



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E
INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LAS INFORMACIÓN

Tema:

**ANÁLISIS DE ALGORITMOS BALANCEADORES DE CARGA
UTILIZANDO MÁQUINAS VIRTUALES PARA LA OPTIMIZACIÓN DE
TIEMPO Y RECURSOS EN SERVIDORES WEB.**

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Tecnologías de la Información.

ÁREA: Software

LÍNEA DE INVESTIGACIÓN: Tecnología de la información y sistemas de control

AUTOR: Paúl Sebastián González Sarabia

TUTOR: Ing. Rubén Eduardo Nogales Portero, Mg.

Ambato – Ecuador

agosto - 2023

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: ANÁLISIS DE ALGORITMOS BALANCEADORES DE CARGA UTILIZANDO MÁQUINAS VIRTUALES PARA LA OPTIMIZACIÓN DE TIEMPO Y RECURSOS EN SERVIDORES WEB, desarrollado bajo la modalidad Proyecto de Investigación por el señor Paúl Sebastián González Sarabia, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, agosto 2023.

Ing. Rubén Eduardo Nogales Portero, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación titulado: ANÁLISIS DE ALGORITMOS BALANCEADORES DE CARGA UTILIZANDO MÁQUINAS VIRTUALES PARA LA OPTIMIZACIÓN DE TIEMPO Y RECURSOS EN SERVIDORES WEB es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, agosto 2023.



Paúl Sebastián González Sarabia
C.C 1804954756
AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, agosto 2023.



Paúl Sebastián González Sarabia
C.C 1804954756
AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Paúl Sebastián González Sarabia, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado ANÁLISIS DE ALGORITMOS BALANCEADORES DE CARGA UTILIZANDO MÁQUINAS VIRTUALES PARA LA OPTIMIZACIÓN DE TIEMPO Y RECURSOS EN SERVIDORES WEB, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, agosto 2023.

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL

Ing. Mg. David Omar Guevara
Aulestia
PROFESOR CALIFICADOR

Ing. Mg. Dennis Vinicio Chicaiza
Castillo
PROFESOR CALIFICADOR

DEDICATORIA

A mis padres, Anita y Paul, les agradezco de todo corazón por su incansable paciencia, dedicación y sacrificio. Han sido mi mayor inspiración y su ejemplo de perseverancia y valentía me ha impulsado a superar cualquier obstáculo. Gracias por creer en mí y por enseñarme que no hay límites para alcanzar mis sueños.

A mi querido hermano Juan Esteban y a Valentina, les agradezco por su incondicional apoyo y cariño a lo largo de este camino. Siempre han estado a mi lado, brindándome fuerza y ánimo en cada paso que he dado. Su presencia ha sido fundamental para mi crecimiento personal y académico.

Finalmente, quiero dedicar este proyecto a mis abuelitos maternos y paternos. Ellos son una luz en mi vida, guiándome y motivándome a alcanzar mis metas y sueños; pero sobre todo a la memoria de Víctor Sarabia que, aunque ya no esté físicamente con nosotros, su legado de lucha y dedicación vive en mi corazón.

Les dedico este trabajo a cada uno de ustedes por formar parte de mi vida y por ser mi mayor motivación. Este proyecto de tesis es un reflejo de nuestro amor, esfuerzo y trabajo en equipo. Sin su presencia y apoyo, nada de esto sería posible.

Paúl Sebastián González Sarabia

AGRADECIMIENTO

En primer lugar, agradezco a Dios y a mi familia por ser parte fundamental de mi desarrollo personal y académico.

Quiero agradecer a Noemi "mi persona" por estar a mi lado a pesar del poco tiempo que nos conocemos. Tu apoyo incondicional y tus palabras de aliento han sido de gran valor para mí durante este tiempo. A Felipe, Luis, Anthony L, Anthony P y David, mi grupo de amigos de universidad, les agradezco por ser una parte fundamental en este proceso. Juntos hemos compartido desafíos, éxitos y momentos de aprendizaje. A Andrea, mi querida "bestie", quiero expresar mi más profundo agradecimiento por tu apoyo constante, consejos sabios y alegría contagiosa que han sido un gran respaldo emocional para mí en todos estos años de amistad.

También quiero agradecer a mi tutor de tesis, el ingeniero Rubén Nogales, por su guía experta y apoyo constante, que han sido fundamentales en el desarrollo de este proyecto. Además, quiero expresar mi gratitud a mis profesores, quienes han dejado su huella en mi formación académica.

A cada una de las personas mencionadas y a todas aquellas que, de alguna manera, han dejado su huella en mi vida, les agradezco de corazón. Sus palabras de aliento, gestos de amistad y momentos compartidos han sido un motor importante en mi camino hacia la culminación de esta tesis.

ÍNDICE GENERAL DE CONTENIDOS

APROBACIÓN DEL TUTOR	ii
AUTORÍA	iii
DERECHOS DE AUTOR	iv
DEDICATORIA.....	ii
AGRADECIMIENTO.....	iii
CAPÍTULO I.- MARCO TEÓRICO	1
1.1 Tema de Investigación	1
1.1.1 Planteamiento del Problema.....	1
1.2 Antecedentes Investigativos	3
1.3 Fundamentación Teórica	11
1.4 Objetivos	18
1.4.1 Objetivo General	18
1.4.2 Objetivos Específicos	18
CAPÍTULO II.- METODOLOGÍA	19
2.1 Materiales.....	19
2.2 Métodos.....	19
2.2.1 Modalidad de la Investigación	19
2.2.2 Población y Muestra	19
2.2.3 Recolección de Información.....	27
2.2.4 Procesamiento y Análisis de Datos.....	28

CAPÍTULO III.- RESULTADOS Y DISCUSIÓN	30
3.1 Análisis y discusión de los resultados.....	30
3.1.1 Análisis comparativo de metodologías Ágiles	31
3.1.2 Determinación del administrador de servidores.....	38
3.1.3 Determinación de la distribución open source para servidores.....	40
3.1.4 WordPress.....	43
3.1.5 LAMP	44
3.1.6 JMETER.....	44
3.1.7 Libvirt	45
3.1.8 Virt Manager	46
3.1.9 Power BI	47
3.1.10 Tableau	48
3.1.11 R Studio.....	48
3.1.12 Excel.....	50
3.2 Desarrollo de la propuesta.....	51
3.2.1 Instalación y configuración del entorno de pruebas.	51
3.2.2 Ajuste de los parámetros para pruebas en JMETER	52
3.2.3 Obtención y normalización de los datos de prueba	57
3.2.4 Graficar las líneas de tiempo en Power BI.....	64
3.2.5 Realizar los gráficos de barras comparativos en Tableau	72
3.2.6 Obtención de ANOVA en R Studio	77

3.2.7 Definición del algoritmo óptimo	79
CAPÍTULO IV.- CONCLUSIONES Y RECOMENDACIONES.....	84
4.1 Conclusiones	84
4.2 Recomendaciones	84
REFERENCIAS BIBLIOGRÁFICAS.....	85
ANEXOS.....	88

ÍNDICE DE FIGURAS

Figura 1. Gráfica de la cantidad de búsquedas Balanceo y Balanceador de carga ...	3
Figura 2. Número de ingresos diarios, Empresa 1	22
Figura 3. Número de ingresos diarios, Empresa 2	22
Figura 4. Número de ingresos diarios, Empresa 3	23
Figura 5. Usuarios usando la plataforma Steam en tiempo real[33].....	23
Figura 6. Diagrama del entorno de pruebas	51
Figura 7. Parámetros de prueba en Jmeter	53
Figura 8. Iniciar agente del servicio Jmeter	53
Figura 9. Tipos de prueba en Jmeter	54
Figura 10. Cantidad de interacciones simultaneas.....	54
Figura 11. Tiempo de respuesta por página.....	55
Figura 12. Cuadro de resumen de tiempo de respuesta.....	55
Figura 13. Gráfica de uso de CPU	56
Figura 14. Gráfica del uso de Memoria	56
Figura 15. Árbol de resultados	57
Figura 16. Transformar datos de csv a xlsx.....	58
Figura 17. Fórmula de transformación de timeStamp	59
Figura 18. Fórmula de transformación de elapsed.....	60
Figura 19. Tiempo y porcentaje de uso por algoritmo	61
Figura 20. Promedio de cada columna de porcentaje	61
Figura 21. Tabla de valores de CPU-VM-ALGORITMO.....	62
Figura 22. Datos crudos del tiempo de respuesta	63
Figura 23. Tabla de resumen del tiempo de respuesta	63
Figura 24. Promedio de cuadros de tiempo de respuesta	64

Figura 25. Columna de datos importados a Power BI	65
Figura 26. Relación de datos en Power BI	66
Figura 27. Ítems del eje x; eje y.....	67
Figura 28. Gráfica de tiempo de Main	68
Figura 29. Gráfica de tiempo de VM1	69
Figura 30. Gráfica de tiempo de VM2	70
Figura 31. Gráfica de tiempo del disco duro de la máquina física	71
Figura 32. Ítems para las gráficas en Tableau	72
Figura 33. Gráfica del consumo de CPU en VM1 y VM2	73
Figura 34. Gráfica del consumo de memoria en VM1 y VM2.....	74
Figura 35. Ítems para la gráfica de tiempos de respuesta	74
Figura 36. Gráfica de tiempo de respuesta de cada página	75
Figura 37. Gráfica de tiempo de respuesta por algoritmo	76
Figura 38. Gráfica de la cantidad de interacciones procesadas por algoritmo	77
Figura 39. ANOVA VM1-CPU.....	78
Figura 40. ANOVA VM2-CPU.....	78
Figura 41. ANOVA VM1-Memoria	79
Figura 42. ANOVA VM2-Memoria	79

ÍNDICE DE TABLAS

Tabla 1. Cadenas de búsqueda	4
Tabla 2. Criterios de inclusión y exclusión.....	5
Tabla 3. Cuadro comparativo de Algoritmos Balanceadores de Carga más usados	10
Tabla 4. Características de los ordenadores con la cantidad representada en porcentajes.[34]	24
Tabla 5. Memoria RAM de los ordenadores.....	24
Tabla 6. Velocidades de CPU de los ordenadores	25
Tabla 7. Número de CPUs físicas de los ordenadores.....	25
Tabla 8. Espacio libre en disco de los ordenadores.....	26
Tabla 9. Espacio total de almacenamiento de los usuarios.....	27
Tabla 10. Problemas que enfrentan los servidores web en máquinas virtuales	30
Tabla 11. Resumen de las metodologías ágiles	34
Tabla 12. Comparación entre administradores de balanceo de carga.	39
Tabla 13. Comparación de las distintas distribuciones open source	41

RESUMEN EJECUTIVO

Las pequeñas empresas del Ecuador que cuentan con servidores web enfrentan diversos problemas para manejar de manera eficiente el tiempo y los recursos. Esto afecta su rendimiento y calidad de servicio. Se propone realizar un análisis comparativo entre los diferentes algoritmos balanceadores de carga los cuales permiten la distribución de solicitudes entre máquinas virtuales según distintos parámetros. El objetivo general del proyecto es determinar cuál de estos algoritmos ofrece el mejor desempeño en la optimización de tiempo y recursos.

La metodología para alcanzar el objetivo implica tres etapas. En primer lugar, se procede con una revisión bibliográfica detallada de los algoritmos balanceadores de carga con el fin de seleccionar los más adecuados para el estudio. En segundo punto, se diseña e implementa un escenario experimental usando servidores web en máquinas virtuales. Esto permite medir y analizar los resultados obtenidos por los algoritmos seleccionados. Finalmente, se compara su rendimiento y se identifica el que presenta el mejor desempeño.

El proyecto busca identificar los principales problemas que afectan el manejo del tiempo y los recursos en servidores web con máquinas virtuales. También se busca generar un análisis comparativo de los diferentes algoritmos balanceadores de carga. De esta forma, se determinará cuál es el más eficiente para la optimización de tiempo y recursos en servidores web con máquinas virtuales.

Palabras clave: Servidores web, tiempo, recursos, máquinas virtuales, algoritmos, balanceador de Carga.

ABSTRACT

Small companies in Ecuador that use web servers often face difficulties in efficiently managing time and resources. These difficulties negatively impact the company's performance as well as the quality of service provided to customers. Thus, it is proposed to conduct a comparative analysis of different load balancing algorithms that allow the distribution of requests among virtual machines according to various parameters. The primary objective of this project is to determine which algorithm offers the best performance in optimizing time and resources.

The methodology used to achieve this objective consists of three stages. Firstly, a detailed bibliographic review of load balancing algorithms is conducted to select the most suitable algorithms for the study. Secondly, an experimental scenario is designed and implemented using web servers and virtual machines, allowing for the measurement and analysis of the obtained results. Finally, the performance of the selected algorithms is compared, and the algorithm that performs the best is identified.

The project aims to identify the primary challenges faced by small companies regarding the management of time and resources on their web servers that employ virtual machines. The analysis also strives to compare the performance of different load balancing algorithms to determine the most efficient one for optimizing time and resources on such web servers.

Keywords: Web servers, time, resources, virtual machines, algorithms, load balancers.

CAPÍTULO I.- MARCO TEÓRICO

1.1 Tema de Investigación

ANÁLISIS DE ALGORITMOS BALANCEADORES DE CARGA UTILIZANDO MÁQUINAS VIRTUALES PARA LA OPTIMIZACIÓN DE TIEMPO Y RECURSOS EN SERVIDORES WEB.

1.1.1 Planteamiento del Problema

En el mundo con la expansión de internet y la rápida evolución de las tecnologías, los servidores se enfrentan con dificultades en su capacidad de manejar grandes cantidades de datos. Hoy en día la conexión a internet no suele ser un problema mayor, el tiempo de respuesta depende solo del servidor y como este maneja los datos.[1]

Por su parte, la virtualización de servidores y escritorios virtuales es una tecnología usada por medianas y grandes empresas. Se la aplica con el fin de conseguir ahorros económicos y energéticos. Si bien la virtualización no es un invento reciente, con la consolidación del modelo de computación en la nube, esta ha pasado a ser uno de los componentes fundamentales.[2]

No se ha podido encontrar información estadística con respecto a la cantidad de proveedores de internet que usan tecnologías balanceadoras de carga dirigidas a sus clientes, por esta razón se infiere dicha información con la cantidad de búsquedas sobre el tema de virtualización y balanceo de carga que se han realizado los últimos 5 años en América Latina.

