Ing. Bryan Aldás

Reto Técnico

1. **¿Cuál es la diferencia entre nube pública, privada e hibrida?**

La principal diferencia entre estos tipos de nube radica principalmente en cómo se gestionan los recursos, quien los administra y el nivel de control y seguridad que ofrecen.

Nube pública: Todos los recursos son proporcionados por un proveedor de nube y son compartidos entre múltiples clientes a través de internet (AWS, GCP AZURE). Como principales características tenemos que esta infraestructura es gestionada por el proveedor de nube, proporciona alta escalabilidad y flexibilidad, ofrece un modelo de pago por uso, y ofrecen un mantenimiento y actualizaciones automáticas.

Nube privada: Es una infraestructura que es utilizada por una sola organización y puede estar alojada en las instalaciones de la empresa o gestionada por un proveedor externo. Como principales características tenemos que este tipo de infraestructura ofrece un mayor control y personalización de recursos, seguridad y cumplimiento más estricto, y requiere una inversión inicial alta.

Nube híbrida: Es una combinación de nube pública y privada que permite mover las diferentes cargas de trabajo entre ambas según la necesidad de negocio de la organización. Como principales características tenemos que ofrecen una mayor flexibilidad aprovechando lo mejor de ambos entornos, permite ejecutar cargas sensibles en la nube privada aprovechando la escalabilidad de la nube pública, y permite cumplir normativas sin perder beneficios de la nube pública.

2. **Describa tres prácticas de seguridad en la nube.**

Implementar controles de acceso: Es importante definir controles de acceso con el principio de menor privilegio. Esto permite que los usuarios y servicios solo tengan los permisos estrictamente necesarios.

Backups y recuperación ante desastres: Configurar backups automáticos en bases de datos y almacenamientos

Gestión y protección de credenciales: No almacenar credenciales en código fuente o repositorios públicos, es importante también implementar rotación de credenciales para evitar accesos prolongados en caso de una filtración.

3. **¿Qué es la IaC, y cuáles son sus principales beneficios?, mencione 2 herramientas de IaC y sus principales características.**

La infraestructura como código (IaC) es una metodología que permite gestionar y aprovisionar infraestructura mediante código, esto nos permite evitar configuraciones manuales. Su uso se basa en archivos de configuración o scripts en donde se define y automatiza la creación de la infraestructura, como por ejemplo redes, bases de datos, servidores, entre otros.

- Los principales beneficios son:

La automatización y consistencia: Nos evita configuraciones manuales y errores humanos y facilita despliegues repetibles y predecibles.

Portabilidad y Reutilización: en IaC se puede reutilizar el código para distintos entornos y es compatible con múltiples proveedores de nube.

- Herramientas IaC

Terraform: Soporta múltiples proveedores de nube, permite manejar dependencias entre recursos y funciona con estado remoto para gestionar cambios. Utiliza un lenguaje declarativo.

CloudFormation: Es un servicio nativo de AWS para definir IaC que utiliza archivos YAML o JSON para describir recursos. Tiene fácil integración con otros servicios de AWS para despliegues automatizados y permite actualizaciones controladas con rollback automático.

4. **¿Qué métricas considera esenciales para el monitoreo de soluciones en la nube?**

Métricas de rendimiento: Latencia, uso de CPU, uso de memoria, tiempos de carga, cantidad de solicitudes procesadas por segundo

Métricas de disponibilidad: Tiempos de actividad, tasas de error, health checks, número de instancias activas.

Métricas de seguridad: intentos de accesos fallidos, tráfico no autorizado, solicitudes bloqueadas.

Métricas de costos y optimización. Uso de recursos infrautilizados, gastos por servicios.

5. **¿Qué es Docker y cuáles son sus componentes principales?**

Docker es una plataforma de código abierto que permite crear, desplegar y gestionar aplicaciones en contenedores, estos son entornos ligeros y portables que

empaquetan una aplicación junto con sus dependencias (bibliotecas, configuraciones, etc), garantizando el correcto funcionamiento en cualquier entorno.

Componentes:

Docker Engine: Es el motor que permite la ejecución de contenedores (Docker Daemon, Api Rest, Cli)

Docker Image: Plantillas de lectura que definen como se construye el contenedor

Docker Container: Es una instancia en ejecución de una imagen

Docker Hub: Repositorio público donde se almacenan las imágenes oficiales y personalizadas

DockerFile: Archivo de texto que tiene todas las instrucciones para crear una imagen Docker personalizada.

Docker Compose: Herramienta que permite definir y gestionar aplicaciones multi contenedor en un archivo YAML, es ideal para orquestar servicios como una app web + base de datos.

Docker Volumes: Permite persistir datos generados por contenedores para evitar pérdida de información al reiniciar

Docker networks: Permite la comunicación segura entre contenedores y con aplicaciones externas.

Docker Swarm: Herramienta nativa de Docker para orquestar clústeres de contenedores.

6. **Caso Práctico:**

Cree un diseño de arquitectura para una aplicación nativa de nube considerando los siguientes componentes:

- Frontend: Una aplicación web que los clientes utilizarán para navegación.
- Backend: Servicios que se comunican con la base de datos y el frontend.
- Base de datos: Un sistema de gestión de base de datos que almacene información.
- Almacenamiento de objetos: Para gestionar imágenes y contenido estático.

Diseño:

a. Seleccione un proveedor de servicios de nube (Aws, Azure o GCP) y sustente su selección.
b. Diseñe una arquitectura de nube. Incluya diagramas que representen la arquitectura y justifique sus decisiones de diseño (Utilice https://app.diagrams.net/).