

# TERMINAL USER INTERFACE FOR THE ETHERNET ENABLED DATA ACQUISITION FOR QPS

CERN  
TE-MPE SOFTWARE

Submitted by:  
Inés Terraza Palanca  
Supervised by:  
Marc-Antoine Galilee

3<sup>rd</sup> July - 7<sup>th</sup> September

**Contents**

**1 Introduction of EDAQ** **2**

**2 Structure of network** **2**

    2.1 Communication . . . . . 2

**3 Grafana implementation** **3**

**4 Terminal User Interface** **5**

**5 Conclusions** **8**

# 1 Introduction of EDAQ

Quench Protection System (QPS) is a part of superconducting infrastructure of LHC in charge of the safety and integrity of the magnets. The system is found along the entire accelerator to supervise and protect most of the superconducting elements. In managing a large and complicated setup like this, a critical element involves overseeing and precisely evaluating the overall condition and performance of the entire system. Additionally, conducting a thorough examination of protective events is essential to ensure the overall integrity of the superconducting infrastructure. The current system in operation has relatively constrained abilities when it comes to delivering data, resulting in a restricted overall data precision. Although this solution supports the existing infrastructure of superconductors, it exhibits significant flaws when upgrading for future applications of the High-Luminosity LHC. In this case, high-definition data is vital for the commissioning process, and also represents a valuable resource for scientific research purposes.

## 2 Structure of network

The network topology is presented in three key modules. Firstly, QPS devices are linked to a network switch serving as a central connection hub. This network switch is connected to a server responsible for tasks like data acquisition and the execution of control and supervisory functions.

- Server: The server is a key component that enables integration of the QPS devices with entire infrastructure. It provides a necessary isolation between Technical Network (TN) and field devices.
- Network switch: The network switch enables to implement a star topology network for QPS devices. It provides an aggregation of the traffic from individual devices to the server.
- Devices: QPS devices implement required safety functions for superconducting circuits.

### 2.1 Communication

Communication shall be compatible with regular Internet Protocol Version 4 (IPv4) traffic. Moreover, should provide reliable and unreliable modes depending on the different use cases, and the transport protocol used shall be widely accepted in the industry, in order to enable portability and flexibility of deployments. On the other hand, QPS field nodes do not provide extensive buffering capabilities, hence the transport protocol should be efficient and not overload them. To improve the performance, the server should handle the data buffering, especially when dealing with different amounts of incoming and outgoing data. It is expected that most of the data will flow in the direction of the QPS field nodes. Furthermore, the application communication protocol shall use User Datagram Protocol (UDP) as a transport protocol. The types of traffic expected are the following:

- Reliable traffic: Traffic requiring all packets to be acknowledged by the receiving device. This mode serves as a method for transmitting small amounts of data while ensuring it reaches its destination without fail. This mode is primarily utilized for carrying out control and configuration operations and is employed by both servers and clients.
- Buffered unreliable traffic: Traffic supporting retransmission requests. This mode allows for data to be resent if it doesn't get through, but there's no guarantee it will be delivered. It's a way to send data at a moderate speed with a lower chance of losing it compared to the completely unreliable traffic mode. This mode is useful for tasks like collecting data and is mainly used by QPS nodes when dealing with smaller amounts of data.

- Unreliable traffic: Traffic without buffering. This mode is meant to send continuous and high throughput data without guarantee of delivery. It enables to perform data acquisition, and it is primarily used by QPS nodes.

### 3 Grafana implementation

Grafana is an open-source platform used for monitoring and observability. It allows users to create and visualize real-time metrics, analytics, and logs through customizable dashboards and charts. The latter software was used to visualize more easily the data received from the QPS simulated devices. Prometheus is used in conjunction with Grafana to create a monitoring and alerting system for tracking the performance of the devices. This software is responsible for collecting and storing time-series data, often referred to as metrics. Hereafter, the Grafana dashboard is displayed:

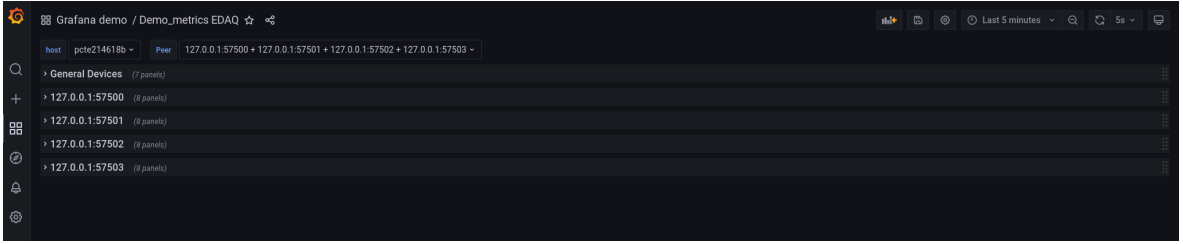


Figure 1: The dashboard is divided into different tabs for each device and one with the set of devices

The dashboard is divided into tabs that depend on the *host* of the device. If chosen, all the devices in that *host* can differ from their personal *Peer* (local address). The dashboard displayed then is divided into the specific devices and the general devices tab which includes a comparison with all of them.

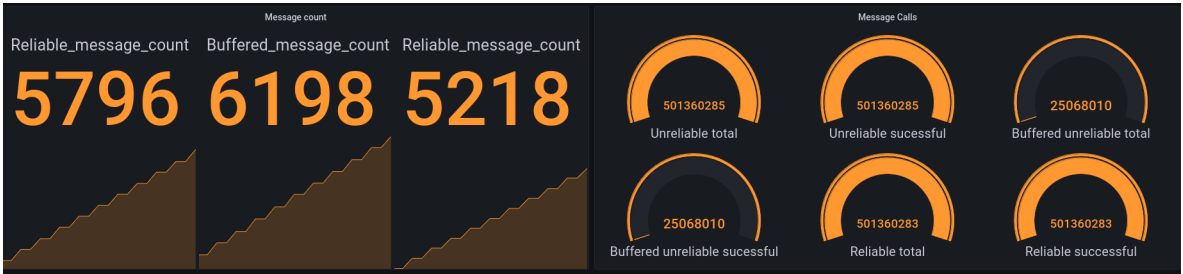


Figure 2: Message count and calls displayed in the tab of general devices

In the general devices tab, the messages count and calls of the three types of traffic are displayed. These are the sum of all the devices.

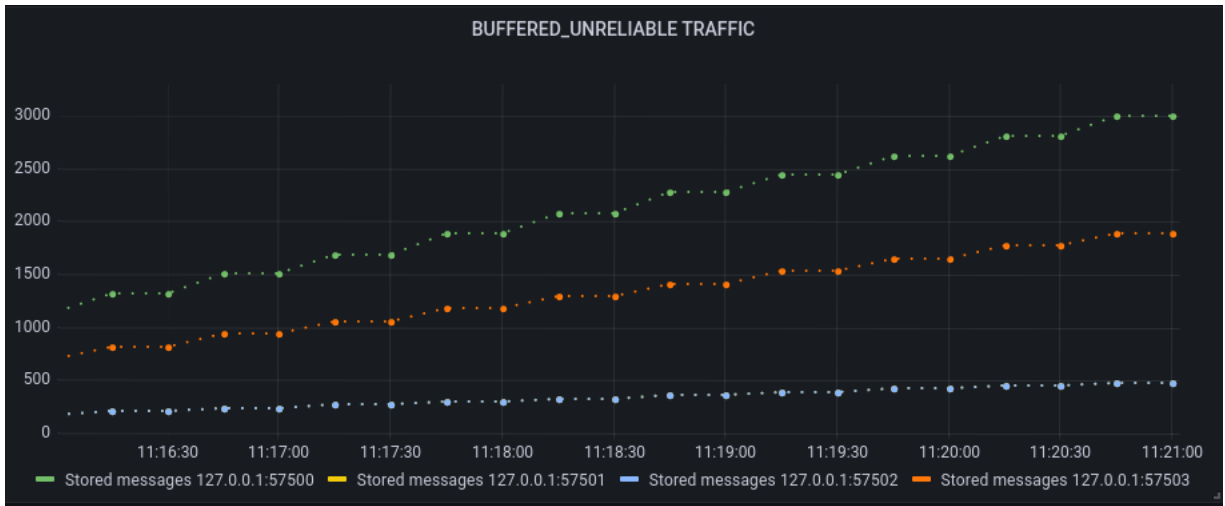


Figure 3: Stored messages for the different devices for the buffered unreliable traffic

