

A man with dark hair and a beard, wearing a blue shirt, is seated at a desk in a modern office. He is looking at several computer monitors. The monitors display various data visualizations, including bar charts, line graphs, and cloud diagrams. The background shows a large window with a view of a city skyline. The overall lighting is blue and professional.

Obligatorio Implementación de soluciones Cloud 2025

Gastón López N°126818
Sergio Nicolás N°101906

OBLIGATORIO IMPLEMENTACIÓN CLOUD

Autores: Sergio Nicolás, Gastón López

Contenido

URL de nuestro Repositorio Git 3

Declaración de Autoría 3

Resume ejecutivo 4

 Restricciones 4

 Requerimientos 4

Diagrama de arquitectura completo 5

Descripción de los Componentes y su Uso: 6

Bibliografía 7

URL de nuestro Repositorio Git

<https://github.com/snicolasg/Cloud2025.git>

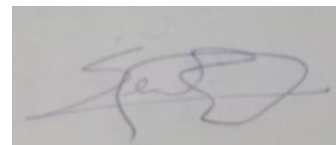
Declaración de Autoría

Nosotros, Sergio Nicolás y Gastón López, declaramos que el trabajo que se presenta en esta obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos el Obligatorio de Implementación de soluciones Cloud.
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega.



Gastón López



Sergio Nicolás

Resume ejecutivo

Planificamos nuestra solución para que sea práctica, sencilla, fácil de leer y que cumpla con los requerimientos solicitados.

Utilizamos buenas prácticas a la hora de desarrollar el código en terraform y kubernetes.

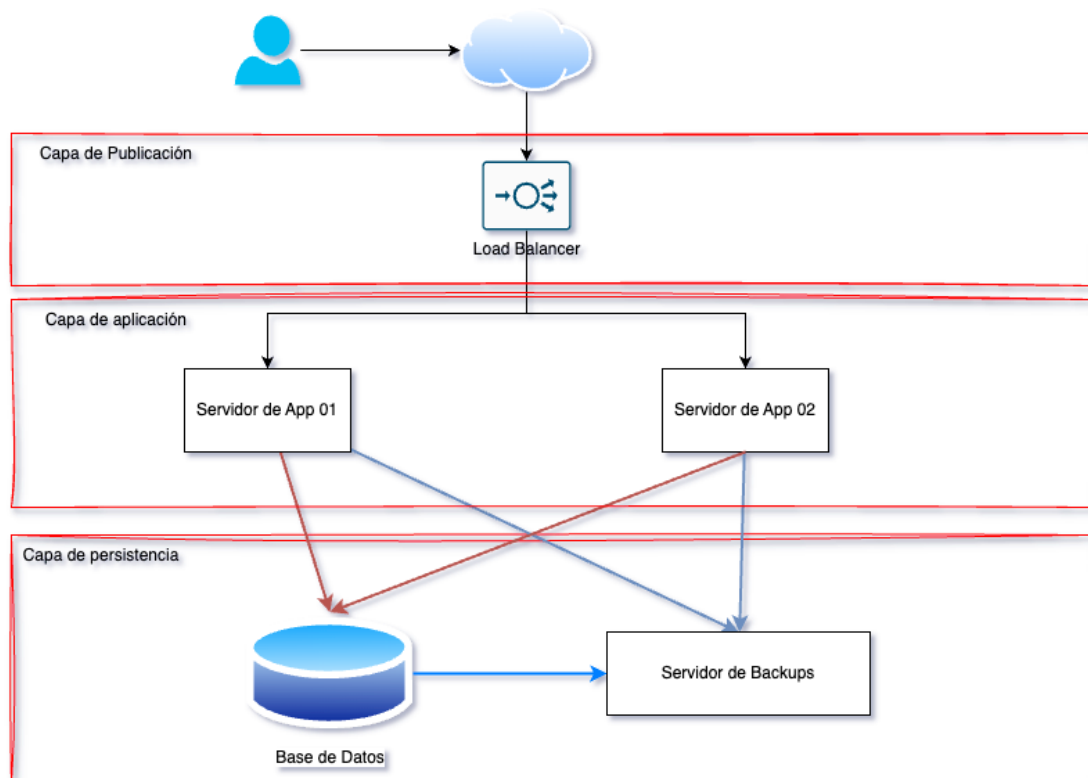
Está orientada a un mínimo costo operativo aplicando los servicios bajo demanda de modo de siempre usar lo que se necesita y liberar recursos que no se están usando. Almacenamiento acorde al uso para bajar los costos al mínimo, si son archivos que se necesitan entonces se almacenan con disponibilidad inmediata que tiene costo más elevado pero si son documentos de poco acceso ej: logs, se usa glacier para ahorrar costos.

