Tuan Hiep/ERODS/LIG

MQTT plugin for Jmeter

I. Introduction

The MQTT Plugin in Jmeter is used for the injection testing of MQTT server. It permits the complete test correspond many scenarios, which depend on type of messages, type of connections. Thanks to it's interface graphic, the fact of testing mqtt protocol is taken easily.

II. How to install MQTT plugin in Jmeter

From the repository: https://github.com/tuanhiep/mqtt-jmeter
Get the source code, go to mqtt-jemeter folder and and use the command maven in terminal (Ubuntu):

mvn clean install package

to obtain the file ***mqtt-jmeter.jar*** in *mqtt-jemeter/target*

Put the ***mqtt-jemeter.jar*** in the folder lib/ext of Jmeter (which is download from http://jmeter.apache.org/download_jmeter.cgi).

Remind that, it's necessary to update the file ***ApacheJMeter_core.jar*** in the repository lib/ext of Jmeter.

Update the file messages.properties in the folder :/org/apache/jmeter/resources/

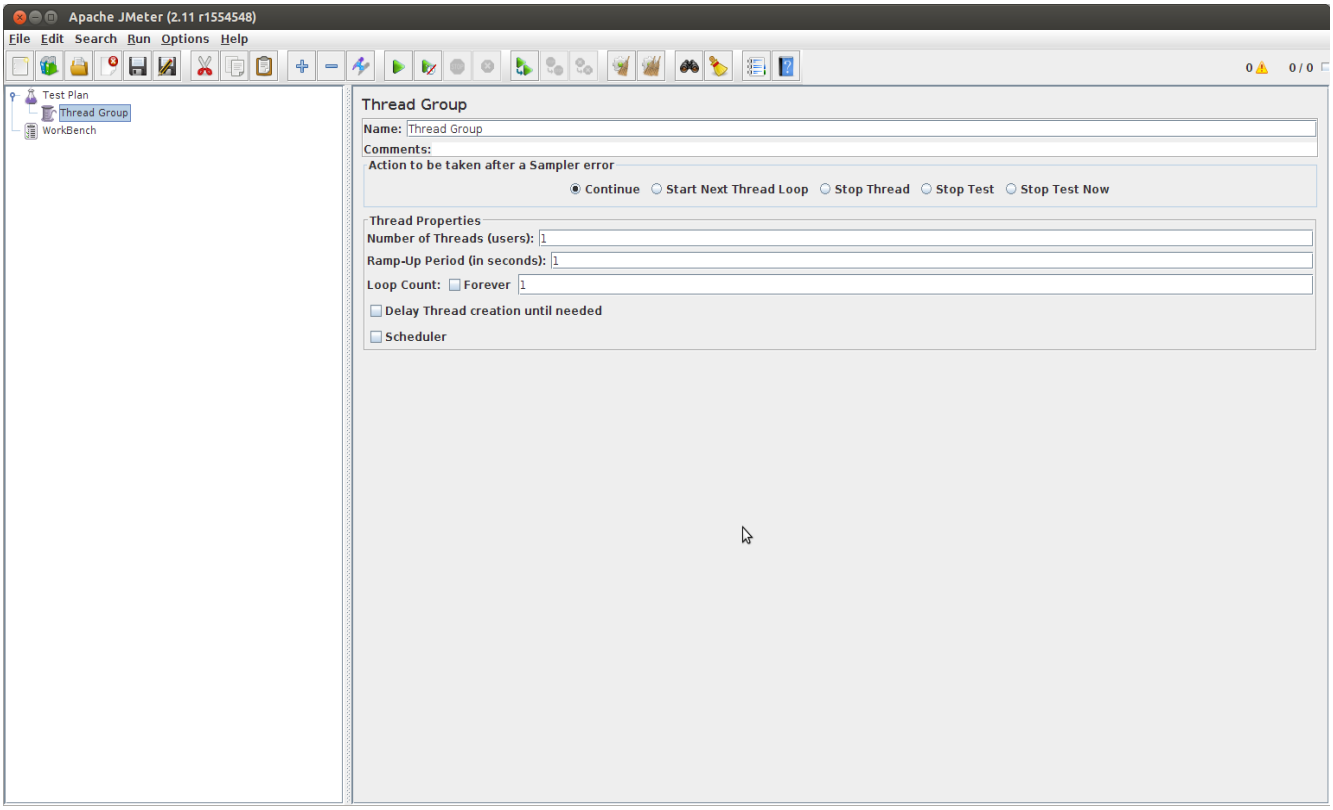
in ***ApacheJMeter_core.jar*** by new file messages.properties from

