MQTT Container

User Guide

September 10, 2021

Contents

1 Introduction

The MQTT container is based on a Debian OS and its intended scope is to provide an MQTT broker functionality with TLS support. The ***mosquitto*** library (`https://mosquitto.org/`) is used for serving the necessary software tools.

The MQTT container integrates perfectly into Perinet's security solution. Please refer to `https://docs.perinet.io` for a detailed explanation of our security concepts.

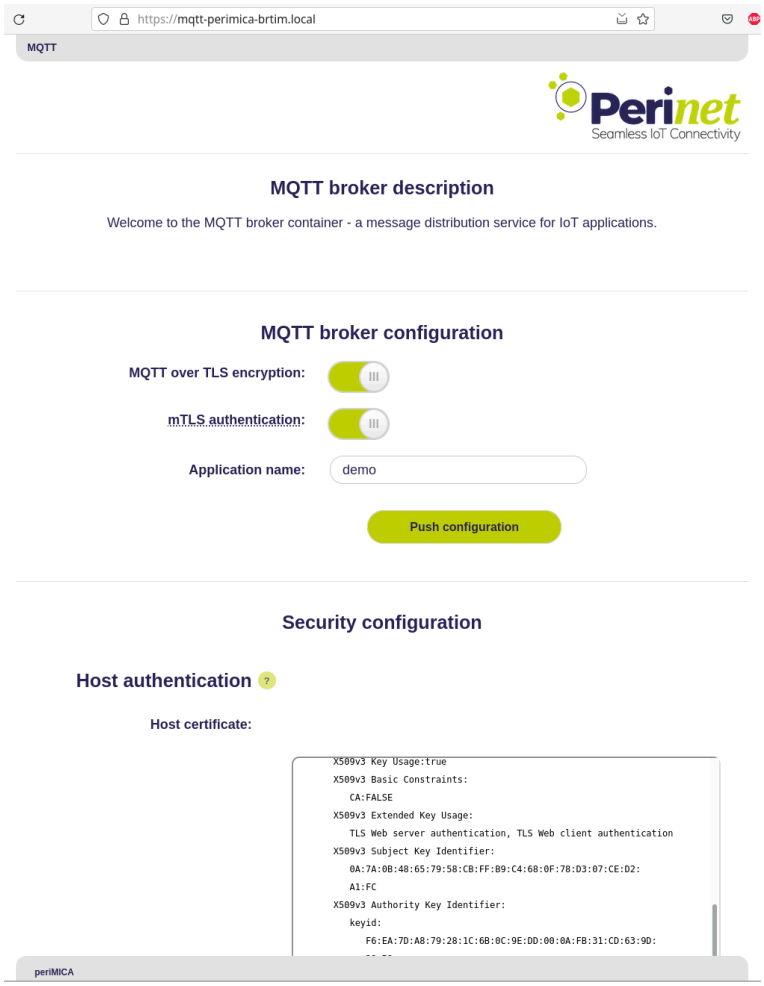

Figure 1: MQTT container

2 Container Configuration

2.1 MQTT Broker Configuration

By default, the MQTT container listens on secure MQTT port `8883` with mTLS authentication disabled.

In order to switch ***MQTT over TLS encryption*** or ***mTLS authentication***, change the corresponding button from either ***off*** to ***on*** or the other way around.

The ***Application name*** defines a subset of IoT devices that belong to one application. It allows to structure elements that belong together and also defines the prefix of the MQTT topics under which the components publish information. The MQTT protocol allows the use of this prefix as general topic to subscribe to all sensor data that belongs to the same application (demo/# in Figure 2).

After the settings are made, activate them by clicking on the ***Push configuration*** button:

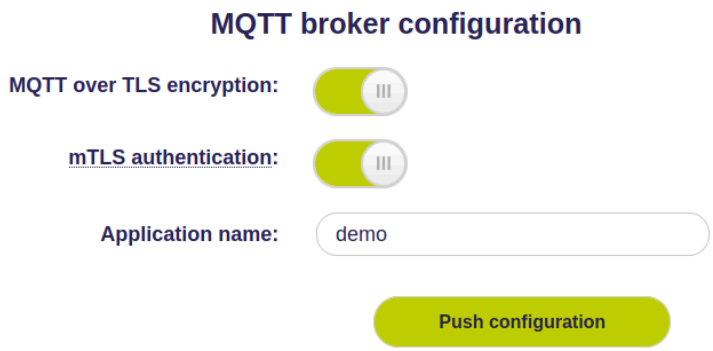

Figure 2: MQTT broker configuration

Disabling ***MQTT over TLS encryption*** will automatically restart the MQTT broker listening to port `1883`.

Enabling ***mTLS authentication*** will check the MQTT client certificates for use of the same root CA (Certification Authority) as the MQTT broker. It is recommended to use this option after setting up an application in the PKI2go container.

2.2 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 two certificates, the *Host certificate* and the *Root 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* is 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.*

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* 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.*

MQTT broker and the MQTT container web server share the same Root and Host certificates.

2.3 Service Discovery

The MQTT broker service can be found in the network by using ***mDNS/DNS - Service Discovery***, e.g. using the Debian container or any other Linux shell:

```
avahi-browse -tpk _mqtt._tcp
```

2.4 Graphical Tools

Graphical tools for service discovery are available, like *Bonjour Browser* for Windows or *avahi-discover* for Linux.

3 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	Dilmari Seidel Heuer	Initial release