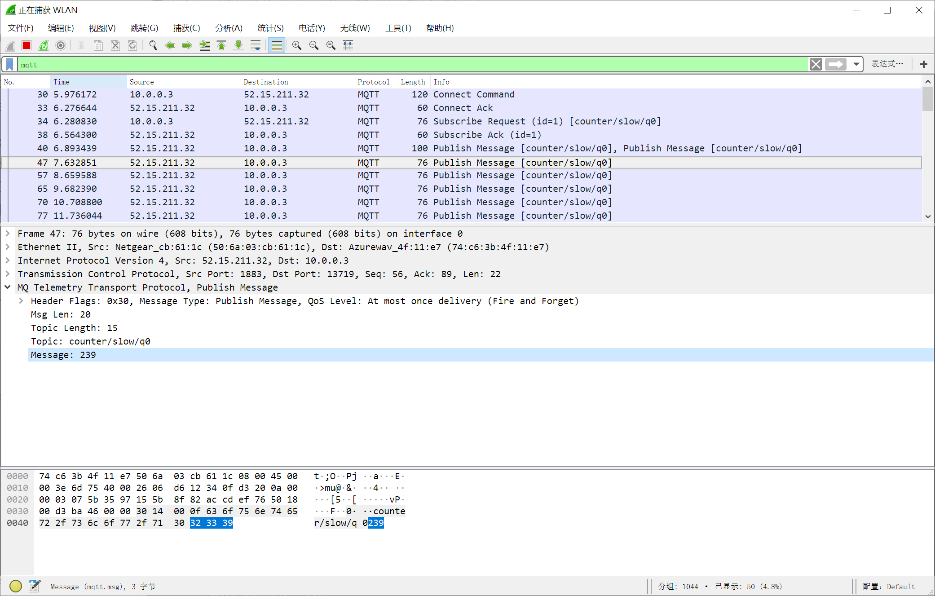
1. In the MQTT protocol, the QoS has three different flags which 0 stands for at most send once, 1 stands for at least send once and 2 stands for send once only. All three of QoS level has initial handshake which are

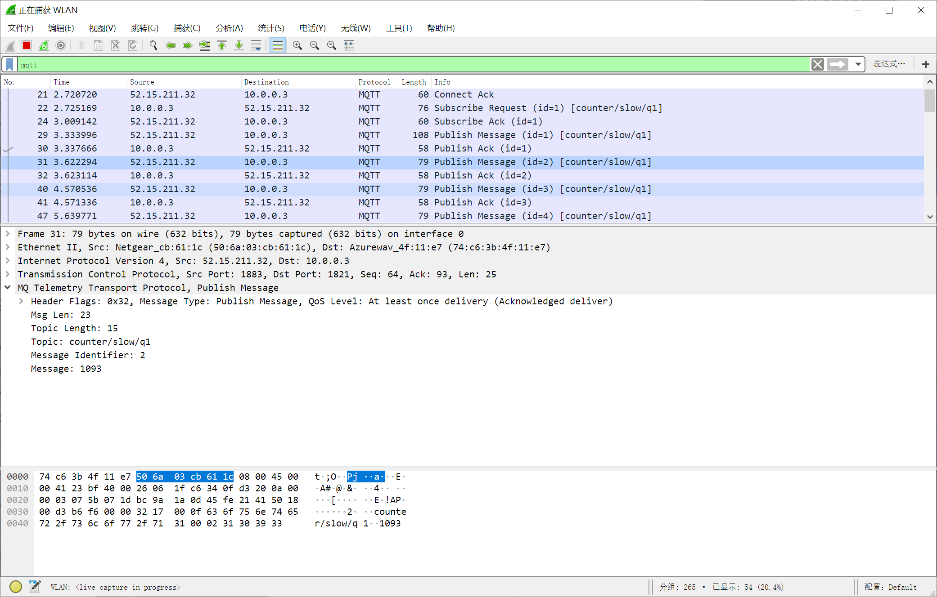
- Sent the connect command from client,

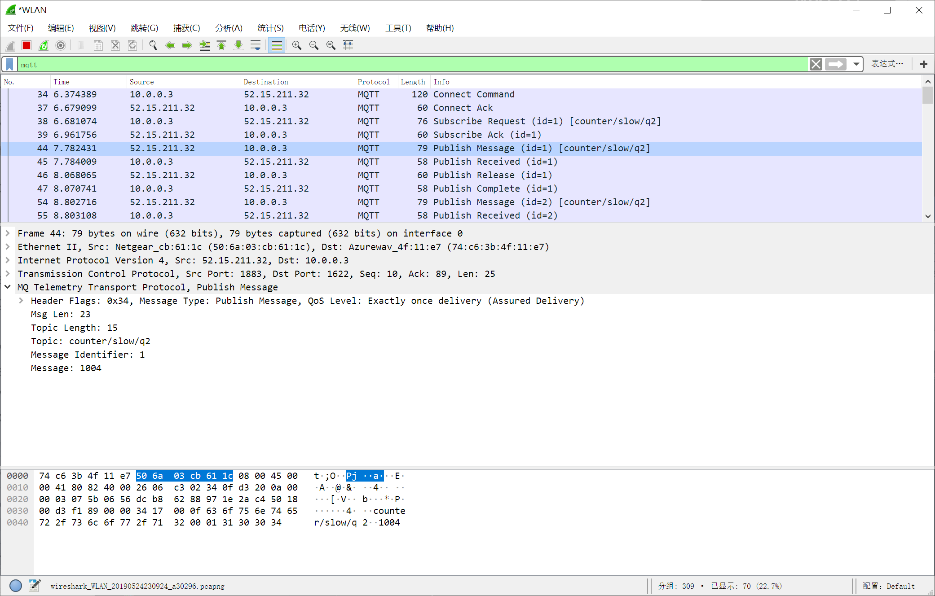
- Server sent the acknowledgement back to client,

- The client sent the subscribe request,

- Subscribe acknowledgement send from Server.

