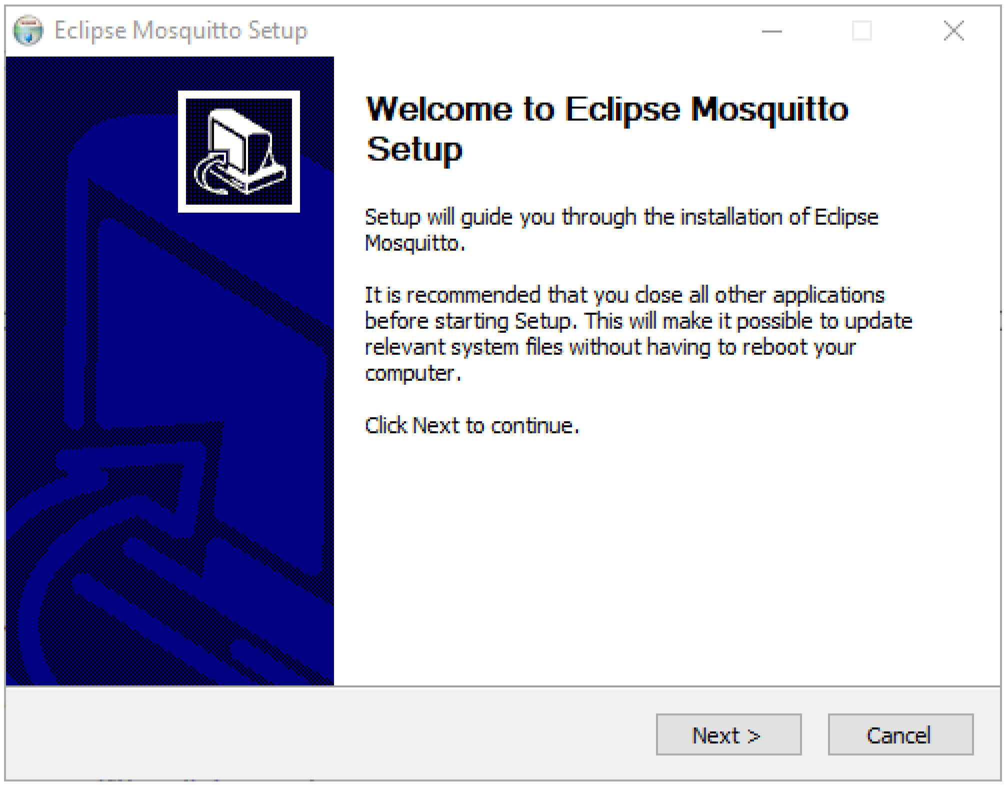
I installed the Mosquitto MQTT broker on my local network from https://mosquitto.org/download/


Eclipse Mosquitto Setup

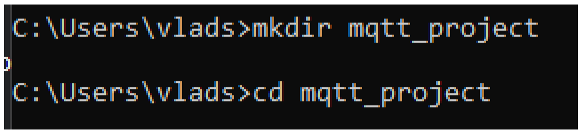
Welcome to Eclipse Mosquitto Setup

Setup will guide you through the installation of Eclipse Mosquitto.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

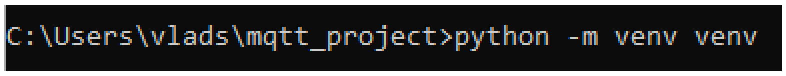
Click Next to continue.

I created a new directory for my project:

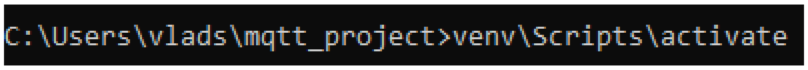


```
C:\Users\vlads>mkdir mqtt_project

C:\Users\vlads>cd mqtt_project
```

I created a Virtual Environment and activated it:

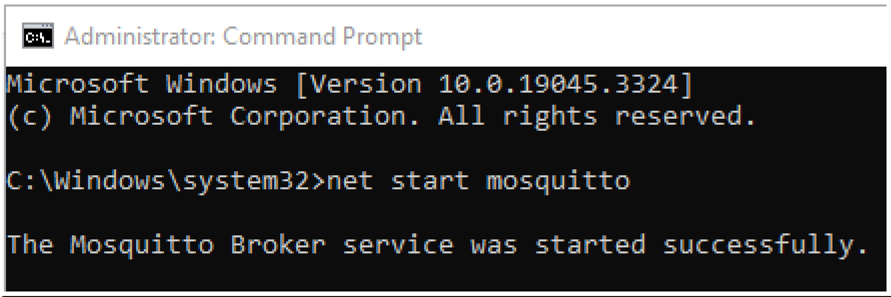


```
C:\Users\vlads\mqtt_project>python -m venv venv
```

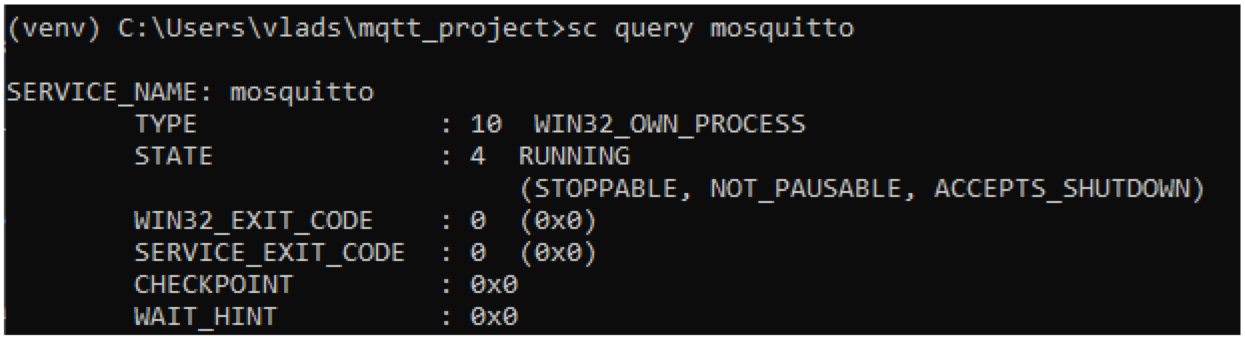


```
C:\Users\vlads\mqtt_project>venv\Scripts\activate
```

I started the process and verified if it is running.

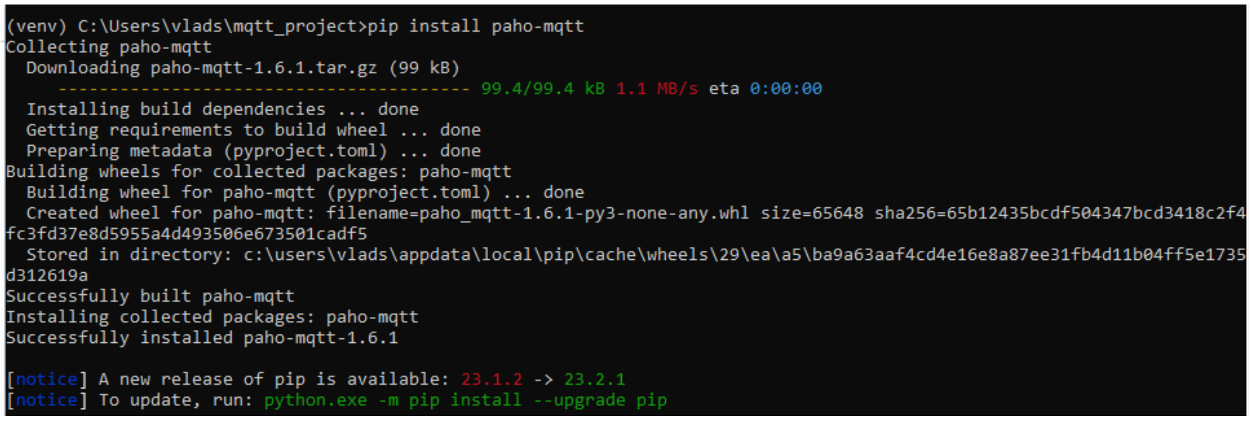


```
Microsoft Windows [Version 10.0.19045.3324]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>net start mosquitto

The Mosquitto Broker service was started successfully.
```

