IoT- From the Microcontroller to the Cloud

The goal of this project is to send the data from a sensor to the cloud.

We have used RIOT OS as an embedded operating system for IoT that provides several useful functions and tools to create, manage network connections and work with protocols and message brokers.

Finally, we will send the data to the AWS-Cloud a cloud service provider to store and present the sensor collected data to the users.

Components

Operating systems

We started to work on the Project with a Windows 10 Operating system and a Linux Ubuntu which is installed on the VMware workstation virtual machine.

Boards

NRF52840dk

The nRF52840 Development Kit (nRF52840 DK) is a versatile and powerful development platform designed by Nordic Semiconductor for creating and prototyping Bluetooth Low Energy (BLE), Bluetooth 5, and other wireless applications. It is built around the nRF52840 SoC (System-on-Chip), which is a feature-rich and highly flexible chip ideal for IoT (Internet of Things) and wireless communication projects. The nRF52840 DK provides a range of connectivity options, including USB, NFC (Near Field Communication), and a wide array of GPIO pins. It supports a variety of development environments, including Nordic's nRF5 SDK and the Arduino IDE, making it a popular choice for both beginners and experienced developers in the field of wireless IoT and wearable technology.

We will extract the Temperature data from the internal temperature sensor of the board.

NRF52840 Dongle

The nRF52840 Dongle is a compact and versatile USB development tool created by Nordic Semiconductor. It is based on the nRF52840 SoC (System-on-Chip) and is designed for evaluating and prototyping Bluetooth Low Energy (BLE) and other wireless applications. This dongle serves as a development platform and can be plugged into a USB port on a computer, making it an excellent tool for testing and debugging wireless devices and applications. It provides a range of features, including a built-in PCB antenna, support for a wide array of development environments like the nRF5 SDK and the Arduino IDE, and can be easily used for development and testing of various IoT and Bluetooth projects.

Saul

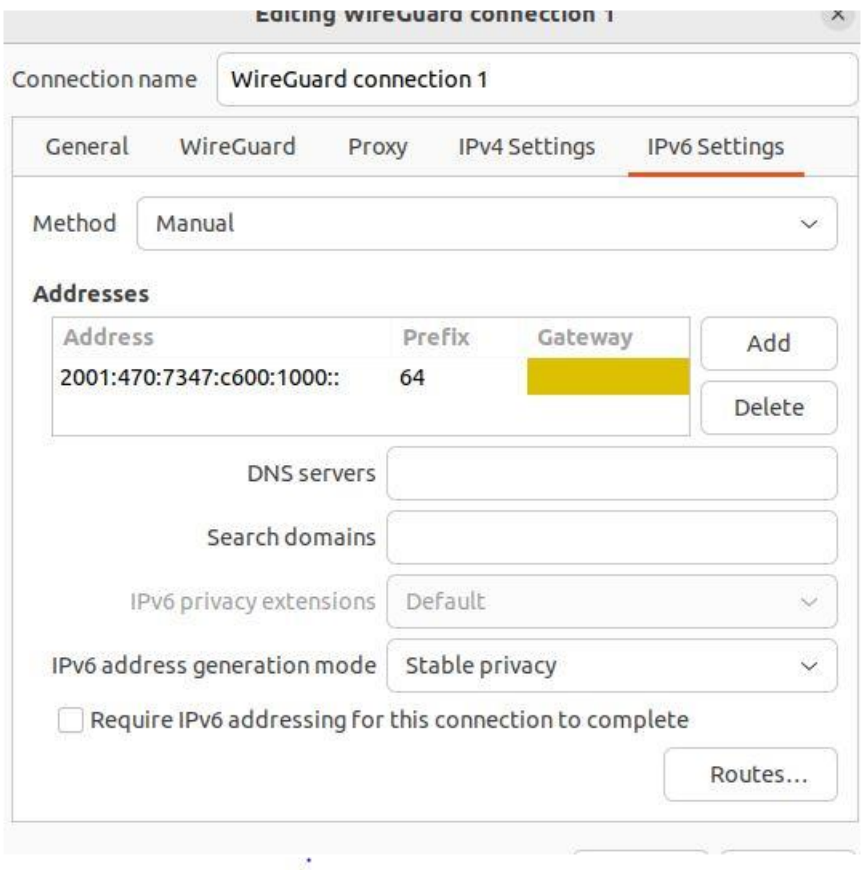
The SAUL interface is used to collect temperature data from the internal sensor.

