National College of Ireland

Configuration Manual

MSc Research Project
MSc in Cloud Computing

Shweta Das
Student ID: x20138547

School of Computing
National College of Ireland

Supervisor: Sean Heeney

Configuration Manual

Shweta Das
x20138547

1 Introduction

1.1 Purpose of the document

This Configuration Manual follows the specifications of the NCI Research project.In this document, we will discuss the software tools and settings that are required to provide efficient task scheduling in serverless platform using Ant Colony Optimization(ACO) Algorithm.

1.2 Document structure

Section	**Purpose**
General Information	This module describes how the serverless platform is set up and the project requirements.
Setup prerequisites	This module describes how to set up the development environment for the development and update of the solution.
Deployment procedure	A proposed model deployment procedure is described in this module
Validations	The purpose of this module is to provide an overview of the validation requirements for the deployment of the solution

2 General Information

2.1 Objective

The objective of this research work is to schedule a task efficiently using Ant Colony Optimization(ACO) algorithm. The ACO algorithm was written in oython 3.8 and utilizes boto3, redis and json library. The experimental simulation was conducted to check the memory and cost utilization by MapReduce application on serverless platform.

2.2 Architecture requirement

Here is a description of the AWS services required to build a Composite model.

2.2.1 AWS Virtual Private Cloud(VPC)

To access Lambda and Elastic Cache, the AWS VPC (Virtual Private Network) is created. The VPC contains subnets, security groups, route tables, and NAT gateways. [1].

2.2.2 AWS Simple Storage Service(S3)

AWS S3 is used for storing input data for MapReduce application and storing output data from MapReduce application [2].

2.2.3 AWS CloudWatch

AWS CloudWatch is used for monitoring MapReduce performance such as CPU utilization, execution time and many more with and without ACO algorithm. [3].

2.3 AWS Lambda

Creation of coordinator, mapper, and reducer functions with Python 3.8 is accomplished by using the Lambda compute service provided by Amazon [4].

2.4 AWS Elastic Cache

Using AWS Elastic Cache service, intermediate files are produced by map functions. This is used to additional power to our application[5].

2.5 Required Skill

You will need to be familiar with Amazon Web Services before you start reading this guide. You will also need to be familiar with the Python language so that you can understand the code.

[1] `https://aws.amazon.com/vpc/`
[2] `https://aws.amazon.com/s3/`
[3] `https://aws.amazon.com/cloudwatch/`
[4] `https://aws.amazon.com/lambda/`
[5] `https://aws.amazon.com/elasticache/`

3 Development Environment Requirement

3.1 Code Repository

Please refer to the zip file I have submitted in the ICT solution.

3.2 Programming language required

- Python Version 3.8
- Boto3
- Shell scripting

3.3 Configuring IAM role

It is necessary, before creating a Lambda function, to create an IAM role which has policy attached to it as shown in the figure 1. This facilitates the execution environment creation for the lambda function.

- AmazonLambdaRole
- AmazonS3FullAccess
- AmazonVPCFullAccess
- Amazon ElasticCacheFullAccess
- AmazonCloudWatchFullAccess