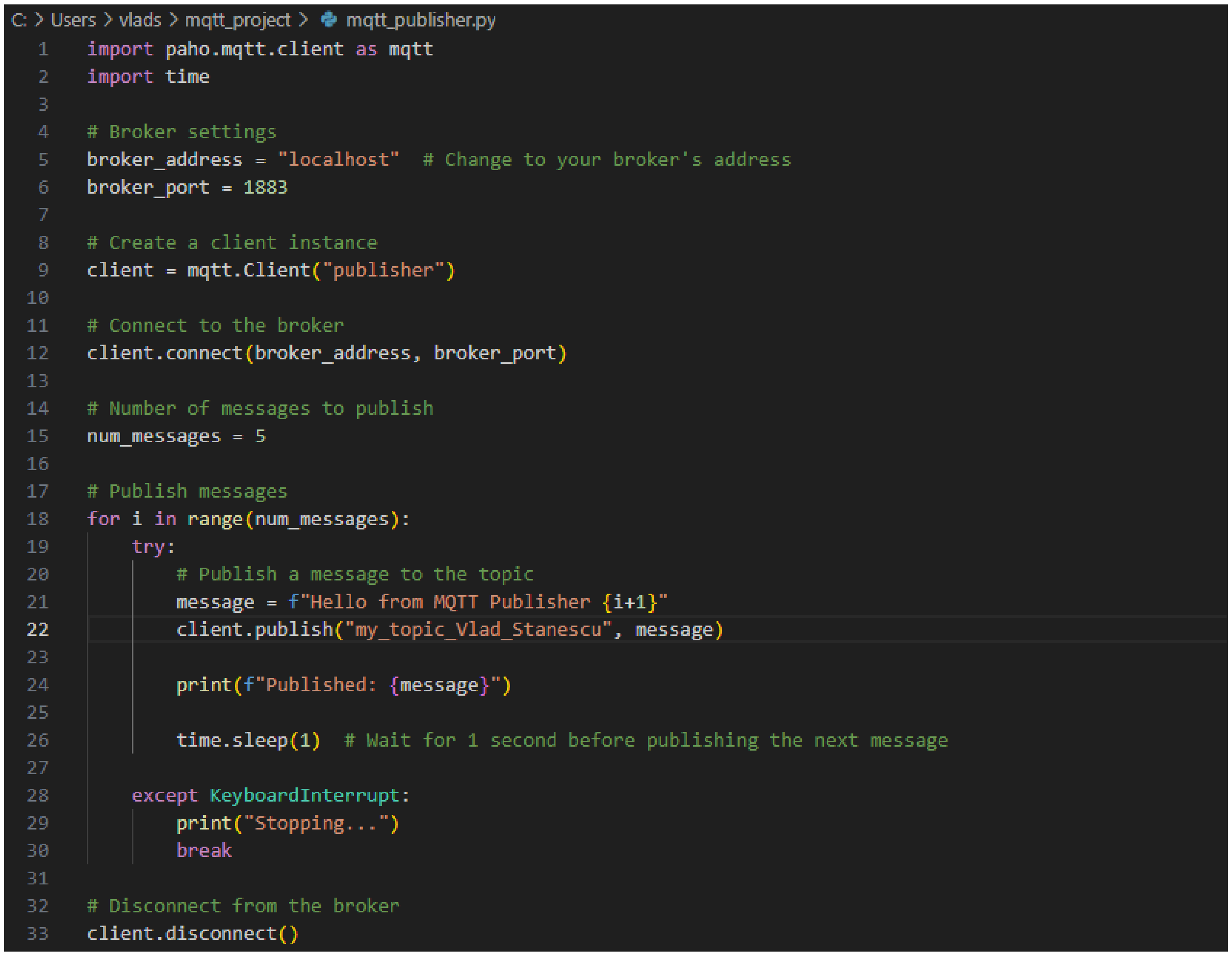


```
(venv) C:\Users\vlads\mqtt_project>sc query mosquitto

SERVICE_NAME: mosquitto
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 4  RUNNING
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x0
```

I installed the MQTT Library:

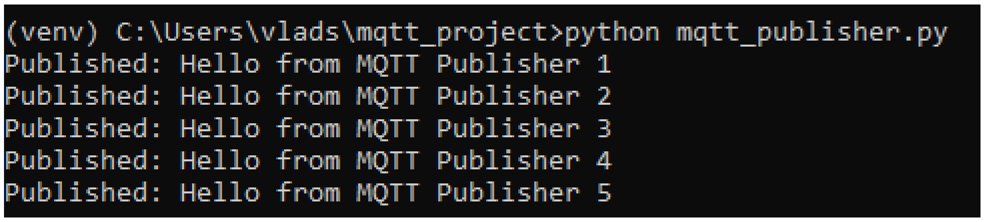


```
(venv) C:\Users\vlads\mqtt_project>pip install paho-mqtt
Collecting paho-mqtt
  Downloading paho-mqtt-1.6.1.tar.gz (99 kB)
     ---------------------------------------- 99.4/99.4 kB 1.1 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Building wheels for collected packages: paho-mqtt
  Building wheel for paho-mqtt (pyproject.toml) ... done
  Created wheel for paho-mqtt: filename=paho_mqtt-1.6.1-py3-none-any.whl size=65648 sha256=65b12435bcdf504347bcd3418c2f4
fc3fd37e8d5955a4d493506e673501cadf5
  Stored in directory: c:\users\vlads\appdata\local\pip\cache\wheels\29\ea\a5\ba9a63aaf4cd4e16e8a87ee31fb4d11b04ff5e1735
d312619a
Successfully built paho-mqtt
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-1.6.1

[notice] A new release of pip is available: 23.1.2 -> 23.2.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

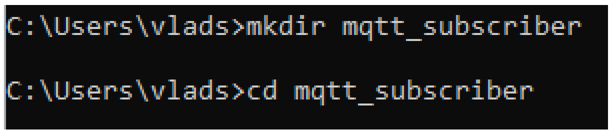

C: > Users > vlads > mqtt_project > mqtt_publisher.py

```python
import paho.mqtt.client as mqtt
import time

# Broker settings
broker_address = "localhost"  # Change to your broker's address
broker_port = 1883

# Create a client instance
client = mqtt.Client("publisher")

# Connect to the broker
client.connect(broker_address, broker_port)

# Number of messages to publish
num_messages = 5

# Publish messages
for i in range(num_messages):
    try:
        # Publish a message to the topic
        message = f"Hello from MQTT Publisher {i+1}"
        client.publish("my_topic_Vlad_Stanescu", message)

        print(f"Published: {message}")

        time.sleep(1)  # Wait for 1 second before publishing the next message

    except KeyboardInterrupt:
        print("Stopping...")
        break

# Disconnect from the broker
client.disconnect()
```

I created a Python Script and wrote a code to create a simple MQTT publisher that sends messages to my topic.

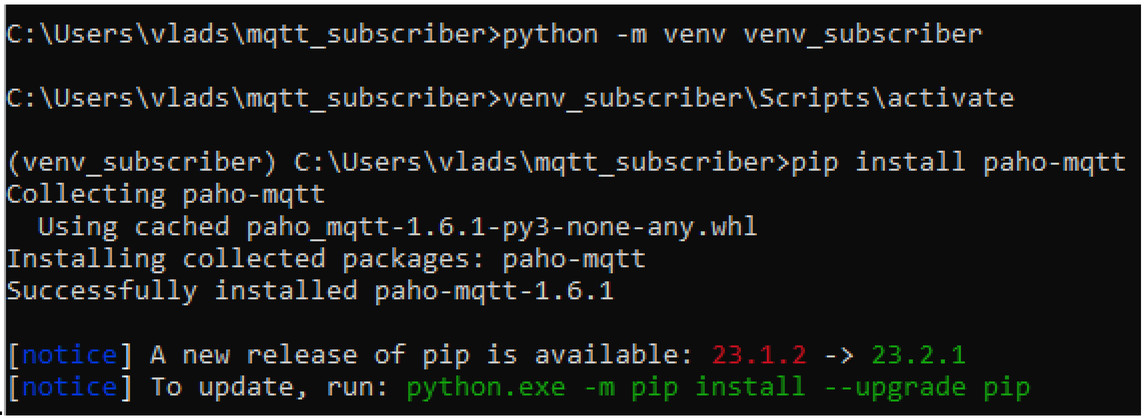


