MQTT-Protocol Test Purposes

Table of Contents

1. Introduction 6

1.1. Open Questions 6

2. The MQTT Protocol 6

2.1. Roles 6

2.1.1. Broker/Server 6

2.1.2. Publisher/Client 7

2.1.3. Subscriber/Client 7

2.2. Test configuration 7

2.2.1. Test configuration CF01 7

2.2.2. Test configuration CF02 7

2.2.3. Test configuration CF03 8

2.2.4. Test configuration CF04 8

3. Test Suite Structure 8

3.1. Broker as SUT 8

3.2. Client as SUT 9

4. Test Purposes for MQTT Server 9

4.1. Mandatory Message Data Fields 9

4.1.1. CONNECT Control Packet 9

4.1.1.1. Fixed Header 9

4.1.1.1.1. Header Flags 9

4.1.1.2. Variable Header 10

4.1.1.2.1. Protocol Name 10

4.1.1.2.2. Protocol Level 10

4.1.1.2.3. Reserved Flag 11

4.1.1.2.4. Will Flag 11

4.1.1.2.5. Will QoS Flags 13

4.1.1.2.6. Will Retain Flag 15

4.1.1.2.7. Credential Flags 16

4.1.1.2.8. Username Flag 17

4.1.1.2.9. Password Flag 18

4.1.1.3. Payload 19

4.1.1.3.1. Client Identifier 19

4.1.1.3.2. Client Identifier with zero-byte length 21

4.1.1.3.3. Client Identifier UTF-8 21

4.1.1.3.4. Will Topic UTF-8 22

4.1.1.3.5. Username UTF-8 24

4.1.2. CONNACK Control Packet 25

4.1.2.1. Fixed Header 25

4.1.2.2. Variable Header 25

4.1.2.2.1. Clean Session 25

4.1.2.2.2. Present Session 26

4.1.2.2.3. Return Code: Unacceptable Protocol Version 28

4.1.2.2.4. Return Code: Identifier rejected 28

4.1.2.2.5. Return Code: MQTT Service unavailable 29

4.1.2.2.6. Return Code: Bad Credentials 30

4.1.2.3. Payload 31

4.1.3. SUBSCRIBE Control Packet 31

4.1.3.1. Fixed Header 31

4.1.3.1.1. Header Flags 31

4.1.3.2. Variable Header 31

4.1.3.2.1. Packet Identifier 31

4.1.3.3. Payload 32

4.1.3.3.1. UTF-8 Encoding 32

4.1.3.3.2. Topic Filter 33

4.1.3.3.3. Requested QoS 34

4.1.4. SUBACK Control Packet 35

4.1.4.1. Fixed Header 35

4.1.4.1.1. Header Flags 35

4.1.4.2. Variable Header 35

4.1.4.2.1. Packet Identifier 35

4.1.4.3. Payload 36

4.1.4.3.1. Return Code: Maximum QoS 0 36

4.1.4.3.2. Return Code: Maximum QoS 1 37

4.1.4.3.3. Return Code: Maximum QoS 2 37

4.1.4.3.4. Return Code: Failure 38

4.1.5. UNSUBSCRIBE Control Packet 39

4.1.5.1. Fixed Header 39

4.1.5.1.1. Header Flags 39

4.1.5.2. Variable Header 39

4.1.5.2.1. Packet Identifier 39

4.1.5.3. Payload 40

4.1.5.3.1. UTF-8 Encoding 40

4.1.5.3.2. Topic Filter 40

4.1.6. UNSUBACK Control Packet 41

4.1.6.1. Fixed Header 41

4.1.6.2. Variable Header 41

4.1.6.3. Payload 42

4.1.7. PINGREQ Control Packet 42

4.1.7.1. Fixed Header 42

4.1.7.1.1. Header Flags 42

4.1.8. PINGRESP Control Packet 43

4.1.8.1. Fixed Header 43

4.1.9. DISCONNECT Control Packet 43

4.1.9.1. Fixed Header 43

4.1.9.1.1. Header Flags 43

4.2. Protocol Features 44

4.2.1. General 44

4.2.1.1. QoS Levels 44

4.2.1.2. Delivery Retransmission 44

4.2.1.3. Retained Messages 44

4.2.1.4. Message ordering 51

4.2.1.5. Anonymous client identifier 51

4.2.2. Session Handling 51

4.2.2.1. Credentials 51

4.2.2.2. Session Initiation 52

4.2.2.3. Session States 53

4.2.3. Subscribe 55

4.2.3.1. Subscribe existing Subscription’s Topic Filter 55

4.2.3.2. Subscribe multiple Topic Filters 57

4.2.4. Unsubscribe 58

4.2.4.1. Unsubscribe existing Subscriptions 58

4.2.4.2. Unsubscribe non existing Subscriptions 59

4.2.4.3. Unsubscribe multiple Topic Filters 59

4.2.5. Publish before CONNACK 60

4.2.6. Last Will Testament 60

4.2.7. Heartbeats 64

4.2.8. Topic Names/Filters 65

4.2.9. Error Handling 65

5. Test Purposes for MQTT Client 65

5.1. Mandatory Message Data Fields 65

5.1.1. CONNECT Control Packet 65

5.1.2. CONNACK Control Packet 66

6. Test Purposes for MQTT Client and Server 68

6.1. Mandatory Message Data Fields 68

6.1.1. PUBLISH Control Packet 68

6.1.1.1. Fixed Header 68

6.1.1.1.1. DUP Flag 68

6.1.1.1.2. QoS Level 69

6.1.1.1.3. Retain Flag 70

6.1.1.2. Variable Header 70

6.1.1.2.1. Topic Name UTF-8 Encoding 70

6.1.1.2.2. Topic Name with Wildcards 71

6.1.2. PUBACK Control Packet 72

6.1.3. PUBREC Control Packet 72

6.1.4. PUBREL Control Packet 73

6.1.5. PUBCOMP Control Packet 73

6.2. Protocol Features 73

6.2.1. QoS Level 1 73

6.2.2. QoS Level 2 74

6.2.3. Publish delivery 75

# Introduction

# The MQTT Protocol

In this document, the MQTT protocol in the version [[mqtt-v3.1.1-plus-errata01]](http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html) will be used. The terminology from the standard will be used to ensure the consistency.

## Roles

MQTT is an extremely lightweight client sever publish/subscribe messaging transport protocol with a star topology on the application layer. The transfer of an application message is always carried over the **Broker (1)** and therefore separated into two distinct steps. Each of these both steps follow strictly the client server model. The communication between an originator of a message (**Publisher (2)**) and any number of receivers (**Subscriber (3)**) is realized with the publish/subscribe model.

Client

Server

### Broker/Server

The MQTT-Broker is the central component between all involved MQTT-Clients. It is the broker’s responsibility to maintain the clients’ connections and their subscriptions.

### Publisher/Client

In context of MQTT clients which are the origin of an application message are called publisher. These clients are sending their application messages on specific topic to the broker.

### Subscriber/Client

In context of MQTT client which signal their interest on a topic with a subscription are called subscribers. Subscribers are usually the final receivers of published application messages.

## Test configuration

The test suite uses three test configurations in order to cover different test scenarios.

The following notation is used to describe the test configuration. The red box denotes the System Under Test (SUT) whereas the green box denotes the Test system.

