ETSI TS 103 597-3 V1.1.1 (2021-01)

Methods for Testing and Specification (MTS);
Test Specification for MQTT;
Part 3: Performance Tests

Reference
DTS/MTS-TSTMQTT-3

Keywords
performance, testing

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

Contents

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 3 of a multi-part deliverable. Full details of the entire series can be found in part 1 [2].

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 ETSI Drafting Rules (Verbal forms for the expression of provisions).

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

Introduction

Technology advancements are bringing ever-increasing computing power and network speed in the communication domain. The number of communicating devices is expected to increase by 2 orders of magnitude in the following decade and with that several challenges emerge. A main challenge pertains to efficiency regarding resource consumption and overall performance.

As existing communication protocols evolve and new ones are created to fit the current technological capabilities and societal needs and the standards that serve the basis for interoperability and compliance. This is most relevant in the foreseen context of the Internet of Things (IoT) which envisions a very high density of connected devices in the near future. The Message Queuing Telemetry Transport (MQTT) protocol is one such example of evolution.

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. Although there are many published evaluations of various MQTT implementations, a lack of common language, methods and presentation of results is slowing down the adoption rate and overall evolution of the protocol.

In the present document the performance testing is presented. It provides a basis for benchmark testing and performance evaluation for the MQTT protocol.

1 Scope

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

NOTE: Available at http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html.

[2] ETSI TS 103 597-1: "Methods for Testing and Specification (MTS); Test Specification for MQTT; Part 1: Conformance Tests".

[3] 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] IETF RFC 2544: "Benchmarking Methodology for Network Interconnect Devices".

[i.2] ETSI TR 101 577: "Methods for Testing and Specifications (MTS); Performance Testing of Distributed Systems; Concepts and Terminology".

[i.3] ISO/IEC 9646-1: " Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 1: General concepts".

[i.4] ETSI ES 202 951: "Methods for Testing and Specification (MTS); Model-Based Testing (MBT); Requirements for Modelling Notations".

3 Definition of terms, symbols and abbreviations

3.1 Terms

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

active server: connected server with at least 1 registered topic

active subscriber: connected client with at least 1 topic subscription

application messages: data carried by the MQTT protocol across the network for the application as defined in the OASIS Standard: "MQTT Version 3.1.1" [1]

NOTE: When an Application Message is transported by MQTT it contains payload data, a Quality of Service (QoS), a collection of Properties, and a Topic Name.

benchmark test: procedure by which a test system interacts with a System Under Test to measure its behaviour and produce a benchmark report

benchmark test report: document generated at the conclusion of a test procedure containing the metrics measured during

client: program or device that uses MQTT

NOTE: A Client always establishes the Network Connection to the Server [1], it can:

- Publish Application Messages that other Clients might be interested in.
- Subscribe to request Application Messages that it is interested in receiving.
- Unsubscribe to remove a request for Application Messages.
- Disconnect from the Server.

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

Design Objective Capacity (DOC): largest load an SUT can sustain while not exceeding design objectives defined for a use-case

disallowed unicode code point: set of Unicode Control Codes and Unicode Noncharacters which should not be included in a UTF-8 Encoded String

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.3])

malformed packet: control packet that cannot be parsed according to the protocol specification

MQTT Protocol Packet: packet of information that is sent across the Network Connection as defined in the OASIS Standard: "MQTT Version 3.1.1" [1]

NOTE: The MQTT specification defines fourteen different types of Control Packet, one of which (the PUBLISH packet) is used to convey Application Messages.

parameter: attribute of a SUT, test system, system load, or traffic set whose value is set externally and prior to a benchmark test, and whose value affects the behaviour of the benchmark test

protocol error: error that is detected after the packet has been parsed and found to contain data that is not allowed by the protocol or is inconsistent with the state of the Client or Server as defined in the OASIS Standard

server: program or device that acts as an intermediary between Clients which publish Application Messages [1] and Clients which have made Subscriptions

NOTE: A Server can:

- Accepts Network Connections from Clients.
- Accepts Application Messages published by Clients.
- Process Subscribe and Unsubscribe requests from Clients.
- Forwards Application Messages that match Client Subscriptions.

session: stateful interaction between a Client and a Server

NOTE: Some Sessions last only as long as the Network Connection, others can span multiple consecutive Network Connections between a Client and a Server [1].

subscription: interaction between client and server within a session, where the client receives messages based on a topic filter

NOTE: A Subscription is associated with a single Session. A Session can contain more than one Subscription. Each Subscription within a session has a different Topic Filter [1]. Subscription comprises a Topic Filter and a maximum QoS as defined in the OASIS Standard: "MQTT Version 3.1.1" [1].

system under test: real open system in which the implementation under test resides (ETSI ES 202 951 [i.4])

test scenario: specific path through a use-case, whose implementation by a test system creates a system load

Topic name: label attached to an Application Message which is matched against the Subscriptions known to the Server as defined in the OASIS Standard: "MQTT Version 3.1.1" [1]

NOTE: The Server sends a copy of the Application Message to each Client that has a matching Subscription [1].

topic filter: expression contained in a Subscription, to indicate an interest in one or more topics as defined in the OASIS Standard: "MQTT Version 3.1.1" [1]

NOTE: A Topic Filter can include wildcard characters [1].

traffic-time profile: evolution of the average scenario over a time interval

use case: description of a goal that a user has in interacting with a system, the various actors and the SUT

3.2 Symbols

Void.

3.3 Abbreviations

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

CoAP	Constrained Application Protocol
CPU	Central Processing Unit
DOC	Design Objective Capacity
GTW	Gateway
IoT	Internet of Things
IUT	Implementation Under Test
KPI	Key Performance Indicator
LAN	Local Area Network
MQTT	Message Queuing Telemetry Transport
NoS	Number of Subscribers
OS	Operating System
PICS	Protocol Implementation Conformance Statement
QoS	Quality of Service

RAM	Random Access Memory
SoC	System on a Chip
SSD	Solid State Drive
SUT	System Under Test
TCP	Transmission Control Protocol
TDL	Test Description Language
TDL-TO	Test Description Language - Test Objectives
TS	Test System
WLF	Workload Factor

4 Performance metrics

4.0 Introduction

The performance metrics specified herein pertain to the specifics of a MQTT IUT. As such, the objective is to use these metrics in order to determine how well the MQTT component (be it client or server) is performing its functions. As MQTT is a transport protocol, the metrics will be focused on how fast, reliable and efficient the transport is handled. The metrics are designed to fit this purpose while covering multiple use-case scenarios. Following below are the specific messages of the MQTT protocol for which the performance metrics are defined.