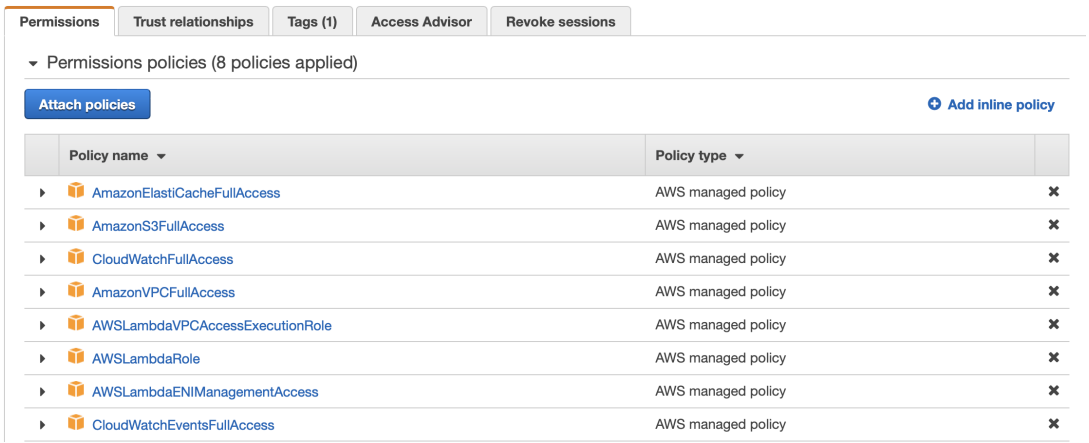

Figure 1: Required IAM policies

3.4 Configuring a VPC

The VPC (server virtual private network) is configured with the 192.168.0.0/16 CIDR block in figure 2

- Use 192.168.10.0/24 v4 CIDR blocks for the public subnet as shown in figure4.
- A public subnet is created by an internet gateway that is assigned to the subnet shown in figure 3
- In figure 5, you can see how to create a private subnet with 192.168.20.0/24 CIDR addresses
- Create a NAT gateway for assigning a private subnet as depicted in the figure6.
- Add public subnet and internet gateway to public route table as shown in figure 7
- Set up private route tables and assign private subnets to them along with their respective NAT gateways and virtual private connections, as shown in figure8.
- Create the VPC endpoint as depicted in figure **??**
- As a virtual firewall, create the security group that allows both inbound and outbound traffic. This security group uses HTTP, HTTPS, SSH, and Custom TCP to allow any traffic from anywhere to Redis and other services as shown in figure 10.