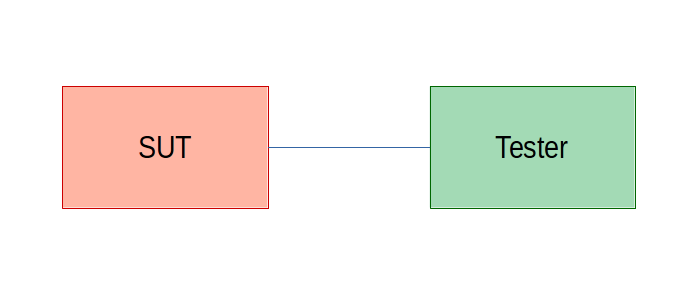


Figure : Generic test configuration

### Test configuration CF01

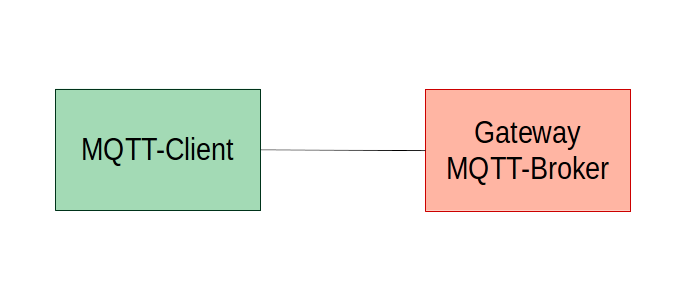


Figure : Test configuration CF01

### Test configuration CF02

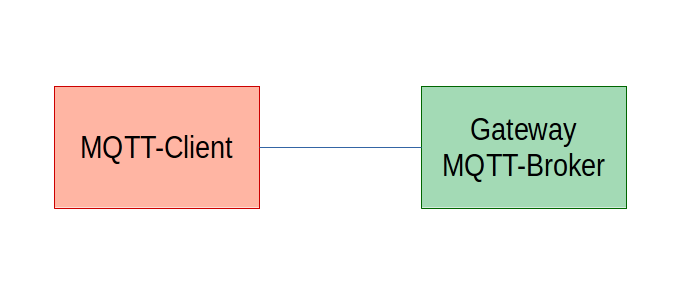


Figure : Test configuration CF02

### Test configuration CF03

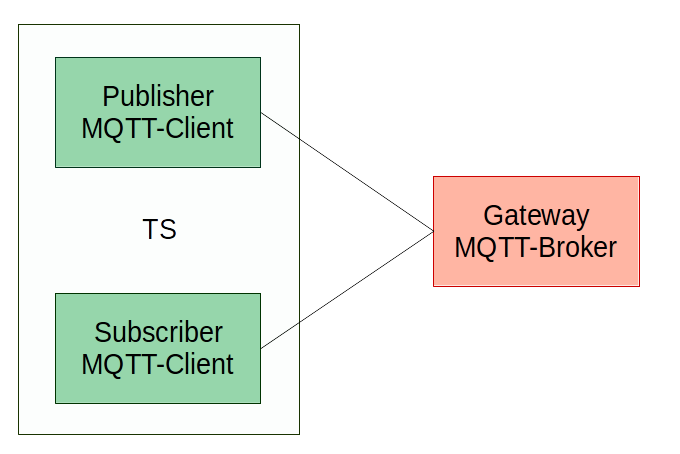


Figure : Test configuration CF03

### Test configuration CF04

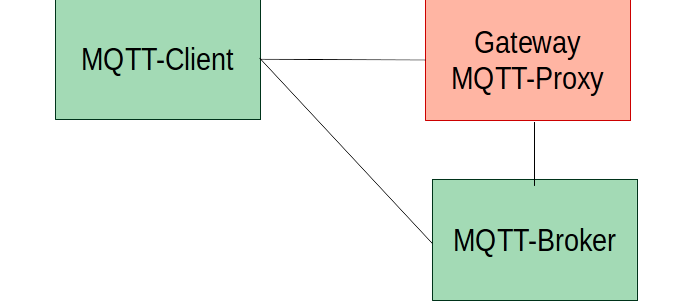


Figure : Test configuration CF04

# Test Suite Structure

## Broker as SUT

1. All mandatory message data fields
   1. Regular and illegal/corrupted data
      1. Control packets: CONNECT, CONNACK, PUBLISH, PUBACK, PUBREC, PUBREL, PUBCOMP, SUBSCRIBE, SUBACK, UNSUBSCRIBE, UNSUBACK, PINGREQ, PINGRESP, DISCONNECT
         1. Fixed Header
         2. Variable Header
         3. Payload
            1. Client identifier length restriction (up to 65535 bytes)
            2. UTF-8 encoding
2. Protocol features
   1. General
      1. QoS levels
      2. Delivery retransmission
      3. Retained messages
      4. Message ordering
      5. Anonymous client identifier
   2. Connect/disconnect (session handling)
      1. Credentials
      2. Session initiation
      3. Session states
   3. Subscribe
   4. Unsubscribe
   5. Immediate publish (w/o awaiting for CONNACK)
   6. Last Will and Testament message
   7. Heartbeats: keepAlive values
   8. Topic names/filters
   9. Error handling

## Client as SUT

**TBD**

# Test Purposes for MQTT Server

## Mandatory Message Data Fields

### CONNECT Control Packet

#### Fixed Header

##### Header Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_001 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST close the network connection if fixed header flags in CONNECT Control Packet are invalid |
| **Reference** | [MQTT-2.2.2-1], [MQTT-2.2.2-2], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **CONNECT** message  containing header\_flags := ‘1111’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

#### Variable Header

##### Protocol Name

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_002 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Protocol Name is incorrect the IUT MAY disconnect the Client or it MAY continue processing the CONNECT packet. |
| **Reference** | [MQTT-3.1.2-1] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name != ”TTQM”  and protocol\_level := 0x04  **sends** a **RESPONSE** message  TODO: response is undefined?? | |
| **Comments**  [MQTT-3.1.2-1] describes what a broker MAY do, but not what a broker MUST do | |

##### Protocol Level

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_003 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST respond to Protocol Levels which it supports (in scope: MQTT-3.1.1) with return code 0x00 |
| **Reference** | [MQTT-3.1.2-2], [MQTT-3.1.4-4] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  **sends** a **RESPONSE** message  containing CONNACK  with connect\_return\_code := 0x00 | |
| **Comments** | |

##### Reserved Flag

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_004 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST validate that the reserved flag in the CONNECT Control Packet is set to zero and disconnect the Client if it is not zero |
| **Reference** | [MQTT-3.1.2-3], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing reserved\_field := 1  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  Mosquitto does not check that flag? | |

##### Will Flag

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_005 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 1, the Will QoS and Will Retain fields in the Connect Flags will be used by the IUT, and the Will Topic and Will Message fields MUST be present in the payload. |
| **Reference** | [MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 1  and will\_qos := 0  and will\_retain := 0  and payload  containing will\_topic := omit  and will\_message := omit  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_006 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 0 the Will QoS and Will retain fields in the Connect Flags MUST be set to zero and the Will Topic and Will Message fields MUST NOT be present in the payload. |
| **Reference** | [MQTT-3.1.2-11], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 0  and will\_qos := 1  and will\_retain := 1  and payload  containing will\_topic := $WILL\_TOPIC  and will\_message := $WILL\_MESSAGE  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

##### Will QoS Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_007 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 0, then the Will QoS MUST be set to 0 |
| **Reference** | [MQTT-3.1.2-13], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 0  and will\_qos := 1  and will\_retain := 0  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_008 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 1, the value of Will QoS can be 0, 1 or 2. It MUST NOT be 3 |
| **Reference** | [MQTT-3.1.2-14], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 1  and will\_qos := 3  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_009 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 1, the value of Will QoS can be 0, 1 or 2. |
| **Reference** | [MQTT-3.1.2-14], [MQTT-3.1.4-4] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 1  and will\_qos := 0  **sends** a **RESPONSE** message  containing CONNACK  with connect\_return\_code := 0x00 | |
| **Comments** | |

##### Will Retain Flag

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_010 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 0, then the Will Retain Flag MUST be set to 0 |
| **Reference** | [MQTT-3.1.2-15], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 0  and will\_qos := 0  and will\_retain := 1  and payload  containing will\_topic := omit  and will\_message := omit  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_011 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 0, then the Will Retain Flag MUST be set to 0 |
| **Reference** | [MQTT-3.1.2-15], [MQTT-3.1.4-4] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 0  and will\_qos := 0  and will\_retain := 0  and payload  containing will\_topic := omit  and will\_message := omit  **sends** a **RESPONSE** message  containing CONNACK  with connect\_return\_code := 0x00 | |
| **Comments** | |

##### Credential Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_012 |
| **Selection** | PIC\_Broker |
| **Summary** | If the User Name Flag is set to 0, the Password Flag MUST be set to 0 |
| **Reference** | [MQTT-3.1.2-22], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 0  and password\_flag := 1  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

