Universidad de los Andes

Documento de Arquitectura de la aplicación ANB en AWS

Autores:

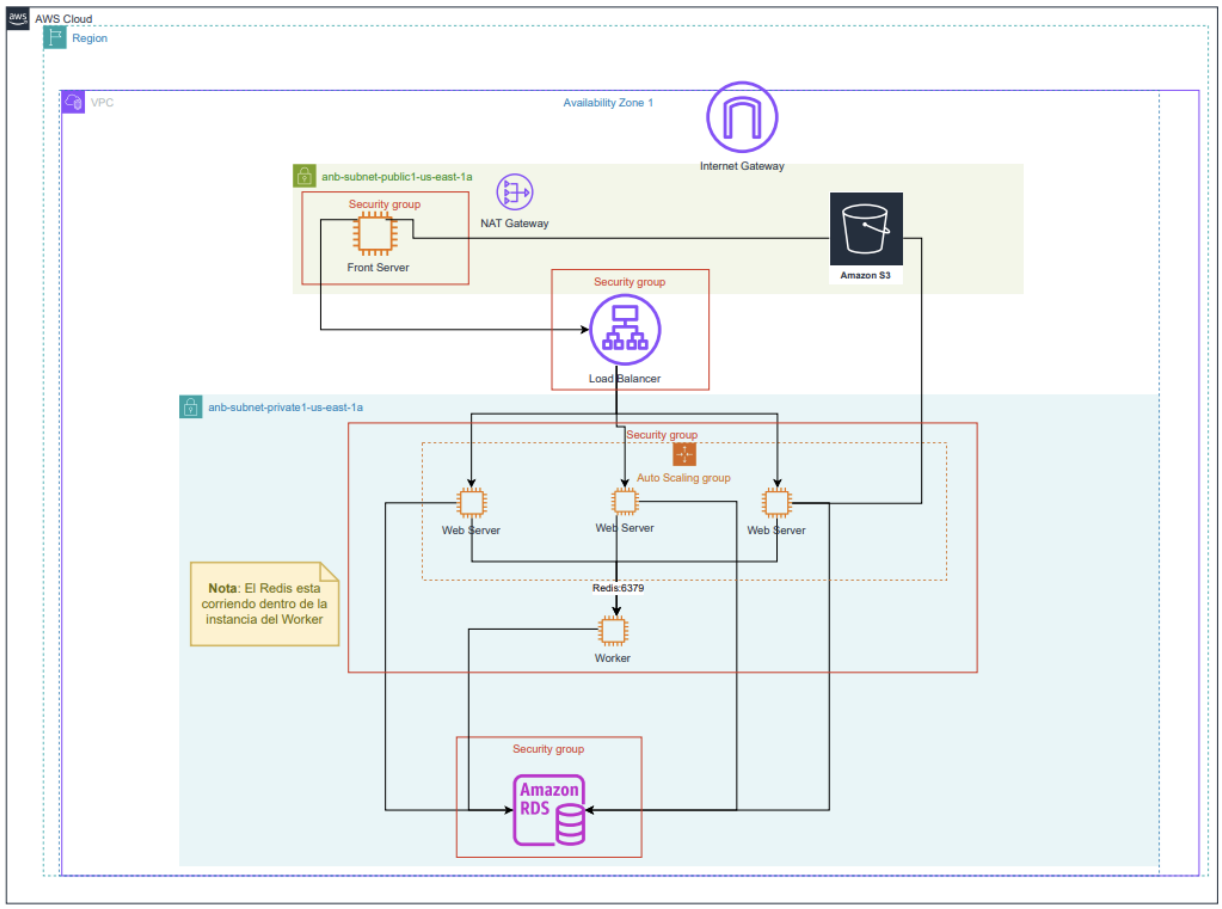
Stiven Andres Viedman Agudelo
Gabriel Gómez Corredor
Andrés Felipe Gámez Vargas
Andrés Felipe Chaparro Diaz

Bogotá – Octubre, 2025

Resumen ejecutivo

Este informe describe los recursos en la nube empleados para la construcción de la arquitectura de despliegue de la aplicación ANB. Se presentan en detalle los tipos de instancias seleccionados, las capacidades de las máquinas (incluyendo la configuración del grupo de autoescalado), la base de datos utilizada y las configuraciones de red implementadas (uso del balanceador de carga), con el fin de garantizar una arquitectura segura, eficiente, escalable y alineada con las mejores prácticas.

Diagrama de arquitectura AWS



Cambios respecto a la entrega 2

Con relación a la entrega anterior, se realizaron varios cambios en la arquitectura de la aplicación con el propósito principal de hacerla escalable (lo cual se evidenciará en los resultados de las pruebas de carga). La escalabilidad se hizo horizontalmente en el API de la aplicación y verticalmente en el almacenamiento de videos. En la primera parte (API) se configuro un balanceador de carga y también un grupo de "Autoscaling" de instancias de EC2 que tienen desplegado el API de la aplicación, esto por supuesto requirió de la creación de un AMI, configuración de grupos de seguridad, subredes y un launch template. Por otro lado, en la entrega anterior se había configurado una instancia EC2 que funcionaba como un NFS para el almacenamiento de los videos procesados y sin procesar. Para escalar esta parte de la arquitectura, se reemplazó la instancia NFS por un bucket de S3 para guardar/consumir los videos. Esto implica una escalabilidad vertical dado que las capacidades de almacenamiento de un bucket son virtualmente infinitas respecto al almacenamiento local de un servidor NFS.

