IoT 2023 CHALLENGE 1

(1) Name: Fontana Nicolo' Person Code: 10581197

(2) Name: Gerosa Andrea Person Code: 10583298

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)**

   Assumptions:

   - We can know whether a resource requested in a GET request is present or not only if there is a response from the server with code 4.04 ("Not Found")

   We used a python script (attached as c1q1.py)

   We filtered the packets retaining only the ones using CoAP

   We filtered them to obtain the GET requests (CoAP code 1) directed to the local server (IP address 127.0.0.1)

   For each GET request, under the assumption (checked manually using Wireshark) that the capture provided to us has the packets ordered, we searched the subsequent packets to find the response of the server and we counted it only if it was a "4.04 – Not Found" (CoAP code 132) message

   If the GET request was of Non-Confirmable type (CoAP type 1) we counted them with a separate counter

   The results we obtained are **11 GET requests directed to non-existing resources in the local CoAP server, 6 of which are of Non-Confirmable type**

---

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)**

   Assumptions:

   - We consider a successful result for a DELETE request only if there is a related response with code 2.02 ("Delete")
   - We consider a DELETE request directed to the "/hello" resource if at least one of the resources addressed by the request is the "/hello" one
   - We consider

   We used Wireshark filters

   dns.qry.name=="coap.me"

   Filter to get IP address of "coap.me" server (134.102.218.18) *ip.dst==134.102.218.18 and coap.code==4*

Filter to get all DELETE requests directed to “coap.me” (115 DELETE requests)

ip.src==134.102.218.18 and coap.code==66

Filter to get all responses from server with 2.02 code (“Deleted”) (10 successful responses to DELETE)

NB: none of them is over the "/hello" resource

We made the difference between all the DELETE requests and the successful ones to get the unsuccessful ones (**105 unsuccessful DELETE**)

ip.dst==134.102.218.18 and coap.code==4 and coap.opt.uri_path=="hello"

Filter to get all DELETE requests directed to the “/hello” resource (5 DELETE requests over the "/hello" resource)

Since we already found that none of the 10 successful DELETE requests was directed to the “/hello” resource we can ensure that every DELETE request directed to the “/hello” resource failed; hence there are **5 not successful DELETE requests directed to the “/hello” resource**

---

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)**

Assumptions:

- We distinguish clients based on IP address and TCP port

We found the IP address of the mosquitto broker with the Wireshark filter *dns.qry.name contains “mosquitto”* (91.121.93.94)

NB: the mosquito broker has also an IPv6 address (2001:41d0:1:925e::1), but we checked with Wireshark that it is never used

We used a python script (attached as c1q3.py)

We filtered the packets retaining only the ones using MQTT

We filtered them to get the SUBSCRIBE requests directed to the mosquitto broker

For each SUBSCRIBE we checked if the subscription is completed by looking for the relative ack

For each completed subscription we checked, for each topic, if it contained at least 1 single-level wildcard and we uniquely saved (using the python built-in function set()) the client which made the SUBSCRIBE request

We found **3 different clients subscribed to the public broker mosquitto**

We found all clients, among those found before, which ones subscribed to any topic and, for each of these topics, if they match (using MQTT wildcards rules) the topic “hospital/room2/area0”

We found **2 clients which would receive a publish message issued to the “hospital/room2/area0” topic**

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)**

   Assumptions:

   - We consider that a client specifies a Last-Will message even if the connection should not complete (i.e. we do not check if the CONNECT requests receive the CONNACK from the server)
   - We assume that the content of a will-message can be considered unique, due to the relatively low amount of packets we collected (39537) and the apparent randomicity of the contents. Because of this assumption, we do not check the topic of publication when looking for the PUBLISH message containing a will-message
   - We distinguish clients based on IP address and TCP port

   We used a python script (attached as c1q4.py)

   We filtered the packets retaining only the ones using MQTT

   We checked all CONNECT requests containing “university” in the first level of the will-topic field and stored the values of those will-messages

   We found **2 different clients specifying a Last-Will message over a topic with “university” as the first level**

   We checked which PUBLISH message contained those will-messages

   We found that **0 Last-Will messages have been published**

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)**

   Assumptions:

   - Given a client with the desired characteristics (connected to HiveMQ with MQTT version 5), we consider **only** the PUBLISH messages sent to it **after** the beginning of the connection with HiveMQ, by **any** device (not only the ones from HiveMQ)
   - We consider a PUBLISH message to be received by the client if it is sent by the broker and captured in the file given to us. We do not check any ack in response to those PUBLISH messages, since it would be the case in which the messages are received (i.e. the focus of the question), but the device performing the capture did not respond back
   - We do not consider edge cases given by differentiating between persistent and non-persistent connections
   - We do not consider edge cases given by clients disconnecting and/or reconnecting from/to HiveMQ (once they connect performing the whole two-way handshake with

MQTTConnect and MQTTConnack they are considered connected and capable of receiving packets)

- If in a single packet there are multiple PUBLISH messages we count each of them, considering their own QoS

We used Wireshark filters

dns.qry.name contains "hivemq"

Filter to get IP address of "broker.hivemq.com" (52.29.173.150 or 3.65.137.17)

(mqtt.msgtype == 1 or mqtt.msgtype==2) and (ip.dst==52.29.173.150 or ip.dst==3.65.137.17) and mqtt.ver==5

Filter to get all CONNECT and CONNACK messages exchanged with the HiveMQ broker over version 5 of MQTT (9 successful connections with clients using 1 IP and 9 TCP ports: we considered 9 different clients)

mqtt.msgtype==3 and mqtt.qos==1 and ip.dst==10.0.2.15 and (tcp.port==47723 or tcp.port==60609 or tcp.port==57265 or tcp.port==36665 or tcp.port==45635 or tcp.port==37401 or tcp.port==46967 or tcp.port==47549 or tcp.port==42827)

Filter to get all PUBLISH messages sent to any of the 9 different found clients where at least one of the messages in the packet has QoS=1 (51 packets, 73 PUBLISH messages in total)

We manually checked how many of those PUBLISH messages actually had QoS=1 (**60 PUBLISH messages are actually received by clients connected to the public HiveMQ broker**)

---

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)**

Assumptions:

- We consider all the packets sent on March 14th, 2023, since we manually check this on Wireshark (first message captured on at 2023-03-14 15:03:33.996283490, last message captured at 2023-03-14 15:43:41.201463787)

We used Wireshark filters

mqttsn.msg.type == 0xc and mqttsn.topic.id==9 and frame.time >= "Mar 14, 2023 15:16:00" and not icmp

Filter to get all MQTT-SN PUBLISH messages directed to the topic with ID=9 and sent after 3.16 PM of the 14th of March 2023

We found **15 PUBLISH messages**

No, the server does not handle them because all the ICMP responses to them are of type=3 (Destination Unreachable) and have code=3 (Port Unreachable)