##### Username Flag

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_013 |
| **Selection** | PIC\_Broker |
| **Summary** | If the user name flag is set to 0, a user name MUST NOT be present in the payload |
| **Reference** | [MQTT-3.1.2-18], [MQTT-3.1.2-22], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 0  and password\_flag := 0  and payload  containing user\_name := $MQTT\_USER\_NAME  and password := omit  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_014 |
| **Selection** | PIC\_Broker |
| **Summary** | If the User Name Flag is set to 1, a user name MUST be present in the payload |
| **Reference** | [MQTT-3.1.2-19], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 1  and password\_flag := 0  and payload  containing user\_name := omit  and password := omit  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

##### Password Flag

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_015 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Password Flag is set to 0, a password MUST NOT be present in the payload |
| **Reference** | [MQTT-3.1.2-20], [MQTT-3.1.2-22], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 0  and password\_flag := 0  and payload  containing user\_name := omit  and password := $MQTT\_PASSWORD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_016 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Password Flag is set to 1, a user password MUST be present in the payload |
| **Reference** | [MQTT-3.1.2-21], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 1  and password\_flag := 1  and payload  containing user\_name := $MQTT\_USER\_NAME  and password := omit  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

#### Payload

##### Client Identifier

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_017 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST allow ClientIDs which are between 1 and 23 UTF-8 encoded bytes in length |
| **Reference** | [MQTT-3.1.3-5] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags   containing clean\_session := 1  and payload  containing client\_identifier := $24\_BYTE\_CLIENT\_ID  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  On concrete implemenations the answer is inconclusive: does the broker close because of invalid (to standard) or incorrect (to credentials) client identifier?  Servers MAY also allow more than 23 Bytes | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_018 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST allow ClientIDs that contain only alphanumeric characters [0-9a-zA-Z]. |
| **Reference** | [MQTT-3.1.3-5] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags   containing clean\_session := 1  and payload  containing client\_identifier := $INVALID\_CHAR\_CLIENT\_ID  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  On concrete implemenations the answer is inconclusive: does the broker close because of invalid (to standard) or incorrect (to credentials) client identifier?  Servers MAY also allow special characters (UTF-8)  Which special characters should be used? | |

##### Client Identifier with zero-byte length

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_019 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MAY allow a Client to supply a Client Identifier that has a length of zero bytes, however if it does so the IUT MUST treat this as a special case and assign a unique Client Identifier to that Client. |
| **Reference** | [MQTT-3.1.3-6], [MQTT-3.1.3-7], [MQTT-3.1.4-4] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags   containing clean\_session := 1  and payload  containing client\_identifier := $ZERO\_BYTE\_CLIENT\_ID  **sends** a **RESPONSE** message  containing CONNACK  with connect\_return\_code := 0x00 | |
| **Comments** | |

##### Client Identifier UTF-8

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_020 |
| **Selection** | PIC\_Broker |
| **Summary** | The ClientId MUST be a well-formed UTF-8 encoded String. |
| **Reference** | [MQTT-1.5.3-1], [MQTT-3.1.3-4], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and payload  containing client\_identifier := U+D800  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  Use @update() to inject erroneous client\_identifier. Otherwise no invalid UTF-8 Data possible | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_021 |
| **Selection** | PIC\_Broker |
| **Summary** | The ClientId MUST be a UTF-8 encoded String. |
| **Reference** | [MQTT-1.5.3-2], [MQTT-3.1.3-4], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and payload  containing client\_identifier := U+0000  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  Use @update() to inject erroneous client\_identifier | |

##### Will Topic UTF-8

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_022 |
| **Selection** | PIC\_Broker |
| **Summary** | The Will Topic MUST be a UTF-8 encoded string |
| **Reference** | [MQTT-1.5.3-1], [MQTT-3.1.3-10], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 1  and payload  containing will\_topic := U+D800  and will\_message := $WILL\_MESSAGE  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_023 |
| **Selection** | PIC\_Broker |
| **Summary** | The Will Topic MUST be a UTF-8 encoded string |
| **Reference** | [MQTT-1.5.3-2], [MQTT-3.1.3-10], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing will\_flag := 1  and payload  containing will\_topic := U+0000  and will\_message := $WILL\_MESSAGE  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

##### Username UTF-8

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_024 |
| **Selection** | PIC\_Broker |
| **Summary** | The User Name MUST be a UTF-8 encoded string. |
| **Reference** | [MQTT-1.5.3-1], [MQTT-3.1.3-11], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 1  and password\_flag := 1  and payload  containing user\_name := U+D800 U+0000 U+D8FFF  and password := $MQTT\_PASSWORD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  [MQTT-1.5.3-1] forbids UTF-8 from U+D800 to U+DFFF [MQTT-1.5.3-2] forbids UTF-8 null character U+0000 -> is one TP here enough? | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNECT\_025 |
| **Selection** | PIC\_Broker |
| **Summary** | The User Name MUST be a UTF-8 encoded string. |
| **Reference** | [MQTT-1.5.3-2], [MQTT-3.1.3-11], [MQTT-3.1.4-1], [MQTT-3.2.2-6] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 1  and password\_flag := 1  and payload  containing user\_name := U+0000  and password := $MQTT\_PASSWORD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

### CONNACK Control Packet

#### Fixed Header

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_001 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT accepts only valid fixed header flags for CONNECT Control Packet and responds with CONNACK |
| **Reference** | [MQTT-2.2.2-1], [MQTT-3.1.4-4] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **CONNECT** message  containing header\_flags := ‘0000’B  **sends** a **RESPONSE** message  containing CONNACK  with header\_flags := ‘0000’B  and connect\_return\_code := 0x00 | |
| **Comments** | |

#### Variable Header

##### Clean Session

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_002 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT accepts a connection with CleanSession set to 1, the IUT MUST set SessionPresent to 0 in the CONNACK packet in addition to setting a zero-return code. |
| **Reference** | [MQTT-3.2.2-1] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 0  and password\_flag := 0   and will\_flag := 0  and will\_qos := 0  and will\_retain := 0  and clean\_session := 1  and payload  containing client\_identifier := $CLIENT\_ID  **sends** a **RESPONSE** message  containing a CONNACK  with session\_present\_flag := 0  and connect\_return\_code := 0x00 | |
| **Comments** | |

##### Present Session

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_003 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT has stored session state, it MUST set session present flag to 1 in the CONNACK control packet. |
| **Reference** | [MQTT-3.2.2-2] |
| **Initial condition** | The IUT has a present session for $CLIENT\_ID |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing clean\_session := 0   and payload  containing client\_identifier := $CLIENT\_ID  **sends** a **RESPONSE** message  containing a CONNACK  with session\_present\_flag := 1  and connect\_return\_code := 0x00 | |
| **Comments**  Username + Password MUST be provided as otherwise the IUT could refuse the connection without even checking for sessions | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_004 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT does not have a stored session state, it MUST set session present flag to 0 in the CONNACK control packet. |
| **Reference** | [MQTT-3.2.2-3] |
| **Initial condition** | The IUT has a no present session for $CLIENT\_ID |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  and clean\_session := 0   and payload  containing client\_identifier := $CLIENT\_ID  **sends** a **RESPONSE** message  containing a CONNACK  with session\_present\_flag := 0  and connect\_return\_code := 0x00 | |
| **Comments** | |

##### Return Code: Unacceptable Protocol Version

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_005 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST respond to Protocol Levels which it does not support (in scope: MQTT-3.1.1) with return code 0x01 |
| **Reference** | [MQTT-3.1.2-2], [MQTT-3.2.2-4], [MQTT-3.2.2-5] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x05  **sends** a **RESPONSE** message  containing CONNACK  with session\_present\_flag := 0  with connect\_return\_code := 0x01  and **closes** the Network Connection | |
| **Comments** | |

##### Return Code: Identifier rejected

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_006 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Client supplies a zero-byte Client Identifier with Clean Session set to 0, the IUT MUST respond to the CONNECT Control Packet with a CONNACK return code 0x02 and then close the Network Connection. |
| **Reference** | [MQTT-3.1.3-8], [MQTT-3.1.3-9], [MQTT-3.2.2-4], [MQTT-3.2.2-5] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags   containing clean\_session := 0  and payload  containing client\_identifier := $ZERO\_BYTE\_CLIENT\_ID  **sends** a **RESPONSE** message  containing CONNACK  with session\_present\_flag := 0  and connect\_return\_code := 0x02  and **closes** the Network Connection | |
| **Comments** | |