La búsqueda en internet con respecto a temas de virtualización en países de América Latina se ha mantenido los últimos 5 años con alrededor de 1400 búsquedas[3], con estos datos se puede inferir que muchas de estas búsquedas recaen en su implementación. Una gran parte del uso son de los proveedores de internet en América Latina que usan tecnologías balanceadoras de carga para asignar las peticiones del usuario en diferentes computadoras dentro de los centros de datos. Este tipo de

tecnologías suelen ser utilizadas para servicios web, FTP, aplicaciones de negocio crítica y entre otras herramientas en línea. Los balanceadores de carga tradicionales, basados en hardware, suelen ser muy costosos y poco flexibles, además de necesitar de un administrador.[4]

Por otra parte, la cantidad de búsquedas de balanceo de carga en Ecuador, se ha mantenido los últimos 3 años, con búsquedas mayores a las 400, sumando ambos términos explicados en el siguiente párrafo.[5] Infiriendo con los datos, en [6] se declara qué, en Ecuador la pequeña empresa ha tenido la necesidad de implementar servicios de gestiones empresariales, mayormente balanceo y aceleración de carga, servicios de backup o de seguridad perimetral en cloud público. Se toman en cuenta tecnologías con bajo costo y fáciles de manejar. La utilización de máquinas virtuales es una buena solución. Sin embargo, si no se cuenta con esquemas de balanceo de cargas, puede llevar a que ciertos nodos trabajen demasiado, mientras que otros se mantengan en modo de reposo.[7], [8]

En términos generales: "Equilibrio de carga" y "Equilibrador de carga" son términos que se refieren a lo mismo: la técnica de distribución de la carga de trabajo entre varios servidores para mejorar el rendimiento, la capacidad de respuesta y la disponibilidad del sistema. La diferencia de terminología puede depender del contexto específico, donde algunas personas pueden preferir usar un término en lugar de otro. En general, se utilizan de manera intercambiable y no hay diferencia en las tecnologías utilizadas para aplicar la técnica.[9] En la **Figura 1** se muestra los gráficos de la cantidad de búsquedas anuales de ambos términos en Ecuador.

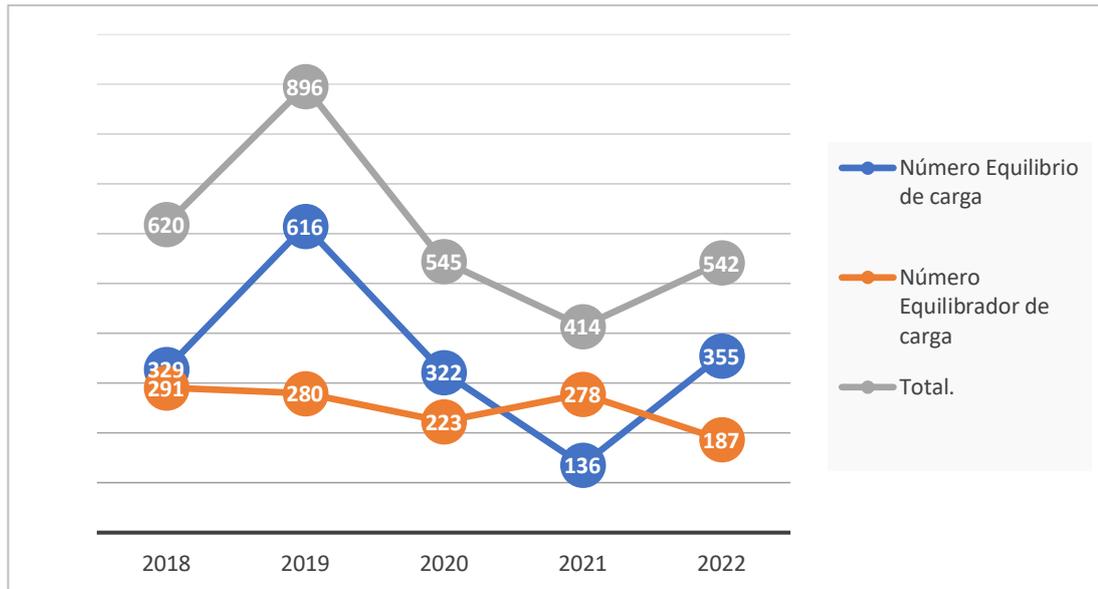


Figura 1. Gráfica de la cantidad de búsquedas Balanceo y Balanceador de carga.

Así mismo, las pequeñas empresas buscan este tipo de soluciones en el balanceo de cargas para sus servicios en línea. Debido a que, si un usuario espera mucho tiempo a que se despliegue una página, este seguramente la abandonará. El alto tiempo de carga será molesto para la experiencia del visitante, cerca del 90% de usuarios se retiran pasados los 2 segundos de espera en carga y el 79% de los mismos no regresarán a dicha página.[10]

En conclusión, con la rápida evolución de las tecnologías y la creciente cantidad de datos que se manejan en línea, los servidores se enfrentan a dificultades en su capacidad para manejar grandes cantidades de carga. La implementación de soluciones de balanceo de carga no solo mejora la experiencia del usuario y previene la pérdida de visitantes en línea, sino que también permite una mejor utilización de los recursos disponibles y una mayor eficiencia en el procesamiento de la carga del servidor.

1.2 Antecedentes Investigativos

En este trabajo, se realizó una revisión de literatura en artículos publicados en las siguientes bases de datos científicas: Springer, IEEE Xplore y ACM Digital Library. Las cadenas de búsqueda se establecieron con las siguientes palabras claves: round robin, random, least connection, Weighted Round Robin, web server, y time or

resources, en el rango de los años 2018 – 2022. Teniendo en cuenta que estos ítems han resultado de la operacionalización de variables. Por consiguiente, la **Tabla 1**. Muestra las cadenas de búsqueda utilizadas y el número de artículos encontrados.

Tabla 1. Cadenas de búsqueda

Elaborado por: Paúl González

	Cadena	Revista	N° artículos
Cadena 1	"Round robin" AND web server AND (time OR resources)	Springer	1400
		IEEE Xplore	8
		ACM Digital Library	1
Cadena 2	"Random" AND web server AND (time OR resources)	Springer	2430
		IEEE Xplore	42
		ACM Digital Library	2
Cadena 3	"Least connection" AND web server AND (time OR resources)	Springer	14
		IEEE Xplore	6
		ACM Digital Library	0
Cadena 4	"Weighted Round Robin" AND web server AND (time OR resources)	Springer	24
		IEEE Xplore	2
		ACM Digital Library	0
Total			3929

El fin de la revisión de literatura es describir los algoritmos balanceadores de carga, que una vez implementados en un web server, disminuyan el tiempo de procesamiento, así como la carga en el sistema.

En este contexto, la revisión de literatura da como resultado 3929 documentos investigativos, donde se encuentran las palabras claves establecidas. Tomando en consideración la gran cantidad de artículos científicos, se generarán criterios de inclusión y exclusión mostrados en la **Tabla 2**, con el fin de reducir la cantidad y obtener únicamente los más acordes con esta investigación.

Tabla 2. Criterios de inclusión y exclusión

Elaborado por: Paúl González

Inclusión	· Artículos que hablen de los algoritmos balanceadores de carga seleccionados.
	· Pruebas en servidores pequeños.
	· Uso de software libre para la experimentación.
	· Artículos que sean de ciencia computacional e ingeniería
Exclusión	· No son de idioma Inglés o Español
	· Aplicados en fog computing.
	· Artículos que son enfocados a redes neuronales.
	· No muestran los resultados de manera clara o están enfocados a otros fines y el implemento de estas herramientas es parte del proceso.
	· Investigaciones que están enfocadas a la seguridad de los servicios.
	· Artículos sobre particionamiento de recursos con multi colas.

De los 3929 documentos se descartaron 14 ya que estaban en idiomas diferentes al Inglés dejando 3915. Se excluyeron 2068 ya que no eran artículos o eran libros que tienen costo para poder acceder a ellos dejando un total de 1847. 108 se excluyeron porque no se hablan a cerca de fog computing. Dejando un total de 1739 artículos restantes de los cuales se discriminaron 1598. Los 1598 son discriminados porque no muestran de manera clara los resultados o el uso de estas herramientas son parte de otros procesos, por lo tanto, se desvían de lo que se necesita. Del análisis anterior quedan 141 artículos. De los 141 artículos, se trabajó con 8 artículos que cumplen todos los criterios de inclusión necesarios para la investigación del estudio.

En [11], Rathore describe un estudio que se realizó con el fin de probar cuatro algoritmos diferentes de equilibrio de carga para un sistema Peer-to-Peer (P2P). El estudio se llevó a cabo en un entorno de red de prueba que constaba de siete máquinas virtuales (VM) en 3 servidores distintos con la misma configuración, dentro de los cuales están instalados una distribución Ubuntu minimal, además cada servidor incluye un equilibrador de carga, servidores y clientes. Los cuatro algoritmos probados fueron Least Clients, Round Robin, Least Load y Fastest First. Se midieron los tiempos de ejecución de cada algoritmo los cuales dieron como resultado que Round Robin es el que más rápido responde con un tiempo promedio de 9.751ms en 3 ambientes de prueba distintos. Los otros algoritmos obtuvieron tiempos promedios de 130.615ms,

122.862ms, 122.771ms (Least Clients, Least Load, Fastest First respectivamente). El estudio concluye que, si bien cada uno tiene un propósito específico y no hay una herramienta que se considere la “mejor”, Round Robin es una buena opción para la selección de clientes, ya que los usuarios conectados a la red tienen una carga despreciable. Sin embargo, puede que no sea la mejor opción para equilibrar la red, puesto que podría sobrecargar al siguiente servidor en la línea cuando se redirige el tráfico. El algoritmo Least Clients puede encontrar un inconveniente similar, pero no en la misma medida, mientras que los otros dos no son propensos a este problema en absoluto. En general, el estudio proporciona información valiosa sobre los algoritmos de equilibrio de carga para los sistemas P2P y puede informar el diseño más efectivo.

Babbar, Parthiban, Radhakrishnan y Rani en [12] hablan del porqué es importante la utilización de los algoritmos balanceadores, para resolver el problema de sobrecarga en servidores web. La arquitectura que se usa en la investigación tiene un CPU Intel i7-855U, 16GB de memoria RAM; Sistemas operativos Ubuntu 18 y Windows 10 con controlador de máquinas virtuales Virtual Box Oracle v6. En este estudio se abarca 6 herramientas balanceadoras de carga, las cuales se pusieron a prueba dentro de un entorno simulado de aplicaciones por medio de la red. Mostrando así el rendimiento en porcentaje en comparación de GLBA a los algoritmos Round Robin, Weighted Round Robin, Server Load, LBBSRT y Dynamic Server. Se tiene como resultado los porcentajes de comparación que se representa en promedio: 74.66%, 18.66%, 68.33%, 64.33%, 75.16% respectivamente. La investigación está implementada en Python utilizando el emulador Mininet y la herramienta de prueba de carga abierta.

En [13] Arreola y Jiménez analizaron dos algoritmos balanceadores de carga (Aleatorio y Round-robin) en una red SDN bajo la supervisión del controlador Ryu. Se usaron switches virtuales en microprocesadores Raspberry Pi 3 B+ y 4 bajo el sistema operativo Raspbian. Los resultados mostraron que el desempeño del balanceador Round-robin superó al Aleatorio en todas las métricas analizadas. En el análisis de porcentaje de paquetes perdidos en el rango de 4 a 200 peticiones, Round Robin presentó un 55% menos pérdida de paquetes. El algoritmo aleatorio alcanza 15% menos paquetes transferidos por unidad de tiempo. Por último, el algoritmo Round Robin tiene un promedio de 10% más información transmitida en un canal por

unidad de tiempo. El balanceador Aleatorio también tiene su utilidad en redes con menor cantidad de nodos. Se sugiere que se puedan implementar algoritmos más complejos en trabajos futuros, considerando factores como el número de peticiones por minuto, la memoria de cada servidor, o la cantidad de procesos que tengan. También se puede agregar un mayor número de servidores o modificar la topología de la red para visualizar los efectos en los tipos de balanceadores mencionados.

En [14] Chitra y Rhymend presentan un algoritmo de asignación de tareas mejorado con round robin ponderado que tiene en cuenta las capacidades de cada máquina virtual (VM) y la duración de las tareas solicitadas para asignarlas a las VM más adecuadas teniendo en cuenta las altas características de los servidores en los que se probó. Los detalles de la configuración son: 500 hosts (200 Nos-4-CoreHyperThreadedOpteron270, 200 Nos-4-Core HyperThreadedOpteron2218 and 100 Nos 8-Core HyperThreaded XeonE5430); Memoria RAM de los hosts: 16/32GB; Número de VM hasta 100 por host. El algoritmo se divide en tres etapas diferentes para manejar distintos escenarios en el ciclo de vida del entorno. El planificador estático se encarga de la colocación inicial de las tareas y las distribuye de igual forma en las VM participantes según sus capacidades y la duración de la tarea solicitada. El planificador dinámico considera la carga de todas las VM configuradas y el tiempo tentativo de finalización de la carga actual junto con el tiempo estimado de finalización de la tarea llegada en cada VM configurada. Después de esto, se identifica el tiempo de finalización mínimo posible para esta tarea en una de las VMs y se le asigna la tarea a esta VM. El balanceador de carga se ejecuta al finalizar cada tarea para distribuir siempre las cargas de manera uniforme en todas las VMs, eliminando así cualquier tiempo inactivo en los recursos participantes. Se demostró que el algoritmo mejorado de round robin ponderado es el más adecuado para trabajos tanto heterogéneos como homogéneos con recursos heterogéneos. El análisis de rendimiento y los resultados experimentales demostraron que considera el tiempo de respuesta como el parámetro principal de calidad del servicio.

