Lab 3: Introduction to Access Control with Amazon MSK

Lab clarity level - 5/10

Obj:

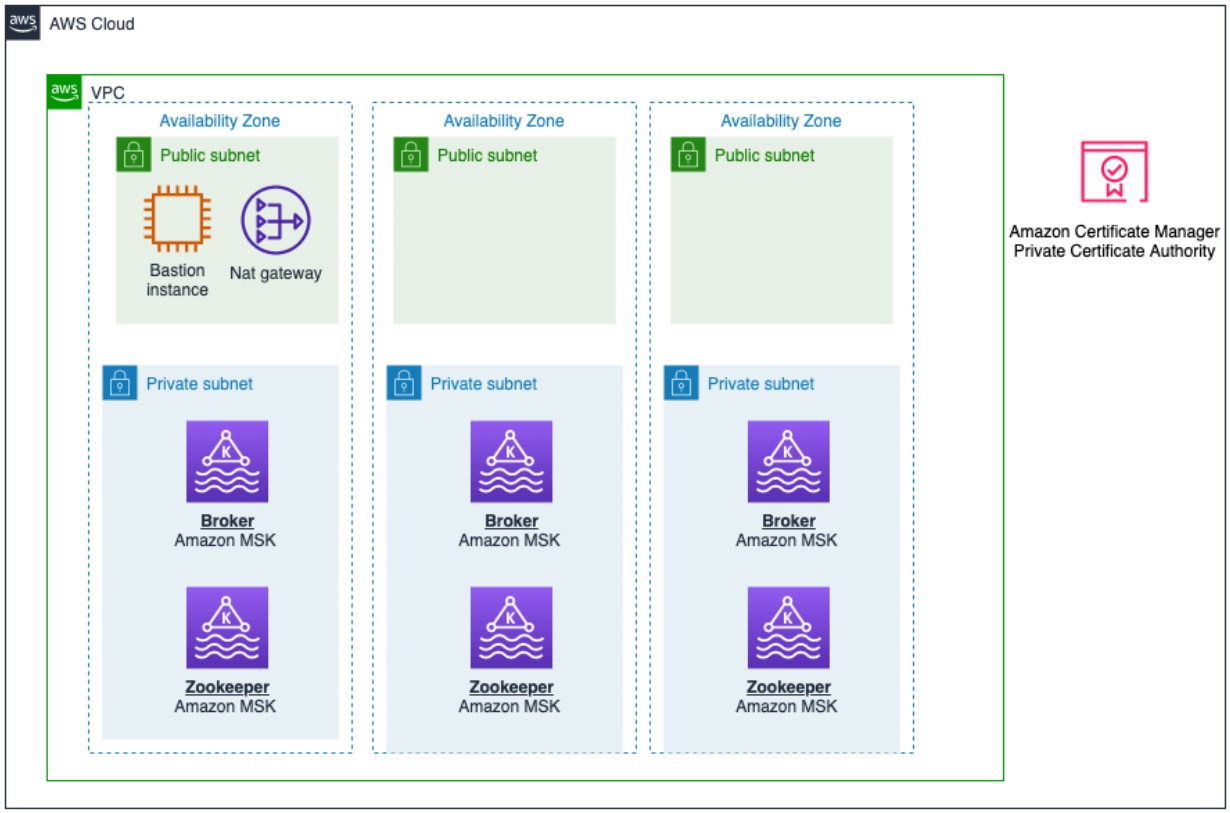
1. Publish to and consume from an MSK cluster using IAM authenticated broker URLs with a Java demo producer and Java demo consumer.

Simply:
Create an MSK cluster -> Use IAM for Authentication and Authorization -> Publish to Kafka using a Java Producer -> Consume from Kafka using a Java Consumer

What is Kafka:-

1. Kafka is a system that lets you send messages between different parts of an application.
    a. Note- we're not explicitly sending messages between two separate applications. Instead, we're interacting with Kafka through various components, such as producers and consumers
2. Think of it like a post office for data. You send messages (like letters) to Kafka, and Kafka makes sure they get to the right place.

- Amazon Managed Streaming for Apache Kafka (Amazon MSK)-
    - Amazon MSK is a service provided by AWS that helps you run Apache Kafka without worrying about the underlying infrastructure.
- Using IAM with Kafka means you can control who can send messages to (publish) and read messages from (consume) your Kafka system.



In each public subnet, we include bastion host to access it from outside
In private (not connected to internet, to ensure security):

1. Broker (Amazon MSK): These are the Kafka servers that handle sending and receiving messages.
2. Zookeeper (Amazon MSK): These are servers that help manage the Kafka brokers.

Certificate mgr ensures secure communication between the different components.

DID NOT FIND - KAFKA ARCH.

Task 1: Inspecting the MSK cluster

1.1 Oen EC2, and bastion host (connect).
Use the in-lab URL for this

1.2 Run the command to display the output of a shell script that has been created to populate your environment with the various network addresses needed to access the MSK cluster.

