MQTT Client Application Example for SIMATIC S7-1500

www.broker4iot.com

23.06.16

VERSION: 1.1

Inhaltsverzeichnis

1 HISTORIE 4
2 MQTT characteristics 5
3 Application implementation 5
4 Main features 5
5 Limitations 5
6 Required components 6
7 MQTT_Client Function Block description 6
Input parameters 7
Output parameters 7
MqttConnectPacket data type 8
MqttPublishPacket data type 8
MqttSubscribePacket data type 8
MqttUnsubscribePacket data type 8
ErrorID 9
ErrorSource 9
8 Integration of the MQTT_Client into user project 10
9 Security function configuration 13
10 Literature and links 13

HISTORIE

Autor	Version	Datum	Änderung
@AR	01.01.00	25.03.21	Ersterstellung

MQTT characteristics

MQTT (Message Queuing Telemetry Transport) is an OASIS standard publish-subscribe-based messaging protocol. It works on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. These characteristics make it ideal for use in many situations, including constrained environments such as for communication in Machine-to-Machine (M2M) and Internet of Things (IoT), contexts under unreliable networks with low bandwidth and high latency.

The MQTT protocol stands out with the following characteristics:

- Light-weight and bandwidth efficient
- Asynchronous publish/subscribe communication model over TCP/IP
- protocol, aimed at low complexity, low power consumption and low footprint implementations
- Decoupling of data producer (publisher) and data consumer (subscriber) trough topics
- Continuous session awareness
- Mechanism for notifying interested parties after a client has disconnected ungracefully
- Provide a Quality of Service (QoS) with three different reliability levels for data delivery
- Optional encrypted communication via SSL/TLS
- Authentication with username/password

Application implementation

This Application Example offers you a solution for implementing the MQTT client protocol into a SIMATIC S7-1500 controller.
This Application Example provides you with a MQTT_Client function block that allows you to transmit MQTT messages to a broker (publisher role) and receive messages from it (subscriber role). It means that both Publisher and Subscriber are implemented in this MQTT_Client function block. Optionally, the communication can also be secured via a TLS connection.

Main features

- The main features of this Application Example are:
- Both Publisher and Subscriber are implemented
- Subscribe and unsubscribe for max. ten topics at a time (expandable)
- Handle max. ten incoming messages from the broker, arrived on the same packet (expandable)
- Demo project for HMI Comfort Panel is integrated
- Full QoS 0/1/2 handling mechanism
- Open code without know-how protection, free for use/modification
- Version for S7-1200 is also possible with minimum change of codes

Limitations

This MQTT_Client function block is compliant with the MQTT V3.1.1, however with the following limitations:

- Nevertheless the max. MQTT payload length is 256MB according to the MQTT specification and the SIMATIC S7-1500 TCP/IP max. data length is 64KB per transmission, the current version of MQTT_Client function block limits the data length per transmission to **500 bytes**. Furthermore, the max. length of a single message to be published to and received from the broker is limited to **254**