Restricciones

Se realizaron todas las pruebas en CentOS Stream 9 en caso de usar otras distribuciones pude que haya que realizar algunos cambios en los comandos de los scripts.

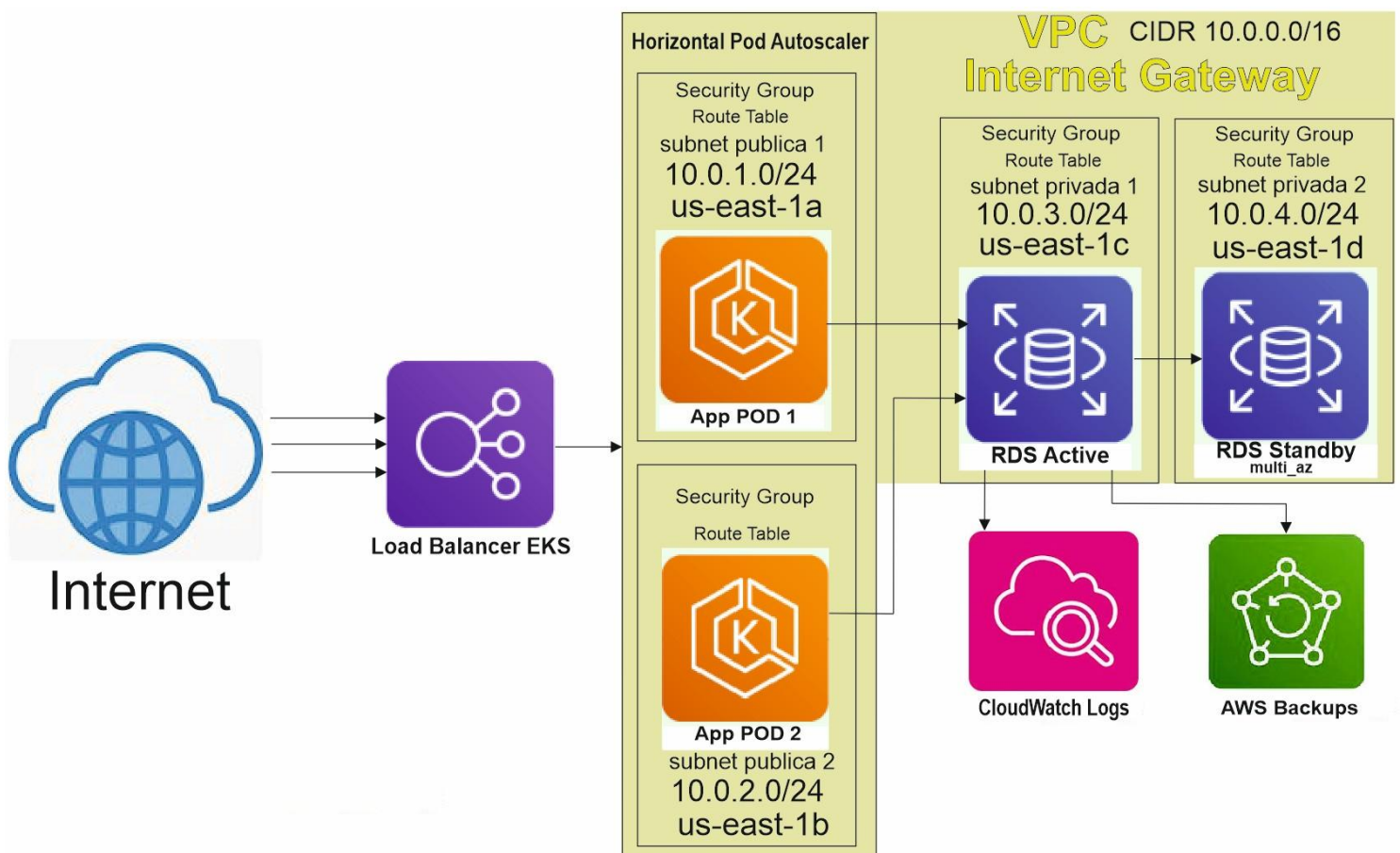
Requerimientos

1) La solución debe contemplar todos los componentes detallados a continuación:



- 2) La infraestructura debe ser tolerante a fallas.
- 3) Ante un escenario de picos de tráfico, la infraestructura debe poder soportar la carga.
- 4) Se debe restringir el firewall al máximo, habilitando solo los accesos necesarios.
- 5) Se debe proponer e implementar (si es posible) mejoras a la solución actual.
Ejemplos de mejoras pueden ser la implementación de monitoreo o concentración de logs.

Diagrama de arquitectura completo



En nuestro diseño el tráfico proveniente de internet hacia la aplicación va directo al load balancer quien se encarga de balancear la carga de trabajo distribuyendo las solicitudes entre los diferentes pods de nuestro cluster eks, contamos con 2 subnets públicas las cuales son 10.0.1.0/24 y 10.0.2.0/24 están en distintas zonas de disponibilidad de modo que si cae una zona la otra siga brindando el servicio a su vez ambas están dentro de un scaling group de este modo logramos que si hay mucho tráfico y cae la performance de nuestro cluster se agreguen más pods con la aplicación bajo demanda.

Los pods con la aplicación almacenan los datos en una base de datos que está en un servidor dentro de una subnet privada 10.0.3.0/24 la cual a su vez replica la información a otro servidor en otra subnet privada 10.0.4.0/24 así nos aseguramos de que si cae el servidor de base de datos principal la réplica pasa a ser activa y el tráfico se redirige hacia ella al instalar el servidor de base de datos con MultiAz y eso hace que use distintas subnets, nos aseguramos de que si una zona de disponibilidad cae la otra siga funcionando.

