Configuración de un Balanceador de Cargas de Aplicaciones (GSP155)

Este documento detalla la implementación de un Balanceador de Cargas de Aplicaciones (L7) en Google Cloud, como parte de mi preparación para la certificación de **Associate Cloud Engineer**.

Detalles del Proyecto	
Certificación	Google Associate Cloud Engineer
Laboratorio	GSP155: Configura balanceadores de cargas de aplicaciones
Tecnologías Clave	Google Cloud Load Balancing (L7), Compute Engine, GCE Instance Groups, `gcloud` CLI
Resultado	Un servicio web global, escalable y de alta disponibilidad.

1. Resumen del Proyecto (Qué es y Para qué es)

Un **Balanceador de Cargas de Aplicaciones (ALB)** de Google Cloud es un servicio global de Capa 7 (L7) que gestiona y distribuye el tráfico HTTP y HTTPS.

A diferencia de un balanceador de cargas de red (Capa 4), un ALB es "inteligente": puede inspeccionar el tráfico y tomar decisiones de enrutamiento basadas en el contenido de la solicitud, como la URL (`/video` , `/images`), los encabezados o las cookies.

El propósito de esta arquitectura es crear un punto de entrada único (una sola dirección IP) para una aplicación, que luego distribuye de manera eficiente las solicitudes de los usuarios a múltiples servidores backend. Esto garantiza:

- **Alta Disponibilidad:** Si un servidor falla, el balanceador deja de enviarle tráfico, y la aplicación sigue funcionando.
- **Escalabilidad:** Se pueden agregar o quitar servidores de forma automática (usando Grupos de Instancias Administrados) para manejar picos de tráfico.
- **Rendimiento:** Dirige a los usuarios al backend más cercano y saludable.

2. Objetivo del Laboratorio

El objetivo de este laboratorio fue configurar y desplegar un Balanceador de Cargas de Aplicaciones externo para distribuir el tráfico entre dos backends de servidor web idénticos. El laboratorio requería usar la CLI de `gcloud` para construir cada componente de la arquitectura, desde las instancias de cómputo hasta las reglas de reenvío globales.

3. Arquitectura Implementada (Qué hice exactamente)

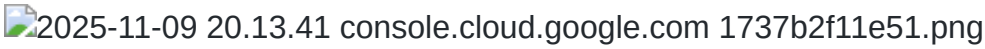
La arquitectura de un balanceador de cargas L7 se divide en dos secciones principales: **Frontend** (la "entrada") y **Backend** (el "destino").

Componentes de Backend (El "Destino"):

1. **Plantilla de Instancias (`lb-backend-template`):** Creé una "receta" para las máquinas virtuales. Esta plantilla definió el tipo de máquina, la imagen del SO (Debian 11) y un script de inicio que instala el servidor web Apache.
2. **Grupo de Instancias Administrado (`lb-backend-group`):** Usando la plantilla, creé un grupo que automáticamente despliega y gestiona un conjunto de VMs idénticas (en este caso, 2). Este grupo es responsable de la auto-reparación y el escalado.
3. **Verificación de Estado (`http-basic-check`):** Configuré una regla que el balanceador usa para preguntar constantemente a las VMs: "¿Estás saludable?". Solo las VMs que responden "sí" a esta verificación (en el puerto 80) reciben tráfico.
4. **Servicio de Backend (`web-backend-service`):** Este es el "cerebro" del backend. Es un recurso global que organiza y gestiona los grupos de instancias (nuestro `lb-backend-group`) y les aplica la verificación de estado (`http-basic-check`).

Componentes de Frontend (La "Entrada"):

1. **Dirección IP Global Estática (`lb-ipv4-1`):** Reservé una única dirección IP pública y estática (`34.110.218.254`). Esta es la dirección que los usuarios de todo el mundo usarán para acceder a la aplicación.

   
   2025-11-09 20.13.41 console.cloud.google.com 1737b2f11e51.png

2. **Mapa de URLs (`web-map-http`):** Define las reglas de enrutamiento. Para este lab, la regla era simple: "enviar todo el tráfico (ruta `/*`) al `web-backend-service`". En un proyecto real, aquí podría configurar rutas como `/api` a un backend y `/static` a otro.
3. **Proxy HTTP de Destino (`http-lb-proxy`):** Es el "portero" que recibe la solicitud HTTP del usuario. Consulta el Mapa de URLs para saber qué hacer con esa solicitud.

