ETSI TS 103 597-1 V1.1.2 (2021-01)



Methods for Testing and Specification (MTS); Test Specification for MQTT; Part 1: Conformance Tests

Reference DTS/MTS-TST9 Keywords conformance, TSS&TP

ETSI

650 Route des Lucioles F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C Association à but non lucratif enregistrée à la Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from: http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status.

Information on the current status of this and other ETSI documents is available at https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services: https://portal.etsi.org/People/CommitteeSupportStaff.aspx

Copyright Notification

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2021. All rights reserved.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

oneM2M[™] logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners.

GSM® and the GSM logo are trademarks registered and owned by the GSM Association.

Contents

Intell	lectual Property Rights	4
Fore	word	4
Mod	lal verbs terminology	4
	oduction	
1	Scope	
2	References	
2.1	Normative references	
2.2	Informative references.	
3	Definition of terms, symbols and abbreviations	5
3.1	Terms	
3.2	Symbols	
3.3	Abbreviations	
4	Test Suite Structure	6
4.0	Introduction	6
4.1	Broker as SUT	6
4.2	Client as SUT	8
4.3	TP naming convention.	9
4.4	TP structure	9
5	Test Purposes for MQTT Broker	10
6	Test Purposes for MQTT Client	72
Anno	ex A (normative): MQTT Test Purposes (TPs)	89
A.0	Introduction	89
Histo	ory	90

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Methods for Testing and Specification (MTS).

The present document is part 1 of a multi-part deliverable covering the MQTT protocol as identified below:

Part 1: "Conformance Tests";

Part 2: "Security Tests";

Part 3: "Performance Tests".

Modal verbs terminology

In the present document "shall", "shall not", "should", "should not", "may", "need not", "will", "will not", "can" and "cannot" are to be interpreted as described in clause 3.2 of the <u>ETSI Drafting Rules</u> (Verbal forms for the expression of provisions).

"must" and "must not" are NOT allowed in ETSI deliverables except when used in direct citation.

Introduction

While the Internet of Things (IoT) is on the rise, the quality assurance of interconnected systems becomes an ever-increasing challenge. Within the last years, many different IoT protocols came to the fore. The MQ Telemetry Transport (MQTT) protocol is one of the most popular representatives as many surveys have shown.

Although many implementations for the MQTT protocol exist, it lacks in satisfying quality assurance. While many IoT components communicate over standardized protocols, communication protocols for IoT like MQTT or CoAP evolved over time without a holistic approach for quality assurance.

In the present document the conformance testing is presented. It provides a basis for interoperability testing and performance testing. The latter is presented in ETSI TS 103 597-3 [i.3].

1 Scope

The present document provides a test specification, i.e. an overall test suite structure and catalogue of test purposes for the MQ Telemetry Transport (MQTT). It will be a reference base for both client-side test campaigns and server-side test campaigns addressing the conformance issues.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

- [1] OASIS Standard: "MQTT Version 3.1.1".
- [2] ETSI ES 203 119-4: "Methods for Testing and Specification (MTS); The Test Description Language (TDL); Part 4: Structured Test Objective Specification (Extension)".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

- [i.1] ISO/IEC 9646-1: "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 1: General concepts".
- [i.2] ETSI ES 202 951: "Methods for Testing and Specification (MTS); Model-Based Testing (MBT); Requirements for Modelling Notations".
- [i.3] ETSI TS 103 597-3: "Methods for Testing and Specification (MTS); Test Specification for MQTT; Part 3: Performance Tests".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

conformance: extent to which an implementation of a standard satisfies the requirements expressed in that standard

conformance testing: process to verify to what extent the IUT conforms to the standard

Implementation Under Test (IUT): implementation of one or more Open Systems Interconnection (OSI) protocols in an adjacent user/provider relationship, being the part of a real open system, which is to be studied by testing (ISO/IEC 9646-1 [i.1])

system under test: real open system in which the implementation under test resides (ETSI ES 202 951 [i.2])

test purpose: non-formal high-level description of a test, mainly using text

test suite structure: document defining (hierarchical) grouping of test cases according to some rules

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

IUT Implementation Under Test
MQTT MQ Telemetry Transport
SUT System Under Test

TDL Test Description Language

TDL-TO Test Description Language - Test Objectives

TSS Test Suite Structure

4 Test Suite Structure

4.0 Introduction

The following two clauses describe the TSS. In the first one an MQTT server as SUT is considered and in the latter, an MQTT client as SUT is considered.

The structure itself is partly derived from the MQTT specification [1] but changed due to overlapping functions that cannot be tested separately.

4.1 Broker as SUT

- 1) All mandatory message data fields
 - a) CONNECT Control Packet
 - i) Fixed Header
 - 1) Header Flags
 - ii) Variable Header
 - 1) Protocol Name
 - 2) Protocol Level
 - 3) Reserved Flags
 - 4) Last Will Testament Flags
 - 5) Credentials Flags

- iii) Payload
 - 1) Client Identifier
 - 2) Will Topic
 - 3) Credentials
- b) CONNACK Control Packet
 - i) Fixed Header
 - ii) Variable Header
 - 1) Clean Session
 - 2) Present Session
 - 3) Return Codes
- c) SUBSCRIBE Control Packet
 - i) Fixed Header
 - 1) Header Flags
 - ii) Variable Header
 - 1) Packet Identifier
 - iii) Payload
 - 1) UTF-8 Encoding
 - 2) Topic Filter
 - 3) Requested QoS
- d) SUBACK Control Packet
 - i) Fixed Header
 - 1) Header Flags
 - ii) Variable Header
 - 1) Packet Identifier
 - iii) Payload
 - 1) Return Codes
- e) UNSUBSCRIBE Control Packet
 - i) Fixed Header
 - 1) Header Flags
 - ii) Variable Header
 - 1) Packet Identifier
 - iii) Payload
 - 1) UTF-8 Encoding
 - 2) Topic Filters

- f) UNSUBACK Control Packet
 - i) Fixed Header
 - ii) Variable Header
- g) PINGREQ Control Packet
 - i) Fixed Header
- h) PINGRESP Control Packet
 - i) Fixed Header
- i) DISCONNECT Control Packet
 - i) Fixed Header
- 2) Protocol features
 - a) General
 - i) QoS levels
 - ii) Delivery retransmission
 - iii) Retained messages
 - iv) Message ordering
 - v) Anonymous client identifier
 - b) Connect/disconnect (session handling)
 - i) Credentials
 - ii) Session initiation
 - iii) Session states
 - c) Subscribe
 - d) Unsubscribe
 - e) Immediate publish (w/o awaiting for CONNACK)
 - f) Last Will and Testament message
 - g) Heartbeats: keepAlive values
 - h) Topic names/filters
 - i) Error handling

4.2 Client as SUT

- 1) All mandatory message data fields
 - a) CONNECT Control Packet
 - b) CONNACK Control Packet
 - c) PUBLISH Control Packet
 - d) PUBACK Control Packet
 - e) PUBREC Control Packet

- f) UNSUBACK Control Packet
- g) PUBREL Control Packet
- h) PUBCOMP Control Packet
- i) SUBSCRIBE Control Packet
- j) UNSUBSCRIBE Control Packet
- k) DISCONNECT Control Packet
- 2) Protocol features
 - a) keepAlive values

4.3 TP naming convention

TPs are numbered, starting at 001, within each main scope. The main scopes are organized according to the TSS. Some TPs may not have a second level scope.

Table 1: TP identifier naming convention scheme

Identifier: TP_ <protocol>_<iut>_<scope>_<2nd_lvl_scope>*_<number>*</number></scope></iut></protocol>				
TP	=	Test Purpose	Fixed to TF	
<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>	=	Protocol name	Fixed to MQTT	
<iut></iut>	=	Type of IUT	Client or Broker	
<scope></scope>	=	Main scope	Scope of the protocol (feature)	
			<contro< td=""><td>DL PACKET> Name of the scoped Control Packet</td></contro<>	DL PACKET> Name of the scoped Control Packet
			FEAT	Protocol Features
<2nd_lvl_scope>	=	Second level scope	RTND	Retained Messages
<number></number>	=	Sequential number	From 001 t	to 999
*optional				

4.4 TP structure

Each TP has been written in TDL-TO and thus in a structured manner which is consistent with all other TPs. The intention of this is to make the TPs more formal. In addition, a more readable format is provided by generating tables out of the TDL-TO format. The defined structure, that has been used, is illustrated in table 2. This table should be read in conjunction with any TP, i.e. please use a TP as an example to facilitate the full comprehension of table 2. All structures are defined formally in the TDL Specification ETSI ES 203 119-4 [2].

Table 2: Structure of a single TP

TP part	Text	Example
Header	<identifier></identifier>	see table 1
	<test objective=""></test>	"The IUT has to close network connect"
	<reference></reference>	[MQTT-3.2.2-6]
	<pics reference=""></pics>	PIC_BROKER_BASIC
Initial	Free text description of the condition that the IUT	the IUT entity having a present session for the
condition	has reached before the test purpose applies.	CLIENT_ID entity
(optional)		
Start point	Describes the full logic of the test purpose.	Expected behavior
	Includes trigger and expected behavior of the IUT.	ensure that { }
Trigger	One or more actions that trigger an expected response of the IUT. Mostly a set of different messages the IUT receives.	when { the IUT entity receives a CONNECT message containing header_flags indicating value '1111'B; }
Expected behavior	Describes the response that the IUT sends after receiving a certain (set of) messages. This response describes the pass criteria	then { the IUT entity closes the TCP_CONNECTION }

5 Test Purposes for MQTT Broker

TP ld	TP_MQTT_BROKER_CONNECT_001		
Test Objective	Verify that the IUT closes 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]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {			
when {			
	ives a CONNECT message containing		
header_flags	indicating value '1111'B;		
}			
then {	then {		
the IUT closes the TCP_CONNECTION			
}			
}			
Final Conditions			

```
TP Id
                    TP_MQTT_BROKER_CONNECT_002
Test Objective
                    Verify that the IUT either disconnects the client or continues processing the CONNECT Control
                    Packet if the protocol name does not correspond to 'MQTT'.
                    [MQTT-3.1.2-1], [MQTT-3.1.4-4]
Reference
                    PICS_BROKER_BASIC
PICS Selection
                                                Initial Conditions
                                               Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME_INVALID,
       protocol_level indicating value 0x04;
  then {
       the IUT closes the TCP_CONNECTION
       // TODO: missing in TTCN-3 Implementation
       or the IUT sends a CONNACK message containing
       connect_return_code indicating value 0x00;
  }
                                                 Final Conditions
```

TP ld	TP_MQTT_BROKER_CONNECT_003		
Test Objective	Verify that the IUT responds to supported protocol levels (in scope: MQTT-3.1.1) with the return		
	code 0x00.		
Reference	[MQTT-3.1.2-2], [MQTT-3.1.4-4]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {	<u> </u>		
when {			
the IUT recei	ives a CONNECT message containing		
header_flags	s indicating value '0000'B,		
protocol_nan	ne corresponding to PROTOCOL_NAME,		
protocol_leve	el indicating value 0x04;		
}	}		
then {			
the IUT send	the IUT sends a CONNACK message containing		
connect_return_code indicating value 0x00;			
}			
}			
	Final Conditions		

```
TP Id
                   TP MQTT BROKER CONNECT 004
Test Objective
                   Verify that the IUT validates the reserved flags in the CONNECT Control Packet.
Reference
                   [MQTT-3.1.2-3], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection
                   PICS_BROKER_BASIC
                                              Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         reserved_field indicating value '1'B;
  then {
       the IUT closes the TCP CONNECTION
                                               Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_CONNECT_005
Test Objective
                    Verify that the IUT validates the will_topic and will_message fields if the will_flag is set to 1.
Reference
                    [MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
                    PICS_BROKER_BASIC and PICS_BROKER_LWT
PICS Selection
                                               Initial Conditions
                                              Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
      header_flags indicating value '0000'B,
      protocol_name corresponding to PROTOCOL_NAME,
      protocol_level indicating value 0x04,
       connect_flags containing
         will_flag indicating value '1'B,
         will_gos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
      payload containing
         will_topic indicating value omit,
         will_message indicating value omit;
  then {
      the IUT closes the TCP CONNECTION
  }
                                                Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_CONNECT_006
Test Objective
                    Verify that the IUT validates the the will_topic and will_message fields to be omitted if the will_flag
                    is set to 0.
                    [MQTT-3.1.2-11], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
Reference
                    PICS_BROKER_BASIC and PICS_BROKER_LWT and PICS_BROKER_RTND
PICS Selection
                                               Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         will_flag indicating value '0'B,
         will_qos corresponding to AT_LEAST_ONCE,
         will_retain indicating value '1'B,
         reserved_field indicating value '0'B;
       payload containing
         will_topic corresponding to PX_WILL_TOPIC,
         will_message corresponding to PX_WILL_MESSAGE;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                               Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNECT_007		
Test Objective	Verify that the IUT validates the will_qos field to be set to 0 if the will_flag is set to 0.		
Reference	[MQTT-3.1.2-13], [MQTT-3.1.4-1], [MQTT-3.2.2-6]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that { when {	CONNECT		
header_flags protocol_nam protocol_leve connect_flags will_flag inc will_qos co will_retain i	ves a CONNECT message containing indicating value '0000'B, see corresponding to PROTOCOL_NAME, I indicating value 0x04, secontaining dicating value '0'B, rresponding to AT_LEAST_ONCE, indicating value '0'B, ield indicating value '0'B;		
then { the IUT closes the TCP_CONNECTION }			
<i>y</i>	Final Conditions		

```
TP Id
                    TP MQTT BROKER CONNECT 008
Test Objective
                    Verify that the IUT validates the will gos field and rejects connections with an invalid will gos
                    value.
                    [MQTT-3.1.2-14], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
Reference
                    PICS_BROKER_BASIC and PICS_BROKER_LWT
PICS Selection
                                               Initial Conditions
                                              Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect flags containing
         will_flag indicating value '1'B,
         will_qos corresponding to INVALID_QOS,
         will_retain indicating value '0'B,
         reserved field indicating value '0'B;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                               Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_CONNECT_009
Test Objective
                    Verify that the IUT validates the will gos field if the will flag is set to 1.
Reference
                    [MQTT-3.1.2-14], [MQTT-3.1.4-4]
                    PICS_BROKER_BASIC and PICS_BROKER_LWT
PICS Selection
                                               Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         will_flag indicating value '1'B,
         will_gos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
  then {
       the IUT sends a CONNACK message containing
       connect_return_code indicating value 0x00;
  }
                                               Final Conditions
```

```
TP Id
                     TP_MQTT_BROKER_CONNECT_010
Test Objective
                     Verify that the IUT validates the will_flag and will_retain flags to be set correctly.
Reference
                     [MQTT-3.1.2-15], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection
                    PICS_BROKER_BASIC
                                                 Initial Conditions
                                                Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
          will_flag indicating value '0'B, will_qos corresponding to AT_MOST_ONCE,
          will_retain indicating value '1'B,
          reserved_field indicating value '0'B;
       payload containing
          will_topic indicating value omit,
          will_message indicating value omit;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                                  Final Conditions
```

TP ld	TP_MQTT_BROKER_CONNECT_011		
Test Objective	Verify that the IUT validates the will_retain flag to be set to 0 if the will_flag is set to 0.		
Reference	[MQTT-3.1.2-15], [MQTT-3.1.4-4]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {			
when {			
the IUT recei	ves a CONNECT message containing		
	indicating value '0000'B,		
	ne corresponding to PROTOCOL_NAME,		
	el indicating value 0x04,		
connect_flag			
	dicating value '0'B,		
	orresponding to AT_MOST_ONCE,		
	indicating value '0'B,		
reserved_	field indicating value '0'B;		
,			
payload cont			
	will_topic indicating value omit,		
will_message indicating value omit;			
; ,			
then (
then { the IUT sends a CONNACK message containing			
connect return code indicating value 0x00;			
to in equation of the contract			
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \			
Final Conditions			
	1 mai voliditiono		
<u> </u>			

```
TP Id
                    TP_MQTT_BROKER_CONNECT_012
Test Objective
                    Verify that the IUT validates the password flag to be set to 0 if the user name flag is set to 0.
Reference
                    [MQTT-3.1.2-22], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection
                   PICS_BROKER_BASIC and PICS_BROKER_AUTH
                                               Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         user name flag indicating value '0'B,
         password_flag indicating value '1'B,
         will_flag indicating value '0'B,
         will gos corresponding to AT_MOST_ONCE,
         will retain indicating value '0'B,
         reserved_field indicating value '0'B;
  }
  then {
       the IUT closes the TCP_CONNECTION
                                               Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_CONNECT_013
Test Objective
                    Verify that the IUT validates the username field to be omitted if the user_name_flag is set to 0.
                    [MQTT-3.1.2-18], [MQTT-3.1.2-22], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
Reference
                   PICS_BROKER_BASIC and PICS_BROKER_AUTH
PICS Selection
                                               Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         user_name_flag indicating value '0'B,
         password_flag indicating value '0'B,
         will_flag indicating value '0'B,
         will_qos corresponding to AT_MOST_ONCE.
         will_retain indicating value '0'B,
         reserved field indicating value '0'B;
       payload containing
         user_name corresponding to PX_MQTT_USER_NAME,
         password indicating value omit;
  then {
       the IUT closes the TCP_CONNECTION
                                               Final Conditions
```