Also, a comparison between the stored messages is shown. In different graphs for each type of traffic, as can be seen in Figure 3, the evolution over time for the messages is represented.

Moving on to the tab for each individual device, the dashboard is divided again into different sections. As it is displayed in Figure 4, the connection state of the device is seen and rises as a new connection is made. The bad counts and bad sources are also shown, but rarely triggered, for it only happens when a device sends badly formed messages or when a device sends messages to the address where another device is already connected. Furthermore, the stored messages for each type of traffic are shown again, this time only for one device.

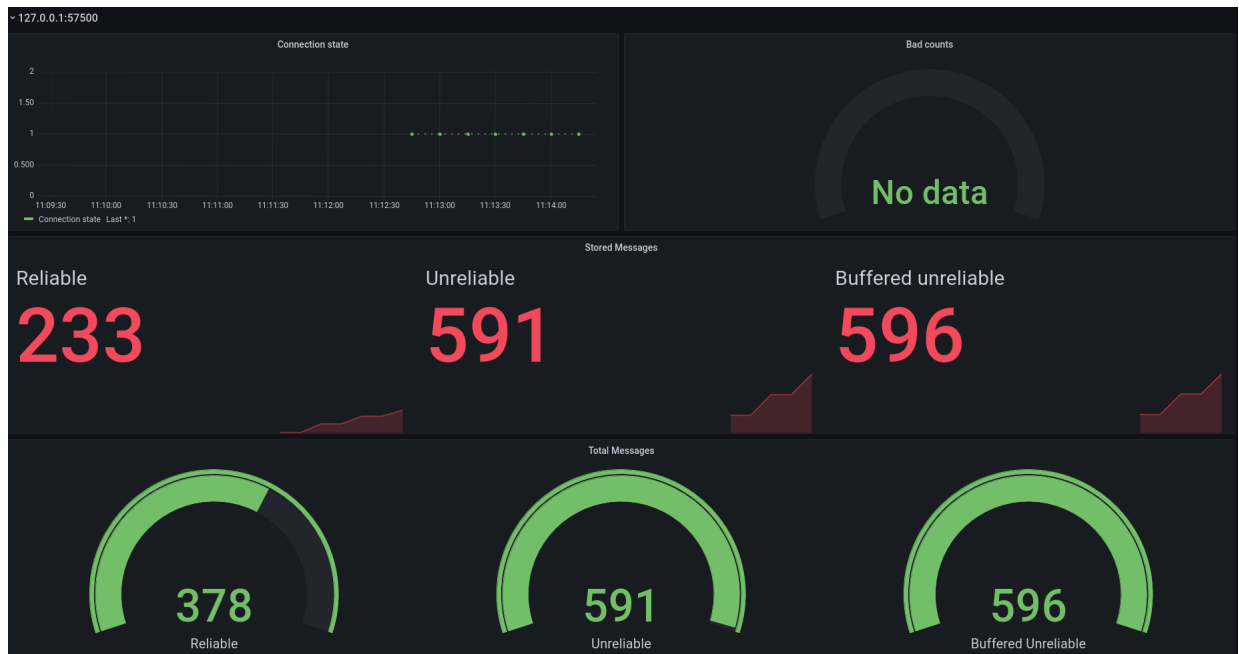


Figure 4: Tab of a single device

Lastly, different boards are designed for distinct types of traffic. For instance, the buffered unreliable traffic displays the retransmissions, nack count, expired nack count... The latter information is displayed when some information sent to the server is lost on the way and the server asks for it again. A similar process happens with reliable traffic, which has its own board. The unexpected nack count and ack count are also displayed when a device does not reply in time to a reliable traffic message. Another example is the ack drop count when a device does not reply at all to a reliable traffic message and the server stops expecting a reply. At the end of the dashboard, the messages out of order are also displayed. It should be pointed out that some metrics are especially hard to trigger and therefore are not displayed. This is the case for message errors for each type of traffic, which are never triggered.



