Dashboard Container

User Guide

September 10, 2021

Contents

1 Introduction

The Dashboard container is based on a Debian OS and is intended to provide a tool to display periNODE sensor data with minimum configuration effort. For this scope, we use the powerful and user-friendly Grafana dashboards (`www.grafana.com`).

The container receives periNODE sensor data via MQTT and stores it inside an InfluxDB database (`www.influxdb.com`). Upon reception of MQTT messages, dashboards will automatically be created.

2 Dashboard Visualization

2.1 Web UI

The access of the dashboards is done by clicking the ***Open dashboard*** button in the Dashboard container Web UI:

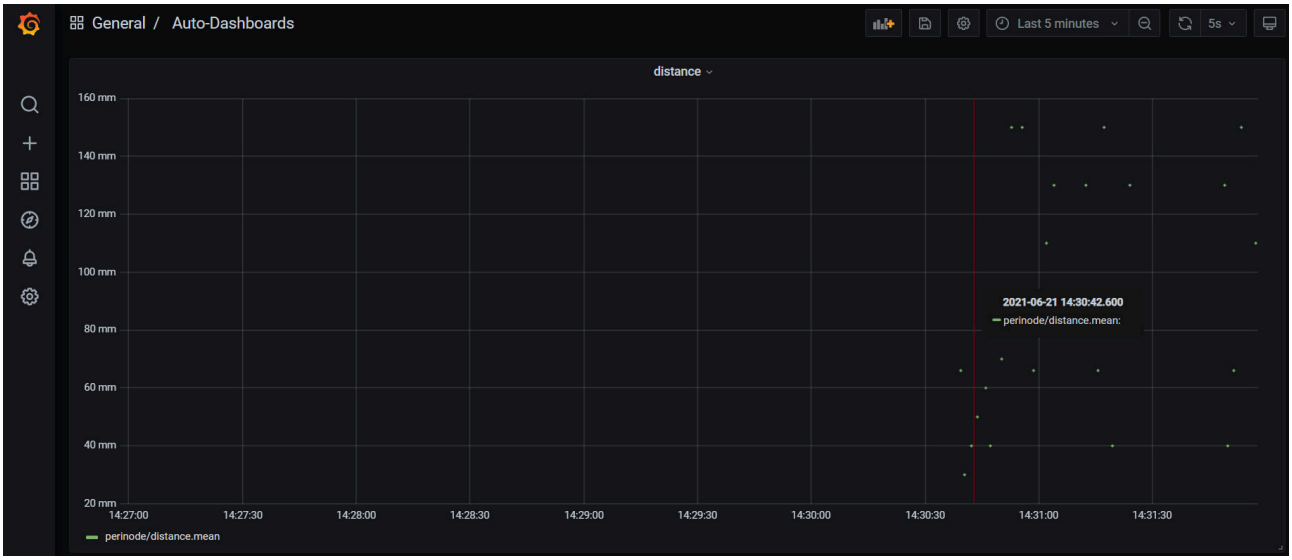

Figure 1: Grafana dashboard UI

2.2 Auto-Configuration

Dashboards are automatically created for each MQTT topic.

The dashboard panel title is created from the element name of the publishing periNODE, hence the suffix of the MQTT topic. Whereas the unit-value of the MQTT message data field is used as the y-axis for the dashboard. Therefore, if MQTT messages are received under the topic "demo/distance" like:

```
{"incarnation":6,"sequence_number":23646,"interface_type":"DIGITAL_IO_SINK",
 "data":[{"unit":"mm","distance":311},{},{}]}
```

the dashboard panel title would be **distance** and the y-axis would have the label **mm**.
The unit-value of the InfluxDB field is configured as unit of the y-axis of the dashboard.

Currently there is no cleanup of automatically generated dashboards. If desired, a ***Container Reset*** via periMICA Web UI can be made. See the context menu in Figure 2:

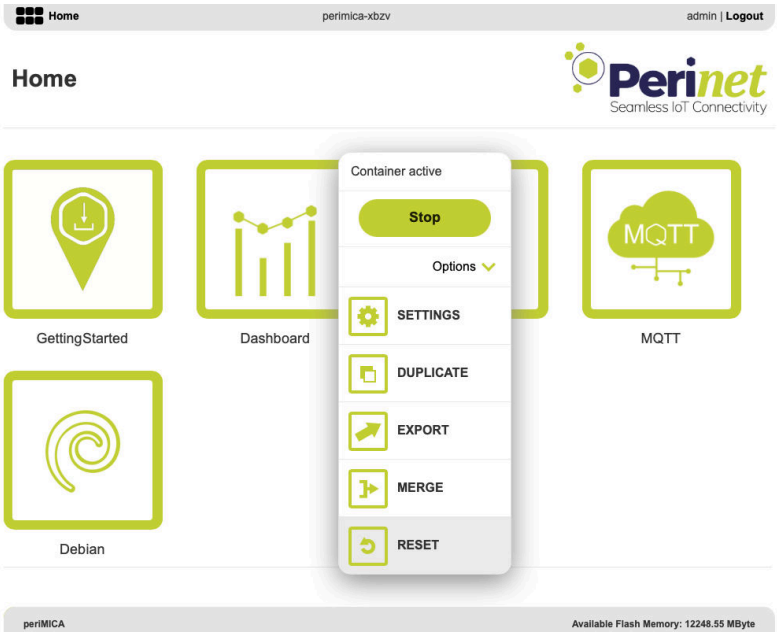

Figure 2: Dashboard container contex menu reset

2.3 Export CSV

Sensor data can be exported as CSV, while residing on the Grafana dashboard, by right-clicking on the panel title of the dashboard and then clicking on ***Inspect -> Data***. Within the Panel Inspector, finally download the CSV by clicking on Download CSV.

