**Caso de Estudio 2 – Análisis de Desempeño**

**Logística y Seguridad Aeroportuaria**

Felipe Guzmán Avendaño – 201813791 y Nicolás Munar González - 201814383

**1. Monitores**

Para el caso del cliente (tanto con o sin seguridad), se automatiza la creación de clientes para dar el mensaje HOLA y hacer aleatorios los envíos de algoritmos para el servidor, para que la carga de clientes generada fuera automática. Se usan dos generadores de carga, uno que llama la clase cliente con seguridad y otro sin seguridad según sea el caso.

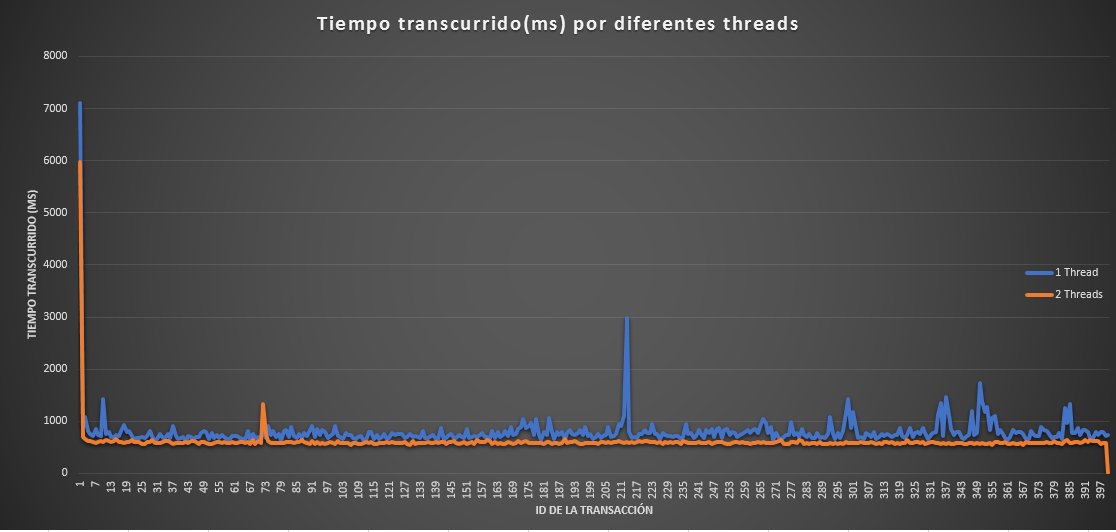
Para el caso del servidor (tanto con o sin seguridad), se agrega el método de obtener el uso de CPU, y se toman en varios puntos durante el proceso para tener un aproximado, principalmente al terminar y comenzar las distintas fases del servidor. Para determinar el tiempo, se inicia antes de recibir el certificado del cliente y finaliza al recibir el mensaje final del cliente, como fue indicado en el proyecto. Se muestran los resultados de uso de CPU en promedio durante el proceso, y el tiempo en el que se tarda en completar la operación para un cliente.

