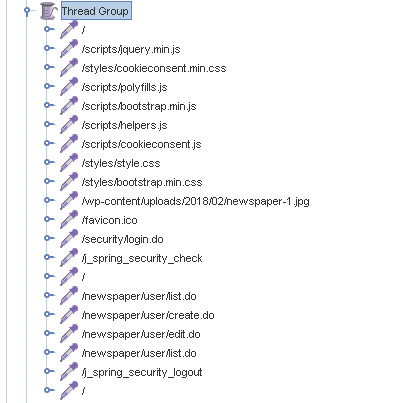
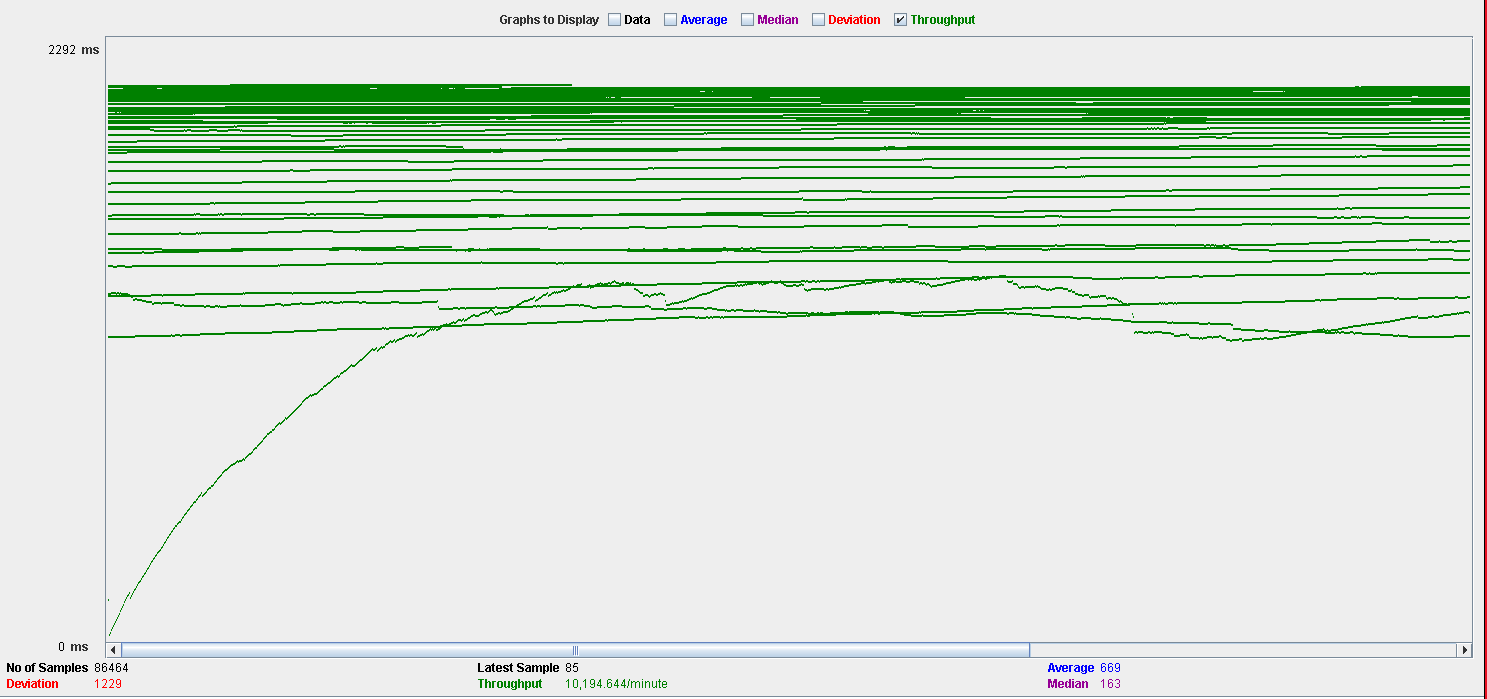
Test de rendimiento de Crear Newspaper

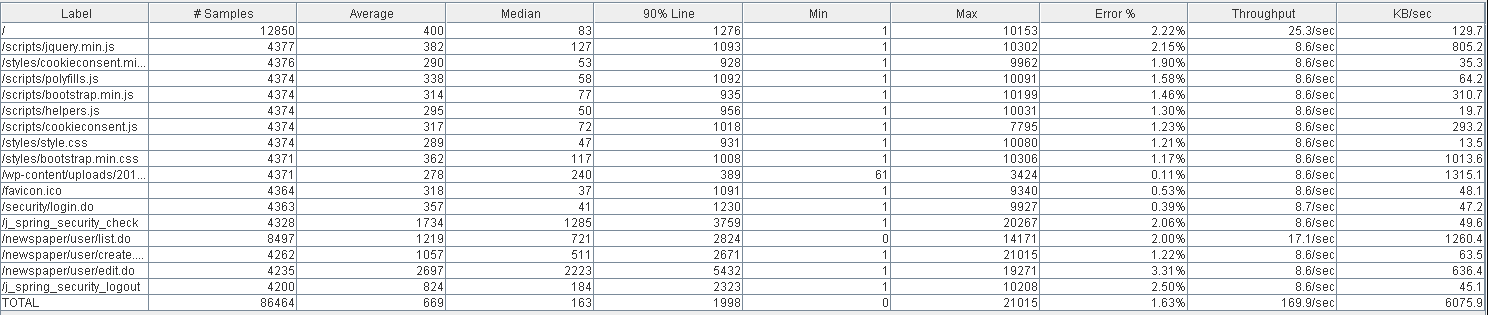
Realizamos una prueba estableciendo como número de usuarios simultáneo 200 personas con un retraso entre usuario y usuario de medio segundo. Con esta configuración tras ejecutar el siguiente script:



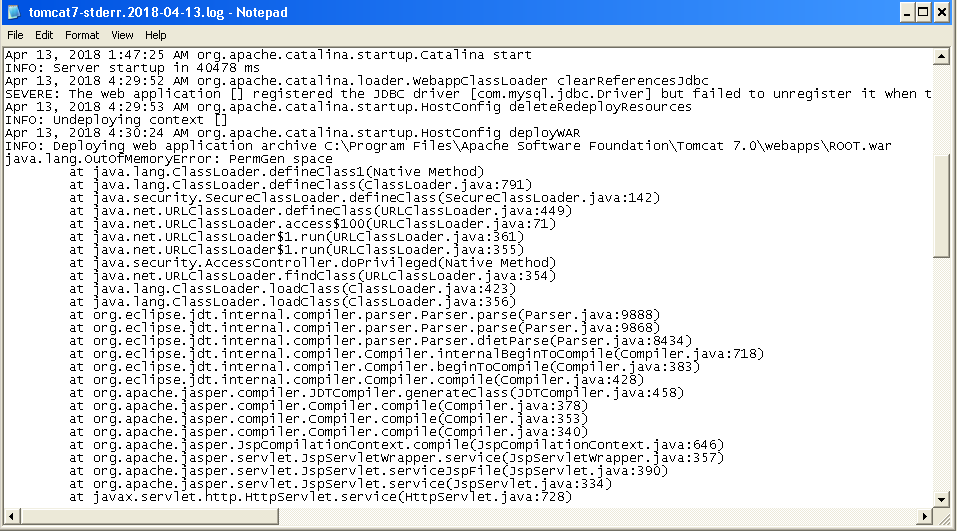
Nos logeamos como user, nos vamos a nuestros newspapers, creamos uno y nos deslogeamos,

Tras tener esta secuencia de pasos ejecutamos los tests y se nos genera las siguientes gráficas.

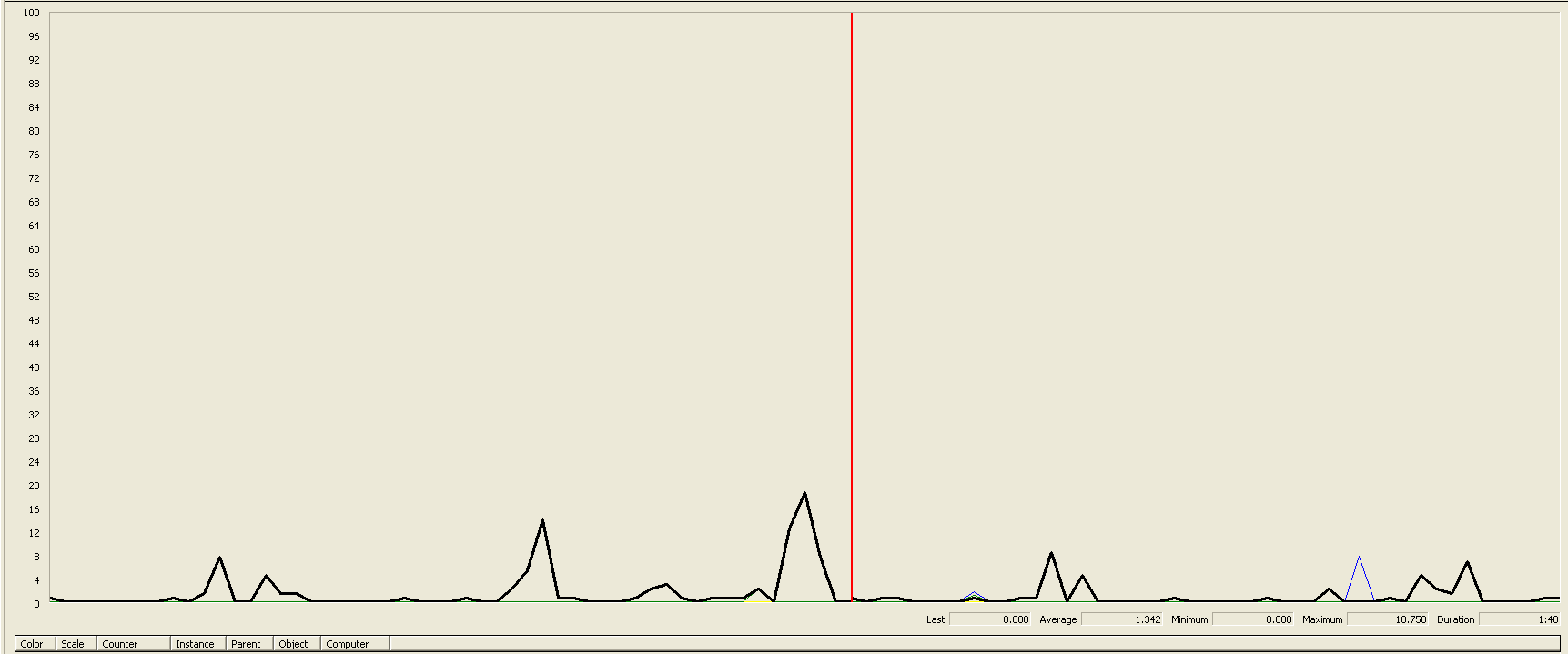


Y se nos genera el siguiente informe:

Como vemos se nos han generado errores pequeños con la configuración escogida por lo que el caso de uso de crear un newspaper es soportado de manera estable por un máximo de 200 personas. El causante del error que ha provocado esto ha sido el procesador de mi máquina. He llegado a esta conclusión ya que he mirado en los archivos de log y decían que el error era por un OutOfMemoryError.

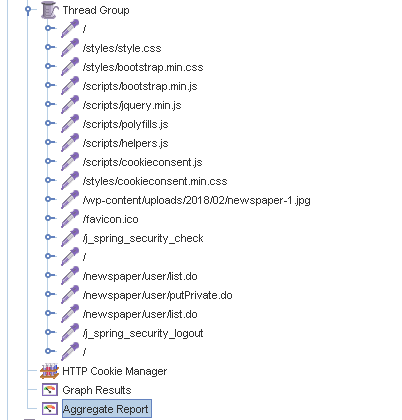


Así que lo que hacemos es observar los componentes de mi Pc para ver que podría estar provocando un fallo y he llegado a la conclusión que se trata del procesador ya que la gráfica asociada a esta tiene muchos picos y al ser menos cercana al suelo es menos buena.