Setting up the Wireguard VPN tunnel:

1.We open the application drawer in Ubuntu and search for the advanced network connections.

2.After making a new connection the connection type must be set as Wireguard.

3.After that we add some details that provided by the Professor. (see the images below)



Editing WireGuard connection 1

Connection name: WireGuard connection 1

Tabs: General | WireGuard | Proxy | IPv4 Settings | IPv6 Settings

Method: Manual

Addresses

Address	Prefix	Gateway
2001:470:7347:c600:1000::	64	🟨

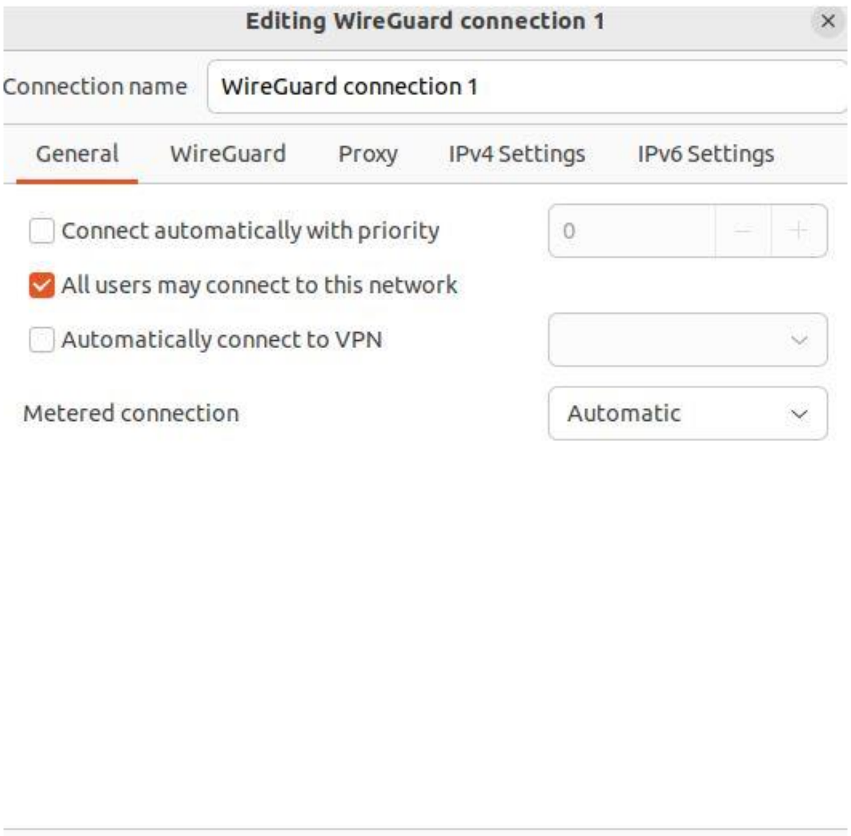
DNS servers:

Search domains:

IPv6 privacy extensions: Default

IPv6 address generation mode: Stable privacy

☐ Require IPv6 addressing for this connection to complete



Editing WireGuard connection 1

Connection name: WireGuard connection 1

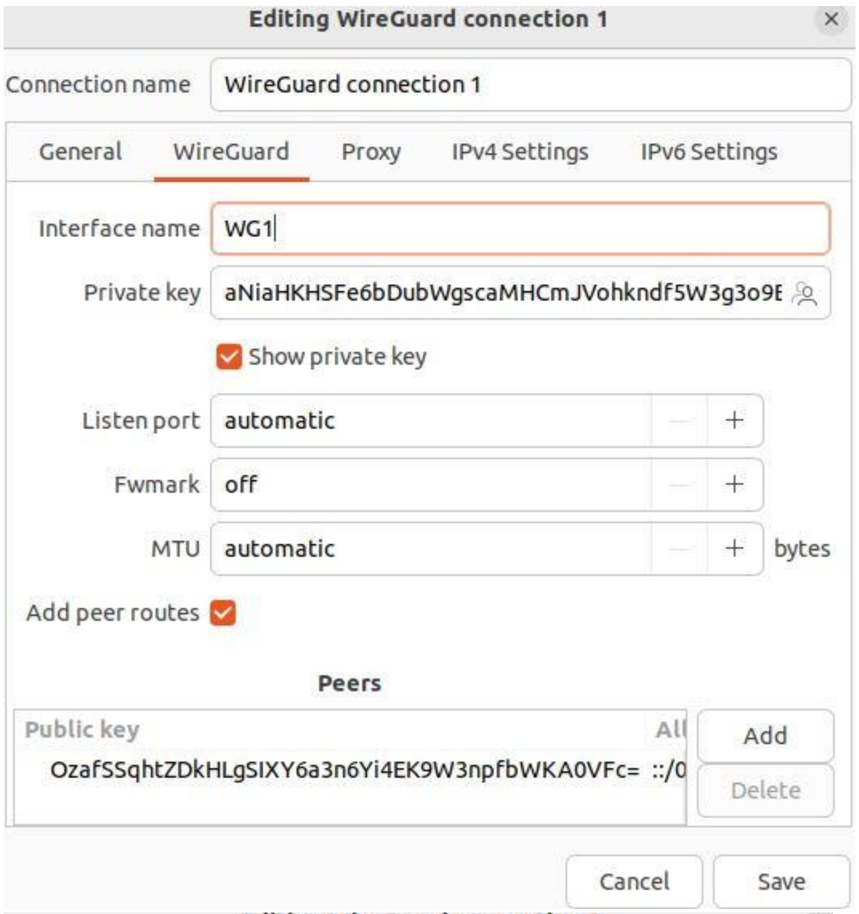
Tabs: General | WireGuard | Proxy | IPv4 Settings | IPv6 Settings