```
TP Id
                     TP_MQTT_BROKER_CONNECT_014
Test Objective
                     Verify that the IUT validates a payload is present if the user_name_flag is set to 1.
                     [MQTT-3.1.2-19], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
Reference
PICS Selection
                    PICS_BROKER_AUTH
                                                 Initial Conditions
                                                Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         user_name_flag indicating value '1'B, password_flag indicating value '0'B,
         will_qos corresponding to AT_MOST_ONCE,
          will_retain indicating value '0'B,
          reserved_field indicating value '0'B;
       payload containing
          user_name indicating value omit,
          password indicating value omit;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                                  Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNECT_015		
Test Objective	Verify that the IUT validates the password field to be omitted if the password_flag is set to 0.		
Reference	[MQTT-3.1.2-20], [MQTT-3.1.2-22], [MQTT-3.1.4-1], [MQTT-3.2.2-6]		
PICS Selection	PICS Selection PICS_BROKER_BASIC and PICS_BROKER_AUTH		
	Initial Conditions		
	Expected Behaviour		
ensure that {	•		
when {			
	ves a CONNECT message containing		
	indicating value '0000'B,		
	ne corresponding to PROTOCOL_NAME,		
<u> </u>	el indicating value 0x04,		
connect_flags			
	e_flag indicating value '0'B,		
	_flag indicating value '0'B,		
	dicating value '0'B,		
	prresponding to AT_MOST_ONCE,		
	indicating value '0'B,		
reserved_r	field indicating value '0'B;		
, payload conta	oining		
	-		
	user_name indicating value omit, password corresponding to PX_MQTT_PASSWORD;		
password corresponding to 1 X_IMQ11_1 ACCIVIOND,			
} '			
then {			
the IUT closes the TCP_CONNECTION			
_			
<u> </u> }			
Final Conditions			

```
TP Id
                     TP_MQTT_BROKER_CONNECT_016
Test Objective
                     Verify that the IUT validates the password field to be present if the password_flag is set to 1.
Reference
                     [MQTT-3.1.2-21], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection
                    PICS_BROKER_AUTH
                                                 Initial Conditions
                                               Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         user_name_flag indicating value '1'B, password_flag indicating value '1'B,
          will_flag indicating value '0'B,
          will_qos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
          reserved_field indicating value '0'B;
       payload containing
          user_name corresponding to PX_MQTT_USER_NAME,
          password indicating value omit;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                                 Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNECT_017	
Test Objective	Verify that the IUT validates the client_identifier to be between 1 and 23 UTF-8 encoded bytes in	
	length.	
Reference	[MQTT-3.1.3-5], [MQTT-3.1.4-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
	Funcated Dehavious	
angura that (Expected Behaviour	
ensure that { when {		
	ives a CONNECT message containing	
	s indicating value '0000'B,	
	me corresponding to PROTOCOL_NAME,	
	el indicating value 0x04,	
	gs containing	
	ssion indicating value '1'B,	
	ne_flag indicating value '0'B,	
	_flag indicating value '0'B,	
	ndicating value '0'B,	
will_qos c	orresponding to AT_MOST_ONCE,	
will_retain indicating value '0'B,		
reserved_	field indicating value '0'B;	
, poylood son	taining	
payload containing		
client_identifier corresponding to CLIENT_ID_24_BYTES;		
}		
then {		
•	es the TCP_CONNECTION	
}	50 mg . 55525 511	
}		

Final Conditions

```
TP Id
                    TP_MQTT_BROKER_CONNECT_018
Test Objective
                    Verify that the IUT validates the client_identifier to contain only alphanumeric characters [0-9a-zA-
Reference
                    [MQTT-3.1.3-5], [MQTT-3.1.4-1]
PICS Selection
                    PICS_BROKER_BASIC
                                               Initial Conditions
                                              Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         clean_session indicating value '1'B,
         user_name_flag indicating value '0'B,
         password_flag indicating value '0'B,
         will_flag indicating value '0'B,
         will_qos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
       payload containing
         client_identifier corresponding to CLIENT_ID_NON_ALPHA_NUM;
  then {
       the IUT closes the TCP_CONNECTION
                                                Final Conditions
```

	<u>, </u>		
TP ld	TP_MQTT_BROKER_CONNECT_019		
Test Objective	Verify that the IUT accepts client_identifiers of zero byte length.		
Reference	[MQTT-3.1.3-6], [MQTT-3.1.3-7], [MQTT-3.1.4-4]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {			
when {			
	ves a CONNECT message containing		
	indicating value '0000'B,		
	ne corresponding to PROTOCOL_NAME,		
	el indicating value 0x04,		
connect_flag			
	sion indicating value '1'B,		
	e_flag indicating value '0'B,		
	flag indicating value '0'B,		
	dicating value '0'B,		
	prresponding to AT_MOST_ONCE,		
1	will_retain indicating value '0'B,		
reserved_field indicating value '0'B;			
, payload containing			
client_identifier corresponding to CLIENT_ID_ZERO_BYTES;			
, }			
then {			
// TODO: Standards says: MAY allow			
	•		

```
the IUT sends a CONNACK message containing connect_return_code indicating value 0x00;
}

Final Conditions
```

ΓP ld			
11 14	TP_MQTT_BROKER_CONNECT_020		
Test Objective			
Reference	[[
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {	·		
when {			
	ves a CONNECT message containing		
	indicating value '0000'B,		
	ne corresponding to PROTOCOL_NAME,		
•	el indicating value 0x04,		
connect_flags			
	sion indicating value '0'B,		
	e_flag indicating value '0'B,		
	flag indicating value '0'B,		
	dicating value '0'B,		
	presponding to AT_MOST_ONCE,		
	indicating value '0'B, ield indicating value '0'B;		
reserveu_n	ieu ilidicatilig value 0 b,		
, payload conta	aining		
client_identifier corresponding to CLIENT_ID_D800;			
·			
}			
then {			
the IUT closes the TCP_CONNECTION			
}			
•			
Final Conditions			

TP ld	TP_MQTT_BROKER_CONNECT_021		
Test Objective	Verify that the IUT validates the client_identifier to be a well-formed 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]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {			
when {			
the IUT recei	ves a CONNECT message containing		
	indicating value '0000'B,		
	ne corresponding to PROTOCOL_NAME,		
protocol_leve	el indicating value 0x04,		
connect_flag	s containing		
clean_ses	sion indicating value '0'B,		
	e_flag indicating value '0'B,		
	flag indicating value '0'B,		
	dicating value '0'B,		
	presponding to AT_MOST_ONCE,		
	will_retain indicating value '0'B,		
reserved field indicating value '0'B;			
Tool Too India malading Value ob,			
payload cont	aining		
client_identifier corresponding to CLIENT_ID_0000;			
	itilier corresponding to OLILIVI_ID_0000,		
,			

```
}
then {
the IUT closes the TCP_CONNECTION
}
}
Final Conditions
```

Test Objective Verify that the IUT validates the will_topic to be a well-formed 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] PICS Selection PICS_BROKER_LWT Initial Conditions Expected Behaviour	r	·
Reference [MQTT-1.5.3-1], [MQTT-3.1.3-10], [MQTT-3.1.4-1], [MQTT-3.2.2-6] PICS Selection PICS BROKER LWT Initial Conditions Expected Behaviour ensure that { when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_retain indicating value '0'B, reserved_field indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; then { the IUT closes the TCP_CONNECTION }	TP Id	
Initial Conditions Expected Behaviour ensure that { when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; , payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE;	Test Objective	Verify that the IUT validates the will_topic to be a well-formed UTF-8 encoded string.
Initial Conditions Expected Behaviour ensure that { when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing	Reference	[MQTT-1.5.3-1], [MQTT-3.1.3-10], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
Expected Behaviour ensure that { when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } } then { the IUT closes the TCP_CONNECTION }	PICS Selection	PICS_BROKER_LWT
when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_flag indicating value '1'B, will_eds corresponding to AT_MOST_ONCE, will_retain indicating value '0'B; reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }		Initial Conditions
when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_flag indicating value '1'B, will_eds corresponding to AT_MOST_ONCE, will_retain indicating value '0'B; reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }		
when { the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }		Expected Behaviour
the IUT receives a CONNECT message containing header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; , payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }	ensure that {	•
header_flags indicating value '0000'B, protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; , payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }	when {	
protocol_name corresponding to PROTOCOL_NAME, protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; then { the IUT closes the TCP_CONNECTION }		eives a CONNECT message containing
protocol_level indicating value 0x04, connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; then { the IUT closes the TCP_CONNECTION }	header_flag	s indicating value '0000'B,
connect_flags containing clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; , payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }	protocol_na	me corresponding to PROTOCOL_NAME,
clean_session indicating value '0'B, user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }	protocol_lev	vel indicating value 0x04,
user_name_flag indicating value '0'B, password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }	connect_fla	gs containing
password_flag indicating value '0'B, will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }		
will_flag indicating value '1'B, will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }		
will_qos corresponding to AT_MOST_ONCE, will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }	password	d_flag indicating value '0'B,
will_retain indicating value '0'B, reserved_field indicating value '0'B; payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; } then { the IUT closes the TCP_CONNECTION }		
reserved_field indicating value '0'B; , payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }		
payload containing will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }		•
will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }	reserved	_field indicating value '0'B;
will_topic indicating value WILL_TOPIC_D800, will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }	,	
will_message corresponding to PX_WILL_MESSAGE; ; } then { the IUT closes the TCP_CONNECTION }		•
; } then { the IUT closes the TCP_CONNECTION }		
the IUT closes the TCP_CONNECTION }	will_mess	sage corresponding to PX_WILL_MESSAGE;
the IUT closes the TCP_CONNECTION }	;	
the IUT closes the TCP_CONNECTION }	} thon (
}	,	pos the TCD, CONNECTION
Final Conditions	ine io i cios	SES THE TOP_CONTRECTION
Final Conditions) 1	
Filial Cultululis	<i>}</i>	Final Conditions
		Final Conditions

TP ld	TP_MQTT_BROKER_CONNECT_023
Test Objective	Verify that the IUT validates the will_topic to be a well-formed 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], [MQTT-4.7.3-2]
PICS Selection	PICS_BROKER_LWT
	Initial Conditions
	Expected Behaviour
header_flag protocol_na protocol_le connect_flag clean_se user_nai passwori will_flag will_qos will_retai reserved	reives a CONNECT message containing gs indicating value '0000'B, ame corresponding to PROTOCOL_NAME, vel indicating value 0x04, ags containing ession indicating value '0'B, me_flag indicating value '0'B, d_flag indicating value '0'B, indicating value '1'B, corresponding to AT_MOST_ONCE, in indicating value '0'B, l_field indicating value '0'B;
payload co	ntaining

```
will_topic indicating value WILL_TOPIC_0000,
    will_message corresponding to PX_WILL_MESSAGE;
;
} then {
    the IUT closes the TCP_CONNECTION
}
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNECT_024
Test Objective	Verify that the IUT validates the will_topic to not contain single-level topic filters.
Reference	[MQTT-4.7.1-1], [MQTT-3.1.4-1]
PICS Selection	PICS_BROKER_LWT
	Initial Conditions
	Expected Behaviour
ensure that {	
when {	
the IUT rece	ives a CONNECT message containing
connect_flag	
	ndicating value '1'B,
	orresponding to AT_MOST_ONCE,
	indicating value '0'B,
reserved_	field indicating value '0'B;
,	
payload cont	· ·
	indicating value TOPIC_FILTER_SINGLE_LEVEL,
will_mess	age corresponding to PX_WILL_MESSAGE;
;	
}	
then {	
the IU1 close	es the TCP_CONNECTION
}	
}	Final Conditions
	Final Conditions

TP Id	TP_MQTT_BROKER_CONNECT_025
Test Objective	Verify that the IUT validates the will_topic to not contain multi-level topic filters.
Reference	[MQTT-4.7.1-1], [MQTT-3.1.4-1]
PICS Selection	PICS_BROKER_LWT
	Initial Conditions
	Expected Behaviour
connect_flag: will_flag in: will_qos co will_retain reserved_f , payload conta will_topic i	dicating value '1'B, orresponding to AT_MOST_ONCE, indicating value '0'B, ield indicating value '0'B;
then { the IUT close } }	s the TCP_CONNECTION

Final Conditions

```
TP Id
                    TP_MQTT_BROKER_CONNECT_026
Test Objective
                    Verify that the IUT validates the user_name to be a well-formed UTF-8 encoded string.
                    [MQTT-1.5.3-1], [MQTT-3.1.3-11], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
Reference
PICS Selection
                   PICS BROKER AUTH
                                              Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         clean_session indicating value '0'B,
         user_name_flag indicating value '1'B,
         password_flag indicating value '1'B,
         will_flag indicating value '0'B,
         will_gos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
       payload containing
         user_name corresponding to MQTT_USER_NAME_INVALID_UTF8,
         password corresponding to PX_MQTT_PASSWORD;
  then {
       the IUT closes the TCP_CONNECTION
                                               Final Conditions
```

TP ld	TP_MQTT_BROKER_CONNECT_027
Test Objective	Verify that the IUT validates the user_name to be a well-formed 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]
PICS Selection	PICS_BROKER_AUTH
	Initial Conditions
	Expected Behaviour
header_flags protocol_nam protocol_leve connect_flags clean_sess user_name password_ will_flag ind will_qos co will_retain i reserved_fi , payload conta user_name	ves a CONNECT message containing indicating value '0000'B, are corresponding to PROTOCOL_NAME, I indicating value 0x04, as containing sion indicating value '0'B, and indicating value '1'B, are sponding value '1'B, are sponding to AT_MOST_ONCE, indicating value '0'B, are sponding to AT_MOST_ONCE, indicating value '0'B, are sponding value '0'B;
then {	

```
the IUT closes the TCP_CONNECTION
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNECT_028		
Test Objective	Verify that the IUT validates the first MQTT Control Packet sent from the client to the server after		
	a TCP connection is a MQTT CONNECT.		
Reference	[MQTT-3.1.0-1], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
-	YSTEM having a TCP_CONNECTION to the IUT		
}	<u> </u>		
	Expected Behaviour		
ensure that {			
when {			
the IUT recei	ives a PUBLISH message		
}	Ü		
then {			
the IUT closes the TCP_CONNECTION			
}			
}			
,	Final Conditions		

TD I I	TO MOTE PROVED COMMENT ON	
TP ld	TP_MQTT_BROKER_CONNECT_029	
Test Objective	Verify that the IUT detects multiple MQTT CONNECT Control Packets sent from a client within a	
	single session as a protocol violation.	
Reference	[MQTT-3.1.0-2]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the TEST_SY	STEM having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT receiv	ves a CONNECT message	
}	•	
then {		
the IUT closes the TCP_CONNECTION		
}		
}		
Final Conditions		

TP Id	TP_MQTT_BROKER_CONNECT_030
Test Objective	Verify that the IUT detects multiple clients with the same client_identifier and disconnects the
-	existing client.
Reference	[MQTT-3.1.4-2]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
with {	
the CLIEN	T_1 having a MQTT_CONNECTION to the IUT
and	
the CLIENT_1	established the MQTT_CONNECTION containing
payload contai	
client_identifi	er corresponding to PX_CLIENT_ID;
;	
to the IUT	
}	

```
ensure that {
    when {
        the IUT receives a CONNECT message containing
        payload containing
        client_identifier corresponding to PX_CLIENT_ID;
    ;
        from the CLIENT_2
    }
    then {
        the IUT closes the TCP_CONNECT to the CLIENT_1
        and
        the IUT sends a CONNACK message to the CLIENT_2
    }
}

Final Conditions
```

TP ld	TP_MQTT_BROKER_CONNECT_031
Test Objective	Verify that the IUT validates all topic names to be at least one character long.
Reference	[MQTT-4.7.3-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
	Expected Behaviour
ensure that {	·
when {	
the IUT recei	ives a CONNECT message containing
header_flags	s indicating value '0000'B,
protocol_nan	ne corresponding to PROTOCOL_NAME,
	el indicating value 0x04,
connect_flag	s containing
will_flag in	dicating value '1'B,
will_qos co	prresponding to AT_LEAST_ONCE,
will_retain	indicating value '0'B,
reserved_t	field indicating value '0'B;
,	
payload cont	aining
will_topic o	corresponding to TOPIC_NAME_ZERO_CHARS,
	age corresponding to PX_WILL_MESSAGE;
;	
}	
then {	
the IUT close	es the TCP_CONNECTION
}	
}	
,	Final Conditions

TP Id	TP_MQTT_BROKER_CONNECT_032	
Test Objective	Verify that the IUT does not process any data sent by the client after a rejected CONNECT	
	Control Packet.	
Reference	[MQTT-3.2.0-1]	
PICS Selection	PICS_BROKER_AUTH	
	Initial Conditions	
with {		
the IUT recei	the IUT received a CONNECT message containing	
connect_flags of	connect_flags containing	
user_name_	flag indicating value '1'B,	
password_fla	ag indicating value '1'B,	
payload containing		
user_name corresponding to MQTT_USER_NAME_INVALID,		
password corresponding to MQTT_PASSWORD_INVALID;;;		
}		

```
ensure that {
   when {
      the IUT receives a SUBSCRIBE message
   }
   then {
      the IUT sends a CONNACK message containing
      connect_return_code indicating value 0x05;
      and
      the IUT sends no SUBACK message
      and
      the IUT closes the TCP_CONNECTION
   }
}
```

TP ld	TP_MQTT_BROKER_CONNACK_001	
Test Objective	Verify that the IUT replies with a CONNACK Control Packet with valid header flags.	
Reference	[MQTT-2.2.2-1], [MQTT-3.1.4-4], [MQTT-3.2.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
	Expected Behaviour	
ensure that { when {	•	
the IUT recei	ves a CONNECT message containing	
header_flags	sindicating value '0000'B,	
	ne corresponding to PROTOCOL_NAME,	
	protocol_level indicating value 0x04,	
connect_flag	· · · · · · · · · · · · · · · · · · ·	
_	sion indicating value '0'B,	
	e_flag indicating value '0'B,	
	_flag indicating value '0'B,	
	dicating value '0'B,	
	orresponding to AT_MOST_ONCE,	
	indicating value '0'B,	
reserved_f	field indicating value '0'B;	
;		
}		
then {		
	s a CONNACK message containing	
header_flags indicating value '0000'B,		
connect_retu	rn_code indicating value 0x00;	
}		
}		
	Final Conditions	

TP Id	TP_MQTT_BROKER_CONNACK_002
Test Objective	Verify that the IUT responds to a CONNECT Control Packet with clean_session set to 1 with a
	CONNACK Control Packet with session_present_flag set to 0.
Reference	[MQTT-3.2.2-1], [MQTT-3.1.4-4], [MQTT-3.2.0-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions

```
Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect flags containing
         clean session indicating value '1'B,
         user_name_flag indicating value '0'B,
         password_flag indicating value '0'B,
         will_flag indicating value '0'B,
         will_qos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
       payload containing
         client_identifier corresponding to VALID_CLIENT_ID;
  then {
       the IUT sends a CONNACK message containing
       header_flags indicating value '0000'B,
       session present flag indicating value '0'B,
       connect return code indicating value 0x00;
  }
                                                 Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_CONNACK_003
Test Objective
                    Verify that the IUT responds to a CONNECT Control Packet for a present session with a
                    CONNACK Control Packet with session_present_flag set to 1.
Reference
                    [MQTT-3.2.2-2], [MQTT-3.1.4-4], [MQTT-3.2.0-1]
PICS Selection
                    PICS_BROKER_RTND
                                                Initial Conditions
with {
       the IUT having a present session for the CLIENT_ID
                                              Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         clean_session indicating value '0'B,
         user name flag indicating value '0'B,
         password_flag indicating value '0'B,
         will_flag indicating value '0'B,
         will_gos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
       payload containing
         client_identifier corresponding to PX_CLIENT_ID;
  then {
       the IUT sends a CONNACK message containing
       header_flags indicating value '0000'B,
       session_present_flag indicating value '1'B,
       connect_return_code indicating value 0x00;
  }
```