https://github.com/tuanhiep/mqtt-jmeter/tree/master/ressource

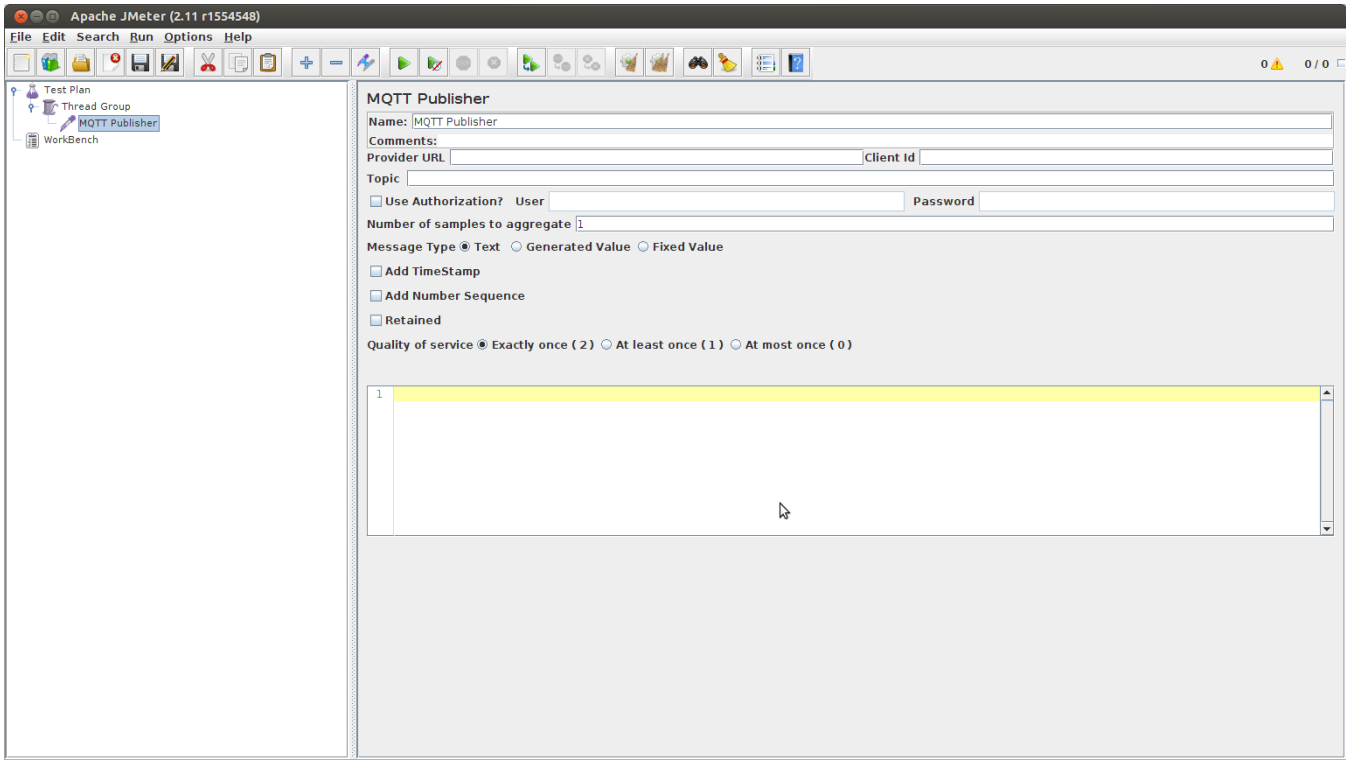
III. How to use MQTT plugin in Jmeter

1. MQTT Publisher

The interface graphic of Jmeter :

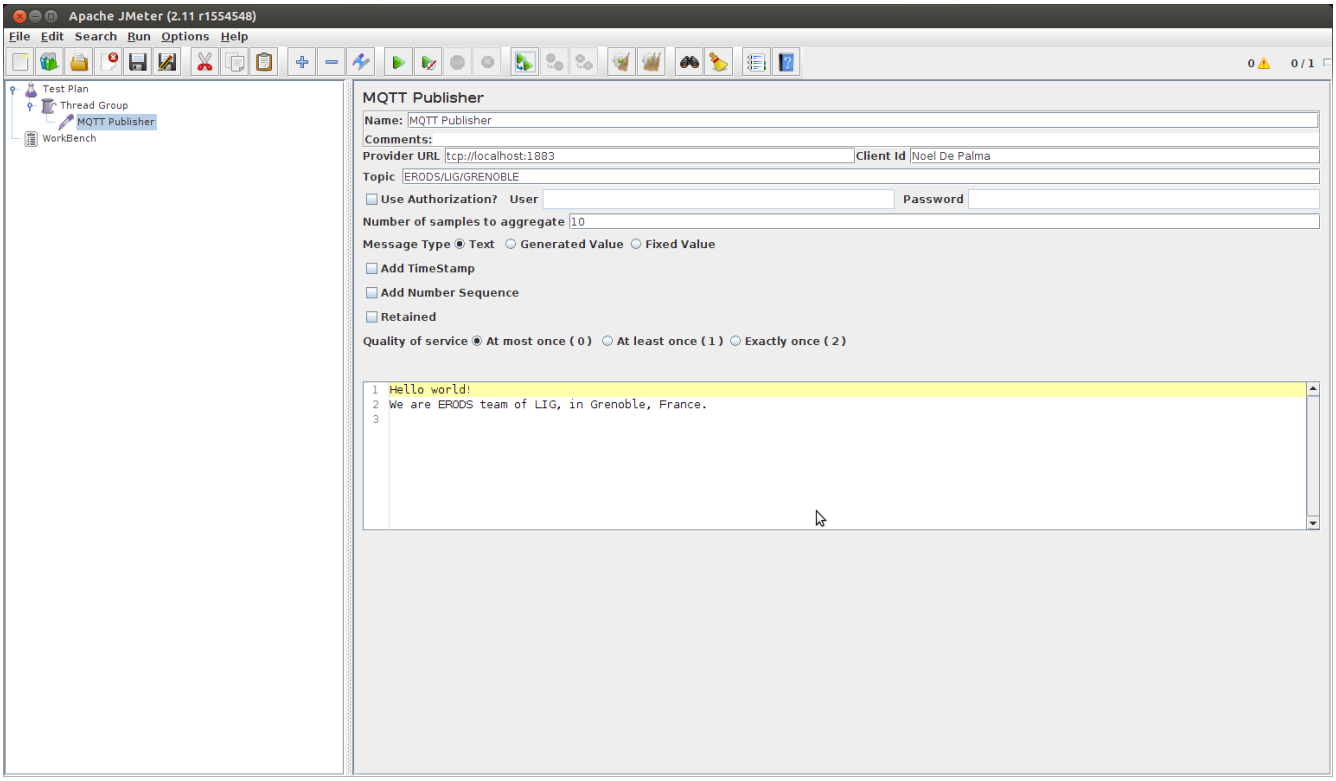


Right-click “Thread” and choose : Add → Sampler → MQTT Publisher



In the principal interface of MQTT Publisher we have the fields:

- *Name :* Name of the MQTT Publisher
- *Comments:* Your comments
- *Provider URL:* the address of MQTT server *example*: tcp://localhost:1883
- *Client Id:* Your Id in the session with MQTT server *example: Noel De Palma*
- *Topic:* The topic's name you want to publish
- *Use Authorization* check box : Necessary in the case the connection needs the username and password
- *User:* Your username
- *Password:* Your password
- *Number of samples to aggregate :* In other way, the number of messages you want to publish to the MQTT sever in this MQTT Publisher thread, with the value like the configuration below.
- *Message Type:* You can choose : Text, Generated Value, Fixed Value (more detail below)



- *Add TimeStamp* check box : Add the timestamps to the message. The timestamps is 8 bytes
- *Add Number Sequence* check box: Add the number sequence to the message. Example: if you publish 100 messages in your session, the message is numbered from 0 to 99. The number sequence field in the message is 4 bytes.
- *Retained* check box: You publish the messages as retained messages or not. The retain flag for an MQTT message is set to false by default. This means that a broker will not hold onto the message so that any subscribers arriving after the message was sent will not see the message. By setting the retain flag, the message is held onto by the broker, so when the late arrivers connect to the broker or clients create a new subscription they get all the relevant retained messages"
- *Quality of service*: Three levels:

  0 : At most once
  1 : At least once
  2 : Exactly once

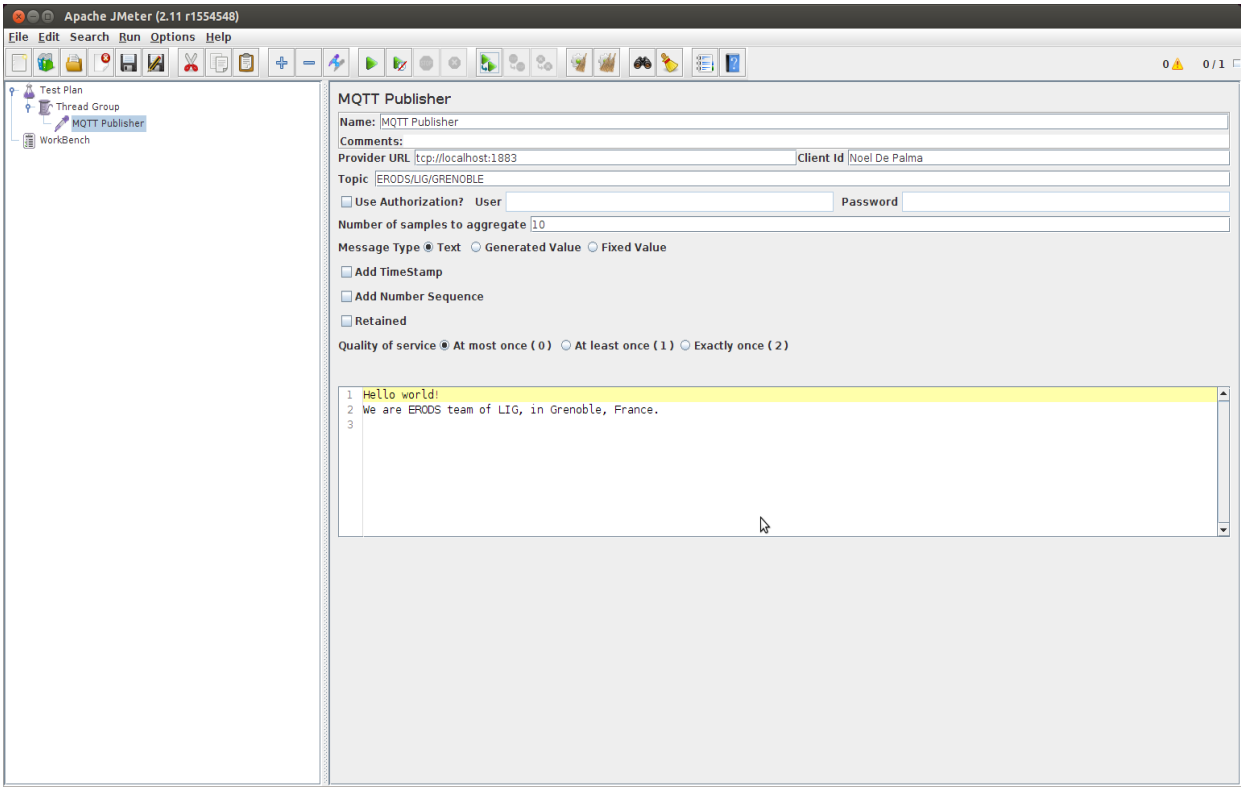
Each message in MQTT can have its quality of service and retain flag set. The quality of service advises the code if and how it should ensure the message arrives. There are three options, 0 (At Most Once),1 (At Least Once) and 2 (Exactly Once). By default, a new message instance is set to "At Least Once", a Quality of Service (QoS) of 1, which means the sender will deliver the message at

least once and, if there's no acknowledgement of it, it will keep sending it with a duplicate flag set until an acknowledgement turns up, at which point the client removes the message from its persisted set of messages.

A QoS of 0, "At Most Once", is the fastest mode, where the client doesn't wait for an acknowledgement. This means, of course, that if there's a disconnection or server failure, a message may be lost. At the other end of the scale is a QoS of 2, "Exactly Once", which uses two pairs of exchanges, first to transfer the message and then to ensure only one copy has been received and is being processed. This does make Exactly Once the slower but most reliable QoS setting.

With MQTT Publisher in Jmeter, three type of messages can be sent (Message Type):

- **Text:** The text message, without flag header and the server MQTT can deliver it like a normal text.



```
strongman@strongman:~$ mosquitto_sub -h localhost -d -t ERODS/LIG/GRENOBLE
Received CONNACK
Received SUBACK
Subscribed (mid: 1): 0
Received PUBLISH (d0, q0, r0, m0, 'ERODS/LIG/GRENOBLE', ... (60 bytes))
Hello world!
We are ERODS team of LIG, in Grenoble, France.

Received PUBLISH (d0, q0, r0, m0, 'ERODS/LIG/GRENOBLE', ... (60 bytes))
Hello world!
We are ERODS team of LIG, in Grenoble, France.

Received PUBLISH (d0, q0, r0, m0, 'ERODS/LIG/GRENOBLE', ... (60 bytes))
Hello world!
We are ERODS team of LIG, in Grenoble, France.

Received PUBLISH (d0, q0, r0, m0, 'ERODS/LIG/GRENOBLE', ... (60 bytes))
Hello world!
We are ERODS team of LIG, in Grenoble, France.

Received PUBLISH (d0, q0, r0, m0, 'ERODS/LIG/GRENOBLE', ... (60 bytes))
Hello world!
We are ERODS team of LIG, in Grenoble, France.

Received PUBLISH (d0, q0, r0, m0, 'ERODS/LIG/GRENOBLE', ... (60 bytes))
Hello world!
We are ERODS team of LIG, in Grenoble, France.
```

1 byte “flag header” for the messages of type: **Generated value, Fixed value**

TimeStamp	Number sequence	INT flag	LONG flag	FLOAT flag	DOUBLE flag	STRING flag	Padding

Flag header

In the flag header, if one field is set to 1, it means, we use the header in the message.

