



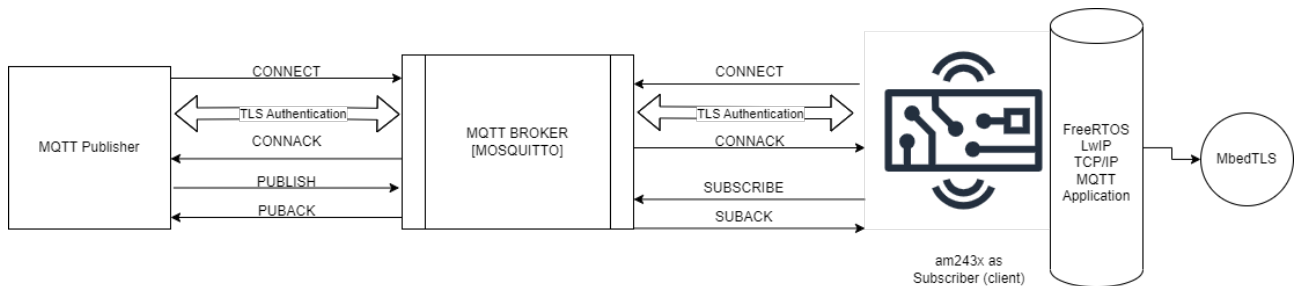
MQTT

Document Version : 6
Document Owner : Shaunak Deshpande

1 Introduction

HTTP	MQTT
Allows unidirectional message transfer. Direct server to client communication	Allows bi-directional message transfer. Communication between publisher and subscriber through the broker
Request response model	Publisher Subscriber model
No features of buffer in default HTTP	Has a Buffer, Last will & Testament,
More overhead since new connections need new headers	Less overhead since the same connection is not closed until the end.
Document centric	Data centric
No choice of QoS.	3 levels of QoS, at most, at least, exactly once.

Sample working of the example:



Using one of the above protocols for communication hooked with MbedTLS will ensure encrypted data transfer by the TLS/ SSL protocol.

The below steps enable the MQTT subscriber (client):

2 Software and Hardware requirements:

Software requirements (ADDITIONAL):

- 1) Mosquitto (open-source MQTT broker)

Hardware requirements:

- 1) **Ethernet**

2.1 Setup the Broker:

```
sudo ifconfig {ethernet-interface-name} 192.168.50.1 netmask 255.255.255.0

# To check ethernet-interface-name, execute the following command:

ifconfig -a
```

2.1.1 Clone mosquitto from git:

```
git clone https://github.com/eclipse/mosquitto.git

cd mosquitto/

make WITH_CJSON=no WITH_DOCS=no
```

2.1.2 Test the working of broker, publisher and subscriber (client) in different terminals:

```
# To start the mosquitto broker:
./src/mosquitto -c my_mosquitto.conf

# To subscribe to the broker from terminal:
LD_LIBRARY_PATH=./lib ./client/mosquitto_sub -h 192.168.50.1 -u pc_sub -P 1234 -t
topic_qos1

# To publish to the broker:
```

```
LD_LIBRARY_PATH=./lib ./client/mosquitto_pub -h 192.168.50.1 -u pc_pub -P 1234 -t  
topic_qos1 -m "hello world"
```

2.1.3 Broker Configuration:

Create new username and password for broker, publisher, subscriber

```
# For simplicity we are using "1234" as password here  
  
# Add the board user and password:  
./apps/mosquitto_passwd/mosquitto_passwd -c my_password_file am243x_evm  
  
# Add the subscriber user and password  
./apps/mosquitto_passwd/mosquitto_passwd my_password_file pc_sub  
  
# Add the publisher user and password  
./apps/mosquitto_passwd/mosquitto_passwd my_password_file pc_pub
```

2.1.4 ACL File changes:

Copy the sample ACL file and add the above created users, and topic, for simplicity, we are using the topic "topic_qos1", which is also defined in lwip mqtt application

```
cd mosquitto/  
cp aclfile.example my_aclfile
```

Add the users in the copied file mosquitto/my_aclfile:

```
# am243X_evm connected to the linux PC  
user am243x_evm  
topic topic_qos1  
  
# Linux PC - Subscriber  
user pc_sub  
topic topic_qos1  
  
# Linux PC - Publisher  
user pc_pub  
topic topic_qos1
```

2.1.5 Mosquitto configuration file:

Copy the existing mosquitto.conf file:

```
cp mosquitto.conf my_mosquitto.conf
```

In the newly copied file, make the following changes as shown in the diff:

```
a0503581@a0503581:~/mosquitto$ diff mosquitto.conf my_mosquitto.conf
37c37
< #per_listener_settings false
---
> per_listener_settings true
234c234
< #listener
---
> listener 1883 192.168.50.1
257c257
< #bind_interface
---
> bind_interface enp0s31f6
532c532
< #allow_anonymous false
---
> allow_anonymous false
550c550
< #password_file
---
> password_file my_password_file
613c613
< #acl_file
---
> acl_file my_aclfile
a0503581@a0503581:~/mosquitto$ S
```

2.2 Steps to enable MQTT with TLS:

3 Sample output:

```

=====
COM9 ×
=====
ENET LwIP App
=====
Enabling clocks!
Mdio_open: MDIO Manual_Mode enabled
EnetPhy_bindDriver: PHY 15: OUI:080028 Model:0f Ver:01 <-> 'dp83869' : OK
EnetPhy_bindDriver: PHY 3: OUI:080028 Model:0f Ver:01 <-> 'dp83869' : OK
PHY 3 is alive
PHY 15 is alive
Starting lwIP, local interface IP is 192.168.50.20
Host MAC address-0 : 70:ff:76:1e:26:fe
[LWIPIF_LWIP] NETIF INIT SUCCESS
[LWIPIF_LWIP] Enet has been started successfully
status_callback==UP, local interface IP is 192.168.50.20
UDP server listening on port 5001
Icsvg_handleLinkUp: icsvg1: Port 1: Link up: 100-Mbps Full-Duplex
link_callback==UP
MQTT client "test" connection cb: status 0
MQTT client "test" request cb: err 0
MQTT client "test" request cb: err 0
5.158s : CPU load = 4.51 %

10.158s : CPU load = 4.0 %
15.158s : CPU load = 3.90 %
20.158s : CPU load = 4.0 %
25.158s : CPU load = 3.91 %
30.158s : CPU load = 4.0 %

5.158s : CPU load = 4.51 %

10.158s : CPU load = 4.0 %
15.158s : CPU load = 3.90 %
20.158s : CPU load = 4.0 %
25.158s : CPU load = 3.91 %
30.158s : CPU load = 4.0 %
MQTT client "test" publish cb: topic topic_qos1, len 11
MQTT client "test" data cb: len 11, flags 1
35.158s : CPU load = 4.3 %

40.158s : CPU load = 3.91 %
45.158s : CPU load = 4.0 %

```