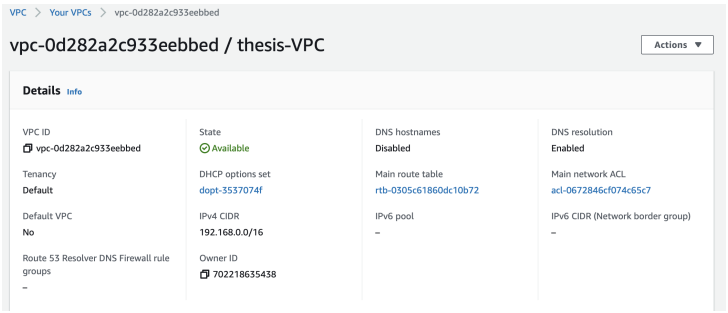

Figure 2: VPC creation

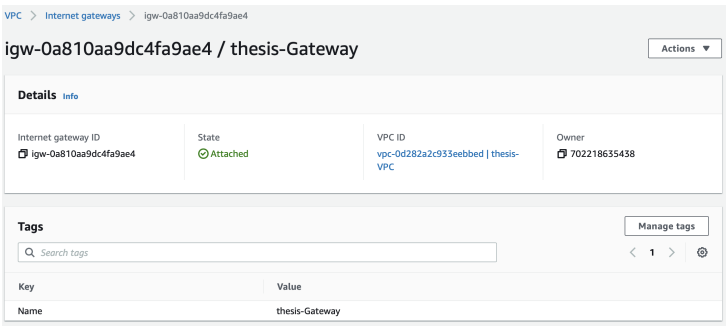

Figure 3: Internet Gateway creation

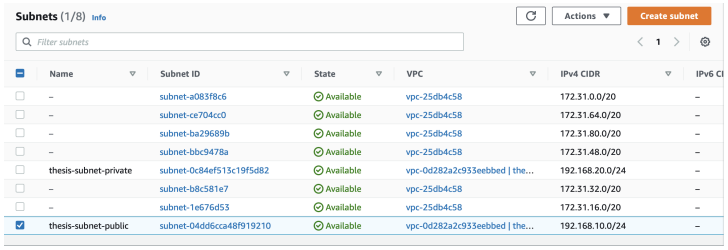

Figure 4: Public subnet creation

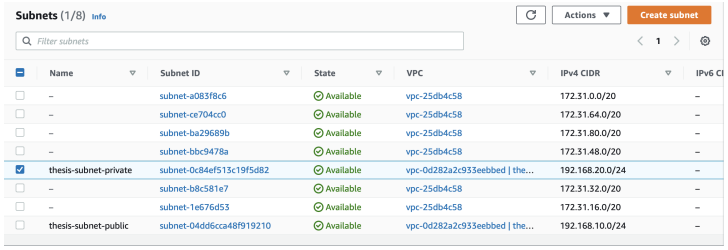

Figure 5: Private subnet creation

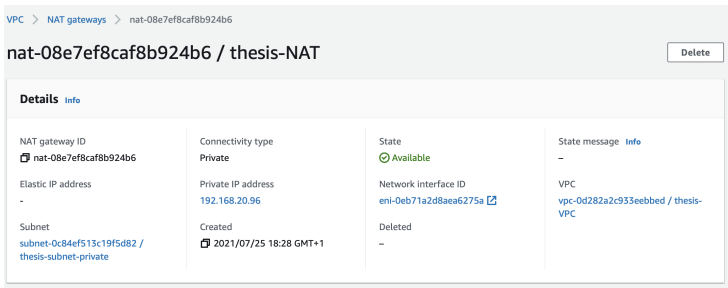

Figure 6: Nat Gateway creation

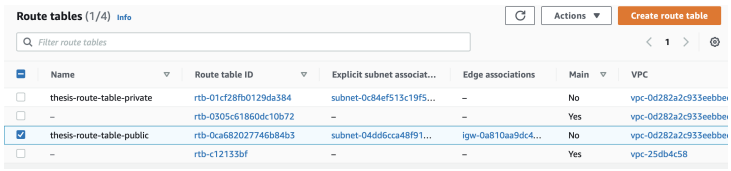

Figure 7: Public routing table creation

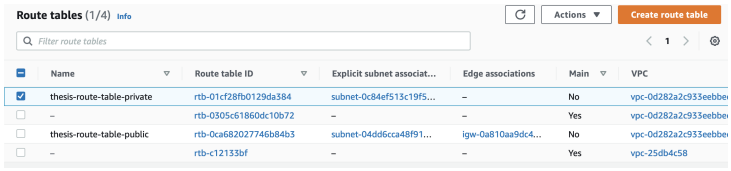

Figure 8: Private route table creation

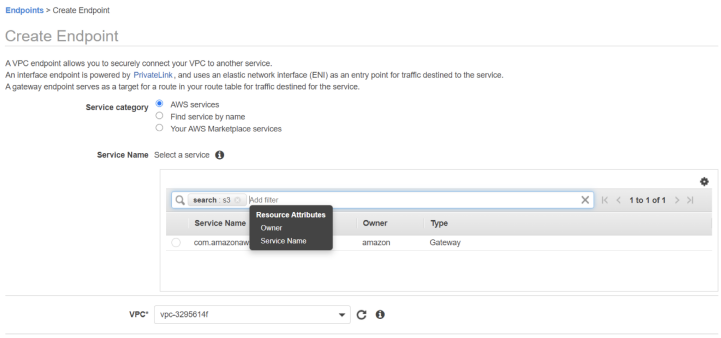

Figure 9: Endpoint creation

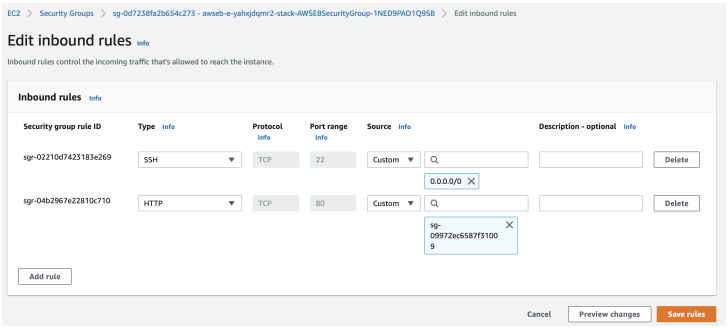

Figure 10: Security group creation

3.5 AWS S3 creation

To store incoming data, create the S3 bucket as shown in figure **??**

3.6 Creation of AWS Lambda function

Once you create the IAM role, create the lambdas for mapper, coordinator, and reducer.

