**zMQTT and CoAP exercises (Part 2)**

March 14, 2024

**Setup of CoAP files (only needs to be done once):**

The commands in this section install the necessary files for the CoAP part of the exercise and are only required once. Start the course’s virtual machine, open a terminal and execute the following commands in the terminal to download and install the relevant software:

* **git clone https://github.com/lstaalhagen/C34351CoAP**
* **cd C34351CoAP**
* **sudo ./coap-setup.sh**

**Note**: The **coap-setup.sh** script may take several minutes to run.

**Task 0 (Verify connectivity – Does not have to be documented in the exercise report):**

Open a terminal window and execute the following commands (note the extra option to the netgenerate.sh script, which specifies that the simulated nodes use IPv6 addresses instead of the default IPv4 addresses):

* **cd ~/C34351CoAP**
* **sudo ./netgenerate.sh --ipv6**

The second of these commands creates a small *virtual* network using Linux network namespaces with three hosts (H1, H2, and H3) connected to a switch, as illustrated in Figure 1. The IPv6 addresses of the hosts are **fd00::1** for H1, **fd00::2** for H2, and **fd00::3** for H3. The command also opens an Xterm window on each host so you can execute commands on a specific host.



Figure 1 – Switched network with three hosts.

Select an arbitrary host and (in that host’s Xterm window) check if you can ping the other two hosts with the command **ping -c1 fd00::x** (where x is 1, 2, or 3, depending on the host).

If the virtual network is not needed anymore, it can be deleted with the script **clearnet.sh** in the C34351CoAP directory. The script requires administrative privileges, so it must be executed with the command **sudo ./clearnet.sh** (If your current directory is not C34351CoAP, you can change to it with **cd ~/C34351CoAP** ).

Note that the virtual network created with the **sudo ./netgenerate.sh --ipv6** command disappears if the virtual machine is restarted, so you must execute the commands at the start of Task 0 to recreate it after a VM restart.

**Task 1 (Basic CoAP):**

Start a CoAP server on host H1 with the command (in H1’s Xterm window):

**coap-server -A fd00::1**

Start Wireshark in the normal terminal window with **sudo wireshark** and start a capture on the **veth11** interface with the display filter set to **coap**.

Task 1a: Use the CoAP client program in H2’s Xterm window to retrieve the default resource (‘**/**’) from the server with the command[[1]](#footnote-1) **coap-client coap://[fd00::1]/**

Stop the capture in Wireshark and examine the contents of the packets captured[[2]](#footnote-2):

* What version of CoAP is used?
* Does the message exchange use confirmable or non-confirmable messages?
* What is the purpose[[3]](#footnote-3) of the two options in the CoAP message sent from **fd00::1** to **fd00::02**?

Task 1b: Start a new Wireshark capture and use the coap-client to retrieve the ‘**/async**’ resource with the command **coap-client coap://[fd00::1]/async**   
  
Stop the Wireshark capture. Explain the differences in the CoAP messages between the retrieval of the default resource in task 1a and the ‘**/async**’ resource in this task.

Task 1c: Start a new capture in Wireshark and use the CoAP client program again to retrieve both the default resource (as in task 1a) and the ‘**/async**’ resource (as in task 1b), but now with the option ‘**-N**’ between the command and the URI.

Explain the difference in terms of the CoAP messages in tasks 1a and 1b and this task.

**Task 2 (Dynamic resources):**

Stop the server (press **Ctrl-C** when H1’s Xterm window has focus). To enable that new resources can be added to the server, the server must be restarted with an extra option: **‑d 999**

The extra option permits the clients to define and set the values of up to 999 dynamic resources.

Task 2a: Start a new capture in Wireshark. Use the coap-client in H2’s Xterm window to define a resource with the name C34351 (i.e., the URI will be **coap://[fd00::1]/C34351**) and the value “**DTU’s best course**”. You will need to use additional options for the CoAP client compared to task 1 to define a) the action/method that creates/instantiates a dynamic resource on the server, b) the value to be stored in the resource, and c) the type of information for the value. Please consult the documentation[[4]](#footnote-4) for the coap-client to determine which options are required and how they are used. Document that the dynamic resource is created with the capture from Wireshark.

Task 2b: Use the client to retrieve the resource with URI **coap://[fd00::1]/C34351** again. Does it contain the expected value?

Task 2c: Now use the client to delete the resource and verify afterwards that it has been deleted.

**Task 3 (Observing a resource):**

Start a new wireshark capture. Use the client to *subscribe* (Hint: consult the client documentation again) to the URI **coap://[fd00::1]/time** for 15 seconds. Stop the Wireshark capture and explain[[5]](#footnote-5) the CoAP packets captured.

**Task 4: (Publish/Subscribe with CoAP):**

Design a setup with one server and two clients that resembles an MQTT network, i.e., with one client acting as a publisher that (in this exercise manually) “publishes” new content to a dynamic resource of your choosing on the server and the other client acting as a subscriber that “subscribes” to this dynamic resource for a suitable time.

List the commands you use to make this setup and document with a Wireshark capture and screenshots that the setup works as expected. Note: You should probably run the server command on H1 (and capture the coap traffic on interface **veth11**) and run the clients on H2 and H3.

**Task 5:**

Stop the CoAP server in H1. Start new CoAP-servers on both H1 and H2 with the command: **coap‑server ‑g ff05::fd**

Start a Wireshark capture on interface **veth31**. Execute the following command in H3’s Xterm window: **coap-client -N coap://[ff05::fd]/**

Stop the Wireshark capture and explain the CoAP packets captured, including:

* What is the type of MAC address used as the destination MAC address in the CoAP packet from the client?
* Why is the ‘-N’ option needed in this scenario?

1. Note that when IPv6 addresses are used directly, the address is enclosed in square brackets, ‘[‘ and ‘]’. [↑](#footnote-ref-1)
2. You can ignore that Wireshark describes an option as “Unknown Option” – this is only because the installed version of Wireshark has not been updated to “recognize” all possible CoAP message options. [↑](#footnote-ref-2)
3. See the RFC-document for CoAP (RFC-7252): <https://www.rfc-editor.org/rfc/rfc7252> [↑](#footnote-ref-3)
4. See <https://libcoap.net/doc/reference/4.3.4/man_coap-client.html> (or the PDF-files on DTU Learn) [↑](#footnote-ref-4)
5. RFC-7641 (<https://www.rfc-editor.org/rfc/rfc7641>) might be relevant to consult. [↑](#footnote-ref-5)