

Internet Of Things (IOT) - Challenge 2

Mohammadreza Zamani (10869960) – Asal Abbasnejadfar (10974178)

Packet Sniffing

Part 1 Questions on the PCAP file

CQ1) How many different Confirmable PUT requests obtained an unsuccessful response from the local CoAP server?

Answer: 22

In the first step to solve this question, we should separate different confirmable put requests from others. We use this filter `coap.type == 0 && coap.code == 3`.

code == 3. `coap.type == 0` for separating confirmable requests and `coap.code == 3` for put requests. This filter returns 45 packets, representing all Confirmable PUT requests.

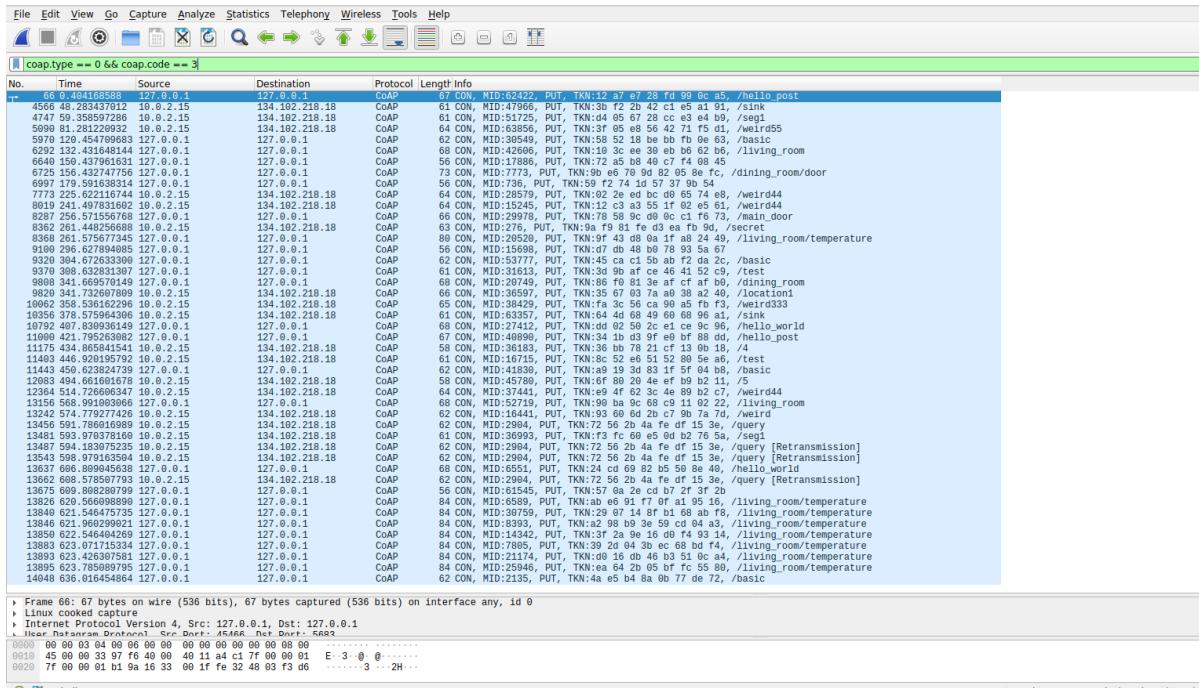


Fig. 1.

Then, we should only consider an unsuccessful response from the local CoAP server. It means their destination address should be 127.0.0.1.

Then, we should consider token number for each pocket in the info section. For example, for the packet number 66, we use this filter `coap.token == 12:a7:e7:28:fd:99:0c:a5` to find which confirmable put requests obtain an unsuccessful response. This filter helps us to see responses related to this token. In the first line of the figure below, you can see the request and in the second line the response to this request. Code 2.04 according to the CoAP standard means a successful response. So, we should not consider this response.

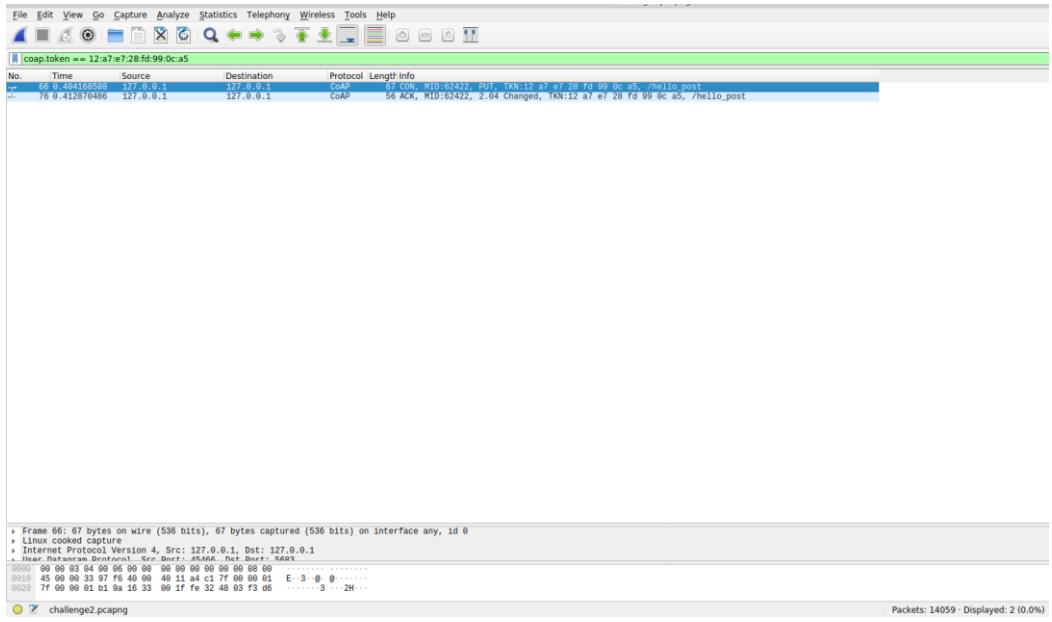


Fig. 2.

For packet number 6725, we obtain 4.04 not found and the response is unsuccessful.

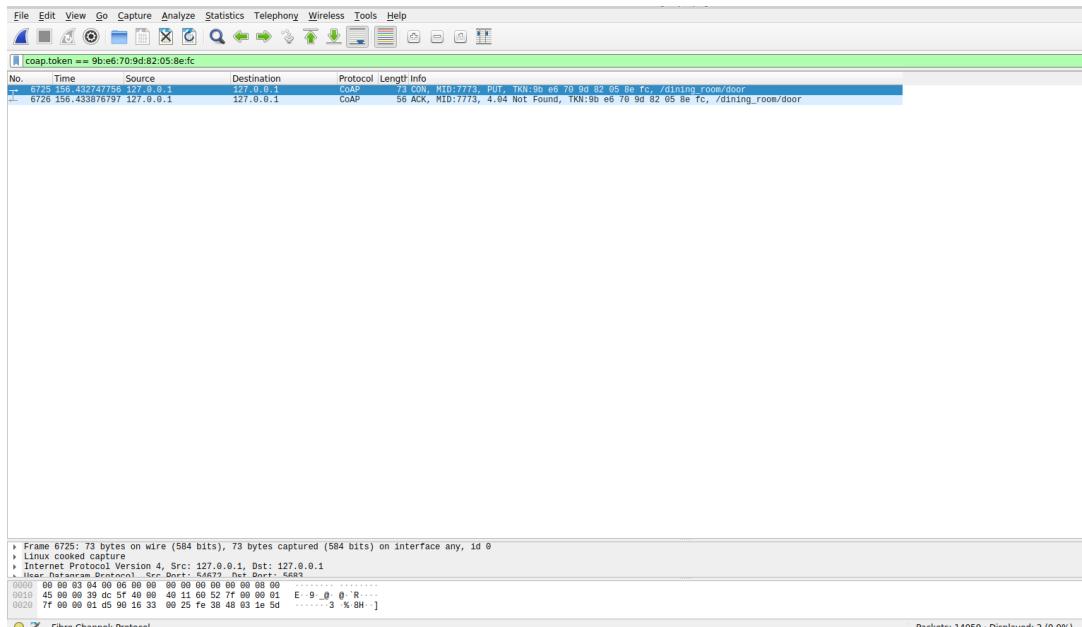


Fig. 3.

