Título

FASE 2: MGN - Rehosting Server with Application Migration Service (MGN)

Estado

Aceptada

Fecha

2023-10-20

Contexto

- La base de datos de nuestro cliente ya está migrada a AWS, y el backend App se conecta a nuestra base de datos migrada. En nuestro servidor origen onprem hemos eliminado nuestra base de datos y aplicativo innecesario para migrar la menor cantidad de datos posible. Hemos conectado los 2 entornos mediante VPN para una migración segura. Por lo que podemos proceder realizar un Lift & Shift (rehosting) del servidor onprem a AWS. Sin embargo, nos planteamos que aunque la base de datos se encuentra en alta disponibilidad ya que durante la fase anterior de migración se ha creado una Read Replica en otra AZ dentro de la misma región, nuestra aplicación monolito no es una aplicación distribuida y por tanto solo puede escalar verticalmente. Nuestro objetivo inicial de migración del monolito hacia una arquitectura serverless basada en microservicios es un proceso que se realizará de forma parcial y a medio-largo plazo.

Decisión

- El equipo decide usar Application Migration Service (MGN) para realizar el Lift & shift de nuestro servidor on prem, la cual nos permite realizar una migración con un mínimo downtime, sin pérdida de datos y reduciendo el riesgo de error humano.

- Nuestro Monolito es stateful (mantiene información de estado y depende de un sistema de almacenamiento persistente), aquí se plantea la posibilidad de dar Alta disponibilidad a nuestro monolito mediante un Autoscaling group que

abarca 2 o 3 AZ, todos los datos de nuestra backend server están dentro de un volumen de EBS el cual está atado a una sola AZ, imaginemos que por algún motivo ya sea caída de la AZ, fallo de red en la AZ etc., nuestra instancia va a ser terminada. Con autoscaling group podemos usar Lifecycle hooks on termination y gracias a esto podemos crear un script que cree un snapshot de dicho volumen de EBS con su correspondiente TAG, seguidamente Autoscaling group que tiene configurado un min, desired an max de 1 va a crear una instancia que reemplazará la anterior de igual crearemos un lifecycle hook en el evento de lanzamiento el cual mediante un script creará un volumen EBS de nuestro snapthot anterior en otra AZ y hará un attachment a la instancia creada por nuestro Autoscaling group. Nuestro monolito está en una subnet privada, para mayor seguridad ponemos delante un Application Load balancer en la parte pública que, aunque no balacea nada nos deja un único punto de exposición, mejorando la seguridad.

Consecuencias

Positivas:

- La replicación es fácil de configurar y no requiere una curva de aprendizaje empinada.
- Es rápido de escalar, mediante la implementación de agentes en las máquinas de origen.
- Puede utilizar Cloud Migration Factory para automatizar la mayoría de las tareas manuales, administrar varias máquinas y orquestar las olas de migración.
- Admite una amplia gama de arquitecturas de sistema operativo x86.
- Es agnóstico a la aplicación, por lo que admite cualquier aplicación que se ejecute en los servidores origen.
- No se requiere licencia ni compra. AWS ofrece precios por uso, por lo que solo paga por el servicio mientras lo usa, sin necesidad de contratos a largo plazo o licencias complejas.

Negativas:

- Los sistemas que tienen una alta tasa de cambios de datos (como las bases de datos OLTP) pueden tener requisitos más altos de rendimiento o red, lo que complica los esfuerzos de migración.
- La ventana de tiempo de inactividad debe ser pequeña, dentro de un RTO (tiempo máximo para recuperarse) de minutos.

Compliance

- Nuestro cliente opera en Europa y Estados Unidos Por lo que ha de cumplir con los estándares GDPR e HIPAA)

Procedimiento

- https://catalog.workshops.aws/well-architected-migration/en-US/2-migration/1-initial-credentials-setup/2-quotas

Notas

- Autor: Fran Díaz
- Version: 0.2
- Changelog:
    - 0.1: Versión Inicial propuesta.
    - 0.2: Cambio propuestos