3 Container Configuration

3.1 MQTT Configuration

Two parameters under MQTT configuration need to be given, in order to use the functionality of the Dashboard container:

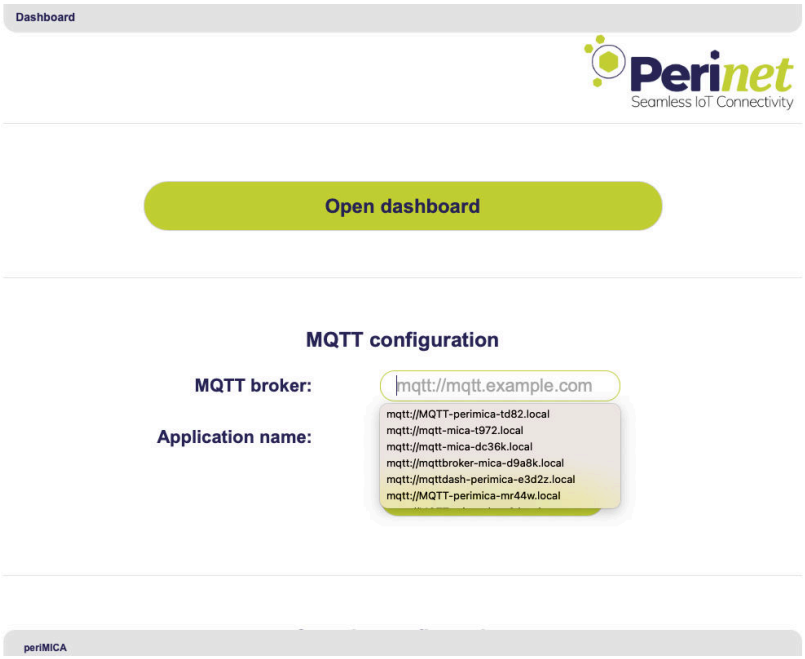

Figure 3: WebUI MQTT broker list of Dashboard container

The MQTT broker needs to be provided for the internal MQTT client of the Dashboard container. The Web UI automatically shows, in a drop-down list, the MQTT brokers that are found in the network via mDNS/DNS-SD protocols.

The second parameter defines the application name. The internal MQTT client subscribes to an MQTT topic that is derived from the application name (***application_name/#***). As an example, if the application name is **'demo'**, the MQTT client subscribes to all topics that obey **'demo/#'**.

After entering both ***MQTT broker*** and ***application name***, please make sure to click the ***Push configuration*** button.

3.2 Security Configuration

The security configuration is usually done automatically when using the PKI2go container. Please refer to the **PKI2go Container User Guide** [1] for a detailed explanation of the security features and the certificates needed.

However, the security configuration can also be done separately or manually, using the provided web-based user interface.

Initial Self Signed Certificate

During the container installation, an initial self-signed certificate is created automatically. The first access to a new container will be authenticated by this new certificate and security warnings are expected on client side. Before configuring the security in the container, the security warnings can be ignored.

Certificates Configuration

The Web UI provides input sections for the three certificates: *Host certificate*, *Root certificate* and *Client certificate* that can be configured in the container.

The certificate encoded and visible in the text area of each certificate is the current stored certificate. If the text area is empty, no certificate has been stored.

The container accepts X.509 certificates, which have been encoded with the PEM format (Base64 ASCII). Usually, the encoding scheme is reflected in the extension `.pem`, but `.crt`, `.cer` and `.key` have also been observed using this scheme.

The *Host certificate* as well as the *Client certificate* are expected to be uploaded with concatenated corresponding private key at the end. A *Root certificate* is expected to be uploaded without the private key.

Enforce mTLS access

Enabling the mTLS feature forces any remote client to authenticate towards the periMICA container with a valid *Client certificate*. The *Client certificate* will be validated with the stored *Root certificate*.

Note: *Before enabling 'Enforce mTLS access' ensure that a valid* Root certificate *has been stored.*

A user role is expected to be encoded in the client certificate. The user role will be used to implement a role based access control mechanism (RBAC). The following roles/capabilities are supported:

- ***admin***: has full read/write access to the container with no restrictions.
- ***super***: can configure the application parameters but **not** the security related.
- ***reader***: read-only access

With mTLS enabled in the container, only clients with valid certificates will be allowed the access, according to the encoded user role.

For more details on how to generate certificates, please refer to `https://docs.perinet.io`.

Security Reset

The **Reset security** button provides the option to reset the security configuration.

This operation will create a new self-signed host certificate and remove any other certificates. The previously configured *Root certificate* and *Client certificate* will be lost.

The mTLS will be disabled after a security reset, which means that the access to the container is not protected anymore.

Note: *Make sure to have a backup of important information before resetting security.*

4 Contact & Support

For customer support, please call us at **+49 30 863 206 701** or send an e-mail to *support@perinet.io*.

For complete contact information visit us at `www.perinet.io`

References

[1] *PKI2go Container User Guide*

Revision History

Revision	Date	Author(s)	Description
1.0	September 10, 2021	Christian Koehler	Initial release