For packet number 10792, we obtain 4.05 method not allowed and the response is unsuccessful.

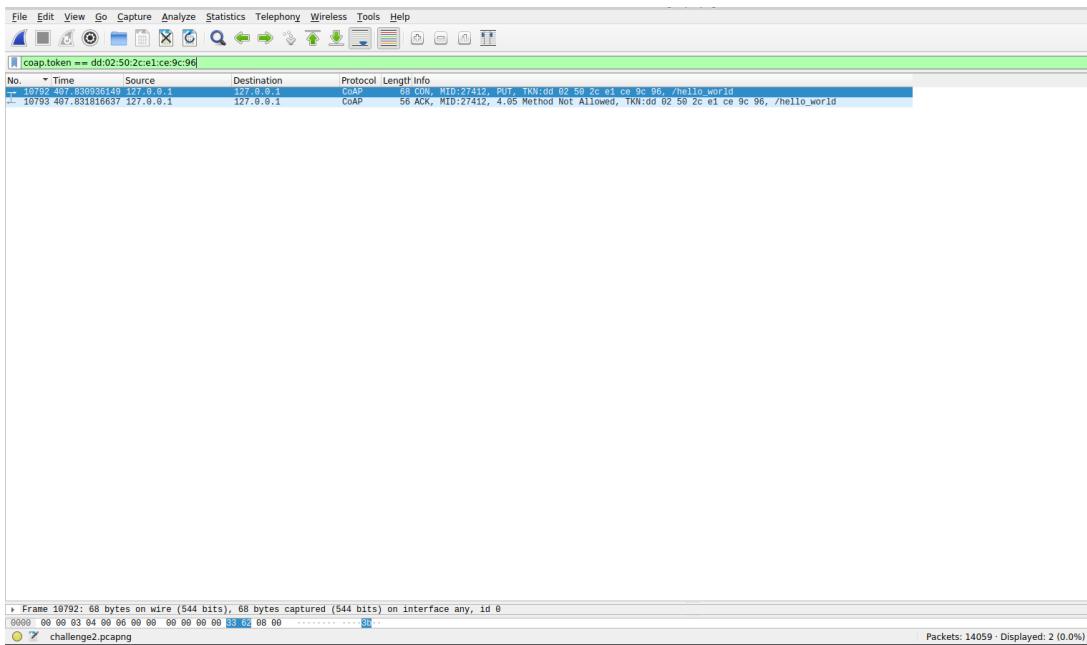


Fig. 4.

We do this process for all packets and finally find 22 different Confirmable PUT requests obtained an unsuccessful response from the local CoAP server.

CQ2) 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 NONCONFIRMABLE GET requests, how many resources have **X=Y**, with **X>0**?

Answer: 3

First, we use this filter `coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18`.

We know that The IP address of the coap.me server is 134.102.218.18. also, `coap.type == 0` use to filter confirmable request and `coap.code == 1` use to filter get requests.

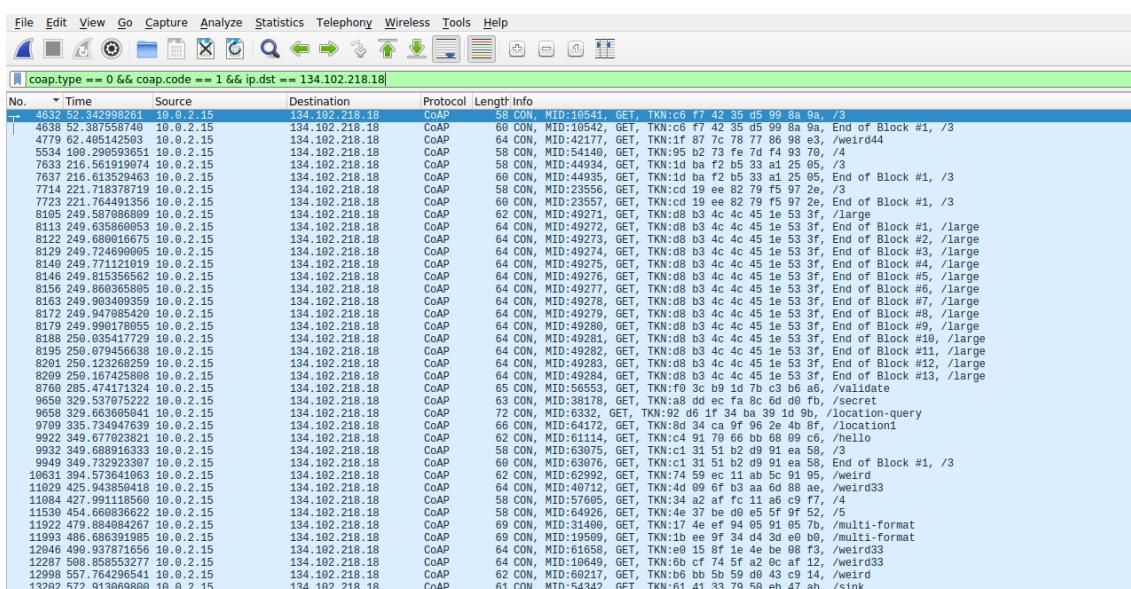


Fig. 5.

Then, we write from the info column the resource paths. The resource paths For confirmable get requests are:

- 1- /3
- 2- /weird44
- 3- /4
- 4- /large
- 5- /5
- 6- /secret
- 7- /validate
- 8- /location-query
- 9- /location1
- 10- /hello
- 11- /weird
- 12- /weird33
- 13- /multi-format
- 14- /sink

Next, we use this filter for non-confirmable request

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

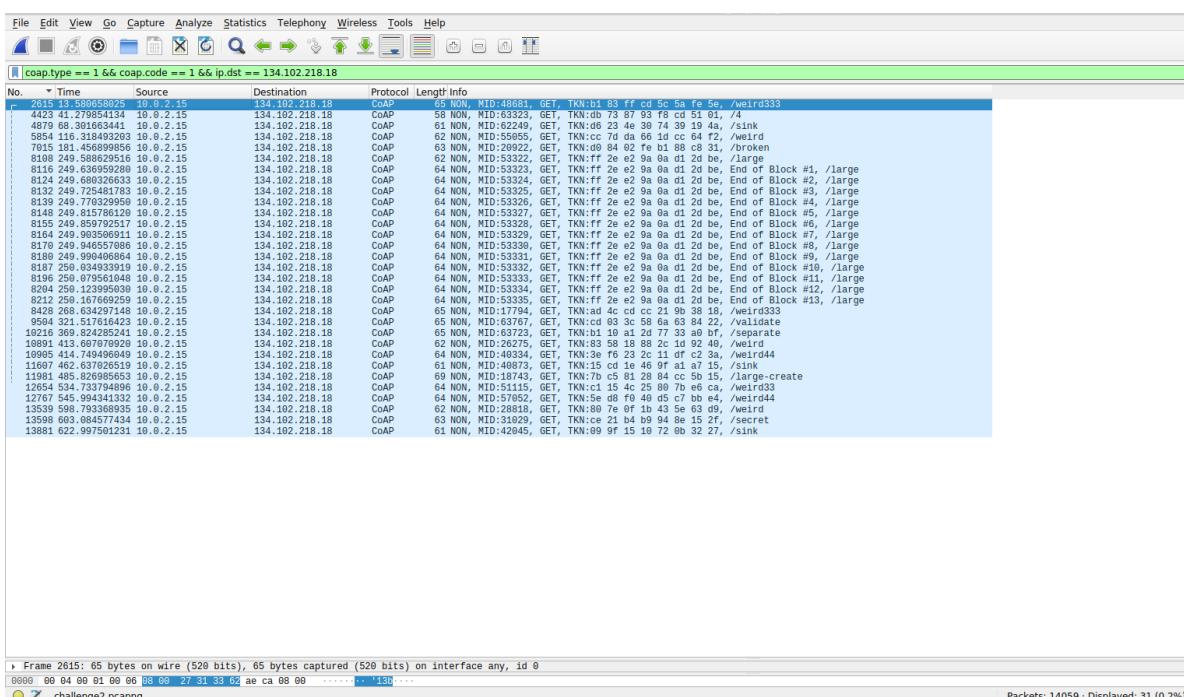


Fig. 6.

