IoT 2023 Challenge 1

Author

Name: Mario Cela

Person Code: 10685242

Comment

I did not included the code in this pdf because there were several function and it would have meant to copy-paste the whole code here. I preferred to keep the code ordered and commented so it is still clear to understand. All the questions are divided, there is a function for each of them (Q1, Q2, Q3, and Q4) in order to facilitate comprehension.

Questions and Answers

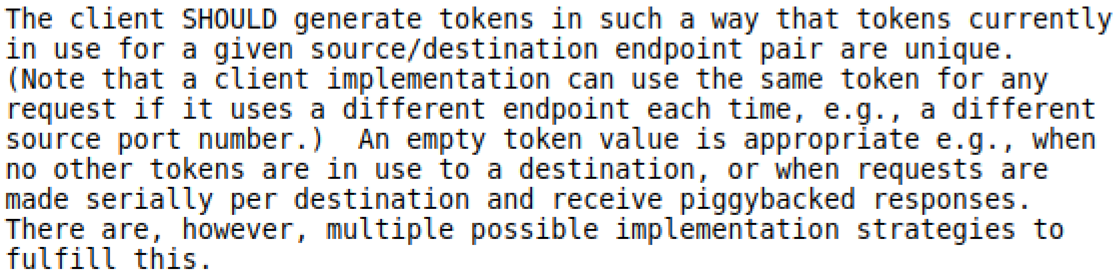
Question 1. How many CoAP GET requests are directed to non-existing resources in the local CoAP server? How many of these are of type Non confirmable? **(0.2 pts)**

Answer a: 11

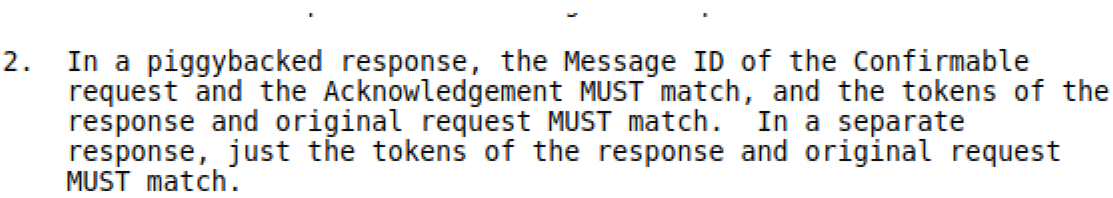
Answer b: 6

The basic concept for the algorithm is to check any correspondence between msg_ID and tokens. In particular, as shown in CoAP's documentation, when we have synchronous communication we shall check that request and response have the same msg_ID; when it is asynchronous, it has to be checked the token.

In particular, from CoAP's documentation we can read that:



```
The client SHOULD generate tokens in such a way that tokens currently
in use for a given source/destination endpoint pair are unique.
(Note that a client implementation can use the same token for any
request if it uses a different endpoint each time, e.g., a different
source port number.)  An empty token value is appropriate e.g., when
no other tokens are in use to a destination, or when requests are
made serially per destination and receive piggybacked responses.
There are, however, multiple possible implementation strategies to
fulfill this.
```



```
2.  In a piggybacked response, the Message ID of the Confirmable
    request and the Acknowledgement MUST match, and the tokens of the
    response and original request MUST match.  In a separate
    response, just the tokens of the response and original request
    MUST match.
```

Question 2. How many CoAP DELETE requests directed to the "coap.me" server did not produce a successful result? How many of these are directed to the "/hello" resource? **(0.2 pts)**

Answer a: 93

Answer b: 5

For the answers, it has been assumed that a DELETE request results to be unsuccessful only when we have a response that contains a code different from 6* (success messages).

Speaking about the type of the communication, the basic concept is the same as in the previous case, in particular it has been used the same concept on msgID and tokenID.

Question 3. How many different MQTT clients subscribe to the public broker mosquitto using single-level wildcards? How many of these *clients* **WOULD** receive a publish message issued to the topic "hospital/room2/area0“ **(0.2 pts)**

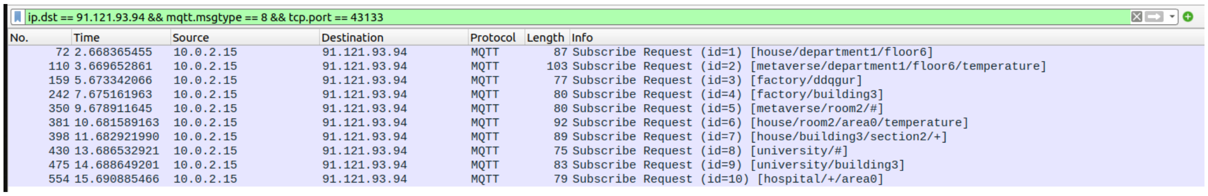
Answer a: 3

Answer b: 2

For the first part of the question, it has been saved the list of ports used for the requests made by the client: in this way we can get the number of different clients that are making requests.