4.1 Concepts

4.1.1 Test Equipment Considerations

For measuring performance of a given Test System (TS), a comprehensive description of the test environment is required. This includes but it is not limited to:

- TS hardware infrastructure: resource specification, type, capacity and distribution.
- test environment type and resources (virtualization technology, allocated resources).
- Measurement equipment hardware/software infrastructure, measurement probe distribution/placement, clock synchronization precision, allocated resources.
- Communication infrastructure: transport network specification, number of switches/hops between TS components, bandwidth capacity.

Additional to the specific characteristics of the SUT, the MQTT protocol specifies sessions as stateful interactions between clients and servers. Because of this, additional performance session-based metrics are considered.

4.1.2 Measurement Preliminary Considerations

In order for the collected measurement data to be useful, special consideration needs to be given to the TS setup. Given that the performance evaluation is targeting one or several IUTs same TS setup characteristics are required in order for the evaluation results to allow valid comparisons between them. Some of the characteristics may refer to infrastructure, hardware, physical or virtual resources as well as network connectivity resources.

4.2 Measurement Methodology

4.2.0 Introduction

This clause presents the test methodology for MQTT performance evaluation. From the performance perspective, all measurable metrics related to the protocol should be considered. Although not exhaustive, these metrics can be categorized as follows.

Powerfulness metrics, as defined in ETSI TR 101 577 [i.2] include 3 sub categories: Responsiveness, Capacity and Scalability. From the Responsiveness category the response time, roundtrip time and latency time metrics are used. From the Capacity category the arrival capacity, peak capacity, in progress capacity, streaming capacity and Throughput capacity metrics are used. From the Scalability category the scaling capacity metric is used.

Reliability metrics, as defined in ETSI TR 101 577 [i.2] include 6 sub categories: Quality of Service (QoS), Stability, Availability, Robustness, Recovery, and Correctness. The Quality of Service sub category refers to well defined requirements which may include acceptable values or ranges for metrics from other categories. Stability refers to the capacity of the System to deliver acceptable performance over time. From the Availability sub category, the logical availability metric is used. From the Robustness sub category, the service capacity reduction and service responsiveness deterioration metrics are used. From the Recovery sub category, the service restart characteristics metric is used. Correctness metrics cover the ability of delivering correctly processed requests under high or odd load conditions.

Efficiency metrics, as defined in ETSI TR 101 577 [i.2], cover resource utilization. The metrics cover the characteristics of resource usage, linearity, scalability and bottleneck. The efficiency metrics used in the present document are referring to the service level and not covering the platform level.

4.2.1 Metric Post-processing

The collection of metric values from a SUT is performed by multiple agents and/or directly by the IUT. Often a better insight into the IUT performance is gained by post-processing these values in order to get more meaningful results. To this scope, the data samples can be aggregates over time intervals in the experiment. From such common practices, the following are used for the metrics listed in this clause:

- Mean Average: $\frac{1}{n}\sum_1^n x_i$, where n is the number of samples and x is a sample value.
- Standard deviation: $\sqrt{\frac{1}{n}\sum_1^n(x_i - \bar{x_i})^2}$, where n is the number of samples, x is a sample value and $\bar{x}$ is the mean average.
- Minimum: min(x_i), the smallest sample value, relative to the rest of the samples.
- Maximum: max(x_i), the greatest sample value, relative to the rest of the samples.

4.2.2 Message Types

Table 1 contains the set control packet messages specified by the MQTT [1] standard.

Table 1: Message Types

Control Packet Name	Description	Client-> Server	Server-> Client	Payload
CONNECT	client requests a connection to the server	✓		Required
CONNACK	acknowledge connection request		✓	None
PUBLISH	Publish message	✓	✓	Optional
PUBACK	Publish acknowledgement	✓	✓	None
PUBREC	Publish received (QoS 2 publish received)	✓	✓	None
PUBREL	Publish release	✓	✓	None
PUBCOMP	Publish complete	✓	✓	None
SUBSCRIBE	Subscribe to topics	✓		Required
SUBACK	Subscribe acknowledgement		✓	Required
UNSUBSCRIBE	Unsubscribe from Topics	✓		Required
UNSUBACK	Unsubscribe acknowledgement		✓	Required
PINGREQ	Ping request	✓		None
PINGRESP	Ping response		✓	None
DISCONNECT	Disconnect notification	✓	✓	None
AUTH	Authentication Exchange	✓	✓	None

4.2.3 Test parameters

The benchmark test parameters are used to control the behaviour of the test script. The data elements required to configure the test system are listed in table 2.

Table 2 is a non-exhaustive list of test parameters defined for the benchmark standard. The list is expected to grow over time, as additional subsystems and system configurations are developed.

Table 2: Test parameters

Parameter	Description
Duration	Amount of time that a system load is presented to a SUT
Type of call	Type of messages contained within a workload
NoC	number of clients generating or subscribing to data/control traffic
NoS	Number of servers handling data/control traffic
Transport interface	Underlying transport interface
WLF for GTW	Work load factor for gateway expressed in number messages received per second, by type of message
Payload	Size of the data in Bytes carried within a message
Monitoring Window(s)	The time interval window for which the monitored metrics are recorded. This reflects the measuring accuracy (e.g. per second, minute, hour, etc.)
Validation threshold(s)	The specific metric thresholds used for validating whether a system performs at specifications

Table 3: Test output

Metric	Description
Minimum call duration	The minimum duration of a successful message request/response interaction within a Monitoring Window
Maximum call duration	The maximum duration of a successful message request/response interaction within a Monitoring Window
Average call duration	The average duration of a successful message request/response interaction within a Monitoring Window
Total number of calls	The total number of workload specified request/response type operations executed during the test
Success rate	Percentage number of successful workload operations relative to the total workload operations
Error rate	Percentage number of failed workload operations relative to the total workload operations
Requests processed per time unit	This metric reflects the average number of successfully processed requests per preferred time unit (second/minute/etc.)

4.2.4 Operation Message Flows

The IUT will be evaluated based on the metric values obtain as a result of the service operations using the messages described in clause 4.1.1. The set of messages exchanged triggered by the initial client message are further referred as operations. For the tests, the metrics use operations rather than specific messages because it is easier to handle the measurements. E.g. for a ping message, the roundtrip time is calculated as the duration between the time the PINGREQ is sent and the time when PINGRESP is received. If specific test system network measurements are available, by subtracting the measured network delayed from the duration, the operation processing time can be deducted:

1) CONNECT: This clause describes the MQTT operation types and message sequences required for test execution using a Client Connection example:

   Preconditions:

   - Unregistered Client, Server/Broker.
   - TCP connection between Client and Server/Broker established.

   Operation sequence:

   - Client sends CONNECT message.
   - Client receives CONNACK message.