```
(venv) C:\Users\vlads\mqtt_project>python mqtt_publisher.py
Published: Hello from MQTT Publisher 1
Published: Hello from MQTT Publisher 2
Published: Hello from MQTT Publisher 3
Published: Hello from MQTT Publisher 4
Published: Hello from MQTT Publisher 5
```

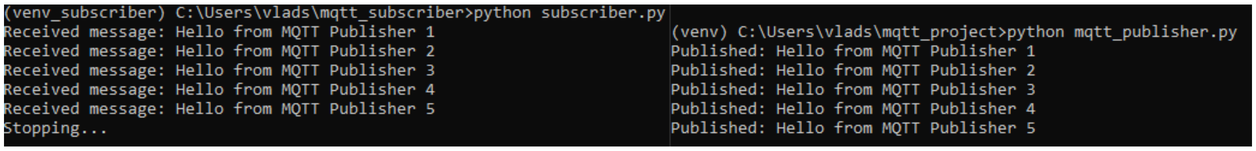


```
C:\Users\vlads>mkdir mqtt_subscriber

C:\Users\vlads>cd mqtt_subscriber
```

I created a separate Python project for the subscriber client following the same steps. This script will connect to the Mosquitto broker, subscribe to the topic, and print received messages to the console.

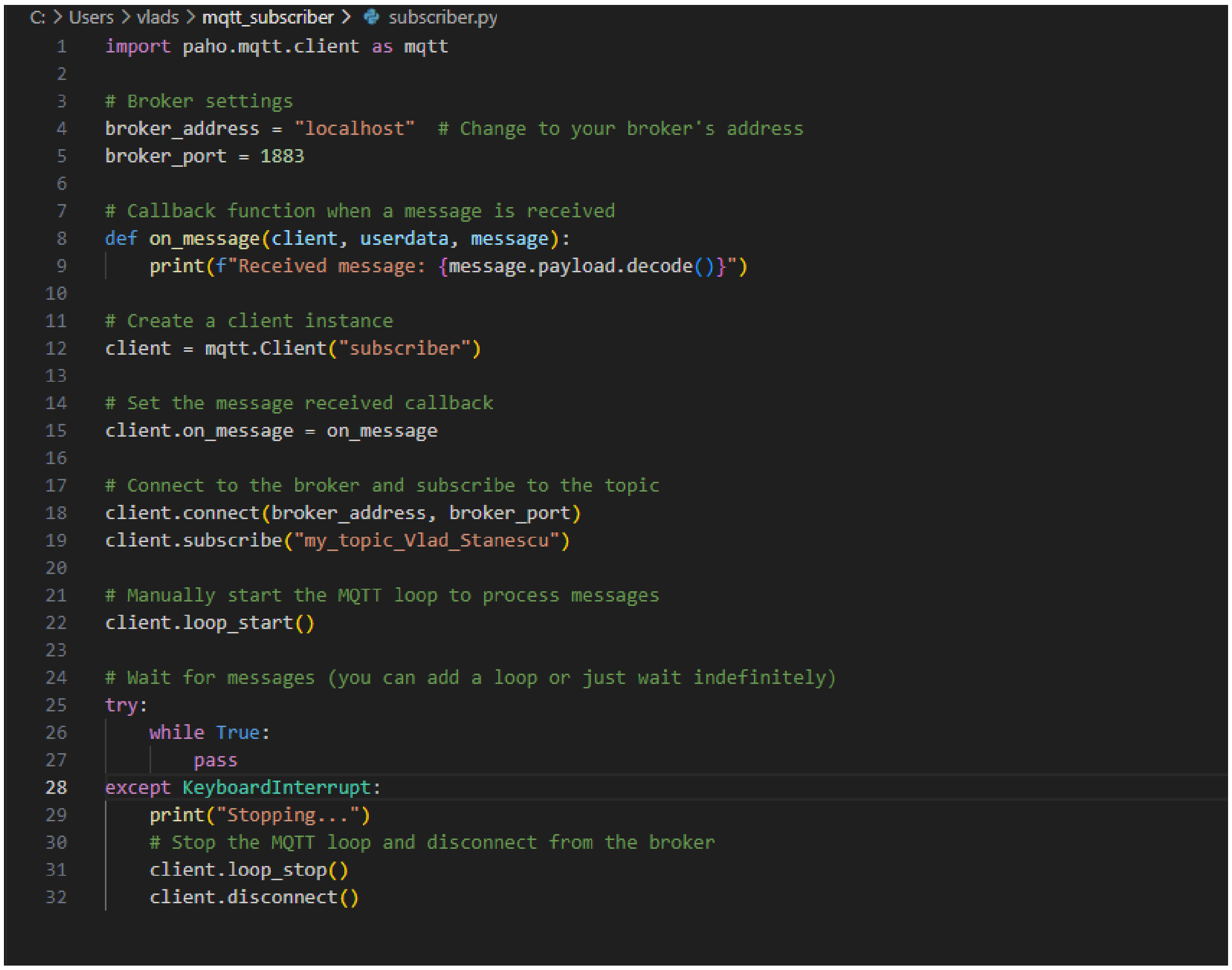


```
C:\Users\vlads\mqtt_subscriber>python -m venv venv_subscriber

C:\Users\vlads\mqtt_subscriber>venv_subscriber\Scripts\activate

(venv_subscriber) C:\Users\vlads\mqtt_subscriber>pip install paho-mqtt
Collecting paho-mqtt
  Using cached paho_mqtt-1.6.1-py3-none-any.whl
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-1.6.1

[notice] A new release of pip is available: 23.1.2 -> 23.2.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```



```
(venv_subscriber) C:\Users\vlads\mqtt_subscriber>python subscriber.py
Received message: Hello from MQTT Publisher 1
Received message: Hello from MQTT Publisher 2
Received message: Hello from MQTT Publisher 3
Received message: Hello from MQTT Publisher 4
Received message: Hello from MQTT Publisher 5
Stopping...
```

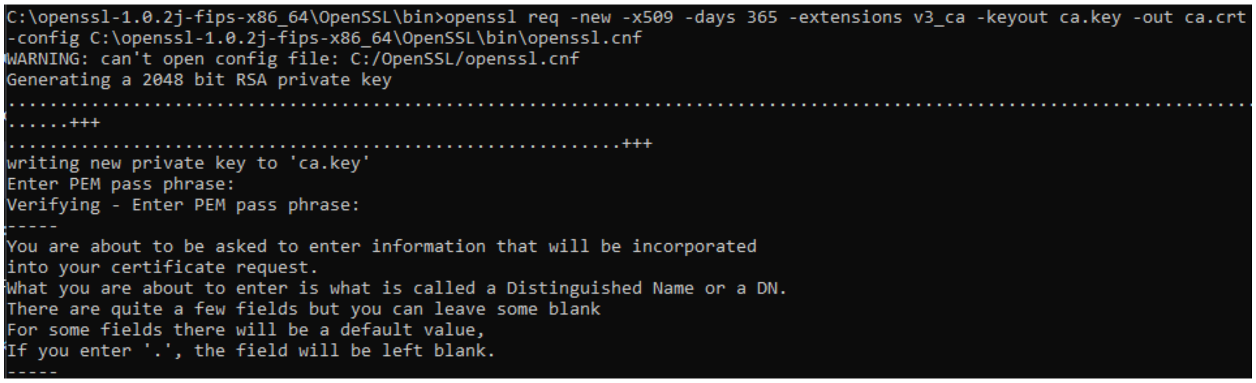
```
(venv) C:\Users\vlads\mqtt_project>python mqtt_publisher.py
Published: Hello from MQTT Publisher 1
Published: Hello from MQTT Publisher 2
Published: Hello from MQTT Publisher 3
Published: Hello from MQTT Publisher 4
Published: Hello from MQTT Publisher 5
```



