IOT MONITORING
PLATFORM

Duțu Alin Călin

CONTENTS

1 INTRODUCTION

1.1 Context

The Internet of Things (IoT) refers to the network of physical objects embedded with sensors, software, and other technologies to connect and exchange data with other devices and systems over the Internet. These "smart" objects can collect, send, and act on data acquired from their environments, often without human intervention. To analyze the data, it is required to create a pipeline to gather the data offered by the sensors and then process it into a useful output.

1.2 Objectives

This program aims to create a simple pipeline that receives the data from sensors, stores and processes it to provide an output in a graph form where the user can visualize the data distribution over a certain period of time, draw conclusions or further process the data for diverse purposes.

The following objectives will be covered after implementing the pipeline:

- Understanding of the mechanism of MQTT, a well-known protocol in the IOT industry
- Understanding the concepts of virtualization and interconnectivity between virtual machines
- Exploring multiple tools and their impact on a multi-functional pipeline

2 PROPOSED SOLUTION

This project aims to implement a pipeline of apps for collecting, storing, and visualizing data received from a significant number of IOT devices. The architecture for the app pipeline implies a simplified version of public cloud services used in real-world applications. However, it is still designed with efficiency as a main objective for this project.

Looking from a general point of view, there are 4 services that perform different actions and interactions to ensure data processing:

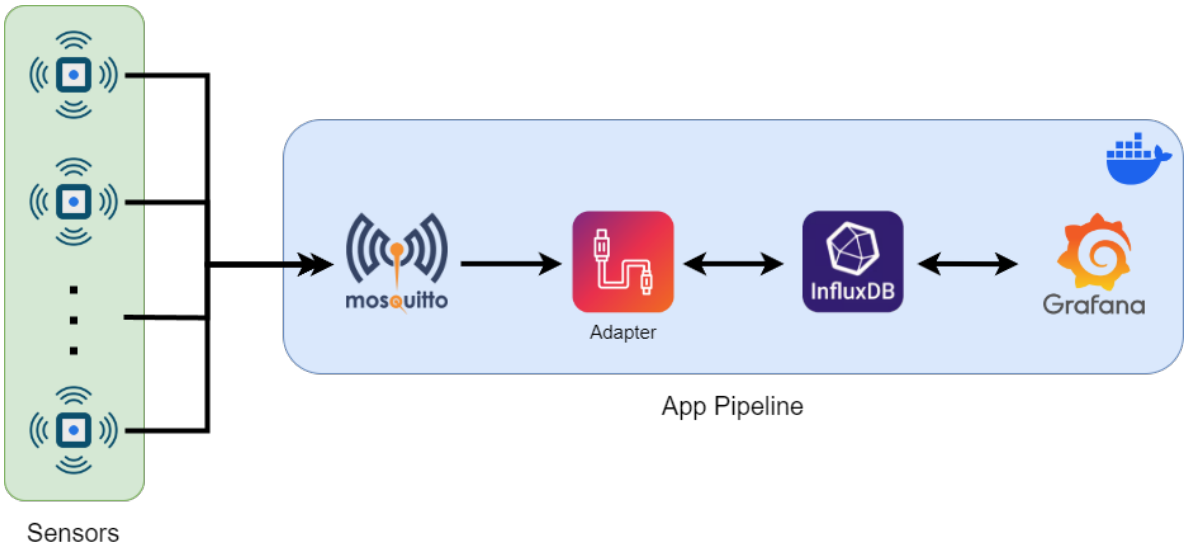

Figure 1: Application architecture [1]

The sensors will be represented by programs implemented by the SPRC team [1] that will simulate the behavior of a real sensor and send pre-defined data as input to the app pipeline.

The App pipeline consists of 4 services:

- Mosquitto broker - an open-source tool that will manage the sensor subscription and data transmission provided by the sensors through topics.[2].
- Adapter - Used to gather all the data the broker received and send it to the InfluxDB database.

[1] `https://www.flaticon.com/icons`. Icons created by RawPixel

- InfluxDB - Stores the data provided by the adapter making it available to the Grafana tool through the provided API[3].
- Grafana - Uses the data provided by the InfluxDB database API as input to create charts for visualizing data over a certain period of time given the set configuration[4].