The resource paths For non-confirmable get requests are:

1. /weird333
2. /weird44
3. /4
4. /sink
5. /weird
6. /broken
7. /large
8. /validate
9. /separate
10. /weird33
11. /large-create
12. /secret

The number of resources that received both Confirmable and Non-confirmable GETs (with X=Y and X>0) is 8. These lists are:

1. /weird44
2. /4
3. /sink
4. /weird
5. /large
6. /validate
7. /weird33
8. /secret

Now, we should create two separate filters for each path above:

For /weird44 path we set this filter `coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "weird44"` for confirmable get and this filter `coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "weird44"` for non-confirmable get.

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "weird44"					
No.	Time	Source	Destination	Protocol	Length Info
4779	62.405142503	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:42177, GET, TKN:1f 87 7c 78 77 86 98 e3, /weird44

In the image above, there is only one packet, which shows that a confirmable GET request was sent to the resource /weird44.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "weird44"					
No.	Time	Source	Destination	Protocol	Length Info
10905	414.749496049	10.0.2.15	134.102.218.18	CoAP	64 NON, MID:40334, GET, TKN:3e f6 23 2c 11 df c2 3a, /weird44
12767	545.994341332	10.0.2.15	134.102.218.18	CoAP	64 NON, MID:57052, GET, TKN:5e d8 f0 40 d5 c7 bb e4, /weird44

In the image above, there are two packets, which show that a non-confirmable GET request was sent to the resource /weird44. So, X is not equal to Y.

For /4 path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/4"					
No.	Time	Source	Destination	Protocol	Length Info
5534	100.290593651	10.0.2.15	134.102.218.18	CoAP	58 CON, MID:54140, GET, TKN:95 b2 73 fe 7d f4 93 70, /4
11084	427.991118560	10.0.2.15	134.102.218.18	CoAP	58 CON, MID:57605, GET, TKN:34 a2 af fc 11 a6 c9 f7, /4

There are two packets which show that a confirmable GET request was sent to the resource /4.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/4"					
No.	Time	Source	Destination	Protocol	Length Info
4423	41.279854134	10.0.2.15	134.102.218.18	CoAP	58 NON, MID:63323, GET, TKN:db 73 87 93 f8 cd 51 01, /4

There is one packet which shows that a confirmable GET request was sent to the resource /4. X is not equal to Y.

For /sink path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/sink"					
No.	Time	Source	Destination	Protocol	Length Info
13202	572.913069800	10.0.2.15	134.102.218.18	CoAP	61 CON, MID:54342, GET, TKN:61 41 33 79 50 eb 47 ab, /sink

There is one packet which shows that a confirmable GET request was sent to the resource /sink.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/sink"					
No.	Time	Source	Destination	Protocol	Length Info
4879	68.301663441	10.0.2.15	134.102.218.18	CoAP	61 NON, MID:62249, GET, TKN:d6 23 4e 30 74 39 19 4a, /sink
11607	462.637026519	10.0.2.15	134.102.218.18	CoAP	61 NON, MID:40873, GET, TKN:15 cd 1e 46 9f a1 a7 15, /sink
13881	622.997501231	10.0.2.15	134.102.218.18	CoAP	61 NON, MID:42045, GET, TKN:09 9f 15 10 72 0b 32 27, /sink

There are three packets which show that a confirmable GET request was sent to the resource /sink. X is not equal to Y.

For /weird path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/weird"					
No.	Time	Source	Destination	Protocol	Length Info
10631	394.573641063	10.0.2.15	134.102.218.18	CoAP	62 CON, MID:62992, GET, TKN:74 59 ec 11 ab 5c 91 95, /weird
12998	557.764296541	10.0.2.15	134.102.218.18	CoAP	62 CON, MID:60217, GET, TKN:b6 bb 5b 59 d0 43 c9 14, /weird

There are two packets which show that a confirmable GET request was sent to the resource /weird.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/weird"					
No.	Time	Source	Destination	Protocol	Length Info
5854	116.318493203	10.0.2.15	134.102.218.18	CoAP	62 NON, MID:55055, GET, TKN:cc 7d da 66 1d cc 64 f2, /weird
10891	413.607076920	10.0.2.15	134.102.218.18	CoAP	62 NON, MID:26275, GET, TKN:83 58 18 88 2c 1d 92 40, /weird
13539	598.793368935	10.0.2.15	134.102.218.18	CoAP	62 NON, MID:28818, GET, TKN:80 7e 0f 1b 43 5e 63 d9, /weird

There are three packets which show that a confirmable GET request was sent to the resource /weird. X is not equal to Y.

For /large path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "/large"					
No.	Time	Source	Destination	Protocol	Length Info
8105	249.587086809	10.0.2.15	134.102.218.18	CoAP	62 CON, MID:49271, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, /large
8113	249.635860053	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49272, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #1, /large
8122	249.680016675	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49273, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #2, /large
8124	249.724690005	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49274, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #3, /large
8146	249.771121019	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49275, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #4, /large
8148	249.813536562	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49276, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #5, /large
8155	249.860365805	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49277, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #6, /large
8163	249.903409359	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49278, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #7, /large
8172	249.947085420	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49279, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #8, /large
8179	249.990178055	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49280, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #9, /large
8188	250.035417729	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49281, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #10, /large
8195	250.079456638	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49282, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #11, /large
8201	250.123268259	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49283, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #12, /large
8209	250.167425808	10.0.2.15	134.102.218.18	CoAP	64 CON, MID:49284, GET, TKN:d8 b3 4c 4c 45 1e 53 3f, End of Block #13, /large

There are 14 packets which show that a confirmable GET request was sent to the resource /large.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "large"						
No.	Time	Source	Destination	Protocol	Length	Info
8108	240.588620516	10.0.2.15	134.102.218.18	CoAP	62	NON, MID:53322, GET, TKN:ff 2e e2 9a 0a d1 2d be, /large
8116	249.636959280	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53323, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #1, /large
8124	249.689326633	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53324, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #2, /large
8132	249.725481783	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53325, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #3, /large
8139	249.770329950	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53326, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #4, /large
8148	249.815786120	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53327, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #5, /large
8155	249.859792517	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53328, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #6, /large
8164	249.903566111	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53329, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #7, /large
8170	249.946556708	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53330, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #8, /large
8188	249.999486864	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53331, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #9, /large
8187	250.034933919	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53332, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #10, /large
8196	250.079561048	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53333, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #11, /large
8204	250.123995030	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53334, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #12, /large
8212	250.167669259	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:53335, GET, TKN:ff 2e e2 9a 0a d1 2d be, End of Block #13, /large

There are 14 packets which show that a confirmable GET request was sent to the resource /large.
X is equal to Y and X>0.