☐ Connect automatically with priority 0

☑ All users may connect to this network

☐ Automatically connect to VPN

Metered connection: Automatic



Editing WireGuard connection 1

Connection name: WireGuard connection 1

General | WireGuard | Proxy | IPv4 Settings | IPv6 Settings

Interface name: WG1

Private key: aNiaHKHSFe6bDubWgscaMHCmJVohkndf5W3g3o9E

☑ Show private key

Listen port: automatic

Fwmark: off

MTU: automatic bytes

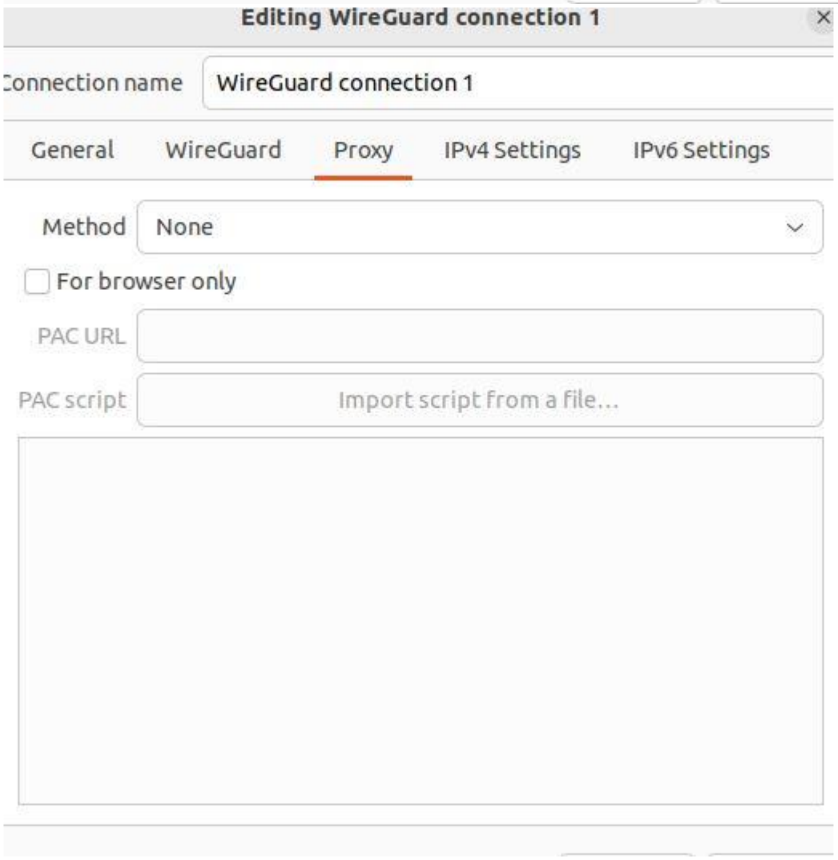
Add peer routes ☑

Peers

Public key	All
OzafSSqhtZDkHLgSIXY6a3n6Yi4EK9W3npfbWKA0VFc=	::/0

Add | Delete

Cancel | Save



Editing WireGuard connection 1

Connection name: WireGuard connection 1

General | WireGuard | Proxy | IPv4 Settings | IPv6 Settings

Method: None

☐ For browser only

PAC URL:

PAC script: Import script from a file…

After the settting we open a terminal and type *nmtui*

From there we can activate our wireguard connection.

We need to enable IPV6 forwarding for the project using this code:

```
sudo sysctl -w net.ipv6.conf.all.forwarding=1
```

First of all, we need to clone the last update of RIOT OS Repository from github.

```
git clone https://github.com/RIOT-OS/RIOT.git
```

Then, we clone the folder from github repository and copy and paste this folder inside of RIOT OS example folder.

Setup gnrc-border-router on the NRF52480 dongle

After connecting the dongle to the USB-Port we navigate to the

```
cd RIOT/examples/gnrc_border_router
```

We will add some lines to the makefile in the folder of gnrc-border-router:

```
DEFAULT_CHANNEL := 24
```

```
UPLINK ?= cdc-ecm
PREFIX_CONF := uhcp
```

Then we should install the Nordic nrfutil packages. Without installation we can’t flash the program on the dongle.

```
.nrfutil install nrf5sdk-tools
```

Permissions for Serial port: In order to access the serial port for flashing the program on the device we run the following code in terminal:

```
sudo dpkg -i nrf-udev_1.0.1-all.deb
```

We will find the USB-Port of dongle using this code:

```
sudo ls -l /dev/ttyACM*
```

Running the border router application

```
PORT=/dev/ttyACM0 IPV6_PREFIX=2001:470:7347:c601::/64 BOARD=nrf52840dongle make flash term
```

After successfully running the example we should always keep this terminal open.

How to flash the application on the Board

We need to unlock the board. Sometimes we will face some problems with flashing and receive an error regarding to unlock the board. We use the following code in linux terminal:

nrfjprog --family nRF52 –recover

After that we need to install some tools packages for using the NRF52480dk board.

sudo apt install gcc-arm-none-eabi

nrf-command-line-tools_10.21.0_amd64.deb

sudo dpkg -i nrf-command-line-tools_10.21.0_amd64.deb

We need to find the USB-Port that belongs to every connected device. In order to find the specific port of each device (dongle and normal board) we type this code in the linux terminal:

*sudo ls -l /dev/ttyACM**

How to start the Temperature application

We start the application using the following pattern:

We clone the folder of project from github and paste it into the RIOT/Examples

We Navigate to the folder through the terminal and run this command:

PORT=/dev/ttyACM1 BOARD=nrf52840dk make term flash PROGRAMMER=openocd

start aws_ipv6_address broker_port

For example: start 2600:1f18:6fc6:8704:33fc:66d7:3d55:20c3 1885

Setup an EC2 instance

We use AWS academy learner lab for students.