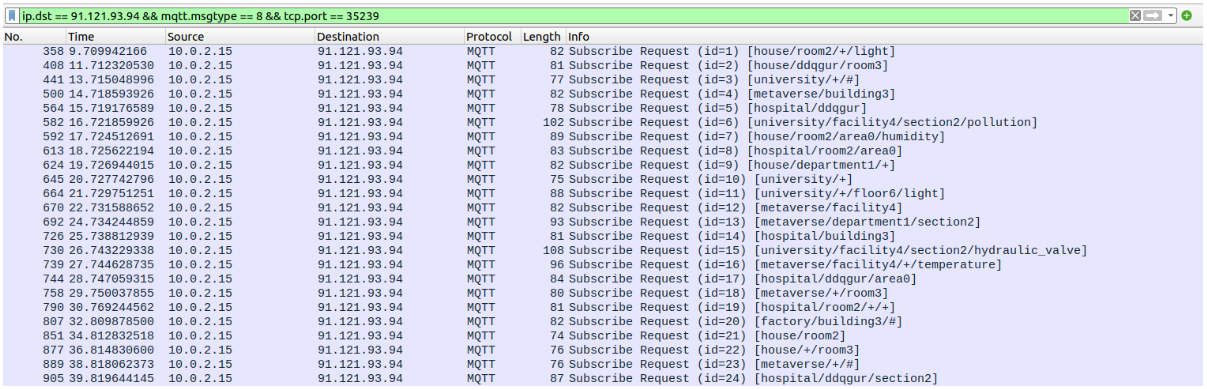
For the second part, considering that there were just 3 ports to study, the analysis has been made manually using Wireshark.

The three filters used are:

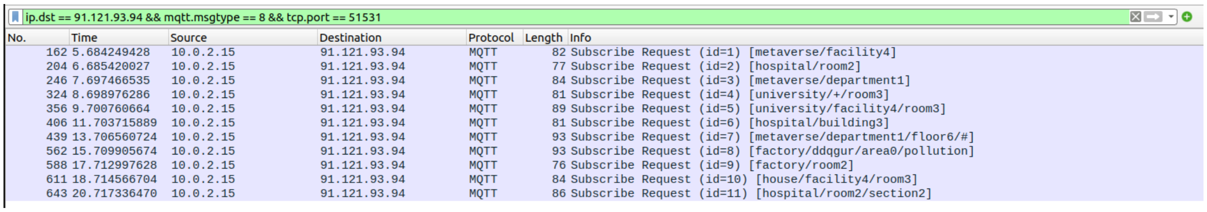


ip.dst == 91.121.93.94 && mqtt.msgtype == 8 && tcp.port == 43133

No.	Time	Source	Destination	Protocol	Length	Info
72	2.668365455	10.0.2.15	91.121.93.94	MQTT	87	Subscribe Request (id=1) [house/department1/floor6]
110	3.669652861	10.0.2.15	91.121.93.94	MQTT	103	Subscribe Request (id=2) [metaverse/department1/floor6/temperature]
159	5.673342066	10.0.2.15	91.121.93.94	MQTT	77	Subscribe Request (id=3) [factory/ddqgur]
242	7.675161963	10.0.2.15	91.121.93.94	MQTT	80	Subscribe Request (id=4) [factory/building3]
350	9.678911645	10.0.2.15	91.121.93.94	MQTT	80	Subscribe Request (id=5) [metaverse/room2/#]
381	10.681589163	10.0.2.15	91.121.93.94	MQTT	92	Subscribe Request (id=6) [house/room2/area0/temperature]
398	11.682921990	10.0.2.15	91.121.93.94	MQTT	89	Subscribe Request (id=7) [house/building3/section2/+]
430	13.686532921	10.0.2.15	91.121.93.94	MQTT	75	Subscribe Request (id=8) [university/#]
475	14.688649201	10.0.2.15	91.121.93.94	MQTT	83	Subscribe Request (id=9) [university/building3]
554	15.690885466	10.0.2.15	91.121.93.94	MQTT	79	Subscribe Request (id=10) [hospital/+/area0]



```
ip.dst == 91.121.93.94 && mqtt.msgtype == 8 && tcp.port == 35239
No.   Time         Source      Destination   Protocol Length Info
  358 9.709942166  10.0.2.15   91.121.93.94  MQTT      82 Subscribe Request (id=1) [house/room2/+/light]
  408 11.712320530 10.0.2.15   91.121.93.94  MQTT      81 Subscribe Request (id=2) [house/ddqgur/room3]
  441 13.715048996 10.0.2.15   91.121.93.94  MQTT      77 Subscribe Request (id=3) [university/+/#]
  500 14.718593926 10.0.2.15   91.121.93.94  MQTT      82 Subscribe Request (id=4) [metaverse/building3]
  564 15.719176589 10.0.2.15   91.121.93.94  MQTT      78 Subscribe Request (id=5) [hospital/ddqgur]
  582 16.721859926 10.0.2.15   91.121.93.94  MQTT     102 Subscribe Request (id=6) [university/facility4/section2/pollution]
  592 17.724512691 10.0.2.15   91.121.93.94  MQTT      89 Subscribe Request (id=7) [house/room2/area0/humidity]
  613 18.725622194 10.0.2.15   91.121.93.94  MQTT      83 Subscribe Request (id=8) [hospital/room2/area0]
  624 19.726944015 10.0.2.15   91.121.93.94  MQTT      82 Subscribe Request (id=9) [house/department1/+]
  645 20.727742796 10.0.2.15   91.121.93.94  MQTT      75 Subscribe Request (id=10) [university/+]
  664 21.729751251 10.0.2.15   91.121.93.94  MQTT      88 Subscribe Request (id=11) [university/+/floor6/light]
  670 22.731588652 10.0.2.15   91.121.93.94  MQTT      82 Subscribe Request (id=12) [metaverse/facility4]
  692 24.734244859 10.0.2.15   91.121.93.94  MQTT      93 Subscribe Request (id=13) [metaverse/department1/section2]
  726 25.738812939 10.0.2.15   91.121.93.94  MQTT      81 Subscribe Request (id=14) [hospital/building3]
  730 26.743229338 10.0.2.15   91.121.93.94  MQTT     108 Subscribe Request (id=15) [university/facility4/section2/hydraulic_valve]
  739 27.744628735 10.0.2.15   91.121.93.94  MQTT      96 Subscribe Request (id=16) [metaverse/facility4/+/temperature]
  744 28.747059315 10.0.2.15   91.121.93.94  MQTT      84 Subscribe Request (id=17) [hospital/ddqgur/area0]
  758 29.750037855 10.0.2.15   91.121.93.94  MQTT      80 Subscribe Request (id=18) [metaverse/+/room3]
  790 30.769244562 10.0.2.15   91.121.93.94  MQTT      81 Subscribe Request (id=19) [hospital/room2/+/+]
  807 32.809878500 10.0.2.15   91.121.93.94  MQTT      82 Subscribe Request (id=20) [factory/building3/#]
  851 34.812832518 10.0.2.15   91.121.93.94  MQTT      74 Subscribe Request (id=21) [house/room2]
  877 36.814830600 10.0.2.15   91.121.93.94  MQTT      76 Subscribe Request (id=22) [house/+/room3]
  889 38.818062373 10.0.2.15   91.121.93.94  MQTT      76 Subscribe Request (id=23) [metaverse/+/#]
  905 39.819644145 10.0.2.15   91.121.93.94  MQTT      87 Subscribe Request (id=24) [hospital/ddqgur/section2]
```



```
ip.dst == 91.121.93.94 && mqtt.msgtype == 8 && tcp.port == 51531
No.   Time         Source      Destination   Protocol Length Info
  162 5.684249428  10.0.2.15   91.121.93.94  MQTT      82 Subscribe Request (id=1) [metaverse/facility4]
  204 6.685420027  10.0.2.15   91.121.93.94  MQTT      77 Subscribe Request (id=2) [hospital/room2]
  246 7.697466535  10.0.2.15   91.121.93.94  MQTT      84 Subscribe Request (id=3) [metaverse/department1]
  324 8.698976286  10.0.2.15   91.121.93.94  MQTT      81 Subscribe Request (id=4) [university/+/room3]
  356 9.700760664  10.0.2.15   91.121.93.94  MQTT      89 Subscribe Request (id=5) [university/facility4/room3]
  406 11.703715889 10.0.2.15   91.121.93.94  MQTT      81 Subscribe Request (id=6) [hospital/building3]
  439 13.706560724 10.0.2.15   91.121.93.94  MQTT      93 Subscribe Request (id=7) [metaverse/department1/floor6/#]
  562 15.709905674 10.0.2.15   91.121.93.94  MQTT      93 Subscribe Request (id=8) [factory/ddqgur/area0/pollution]
  588 17.712997628 10.0.2.15   91.121.93.94  MQTT      76 Subscribe Request (id=9) [factory/room2]
  611 18.714566704 10.0.2.15   91.121.93.94  MQTT      84 Subscribe Request (id=10) [house/facility4/room3]
  643 20.717336470 10.0.2.15   91.121.93.94  MQTT      86 Subscribe Request (id=11) [hospital/room2/section2]
```