```
Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_CONNACK_004
Test Objective
                    Verify that the IUT responds to a CONNECT Control Packet with clean session set to 1 but not
                    having a present session for this client_identifier with a CONNACK Control Packet with
                    session_present_flag set to 0.
                    [MQTT-3.2.2-3], [MQTT-3.1.4-4], [MQTT-3.2.0-1]
Reference
PICS Selection
                    PICS_BROKER_BASIC and PICS_BROKER_RTND
                                                Initial Conditions
with {
       the IUT having no present session for the CLIENT_ID
                                              Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
       header_flags indicating value '0000'B,
       protocol_name corresponding to PROTOCOL_NAME,
       protocol_level indicating value 0x04,
       connect_flags containing
         clean_session indicating value '0'B,
         user_name_flag indicating value '0'B,
         password_flag indicating value '0'B,
         will_flag indicating value '0'B,
         will_gos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
       payload containing
         client_identifier corresponding to PX_CLIENT_ID;
  then {
       the IUT sends a CONNACK message containing
       header_flags indicating value '0000'B,
       session_present_flag indicating value '0'B,
       connect_return_code indicating value 0x00;
  }
                                                Final Conditions
```

TP ld	TP_MQTT_BROKER_CONNACK_005		
Test Objective	Verify that the IUT responds to protocol levels which it does not support with return code 0x01.		
Reference	[MQTT-3.1.2-2], [MQTT-3.2.2-4], [MQTT-3.2.2-5], [MQTT-3.2.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {			
when {			
the IUT recei	ves a CONNECT message containing		
header_flags	sindicating value '0000'B,		
protocol_nan	ne corresponding to PROTOCOL_NAME,		
protocol leve	el indicating value 0xFF,		
connect_flag			
_	clean session indicating value '0'B,		
_	e_flag indicating value '0'B,		
	password_flag indicating value '0'B,		
will_flag indicating value '0'B,			
will_qos corresponding to AT_MOST_ONCE,			
_	indicating value '0'B,		
reserved_f	field indicating value '0'B;		
,	· ·		

```
payload containing
    client_identifier corresponding to PX_CLIENT_ID;
;
}
then {
    the IUT sends a CONNACK message containing
    header_flags indicating value '0000'B,
    session_present_flag indicating value '0'B,
    connect_return_code indicating value 0x01;
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNACK_006
Test Objective	Verify that the IUT responds to CONNECT Control Packets with a zero-byte client_identifier and
1031 Objective	clean_session set to 0 with CONNACK with connect_return_code set to 0x02 and 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], [MQTT-3.2.0-1]
PICS Selection	PICS_BROKER_RTND
i ioo oelection	Initial Conditions
	Expected Behaviour
ensure that {	
when {	
	eives a CONNECT message containing
	s indicating value '0000'B,
	me corresponding to PROTOCOL_NAME,
	vel indicating value 0x04,
	gs containing
	ssion indicating value '0'B,
	ne_flag indicating value '0'B,
	I_flag indicating value '0'B,
	ndicating value '0'B,
	corresponding to AT_MOST_ONCE,
	n indicating value '0'B,
reserved_	_field indicating value '0'B;
,	
payload cor	
client_ide	entifier corresponding to CLIENT_ID_ZERO_BYTES;
;	
}	
then {	da a CONNIA OK masaa ma aanta'n in m
	ds a CONNACK message containing
	s indicating value '0000'B,
	esent_flag indicating value '0'B,
	urn_code indicating value 0x02;
_	closes the TCP_CONNECTION
}	
}	Final Conditions
	i mai conditiono
L	

TP Id	TP_MQTT_BROKER_CONNACK_007
Test Objective	Verify that the IUT responds to a CONNECT Control Packet with a malformed user_name with
	CONNACK with connect_return_code set to 0x04.
Reference	[MQTT-3.2.2-4], [MQTT-3.2.2-5], [MQTT-3.2.0-1]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_AUTH
Initial Conditions	

```
Expected Behaviour
ensure that {
  when {
      the IUT receives a CONNECT message containing
      header_flags indicating value '0000'B,
      protocol_name corresponding to PROTOCOL_NAME,
      protocol_level indicating value 0x04,
      connect_flags containing
         clean_session indicating value '0'B,
         user_name_flag indicating value '1'B,
         password_flag indicating value '1'B,
         will_flag indicating value '0'B,
         will_qos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
      payload containing
         client_identifier corresponding to PX_CLIENT_ID,
         user_name corresponding to MQTT_USER_NAME_INVALID_UTF8,
         password corresponding to PX_MQTT_PASSWORD;
  then {
      the IUT sends a CONNACK message containing
      header_flags indicating value '0000'B,
      session_present_flag indicating value '0'B,
      connect_return_code indicating value 0x04;
      and the IUT closes the TCP_CONNECTION
  }
                                               Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNACK_008
Test Objective	Verify that the IUT responds to a CONNECT Control Packet with a invalid user_name with
-	CONNACK with connect_return_code set to 0x05.
Reference	[MQTT-3.2.2-4], [MQTT-3.2.2-5], [MQTT-3.2.0-1]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_AUTH
	Initial Conditions
	Expected Behaviour
ensure that {	
when {	
	ves a CONNECT message containing
	indicating value '0000'B,
	ne corresponding to PROTOCOL_NAME,
	el indicating value 0x04,
connect_flag	s containing sion indicating value '0'B,
	e_flag indicating value '0'B,
	flag indicating value '1'B,
	dicating value '0'B,
	orresponding to AT_MOST_ONCE,
	indicating value '0'B,
	ield indicating value '0'B;
,	,
payload conta	aining
client_iden	tifier corresponding to PX_CLIENT_ID,
	e corresponding to MQTT_USER_NAME_INVALID,
password	corresponding to PX_MQTT_PASSWORD;
;	
}	
then {	OONINAOK
the IUT sends a CONNACK message containing	
header_flags indicating value '0000'B, session_present_flag indicating value '0'B,	
connect_retu	rn_code indicating value 0x05;

```
and the IUT closes the TCP_CONNECTION
}

Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_CONNACK_009
Test Objective
                   Verify that the IUT responds to a CONNECT Control Packet with a invalid password with
                   CONNACK with connect_return_code set to 0x05.
Reference
                   [MQTT-3.2.2-4], [MQTT-3.2.2-5], [MQTT-3.2.0-1]
PICS Selection
                   PICS BROKER BASIC and PICS BROKER AUTH
                                              Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a CONNECT message containing
      header_flags indicating value '0000'B,
      protocol_name corresponding to PROTOCOL_NAME,
      protocol level indicating value 0x04,
      connect_flags containing
         clean_session indicating value '0'B,
         user_name_flag indicating value '1'B,
         password_flag indicating value '1'B,
         will_flag indicating value '0'B,
         will_gos corresponding to AT_MOST_ONCE,
         will_retain indicating value '0'B,
         reserved_field indicating value '0'B;
      payload containing
         client_identifier corresponding to PX_CLIENT_ID,
         user_name corresponding to PX_MQTT_USER_NAME,
         password corresponding to MQTT_PASSWORD_INVALID;
  then {
       the IUT sends a CONNACK message containing
      header_flags indicating value '0000'B,
      session present flag indicating value '0'B.
      connect_return_code indicating value 0x05;
      and the IUT closes the TCP_CONNECTION
  }
                                               Final Conditions
```

TP Id	TP_MQTT_BROKER_CONNACK_010	
Test Objective	Verify that the IUT responds with CONNECT with connect_return_code set to 0x03 if the MQTT	
	service is unavailable.	
Reference	[MQTT-3.2.2-4], [MQTT-3.2.2-5], [MQTT-3.2.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the IUT havir	ng no available service for the MQTT_CONNECTION	
}	}	
	Expected Behaviour	
ensure that {	ensure that {	
when {		
the IUT recei	ves a CONNECT message containing	
header_flags	header_flags indicating value '0000'B,	
protocol_name corresponding to PROTOCOL_NAME,		
protocol_level indicating value 0x04,		
connect_flags containing		
clean_session indicating value '0'B,		
user_name_flag indicating value '0'B,		

```
password_flag indicating value '0'B,
will_flag indicating value '0'B,
will_qos corresponding to AT_MOST_ONCE,
will_retain indicating value '0'B,
reserved_field indicating value '0'B;

payload containing
client_identifier corresponding to PX_CLIENT_ID;
;
}
then {
the IUT sends a CONNACK message containing
header_flags indicating value '0000'B,
session_present_flag indicating value '0'B,
connect_return_code indicating value 0x03;
and the IUT closes the TCP_CONNECTION
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_PUBLISH_001		
Test Objective	Verify that the IUT accepts only QoS 0 PUBLISH Control Packets with the dup_flag set to 0.		
Reference	[MQTT-3.3.1-2]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the TEST_SY	STEM having a MQTT_CONNECTION to the IUT		
}			
	Expected Behaviour		
ensure that {			
when {			
the IUT receive	ves a PUBLISH message containing		
header containii			
dup_flag indica	ating value '1'B,		
qos_level corre	qos_level corresponding to AT_MOST_ONCE;		
;			
}	}		
then {	then {		
the IUT closes the TCP_CONNECTION			
}			
}			
Final Conditions			
	_		

TP ld	TP_MQTT_BROKER_PUBLISH_002	
Test Objective	Verify that the IUT accepts only PUBLISH Control Packets with a valid QoS level.	
Reference	[MQTT-2.2.2-1], [MQTT-3.3.1-4]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the TEST_S\	YSTEM having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT recei	ves a PUBLISH message containing	
header containi	header containing	
qos_level corre	qos_level corresponding to INVALID_QOS;	
;		
}		
then {	then {	
the IUT closes the TCP_CONNECTION		
}		
}		

Final Conditions

TP MQTT BROKER PUBLISH 003	
Verify that the IUT validates the topic_name in a PUBLISH Control Packet to be a well-formed	
UTF-8 encoded string.	
0	
[MQTT-3.3.2-1], [MQTT-4.8.0-1]	
PICS_BROKER_BASIC	
Initial Conditions	
STEM having a MQTT_CONNECTION to the IUT	
5 –	
Expected Behaviour	
ves a PUBLISH message containing	
topic_name corresponding to TOPIC_NAME_INVALID_UTF8;	
topic_name conceptioning to 10.110_17.181E_17477.EED_011.0,	
then { the IUT closes the TCP_CONNECTION	
the for closes the for_connection	
F. 10 W.	
Final Conditions	

TP Id	TP_MQTT_BROKER_PUBLISH_004	
Test Objective	Verify that the IUT validates the topic_name in a PUBLISH Control Packet to not contain multi-	
-	level wildcard characters.	
Reference	[MQTT-3.3.2-2], [MQTT-4.7.1-1], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the TEST_SY	STEM having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT receive	ves a PUBLISH message containing	
topic_name corresponding to TOPIC_NAME_WC_MULTI_LVL;		
}		
then {		
the IUT closes the TCP_CONNECTION		
}		
}		
Final Conditions		

TP ld	TP_MQTT_BROKER_PUBLISH_005
Test Objective	Verify that the IUT validates the topic_name in a PUBLISH Control Packet to not contain single-
	level wildcard characters.
Reference	[MQTT-3.3.2-2], [MQTT-4.7.1-1], [MQTT-4.8.0-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
with {	
the TEST	SYSTEM having a MQTT_CONNECTION to the IUT
}	•
	Expected Behaviour
ensure that {	
when {	
the IUT rec	eives a PUBLISH message containing
topic_name corresponding to TOPIC_NAME_WC_SINGLE_LVL;	
}	·
then {	
the IUT clo	ses the TCP_CONNECTION

```
}
Final Conditions
```

TO MOTE DEGLED DUBLICH AND		
TP_MQTT_BROKER_PUBLISH_006		
Verify that the IUT validates the topic_name in a PUBLISH Control Packet to be at least on		
character long.		
[MQTT-4.7.3-1], [MQTT-4.8.0-1]		
PICS_BROKER_BASIC		
Initial Conditions		
/STEM having a MQTT_CONNECTION to the IUT		
-		
Expected Behaviour		
ensure that { when {		
ves a PUBLISH message containing		
topic_name corresponding to TOPIC_NAME_ZERO_CHARS;		
then {		
the IUT closes the TCP_CONNECTION		
Final Conditions		
,		

TP Id	TP_MQTT_BROKER_PUBLISH_007	
Test Objective	Verify that the IUT validates the topic_name in a PUBLISH Control Packet to not contain the null	
	character (Unicode U+0000).	
Reference	[MQTT-4.7.3-2], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the TEST_S	YSTEM having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
	ves a PUBLISH message containing	
topic_name cor	topic_name corresponding to TOPIC_NAME_0000;	
}		
then {		
the IUT closes the TCP_CONNECTION		
}		
<u>}</u>		
Final Conditions		

TP Id	TP_MQTT_BROKER_PUBLISH_008	
Test Objective	Verify that the IUT rejects QoS 0 PUBLISH Control Packets with the dup_flag set to 1.	
Reference	[MQTT-4.3.1-1], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
Initial Conditions		
with {		
the TEST_SYSTEM having a MQTT_CONNECTION to the IUT		
}		

```
Expected Behaviour

ensure that {
    when {
        the IUT receives a PUBLISH message containing
        header containing
        dup_flag indicating value '1'B,
        qos_level corresponding to AT_MOST_ONCE;
    ;
    }
    then {
        the IUT closes the TCP_CONNECTION
    }
}
```

TP Id	TP_MQTT_BROKER_PUBLISH_009			
Test Objective	Verify that the IUT validates a QoS 0 PUBLISH Control Packet to not contain a packet_identifier.			
Reference	[MQTT-2.3.1-5], [MQTT-4.8.0-1]			
PICS Selection	PICS_BROKER_BASIC			
	Initial Conditions			
with {				
the TEST_SYSTEM having a MQTT_CONNECTION to the IUT				
}				
Expected Behaviour				
ensure that {				
when {	when {			
the IUT receive	the IUT receives a PUBLISH message containing			
	header containing			
	esponding to AT_MOST_ONCE,			
packet_identifier corresponding to PACKET_ID;				
;				
}				
then {	then {			
the IUT closes the TCP_CONNECTION				
}				
}				
Final Conditions				

	-		
TP Id	TP_MQTT_BROKER_PUBLISH_010		
Test Objective	Verify that the IUT validates a QoS 1 PUBLISH Control Packet to contain a non-zero 16-bit		
-	Packet Identifier.		
Reference	[MQTT-2.3.1-1], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_QOS_1		
Initial Conditions			
with {			
the TEST SY	the TEST_SYSTEM having a MQTT_CONNECTION to the IUT		
}	5		
•	Expected Behaviour		
ensure that {	·		
when {			
the IUT receives a PUBLISH message containing			
	header containing		
	esponding to AT_LEAST_ONCE,		
	packet_identifier corresponding to PACKET_ID_ZERO;		
:			
} '			
then {			
the IUT closes the TCP_CONNECTION			
}	*** * 2** * * *		
}			
Final Conditions			

```
TP Id
                    TP_MQTT_BROKER_PUBLISH_011
                    Verify that the IUT validates a QoS 2 PUBLISH Control Packet to contain a non-zero 16-bit
Test Objective
                    Packet Identifier.
                    [MQTT-2.3.1-1], [MQTT-4.8.0-1]
Reference
PICS Selection
                    PICS_BROKER_QOS_2
                                                Initial Conditions
with {
       the TEST_SYSTEM having a MQTT_CONNECTION to the IUT
                                               Expected Behaviour
ensure that {
  when {
       the IUT receives a PUBLISH message containing
    header containing
    qos_level corresponding to EXACTLY_ONCE, packet_identifier corresponding to PACKET_ID_ZERO;
  then {
       the IUT closes the TCP_CONNECTION
                                                 Final Conditions
```

TP Id	TP MQTT BROKER PUBLISH 012		
Test Objective	Verify that the IUT handles not authorized PUBLISH Control Packets with either a positive		
Test objective	PUBACK Control Packet or by closing the network connection.		
Reference			
	[MQTT-3.3.5-2]		
PICS Selection	PICS_BROKER_QOS_1 Initial Conditions		
	initial Conditions		
with {	CTEM having a MOTT CONNECTION to the ILIT		
the rest_st	STEM having a MQTT_CONNECTION to the IUT		
})		
Expected Behaviour			
ensure that {			
•	when {		
	the IUT receives a PUBLISH message containing		
header containing			
qos_level corresponding to AT_LEAST_ONCE,			
topic_name cor	responding to TOPIC_NAME_RESTRICTED;		
;			
}			
then {			
the IUT sends a PUBACK message			
or	•		
the IUT closes the TCP_CONNECTION			
}			
}			
Final Conditions			
L			

TP Id	TP_MQTT_BROKER_PUBACK_001		
Test Objective	Verify that the IUT closes the network connection if fixed header flags in PUBACK Control Packet		
_	are invalid.		
Reference	[MQTT-2.2.2-1], [MQTT-2.2.2-2]		
PICS Selection	PICS_BROKER_QOS_1		
Initial Conditions			
with {			
the CLIENT_1 having a MQTT_CONNECTION to the IUT			
and			
the CLIENT_2 having a MQTT_CONNECTION to the IUT			
and			
the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing			
qos_level o	qos_level corresponding to AT_LEAST_ONCE;		

```
and
    the CLIENT_1 delivered a PUBLISH message containing
      qos_level corresponding to AT_LEAST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC;
    to the IUT
                                           Expected Behaviour
ensure that {
  when {
      the IUT sends a PUBLISH message containing
   header containing
    gos_level corresponding to AT_LEAST_ONCE,
   topic_name corresponding to PX_PUBLISH_TOPIC;
   to the CLIENT_2
   and
   the IUT receives a PUBACK message containing
   header_flags indicating value '1111'B;
   from the CLIENT_2
  then {
      the IUT closes the TCP CONNECTION to the CLIENT 2
  }
                                              Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_PUBACK_002
Test Objective
                   Verify that the IUT sends PUBACK Control Packets in the order in which the corresponding QoS
                   1 PUBLISH Control Packets were received.
Reference
                   [MQTT-4.6.0-2], [MQTT-3.3.4-1], [MQTT-4.6.0-6], [MQTT-2.3.1-6]
PICS Selection
                   PICS_BROKER_QOS_1
                                             Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    header containing
     qos_level corresponding to AT_LEAST_ONCE,
    topic name corresponding to PX PUBLISH TOPIC,
    packet_identifier corresponding to PACKET_ID_1;
    the IUT receives a PUBLISH message containing
    header containing
     qos_level corresponding to AT_LEAST_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_2;
  then {
      the IUT sends a PUBACK message containing
     packet_identifier corresponding to PACKET_ID_1;
    and
    the IUT sends a PUBACK message containing
     packet_identifier corresponding to PACKET_ID_2;
  }
                                              Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_PUBREC_001
Test Objective
                   Verify that the IUT closes the network connection if fixed header flags in PUBREC Control Packet
                   are invalid.
Reference
                   [MQTT-2.2.2-1], [MQTT-2.2.2-2]
PICS Selection
                   PICS_BROKER_QOS_2
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_2 having a MQTT_CONNECTION to the IUT
    the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
      gos_level corresponding to EXACTLY_ONCE;
    and
    the CLIENT_1 delivered a PUBLISH message containing
      qos_level corresponding to EXACTLY_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC;
    to the IUT
                                           Expected Behaviour
ensure that {
  when {
      the IUT sends a PUBLISH message containing
   header containing
    gos_level corresponding to EXACTLY_ONCE,
   topic_name corresponding to PX_PUBLISH_TOPIC;
   to the CLIENT_2
   and
   the IUT receives a PUBREC message containing
   header_flags indicating value '1111'B;
   from the CLIENT_2
  then {
      the IUT closes the TCP_CONNECTION to the CLIENT_2
                                             Final Conditions
```

