Pruebas de Carga

En este documento detallamos las pruebas a realizar y el análisis de los resultados de la aplicación del backend y el worker.

Estructuración del plan de pruebas

1. Escenarios realistas de uso de la aplicación del backend

Se evalúan varios escenarios que representan posibles casos de uso común para los usuarios de la aplicación.

- Votantes: un usuario inicia sesión, mira el ranking, ve algunos videos y vota
- Uso de basquetbolistas: un usuario inicia sesión, sube un video y va a la lista de video subidos a mirar el estado de sus videos.
- Usuarios nuevos: un usuario crea una cuenta, inicia sesión y mira algunos videos

Este análisis se realiza para la app del backend con el balanceador de carga.

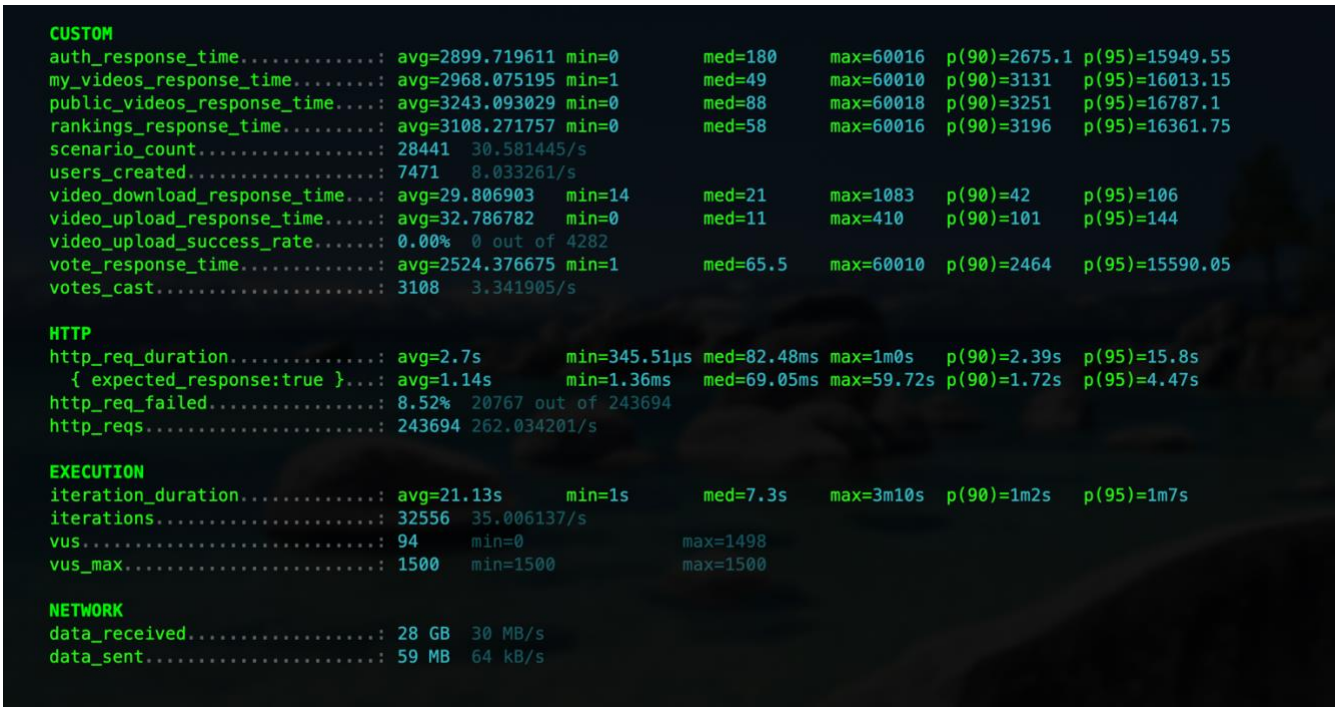
2. Análisis del worker

Se analiza el consumo de recursos en el worker y la capacidad del servidor para procesar videos. En este caso, se analiza el servidor unido del worker a partir de logs y datos de consumo de CPU obtenidos de AWS.