After logging into the website, navigate to modules/Launch learner lab.

Press the play button to start the lab. After seeing the green light, we should click on it to enter the AWS management console.

Create a VPC with IPV6

In Amazon Web Services (AWS), VPC stands for "Virtual Private Cloud." A VPC is a logically isolated section of the AWS cloud where you can launch AWS resources, such as Amazon EC2 instances, RDS databases, and more. VPC allows you to define your own virtual network environment, including configuring your own IP address range, subnets, route tables, and network gateways.

1.We should navigate to VPC

2.Then Click on Create VPC.Then Choose IPv6 CIDR block and provide the IPv6 block we want to use.

3.Click on Create VPC.

Subnets and Internet gateways

1.After creating a VPC we should choose the subnets from the sidebar.

2. We will see a list of subnets.

3. We should select each subnet and from actions menu we should enter IPv6 CIDR block and provide the IPv6 block for the subnets.

4.After changes we should select internet gateways from the sidebar.

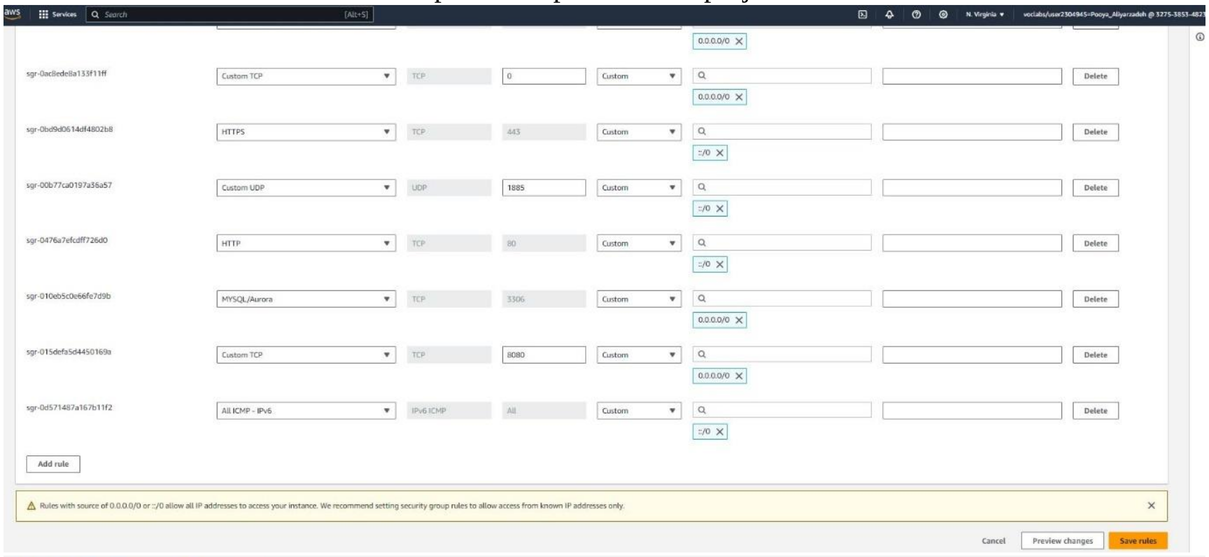
5. We should create a gateway and attach the gateway to the VPC.

Create security group

1.We select security groups from the sidebar.