We can see that the clients of the first and of the second image are subscribed to the topic hospital/room2/area0, so the answer is 2.

Question 4. How many MQTT clients specify a last Will Message directed to a topic having as first level "university"? How many of these Will Messages are sent from the broker to the subscribers? **(0.2 pts)**

Answer a: 2

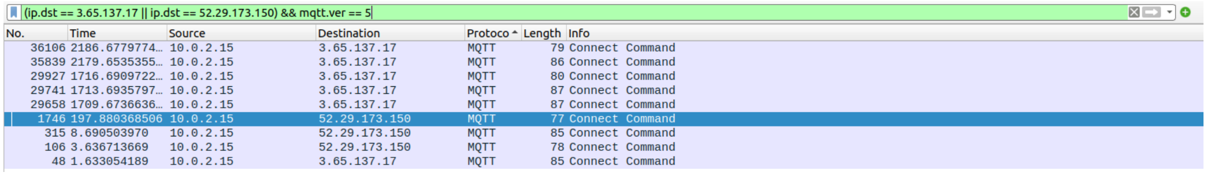
Answer b: 0

The algorithms saves those packets that has the willflag to 1 and that have a topic that startswith university. For the second part of the question, the algorithm saves into a list the content of the last will messages and check whether there is any PUBLISH messages which content is the same. From the anaylsis, it comes out that those last will messages have never been sent to the subscribers.

Question 5. How many Publish messages with QoS = 1 are received by the MQTT clients connected to the HiveMQ broker with MQTT version 5? **(0.1 pts)**

Answer: 60

With the first filter we get all the clients we are interested in, in particular we focus on their ports.

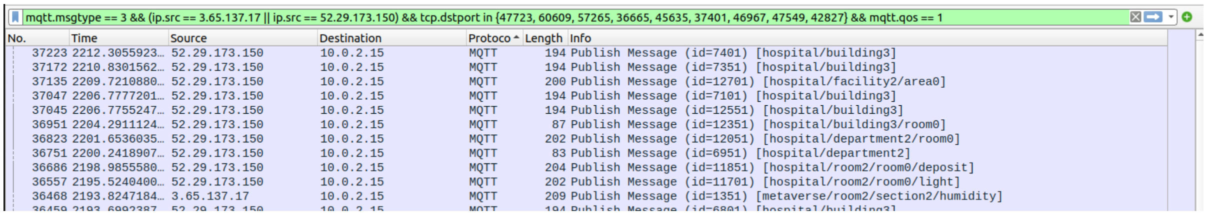


(ip.dst == 3.65.137.17 || ip.dst == 52.29.173.150) && mqtt.ver == 5

No.	Time	Source	Destination	Protoco	Length	Info
36106	2186.6779774…	10.0.2.15	3.65.137.17	MQTT	79	Connect Command
35839	2179.6535355…	10.0.2.15	3.65.137.17	MQTT	86	Connect Command
29927	1716.6909722…	10.0.2.15	3.65.137.17	MQTT	80	Connect Command
29741	1713.6935797…	10.0.2.15	3.65.137.17	MQTT	87	Connect Command
29658	1709.6736636…	10.0.2.15	3.65.137.17	MQTT	87	Connect Command
1746	197.880368506	10.0.2.15	52.29.173.150	MQTT	77	Connect Command
315	8.690503970	10.0.2.15	52.29.173.150	MQTT	85	Connect Command
106	3.636713669	10.0.2.15	52.29.173.150	MQTT	78	Connect Command
48	1.633054189	10.0.2.15	3.65.137.17	MQTT	85	Connect Command

From the content of those packets, we can get the ports of the clients.