   Measurement:

   - Time period expressed in milliseconds between the moment client forwards the CONNECT Message and the moment Client receives CONNACK message from server.

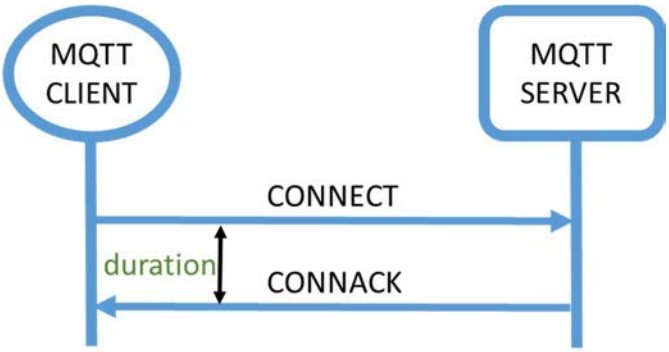


Figure 1: MQTT Server-Client CONNECT message flow

2) PINGREQ: This section describes the MQTT operation types and message sequences required for test execution using a Client ping example:

   Preconditions:

   - Connected Client, Server/Broker.
   - TCP connection between Client and Server/Broker established.

   Operation sequence:

   - Client sends PINGREQ message.
   - Client receives PINGRESP message.

   Measurement:

   - Time period expressed in milliseconds between the moment client forwards the PINGREQ Message and the moment Client receives PINGRESP message from server.

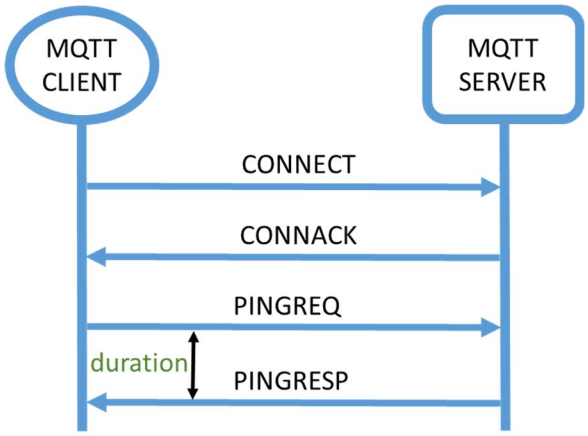


Figure 2: MQTT Server-Client PING message flow

3) SUBSCRIBE: This section describes the MQTT operation types and message sequences required for test execution using a Client SUBSCRIBE example:

Preconditions:

- Connected Client, Server/Broker.
- TCP connection between Client and Server/Broker established.

Operation sequence:

- Client sends SUBSCRIBE message.
- Client receives SUBACK message.

Measurement:

- Time period expressed in milliseconds between the moment client forwards the SUBSCRIBE Message and the moment Client receives SUBACK message from server.

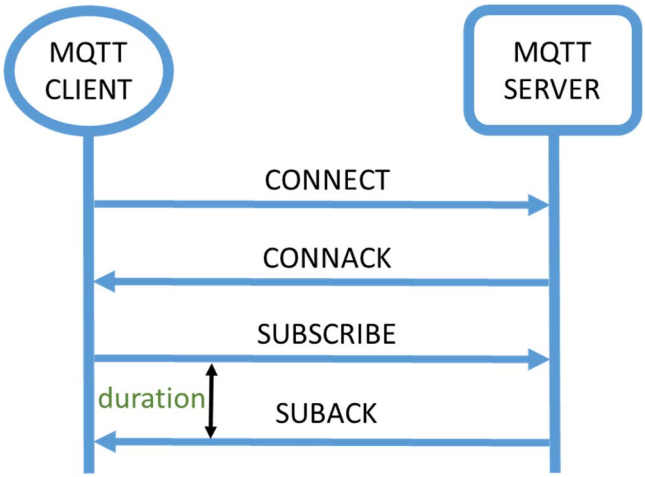


Figure 3: MQTT Server-Client SUBSCRIBE message flow

4) PUBLISH: This section describes the MQTT operation types and message sequences required for test execution using a Client PUBLISH example:

Preconditions:

- Connected Client, Server/Broker.
- TCP connection between Client and Server/Broker established.

Operation sequence QOS1:

- Client sends PUBLISH message.
- Client receives PUBACK message.

Measurement:

- Time period expressed in milliseconds between the moment client forwards the PUBLISH Message and the moment Client receives PUBACK message from server.

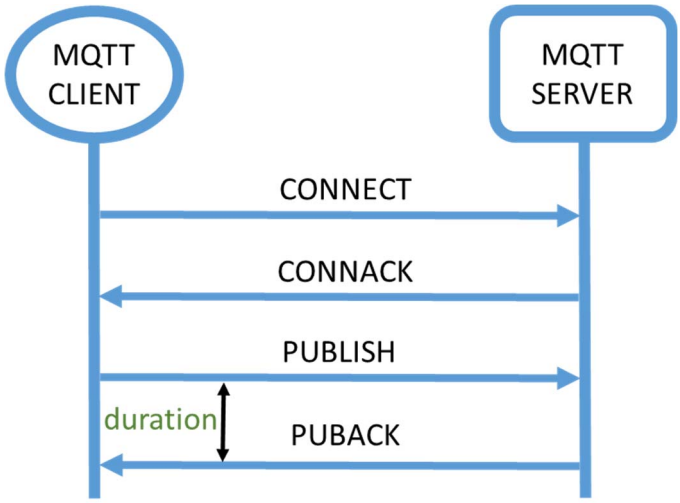


Figure 4: MQTT Server-Client QoS 1 PUBLISH message flow

Alternative operation sequence QOS2:

- Client sends PUBLISH message.
- Client receives PUBREC message.
- Client sends PUBREL message.
- Client receives PUBCOMP message.

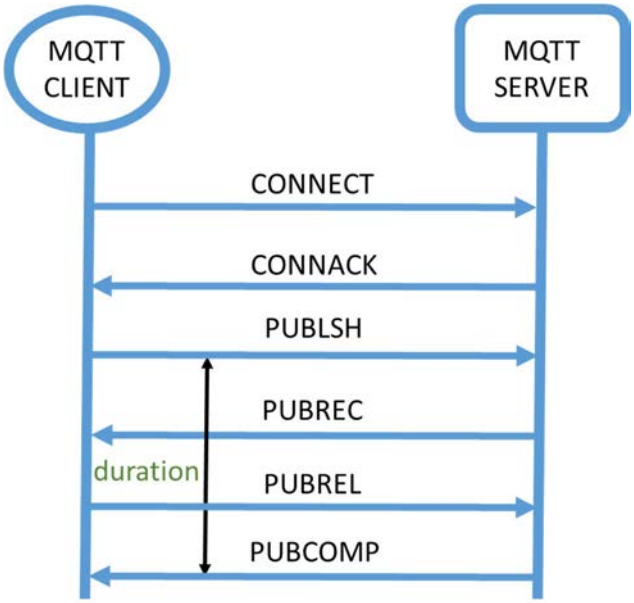


Figure 5: MQTT Server-Client QOS 2 PUBLISH message flow

Measurement:

- Time period expressed in milliseconds between the moment client forwards the PUBLISH Message and the moment Client receives PUBCOMP message from server.