4. **Regla de Reenvío Global (`http-content-rule`):** La pieza final que conecta todo. Esta regla vincula la dirección IP pública (`34.110.218.254` en el puerto 80) con el proxy (`http-lb-proxy`).

Flujo de Tráfico:

Usuario ➔ IP Global (34.110.218.254) ➔ Regla de Reenvío ➔ Proxy HTTP ➔ Mapa de URLs ➔ Servicio de Backend ➔ Grupo de Instancias ➔ VM saludable

4. Detalles Técnicos y Comandos Clave

Todo el despliegue se realizó a través de la CLI `gcloud` . A continuación, se detallan los comandos más críticos y su propósito:

```
# --- 1. CONFIGURACIÓN DEL BACKEND ---

# Crear la plantilla para las VMs
gcloud compute instance-templates create lb-backend-template \
   --region=us-west1 \
   --tags=allow-health-check \
   --machine-type=e2-medium \
   --metadata=startup-script='...'

# Crear el grupo de 2 VMs usando la plantilla
gcloud compute instance-groups managed create lb-backend-group \
   --template=lb-backend-template --size=2 --zone=us-west1-c

# Crear la regla de firewall para permitir que Google verifique la salud
gcloud compute firewall-rules create fw-allow-health-check \
  --source-ranges=130.211.0.0/22,35.191.0.0/16 \
  --target-tags=allow-health-check \
  --rules=tcp:80

# Crear la verificación de estado (Health Check)
gcloud compute health-checks create http http-basic-check --port 80

# Crear el servicio de backend (el "cerebro")
gcloud compute backend-services create web-backend-service \
  --protocol=HTTP \
  --health-checks=http-basic-check \
  --global

# Conectar el grupo de VMs al servicio de backend
gcloud compute backend-services add-backend web-backend-service \
  --instance-group=lb-backend-group \
  --instance-group-zone=us-west1-c \
  --global

# --- 2. CONFIGURACIÓN DEL FRONTEND ---

# Reservar la IP pública global
gcloud compute addresses create lb-ipv4-1 \
  --ip-version=IPV4 \
  --global

# Crear el mapa de URLs para dirigir el tráfico
gcloud compute url-maps create web-map-http \
    --default-service web-backend-service

# Crear el proxy que recibe las solicitudes
gcloud compute target-http-proxies create http-lb-proxy \
    --url-map web-map-http

# Crear la regla de reenvío: la conexión final
```

```
gcloud compute forwarding-rules create http-content-rule \
   --address=lb-ipv4-1 \
   --global \
   --target-http-proxy=http-lb-proxy \
   --ports=80
```

5. Verificación y Logros (Qué logré)

El logro principal fue desplegar con éxito un servicio web resiliente y distribuido.

La verificación se realizó de dos maneras:

1. **Consola de Google Cloud:** Se confirmó que el servicio de backend (`web-backend-service`) mostraba **2 de 2** instancias en estado **"HEALTHY"** (Saludable).
2. Prueba del Navegador: Al acceder a la dirección IP pública reservada (http://34.110.218.254), el navegador mostró la página web con el mensaje:

   Page served from: lb-backend-group-rhg5

   (Al recargar la página, el nombre del servidor cambiaba, confirmando que el balanceo de carga estaba activo y distribuyendo las solicitudes entre las diferentes VMs del grupo).

6. Conclusión

Este laboratorio fue una demostración práctica de cómo construir una arquitectura web fundamental en Google Cloud. Comprendí la diferencia crítica entre los componentes de frontend y backend de un balanceador de cargas y cómo cada recurso (`Forwarding Rule`, `Target Proxy`, `URL Map`, `Backend Service`) juega un papel específico e interconectado.

Este proyecto solidifica mi comprensión de la alta disponibilidad y la escalabilidad, conceptos centrales para el rol de un Ingeniero de la Nube.