Figure 5: Messages out of order of a single device

## 4 Terminal User Interface

TUI serves as a platform for constructing terminal-based user interfaces. It offers multiple "backends" for rendering content on the terminal. These backends assume responsibility for the fundamental operations involving terminal interaction, like configuring characters and screen clearing. The latter was used for a better triggering of the metrics and design of the devices in a more user-friendly environment. Hereafter, the interface is explained.

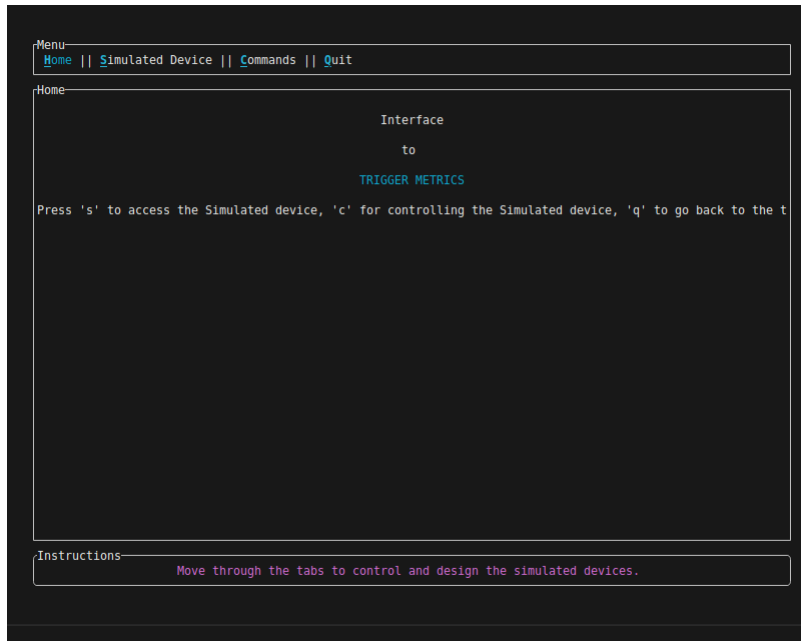


Figure 6: Home menu of the interface

The "Home" page explains how to move between tabs in the interface. If the first letter of each tab in the menu is pressed, it automatically changes the terminal to the selected tab.

Once in the simulated device tab, one can design the device by selecting each flag, the information is displayed as the flag is selected. As can be seen in Figure 7, pressing 'e' will bring it to edit mode in order to modify the default mode of each flag. Additionally, by pressing 'Enter' in the edit mode, the information is flashed. Finally, to create the device, one must press 'Enter'. Although the latter does not mean the device will be activated, in order to activate it one must move to the command tab.