For /validate path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "validate"						
No.	Time	Source	Destination	Protocol	Length	Info
8760	285.474171324	10.0.2.15	134.102.218.18	CoAP	65	CON, MID:56553, GET, TKN:f0 3c b9 1d 7b c3 b6 a6, /validate

There is one packet which shows that a confirmable GET request was sent to the resource /validate.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "validate"						
No.	Time	Source	Destination	Protocol	Length	Info
9504	321.517616423	10.0.2.15	134.102.218.18	CoAP	65	NON, MID:63767, GET, TKN:cd 03 3c 58 6a 63 84 22, /validate

There is one packet which shows that a confirmable GET request was sent to the resource /validate.
X is equal to Y and X>0.

For /secret path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "secret"						
No.	Time	Source	Destination	Protocol	Length	Info
9650	329.537075222	10.0.2.15	134.102.218.18	CoAP	63	CON, MID:38178, GET, TKN:a8 dd ec fa 8c 6d d0 fb, /secret

There is one packet which shows that a confirmable GET request was sent to the resource /secret.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "secret"						
No.	Time	Source	Destination	Protocol	Length	Info
13598	603.084577434	10.0.2.15	134.102.218.18	CoAP	63	NON, MID:31029, GET, TKN:ce 21 b4 b9 94 8e 15 2f, /secret

There is one packet which shows that a confirmable GET request was sent to the resource /secret.
X is equal to Y and X>0.

For /weird33 path:

coap.type == 0 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "weird33"						
No.	Time	Source	Destination	Protocol	Length	Info
11029	425.943850418	10.0.2.15	134.102.218.18	CoAP	64	CON, MID:40712, GET, TKN:4d 09 6f b3 aa 6d 88 ae, /weird33
12046	490.937871656	10.0.2.15	134.102.218.18	CoAP	64	CON, MID:61658, GET, TKN:e0 15 8f 1e 4e be 08 f3, /weird33
12287	508.88553277	10.0.2.15	134.102.218.18	CoAP	64	CON, MID:10649, GET, TKN:6b cf 74 5f a2 0c af 12, /weird33

There are three packets which show that a confirmable GET request was sent to the resource /weird33.
X is equal to Y and X>0.

coap.type == 1 && coap.code == 1 && ip.dst == 134.102.218.18 && coap.opt.uri_path == "weird33"						
No.	Time	Source	Destination	Protocol	Length	Info
12654	534.733794896	10.0.2.15	134.102.218.18	CoAP	64	NON, MID:51115, GET, TKN:c1 15 4c 25 80 7b e6 ca, /weird33

There is one packet which shows that a confirmable GET request was sent to the resource /weird33.
X is not equal to Y.

CQ3) How many different MQTT clients subscribe to the public broker HiveMQ using multi-level wildcards?

Answer: 4

To solve this question, first we use this filter `mqtt.msgtype == 8 && mqtt.topic contains "#".`
 This filter displays only MQTT SUBSCRIBE messages (`mqtt.msgtype == 8`) where the topic contains the multi-level wildcard `#`.

No.	Time	Source	Destination	Protocol	Length	Info
238	3.087766995	::1	::1	MQTT	106	Subscribe Request (id=2) [metaverse/#]
375	5.113641615	10.0.2.15	18.192.151.104	MQTT	80	Subscribe Request (id=3) [university/+/#]
468	6.092938713	::1	::1	MQTT	114	Subscribe Request (id=4) [metaverse/+room1/#]
499	6.106824619	::1	::1	MQTT	117	Subscribe Request (id=4) [university/facility0/#]
563	7.094231915	::1	::1	MQTT	127	Subscribe Request (id=5) [metaverse/department3/section0/#]
1138	18.102492445	::1	::1	MQTT	108	Subscribe Request (id=6) [university/#]
1177	18.112498473	::1	::1	MQTT	114	Subscribe Request (id=7) [hospital/fxLugs/+/#]
1178	18.112541241	::1	::1	MQTT	106	Subscribe Request (id=6) [metaverse/#]
2338	13.118621965	::1	::1	MQTT	119	Subscribe Request (id=4) [university/department3/#]
2368	13.118711456	::1	::1	MQTT	106	Subscribe Request (id=8) [factory/+/#]
2442	13.175483992	10.0.2.15	18.192.151.104	MQTT	87	Subscribe Request (id=5) [university/room0/room1/#]
2994	16.145974080	::1	::1	MQTT	111	Subscribe Request (id=12) [house/+/room1/#]
3276	20.152592426	::1	::1	MQTT	119	Subscribe Request (id=14) [university/department3/#]
3293	20.163621204	10.0.2.15	18.192.151.104	MQTT	79	Subscribe Request (id=10) [house/#]
3363	20.224858918	10.0.2.15	18.192.151.104	MQTT	75	Subscribe Request (id=9) [university/#]
3362	21.206357493	10.0.2.15	18.192.151.104	MQTT	94	Subscribe Request (id=10) [university/building2/section0/#]
3483	23.228257088	::1	::1	MQTT	113	Subscribe Request (id=14) [university/room0/#]
3612	25.177664386	::1	::1	MQTT	126	Subscribe Request (id=12) [university/building2/section0/#]
3693	26.268559277	10.0.2.15	18.192.151.104	MQTT	91	Subscribe Request (id=13) [factory/department3/floor0/#]

We can concern from above image that which clients connect to the HiveMQ public broker. Some clients are using IP ::1 (localhost). This means that some of these clients are not connected to the public broker at all but are connected to their own system. So, we should consider only those clients that connect to IP 18.192.151.104 .

For each of these six packets, we should do the same procedure. We should find different Client ID for these packets.

For packet 375:

Use its source port (38641) and then use this filter `tcp.port == <source port> && mqtt.msgtype == 1` to find Client ID.