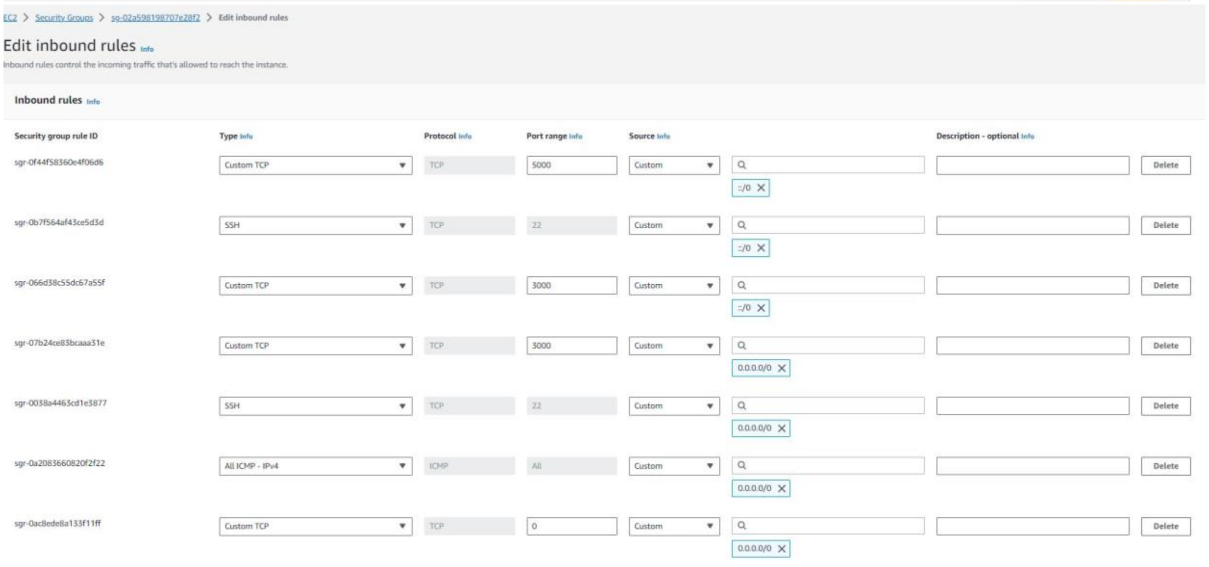
2.After that click on Create security group.

3.We will edit the inbound rules to open some ports for the project as bellow:



					0.0.0.0/0		
sgr-0ac8ede8a133f11ff	Custom TCP	TCP	0	Custom	0.0.0.0/0		Delete
sgr-0bd9d0614df4802b8	HTTPS	TCP	443	Custom	::/0		Delete
sgr-00b77ca0197a36a57	Custom UDP	UDP	1885	Custom	::/0		Delete
sgr-0476a7efcdff726d0	HTTP	TCP	80	Custom	::/0		Delete
sgr-010eb5c0e66fe7d9b	MYSQL/Aurora	TCP	3306	Custom	0.0.0.0/0		Delete
sgr-015defa5d4450169a	Custom TCP	TCP	8080	Custom	0.0.0.0/0		Delete
sgr-0d571487a167b11f2	All ICMP - IPv6	IPv6 ICMP	All	Custom	::/0		Delete

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.


EC2 > Security Groups > sg-02a598198707e28f2 > Edit inbound rules

Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source		Description - optional	
sgr-0f44f58360e4f06d6	Custom TCP	TCP	5000	Custom	::/0		Delete
sgr-0b7f564af43ce5d3d	SSH	TCP	22	Custom	::/0		Delete
sgr-066d38c55dc67a55f	Custom TCP	TCP	3000	Custom	::/0		Delete
sgr-07b24ce83bcaaa31e	Custom TCP	TCP	3000	Custom	0.0.0.0/0		Delete
sgr-0038a4463cd1e3877	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-0a2083660820f2f22	All ICMP - IPv4	ICMP	All	Custom	0.0.0.0/0		Delete
sgr-0ac8ede8a133f11ff	Custom TCP	TCP	0	Custom	0.0.0.0/0		Delete

Create an Instance

1.we select Launch instance from the console.

2.After clicking Launch instance we should choose the last version of Ubuntu image.

3. Select t2.micro as the instance type.

4. We should select a keypair. We need to enter the name of keypair and save it as a .pem file on our local machine. IMPORTANT: We will need this file for future steps.

5. In section of network settings we should choose the created VPC and one of its subnets.

6.We will select the existing Security group that we created before.