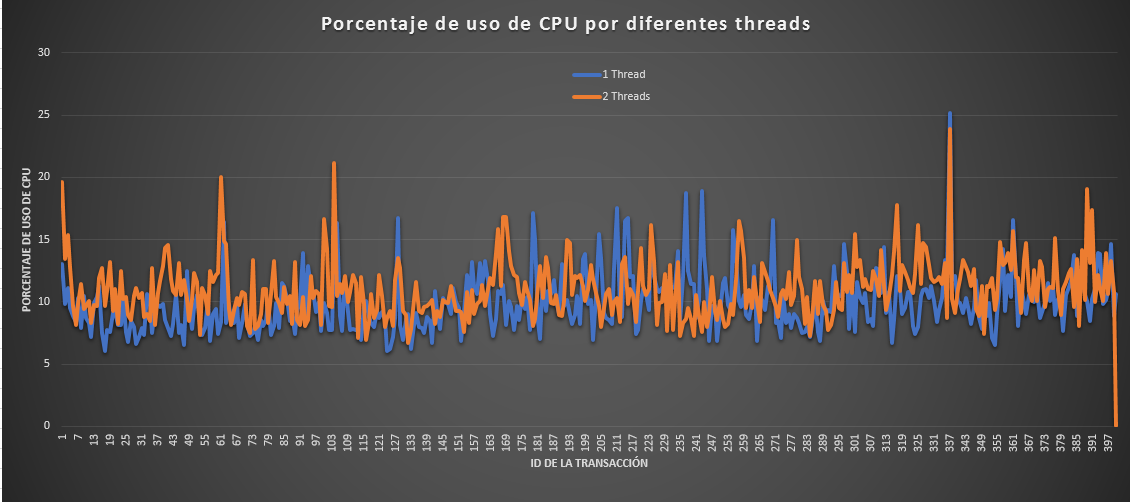
**2. Plataforma**

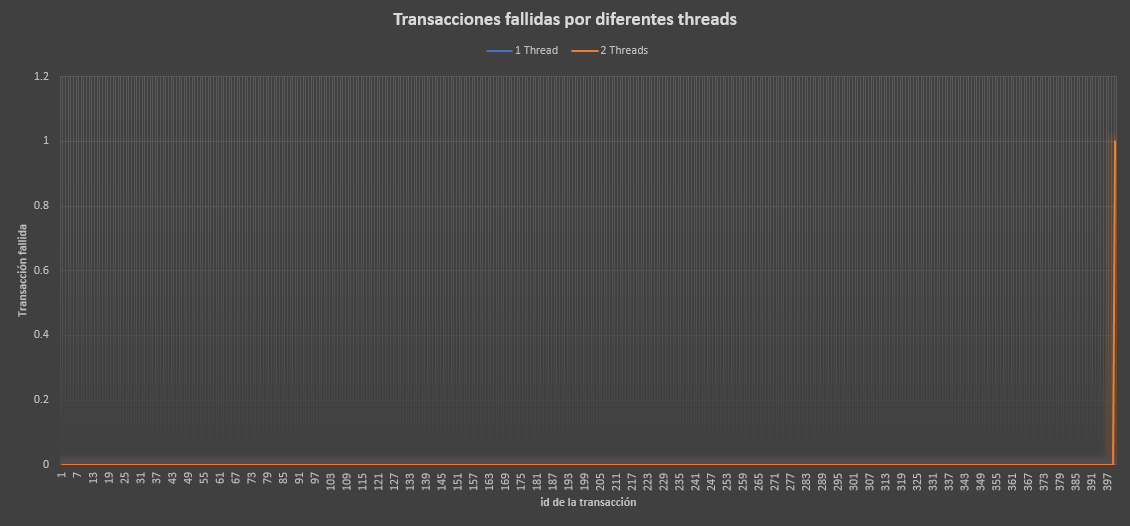
Se usa la máquina virtual proveída, la cual tiene una arquitectura de 64 bits, 2 núcleos, una velocidad del procesador de 2.7GHz, cuenta con 4GB de RAM y la memoria asignada a la JVM es de 910.50MB.