TP Id	TP_MQTT_BROKER_PUBREC_002	
Test Objective	Verify that the IUT sends PUBREC Control Packets in the order in which the corresponding QoS	
	2 PUBLISH Control Packets were received.	
Reference	[MQTT-4.6.0-3], [MQTT-3.3.4-1], [MQTT-4.6.0-6], [MQTT-2.3.1-6]	
PICS Selection	PICS_BROKER_QOS_2	
	Initial Conditions	
with {		
the CLIENT	having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT rece	ives a PUBLISH message containing	
header contain		
	responding to EXACTLY_ONCE,	
	rresponding to PX_PUBLISH_TOPIC,	
packet_identifie	er corresponding to PACKET_ID_1;	
;		
and		
	es a PUBLISH message containing	
header contain		
	responding to EXACTLY_ONCE,	
topic_name corresponding to PX_PUBLISH_TOPIC,		
packet_identifie	er corresponding to PACKET_ID_2;	
;		
}		
then {		

```
the IUT sends a PUBREC message containing
   packet_identifier corresponding to PACKET_ID_1;
   and
   the IUT sends a PUBREC message containing
   packet_identifier corresponding to PACKET_ID_2;
}

Final Conditions
```

	1		
TP ld	TP_MQTT_BROKER_PUBREL_001		
Test Objective	Verify that the IUT closes the network connection if fixed header flags in PUBREL Control Packet		
	are invalid.		
Reference	[MQTT-2.2.2-1], [MQTT-2.2.2-2], [MQTT-3.6.1-1]		
PICS Selection	PICS_BROKER_QOS_2		
	Initial Conditions		
with {			
the CLIENT_	1 having a MQTT_CONNECTION to the IUT		
}			
	Expected Behaviour		
ensure that {			
when {			
	ves a PUBLISH message containing		
header containin			
. –	sponding to EXACTLY_ONCE,		
	topic_name corresponding to PX_PUBLISH_TOPIC;;		
from the CLIENT	<u>_</u> 1		
and			
	PUBREC message		
from the CLIENT	from the CLIENT_1		
and			
	a PUBREL message containing		
header_flags ind	icating value '1101'B;		
from the CLIENT	_1		
}			
then {			
the IUT closes the TCP_CONNECTION to the CLIENT_1			
\ \			
}			
	Final Conditions		

TP ld	TP_MQTT_BROKER_PUBREL_002	
Test Objective	Verify that the IUT sends PUBREL Control Packets in the order in which the corresponding	
	PUBREC Control Packets were received.	
Reference	[MQTT-4.6.0-4], [MQTT-4.6.0-6], [MQTT-2.3.1-6]	
PICS Selection	PICS_BROKER_QOS_2	
	Initial Conditions	
with {		
the CLIENT	_1 having a MQTT_CONNECTION to the IUT	
and		
the CLIENT_2	having a MQTT_CONNECTION to the IUT	
and		
	subscribed the PX_PUBLISH_TOPIC containing	
qos_level co	qos_level corresponding to EXACTLY_ONCE;	
and		
	delivered a PUBLISH message containing	
	prresponding to EXACTLY_ONCE,	
	topic_name corresponding to PX_PUBLISH_TOPIC,	
	packet_identifier corresponding to PACKET_ID_1;	
to the IUT		
and		
the CLIENT_1 delivered a PUBLISH message containing		
	qos_level corresponding to EXACTLY_ONCE,	
	corresponding to PX_PUBLISH_TOPIC,	
packet_iden	tifier corresponding to PACKET_ID_2;	

```
to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT sends a PUBLISH message containing
   header containing
    qos_level corresponding to EXACTLY_ONCE,
   topic_name corresponding to PX_PUBLISH_TOPIC,
   packet_identifier corresponding to PACKET_ID_1;;
   to the CLIENT_2
   and
   the IUT receives a PUBREC message containing
   packet_identifier corresponding to PACKET_ID_1;
   from the CLIENT_2
   and
   the IUT sends a PUBLISH message containing
   header containing
    qos_level corresponding to EXACTLY_ONCE,
   topic_name corresponding to PX_PUBLISH_TOPIC,
   packet_identifier corresponding to PACKET_ID_2;;
   to the CLIENT_2
   and
   the IUT receives a PUBREC message containing
   packet_identifier corresponding to PACKET_ID_2;
   from the CLIENT_2
  then {
      the IUT sends a PUBREL message containing
     packet_identifier corresponding to PACKET_ID_1;
    and
    the IUT sends a PUBREL message containing
     packet_identifier corresponding to PACKET_ID_2;
  }
                                              Final Conditions
```

TP Id	TP_MQTT_BROKER_PUBCOMP_001		
Test Objective	Verify that the IUT closes the network connection if fixed header flags in PUBCOMP Control		
-	Packet are invalid.		
Reference	[MQTT-2.2.2-1], [MQTT-2.2.2-2]		
PICS Selection	PICS_BROKER_QOS_2		
	Initial Conditions		
with {			
the CLIENT	_1 having a MQTT_CONNECTION to the IUT		
and			
the CLIENT_2	having a MQTT_CONNECTION to the IUT		
and			
the CLIENT_2	subscribed the PX_PUBLISH_TOPIC containing		
qos_level co	presponding to EXACTLY_ONCE;		
and			
	delivered a PUBLISH message containing		
	orresponding to EXACTLY_ONCE,		
	corresponding to PX_PUBLISH_TOPIC;		
to the IUT			
}			
	Expected Behaviour		
ensure that {			
when {			
the IUT sends a PUBLISH message containing			
header containii			
	esponding to EXACTLY_ONCE,		
	responding to PX_PUBLISH_TOPIC;;		
to the CLIENT_2			
and			
the IUT receives a PUBREC message			

```
from the CLIENT_2
and
the IUT sends a PUBREL message
to the CLIENT_2
and
the IUT receives a PUBCOMP message containing
header_flags indicating value '1111'B;
from the CLIENT_2
}
then {
the IUT closes the TCP_CONNECTION to the CLIENT_2
}
Final Conditions
```

TP Id	TP_MQTT_BROKER_SUBSCRIBE_001	
Test Objective	Verify that 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], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT	having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
	vives a SUBSCRIBE message containing	
header_flags	header_flags indicating value '1101'B;	
}		
then {		
the IUT closes the TCP_CONNECTION		
}		
}		
Final Conditions		

TP Id	TP_MQTT_BROKER_SUBSCRIBE_002	
Test Objective	Verify that the IUT validates a SUBSCRIBE Control Packet to contain a non-zero 16-bit Packet	
	Identifier.	
Reference	[MQTT-2.3.1-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT I	having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
	ves a SUBSCRIBE message containing	
	header_flags indicating value '0010'B,	
	ifier corresponding to PACKET_ID_ZERO,	
payload cont	•	
	topic_filter corresponding to PX_SUBSCRIBE_TOPIC_FILTER,	
requested_	requested_qos corresponding to AT_LEAST_ONCE;	
;		
}		
then {		
the IUT close	the IUT closes the TCP_CONNECTION	
}		
}		
Final Conditions		

```
TP Id
                   TP_MQTT_BROKER_SUBSCRIBE_003
Test Objective
                   Verify that the IUT validates the topic_filter in a SUBSCRIBE Control Packet to be a well-formed
                   UTF-8 encoded string and do not contain code points between U+D800 and U+DFFF.
                   [MQTT-1.5.3-1], [MQTT-3.8.3-1], [MQTT-4.8.0-1]
Reference
PICS Selection
                   PICS_BROKER_BASIC
                                             Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
       the IUT receives a SUBSCRIBE message containing
       header_flags indicating value '0010'B,
       packet_identifier corresponding to PACKET_ID,
       payload containing
         topic_filter corresponding to TOPIC_FILTER_D800,
         requested_qos corresponding to AT_LEAST_ONCE;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                              Final Conditions
```

TP Id	TO MOTT PROVED SUBSCIDE 004		
	TP_MQTT_BROKER_SUBSCRIBE_004		
Test Objective	Verify that the IUT validates the topic_filter in a SUBSCRIBE Control Packet to be a well-formed		
	UTF-8 encoded string and do not contain the null character (Unicode U+0000).		
Reference	[MQTT-1.5.3-2], [MQTT-3.8.3-1], [MQTT-4.7.3-2], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT	having a MQTT_CONNECTION to the IUT		
}	5		
•	Expected Behaviour		
ensure that {	Francis Control		
when {			
,	ives a SUBSCRIBE massage containing		
	the IUT receives a SUBSCRIBE message containing		
	header_flags indicating value '0010'B, packet_identifier corresponding to PACKET_ID,		
payload con			
	topic_filter corresponding to TOPIC_FILTER_0000,		
requested	requested_qos corresponding to AT_LEAST_ONCE;		
;			
}			
then {			
the IUT closes the TCP_CONNECTION			
}			
}			
•	Final Conditions		

TP ld	TP MQTT BROKER SUBSCRIBE 005		
Test Objective	Verify that the IUT validates the topic_filter in a SUBSCRIBE Control Packet to be at least on		
-	character long.		
Reference	[MQTT-4.7.3-1], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT	the CLIENT having a MQTT_CONNECTION to the IUT		
Expected Behaviour			
ensure that {			

```
when {
    the IUT receives a SUBSCRIBE message containing
    header_flags indicating value '0010'B,
    packet_identifier corresponding to PACKET_ID,
    payload containing
    topic_filter corresponding to TOPIC_FILTER_ZERO_CHARS,
    requested_qos corresponding to AT_LEAST_ONCE;
    ;
}
then {
    the IUT closes the TCP_CONNECTION
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_SUBSCRIBE_006
Test Objective	Verify that the IUT validates a SUBSCRIBE Control Packet to contain at least one topic filter/QoS
	pair.
Reference	[MQTT-3.8.3-3], [MQTT-4.8.0-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
with { the CLIENT h }	naving a MQTT_CONNECTION to the IUT
	Expected Behaviour
header_flags packet_ident payload cont omit; ; } then {	es the TCP_CONNECTION
Final Conditions	

TP ld	TP_MQTT_BROKER_SUBSCRIBE_007		
Test Objective	Verify that the IUT validates in a SUBSCRIBE Control Packet the upper 6 bits of a requested QoS		
	byte (reserved bits) to be set to 0.		
Reference	[MQTT-3.8.3-4], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT h	naving a MQTT_CONNECTION to the IUT		
}			
	Expected Behaviour		
ensure that {			
when {			
	ves a SUBSCRIBE message containing		
	indicating value '0010'B,		
	packet_identifier corresponding to PACKET_ID,		
	payload containing		
-	corresponding to PX_SUBSCRIBE_TOPIC_FILTER,		
	requested_qos corresponding to AT_MOST_ONCE,		
requested_	_qos_flags indicating value '111111'B;		
;			
}			
then {			
the IUT close	s the TCP_CONNECTION		

```
}
Final Conditions
```

```
TP Id
                    TP_MQTT_BROKER_SUBSCRIBE_008
Test Objective
                    Verify that the IUT validates the requested_qos field to be a valid QoS level.
                   [MQTT-3.8.3-4], [MQTT-4.8.0-1]
PICS_BROKER_BASIC
Reference
PICS Selection
                                               Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a SUBSCRIBE message containing
       header_flags indicating value '0010'B,
       packet_identifier corresponding to PACKET_ID,
       payload containing
         topic_filter corresponding to PX_SUBSCRIBE_TOPIC_FILTER,
         requested_qos corresponding to INVALID_QOS;
  then {
       the IUT closes the TCP_CONNECTION
  }
                                               Final Conditions
```

TP ld	P Id TP_MQTT_BROKER_SUBSCRIBE_009	
Test Objective	Verify that the IUT validates topic_filter field to be a valid multi-level Topic Filter.	
Reference	[MQTT-4.7.1-2], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with { the CLIENT }	with { the CLIENT having a MQTT_CONNECTION to the IUT }	
	Expected Behaviour	
header_flage packet_iden payload con topic_filten requested ; } then {	eives a SUBSCRIBE message containing s indicating value '0010'B, tifier corresponding to PACKET_ID, taining r corresponding to TOPIC_FILTER_MULTI_LEVEL_INVALID, I_qos corresponding to AT_MOST_ONCE; es the TCP_CONNECTION	
Final Conditions		

TP ld	TP_MQTT_BROKER_SUBSCRIBE_010	
Test Objective	Verify that the IUT validates topic_filter field to be a valid single-level Topic Filter.	
Reference	[MQTT-4.7.1-3], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
Initial Conditions		
with {		
the CLIENT having a MQTT_CONNECTION to the IUT		

```
Expected Behaviour

ensure that {
    when {
        the IUT receives a SUBSCRIBE message containing
        header_flags indicating value '0010'B,
        packet_identifier corresponding to PACKET_ID,
        payload containing
        topic_filter corresponding to TOPIC_FILTER_SINGLE_LEVEL_INVALID,
        requested_qos corresponding to AT_MOST_ONCE;
    ;
}
then {
        the IUT closes the TCP_CONNECTION
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_SUBSCRIBE_011		
Test Objective	Verify that the IUT allows topic_filter field to include the 'zero width no-break space character'		
Reference	[MQTT-1.5.3-3]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with { the CLIENT h }	with { the CLIENT having a MQTT_CONNECTION to the IUT		
,	Expected Behaviour		
packet_identi payload conti topic_filter requested_ ; } then { the IUT send	ves a SUBSCRIBE message containing ifier corresponding to PACKET_ID, aining corresponding to TOPIC_FILTER_WITH_ZWNBS, _qos corresponding to AT_MOST_ONCE; s a SUBACK message containing ntifier corresponding to PACKET_ID;		
Final Conditions			

TP Id	TP_MQTT_BROKER_SUBACK_001		
Test Objective	Verify that the IUT replies with a SUBACK Control Packet with valid header flags.		
Reference	[MQTT-2.2.2-1], [MQTT-3.8.1-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT I	having a MQTT_CONNECTION to the IUT		
}			
	Expected Behaviour		
ensure that {			
when {			
the IUT recei	the IUT receives a SUBSCRIBE message containing		
header_flags indicating value '0010'B;			
}			
then {			
the IUT sends a SUBACK message containing			
header_flags	header_flags indicating value '0000'B;		
}			
}			

Final Conditions

```
TP Id
                    TP_MQTT_BROKER_SUBACK_002
Test Objective
                    Verify that the IUT replies with a SUBACK Control Packet containing a packet identifier
                    corresponding to the SUBSCRIBE Control Packet.
                   [MQTT-2.3.1-1], [MQTT-2.3.1-7], [MQTT-3.8.4-1], [MQTT-3.8.4-2]
PICS_BROKER_QOS_1
Reference
PICS Selection
                                               Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                             Expected Behaviour
ensure that {
  when {
       the IUT receives a SUBSCRIBE message containing
       header_flags indicating value '0010'B,
       packet_identifier corresponding to PACKET_ID,
       payload containing
         topic_filter corresponding to PX_SUBSCRIBE_TOPIC_FILTER,
         requested_qos corresponding to AT_LEAST_ONCE;
  then {
       the IUT sends a SUBACK message containing
       header_flags indicating value '0000'B,
       packet_identifier corresponding to PACKET_ID;
  }
                                               Final Conditions
```

TP Id	TP_MQTT_BROKER_SUBACK_003		
Test Objective	Verify that the IUT replies with a SUBACK Control Packet with a valid maximum QoS level.		
Reference	[MQTT-3.9.3-1], [MQTT-3.9.3-2]		
PICS Selection	PICS_BROKER_QOS_2		
	Initial Conditions		
with {			
the CLIENT	having a MQTT_CONNECTION to the IUT		
}			
	Expected Behaviour		
ensure that { when {			
	ives a SUBSCRIBE message containing s indicating value '0010'B,		
packet_ident	packet_identifier corresponding to PACKET_ID,		
payload conf	· ·		
	topic_filter corresponding to PX_SUBSCRIBE_TOPIC_FILTER, requested_qos corresponding to AT_MOST_ONCE;		
;			
than (
then { the IUT sends a SUBACK message containing header_flags indicating value '0000'B, packet_identifier corresponding to PACKET_ID, return_code indicating value 0x00;			
}			
J	Final Conditions		
	I mai denamento		

```
TP Id
                   TP_MQTT_BROKER_SUBACK_004
                   Verify that the IUT replies with a SUBACK Control Packet with a valid maximum QoS level.
Test Objective
Reference
                   [MQTT-3.9.3-1], [MQTT-3.9.3-2]
PICS Selection
                   PICS_BROKER_BASIC and PICS_BROKER_QOS_1
                                              Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
       the IUT receives a SUBSCRIBE message containing
       header_flags indicating value '0010'B,
       packet_identifier corresponding to PACKET_ID,
       payload containing
         topic_filter corresponding to PX_SUBSCRIBE_TOPIC_FILTER,
         requested_qos corresponding to AT_LEAST_ONCE;
  then {
       the IUT sends a SUBACK message containing
       header_flags indicating value '0000'B,
       packet_identifier corresponding to PACKET_ID,
       return_code indicating value 0x01;
       the IUT sends a SUBACK message containing // Note: if the IUT supports only QoS 0
       header_flags indicating value '0000'B,
       packet_identifier corresponding to PACKET_ID,
       return_code indicating value 0x00;
  }
                                              Final Conditions
```

