StreamX: Streaming Over HLS

Aaron González Araya
ITCR
Santa Clara, San Carlos
lamgonzalez@estudiantec.cr

Daniel Porras Narvaez
ITCR
Santa Clara, San Carlos
daporras@estudiantec.cr

***Abstract*— The StreamX system is designed to handle large volumes of simultaneous requests for multimedia content delivery using the HTTP Live Streaming (HLS) protocol. The architecture is based on a scalable structure that includes a load balancer, a Go server, and blob storage in PostgreSQL. The system allows users to stream video and audio without downloading entire files, offering a real-time experience similar to platforms like Netflix. Performance tests under adverse network conditions, such as packet loss and limited bandwidth, demonstrated the system's adaptive behavior, stabilizing over time despite initial spikes in blocked or failed requests. In optimal network environments, the system operates without failures, confirming its efficiency in handling multimedia content delivery.**

Index terms*—HTTP Live Streaming (HLS), StreamX, Multimedia content delivery, Scalability, Load balancing, PostgreSQL blob storage, Network performance, Real-time streaming*

I. Diseño Arquitectónico

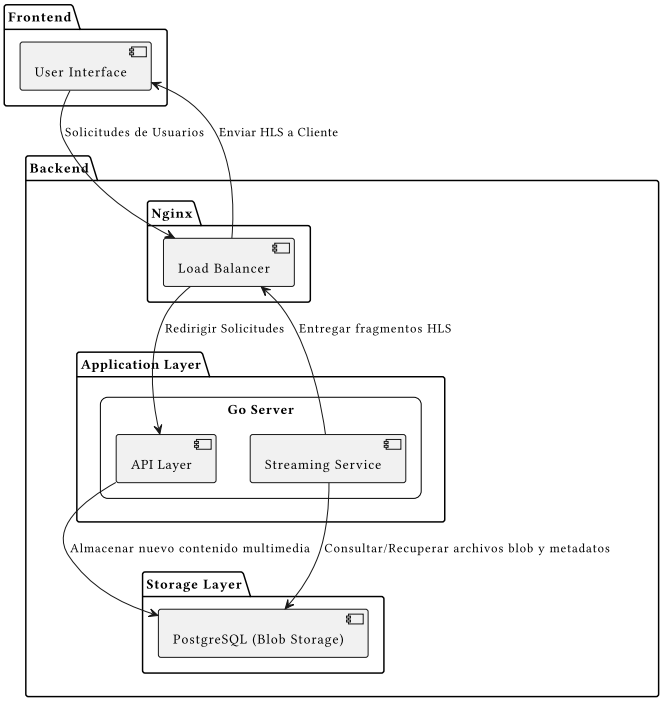

Fig. 1: System's Architectural Design

El diseño arquitectónico de StreamX se basa en una infraestructura escalable y eficiente para la entrega de contenido multimedia mediante el protocolo HTTP Live Streaming (HLS) [1]. El sistema está compuesto por varios componentes distribuidos que colaboran para asegurar una experiencia de usuario fluida y rápida, capaz de manejar múltiples solicitudes simultáneas.

El frontend de StreamX está constituido por una interfaz de usuario que puede ser accedida a través de navegadores web en plataformas de escritorio. Esta interfaz permite a los usuarios explorar el catálogo de contenido multimedia, como videos y música, y enviar solicitudes de reproducción. Las interacciones entre los usuarios y el sistema se realizan mediante solicitudes HTTP, que inician el proceso de entrega de contenido.

En el backend, Nginx [2] desempeña un papel crucial como balanceador de carga y proxy inverso. Su función es recibir las solicitudes HTTP de los usuarios y distribuirlas de manera eficiente entre las diferentes instancias del servidor backend implementado en Go. Este balanceo de carga permite que el sistema maneje grandes volúmenes de tráfico sin sobrecargar una sola instancia, lo que asegura la alta disponibilidad y escalabilidad de la plataforma.

