Challenge 1 -

1) What are the differences between the request with MID: 53533 and the one with MID: 42804?

We used the filter *“coap.mid == 42804* || *coap.mid==53533”* to get the required message. Analyzing the packets we noticed:

- The message with mid=5355 is a GET request of type confirmable, as a matter of fact using this filter we can also see the reply message.
- The message with mid = 42804 is a DELETE request of type Not-confirmable

We also noticed that the ip of the destination is not the same.

2) What is the response of message No. 2428, if any?

After finding the message No. 2428, which is a delete request, we used the filter “*coap.token == 67:c7:22:9a”* to check if there was a response.
We found message No.2429 that has a coap.code == 66 (deleted), the same token but a different message ID and it contains 16 bytes of data corresponding to the deleted elements.
We didn’t use the filter on the message ID because the request was of type Non-Confirmable.

3) How many replies to requests of type confirmable, having the result code “Content” are received by the client “localhost”?

We used the filter “*coap.code==69 && ip.dst_host == 127.0.0.1*” to get all the packets with the required code (Content = 69) received by localHost. All the 8 packets filtered are ACK sent to some confirmable requests.

4) How many GET requests, excluding OBSERVE requests, have been directed to non existing resources?

We started using the filter *“coap.code==132*” in order to find all the “resource not found “ response messages.
Then, for each of the 8 responses, we applied the filter with its token and the coap code equal to GET (“*coap.code == GET && coap.token == message_token”*) and we found a GET request for only 6 of the 8 responses.
As an additional proof we used the filter “*coap.opt.observe==0* ” to analyze all the OBSERVE subscriptions and we found subscriptions linked to the same topics of the other 2 response messages.
In conclusion we found 6 GET request messages directed to non existing resources.

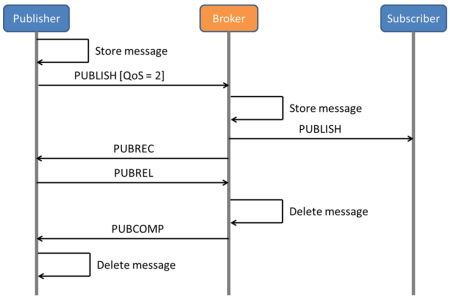
5) How many messages containing the topic “factory/department*/+” are published by a client with user password: “admin”?

At first we used the filter “*mqtt.topic contains factory/departmentX* ” (X that changes between 0 and 9) and we noticed that all the messages published do not have just one level after department as requested by the wildcard +.
Our conclusion is that the client with user password “admin” did not publish messages containing the required topic. Note: from the question we assume we are looking for messages that contain one single level after department

6) How many clients connected to the public broker “mosquitto” have specified a will message?

Using the terminal we connected to the broker *test.mosquitto.org* and through wireshark we recovered the broker's IP (5.196.95.108). Then we used the filter *“ip.dest == 5.196.95.108 && mqtt.willmsg”* finding 9 clients who connected to the broker and setted a will message.
Note: we have assumed that all the connections are from different clients.

7) How many publishes with QoS 2 don’t receive the PUBREL?



We used the filter *“mqtt.qos == 2 && mqtt.msgtype == 3”* to count all the messages published with qos= 2 and we got 94 messages. After that we used the filter “*mqtt.msgtype==6”* and we noticed that no message of type PUBREL was sent. So all the 94 publish messages don’t receive the PUBREL
Note: As we can see from the slide the PUBREL message is sent from the publisher, not received.

8) What is the average Will Topic Length specified by clients with empty Client ID?

We used the filter *“!(_ws.malformed.expert) && mqtt.clientid_len==0 && mqtt.willtopic_len”* to get all packets with connect messages where a will message is setted and the client ID is empty. We count all the will message length and we average it counting the client connection messages :

- average including client without will message settled (length=0) -> 831/108 = 7.69
- average without including client without will message settled -> 831/22 = 37.7

9) How many ACKs received the client with ID "6M5H8y3HJD5h4EEscWknTD"? What type(s) is(are) it(them)?

Using the filter “*mqtt.clientid == 6M5H8y3HJD5h4EEscWknTD”* we took the IP and the port of the targeted client. Then we filtered as: “*ip.dst==10.0.2.15 && tcp.dstport==46295 && mqtt*” to get all the mqtt messages received from the client.
We counted 5 ACK of which one is a connect, three subscribe and one publish.

10)What is the average MQTT message length of the CONNECT messages using mqttv3.1 protocol? Why messages have different size?

We used the filter “*mqtt.msgtype==1 && mqtt.ver==3”* to get all the connected messages using mqttv3.1. We averaged the mqtt message length of the 47 messages and got a length of approx 63 bytes.
The different length can be given by the different selections during the connection: username, password, willmessage, willtopic.