TP Id	TD MOTT PROVED SUBACK OOS		
	TP_MQTT_BROKER_SUBACK_005		
Test Objective Reference	Verify that the IUT replies with a SUBACK Control Packet with a valid maximum QoS level.		
	[MQTT-3.9.3-1], [MQTT-3.9.3-2]		
PICS Selection	PICS Selection PICS_BROKER_BASIC and PICS_BROKER_QOS_1 and PICS_BROKER_QOS_2		
201 (Initial Conditions		
with { the CLIENT h }	naving a MQTT_CONNECTION to the IUT		
	Expected Behaviour		
ensure that {			
when {			
	ves a SUBSCRIBE message containing		
	indicating value '0010'B,		
	ifier corresponding to PACKET_ID,		
payload conta			
	corresponding to PX_SUBSCRIBE_TOPIC_FILTER,		
requested_	_qos corresponding to EXACTLY_ONCE;		
;			
} *h=== (
then {	a CURACK recessor containing		
	s a SUBACK message containing		
	header_flags indicating value '0000'B,		
	packet_identifier corresponding to PACKET_ID, return_code indicating value 0x02;		
or			
the IUT sends a SUBACK message containing // Note: if the IUT supports only up to QoS 1 header_flags indicating value '0000'B,			
packet_identifier corresponding to PACKET_ID,			
return_code indicating value 0x01;			
or	management of the second of th		
_	s a SUBACK message containing // Note: if the IUT supports only QoS 0		
	indicating value '0000'B,		
packet_identifier corresponding to PACKET_ID,			

```
return_code indicating value 0x00;
}

Final Conditions
```

TP ld	TP_MQTT_BROKER_SUBACK_006	
Test Objective	Verify that the IUT replies to a failed subscription with a SUBACK Control Packet with the return	
•	code 0x80.	
Reference	[MQTT-3.9.3-1], [MQTT-3.9.3-2]	
PICS Selection	PICS_BROKER_BASIC or PICS_BROKER_QOS_2	
	Initial Conditions	
with {		
the CLIEN	Γ having a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
	eives a SUBSCRIBE message containing	
	gs indicating value '0010'B,	
	ntifier corresponding to PACKET_ID,	
payload co		
	er corresponding to TOPIC_FILTER_INVALID,	
requeste	requested_qos corresponding to AT_MOST_ONCE;	
;		
}		
then {		
	nds a SUBACK message containing	
	gs indicating value '0000'B,	
packet_ide	ntifier corresponding to PACKET_ID,	
return_code indicating value 0x80;		
}		
}		
	Final Conditions	

TP Id	TP MQTT BROKER UNSUBSCRIBE 001	
Test Objective	Verify that 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], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT	having a MQTT_CONNECTION to the IUT	
}	-	
	Expected Behaviour	
ensure that {		
when {		
the IUT rece	ives a UNSUBSCRIBE message containing	
header_flags indicating value '1101'B;		
}		
then {		
the IUT closes the TCP_CONNECTION		
}		
}		
	Final Conditions	

```
TP Id
                   TP_MQTT_BROKER_UNSUBSCRIBE_002
                   Verify that the IUT validates a UNSUBSCRIBE Control Packet to contain a non-zero 16-bit Packet
Test Objective
                   Identifier.
                   [MQTT-2.3.1-1], [MQTT-4.8.0-1]
Reference
PICS Selection
                   PICS_BROKER_BASIC
                                             Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a UNSUBSCRIBE message containing
      header_flags indicating value '0010'B,
      packet_identifier corresponding to PACKET_ID_ZERO;
  then {
      the IUT closes the TCP_CONNECTION
      the IUT sends a UNSUBACK message containing
      packet_identifier corresponding to PACKET_ID_ZERO;
  }
                                             Final Conditions
```

TP Id	TP_MQTT_BROKER_UNSUBSCRIBE_003
Test Objective	Verify that the IUT validates the topic_filter in a UNSUBSCRIBE Control Packet to be a well-
-	formed UTF-8 encoded string and do not contain code points between U+D800 and U+DFFF.
Reference	[MQTT-1.5.3-1], [MQTT-3.10.3-1], [MQTT-4.8.0-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
with { the CLIENT }	having a MQTT_CONNECTION to the IUT
	Expected Behaviour
header_flag packet_iden payload con topic_filte ; } then {	r corresponding to TOPIC_FILTER_D800; es the TCP_CONNECTION
	Final Conditions

TP Id	TP_MQTT_BROKER_UNSUBSCRIBE_004		
Test Objective	Verify that the IUT validates all topic filters to be at least one character long.		
Reference	[MQTT-4.7.3-1], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT h	the CLIENT having a MQTT_CONNECTION to the IUT		
}			
Expected Behaviour			
ensure that {			
when {			
the IUT receives a UNSUBSCRIBE message containing			
header_flags indicating value '0010'B,			
	-		

```
packet_identifier corresponding to PACKET_ID,
payload containing
topic_filter corresponding to TOPIC_FILTER_ZERO_CHARS;
;
}
then {
the IUT closes the TCP_CONNECTION
}
}

Final Conditions
```

TP Id	TP MQTT BROKER UNSUBSCRIBE 005
Test Objective	Verify that the IUT validates the topic filter in a UNSUBSCRIBE Control Packet not to contain the
•	null character (Unicode U+0000).
Reference	[MQTT-4.7.3-2], [MQTT-4.8.0-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
with { the CLIENT	having a MQTT_CONNECTION to the IUT
	Expected Behaviour
header_flags packet_ident payload cont topic_filter ; } then {	ives a UNSUBSCRIBE message containing indicating value '0010'B, ifier corresponding to PACKET_ID, aining corresponding to TOPIC_FILTER_0000; es the TCP_CONNECTION
	Final Conditions

TP Id	TP_MQTT_BROKER_UNSUBSCRIBE_006		
Test Objective	Verify that the IUT validates a UNSUBSCRIBE Control Packet to contain at least on topic filter.		
Reference	[MQTT-3.10.3-2], [MQTT-4.8.0-1], [MQTT-4.8.0-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with { the CLIENT h	with { the CLIENT having a MQTT_CONNECTION to the IUT		
,	Expected Behaviour		
header_flags packet_identi payload conta omit; ; } then {	ensure that { when { the IUT receives a UNSUBSCRIBE message containing header_flags indicating value '0010'B, packet_identifier corresponding to PACKET_ID, payload containing omit; ; }		
Final Conditions			

```
TP_MQTT_BROKER_UNSUBACK_001
TP Id
Test Objective
                   Verify that the IUT replies with an UNSUBACK Control Packet with valid header flags.
Reference
                   [MQTT-2.2.2-1], [MQTT-3.10.1-1]
PICS Selection
                   PICS_BROKER_BASIC
                                             Initial Conditions
with {
      the CLIENT having a MQTT_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a UNSUBSCRIBE message containing
      header_flags indicating value '0010'B;
  then {
      the IUT sends a UNSUBACK message containing
      header_flags indicating value '0000'B;
  }
                                              Final Conditions
```

TP Id	TP_MQTT_BROKER_UNSUBACK_002	
Test Objective	Verify that the IUT replies with a UNSUBACK Control Packet containing a packet identifier	
	corresponding to the UNSUBSCRIBE Control Packet.	
Reference	[MQTT-3.10.4-4], [MQTT-2.3.1-7]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT	having a MQTT_CONNECTION to the IUT	
and		
	bscribed the PX_PUBLISH_TOPIC containing	
qos_level c	corresponding to AT_MOST_ONCE;	
}		
	Expected Behaviour	
ensure that {		
when {		
	eives a UNSUBSCRIBE message containing	
	s indicating value '0010'B,	
	tifier corresponding to PACKET_ID,	
payload con		
topic_filter	topic_filter corresponding to PX_PUBLISH_TOPIC;	
;		
then {		
the IUT send	ds a UNSUBACK message containing	
header_flags	s indicating value '0000'B,	
packet_iden	tifier corresponding to PACKET_ID;	
}		
}		
	Final Conditions	

TP Id	TP MQTT BROKER UNSUBACK 003		
Test Objective	Verify that the IUT replies with an UNSUBACK Control Packet even if no topic subscriptions are		
	deleted.		
Reference	[MQTT-3.10.4-5]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {	with {		
the CLIENT	the CLIENT having a MQTT_CONNECTION to the IUT		
and	and		
the CLIENT not subscribed the PX_PUBLISH_TOPIC			
}			
Expected Behaviour			

```
ensure that {
    when {
        the IUT receives a UNSUBSCRIBE message containing
        header_flags indicating value '0010'B,
        packet_identifier corresponding to PACKET_ID,
        payload containing
        topic_filter corresponding to PX_PUBLISH_TOPIC;
    ;
}
then {
        the IUT sends a UNSUBACK message containing
        header_flags indicating value '0000'B;
}

Final Conditions
```

TP ld	TP_MQTT_BROKER_UNSUBACK_004	
Test Objective	Verify that the IUT replies to UNSUBSCRIBE Control Packets with multiple topic filters with one	
_	single UNSUBACK Control Packet.	
Reference	[MQTT-3.10.4-6]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT h	naving a MQTT_CONNECTION to the IUT	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT receiv	ves a UNSUBSCRIBE message containing	
header_flags	header_flags indicating value '0010'B,	
packet_identi	packet_identifier corresponding to PACKET_ID,	
payload conta	aining	
	orresponding to PX_PUBLISH_TOPIC,	
topic_filter corresponding to TOPIC_FILTER_VALID; // second topic filter		
;		
}		
then {		
the IUT sends	the IUT sends a UNSUBACK message	
and		
the IUT sends no second UNSUBACK message		
}		
}		
Final Conditions		

TP ld	TP_MQTT_BROKER_PINGREQ_001	
Test Objective	Verify that 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], [MQTT-4.8.0-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT I	the CLIENT having a MQTT_CONNECTION to the IUT	
}	 }	
Expected Behaviour		
ensure that {		
when {		
the IUT recei	ives a PINGREQ message containing	
header_flags	header flags indicating value '1111'B;	
}		
then {		
the IUT closes the TCP_CONNECTION		
}	_	
} ′		

Final Conditions

```
TP Id
                    TP_MQTT_BROKER_PINGRESP_001
                    Verify that the IUT replies with a PINGRESP Control Packet with valid header flags. [MQTT-2.2.2-1], [MQTT-3.12.4-1]
Test Objective
Reference
PICS Selection
                    PICS_BROKER_BASIC
                                                Initial Conditions
with {
       the CLIENT having a MQTT_CONNECTION to the IUT
                                               Expected Behaviour
ensure that {
  when {
       the IUT receives a PINGREQ message containing
       header_flags indicating value '0000'B;
  then {
       the IUT sends a PINGRESP message containing
       header_flags indicating value '0000'B;
  }
                                                 Final Conditions
```

TP ld	TP_MQTT_BROKER_DISCONNECT_001	
Test Objective	Verify that the IUT closes the network connection if fixed header flags in DISCONNECT Control	
	Packet are valid.	
Reference	[MQTT-2.2.2-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT	the CLIENT having a MQTT_CONNECTION to the IUT	
}	•	
	Expected Behaviour	
ensure that {		
when {		
	eives a DISCONNECT message containing	
header_flag	header_flags indicating value '0000'B;	
}		
•	then {	
the IUT closes the TCP_CONNECTION		
}		
}		
Final Conditions		

TP Id	TP_MQTT_BROKER_DISCONNECT_002
Test Objective	Verify that the IUT closes the network connection if fixed header flags in DISCONNECT Control
	Packet are valid.
Reference	[MQTT-2.2.2-2], [MQTT-3.14.1-1], [MQTT-4.8.0-1]
PICS Selection	PICS_BROKER_BASIC
	Initial Conditions
with {	
	having a MQTT CONNECTION to the IUT
}	
	Expected Behaviour
ensure that {	
when {	
the IUT rec	eives a DISCONNECT message containing
header flags indicating value '1111'B;	
}	,
then {	
	ses the TCP_CONNECTION

```
}
Final Conditions
```

```
TP_MQTT_BROKER_FEAT_REMLEN_001
TP Id
                   Verify that the IUT forwards PUBLISH Control Packets with Remaining Length fields encoded in
Test Objective
                   one byte.
Reference
                   MQTT 2.2.3
PICS Selection
                   PICS_BROKER_BASIC
                                             Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
      and
      the CLIENT_2 having a MQTT_CONNECTION to the IUT
      the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
        qos_level corresponding to AT_MOST_ONCE;
      to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         qos_level corresponding to AT_MOST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier indicating value omit,
      payload corresponding to PAYLOAD;
      from the CLIENT_1
  then {
      the IUT sends a PUBLISH message containing
      publish_header containing
        dup_flag indicating value '0'B,
         qos_level corresponding to AT_MOST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier indicating value omit,
      payload corresponding to PAYLOAD;
      to the CLIENT_2
  }
                                             Final Conditions
```

TP Id	TP MQTT BROKER FEAT REMLEN 002	
Test Objective	Verify that the IUT forwards PUBLISH Control Packets with Remaining Length fields encoded in	
	two bytes.	
Reference	MQTT 2.2.3	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT_	_1 having a MQTT_CONNECTION to the IUT	
and	and	
the CLIENT_	2 having a MQTT_CONNECTION to the IUT	
and	and	
_	2 subscribed the PX_PUBLISH_TOPIC containing	
· –	qos_level corresponding to AT_MOST_ONCE;	
to the IUT		
}		
	Expected Behaviour	
ensure that {		

```
when {
    the IUT receives a PUBLISH message containing
    publish_header containing
       dup_flag indicating value '0'B,
       gos level corresponding to AT MOST ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier indicating value omit,
    payload corresponding to PAYLOAD_REM_LEN_2;
    from the CLIENT_1
then {
    the IUT sends a PUBLISH message containing
    publish_header containing
       dup_flag indicating value '0'B,
       qos_level corresponding to AT_MOST_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier indicating value omit,
    payload corresponding to PAYLOAD_REM_LEN_2;
    to the CLIENT 2
}
                                            Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_REMLEN_003
Test Objective
                   Verify that the IUT forwards PUBLISH Control Packets with Remaining Length fields encoded in
                   three bytes.
Reference
                   MQTT 2.2.3
PICS Selection
                   PICS BROKER BASIC
                                             Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
      the CLIENT_2 having a MQTT_CONNECTION to the IUT
      the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
        qos_level corresponding to AT_MOST_ONCE;
      to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         qos_level corresponding to AT_MOST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier indicating value omit,
      payload corresponding to PAYLOAD_REM_LEN_3;
      from the CLIENT_1
  then {
      the IUT sends a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         qos_level corresponding to AT_MOST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier indicating value omit,
      payload corresponding to PAYLOAD_REM_LEN_3;
      to the CLIENT_2
  }
```

```
Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_REMLEN_004
Test Objective
                   Verify that the IUT forwards PUBLISH Control Packets with Remaining Length fields encoded in
                   four bytes.
Reference
                   MQTT 2.2.3
PICS Selection
                   PICS_BROKER_BASIC
                                             Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
      the CLIENT_2 having a MQTT_CONNECTION to the IUT
      and
      the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
        qos_level corresponding to AT_MOST_ONCE;
      to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         qos_level corresponding to AT_MOST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier indicating value omit,
      payload corresponding to PAYLOAD_REM_LEN_4;
      from the CLIENT_1
  then {
      the IUT sends a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         qos_level corresponding to AT_MOST_ONCE,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier indicating value omit,
      payload corresponding to PAYLOAD_REM_LEN_4;
      to the CLIENT_2
  }
                                              Final Conditions
```

TP_MQTT_BROKER_FEAT_KEEPALIVE_001	
Verify that the IUT disconnects a client if it does not receive a Control Packet from it within one	
and a half times of the given Keep Alive time period.	
[MQTT-3.1.2-24]	
PICS_CLIENT_BASIC	
Initial Conditions	
having a MQTT_CONNECTION to the IUT	
the CLIENT established the MQTT_CONNECTION containing	
keep_alive corresponding to PX_KEEP_ALIVE;	
Expected Behaviour	
ensure that { when {	
the IUT times_out	
}	
then {	
the IUT closes the TCP_CONNECTION to the CLIENT	

```
}
Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_RTND_001
Test Objective
                   Verify that the IUT does not delete Retained Messages when a session with the corresponding
                   client ends.
Reference
                   [MQTT-3.1.2-7]
PICS Selection
                   PICS_BROKER_RTND
                                            Initial Conditions
with {
      the IUT having a UTF8_MESSAGE_VALID in the RETAIN_TOPIC
    and
    the CLIENT having a MQTT_CONNECTION to the IUT
    the CLIENT having a CLEAN_SESSION
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a SUBSCRIBE message containing
      packet_identifier corresponding to PACKET_ID,
      payload containing
         topic_filter corresponding to PX_SUBSCRIBE_TOPIC_FILTER,
         requested_qos corresponding to AT_MOST_ONCE;
  then {
      the IUT sends a SUBACK message containing
      return_code indicating value 0x00;
      and the IUT sends a PUBLISH message containing
      topic_name corresponding to PX_SUBSCRIBE_TOPIC_FILTER,
      payload corresponding to UTF8_MESSAGE_VALID;
  }
                                             Final Conditions
```

TP Id	TP_MQTT_BROKER_FEAT_RTND_002	
Test Objective	Verify that the IUT stores Retained Messages for future deliveries.	
Reference	[MQTT-3.3.1-5], [MQTT-3.3.1-6], [MQTT-3.3.1-8]	
PICS Selection	PICS_BROKER_RTND	
	Initial Conditions	
with {		
the CLIENT_	1 published a Message containing	
dup_flag in	dicating value '0'B,	
qos_level c	corresponding to AT_LEAST_ONCE,	
	retain_flag indicating value '1'B,	
	e corresponding to PX_PUBLISH_TOPIC,	
	packet_identifier corresponding to PACKET_ID_1,	
	payload corresponding to PAYLOAD;	
to the IUT		
and	and	
the CLIENT_2 having a MQTT_CONNECTION to the IUT		
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT receives a SUBSCRIBE message containing		
packet_identifier corresponding to PACKET_ID_2,		
payload containing		
topic_filter corresponding to PX_PUBLISH_TOPIC,		
requested_qos corresponding to AT_MOST_ONCE;		
;		

```
from the CLIENT_2
then {
    the IUT sends a SUBACK message containing
    packet_identifier corresponding to PACKET_ID_2,
    payload containing
       return_code indicating value 0x00;
    to the CLIENT_2
    and
    the IUT sends a PUBLISH message containing
    publish_header containing
       dup_flag indicating value '0'B,
       qos_level corresponding to AT_MOST_ONCE,
       retain_flag indicating value '1'B,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_3,
    payload corresponding to PAYLOAD;
    to the CLIENT_2
}
                                             Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_RTND_003
                   Verify that the IUT sets the retain_flag to 0 when Retained Messages are delivered directly to
Test Objective
                   existing subscriptions.
Reference
                   [MQTT-3.3.1-9]
PICS Selection
                   PICS BROKER RTND
                                              Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_2 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
       qos_level corresponding to AT_MOST_ONCE;
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         gos_level corresponding to AT_LEAST_ONCE,
         retain_flag indicating value '1'B,
      topic_name corresponding to PX_PUBLISH_TOPIC,
       packet_identifier corresponding to PACKET_ID_1,
      payload corresponding to PAYLOAD;
      from the CLIENT_1
  then {
      the IUT sends a PUBACK message containing
      packet_identifier corresponding to PACKET_ID_1;
      to the CLIENT_1
      the IUT sends a PUBLISH message containing
      publish_header containing
         dup_flag indicating value '0'B,
         gos_level corresponding to AT_MOST_ONCE,
         retain_flag indicating value '0'B,
      topic_name corresponding to PX_PUBLISH_TOPIC,
      packet_identifier corresponding to PACKET_ID_2,
       payload corresponding to PAYLOAD;
```

```
to the CLIENT_2
}

Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_RTND_004
Test Objective
                   Verify that the IUT forwards Retained Messages with a zero-bytes payload.
Reference
                   [MQTT-3.3.1-10]
PICS Selection
                   PICS_BROKER_RTND
                                             Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
   the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
      qos_level corresponding to AT_MOST_ONCE;
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    publish_header containing
     qos_level corresponding to AT_MOST_ONCE,
     retain_flag indicating value '1'B,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    payload corresponding to PAYLOAD_ZERO_BYTE;
    from the CLIENT_1
  then {
      the IUT sends a PUBLISH message containing
    publish_header containing
     qos_level corresponding to AT_MOST_ONCE,
     retain_flag indicating value '0'B,
    payload corresponding to PAYLOAD_ZERO_BYTE;
    to the CLIENT_2
  }
                                             Final Conditions
```