- The AWS lambda function can be created by going to Lambda and clicking Create a Function as shown in figure 12.
- Click the blueprint button once you've selected the create function as shown in figure 13
- As shown in figure 14, select the basic information section with the function name, run-time language, and permissions – a permissions role with everything required.
- You need to edit the vpc section after creating the lambda function in the project. As illustrated in figure 16, add the network configuration
- Whenever input data is entered into the bucket, it should be notified to the lambda function. To do this, we must set a trigger event on the bucket in the lambda function **??**

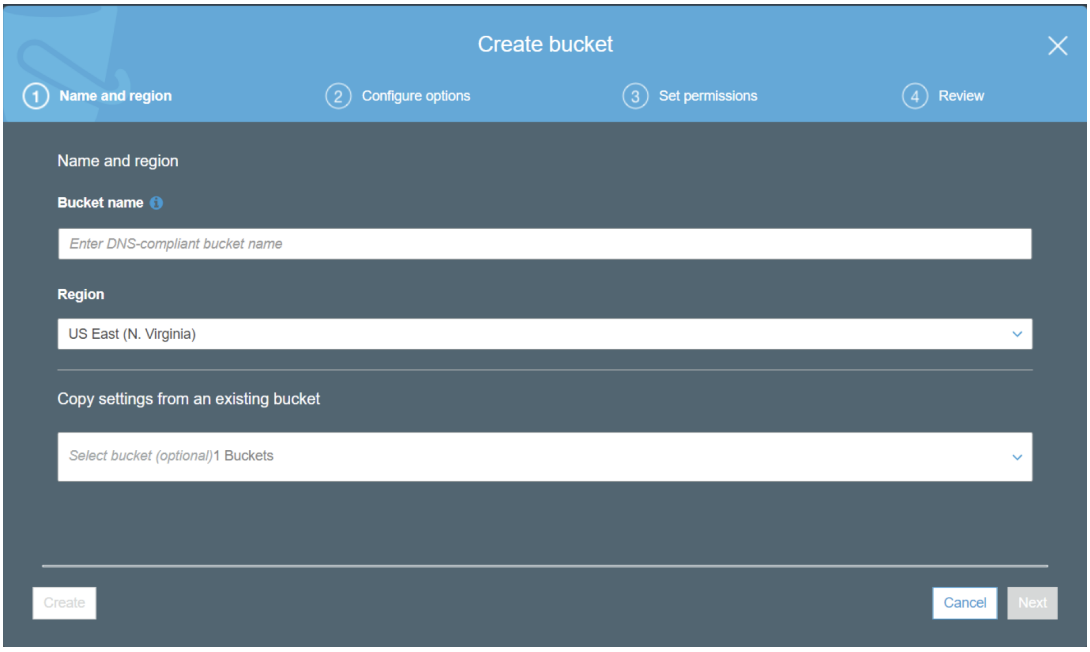

Figure 11: AWS S3 bucket interface

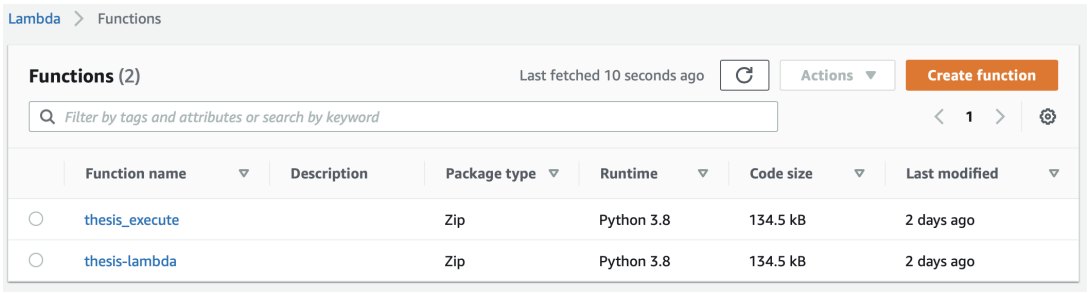

Figure 12: Function Creation

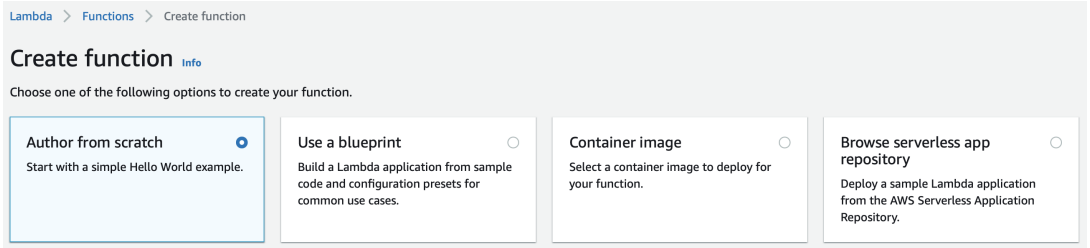

Figure 13: Blueprint Selection

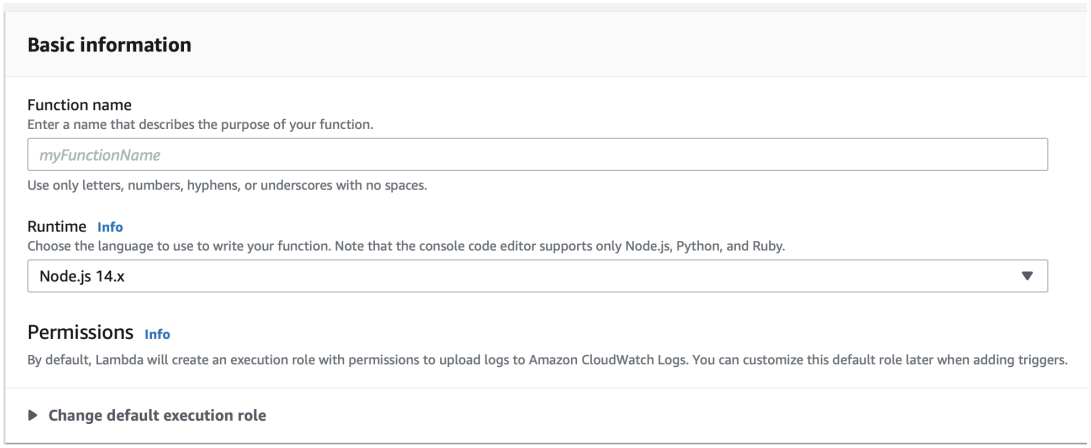

Figure 14: Configuration details

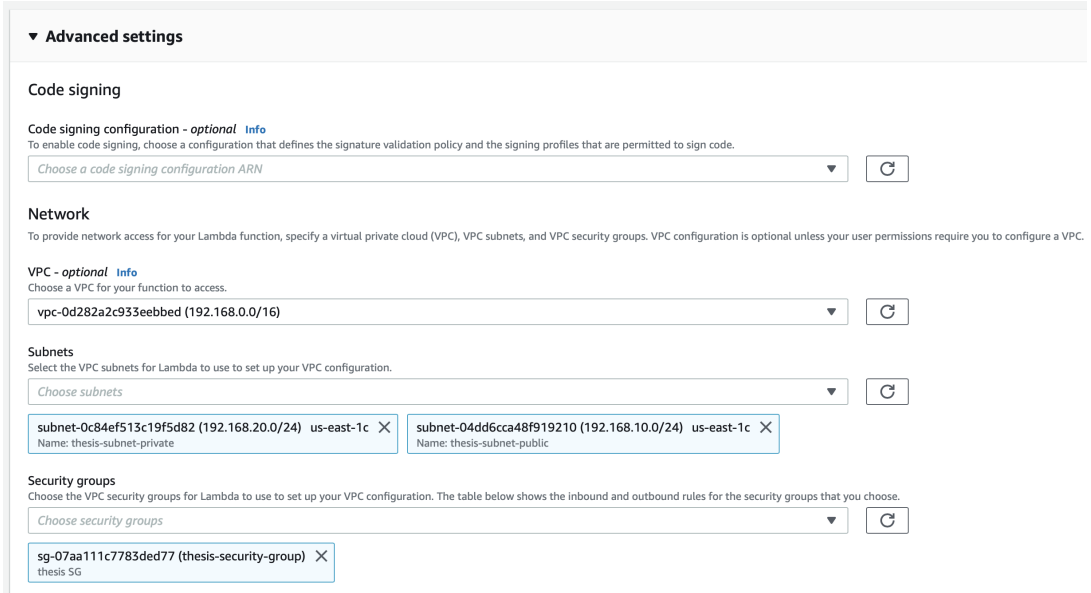

Figure 15: Configuring network

