Análise de Arquitetura da Aplicação Flask na AWS

1. Descrição da Arquitetura

A arquitetura provisionada via Terraform na AWS foi projetada para hospedar uma aplicação Flask de forma segura, escalável e resiliente. A infraestrutura é modularizada e segue as melhores práticas de nuvem, utilizando uma topologia de Virtual Private Cloud (VPC) com sub-redes públicas e privadas distribuídas em múltiplas Zonas de Disponibilidade (AZs) para garantir alta disponibilidade.

Componentes Principais:

Componente	Descrição	F
VPC e Sub-redes	Uma VPC com CIDR `10.0.0.0/16` é dividida em sub-redes públicas e privadas em duas AZs.	
Application Load Balancer (ALB)	Posicionado na sub-rede pública, distribui o tráfego HTTP de entrada para os servidores web.	
Servidores Web (EC2)	Duas instâncias EC2 `t3.micro` em sub-redes privadas, executando a aplicação Flask.	
Banco de Dados (RDS)	Uma instância de banco de dados MySQL Multi-AZ em sub-redes privadas dedicadas.	
NAT Gateway	Provisionado em cada AZ para permitir que os recursos em sub-redes privadas acessem a internet.	
Bastion Host	Uma instância EC2 na sub-rede pública para acesso administrativo seguro aos servidores web.	
Logs e Monitoramento	Um bucket S3 centraliza os logs do ALB e CloudTrail. O VPC Flow Logs é enviado para o CloudWatch.	

2. Framework CSA CCM AWS

O desenvolvimento desta arquitetura foi orientado pelos princípios do **Cloud Security Alliance (CSA) Cloud Controls Matrix (CCM)**, um framework de controles de segurança

específico para ambientes em nuvem. A implementação na AWS segue as diretrizes do CSA para garantir uma postura de segurança robusta.

> O CSA CCM é projetado para fornecer princípios fundamentais de segurança para guiar os fornecedores de nuvem e auxiliar os potenciais clientes de nuvem na avaliação da segurança geral de um provedor de nuvem. [1]

Controles Aplicados:

- **IVS-01: Identity & Access Management:** O uso de roles do IAM para VPC Flow Logs e a política de bucket S3 com `Principal` e `Condition` específicos demonstram o princípio do menor privilégio.
- **DCS-01: Data-in-Transit Security:** O tráfego entre o ALB e os servidores web, e entre os servidores web e o RDS, ocorre dentro da VPC, protegido por Security Groups.
- **BCR-10: Business Continuity & Disaster Recovery:** A utilização de múltiplas AZs para o ALB, servidores web e a configuração Multi-AZ do RDS garantem a continuidade dos negócios em caso de falha de uma AZ.
- **LOG-01: Log Management:** A centralização de logs do ALB, CloudTrail e VPC Flow Logs em um bucket S3 e CloudWatch permite a análise e monitoramento contínuo.

3. Sugestões de Melhorias

A arquitetura atual é sólida, mas pode ser aprimorada em termos de escalabilidade e segurança. Apresentamos duas sugestões de melhorias:

Melhoria 1: Implementar Auto Scaling para os Servidores Web

Problema: A arquitetura atual utiliza um número fixo de duas instâncias EC2. Isso pode levar a problemas de performance em picos de tráfego ou a custos desnecessários em períodos de baixa utilização.

Solução: Substituir as instâncias EC2 fixas por um **Auto Scaling Group (ASG)**. O ASG pode ser configurado para escalar o número de instâncias horizontalmente com base em métricas como o uso de CPU ou o número de requisições.

Benefícios:

- **Elasticidade:** Ajusta a capacidade automaticamente para atender à demanda.
- **Disponibilidade:** Garante que um número mínimo de instâncias esteja sempre em execução.
- **Custo-benefício:** Paga-se apenas pela capacidade utilizada.

Melhoria 2: Gerenciamento de Segredos com AWS Secrets Manager

Problema: A senha do banco de dados é passada como uma variável sensível no Terraform. Embora o Terraform trate a variável como `sensitive`, ela ainda pode ser exposta em logs ou no estado do Terraform.

Solução: Utilizar o **AWS Secrets Manager** para armazenar e gerenciar a senha do banco de dados. A aplicação nos servidores web pode então recuperar a senha do Secrets Manager em tempo de execução, utilizando uma role do IAM com as permissões necessárias.

Benefícios:

- **Segurança:** A senha não é mais codificada no Terraform, reduzindo o risco de exposição.
- **Gerenciamento Centralizado:** Facilita a rotação de senhas e o controle de acesso.
- **Conformidade:** Atende a requisitos de conformidade que exigem o gerenciamento seguro de credenciais.

Referências

[1] Cloud Security Alliance. (2023). *Cloud Controls Matrix (CCM) v4*. https://cloudsecurityalliance.org/research/cloud-controls-matrix/