El Servidor de Go [3] es el núcleo del backend y gestiona toda la lógica de negocio del sistema. Cuando una solicitud es redirigida desde Nginx, el servidor en Go se encarga de procesarla, lo que incluye la autenticación del usuario y la validación de la solicitud. A partir de este punto, el Servidor de Go consulta la base de datos PostgreSQL [4], que actúa como el sistema de almacenamiento principal para los archivos multimedia y los metadatos relacionados. PostgreSQL almacena los videos y las canciones como blobs[1] [5], permitiendo un acceso eficiente a grandes volúmenes de datos, mientras que los metadatos como títulos, autores y duración se gestionan a través de consultas SQL.
[1]

Una vez que el Servidor de Go obtiene el contenido multimedia desde PostgreSQL, utiliza el protocolo HLS para dividir los archivos en fragmentos pequeños, los cuales son entregados al cliente de manera escalonada. Esto permite a los usuarios comenzar a reproducir el contenido sin tener que descargar el archivo completo, lo que mejora la experiencia de transmisión en tiempo real. Los fragmentos de video o audio son enviados al usuario a través de Nginx, que se encarga de gestionar la conexión entre el servidor y el cliente final.

[1]Binary Large Object

El flujo completo del sistema comienza con una solicitud de reproducción por parte del usuario, que es recibida por Nginx y redirigida al Servidor de Go. El servidor procesa la solicitud, obtiene los datos necesarios desde PostgreSQL , y luego entrega los fragmentos HLS de vuelta al usuario para su reproducción. Este proceso asegura una experiencia fluida y de alta calidad, similar a la que ofrecen servicios como Netflix o Spotify.

II. Informe de Rendimiento

Los datos presentados en esta sección no representan métricas absolutas para una única conexión. En su lugar, son una amalgama de los promedios de los datos recopilados de 10 usuarios virtuales utilizados en las pruebas de rendimiento. Cada gráfico refleja el comportamiento promedio de estas múltiples conexiones simultáneas, lo que permite una visión más general del rendimiento del sistema bajo condiciones de carga representativas.

A. Usuario Subiendo un Video

Las pruebas relacionadas a un usuario subiendo un video fueron simuladas mediante el uso de `k6`, estas se simularon con un total de 10 usuarios virtuales en un periodo de 10 segundos.

1) *Peticiones Http Bloquadas:*

20% Pérdida de Paquetes:

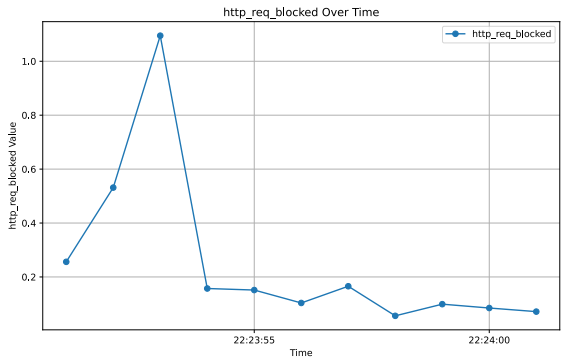

Fig. 2: Http Requests Blocked Over Time @ 20% Packet Loss

El gráfico proporcionado en la Fig. 2 muestra las métricas relacionadas con el bloqueo de solicitudes HTTP a lo largo del tiempo bajo una configuración de red con un 20% de pérdida de paquetes. En el eje X se representa el tiempo, mientras que el eje Y muestra la cantidad de solicitudes bloqueadas. A partir de la observación de este gráfico, es posible extraer varios aspectos clave sobre el comportamiento del sistema bajo estas condiciones adversas de red.

