Probando MQTT con Python seguro.

Grupo 3 (Garai, Pablo y Markel)

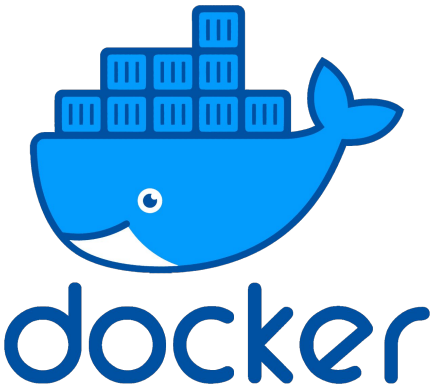
Objetivo del proyecto

El objetivo principal de este proyecto es demostrar cómo montar un broker MQTT seguro gracias a establecer una autenticación basada en certificados de cliente y servidor.

Durante esta presentación se mostrarán los pasos seguidos para montar este sistema tanto con scripts de Python como por línea de comandos.

Decisión de usar o no contenedores

Tras valorar los pros y contras de la utilización de contenedores para este trabajo, finalmente se ha optado por su uso, ya que si se entienden los fundamentos y el funcionamiento de Docker y de Docker Compose, es bastante más sencillo realizar este tipo de retos.

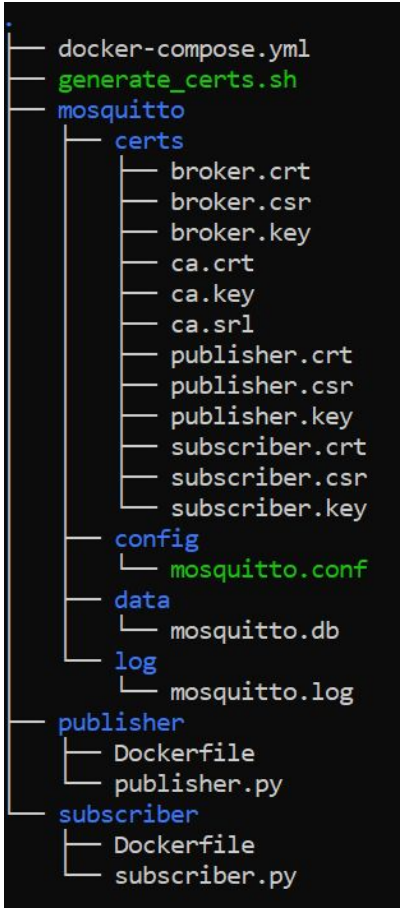


Ejemplo de uso

Para este reto hemos elegido utilizar como ejemplo de la comunicación MQTT, unas suscripciones a periódicos y secciones de periódicos.

El suscriptor podrá suscribirse a periódicos completos o a subsecciones de los mismos, además, también podrá suscribirse a todos los periódicos pero solo a secciones determinadas.

Descripción de nuestro ejemplo
-
Estructura de carpetas



Generación de los certificados

Para garantizar que la comunicación MQTT es segura, se configura MQTT con TLS mediante certificados.

Para generar los certificados, hemos generado un script de bash de Linux el cual genera paso por paso lo necesario para una comunicación MQTT segura mediante TLS.

A continuación se describen los pasos seguidos en el script.

Generación de certificados

Paso 1: Configuración inicial y creación de la entidad certificadora (CA).

```
#!/bin/bash

# Directorio donde se guardarán los certificados
CERTS_DIR="mosquitto/certs"
mkdir -p $CERTS_DIR
cd $CERTS_DIR

echo "🔷 Generando Autoridad Certificadora (CA)..."
openssl genpkey -algorithm RSA -out ca.key
openssl req -new -x509 -days 365 -key ca.key -out ca.crt -subj "/C=ES/ST=Álava/L=Vitoria-
Gasteiz/O=Deusto/OU=IoT/CN=Deusto CA"
```

Generación de certificados

Paso 2: Generación de certificados para el broker

```
echo "🔹 Generando clave y CSR para el broker (Mosquitto)..."
openssl genpkey -algorithm RSA -out broker.key
openssl req -new -key broker.key -out broker.csr -subj "/C=ES/ST=Álava/L=Vitoria-
Gasteiz/O=Deusto/OU=IoT/CN=mosquitto"

echo "🔹 Firmando el certificado del broker con la CA..."
openssl x509 -req -in broker.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out broker.crt -days 365
```

