AWS Deployment Guide

Amit Christian

2024-08-14

# AWS Deployment Guide

This comprehensive guide will walk you through the process of deploying a basic web application on AWS. The deployment will utilize a custom VPC, EC2 instances (including a Bastion host), RDS (PostgreSQL Aurora), an Application Load Balancer (ALB), and an Auto Scaling Group (ASG). Each step includes detailed explanations, form fill-ups, and image captions where evidence is required.

## 1. VPC Configuration

### Objective

Create a custom VPC with both public and private subnets, distributed across two Availability Zones (AZs), and configure necessary routing and gateways.

### Steps

1. **Create a Custom VPC**:
   * Navigate to the VPC Dashboard in the AWS Management Console.
   * Click on “Create VPC”.
   * **Form Details**:
     + VPC Name: christian-vpc
     + IPv4 CIDR block: 10.0.0.0/16
     + Tenancy: default
   * Click “Create VPC”.
2. **Create Subnets**:
   * Go to the Subnets section under your VPC.
   * Click “Create Subnet”.
   * **Form Details**:
     + **Public Subnet 1**:
       - Name: christian-public-subnet-1
       - Availability Zone: us-east-1a
       - IPv4 CIDR block: 10.0.1.0/24
     + **Public Subnet 2**:
       - Name: christian-public-subnet-2
       - Availability Zone: us-east-1b
       - IPv4 CIDR block: 10.0.2.0/24
     + **Private Subnet 1**:
       - Name: christian-private-subnet-1
       - Availability Zone: us-east-1a
       - IPv4 CIDR block: 10.0.3.0/24
     + **Private Subnet 2**:
       - Name: christian-private-subnet-2
       - Availability Zone: us-east-1b
       - IPv4 CIDR block: 10.0.4.0/24
   * Create each subnet accordingly.
3. **Create an Internet Gateway (IGW)**:
   * In the VPC Dashboard, go to the Internet Gateways section.
   * Click “Create Internet Gateway”.
   * **Form Details**:
     + Name: christian-igw
   * Attach it to christian-vpc.
4. **Deploy a NAT Gateway**:
   * Go to the NAT Gateways section in the VPC Dashboard.
   * Click “Create NAT Gateway”.
   * **Form Details**:
     + Subnet: christian-public-subnet-1
     + Allocate a new Elastic IP.
   * Click “Create NAT Gateway”.
5. **Configure Route Tables**:
   * **Public Route Table**:
     + Go to Route Tables in the VPC Dashboard.
     + Click “Create Route Table”.
     + **Form Details**:
       - Name: christian-public-rtb
       - VPC: christian-vpc
     + Click “Create”.
     + Add a route with Destination 0.0.0.0/0 and Target as christian-igw.
     + Associate it with christian-public-subnet-1 and christian-public-subnet-2.
   * **Private Route Table**:
     + Create another route table named christian-private-rtb.
     + Add a route with Destination 0.0.0.0/0 and Target as the NAT Gateway.
     + Associate it with christian-private-subnet-1 and christian-private-subnet-2.
6. **Generate a Key Pair**:
   * Go to the EC2 Dashboard.
   * Click “Key Pairs” and then “Create Key Pair”.
   * **Form Details**:
     + Name: christian-key
   * Download the key pair .pem file.

### Image Captions

A screenshot of a computer

Description automatically generated

* *Screenshot 1: VPC dashboard showing the custom VPC with associated public and private subnets.*

## 2. Security Groups (SGs)

### Objective

Create and configure Security Groups for the Bastion host, EC2 instances, RDS, and ALB to ensure secure access and traffic flow.

### Steps

1. **Create Security Group for Bastion Host (christian-bastion-sg)**:
   * Go to Security Groups in the VPC Dashboard.
   * Click “Create Security Group”.
   * **Form Details**:
     + Name: christian-bastion-sg
     + VPC: christian-vpc
   * Inbound Rules:
     + SSH (port 22) from your IP (e.g., 203.0.113.0/24).
   * Outbound Rules:
     + Allow all traffic (0.0.0.0/0).
2. **Create Security Group for Web Servers (christian-web-sg)**:
   * **Form Details**:
     + Name: christian-web-sg
     + VPC: christian-vpc
   * Inbound Rules:
     + HTTP (port 80) and HTTPS (port 443) from christian-alb-sg.
   * Outbound Rules:
     + Allow all traffic (0.0.0.0/0).
3. **Create Security Group for RDS (christian-rds-sg)**:
   * **Form Details**:
     + Name: christian-rds-sg
     + VPC: christian-vpc
   * Inbound Rules:
     + PostgreSQL (port 5432) from christian-web-sg.
   * Outbound Rules:
     + Allow all traffic (0.0.0.0/0).
4. **Create Security Group for ALB (christian-alb-sg)**:
   * **Form Details**:
     + Name: christian-alb-sg
     + VPC: christian-vpc
   * Inbound Rules:
     + HTTP (port 80) and HTTPS (port 443) from anywhere (0.0.0.0/0).
   * Outbound Rules:
     + Allow all traffic to christian-web-sg.

### Image Captions

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* *Screenshot 2: Security Groups settings showing inbound and outbound rules for Bastion, EC2 instances, RDS, and ALB.*

## 3. EC2 Instance Setup

### Objective

Launch EC2 instances using Amazon Linux 2 AMI, configure SSH access via the Bastion host, and create a custom AMI.