En la fase inicial del período de observación, se aprecia un aumento rápido en la cantidad de peticiones bloqueadas, alcanzando un máximo cercano a 1.0 en el eje Y. Este incremento inicial sugiere que, durante el inicio del período de prueba, hubo una alta cantidad de solicitudes HTTP bloqueadas, probablemente debido a la pérdida de paquetes que impide que las solicitudes lleguen al servidor de manera adecuada. Este comportamiento es típico en entornos de red con pérdida de paquetes, especialmente al inicio de una prueba de carga, donde el sistema aún no ha ajustado sus mecanismos para compensar dicha pérdida.

Posteriormente, el gráfico muestra una caída abrupta en la cantidad de solicitudes bloqueadas, la cual casi llega a 0 en un corto período de tiempo. Esta caída rápida sugiere que el sistema se ha recuperado o adaptado momentáneamente a las condiciones de la red, lo que reduce drásticamente el número de solicitudes bloqueadas. Esta respuesta puede deberse a diversos factores, como mecanismos automáticos de recuperación, retransmisiones de las solicitudes bloqueadas o ajustes internos que permiten mejorar temporalmente el procesamiento de las solicitudes a pesar de la pérdida de paquetes en la red.

Tras esta recuperación inicial, se observa una estabilización en la métrica de peticiones bloqueadas, con fluctuaciones pequeñas que se mantienen alrededor de valores bajos, aproximadamente entre 0.1 y 0.2. Esto indica que, aunque el sistema logra estabilizarse, no ha alcanzado un estado completamente óptimo, ya que las pequeñas variaciones sugieren que sigue lidiando con las condiciones adversas de la red, aunque en menor medida que al inicio del periodo.

500kbits de Ancho de Banda:

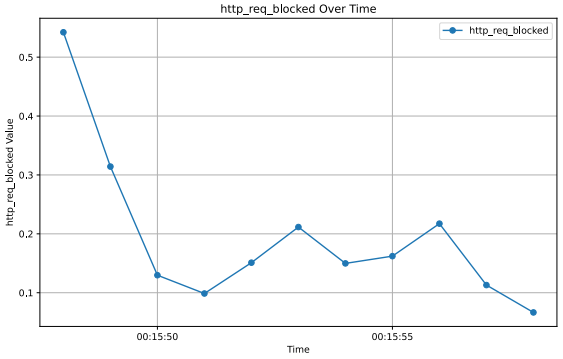

Fig. 3: Http Requests Blocked Over Time @ 500kbits bandwidth

El gráfico proporcionado en la Fig. 3 muestra la cantidad de solicitudes HTTP bloqueadas a lo largo del tiempo bajo una condición de red con un ancho de banda limitado a 500kbits. Este escenario simula una red con ancho de banda reducido, lo que afecta directamente el rendimiento de las solicitudes.

En el eje X se visualiza el tiempo, mientras que en el eje Y se muestra la cantidad de solicitudes HTTP bloqueadas. El análisis de este gráfico revela un comportamiento interesante del sistema bajo estas condiciones.

Inicialmente, el gráfico muestra un valor elevado de solicitudes bloqueadas, cercano a 0.5, lo que indica que, al comienzo de la prueba, la red limitada en ancho de banda resulta en una alta cantidad de solicitudes bloqueadas. Este comportamiento es esperable, dado que la baja velocidad de transmisión de

datos restringe la capacidad del sistema para procesar eficazmente las solicitudes en los primeros momentos de la prueba.

Después de este pico inicial, la cantidad de solicitudes bloqueadas disminuye gradualmente, alcanzando un valor más bajo cercano a 0.1. Este descenso refleja que el sistema se ajusta a las limitaciones de ancho de banda, posiblemente mediante la adaptación de los mecanismos de control de flujo o las políticas de retransmisión. A pesar de esta mejora, el sistema no alcanza una estabilidad total, como lo demuestran las fluctuaciones observadas en los datos posteriores.