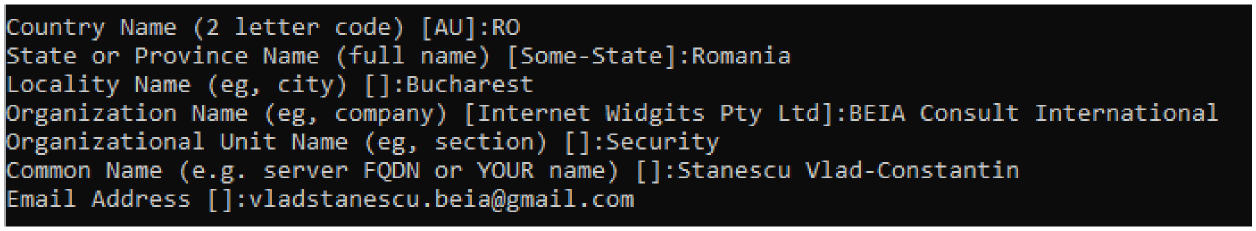
C: > Users > vlads > mqtt_subscriber > subscriber.py

```python
import paho.mqtt.client as mqtt

# Broker settings
broker_address = "localhost"  # Change to your broker's address
broker_port = 1883

# Callback function when a message is received
def on_message(client, userdata, message):
    print(f"Received message: {message.payload.decode()}")

# Create a client instance
client = mqtt.Client("subscriber")

# Set the message received callback
client.on_message = on_message

# Connect to the broker and subscribe to the topic
client.connect(broker_address, broker_port)
client.subscribe("my_topic_Vlad_Stanescu")

# Manually start the MQTT loop to process messages
client.loop_start()

# Wait for messages (you can add a loop or just wait indefinitely)
try:
    while True:
        pass
except KeyboardInterrupt:
    print("Stopping...")
    # Stop the MQTT loop and disconnect from the broker
    client.loop_stop()
    client.disconnect()
```

I downloaded OpenSSL. I generated a certificate authority (CA) key and a self-signed certificate:

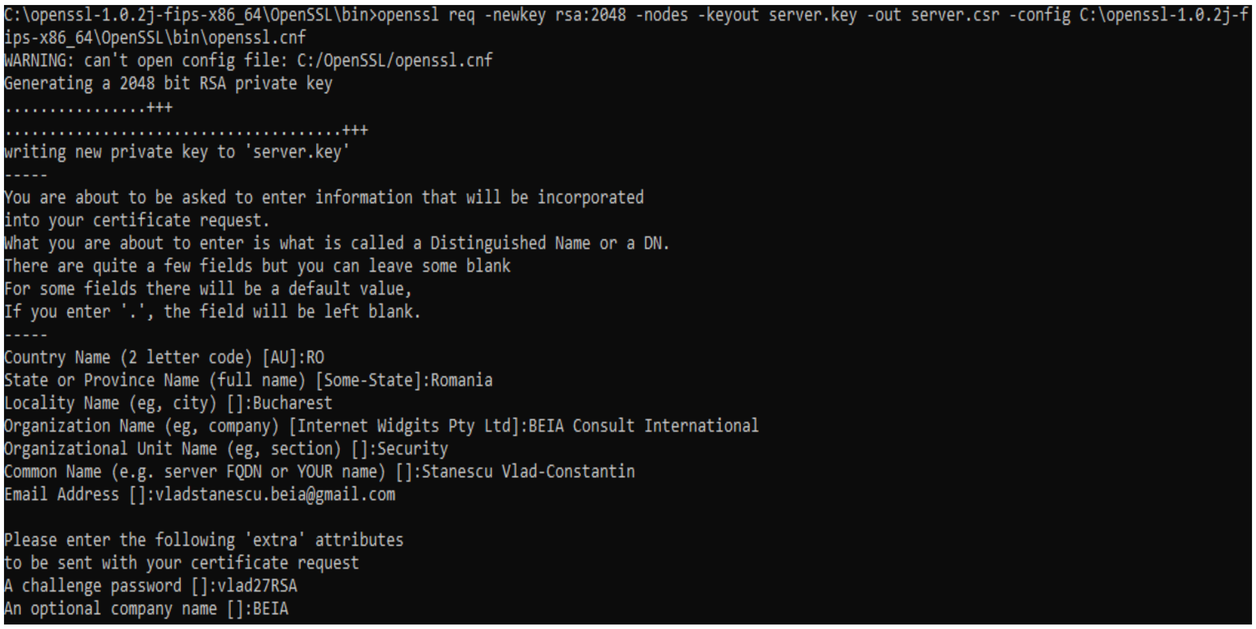


```
C:\openssl-1.0.2j-fips-x86_64\OpenSSL\bin>openssl req -new -x509 -days 365 -extensions v3_ca -keyout ca.key -out ca.crt
-config C:\openssl-1.0.2j-fips-x86_64\OpenSSL\bin\openssl.cnf
WARNING: can't open config file: C:/OpenSSL/openssl.cnf
Generating a 2048 bit RSA private key
..........................................................................................................................
......+++
.............................................................+++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
```

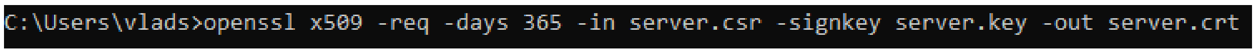


```
Country Name (2 letter code) [AU]:RO
State or Province Name (full name) [Some-State]:Romania
Locality Name (eg, city) []:Bucharest
Organization Name (eg, company) [Internet Widgits Pty Ltd]:BEIA Consult International
Organizational Unit Name (eg, section) []:Security
Common Name (e.g. server FQDN or YOUR name) []:Stanescu Vlad-Constantin
Email Address []:vladstanescu.beia@gmail.com
```

I generated a server key and a certificate signed by the CA:



```
C:\openssl-1.0.2j-fips-x86_64\OpenSSL\bin>openssl req -newkey rsa:2048 -nodes -keyout server.key -out server.csr -config C:\openssl-1.0.2j-f
ips-x86_64\OpenSSL\bin\openssl.cnf
WARNING: can't open config file: C:/OpenSSL/openssl.cnf
Generating a 2048 bit RSA private key
...............+++
.....................................+++
writing new private key to 'server.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:RO
State or Province Name (full name) [Some-State]:Romania
Locality Name (eg, city) []:Bucharest
Organization Name (eg, company) [Internet Widgits Pty Ltd]:BEIA Consult International
Organizational Unit Name (eg, section) []:Security
Common Name (e.g. server FQDN or YOUR name) []:Stanescu Vlad-Constantin
Email Address []:vladstanescu.beia@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:vlad27RSA
An optional company name []:BEIA
```

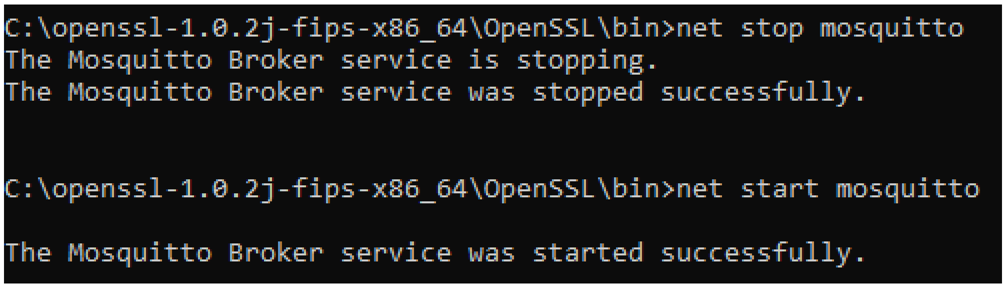


```
C:\Users\vlads>openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

After generating the certificates, I needed to copy them to a directory where my Mosquitto broker can access them. I copied them to the Mosquitto configuration directory. Then I needed to configure Mosquitto, so I opened my Mosquitto configuration file (mosquitto.conf) and add the following lines to enable encrypted communication using the generated certificates:

```
listener 8883
cafile C:/Program Files/mosquitto/ca.crt
certfile C:/Program Files/mosquitto/server.crt
keyfile C:/Program Files/mosquitto/server.key
```

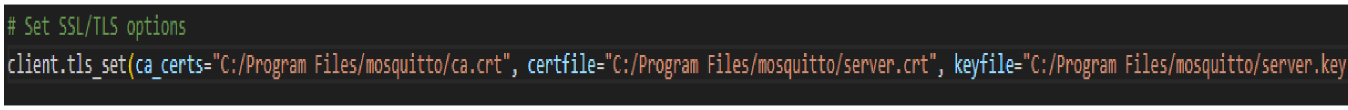
I restarted the Mosquitto broker for the changes to take effect. After completing these steps, my Mosquitto broker should be configured to use encrypted communication with the SSL/TLS certificates I've generated.



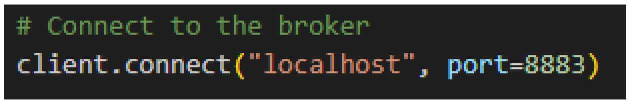
```
C:\openssl-1.0.2j-fips-x86_64\OpenSSL\bin>net stop mosquitto
The Mosquitto Broker service is stopping.
The Mosquitto Broker service was stopped successfully.