##### Return Code: MQTT Service unavailable

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_007 |
| **Selection** | PIC\_Broker |
| **Summary** | The Network Connection has been made but the MQTT service is unavailable |
| **Reference** | [MQTT-3.2.2-4], [MQTT-3.2.2-5] |
| **Initial condition** | MQTT Service is unavailable |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 0  and password\_flag := 0   and will\_flag := 0  and will\_qos := 0  and will\_retain := 0  and clean\_session := 1  and payload  containing client\_identifier := $CLIENT\_ID  **sends** a **RESPONSE** message  containing a CONNACK  with session\_present\_flag := 0  and connect\_return\_code := 0x03  and **closes** the Network Connection | |
| **Comments**  How can that happen? No requirement found when the server sends 0x03 | |

##### Return Code: Bad Credentials

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_CONNACK\_008 |
| **Selection** | PIC\_Broker |
| **Summary** | The data in the username or password is malformed |
| **Reference** | [MQTT-3.1.2-19], [MQTT-3.1.2-21], [MQTT-3.2.2-4], [MQTT-3.2.2-5] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 1  and password\_flag := 1  and payload  containing user\_name := $MQTT\_USER\_NAME\_INVALID  and password := $MQTT\_PASSWORD\_INVALID  **sends** a **RESPONSE** message  containing CONNACK  with session\_present\_flag := 0  and connect\_return\_code := 0x04  and **closes** the Network Connection | |
| **Comments**  Negative TP with invalid credentials. | |

#### Payload

The CONNACK Control Packet has no payload.

### SUBSCRIBE Control Packet

#### Fixed Header

##### Header Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_001 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT closes the network connection if fixed header flags in SUBSCRIBE Control Packet are invalid |
| **Reference** | [MQTT-2.2.2-1], [MQTT-2.2.2-2], [MQTT-3.8.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **SUBSCRIBE** message  containing header\_flags := ‘1101’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |
|  | |

#### Variable Header

##### Packet Identifier

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_002 |
| **Selection** | PIC\_Broker |
| **Summary** | SUBSCRIBE Control Packet MUST contain a **non-zero** 16-bit Packet Identifier |
| **Reference** | [MQTT-2.3.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := 0  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 1  **sends** a **RESPONSE** message  containing a SUBACK  with packet\_identifier := 0 | |
| **Comments**  TODO: response/behavior is not described in standard Mosquitto answers regularly with SUBACK and packet\_id := 0  Positive TP is described in SUBACK -> Variable Header | |

#### Payload

##### UTF-8 Encoding

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_003 |
| **Selection** | PIC\_Broker |
| **Summary** | The Topic Filters in a Subscribe packet payload MUST be UTF-8 encoded strings. This data MUST NOT include encodings of code points between U+D800 and U+DFFF |
| **Reference** | [MQTT-1.5.3-1], [MQTT-3.8.3-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := 0x5757  and payload  containing topic\_filter := U+D800  and requested\_qos := 1  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  UTF-8 encoding | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_004 |
| **Selection** | PIC\_Broker |
| **Summary** | The Topic Filters in a Subscribe packet payload MUST be UTF-8 encoded strings. A UTF-8 encoded string MUST NOT include an encoding of the null character U+0000. |
| **Reference** | [MQTT-1.5.3-2], [MQTT-3.8.3-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := 0x5757  and payload  containing topic\_filter := U+0000  and requested\_qos := 1  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  UTF-8 encoding | |

##### Topic Filter

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_005 |
| **Selection** | PIC\_Broker |
| **Summary** | The payload of a SUBSCRIBE packet MUST contain at least one Topic Filter /QoS pair. |
| **Reference** | [MQTT-3.8.3-3], [MQTT-4.8.0-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload **with** length 0  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  Empty record MQTT\_v3\_1\_1\_SubscribePayloadList | |

##### Requested QoS

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_006 |
| **Selection** | PIC\_Broker |
| **Summary** | The upper 6 bits of the Requested QoS byte are reserved. Reserved bits for Requested QoS in the payload must be set to 0. |
| **Reference** | [MQTT-3.8.3-4] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := 0x5757  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 0  and requested\_qos\_flags := ‘111111’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBSCRIBE\_007 |
| **Selection** | PIC\_Broker |
| **Summary** | The Requested QoS in the payload MUST be 0, 1 or 2 |
| **Reference** | [MQTT-3.8.3-4] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 3  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

### SUBACK Control Packet

#### Fixed Header

##### Header Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBACK\_001 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT accepts only valid fixed header flags for SUBSCRIBE Control Packet and responds with SUBACK |
| **Reference** | [MQTT-2.2.2-1], [MQTT-3.8.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  **sends** a **RESPONSE** message  containing a SUBACK  with header\_flags := ‘0000’B | |
| **Comments** | |

#### Variable Header

##### Packet Identifier

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBACK\_002 |
| **Selection** | PIC\_Broker |
| **Summary** | SUBSCRIBE (in case where QoS > 0) Control Packet MUST contain a non-zero 16-bit Packet Identifier |
| **Reference** | [MQTT-2.3.1-1], [MQTT-2.3.1-7], [MQTT-3.8.4-1], [MQTT-3.8.4-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 1  **sends** a **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID | |
| **Comments** | |

#### Payload

##### Return Code: Maximum QoS 0

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBACK\_003 |
| **Selection** | PIC\_Broker |
| **Summary** | SUBACK return codes other than 0x00, 0x01, 0x02 and 0x80 are reserved and MUST NOT be used. |
| **Reference** | [MQTT-3.9.3-1], [MQTT-3.9.3-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 0  **sends** no **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and return\_code := 0x00 | |
| **Comments** | |

##### Return Code: Maximum QoS 1

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBACK\_004 |
| **Selection** | PIC\_Broker |
| **Summary** | SUBACK return codes other than 0x00, 0x01, 0x02 and 0x80 are reserved and MUST NOT be used. |
| **Reference** | [MQTT-3.9.3-1], [MQTT-3.9.3-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 1  **sends** no **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and return\_code := 0x01 | |
| **Comments**  Only in the case if the IUT supports QoS 1 returns 0x00 if IUT supports only QoS 0 | |

##### Return Code: Maximum QoS 2

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBACK\_005 |
| **Selection** | PIC\_Broker |
| **Summary** | SUBACK return codes other than 0x00, 0x01, 0x02 and 0x80 are reserved and MUST NOT be used. |
| **Reference** | [MQTT-3.9.3-1], [MQTT-3.9.3-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 2  **sends** no **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and return\_code := 0x02 | |
| **Comments**  Only in the case if the IUT supports QoS 2 returns 0x00/0x01 if IUT supports only QoS 0/1 | |

##### Return Code: Failure

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_SUBACK\_006 |
| **Selection** | PIC\_Broker |
| **Summary** | SUBACK return codes other than 0x00, 0x01, 0x02 and 0x80 are reserved and MUST NOT be used. |
| **Reference** | [MQTT-3.9.3-1], [MQTT-3.9.3-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_\*\*  and requested\_qos := 2  **sends** no **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and return\_code := 0x80 | |
| **Comments**  TODO: When does this happen? -> subscription denied for $TOPIC\_\*\* ??  <https://github.com/erlio/vernemq/issues/72> https://groups.google.com/forum/#!topic/mqtt/ppdmL4-v2Yk | |

### UNSUBSCRIBE Control Packet

#### Fixed Header

##### Header Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_047 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT closes the network connection if fixed header flags in UNSUBSCRIBE Control Packet are invalid |
| **Reference** | [MQTT-2.2.2-1], [MQTT-2.2.2-2], [MQTT-3.10.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **UNSUBSCRIBE** message  containing header\_flags := ‘1101’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |
|  | |

#### Variable Header

##### Packet Identifier

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_048 |
| **Selection** | PIC\_Broker |
| **Summary** | UNSUBSCRIBE Control Packet MUST contain a **non-zero** 16-bit Packet Identifier |
| **Reference** | [MQTT-2.3.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := 0  and payload  containing topic\_filter := $TOPIC\_FILTER  **sends** a **RESPONSE** message  containing a UNSUBACK  with packet\_identifier := 0 | |
| **Comments**  TODO: response/behavior is not described in standard!! | |