Cubas en [15] compara distintas herramientas open source para realizar pruebas de carga y estrés, simulando la capacidad máxima que tiene el servidor web, para atender un conjunto de usuarios de manera simultánea y de esa manera medir el rendimiento

de los servidores web. Se compararon las características de varias herramientas de código abierto para seleccionar la herramienta JMETER como la mejor opción para ejecutar las pruebas de carga y estrés en la investigación debido a sus constantes actualizaciones, compatibilidad con plataformas, soporte de idiomas y pocos requerimientos de uso. Se desarrolló un Aplicativo web y se configuró en cada nodo para evaluar el rendimiento de los servidores web bajo la métrica eficiencia de la norma ISO 9126. Se evaluaron las medidas y se identificó que el servidor web IIS tiene un mejor rendimiento que el montado con NGINX en las pruebas de carga, estrés y esfuerzo. IIS pudo soportar entradas de 10000 usuarios concurrentes en un tiempo máximo promedio de 3126ms, mientras que el servidor NGINX tardó un tiempo promedio de 488ms para soportar la misma carga.

P. Kumar y R. Kumar en [16] muestran que el balanceo de carga de tareas en máquinas virtuales es un desafío importante en la computación en la nube y ha recibido mucha atención por parte de los investigadores. Este artículo es una revisión del estado del arte de los problemas y desafíos del equilibrio de carga. Según este estudio, se ha realizado una extensa encuesta sobre numerosas técnicas de equilibrio de carga considerando diferentes métricas. Basados en esto, los autores clasifican las técnicas de balanceo en varias categorías y discuten sus ventajas, desventajas, conceptos y desafíos. Se destaca la necesidad de perfeccionar las técnicas existentes para mejorar el rendimiento y la gestión de carga de tareas en el futuro, así como desarrollar nuevos algoritmos que también tengan en cuenta la eficiencia energética y la gestión de tareas. Los resultados y conclusiones de este estudio pueden ayudar a los investigadores a identificar problemas en el campo del equilibrio de carga y proporcionar un resumen de las técnicas disponibles.

Jyoti, Shrimali, Tiwari y Pratap en [17] discuten los problemas que existen en el entorno de la computación en la nube, como uno de estos el balanceo de carga. Hace una comparación de 5 algoritmos balanceadores: Round Robin, Weighted Round Robin, Least Connection, Weighted Least Connection y Random, declaradas como las técnicas más predominantes. Todo con la finalidad de obtener un buen funcionamiento de los servicios en la nube. También se discuten las políticas de estos, la gestión de calidad y la optimización de los recursos. Se proponen métricas con el fin de evaluar

el rendimiento del sistema y se presentan soluciones para mejorar la eficiencia y la calidad de los servicios en la nube.

Chiang, Cheng, Liu y Ching-Yi Chiang, en [18] toman como punto de partida una red definida por software (SDN, por sus siglas en inglés) la cual es una forma de gestionar una red a gran escala de manera flexible. Con el fin de mejorar la eficiencia y la disponibilidad de los servicios, se utilizaron agrupamiento de servidores basados en Open Flow. Los servidores utilizados se componen de 2 host Dell PowerEdge R730 con 80 Gb de memoria; Un host de máquinas virtuales con Ubuntu 16.04 y 1 Gb de memoria; Un controlador Open Flow i7-6700 3.4Ghz con Sistema Ubuntu 16.04; Un Switch virtual Open Flow con un CPU i5-4590 3.30 GHz con Sistema Ubuntu 14.04 y 12 Gb de memoria; Y un cliente con un CPU i5-4590 a 3.30 GHz Con sistema Ubuntu 16.04 y 8 Gb de memoria. Para el manejo de las cargas se usaron distintos algoritmos balanceadores ya existentes: Least Connection, Weighted Round-Robin, Round-Robin, Least Link Load; Y un adaptado para el problema propuesto por los investigadores: Dynamic Weighted Random Selection. Antes de los experimentos, se mide el rendimiento de una MV para determinar el rendimiento de cada servidor. Se muestran los sistemas en los cuales los algoritmos se montaron y la configuración que les dan a estos, aparte de presentar los resultados obtenidos tras la ejecución de peticiones a los servidores.

Tabla 3. Cuadro comparativo de Algoritmos Balanceadores de Carga más usados

Elaborado por: Paúl González

Algoritmo	Descripción	Ventajas	Desventajas	Rendimiento	Porcentaje de mejora
Round Robin	Distribuye las solicitudes a través de los servidores disponibles de forma circular.	Fácil de implementar y justo para todos los servidores.	No considera la carga del servidor ni la capacidad disponible.	Bueno para un número pequeño de servidores; Puede llevar a una sobrecarga en servidores menos potentes.	Hasta un 50% de mejora en el rendimiento para sistemas con un número limitado de servidores.
Weighted Round Robin	Similar al Round Robin, pero asigna un peso a cada servidor para dar prioridad a los más potentes.	Permite equilibrar mejor la carga, asignando más solicitudes a los servidores más capaces.	El peso asignado a cada servidor puede ser difícil de determinar y ajustar.	Mejora la distribución de carga; Puede llevar a una sobrecarga en servidores menos potentes.	Hasta un 80% de mejora en el rendimiento en sistemas con una amplia gama de servidores.
Least Connection	Envía las solicitudes al servidor con menos conexiones activas en ese momento.	Distribuye la carga de forma más equilibrada, evitando que un servidor se sobrecargue.	No tiene en cuenta la capacidad o recursos del servidor, por lo que no es adecuado para servidores con diferentes características.	Muy eficiente para grandes conjuntos de servidores; Requiere un equilibrio constante de conexiones para mantener la distribución de carga.	Hasta un 90% de mejora en el rendimiento en sistemas con una alta variación de carga.
Random	Elige un servidor al azar para manejar cada solicitud.	Simple y fácil de implementar.	Puede llevar a una distribución desequilibrada de la carga.	Buen rendimiento para sistemas con una distribución uniforme de carga; Puede generar una carga dispar en servidores.	Mejora limitada en el rendimiento en sistemas con una alta variación de carga.

En la **Tabla 3** se han elegido diferentes algoritmos de balanceo de carga en función de sus ventajas y desventajas, así como su rendimiento y porcentaje de mejora en el sistema. Se ha seleccionado el algoritmo de Round Robin, por su facilidad de implementación y distribución justa de solicitudes a todos los servidores, aunque puede llevar a una sobrecarga en servidores menos potentes. Se ha optado por Weighted Round Robin, porque permite equilibrar mejor la carga, aunque puede ser difícil de determinar y ajustar el peso asignado a cada servidor. También se ha utilizado Least Connection, porque distribuye la carga de forma más equilibrada, evitando que un servidor se sobrecargue, pero no tiene en cuenta la capacidad o recursos del servidor. Finalmente, se ha incluido el algoritmo Random, porque es simple y fácil de implementar, aunque puede llevar a una distribución desequilibrada de la carga. En resumen, se ha seleccionado cada algoritmo en función de su adecuación y su porcentaje de mejora en el sistema, para determinar cuál es el más apropiado para usarse en la investigación.

1.3 Fundamentación Teórica

Tipos de balanceado de carga

Existen dos formas de balanceo de carga: balanceo de carga estático y balanceo de carga dinámico. En el primer caso, la distribución de las tareas se realiza al comienzo de la computación, lo cual permite al nodo maestro participar en la computación una vez que haya asignado una fracción del trabajo a cada nodo esclavo.[19]

- **Balanceo de carga estático**

Los algoritmos estáticos son apropiados para los sistemas con baja variación en carga. En algoritmos estáticos el tráfico está dividido uniformemente entre los servidores. Este algoritmo requiere un previo conocimiento de los recursos del sistema ya que el rendimiento de los procesadores está determinado al inicio de la ejecución de una aplicación, por lo tanto, la decisión de ir cambiando la carga no depende del estado actual del sistema.[20]

Teniendo en cuenta que la comunicación entre servidores no siempre será óptima y que las cargas en solicitudes a los mismos pueden variar, dificulta incorporar la variable retardo de comunicación en el balanceo de carga estático.

- **Balanceo de carga dinámico**

En los algoritmos dinámicos los servidores más libres en toda la red o sistema son los elegidos para el balanceo de carga. Se toma en cuenta el estado actual del sistema para tomar decisiones de la administración de la carga. Gracias a esto, se permite que los procesos se envíen entre una máquina con mucha carga a otra que esté siendo menos usada, todo esto de manera automática.[20]

En este tipo de balanceo, las tareas se reparten entre los procesadores durante la ejecución del programa. Dependiendo de dónde y cómo se almacenen y repartan las tareas el balanceo de carga dinámico se divide en:

- a) Balanceo de carga dinámico centralizado. Se corresponde con la estructura típica de Maestro/Esclavo.
- b) Balanceo de carga dinámico distribuido o descentralizado. Se utilizan varios maestros y cada uno controla a un grupo de esclavos.[21]

Métodos de balanceo de carga

El Load Balancing es de gran ayuda para las granjas de servidores, debido a que su implementación les permite optimizar recursos y manejar pesadas y poderosas aplicaciones de una forma distribuida, garantizando una alta disponibilidad y tolerancia a fallos.

El balanceo de carga distribuye el tráfico IP a múltiples copias o instancias de servicios TCP/IP, cada uno corriendo en un host dentro de una granja de servidores o clústeres refiriéndose a ambos como una agrupación de servidores. Se realizan particiones transparentes de las peticiones de los clientes a través de los hosts y se deja que los clientes accedan a la granja de servidores utilizando una o más direcciones IP virtuales.

Desde el punto de vista del cliente, la granja parece ser un solo servidor que responde a sus peticiones. Si el tráfico se incrementa, el administrador de red simplemente conecta otro servidor a la granja.[22]

Los métodos de balanceo de carga pueden ser de tres tipos:

- **Balanceadores de carga de aplicación**

Toma decisiones de enrutamiento en la capa de aplicación (HTTP/HTTPS), Protocolo de Transferencia de Hiper Textos, admite el enrutamiento basado en rutas y puede dirigir solicitudes a uno o varios puertos de cada instancia del contenedor del clúster. Estos admiten el mapeo de puertos de host dinámico.[23]

- **Balanceadores de carga en red**

Toma las decisiones de enrutamiento en la capa de transporte (TCP/SSL), Protocolo de Control de Transmisión / Capa de Sockets Seguros. Puede atender millones de solicitudes por segundo. Una vez que el balanceador de carga ha recibido una conexión, selecciona un destino del grupo de destinos para la regla predeterminada por medio de un algoritmo hash de flujo de direccionamiento. Intenta abrir una conexión TCP con el destino seleccionado en el puerto especificado en la configuración del agente de escucha. Reenvía la solicitud sin modificar los encabezados. Admiten el mapeo de puertos de host dinámico.[24]

- **Balanceadores de carga clásicos**

Automáticamente distribuye el tráfico entrante de múltiples clientes, como pueden ser las instancias EC2, contenedores y direcciones IP. Monitorea el estado de los clientes conectados, las rutas de tráfico de las conexiones estables. Esta se balancea automáticamente en respuesta a los cambios de tráfico existente.[25]

Algoritmos balanceadores de carga

Los diferentes algoritmos son:

- **Random**

Distribuye aleatoriamente las cargas a través de los servidores disponibles.

- **Round Robin**

Este algoritmo envía todas las nuevas peticiones a los siguientes servidores en línea, eventualmente distribuyendo las conexiones por medio de un array de máquinas.

- **Fastest**

Envía las nuevas conexiones basándose en el tiempo más rápido de respuesta de cada servidor.

- **Least Connection**

Con este método, el sistema envía una nueva conexión al servidor que tiene menor número de conexiones simultaneas.

- **Observed**

Usa una combinación de los algoritmos “Least Connection” y “Fastest” para poder categorizar en que orden se pueden usar los servidores acordes a sus conexiones y tiempo de respuesta.

- **Predictive**

Usa el mismo método del algoritmo “Observed” pero con la diferencia que analiza cada servidor a través del tiempo con el fin de conectar a los servidores con mayor rendimiento.[26]

- **Weighted Round Robin**

Muy similar al tradicional RR, la diferencia es que este algoritmo considera el peso de cada nodo. Estos pesos son asignados por el desarrollador y tienen su base en la más alta capacidad de la máquina virtual. Esta herramienta es de gran utilidad para medir los tiempos de espera, sin embargo, no toma en cuenta la longitud de cada tarea. [27]

Virtualización

Virtualización es una tecnología de software que permite ejecutar varias máquinas virtuales en una única máquina física, compartiendo los recursos de ese ordenador físico entre dos o más entornos diferentes y aislados. Las máquinas virtuales diferentes consiguen ejecutar sistemas operativos diferentes e incontables aplicaciones en el mismo ordenador físico. Así como un servidor físico, la máquina virtual tiene placa madre, procesador, memoria, disco e interface de red. Sin embargo, esos recursos están disponibles y limitados por el servidor físico que los gestiona.[28]

La virtualización es una tecnología que divide un ordenador en diversas máquinas independientes que pueden soportar diferentes sistemas operativos y aplicaciones siendo ejecutados concurrentemente.[29]

Sistemas operativos

Software cuya labor es administrar todos los dispositivos de una computadora y proporcionar una interfaz más sencilla a los programas de usuario para comunicarse con el hardware.[30]

Un sistema operativo es un programa que controla la ejecución de los programas de aplicación y que actúa como interfaz entre el usuario de un computador y el hardware de esta. Puede considerarse que un sistema operativo tiene tres objetivos o lleva a cabo tres funciones:

- **Comodidad:** Un sistema operativo hace que un computador sea más cómodo de utilizar.
- **Eficiencia:** Un sistema operativo permite que los recursos de un sistema informático se aprovechen de una manera más eficiente.
- **Capacidad de evolución:** Un sistema operativo debe construirse de modo que permita el desarrollo efectivo, la verificación y la introducción de nuevas funciones en el sistema y, a la vez, no interferir en los servicios que brinda.