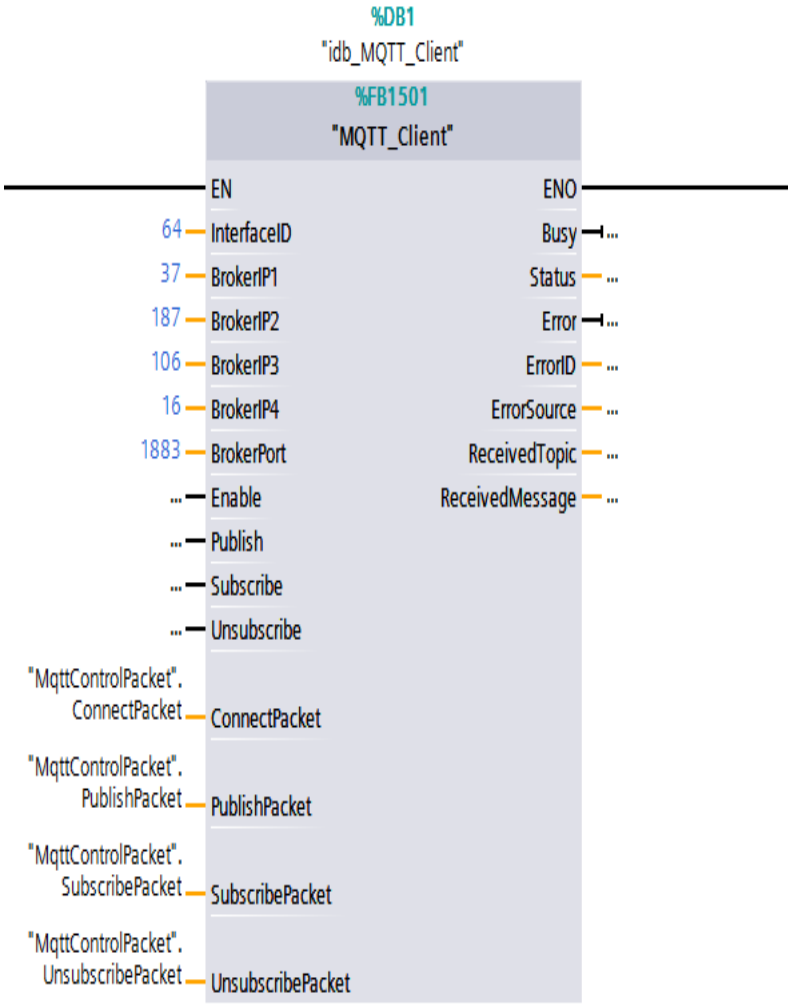
characters (standard String data type).

- Session State mechanism according to MQTT specification [MQTT-3.1.2-4] is not implemented. The session state is not stored and after resuming connection from an ungraceful disconnection, the action according to MQTT specification [MQTT-3.1.2-5] will not be executed. Therefore, the Clean Session bit of the CONNECT control packet is set to "true" unchangeably.
- The specified UTF-8 format according to MQTT specification [MQTT-1.5.3-1], [MQTT-1.5.3-2] and [MQTT-1.5.3-3] is not validated while compile the control packets.
- The IP and Port addresses are not validated before sending the CONNECT control packet

Required components

- Any SIMATIC S7-1500 CPU with FW 2.0 and above
- TIA Portal V14 SP1 (V15 not yet tested)
- A MQTT broker

MQTT_Client Function Block description



Input parameters

Parameter	Data type	Function
InterfaceID	UInt	Hardware identifier of the PROFINET interface of the CPU.
BrokerIP1	Byte	IP_v4 address of the broker.
BrokerIP2	Byte	
BrokerIP3	Byte	
BrokerIP4	Byte	
BrokerPort	UInt	Port address of the broker.
Enable	Bool	Positive edge, initialize TCP connection and send MQTT CONNECT control packet to the broker. Negative edge, send MQTT DISCONNECT control packet to the broker and disconnect TCP & MQTT connections.
Publish	Bool	Positive edge, send MQTT PUBLISH control packet to the broker. **Automatically** set back to "false" once completed the publishing.
Subscribe	Bool	Positive edge, send MQTT SUBSCRIBE control packet to the broker. **Automatically** set back to "false" once completed the subscription.
Unsubscribe	Bool	Positive edge, send MQTT UNSUBSCRIBE control packet to the broker. **Automatically** set back to "false" once completed the unsubscription.
ConnectPacket	"MqttConnectPacket"	Data area of CONNECT packet.
PublishPacket	"MqttPublishPacket"	Data area of PUBLISH packet.
SubscribePacket	"MqttSubscribePacket"	Data area of SUBSCRIBE packet.
UnsubscribePacket	"MqttUnsubscribePacket"	Data area of UNSUBSCRIBE packet.

