# LOMBA KOMPETENSI SISWA (LKS) SEKOLAH MENENGAH KEJURUAN TINGKAT PROVINSI JAWA BARAT TAHUN 2024

## NASKAH SOAL

## Bidang Lomba

## Cloud Computing

## Modul 2 - Using S3 to store data from IoT core

May 13, 2024


**PEMERINTAH DAERAH PROVINSI JAWA BARAT**

**DINAS PENDIDIKAN**

Jalan Dr. Radjiman No. 6 Telp. (022) 4264813 Fax. (022) 4264881
Website: disdik.jabarprov.go.id
Email: disdik@jabarprov.go.id/sekretariatdisdikjabar@gmail.com
BANDUNG - 40171

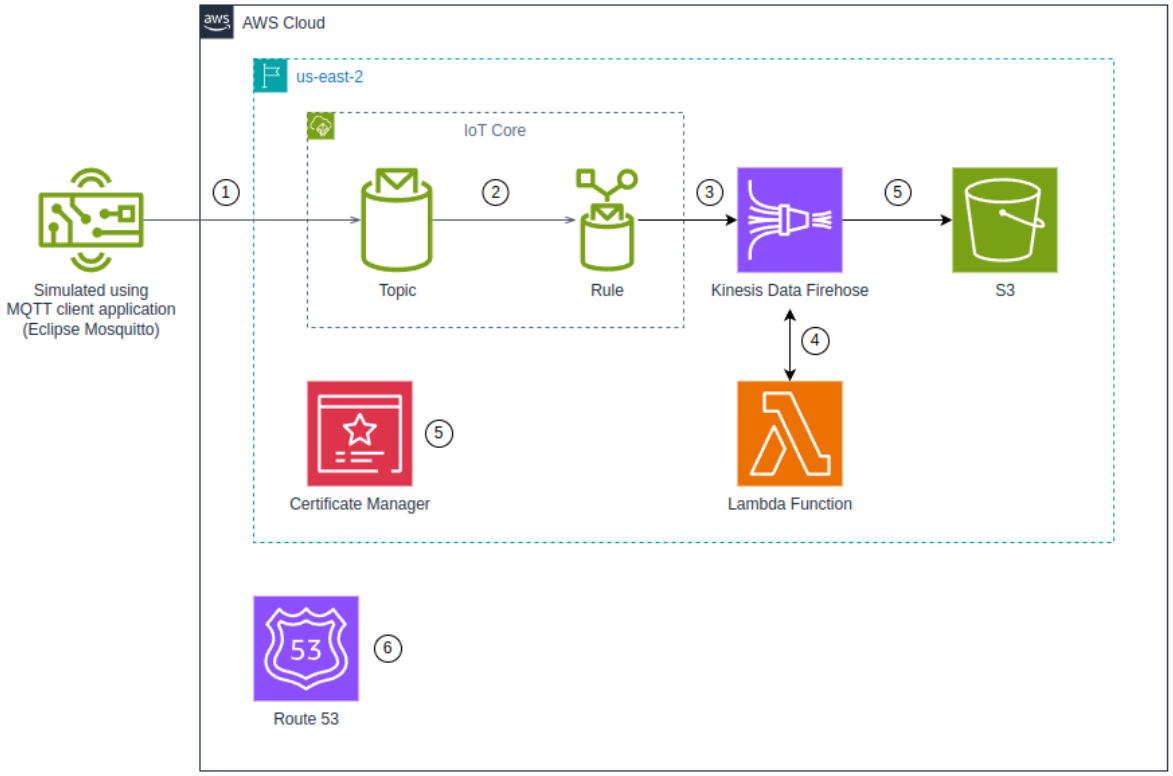


Figure 1: Architecture Diagram

# 1 Overview

Storing raw IoT data may incur significant expenses due to the sheer volume of the data involved. S3 offers a cost-effective solution for storing data, especially when dealing with large datasets. It follows a pay-as-you-go pricing model, meaning you only pay for the storage you use, without any upfront costs or long-term commitments. Your task is to build a solution to efficiently store raw data from IoT core to S3 using Kinesis Firehose. Additional preprocessing such as using Glue, Athena dan QuickSight is beyond the scope of this task.

# 2 General Rules

1. Failure to comply with the rules will result in immediate disqualification.
2. You have 3 hours to finish the tasks.
3. You may not open any website unless otherwise specified in section 7 and you may open the control panel of your domain provider to update the nameserver to Route 53.
4. Using the search feature from the AWS documentation website is allowed. However opening other AWS website such as AWS Solutions Library is NOT permitted. If you are unsure about the eligibility of the site you want to open, please ask.
5. You may use AWS Console and AWS CLI to deploy the solutions. You may not use SAM, CloudFormation, or CDK.
6. During the event, multiple login is not permitted.
7. If you have any question, do not hesitate to ask.

# 3 Architecture

Refer to Figure 1.

1. The MQTT software stream payloads to the AWS IoT Core message broker to a specific MQTT topic.
2. The AWS IoT rule activates upon detecting a payload within its designated topic. The rule is configured with an Amazon Kinesis Data Firehose action.
3. Amazon Kinesis Data Firehose buffers the incoming payloads, then it will deliver them to the data store once either size or time thresholds are met, whichever comes first.
4. Before delivering to the destination, Kinesis Data Firehose execute an AWS Lambda function to transform the payloads in batches. In this case, the lambda function will add and attributed named "processedAt" with current timestamp as the value and then return the modified payload to Kinesis Data Firehose.
5. The modified payloads are compressed and put into an Amazon S3 bucket.
6. Certificate Manager is used to provide certificate for IoT Core's custom domain.
7. Route 53 is used to point the custom domain to IoT Core's endpoint.