El sistema operativo como interfaz usuario/computadora

El programa de sistemas más importante es el sistema operativo. El sistema operativo oculta al programador los detalles del hardware y le proporciona una interfaz cómoda para utilizar el sistema. Actúa como mediador, facilitándole al programador y a los programas de aplicación el acceso y uso de todas esas características y servicios.

De forma resumida, un sistema operativo ofrece servicios en las áreas siguientes:

- **Creación de programas:** El sistema operativo ofrece una variedad de características y servicios, tales como los editores y los depuradores también llamados debuggers, para ayudar al programador en la creación de programas.
- **Ejecución de programas:** Para ejecutar un programa se necesita un cierto número de tareas. Las instrucciones y los datos se deben cargar en la memoria principal, los archivos y los dispositivos de E/S se deben inicializar y se deben preparar otros recursos.
- **Accesos a los dispositivos de E/S:** Cada dispositivo de E/S requiere un conjunto propio y peculiar de instrucciones o de señales de control para su funcionamiento.

- **Acceso controlado a los archivos:** En el caso de los archivos, el control debe incluir una comprensión, no sólo de la naturaleza del dispositivo de entrada y salida “E/S”, sino del formato de los archivos y del medio de almacenamiento.
- **Acceso al sistema:** En el caso de un sistema compartido o público, el sistema operativo controla el acceso al sistema como un todo y a los recursos específicos del sistema.
- **Detección y respuesta de errores:** Cuando un sistema informático está en funcionamiento pueden producirse varios errores.
- **Contabilidad:** Un buen sistema operativo debe recoger estadísticas de utilización de los diversos recursos y supervisar los parámetros de rendimiento tales como el tiempo de respuesta.

El sistema operativo como administrador de recursos

Un computador es un conjunto de recursos para el traslado, almacenamiento y proceso de datos y para el control de estas funciones. El sistema operativo es el responsable de la gestión de estos recursos.

- El sistema operativo funciona de la misma manera que el software normal de un computador, es decir, es un programa ejecutado por el procesador.
- El sistema operativo abandona con frecuencia el control y debe depender del procesador para recuperarlo.[30]

Optimización de tiempo y recursos

La administración de recursos es un tema relacionado usualmente con los mecanismos de sistemas operativos. Toda la responsabilidad se deja a los autores o arquitectos de sistemas operativos. La mayoría de las veces es requerido que el sistema operativo maneje todos los recursos, sin embargo, se suele dejar en ciertos puntos, que la aplicación maneje dichos recursos con el fin que se pueda sacar el máximo partido a los componentes.

Con el uso continuo de aplicaciones web, los servidores suelen encontrarse con enormes cargas de peticiones hechas por clientes alrededor del mundo. Estas cargas conllevan a que, en los sistemas operativos modernos, el uso de los recursos del sistema sea considerado parte de la solución al problema. La administración de recursos se realiza eficientemente a los niveles más bajos del sistema operativo, al nivel de kernel, donde puede ser configurado según el uso que se le dará al sistema.[31]

1.4 Objetivos

1.4.1 Objetivo General

Establecer un análisis entre algoritmos balanceadores de carga utilizando máquinas virtuales para la optimización de tiempo y recursos en servidores web.

1.4.2 Objetivos Específicos

- Determinar los problemas que enfrentan los servidores en máquinas virtuales con respecto al manejo de tiempo y recursos.
- Evaluar los algoritmos balanceadores de carga en servidores web según el tiempo de respuesta y su carga dentro del sistema.
- Identificar el algoritmo balanceador de carga que mejor performance presente con respecto al tiempo y carga computacional del sistema.

CAPÍTULO II.- METODOLOGÍA

2.1 Materiales

Debido a la naturaleza del proyecto de investigación se utilizó cadenas investigativas para la recolección de información descritas en la sección **1.2 Antecedentes Investigativos**.

2.2 Métodos

2.2.1 Modalidad de la Investigación

- **Investigación Bibliográfica-Documental**

La investigación será Bibliográfica-Documental debido a que se fundamenta en la recolección de datos a través de artículos, revistas, tesis realizadas, etc., los que se utilizarán para la construcción de la fundamentación teórica que estudiará temas a cerca de: algoritmos balanceadores de carga, arquitectura de servidores, servicios web, entre otros.

- **Investigación Experimental**

La investigación será experimental puesto a que se usarán distintos algoritmos especializados en balanceo de carga dentro de máquinas virtuales con sistemas open source, con la finalidad de escoger el algoritmo que mejor optimice los recursos computacionales y reduzca el tiempo de respuesta de los servidores.

2.2.2 Población y Muestra

Para esta investigación se considera como población el número de instancias de conexión hacia una máquina que aloja servidores virtuales. En este sentido, la población es considerada como desconocida. Con respecto a la muestra, utilizaremos

la desigualdad de Hoeffding simplificada para obtener la cantidad mínima de instancias que sirvan para testear los algoritmos balanceadores de carga.

$$P(|\bar{x} - \mu| > E) \leq e^{(2e^2N)}$$

Donde:

P = representa la probabilidad de que la diferencia entre la media muestral (\bar{x}) y la media poblacional (μ) sea mayor que un cierto valor E

\bar{x} = media de n muestras de una variable aleatoria.

μ = valor esperado verdadero de la variable aleatoria.

E = es la cota superior deseada del error absoluto de la media.

e = es el tamaño de la desviación estándar de x .

e^2 = es la base del logaritmo natural.

N = es el número de la población que se quiere estimar

El término izquierdo de la igualdad se la puede remplazar con un grado de confianza del 99.99% la cual se puede representar con dos desviaciones estándar o cuatro σ (sigma) el cual da como resultado 7.84.

$$7.84 \leq e^{(2e^2N)}$$

Ahora para poder obtener el valor de N se procede a despejar, con la finalidad de poder tener un valor de población.

Aplicamos el logaritmo natural en ambos lados de la desigualdad para eliminar el exponente de e y obtener:

$$\ln(7.84) \leq \ln(e^{2e^2N})$$

Utilizamos la propiedad de los logaritmos que dice que $\ln(a^b) = b \ln(a)$ para obtener:

$$\ln(7.84) \leq 2e^2N \ln(e)$$

Como $\ln(e) = 1$, podemos simplificar la expresión anterior a:

$$\ln(7.84) \leq 2e^2 N$$

Despejamos N dividiendo ambos lados de la desigualdad por $2e^2$ y haciendo uso del inverso aditivo de ln para obtener:

$$N \geq (\ln(7.84))/(2e^2)$$

Utilizando la fórmula y considerando un error absoluto del 1%, se ha obtenido un valor poblacional de **10,296**. Cabe recalcar que los datos obtenidos empíricamente en la fórmula, se asemejan a los obtenidos de manera real y demostrados en las **imágenes 2, 3 y 4**.

El valor obtenido por medio de la desigualdad servirá como referencia para la investigación experimental sobre la cantidad de conexiones que manejan los servidores de la pequeña empresa de Ecuador. Se utilizará una muestra superior a este tamaño para evaluar el rendimiento de los servidores antes y después de la aplicación de algoritmos balanceadores de carga. Se realizarán pruebas de estrés para simular una población real enviando solicitudes a los servidores y así determinar su capacidad y eficiencia.

A continuación, se presenta 3 gráficos sobre el número de instancias conectadas a servidores de empresas pequeñas en el Ecuador. Las **Figuras 2, 3 y 4** muestran esta

cantidad de interacciones con cada página web. Cabe resaltar que se ha pedido el anonimato con respecto a los datos brindados, por esta razón a las empresas se las tratará como tal, sin mostrar sus nombres.

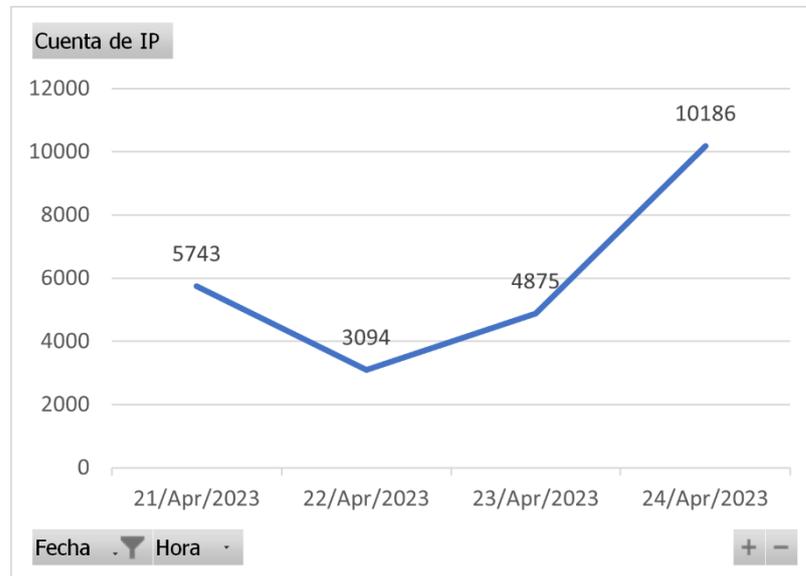


Figura 2. Número de ingresos diarios, Empresa 1

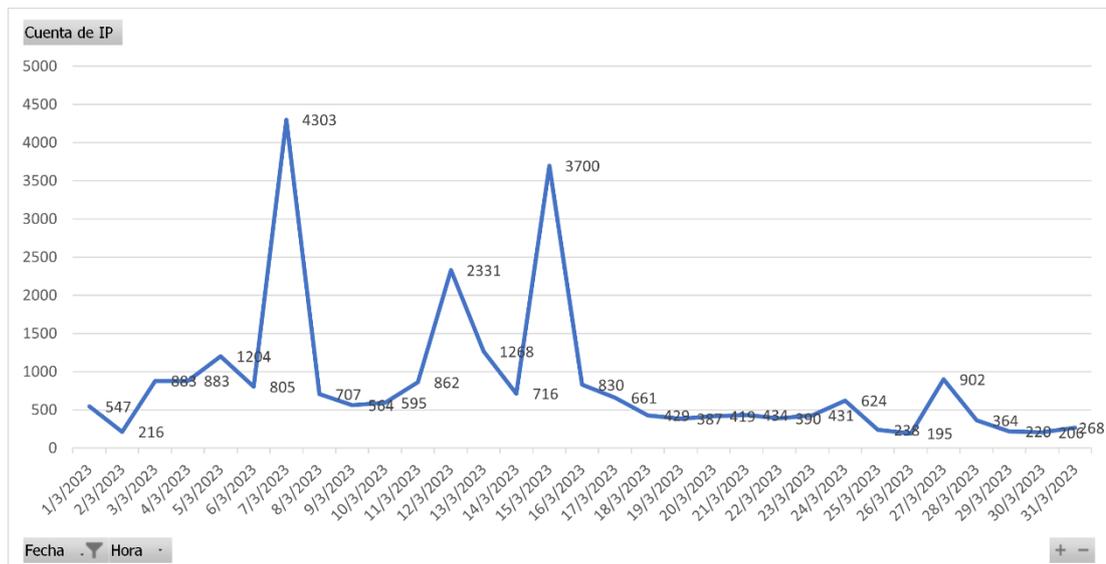


Figura 3. Número de ingresos diarios, Empresa 2

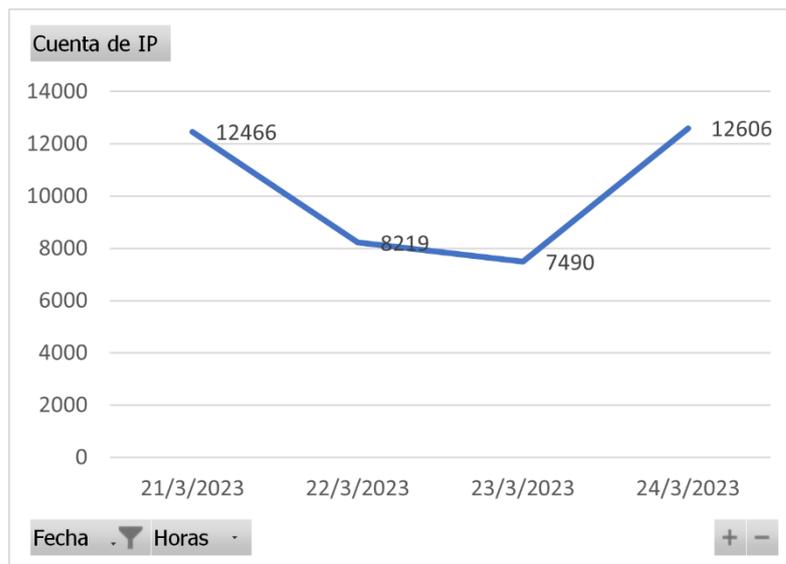


Figura 4. Número de ingresos diarios, Empresa 3

La **Figura 5** y las **Tablas 4 – 9** muestran estadísticas sobre el número de usuarios que utilizan la plataforma de Steam de la corporación Valve en tiempo real[32]. Estos datos estadísticos son utilizados para analizar las características de las computadoras a nivel mundial y, como punto de partida, armar una arquitectura de computadora para probar algoritmos balanceadores de carga en servidores virtuales.