Output parameters

Parameter	Data type	Function
Busy	Bool	True, while a MQTT control packet is being sent to the broker.
Status	Byte	Status of MQTT client: Bit 0: TCP connected Bit 1: MQTT connected Bit 2: Published successfully Bit 3: Subscribed successfully Bit 4: Unsubscribed successfully Bit 5: Ping responded Bit 6: DISCONNECT packet sent Bit 7: Message received from the broker
Error	Bool	True, if error is present.
ErrorID	Word	Cause of the errors, see the ErrorID table below
ErrorSource	Byte	Source of the errors, see the ErrorSource table below.
ReceivedTopic	Array[0..9] of String	Data buffer where the incoming published topic names from the broker are stored (expandable).
ReceivedMessage	Array[0..9] of String	Data buffer where the incoming

messages

MqttConnectPacket data type

Parameter	Data type	Function
cleanSession	Bool	True: discard any previous session and start a new one. Since this MQTT_Client does not support the MQTT session state by now, this bit is always set to "true".
willFlag	Bool	True: indicates that a will message is to be sent upon ungraceful client disconnection.
willQoS_level	Int	Indicates QoS level for the will message.
willRetain	Bool	True: the will message must be retained in the broker.
passwordFlag	Bool	True: if the broker requires a password for login. If userNameFlag is "false", passwordFlag will be reset automatically.
userNameFlag	Bool	True: if the broker requires a user name for login
keepAliveTime	Word	Max. time interval in seconds that can elapse between client transmission of control packets (Min. 2 seconds).
clientIdentifier	String[23]	The unique name with which the client identify itself at the broker.
willTopic	String[100]	If willFlag is "true", the topic for the last will must be present.
willMessage	String[100]	If willFlag is "true", the message for the last will should be present.
userName	String[20]	If userNameFlag is "true", the user name should be present.
password	String[20]	If passwordFlag is "true", the

MqttPublishPacket data type

Parameter	Data type	Function
DUP_flag	Bool	True: re-delivery attempt of a previous message.
QoS_level	Int	Quality of service for an application message
Retain	Bool	True: the message and its QoS must be stored in the broker and delivered to future subscribers of that topic.
PublishTopic	String	The topic to be published. Must not

MqttSubscribePacket data type

Parameter	Data type	Function
TopicFilter	Array[0..9] of String	The topics to be subscribed (expandable)
RequestedQoS	Array[0..9] of Byte	QoS level required for this subscribtion (expandable).

MqttUnsubscribePacket data type

Parameter	Data type	Function
TopicFilter	Array[0..9] of String	The topics to be unsubscribed (expandable).

ErrorID

ErrorID (W#16#...)	Description	MQTT Disconnected
16#8000~ 16#8999	Errors caused by T-Block commands (TSEND_C or TRCV)	Yes
16#9001	Watchdog timeout during MQTT packets transmission. The watchdog timer is fixed to 3 seconds.	Yes
16#9003	Receive wrong CONNACK fixed header from the broker	Yes
16#9004	Receive CONNACK error code from the broker	Yes
16#9011	Receive wrong PUBREC fixed header from the broker (QoS=2)	Yes
16#9012	Receive wrong PUBCOMP fixed header from the broker (QoS=2)	Yes
16#9013	Receive wrong PUBACK fixed header from the broker (QoS=1)	Yes
16#9021	Receive wrong SUBACK fixed header from the broker	Yes
16#9022	SUBACK Remaining Length (RL) exceeds the max. permissible length	Yes
16#9023	Receive inconsistent SUBACK packet ID from the broker	Yes
16#9024	Receive inconsistent SUBACK QoS level from the broker	Yes
16#9025	Receive inconsistent UNSUBACK packet ID from the broker	Yes
16#9030	Receive wrong UNSUBACK fixed header from the broker	Yes
16#9040	Receive wrong PINGRESP fixed header from the broker	Yes
16#9050	Incoming published message Remaining Length (RL) exceeds the max. permissible length	Yes
16#9051	Incoming message wrong QoS level (>2)	Yes
16#9052	Receive wrong PUBREL fixed header from the broker	Yes
16#9053	Receive inconsistent PUBREL packet ID from the broker	Yes
16#9054	Incoming topic length exceeds the buffer size	Yes
16#9055	Incoming message length exceeds the buffer size	Yes
16#9100	Wrong CONNECT WillQoS level (>2)	No
16#9101	PUBLISH topic name is empty	No
16#9102	Wrong PUBLISH QoS level (>2)	No
16#9103	All of the ten SUBSCRIBE topic names are empty	No
16#9104	MQTT is not connected	No
16#9105	Wrong SUBSCRIBE QoS level (<2)	No
16#9106	All of the ten UNSUBSCRIBE topic names are empty	No
16#9107	CONNECT Client Identifier is empty	No
16#9108	CONNECT WillFlag is "true" but WillTopic is empty	No