However, each QoS level has different message handshakes. When the QoS level is 0 each message at most send once. There is no extra handshake, the server will send the message only. Thus, the server will not get any acknowledgement from clients. In this case, some message could be lost if the connection between the sever and clients is not stable. Nowadays, this level could be used for the environment sensor data transportation, lost some packages is not critical in this case because the newer data will be sent very soon.

When the QoS level is 1, each message will be sent at least once. In this case, the client will send an extra acknowledgement back to sever to ensure the client has received that message successfully. If the server did not receive that acknowledgement on time, the server will send that message again. Compare with level 0 and level 2, this level is more reliable then level 0 and faster than level 2. However, if the client communication jammed to cause the receiving acknowledgement not sent out on time the server will send same message as well. Thus, that may cause duplicate packages. This level could be used for fire alarm system. Because for the system both response speed and reliability are important. Furthermore, in that case duplicate package is not affect anything. Thus, the level 1 is most suitable for that system.

When the QoS level is 2, which send one package exactly once. In this case, there are 4 steps between the server and the client for each package. Firstly, the server will send the message to clients. Secondly, the client will send the receiving acknowledgement to server. Thirdly, after the server receive that acknowledgement, server will send to client again about that publish is released. Finally, the client will send back to server about the publish is complete. Compare with level 0 and 1, this level is the most complicate QoS level. It requires 4 communications for each message. Due to that, this level also cost more time than level 0 and 1. However, level 2 is the most reliable QoS level. It will guarantee the message arrived successfully and without duplicate in most time. Due to the reliability, the level 2 could use for paying system. Usually, the duplicate and lost message might cause incorrect amount to pay. But the level 2 will not generate the lost and duplicate message in the most of time.

2. a. For fast counter channels, the variation impacting the lost rate. The variation is a measure of the degree of dispersion when probability theory and statistical variance measure a random variable or a set of data. In my statistics, fast counter with the QoS level 0 does not lose any message also has a low variation about 473.29 which means the time between each message is close to the average. QoS level 1 losing about 88% message in my testing but also with a vert huge variation which is 10780.73. Furthermore, the QoS level 2 lost rate is higher than QoS level 1 around 93.98%. that level also has a higher variation that level 1. The average gap is impacting the receive rate. In my testing, the QoS level 0 receive 272 messages pre second, the average gap is 3.69 milliseconds. The QoS level 1 should received 60729 messages but only receiver 7000 messages and the receive rate is 23.34 messages pre second. The average gap between each message for level1 is 43.02 milliseconds. For the level 2, the receive rate is 11.86 message pre second and the average gap is 84.66 milliseconds. Compare with level 1 and level 2, the receive rate of level 1 is faster twice than level 2 and the average gap is lower twice than level 2.

b.

c.

3. For this assignment, I used the Java to program it and use an extra library which is the org.eclipse.paho.client.mqttv3. The LAN is 5GHz wireless, WAN is the nbn100.

In my code, I use the HashSet to check the duplicate rate. Because the HashSet does not same data twice. I use one ArrayList to storage all message I received and put every message into a HashSet. Then I compare the size of two ADT to determinate the duplicate rate. For the out of order, I compare current message and 10 previous messages before that message. Due to the counter is continuous increased. Thus, out of order will be recorded if any one of previous messages greater than current message.

On the one hand, I test the lost rate, duplicate rate and out of order rate both 0 and the received rate is 0.98 message pre second on three QoS level on the slow counter channel. That shows network between the client and the server is stable also the client and the server is powerful enough to handle those messages. Because the lost rate is 0% on level 0 which send no more than once. If the connection is unstable or client and server not powerful, there are lost package on level 0. The level 1 and level 2 will not lost message as I mentioned in Q1. However, the level 1 might contain duplicate message if the connection is unstable. If the sever does not receive the acknowledgement from client on time, it will send same message to client again. Moreover, the server counter is continuing increase. Thus, the message out of order issue should not cause by the server and connection. On the level 2 might be message out of order, because the level 2 need time to make 4 steps for each message. Thus, if the client is not fast enough, the message might be out of order.

On the other hand, in the fast counter channel, I got no lost on QoS level 0, but over 80% lost on level 1 and level 2. As I mentioned in the Q1 the level 1 and 2 should not lost any package. However, in the situation which the sever sending a lot of message fast. There is latency between the broker and the client. Thus, there might be many messages pending for QoS level 1 and 2, before the sever receive the acknowledgement from client about clear to send the next message. Moreover, the CPU usage and memory are not infinity big on both client machine and server. Thus, in that situation the server or client will drop messages due to there not fast enough to handle all message in that speed or insufficient memory. That dropping packages will show as lost packages in the sequence of MQTT message. There is no message out of order because the server dropped all message which unable to handle.

4.

In the scenario with millions of sensors and thousands of subscribers. The minimum length of each MQTT message is 3 byte include 2 bytes fixed header and 1 byte message payload plus variable header (IBM, 2010). Assume each sensor only send one minimum message to broker per second, the broker have to handle 3MB different message from different source per second. Assume that broker has 20 cores CPU. The CPU have to process each second within