**3. Concurrencia**

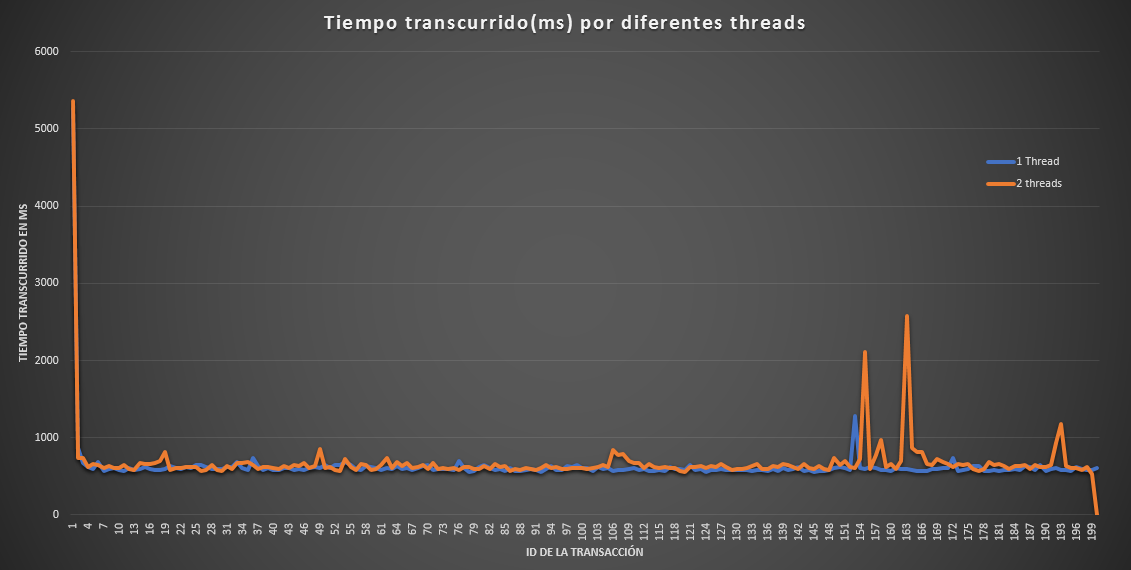
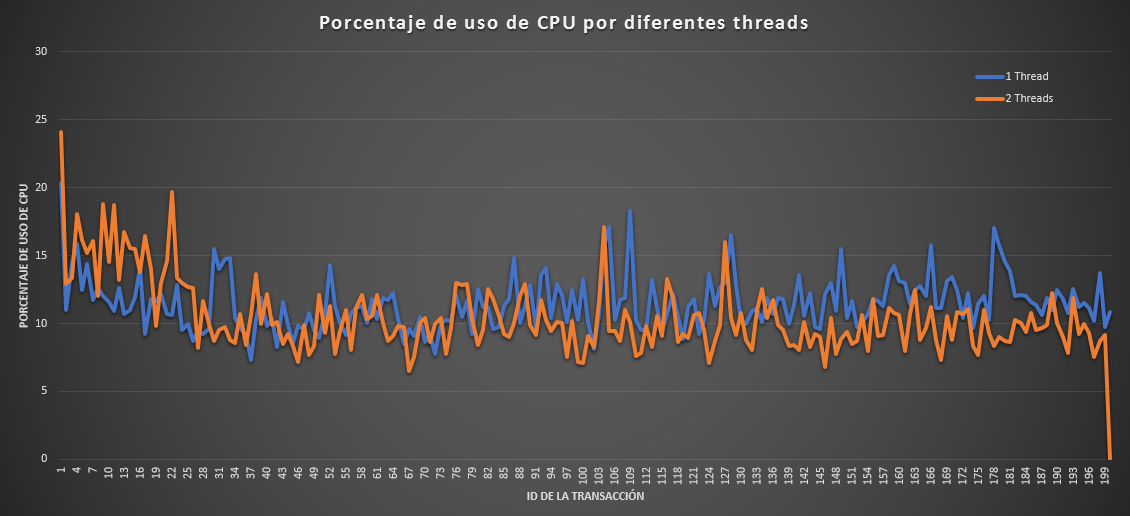
**Primer escenario:** 400 transacciones, retardos de 20ms