1.3 To output the various network addresses needed to access the MSK cluster

```
sh-4.2$ cat /opt/kafka_2.12-2.2.1/msk.env
export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/opt/kafka_2.12-2.2.1/bin:/opt/apache-maven-
3.8.8/bin:/usr/local/bin/
export MSK_ARN=arn:aws:kafka:us-west-2:854514635624:cluster/MSK-Demo/71fb316b-661b-4830-a002-2f18d52171c5-
10
export MSK_BOOTSTRAP="b-2.mskdemo.x8dqeo.c10.kafka.us-west-2.amazonaws.com:9094,b-3.mskdemo.x8dqeo.c10.kaf
ka.us-west-2.amazonaws.com:9094,b-1.mskdemo.x8dqeo.c10.kafka.us-west-2.amazonaws.com:9094"
export MSK_ZOOKEEPER="z-1.mskdemo.x8dqeo.c10.kafka.us-west-2.amazonaws.com:2181,z-2.mskdemo.x8dqeo.c10.kaf
ka.us-west-2.amazonaws.com:2181,z-3.mskdemo.x8dqeo.c10.kafka.us-west-2.amazonaws.com:2181"
export MSKIAM_BOOTSTRAP="b-1.mskdemo.x8dqeo.c10.kafka.us-west-2.amazonaws.com:9098,b-2.mskdemo.x8dqeo.c10.
kafka.us-west-2.amazonaws.com:9098,b-3.mskdemo.x8dqeo.c10.kafka.us-west-2.amazonaws.com:9098"
sh-4.2$
```

o/p has bootstrap brokers, and the Apache Zookeeper address for the cluster.

1.4 Inspection means viewing these things:

MSK_ARN: The Amazon Resource Name for the MSK cluster.

MSK_BOOTSTRAP: The bootstrap servers' addresses for TLS-authenticated communication (port 9094).

MSK_ZOOKEEPER: The addresses of the Zookeeper nodes used for managing the Kafka cluster.

MSKIAM_BOOTSTRAP: The bootstrap servers' addresses for IAM-authenticated communication (port 9098).

GPT - The bootstrap servers (under MSK_BOOTSTRAP (for secure communication) and MSKIAM_BOOTSTRAP (for authentication)) are

the endpoints you will use to connect to the Kafka brokers within your MSK cluster.

Task 2: Publish to and consume from an IAM authenticated MSK cluster

The goal of Task 2 is to publish a simple message to your Amazon MSK cluster. This involves using a producer to send a message to a Kafka topic within the cluster.

K. Topic example:
Imagine a system that tracks user interactions on a website. Each interaction (e.g., click, page view, purchase) is sent to a Kafka topic called user-interactions. Multiple producers (e.g., different web servers) send interaction events to this topic. Downstream, multiple consumers (e.g., analytics services, recommendation engines) read from the user-interactions topic to process and analyze the data.

2.1 Create and list a topic.
2.2 To load the environment variable that references the MSK cluster into your shell environment, run the following command:

source /opt/kafka_2.12-2.2.1/msk.env

2.3 Run this to create a topic:
kafka-topics.sh --create --topic ExampleTopic --partitions 5 --replication-factor 3 --bootstrap-server $MSK_BOOTSTRAP --command-config /opt/client.properties

2.4 Verify creation

```
sh-4.2$ kafka-topics.sh --list --bootstrap-server $MSK_BOOTSTRAP --command-config /opt/client.properties
ExampleTopic
__amazon_msk_canary
__consumer_offsets
sh-4.2$
```

2.5 Compile JAVA producer and consumer
Code for Prod and cons already done by the lab, and is stored in “maven” package.

We compile that package using:

mvn package

```
[INFO] Dependency-reduced POM written at: /opt/msk-java/dependency-reduc
[INFO] ------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------
[INFO] Total time:  21.086 s
```

2.6 Publish to Example Topic using JAVA Producer

java -cp target/msk-auth-demo-1.0-SNAPSHOT.jar
com.amazonaws.examples.DemoProducer $MSKIAM_BOOTSTRAP
ExampleTopic

```
Hello World!
sent message to topic:ExampleTopic partition:1  offset:0
>
```

Sent hello world to topic (typed hello world in CLI)

2.7 Consume from AN IAM AUTHENTICATED MSK CLUSTER

We consume message from the example topic

2.8 Open a new bastion host connection using url given
And
Load the env variables that reference to the MSK cluster using

source /opt/kafka_2.12-2.2.1/msk.env

2.9 Change to the directory where prod and cons code is compiled using

cd /opt/msk-java

2.10 Run this command to consume from the topic

java -cp target/msk-auth-demo-1.0-SNAPSHOT.jar com.amazonaws.examples.DemoConsumer $MSKIAM_BOOTSTRAP ExampleTopic

```
tId=consumer-1, groupId=foobar] Resetting offset for partition ExampleTopic-2 to offset 0.
Hello World!
```

We see the message that we input using producer

2.11 more messages

```
>It is a great day.
sent message to topic:ExampleTopic partition:4  offset:0
>I got this.
sent message to topic:ExampleTopic partition:4  offset:1
>
```

```
Hello World!
It is a great day.
I got this.
```

You can see the new messages consumed in the *Java consumer session window*.