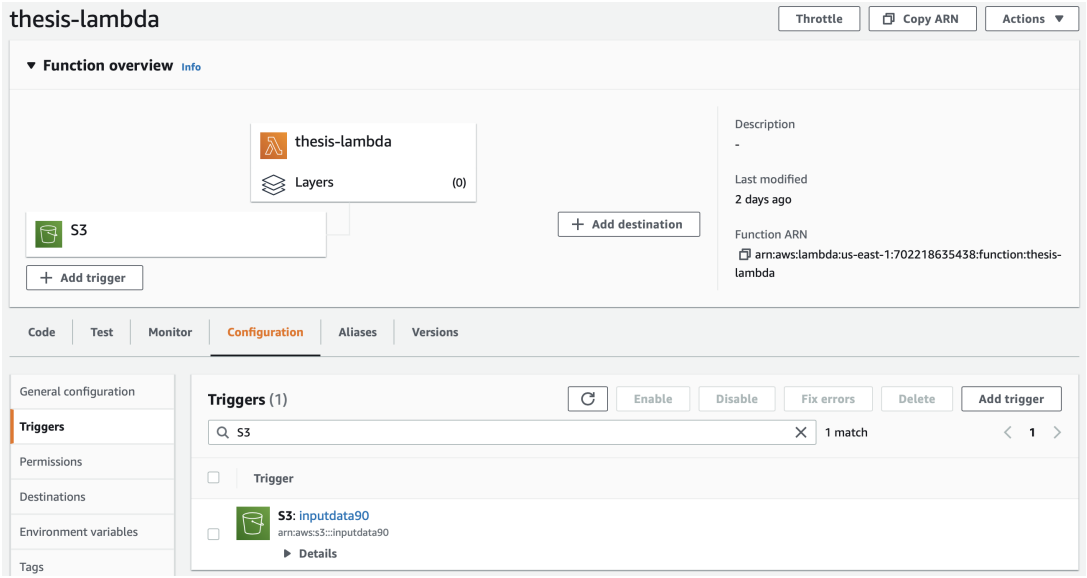

Figure 16: Configuring network

3.7 AWS Elastic Cache Creation

To store intermediate data generated from mapper functions, create an elastic cache memory for redis.

- The cluster engine should be selected as redis and we will configure the redis setting in which we will select engine version, port, node type, and the number of replicas as shown in figure 17
- There are a number of advance settings that need to be selected, including subnet group, security group, backup and maintenance windows as shown in figure 18.