La **Figura 5** muestra el número de conexiones simultáneas que tiene la plataforma de Steam alrededor del mundo con picos superiores a los 31 millones de usuarios:



Figura 5. Usuarios usando la plataforma Steam en tiempo real[33]

La **Tabla 4** muestra el porcentaje de las características de ordenadores de los usuarios que acceden a la plataforma:

Tabla 4. Características de los ordenadores con la cantidad representada en porcentajes.[34]

Elaborado por: Paúl González

Ítem	Más popular	Porcentaje de uso	Cambio con respecto a meses anteriores
Versión de Sistema Operativo	Windows 10 - 64 bits	61,21%	-12,74%
Memoria RAM	16 GB	52,19%	-4,73%
Velocidad de CPU	2.3 GHz a 2.69 GHz	20,65%	1,55%
CPUs físicas	6 CPUs	33,49%	-12,29%
Tarjeta Gráfica	NVIDIA GeForce GTX 1650	5,98%	2,02%
RAM virtual	8 GB	27,90%	-4,26%
Resolución de Pantalla Principal	1920 x 1080	64,52%	0,24%
Espacio Libre en Disco	100 GB a 249 GB	25,65%	6,26%
Espacio Total en Disco	Más de 1 TB	49,32%	-18,61%

La **Tabla 5** muestra los porcentajes de la cantidad de memoria RAM que tienen los dispositivos conectados a la plataforma. [34]

Tabla 5. Memoria RAM de los ordenadores

Elaborado por: Paúl González

Memoria RAM		
Detalle	Porcentaje	Cambio
< 4 Gb	0.15%	0,06%
4	5.00%	1,96%
8	19.35%	6,66%
12	3.49%	1,18%
16	52.19%	-4,73%
24	1.16%	0,27%
32	16.10%	-6,31%
64	1.57%	0,55%
> 64 Gb	0.16%	0,07%
Otros	0.84%	0,31%

La **Tabla 6** muestra los porcentajes de las velocidades de CPU que tienen los dispositivos conectados a la plataforma. [34]

Tabla 6. Velocidades de CPU de los ordenadores

Elaborado por: Paúl González

Velocidad de CPU		
Detalle	Porcentaje	Cambio
< 1,4 GHz	1,21%	0,49%
1,4 GHz - 1,49 GHz	0,34%	0,14%
1,5 GHz - 1,69 GHz	0,40%	0,16%
1,7 GHz - 1,99 GHz	1,83%	0,71%
2,0 GHz - 2,29 GHz	3,50%	-0,63%
2,3 GHz - 2,69 GHz	20,65%	1,55%
2,7 GHz - 2,99 GHz	12,99%	-12,14%
3,0 GHz - 3,29 GHz	7,24%	-1,27%
3,3 GHz - 3,69 GHz	14,74%	1,77%
> 3,7 GHz	3,41%	0,08%

La **Tabla 7** muestra la cantidad de CPUs físicas que tienen los dispositivos conectados a la plataforma. [34]

Tabla 7. Número de CPUs físicas de los ordenadores

Elaborado por: Paúl González

Número de CPUs		
Detalle	Porcentaje	Cambio
1 CPU	0,12%	0,04%
2 CPU	8,75%	3,34%
3 CPU	0,36%	0,10%
4 CPU	27,70%	6,76%
5 CPU	0,02%	0,00%
6 CPU	33,49%	-12,29%
7 CPU	0,00%	0,00%
8 CPU	19,60%	1,15%
9 CPU	0,01%	0,01%
10 CPU	2,10%	0,25%
11 CPU	0,00%	0,00%
12 CPU	3,90%	-0,72%

13 CPU	0,00%	0,00%
14 CPU	2,16%	0,72%
15 CPU	0,00%	0,00%
16 CPU	1,34%	0,39%
18 CPU	0,02%	0,01%
20 CPU	0,01%	0,00%
22 CPU	0,00%	0,00%
24 CPU	0,40%	0,21%
28 CPU	0,00%	0,00%

La **Tabla 8** muestra la cantidad de espacio libre de almacenamiento que tienen los dispositivos conectados a la plataforma. [34]

Tabla 8. Espacio libre en disco de los ordenadores

Elaborado por: Paúl González

Espacio libre en disco		
Detalle	Porcentaje	Cambio
< 10 Gb	1,11%	0,42%
10 Gb - 99 Gb	18,80%	6,27%
100 Gb - 249 Gb	25,65%	6,26%
250 Gb - 499 Gb	21,92%	3,87%
500 Gb - 749 Gb	10,83%	1,21%
750 Gb - 999 Gb	8,61%	1,29%
1 Tb - 2 Tb	7,81%	-2,56%
2 Tb - 3 Tb	1,68%	-3,44%
3 Tb - 4 Tb	1,35%	-2,87%
> 4 Tb	2,23%	-10,46%

La **Tabla 9** muestra la cantidad total de espacio de almacenamiento que tienen los dispositivos conectados a la plataforma. [34]

Tabla 9. Espacio total de almacenamiento de los usuarios

Elaborado por: Paúl González

Espacio total en disco		
Detalle	Porcentaje	Cambio
< 10 Gb	0,00%	0,00%
10 Gb - 99 Gb	0,26%	0,11%
100 Gb - 249 Gb	8,54%	3,47%
250 Gb - 499 Gb	23,18%	8,26%
500 Gb - 749 Gb	4,08%	1,48%
750 Gb - 999 Gb	14,62%	5,29%
> 1 Tb	49,32%	-18,61%

Teniendo en cuenta estos valores se puede estimar los recursos de una máquina y determinar su costo total. La pequeña empresa en el Ecuador puede generar ganancias alrededor de los cien mil dólares anuales.[35] Por esta razón los componentes de las computadoras que se usarán para este estudio tendrán los siguientes componentes:

- 8GB de RAM
- 3.40 GHz
- 8 núcleos de CPU
- 1 Terabyte de espacio en disco

El costo de una computadora de estas características en Estados Unidos ronda por los 965 dólares libre de importaciones, el precio varía con respecto a marcas y distribuidores dentro del país. Este precio no es tan elevado para una empresa pequeña que no puede permitirse comprar computadoras con mejores prestaciones o varias de las mismas.

2.2.3 Recolección de Información

La información requerida ya se encuentra presentada en el documento actual. En particular, la recolección de información ha sido detallada en la sección **1.2 Antecedentes investigativos** y se puede encontrar en la **Tabla 1**. Se sugiere que se refiera a ese capítulo y a dicha tabla para obtener la información necesaria.

2.2.4 Procesamiento y Análisis de Datos

Gracias a la información recolectada de temas similares al planteado se puede decir que:

- Es importante tener en cuenta que actualmente existen varios algoritmos balanceadores de carga, incluso se pueden modificar los existentes con el fin de que se adapten a nuestras necesidades, aunque solo recomendable para servidores que manejen enormes cantidades de datos.
- En el pasado se consideraba solo servidores de tipo Linux por ser versátiles a la hora de configurar, sin embargo, existen alternativas incluyendo Windows server que ha demostrado que puede ser una opción plausible al implemento de prácticas en tema de rendimiento.
- Es de gran ayuda saber que estás herramientas de balanceo de carga son ampliamente utilizadas, tanto en entornos de trabajo grandes como empresas internacionales, o en entornos pequeños como puede ser la implementación hasta en controladores Ryu.
- Se debe tomar en cuenta que algunos algoritmos balanceadores de carga ayudarán con el rendimiento en general a los dispositivos, pero hay que tener en cuenta que en ciertos casos puede generar incremento en el uso de memoria o viceversa disminuir el uso de memoria a coste de aumentar el uso de CPU.
- Es de mucha ayuda saber que las prácticas se han hecho en entornos simulados de manera virtualizada, pero con servidores de altas características, no se ha intentado realizar un análisis comparativo de algoritmos de esta manera en computadores con pocos recursos.
- Es crucial tener en cuenta que, para comparar los algoritmos balanceadores, es necesario realizar pruebas un mínimo de 3 veces cambiando ciertos parámetros con la finalidad de ver el verdadero rendimiento y versatilidad de la herramienta.

- Para llevar a cabo la implementación del proceso de balanceo de carga, es necesario realizar una selección rigurosa entre diversas herramientas y metodologías disponibles, con el fin de determinar cuál es la más adecuada para cumplir con los objetivos y necesidades específicas planteadas en el proceso.
- Es imperativo aplicar algoritmos balanceadores de carga a computadores con pocos recursos, con la finalidad de demostrar su posible mejora al rendimiento y así cumplir los objetivos propuestos.

CAPÍTULO III.- RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

Después de haber investigado en diferentes artículos, los problemas más comunes que se encuentran en servidores web en máquinas virtuales se representan en la **Tabla 10** a modo de resumen:

Tabla 10. Problemas que enfrentan los servidores web en máquinas virtuales

Elaborado por: Paúl González

Problema	Descripción
Alta utilización de la CPU	Esto puede ocurrir cuando el servidor web está bajo una carga pesada o procesando grandes cantidades de datos.
Uso elevado de memoria	Esto puede ocurrir cuando el servidor web está ejecutando muchos procesos en segundo plano o procesando grandes cantidades de datos.
Latencia de E/S de disco	Esto puede ocurrir cuando el disco virtual está fragmentado o cuando el servidor web está ejecutando muchos procesos en segundo plano o procesando grandes cantidades de datos.
Problemas de sincronización de tiempo entre VM	Esto puede ocurrir cuando el tiempo en las VM no está sincronizado correctamente con el host o entre ellas, lo que puede provocar problemas con la programación y la asignación de recursos.
Falta de monitoreo y gestión adecuados	Sin herramientas de monitoreo y gestión adecuadas, puede resultar difícil identificar y resolver problemas de rendimiento antes de que se conviertan en problemas importantes.
Sobredimensionamiento o subdimensionamiento de las VMs	Esto puede ocurrir si los recursos asignados a una VM no son suficientes para su carga de trabajo, o si la VM está sobredimensionada con más recursos de los que necesita, lo que desperdicia recursos.

3.1.1 Análisis comparativo de metodologías Ágiles

La elección de la metodología se llevó a cabo con el objetivo de establecer la forma en que se va a gestionar el desarrollo del proyecto, dadas las limitaciones del equipo de trabajo, los cortos plazos para presentar los entregables, la posibilidad de eventos inesperados y la necesidad de completar tareas en un orden específico.

Metodología Kanban

La metodología Kanban es una herramienta de gestión visual que sirve para ordenar y estructurar el proceso de ejecución de un proyecto de una manera ágil y eficiente.[36] Es una subcategoría de la metodología Agile, y su objetivo es mejorar la productividad y eficacia del equipo mediante la eliminación de desechos y la maximización de los flujos de trabajo.

Kanban se basa en la premisa de que debe haber una limitación en la cantidad de tareas en proceso en un momento dado, lo que evita la sobrecarga del equipo. La metodología se divide en etapas claras y se utilizan tarjetas y un tablero Kanban para visualizar las tareas en cada parte del proceso.

Los principios clave de Kanban incluyen la limitación del trabajo en progreso, la eliminación de desperdicios, la mejora continua y la entrega temprana. Al limitar la cantidad de trabajo en progreso, se puede mejorar la calidad del trabajo y reducir el tiempo de espera en el proceso.

Además, Kanban se enfoca en la mejora continua del proceso, fomentando la retroalimentación constante y la revisión de los resultados para identificar oportunidades de mejora.

Metodología Scrum

La metodología Scrum es un marco de trabajo Agile para la gestión de proyectos. Se enfoca en la entrega de valor de forma iterativa e incremental, en lugar de esperar a tener un producto completo al final del ciclo. Se divide el proceso en Sprints, ciclos de trabajo que suelen durar de dos a cuatro semanas y permiten crear entregables tangibles.

Las etapas de Scrum son:

- 1) Planificación del Sprint, en donde se decide qué se va a trabajar
- 2) Ejecución, se realiza el trabajo en sí
- 3) Revisión, se presenta el trabajo al equipo y se obtiene feedback
- 4) Retrospectiva, se reflexiona sobre lo sucedido y se plantean mejoras.

El equipo de Scrum se compone de tres roles principales:

- Product Owner, que se encarga de definir el objetivo del proyecto y priorizar el trabajo basándose en el valor que genera para el cliente
- Scrum Master, que se encarga de asegurar que el equipo está siguiendo las prácticas de Scrum y se compromete en su mejora continua
- Development Team, que se encargan de llevar a cabo el trabajo del proyecto.

Además existen tres artefactos principales de Scrum: el Sprint Backlog, que es una lista de todas las tareas a realizar en un Sprint, el Product Backlog, que es una lista de todas las funcionalidades del proyecto ordenadas por prioridad, y el Incremento, que es el resultado del trabajo realizado durante el Sprint.

Metodología Lean

La metodología Lean es una filosofía de gestión enfocada en la eliminación de desperdicios o procesos innecesarios en la producción para crear la máxima eficiencia y calidad en el resultado final. Esta metodología se originó en la industria manufacturera japonesa, pero en la actualidad se aplica en una amplia variedad de

contextos, incluyendo la gestión de proyectos, el desarrollo de productos y la atención al cliente.

Los cinco principios clave de la metodología Lean son:

- 1) Identificar valor: El primer principio se enfoca en identificar qué es lo que realmente aporta valor al proyecto o proceso.
- 2) Mapear el flujo de valor: El segundo principio se enfoca en evaluar cada etapa del proceso y eliminar aquellas que no aportan valor o que generan desperdicios.
- 3) Crear flujo: El tercer principio se enfoca en crear un flujo de trabajo continuo para reducir los tiempos muertos y lograr una mayor eficiencia.
- 4) Establecer un Pull: El cuarto principio se enfoca en implementar un sistema "Pull", que permita que la cantidad de trabajo realizado se ajuste a medida que se van obteniendo resultados.
- 5) Buscar la mejora constante: El quinto principio se enfoca en buscar constantemente maneras de mejorar el proceso, estableciendo métricas para medir el éxito en cada etapa y permitiendo la reevaluación constante.

En el contexto del ciclo de vida del producto, la aplicación de la metodología Lean implica diseñarlo de forma eficiente, producirlo con rapidez y calidad, y entregarlo al cliente de manera oportuna. La metodología busca eliminar los desperdicios, que pueden incluir la sobreproducción, inventarios innecesarios, movimientos excesivos, esperas, procesos defectuosos y exceso de procesamiento.

También implica un enfoque en el avance continuo, mediante la retroalimentación constante y la implementación de mejoras graduales y sostenibles.