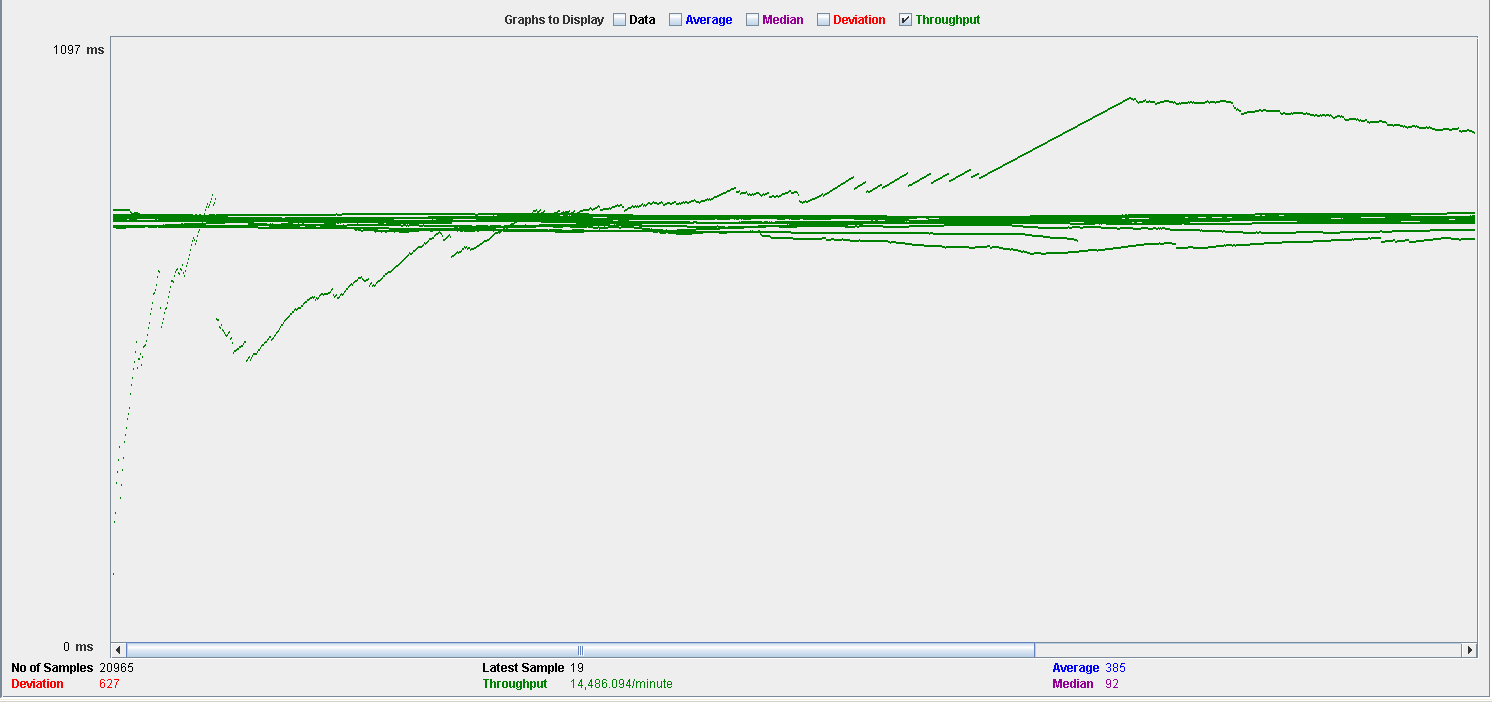
Test de rendimiento poner a privado:

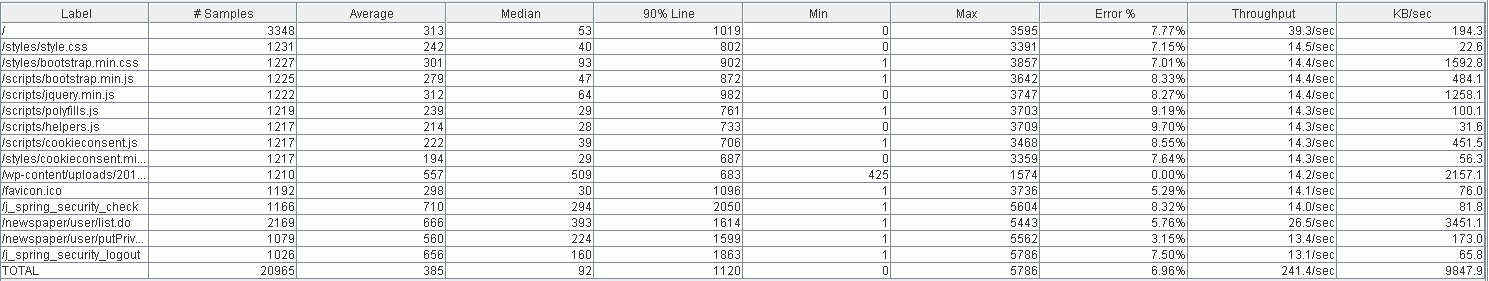
Realizamos una prueba estableciendo como número de usuarios simultáneo 250 personas con un tipo de apertura entre persona y persona de 1/250. Con esta configuración tras ejecutar el siguiente script:



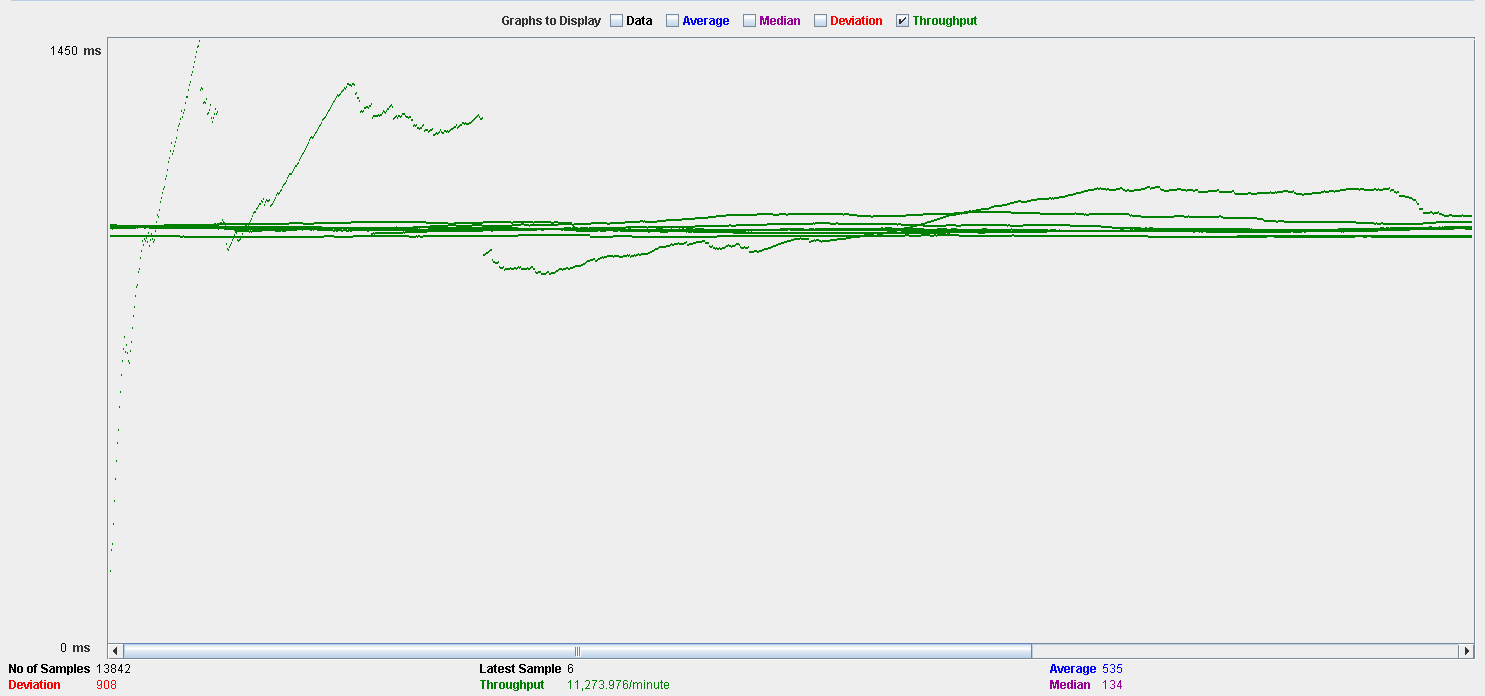
Nos logeamos como user, nos vamos a nuestros newspapers, ponemos uno público en privado y nos deslogeamos.

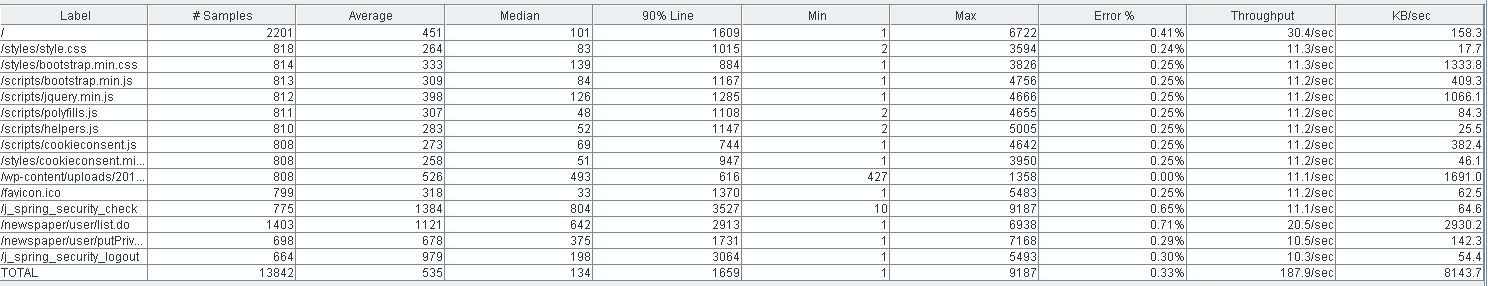
Tras tener esta secuencia de pasos ejecutamos los tests y se nos genera las siguientes gráficas.