TP_MQTT_BROKER_FEAT_RTND_005	
Verify that the IUT deletes a stored Retained Messages if it receives a new Retained Message	
with a zero-bytes payload.	
[MQTT-3.3.1-10], [MQTT-3.3.1-11]	
PICS_BROKER_RTND	
Initial Conditions	
with { the CLIENT_1 having a MQTT_CONNECTION to the IUT	
aving a MQTT_CONNECTION to the IUT	
Expected Behaviour	
ives a PUBLISH message containing	
containing	
gos_level corresponding to AT_MOST_ONCE,	
retain_flag indicating value '1'B,	
topic_name corresponding to PX_PUBLISH_TOPIC,	
payload corresponding to PAYLOAD;	
-	
i	

```
from the CLIENT_1
  the IUT receives a PUBLISH message containing
  publish_header containing
   qos_level corresponding to AT_MOST_ONCE,
   retain_flag indicating value '1'B,
  topic_name corresponding to PX_PUBLISH_TOPIC,
  payload corresponding to PAYLOAD_ZERO_BYTE;
  from the CLIENT_1
  and
  the IUT receives a SUBSCRIBE message containing
  payload containing
   topic_filter corresponding to PX_PUBLISH_TOPIC,
   requested_gos corresponding to AT_MOST_ONCE;
then {
    the IUT sends no PUBLISH message to the CLIENT_2
}
                                           Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_RTND_006
Test Objective
                   Verify that the IUT does neither store a Retained Message nor removes or replaces any existing
                   Retained Messages if the retained_flag is set to 0.
Reference
                   [MQTT-3.3.1-12]
PICS Selection
                   PICS_BROKER_RTND
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    publish_header containing
     qos_level corresponding to AT_MOST_ONCE,
     retain_flag indicating value '1'B,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    payload corresponding to PAYLOAD;
    from the CLIENT_1
    and
    the IUT receives a PUBLISH message containing
    publish_header containing
     qos_level corresponding to AT_MOST_ONCE,
     retain flag indicating value '0'B.
    topic_name corresponding to PX_PUBLISH_TOPIC,
    payload corresponding to PAYLOAD_2;
    from the CLIENT_1
    and
    the IUT receives a SUBSCRIBE message containing
    payload containing
     topic_filter corresponding to PX_PUBLISH_TOPIC,
     requested_qos corresponding to AT_MOST_ONCE;
    from the CLIENT_2
  then {
      the IUT sends a PUBLISH message containing
    publish_header containing
     qos_level corresponding to AT_MOST_ONCE
```

```
retain_flag indicating value '1'B,
    payload corresponding to PAYLOAD;
    ;
    to the CLIENT_2
    }
}
Final Conditions
```

```
TP Id
                   TP MQTT BROKER FEAT RTND 007
Test Objective
                   Verify that the IUT stores Retained Messages with a QoS level of 0 for future deliveries. However,
                   the IUT may choose to discard Retained Messages with a QoS level of 0 at any time.
Reference
                   [MQTT-3.3.1-7]
PICS Selection
                   PICS BROKER RTND
                                              Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
       the IUT receives a PUBLISH message containing
    publish_header containing
      qos_level corresponding to AT_LEAST_ONCE,
      retain_flag indicating value '1'B,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    payload corresponding to PAYLOAD;
    from the CLIENT_1
    and
    the IUT receives a PUBLISH message containing
    publish_header containing
      qos_level corresponding to AT_MOST_ONCE,
      retain_flag indicating value '1'B,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    payload corresponding to PAYLOAD_2;
    from the CLIENT_1
    and
    the IUT receives a SUBSCRIBE message containing
    payload containing
     topic_filter corresponding to PX_PUBLISH_TOPIC,
      requested_qos corresponding to AT_MOST_ONCE;
    from the CLIENT_2
  then {
       the IUT sends a PUBLISH message containing
    publish header containing
      gos_level corresponding to AT_MOST_ONCE,
      retain_flag indicating value '1'B,
    payload corresponding to PAYLOAD_2;
    to the CLIENT_2
    the IUT sends no PUBLISH message to the CLIENT_2
  }
                                              Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_LWT_001
Test Objective
                   Verify that the IUT sends a Will Messages to subscribes if a client with LWT disconnects
                   unexpectedly.
Reference
                   [MQTT-3.1.2-8]
PICS Selection
                   PICS_BROKER_LWT
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_1 established the MQTT_CONNECTION containing
   flags containing
    will_flag indicating value '1'B,
   payload containing
    will_topic corresponding to PX_PUBLISH_TOPIC,
    will_message corresponding to PX_WILL_MESSAGE;
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_2 subscribed the PX_PUBLISH_TOPIC
                                           Expected Behaviour
ensure that {
  when {
      the CLIENT_1 closes the TCP_CONNECTION to the IUT
  then {
      the IUT sends a PUBLISH message containing
    topic_name corresponding to PX_PUBLISH_TOPIC,
    payload corresponding to PX_WILL_MESSAGE;
    to the CLIENT_2
  }
                                             Final Conditions
```

TP Id	TP_MQTT_BROKER_FEAT_LWT_002	
Test Objective	Verify that the IUT deletes a Will Messages if the client with LWT disconnects correctly with a	
	DISCONNECT Control Packet.	
Reference	[MQTT-3.1.2-8], [MQTT-3.1.2-10], [MQTT-3.14.4-3]	
PICS Selection	PICS_BROKER_LWT	
	Initial Conditions	
with {		
the CLIENT	「⊆1 having a MQTT_CONNECTION to the IUT	
and		
the CLIENT_1	established the MQTT_CONNECTION containing	
flags containing		
will_flag indicating value '1'B,		
	payload containing	
will_topic corr	will_topic corresponding to PX_PUBLISH_TOPIC,	
will_message	corresponding to PX_WILL_MESSAGE;	
,,		
and		
_	having a MQTT_CONNECTION to the IUT	
and		
the CLIENT_2	subscribed the PX_PUBLISH_TOPIC	
}		
	Expected Behaviour	
ensure that {		
when {	· PIOCONNECT / / OUTSIT /	
the IUT rec	eives a DISCONNECT message from the CLIENT_1	
}		
then {	ada na DUDUJOU nasasana ta tha OUTANT O	
the IUT ser	nds no PUBLISH message to the CLIENT_2	
}		
,}		

Final Conditions

```
TP Id
                   TP_MQTT_BROKER_FEAT_LWT_003
Test Objective
                   Verify that the IUT sends no Will Message if a client without LWT disconnects unexpectedly.
Reference
                   [MQTT-3.1.2-12]
PICS Selection
                   PICS_BROKER_LWT
                                             Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_1 established the MQTT_CONNECTION containing
   flags containing
    will_flag indicating value '0'B;
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
   the CLIENT_2 subscribed the PX_PUBLISH_TOPIC
                                           Expected Behaviour
ensure that {
  when {
      the CLIENT_1 closes the TCP_CONNECTION to the IUT
  then {
      the IUT sends no PUBLISH message to the CLIENT_2
                                             Final Conditions
```

```
TP Id
                   TP MQTT BROKER FEAT LWT 004
Test Objective
                   Verify that the IUT handles Will Messages with will retain set to 1 as Retained Will Messages.
Reference
                   [MQTT-3.1.2-17]
PICS Selection
                   PICS BROKER LWT
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
   the CLIENT_1 established the MQTT_CONNECTION containing
   flags containing
    will_retain indicating value '1'B,
    will_flag indicating value '1'B,
   payload containing
    will_topic corresponding to PX_PUBLISH_TOPIC
    will_message corresponding to PX_WILL_MESSAGE;
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
   the CLIENT_1 closed the TCP_CONNECTION to the IUT
                                            Expected Behaviour
ensure that {
  when {
      the IUT receives a SUBSCRIBE message containing
    payload containing
      topic_filter corresponding to PX_PUBLISH_TOPIC;
    from the CLIENT_2
  then {
      the IUT sends a PUBLISH message containing
    topic_name corresponding to PX_PUBLISH_TOPIC,
```

```
payload corresponding to PX_WILL_MESSAGE;
to the CLIENT_2
}

Final Conditions
```

	P_MQTT_BROKER_FEAT_LWT_005	
	erify that the IUT handles Will Messages with will_retain set to 1 as non-retained Will Messages.	
Reference [Mo	QTT-3.1.2-16]	
PICS Selection PIC	CS_BROKER_LWT	
	Initial Conditions	
with {		
the CLIENT_1 ha	aving a MQTT_CONNECTION to the IUT	
and		
the CLIENT_1 estab	blished the MQTT_CONNECTION containing	
flags containing	-	
will_retain indicating	ng value '0'B,	
will_flag indicating	value '1'B,	
payload containing	·	
will_topic corresponding to PX_PUBLISH_TOPIC,		
	will_message corresponding to PX_WILL_MESSAGE;	
- :		
and		
the CLIENT_2 having a MQTT_CONNECTION to the IUT		
and		
the CLIENT_1 close	ed the TCP_CONNECTION to the IUT	
-		
	Expected Behaviour	
ensure that {		
when {		
the IUT receives	s a SUBSCRIBE message containing	
payload containing		
topic_filter corresponding to PX_PUBLISH_TOPIC;		
;		
from the CLIENT_2		
}		
then {		
the IUT sends no PUBLISH message to the CLIENT_2		
}		
}		
	Final Conditions	

TP_MQTT_BROKER_FEAT_PUBSUB_001	
Verify that the IUT validates the UTF-8 encoded sequence 0xEF 0xBB 0xBF as Unicode U+FEFF	
('ZERO WIDTH NO-BREAK SPACE') within the topic name of a PUBLISH Control Packet.	
[MQTT-1.5.3-3], [MQTT-4.7.3-4]	
PICS_BROKER_BASIC	
Initial Conditions	
_1 having a MQTT_CONNECTION to the IUT	
the CLIENT_2 having a MQTT_CONNECTION to the IUT	
and	
the CLIENT_2 subscribed the PX_PUBLISH_TOPIC	
Expected Behaviour	
the IUT receives a PUBLISH message containing	
publish_header containing	
qos_level corresponding to AT_MOST_ONCE,	
topic_name corresponding to TOPIC_FILTER_WITH_ZWNBS;;	
Γ_1	

```
}
then {
the IUT sends no PUBLISH to the CLIENT_2
}

Final Conditions
```

TP Id	TP_MQTT_BROKER_FEAT_PUBSUB_002		
Test Objective	Verify that the IUT validates the UTF-8 encoded sequence 0xEF 0xBB 0xBF as Unicode U+FEFF		
•	('ZERO WIDTH NO-BREAK SPACE') within the topic filter of a SUBSCRIBE Control Packet.		
Reference	[MQTT-1.5.3-3], [MQTT-4.7.3-4]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT_	1 having a MQTT_CONNECTION to the IUT		
and			
the CLIENT_2 ha	the CLIENT_2 having a MQTT_CONNECTION to the IUT		
and	and		
the CLIENT_2 su	ubscribed the TOPIC_FILTER_WITH_ZWNBS		
}	}		
Expected Behaviour			
ensure that {			
when {			
	ves a SUBSCRIBE message containing		
. ,	payload containing		
	topic_filter corresponding to PX_PUBLISH_TOPIC,		
requested_qos	s corresponding to AT_MOST_ONCE;		
;	.,		
from the CLIENT	_1		
}			
then {			
the IUT send	s no PUBLISH to the CLIENT_2		
)			
}	Final Conditions		
Final Conditions			

TP Id	TP_MQTT_BROKER_FEAT_PUBSUB_003	
Test Objective	Verify that the IUT does not match topic filters starting with a multi-level wildcard character (#)	
	with topic names beginning with a \$ character	
Reference	[MQTT-4.7.2-1]	
PICS Selection	PICS_BROKER_BASIC	
	Initial Conditions	
with {		
the CLIENT_	the CLIENT_1 having a MQTT_CONNECTION to the IUT	
and	and	
	aving a MQTT_CONNECTION to the IUT	
and		
the CLIENT_2 su	ubscribed the TOPIC_FILTER_MULTI_LEVEL_ALL	
}		
	Expected Behaviour	
ensure that {		
when {	DIDLICH management and significant	
	ves a PUBLISH message containing	
	responding to TOPIC_NAME_SYS;	
1 ITOMI THE CLIEN	from the CLIENT_1	
then {		
,	·	
\ \	the IUT sends no PUBLISH message to the CLIENT_2	
}		
Final Conditions		

```
TP Id
                   TP_MQTT_BROKER_FEAT_PUBSUB_004
Test Objective
                   Verify that the IUT does not match topic filters starting with a single-level wildcard character (+)
                   with topic names beginning with a $ character
Reference
                   [MQTT-4.7.2-1]
                   PICS_BROKER_BASIC
PICS Selection
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_2 subscribed the TOPIC_FILTER_SINGLE_LEVEL
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    topic_name corresponding to TOPIC_NAME_SYS;
    from the CLIENT_1
  then {
      the IUT sends no PUBLISH message to the CLIENT_2
  }
                                             Final Conditions
```

TP Id	TP_MQTT_BROKER_FEAT_PUBSUB_005		
Test Objective	Verify that the IUT does match topic names and filters beginning with a \$ character.		
Reference	[MQTT-4.7.2-1]		
PICS Selection	PICS_BROKER_BASIC		
	Initial Conditions		
with {			
the CLIENT_	1 having a MQTT_CONNECTION to the IUT		
and	and		
the CLIENT_2 ha	the CLIENT_2 having a MQTT_CONNECTION to the IUT		
and	and		
the CLIENT_2 su	ubscribed the TOPIC_FILTER_MULTI_LEVEL_SYS_ALL		
}			
Expected Behaviour			
ensure that {			
when {			
	ves a PUBLISH message containing		
	responding to TOPIC_NAME_SYS;		
from the CLIEN	from the CLIENT_1		
}			
then {			
	the IUT sends a PUBLISH message containing		
	topic_name corresponding to TOPIC_NAME_SYS;		
to the CLIENT_	to the CLIENT_2		
}			
<u> </u>			
Final Conditions			

TP Id	TP MQTT BROKER FEAT PUBSUB 006
Test Objective	Verify that the IUT resends PUBLISH Control Packets in the order in which the original PUBLISH
	Control Packets were sent.
Reference	[MQTT-4.6.0-1], [MQTT-4.6.0-6], [MQTT-4.4.0-1]
PICS Selection	PICS_BROKER_QOS_1
Initial Conditions	
with {	
the CLIENT_1 having a MQTT_CONNECTION to the IUT	
and	
the CLIENT_2 having a MQTT_CONNECTION to the IUT	

```
and
   the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
    requested_qos corresponding to AT_LEAST_ONCE;
   to the IUT
   and
   the CLIENT_1 sent a PUBLISH message containing
    publish_header containing
      gos_level corresponding to AT_LEAST_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_1;
   to the IUT
   and
   the CLIENT_1 sent a PUBLISH message containing
    publish_header containing
      qos_level corresponding to AT_LEAST_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_2;
   to the IUT
   and
   the IUT sent a PUBLISH message containing
    publish_header containing
      gos_level corresponding to AT_LEAST_ONCE.
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_1;
   to the CLIENT_2
   and
   the IUT sent a PUBLISH message containing
    publish_header containing
       qos_level corresponding to AT_LEAST_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_2;
   to the CLIENT_2
                                            Expected Behaviour
ensure that {
  when {
      the IUT received no PUBACK message containing
     packet_identifier corresponding to PACKET_ID_1;
    the IUT received no PUBACK message containing
     packet_identifier corresponding to PACKET_ID_2;
  then {
      the IUT sends a PUBLISH message containing
     publish_header containing
        qos_level corresponding to AT_LEAST_ONCE,
     topic_name corresponding to PX_PUBLISH_TOPIC,
     packet_identifier corresponding to PACKET_ID_1;
    the IUT sends a PUBLISH message containing
     publish_header containing
        qos_level corresponding to AT_LEAST_ONCE,
     topic_name corresponding to PX_PUBLISH_TOPIC,
     packet_identifier corresponding to PACKET_ID_2;
  }
                                              Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_PUBSUB_007
Test Objective
                   Verify that the IUT resends PUBLISH Control Packets in the order in which the original PUBLISH
                   Control Packets were sent.
                   [MQTT-4.6.0-1], [MQTT-4.6.0-6], [MQTT-4.4.0-1]
Reference
PICS Selection
                   PICS_BROKER_QOS_2
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
   and
   the CLIENT_2 having a MQTT_CONNECTION to the IUT
   the CLIENT_2 subscribed the PX_PUBLISH_TOPIC containing
    requested_qos corresponding to EXACTLY_ONCE;
   to the IUT
   the CLIENT_1 sent a PUBLISH message containing
    publish_header containing
      qos_level corresponding to EXACTLY_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_1;
   to the IUT
   and
   the CLIENT_1 sent a PUBLISH message containing
    publish_header containing
      qos_level corresponding to EXACTLY_ONCE,
    topic name corresponding to PX PUBLISH TOPIC,
    packet_identifier corresponding to PACKET_ID_2;
   to the IUT
   and
   the IUT sent a PUBLISH message containing
    publish_header containing
      qos_level corresponding to EXACTLY_ONCE
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_1;
   to the CLIENT_2
   and
   the IUT sent a PUBLISH message containing
    publish_header containing
      qos_level corresponding to EXACTLY_ONCE,
    topic_name corresponding to PX_PUBLISH_TOPIC,
    packet_identifier corresponding to PACKET_ID_2;
   to the CLIENT_2
                                            Expected Behaviour
ensure that {
  when {
      the IUT not received a PUBREC message containing
     packet_identifier corresponding to PACKET_ID_1;
    the IUT not received a PUBREC message containing
     packet_identifier corresponding to PACKET_ID_2;
  then {
      the IUT sends a PUBLISH message containing
     publish_header containing
        qos_level corresponding to EXACTLY_ONCE,
     topic_name corresponding to PX_PUBLISH_TOPIC,
     packet_identifier corresponding to PACKET_ID_1;
    the IUT sends a PUBLISH message containing
     publish_header containing
        gos_level corresponding to EXACTLY_ONCE
     topic_name corresponding to PX_PUBLISH_TOPIC
```

```
packet_identifier corresponding to PACKET_ID_2;
;
}

Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_QOS_001
                   Verify that the IUT delivers PUBLISH Control Packets (in case of overlapping topic filter)
Test Objective
                   respecting the maximum QoS level of all matching subscriptions
Reference
                   [MQTT-3.3.5-1]
PICS Selection
                   PICS_BROKER_QOS_1
                                             Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
    the CLIENT_2 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_1 subscribed the TOPIC_FILTER_VALID containing
       qos_level corresponding to AT_MOST_ONCE;
    and
    the CLIENT_1 subscribed the TOPIC_FILTER_VALID_OVERLAP containing
      qos_level corresponding to AT_LEAST_ONCE;
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    header containing
     gos_level corresponding to AT_MOST_ONCE,
    topic_name corresponding to TOPIC_NAME_VALID_OVERLAP;
    from the CLIENT_2
  then {
      the IUT sends a PUBLISH message containing
    header containing
     qos_level corresponding to AT_LEAST_ONCE;
    to the CLIENT_1
    or
    the IUT sends a PUBLISH message containing
    header containing
     qos_level corresponding to AT_MOST_ONCE;
    to the CLIENT_1
  }
                                             Final Conditions
```

