POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

IoT Challenge #2, Packet Sniffing

Internet of Things

Authors: **Kevin Ziroldi - 10764177**
Matteo Volpari - 10773593

Professors: Alessandro Redondi, Fabio Palmese
Academic Year: 2024-2025
Version: 1.0
Release date: 6-4-2025

Contents

1 | Packet sniffing PCAP file

1.1. CQ1

Question

How many different Confirmable PUT requests obtained an unsuccessful response from the local CoAP server?

Answer

TODO

Explanation

Domanda 1 Confirmable put request

```
coap && coap.type == 0 && coap.code == 3
```

45 frames Response

```
coap && (coap.code >= 128)
```

228 frames Dovrei matcharli per token o message id (quale????), troppi. In realtà posso filtrare anche ip src = ip dst. Quindi pyshark.

1.2. CQ2

Question

How many CoAP resources in the coap.me public server received the same number of unique Confirmable and Non Confirmable GET requests?
Assuming a resource receives X different CONFIRMABLE requests and Y different NON-CONFIRMABLE GET requests, how many resources have X=Y, with X>0?

Answer

TODO

Explanation

Domanda 2 Get request confirmable a coap.me

```
coap.type == 0 && coap.code == 1 && ip.dst==134.102.218.18
```

39 frames Get non confirmable a coap.me

```
coap.type == 1 && coap.code == 1 && ip.dst==134.102.218.18
```

31 frames Dovrei vedere a quale risorsa fanno riferimento e poi confrontare. Troppo, quindi pyshark.

1.3. CQ3

Question

How many different MQTT clients subscribe to the public broker HiveMQ using multi-level wildcards?

Answer

The number of clients who subscribe to the public broker HiveMQ using multi-level wildcards is 4.

Explanation

In order to find the IP address of the HiveMQ broker, we filter the response of the DNS server using the following Wireshark filter:

```
dns.qry.name == "broker.hivemq.com"
```

All DNS responses return 3 addresses: 18.192.151.104, 35.158.34.213 and 35.158.43.69.

We use a second filter to find SUBSCRIBE messages, with message type 8, sent to HiveMQ broker, to one of the IP addresses found above, with a multi-level wildcard, ending with "#":

```
mqtt && mqtt.msgtype == 8 &&
(ip.dst == 18.192.151.104 || ip.dst == 35.158.34.213
```

```
|| ip.dst == 35.158.43.69) && mqtt.topic contains "#"
```

We find out that HiveMQ broker receives 6 messages of this type, all at the IP address 18.192.151.104.

Since the question asks for the number of MQTT clients who subscribe, we need to identify the clients who sent these messages. For each message, we select the TCP stream, which identifies the client.

Message number	TCP stream
375	8
2442	15
3293	20
3303	15
3362	3
3693	15

Table 1.1: TCP streams

Since there are 4 TCP streams, the 6 messages have been sent by 4 different client.
We can also find the Client ID of these clients by finding the CONNECT message, of type 1, they sent to the broker. For the TCP stream 8, we can use the following filter:

```
mqtt && mqtt.msgtype == 1 && tcp.stream == 3
```

The same filter with different TCP stream can be used for other clients.

TCP stream	Client ID
3	cpoepjzkhibxgjiu
8	dzcxnwdqef
15	tukvxesuhe
20	fcthvjikxjul

Table 1.2: Client IDs table

1.4. CQ4

Question

How many different MQTT clients specify a Last Will Message to be directed to a topic having as first level "university"?

Answer

The number of clients who specify a Last Will Message to be directed to a topic having as first level "university" is 1.

Explanation

MQTT clients can specify a Last Will Message in the CONNECT message. In order to find the described messages, we filter CONNECT messages, of type 1, with a Last Will Topic:

```
mqtt && mqtt.msgtype == 1 && mqtt.willtopic
```

We find four messages, but only one of them has a Last Will Topic having as first level "university".
We can find the result by enriching the filter and avoiding manually checking the topics, using the following filter:

```
mqtt && mqtt.msgtype == 1 && mqtt.willtopic matches "^university"
```

Using this filter, we directly get the only message asked by CQ4.

1.5. CQ5

Question

How many MQTT subscribers receive a last will message derived from a subscription without a wildcard?

Answer

TODO

Explanation

Domanda 5:

```
mqtt && mqtt.msgtype== 1 && mqtt.willmsg
```

Ottengo 4 fram di tipo CONNECT che specificano un last will message e last will topic.

4 0.000117188 ::1 ::1 MQTT 176 Connect Command 196 2.116585177 10.0.2.15 5.196.78.28 MQTT 126 Connect Command 352 5.034840089 10.0.2.15 5.196.78.28 MQTT 123 Connect Command 557 7.043177949 10.0.2.15 5.196.78.28 MQTT 120 Connect Command

1 Will Topic: university/department12/room1/temperature Will Message Length: 29 Will Message: 6572726f723a20612056495020436c69656e74206a7573742064696564 2 Will Topic: metaverse/room2/floor4 Will Message Length: 15 Will Message: 6572726f723a20706575697664716c
3 Will Topic: hospital/facility3/area3 Will Message Length: 15 Will Message: 6572726f723a20786c7a6
4 Will Topic: metaverse/room2/room2 Will Message Length: 15 Will Message: 6572726f723a207a6a7a

Per il primo: Messaggi pubblicati su quel topic:

```
mqtt && mqtt.msgtype==3 && mqtt.topic == "university/department12/room1/temperature
```

4 risultati Messaggi pubblicati su quel topic con messaggio uguale a last will message:

```
mqtt && mqtt.msgtype==3 && mqtt.topic == "university/department12/room1/temperature
```

6572726f723a20612056495020436c69656e74206a7573742064696564 Stessi 4 risultati Però non ha retain true!

```
mqtt && mqtt.msgtype==3 && mqtt.topic == "university/department12/room1/temperature
6572726f723a20612056495020436c69656e74206a7573742064696564 && mqtt.retain == 1
```

Nessun risultato

Devono avere il retain? Se sì, allora non sono last will, altrimenti??? Non penso. Il retain dovrebbe essere sul messaggio di connect, non su quello di last will!