C:\openssl-1.0.2j-fips-x86_64\OpenSSL\bin>net start mosquitto

The Mosquitto Broker service was started successfully.
```

To verify if I have successfully implemented encrypted communication for the Mosquitto broker using SSL/TLS I used the MQTT clients created earlier (publisher and subscriber) and modified them to connect over SSL/TLS. I updated the client code to use the SSL/TLS configuration parameters (cafile, certfile, keyfile) in addition to specifying the SSL/TLS port (typically 8883).

I needed the ssl library. In both the publisher and subscriber scripts, after creating the MQTT client instance, I configured the SSL/TLS options using the tls_set method. In both scripts, I updated the connection method to connect using the encrypted SSL/TLS port (usually 8883).

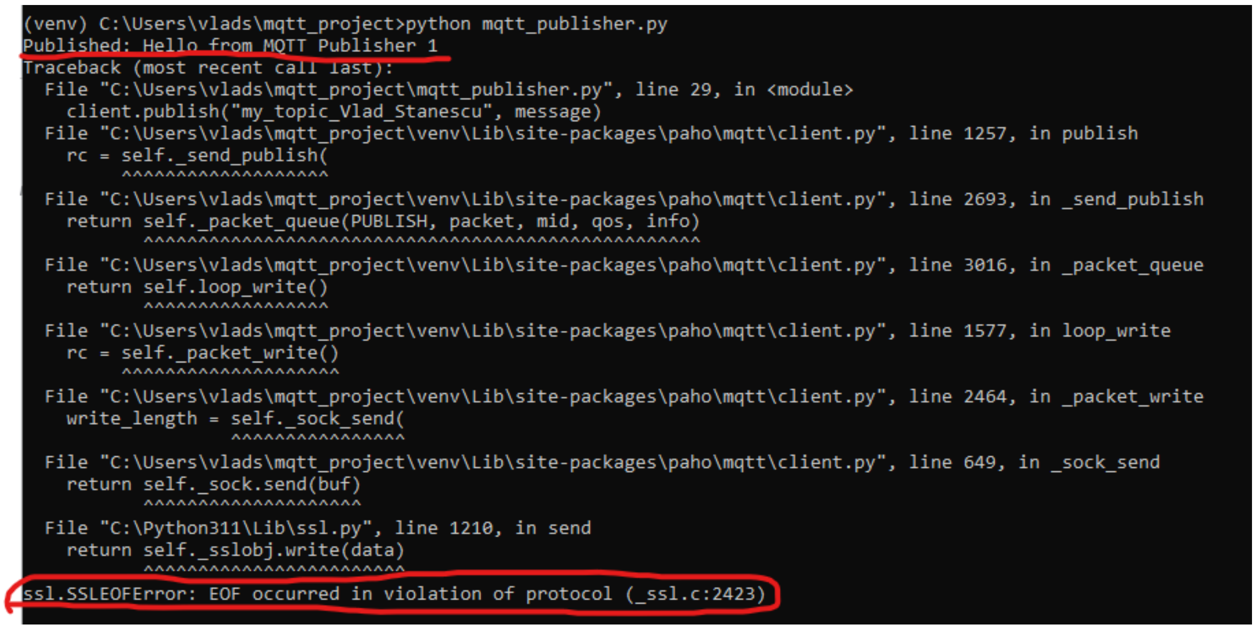


```
# Set SSL/TLS options
client.tls_set(ca_certs="C:/Program Files/mosquitto/ca.crt", certfile="C:/Program Files/mosquitto/server.crt", keyfile="C:/Program Files/mosquitto/server.key
```



```
# Connect to the broker
client.connect("localhost", port=8883)
```

I ran the scripts and after the message from the publisher was printed, I received an error indicating that the SSL/TLS connection between my MQTT client and the Mosquitto broker was terminated unexpectedly. This error often occurs when there's a problem with the SSL/TLS communication, including issues with certificates, handshakes, or the underlying network connection. I tried every step to resolve the issue but I couldn’t.