# 4 Information

1. The repository URL for the required source code to deploy this solution is: https://github.com/kensasongko/lksccjabar2024modul3_aplikasi
2. This solution must be deployed in **us-east-2 (Ohio)** region. Deploying in another region will result in a major point reduction.
3. An automated scoring system will check your AWS account to calculate the score of your task.
4. In order check your progress, the scoring system needs an AWS credential with administrator privilege. You will be asked to provide the AWS credential.
5. AWS tag is used by the scoring system to find your solution. Forgetting to add the required tag or adding the same tag value to multiple instance may result in an unexpected behavior and thus lower your score.
6. The system will calculate 90% of the final score. The remaining 10% will be evaluated by checking manually in front of the participants.
7. If multiple participants complete the assignment, their completion times will factor into the manual scoring process. For instance, the first participant to finish will receive a full score, while subsequent finishers will incur minor penalties.
8. Mosquitto MQTT client is used as an example to test the deployment.

# 5 Task

Your task is to create the solution from the section 3.

1. Create an S3 bucket with the following configuration:
   - Block all public access.
   - Bucket Versioning: Disable
   - Tag: Key=LKS-CC-2024, Value=LKS-IOT-S3

2. Deploy Lambda.
    - The source code of the lambda function can be found in section 4.
    - Create and deploy the lambda function with the following configuration:
        - Lambda runtime: python3.12
        - Timeout: 1 minutes
        - Memory size: 256 MB
        - Ephemeral storage: 512 MB
        - Architecture: arm64
        - Tag: Key=LKS-CC-2024, Value=LKS-IOT-LAMBDA
3. Create Kinesis Data Firehose with the following configuration:
    - Source: Direct PUT
    - Destination: S3
    - Enable transform source records with AWS Lambda:
        - Choose the Lambda function you have previously deployed.
        - Version or alias: $LATEST
        - Buffer size: 1 MB
        - Buffer interval: 60 seconds
    - New line delimiter: Enabled
    - S3 bucket prefix: data/
    - S3 bucket error output prefix: error/
    - S3 bucket and S3 error output prefix time zone: UTC
    - S3 buffer hints:
        - Buffer size: 1 MB
        - Buffer interval: 60 seconds
    - Compression for data records: GZIP
    - Tag: Key=LKS-CC-2024, Value=LKS-IOT-FIREHOSE
4. Create an IoT Rule.
    - Rule name: ToFirehose
    - SQL version: 2016-03-23
    - SQL statement: SELECT * FROM 'location/#'
    - Rule action: Amazon Data Firehose stream
    - Amazon Data Firehose stream: The stream you have previously created
    - Separator: \n
    - Batch mode: Enabled
    - IAM Role: Create new role named IoTRuleToFirehoseRole
    - Tag: Key=LKS-CC-2024, Value=LKS-IOT-RULE
5. Create IoT Policy with the following configuration:
    - Allow iot:Connect to *
    - Allow iot:Publish to *
    - Tag: Key=LKS-CC-2024, Value=LKS-IOT-POLICY
6. Create IoT certificate with the following configuration:
    - Auto-generate new certificate.

- Attach the IoT Policy you have created.

7. Create a CNAME record in Route 53 named lks-iot from your domain with your IoT Core device data endpoint as the value of the record.
8. Request a public certificate for *.[YOUR_DOMAIN] from AWS Certificate Manager and validate the certificate using DNS validation.
9. Create IoT domain configurations with the following configuration:
   - Name: lks-iot
   - Domain type: Customer managed domain
   - Domain name: lks-iot.[YOUR_DOMAIN]
   - Server certificate: Choose the certificate you have previously requested in ACM.

# 6 How to test your deployment

1. Using the IoT certificate you have downloaded, execute the following command on your terminal:

```
mosquitto_pub -h lks-iot.[YOUR_DOMAIN] -p 8883 \
  --key /path/to/xxx.private.pem.key \
  --cert /path/to/xxx.certificate.pem.crt \
  --cafile /path/to/AmazonRootCA1.pem \
  -t location/device1 -m '{"hello":"test"}' -i d1 -d
```

   The output should be similar to:

```
Client d1 sending CONNECT
Client d1 received CONNACK (0)
Client d1 sending PUBLISH (d0, q0, r0, m1, 'location/d1', .(9 bytes))
Client d1 sending DISCONNECT
```

2. Check your S3 bucket, ensure the bucket contains your payload along with processedAt attribute. For example:

```
{"hello": "test", "processedAt": "2024-05-05␣03:57:41"}
```

# 7 References

- Eclipse Mostquitto
- AWS IoT Core
- Amazon Data Firehose
- AWS Lambda
- Amazon S3
- Amazon Route 53
- AWS Certificate Manager

# Good luck!

$$cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \cdot \|\vec{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{1}$$

$$KernelKMeansDistance(K_{ii}, K) = K_{ii} - \frac{2 * \sum_{x_j \in \pi_c} \alpha_j K_{ij}}{\sum_{x_j \in \pi_c} \alpha_j} + \frac{\sum_{x_j, x_l \in \pi_c} \alpha_j \alpha_l K_{jl}}{(\sum_{xj \in \pi_c} \alpha_j)^2} \tag{2}$$