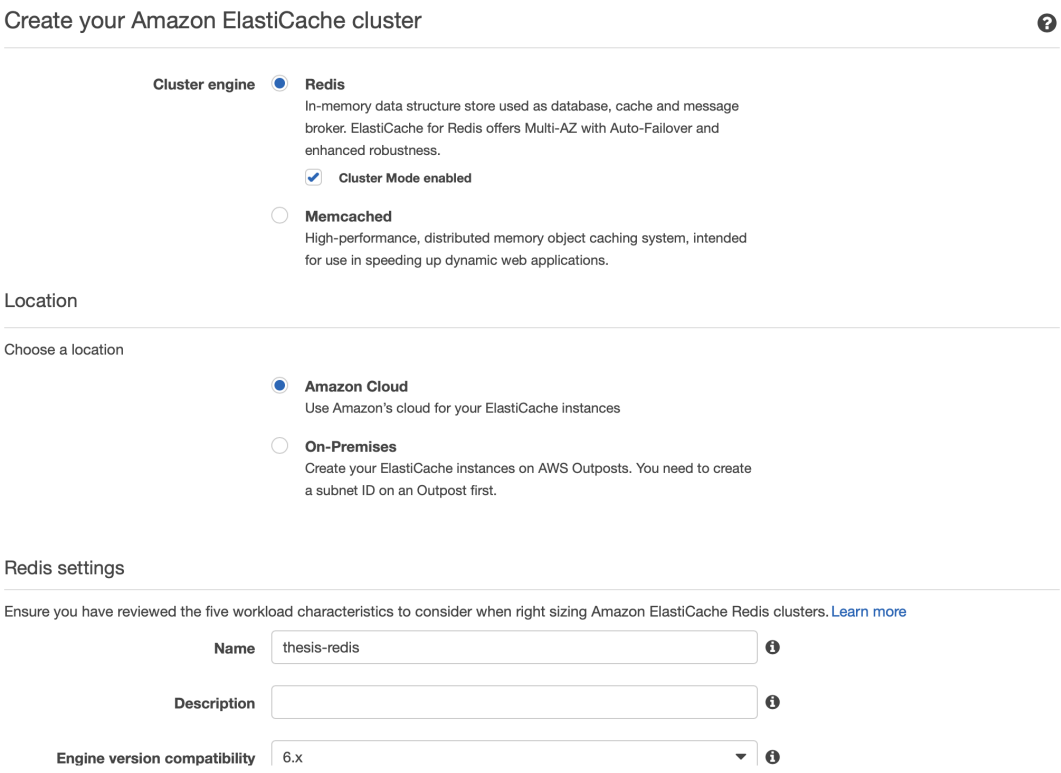

Figure 17: Redis configuration

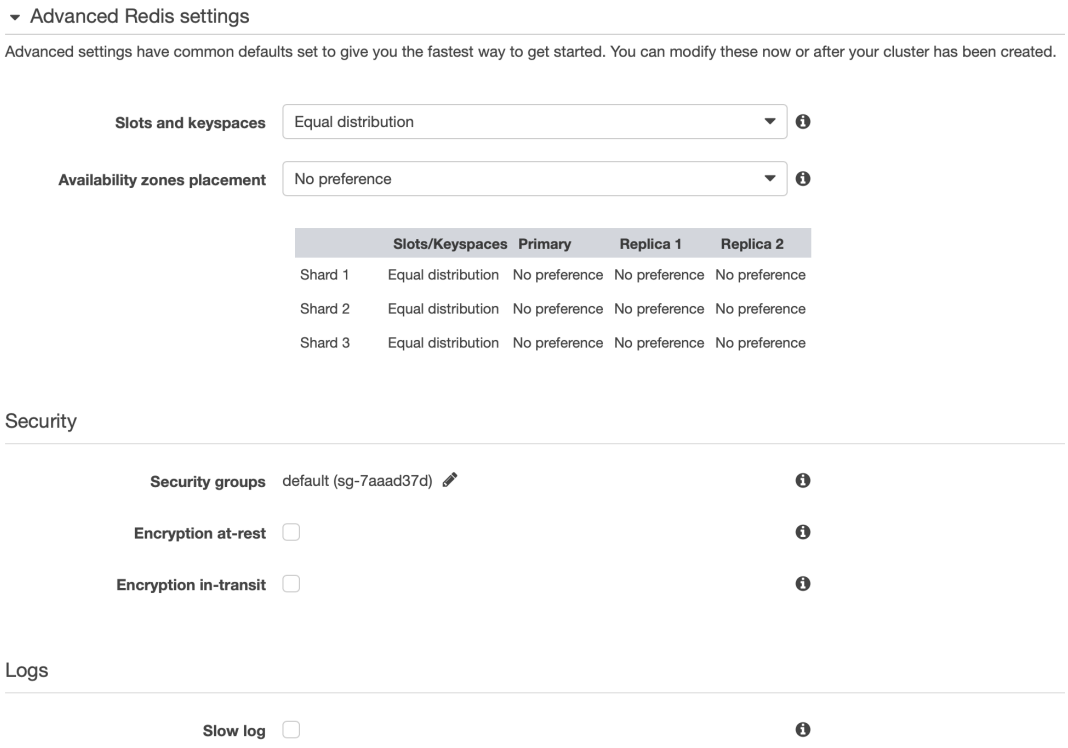

Figure 18: Additional Redis settings

4 Validation

Prepare the code and upload the files using the upload zip file option. now create a lambda function and inside that function, set up a mapper, a coordinator, and a reducer. After creating all the services in the VPC section, you should now configure the network. The configuration is tested in two ways as shown below:

- An alternative way is to create tests within functions, as shown in the figure 19. Run the tests, and then view the results displayed in the figure 20. When the configuration is incorrect, an error will be thrown.
- As shown in figure 21, the second way involves adding input to the S3 bucket so that whenever there is input at the bucket, a S3 event trigger is triggered. Figure 22 shows the output in the cloud watch service when there is a correct connection between the trigger event in S3 and lambda function.