A continuación se presenta la **Tabla 11** en la cual se resumen estas 3 metodologías de manera detallada:

Tabla 11. Resumen de las metodologías ágiles

Elaborado por: Paúl González

	Scrum	Kanban	Lean
Orígenes	Desarrollo de software	Manufactura Toyota	Manufactura Toyota
Principales características	División del trabajo en Sprints, roles de equipo definidos, reuniones diarias, backlog de producto y sprint	Visualización del flujo de trabajo, limitación de trabajo en progreso, mejora continua	Reducción del desperdicio, mejora continua, flujo de valor
Enfoque	Gestión de proyectos	Gestión de procesos	Gestión de procesos
Papel del equipo	Scrum Master, propietario de producto, equipo de desarrollo	No hay roles específicos	No hay roles específicos
Procesos	Planificación de sprint, reunión diaria de Scrum, revisión de sprint, retrospectiva de sprint	Visualización de Kanban, límites de trabajo en progreso, mejora continua	Identificación del valor, mapeo del flujo de valor, flujo continuo, mejora continua

Dado el tamaño limitado del equipo de investigación y el plazo acotado, se sugiere el uso de la metodología **Kanban** como enfoque efectivo. Esta herramienta aplicada a un proyecto de investigación con un personal y tiempo limitados aporta varias ventajas:

Visualización: Permite la visualización clara y concisa del flujo de trabajo, en especial útil para equipos pequeños. Cada miembro puede ver las tareas en curso, completadas

y próximas, lo que facilita la coordinación y el seguimiento del avance en un entorno donde la comunicación directa es más factible.

Limitación de trabajo en progreso: Kanban establece límites en la cantidad de trabajo que se puede realizar de manera simultánea, así evitando la sobrecarga de trabajo y manteniendo el enfoque en las tareas críticas y urgentes. Al limitar el trabajo en progreso, se puede priorizar las actividades clave y evitar la dispersión de esfuerzos en tareas menos relevantes, maximizando así el uso del breve periodo de investigación.

Mejora continua: Kanban fomenta esta práctica al permitir al equipo reflexionar sobre el flujo de trabajo y hacer ajustes con el fin de mejorar la eficiencia y la calidad. En un proyecto con restricciones de tiempo, es esencial aprovechar cualquier oportunidad para optimizar el proceso y maximizar los resultados en el plazo disponible.

Además de estas características específicas de Kanban, es importante destacar que esta metodología no requiere roles específicos, lo cual es beneficioso para un equipo de investigación pequeño. Al no estar sujeto a una estructura rígida de roles, los miembros del grupo pueden adaptarse y colaborar de manera más flexible y ágil.

Flujo de trabajo

Con el fin de cumplir con los objetivos establecidos en esta investigación, se han planteado tareas que se adecuan a cada objetivo. Algunas de estas tareas incluyen subtareas que sirven para complementar cada acción de la manera más efectiva posible.

Objetivo 1:

- Realizar una revisión bibliográfica sobre la administración de recursos en máquinas virtuales y los problemas típicos que se presentan en su manejo.

- Identificar herramientas y métodos útiles para la recolección de datos de rendimiento.
- Revisar cuales son los algoritmos balanceadores de carga más utilizados y por qué.
- Definir una muestra de una población desconocida para en el futuro aplicar esta cantidad de interacciones.
- Revisar características de computadoras comúnmente utilizados.

Objetivo 2:

- Estudio de las metodologías ágiles para definir cual se adapta mejor al desarrollo del proyecto.
- Estudio de los sistemas operativos que serán utilizados en la propuesta.
 - Verificar la robustez, tiempo de soporte, facilidad de uso.
- Configurar el ambiente de desarrollo necesario para las siguientes etapas.
 - Instalar el sistema operativo correspondiente en 4 máquinas físicas de iguales características.
 - Verificar actualizaciones e instalar software necesario para las prácticas
- Configurar las máquinas virtuales y la infraestructura necesaria básica.
 - Montar bases de datos en servidor 1 y servidor 2 virtuales con datos de prueba como se presenta en la imagen 7.
 - Generar una página web sencilla conectada a las bases de datos para la prueba de ingreso.
 - Armar la arquitectura según el diagrama en cada máquina
 - Tener en cuenta que las máquinas no tendrán balanceadores de carga al inicio

- Generar conexiones sintéticas para cantidad de instancias que ingresarán al sistema.
- Toma de datos en primera instancia.
- Evaluar acorde a las métricas establecidas (tiempo de respuesta y cantidad de carga al sistema) antes de la implementación de los algoritmos para tener referencia del rendimiento actual de las máquinas
- Realizar la implementación de los algoritmos balanceadores de carga uno en cada máquina.
- Generar conexiones sintéticas para cantidad de instancias que ingresarán al sistema.
- Realizar pruebas para medir el rendimiento de los algoritmos balanceadores de carga en diferentes situaciones.
- Analizar la información obtenida de las pruebas y descubrir patrones y tendencias en los datos recolectados.
- Análisis estadístico de datos recolectados.

Objetivo 3:

- Seleccionar el algoritmo balanceador de carga que mejor performance presenta con respecto al tiempo y carga computacional del sistema.
- Redactar y revisar el documento final, que incluirá los hallazgos del proyecto, el proceso llevado a cabo, así como las conclusiones y recomendaciones.
- Preparar la presentación de los resultados en un lenguaje claro y entendible para el público junto con gráficas estadísticas que representen a los datos obtenidos.

Visualizar el flujo de trabajo

Para visualizar el flujo de trabajo se creó un tablero Kanban con la aplicación **Trello** representado en la **Imagen 6**, en donde se especifican las actividades a desarrollar, las que se están desarrollando y las finalizadas del proyecto.



Figura 1. Tablero Kanban

3.1.2 Determinación del administrador de servidores

Los balanceadores de carga son una pieza fundamental en arquitecturas de aplicaciones web escalables y de alta disponibilidad. Estos sistemas actúan como intermediarios entre los clientes y los servidores finales, distribuyendo la carga de trabajo de manera equilibrada entre los nodos disponibles. Esto permite que los sistemas sean más eficientes y fiables, al mismo tiempo que reducen el riesgo de interrupciones y fallos en caso de sobrecarga o errores en un servidor individual. En la comparación de la

Tabla 12, se mostrará algunas de las principales cualidades para ayudar a seleccionar la que mejor se ajuste a las necesidades de la investigación.

Tabla 12. Comparación entre administradores de balanceo de carga.

Elaborado por: Paúl González

Característica	HAProxy	F5 BIG-IP	Citrix ADC
Balanceo de carga	Round Robin, Weighted Round Robin, Least Connection, Random	Round Robin, Weighted Round Robin, Least Connection, Random	Round Robin, Weighted Round Robin, Least Connection, Random
Escalabilidad	Escala de manera horizontal	Escala de manera horizontal y vertical.	Escala de manera horizontal y vertical.
Seguridad	Soporte SSL/TLS	WAF	WAF
Capacidad de monitoreo	Registra estadísticas y eventos.	Registra estadísticas y eventos.	Registra estadísticas y eventos.
Soporte de Protocolos	TCP, HTTP, HTTPS, SSL, FTP, SMTP, SNMP, IMAP, etc.	TCP, HTTP, HTTPS	TCP, HTTP, HTTPS
Administración del tráfico	Soporte para balanceo de carga de servidor, balanceo de carga global y conmutación por error.	Soporte para balanceo de carga de servidor, balanceo de carga global y conmutación por error.	Soporte para balanceo de carga de servidor, balanceo de carga global y conmutación por error.
Integración de Redes	Soporte para múltiples interfaces de red.	Soporte para múltiples interfaces de red.	Soporte para múltiples interfaces de red.
Integración de Data Center	Soporte para múltiples centros de datos.	Soporte para múltiples centros de datos.	Soporte para múltiples centros de datos.

En la **Tabla 12** se tomaron en cuenta 3 herramientas que pueden controlar los 4 algoritmos balanceadores de carga que se usarán en esta investigación. Las herramientas son muy similares en características, sin embargo, la que se va a usar es la denominada HAProxy por su gran soporte de protocolos. Además, proporciona estadísticas detalladas sobre el rendimiento del servidor y la aplicación; También puede agregar o eliminar servidores de forma dinámica, lo que permite escalar fácilmente la aplicación según las necesidades.

3.1.3 Determinación de la distribución open source para servidores

Se decidió comparar las siguientes distros, tomando en cuenta su soporte, seguridad, versatilidad y rendimiento en entornos virtualizados para saber cuál se acopla mejor a la investigación, la comparación se presenta en la **Tabla 13**.

Tabla 13. Comparación de las distintas distribuciones open source

Elaborado por: Paúl González

Distribución Linux	Soporte	Seguridad	Versatilidad	Rendimiento en entornos virtualizados
Ubuntu Server	Ofrece soporte a largo plazo (LTS) de hasta 5 años	Tiene una amplia gama de herramientas de seguridad, como AppArmor y UFW	Excelente para servidores web y de bases de datos	Es una de las distribuciones más utilizadas en entornos virtualizados, con un buen rendimiento en máquinas virtuales
CentOS	Ofrece soporte a largo plazo (LTS) de hasta 10 años	Tiene un sistema de seguridad avanzado utilizando SELinux y otras herramientas de seguridad avanzadas	Es adecuado para diferentes propósitos como servidores web, de archivo y de correo	Se desempeña bien en entornos virtualizados, con un optimizado rendimiento gracias a su bajo consumo de recursos
Debian	Ofrece soporte a largo plazo (LTS) de hasta 3 años	Cuenta con un sistema de seguridad robusto con SELinux y otras herramientas de seguridad avanzadas	Altamente personalizable, adecuado para cualquier tipo de servidor y compatible con la mayoría de los servicios de red	Se desempeña adecuadamente en entornos virtualizados gracias a su bajo consumo de recursos y su compatibilidad con diferentes tecnologías de virtualización.
Fedora Server	Ofrece soporte a corto plazo (6 meses)	Tiene un sistema de seguridad avanzado utilizando SELinux y otras herramientas de seguridad avanzadas	Es altamente personalizable, adecuado para diferentes propósitos, incluidos servidores web y de bases de datos	Tiene un buen rendimiento en entornos virtualizados, gracias a su compatibilidad con tecnologías de virtualización como KVM y VirtualBox

Red Hat Enterprise Linux (RHEL)	Ofrece un soporte a largo plazo de hasta 10 años	Tiene una política de seguridad rigurosa y una gran cantidad de herramientas de seguridad	Es altamente personalizable y puede utilizarse para diferentes propósitos, incluidos servidores web, de bases de datos, de correo, etc.	Ofrece una excelente performance en entornos virtualizados, con un alto rendimiento y una buena compatibilidad con tecnologías de virtualización como KVM y Xen
SUSE Linux Enterprise Server (SLES)	Ofrece soporte a largo plazo de hasta 13 años	Cuenta con una gran cantidad de herramientas de seguridad y políticas de seguridad estrictas	Es adecuado para diferentes propósitos, incluidos servidores de archivos, correo y web	Ofrece un buen rendimiento en entornos virtualizados gracias a su compatibilidad con tecnologías de virtualización como Xen
FreeBSD	Ofrece soporte a largo plazo (LTS) de hasta 5 años	Cuenta con una política de seguridad robusta y una gran cantidad de herramientas de seguridad	Es altamente personalizable y puede utilizarse para diferentes tipos de servidores	Se desempeña adecuadamente en entornos virtualizados gracias a su compatibilidad con tecnologías de virtualización como bhyve y VirtualBox

En base a la **Tabla 13** se puede afirmar que Ubuntu Server es una distribución de Linux confiable y sólida, que ofrece un buen rendimiento en entornos virtuales y una amplia variedad de herramientas de seguridad. Ubuntu es ampliamente utilizado en entornos virtuales y es compatible con diversas tecnologías de virtualización. Su facilidad de uso y configuración lo convierte en una buena opción para administradores de sistemas, independientemente de su nivel de experiencia. Se utilizará Ubuntu Server debido a que es una opción atractiva para la virtualización, gracias a su robustez, seguridad,

rendimiento y facilidad de uso. Además, es muy utilizado, lo que ofrece una amplia documentación y una comunidad activa que facilita su uso.

3.1.4 WordPress

WordPress es una plataforma de gestión de contenidos (CMS) que permite crear y mantener sitios web de forma sencilla y eficiente. WordPress ofrece varias ventajas para el desarrollo de servidores web, tales como:

- Es de código abierto y gratuito, lo que significa que se puede personalizar y adaptar a las necesidades específicas de cada proyecto, así como contar con el apoyo de una amplia comunidad de desarrolladores y usuarios.
- Es fácil de instalar y configurar, ya que cuenta con un asistente que guía el proceso paso a paso y ofrece múltiples opciones de alojamiento, desde servidores propios hasta servicios en la nube.
- Es flexible y versátil, puesto que permite generar sitios web tanto simples como complejos portales o tiendas en línea, gracias a la gran variedad de temas, plugins y widgets disponibles, que amplían las funcionalidades y el diseño del sitio.
- Es seguro y confiable, ya que cuenta con un sistema de actualizaciones automáticas que protege el sitio de posibles vulnerabilidades, así como con herramientas de copia de seguridad y restauración en caso de emergencia.

Por estas razones, WordPress es una buena opción para utilizar en el proyecto, porque permite optimizar el tiempo y los recursos en los servidores web, al facilitar el desarrollo, la gestión y el mantenimiento de los sitios web.

3.1.5 LAMP

LAMP es un conjunto de tecnologías de código abierto que permite crear y desplegar aplicaciones web de forma rápida y eficiente. Está compuesto por Linux, un sistema operativo robusto y seguro; MySQL, un sistema de gestión de bases de datos relacional y escalable; Apache, un servidor web potente y flexible; y PHP, un lenguaje de programación interpretado y orientado a objetos.

LAMP tiene varias ventajas frente a otras tecnologías para el desarrollo web, tales como:

- Es económico, ya que no requiere licencias ni costes de mantenimiento.
- Es compatible, porque se puede ejecutar en diferentes plataformas y sistemas operativos.
- Es modular, puesto que se integra con otras herramientas y frameworks.
- Es versátil, porque se adapta a diferentes tipos de aplicaciones y necesidades.
- Es popular, por el hecho de que cuenta con una amplia comunidad de desarrolladores y usuarios que ofrecen soporte y recursos.