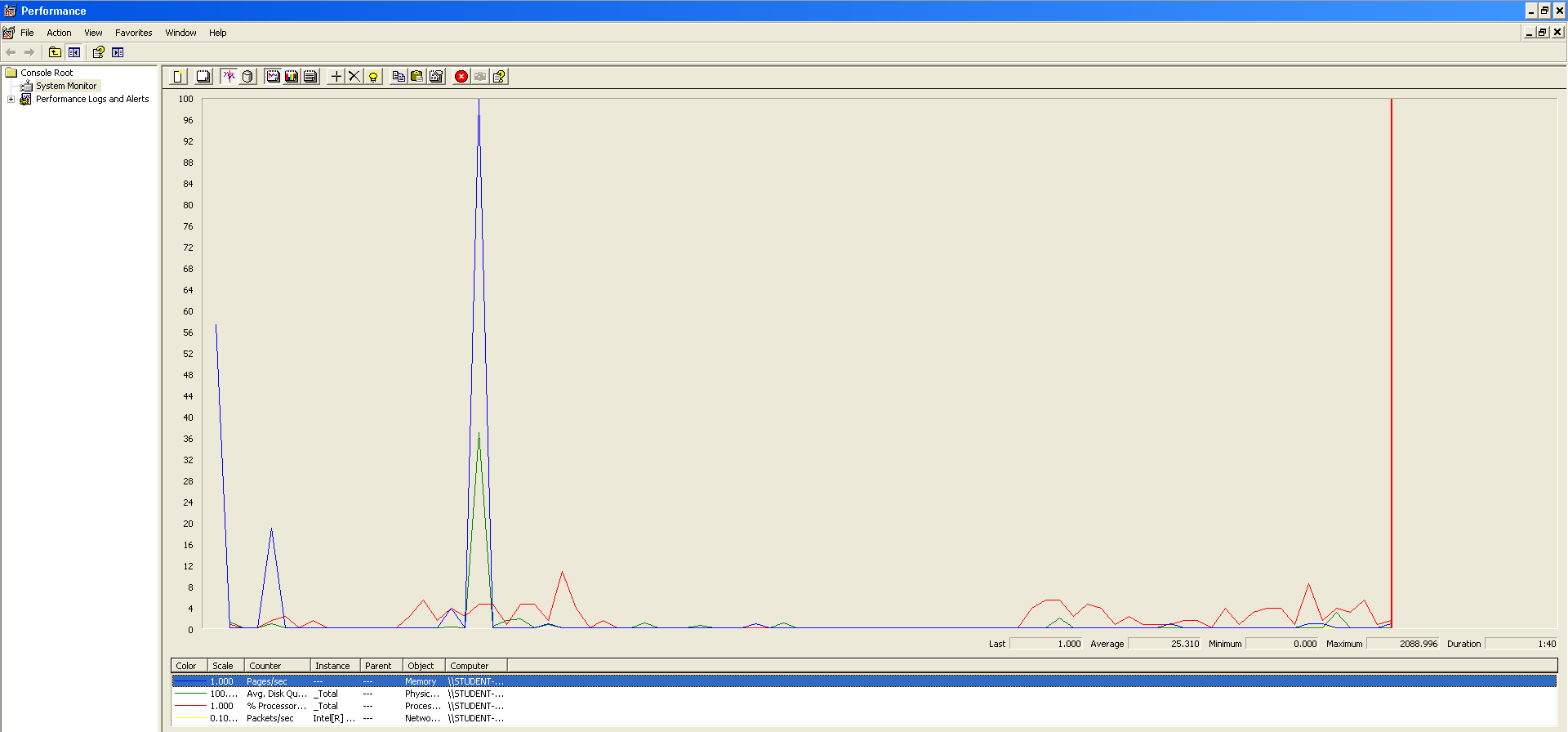


Como vemos se han producido bastantes errores por lo tanto el valor de usuarios máximos simultaneo que puede soporta la aplicación es menor. Por lo tanto, buscamos con un número menor y probamos con 200 usuarios simultáneos. Las gráficas que nos salen al hacer esto son las siguientes:



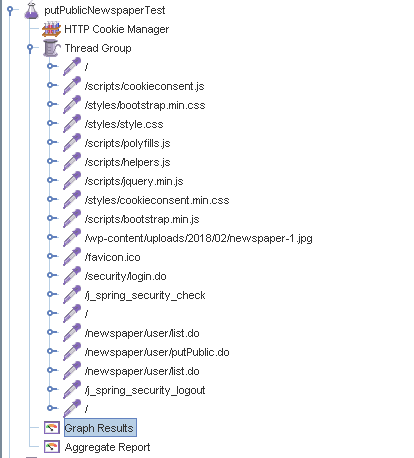
Como vemos el porcentaje del error es muy cercano a cero, por lo tanto 200 usuarios es un valor muy cercano al límite máximo que es capaz de soportar nuestra aplicación de forma simultánea para que se haga el caso de uso sin que haya ningún fallo. Cabe destacar por la columna de 90% Line que los pasos que más tiempo producen son aquellos realizados por el usuario donde se hace una petición Post.

Haciendo una comprobación con el performance vemos que se está produciendo un cuello de botella en la memoria del sistema debido al alto valor que produce esta al principio.



Test de rendimiento de poner a público un newspaper

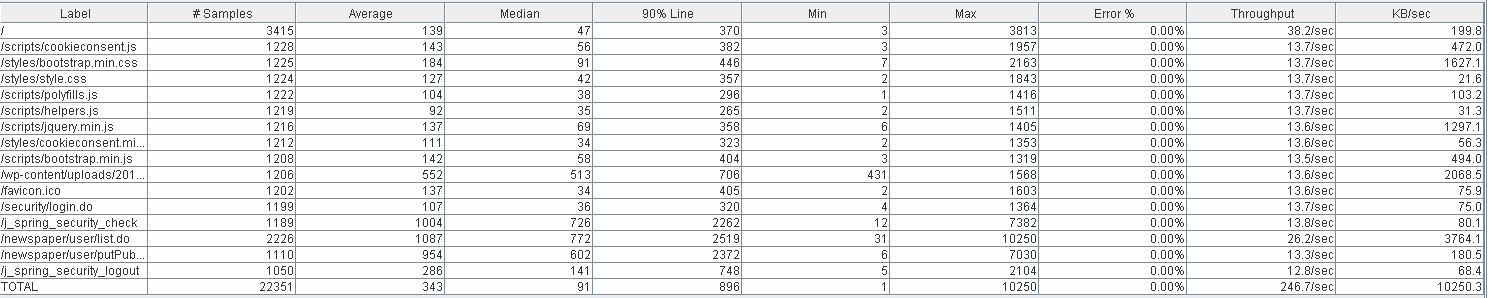
Realizamos una prueba estableciendo como número de usuarios simultáneo de 200 con un retraso de medio segundo entre usuario y usuario. Con esta configuración tras ejecutar el siguiente script:



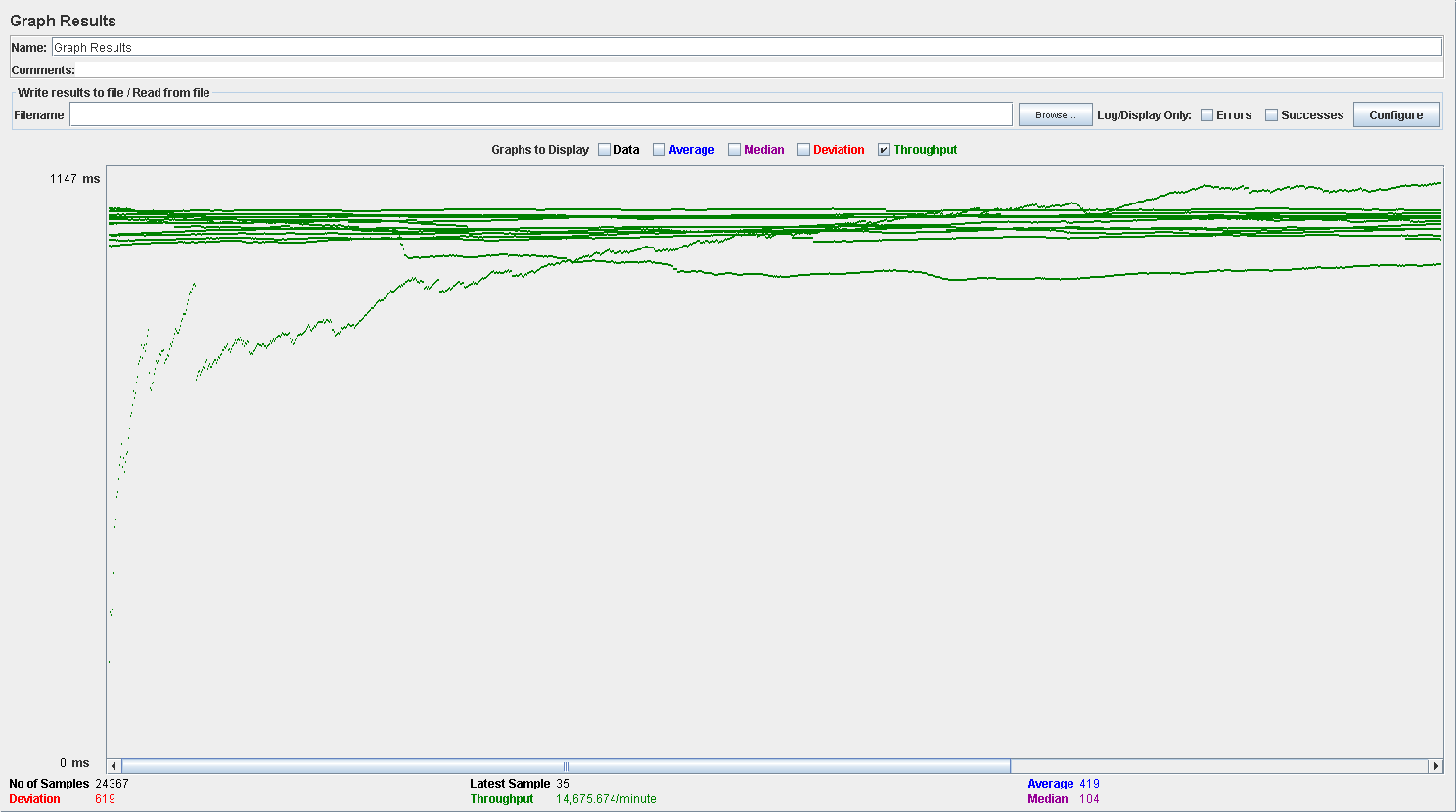
Nos logeamos como user, nos vamos a nuestros newspapers, ponemos uno privado en público y nos deslogeamos.

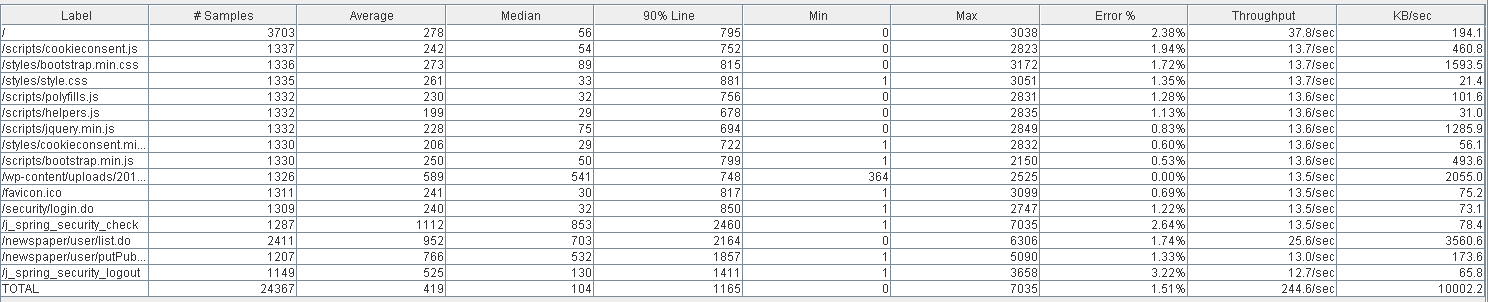
Tras tener esta secuencia de pasos ejecutamos los tests y se nos genera las siguientes gráficas.