#### Payload

##### UTF-8 Encoding

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_049 |
| **Selection** | PIC\_Broker |
| **Summary** | The Topic Filters in an Unsubscribe packet payload MUST be UTF-8 encoded strings. This data MUST NOT include encodings of code points between U+D800 and U+DFFF |
| **Reference** | [MQTT-1.5.3-1], [MQTT-3.10.3-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := U+D800  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  UTF-8 encoding | |

##### Topic Filter

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_050 |
| **Selection** | PIC\_Broker |
| **Summary** | The payload of a UNSUBSCRIBE packet MUST contain at least one Topic Filter. |
| **Reference** | [MQTT-3.10.3-2], [MQTT-4.8.0-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload **with** length 0  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  Empty record MQTT\_v3\_1\_1\_UnsubscribePayload | |

### UNSUBACK Control Packet

#### Fixed Header

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_051 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT accepts only valid fixed header flags for UNSUBSCRIBE Control Packet and responds with SUBACK |
| **Reference** | [MQTT-2.2.2-1], [MQTT-3.10.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  **sends** a **RESPONSE** message  containing a UNSUBACK  with header\_flags := ‘0000’B | |
| **Comments** | |

#### Variable Header

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_052 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST respond to an UNSUBSCRIBE request by sending an UNSUBACK control packet. The UNSUBACK control packet MUST have the same Packet Identifier as the UNSUBSCRIBE Control Packet. |
| **Reference** | [MQTT-3.10.4-4], [MQTT-3.10.4-5] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  **sends** a **RESPONSE** message  containing a UNSUBACK  with packet\_identifier := $PACKET\_ID | |
| **Comments**  NOTE: Do [MQTT-3.10.4-4] & [MQTT-3.10.4-5] need 2 separate TPs? | |

#### Payload

The UNSUBACK Control Packet has no payload

### PINGREQ Control Packet

#### Fixed Header

##### Header Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_053 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT closes the network connection if fixed header flags in PINGREQ Control Packet are invalid |
| **Reference** | [MQTT-2.2.2-1], [MQTT-2.2.2-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PINGREQ** message  containing header\_flags := ‘1111’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |
|  | |

### PINGRESP Control Packet

#### Fixed Header

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_054 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT accepts only valid fixed header flags for PINGREQ Control Packet and responds with a PINGRESP |
| **Reference** | [MQTT-2.2.2-1], [MQTT-3.12.4-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **PINGREQ** message  containing header\_flags := ‘0000’B  **sends** a **RESPONSE** message  containing a PINGRESP  with header\_flags := ‘0000’B | |
| **Comments** | |

### DISCONNECT Control Packet

#### Fixed Header

##### Header Flags

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_055 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST validate that reserved bits are set to zero and disconnect the Client if they are not zero. |
| **Reference** | [MQTT-2.2.2-1], ~~[MQTT-3.14.1-1]~~ |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **DISCONNECT** message  containing header\_flags := ‘0000’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_056 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST validate that reserved bits are set to zero and disconnect the Client if they are not zero. |
| **Reference** | [MQTT-2.2.2-2], [MQTT-3.14.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **DISCONNECT** message  containing header\_flags := ‘1111’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

## Protocol Features

### General

#### QoS Levels

#### Delivery Retransmission

#### Retained Messages

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_057 |
| **Selection** | PIC\_Broker |
| **Summary** | Retained messages do not form part of the Session state in the IUT, they MUST NOT be deleted when the Session ends |
| **Reference** | [MQTT-3.1.2-7] |
| **Initial condition** | * A retained message is present in the topic $TOPIC * Client connected to the Broker with CleanSession set to 1 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC  and requested\_qos := 0  **sends** a **RESPONSE** message  containing SUBACK  with return\_code = 0x00  **sends** a **RESPONSE** message  containing PUBLISH  with topic\_name $TOPIC  and the retained message as payload | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_058 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Retain Flag is set to 1, in a PUBLISH Control Packet sent by a Client to the IUT, the IUT MUST store the Application Message and its QoS, so that It can be delivered to future subscribers. |
| **Reference** | [MQTT-3.3.1-5], [MQTT-3.3.1-6], [MQTT-3.3.1-8] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 1  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $PAYLOAD  **sends** a **RESPONSE** message to Client 1  containing PUBACK  containing packet\_identifier := $PACKET\_ID\_1  **on receipt** of an **SUBSCRIBE** message from Client 2  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID\_2  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 0  **sends** a **RESPONSE** message to Client 2  containing a SUBACK  with packet\_identifier := $PACKET\_ID\_2  and payload  containing return\_code := 0x00  **sends** a **PUBLISH** message to Client 2  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 1  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_3  and payload := $PAYLOAD | |
| **Comments**  Derived from [MQTT-3.3.1-8] | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_059 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST set the Retain Flag to 0 when a PUBLISH Control Packet is sent to a Client because it matches an established subscription regardless of how the flag was set in the message it received. |
| **Reference** | [MQTT-3.3.1-9] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 connected to the Broker   + Client 2 subscribed on topic $TOPIC with QoS 0 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 1  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $PAYLOAD  **sends** a **RESPONSE** message to Client 1  containing PUBACK  containing packet\_identifier := $PACKET\_ID\_1  **sends** a **PUBLISH** message to Client 2  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_3  and payload := $PAYLOAD | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_060 |
| **Selection** | PIC\_Broker |
| **Summary** | A PUBLISH Control Packet with a Retain Flag set to 1 and a payload containing zero bytes will be processed as normal by the IUT and sent to Clients with a subscription matching the topic name. Additionally, any existing retained message with the same topic name MUST be removed and any future subscribers for the topic will not receive a retained message. |
| **Reference** | [MQTT-3.3.1-10], [MQTT-3.3.1-11] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 connected to the Broker   + Client 2 subscribed on topic $TOPIC with QoS 0 * Client 3 connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 1  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $ZERO\_BYTE\_PAYLOAD  **sends** a **RESPONSE** message to Client 1  containing PUBACK  containing packet\_identifier := $PACKET\_ID\_1  **sends** a **PUBLISH** message to Client 2  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_2  and payload := $ZERO\_BYTE\_PAYLOAD  **on receipt** of an **SUBSCRIBE** message from Client 3  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID\_3  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 0  **sends** a **RESPONSE** message to Client 2  containing a SUBACK  with packet\_identifier := $PACKET\_ID\_3  and payload  containing return\_code := 0x00  **sends** no **PUBLISH** message to Client 3 | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_061 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Retain Flag is 0, in a PUBLISH Control Packet sent by a Client to the IUT, the IUT MUST NOT store the message and MUST NOT remove or replace any existing retained message. |
| **Reference** | [MQTT-3.3.1-12] |
| **Initial condition** | * Client 1 connected to the Broker   + Client 1 published retained message to topic $TOPIC with $RETAIN\_PAYLOAD * Client 2 connected to the Broker   + Client 2 subscribed on topic $TOPIC with QoS 0 * Client 3 connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $PAYLOAD  **sends** a **RESPONSE** message to Client 1  containing PUBACK  containing packet\_identifier := $PACKET\_ID\_1  **sends** a **PUBLISH** message to Client 2  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_2  and payload := $PAYLOAD  **on receipt** of an **SUBSCRIBE** message from Client 3  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID\_2  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 0  **sends** a **RESPONSE** message to Client 3  containing a SUBACK  with packet\_identifier := $PACKET\_ID\_2  and payload  containing return\_code := 0x00  **sends** a **PUBLISH** message to Client 3  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 1  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_3  and payload := $RETAIN\_PAYLOAD | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_062 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT receives a QoS 0 message with a Retain Flag set to 1 it MUST discard any message previously retained for that topic. |
| **Reference** | [MQTT-3.3.1-7] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 1  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $PAYLOAD  **sends** no **RESPONSE** message to Client 1  **on receipt** of an **SUBSCRIBE** message from Client 2  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID\_2  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 0  **sends** a **RESPONSE** message to Client 2  containing a SUBACK  with packet\_identifier := $PACKET\_ID\_2  and payload  containing return\_code := 0x00  **sends** no **PUBLISH** message to Client 2 | |
| **Comments** | |

#### Message ordering

#### Anonymous client identifier

What is meant by “Anonymous client identifier”? Section 4.1.1.3.2 describes a CONNECT without clientId