LAMP es especialmente conveniente para el proyecto actual, ya que:

- Permite generar aplicaciones web dinámicas y personalizadas que se ajusten a los requisitos del problema.
- Facilita la implementación y el despliegue de las aplicaciones en diferentes entornos virtuales.

3.1.6 JMETER

JMETER es una herramienta de software libre que permite realizar pruebas de rendimiento y funcionalidad de aplicaciones web. Su uso es conveniente para el

análisis de algoritmos balanceadores de carga utilizando máquinas virtuales, ya que ofrece las siguientes ventajas:

- Permite simular una gran cantidad de usuarios concurrentes que acceden a un servidor web, generando una alta demanda de recursos y tiempo de respuesta.
- Puede medir el comportamiento del servidor web bajo diferentes escenarios de carga.
- Permite configurar diversos parámetros de las pruebas, como el número de usuarios, el tiempo de espera, el tipo de petición, los datos enviados y recibidos, etc.
- Permite visualizar los resultados de las pruebas en forma de gráficos, tablas y reportes, facilitando el análisis y la comparación de los algoritmos balanceadores de carga.
- Permite ejecutar pruebas sobre cualquier tipo de aplicación web, independientemente del lenguaje o la plataforma utilizada.

Por estas razones JMETER es una herramienta adecuada para la investigación, ya que permite obtener datos objetivos y fiables sobre el funcionamiento y la optimización de tiempo y recursos en servidores web.

3.1.7 Libvirt

Libvirt es un software libre que proporciona una interfaz común para interactuar con diferentes tecnologías de virtualización, como KVM, Xen, VMware, VirtualBox, entre otras. Esto facilita la portabilidad y la interoperabilidad de las máquinas virtuales, así como la integración con otras herramientas de gestión. Además, Libvirt ofrece una serie de funcionalidades que mejoran el rendimiento y la seguridad de las máquinas virtuales, tales como:

- La migración en vivo, que permite trasladar una máquina virtual en ejecución de un servidor a otro sin interrumpir su funcionamiento.
- El aislamiento de recursos, que evita que una máquina virtual consuma más de los asignados y afecte al resto del sistema.
- El control de acceso, que restringe el ingreso a las máquinas virtuales según el rol y los permisos del usuario.
- El monitoreo y la auditoría, que permiten supervisar el estado y el comportamiento de las máquinas virtuales y registrar los eventos relevantes.

Por estas razones Libvirt es una buena opción para administrar las máquinas virtuales que se utilizarán en el análisis de los algoritmos balanceadores de carga, ya que ofrece una mayor flexibilidad, eficiencia y seguridad que otras opciones disponibles en el mercado.

3.1.8 Virt Manager

VM es un administrador de máquinas virtuales basado en la librería Libvirt. Permite crear, modificar, eliminar y controlar el estado de las máquinas virtuales de forma sencilla y eficiente, mediante una interfaz gráfica o una línea de comandos. Ofrece varias ventajas para el desarrollo de este trabajo, tales como:

- Es compatible con diversos sistemas operativos y plataformas de virtualización, lo que permite una mayor flexibilidad y portabilidad de las máquinas virtuales.
- Es capaz de gestionar múltiples nodos y grupos de máquinas virtuales, lo que facilita la implementación y el análisis de los algoritmos balanceadores de carga en diferentes escenarios.

- Puede monitorizar el rendimiento y el consumo de recursos de las máquinas virtuales, lo que permite evaluar el impacto de los algoritmos balanceadores de carga en los servidores web.
- Tiene la capacidad de configurar redes virtuales y dispositivos de almacenamiento para las máquinas virtuales, lo que permite una mayor seguridad y eficiencia en el acceso a los datos.

Por estas razones VM es una herramienta adecuada para la gestión de máquinas virtuales en este trabajo, ya que permite crear y manipular los entornos virtuales donde se ejecutarán los algoritmos balanceadores de carga, así como medir su desempeño y su impacto en la optimización de los servidores web.

3.1.9 Power BI

Power BI es una herramienta de análisis de datos que permite crear informes interactivos y visualizaciones dinámicas. Con este software, se puede acceder a diferentes fuentes de datos, transformarlos y modelarlos según las necesidades del usuario, y compartir los resultados con otros usuarios o aplicaciones. Tiene una interfaz intuitiva y fácil de usar, que facilita la exploración y el descubrimiento de insights a partir de los datos.

Una de las ventajas de Power BI es que permite generar líneas de tiempo para mostrar la evolución de una variable o un indicador. Las mismas son útiles para comparar diferentes períodos, identificar tendencias o patrones, y analizar el impacto de eventos o acciones. Con Power BI, se puede personalizar el diseño y el formato de las líneas de tiempo, agregar filtros o segmentaciones, y combinarlas con otros tipos de gráficos o tablas.

Power BI es una solución eficiente y versátil para el análisis de datos, que ofrece múltiples beneficios para los usuarios que quieren obtener información valiosa y tomar mejores decisiones basadas en los datos. Es una herramienta que se adapta a diferentes escenarios y necesidades, y que se integra con otras plataformas y servicios de Microsoft y de terceros.

3.1.10 Tableau

Tableau es una herramienta de visualización de datos que permite crear gráficos interactivos y dinámicos a partir de diferentes fuentes de información. Su uso es beneficioso para analizar y comunicar los resultados de una investigación, un proyecto o una actividad de forma clara y efectiva. Tableau ofrece diversas opciones para personalizar los gráficos según las necesidades y preferencias del usuario, así como para explorar los datos con filtros, cálculos y parámetros. Uno de los tipos de gráficos que se pueden producir con Tableau son los de columnas, que son útiles para comparar valores numéricos entre varias categorías o grupos. Los gráficos de barras facilitan la identificación de tendencias, patrones y diferencias entre los datos, así como la evaluación del rendimiento o la calidad de un proceso o producto. Es una herramienta poderosa y versátil que permite crear visualizaciones de datos atractivas y significativas. Estas pueden ayudar a tomar mejores decisiones y a transmitir un mensaje con mayor impacto.

3.1.11 R Studio

R Studio es un entorno de desarrollo integrado (IDE) para el lenguaje de programación R, que se utiliza para el análisis estadístico y la visualización de datos. R Studio ofrece una serie de ventajas para los usuarios, tales como:

- Una interfaz gráfica de usuario que facilita la escritura, la depuración y la ejecución de código R, así como la gestión de proyectos, archivos y paquetes.
- Una consola interactiva que permite introducir y ejecutar comandos R al instante, así como ver los resultados y los gráficos generados.
- Un editor que resalta la sintaxis, completa automáticamente el código, ofrece sugerencias y ayuda en línea, y permite crear y ejecutar fragmentos de código (chunks) con un solo clic.
- Es un visor de datos que muestra las variables, las tablas y las listas generadas en R, permitiendo explorar, filtrar, ordenar y modificar los datos de forma sencilla.
- También es un visor de documentos que permite producir informes dinámicos con R Markdown, un instrumento que combina código R con texto formateado en Markdown, HTML, LaTeX o Word.
- Una integración con herramientas externas como Git, GitHub, Shiny, SQL, Python y C++, que amplían las posibilidades y funcionalidades de R.

R Studio es una herramienta muy útil para obtener datos y resultados estadísticos para comparar pares de resultados, ya que permite:

- Acceder a una gran variedad de fuentes de datos, tanto locales como remotas, mediante funciones específicas o paquetes especializados.
- Manipular, transformar y limpiar los datos con facilidad, usando diferentes librerías.
- Realizar todo tipo de análisis estadísticos con R, desde los más básicos hasta los más avanzados, empleando funciones incorporadas o paquetes adicionales.

- Comunicar los hallazgos y las conclusiones con informes reproducibles y profesionales, utilizando R Markdown o el paquete knitr.

R Studio es, por tanto, un entorno de desarrollo ideal para trabajar con R y aprovechar todo su potencial para el análisis estadístico y la visualización de datos.

3.1.12 Excel

Excel es una aplicación de hoja de cálculo que permite crear, modificar y analizar datos numéricos y alfanuméricos. Es una herramienta muy útil para realizar tareas de contabilidad, estadística, finanzas y otras que requieren el manejo de información cuantitativa. Excel tiene varias ventajas que lo hacen un programa muy popular y versátil:

- Permite trabajar con archivos csv, que son un formato estándar para almacenar y compartir elementos tabulares. Estos csv se pueden importar y exportar de manera sencilla, lo que facilita la integración con otras fuentes de datos o aplicaciones.
- Permite normalizar datos, es decir, ajustarlos a un formato común y eliminar inconsistencias o errores. Esto mejora la calidad y la fiabilidad de los datos, así como su presentación y comparación.
- Permite generar cálculos de manera rápida y sencilla, con el uso de fórmulas, funciones y referencias que automatizan los procesos matemáticos. Excel ofrece una gran variedad de funciones para realizar operaciones aritméticas, lógicas, financieras, estadísticas, trigonométricas y más.
- Permite crear gráficos, tablas dinámicas y otros elementos visuales que facilitan la interpretación y la comunicación de los resultados. Ofrece múltiples opciones para personalizar el diseño, el estilo y el formato de estos elementos.

Excel es una aplicación muy completa y eficiente para el manejo de datos numéricos y alfanuméricos, que ofrece múltiples posibilidades para el análisis, la organización y la presentación de la información.

3.2 Desarrollo de la propuesta

3.2.1 Instalación y configuración del entorno de pruebas.

Para comenzar la instalación es bueno tener un diagrama en el cual guiarse para tener claro las conexiones que se realizarán y las configuraciones necesarias de cada máquina. En la **figura 6** se representa el diagrama de lo que será el entorno de pruebas ya armado. Las dos máquinas principales (Main) con iguales componentes y sistemas. Dentro de las mismas las 3 máquinas virtualizadas con sus direcciones IP y con los algoritmos que se usarán, por temas de tiempo se usaron dos máquinas y cada uno asignado diferentes algoritmos para poder acortar tiempos de pruebas.

Hay que tomar en cuenta que a las máquinas MAIN se ha asignado una configuración SWAP de 20Gb a cada una para que no afecte con el rendimiento de las pruebas ya que con los 8Gb disponibles no tenían la capacidad suficiente para manejar los datos.

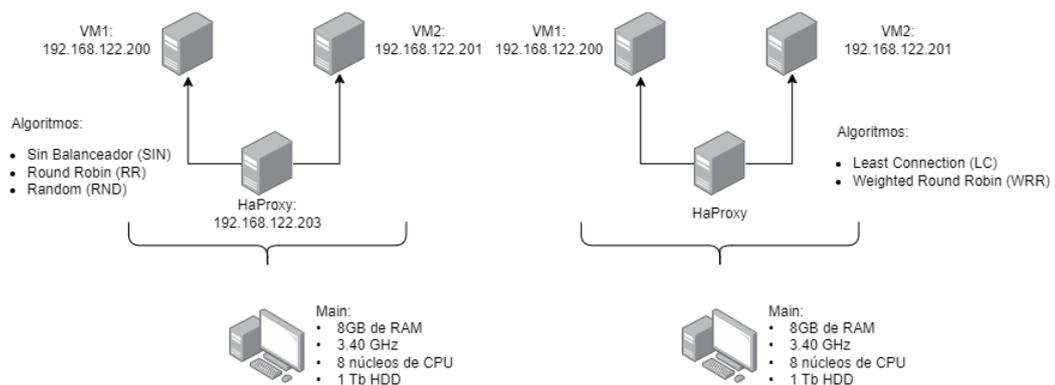


Figura 6. Diagrama del entorno de pruebas

El proceso de instalación y configuración de las máquinas virtuales se muestra en los **anexos (1-5)**. Para la instalación de Ubuntu Server se muestra en los **anexos (6-9)**. Una vez montada la arquitectura de servidores virtuales se procede a instalar, configurar y añadir una página de uso libre en WordPress presentados en los **anexos (11-14)**. Con la página de WordPress montada se procede a clonar la máquina virtual 1 (VM1) y se configura HAProxy con las IP de ambas máquinas virtuales VM1 y VM2 como se representa en los **anexos (16-20)**. Finalmente se realiza la instalación de JMETER y sus componentes en la máquina principal para realizar las pruebas, esto representado en los **anexos (21-31)**.

3.2.2 Ajuste de los parámetros para pruebas en JMETER

En la **figura 7** en el **punto 1** se indica el número de usuarios que se conectarán simultáneamente a la página para probarla. En el **punto 2** se debe poner el tiempo en que los usuarios se irán conectando hasta llegar al límite indicado en el punto 1. En el **punto 3** se especifica el tiempo que durará la prueba en segundos. Como **punto 4** se puede indicar el tiempo en que se desea que la prueba tarde en iniciar, esto puede servir para generar varias pruebas en cola.

Para el caso actual de estudio se ha puesto 50 usuarios simultáneos, con el fin de exigir a la máquina que utilice su máximo performance acorde a sus componentes instalados. También 60 segundos para inicio de conexión de usuarios y un total de 12 horas para la prueba por algoritmo.