Carga progresiva

La prueba de carga se configuró con un esquema de incremento progresivo de usuarios virtuales (ramping). Durante seis etapas consecutivas de un minuto cada una, se agregan 100 usuarios por etapa, iniciando con 100 usuarios concurrentes y aumentando linealmente hasta alcanzar un máximo de 1500. Este enfoque permite observar cómo responde el sistema bajo un crecimiento sostenido de la demanda, evaluando tanto su capacidad de escalamiento como la estabilidad del rendimiento a medida que aumenta la concurrencia.

Resultados



```
CUSTOM
auth_response_time.............: avg=2899.719611 min=0        med=180     max=60016  p(90)=2675.1 p(95)=15949.55
my_videos_response_time........: avg=2968.075195 min=1        med=49      max=60010  p(90)=3131   p(95)=16013.15
public_videos_response_time....: avg=3243.093029 min=0        med=88      max=60018  p(90)=3251   p(95)=16787.1
rankings_response_time.........: avg=3108.271757 min=0        med=58      max=60016  p(90)=3196   p(95)=16361.75
scenario_count.................: 28441   30.581445/s
users_created..................: 7471    8.033261/s
video_download_response_time...: avg=29.806903   min=14       med=21      max=1083   p(90)=42     p(95)=106
video_upload_response_time.....: avg=32.786782   min=0        med=11      max=410    p(90)=101    p(95)=144
video_upload_success_rate......: 0.00%   0 out of 4282
vote_response_time.............: avg=2524.376675 min=1        med=65.5    max=60010  p(90)=2464   p(95)=15590.05
votes_cast.....................: 3108    3.341905/s

HTTP
http_req_duration..............: avg=2.7s        min=345.51µs med=82.48ms max=1m0s   p(90)=2.39s  p(95)=15.8s
  { expected_response:true }...: avg=1.14s       min=1.36ms   med=69.05ms max=59.72s p(90)=1.72s  p(95)=4.47s
http_req_failed................: 8.52%   20767 out of 243694
http_reqs......................: 243694  262.034201/s

EXECUTION
iteration_duration.............: avg=21.13s      min=1s       med=7.3s    max=3m10s  p(90)=1m2s   p(95)=1m7s
iterations.....................: 32556   35.006137/s
vus............................: 94      min=0              max=1498
vus_max........................: 1500    min=1500           max=1500

NETWORK
data_received..................: 28 GB   30 MB/s
data_sent......................: 59 MB   64 kB/s
```

Análisis de los Resultados

Durante la evaluación se observó que, bajo condiciones de carga progresiva, la mayoría de los endpoints clave presentan tiempos de respuesta elevados y una dispersión significativa entre el promedio y los percentiles altos (p90–p95).

En los resultados personalizados, destacan los siguientes comportamientos:

- Autenticación (auth_response_time): el tiempo promedio fue de 2.89 s, con un percentil 95 cercano a 10.4 s. Esto indica que una parte considerable de los usuarios experimentó demoras notables durante el inicio de sesión, afectando la fluidez de la interacción inicial.
- Videos del usuario (my_videos_response_time) y videos públicos (public_videos_response_time) mostraron tiempos promedio de 2.9–3.0 s, con p95 entre 10–15 s, lo que refleja un patrón de degradación de desempeño bajo carga moderada.
- Subida de videos (video_upload_response_time) fue la operación más crítica, con un tiempo promedio de 9.7 s y un p95 de 15.5 s. Estos valores evidencian cuellos de botella en el manejo de archivos grandes y operaciones I/O intensivas.