Generación de certificados

Paso 3: Generación de certificados para los clientes (publicador/suscriptor)

```
echo "🔹 Generando clave y CSR para el publicador..."
openssl genpkey -algorithm RSA -out publisher.key
openssl req -new -key publisher.key -out publisher.csr -subj "/C=ES/ST=Álava/L=Vitoria-
Gasteiz/O=Deusto/OU=IoT/CN=publisher"

echo "🔹 Firmando el certificado del publicador con la CA..."
openssl x509 -req -in publisher.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out publisher.crt -days 365

echo "🔹 Generando clave y CSR para el suscriptor..."
openssl genpkey -algorithm RSA -out subscriber.key
openssl req -new -key subscriber.key -out subscriber.csr -subj "/C=ES/ST=Álava/L=Vitoria-
Gasteiz/O=Deusto/OU=IoT/CN=subscriber"

echo "🔹 Firmando el certificado del suscriptor con la CA..."
openssl x509 -req -in subscriber.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out subscriber.crt -days
365
```

Comunicación MQTT - Publicador

A continuación se muestra el archivo **publisher.py** el cual va publicando artículos preestablecidos en una lista cada 10 segundos.

```python
# Crear el cliente MQTT
client = mqtt.Client(client_id="news-publisher")
# Configurar TLS usando los certificados del publisher
client.tls_set(
    ca_certs="/mosquitto/certs/ca.crt",
    certfile="/mosquitto/certs/publisher.crt",
    keyfile="/mosquitto/certs/publisher.key",
    tls_version=ssl.PROTOCOL_TLSv1_2
)
client.tls_insecure_set(False)
broker = "mosquitto"  # En la red de Docker, el broker se llama "mosquitto"
port = 8883
client.connect(broker, port)
client.loop_start()
print("📢 Publicador de artículos activo. Enviando artículos cada 10
segundos.")
articles = [
    {
        "newspaper": "marca",
        "category": "futbol",
        "title": "El triunfo de Deusto FC",
        "content": "El Deusto FC gana en un partido épico."
    },

    ....
]
try:
    while True:
        article = random.choice(articles)
        topic = f"articulos/{article['newspaper']}/{article['category']}"
        message = f"Título: {article['title']} | Contenido: {article['content']}"
        print(f"📰 Publicando en {topic}: {message}")
        client.publish(topic, payload=message, qos=2)
        time.sleep(10)
except KeyboardInterrupt:
    print("🛑 Cerrando publicador...")
client.loop_stop(
client.disconnect()
```

Comunicación MQTT - Publicador

```
FROM python:3.9-slim
RUN pip install "paho-mqtt<2.0.0"
WORKDIR /app
COPY publisher.py .
CMD ["python", "-u", "publisher.py"]
```

Aquí podemos ver el archivo **DockerFile**, que a partir de una versión slim de Python genera una imagen Docker la cual está diseñada para ejecutar el archivo publisher.py, y la cual utiliza la biblioteca **paho-mqtt** (una biblioteca para la comunicación MQTT en Python).

Comunicación MQTT - Suscriptor

A continuación se muestra el archivo **suscriber.py** el cual se suscribe a los topics que se le indiquen en la lista con ese nombre (en este ejemplo se suscribe a las secciones de fútbol de todos los periódicos y a la sección de política de “El País”

```python
def on_message(client, userdata, msg):
    print(f"📩 Mensaje recibido en {msg.topic}:
{msg.payload.decode()}")
client = mqtt.Client(client_id="news-subscriber")
client.tls_set(
    ca_certs="/mosquitto/certs/ca.crt",
    certfile="/mosquitto/certs/subscriber.crt",
    keyfile="/mosquitto/certs/subscriber.key",
    tls_version=ssl.PROTOCOL_TLSv1_2
)
client.tls_insecure_set(False)
client.on_message = on_message
broker = "mosquitto"
port = 8883
client.connect(broker, port)
# Ejemplo: suscribirse a dos topics
topics = [
    ("articulos/+/futbol", 2),
    ("articulos/elpais/politica", 2)
]
client.subscribe(topics)
client.loop_start()
print("🔗 Conectado al broker MQTT.")
print("📥 Suscrito a los siguientes topics:")
for t, _ in topics:
    print(f"  - {t}")
try:
    while True:
        pass
except KeyboardInterrupt:
    print("🛑 Cerrando subscriber...")
client.loop_stop()
client.disconnect()
```

Comunicación MQTT - Suscriptor

```
FROM python:3.9-slim
RUN pip install "paho-mqtt<2.0.0"
WORKDIR /app
COPY subscriber.py .
CMD ["python", "-u", "subscriber.py"]
```

De manera similar al caso del publicador, el archivo **DockerFile** presenta cómo genera una imágen Docker diseñada, en este caso, para ejecutar el script suscriber.py

Demo

Prueba de funcionamiento del proyecto en directo, con el uso de comandos y sin ellos.

Vídeo Demo:

https://drive.google.com/file/d/1PP2NE2EnTba0R-K8ptP56ONsnodui1uf/view?usp=sharing