ErrorSource

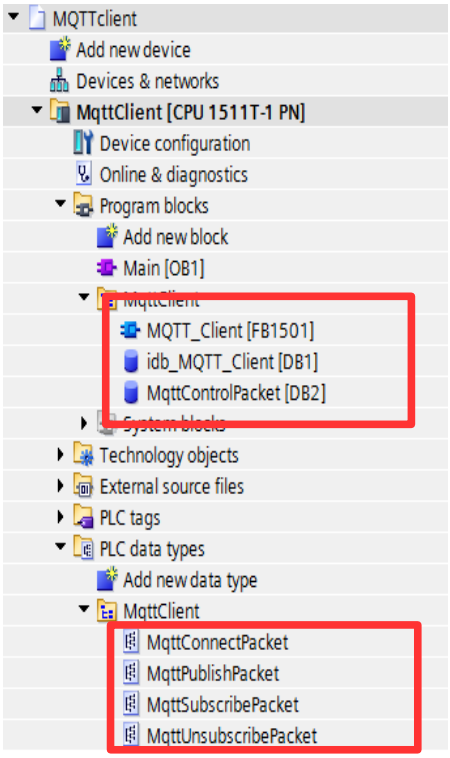
Error source	Value	Direction of flow	Description
TCP commands	16#00	-	See the STATUS parameter description of TSEND_C &

			TRCV for detail.
CONNECT	16#01	Client → Server	Request to connect to the broker
CONNACK	16#02	Server → Client	Connect acknowledgment
PUBLISH	16#03	Client ↔ Client	Publish message
PUBACK	16#04	Client ↔ Server	Publish acknowledgment
PUBREC	16#05	Client ↔ Server	Publish received (assured delivery part 1)
PUBREL	16#06	Client ↔ Server	Publish release (assured delivery part 2)
PUBCOMP	16#07	Client ↔ Server	Publish complete (assured delivery part 3)
SUBSCRIBE	16#08	Client → Server	Client subscribe request
SUBACK	16#09	Server → Client	Subscribe acknowledgment
UNSUBSCRIBE	16#0A	Client → Server	Unsubscribe request
UNSUBACK	16#0B	Server → Client	Unsubscribe acknowledgment
PINGREQ	16#0C	Client → Server	Ping request
PINGRESP	16#0D	Server → Client	Ping response
DISCONNECT	16#0E	Client → Server	Client is disconnecting
Subscribed message arrived	16#0F	Servo → Client	Receiving subscribed message from the broker