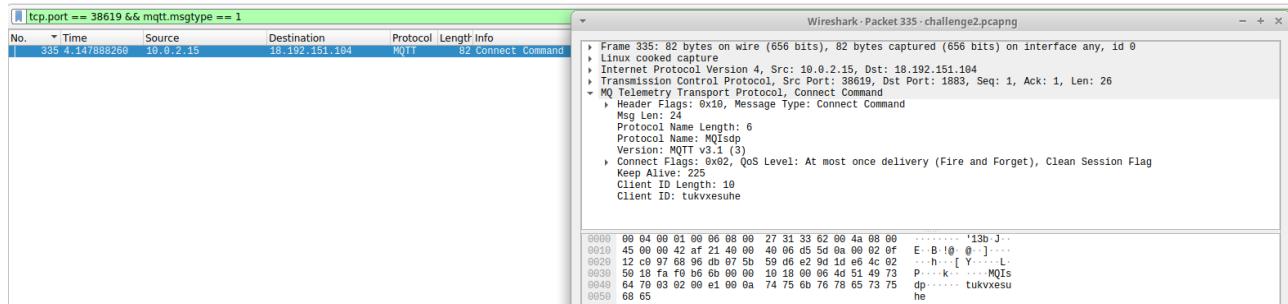
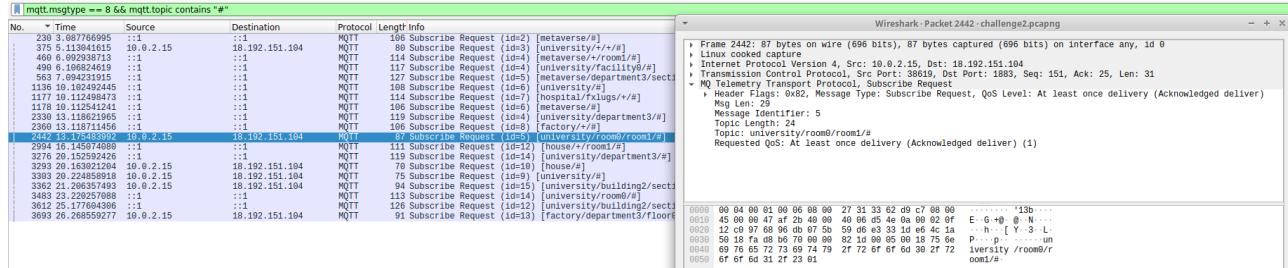
No.	Time	Source	Destination	Protocol	Length	Info
238	3.087766995	::1	::1	MQTT	106	Subscribe Request (id=2) [metaverse/#]
375	5.113641615	10.0.2.15	18.192.151.104	MQTT	80	Subscribe Request (id=3) [university/+/#]
468	6.092938713	::1	::1	MQTT	114	Subscribe Request (id=4) [metaverse/+room1/#]
499	6.106824619	::1	::1	MQTT	117	Subscribe Request (id=4) [university/facility0/#]
563	7.094231915	::1	::1	MQTT	127	Subscribe Request (id=5) [metaverse/department3/section0/#]
1138	18.102492445	::1	::1	MQTT	108	Subscribe Request (id=6) [university/#]
1177	18.112498473	::1	::1	MQTT	114	Subscribe Request (id=7) [hospital/fxLugs/+/#]
1178	18.112541241	::1	::1	MQTT	106	Subscribe Request (id=6) [metaverse/#]
2338	13.118621965	::1	::1	MQTT	119	Subscribe Request (id=4) [university/department3/#]
2368	13.118711456	::1	::1	MQTT	106	Subscribe Request (id=8) [factory/+/#]
2442	13.175483992	10.0.2.15	18.192.151.104	MQTT	87	Subscribe Request (id=5) [university/room0/room1/#]
2994	16.145974080	::1	::1	MQTT	111	Subscribe Request (id=12) [house/+/room1/#]
3276	20.152592426	::1	::1	MQTT	119	Subscribe Request (id=14) [university/department3/#]
3293	20.163621204	10.0.2.15	18.192.151.104	MQTT	79	Subscribe Request (id=10) [house/#]
3363	20.224858918	10.0.2.15	18.192.151.104	MQTT	75	Subscribe Request (id=9) [university/#]
3362	21.206357493	10.0.2.15	18.192.151.104	MQTT	94	Subscribe Request (id=10) [university/building2/section0/#]
3483	23.228257088	::1	::1	MQTT	113	Subscribe Request (id=14) [university/room0/#]
3612	25.177664386	::1	::1	MQTT	126	Subscribe Request (id=12) [university/building2/section0/#]
3693	26.268559277	10.0.2.15	18.192.151.104	MQTT	91	Subscribe Request (id=13) [factory/department3/floor0/#]

No.	Time	Source	Destination	Protocol	Length	Info
130	1.107593686	10.0.2.15	18.192.151.104	MQTT	81	Connect Command
						WireShark - Packet 375 - challenge2.pcapng
						Frame 375: 80 bytes on wire (640 bits), 80 bytes captured (640 bits) on interface any, id 8 Internet Protocol Version 4, Src: 10.0.2.15, Dst: 18.192.151.104 Transmission Control Protocol, Src Port: 38641, Dst Port: 1883, Seq: 74, Ack: 24, Len: 24 MQ Telemetry Transport Protocol, Subtype: Challenge Response Header Flags: 0x02, Message Type: Subscribe Request Msg Len: 22 Message Identifier: 3 Properties: Topic Length: 16 Topic: university/+/# Subscription Options: 0x01, Retain Handling: Send msgs at subscription time, QoS: At least once delivery (Acknowl)
						0000 00 04 00 00 01 00 00 00 00 27 31 33 62 69 00 00 00 00 E.....'13b... 0001 45 00 00 40 01 77 40 00 00 22 8a 00 00 00 00 E-Aa 0 0 .. 0002 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 E-h .. M-v 0003 12 c6 97 68 96 f1 07 50 ec 8f 4d fd 1d df 76 19 P .. i .. M-v 0004 50 18 fa f9 b6 6a 00 00 10 17 00 04 4d 51 54 54 P .. j .. M-QTT 0005 6e 68 78 65 72 73 69 74 79 2f 2b 2f 2b 2f 23 01 niversit y/+/#

The client id for this packet is **dzcxnwdqef**.

For packet 2442:

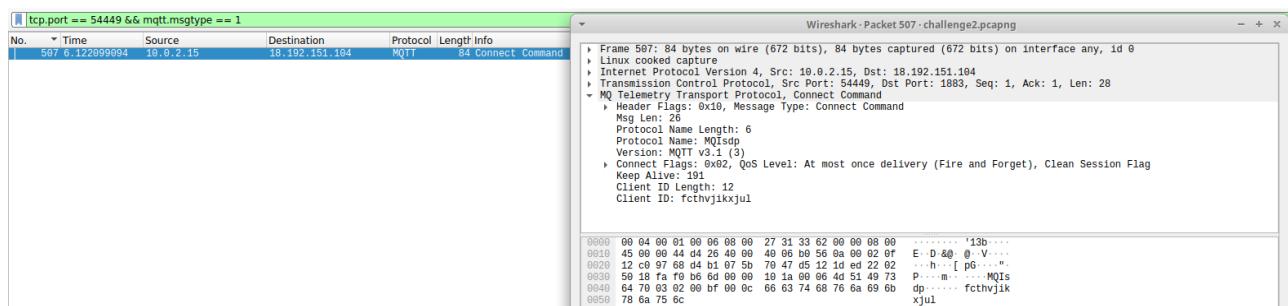
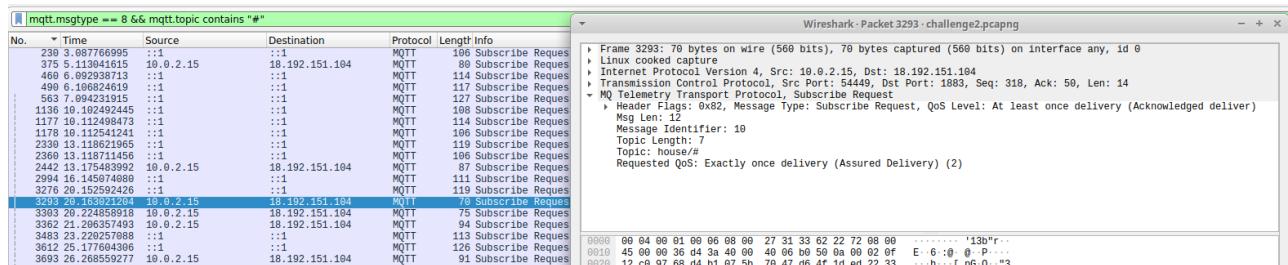
Use its source port (38619).



The client id for packet 2442 is **tukvxesu**.

For packet 3293:

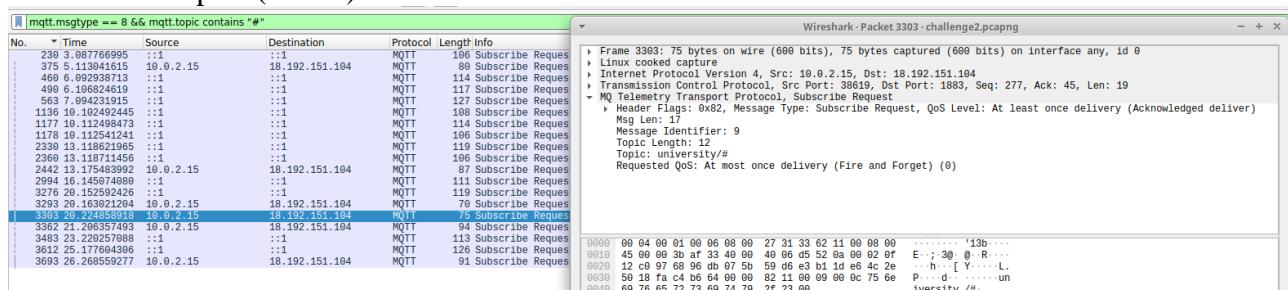
Use its source port (54449).