El gráfico evidencia un patrón oscilante en la cantidad de solicitudes bloqueadas después de la caída inicial, con valores que varían entre 0.1 y 0.2 en diferentes momentos. Esto indica que el sistema sigue experimentando dificultades para manejar el tráfico de solicitudes bajo un ancho de banda restringido, aunque de forma menos severa que al inicio de la prueba. Las fluctuaciones sugieren que, aunque se logra un cierto nivel de adaptación, el sistema no consigue eliminar por completo los bloqueos debido a la falta de recursos de red.

Configuración de Red Óptima:

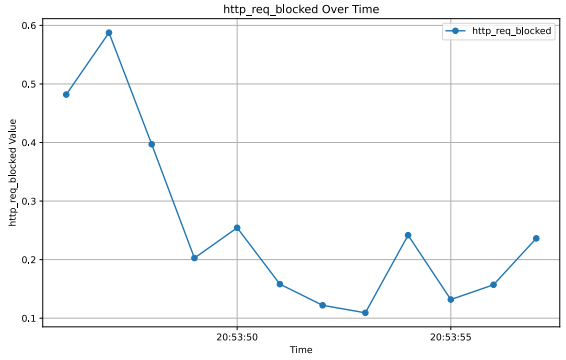

Fig. 4: Http Blocked Requests Over Time @ Optimal Settings

El gráfico proporcionado en la Fig. 4, que ilustra la cantidad de solicitudes HTTP bloqueadas bajo una configuración de red óptima, destaca varios aspectos importantes sobre el comportamiento del sistema. A diferencia de los escenarios anteriores con limitaciones de red, este gráfico refleja el desempeño del servidor en un entorno sin restricciones significativas. Sin embargo, los resultados permitieron identificar que los parámetros de configuración del servidor no eran adecuados para las demandas del sistema, en particular en relación con el mecanismo de rate limit que implementamos.

Desde el inicio de la prueba, se observa un pico en la cantidad de solicitudes bloqueadas, alcanzando un valor de 0.6. A pesar de la disminución rápida que se produce posteriormente, donde la cantidad de solicitudes bloqueadas cae por debajo de 0.2, se siguen observando fluctuaciones menores en los valores a lo largo del tiempo. Estas oscilaciones, con variaciones que se mueven entre 0.1 y 0.3 en diferentes puntos, sugieren que el sistema aún enfrenta bloqueos de solicitudes, incluso bajo condiciones de red ideales.

Al revisar la configuración del servidor, notamos que habíamos implementado un mecanismo de rate limit para controlar el tráfico de solicitudes. Este mecanismo fue configurado para limitar el número de solicitudes que pueden procesarse en un periodo de tiempo determinado, con el objetivo de evitar la sobrecarga del servidor. Sin embargo, las pruebas revelaron que los parámetros de configuración del rate limit no eran óptimos para la demanda real que implica servir un solo video en condiciones de alta carga.

El límite configurado inicialmente resultó ser demasiado bajo para satisfacer las exigencias del sistema de streaming. Esto provocó que, incluso en un entorno con condiciones de red óptimas, se bloquearan solicitudes que el servidor podría haber manejado si el límite hubiera sido más alto. Estas pruebas nos permitieron identificar la necesidad de ajustar los parámetros del rate limit, para que el sistema pueda procesar un mayor volumen de solicitudes sin que se produzcan bloqueos innecesarios, especialmente al atender a varios usuarios simultáneos que acceden al contenido multimedia.

2) *Datos Enviados:*

500kbits de ancho de banda:

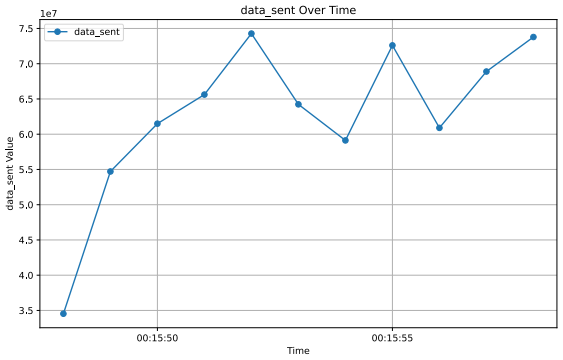

Fig. 5: Data Sent Over Time @ 500kbits bandwidth

El gráfico mostrado en la Fig. 5 muestra la cantidad de datos enviados durante la subida de un video bajo una restricción de ancho de banda de 500kbits. En el eje X se representa el tiempo, mientras que en el eje Y se muestra la cantidad de datos enviados, con un valor total en el rango de 10^7 bytes.

