Propuestos:

**1. Conceptos Básicos de la Nube**

**Problema Propuesto:**

Una empresa tiene un servidor físico para almacenar datos de clientes y necesita mover su infraestructura a la nube para reducir costos, mejorar escalabilidad y garantizar la disponibilidad.

**Solución:**

1. **Entender los Modelos de Nube:**
   * **IaaS (Infraestructura como Servicio):** Proveedores como AWS, Azure o Google Cloud ofrecen servidores virtuales configurables.
   * **PaaS (Plataforma como Servicio):** Herramientas como AWS Elastic Beanstalk facilitan la implementación de aplicaciones.
   * **SaaS (Software como Servicio):** Aplicaciones listas para usar, como Gmail o Dropbox.
2. **Seleccionar un Proveedor de Nube:**
   * Investiga los costos y servicios ofrecidos por proveedores como AWS, Azure o Google Cloud.
   * Evalúa el nivel de soporte que necesitarás.
3. **Configurar la Infraestructura:**
   * Crea una cuenta en AWS (puedes usar la capa gratuita para pruebas).
   * Accede a la consola de AWS y explora los servicios disponibles, como **Amazon EC2** (servidores virtuales) y **Amazon S3** (almacenamiento de objetos).
4. **Migrar los Datos:**
   * Usa **AWS S3** para cargar los archivos de clientes.
   * Configura buckets con políticas adecuadas para controlar el acceso.
5. **Beneficios Logrados:**
   * Reducción de costos de hardware.
   * Escalabilidad en función de la demanda.
   * Alta disponibilidad de datos.

**2. Introducción a los Servicios y Arquitecturas de AWS**

**Problema Propuesto:**

Diseñar una arquitectura básica en AWS que permita alojar una aplicación web con alta disponibilidad y escalabilidad.

**Solución:**

1. **Definir los Componentes Necesarios:**
   * **Frontend:** Código estático de tu aplicación.
   * **Backend:** Lógica y procesamiento.
   * **Base de Datos:** Almacenamiento de datos de la aplicación.
2. **Implementar la Arquitectura:**
   * **Frontend:**
     + Usa **Amazon S3** para almacenar y servir contenido estático. (mybucketacademiadataengineer)
     + Configura **Amazon CloudFront** para acelerar el acceso a nivel global.
   * **Backend:**
     + Crea una instancia EC2 para alojar la lógica de la aplicación. (curso-data-engineer)
     + Configura un balanceador de carga con **Elastic Load Balancer (ELB)**.
   * **Base de Datos:**
     + Usa **Amazon RDS** para configurar una base de datos relacional como MySQL o PostgreSQL.
3. **Configurar la Escalabilidad:**
   * Usa **Auto Scaling Groups** para ajustar automáticamente el número de instancias EC2 según la demanda.
   * Habilita escalabilidad automática en Amazon RDS para manejar cargas de base de datos más pesadas.
4. **Conectar los Componentes:**
   * Configura permisos de seguridad con **Security Groups**.
   * Asegúrate de que las instancias EC2 y el RDS puedan comunicarse.
5. **Prueba y Monitoreo:**
   * Usa **Amazon CloudWatch** para monitorear el rendimiento de los servicios.
   * Configura alertas para eventos críticos.

|  |  |  |
| --- | --- | --- |
| ***Componentes*** | ***Nombre*** | ***Descripción*** |
| EC2 y Bucket en S3 | curso-data-engineer, mybucketacademiadataengineer |  |
| Cloud Watch (Amazon CloudFront) |  |  |
| Elastic Load Balancer (ELB). |  | Para monitoreo |
| Amazon RDS |  |  |
| IAM (Identity and Access Management) |  |  |

**3. Configuración y Seguridad Básica en AWS**

**Problema Propuesto:**

Tu equipo quiere asegurarse de que los datos almacenados en AWS están protegidos contra accesos no autorizados y pérdidas.

**Solución:**

1. **Configurar Seguridad Básica:**
   * Usa **IAM (Identity and Access Management)** para:
     + Crear usuarios y roles con permisos mínimos necesarios.
     + Habilitar el acceso programático con claves de acceso.
   * Configura el **Multi-Factor Authentication (MFA)** para usuarios con acceso a la consola.
2. **Asegurar los Datos en S3:**
   * Configura políticas de bucket para restringir el acceso.
   * Habilita el cifrado de datos en reposo usando **AWS KMS** (Key Management Service).
   * Usa **S3 Versioning** para mantener versiones anteriores de los archivos.
3. **Configurar Auditorías y Monitoreo:**
   * Habilita **AWS CloudTrail** para registrar todas las acciones en la cuenta.
   * Usa **AWS Config** para verificar que los recursos cumplen con las mejores prácticas.
4. **Proteger la Red:**
   * Configura **Security Groups** y **Network ACLs** para controlar el tráfico de red entrante y saliente.
   * Implementa un **Web Application Firewall (WAF)** para proteger contra ataques comunes como SQL Injection o Cross-Site Scripting.
5. **Simular Incidentes:**
   * Realiza simulaciones de recuperación de datos.
   * Prueba la rotación de claves de acceso y contraseñas.