Inoltre sono i pacchetti 6560, 6562, 6564, 6566. A 6559 c'è un reset ??. Si tratta di questi pacchetti 6559 146.691800286 ::1 ::1 TCP 88 38083 → 1883 [RST, ACK] Seq=9316 Ack=85 Win=65536 Len=0 TSval=2654936537 TSecr=2654934931 6560 146.692096889 ::1 ::1 MQTT 162 Publish Message [university/department12/room1/temperature] 6561 146.692172650 ::1 ::1 TCP 88 39551 → 1883 [ACK] Seq=77 Ack=86 Win=65536 Len=0 TSval=2654936537 TSecr=2654936537 6562 146.692187516 ::1 ::1 MQTT 164 Publish Message (id=1) [university/department12/room1/temperature] 6563 146.692189976 ::1 ::1 TCP 88 53557 → 1883 [ACK] Seq=94 Ack=86 Win=65536 Len=0 TSval=2654936537 TSecr=2654936537 6564 146.692199911 ::1 ::1 MQTT 165 Publish Message (id=12) [university/department12/room1/temperature] 6565 146.692202117 ::1 ::1 TCP 88 51743 → 1883 [ACK] Seq=573 Ack=14152 Win=64896 Len=0 TSval=2654936537 TSecr=2654936537 6566 146.692209983 ::1 ::1 MQTT 164 Publish Message (id=1) [university/department12/room1/temp

6560, 6562, 6564 e 6466 sono quelli con contenuto pari al last will message sul last will topic. Per questi pacchetti, ho notato che la source port è sempre 1883, mentre la destination port cambia. Infine il tcp stream cambia.

Devo controllare se si sono iscritti con wildcard, seguo il tcp stream e identifico le richieste di SUBSCRIBE.

6560 Destination Port: 39551 [Stream index: 2]

```
mqtt && mqtt.msgtype == 8 && tcp.stream == 2
```

Non usa wildcard

```
mqtt && mqtt.msgtype == 1 && tcp.stream == 2
```

Client ID: auyvhrhdudnm

6562 Destination Port: 53557 [Stream index: 6]

```
mqtt && mqtt.msgtype == 8 && tcp.stream == 6
```

Non usa wildcard

```
mqtt && mqtt.msgtype == 1 && tcp.stream == 6
```

Client ID: pcdwkgeslfh

6564 Destination Port: 51743 [Stream index: 10]

```
mqtt && mqtt.msgtype == 8 && tcp.stream == 10
```

Molte subscription: una sola che va bene per il nostro topic, con wildcard

```
1136 10.102492445 ::1 ::1 MQTT 108 Subscribe Request (id=6) [university/#]
```

Destination Port: 41789 [Stream index: 14]

```
mqtt && mqtt.msgtype == 8 && tcp.stream == 14
```

Non usa wildcard

```
mqtt && mqtt.msgtype == 1 && tcp.stream == 14
```

Client ID: mjdocmjxt

RST è un reset, indica disconnessione brusca. Successivamente il broker, sempre porta 1883, manda messaggi tutti con last will topic e last will message in loopback, sempre a porte diverse. Secondo me sono dei last will messages.

Per quelli da 2 a 4: 5.196.78.28 è src solamente per PUBACK, PUBREC, CONNACK. Filtro che estrae i messaggi pubblicati:

```
mqtt && ip.src == 5.196.78.28 && mqtt.msgtype == 3
```

non ritorna risultati Se non ha pubblicato nulla, non può aver mandato dei last will message.

Conclusione: ci sono 4 CONNECT con last will, una di queste, quella in locale, ha dei messaggi di last will. Il messaggio viene inviato dopo un reset dal broker a 4 subscriber, vedendo la loro subscription, solo 3 su 4 non hanno usato la wildcard.

References:

`https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html#_Toc3901022`

1.6. CQ6

Question

How many MQTT publish messages directed to the public broker mosquitto are sent with the retain option and use QoS “At most once”?

Answer

TODO

Explanation

Domanda 6:

utilizzando il filtro: dns.qry.name == "test.mosquitto.org" trovo l’indirizzo ip collegato al dominio richiesto che poi mi servirà per filtrare i pacchetti, l’IP è: 5.196.78.28. quindi filtro i pacchetti con: mqtt.msgtype == 3 and mqtt.qos == 0 and mqtt.retain == 1 and ip.dst == 5.196.78.28 e trovo tutti quelli che rispettano la richiesta: 208 pacchetti.

1.7. CQ7

Question

How many MQTT-SN messages on port 1885 are sent by the clients to a broker in the local machine?

Answer

TODO

Explanation

Domanda 7: Il protocollo MQTT-SN non è riconosciuto nativamente da Wireshark ma sappiamo da teoria che è un protocollo udp. Filtrando quindi con udp.port == 1885 non trovo nessun pacchetto e quindi non sono stati inviati pacchetti sulla porta 1885.