4.2.5 Test Campaign Parameters

For test-campaign evaluations, a sequence of control packet exchanges are specified. This allows for defining multiple types of test benchmarks based on client-server interactions. This may address but not be limited to the following:

- duration;
- control packet sending order;
- number of control packets sent per session;
- number of application message packets sent.

The parameters also cover the type, sequence, characteristics and flow of messages sent towards the IUT:

- Type: message types from the ones listed in the previous tables.
- Sequence: a logical message flow sequence targeting the IUT behaviour (e.g. CONNECT, PINGREQ, PUBLISH, DISCONNECT).
- Characteristics: the specific message header flags and payload size (e.g. PUBLISH (AT_LEAST_ONCE_DELIVERY, 4Kb payload).
- Flow: the explicit message flow to be sent towards the IUT (e.g. 1xCONNECT, 1000x PUBLISH, 1xDISCONNECT).

The metrics listed in clauses 4.3 to 4.5 are derived from the ETSI TR 101 577 [i.2]. The measuring intervals (also referred in clause 4.2.3 as "monitoring windows") and metric validation thresholds for all metrics can vary depending on test requirements and are part of the test input parameters.

4.3 Powerfulness metrics

The powerfulness category of performance characteristics contains indicators of speed and quantity of service production. The following metrics are derived from ETSI TR 101 577 [i.2]:

- Capacity: ability to accept incoming service requests per time unit. For this metric set, the type of messages considered for evaluation are CONNECT, PUBLISH and PINGREQ. For each of the messages the input parameters specify number and type of requests per second and duration. The metrics consist of minimum, maximum and average response time, total number of successful calls, failed calls and pending calls.
- Responsiveness: time to handle a service request, e.g. subscription, connect request, etc.; transmission time, roundtrip time; publication (publish) time. Messages used are CONNECT, PUBLISH, PINGREQ. The metrics consist of minimum, maximum and average response time, total number of successful calls, failed calls and pending calls. The measuring interval and metric validation thresholds can vary depending on user requirements.
- Number of registered subscribers: maximum number active subscribers supported by the system). This metric is measured by connecting subscribers to a gateway in order to determine its maximum capacity. This is achieved by gradually connecting subscribers until no additional connections are accepted. The metric is measured by recording the maximum number of connected subscribers during a test.
- Number of connected servers: maximum number of servers supported by a client. This is achieved by connecting a gateway to an increasing number of gateways until no additional connections are accepted. The metric is measured by recording the maximum number of connected servers during a test.

4.4 Reliability metrics

The reliability category of performance characteristics contains indicators of how predictable a system's service production is. The performance category has subcategories for Quality-of-Service, Stability, Availability, Robustness, Recovery and Correctness:

- Stability: Stability characteristics are indicators of a system's ability to maintain measured performance figures for powerfulness and efficiency during service delivery regardless of time. This is a performance metric that is usually validated through endurance testing i.e. testing over longer periods of time.
- Availability: Availability characteristics are indicators of a system's ability to delivery services over time. Different availability characteristics are applied on hardware (physical availability) and on software (logical availability). This is a performance metric that is usually validated through endurance testing i.e. testing over longer periods of time.
- Robustness: Robustness characteristics are indicators of a system's services levels, i.e. services capacity and/or service responsiveness under extreme conditions. Extreme conditions can be caused internally by hardware failure or software malfunctioning, or externally by extreme peak load conditions, or by denial-of-service attacks or other malice attempts. Messages used are CONNECT, PUBLISH, PINGREQ. The metrics consist of minimum, maximum and average response time, total number of successful calls, failed calls and pending calls.
- Recovery: Recovery characteristics are indicators of production disturbances from hardware or software malfunctioning. Recovery covers a large number of different operations. In this context the system recovery and service recovery is considered. i.e. the time duration of a system to become operational after a reset.
- Correctness: Correctness characteristics are indicators of a system's ability to deliver correctly processed service requests under high or odd load conditions. Messages used are CONNECT, PUBLISH, PINGREQ. The metrics consist of minimum, maximum and average response time, total number of successful calls, failed calls and pending calls.

4.5 Efficiency metrics

The following metrics are derived from ETSI TR 101577 [i.2].

The performance category efficiency contains different types of indicators of resource usage and resource utilization. These metrics can be recorded additional to the Powerfulness metrics during the specified tests. The scope of these are to determine system deployment resource requirements as well as scaling capabilities:

- Resource usage: amount of CPU, Memory, Disk used by a server/client component during a test. Metric measurements are to be correlated with the other metrics, i.e. CPU usage during a 1 000 PUBLISH messages per second monitoring window, with a min, max and average load as output.
- Scalability: amount of resource usage increase through scaling horizontally (multiple client/server instances) or vertically (increase resource usage for CPUs/bandwidth capacity) relative to the load capacity increase. E.g. the CPU usage at 1 000 requests processed per time unit is 1 % and at 10 000 requests per processed time unit is 10 % would express a linear relation between the CPU usage and workload for the given parameters. This varies greatly with the test system characteristics and testing parameters, nevertheless it is very relevant for production services as an operational feature.

5 Configurations

The performance test configurations are derived from the SUT access points and functional test configurations.