### Session Handling

#### Credentials

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_063 |
| **Selection** | PIC\_Broker |
| **Summary** | If the User Name and Password Flags are set to 1, then User Name and Password MUST be present in the payload. |
| **Reference** | [MQTT-3.1.2-19], [MQTT-3.1.2-21], [MQTT-3.1.4-4] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 1  and password\_flag := 1  and payload  containing user\_name := $MQTT\_USER\_NAME  and password := $MQTT\_PASSWORD  **sends** a **RESPONSE** message  containing CONNACK  with connect\_return\_code := 0x00 | |
| **Comments**  Positive TP with valid credentials. Invalid counterpart see CONNACK Return Codes -> Bad Credentials | |

#### Session Initiation

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_064 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT expects only a CONNECT packet as the first packet for each new connection |
| **Reference** | [MQTT-3.1.0-1] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := 0x00   and protocol\_name := “MQTT”  and protocol\_level := 0x04  **sends** a **RESPONSE** message  containing CONNACK  with connect\_return\_code := 0 | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_065 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT expects only a CONNECT packet as the first packet for each new connection. Interaction with the IUT before a CONNECT is not allowed. |
| **Reference** | [MQTT-3.1.0-1] |
| **Initial condition** | Client not connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_066 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT can only receive exactly one CONNECT packet over a network connection. |
| **Reference** | [MQTT-3.1.0-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message  containing header\_flags := 0  and protocol\_name := “MQTT”  and protocol\_level := 4  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

#### Session States

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_067 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT must be able to restore a session for a client |
| **Reference** | [MQTT-3.1.2-4], [MQTT-3.1.2-5], [MQTT-3.1.4-3] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 not connected to the Broker   + Client 2 has a Session with the Broker   + Client 2 subscribed for Topic $SESSION\_TOPIC with QoS 0 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **PUBLISH** message from Client 1  containing header\_flags   containing dup\_flag := 0  and qos\_level := 1   and retain\_flag := 0  and topic\_name := $SESSION\_TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $PAYLOAD  **sends** a **RESPONSE** message to Client 1  containing a PUBACK  with packet\_identifier := $PACKET\_ID\_1  **on receipt** of a **CONNECT** message from Client 2  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing clean\_session := 0  **sends** a **RESPONSE** message to Client 2  containing a CONACK  with connect\_return\_code = 0x00  **sends** a **RESPONSE** message to Cient 2  containing a PUBLISH  with header\_flags   containing dup\_flag := 0  and qos\_level := 0  and retain\_flag := 0  and topic\_name := $SESSION\_TOPIC  and payload := $PAYLOAD | |
| **Comments**  Needs to be checked!! | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_068 |
| **Selection** | PIC\_Broker |
| **Summary** | If CleanSession is set to 1 the IUT MUST discard any previous sessions for this client and start a new one. |
| **Reference** | [MQTT-3.1.2-6], [MQTT-3.1.4-3] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 not connected to the Broker   + Client 2 has a Session with the Broker   + Client 2 subscribed for Topic $SESSION\_TOPIC with QoS 0 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message from Client 2  containing header\_flags := 0  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing clean\_session := 1  **sends** a **RESPONSE** message  containing CONACK  with connect\_return\_code = 0x00  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $SESSION\_TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message to Client 2  containing PUBLISH  with topic\_name of subscription from previous session  and payload | |
| **Comments**  If receive Publish => fail because the standard says explicitly that no PUBLISH to client 2 is expected | |

### Subscribe

#### Subscribe existing Subscription’s Topic Filter

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_069 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT receives a SUBSCRIBE packet containing a Topic Filter that is identical to an existing Subscription’s Topic Filter then it MUST completely replace that existing Subscription. |
| **Reference** | [MQTT-3.8.4-3] |
| **Initial condition** | * Client 1 connected to the IUT * Client 1 subscribed to the topic $TOPIC with QoS 0 * Client 2 connected to the IUT |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message from Client 1  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  and requested\_qos := 1  **sends** a **RESPONSE** message to Client 1  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and payload  containing return\_code := 0x01  **on receipt** of a **PUBLISH** message from Client 2  containing header  with dup\_flag := 0  and qos\_level := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and payload := $PAYLOAD   **sends** a **PUBLISH** message to Client 1  containing header  with dup\_flag := 0  and qos\_level := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and payload := $PAYLOAD | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_070 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT receives a SUBSCRIBE packet containing a Topic Filter that is identical to an existing Subscription’s Topic Filter then it MUST completely replace that existing Subscription. Any existing retained messages matching the Topic Filter MUST be re-sent. |
| **Reference** | [MQTT-3.8.4-3] |
| **Initial condition** | * Client connected to the IUT * Client subscribed to the topic $TOPIC with QoS 0 * Retained Message is published to topic $TOPIC with QoS 0 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC \_FILTER  and requested\_qos := 1  **sends** a **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and payload  containing return\_code := 0x01  **sends** a **PUBLISH** message  containing header  with dup\_flag := 0  and qos\_level := 1  and retain\_flag := 1  and topic\_name := $TOPIC   and payload := $PAYLOAD | |
| **Comments** | |

#### Subscribe multiple Topic Filters

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_071 |
| **Selection** | PIC\_Broker |
| **Summary** | If the IUT receives a SUBSCRIBE packet containing multiple Topic Filters it MUST handle that packet as if it had received a sequence of multiple SUBSCRIBE packets, except that it combines their responses into a single SUBACK response. |
| **Reference** | [MQTT-3.8.4-4], [MQTT-3.8.4-5], [MQTT-3.9.3-1] |
| **Initial condition** | Client connected to the IUT |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **SUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC \_0  and requested\_qos := 0  and topic\_filter := $TOPIC \_1  and requested\_qos := 1  and topic\_filter := $TOPIC \_2  and requested\_qos := 2  **sends** a **RESPONSE** message  containing a SUBACK  with packet\_identifier := $PACKET\_ID  and payload  containing return\_code := 0x00  and return\_code := 0x01  and return\_code := 0x02 | |
| **Comments**  [MQTT-3.8.4-4]: multiple topic filter descriptions  [MQTT-3.9.3-1]: return codes match the order of the Topic Filters What happens, when the IUT does not support QoS > 0? | |

### Unsubscribe

#### Unsubscribe existing Subscriptions

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_072 |
| **Selection** | PIC\_Broker |
| **Summary** | UNSUBSCRIBE (in case where QoS > 0) Control Packet MUST contain a non-zero 16-bit Packet Identifier |
| **Reference** | [MQTT-2.3.1-1], [MQTT-2.3.1-7], [MQTT-3.10.4-4] |
| **Initial condition** | * Client connected to the Broker * Client subscribed on topic $TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  **sends** a **RESPONSE** message  containing a UNSUBACK  with packet\_identifier := $PACKET\_ID | |
| **Comments** | |

#### Unsubscribe non existing Subscriptions

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_073 |
| **Selection** | PIC\_Broker |
| **Summary** | UNSUBSCRIBE (in case where QoS > 0) Control Packet MUST contain a non-zero 16-bit Packet Identifier |
| **Reference** | [MQTT-2.3.1-1], [MQTT-2.3.1-7], [MQTT-3.10.4-5] |
| **Initial condition** | * Client connected to the Broker * Client is not subscribed on topic $TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC\_FILTER  **sends** a **RESPONSE** message  containing a UNSUBACK  with packet\_identifier := $PACKET\_ID | |
| **Comments** | |

#### Unsubscribe multiple Topic Filters

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_074 |
| **Selection** | PIC\_Broker |
| **Summary** | If an IUT receives an UNSUBSCRIBE packet that contains multiple Topic Filters it MUST handle that packet as if it had received a sequence of multiple UNSUBSCRIBE packets, except that it sends just one UNSUBACK response. |
| **Reference** | [MQTT-3.10.4-6] |
| **Initial condition** | Client connected to the IUT |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **UNSUBSCRIBE** message  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $TOPIC \_1  and topic\_filter := $TOPIC \_2  **sends** a **RESPONSE** message  containing a UNSUBACK  with packet\_identifier := $PACKET\_ID | |
| **Comments** | |