For example: With this flag header

1	1	0	0	0	1	0	0

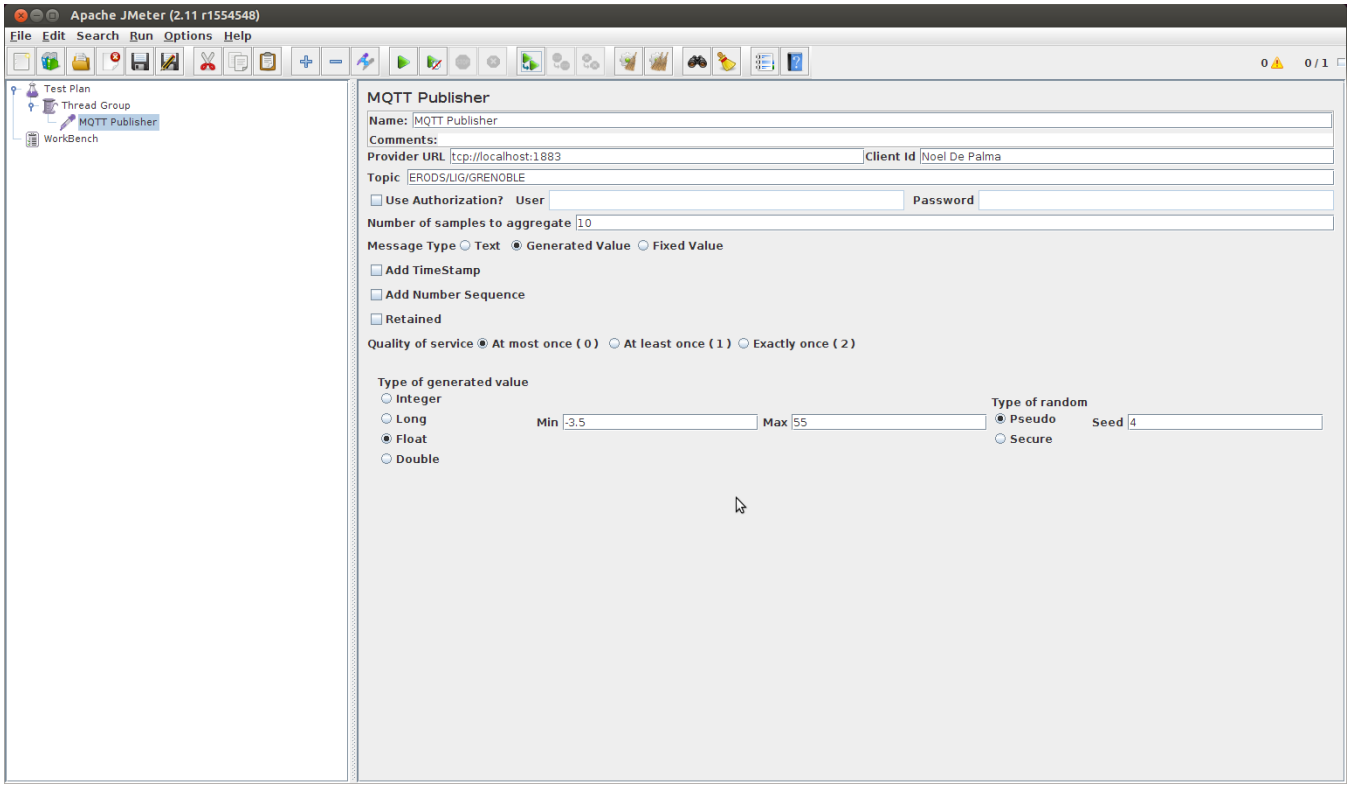
It means that, in the message, we have :

Flag Header 1 byte	TimeStamp Header 8 bytes	Number Sequence Header 4 bytes	Value (a double) 8 bytes

Message

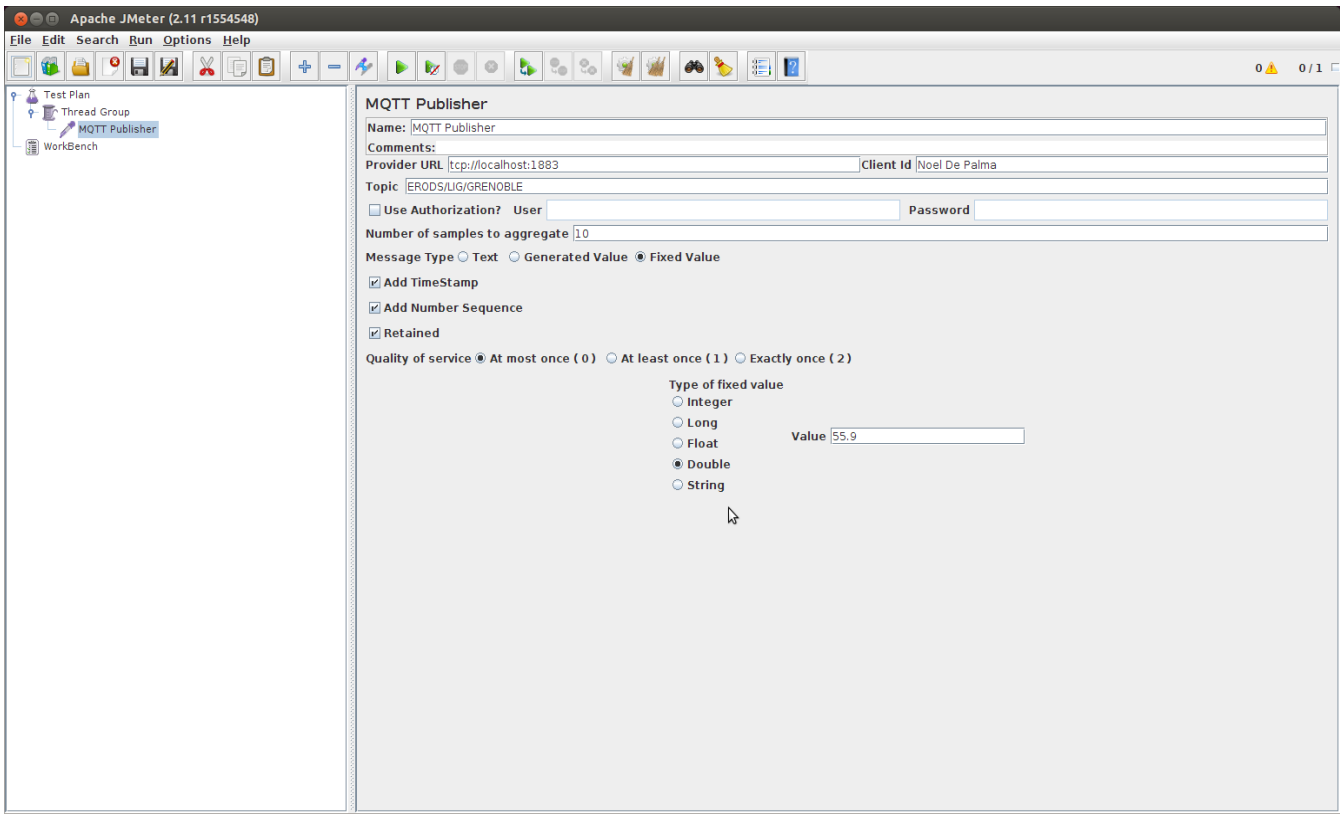
- **Generated Value :**

The generated value can be of type: Integer, Long, Float, Double within the range [Min,Max] . The type of random can be: Pseudo random or Secure random. In the two cases, we can set the Seed for the generator.