The client id for packet 3293 is **fcthvjkkxjul**.

For packet 3303:

Use its source port (38619).



tcp.port == 38619 && mqtt.msctype == 1					
No.	Time	Source	Destination	Protocol	Length/Info
335	4.14788260	10.0.2.15	18.192.151.104	MQTT	62 Connect Command

Frame 335: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface any, id 0					
> Linux cooked capture					
> Internet Protocol Version 4, Src: 10.0.2.15, Dst: 18.192.151.104					
> Transmission Control Protocol, Src Port: 38619, Dst Port: 1883, Seq: 1, Ack: 1, Len: 26					
> MQ Telemetry Transport Protocol, Connect Command					
0x0000	00	04	00	01	00 06 08 00 27 31 33 62 00 4a 08 00
0x0001	45	09	00	42	a7 21 49 00 40 66 d5 5d 0a 00 02 0f
0x0002	12	c0	97	68	96 d0 07 5b 59 6e e2 9d 1d e6 4c 02
0x0003	50	18	fa	f0	b6 60 00 00 10 18 00 06 4d 51 49 73
0x0004	64	70	03	02	09 e1 00 0a 74 75 6b 76 78 65 73 75
0x0005	68	65			dp..... tukvxesuhe

The client id for packet 3303 is **tukvxesuhe**.

For packet 3362:

Use its source port (57863).

mqtt.msctype == 8 && mqtt.topic contains "#"					
No.	Time	Source	Destination	Protocol	Length/Info
239	3.88766995	::1	18.192.151.104	MQTT	106 Subscribe Request (id=2) [meta]
375	5.113041515	10.0.2.15	::1	MQTT	89 Subscribe Request (id=3) [unive]
469	6.692938713	::1	18.192.151.104	MQTT	114 Subscribe Request (id=4) [meta]
499	6.106824619	::1	::1	MQTT	117 Subscribe Request (id=4) [univer]
563	7.094231915	::1	18.192.151.104	MQTT	127 Subscribe Request (id=5) [meta]
1151	9.103249245	::1	::1	MQTT	100 Subscribe Request (id=6) [meta]
1177	10.112498473	::1	::1	MQTT	114 Subscribe Request (id=7) [hospi]
1178	10.112541241	::1	::1	MQTT	119 Subscribe Request (id=8) [univer]
2339	13.118621965	::1	::1	MQTT	100 Subscribe Request (id=9) [corpor]
2360	14.118714587	::1	::1	MQTT	101 Subscribe Request (id=10) [univer]
2442	13.175483992	10.0.2.15	18.192.151.104	MQTT	87 Subscribe Request (id=11) [univer]
2994	16.145674088	::1	::1	MQTT	111 Subscribe Request (id=12) [house]
3276	20.152594246	::1	::1	MQTT	119 Subscribe Request (id=14) [unive]
3293	20.163621208	10.0.2.15	18.192.151.104	MQTT	75 Subscribe Request (id=15) [univer]
3303	20.224858918	10.0.2.15	18.192.151.104	MQTT	75 Subscribe Request (id=16) [univer]
3302	21.2065357493	10.0.2.15	18.192.151.104	MQTT	94 Subscribe Request (id=15) [unive]
3483	23.229257088	::1	18.192.151.104	MQTT	113 Subscribe Request (id=14) [unive]
3612	25.177664306	::1	::1	MQTT	126 Subscribe Request (id=12) [unive]
3693	26.268659277	10.0.2.15	18.192.151.104	MQTT	91 Subscribe Request (id=13) [facto]

Frame 3362: 64 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface any, id 0					
> Linux cooked capture					
> Internet Protocol Version 4, Src: 10.0.2.15, Dst: 18.192.151.104					
> Transmission Control Protocol, Src Port: 57863, Dst Port: 1883, Seq: 459, Ack: 351, Len: 38					
> MQ Telemetry Transport Protocol, Subscribe Request					
0x0000	00	04	00	01	00 06 08 00 27 31 33 62 00 4a 08 00
0x0001	45	09	00	42	a7 21 49 00 40 66 d5 5d 0a 00 02 0f
0x0002	12	c0	97	68	96 d0 07 5b 59 6e e2 9d 1d e6 4c 02
0x0003	50	18	fa	f0	b6 60 00 00 10 18 00 06 4d 51 49 73
0x0004	64	70	03	02	09 e1 00 0a 74 75 6b 76 78 65 73 75
0x0005	68	65			dp..... tukvxesuhe

tcp.port == 57863 && mqtt.msctype == 1					
No.	Time	Source	Destination	Protocol	Length/Info
43	0.111075257	10.0.2.15	18.192.151.104	MQTT	88 Connect Command

Frame 43: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) on interface any, id 0					
> Linux cooked capture					
> Internet Protocol Version 4, Src: 10.0.2.15, Dst: 18.192.151.104					
> Transmission Control Protocol, Connect Command					
> MQ Telemetry Transport Protocol, Connect Command					
0x0000	00	04	00	01	00 06 08 00 27 31 33 62 72 61 08 00
0x0001	45	09	00	4e	69 69 40 00 40 86 7b 6d 0a 00 02 0f
0x0002	12	c0	97	68	e2 07 07 5b c8 89 f5 1d dc 89 60
0x0003	50	18	fa	f9	b6 77 00 00 10 24 00 07 0f 75 6e
0x0004	64	70	03	02	2b 00 19 63 70 6f 65 70 6a 7a 6b
0x0005	67	32	21	73	65 63 74 69 6f 36 39 2f 23 01 g2/secti on0/#

The client id for packet 3362 is **cpoepjzhixbxgjiu**.

For packet 3693:

Use its source port (38619).

mqtt.msctype == 8 && mqtt.topic contains "#"					
No.	Time	Source	Destination	Protocol	Length/Info
239	3.88766995	::1	18.192.151.104	MQTT	106 Subscribe Request (id=2) [meta]
375	5.113041515	10.0.2.15	18.192.151.104	MQTT	89 Subscribe Request (id=3) [univer]
469	6.692938713	::1	::1	MQTT	114 Subscribe Request (id=4) [meta]
499	6.106824619	::1	::1	MQTT	117 Subscribe Request (id=4) [univer]
563	7.094231915	::1	18.192.151.104	MQTT	127 Subscribe Request (id=5) [meta]
1151	9.103249245	::1	::1	MQTT	100 Subscribe Request (id=6) [meta]
1177	10.112498473	::1	::1	MQTT	114 Subscribe Request (id=7) [hospi]
1178	10.112541241	::1	::1	MQTT	116 Subscribe Request (id=6) [meta]
2339	13.118621965	::1	18.192.151.104	MQTT	119 Subscribe Request (id=8) [univer]
2360	14.118714587	::1	::1	MQTT	108 Subscribe Request (id=9) [facto]
2442	13.175483992	10.0.2.15	18.192.151.104	MQTT	87 Subscribe Request (id=10) [univer]
2994	16.145674088	::1	::1	MQTT	111 Subscribe Request (id=12) [house]
3276	20.152594246	::1	::1	MQTT	119 Subscribe Request (id=14) [univer]
3293	20.163621208	10.0.2.15	18.192.151.104	MQTT	75 Subscribe Request (id=15) [univer]
3302	21.2065357493	10.0.2.15	18.192.151.104	MQTT	94 Subscribe Request (id=16) [univer]
3483	23.229257088	::1	18.192.151.104	MQTT	113 Subscribe Request (id=14) [univer]
3612	25.177664306	::1	::1	MQTT	126 Subscribe Request (id=12) [univer]
3693	26.268659277	10.0.2.15	18.192.151.104	MQTT	91 Subscribe Request (id=13) [facto]