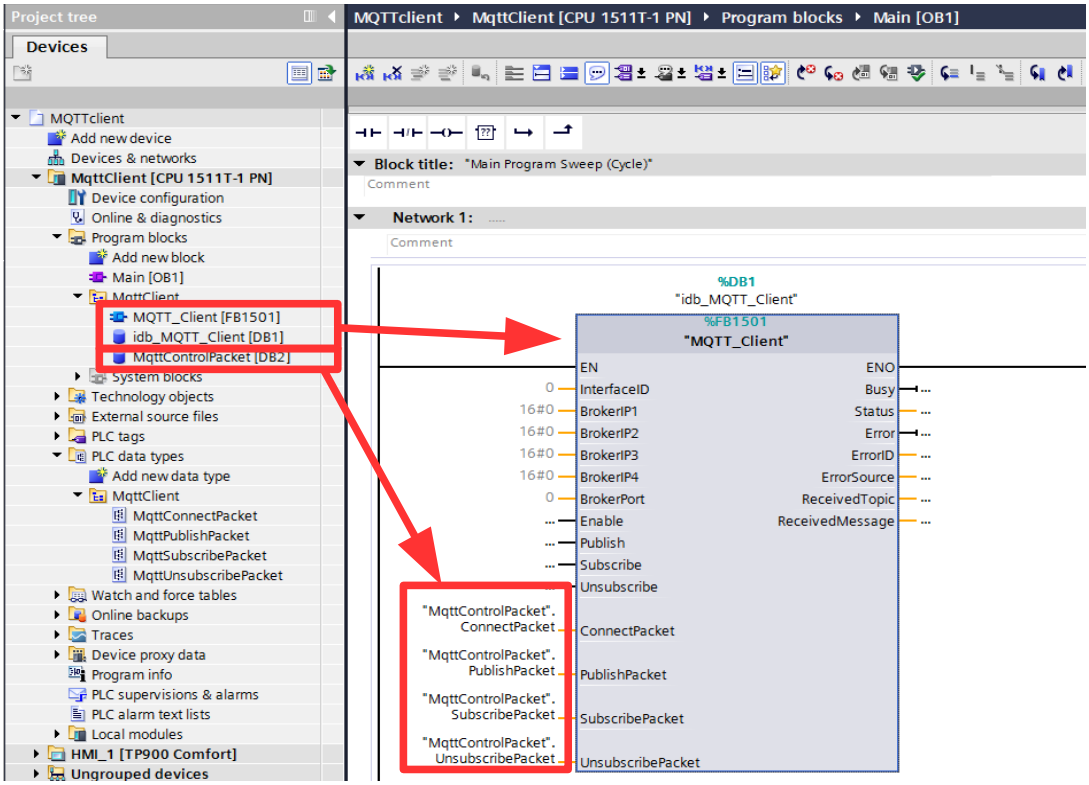
Integration of the MQTT_Client into user project

The MQTT_Client is not released as a library; therefore you should integrate it into your project simply by copy/paste method directly from the example project.
The blocks needed to be copied into your project are:

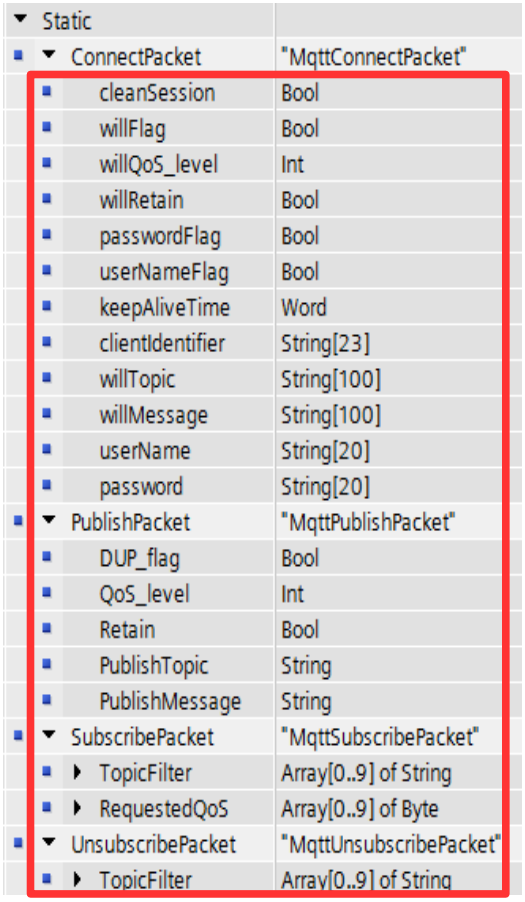
- Under Program blocks: The whole **MqttClient** group which contains MQTT_Client FB, idb_MQTT_Client data block and MqttControlPacket data block.
- Under PLC data types: The whole **MqttClient** group which contains ConnectPacket, PublishPacket, SubscribePacket and UnsubscribePacket data types.