TP ld	TP_MQTT_BROKER_FEAT_QOS_002	
Test Objective	Verify that the IUT delivers PUBLISH Control Packets (in case of overlapping topic filter)	
-	respecting the maximum QoS level of all matching subscriptions.	
Reference	[MQTT-3.3.5-1]	
PICS Selection	PICS_BROKER_QOS_2	
Initial Conditions		
with {		
the CLIENT_1 having a MQTT_CONNECTION to the IUT		
and		
the CLIENT_2 having a MQTT_CONNECTION to the IUT		
and		
the CLIENT_1 subscribed the TOPIC_FILTER_VALID containing		
qos_level corresponding to AT_MOST_ONCE;		
and		
the CLIENT_1	the CLIENT_1 subscribed the TOPIC_FILTER_VALID_OVERLAP containing	

```
gos_level corresponding to EXACTLY_ONCE;
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    header containing
     gos_level corresponding to AT_MOST_ONCE,
    topic_name corresponding to TOPIC_NAME_VALID_OVERLAP;
    from the CLIENT_2
  then {
      the IUT sends a PUBLISH message containing
    header containing
     qos_level corresponding to EXACTLY_ONCE;
    to the CLIENT_1
    or
    the IUT sends a PUBLISH message containing
    header containing
     qos_level corresponding to AT_MOST_ONCE;
    to the CLIENT_1
  }
                                             Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_QOS_003
Test Objective
                   Verify that the IUT delivers PUBLISH Control Packets (in case of overlapping topic filter)
                   respecting the maximum QoS level of all matching subscriptions.
Reference
                   [MQTT-3.3.5-1]
PICS Selection
                   PICS_BROKER_QOS_2
                                             Initial Conditions
with {
       the CLIENT_1 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_2 having a MQTT_CONNECTION to the IUT
    the CLIENT_1 subscribed the TOPIC_FILTER_VALID containing
      qos_level corresponding to AT_LEAST_ONCE;
    and
    the CLIENT_1 subscribed the TOPIC_FILTER_VALID_OVERLAP containing
      qos_level corresponding to EXACTLY_ONCE;
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    header containing
     gos_level corresponding to AT_LEAST_ONCE,
    topic_name corresponding to TOPIC_NAME_VALID_OVERLAP;
    from the CLIENT_2
  then {
      the IUT sends a PUBLISH message containing
    header containing
     qos_level corresponding to EXACTLY_ONCE;
    to the CLIENT_1
    the IUT sends a PUBLISH message containing
    header containing
     gos_level corresponding to AT_LEAST_ONCE;
```

```
to the CLIENT_1
}

Final Conditions
```

```
TP Id
                   TP_MQTT_BROKER_FEAT_QOS_004
Test Objective
                   Verify that the IUT delivers PUBLISH Control Packets (in case of overlapping topic filter)
                   respecting the maximum QoS level of all matching subscriptions.
Reference
                   [MQTT-3.3.5-1]
PICS Selection
                   PICS BROKER QOS 1
                                            Initial Conditions
with {
      the CLIENT_1 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_2 having a MQTT_CONNECTION to the IUT
    and
    the CLIENT_1 subscribed the TOPIC_FILTER_VALID containing
      qos_level corresponding to AT_LEAST_ONCE;
    and
    the CLIENT_1 subscribed the TOPIC_FILTER_VALID_OVERLAP containing
      qos_level corresponding to AT_MOST_ONCE;
                                           Expected Behaviour
ensure that {
  when {
      the IUT receives a PUBLISH message containing
    header containing
     qos_level corresponding to AT_LEAST_ONCE,
    topic_name corresponding to TOPIC_NAME_VALID_OVERLAP;
    from the CLIENT_2
  then {
      the IUT sends a PUBLISH message containing
    header containing
     qos_level corresponding to AT_MOST_ONCE;
    to the CLIENT_1
  }
                                             Final Conditions
```

TP Id	TP MQTT BROKER FEAT QOS 005	
Test Objective	Verify that the IUT delivers PUBLISH Control Packets (in case of overlapping topic filter)	
•	respecting the maximum QoS level of all matching subscriptions.	
Reference	[MQTT-3.3.5-1]	
PICS Selection	PICS_BROKER_QOS_2	
	Initial Conditions	
with {		
the CLIENT_1 having a MQTT_CONNECTION to the IUT		
and	and	
the CLIENT_2 having a MQTT_CONNECTION to the IUT		
and		
	the CLIENT_1 subscribed the TOPIC_FILTER_VALID containing	
qos_level co	qos_level corresponding to EXACTLY_ONCE;	
and		
<u> </u>	the CLIENT_1 subscribed the TOPIC_FILTER_VALID_OVERLAP containing	
qos_level co	qos_level corresponding to AT_LEAST_ONCE;	
}		
Expected Behaviour		
ensure that {		
when {		
the IUT rece	the IUT receives a PUBLISH message containing	

```
header containing
    qos_level corresponding to EXACTLY_ONCE,
    topic_name corresponding to TOPIC_NAME_VALID_OVERLAP;
;
from the CLIENT_2
}
then {
    the IUT sends a PUBLISH message containing
    header containing
    qos_level corresponding to AT_LEAST_ONCE;
;
    to the CLIENT_1
}

Final Conditions
```

6 Test Purposes for MQTT Client

TP Id	TP_MQTT_CLIENT_CONNECT_001		
Test Objective	Verify that the IUT is able to send CONNECT Control Packets with valid Header Flags.		
Reference	[MQTT-2.2.2-1]		
PICS Selection	PICS_CLIENT_BASIC		
Initial Conditions			
	Expected Behaviour		
ensure that { when { the IUT is triggered to send a CONNECT message } then { the IUT sends a CONNECT message containing header_flags indicating value '0000'B; } }			
Final Conditions			

TP Id	TP_MQTT_CLIENT_CONNECT_002		
Test Objective	The protocol name representing the protocol is a UTF-8 encoded 'MQTT' string. Verify that the		
	IUT is able to send CONNECT Control Packets with valid protocol name represented by a UTF-8 encoded 'MQTT' string.		
Reference	[MQTT-3.1.2-1]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {			
when {			
the IUT is triggered to send a CONNECT message			
}	}		
then {			
I .	ds a CONNECT message containing		
protocol_name corresponding to PROTOCOL_NAME;			
}			
}			
Final Conditions			

```
TP Id
                   TP_MQTT_CLIENT_CONNECT_003
Test Objective
                   Verify that the IUT is able to send CONNECT Control Packets with Protocol Level 0x04 for MQTT
                   3.1.1
Reference
                   [MQTT-3.1.2-2]
PICS Selection
                   PICS_CLIENT_BASIC
                                             Initial Conditions
                                            Expected Behaviour
ensure that {
  when {
       the IUT is triggered to send a CONNECT message
  then {
       the IUT sends a CONNECT message containing
        protocol_level indicating value 0x04;
  }
                                              Final Conditions
```

TP Id	TP_MQTT_CLIENT_CONNECT_004
Test Objective	Verify that the IUT is able to send CONNECT Control Packets with valid reserved flag.
Reference	[MQTT-3.1.2-3]
PICS Selection	PICS_CLIENT_BASIC
	Initial Conditions
	Expected Behaviour
then { the IUT send connect_flat	ggered to send a CONNECT message Is a CONNECT message containing Igs containing field indicating value '0'B;
}	
Final Conditions	

	T	
TP ld	TP_MQTT_CLIENT_CONNECT_005	
Test Objective	Verify that the IUT is able to send CONNECT Control Packets with valid Last Will Testament	
	settings.	
Reference	[MQTT-3.1.2-9], [MQTT-3.1.2-14]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
	Expected Behaviour	
ensure that {		
when {		
the IUT is to	riggered to send a CONNECT message containing	
	ags containing	
	will_flag indicating value '1'B;	
wiii_nag	maleating value 12,	
,		
}		
then {		
the IUT sen	the IUT sends a CONNECT message containing	
connect_f	connect_flags containing	
will_flag indicating value '1'B,		
will_gos corresponding to VALID_QOS;		
11900		
, payload o	potaining	
	payload containing	
will_topic	indicating value not omit,	

```
will_message indicating value not omit;
;
}
Final Conditions
```

TP ld	TP_MQTT_CLIENT_CONNECT_006		
Test Objective	Verify that the IUT is able to send CONNECT Control Packets without Last Will Testament.		
Reference	[MQTT-3.1.2-11], [MQTT-3.1.2-13], [MQTT-3.1.2-15]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that { when {			
	riggered to send a CONNECT message containing		
	lags containing		
will_flag i	will_flag indicating value '0'B;		
;			
}			
then {	L CONTEST		
	ds a CONNECT message containing		
	lags containing		
	indicating value '0'B,		
	will_qos corresponding to AT_MOST_ONCE,		
wiii_retaii	n indicating value '0'B;		
,			
	payload containing		
will_topic indicating value omit,			
wiii_mess	sage indicating value omit;		
,			
} \			
Final Conditions			
	1 mai vonamono		
i .			

TP Id	TP_MQTT_CLIENT_CONNECT_007	
Test Objective	Verify that the IUT is able to send CONNECT Control Packets with valid settings for a connection	
	without authentication.	
Reference	[MQTT-3.1.2-18], [MQTT-3.1.2-20], [MQTT-3.1.2-22]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
	Expected Behaviour	
ensure that {	·	
when {		
	riggered to send a CONNECT message containing	
connect_f	lags containing	
user_nar	me_flag indicating value '0'B;	
;		
}		
then {		
the IUT ser	nds a CONNECT message containing	
	lags containing	
user_nar	me_flag indicating value '0'B,	
passwore	d_flag indicating value '0'B;	
,		
payload containing		
user_name indicating value omit,		
passwore	password indicating value omit;	
;		
}		
}		

Final Conditions

```
TP Id
                    TP_MQTT_CLIENT_CONNECT_008
                    Verify that the IUT is able to send CONNECT Control Packets with only a User Name.
Test Objective
Reference
                    [MQTT-3.1.2-19]
PICS Selection
                    PICS_CLIENT_BASIC
                                               Initial Conditions
                                              Expected Behaviour
ensure that {
  when {
       the IUT is triggered to send a CONNECT message containing
        connect_flags containing
         user_name_flag indicating value '1'B,
         password_flag indicating value '0'B;
  }
then {
       the IUT sends a CONNECT message containing
        connect_flags containing
         user_name_flag indicating value '1'B,
         password_flag indicating value '0'B;
        payload containing
         user_name indicating value not omit,
         password indicating value omit;
  }
                                                Final Conditions
```

TP Id	TP_MQTT_CLIENT_CONNECT_009		
Test Objective	Verify that the IUT is able to send CONNECT Control Packets with a User Name and Password.		
Reference	[MQTT-3.1.2-21], [MQTT-3.1.2-21]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
	Expected Behaviour		
ensure that {	·		
when {			
the IUT is tr	iggered to send a CONNECT message containing		
connect_fl	ags containing		
user_nan	ne_flag indicating value '1'B,		
password	d_flag indicating value '1'B;		
÷,			
}			
then {			
	ds a CONNECT message containing		
	ags containing		
user_name_flag indicating value '1'B,			
password_flag indicating value '1'B;			
, ,			
	payload containing		
user_name indicating value not omit,			
password	d indicating value not omit;		
, ;			
}			
J	Final Conditions		
	i iiiai ooliaitiolis		

```
TP Id
                    TP MQTT CLIENT CONNECT 010
Test Objective
                    Verify that the IUT sends CONNECT Control Packets with Payload fields appearing in a correct
                    order.
Reference
                    [MQTT-3.1.3-1]
PICS Selection
                    PICS_CLIENT_BASIC
                                               Initial Conditions
                                             Expected Behaviour
ensure that {
  when {
       the IUT is triggered to send a CONNECT message containing
        connect_flags containing
         user_name_flag indicating value '1'B,
         password_flag indicating value '1'B,
         will flag indicating value '1'B,
         will_gos corresponding to AT_MOST_ONCE;
        payload containing
         client identifier corresponding to VALID CLIENT ID.
         will_topic corresponding to PX_WILL_TOPIC
         will_message corresponding to PX_WILL_MESSAGE,
         user_name corresponding to PX_MQTT_USER_NAME,
         password corresponding to PX_MQTT_PASSWORD;
 }
then {
       // Assumption: by comparing each field with the sent values, the order is checked implicitly.
        // Wrong order would silently swap the fields during decoding
        the IUT sends a CONNECT message containing
        connect_flags containing
         user_name_flag indicating value '1'B,
         password_flag indicating value '1'B,
         will_flag indicating value '1'B;
        payload containing
         client_identifier corresponding to VALID_CLIENT_ID,
         will_topic corresponding to PX_WILL_TOPIC,
         will_message corresponding to PX_WILL_MESSAGE.
         user_name corresponding to PX_MQTT_USER_NAME,
         password corresponding to PX_MQTT_PASSWORD;
  }
                                               Final Conditions
```

TP Id	TP_MQTT_CLIENT_CONNECT_011	
Test Objective	Verify that the IUT is able to send CONNECT Control Packets with a well-formed UTF-8 encoded	
	client identifier.	
Reference	[MQTT-3.1.3-4], [MQTT-1.5.3-1]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
	Expected Behaviour	
ensure that {		
when {		
the IUT is t	riggered to send a CONNECT message containing	
payload c	ontaining	
client_id	entifier corresponding to VALID_CLIENT_ID; // TODO: required to trigger a concrete Client ID?	
;		
}		
then {		
// TODO: s	// TODO: sufficient for [MQTT-3.1.3-4] ?	
the IUT se	the IUT sends a CONNECT message containing	
payload c	payload containing	
client_id	client_identifier corresponding to VALID_CLIENT_ID;	

```
;
}
Final Conditions
```

TP ld	TP_MQTT_CLIENT_CONNECT_012
Test Objective	Verify that the IUT sets the clean_session flag to 1 if it connects with a zero-byte client identifier.
Reference	[MQTT-3.1.3-7]
PICS Selection	PICS_CLIENT_BASIC
	Initial Conditions
	Expected Behaviour
payload co client_ide ; } then { the IUT sen- connect_flater clean_set ,	chtifier corresponding to CLIENT_ID_ZERO_BYTES; ds a CONNECT message containing ags containing ssion indicating value '1'B;
,	Final Conditions
	i mai conditions

TP Id	TP_MQTT_CLIENT_CONNECT_013		
Test Objective	Verify that IUT encodes the Will Topic to well-formed UTF-8 encoded string.		
Reference	[MQTT-3.1.3-10]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
	Expected Behaviour		
	ggered to send a CONNECT message containing		
	connect_flags containing will_flag indicating value '1'B;		
;	ntaining corresponding to PX_WILL_TOPIC; // TODO: required to trigger a concrete topic?		
then { // TODO: sufficient for [MQTT-3.1.3-10] ? the IUT sends a CONNECT message containing connect_flags containing will_flag indicating value '1'B;			
payload cor will_topic o ;	ntaining corresponding to TOPIC_NAME_VALID;		
}	Final Canditions		
	Final Conditions		

```
TP Id
                   TP_MQTT_CLIENT_CONNECT_014
Test Objective
                   Verify that IUT encodes the User Name to well-formed UTF-8 encoded string.
Reference
                   [MQTT-3.1.3-11]
PICS Selection
                   PICS_CLIENT_BASIC
                                             Initial Conditions
                                            Expected Behaviour
ensure that {
  when {
       the IUT is triggered to send a CONNECT message containing
        connect_flags containing
         user_name_flag indicating value '1'B;
        payload containing
         user_name corresponding to PX_MQTT_USER_NAME; // TODO: required to trigger a concrete
username?
  then {
       // TODO: sufficient for [MQTT-3.1.3-11]?
        the IUT sends a CONNECT message containing
        connect_flags containing
         user_name_flag indicating value '1'B;
        payload containing
         user_name corresponding to USER_NAME_VALID_UTF8;
  }
                                              Final Conditions
```

T		
TP Id	TP_MQTT_CLIENT_CONNACK_001	
Test Objective	Verify that the IUT closes the network connection on receiption of a CONNACK Control Packet	
	with invalid fixed header.	
Reference	[MQTT-2.2.2-1], [MQTT-2.2.2-2], [MQTT-4.8.0-1]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
with {		
the IUT is trig	gered to send a CONNECT message	
}	· ·	
Expected Behaviour		
ensure that {		
when {		
the IUT recei	ves a CONNACK message containing	
header_flags i	header_flags indicating value '1111'B;	
}		
then {		
the IUT closes the TCP_CONNECTION		
}		
}		
Final Conditions		

TP Id	TP_MQTT_CLIENT_PUBLISH_001		
Test Objective	Verify that the IUT sets the DUP flag to 0 for all QoS 0 PUBLISH Control Packets.		
Reference	[MQTT-3.3.1-2]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
with {	with {		
the IUT havin	the IUT having a MQTT_CONNECTION to the TEST_SYSTEM		
\mathbf{B}			
Expected Behaviour			
ensure that {			
when {			

```
the IUT is triggered to send a PUBLISH message containing
header containing
qos_level corresponding to AT_MOST_ONCE;
;
}
then {
the IUT sends a PUBLISH message containing
header containing
qos_level corresponding to AT_MOST_ONCE,
dup_flag indicating value '0'B;
;
}

Final Conditions
```