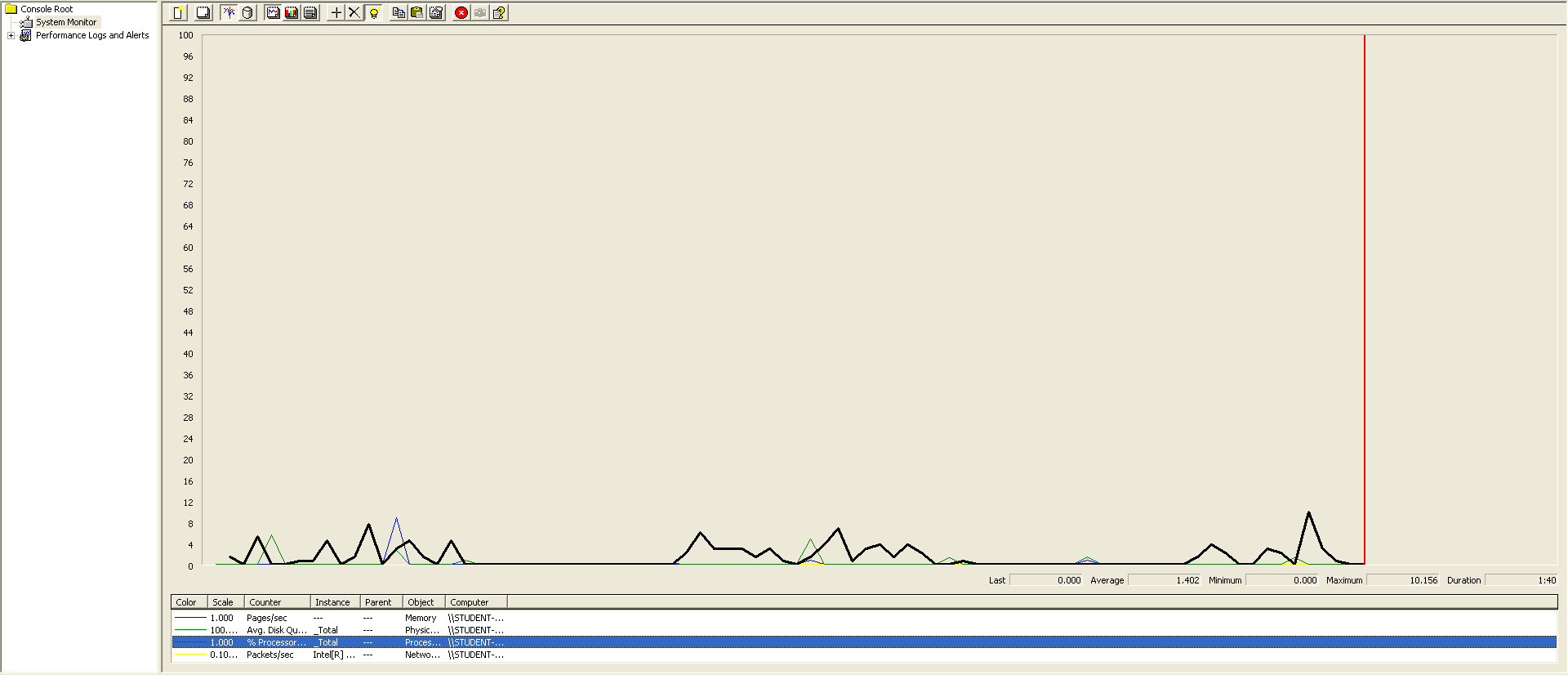


Como vemos no se producen errores con esta cantidad de usuarios, por lo que vamos a aumentar el número para ver si aparecen errores. Aumentamos el número a 230 usuarios simultáneos. Las gráficas que aparecen son los siguientes:



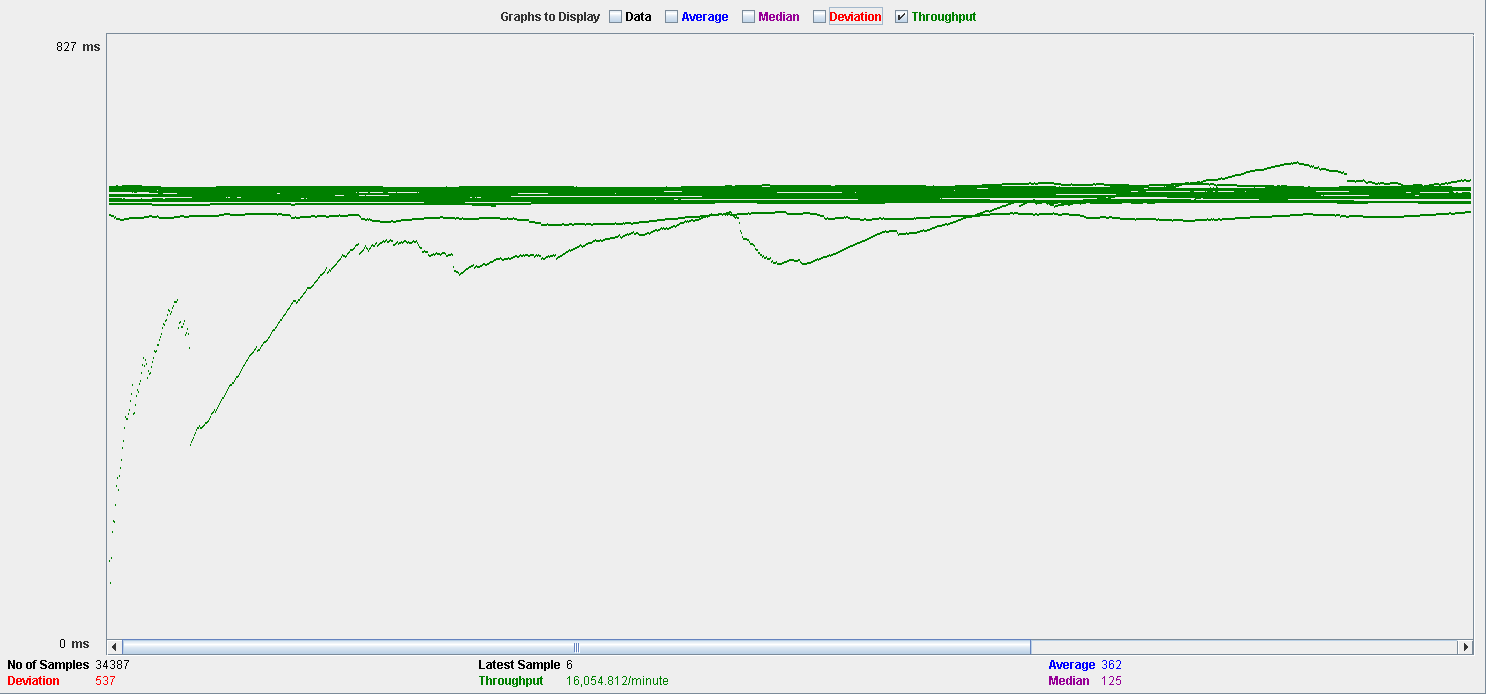


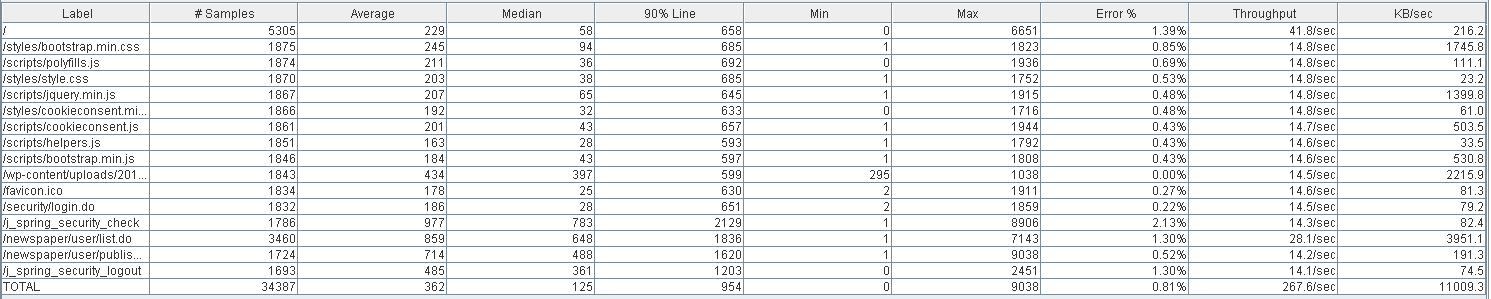
Con este valor empiezan a aparecer pequeños errores por lo que seguramente el valor máximo de usuarios concurrentes sea alrededor de 220. Viendo el performance podemos observar que lo que tiene una mayor carga es la memoria, por lo tanto, el cuello de botella se produciría ahí.



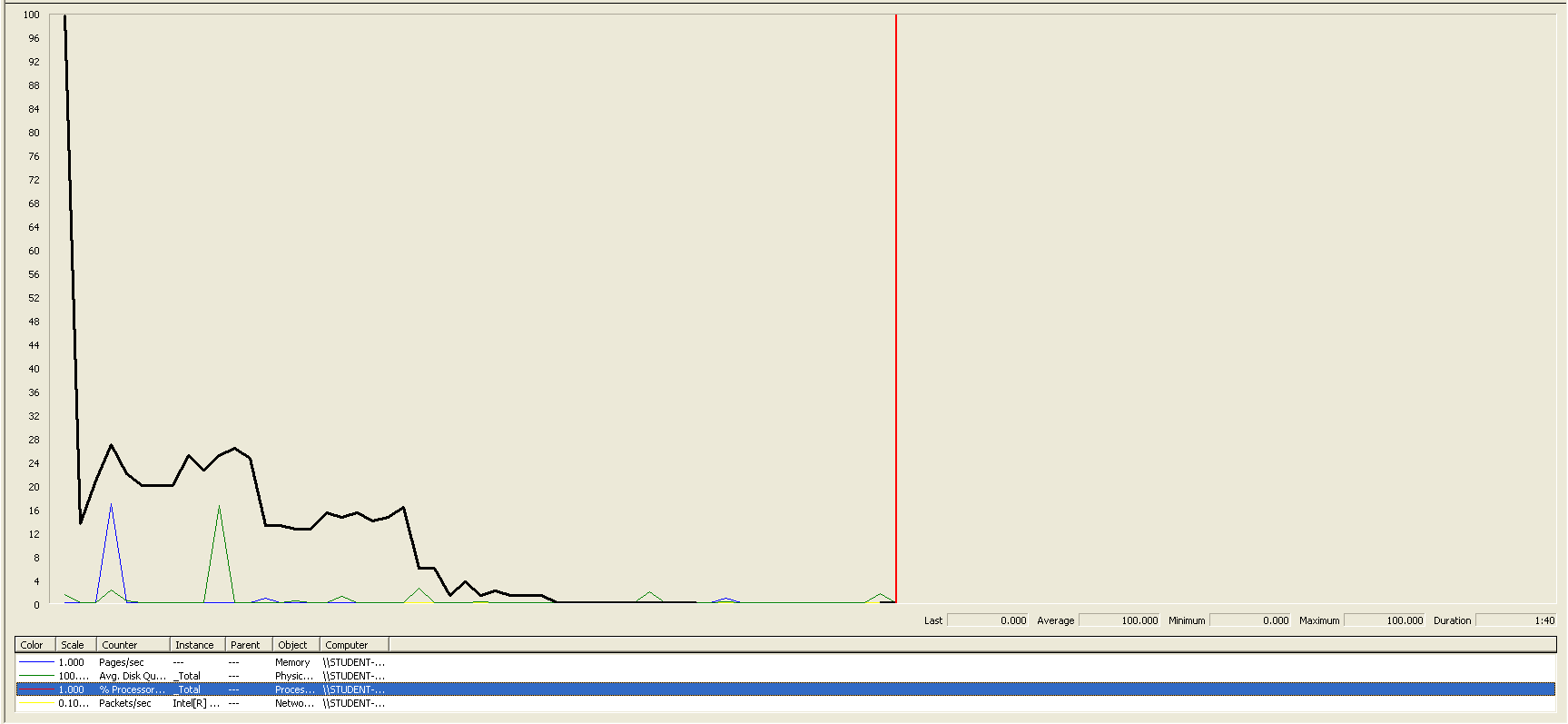
Test de rendimiento publicar Newspaper

Probamos inicialmente con 200 y 210 usuarios simultáneos, en ambos la probabilidad era cero, por lo tanto, aumentamos hasta 225 donde se empezó a producir errores, por lo tanto este es aproximadamente el valor máximo de usuarios simultáneos. Las gráficas que se obtienen son:



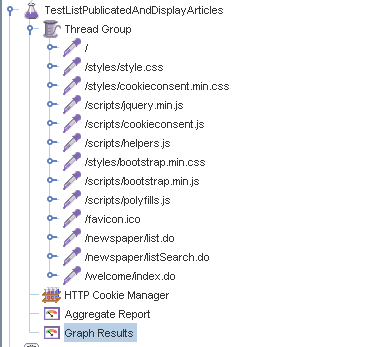


Se han producido los errores, mirando en el performance podemos observar que el cuello de botella se produce en el procesador debido a los altos valores que esta toma.