### Steps

1. **Launch Bastion Host**:
   * Go to the EC2 Dashboard and click “Launch Instance”.
   * **Form Details**:
     + AMI: Amazon Linux 2 AMI
     + Instance Type: t2.micro
     + Network: christian-public-subnet-1
     + Security Group: christian-bastion-sg
     + Key Pair: christian-key
   * Click “Launch”.
2. **Launch Web Server EC2 Instances**:
   * **Form Details**:
     + AMI: Amazon Linux 2 AMI
     + Instance Type: t2.micro
     + Network: Private subnets (christian-private-subnet-1 and christian-private-subnet-2)
     + Security Group: christian-web-sg
     + Key Pair: christian-key
   * Configure the User Data script:
   * #!/bin/bash  
     yum update -y  
     yum install -y httpd php php-pgsql postgresql  
     systemctl start httpd  
     systemctl enable httpd  
     echo "<?php phpinfo(); ?>" > /var/www/html/index.php  
     aws s3 cp s3://christian-bucket/index.php /var/www/html/
3. **SSH Access via Bastion Host**:
   * SSH into the Bastion host:
   * ssh -i christian-key.pem ec2-user@<Bastion-IP-Address>
   * From the Bastion host, SSH into the private EC2 instance:
   * ssh -i /home/ec2-user/.ssh/christian-key.pem ec2-user@<Private-EC2-Private-IP>
4. **Deploy Web Application**:
   * Upload index.php to /var/www/html/ on the private EC2 instance.
   * Ensure index.php is configured with the correct RDS connection details.
5. **Create Custom AMI**:
   * After configuring the EC2 instance, create a custom AMI.
   * Name the AMI christian-web-ami.

### Image Captions

A computer screen with white text

Description automatically generated

* *Screenshot 3: Successful SSH connection to the private EC2 instance via the Bastion host.*
* \*Screenshot 4: Web application (index.php

) running and accessible via ALB, connected to RDS.\*

## 4. RDS (PostgreSQL Aurora) Setup

### Objective

Set up a PostgreSQL database using AWS RDS, ensuring it’s only accessible from private subnets.

### Steps

1. **Create RDS Instance**:
   * Go to the RDS Dashboard and click “Create database”.
   * **Form Details**:
     + Engine: PostgreSQL version 15
     + Instance Class: db.t3.micro
     + Multi-AZ: Disabled
     + Storage: Default settings
     + Subnet Group: Use private subnets (christian-private-subnet-1 and christian-private-subnet-2)
     + Security Group: christian-rds-sg
2. **Database Seeding**:
   * SSH into the private EC2 instance.
   * Connect to the RDS instance and create the required table:

* psql -h <RDS-Endpoint> -U postgres -d <dbname> -c "CREATE TABLE final ( id SERIAL PRIMARY KEY, fname VARCHAR NOT NULL, lname VARCHAR NOT NULL, created\_at TIMESTAMPTZ DEFAULT Now() );"

### Image Captions

A screenshot of a computer

Description automatically generated

* *Screenshot 5: RDS dashboard showing the single-AZ configuration and associated subnets.*
* *Screenshot 6: Security group settings for RDS showing no public access.*

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## 5. ALB and ASG Configuration

### Objective

Set up an ALB to distribute traffic and configure an ASG for automatic scaling.

### Steps

1. **Setup ALB**:
   * Go to the EC2 Dashboard and click “Create Load Balancer”.
   * **Form Details**:
     + Type: Application Load Balancer
     + Scheme: Internet-facing
     + Network: Public Subnets (christian-public-subnet-1 and christian-public-subnet-2)
     + Security Group: christian-alb-sg
   * Add Listeners:
     + HTTP (port 80) and HTTPS (port 443) forwarding to the target group.
2. **Create Target Groups**:
   * Go to the Target Groups section and click “Create Target Group”.
   * **Form Details**:
     + Protocol: HTTP
     + VPC: christian-vpc
     + Health Check Path: /index.php
   * Register the EC2 instances.
3. **Create Auto Scaling Group (ASG)**:
   * Go to the Auto Scaling section in the EC2 Dashboard.
   * Click “Create Auto Scaling Group”.
   * **Form Details**:
     + Launch Template: Use the custom AMI christian-web-ami
     + VPC: christian-vpc
     + Subnets: christian-private-subnet-1 and christian-private-subnet-2
     + Desired Capacity: 2
     + Min Capacity: 2
     + Max Capacity: 5
   * Configure scaling policies to scale out by 1 instance when CPU usage exceeds 80% for 2 minutes

.

### Image Captions

* *Screenshot 7: ALB configuration, listeners, and associated target group*

A screenshot of a computer

Description automatically generated*.*

* *Screenshot 8: ASG configuration, showing minimum, maximum, and desired instance count, along with scaling policies.*

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

## 6. Resource Cleanup

### Objective

Clean up AWS resources in a logical order to avoid unnecessary charges.

### Steps

1. **Delete Listener Rules**:
   * Navigate to the ALB Listeners section and remove the listener rules for HTTP and HTTPS.
2. **Delete Target Group**:
   * Go to the Target Groups section and delete the target group associated with the ALB.
3. **Delete Auto Scaling Group (ASG)**:
   * Go to the Auto Scaling Groups section and delete the ASG.
4. **Delete ALB**:
   * Go to the Load Balancers section and delete the ALB.
5. **Terminate EC2 Instances**:
   * Go to the EC2 Instances section and terminate all running instances.
6. **Delete RDS Instance**:
   * Navigate to the RDS Dashboard and delete the RDS instance.
7. **Delete NAT Gateway**:
   * Go to the NAT Gateways section and delete the NAT Gateway.
8. **Detach and Delete Internet Gateway (IGW)**:
   * Go to the Internet Gateways section, detach christian-igw from the VPC, and then delete it.
9. **Delete VPC and Subnets**:
   * Finally, go to the VPC Dashboard and delete the christian-vpc, including all associated subnets.

*.*