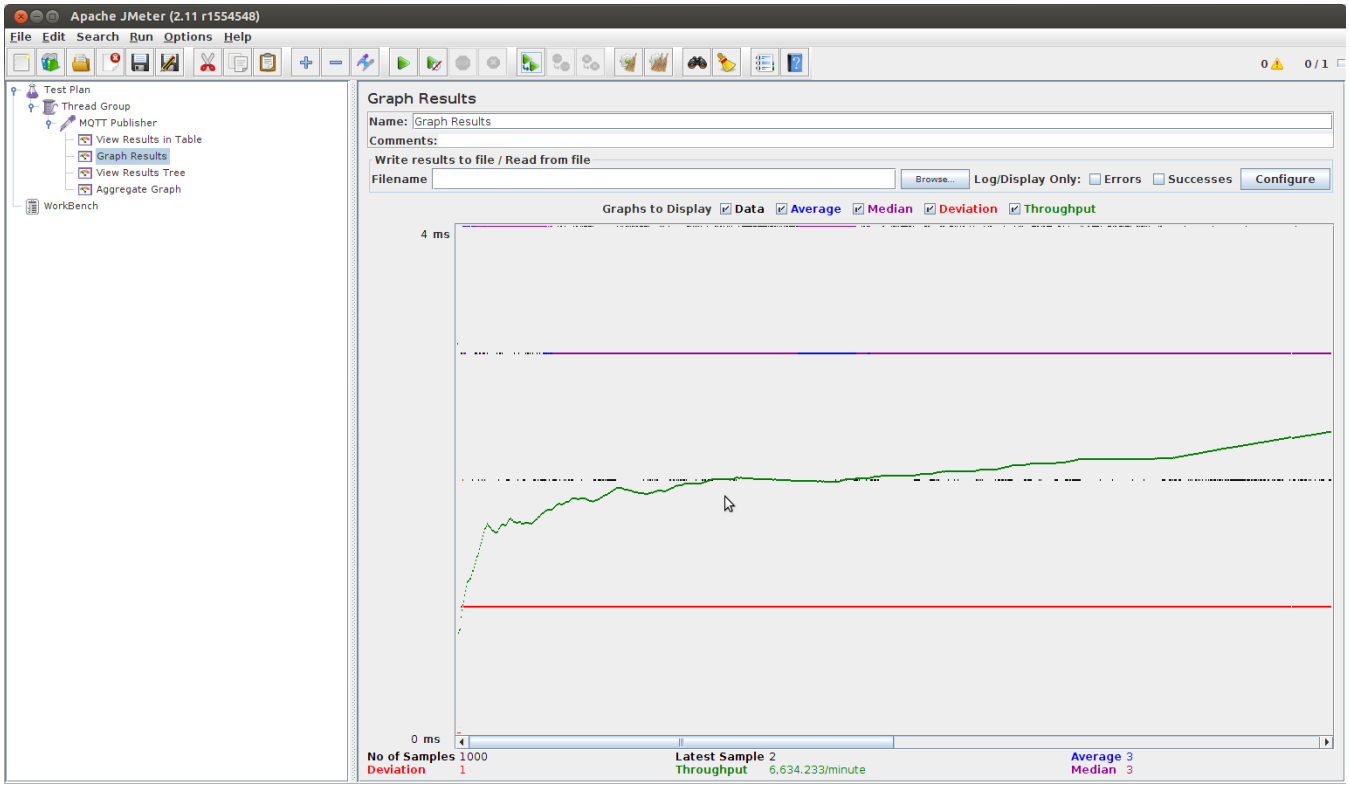


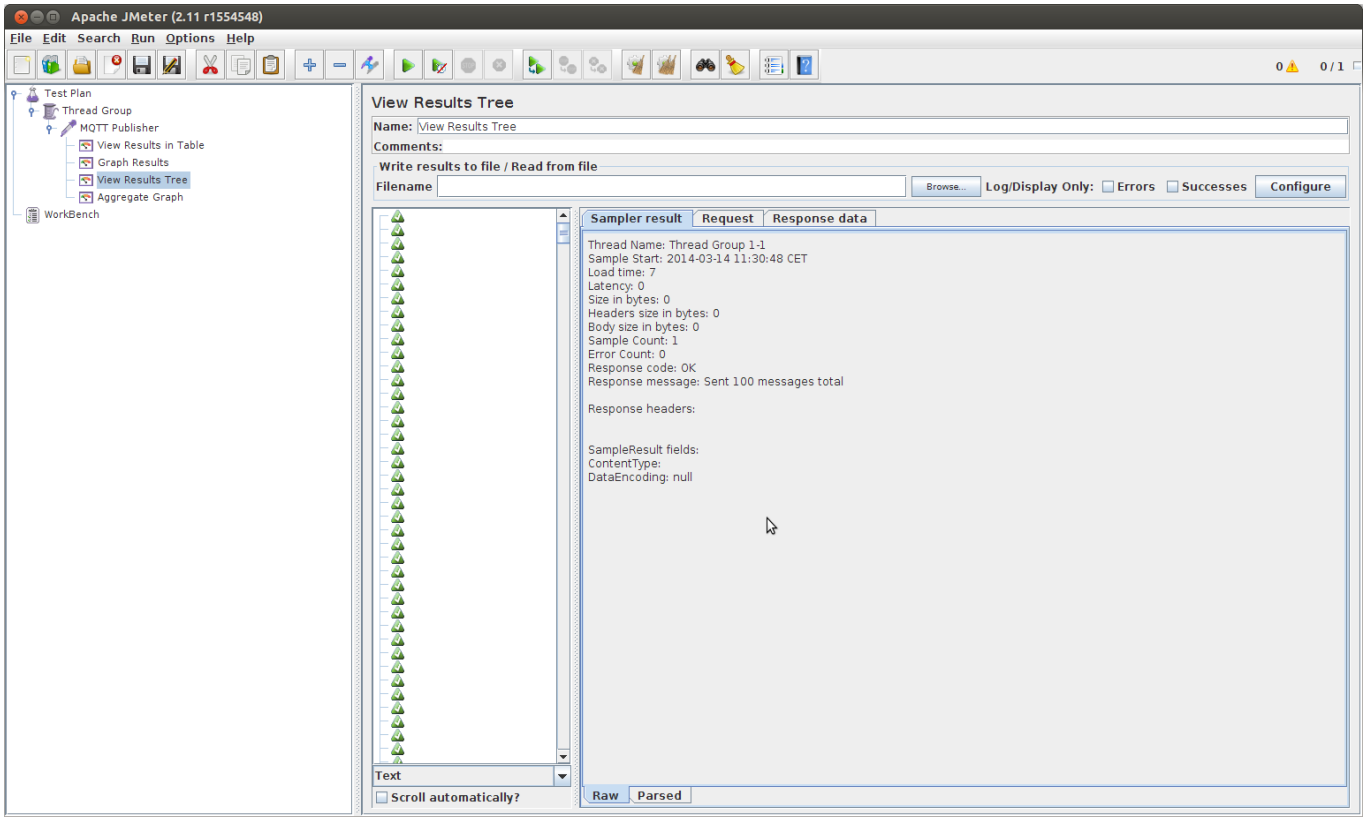
- **Fixed Value:**

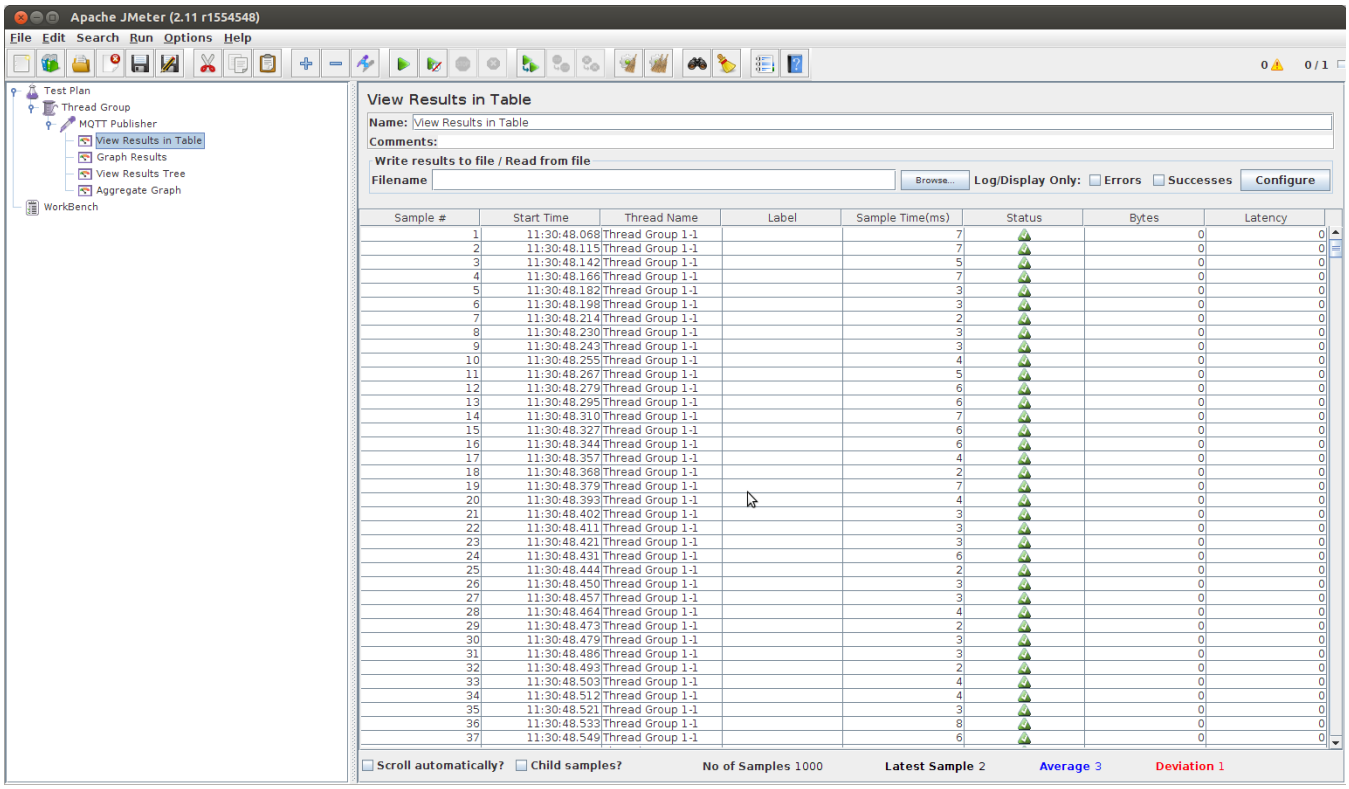
The fixed value can be of type: Integer, Long, Float, Double, String within the range [Min,Max] .



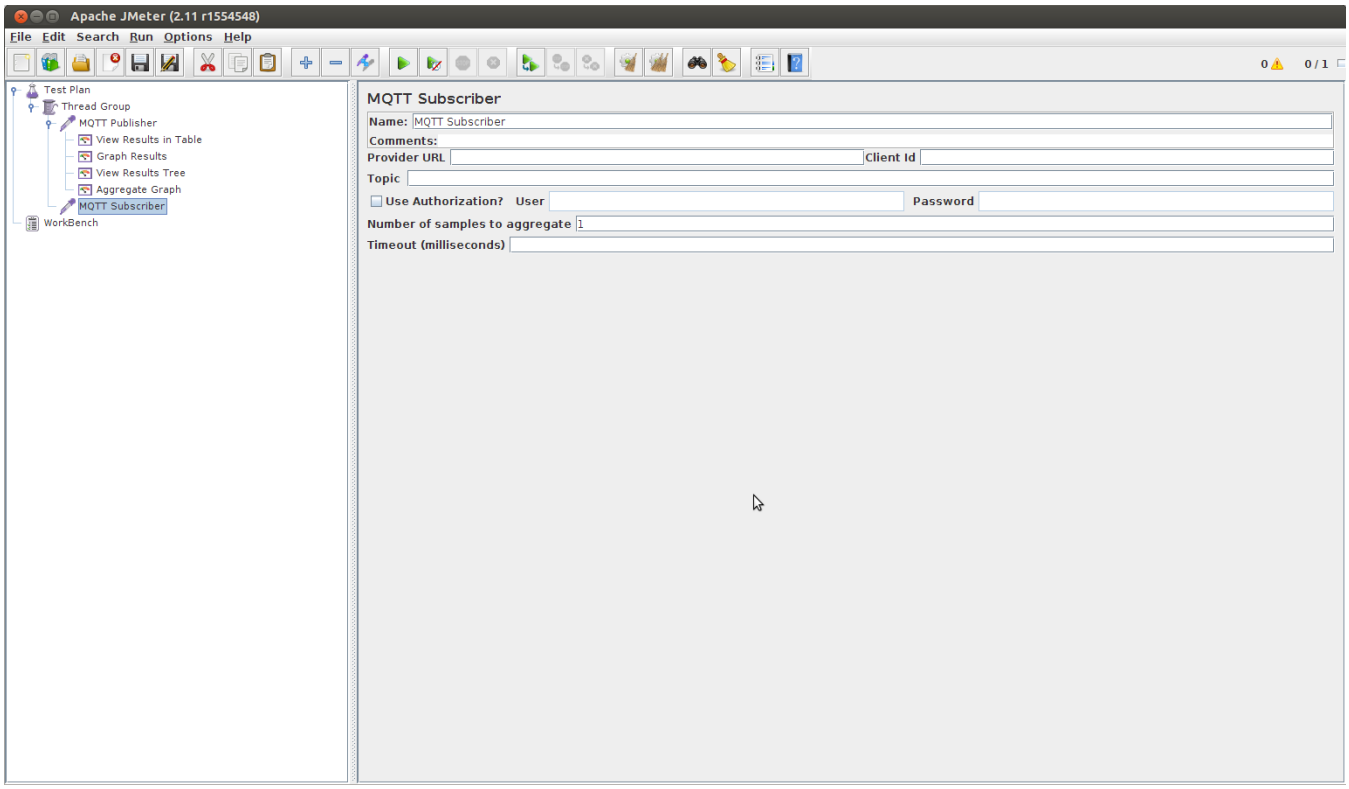
For mesuring, thanks to Jmeter, we can add some listeners:



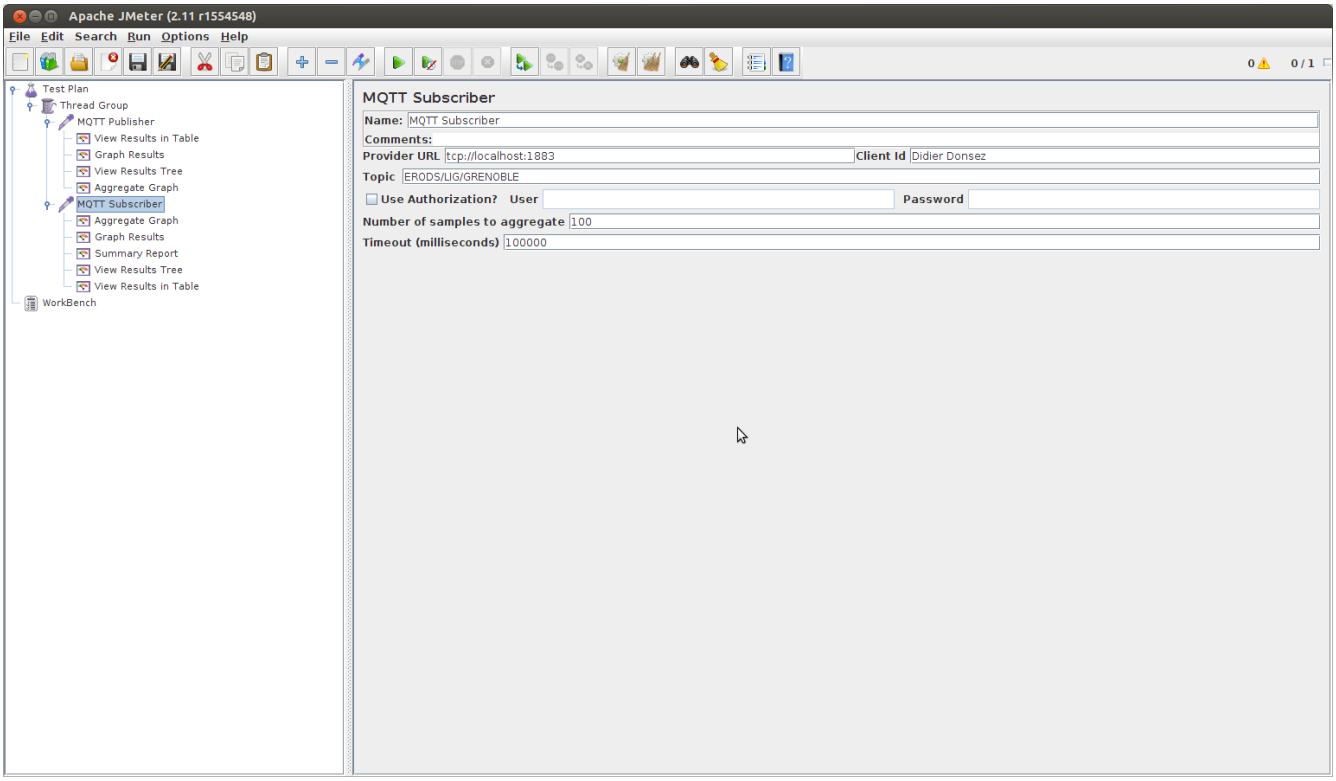

Apache JMeter (2.11 r1554548)
File Edit Search Run Options Help
0 0 / 1
Test Plan
Thread Group
MQTT Publisher
View Results in Table
Graph Results
View Results Tree
Aggregate Graph
WorkBench
View Results Tree
Name: View Results Tree
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Sampler result
Request
Response data
Thread Name: Thread Group 1-1
Sample Start: 2014-03-14 11:30:48 CET
Load time: 7
Latency: 0
Size in bytes: 0
Headers size in bytes: 0
Body size in bytes: 0
Sample Count: 1
Error Count: 0
Response code: OK
Response message: Sent 100 messages total
Response headers:
SampleResult fields:
ContentType:
DataEncoding: null
Text
Scroll automatically?
Raw
Parsed