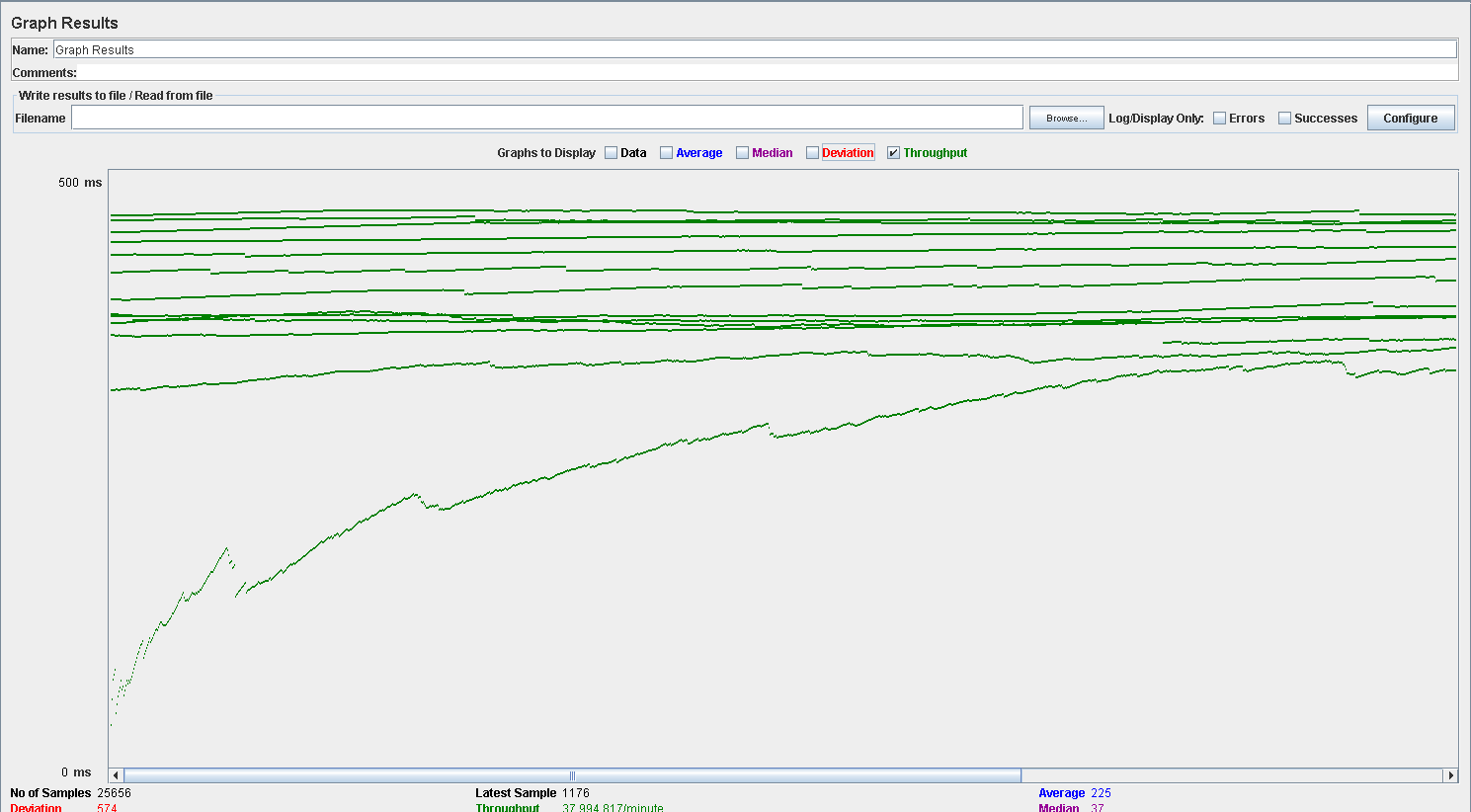


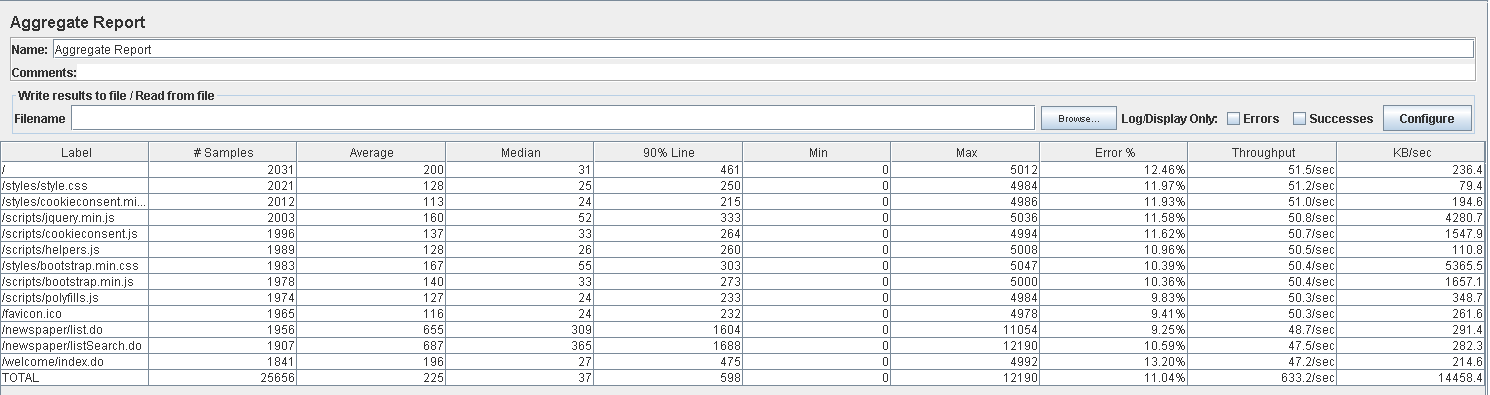
Test de rendimiento del buscador de newspaper:

Realizamos una prueba estableciendo como número de usuarios simultáneo 200 con un retraso de medio segundo entre persona y persona. Con esta configuración tras ejecutar el siguiente script:



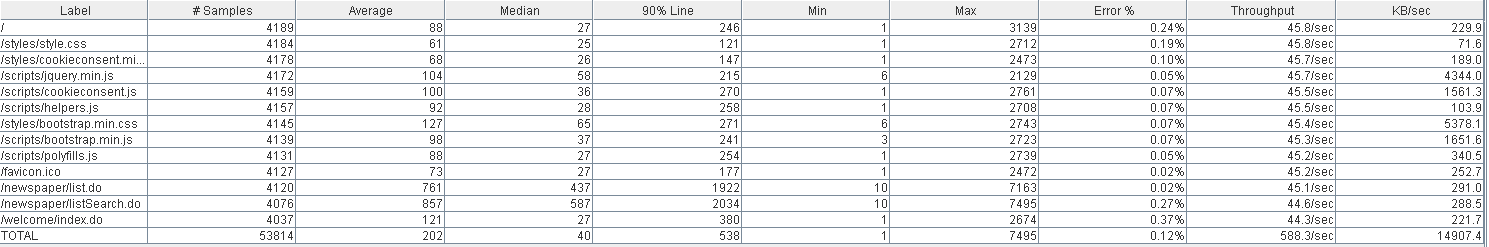
Al realizar el test nos salen las siguientes gráficas:



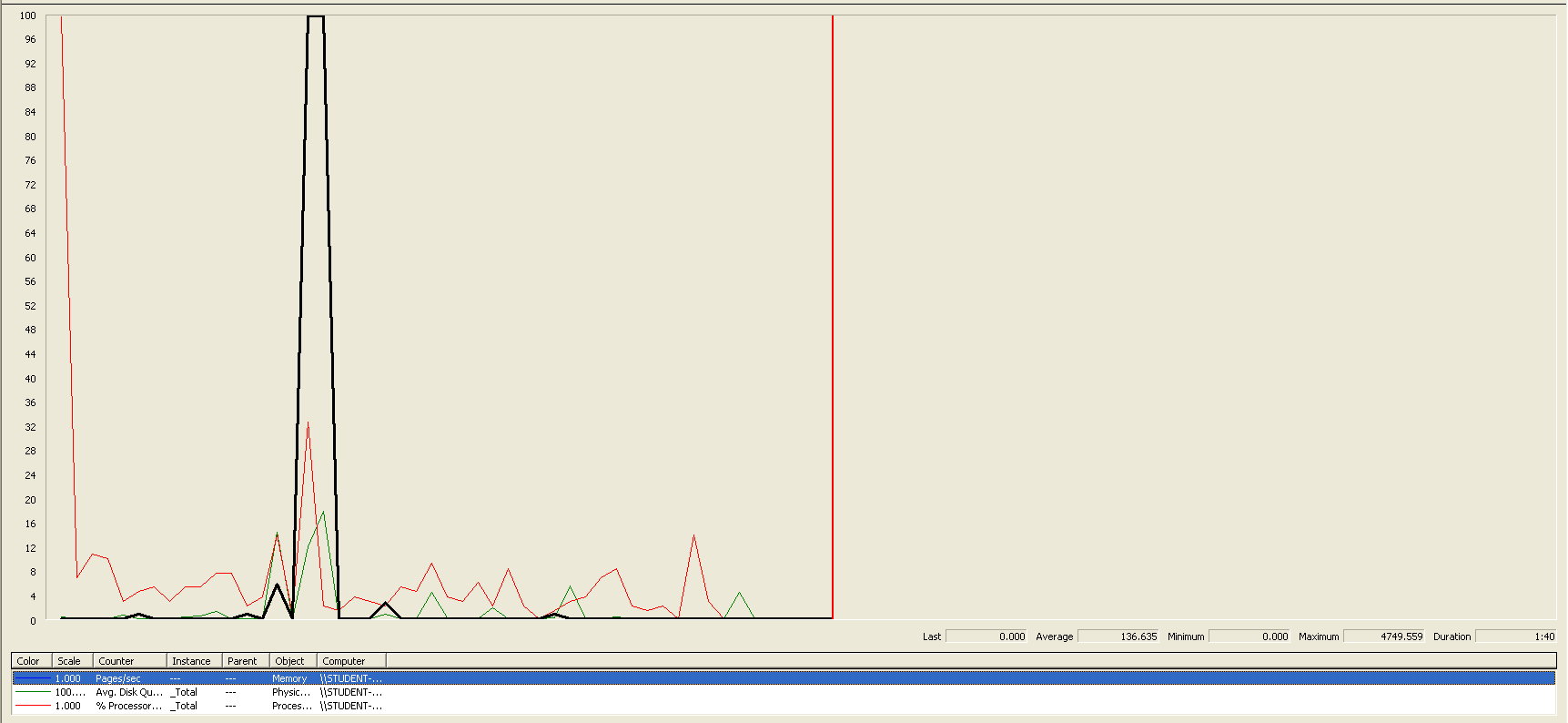


Como vemos los valores del error son bastante elevados por lo que probamos a buscar el máximo de usuarios concurrentes disminuyendo el número de usuarios. Probamos con 160 y en ese caso las gráficas resultantes son:



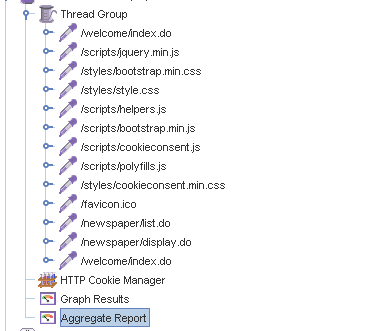


Con este valor los errores son muy cercanos a cero por lo que podemos ver que 160 es un valor muy aproximado al número de usuarios máximo simultaneo que puede tener la aplicación para este caso de uso. Analizando el performance podemos ver que es en la memoria donde ocurre un cuello de botella.

