

IoT 2023 CHALLENGE 1

- (1) Name: Nicolò Pisanu Person Code: 10827469
- (2) Name: Marco Viviani Person Code: 10528650

Referring to the file [challenge2023_1.pcap](#), answer the following questions:

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)

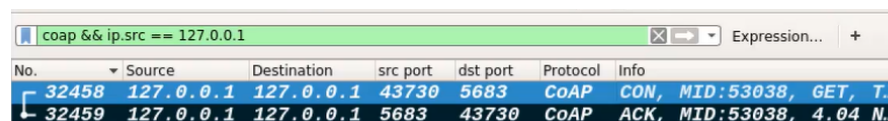
We didn't consider ICMP and TCP messages because they're related to messages that never reached the destination or the port (and not related with the non existing resources).

a) **not_ws.malformed && coap.code==132 && ip.src==127.0.0.1**

We obtained 38 not-found (error 4.04) server responses to the requests directed to non-existent resources. However, not all these 38 responses come from GETs requests. We filtered all the GET requests:

not_ws.malformed && coap.code == 1 && ip.dst == 127.0.0.1

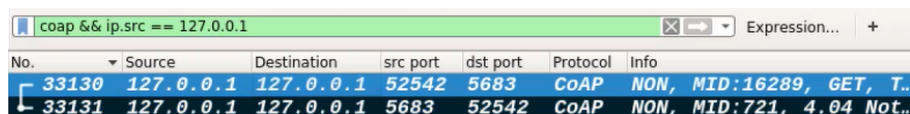
and checked which of them have a related 4.04 error code response.



No.	Source	Destination	src port	dst port	Protocol	Info
32458	127.0.0.1	127.0.0.1	43730	5683	CoAP	CON, MID:53038, GET, T...
32459	127.0.0.1	127.0.0.1	5683	43730	CoAP	ACK, MID:53038, 4.04 N...

ANSWER: 11

- b) Of the above 11 GET requests, is possible to check if they are CON or NON type.



No.	Source	Destination	src port	dst port	Protocol	Info
33130	127.0.0.1	127.0.0.1	52542	5683	CoAP	NON, MID:16289, GET, T...
33131	127.0.0.1	127.0.0.1	5683	52542	CoAP	NON, MID:721, 4.04 Not...

ANSWER: 6

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)

a) **(not_ws.malformed && coap.code==DELETE && ip.dst==134.102.218.18)**

In this way we find 115 messages DELETE to the coap.me server.

Then we subtract the number of the successful DELETE (we decided to consider as successful only the requests that obtained response code 2.02):

not_ws.malformed && ip.src_host==134.102.218.18 && coap.code==66

We find 10 of these messages. So we subtract these 10 messages to the starting 115 DELETE messages and we obtain 105 failed DELETE requests (we considered also multiple requests with the same MID)

ANSWER: 105

- b) **not_ws.malformed && coap.code==DELETE && ip.dst==134.102.218.18 && coap.opt.uri_path=="hello"**

From this filter we obtain 5 DELETE requests directed to the “/hello” resource; by comparing the MIDs of these requests with the ones of the previous successful 10 (point a) responses we can notice that none of the 5 DELETE requests were successful.

ANSWER: 5

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)

a) **not_ws.malformed && mqtt.msgtype==8 && ip.addr==91.121.93.94**

We obtained 13 subscriptions sent by several clients. To find out these clients we looked for their *tcp.src_port*. These 13 messages are sent by 3 clients: 43133, 51531, 43133.

ANSWER: 3

- b) To answer to this question we looked for ALL the subscriptions of the above 3 clients, not only the 13 messages of point (a):

mqtt.msqtpe==8 && ip.dst==91.121.93.94 && tcp.srcport in {43133 51531 43133}

We obtained 21 subscriptions. By looking at the topics we observed that only 2 of the 3 starting clients **WOULD** receive a publish message issued to the topic “hospital/room2/area0”

ANSWER: 2

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)

a) **mqtt.willtopic contains university**

Both of them have “university” as first level in the topic

ANSWER: 2

- b) The broker doesn’t publish will messages because the client does not die. So no client publishes the will messages.

ANSWER: 0

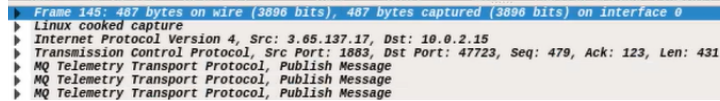
5. How many Publish messages with QoS = 1 are received by the MQTT clients connected to the HiveMQ broker with MQTT version 5? (0.1 pts)

a) **mqtt.ver==5 && (ip.addr==52.29.173.150 || ip.addr==3.65.137.17)**

With this filter we obtain the 9 clients (36665, 37401, 42827, 45635, 46967, 47549, 47723, 57265, 60609) connected to HiveMQ broker with MQTT version 5.

mqtt.qos==1 && tcp.port in {36665 37401 42827 45635 46967 47549 47723 57265 60609} && not _ws.malformed && (ip.src_host==52.29.173.150 || ip.src_host==3.65.137.17)

With this filter we obtain 51 packets received by the 9 clients. Some of these packets contain multiple messages with several QoS levels. For example, the packet 145 contains 3 MQTT publish messages, all with QoS=1:



Frame 145: 487 bytes on wire (3896 bits), 487 bytes captured (3896 bits) on interface 0
Linux cooked capture
Internet Protocol Version 4, Src: 3.65.137.17, Dst: 10.0.2.15
Transmission Control Protocol, Src Port: 1883, Dst Port: 47723, Seq: 479, Ack: 123, Len: 431
MQ Telemetry Transport Protocol, Publish Message
MQ Telemetry Transport Protocol, Publish Message
MQ Telemetry Transport Protocol, Publish Message

For a total of 73 publish messages, of which only 60 have QoS=1.

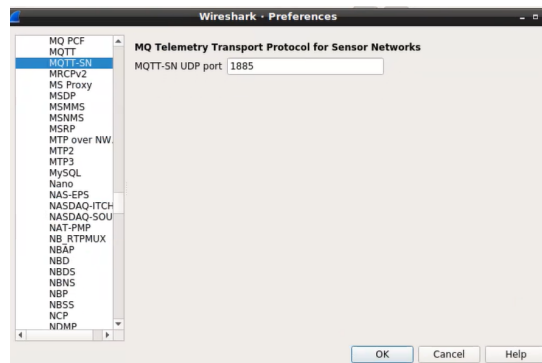
Indeed with the following filter we find exactly 60 PUB ACK:

mqtt.msgtype==4 && tcp.srcport in {36665 37401 42827 45635 46967 47549 47723 57265 60609}

ANSWER: 60

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)**

a) In the Wireshark's preferences we set as MQTT-SN UDP port the 1885.



So we can apply this filter:

not icmp && mqttsn && frame.time >= "Mar 14, 2023 15:16:00" && mqttsn.topic.id==9

ANSWER: 15

b) They are not handled by the server because the port is unreachable.