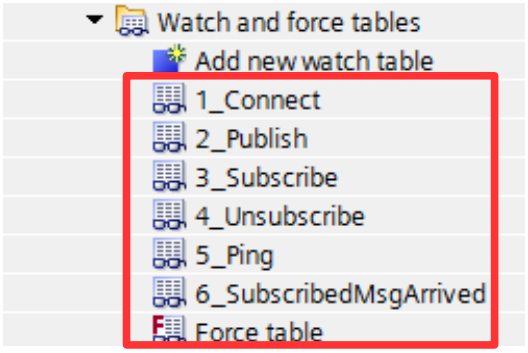
Once the abovementioned blocks have integrated into your project, you need to call up the MQTT_Client (this case, FB1501) in your program cycle (e.g. OB1), assign an instance data block to it (this case, DB1) and interconnect the input and/or output parameters to your preference (especially the Interface ID, the broker's IP and Port addresses and/or user name and password).



And then, set up the MQTT control packet parameters in the MqttControlPacket data block (this case, DB2) in your program or by means of the HMI interface.

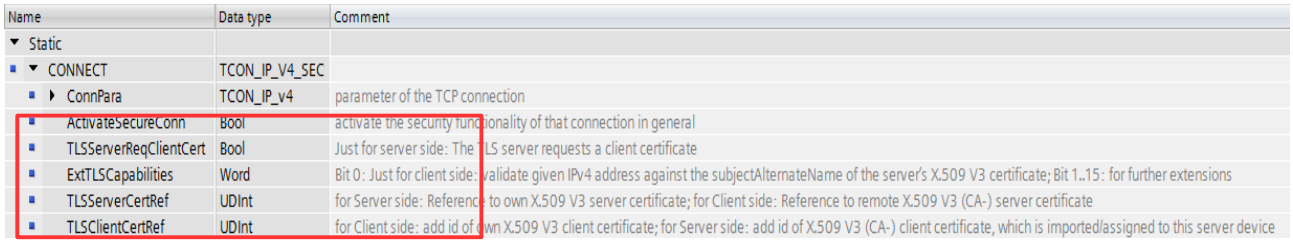


Now, you can compile and download the project to CPU for testing. Several ready-to-use watch tables are provided in the example project for easing the test.



Security function configuration

To activate the security functionality of the connection, the related parameters declared in the Static of the instance DB of MQTT_Client FB must be set according to your specifications.



Name	Data type	Comment
Static		
CONNECT	TCON_IP_V4_SEC	
ConnPara	TCON_IP_v4	parameter of the TCP connection
ActivateSecureConn	Bool	activate the security functionality of that connection in general
TLSServerReqClientCert	Bool	Just for server side: The TLS server requests a client certificate
ExtTLSCapabilities	Word	Bit 0: Just for client side: validate given IPv4 address against the subjectAlternateName of the server's X.509 V3 certificate; Bit 1..15: for further extensions
TLSServerCertRef	UDInt	for Server side: Reference to own X.509 V3 server certificate; for Client side: Reference to remote X.509 V3 (CA-) server certificate
TLSClientCertRef	UDInt	for Client side: add id of own X.509 V3 client certificate; for Server side: add id of X.509 V3 (CA-) client certificate, which is imported/assigned to this server device

The security function configuration is not included in the scope of this document. For secured communication via SSL/TLS, please refer to the chapter 3.6 of "SIMATIC S7-1500, ET 200MP, ET 200SP, ET 200AL, ET 200pro Communication Function Manual" (https://support.industry.siemens.com/cs/document/59192925/).
The information on the secured communication between MQTT broker and S7-1500 CPU via TLS can also be found in the chapter 2.3 of "MQTT Publisher for SIMATIC S7-1500" (https://support.industry.siemens.com/cs/document/109748872/)

Literature and links

- Siemens Industry Online Support: MQTT Publisher for SIMATIC CPU https://support.industry.siemens.com/cs/document/109748872/
- MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta. 29 October 2014. OASIS Standard. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html
- ChristofGroschke/MQTT-Siemens-S7-1500 https://github.com/ChristofGroschke/MQTT-Siemens-S7-1500
- Eclipse Mosquitto™ An open source MQTT broker https://mosquitto.org/
- SIMATIC S7-1500, ET 200MP, ET 200SP, ET 200AL, ET 200pro
- MQTT Client Application Example for SIMATIC S7-1500 (YU-WEN TANG)