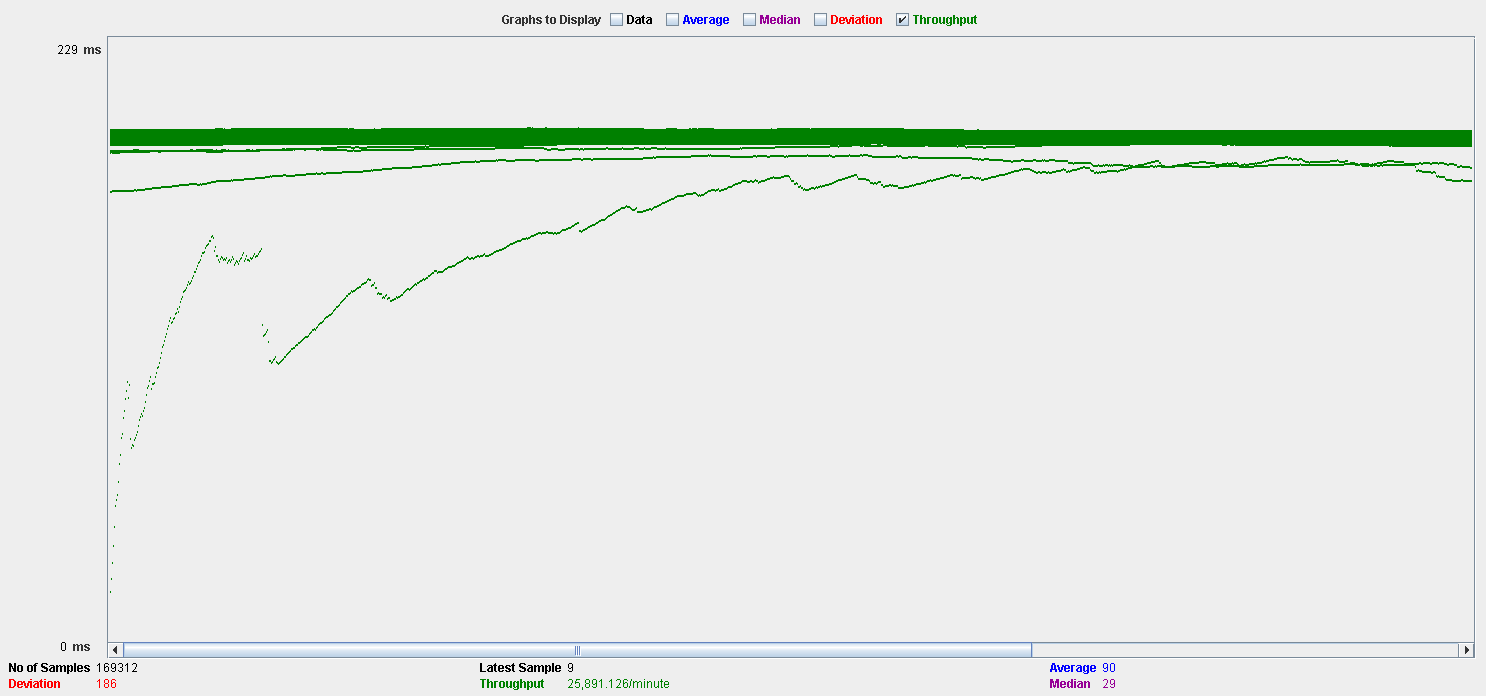


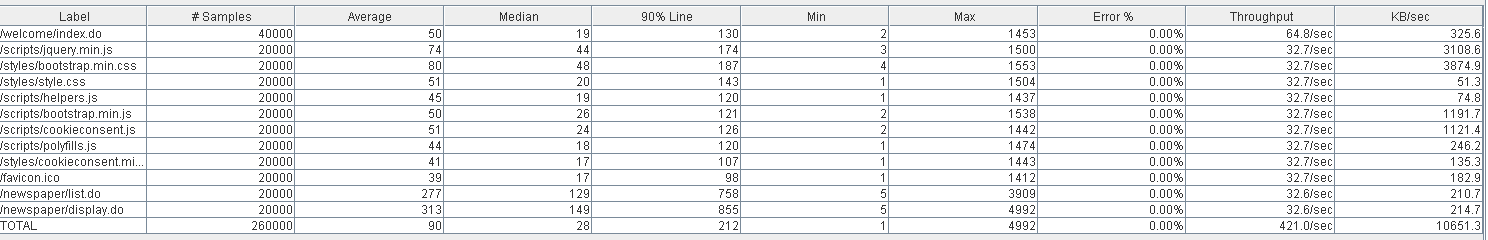
Test de rendimiento de listar todos los newspapers y desplegar sus artículos

Realizamos una prueba estableciendo como número de usuarios simultáneo con un retraso de medio segundo entre usuario y usuario.Con esta configuración tras ejecutar el siguiente script:

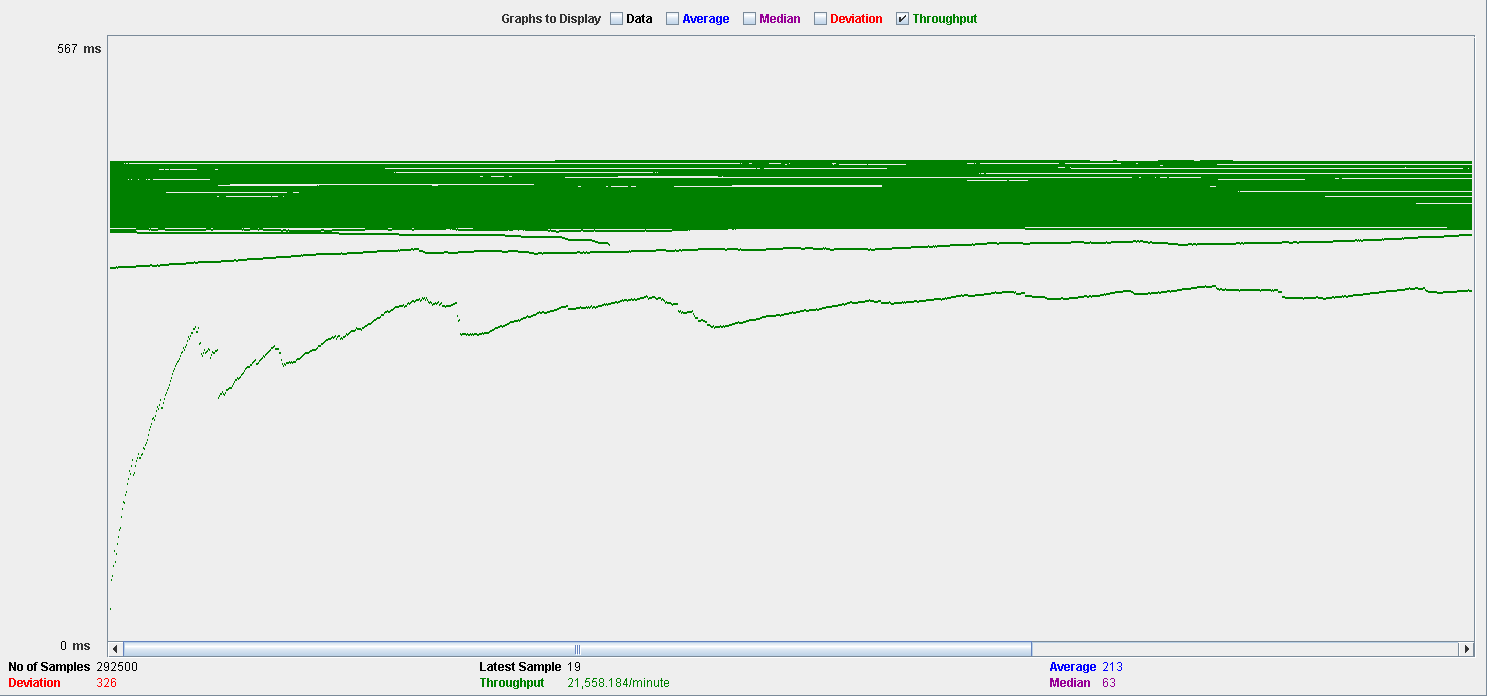


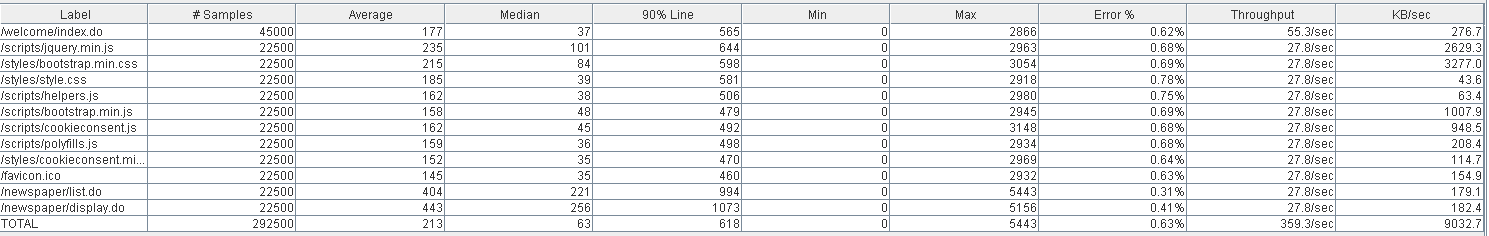
Al realizar el test aparecen las siguientes gráficas:





Como vemos el error es cero, así que vamos a aumentar el número de personas conectadas simultáneamente conectadas. Probamos con 225 y en ese caso las gráficas resultantes son:



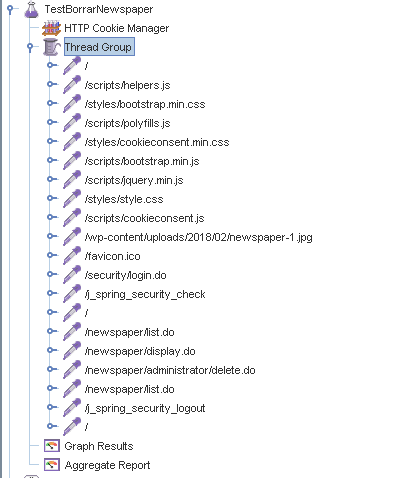


Vemos que los errores son muy parecidos a cero por lo tanto el número máximo de usuarios simultáneamente haciendo lo mismo es de 225. Analizando el performance podemos ver que el procesador es el que tiene mayor peso.