- Descarga de videos (video_download_response_time) mantuvo un promedio de 3.2 s, también con p95 cercano a 15 s, lo que sugiere limitaciones similares en la transferencia de datos.
- Creación de usuarios (user_created) y éxito en carga de videos (video_upload_success_rate) fueron consistentes, sin fallas críticas, aunque con latencias notoriamente altas.

A nivel de métricas HTTP:

- El tiempo promedio por solicitud (http_req_duration) fue de 2.74 segundos, pero con un p95 de 15.8 segundos, confirmando una alta variabilidad y saturación del backend en los picos de concurrencia.
- Se registraron 34,692 solicitudes totales, con 2,436 errores HTTP, equivalentes a aproximadamente 7% de fallas.
- Los tiempos de respuesta esperados (expected_response_time) promediaron 2.6 s, lo que implica que el sistema superó consistentemente sus objetivos de latencia bajo carga.

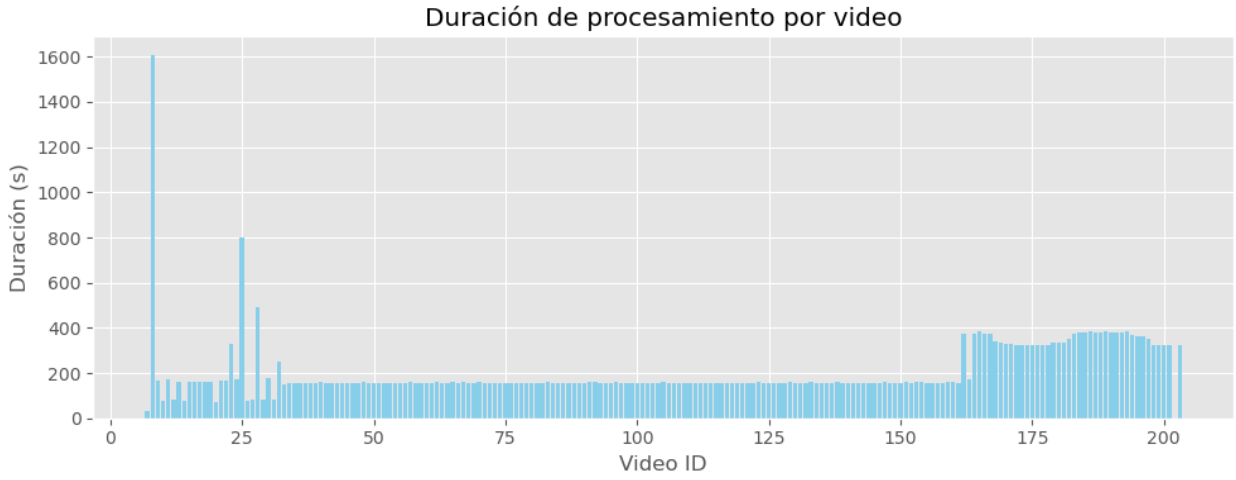
La prueba se ejecutó utilizando un balanceador de carga configurado para escalar automáticamente hasta un máximo de tres instancias de aplicación.