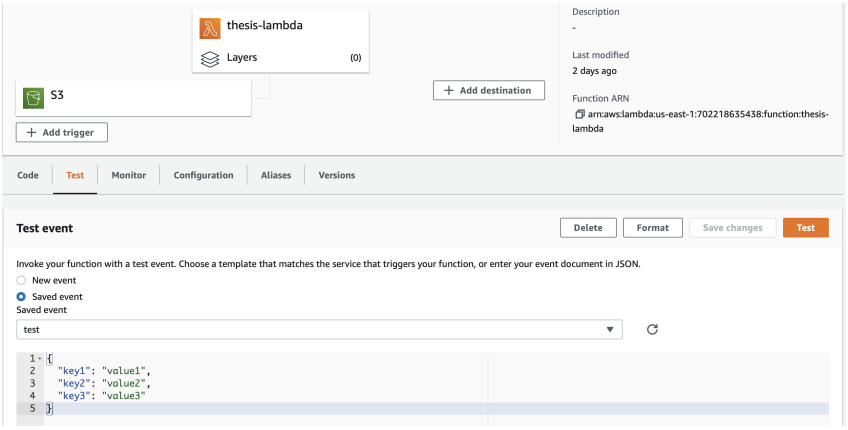

Figure 19: MapReduce application execution test case creation

Figure 20: Execution result of MapReduce application

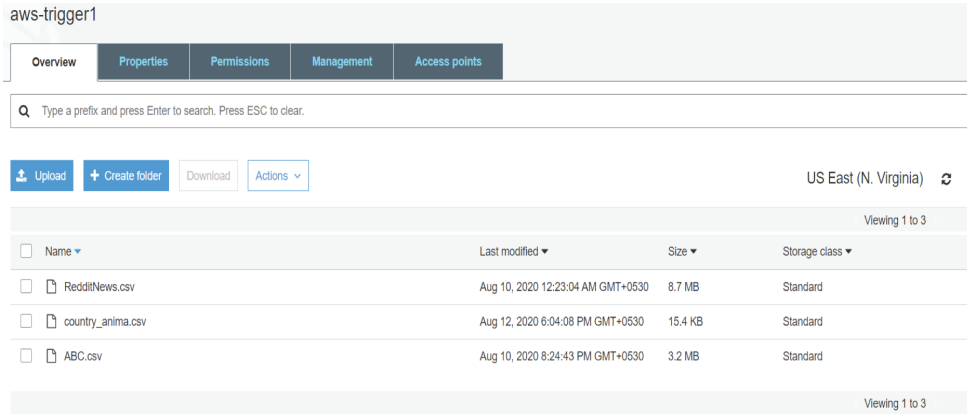

Figure 21: Event trigger for AWS S3

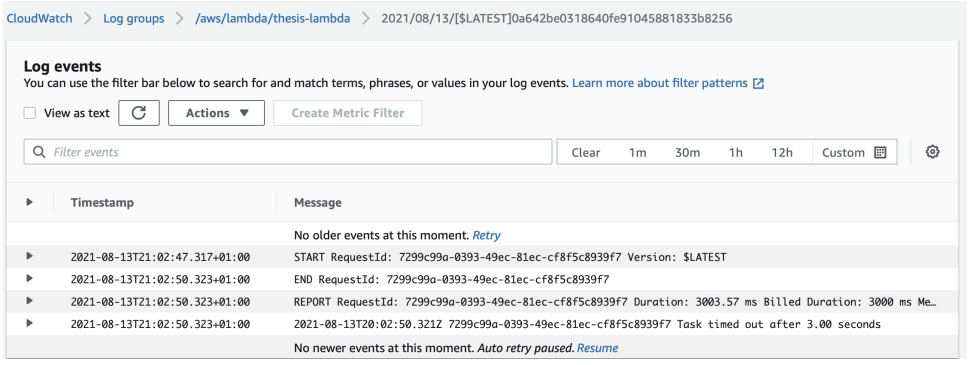

Figure 22: Execution logs provided by AWS Cloud Watch