El comportamiento del sistema bajo esta restricción de ancho de banda revela varios patrones importantes sobre la forma en que el servidor gestiona la transmisión de datos cuando el ancho de banda es limitado.

Al inicio del gráfico, la cantidad de datos enviados comienza en un valor cercano a 3.5×10^7 bytes y crece rápidamente, alcanzando un pico de aproximadamente 7.5×10^7 bytes hacia la mitad del período observado. Este comportamiento refleja que, a pesar de las limitaciones impuestas por el ancho de banda, el sistema puede mantener un ritmo de subida relativamente constante en los primeros momentos del proceso de subida del video.

Sin embargo, tras alcanzar este pico, el gráfico muestra una serie de fluctuaciones, con caídas y picos en la cantidad de datos enviados a lo largo del tiempo. Estas oscilaciones indican que el sistema no puede mantener un flujo de datos constante debido a las restricciones de ancho de banda. Es probable que, en los momentos donde se observan descensos en la cantidad de datos enviados, el ancho de banda limitado esté impactando la capacidad del servidor para continuar transmitiendo datos de manera estable.

Estas fluctuaciones podrían estar relacionadas con la forma en que el mecanismo de control de flujo del servidor interactúa con la limitación de ancho de banda. El servidor intenta enviar tantos datos como sea posible en los momentos de disponibilidad de la red, pero la restricción del ancho de banda hace que este ritmo no sea uniforme, generando momentos donde la velocidad de subida disminuye significativamente.

20% Pérdida de Paquetes:

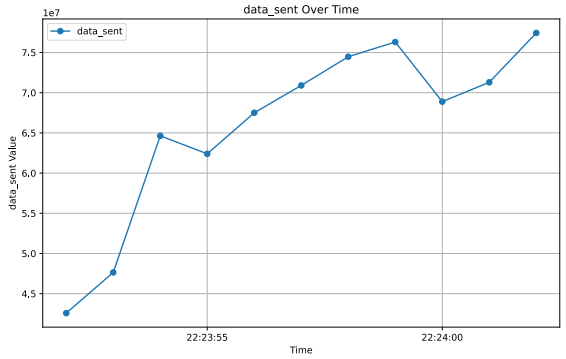

Fig. 6: Data Sent Over Time @ 20% Packet Loss

El gráfico que proporcionamos en la Fig. 6 muestra el comportamiento del sistema en términos de datos enviados durante la subida de un video, esta vez bajo una condición de red con 20% de pérdida de paquetes (packet loss). A diferencia del gráfico anterior, que estaba limitado por un ancho de banda de 500kbits, aquí el impacto en la transmisión de datos es causado por la pérdida de paquetes en lugar de un ancho de banda reducido.

En el eje X se representa el tiempo, mientras que el eje Y muestra la cantidad de datos enviados, con valores en el rango de 10^7 bytes.

Al comienzo del gráfico, se observa un crecimiento relativamente constante en la cantidad de datos enviados, que pasa de 4.5×10^7 bytes a un máximo cercano a 7.5×10^7 bytes. Esta tendencia indica que, incluso bajo condiciones de pérdida de paquetes, el servidor es capaz de enviar datos de manera eficiente durante los primeros momentos de la transmisión. Sin embargo, la pérdida de paquetes empieza a afectar el rendimiento posteriormente.