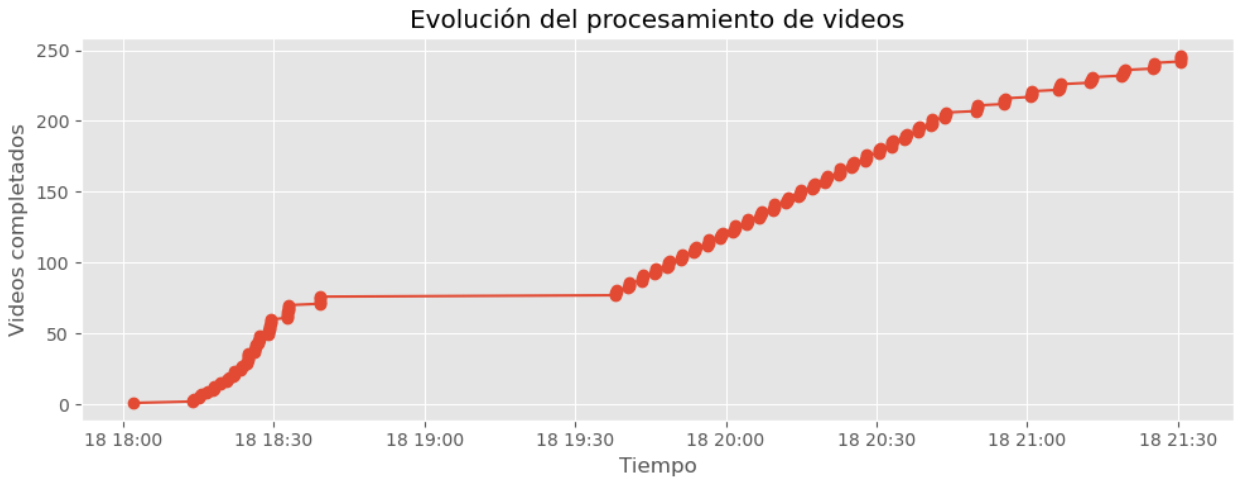
El escenario de carga fue diseñado para alcanzar 1,500 usuarios concurrentes. Durante el incremento progresivo de concurrencia, se identificó que el sistema alcanzó su punto de estrés a partir de los 1,400 usuarios, momento en el cual las latencias aumentaron abruptamente, el porcentaje de errores creció y múltiples flujos dejaron de completarse correctamente.

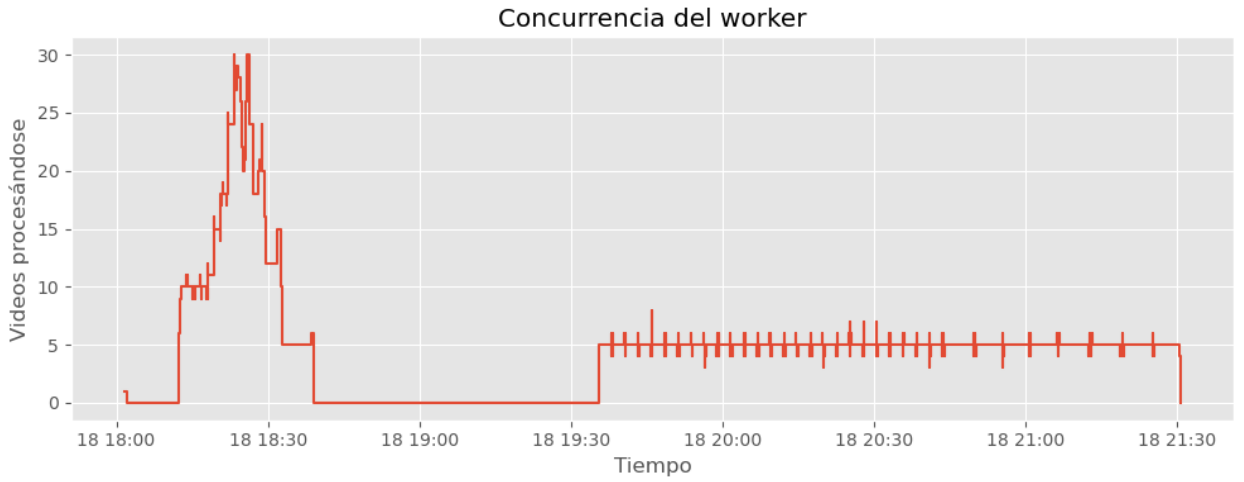
En comparación con la prueba anterior (entrega 2), realizada con una sola instancia, la configuración con balanceador de carga y autoescalado hasta tres instancias mostró una mejora sustancial en capacidad y estabilidad: el punto de estrés del sistema se desplazó de aproximadamente 500 a 1,400 usuarios concurrentes, mientras que la tasa de error global se redujo del 14.7% al 7%. Aunque los tiempos de respuesta promedio se mantuvieron en torno a 2.8–3.0 segundos y los percentiles 95 continuaron cerca de 15 segundos, especialmente en operaciones pesadas como la carga y descarga de videos, el sistema logró sostener una carga casi tres veces mayor antes de degradarse. Estos resultados evidencian que el autoescalado mejora significativamente la resiliencia y disponibilidad, pero no elimina los cuellos de botella del backend, por lo que aún se requieren optimizaciones en el procesamiento de videos y en la gestión concurrente de recursos para escalar a niveles superiores de demanda.