### Publish before CONNACK

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_075 |
| **Selection** | PIC\_Broker |
| **Summary** | Clients can send further Control Packets immediately after sending a CONNECT Control Packet. If the IUT rejects the CONNECT, it MUST NOT process any data sent by the Client after the CONNECT Control Packet |
| **Reference** | [MQTT-3.1.4-5] |
| **Initial condition** | * Client 1 connected to the Broker * Client 2 not connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **CONNECT** message from Client 2  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04 | |
| **Comments**  **TODO: how to describe this TP??** | |

### Last Will Testament

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_076 |
| **Selection** | PIC\_Broker |
| **Summary** | A Will Message MUST be stored on the IUT and associated with the Network Connection. The Will Message MUST be published when the network connection is closed before a DISCONNECT Packet. |
| **Reference** | [MQTT-3.1.2-8] |
| **Initial condition** | * Client 1 connected to the Broker with LWT   + Will\_flag := 1   + Will\_topic := $WILL\_TOPIC   + Will\_message := $WILL\_MESSAGE * Client 2 connected to the Broker and subscribed on $WILL\_TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on** **close** of the Network Connection from/to Client 1  **sends** no **RESPONSE** message to Client 1  and **closes** the Network Connection  **sends** a **RESPONSE** message to Client 2  containing PUBLISH  with topic\_name := $WILL\_TOPIC  and payload := $WILL\_MESSAGE | |
| **Comments**  -> Network Connection is closed here without DISCONNECT Control Packet | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_077 |
| **Selection** | PIC\_Broker |
| **Summary** | When the IUT receives a DISCONNECT Control Packet it MUST discard any Will Message associated with the current connection without publishing it. |
| **Reference** | [MQTT-3.1.2-10], [MQTT-3.14.4-3] |
| **Initial condition** | * Client 1 connected to the Broker with LWT   + Will\_flag := 1   + Will\_topic := $WILL\_TOPIC   + Will\_message := $WILL\_MESSAGE * Client 2 connected to the Broker and subscribed on $WILL\_TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **DISCONNECT** message from Client 1  containing header\_flags := ‘0000’B  **sends** no **RESPONSE** message to Client 1  and **closes** the Network Connection  **sends** no **PUBLISH** message to Client 2 | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_078 |
| **Selection** | PIC\_Broker |
| **Summary** | The Will Message MUST be removed from the stored Session state in the IUT once it has been published |
| **Reference** | [MQTT-3.1.2-10], [MQTT-3.1.2-16] |
| **Initial condition** | * Client 1 connected to the Broker with LWT   + Will\_flag := 1   + Will\_retain := 0   + Will\_topic := $WILL\_TOPIC   + Will\_message := $WILL\_MESSAGE * Client 2 connected to the Broker and subscribed on $WILL\_TOPIC * Client 3 connected to the Broker and not subscribed on $WILL\_TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on close** of Network Connection from/to Client 1  **sends** a **RESPONSE** message to Client 2  containing PUBLISH  with topic\_name := $WILL\_TOPIC  and payload := $WILL\_MESSAGE  **sends** no **PUBLISH** message to Client 3  **on receipt** of a **SUBSCRIBE** message from Client 3  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $WILL\_TOPIC  **sends** a **RESPONSE** message to Client 3  containing SUBACK  **sends** no **PUBLISH** message to Client 3 | |
| **Comments**  Not mentioned in [MQTT-3.1.2-10] -> [MQTT-3.1.2-16]  Publish really required here?  If nobody subscribed to /client1/will\_topic until Client1 „dies“ no publish will be triggered. Nevertheless. the Will\_Message will be removed because of the will\_retain flag | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_079 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set to 1 and Will Retain is set to 1, the IUT MUST publish the Will Message as a retained message. |
| **Reference** | [MQTT-3.1.2-17] |
| **Initial condition** | * Client 1 connected to the Broker with LWT   + Will\_flag := 1   + Will\_retain := 1   + Will\_topic := $WILL\_TOPIC   + Will\_message := $WILL\_MESSAGE * Client 2 connected to the Broker and subscribed on $WILL\_TOPIC * Client 3 connected to the Broker and not subscribed on $WILL\_TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on close** of Network Connection from/to Client 1  **sends** a **RESPONSE** message to Client 2  containing PUBLISH  with topic\_name := $WILL\_TOPIC  and payload := $WILL\_MESSAGE  **sends** no **PUBLISH** message to Client 3  **on receipt** of a **SUBSCRIBE** message from Client 3  containing header\_flags := ‘0010’B  and packet\_identifier := $PACKET\_ID  and payload  containing topic\_filter := $WILL\_TOPIC  **sends** a **RESPONSE** message to Client 3  containing SUBACK  **sends** a **RESPONSE** message to Client 3  containing PUBLISH  with topic\_name := $WILL\_TOPIC  and payload := $WILL\_MESSAGE | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_080 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Will Flag is set 0, a Will Message MUST NOT be published when this Network Connection ends. |
| **Reference** | [MQTT-3.1.2-12] |
| **Initial condition** | * Client 1 connected to the Broker with LWT   + Will\_flag := 0   + Will\_retain := 0   + Will\_topic := omit   + Will\_message := omit * Client 2 connected to the Broker and subscribed on $WILL\_TOPIC |
| **Test purpose** | |
| Ensure that the IUT  **on close** of Network Connection from/to Client 1  **sends** no **PUBLISH** message to Client 2 | |
| **Comments**  If the Will\_flag is set to 0, then Will\_Message and Will\_Topic MUST NOT be set. How should Will\_Message be sent? [MQTT-3.1.2-12] is implicitly covered by [MQTT-3.1.2-11]? | |

### Heartbeats

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_081 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Keep Alive value is non-zero and the IUT does not receive a Control Packet from the Client within $T\_KA \* 1.5 it MUST disconnect the Network Connection to the Client as if the network had failed. |
| **Reference** | [MQTT-3.1.2-24] |
| **Initial condition** | * Client connected to the Broker * Keep Alive Value $T\_KA provided by the Client |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of no message from Client within $T\_KA \* 1.5  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_082 |
| **Selection** | PIC\_Broker |
| **Summary** | If the Keep Alive value is non-zero and the IUT does not receive a Control Packet from the Client within $T\_KA \* 1.5 it MUST disconnect the Network Connection to the Client as if the network had failed. |
| **Reference** | [MQTT-3.1.2-24] |
| **Initial condition** | * Client connected to the Broker * Keep Alive Value $T\_KA provided by the Client |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **PINGREQ** message  **sends** a **PINGRESP** message | |
| **Comments** | |

### Topic Names/Filters

### Error Handling

# Test Purposes for MQTT Client

## Mandatory Message Data Fields

### CONNECT Control Packet

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_083 |
| **Selection** | PIC\_Client |
| **Summary** | The first packet sent from the Server to the IUT MUST be a CONNACK packet. |
| **Reference** | [MQTT-3.2.0-1] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that   **when** the IUT is requested to connect  **then** the IUT sends a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  and connect\_flags  containing user\_name\_flag := 0  and password\_flag := 0  and will\_flag := 0  and will\_qos := 0  and will\_retain := 0  and clean\_session := 1  and payload  containing client\_identifier := ?   and will\_topic := omit  and will\_message := omit  and user\_name := omit  and password := omit | |
| **Comments**  UpperTester required to trigger Connect Request!  Different options for Connect Request required | |

### CONNACK Control Packet

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_084 |
| **Selection** | PIC\_Client |
| **Summary** | The first packet sent from the Server to the IUT MUST be a CONNACK packet. |
| **Reference** | [MQTT-3.2.0-1] |
| **Initial condition** |  |
| **Test purpose** | |
| Ensure that the IUT  **on send** of a **CONNECT** message  containing header\_flags := ‘0000’B  and protocol\_name := “MQTT”  and protocol\_level := 0x04  **receives** a **RESPONSE** message  containing a CONNACK  with connect\_return\_code := 0x00 | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_085 |
| **Selection** | PIC\_Client |
| **Summary** | The first packet sent from the Server to the IUT MUST be a CONNACK packet. |
| **Reference** | [MQTT-3.2.0-1] |
| **Initial condition** | The IUT sent a CONNECT Control Packet to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **CONNACK** message  containing header\_flags := ‘0000’B  and connect\_return\_code := 0x00  TODO | |
| **Comments** | |