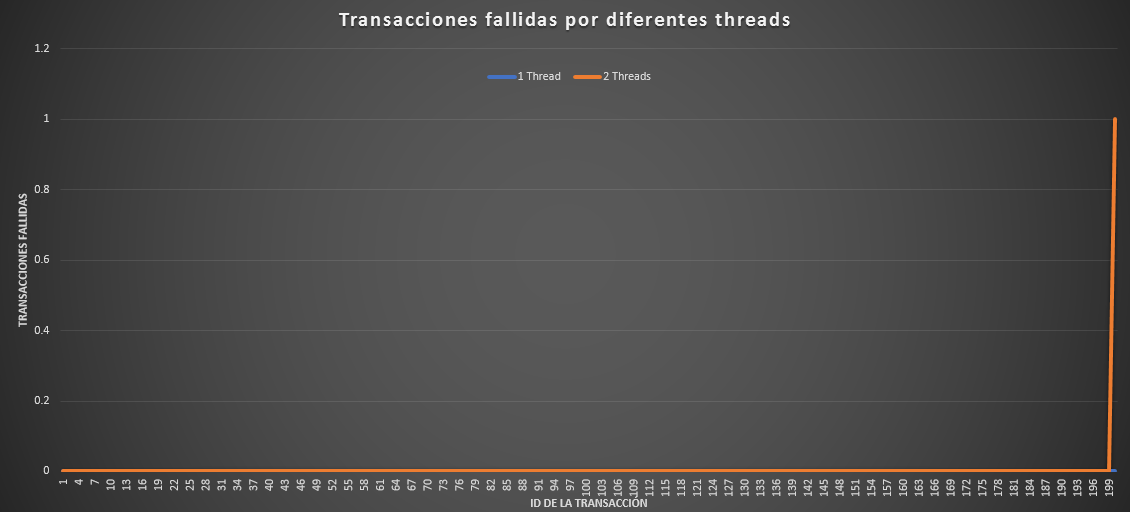


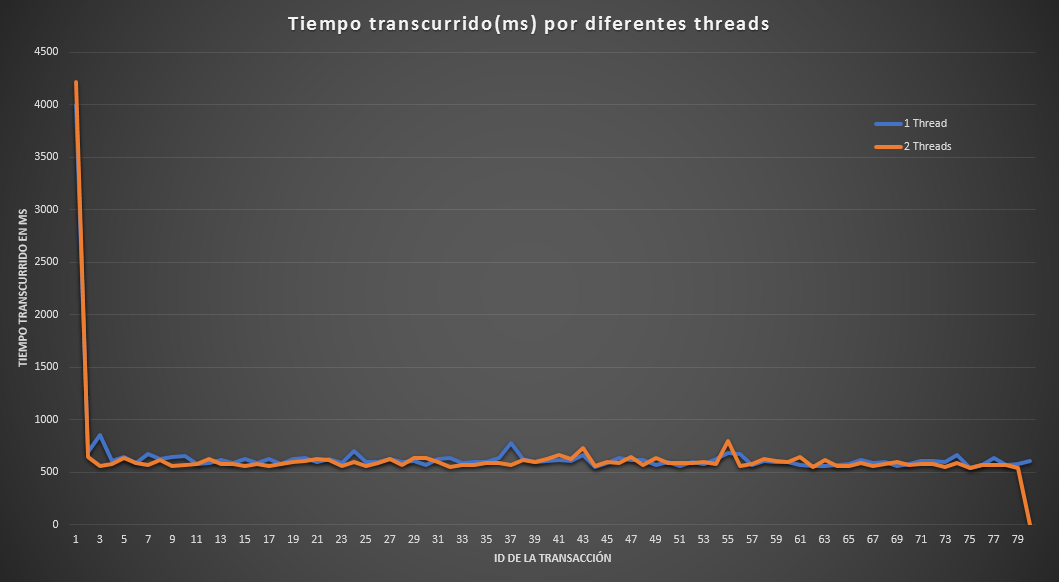


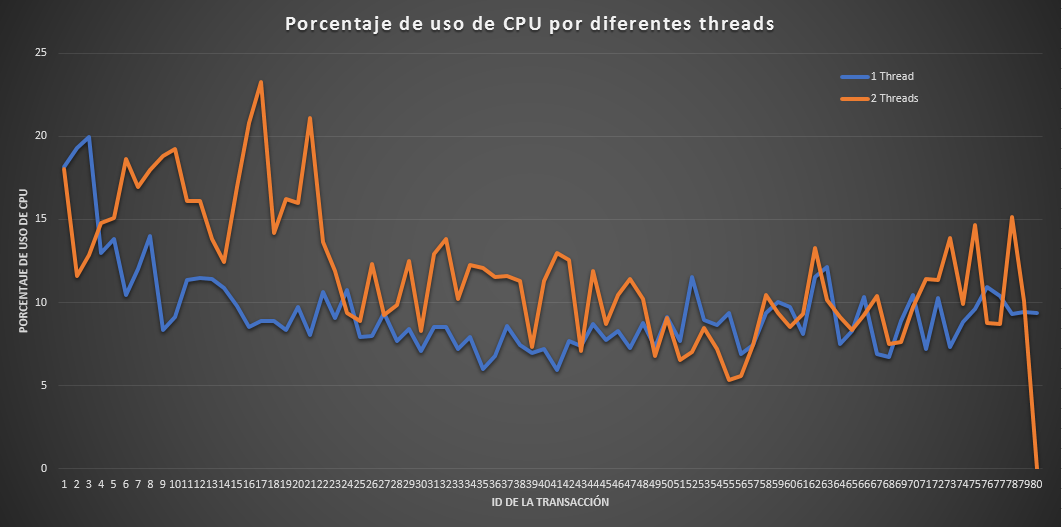
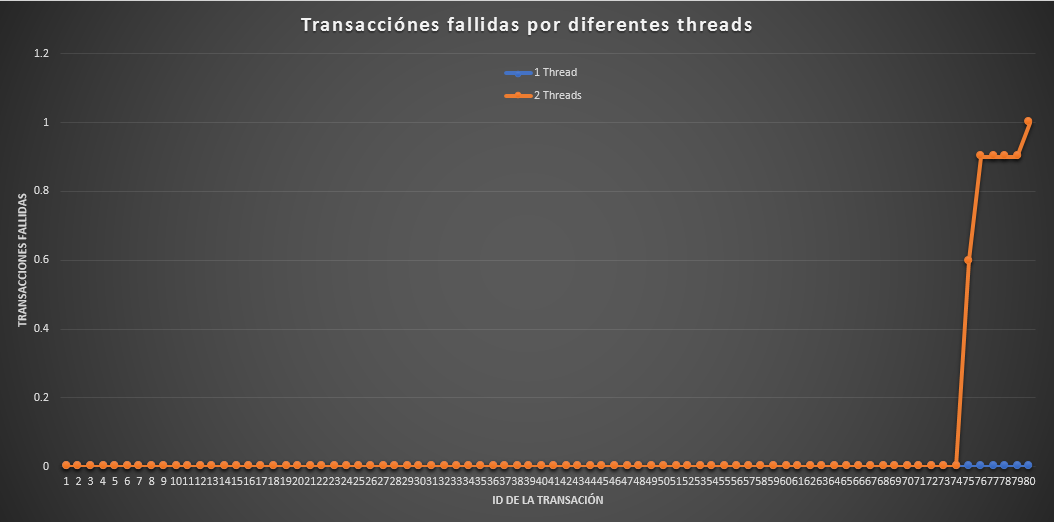


