Lomba Kompetensi Siswa
Sekolah Menengah Kejuruan
Tingkat Kabupaten Tasikmalaya
Tahun 2023

Modul 3 - Highly Scalable and Fault Tolerant Architecture

Bidang Lomba Cloud Computing

1 Overview

You have been given a task to deploy a web application on AWS. The application is built using nodejs and it will be hosted on EC2. The architecture must be designed to highly scalable and fault tolerant. You will deploy the application to an Auto Scaling Group in a multi-AZ environment, EFS is used to store uploaded images, ElastiCache for Redis is used to store login session information and Aurora Serverless is used to store data.

2 General Rules

1. Failure to comply with the rules will result in immediate disqualification.
2. You have 6 hours to finish the tasks.
3. This is an open book test.
4. You may use AWS Console and AWS CLI to deploy the solutions. You may not use CloudFormation or CDK.
5. Between and after the event, you may not access your account. Any activity on AWS during this period is not allowed.
6. During the event, multiple login is not permitted.
7. If you have any question, do not hesitate to ask.

3 External Resources

1. Located for the project in folder “lks-jabar-2022-modul3-master”
2. The queries used to create required tables for the project is located in file “create-table.sql”

4 Architecture

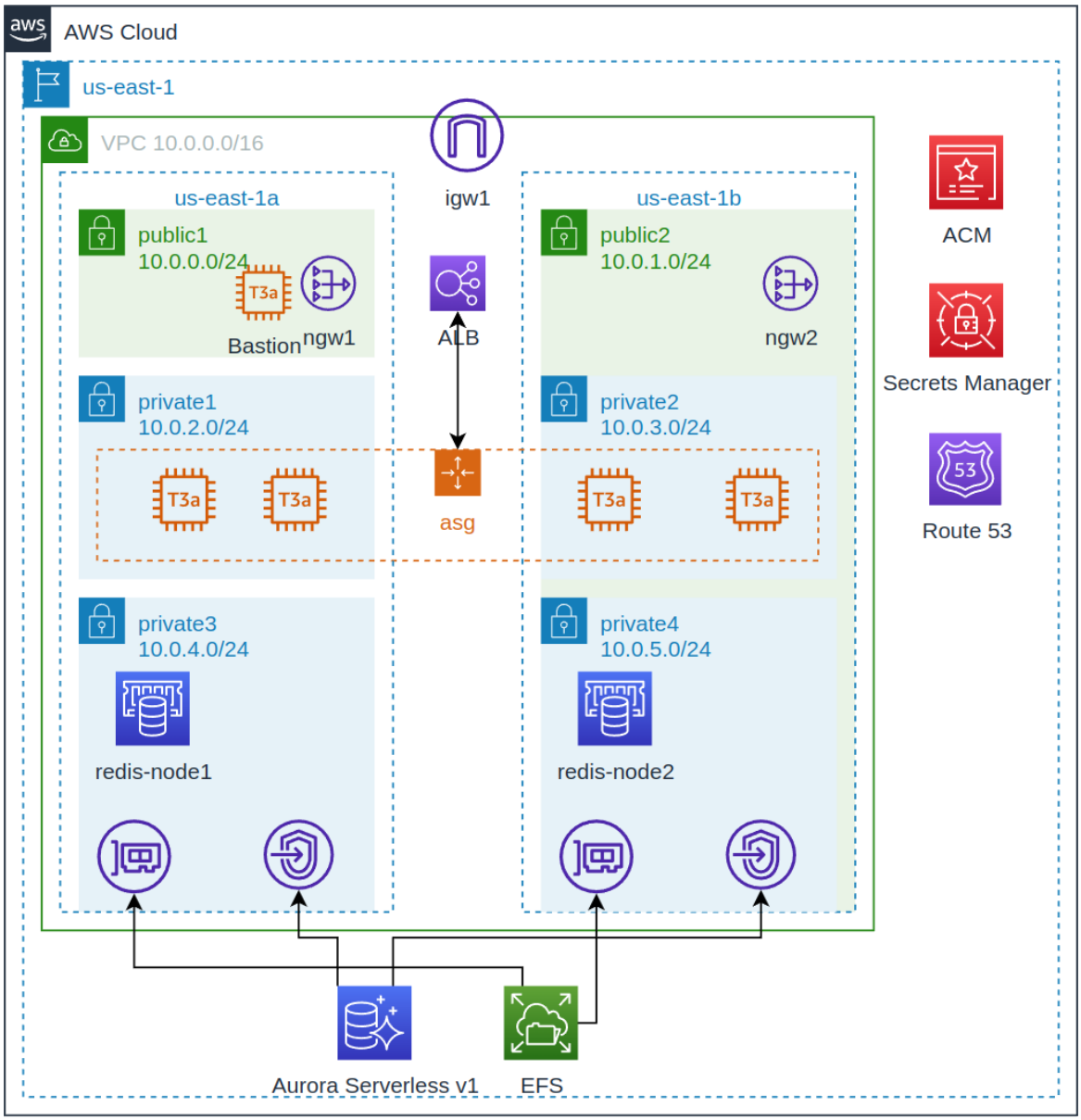

Figure 1: Architecture Diagram

5 Task

1. Create VPC with the following specifications:

   IPv4 CIDRs: 10.0.0.0/16

   Number of NAT Gateways: 2

   (a) Name: ngw1, Subnet: public1

   (b) Name: ngw2, Subnet: public2

   Subnets:

   (a) – Subnet Name: public1

- IPv4 CIDR block: 10.0.0.0/24
- 0.0.0.0/0 is routed to: Internet Gateway

(b)
- Subnet Name: public2
- IPv4 CIDR block: 10.0.1.0/24
- 0.0.0.0/0 is routed to: Internet Gateway

(c)
- Subnet Name: private1
- IPv4 CIDR block: 10.0.2.0/24
- 0.0.0.0/0 is routed to: NAT Gateway (ngw1)

(d)
- Subnet Name: private2
- IPv4 CIDR block: 10.0.3.0/24
- 0.0.0.0/0 is routed to: NAT Gateway (ngw2)

(e)
- Subnet Name: private3
- IPv4 CIDR block: 10.0.4.0/24
- No additional route.

(f)
- Subnet Name: private4
- IPv4 CIDR block: 10.0.5.0/24
- No additional route.

2. Create an EC2 key pair.

   Name: lksjabar2022modul3
   Key pair type: RSA
   Private key file format: .pem

3. Create a bastion host (EC2 instance) to access resources in private subnet remotely.

   Name: Bastion
   Subnet: public1
   Instance Type: t3a.micro
   Public IP Address: Enabled
   Security Group Rule: Allow SSH traffic from anywhere
   Key pair: lksjabar2022modul3

4. Create an RDS database with the following specifications:

   Aurora Serverless with PostgreSQL compatibility (version 10.18)
   DB instance class: **Serverless v1**
   Minimum ACU: 2
   Maximum ACU: 4
   Network Subnet: private3 and private4
   Enable scale the capacity to 0 ACUs when cluster is idle (00:05:00).
   Web Service Data API: Enabled
   Note: Check section 3 to find create table queries.

5. Create a database credential on Secrets Manager and attach it to the database

6. Create ElastiCache with the following specifications:

Redis Cluster

Enable cluster mode

Enable multi availability zone

Instance Type: t4g.micro

Port: 6379

Number of shards: 1

Number of replicas: 2

Network Subnet: private3 and private4

7. Create an EFS file system with the following specifications:

   Storage class: Standard

   Transition into IA: 7 days since last access

   Transition out of IA: On first access

   Performance mode: General Purpose

   Throughput mode: Bursting

   Encryption: Enable encryption of data at rest

   Mount targets:

   (a) AZ: us-east-1a, Subnet: private3
   (b) AZ: us-east-1b, Subnet: private4

8. Create an EC2 instance

   Name: InstanceTemplate-modul3

   Key pair: lksjabar2022modul3

   Instance Type: t3a.micro

   Storage: 8 GIB of GP2

   Connect to the instance and finish the following instructions inside the instance:

   - Clone the application repository from section 3. Follow the instruction in README.md to install the application.
   - Mount the EFS file system to APP_ROOT/public/images/ where APP_ROOT is the base directory of the application. Make sure the EFS file system is mounted on startup.
   - Restart the VM, make sure the application starts on startup.

9. Create an EC2 Launch Template from InstanceTemplate-modul3 with the following specifications:

   Launch template name: ASG-template-modul3

   Key pair: lksjabar2022modul3

   Instance Type: t3a.micro

10. Create Auto Scaling Group (ASG) with the following specifications:

    Network Subnet: private1 and private2

    Load balancing: Attach to a new load balancer

    - Load balancer name: ALB-modul3
    - Load balancer type: Application Load Balancer
    - Load balancer scheme: Internet-facing

    Minimum Capacity: 2

    Desired: 2

Max: 6

Scaling policies: Target tracking scaling policy

- Metric type: Average CPU utilization
- Target value: 70

11. Configure ALB-modul3 to redirect HTTP request to HTTPS and forward HTTPS request to the ASG's target group.
12. Create a certificate in ACM

    Domain Name: modul3.[YOUR_DOMAIN]

    Validation Method: DNS validation

13. Add custom domain to the Application Load Balancer
14. Open http://modul3.[YOUR DOMAIN] on your browser to check if HTTP request is redirected to HTTPS and make sure the web works correctly.

6 References

Amazon Aurora documentation

Application Load Balancer documentation

Certificate Manager documentation

EFS documentation

EC2 documentation

EC2 Auto Scaling documentation

ElastiCache for Redis documentation

Route 53 documentation

AWS Secrets Manager documentation

VPC documentation

npm documentation

yarn documentation

PM2 documentation

Good luck!