# Test Purposes for MQTT Client and Server

## Mandatory Message Data Fields

### PUBLISH Control Packet

#### Fixed Header

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_086 |
| **Selection** | PIC\_Broker, PIC\_Client, QoS\_0 |
| **Summary** | The DUP flag MUST be set to 0 for all QoS 0 messages |
| **Reference** | [MQTT-2.2.2-1], [MQTT-3.3.1-2], [MQTT-4.3.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message | |
| **Comments**  TODO: is second client for receive required here? | |

##### DUP Flag

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_087 |
| **Selection** | PIC\_Broker, PIC\_Client, QoS\_0 |
| **Summary** | The DUP flag MUST be set to 0 for all QoS 0 messages |
| **Reference** | [MQTT-3.3.1-2], [MQTT-4.3.1-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 1  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_088 |
| **Selection** | PIC\_Broker, PIC\_Client, QoS\_1 |
| **Summary** | The DUP flag MUST be set to 1 when a re-delivery of a PUBLISH Control Packet is attempted. |
| **Reference** | [MQTT-3.3.1-1] |
| **Initial condition** | * Client connected to the Broker * Assume the first PUBLISH message is lost on the Network |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 1  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** a **RESPONSE** message  containing a PUBACK  with header\_flags := ‘0000’B | |
| **Comments**  Assumption makes no difference for the TP | |

##### QoS Level

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_089 |
| **Selection** | PIC\_Broker, PIC\_Client |
| **Summary** | A PUBLISH Control Packet MUST NOT have QoS Level set to 3. |
| **Reference** | [MQTT-2.2.2-2], [MQTT-3.3.1-4] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 3 (RESERVED)  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  Mosquitto does not close the Network Connection nor forwards the Message | |

##### Retain Flag

The interpretation and behavior of the retain flag differs depending on the role of the IUT. TODO -> Retain Flag TPs in Chapter 4 + 5

#### Variable Header

##### Topic Name UTF-8 Encoding

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_090 |
| **Selection** | PIC\_Broker, PIC\_Client |
| **Summary** | The Topic Name MUST be present as the first field in the PUBLISH Control Packet variable header. It MUST be a UTF-8 encoded string. |
| **Reference** | [MQTT-1.5.3-1], [MQTT-3.3.2-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := U+D800  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  UTF-8 encoding | |

##### Topic Name with Wildcards

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_091 |
| **Selection** | PIC\_Broker, PIC\_Client |
| **Summary** | The Topic Name in the PUBLISH Control Packet MUST NOT contain wildcard characters. |
| **Reference** | [MQTT-3.3.2-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $WILDCARD\_TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message  and **closes** the Network Connection | |
| **Comments**  The standard says, „MUST NOT contain wildcard characters” but it does not say what will happen if it contains wildcard characters | |

### PUBACK Control Packet

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_092 |
| **Selection** | PIC\_Broker, PIC\_Client, PIC\_QoS1 |
| **Summary** | If the IUT receives a PUBLISH Control Packet with QoS 1, it MUST respond with a PUBACK Control Packet. |
| **Reference** | [MQTT-3.3.4-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** a **RESPONSE** message  containing PUBACK  with header\_flags := ’0000’B   and packet\_identifier := $PACKET\_ID | |
| **Comments** | |

### PUBREC Control Packet

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_093 |
| **Selection** | PIC\_Broker, PIC\_Client, PIC\_QoS2 |
| **Summary** | If the IUT receives a PUBLISH Control Packet with QoS 2, it MUST respond with a PUBREC Control Packet. |
| **Reference** | [MQTT-3.3.4-1] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 2  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** a **RESPONSE** message  containing PUBREC  with header\_flags := ’0000’B   and packet\_identifier := $PACKET\_ID | |
| **Comments** | |

### PUBREL Control Packet

### PUBCOMP Control Packet

## Protocol Features

### QoS Level 1

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_094 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST respond with PUBACK Control Packet containing the Packet Identifier from the incoming PUBLISH Control Packet, having accepted ownership of the Application Message. After the IUT has sent a PUBACK it MUST treat any incoming PUBLISH Control Packets that contain the same Packet Identifier as being a new publication, irrespective of the setting of its DUP flag. |
| **Reference** | [MQTT-4.3.2-2] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  with dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** a **RESPONSE** message  containing PUBACK  with packet\_identifier := $PACKET\_ID  **on receipt** of an **PUBLISH** message  containing publish\_header  with dup\_flag := 1  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** a **RESPONSE** message  containing PUBACK  with packet\_identifier := $PACKET\_ID | |
| **Comments**  Is the Conformance Statement checked by that TP? | |

### QoS Level 2

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_095 |
| **Selection** | PIC\_Broker |
| **Summary** | The IUT MUST assign an unused Packet Identifier when it has a new Application Message to publish and send a PUBLISH Control Packet with QoS=2 and DUP=0… |
| **Reference** | [MQTT-4.3.3-1] |
| **Initial condition** | * Client connected to the Broker * Client subscribed to Topic $TOPIC with QoS 2 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of a **PUBLISH** message  containing publish\_header  with dup\_flag := 0  and qos\_flag := 2  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_1  and payload := $PAYLOAD  **sends** a **PUBREC** message  containing packet\_identifier := $PACKET\_ID\_1  **on receipt** of a **PUBREL** message  containing packet\_identifier := $PACKET\_ID\_1  **sends** a **PUBCOMP** message  containing packet\_identifier := $PACKET\_ID\_1  **sends** a **PUBLISH** message  containing publish\_header  with dup\_flag := 0  and qos\_flag := 2  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID\_2  and payload := $PAYLOAD  **on receipt** of a **PUBREC** message  containing packet\_identifier := $PACKET\_ID\_2  **sends** a **PUBREL** message  containing packet\_identifier := $PACKET\_ID\_2  **on receipt** of a **PUBCOMP** message  containing packet\_identifier := $PACKET\_ID\_2  **sends** no **RESPONSE** message | |
| **Comments** | |

### Publish delivery

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_096 |
| **Selection** | PIC\_Broker, PIC\_Client |
| **Summary** | The value of the DUP flag from an incoming PUBLISH Control Packet is not propagated when the PUBLISH Control Packet is sent to subscribers by the IUT. The DUP flag in the outgoing PUBLISH Control Packet is set independently to the incoming PUBLISH Control Packet, its value MUST be determined solely by whether the outgoing PUBLISH Control Packet is a retransmission. |
| **Reference** | [MQTT-3.3.1-3] |
| **Initial condition** | * Client 1 connected to the Broker   + Assume the first PUBLISH message is lost on the Network * Client 2 connected to the Broker   + Client 2 subscribed for topic $TOPIC with QoS 1 |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message from Client 1  containing publish\_header  with dup\_flag := 1  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** a **RESPONSE** message to Client 1  containing a PUBACK  with header\_flags := ‘0000’B  **send** a **PUBLISH** message to Client 2containing publish\_header  with dup\_flag := 0  and qos\_flag := 1  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD | |
| **Comments**  Assumption makes no difference for the TP | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_097 |
| **Selection** | PIC\_Broker |
| **Summary** |  |
| **Reference** | [MQTT-3.3.1-6] |
| **Initial condition** | Client connected to the Broker |
| **Test purpose** | |
| Ensure that the IUT  **on receipt** of an **PUBLISH** message  containing publish\_header  containing dup\_flag := 0  and qos\_flag := 0  and retain\_flag := 0  and topic\_name := $TOPIC  and packet\_identifier := $PACKET\_ID  and payload := $PAYLOAD  **sends** no **RESPONSE** message | |
| **Comments** | |

|  |  |
| --- | --- |
| **TP-ID** | TP\_MQTT\_Broker\_098 |
| **Selection** | PIC\_Client |
| **Summary** | It is the responsibility of the Client to ensure that the interval between Control Packets being sent does not exceed the Keep Alive value. In the absence of sending any other Control Packets, the Client MUST send a PINGREQ Packet |
| **Reference** | [MQTT-3.1.2-23] |
| **Initial condition** | * Client connected to the Broker * Keep Alive Value $T\_KA provided by the Client |
| **Test purpose** | |
| Ensure that the IUT  **on timeout** of **T\_KA**  sends PINGREQ Packet | |
| **Comments** | |