Frame 3693: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface any, id 0					
> Linux cooked capture					
> Internet Protocol Version 4, Src: 10.0.2.15, Dst: 18.192.151.104					
> Transmission Control Protocol, Connect Command					
> MQ Telemetry Transport Protocol, Connect Command					
0x0000	00	04	00	01	00 06 08 00 27 31 33 62 75 72 00 08 00
0x0001	45	09	00	4b	40 86 d5 5d 0a 00 02 0f
0x0002	12	c9	97	68	00 07 5b 59 6e 04 22 1d e6 66 cc
0x0003	50	18	fa	f9	b6 71 00 00 10 1e 89 06 4d 51 49 73
0x0004	64	70	03	02	2b 00 19 63 70 6f 65 70 6a 7a 6b
0x0005	68	62	27	66	65 63 74 69 6f 36 39 3/floor #/a

Frame 3693: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface any, id 0					
> Internet Protocol Version 4, Src: 10.0.2.15, Dst: 18.192.151.104					
> Transmission Control Protocol, Connect Command					
> MQ Telemetry Transport Protocol, Connect Command					
0x0000	00	04	00	01	00 06 08 00 27 31 33 62 00 4a 08 00
0x0001	45	09	00	42	a7 21 49 00 40 66 d5 5d 0a 00 02 0f
0x0002	12	c0	97	68	96 d0 07 5b 59 6e e2 9d 1d e6 4c 02
0x0003	50	18	fa	f9	b6 60 00 00 10 18 00 06 4d 51 49 73
0x0004	64	70	03	02	09 e1 00 0a 74 75 6b 76 78 65 73 75
0x0005	68	65			dp..... tukvxesuhe

The client id for packet 3693 is **tukvxesuhe**.

The list of different client ids is **dzcxnwdqef, tukvxesuhe, fcthvjkkxjul, cpoepjzkhibxgjiu**.

CQ4) How many different MQTT clients specify a last Will Message to be directed to a topic having as first level "university"?

Answer: 1

We use this filter `mqtt.msgtype == 1 && mqtt.willmsg`. This filter shows only MQTT CONNECT packets (mqtt.msgtype == 1) that include a Last Will Message (mqtt.willmsg).

mqtt.msgtype == 1 && mqtt.willmsg						
No.	Time	Source	Destination	Protocol	Length	Info
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

Now we need to check in each packet if Will Topic starts with "/university/" or not.

For packet 4:

No.	Time	Source	Destination	Protocol	Length	Info
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

The Will Topic starts with /university.

For packet 196:

No.	Time	Source	Destination	Protocol	Length	Info
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

The will topic doesn't start with /university.

For packet 352:

The Will Topic doesn't start with /university.

For packet 557:

The Will Topic doesn't start with /university.

CQ5) How many MQTT subscribers receive a last will message derived from a subscription without a wildcard?

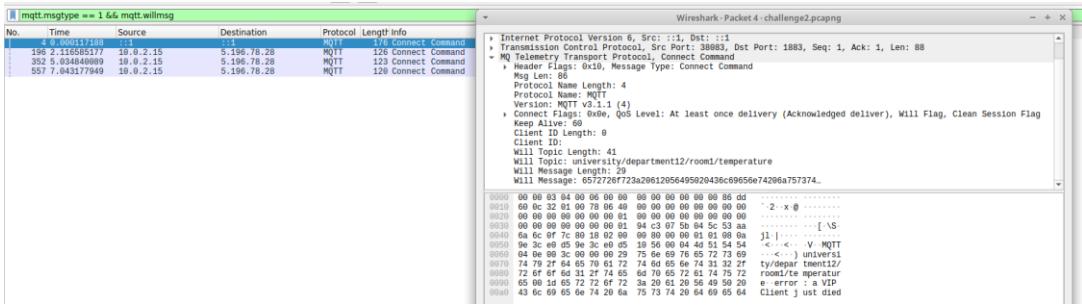
Answer: 3

First we use this filter `mqtt.msgtype == 1 && mqtt.willmsg`.

This filter identifies clients (`mqtt.msgtype == 1`) that defines a last will message (`mqtt.willmsg`) derived from a subscription without a wildcard. Now we need to check what will topic each packet has.

mqtt.msgtype == 1 && mqtt.willmsg						
No.	Time	Source	Destination	Protocol	Length	Info
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

For packet 4:



The Will Topic is "**university/department12/room1/temperature**".

In this step, we check if a client has subscribed to this topic without using a wildcard or not. We use this filter `mqtt.msgtype == 8 && mqtt.topic == "university/department12/room1/temperature"`. From above image we can see three packets, but we should analyze that these three packets belong to different TCP streams, meaning they are from different clients.

For this packet, three clients received messages without wildcards.

For packet 196:

The Will Topic is "**metaverse/room2/floor4**". We use this filter `mqtt.msgtype == 8 && mqtt.topic == "metaverse/room2/floor4"`.

mqtt.msgtype == 8 && mqtt.topic == "metaverse/room2/floor4"						
No.	Time	Source	Destination	Protocol	Length	Info

The number of subscriptions without wildcards on this topic is zero.

For packet 352:

mqtt.msgtype == 1 && mqtt.willmsg						
No.	Time	Source	Destination	Protocol	Length	Info
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	128	Connect Command

Frame 352: 128 bytes on wire (984 bits), 128 bytes captured (984 bits) on interface any, id 0
 Linux cooked capture
 Internet Protocol Version 4, Src: 10.0.2.15, Dst: 5.196.78.28
 Transmission Control Protocol, Src Port: 53485, Dst Port: 10883, Seq: 1, Ack: 1, Len: 67
 MQ Telemetry Transport Protocol, Connect Command
 > Header Flags: 0x10, Message Type: Connect Command
 > Msg Len: 67
 > Protocol Name Length: 4
 > Protocol Name: MQTT
 > Version: MQTT v5.0 (5)
 > Connect Flags: 0x06, QoS Level: At most once delivery (Fire and Forget), Will Flag, Clean Session Flag
 > Keep Alive: 264
 > Client ID: 1stvfwz
 > Client ID Length: 8
 > Client ID: 1stvfwz
 > Will Properties

The Will Topic is "hospital/facility3/area3". We use this filter mqtt.msgtype == 8 && mqtt.topic == "hospital/facility3/area3".

mqtt.msgtype == 8 && mqtt.topic == "hospital/facility3/area3"						
No.	Time	Source	Destination	Protocol	Length	Info

For packet 557:

mqtt.msgtype == 1 && mqtt.willmsg						
No.	Time	Source	Destination	Protocol	Length	Info
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	128	Connect Command

Frame 557: 128 bytes on wire (960 bits), 128 bytes captured (960 bits) on interface any, id 0
 Linux cooked capture
 Internet Protocol Version 4, Src: 10.0.2.15, Dst: 5.196.78.28
 Transmission Control Protocol, Src Port: 42665, Dst Port: 1883, Seq: 1, Ack: 1, Len: 64
 MQ Telemetry Transport Protocol, Connect Command
 > Header Flags: 0x10, Message Type: Connect Command
 > Msg Len: 64
 > Protocol Name Length: 4
 > Protocol Name: MQTT
 > Version: MQTT v3.1.1 (4)
 > Connect Flags: 0x06, QoS Level: At least once delivery (Acknowledged deliver), Will Flag, Clean Session Flag
 > Keep Alive: 337
 > Client ID: kiurrzngj
 > Client ID Length: 10
 > Client ID: kiurrzngj
 > Will Topic Length: 21
 > Will Topic: metaverse/room2/room2
 > Will Topic Length: 21
 > Will Topic: metaverse/room2/room2

The Will Topic is "metaverse/room2/room2". We use this filter mqtt.msgtype == 8 && mqtt.topic == "metaverse/room2/room2".

mqtt.msgtype == 8 && mqtt.topic == "metaverse/room2/room2"						
No.	Time	Source	Destination	Protocol	Length	Info

CQ6) How many MQTT publish messages directed to the public broker mosquitto are sent with the retain option and use QoS “At most once”?