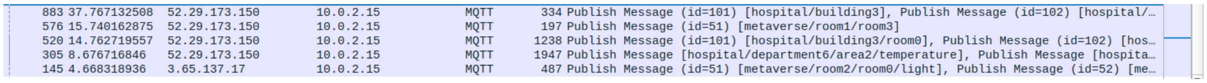
Then, with the second filter we get all the packets filtering on the identified ports. The filter shows a total of 51 packets.



mqtt.msgtype == 3 && (ip.src == 3.65.137.17 || ip.src == 52.29.173.150) && tcp.dstport in {47723, 60609, 57265, 36665, 45635, 37401, 46967, 47549, 42827} && mqtt.qos == 1

No.	Time	Source	Destination	Protoco	Length	Info
37223	2212.3055923…	52.29.173.150	10.0.2.15	MQTT	194	Publish Message (id=7401) [hospital/building3]
37172	2210.8301562…	52.29.173.150	10.0.2.15	MQTT	194	Publish Message (id=7351) [hospital/building3]
37135	2209.7210880…	52.29.173.150	10.0.2.15	MQTT	200	Publish Message (id=12701) [hospital/facility2/area0]
37047	2206.7777201…	52.29.173.150	10.0.2.15	MQTT	194	Publish Message (id=7101) [hospital/building3]
37045	2206.7755247…	52.29.173.150	10.0.2.15	MQTT	194	Publish Message (id=12551) [hospital/building3]
36951	2204.2911124…	52.29.173.150	10.0.2.15	MQTT	87	Publish Message (id=12351) [hospital/building3/room0]
36823	2201.6536035…	52.29.173.150	10.0.2.15	MQTT	202	Publish Message (id=12051) [hospital/department2/room0]
36751	2200.2418907…	52.29.173.150	10.0.2.15	MQTT	83	Publish Message (id=6951) [hospital/department2]
36686	2198.9855580…	52.29.173.150	10.0.2.15	MQTT	204	Publish Message (id=11851) [hospital/room2/room0/deposit]
36557	2195.5240400…	52.29.173.150	10.0.2.15	MQTT	202	Publish Message (id=11701) [hospital/room2/room0/light]
36468	2193.8247184…	3.65.137.17	10.0.2.15	MQTT	209	Publish Message (id=1351) [metaverse/room2/section2/humidity]
36459	2193.6992387…	52.29.173.150	10.0.2.15	MQTT	194	Publish Message (id=6801) [hospital/building3]

The problem is that some of them contain multiple publish message. In particular, the packets shown in the next image contain multiple messages:



883	37.767132508	52.29.173.150	10.0.2.15	MQTT	334	Publish Message (id=101) [hospital/building3], Publish Message (id=102) [hospital/…
576	15.740162875	52.29.173.150	10.0.2.15	MQTT	197	Publish Message (id=51) [metaverse/room1/room3]
520	14.762719557	52.29.173.150	10.0.2.15	MQTT	1238	Publish Message (id=101) [hospital/building3/room0], Publish Message (id=102) [hos…
305	8.676716846	52.29.173.150	10.0.2.15	MQTT	1947	Publish Message [hospital/department6/area2/temperature], Publish Message [hospita…
145	4.668318936	3.65.137.17	10.0.2.15	MQTT	487	Publish Message (id=51) [metaverse/room2/room0/light], Publish Message (id=52) [me…

In this way, the total becomes 73.

But, we have to check that also these other messages are sent using QoS = 1. Doing that, we find 13 packets that are sent with QoS equal to 0 or equal to 2.