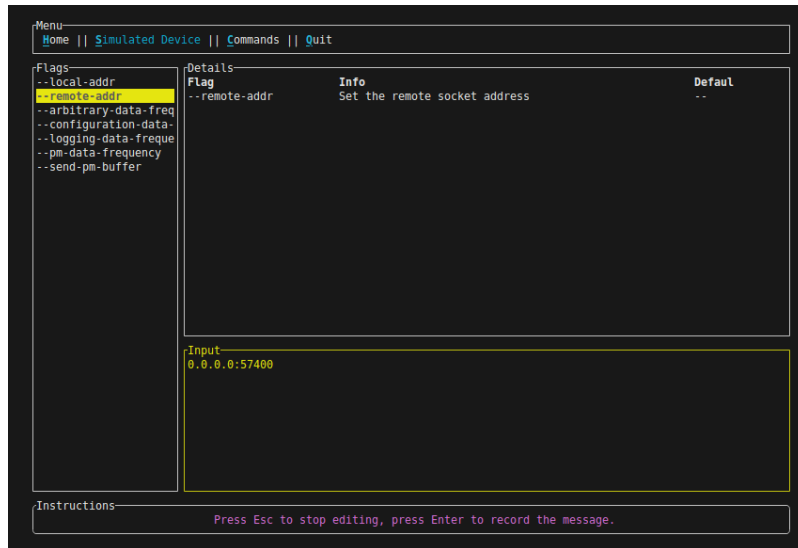


Figure 7: Simulated device tab

In this tab, Figure 8, the device is activated with 'Start' through a port to the server, and the metrics can begin to be seen. With 'Stop' the device goes to idle and can be activated again with 'Start' button. If 'Delete' is selected, the device is completely erased, and it should be redefined in the 'Simulated Device' tab if one wants to re-activate it. The 'Re-use' button sets in the 'Simulated Device' the details of the selected device for another use. On the other hand, to move between the activated devices, one must press the number of the device. If the device is in blue, it is selected, if it is in red, it's stopped, if otherwise is white it is not selected but activated. When a device is selected, its details are shown in the bottom part of the dashboard.



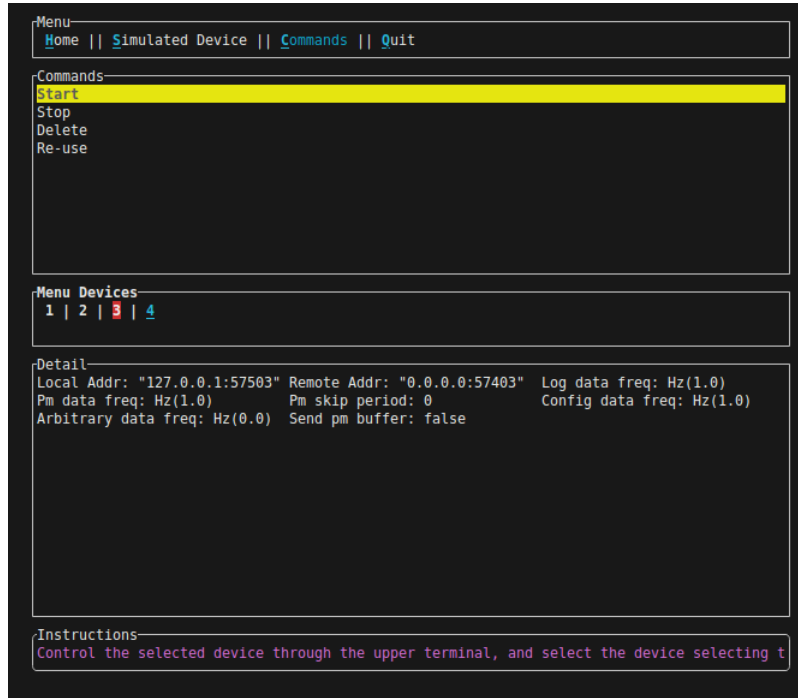


Figure 8: Command tab

## 5 Conclusions

In conclusion, the Quenching Protection System (QPS) stands as an indispensable component in the safety of the Large Hadron Collider (LHC) at CERN. My two-month engagement in this project has been a profoundly rewarding and self-fulfilling experience. The introduction of Grafana into the QPS infrastructure will enhance the visibility and comprehension of critical metrics, thereby expediting the detection of anomalies. Moreover, the development of the Terminal user interface will simplify the device interaction and server-based activations for users. This enriching experience has not only allowed for substantial growth with the Rust programming language but has also broadened my understanding of transport protocols and the intricacies of the LHC at CERN. Lastly, my involvement in this project has definitely enriched my knowledge and skill set in a multitude of ways, reinforcing the importance of continued advancements in safety systems.

## References

- [1] Podzorny, Tomas, CHristensen, Magnus. (2022). Ethernet enabled data acquisition for QPS.
- [2] (2023). The Large Hadron Collider, CERN Accelerating Science. <https://www.home.cern/science/accelerators/large-hadron-collider>.