Una diferencia clave respecto al gráfico anterior es que, aunque hay un patrón de aumento en los datos enviados, se observan picos y caídas más marcadas en el volumen de datos transmitidos. Después de alcanzar un máximo alrededor de los 7.5×10^7 bytes, el gráfico muestra una caída visible, seguida de una recuperación parcial. Esta fluctuación refleja cómo la pérdida de paquetes interrumpe la transmisión de datos, lo que obliga al servidor a realizar retransmisiones o ajustes para compensar los datos que no llegaron correctamente al destinatario.

A pesar de estas interrupciones, el sistema logra recuperar su ritmo de envío en varios puntos, lo que indica que, aunque el 20% de packet loss impacta el flujo de datos, el servidor implementa mecanismos que le permiten mitigar parcialmente este efecto. No obstante, a lo largo del tiempo, las fluctuaciones son evidentes y muestran que la pérdida de paquetes tiene un impacto más dinámico en comparación con el ancho de banda limitado, lo que genera caídas repentinas en la cantidad de datos enviados.

Configuración de Red Óptima:

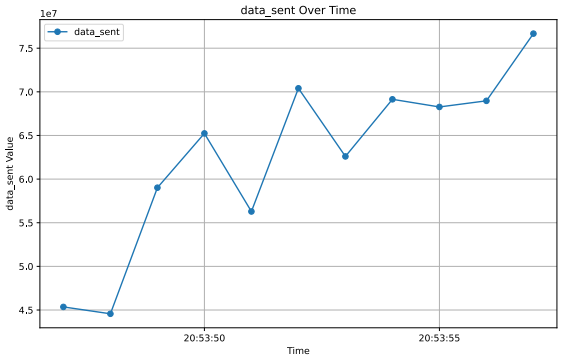

Fig. 7: Data Sent Over Time @ Optimal Settings

El gráfico presentado en la Fig. 7 muestra la cantidad de datos enviados durante la subida de un video bajo una configuración de red óptima, es decir, sin restricciones significativas como pérdida de paquetes o limitaciones de ancho de banda. En el eje X se representa el tiempo, mientras que el eje Y muestra la cantidad de datos enviados, con valores en el rango de 10^7 bytes.

El comportamiento observado en esta configuración refleja un flujo de datos fluido y relativamente estable. Desde el inicio, se aprecia un crecimiento continuo en la cantidad de datos enviados, comenzando desde 4.5×10^7 bytes y subiendo progresivamente hacia 7.5×10^7 bytes. A lo largo del tiempo, el gráfico muestra varios picos y caídas, aunque las fluctuaciones son leves y no interrumpen significativamente el flujo de datos.

El sistema parece funcionar de manera eficiente, ya que las caídas observadas no son pronunciadas y no afectan el rendimiento de manera importante. A pesar de algunas oscilaciones, el flujo general de datos es sostenido, lo que indica que el servidor está gestionando bien la transmisión de grandes volúmenes de datos sin interrupciones evidentes. Los picos hacia el final del gráfico muestran que el sistema continúa enviando

datos a una velocidad consistente y alta, lo que es indicativo de un entorno de red saludable.

3) *Peticiones Http Fallidas:*

20% Pérdida de Paquetes:

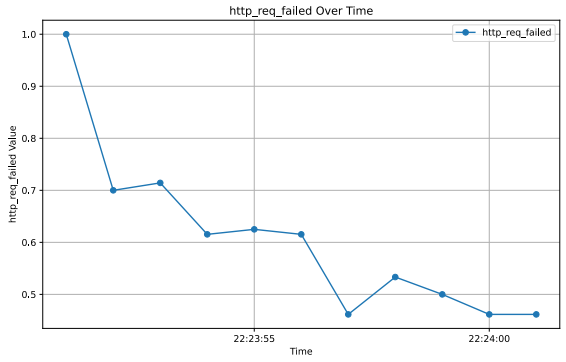

Fig. 8: Failed Http Requests Over Time @ 20% Packet Loss

El gráfico de la Fig. 8 muestra la evolución de las conexiones fallidas a lo largo del tiempo en un entorno donde existe una restricción de 20% de pérdida de paquetes (packet loss). En el eje X se presenta el tiempo, mientras que en el eje Y se representan los valores de las solicitudes HTTP que fallaron. Al observar la tendencia del gráfico, se puede apreciar cómo el sistema se comporta en condiciones adversas de red, con una alta tasa de pérdida de paquetes.