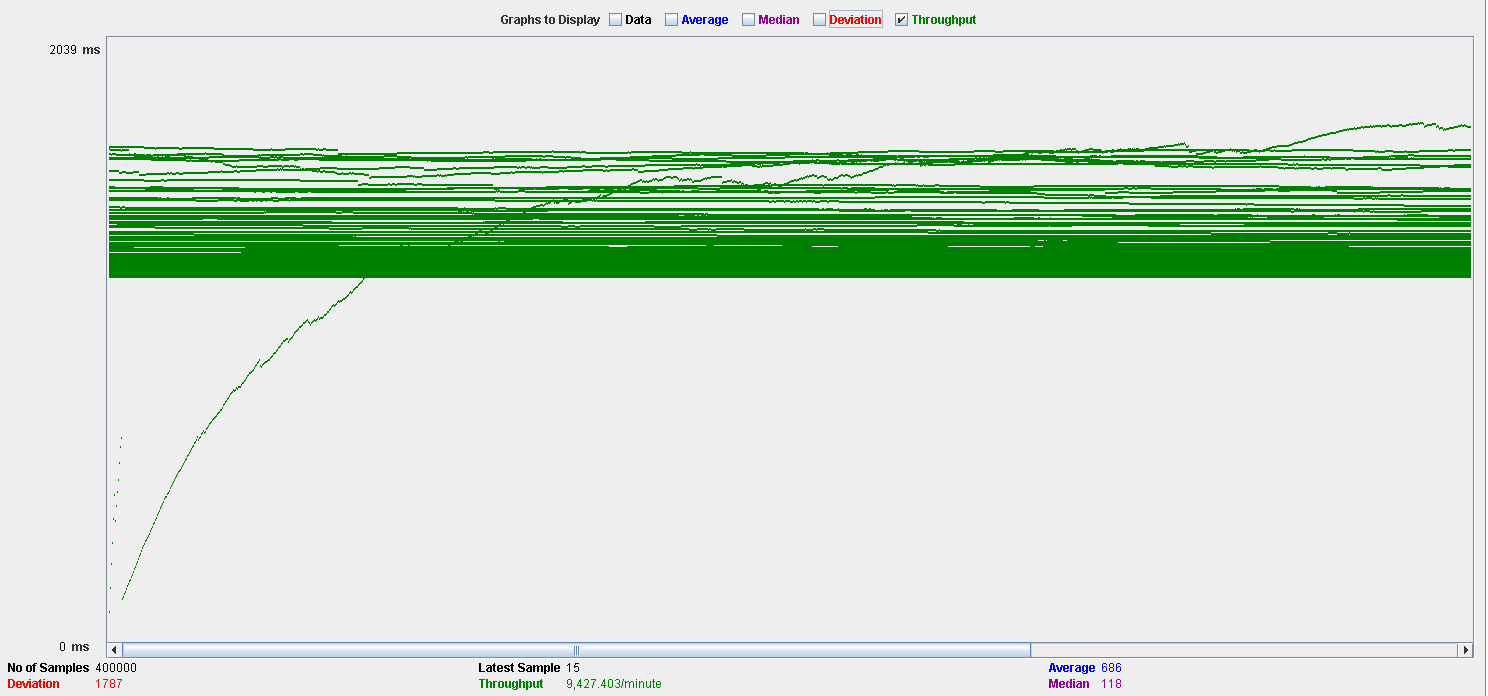


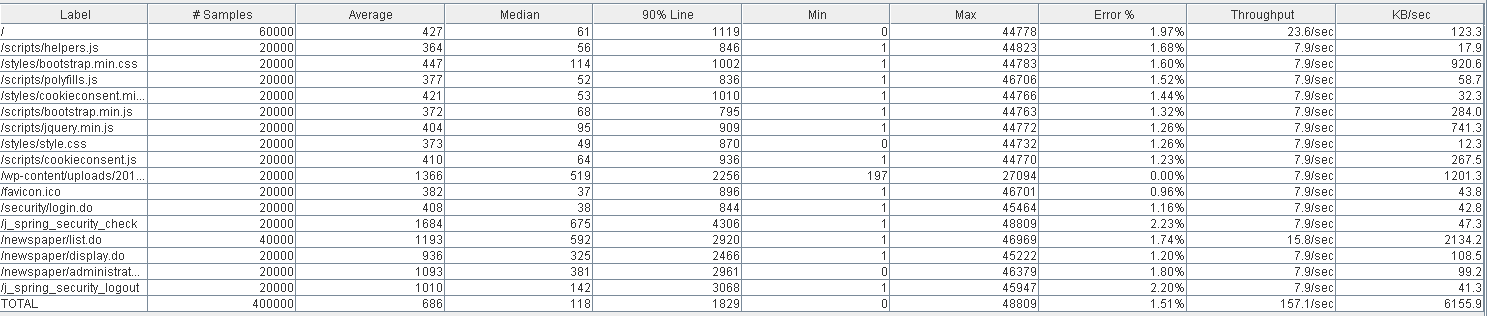
Test de borrar un newspaper

Realizamos una prueba estableciendo como número de usuarios simultáneo 200 con un retraso de medio segundo entre usuario y usuario.Con esta configuración tras ejecutar el siguiente script:



Al realizar el test aparecen las siguientes gráficas:





Vemos que los errores son muy parecidos a cero por lo tanto el número máximo de usuarios simultáneamente haciendo lo mismo es de 200. Analizando el performance podemos ver que el procesador, tarjeta de red y memoria son los principales causantes de que haya un cuello de botella.



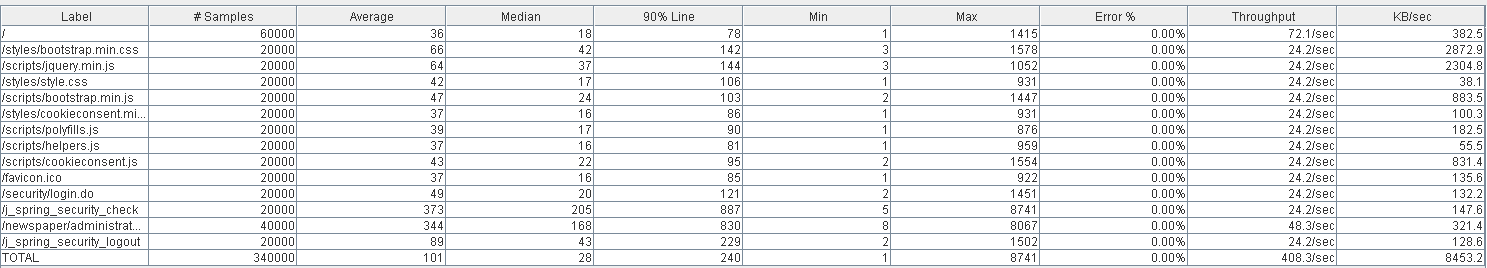
Test de listar los newspapers con palabras taboo

Realizamos una prueba estableciendo como número de usuarios simultáneo 200 con un retraso de medio segundo entre usuario y usuario.. Con esta configuración tras ejecutar el siguiente script:

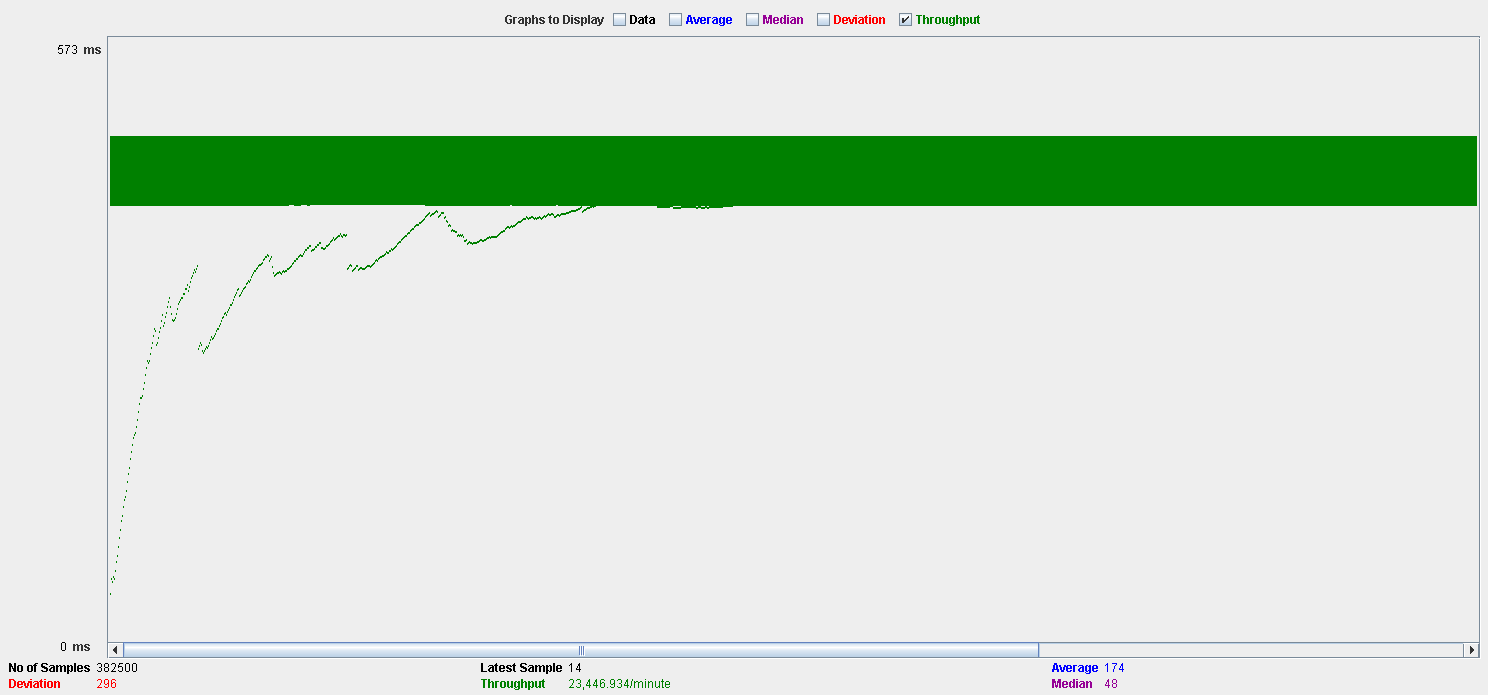


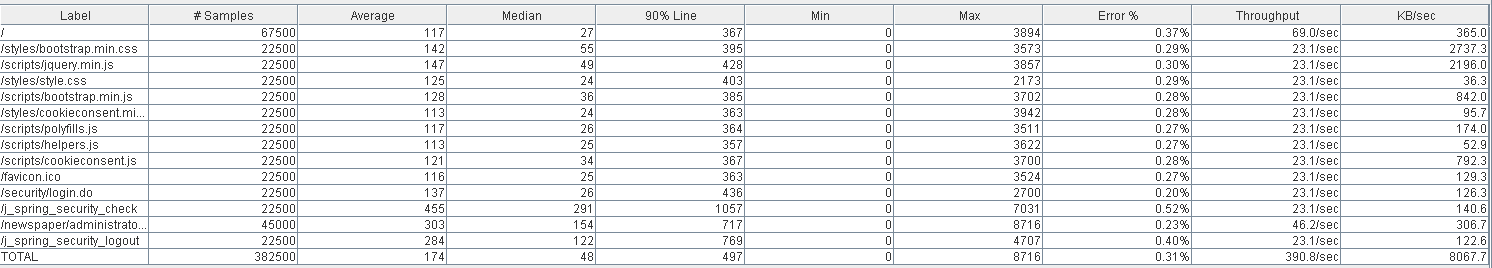
Al realizar el test aparecen las siguientes gráficas:



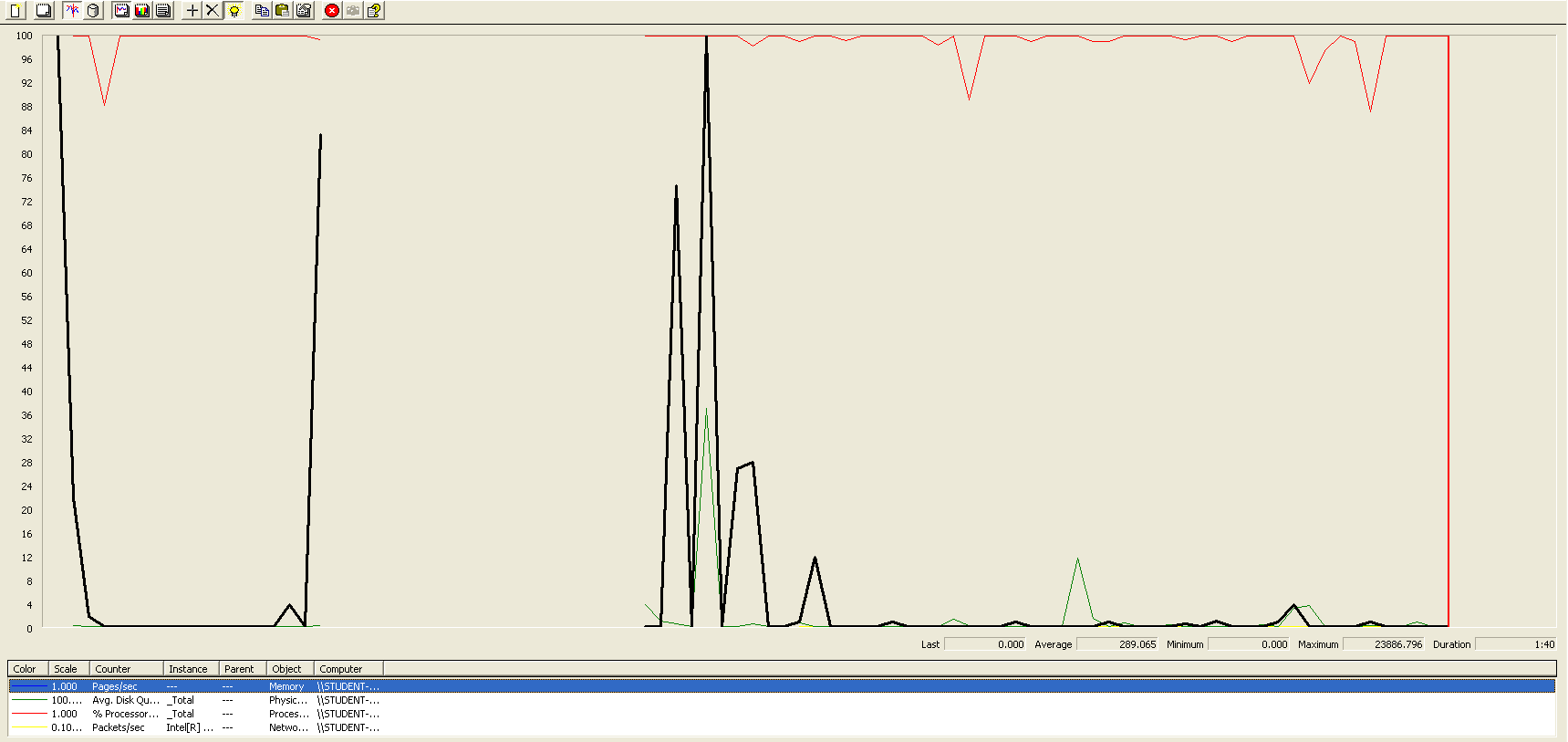


Como vemos el error es cero, así que vamos a aumentar el número de personas conectadas simultáneamente. Probamos con 225 y en ese caso las gráficas resultantes son:



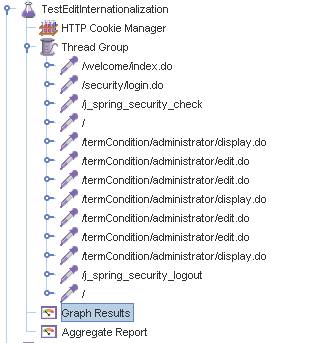


Vemos que los errores son muy parecidos a cero por lo tanto el número máximo de usuarios simultáneamente haciendo lo mismo es de 225. Analizando el performance podemos ver que la memoria es el principal causante de que haya un cuello de botella.

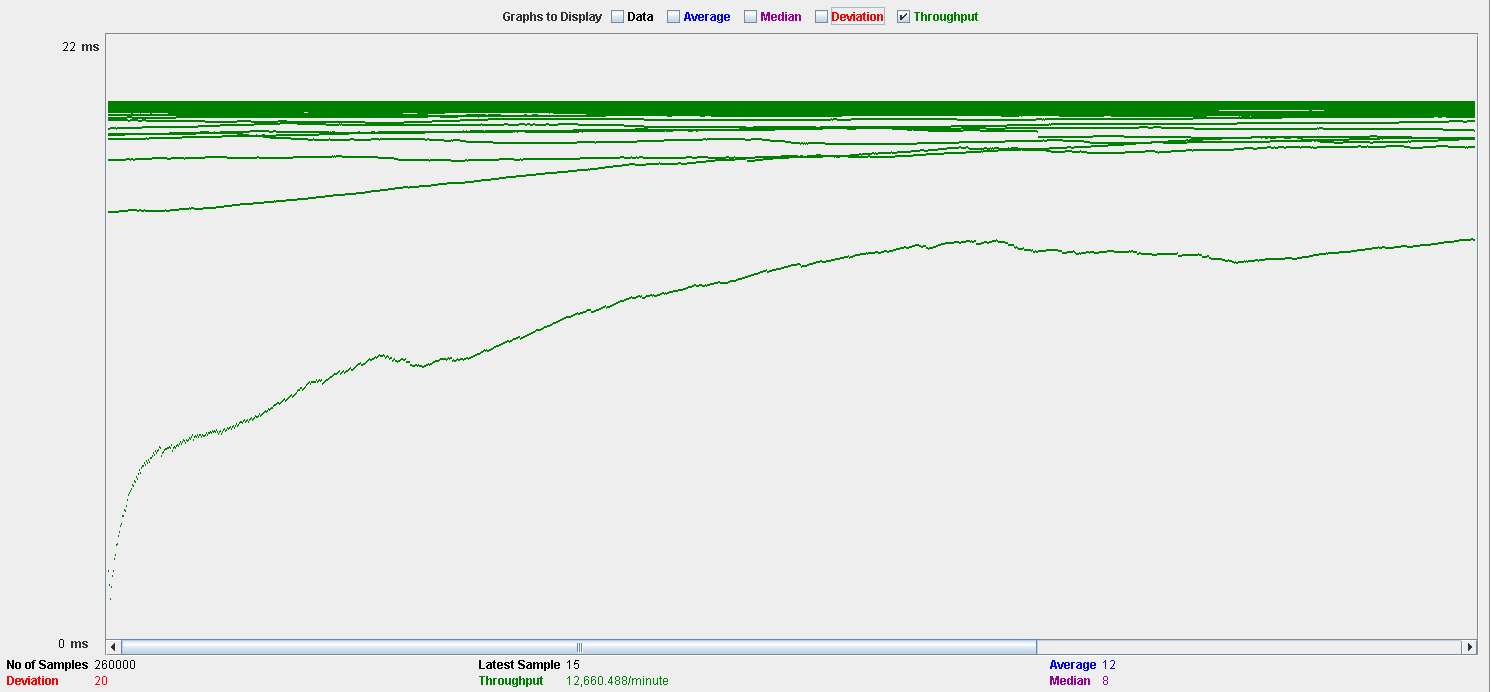
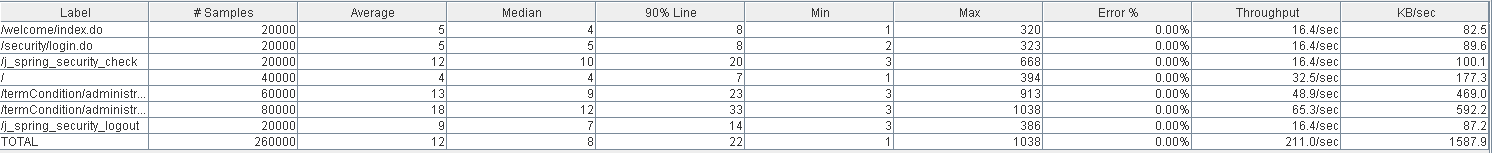


Test editar los términos y condiciones

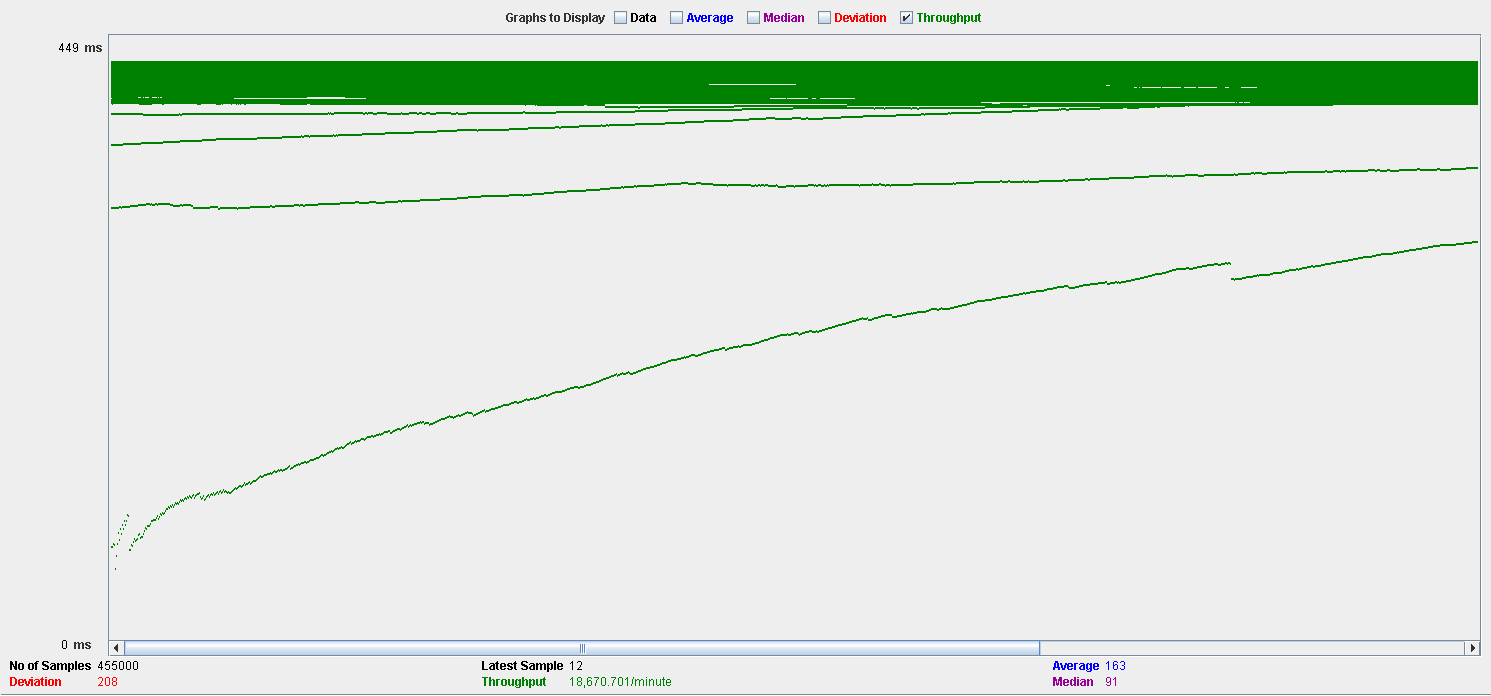
Realizamos una prueba estableciendo como número de usuarios simultáneo 200 con un retraso de medio segundo entre usuario y usuario. Con esta configuración tras ejecutar el siguiente script:

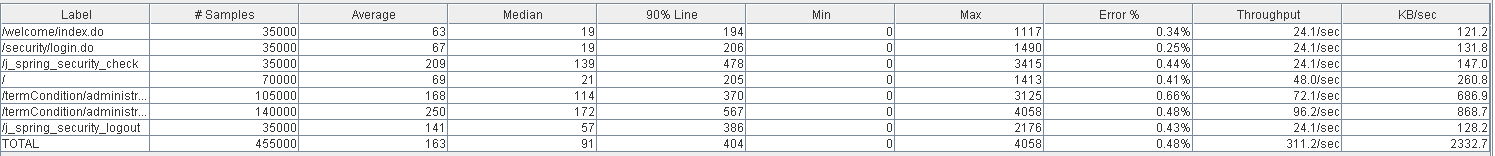


Al realizar el test aparecen las siguientes gráficas:



Como vemos el error es cero, así que vamos a aumentar el número de personas conectadas simultáneamente conectadas. Probamos con 350 y en ese caso las gráficas resultantes son:





Vemos que los errores son muy parecidos a cero por lo tanto el número máximo de usuarios simultáneamente haciendo lo mismo es de aproximadamente 350 para este caso de uso. Analizando el performance podemos ver que la memoria es el principal causante de que haya un cuello de botella.