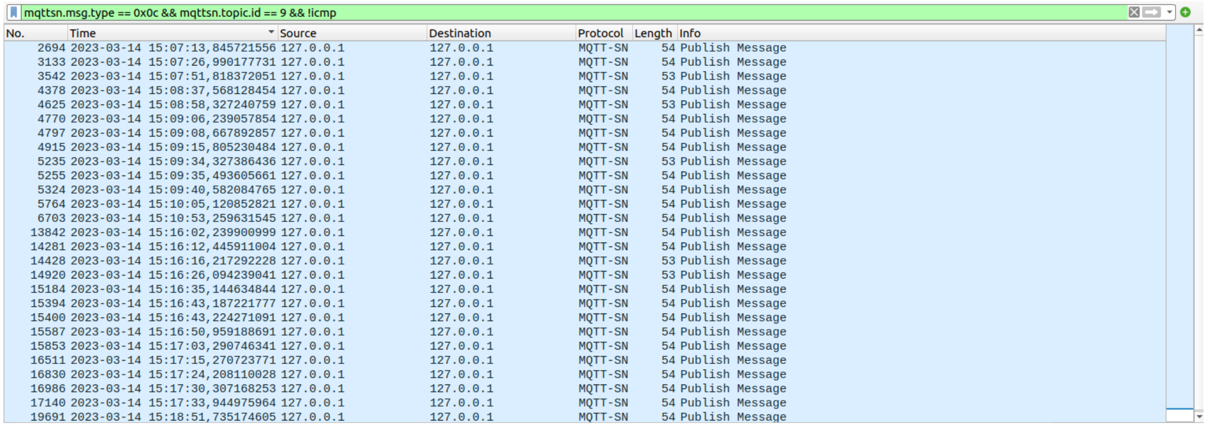
In this way, we get the final answer: 60 packets.

Question 6. How many MQTT-SN (on port 1885) publish messages sent after the hour 3.16PM (Milan Time) are directed to topic 9? Are these messages handled by the server? **(0.1 pts)**

Answer a: 15

Answer b: not handled by the server

After having set that all the packets that use port 1885 use MQTT-SN as communication protocol, we can get the answer using the following filter:



```
mqttsn.msg.type == 0x0c && mqttsn.topic.id == 9 && !icmp
No.     Time                          Source          Destination      Protocol  Length Info
   2694 2023-03-14 15:07:13,845721556 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   3133 2023-03-14 15:07:26,990177731 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   3542 2023-03-14 15:07:51,818372051 127.0.0.1       127.0.0.1        MQTT-SN       53 Publish Message
   4378 2023-03-14 15:08:37,568128454 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   4625 2023-03-14 15:08:58,327240759 127.0.0.1       127.0.0.1        MQTT-SN       53 Publish Message
   4770 2023-03-14 15:09:06,239057854 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   4797 2023-03-14 15:09:08,667892857 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   4915 2023-03-14 15:09:15,805230484 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   5235 2023-03-14 15:09:34,327386436 127.0.0.1       127.0.0.1        MQTT-SN       53 Publish Message
   5255 2023-03-14 15:09:35,493605661 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   5324 2023-03-14 15:09:40,582084765 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   5764 2023-03-14 15:10:05,120852821 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
   6703 2023-03-14 15:10:53,259631545 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  13842 2023-03-14 15:16:02,239900999 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  14281 2023-03-14 15:16:12,445911004 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  14428 2023-03-14 15:16:16,217292228 127.0.0.1       127.0.0.1        MQTT-SN       53 Publish Message
  14920 2023-03-14 15:16:26,094239041 127.0.0.1       127.0.0.1        MQTT-SN       53 Publish Message
  15184 2023-03-14 15:16:35,144634844 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  15394 2023-03-14 15:16:43,187221777 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  15400 2023-03-14 15:16:43,224271091 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  15587 2023-03-14 15:16:50,959188691 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  15853 2023-03-14 15:17:03,290746341 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  16511 2023-03-14 15:17:15,270723771 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  16830 2023-03-14 15:17:24,208110028 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  16986 2023-03-14 15:17:30,307168253 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  17140 2023-03-14 15:17:33,944975964 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
  19691 2023-03-14 15:18:51,735174605 127.0.0.1       127.0.0.1        MQTT-SN       54 Publish Message
```

The first filter stands for “PUBLISH” messages, the second one identifies those messages that are sent to the topic 9. Finally, the third one is introduced because there were 15 ICMP packets that have an error (unreachable destination).

Keeping only the MQTT-SN packets that satisfied the constraints, we get a total amount of 15 packets (the filter gives more than 15 packets, but counting from the packet that has 15:16:02 as time will give a count of 15 packets).

They are not handled by the server because the destination in unreachable, since for each of those packets we have also an error message included into the ICMP packets mentioned before.