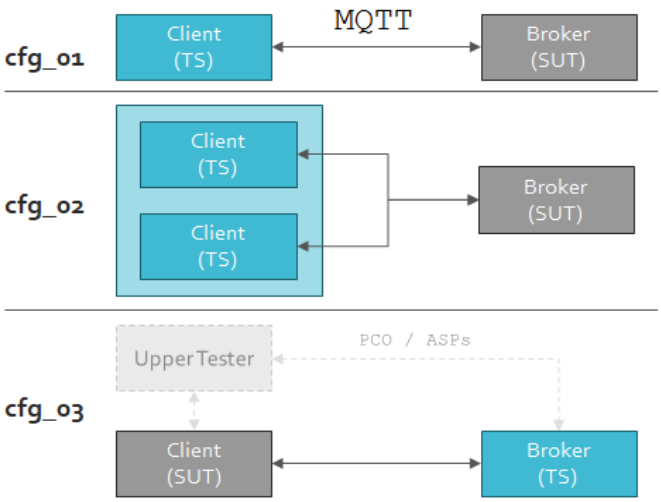
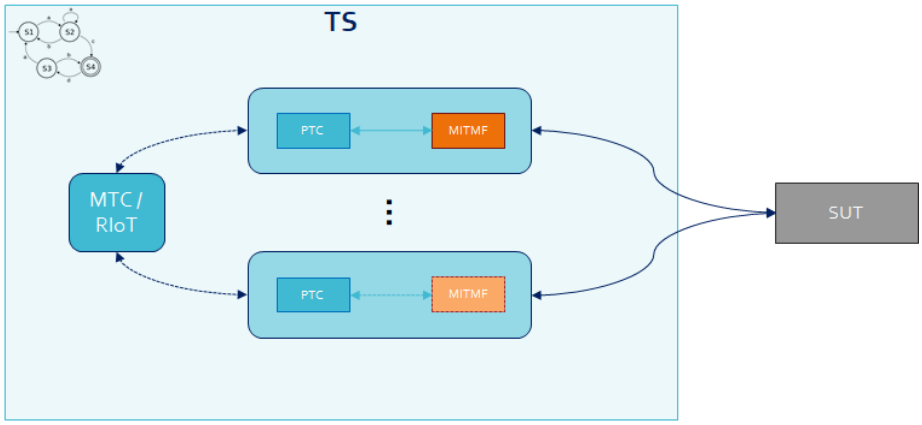


Figure 6: MQTT Server-Client test configurations

Furthermore, there are consideration following possible performance specific load and measurement tools.



MTC: Main Test Component
PTC: Parallel Test Component
MITMF: Man-in-the-Middle Fuzzer / Fuzzing Proxy
SUT: System Under Test

Figure 7: MQTT Server-Client test configuration

6 Benchmarking

6.1 Generic adjustments

This clause specifies the MQTT protocol performance metrics. As a protocol evaluation rationale, the metrics presented herein are directly addressed to IUTs: systems and associated components implementing the protocol, namely clients and servers. No requirements will be expressed regarding specifics of such IUTs in terms of implementation, technology, or proprietary elements. The metrics expressed herein are derived from the protocol specification and assume that both client and server components are conformant with the MQTT protocol specification. For conformance testing specifications, please refer to ETSI TS 103 597-1 [2].

6.2 Benchmarking Methodology

This clause aims to describe a viable methodology for benchmarking the performance of an IUT. In the case of MQTT, an IUT may consist of one or several Brokers or one or several Clients. The approach is inspired from the examples in IETF RFC 2544 [i.1] from the point of view of measurement preconditions, approach and statistically consistent measurement sampling.

As a general precondition for performance benchmarking, a functionally correct implementation is a prerequisite. For this the general assumption is that the IUT has passed the conformance testing described in ETSI TS 103 597-1 [2].

First, the System under Test is described, including hardware, resource manager (bare-metal/virtualization technology) and network connectivity (type-wired/ait, latency, throughput capacity). This includes both the resources dedicated to the IUT as well the ones for the test system.

Second, the type of performance benchmarking is established: whether the tests aim to determine the system KPI values or the tests aim to check whether the system meets established performance requirements. Depending on the objective, the approach will differ. In this step the KPIs and metrics w/o thresholds are selected. Two examples are presented in clause 6.4 reflecting the specific approach.

Third, the appropriate tests are selected and the test input parameters are specified. These include the test types, duration, metric threshold requirements, sample size and validation checks. Then, the monitoring system is configured, and the appropriate metrics (clauses 4.3 to 4.5) are selected for observation.

Fourth, the tests are executed and the metrics are collected. For this stage it is highly relevant that the TS and IUT are not affected by external factors in terms of compute and network resources. As an example, the monitoring system load on both the network and compute resources is commonly not taken into consideration and this leads to skewed results.

Finally, the test results are checked and validated leading to a verdict whether the performance tests are passed or failed.

6.3 Metric examples

This clause presents a series of benchmark metric examples:

- Connection-release delay: the time delay between DISCONNECT message and TCP connection closing. Value expressed in milliseconds (ms).
- Setup-delay: the time interval starts when a CONNECT message is sent and ends when the corresponding CONNACK message has been received back. Value expressed in milliseconds (ms).
- Publish delay: the time interval starts when a PUBLISH message is sent and ends when the corresponding PUBACK/PUBCOMP message has been received. Value expressed in milliseconds (ms).
- Subscription delay: the time between SUBSCRIBE and SUBACK message. Value expressed in milliseconds (ms).
- Unsubscription delay: the time between UNSUBSCRIBE and UNSUBACK message. Value expressed in milliseconds (ms).
- Ping delay: the time between PINGREQ and PINGRESP message. Value expressed in milliseconds (ms).

For each of the measurements enumerated above, the minimum, maximum and average values are also calculated according to clause 4.2.1 and reported to the specified monitoring windows.

6.4 Benchmark Examples

6.4.0 Introduction

For evaluating the metrics described in clause 4, the benchmark tests can be grouped in 3 main categories:

1) Load Tests: These tests are used for determining or validating the IUT workload range. The workload consists of one or multiple message exchanges between the IUT and the Test System. The aim is to observe the Powerfulness and Efficiency metrics as well as the Correctness (Reliability category) metric in order to determine or validate the maximum operating workload handled by the IUT.

2) Endurance Tests: These tests are used for determining or validating the IUT Reliability and Efficiency. These tests generally consist of exposing the IUT to a variable or high operational workload for long periods of time. The metrics observed are the Reliability and Efficiency ones. This type of testing covers operational aspects such as degradation over time, memory leaks and resource consumption estimates.

3) Stress Tests: These tests are used for determining or validating the IUT Robustness and Recovery (Reliability category) metrics. This is achieved by injecting workload spikes throughout the test and observing the degradation and recovery patterns of the system as well as the maximum workload operational limits.

The following two benchmarking examples reflect the two types of performance benchmarking evaluations specified in clause 7.2.

6.4.1 KPI Determination

The KPI determination benchmark is an exploratory performance evaluation that aims to determine the operational performance of an IUT. The KPIs are specified as an input. The scope of this evaluation is to establish a reliable indication of how the SUT is expected to perform in production. The KPIs are determined according to the intended use-case scenario for the IUT.

As a first example, a device manufacturer has finished a hardware MQTT broker prototype for the industrial IoT market. The target objective is to provide communication in small industrial buildings serving a potential capacity of 50 to 5 000 MQTT clients. The expected use-case requires QoS1 for data transmission and foresees a 1 000 to 10 000 published messages per second load. Additionally, the manufacturer wants to determine the system reliability, specifically Stability, Availability and Correctness.