Componentes de la arquitectura

Computo

En cuanto a los recursos de cómputo, se empleó el servicio Amazon Elastic Compute Cloud (EC2) para el despliegue de las aplicaciones de backend (junto con todas las instancias de auto-escalado), worker y frontend. A continuación, se detallan las características de cada uno de estos nodos. Todas las instancias están alojadas dentro de la misma Virtual Private Cloud (**VPC**).

- **VPC**:
    - **Id:** vpc-07e55b2b1ab1ff090f4
    - **Nombre**: anb-vpc

Front

- **Nombre de instancia:** Front Server
- **Tipo de instancia:** t2.large
- **Capacidad:** 2 vCPU – 8G Ram
- **SubRed:**
    - **Tipo**: Pública
    - **Nombre**: anb-subnet-public1-us-east-1a

- **Grupo de seguridad:** Se creó un grupo de seguridad que protegiera la instancia del front.

Nombre	AWS ID	Reglas de entrada	Reglas de salida
Fron-security-group	sg-04c0ec4385f642e2c	SSH:22 – 0.0.0.0/0 HTTP:80 – 0.0.0.0/0 TCP Pers:3000 – 0.0.0.0/0	TCP Pers:8080 – 0.0.0.0/0 Todo el tráfico

- **Descripción**: El servidor del front es la única instancia que se encuentra en la subred publica, porque será el punto de entrada del usuario desde el exterior, las demás instancias al estar en la subred privada no tendrán acceso directo desde el internet público. Por esta razón, esta instancia del front tiene dos objetivos. El primero, exponer la interfaz de la aplicación, y las peticiones que vengan desde el navegador debe dirigirlas ahora al DNS del balanceador de carga para que se pueda redirigir/asignar la petición a una de las instancias del grupo de autoescalado. También el front server debe redireccionar las peticiones de videos al almacenamiento existente en S3. El front esta desarrollado en Next.js, aprovechamos el servidor de Next.js para que actúe como un middlware, un reverse proxy que nos ayude a redireccionar esas peticiones al balanceador de carga. El segundo objetivo de esta instancia es actuar como un servidor Bastión, porque nuestros servidores Web Server (backend) y Worker se encuentran en la subred privada no se puede acceder desde el internet público, el Front Server actúa como puente para poder comunicarnos con los servidores privados.

Backend

- **Nombre de instancia:** Web Server
- **Tipo de instancia:** t2.larger
- **Capacidad:** 2 vCPU – 8G Ram
- **SubRed:**
  - **Tipo**: Privada
  - **Nombre**: anb-subnet-private1-us-east-1a
- **Grupo de seguridad:** Se creó un grupo de seguridad que protegiera la instancia del back y del worker.

Nombre	AWS ID	Reglas de entrada	Reglas de salida

Back-security-group	sg-080b3035f29fb99c9	SSH:22 – 0.0.0.0/0 TCP Pers:6379– 0.0.0.0/0 (Redis) PostgreSQL TCP: 5432 – 0.0.0.0/0 TCP Pers:8080 – 0.0.0.0/0	Todo el tráfico

- **Descripción**: La instancia del back se comunica con la base de datos PostgreSQL, la cola (Redis) donde va a encolar los videos que serán procesados por el worker y también se autentica y conecta con el servicio de S3 para almacenar los videos en la carpeta "/uploads" dentro del bucket "anb-videos" para que el worker pueda acceder a esa carpeta y procesar los videos pendientes.

Grupo de auto-escalado

- **Nombre del Grupo:** BackendASG
- **Tipo de instancias:** t2.larger
- **Capacidad:** 2 vCPU – 8G Ram
- **SubRed:**
  - **Tipo**: Privada
  - **Nombre**: anb-subnet-private1-us-east-1a
- **Grupo de seguridad**: Es el mismo grupo configurado para la instancia del backend. Esto implica que contiene las mismas reglas de entrada y salida de tráfico.
- **Capacidad deseada**: 2
- **Instancias mínimo**: 1
- **Instancias máximo**: 3
- **AMI**: Se creo la AMI (ID: ami-0ef0387be4a859051) de la imagen del servidor del backend (con Amazon Linux 2023 como OS) con la configuración de la imagen y el contenedor que lanzan el API de la aplicación y sus respectivas variables de entorno.
- **Launch Template**: Se configuró la plantilla de lanzamiento (ID lt-096431fbc49142e9) de las instancias de auto-escalado para que fuera una copia directa del servidor Backend, de modo que su creación y puesta en marcha fuera rápida. Se realizaron algunas pruebas y se encontró que configurar la plantilla de esta manera y no con un

script (user_data) hacia que las instancias se lanzaran rápido y el proceso de escalado fuera más óptimo.