**1. Migración Segura de una Aplicación Web a la Nube**

**Problema Propuesto:**

Una empresa tiene una aplicación web alojada en servidores locales. Necesitan migrarla a AWS para mejorar la escalabilidad y seguridad, garantizando que los datos de los usuarios estén protegidos y que la aplicación tenga alta disponibilidad.

**Solución:**

1. **Preparación para la Migración:**
   * Realiza un inventario de los recursos actuales de la aplicación (frontend, backend, base de datos, etc.).
   * Identifica los permisos necesarios para el equipo que trabajará en la migración y configúralos usando **IAM**.
2. **Configuración de la Infraestructura:**
   * **Frontend:**
     + Almacena los archivos estáticos en **Amazon S3** y configura **CloudFront** para la distribución global.
   * **Backend:**
     + Configura instancias **EC2** detrás de un **Elastic Load Balancer (ELB)** para manejar el tráfico.
     + Usa **Auto Scaling Groups** para ajustar el número de instancias según la demanda.
   * **Base de Datos:**
     + Migra los datos a una instancia de **Amazon RDS** con cifrado habilitado.
3. **Garantizar la Seguridad:**
   * Habilita el cifrado en los buckets de S3 con **AWS KMS**.
   * Crea **Security Groups** que permitan únicamente el tráfico necesario.
   * Configura políticas para que solo ciertos usuarios o roles puedan acceder a los recursos.
4. **Pruebas y Optimización:**
   * Usa **CloudWatch** para monitorear el rendimiento de la aplicación.
   * Habilita **CloudTrail** para registrar las acciones realizadas durante la migración.
5. **Lanzamiento y Evaluación:**
   * Realiza pruebas de carga para verificar la escalabilidad.
   * Implementa alertas para detectar problemas de rendimiento o intentos de acceso no autorizados.

**2. Construcción de una Plataforma de Datos Segura y Escalable**

**Problema Propuesto:**

Un startup de análisis de datos necesita una solución en la nube para almacenar y procesar grandes volúmenes de datos de manera segura, asegurando el acceso controlado para el equipo.

**Solución:**

1. **Diseño de la Arquitectura:**
   * Almacena los datos en bruto en **Amazon S3** con **Versioning** habilitado.
   * Usa **AWS Glue** para transformar y limpiar los datos.
   * Almacena los datos procesados en **Amazon Redshift** para análisis rápido.
2. **Configuración de Seguridad:**
   * Limita el acceso a los buckets de S3 usando políticas específicas.
   * Cifra los datos almacenados en S3 y Redshift usando **AWS KMS**.
   * Configura roles en **IAM** para que los servicios (como Glue) accedan a los recursos necesarios.
3. **Automatización de Procesos:**
   * Usa **AWS Step Functions** para orquestar tareas de extracción, transformación y carga (ETL).
   * Configura un job recurrente en Glue para procesar los datos automáticamente.
4. **Escalabilidad:**
   * Habilita **Auto Scaling** para Glue y Redshift para manejar picos de carga.
   * Usa **CloudFront** para distribuir reportes o dashboards a usuarios finales.
5. **Monitoreo y Auditoría:**
   * Usa **CloudWatch** para monitorear el rendimiento de los jobs de Glue y el uso de Redshift.
   * Habilita **CloudTrail** para registrar quién accede a los datos y realiza cambios.

**3. Implementación de un Sitio Web de Comercio Electrónico Escalable y Seguro**

**Problema Propuesto:**

Un pequeño negocio de comercio electrónico quiere lanzar su tienda en AWS, asegurando que los datos de los clientes estén protegidos y que el sitio pueda manejar incrementos de tráfico, especialmente en temporadas de ventas altas.

**Solución:**

1. **Diseño de la Arquitectura:**
   * **Frontend:**
     + Almacena el contenido estático (imágenes, CSS, JS) en **Amazon S3** y distribúyelo con **CloudFront**.
   * **Backend:**
     + Despliega el backend en instancias **EC2** detrás de un **Elastic Load Balancer**.
     + Configura un **Auto Scaling Group** para manejar aumentos de tráfico.
   * **Base de Datos:**
     + Usa **Amazon RDS** para almacenar la información de productos y pedidos.
     + Almacena la sesión de los usuarios en **Amazon ElastiCache** para mejorar el rendimiento.
2. **Configuración de Seguridad:**
   * Usa **IAM Roles** para que los servicios accedan a los datos necesarios.
   * Implementa un **Web Application Firewall (WAF)** para proteger contra ataques comunes como SQL Injection.
   * Cifra los datos sensibles almacenados en RDS.
3. **Implementación de Medidas de Alta Disponibilidad:**
   * Configura **Multi-AZ Deployment** en RDS para garantizar que la base de datos esté disponible incluso si una zona de disponibilidad falla.
   * Habilita la replicación de contenido en CloudFront para usuarios globales.
4. **Optimización y Escalabilidad:**
   * Configura **Auto Scaling** tanto en EC2 como en ElastiCache para manejar variaciones en la carga.
   * Monitorea el tráfico con **CloudWatch** y ajusta las reglas de escalado si es necesario.
5. **Pruebas y Lanzamiento:**
   * Realiza pruebas de carga para simular temporadas altas de tráfico.
   * Implementa alertas para fallos en los componentes clave de la arquitectura.