According to clause 7.2, the first step is to specify the SUT hardware and network resources. In this example the underlying hardware to be a bare-metal SoC box running Ubuntu 18.04 OS is considered. It has a dual core 2,4 GHz CPU, 2 Gb or RAM, 120 GB SSD and a 1 Gb Ethernet connection. For simplification, the network is considered wired, with a 1 Gbps throughput running TCP/IP over a 1000BASE-T Ethernet LAN connection with an estimated 1ms end-to-end delay for all connections. The Test System (TS) consists of a quad-core power pc with 2,4 GHz CPU, 8 Gb of RAM, 500 GB SSD and 1 Gb Ethernet connection. The TS is deployed in virtual containers running over a Unix environment with direct access to the network (non-virtualised network connection). The monitoring system resource consumption is considered negligible.

The second step is to select the KPIs and metrics of interest. The KPIs selected by the manufacturer are Capacity(max number of publish requests handled per second), Responsiveness (average delay of processing client requests), Number of registered subscribers, resource usage and stability. The Broker operates with QoS 1. The selected associated metrics are as follows:

- Publish delay: the time interval starts when a PUBLISH message is sent and ends when the corresponding PUBACK (QoS1) message has been received. Value expressed in milliseconds (ms).
- Subscriber Count: the maximum number of registered subscribers within a measurement window.
- Capacity: number of successfully handled Publish messages per second. Threshold is initially set to 1 000.
- Correctness: percentage of successfully handled Publish messages per second.
- Resource usage: amount of CPU, Memory used by the IUT during the Capacity evaluation. with a min, max and average values.

The third step consists of determining the tests and configuring the monitoring system. According to the KPI requirements, the type of tests required are load testing for determining the IUT Capacity and Resource usage and Endurance testing for determining the system Responsiveness, Correctness and Availability. The monitoring system is configured to record number of subscribers, PUBLISH delay, rate of success for PUBLISH messages, CPU, Mem and Network in/out usage. The monitoring window is set to 1 second and post-processing averages are configured to cover 1 minute for load tests and 1 hour for endurance tests.

Load Testing

EXAMPLE 1:

1 000 Clients connect to the IUT and subscribe to topics.

Each client starts sending Publish messages at a rate of 1 message per second. The rate increases by 1 up to a maximum of 10 every minute. The test duration is set to 10 minutes and executed 10 times. The test is repeated for an incremented number of connected clients up to 5 000 in incremental steps of 1 000 clients. The input parameters are no longer incremented if the test fails. Test duration: 10 minutes per test:

a) Metric: number of connected clients.

b) Metric: number of PUBLISH messages processed per second.

c) Metric: average PUBLISH messages processing delay.

d) Metric: rate of successfully processed PUBLISH messages.

e) Metric: CPU load user time %.

f) Metric: Memory load (Mb).

g) Metric: Network Packet In (Kb).

h) Metric: Network Packet Out (Kb).

Fail Criteria:

1) Rate of successfully processed PUBLISH messages falls under 90 %.

2) CPU load goes over 80 %.

3) Memory load goes over 90 %.

Pass criteria: Test ended without fail criteria triggered.

Endurance Test

Starting from the highest load successfully passed, the endurance test parameters are noted with max(X) where X is the metric value from the Load test.

EXAMPLE 2:

max Clients connect to the IUT and subscribe to topics.

Each client starts sending Publish messages at a rate of max message per second. The rate remains constant. The test duration is set to 600 minutes and executed 10 times. The test is repeated for an decremented number of connected clients down to 1 000 in decremented steps of 1 000 clients in case of failure. The input parameters are no longer decremented if the test succeeds. Test duration: 600 minutes per test:

a) Metric: number of connected clients.

b) Metric: number of PUBLISH messages processed per second.

c) Metric: average PUBLISH messages processing delay.

d) Metric: rate of successfully processed PUBLISH messages.

e) Metric: CPU load user time %.

f) Metric: Memory load (Mb).

g) Metric: Network Packet In (Kb).

h) Metric: Network Packet Out (Kb).

Fail Criteria:

1) Rate of successfully processed PUBLISH messages falls under 90 %.

2) CPU load goes over 80 %.

3) Memory load goes over 90 %.

Pass criteria: Test ended without fail criteria triggered.

The fourth step consists of executing the tests. As a general precondition: the SUT is operational - IUT MQTT broker is active. TS is operational and connected to the SUT. Finally the results are collected the Powerfulness, Reliability and Efficiency selected KPI values are determined.

6.4.2 KPI Validation

The KPI validation benchmark is performance evaluation that aims to validate whether the IUT performs according to requirement specifications. The KPIs and their thresholds are specified as an input. The scope of this evaluation is to establish a reliable indication of how the SUT is expected to perform in production. The KPIs are determined according to the intended use-case scenario for the IUT.

In this second example, a factory owner contacts the previous device manufacturer and specifies its requirements. These consist of providing communication in small industrial buildings serving a total of 1 000 MQTT clients. The expected use-case requires QoS1 for data transmission and a variable load of 1 000 to 5 000 published messages per second load. Additionally, the factory owner requires that the publish delay is no longer than 1 second and that 99 % of the messages are successfully transmitted.

According to clause 7.2, the first step is to specify the SUT hardware and network resources. In this example the underlying hardware is considered to be a bare-metal SoC box running Ubuntu 18.04 OS. It has a dual core 2,4 GHz CPU, 2 Gb or RAM, 120 GB SSD and a 1 Gb Ethernet connection. For simplification, the network is considered wired, with a 1 Gbps throughput running TCP/IP over a 1000BASE-T Ethernet LAN connection with an estimated 1ms end-to-end delay for all connections. The Test System (TS) consists of a quad-core power pc with 2,4 GHz CPU, 8Gb of RAM, 500 GB SSD and 1 Gb Ethernet connection. The TS is deployed in virtual containers running over a Unix environment with direct access to the network (non-virtualised network connection). The monitoring system resource consumption is considered negligible.

The second step is to select the KPIs and metrics of interest. The KPIs selected by the manufacturer are Capacity(max number of publish requests handled per second), Responsiveness (average delay of processing client requests), Number of registered subscribers, resource usage and stability. The Broker operates with QoS 1. The selected associated metrics are as follows:

- Publish delay: the time interval starts when a PUBLISH message is sent and ends when the corresponding PUBACK (QoS1) message has been received. Value expressed in milliseconds (ms).Threshold set to 1 000 ms.
- Subscriber Count: the maximum number of registered subscribers within a measurement window. Threshold set to 1 000.
- Capacity: number of successfully handled Publish messages per second. Threshold is set to 5 000.
- Correctness: percentage of successfully handled Publish messages per second. Threshold is set to 99 %.
- Resource usage - amount of CPU, Memory used by the IUT during the Capacity evaluation. with a min, max and average values. Threshold is set to 80 % CPU load, 90 % Mem load.