Inicialmente, se observa un pico cercano a 1.0, lo que indica que, al comienzo del período de prueba, la mayoría de las solicitudes estaban fallando. Este comportamiento es típico al inicio de un entorno de red con alta pérdida de paquetes, donde muchas de las solicitudes no logran completarse exitosamente debido a la inestabilidad de la red.

A medida que avanza el tiempo, se evidencia una tendencia descendente en el número de solicitudes fallidas. El gráfico muestra una serie de caídas sucesivas, con algunos pequeños picos de recuperación intercalados, pero en general, la cantidad de solicitudes que fallan va disminuyendo progresivamente. Esta reducción puede estar relacionada con la capacidad del sistema para adaptarse y gestionar las retransmisiones de paquetes perdidos, o bien porque algunas conexiones logran completarse a pesar de la pérdida de paquetes, mejorando los resultados a lo largo del tiempo.

En las fases finales del gráfico, los valores caen por debajo de 0.5, lo que indica que menos del 50% de las solicitudes están fallando al final de la prueba. Aunque esta mejora es notable, sigue reflejando un entorno de red con importantes limitaciones, ya que un 20% de pérdida de paquetes sigue afectando negativamente la eficiencia del sistema.

500kbits de Ancho de Banda:

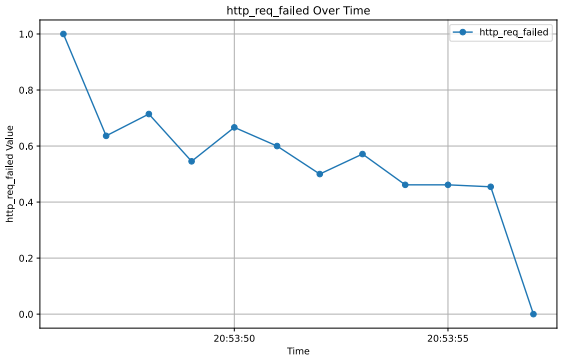

Fig. 9: Failed Http Requests Over Time @ 500kbits bandwidth

El gráfico proporcionado en la Fig. 9 refleja la cantidad de solicitudes HTTP fallidas a lo largo del tiempo en un entorno donde la restricción es un ancho de banda limitado a 500kbits. El eje X representa el tiempo, mientras que el eje Y muestra el valor de las solicitudes que fallaron.

Al inicio del gráfico, se observa un valor elevado cercano a 0.9, lo que indica que al comienzo del período de prueba, una gran cantidad de solicitudes estaban fallando debido a la restricción en el ancho de banda. Esta alta tasa de fallos es típica en escenarios donde el servidor se ve limitado en su capacidad para procesar y enviar datos a una velocidad suficiente para manejar la demanda.

A medida que la prueba avanza, el gráfico muestra una caída significativa en las solicitudes fallidas, llegando a aproximadamente 0.5. Sin embargo, tras esta mejora inicial, el sistema parece estabilizarse con fluctuaciones alrededor de 0.5 y 0.7 en el valor de solicitudes fallidas. Estas fluctuaciones indican que, aunque el sistema ha reducido las fallas iniciales, el ancho de banda limitado sigue afectando el rendimiento y una parte considerable de las solicitudes continúa fallando.

Cerca del final del gráfico, se observa una caída pronunciada en las solicitudes fallidas, descendiendo hasta 0.2. Esto sugiere que, con el paso del tiempo, el sistema puede haber ajustado el manejo de las solicitudes o reducido la carga de red, lo que permite una mayor cantidad de solicitudes exitosas a pesar de las limitaciones de ancho de banda.

