 11 00 00   ····E··6 ········
0010  7f 00 00 01 7f 00 00 01  fa 54 0b b8 00 22 ea 00   ········ ·T···"··
0020  0a a0 f9 f9 0d 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 00 00 00 00  00 00                     ········ ··
```

Software Changelog

This section presents the changes between versions of **Veronte Link** application.

6.12.22

This section presents the changes between the previous software version **v.6.8** and **v.6.12.22**.

Improved

- Synchronization time with Veronte products.

6.14.28

This section presents the changes between the previous software version **v.6.12.22** and **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

6.14.60

This section presents the changes between the previous software version **v.6.14.28** and **v.6.14.60**.

Added

- Buffer size and TTL values configuration parameters for UDP/TCP connections
- Button to upload the device's PDIFs to the cloud (Upload PDIFs to cloud) in the Cloud connection tab

- Show UDP port in the Connection tab