The third step consists of determining the tests and configuring the monitoring system. According to the KPI requirements, the type of tests required are load testing for validating the IUT Capacity and Resource usage and Endurance testing for determining the system Responsiveness, Correctness and Availability. The monitoring system is configured to record number of subscribers, PUBLISH delay, rate of success for PUBLISH messages, CPU, Mem and Network in/out usage. The monitoring window is set to 1 second and post-processing averages are configured to cover 1 minute for load tests and 1 hour for endurance tests.

Load Testing

EXAMPLE:

1 000 Clients connect to the IUT and subscribe to topics.

Each client starts sending Publish messages at a rate of 1 message per second. The rate increases by 1 up to a maximum of 5 every 2 minutes. The test duration is set to 10 minutes and executed 10 times. Test duration: 10 minutes per test:

a) Metric: number of connected clients.

a) Metric: number of PUBLISH messages processed per second.

b) Metric: average PUBLISH messages processing delay.

c) Metric: rate of successfully processed PUBLISH messages.

d) Metric: CPU load user time %.

e) Metric: Memory load (Mb).

f) Metric: Network Packet In (Kb).

g) Metric: Network Packet Out (Kb).

Fail criteria:

1) Rate of successfully processed PUBLISH messages falls under 99 %.

2) CPU load goes over 80 %.

3) PUBLISH delay max value is greater than 1 000 ms.

4) Memory load goes over 90 %.

Pass criteria: Test ended without fail criteria triggered.

Endurance Test

1 000 Clients connect to the IUT and subscribe to topics.

Each client starts sending Publish messages at a rate of 5 messages per second. The rate remains constant. The test duration is set to 600 minutes and executed 10 times. Test duration: 600 minutes per test:

a) Metric: number of connected clients.

b) Metric: number of PUBLISH messages processed per second.

c) Metric: average PUBLISH messages processing delay.

d) Metric: rate of successfully processed PUBLISH messages.

e) Metric: CPU load user time %.

f) Metric: Memory load (Mb).

g) Metric: Network Packet In (Kb).

h) Metric: Network Packet Out (Kb).

Fail criteria:

1) Rate of successfully processed PUBLISH messages falls under 99 %.
2) CPU load goes over 80 %.
3) Memory load goes over 90 %.
4) PUBLISH delay max value is greater than 1 000 ms.

Pass criteria: Test ended without fail criteria triggered.

The fourth step consists of executing the tests. As a general precondition: the SUT is operational - IUT MQTT broker is active. TS is operational and connected to the SUT. Finally the results are collected the Powerfulness, Reliability and Efficiency selected KPI values are determined.

7 Examples of Tests

7.0 Introduction

A test should be presentable as a document, with accompanying data files, that provides a full description of an execution of a performance test on a test system. Description of the test case and objective of the test case, e.g. the definition of the targeted metrics should be contained therein. The following sections should be addressed in general:

- Test procedure: Description of the execution of the test:
  - Test sequence to run the test case: sequence of actions for running the experiment and collect the measurements needed to compute the metrics.
  - Test duration (per iteration).
  - Number of iterations of the experiment. Number of repetitions of the experiments to obtain relevant statistical results.
  - Measurements collected to compute the metrics.
- Procedure for metrics calculation Description of the procedure applied (statistical aggregation, algorithm, etc.) to compute the metrics based on the raw measurements collected.
- Expected output of the test case:
  - Test report.

A set of Test Purposes (TPs) are provided as reference examples in Annex A.

7.1 Test Objectives

In this clause example 2 formalized test objective examples are presented. A set of test purposes are presented in Annex A.

EXAMPLE 1: Broker Publish Load Test:

- Determine if the IUT (broker) can handle the given publish message load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.

EXAMPLE 2: Broker PUB/SUB Load Test:

- Determine if the IUT (broker) can handle the given publish message load and forward them to a given number of subscribers for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.

7.2 Test Purpose

In this clause a formalized test purpose example is provided. A set of test purposes are presented in Annex A.

Precondition:

- The IUT is running and configured to accept client connections.
- Client(s) send CONNECT messages.
- Client(s) start sending PUBLISH messages at a specified rate (messages per second) totalling a specified duration (time) or number of messages (e.g. 10 000 messages).
- After finishing the execution, the client(s) send DISCONNECT messages.

Expected behaviour:

- The IUT will reach the given time interval of handling the given load without exceeding the delay/message loss threshold.
- The IUT will not reach the given time interval of handling the given load before exceeding the delay/message loss threshold.

Expected output:

- Minimal, maximal and average measurements of selected metrics.

7.3 Test Report

The test report will include the test execution time, test parameters (e.g. number of messages, rate) and measurement/aggregated measurement results (e.g. the average PUBLISH/PUBACK message sequence latency was 3 ms end-to-end).

Table 4: Performance load test report for PUB/ACK example

Test Number	T-01	**Test Category**	Performance	**Test Type**	Load Testing
Test Objective	"Determine if the IUT(broker) can handle the given incremental load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate."				
Test Description	Test Scenario 1 Test Case 1 Configuration 1: Against Mosca Server	Test Scenario 1 Test Case 1 Configuration 2: Against Mosquitto Server	Reference 2 ms		
Preconditions	the CLIENT having a MQTT_CONNECTION to the IUT				

Expected Behaviour	ensure that {
	when {
	(.) at time point t1: the tester entity send multiple PUBLISH messages and assure the RATE and
	(!) during the INTERVAL after t1: the IUT entity receive multiple PUBLISH message containing
	topic_name corresponding to TOPIC,
	payload corresponding to RETAINED_MESSAGE;
	}
	then {
	(!) INTERVAL after t1:
	the IUT entity assure and send the PUBACK messages and
	the IUT entity assure the packet_loss_limit and
	the IUT entity assure the DELAY;
	}				

Output	Average PUBLISH/PUBACK delay in ms (KPIx),				
Measurements	Publish Success Rate	Sequence Delay	Reference		
Values C1	100 %	0,998 ms	2 ms		
Values C2	100 %	0,93 ms	2 ms		

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

These TPs has been produced using the TDL extension TDL-TO according to [3]. They serve as informative references to performance test examples that are structured and numbered according to [3], clause 4.