Configuración de Red Óptima:

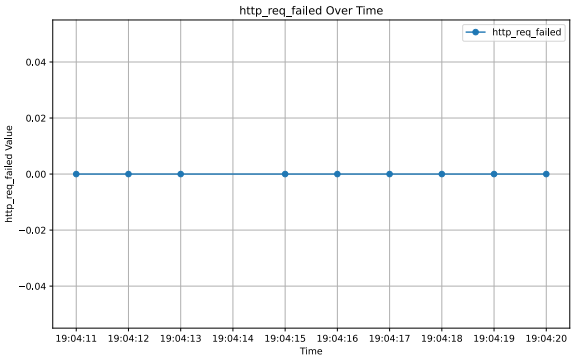

Fig. 10: Failed Http Requests Over Time @ Optimal Settings

El gráfico de la Fig. 10 muestra la cantidad de solicitudes HTTP fallidas a lo largo del tiempo en un entorno con configuración de red óptima, es decir, sin restricciones significativas como limitaciones de ancho de banda o pérdida de paquetes. En el eje X se representa el tiempo, mientras que en el eje Y se observa el valor de las solicitudes fallidas.

En este gráfico, los valores se mantienen consistentemente en 0 durante todo el período de tiempo analizado. Esto significa que no se registraron solicitudes HTTP fallidas bajo esta configuración de red, lo que es indicativo de un entorno de red completamente estable y capaz de manejar la demanda de tráfico sin que ocurra ningún fallo en las solicitudes.

La ausencia total de fallos en las solicitudes HTTP bajo una red óptima refleja que el sistema es capaz de gestionar las conexiones de manera eficiente cuando no hay limitaciones impuestas por factores externos, como ancho de banda limitado o pérdida de paquetes. Esto demuestra que el sistema está bien optimizado para operar en condiciones ideales, garantizando que todas las solicitudes lleguen a su destino sin interrupciones.

III. Conclusiones

El sistema StreamX está diseñado para manejar grandes volúmenes de solicitudes simultáneas y entregar contenido multimedia de manera eficiente utilizando el protocolo HLS. La arquitectura se basa en una estructura escalable con un balanceador de carga, un servidor implementado en Go y almacenamiento de blobs en PostgreSQL. El sistema permite a los usuarios reproducir videos sin necesidad de descargarlos completamente, proporcionando una experiencia en tiempo real similar a plataformas como Netflix.

En cuanto al rendimiento, las pruebas bajo condiciones de red adversas como pérdida de paquetes y limitaciones de ancho de banda revelaron un comportamiento adaptativo del sistema. A pesar de los picos iniciales de solicitudes bloqueadas o fallidas, el sistema logró estabilizarse, aunque no alcanzó un rendimiento completamente óptimo bajo estas condiciones. En un entorno ideal, el sistema no mostró fallos en las solicitudes, lo que confirma que StreamX funciona de manera eficiente cuando no hay limitaciones significativas en la red.

References

[1] R. Pantos and W. May, “HTTP live streaming,” 2017.

[2] R. Soni, *Nginx*. Springer, 2016.

[3] M. Tsoukalos, *Mastering Go: Create Golang production applications using network libraries, concurrency, machine learning, and advanced data structures*. Packt Publishing Ltd, 2019.

[4] K. Douglas and S. Douglas, *PostgreSQL: a comprehensive guide to building, programming, and administering PostgresSQL databases*. SAMS publishing, 2003.

[5] M. Shapiro and E. Miller, “Managing databases with binary large objects,” in *16th IEEE Symposium on Mass Storage Systems in cooperation with the 7th NASA Goddard Conference on Mass Storage Systems and Technologies (Cat. No. 99CB37098)*, 1999, pp. 185–193.