7. For storage we can select up to 30GB for free. We will select 20GB.

8. Finally we can run our Instance.

Elastic IP Creation:

When accessing an EC2 instance, the IP address would alter each time we logged in. To rectify this, we assigned an Elastic IP (EIP) with a static IPv4 address, ensuring its stability even when the instance is stopped or started. You can access our project via the following unchanging IP address: http://54.158.73.35.

How to create:

1.From EC2 dashboard we select Elastic IP which is under the network and security in side panel.

2.Then we click on Allocate elastic IP address button.

3.We allocate an IPV4 as the network border group

4. We can Assosiate the elastic IP to a special instance.

Connecting to the instance:

After selecting the instance in the console we can connect to it. We click on the connect button and the linux terminal will be opened.

We need to Update and install paho MQTT on the instance.

sudo apt update

sudo apt install python3-pip

pip3 install paho-mqtt

sudo apt install awscli

Mosquitto RSMB Broker

RSMB, or Really Small Message Broker, is a lightweight and open-source message broker implementation that supports the MQTT (Message Queuing Telemetry Transport) and MQTT-SN (MQTT for Sensor Networks) protocols. It is designed to be highly efficient, making it well-suited for resource-constrained devices and low-bandwidth networks

Data transfer from the sensor node to AWS can be achieved using port 1885, which is the MQTT-SN (MQTT for Sensor Networks) port. Additionally, port 1886 is designated for MQTT (Message Queuing Telemetry Transport) and is monitored by the mqtt_subscriber_client for data reception.

We should clone the repository and navigate to src folder

```
git clone https://github.com/eclipse/mosquitto.rsmb.git

cd mosquitto.rsmb/rsmb/src
```

Then we open the notepad and save the following text as config.conf data

```
# add some debug output

trace_output protocol

# listen for MQTT-SN traffic on UDP port 1885

listener 1885 INADDR_ANY mqtts

ipv6 true

# listen to MQTT connections on tcp port 1886

listener 1886 INADDR_ANY

ipv6 true
```

Accessing the EC2 Instance from local machine

We use windows command line to access the instance.

We need the keypair file that we created in last steps.

This is our example for this project:

```
C:\Users\username>ssh -i “location of keypair/key1.pem” ubuntu@ec2-52-87-214-199.compute-1.amazonaws.com
```

Then we can use following command to upload the whole RSMB directory to the instance terminal:

```
C:\Users\username>scp -r -i location of keypair/key1.pem location of folder ubuntu@ec2-52-87-214-199.compute-1.amazonaws.com:
```

Now we can start the broker in the terminal

```
./broker_mqtts RSMBconfig.conf
```

Start MQTT message subscriber client

The MQTT message subscriber saves incoming temperature from sensor node and shows the updates during the time that data is being saved.

We use paho mqtt python library which is an open-source set of client libraries and tools for implementing MQTT communication in various programming languages.

How to upload the python file to the terminal:

C:\Users\username>scp -i location of keypair/key1.pem location of file/filename ubuntu@ec2-52-87-214-199.compute-1.amazonaws.com:

Installing mySQL database

MySQL uses port 3306 by default. We opened this ports in security groups.

We use these codes:

sudo apt update

sudo apt install mysql-server

To start MySQL: *sudo systemctl start mysql*

The code to ensure the running of MySQL in every reboot: *sudo systemctl enable mysql*

Check the status of the MySQL service: *sudo systemctl status mysql*

Log into the MySQL from the terminal : *sudo mysql*

username: root

password: admin

mysql> mysql -u root -p

In order to create the tables for the database we use the following code:

```
CREATE DATABASE `TemperatureReadings`;

USE TemperatureReadings;

CREATE TABLE `Readings` (

 `datetime` datetime NOT NULL,

 `temperature` int DEFAULT NULL,

 PRIMARY KEY (`datetime`)

)
```

Install Phpmyadmin and Apache

Apache is one of the most popular open-source web servers globally. It's known for its robustness, security, and flexibility. Apache is used to serve web content and can handle various web technologies, such as PHP, Python, and more. It supports a wide range of modules and configurations, making it a versatile choice for web hosting and application deployment.