```
(venv) C:\Users\vlads\mqtt_project>python mqtt_publisher.py
Published: Hello from MQTT Publisher 1
Traceback (most recent call last):
  File "C:\Users\vlads\mqtt_project\mqtt_publisher.py", line 29, in <module>
    client.publish("my_topic_Vlad_Stanescu", message)
  File "C:\Users\vlads\mqtt_project\venv\Lib\site-packages\paho\mqtt\client.py", line 1257, in publish
    rc = self._send_publish(
         ^^^^^^^^^^^^^^^^^^^
  File "C:\Users\vlads\mqtt_project\venv\Lib\site-packages\paho\mqtt\client.py", line 2693, in _send_publish
    return self._packet_queue(PUBLISH, packet, mid, qos, info)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\vlads\mqtt_project\venv\Lib\site-packages\paho\mqtt\client.py", line 3016, in _packet_queue
    return self.loop_write()
           ^^^^^^^^^^^^^^^^^
  File "C:\Users\vlads\mqtt_project\venv\Lib\site-packages\paho\mqtt\client.py", line 1577, in loop_write
    rc = self._packet_write()
         ^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\vlads\mqtt_project\venv\Lib\site-packages\paho\mqtt\client.py", line 2464, in _packet_write
    write_length = self._sock_send(
                   ^^^^^^^^^^^^^^^^
  File "C:\Users\vlads\mqtt_project\venv\Lib\site-packages\paho\mqtt\client.py", line 649, in _sock_send
    return self._sock.send(buf)
           ^^^^^^^^^^^^^^^^^^^^
  File "C:\Python311\Lib\ssl.py", line 1210, in send
    return self._sslobj.write(data)
           ^^^^^^^^^^^^^^^^^^^^^^^^
ssl.SSLEOFError: EOF occurred in violation of protocol (_ssl.c:2423)
```

Here are some methods I tried to solve the error:

1. Checked Network Connection

2. Verified Broker's SSL/TLS Configuration: Double-check the SSL/TLS configuration in my Mosquitto broker “mosquitto.conf” file.

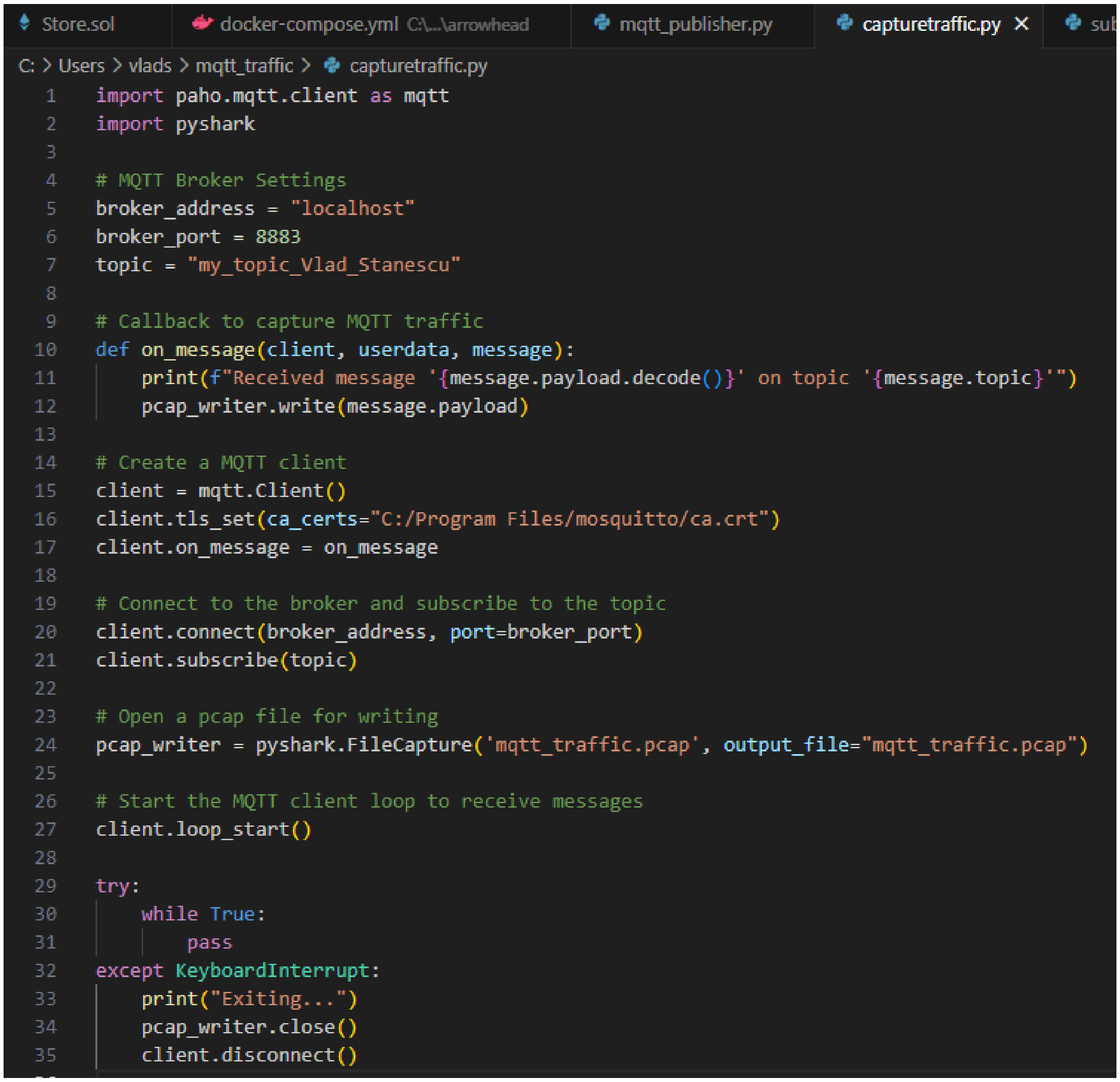
3. Checked for Errors in the Broker Logs: Reviewed the Mosquitto broker's log files for any errors or warnings related to the SSL/TLS connection.

4. Updated Library Versions

5. Restarted Broker

None of them helped me to solve the problem, so I moved to the next step.

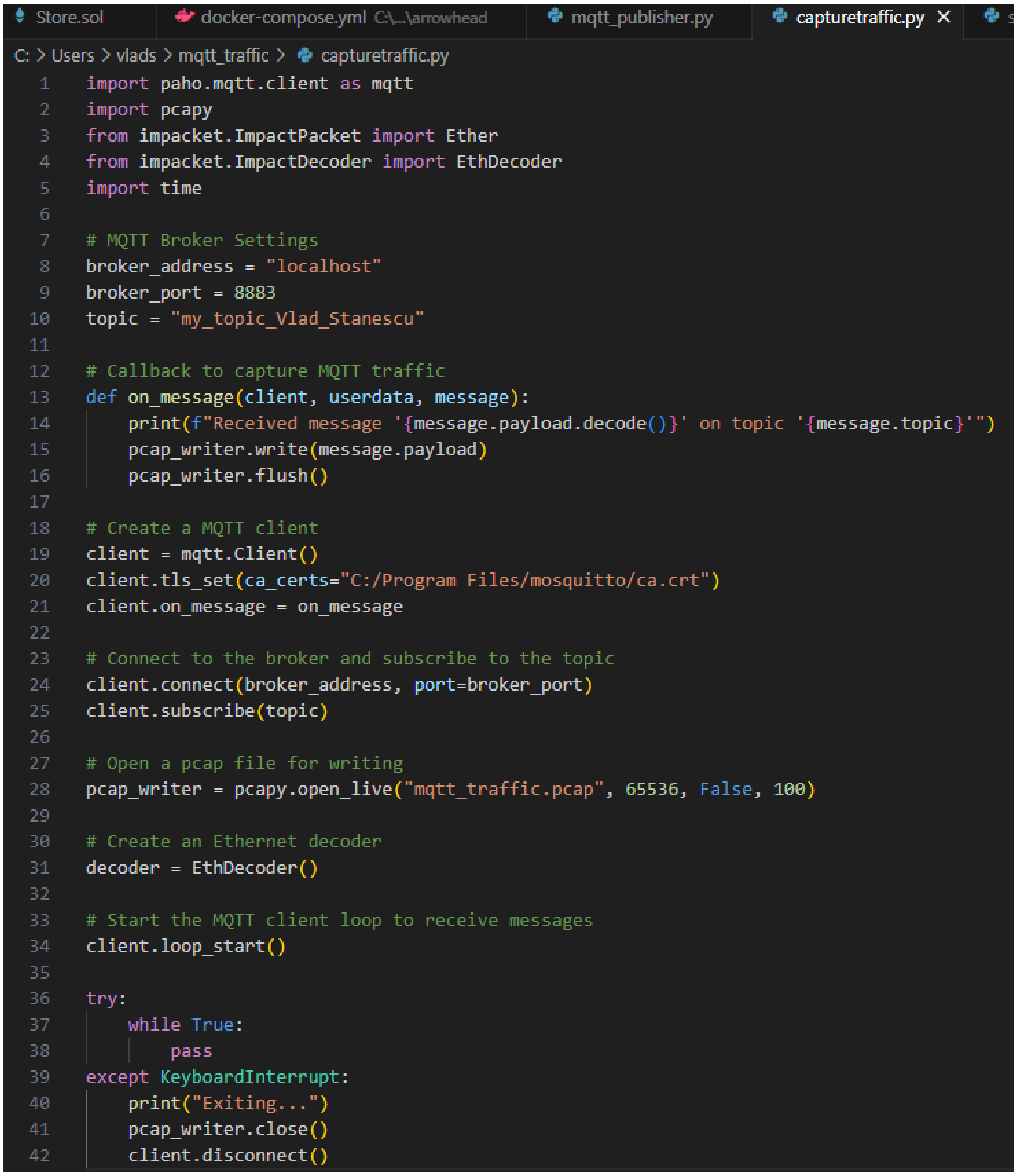
I moved on to create a Python script that captures MQTT traffic and saves it as pcap files.


Store.sol
docker-compose.yml C:\...\arrowhead
mqtt_publisher.py
capturetraffic.py
C: > Users > vlads > mqtt_traffic > capturetraffic.py

```python
import paho.mqtt.client as mqtt
import pyshark

# MQTT Broker Settings
broker_address = "localhost"
broker_port = 8883
topic = "my_topic_Vlad_Stanescu"

# Callback to capture MQTT traffic
def on_message(client, userdata, message):
    print(f"Received message '{message.payload.decode()}' on topic '{message.topic}'")
    pcap_writer.write(message.payload)

# Create a MQTT client
client = mqtt.Client()
client.tls_set(ca_certs="C:/Program Files/mosquitto/ca.crt")
client.on_message = on_message

# Connect to the broker and subscribe to the topic
client.connect(broker_address, port=broker_port)
client.subscribe(topic)