The communication between these 4 services is covered by Docker[5], a well-known platform designed to help developers build, share, and run container applications. The enumerated services will be built as isolated containers connected through internal networks that ensure data transmission between services. The app will be released in clusters, one for each service, using the Docker Swarm mode[6].

3 IMPLEMENTATION DETAILS

3.1 Proposed implementation

The proposed solution mainly consists of an application pipeline that gathers the data provided by multiple IoT sensors, stores it in a local database configured specifically for various IoT data streams, and creates a dashboard with charts for visualizing data over a certain period of time.

3.1.1 Message broker

The message broker will be implemented using the Mosquitto broker, an open-source tool for message distribution that implements the MQTT protocol[7]. In this case, the broker will only be equipped with the main functionality. So, the broker after initialization, will wait to receive new requests.

The first type of request is from the sensors which will send data on a specific topic defined by each one of them. The broker will create a new topic if it hasn't already been created and associate the received data with it.

The second type of request is from the adapter which will ask to subscribe to all the topics created by the sensors. As soon as the broker honors the request, the adapter will receive the stored data from all the topics associated with the sensors and all the upcoming updates, so that a message received from a sensor will result in a message sent to the adapter with the provided sensor data.

3.1.2 Adapter

The adapter is a program written in Python used to adapt the data stream from sensors to a format accepted by the InfluxDB database. At initialization, the adapter will set a connection with the database first and then send a request to subscribe to all broker's topics.

After initialization, the adapter will wait for the broker to send messages with sensor data. As soon as one is received, it will be converted to the following JSON format:

```
{
    "measurement": statie.cheie (string),
    "tags": {
        "location": locatie (string),
        "station": statie (string)
    },

    "time": timestamp (string),
    "fields": {
        "value": valoare (float)
    },
}
```

The converted data will be sent to InfluxDB to be stored.

3.1.3 Database

The database used is an instance of InfluxDB, a vertical scaling database capable of withstanding multiple data streams that send a small amount of data in a short time, such as multiple IoT devices[3]. The input comes from the adapter as a database command to store the data. The stored data will be used by Grafana to create the charts for data visualization.

3.1.4 Grafana

For data visualization, Grafana[4] is a great choice for this specific pipeline as it is an open-source analytics and monitoring solution capable of gathering the sensor data from the InfluxDB database using the database API for connection and data selection and creating dashboard and configurable charts either programmatic utilizing a specific JSON format or manually on its user interface to visualize data over a certain amount of time.

3.1.5 Docker

Docker[5] is the main component that makes the pipeline creation possible. Docker is a platform designed to help developers build, share, and run container applications. In this case, Docker helps to build apps and deploy them as services in isolated containers, create internal network connections between them as shown in the architecture diagram (Figure 1), expose the broker service to the outside world, and monitor the activity in the pipeline. Along with the Docker Swarm mode [6], the app can be easily monitored and released to production.

3.2 Testing

For testing the proposed solution, the manual method is the best choice as it can be used for testing the pipeline configuration, performance, and practicality in all possible conditions.

4 CONCLUSIONS

In conclusion, this program aims to create a simple pipeline designed to gather data from multiple IoT sensors and process it to provide an output in a chart form where the user can visualize the data distribution over a certain period of time, and draw conclusions or further process the data for other purposes.

BIBLIOGRAPHY

[1] SPRC team @ Polytechnic University of Bucharest. Systems for Programming Network Computers course. https://curs.upb.ro/2022/mod/resource/view.php?id=45681. The course is written in Romanian; Latest access: 14.07.2024;.

[2] Eclipse Foundation. Eclipse Mosquitto. https://mosquitto.org/. Latest access: 14.07.2024.

[3] InfluxData. InfluxDB. https://www.influxdata.com/. Latest access: 14.07.2024.

[4] GrafanaLabs. Grafana. https://grafana.com/. Latest access: 14.07.2024.

[5] Docker. Docker. https://www.docker.com/. Latest access: 14.07.2024.

[6] Docker. Docker Swarm Docs. https://docs.docker.com/engine/swarm/stack-deploy/. Latest access: 14.07.2024.

[7] Eclipse Foundation. MQTT prtocol. https://mqtt.org/. Latest access: 14.07.2024.