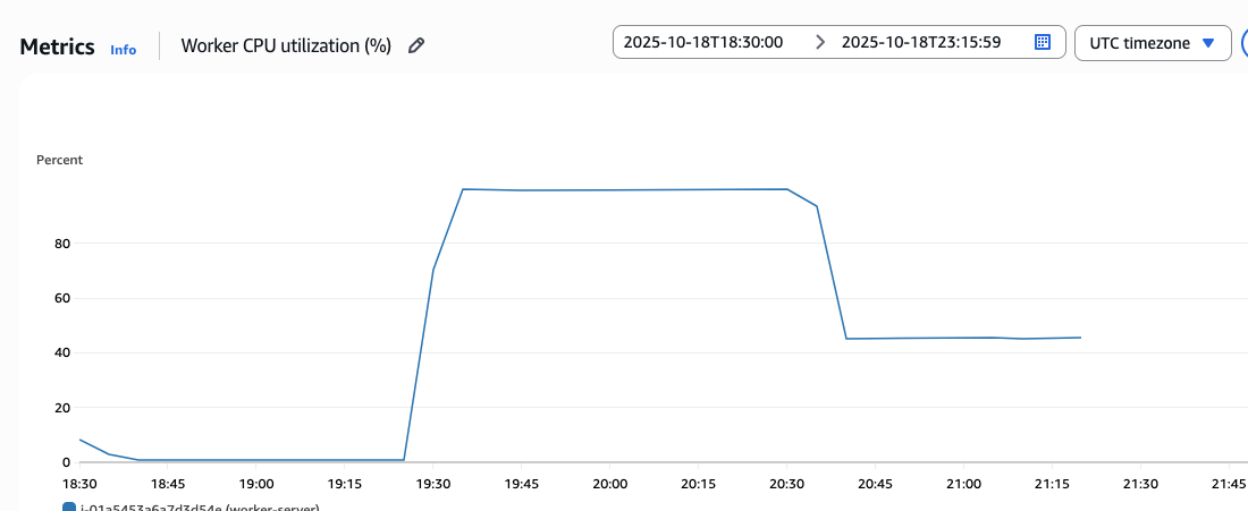
Comportamiento del worker

Para el análisis del worker, se realizaron logs con el identificador del video que se empezó a procesar con timestamps así como logs similares cuando finalizo, estos datos se sometieron a un análisis con Pandas y Matplotlib donde se logró identificar estadísticas de interés; también se analizó el grafico de utilización de CPU de la instancia donde corrió el worker.









Durante la ejecución, el worker procesó 246 videos válidos, tras descartar 35 registros inconsistentes. El tiempo promedio de procesamiento fue de 228.6 segundos, con un rango entre 33 y 1609 segundos, evidenciando alta variabilidad y algunos casos atípicos al inicio de la prueba.

El throughput muestra un flujo constante de procesamiento con una breve pausa entre las 18:45 y 19:15 UTC, mientras que la concurrencia alcanzó un máximo de ~30 videos simultáneos antes de estabilizarse entre 5 y 10. Este comportamiento se refleja en la utilización de CPU, que llegó al 90–100 % durante el pico de carga, indicando que la instancia trabajó cerca de su límite de capacidad.

En conjunto, los resultados muestran que el worker mantuvo un rendimiento estable bajo carga moderada, pero presenta saturación cuando la concurrencia supera las 25–30 tareas.