# Open a pcap file for writing
pcap_writer = pyshark.FileCapture('mqtt_traffic.pcap', output_file="mqtt_traffic.pcap")

# Start the MQTT client loop to receive messages
client.loop_start()

try:
    while True:
        pass
except KeyboardInterrupt:
    print("Exiting...")
    pcap_writer.close()
    client.disconnect()
```

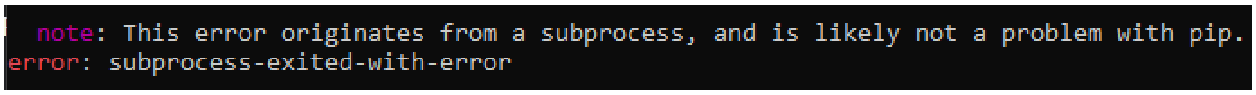
Now it seems that the “pyshark” library does not directly support writing captured packets to a pcap file in this manner. So to capture and save MQTT traffic as a pcap file, I will need to use a different approach.

Here's an alternative approach using the “pcapy” library to capture packets and write them to a pcap file:


Store.sol
docker-compose.yml C:\...\arrowhead
mqtt_publisher.py
capturetraffic.py
C: > Users > vlads > mqtt_traffic > capturetraffic.py

```python
import paho.mqtt.client as mqtt
import pcapy
from impacket.ImpactPacket import Ether
from impacket.ImpactDecoder import EthDecoder
import time

# MQTT Broker Settings
broker_address = "localhost"
broker_port = 8883
topic = "my_topic_Vlad_Stanescu"

# Callback to capture MQTT traffic
def on_message(client, userdata, message):
    print(f"Received message '{message.payload.decode()}' on topic '{message.topic}'")
    pcap_writer.write(message.payload)
    pcap_writer.flush()

# Create a MQTT client
client = mqtt.Client()
client.tls_set(ca_certs="C:/Program Files/mosquitto/ca.crt")
client.on_message = on_message

# Connect to the broker and subscribe to the topic
client.connect(broker_address, port=broker_port)
client.subscribe(topic)

# Open a pcap file for writing
pcap_writer = pcapy.open_live("mqtt_traffic.pcap", 65536, False, 100)

# Create an Ethernet decoder
decoder = EthDecoder()

# Start the MQTT client loop to receive messages
client.loop_start()

try:
    while True:
        pass
except KeyboardInterrupt:
    print("Exiting...")
    pcap_writer.close()
    client.disconnect()
```

When I wanted to install the “pcapy” library, I encountered that error:

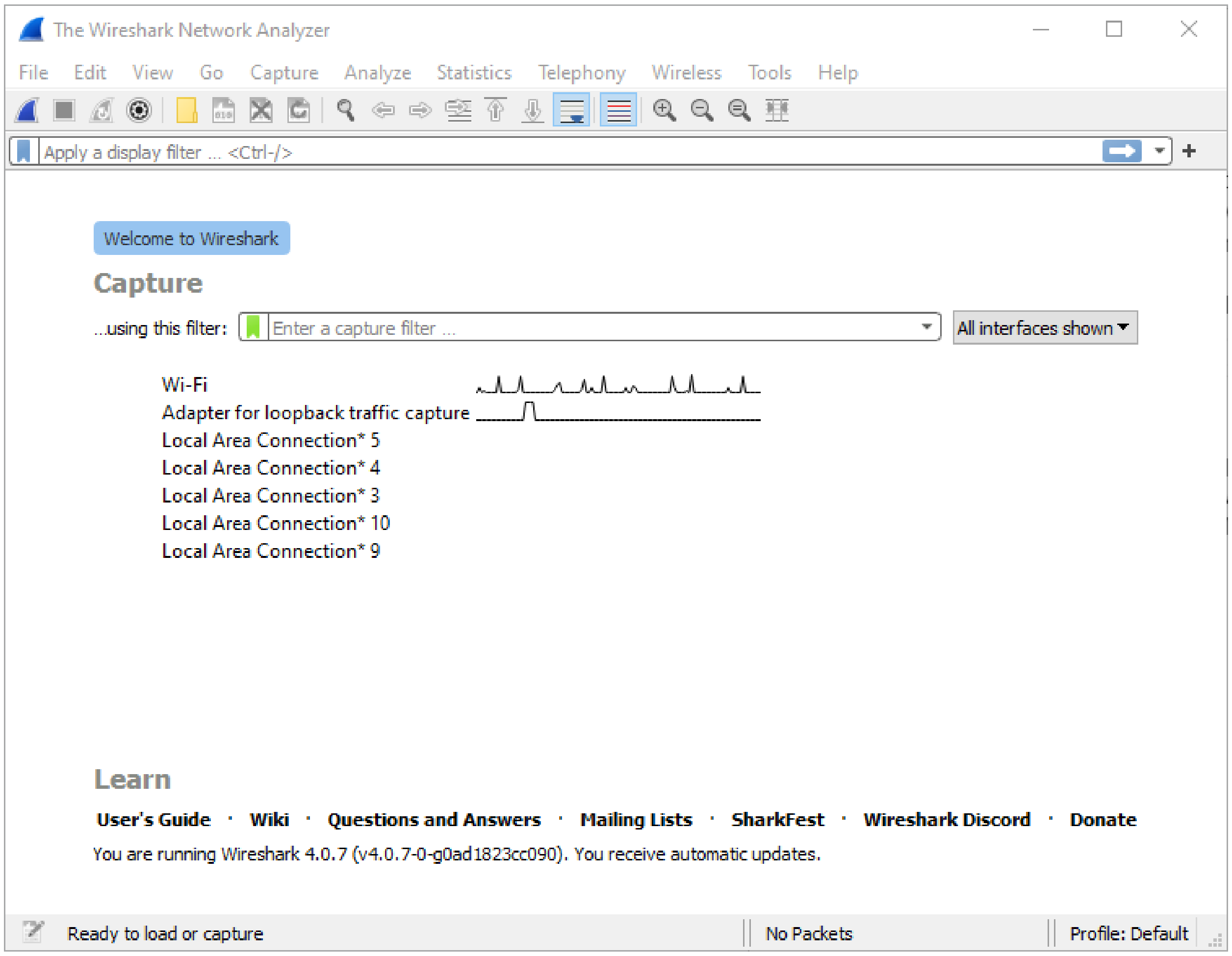