Hacemos respaldo con AWS Backup y se almacenan los Logs de la base de datos en CloudWatch

Descripción de los Componentes y su Uso:

1. Load Balancer EKS

Recibe tráfico desde Internet y lo redistribuye hacia la aplicación.

2. Horizontal POD Autoscaler

Escala automáticamente bajo demanda los pods que forman parte del clúster EKS para asegurar alta disponibilidad de la aplicación.

3. VPC 10.0.0.0/16 con cuatro Subredes publicas 10.0.1.0/24 10.0.2.0/24 y las privadas 10.0.3.0/24 y 10.0.4.0/24

Dos Subnets Públicas: Contienen los pods de aplicación en principio cuando la demanda es baja solo un pod por subnet, estas reciben tráfico del Load Balance EKS.

Dos Subnets Privadas: Una contiene el pod con la base de datos principal, la otra contiene el pod con la réplica de la base de datos. Están aisladas del acceso directo desde Internet.

4. Internet Gateway

Permite el acceso hacia internet a los pods con la aplicación.

5. Route Table

Creamos dos route table una para las subnets públicas que es donde se permite el tráfico a internet y la otra está asociada a las privadas.

6. Amazon EKS

Orquesta los pods que contienen los contenedores que a su vez ejecutan la aplicación y la base de datos.

7. Base de datos (RDS MultiAz)

Usamos una base de datos relacional My-Sql con MultiAz.

8. Amazon AWS Backups

Backup de la base de datos.

9. Amazon CloudWatch Logs

Se almacenan los logs de la base de datos.

10. Security Group (SG)

Lo usamos para controlar que el tráfico proveniente de Internet solo sea permitido hacia a nuestra aplicación vía HTTP y HTTPS en los puertos 80 y 443. También un SG que permite tráfico de ingreso al puerto 3306 de la base de datos.

Bibliografía

Documentación Terraform

https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/eks_cluster

Consulta con IA ChatGPT