Para el primer escenario, se nota un comportamiento similar en las 400 transacciones (promediadas), con 1 o 2 threads. Para dos threads se usa más CPU que con 1 thread aunque no por mucho, sin embargo, se demora menos tiempo en completar la transacción cuando se usan 2 threads. Igualmente solo se presenta una transacción perdida cuando se usan 2 threads. Ambos presentan un pico de tiempo en completar y de uso de CPU al iniciar, en la primera transacción.

**Segundo escenario:** 200 transacciones, 40ms de retardo. 

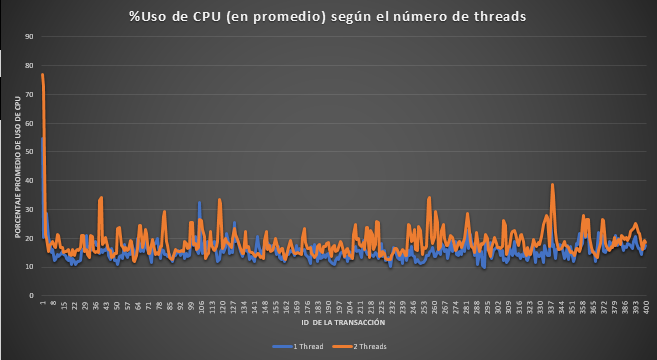
Para el segundo escenario, tanto el uso de CPU como el tiempo transcurrido para completar la transacción, son similares, siendo en promedio mayor el tiempo transcurrido para 2 threads, mientras que para un thread se usa más CPU, esto debido a unos picos que tienen en sus gráficas correspondientes. Para las transacciones fallidas, se presenta una cuando se usan 2 threads. Igualmente se presentan picos al inicio de las transacciones.

 **Tercer escenario:** 80 transacciones, 100ms de retardo.



Para el tercer escenario, al usar 2 threads se tienen más picos de porcentaje de uso de CPU, y por ello el uso es mayor que usar 1 thread. El tiempo transcurrido es similar en ambos casos, con una demora promedio de 640 ms para completar la transacción. Sobre transacciones fallidas, se presentan más con 2 threads, específicamente al final que se presentan más fallos.

En general, el comportamiento del servidor se mantiene similar en los 3 escenarios, manteniendo 10.5% de uso de CPU en promedio sin importar los threads o la carga. Para el tiempo transcurrido se mantiene también relativamente constante en 647.69ms, excepto para el primer escenario con 1 thread, el cual tuvo un tiempo transcurrido promedio cercano a 800ms. Para los 3 escenarios hubo una mayor carga al inicio de las transacciones, e igualmente para 2 threads había fallos al finalizar las transacciones, y además se tiene una mayor cantidad de picos en el uso de CPU para estos casos con 2 threads.

**4. Sin Seguridad**

De la gráfica se puede observar que al usar 2 threads se tienen más picos de uso de CPU, casi periódicamente, aunque para 1 thread se tienen picos no son tan pronunciados. Al inicio de ambas transacciones se tiene la mayor cantidad de uso de CPU. En general 1 thread no usa tanta CPU como si se implementaran 2 threads.

Se espera que al no usar seguridad, se use menos CPU que cuando tenga seguridad, pues no se tiene que cifrar/descifrar los mensajes, son más rápidas las transacciones, y estas son solo recibir y enviar mensajes correspondientes.

Sin embargo, se tiene un mayor uso de CPU, el promedio de estas pruebas para 2 threads es de 15.68% y 1 thread es 18.57%, mientras que para el servidor con seguridad, se tiene 9.92% de uso de CPU para 1 thread y 10.98% para dos threads. Esto puede ocurrir por el momento en el que se toman las medidas de uso de CPU, es decir que no ocurran durante un cifrado/descifrado de mensajes.