Answer: 208

I use this filter mqtt.msgtype == 3 && mqtt.retain == 1 && mqtt.qos == 0 && ip.dst == 5.196.78.28. This filter displays all MQTT PUBLISH messages sent to the Mosquitto broker (ip.dst == 5.196.78.28) that have the retain flag set (mqtt.retain == 1) and use QoS level 0 (mqtt.qos == 0), which means the message is delivered at most once, without any acknowledgment or retransmission if delivery fails.

No.	Time	Source	Destination	Protocol	Length	Info
343	15.180262925	10.0.2.15	5.196.78.28	MQTT	206	Publish Message [metaverse/department2/area3]
2877	14.980932029	10.0.2.15	5.196.78.28	MQTT	206	Publish Message [hospital/room2/area3/pollution]
2914	15.186698287	10.0.2.15	5.196.78.28	MQTT	192	Publish Message [factory/room2/area3]
2936	15.358964848	10.0.2.15	5.196.78.28	MQTT	193	Publish Message [factory/department2]
3269	15.185145268	10.0.2.15	5.196.78.28	MQTT	95	Publish Message [metaverse/building3/floor4/deposit]
3778	27.988951316	10.0.2.15	5.196.78.28	MQTT	206	Publish Message [metaverse/building3/room2/light]
3884	27.493741427	10.0.2.15	5.196.78.28	MQTT	205	Publish Message [factory/department2/floor4/light]
3890	27.499211840	10.0.2.15	5.196.78.28	MQTT	192	Publish Message [factory/room2/area3]
3909	27.499211840	10.0.2.15	5.196.78.28	MQTT	206	Publish Message [metaverse/room2/hospital/pollution]
4699	31.779645783	10.0.2.15	5.196.78.28	MQTT	197	Publish Message [factory/building3/floor4]
4916	31.855909851	10.0.2.15	5.196.78.28	MQTT	207	Publish Message [hospital/building3/room2/humidity]
4347	39.332200710	10.0.2.15	5.196.78.28	MQTT	199	Publish Message [metaverse/building3/room2]
4444	42.944009077	10.0.2.15	5.196.78.28	MQTT	198	Publish Message [university/room2/section1]
4508	45.188908908	10.0.2.15	5.196.78.28	MQTT	201	Publish Message [university/building3/room1/light]
4508	45.475138168	10.0.2.15	5.196.78.28	MQTT	198	Publish Message [factory/department2/area3]
4512	46.608094344	10.0.2.15	5.196.78.28	MQTT	202	Publish Message [university/building3/section1]
4548	47.797638269	10.0.2.15	5.196.78.28	MQTT	200	Publish Message [university/facility3/floor4]
4633	52.143451988	10.0.2.15	5.196.78.28	MQTT	194	Publish Message [university/room2/area3]
4640	54.174208203	10.0.2.15	5.196.78.28	MQTT	194	Publish Message [metaverse/room2/area3]
4699	57.451814924	10.0.2.15	5.196.78.28	MQTT	86	Publish Message [metaverse/facility3/room2]
4781	62.459947130	10.0.2.15	5.196.78.28	MQTT	192	Publish Message [metaverse/building3]
4811	65.284771761	10.0.2.15	5.196.78.28	MQTT	200	Publish Message [metaverse/building3/floor4]
4821	66.185134471	10.0.2.15	5.196.78.28	MQTT	193	Publish Message [metaverse/facility3]
4832	73.185134471	10.0.2.15	5.196.78.28	MQTT	210	Publish Message [metaverse/room2/facility3/floor4/hydraulic_valve]
4955	75.314253780	10.0.2.15	5.196.78.28	MQTT	215	Publish Message [metaverse/room2/hydraulic_valve]
5057	79.805194077	10.0.2.15	5.196.78.28	MQTT	195	Publish Message [metaverse/department2]
5188	83.649617878	10.0.2.15	5.196.78.28	MQTT	200	Publish Message [university/building3/area3]
5200	83.779645783	10.0.2.15	5.196.78.28	MQTT	200	Publish Message [metaverse/section1/section1]
5420	93.606654698	10.0.2.15	5.196.78.28	MQTT	193	Publish Message [university/section1/section1]
5474	98.842901374	10.0.2.15	5.196.78.28	MQTT	81	Publish Message [hospital/room2/floor4]
5569	101.536321277	10.0.2.15	5.196.78.28	MQTT	202	Publish Message [metaverse/department2/room2]
5693	106.782216803	10.0.2.15	5.196.78.28	MQTT	97	Publish Message [university/building3/room2/pollution]
5705	111.832347178	10.0.2.15	5.196.78.28	MQTT	101	Publish Message [factory/facility3/temperature]
5850	116.832347178	10.0.2.15	5.196.78.28	MQTT	216	Publish Message [factory/facility3/section1/hydraulic_valve]
5929	118.459690918	10.0.2.15	5.196.78.28	MQTT	202	Publish Message [university/room2/floor4/light]
6523	143.684190769	10.0.2.15	5.196.78.28	MQTT	187	Publish Message [hospital/room2]
6527	144.085199419	10.0.2.15	5.196.78.28	MQTT	212	Publish Message [university/building3/room2/temperature]
6749	157.808672234	10.0.2.15	5.196.78.28	MQTT	193	Publish Message [university/facility3]
6812	160.755841339	10.0.2.15	5.196.78.28	MQTT	86	Publish Message [metaverse/building3/floor4]
6859	164.144895763	10.0.2.15	5.196.78.28	MQTT	190	Publish Message [factory/facility3]
6914	171.764167937	10.0.2.15	5.196.78.28	MQTT	188	Publish Message [metaverse/room2]
6985	178.207094988	10.0.2.15	5.196.78.28	MQTT	201	Publish Message [metaverse/building3/section1/pollution]
7011	181.042926672	10.0.2.15	5.196.78.28	MQTT	211	Publish Message [metaverse/facility3/floor4/temperature]
7935	183.461848815	10.0.2.15	5.196.78.28	MQTT	208	Publish Message [factory/facility3/room4/pollution]
7865	184.441959147	10.0.2.15	5.196.78.28	MQTT	188	Publish Message [hospital/room2]
7977	185.765212912	10.0.2.15	5.196.78.28	MQTT	200	Publish Message [metaverse/department2/area3]
7981	185.765212912	10.0.2.15	5.196.78.28	MQTT	200	Publish Message [university/facility3/room2]

In the above image, the displayed show 208.

CQ7) How many MQTT-SN messages on port 1885 are sent by the clients to a broker in the local machine?

Answer: 0

To solve Question 7, we use this filter `udp.port == 1885 && ip.dst == 127.0.0.1`.

File	Edit	View	Go	Analyze	Statistics	Telephony	Wireless	Tools	Help
udp.port == 1885 && ip.dst == 127.0.0.1									
No.	Time	Source	Destination	Protocol	Length	Info			

The condition `udp.port == 1885` ensures that only messages associated with the MQTT-SN service (which typically runs on UDP port 1885) are included. The condition `ip.dst == 127.0.0.1` restricts the results to packets whose destination is the local machine (localhost), indicating that the messages are being sent to the broker.