TP Id	TP_MQTT_CLIENT_PUBLISH_002	
Test Objective	Verify that the IUT sets the DUP flag to 0 for all QoS 1 PUBLISH Control Packets.	
Reference	[MQTT-3.3.1-1]	
PICS Selection	PICS_CLIENT_QOS_1	
	Initial Conditions	
with { the IUT hat }	with { the IUT having a MQTT_CONNECTION to the TEST_SYSTEM }	
	Expected Behaviour	
header co qos_leve ; } then { the IUT is header co qos_lev	ensure that { when { the IUT is triggered to send a PUBLISH message containing header containing qos_level corresponding to AT_LEAST_ONCE; ; }	
Final Conditions		

TP Id	TP_MQTT_CLIENT_PUBLISH_003	
Test Objective	Verify that the IUT sets the DUP flag to 0 for all QoS 2 PUBLISH Control Packets.	
Reference	[MQTT-3.3.1-1]	
PICS Selection	PICS_CLIENT_QOS_2	
	Initial Conditions	
with { the IUT having a MQTT_CONNECTION to the TEST_SYSTEM }		
	Expected Behaviour	
qos_level \(\) then { the IUT send \(\) qos_level	ggered to send a PUBLISH message containing corresponding to EXACTLY_ONCE; Is a PUBLISH message containing header containing corresponding to EXACTLY_ONCE, ndicating value '0'B;	

Final Conditions

```
TP Id
                   TP_MQTT_CLIENT_PUBLISH_004
Test Objective
                   Verify that IUT encodes the topic name to a well-formed UTF-8 encoded string.
                   [MQTT-3.3.2-1]
Reference
PICS Selection
                  PICS_CLIENT_BASIC
                                            Initial Conditions
with {
      the IUT having a MQTT_CONNECTION to the TEST_SYSTEM
                                           Expected Behaviour
ensure that {
  when {
      the IUT is triggered to send a PUBLISH message
  then {
      the IUT sends a PUBLISH message containing
         topic_name not corresponding to TOPIC_NAME_INVALID_UTF8;
  }
                                             Final Conditions
```

TP Id	TP_MQTT_CLIENT_PUBLISH_005	
Test Objective	Verify that the IUT does not send PUBLISH Control Packets which contain only valid topic names	
	without wildcard characters.	
Reference	[MQTT-3.3.2-2]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
with {		
the IUT havi	ng a MQTT_CONNECTION to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT is trig	the IUT is triggered to send a PUBLISH message	
}		
then {	then {	
the IUT sends a PUBLISH message containing		
topic_name corresponding to TOPIC_NAME_VALID;		
}		
}		
Final Conditions		

TP ld	TP_MQTT_CLIENT_PUBLISH_006		
Test Objective	Verify that the IUT does not send any response on reception of a QoS level 0 PUBLISH Control		
-	Packet.		
Reference	[MQTT-3.3.4-1]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
with {			
the IUT hav	ring a MQTT_CONNECTION to the TEST_SYSTEM		
and			
the IUT sub	the IUT subscribed the PX_PUBLISH_TOPIC to the TEST_SYSTEM		
}	}		
	Expected Behaviour		
ensure that {			
when {			
the IUT receives a PUBLISH message containing			
header o	header containing		
gos_level corresponding to AT_MOST_ONCE,			
topic_na	topic_name corresponding to PX_PUBLISH_TOPIC;;		

```
}
then {
the IUT sends no response message
}
}
Final Conditions
```

TP ld	TP_MQTT_CLIENT_PUBLISH_007	
Test Objective	Verify that the IUT responds to a QoS level 1 PUBLISH Control Packet with a PUBACK Control	
	Packet.	
Reference	[MQTT-3.3.4-1]	
PICS Selection	PICS_CLIENT_QOS_1	
	Initial Conditions	
with {		
the IUT havi	ng a MQTT_CONNECTION to the TEST_SYSTEM	
and		
the IUT subsc	cribed the PX_PUBLISH_TOPIC to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {		
when {		
	ives a PUBLISH message containing	
header co		
qos_level corresponding to AT_LEAST_ONCE,		
packet_identifier corresponding to PACKET_ID,		
topic_nam	topic_name corresponding to PX_PUBLISH_TOPIC;;	
}		
then {	L. DUDAGIC	
the IUT sends a PUBACK message containing		
packet_ide	entifier corresponding to PACKET_ID;	
}		
Final Conditions		
	Final Conditions	

```
TP Id
                   TP_MQTT_CLIENT_PUBLISH_009
Test Objective
                   Verify that the IUT assigns a non-zero packet identifier on each new PUBLISH Control Packet
                   with QoS level > 0
Reference
                   [MQTT-2.3.1-1]
PICS Selection
                   PICS_CLIENT_QOS_1
                                             Initial Conditions
with {
      the IUT having a MQTT_CONNECTION to the TEST_SYSTEM
                                            Expected Behaviour
ensure that {
  when {
      the IUT is triggered to send a PUBLISH message containing
     header containing
        qos_level corresponding to AT_LEAST_ONCE;;
  then {
      the IUT sends a PUBLISH message containing
       packet_identifier corresponding to PACKET_ID_NON_ZERO;
  }
                                             Final Conditions
```

TP ld	TP_MQTT_CLIENT_PUBLISH_010	
Test Objective	Verify that the IUT assigns a currently unused packet identifier on each new PUBLISH Control	
-	Packet with QoS level > 0	
Reference	[MQTT-2.3.1-2]	
PICS Selection	PICS_CLIENT_QOS_1	
	Initial Conditions	
with { the IUT having a MQTT_CONNECTION to the TEST_SYSTEM }		
	Expected Behaviour	
ensure that { when {		
	ggered to send a PUBLISH message containing	
header contair		
	qos_level corresponding to AT_LEAST_ONCE;;	
	and	
header contain	gered to send a PUBLISH message containing	
qos_level corresponding to AT_LEAST_ONCE;;		
then {		
	s a PUBLISH message containing	
	packet_identifier corresponding to PACKET_ID_1;	
and		
the IUT sends a PUBLISH message containing		
packet_identifier corresponding to PACKET_ID_2;		
}		
}		
Final Conditions		

TP Id	TP_MQTT_CLIENT_PUBLISH_011	
Test Objective	Verify that the IUT does not assign a packet identifier on PUBLISH Control Packet with QoS level	
	equals 0	
Reference	[MQTT-2.3.1-5]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
with {	with {	
the IUT having a MQTT_CONNECTION to the TEST_SYSTEM		
· · · · · · · · · · · · · · · · · · ·		
Expected Behaviour		

```
ensure that {
    when {
        the IUT is triggered to send a PUBLISH message containing
        header containing
        qos_level corresponding to AT_MOST_ONCE;;
}
then {
        the IUT sends a PUBLISH message containing
        packet_identifier indicating value omit;
}

Final Conditions
```

TP Id	TP_MQTT_CLIENT_PUBACK_001	
Test Objective	Verify that the IUT is able to send PUBACK Control Packets with valid Header Flags.	
Reference	[MQTT-2.2.2-1]	
PICS Selection	PICS_CLIENT_BASIC	
	Initial Conditions	
with {		
the IUT havin	ng a MQTT_CONNECTION to the TEST_SYSTEM	
and		
the IUT subscril	bed the PX_PUBLISH_TOPIC to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT recei	the IUT receives a PUBLISH message containing	
header cont	taining	
qos_leve	qos_level corresponding to AT_LEAST_ONCE,	
topic_name corresponding to PX_PUBLISH_TOPIC;;		
}		
then {		
the IUT sends a PUBACK message containing		
header_flags indicating value '0000'B;		
}	-	
}		
Final Conditions		

TP Id	TP MQTT CLIENT PUBACK 002	
Test Objective	Verify that the IUT acknowledges a PUBLISH Control Packet with the correct packet identifier.	
Reference	[MQTT-2.3.1-6]	
PICS Selection	PICS_CLIENT_QOS_1	
	Initial Conditions	
with {		
the IUT havin	ng a MQTT_CONNECTION to the TEST_SYSTEM	
and		
the IUT subscrib	ped the PX_PUBLISH_TOPIC to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {	•	
when {		
•	the IUT receives a PUBLISH message containing	
header cont	· ·	
	qos_level corresponding to AT_LEAST_ONCE,	
	topic_name corresponding to PX_PUBLISH_TOPIC,	
packet_identifier corresponding to PACKET_ID;;		
}		
then {		
the IUT sends a PUBACK message containing		
packet_identifier corresponding to PACKET_ID;		
1	pasion derivation corresponding to 1 None 1 _ ib,	
, ,		
}		

Final Conditions

```
TP Id
                   TP_MQTT_CLIENT_PUBREC_001
                   Verify that the IUT is able to send PUBREC Control Packets with valid Header Flags.
Test Objective
Reference
                   [MQTT-2.2.2-1]
PICS Selection
                   PICS_CLIENT_QOS_2
                                             Initial Conditions
with {
       the IUT having a MQTT_CONNECTION to the TEST_SYSTEM
    the IUT subscribed the PX_PUBLISH_TOPIC to the TEST_SYSTEM
                                           Expected Behaviour
ensure that {
  when {
       the IUT receives a PUBLISH message containing
        header containing
          qos_level corresponding to EXACTLY_ONCE,
        topic_name corresponding to PX_PUBLISH_TOPIC;;
  }
  then {
       the IUT sends a PUBREC message containing
        header_flags indicating value '0000'B;
  }
                                             Final Conditions
```

TP Id	TP_MQTT_CLIENT_PUBREC_002	
Test Objective	Verify that the IUT acknowledges a PUBLISH Control Packet with the correct packet identifier.	
Reference	[MQTT-2.3.1-6]	
PICS Selection	PICS_CLIENT_QOS_2	
	Initial Conditions	
with {		
the IUT havin	g a MQTT_CONNECTION to the TEST_SYSTEM	
and		
the IUT subscrib	ped the PX_PUBLISH_TOPIC to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT receive	ves a PUBLISH message containing	
header cont		
	corresponding to EXACTLY_ONCE,	
topic_name	topic_name corresponding to PX_PUBLISH_TOPIC,	
packet_identifier corresponding to PACKET_ID;;		
}		
then {	then {	
the IUT sends a PUBREC message containing		
acket_identi	acket_identifier corresponding to PACKET_ID;	
}		
}		
Final Conditions		

TP Id	TP_MQTT_CLIENT_PUBREL_001	
Test Objective	Verify that the IUT is able to send PUBREL Control Packets with valid Header Flags.	
Reference	[MQTT-2.2.2-1]	
PICS Selection	PICS_CLIENT_QOS_2	
Initial Conditions		
with { the IUT having a MQTT_CONNECTION to the TEST_SYSTEM }		

```
ensure that {
   when {
       the IUT receives a PUBREC message
   }
   then {
       the IUT sends a PUBREL message containing
       header_flags indicating value '0000'B;
   }
}

Final Conditions
```

TP ld	TP_MQTT_CLIENT_PUBREL_002	
Test Objective	Verify that the IUT acknowledges a PUBREC Control Packet with the correct packet identifier with	
_	a PUBREL Control Packet.	
Reference	[MQTT-2.3.1-6]	
PICS Selection	PICS_CLIENT_QOS_2	
	Initial Conditions	
with {		
the IUT havin	ng a MQTT_CONNECTION to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT recei	the IUT receives a PUBREC message containing	
packet_identifier corresponding to PACKET_ID;		
,		
then {		
the IUT send	s a PUBREL message containing	
packet_identifier corresponding to PACKET_ID;		
}	· · · · · · · · ·	
}		
Final Conditions		

TP Id	TP_MQTT_CLIENT_PUBCOMP_001	
Test Objective	Verify that the IUT is able to send PUBCOMP Control Packets with valid Header Flags.	
Reference	[MQTT-2.2.2-1]	
PICS Selection	PICS_CLIENT_QOS_2	
	Initial Conditions	
with {		
the IUT havir	ng a MQTT_CONNECTION to the TEST_SYSTEM	
and		
the IUT subscril	bed the PX_PUBLISH_TOPIC to the TEST_SYSTEM	
}		
	Expected Behaviour	
ensure that {		
when {		
the IUT recei	the IUT receives a PUBREL message	
}	}	
then {	then {	
the IUT sends a PUBCOMP message containing		
header_flags indicating value '0000'B;		
}		
}		
Final Conditions		

```
TP Id
                   TP_MQTT_CLIENT_SUBSCRIBE_001
Test Objective
                   Verify that the IUT is able to send SUBSCRIBE Control Packets with valid Header Flags.
Reference
                   [MQTT-2.2.2-1], [MQTT-3.8.1-1]
PICS Selection
                   PICS_CLIENT_BASIC
                                             Initial Conditions
with {
       the IUT having a MQTT_CONNECTION to the TEST_SYSTEM
                                            Expected Behaviour
ensure that {
  when {
       the IUT is triggered to send a SUBSCRIBE message
  then {
       the IUT sends a SUBSCRIBE message containing
         header_flags indicating value '0010'B;
  }
                                              Final Conditions
```

TP Id	TP_MQTT_CLIENT_SUBSCRIBE_002		
Test Objective	Verify that the IUT assigns a non-zero packet identifier on each new SUBSCRIBE Control Packe		
Reference	[MQTT-2.3.1-2]		
PICS Selection	PICS_CLIENT_BASIC		
	Initial Conditions		
with { the IUT having a MQTT_CONNECTION to the TEST_SYSTEM }			
	Expected Behaviour		
ensure that { when { the IUT is triggered to send a SUBSCRIBE message } then { the IUT sends a SUBSCRIBE message containing packet_identifier corresponding to PACKET_ID_1; } }			
Final Conditions			

TD L	TR MOTT OUTENT OUTPOORING 000			
TP Id	TP_MQTT_CLIENT_SUBSCRIBE_003			
Test Objective	Verify that the IUT assigns a currently unsued packet identifier on each new SUBSCRIBE Control			
	Packet			
Reference	[MQTT-2.3.1-2]			
PICS Selection	PICS_CLIENT_BASIC			
	Initial Conditions			
with {				
the IUT havir	ng a MQTT_CONNECTION to the TEST_SYSTEM			
}				
	Expected Behaviour			
ensure that {	·			
when {				
the IUT is trig	agered to send a SUBSCRIBE message			
and	•			
the IUT is trigo	gered to send a SUBSCRIBE message			
}				
then {				
	ls a SUBSCRIBE message containing			
packet_identifier corresponding to PACKET_ID_1;				
and				
the IUT sends a SUBSCRIBE message containing				
packet_identifier corresponding to PACKET_ID_2;				
packet_idei	}			

```
Final Conditions
```

TP Id	TP_MQTT_CLIENT_UNSUBSCRIBE_001			
Test Objective	Verify that the IUT is able to send UNSUBSCRIBE Control Packets with valid Header Flags.			
Reference	[MQTT-2.2.2-1], [MQTT-3.10.1-1]			
PICS Selection	PICS_CLIENT_BASIC			
	Initial Conditions			
with {				
the IUT havin	ng a MQTT_CONNECTION to the TEST_SYSTEM			
}				
	Expected Behaviour			
ensure that {				
when {	when {			
the IUT is trig	gered to send a UNSUBSCRIBE message			
}	· ·			
then {				
the IUT send	s a UNSUBSCRIBE message containing			
header_flags indicating value '0010'B;				
}				
)				
Final Conditions				

TP ld	TP_MQTT_CLIENT_UNSUBSCRIBE_002		
Test Objective	Verify that IUT encodes the topic filter to a well-formed UTF-8 encoded string.		
Reference	[MQTT-3.10.3-1]		
PICS Selection	ction PICS_CLIENT_BASIC		
	Initial Conditions		
with { the IUT having a MQTT_CONNECTION to the TEST_SYSTEM }			
	Expected Behaviour		
ensure that { when { the IUT is triggered to send a UNSUBSCRIBE message } then { the IUT sends a UNSUBSCRIBE message containing payload containing topic_filter not corresponding to TOPIC_FILTER_INVALID_UTF8;; } }			
Final Conditions			

TP ld	TP_MQTT_CLIENT_DISCONNECT_001			
Test Objective	Verify that the IUT is able to send DISCONNECT Control Packets with valid Header Flags.			
Reference	[MQTT-2.2.2-1]			
PICS Selection	PICS_CLIENT_BASIC			
Initial Conditions				
with {				
the IUT havir	the IUT having a MQTT_CONNECTION to the TEST_SYSTEM			
}				
Expected Behaviour				
ensure that {				
when {				
the IUT is tric	the IUT is triggered to send a DISCONNECT message			
}	· ·			
then {	then {			
the IUT sends a DISCONNECT message containing				
header_flags indicating value '0000'B;				
}	goggg			

```
Final Conditions
```

```
TP Id
                   TP_MQTT_CLIENT_DISCONNECT_002
Test Objective
                   Verify that the IUT closes the network connection after sending a DISCONNECT Control Packet.
                   [MQTT-3.14.4-1]
PICS_CLIENT_BASIC
Reference
PICS Selection
                                             Initial Conditions
with {
      the IUT having a MQTT_CONNECTION to the TEST_SYSTEM
                                            Expected Behaviour
ensure that {
  when {
      the IUT is triggered to send a DISCONNECT message
  then {
      the IUT sends a DISCONNECT message
        the IUT closes the TCP_CONNECTION
  }
                                              Final Conditions
```

TP Id	TP_MQTT_CLIENT_FEAT_KEEPALIVE_001			
Test Objective	Verify that the IUT ensures that the interval between Control Packets being sent does not exceed			
	the Keep Alive value.			
Reference	[MQTT-3.1.2-23]			
PICS Selection	PICS_CLIENT_BASIC			
	Initial Conditions			
with {				
the IUT havin	the IUT having a MQTT_CONNECTION containing			
keep_alive	keep_alive corresponding to PX_KEEP_ALIVE;			
}				
	Expected Behaviour			
ensure that {				
when {				
the MQTT_C	ONNECTION times_out			
}	}			
then {				
the IUT send	the IUT sends a PINGREQ message			
}				
}				
Final Conditions				

Annex A (normative): MQTT Test Purposes (TPs)

A.0 Introduction

This Test purpose catalogue has been produced using the Test Description Language (TDL-TO) according to ETSI ES 203 119-4 [2]. The TDL-TO library modules corresponding to the Test purpose catalogue are contained in archive ts_10359701v010102p0.zip which accompanies the present document.

History

Document history				
V1.1.1	January 2021	Publication		
V1.1.2	January 2021	Publication		