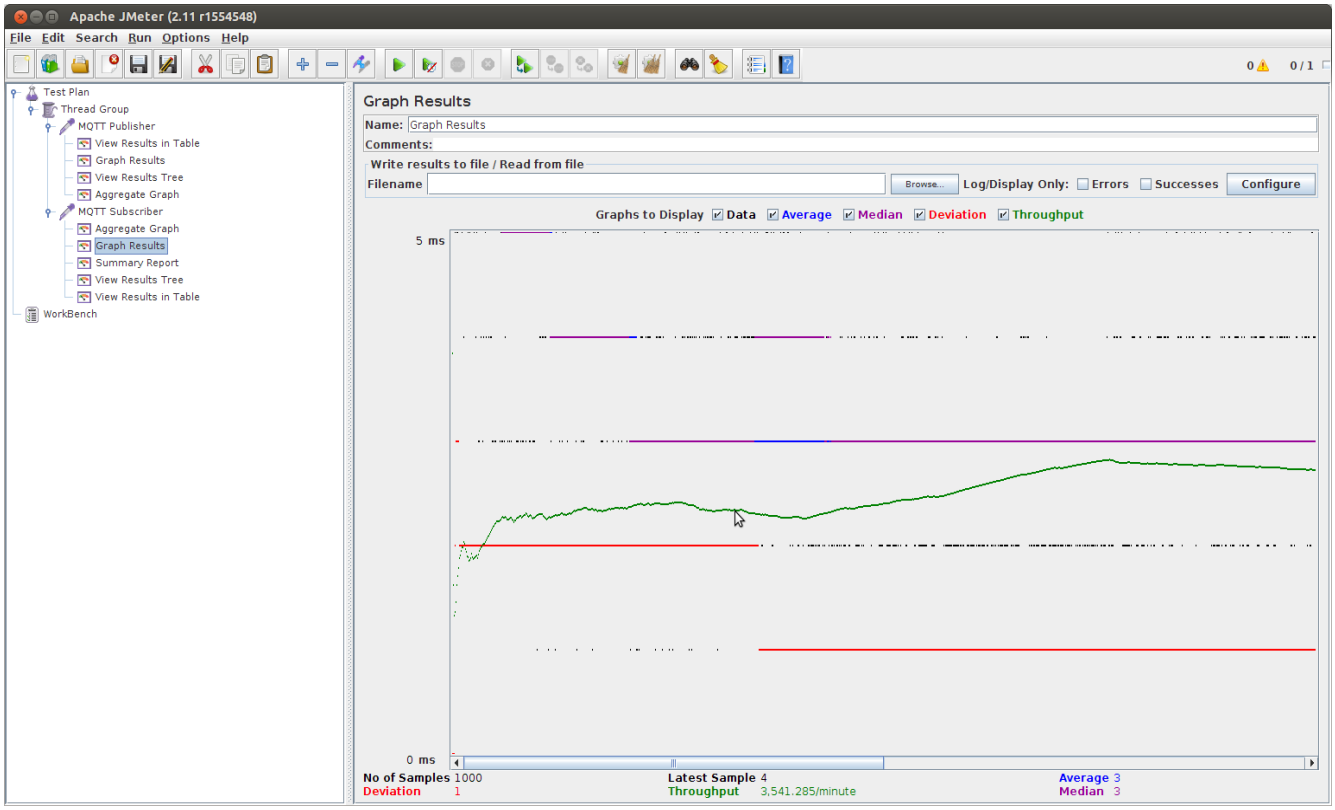


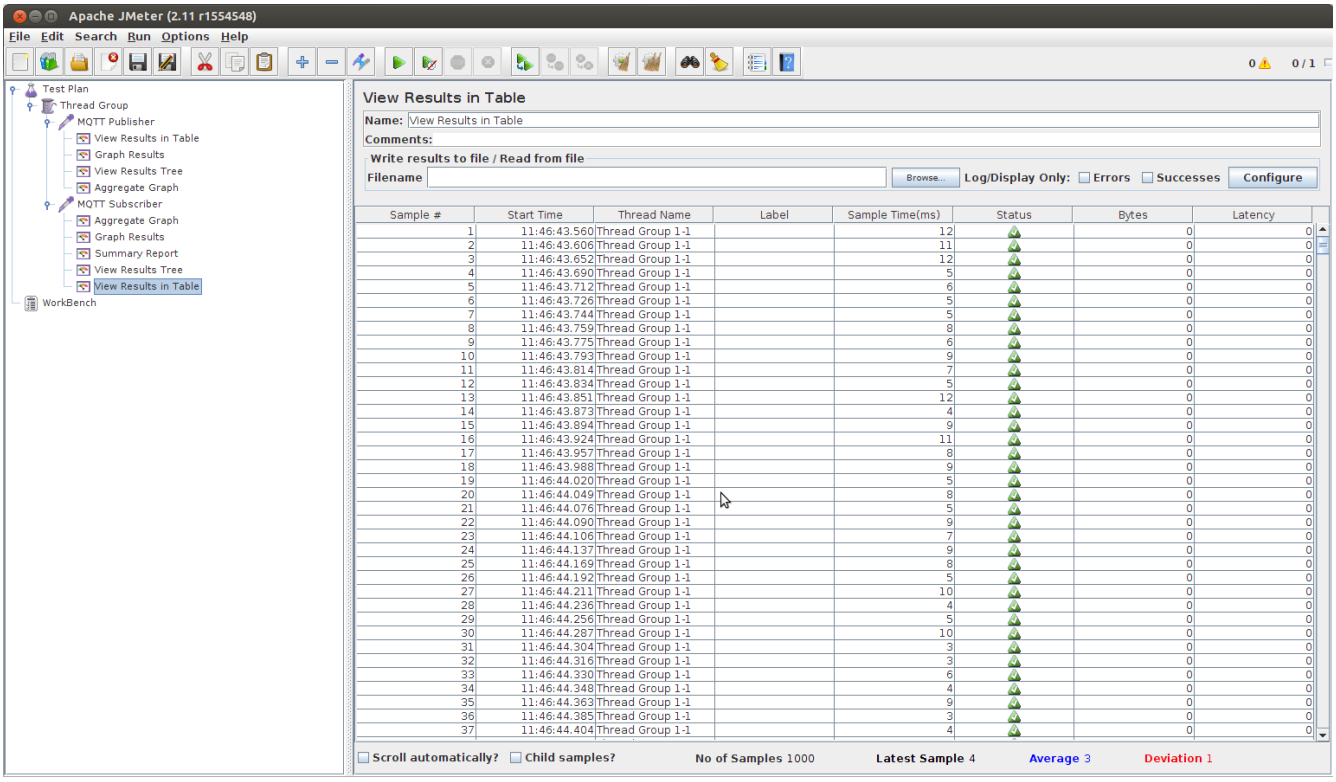
2. MQTT Subscriber



- *Name:* Name of the MQTT Subscriber
- *Comments:* Your comments
- *Provider URL:* The address of MQTT server
- *Client Id:* Your Id in the session
- *Topic:* The topic you want to subscribe.
- *Use Authorization :* Necessary in the case the connection need username and password
- *User:* your username
- *Password:* your password
- *Number of samples to aggregate :* In other way, the number of message you want to receive from the topic in one session
- *Time out (milliseconds):* timeout for the connection to receive message from the topic


Apache JMeter (2.11 r1554548)
File Edit Search Run Options Help
Test Plan
Thread Group
MQTT Publisher
View Results in Table
Graph Results
View Results Tree
Aggregate Graph
MQTT Subscriber
Aggregate Graph
Graph Results
Summary Report
View Results Tree
View Results in Table
WorkBench
MQTT Subscriber
Name: MQTT Subscriber
Comments:
Provider URL tcp://localhost:1883
Client Id Didier Donsez
Topic ERODS/LIG/GRENOBLE
Use Authorization? User Password
Number of samples to aggregate 100
Timeout (milliseconds) 100000


Apache JMeter (2.11 r1554548)
File Edit Search Run Options Help
Test Plan
Thread Group
MQTT Publisher
View Results in Table
Graph Results
View Results Tree
Aggregate Graph
MQTT Subscriber
Aggregate Graph
Graph Results
Summary Report
View Results Tree
View Results in Table
WorkBench
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
5 ms
0 ms
No of Samples 1000
Latest Sample 4
Average 3
Deviation 1
Throughput 3,541.285/minute
Median 3



Grenoble, France 14/03/2014,

ERODS Team