Table A.1

TP Id	TP_MQTT_Performance_Broker_Load_001
Test Objective	MQTT CONNECT load test for broker: Determine if the IUT(broker) can handle the given load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the TEST_SYSTEM having a TCP_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple CONNECT messages and assure the INCREMENTAL_RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple CONNECT message containing payload containing
client_identifier corresponding to PX_CLIENT_ID,
user_name corresponding to PX_MQTT_USER_NAME,
password corresponding to PX_MQTT_PASSWORD;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the CONNACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.2

TP Id	TP_MQTT_Performance_Broker_Load_002
Test Objective	MQTT PING load test for broker: Determine if the IUT(broker) can handle the given load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PINGREQ messages and assure the INCREMENTAL_RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple PINGREQ message containing
header_flags indicating value '0000'B;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the PINGRESP messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.3

TP Id	TP_MQTT_Performance_Broker_Load_003
Test Objective	MQTT PUBLISH load test for broker: Determine if the IUT(broker) can handle the given load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PUBLISH messages and assure the INCREMENTAL_RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple PUBLISH message containing
topic_name corresponding to TOPIC,
payload corresponding to RETAINED_MESSAGE;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the PUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.4

TP Id	TP_MQTT_Performance_Broker_Load_004
Test Objective	MQTT SUBSCRIBE load test for broker: Determine if the IUT(broker) can handle the given load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple SUBSCRIBE messages and assure the INCREMENTAL_RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple 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 {
(!) INTERVAL after t1:
the IUT assure and send the SUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.5

TP Id	TP_MQTT_Performance_Broker_Endurance_001
Test Objective	MQTT CONNECT endurance test for broker: Determine if the IUT(broker) can handle the given incremental load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the TEST_SYSTEM having a TCP_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple CONNECT messages and assure the RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple CONNECT message containing
payload containing
client_identifier corresponding to PX_CLIENT_ID,
user_name corresponding to PX_MQTT_USER_NAME,
password corresponding to PX_MQTT_PASSWORD;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the CONNACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.6

TP Id	TP_MQTT_Performance_Broker_Endurance_002
Test Objective	MQTT PING endurance test for broker: Determine if the IUT(broker) can handle the given incremental load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PINGREQ messages and assure the RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple PINGREQ message containing
header_flags indicating value '0000'B;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the PINGRESP messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.7

TP Id	TP_MQTT_Performance_Broker_Endurance_003
Test Objective	MQTT PUBLISH endurance test for broker: Determine if the IUT(broker) can handle the given incremental load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PUBLISH messages and assure the RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple PUBLISH message containing
topic_name corresponding to TOPIC,
payload corresponding to RETAINED_MESSAGE;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the PUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.8

TP Id	TP_MQTT_Performance_Broker_Endurance_004
Test Objective	MQTT SUBSCRIBE endurance test for broker: Determine if the IUT(broker) can handle the given incremental load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple SUBSCRIBE messages and assure the RATE and
(!) during the INTERVAL after t1: the IUT receive multiple 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 {
(!) INTERVAL after t1:
the IUT assure and send the SUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.9

TP Id	TP_MQTT_Performance_Broker_Stress_001
Test Objective	MQTT PUBLISH stress test for broker: Determine if the IUT(broker) can handle the given spiking load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the IUT reach an initial_state
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PUBLISH messages and assure the RATE and
(!) during INTERVAL after t1:
the IUT receive several PUBLISH messages containing
topic_name corresponding to TOPIC,
payload corresponding to RETAINED_MESSAGE; and
(.) at time point t2: the tester send multiple PUBLISH messages and assure the SPIKE_RATE and
(!) during INTERVAL of t2:
the IUT receive several PUBLISH messages containing
topic_name corresponding to TOPIC,
payload corresponding to RETAINED_MESSAGE;
}
then {
(!) INTERVAL after t2:
the IUT assure and send the PUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.10

TP Id	TP_MQTT_Performance_Broker_Stress_002
Test Objective	MQTT PING stress test for broker: Determine if the IUT(broker) can handle the given spiking load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PINGREQ messages and assure the RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple PINGREQ message containing
header_flags indicating value '0000'B; and
(.) at time point t2:
the tester send multiple PINGREQ messages and assure the SPIKE_RATE and
(!) during the INTERVAL after t2:
the IUT receive multiple PINGREQ message containing
header_flags indicating value '0000'B;
}
then {
(!) INTERVAL after t1:
the IUT assure and send the PINGRESP messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.11

TP Id	TP_MQTT_Performance_Broker_Stress_003
Test Objective	MQTT PUBLISH stress test for broker: Determine if the IUT(broker) can handle the given spiking load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple PUBLISH messages and assure the RATE and
(!) during the INTERVAL after t1:
the IUT receive multiple PUBLISH message containing
topic_name corresponding to TOPIC,
payload corresponding to RETAINED_MESSAGE; and
(.) at time point t2:
the tester send multiple PUBLISH messages and assure the SPIKE_RATE and
(!) during the INTERVAL after t1: the IUT receive multiple PUBLISH message containing
topic_name corresponding to TOPIC,
payload corresponding to RETAINED_MESSAGE;
}
then {
(!) INTERVAL after t2:
the IUT assure and send the PUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions	

Table A.12

TP Id	TP_MQTT_Performance_Broker_Stress_004
Test Objective	MQTT SUBSCRIBE stress test for broker: Determine if the IUT(broker) can handle the given spiking load for a determined period of time without exceeding the delay threshold within a given acceptable message loss rate.
Reference	[MQTT-3.1.2-9], [MQTT-3.1.4-1], [MQTT-3.2.2-6]
PICS Selection	PICS_BROKER_BASIC and PICS_BROKER_PERFORMANCE and PICS_CLIENT_BASIC
Initial Conditions	

with {
the CLIENT having a MQTT_CONNECTION to the IUT
}	

Expected Behaviour	

ensure that {
when {
(.) at time point t1:
the tester send multiple SUBSCRIBE messages and assure the RATE and
(!) during the INTERVAL after t1: the IUT receive multiple 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; and
(.) at time point t2: the tester send multiple SUBSCRIBE messages and assure the SPIKE_RATE and
(!) during the INTERVAL after t1: the IUT receive multiple 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 {
(!) INTERVAL after t2:
the IUT assure and send the SUBACK messages and
the IUT assure the packet_loss_limit and
the IUT assure the DELAY
}
}	

Final Conditions

This Test purpose catalogue has been produced using the Test Description Language (TDL-TO) according to ETSI ES 203 119-4 [3]. The TDL-TO library modules corresponding to the Test purpose catalogue are contained in archive ts_10359703v010101p0.zip which accompanies the present document.

History

Document history		
V1.1.1	January 2021	Publication