- **Descripción**: Las instancias del grupo de auto-escalado se lanzan cuando se detecta con CloudWatch un uso promedio de CPU del 60% en el servidor del backend. De acuerdo con la configuración, el balanceador de carga equilibra las peticiones entre todas las instancias que se creen.

Worker

- **Nombre de instancia:** Worker
- **Tipo de instancia:** t2.larger
- **Capacidad:** 2 vCPU – 8G Ram
- **SubRed:**
  - **Tipo**: Privada
  - **Nombre**: anb-subnet-private1-us-east-1a
- **Grupo de seguridad:** Se creó un grupo de seguridad que protegiera la instancia del back y del worker.

Nombre	**AWS ID**	**Reglas de entrada**	**Reglas de salida**
Back-security-group	sg-080b3035f29fb99c9	SSH:22 – 0.0.0.0/0 TCP Pers:6379– 0.0.0.0/0 (Redis) PostgreSQL TCP: 5432 – 0.0.0.0/0 TCP Pers:8080 – 0.0.0.0/0	Todo el tráfico

- **Descripción**: La instancia del worker se comunica con la base de datos PostgreSQL, la cola (Redis) donde va a consultar los videos encolados previamente por el backend y los procesará. También tendrá conexión con S3, allí obtendrá los videos de la carpeta “/uploads”, y una vez procesado el video, el worker deberá almacenarlo en la carpeta “/processed” del mismo bucket (anb-videos) y actualizar el estado del video en la base de datos.

Base de datos

- **Nombre de instancia de base de datos (en RDS):** database-anb
- **Tipo de instancia:** db.t4g.micro
- **Capacidad:** 2vCPU - 1G RAM – 200 GB de almacenamiento.
- **SubRed**:
    - **Tipo:** Privada
    - **Nombre:** anb-subnet-private1-us-east-1a
- **Nombre de base de datos:** dbanb
- **Motor de BD:** PostgreSQL 17.4
- **Grupo de seguridad:** Se creo un grupo de seguridad para la instancia RDS de la base de datos.

Nombre	AWS ID	Reglas de entrada	Reglas de salida
DB-security-group	sg-0fdc961799a912430	Todo el tráfico	Todo el tráfico

Aunque las reglas de entrada estan con todo el trafico, no existe un riesgo latente dado que la instancia de la base de datos esta en la subred privada.

- **Descripción:** Esta instancia contiene todas las tablas que conforma nuestro modelo de datos definido en la entrega anterior. Únicamente interactúa con el web server para guardar información o entregarla (para temas de autenticación, votos, rankings) y con el Worker para guardar información de videos procesados y entregar información de los videos que se subieron (uploaded).

Redes

- **Descripción**: Se creo una VPC para el proyecto en us-east-1 y dentro de ella se crearon 2 subredes: una publica (para el front) y otra privada (para el backend, base de datos y worker). Ambas en la misma zona de disponibilidad (us-east-1a). A continuación, se destacan atributos importantes de cada una:
    - VPC:
        - ID: vpc-07e55b2b1ab1ff090f4 (anb-vpc)
        - CIDR: 10.0.0.1/16 (65531 IP disponibles)
    - Subnet Pública:
        - ID: subnet-0e4991e6dfee69a4 (anb-subnet-public1-us-east-1a)
        - CIDR: 10.0.0.0/24 (251 IP disponibles)
        - Conexión de red: anb-igw (Internet Gateway)

    - Subnet Privada:
        - ID: subnet-09b01b4cd8e80354a (anb-subnet-private1-us-east-1a)
        - CIDR: 10.0.1.0/24 (251 IP disponibles)
        - Conexión de red: anb-nat-public1-us-east-1a (NAT Gateway).
- De manera general, no tiene configurados ACL especiales, sino los de por defecto que se asocian desde AWS.

Balanceador de carga

- **Nombre del balanceador de carga:** BackendELB
- **Tipo de LB:** Aplicación
- **Zonas de disponibilidad:** us-east-1a y us-east-1b.
- **Nombre DNS**: internal-BackendELB-810693461.us-east-1.elb.amazonaws.com
- **Descripción**: El balanceador fue configurado para equilibrar la carga de peticiones entre las instancias del backend cuando se llegue a un 60% de uso promedio de CPU y el grupo de autoescalado se active, en ese momento, tendrá sentido y utilidad el LB. Esto permitirá aumentar el número de peticiones que se pueden procesar de manera concurrente dando robustez y escalabilidad a nuestra aplicación.

Almacenamiento de objetos -S3

NFS (Bucket)

- **Nombre bucket**: anb-videos
- **Subcarpetas**: uploads/ y processed/
- **Subred**: anb-subnet-public1-us-east-1a
- **Descripción**: El bucket será el reemplazo del servidor NFS, permitiendo almacenar los videos procesados y sin procesar en las respectivas subcarpetas. El acceso a este bucket se hará desde el front para entregar en streaming los videos, desde la API para almacenar los videos sin procesar y desde el worker para descargar los videos y subir los procesados. Este cambio proporciona mejoras en la capacidad de almacenamiento, pero también en temas de seguridad de la información (encriptación del bucket) y en facilidad de acceso al recurso de S3.