PhpMyAdmin is a web-based administration tool for managing MySQL and MariaDB databases. It provides a user-friendly interface for database management tasks such as creating, modifying, and deleting databases and tables.

1.We should install apache and PHP using the following codes:

```
sudo apt update
sudo apt install apache2 php libapache2-mod-php
```

2.We change the apache port to 8080.

First, we open the ports .conf file with a text editor:

```
sudo nano /etc/apache2/ports.conf
```

and change the line Listen 80 to Listen 8080.

We save and close the text editor using Ctrl+O,Enter, then Ctrl+X

3.Restart apache to see the effects:

```
sudo service apache2 restart
```

4.Install PHPMyadmin

```
sudo apt install phpmyadmin
```

If the terminal asked to choose a webserver, we select Apache2.

If it asks to use dbconfig-common to setup the database, we will choose yes.

5. We must configure the apache to recognize PHPMyadmin. We create a symbolic link for PHPMyadmin in the apache web directory:

```
sudo ln -s /usr/share/phpmyadmin /var/www/html/phpMyAdmin
```

How to access from webbrowser:

http://54.158.73.35:8080/phpmyadmin

Username and password are:

username: root

password: admin

Install Grafana with NGINX

Grafana is an open-source data visualization and monitoring platform that allows you to query, visualize, and understand your data through a variety of interactive and customizable dashboards. It is commonly used for monitoring and observability in the context of IT infrastructure, application performance, and IoT (Internet of Things) environments.

```
sudo apt-get update

sudo apt-get install -y apt-transport-https

sudo apt-get install -y software-properties-common wget

wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -

sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"

sudo apt-get update
sudo apt-get install -y Grafana
```

Start Grafana:

```
sudo systemctl start grafana-server

sudo systemctl enable grafana-server
```

Install NGINX

```
sudo apt-get install -y nginx
```

We should configure NGINX as a reverse Proxy for Grafana.

We create a new server block using text editor

```
sudo nano /etc/nginx/sites-available/grafana.conf
```

Then we add the following configuration to the file. The domain name or public ID can replace this example. Then save the file and exit the editor.

```
server {

  listen 80;

  ec2-52-87-214-199.compute-1.amazonaws.com;


  location / {

    proxy_pass http://localhost:3000;

    proxy_set_header Host $host;

    proxy_set_header X-Real-IP $remote_addr;

    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

    proxy_set_header X-Forwarded-Proto $scheme;

  }


}
```

Enabling the NGINX server block configuration

```
sudo ln -s /etc/nginx/sites-available/grafana.conf /etc/nginx/sites-enabled/
```

Test NGINX

```
sudo nginx -t
```

If there are no errors, restart NGINX with:

```
sudo systemctl restart nginx
```

Access to Grafana

We open a web browser page and navigate to http://your_public_ip of EC2 instance: 3000.

The Grafana login page will be visible.

1.We should enter the default username and password which are admin/admin.

2.Then we can change our password.

If everything is correctly working, you can login to through the page.

http://54.158.73.35:3000

Connect Grafana to the Database

We should add a datasource:

From configurations>Datasources>add Datasource

We will select MySql and edit the necessary details.

After Save and Test we are successfully connected.

Setup Grafana Dashboard

After logging in:

1.Click on + icon> New dashboard> add Visualization> MySql-TemperatureReadings as Datasource

2. We select dataset Temperature readings and Readings as table.

3.We select one of charts to visualize the data.

We can dedicate the first column to datetime and the second one to temperature.

After successfully setting the zoom data will be shown.

From the sensor node it submit 5 data points from the environment with 5 seconds delay.

References:

https://youtu.be/N4S6UjR6gUY?si=14vFWWUGAyWp5lYD

https://4sysops.com/archives/assign-an-ipv6-address-to-an-ec2-instance-dual-stack/

https://www.hackster.io/zakilive/iot-from-microcontroller-to-cloud-with-riot-os-459275