```
  note: This error originates from a subprocess, and is likely not a problem with pip.
error: subprocess-exited-with-error
```

I tried to update setuptools and pip and to use a virtual environment, but the same error appeared. If the “pcapy” library is not available or suitable for my environment, I considered using the “scapy” library, but the same error appeared.

The last solution I had left was to use Wireshark, which is a widely used network protocol analyzer. Wireshark provides a user-friendly interface to capture and analyze network packets. I can capture MQTT traffic by setting a filter for MQTT messages and then save the captured packets to a “pcap” file.

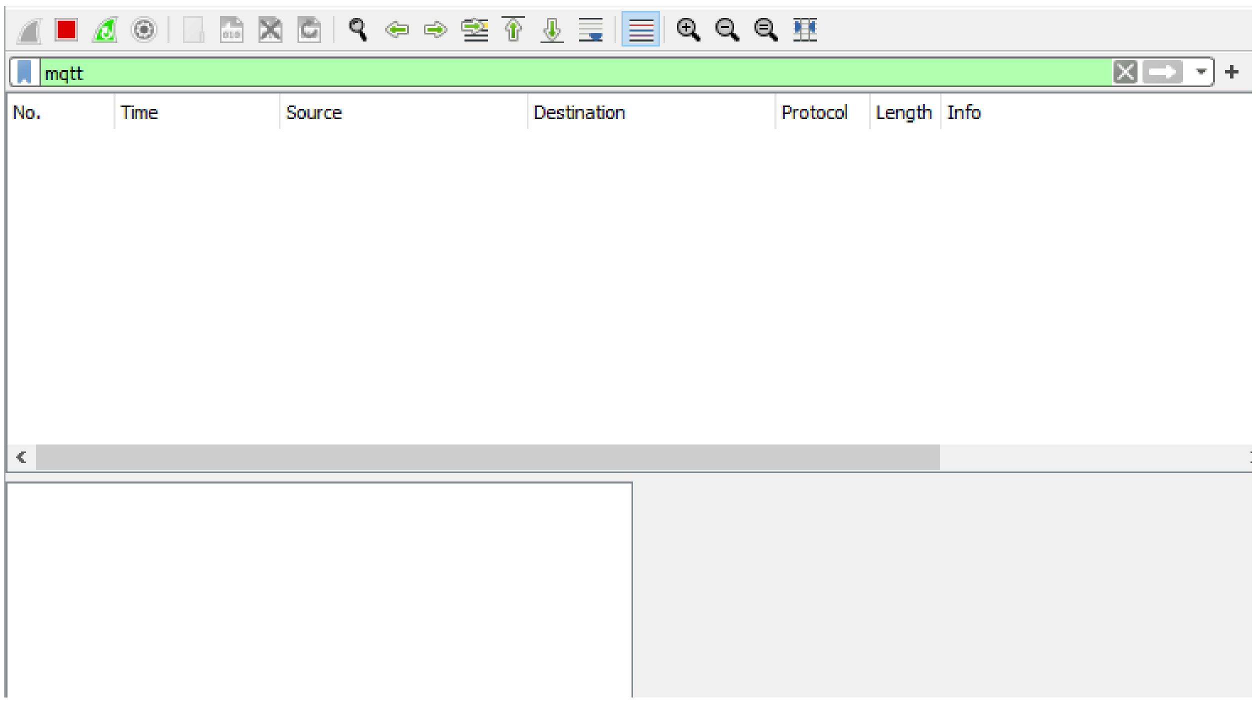
I installed Wireshark, opened it and started capturing packets by selecting the appropriate network interface.While capturing, I applied a display filter to only show MQTT traffic, but wasn't capturing the MQTT signal.


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists · SharkFest · Wireshark Discord · Donate

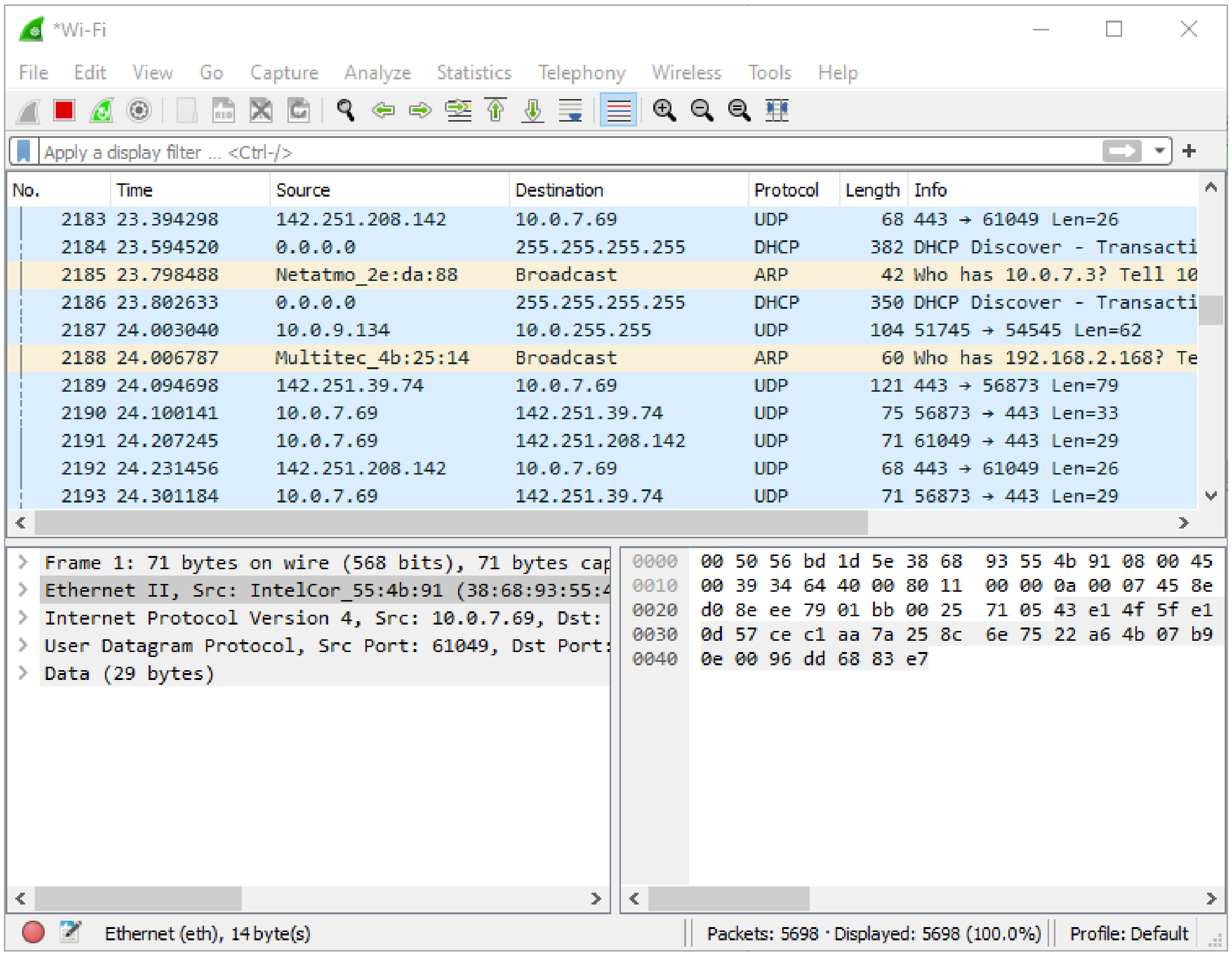
Wi-Fi
Adapter for loopback traffic capture
Local Area Connection* 5
Local Area Connection* 4
Local Area Connection* 3
Local Area Connection* 10
Local Area Connection* 9

You are running Wireshark 4.0.7 (v4.0.7-0-g0ad1823cc090). You receive automatic updates.

Ready to load or capture | No Packets | Profile: Default



```
mqtt
No.   Time   Source   Destination   Protocol   Length   Info
```



*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.    Time       Source            Destination       Protocol  Length  Info
2183   23.394298  142.251.208.142   10.0.7.69         UDP         68  443 → 61049 Len=26
2184   23.594520  0.0.0.0           255.255.255.255   DHCP       382  DHCP Discover - Transacti
2185   23.798488  Netatmo_2e:da:88  Broadcast         ARP         42  Who has 10.0.7.3? Tell 10
2186   23.802633  0.0.0.0           255.255.255.255   DHCP       350  DHCP Discover - Transacti
2187   24.003040  10.0.9.134        10.0.255.255      UDP        104  51745 → 54545 Len=62
2188   24.006787  Multitec_4b:25:14 Broadcast         ARP         60  Who has 192.168.2.168? Te
2189   24.094698  142.251.39.74     10.0.7.69         UDP        121  443 → 56873 Len=79
2190   24.100141  10.0.7.69         142.251.39.74     UDP         75  56873 → 443 Len=33
2191   24.207245  10.0.7.69         142.251.208.142   UDP         71  61049 → 443 Len=29
2192   24.231456  142.251.208.142   10.0.7.69         UDP         68  443 → 61049 Len=26
2193   24.301184  10.0.7.69         142.251.39.74     UDP         71  56873 → 443 Len=29
```

```
> Frame 1: 71 bytes on wire (568 bits), 71 bytes cap
> Ethernet II, Src: IntelCor_55:4b:91 (38:68:93:55:4
> Internet Protocol Version 4, Src: 10.0.7.69, Dst:
> User Datagram Protocol, Src Port: 61049, Dst Port:
> Data (29 bytes)
```

```
0000  00 50 56 bd 1d 5e 38 68  93 55 4b 91 08 00 45
0010  00 39 34 64 40 00 80 11  00 00 0a 00 07 45 8e
0020  d0 8e ee 79 01 bb 00 25  71 05 43 e1 4f 5f e1
0030  0d 57 ce c1 aa 7a 25 8c  6e 75 22 a6 4b 07 b9
0040  0e 00 96 dd 68 83 e7
```

```
Ethernet (eth), 14 byte(s)    Packets: 5698 · Displayed: 5698 (100.0%)    Profile: Default
```