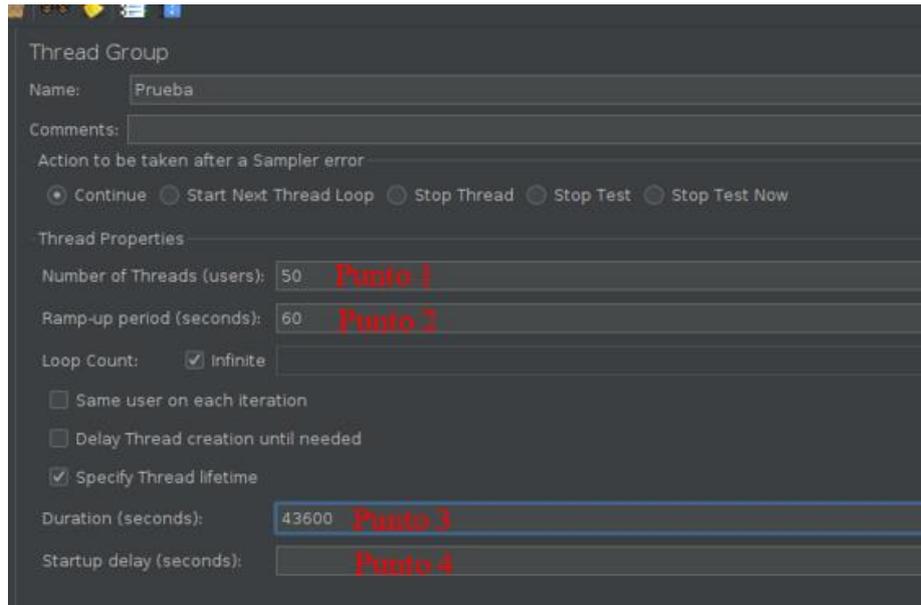


Figura 7. Parámetros de prueba en Jmeter

En las máquinas que se vayan a realizar las pruebas se debe iniciar el agente del servicio para Jmeter, en este caso las máquinas virtuales VM1, VM2 y Main. En la **figura 8** se muestra los mensajes de respuesta de los servidores, los que indican el puerto 4444 en el que escucha el servicio y el mensaje que el agente ha iniciado sin presentar errores.

```
root@web1:~# cd ServerAgent-2.2.3/
root@web1:~/ServerAgent-2.2.3# ls
CMDRunner.jar LICENSE ServerAgent.jar lib startAgent.bat startAgent.sh
root@web1:~/ServerAgent-2.2.3# ./startAgent.sh
INFO 2023-07-14 12:07:06.650 [kg.apc.p] (): Binding UDP to 4444
INFO 2023-07-14 12:07:06.710 [kg.apc.p] (): Binding TCP to 4444
INFO 2023-07-14 12:07:06.723 [kg.apc.p] (): JP@GC Agent v2.2.3 started
```

Figura 8. Iniciar agente del servicio Jmeter

En la **figura 9** se presentan las diferentes pruebas que se han añadido a Jmeter. Las 6 pruebas se explican a detalle en las **figuras 10 – 14**. Además, hay que tener en cuenta que los resultados de estas pruebas se obtienen en formatos csv, con los cuales se trabajará en la sección **3.2.3 Obtención y normalización de los datos de prueba**.

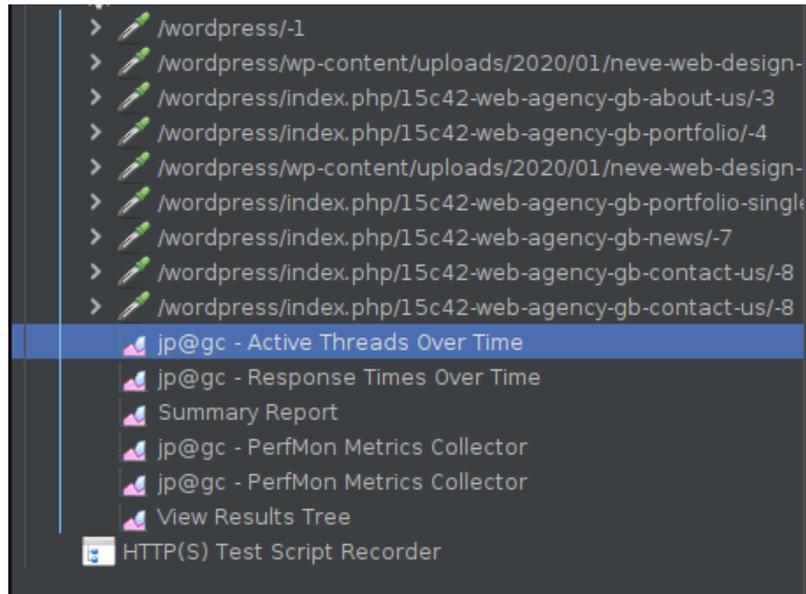


Figura 9. Tipos de prueba en Jmeter

En la **figura 10** se presenta la prueba “Active Threads Over Time” la cual indica la cantidad de interacciones simultaneas conectadas a la máquina a través del tiempo. Como se indicó en la **figura 7**, las interacciones configuradas son 50 y el tiempo total son 12 horas de prueba.



Figura 10. Cantidad de interacciones simultaneas

En la **figura 11** se indica la prueba “Response Times Over Time” la cual muestra el tiempo de respuesta en milisegundos que cada página ha presentado a través del tiempo.

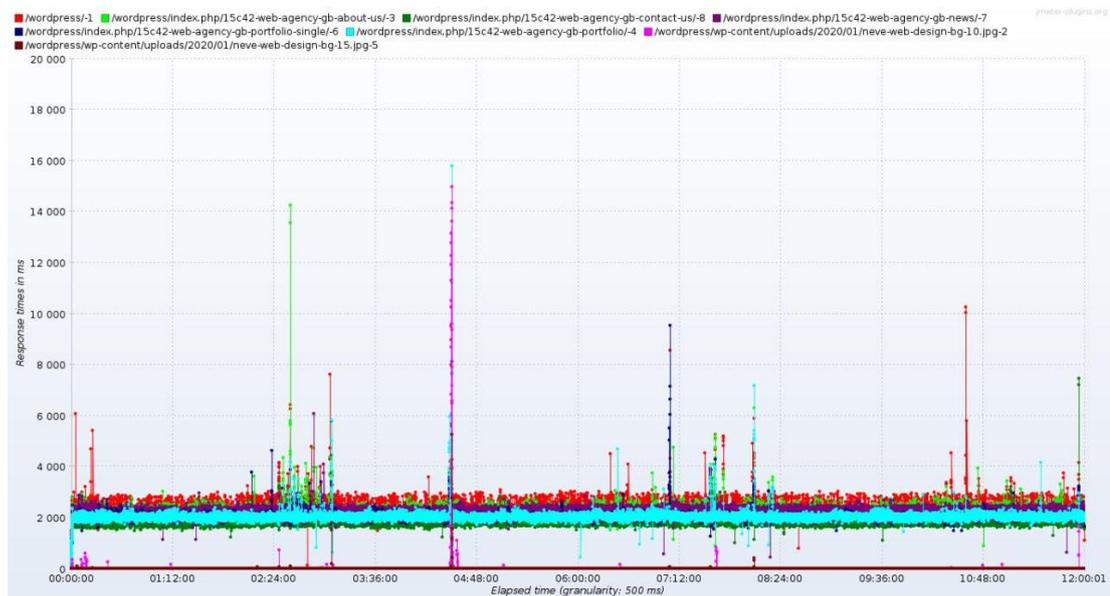


Figura 11. Tiempo de respuesta por página

En la **figura 12** se presenta el ítem “Summary Report” el cual presenta un resumen del gráfico de la **figura 11**, en la cual se detalla el nombre de la página, el número de interacciones y el mínimo, máximo, promedio y desviación estándar de los tiempos de respuesta.

Summary Report

Name:

Comments:

Write results to file / Read from file

Filename

Label	# Samples	Average	Min	Max	Std. Dev.	Error %
/wordpress/-1	146286	2304	89	10967	237.96	0.00%
/wordpress/wp-c...	146285	3	0	15080	182.64	0.00%
/wordpress/inde...	146285	2245	75	16814	170.64	0.00%
/wordpress/inde...	146285	2059	72	15859	158.03	0.00%
/wordpress/wp-c...	146285	0	0	5262	22.36	0.00%
/wordpress/inde...	146285	2034	71	15745	126.66	0.00%
/wordpress/inde...	146284	2298	79	6082	114.46	0.00%
/wordpress/inde...	292534	1903	64	11167	126.96	0.00%
TOTAL	1316529	1639	0	16814	899.52	0.00%

Figura 12. Cuadro de resumen de tiempo de respuesta

En la **figura 13** y **14** presentan las gráficas de “PerfMon Metrics Collector” las que muestran el uso de CPU y Memoria respectivamente. La medida se la realiza en un intervalo de 0 a 100 que es el porcentaje de uso del CPU o la Memoria



Figura 13. Gráfica de uso de CPU

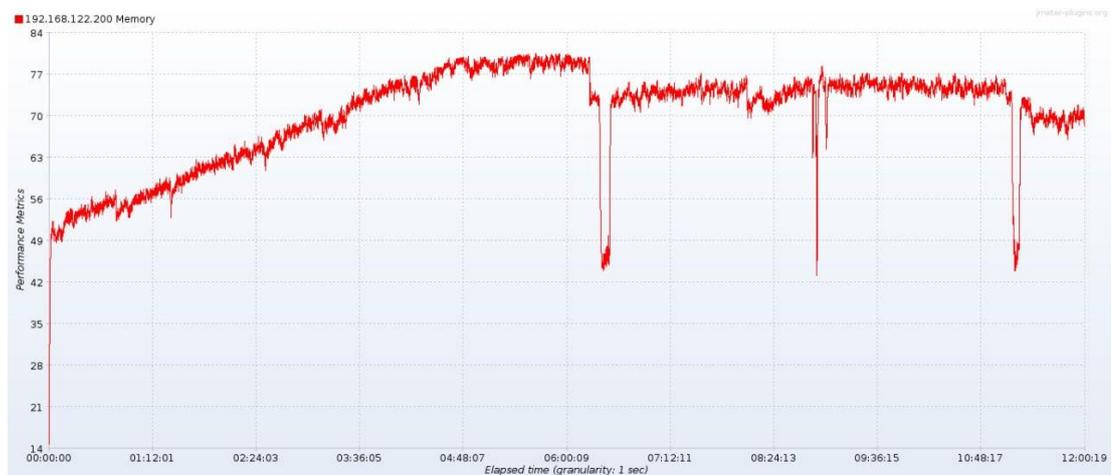


Figura 14. Gráfica del uso de Memoria

En la **figura 15** se muestra el árbol de resultados “View Results Tree”, este detalla si la petición a la página por interacción fue correcta o presentó un error. Se puede diferenciar los dos colores: verde si la petición fue correctamente presentada o en rojo si la página presento un problema. A modo de ejemplo se tomó una parada abrupta de

la ejecución para mostrar ambos resultados. En el detalle del problema señalado se lee que fue por el cierre de socket.

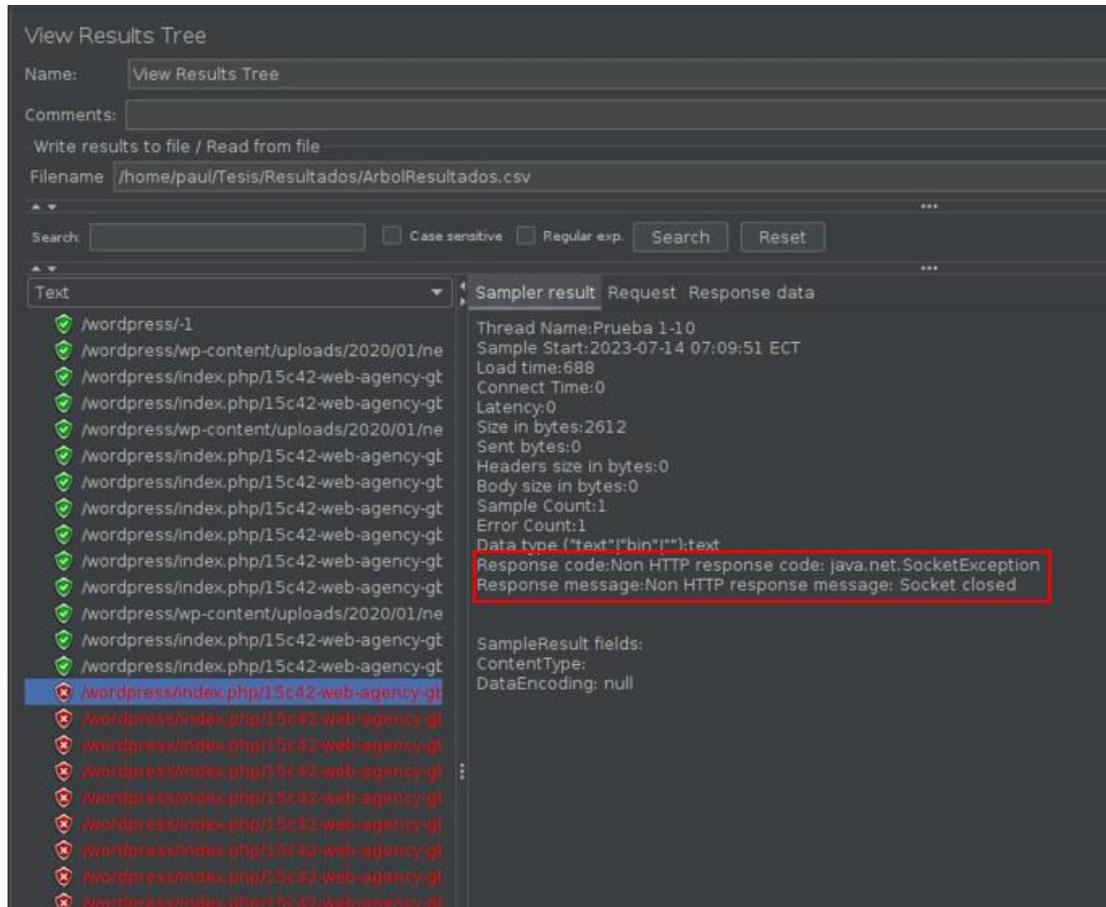


Figura 15. Árbol de resultados

3.2.3 Obtención y normalización de los datos de prueba

Una vez obtenidos los resultados de todas las pruebas, se requiere graficarlos utilizando archivos csv. No obstante, se debe normalizar los datos a formatos legibles antes de proceder. Para lograr esto, se utilizará la herramienta Excel con el fin de

cambiar el formato de los archivos y homogeneizarlos. El primer paso para lograr este objetivo es separar los datos en columnas, tal como se muestra en la **figura 16**.

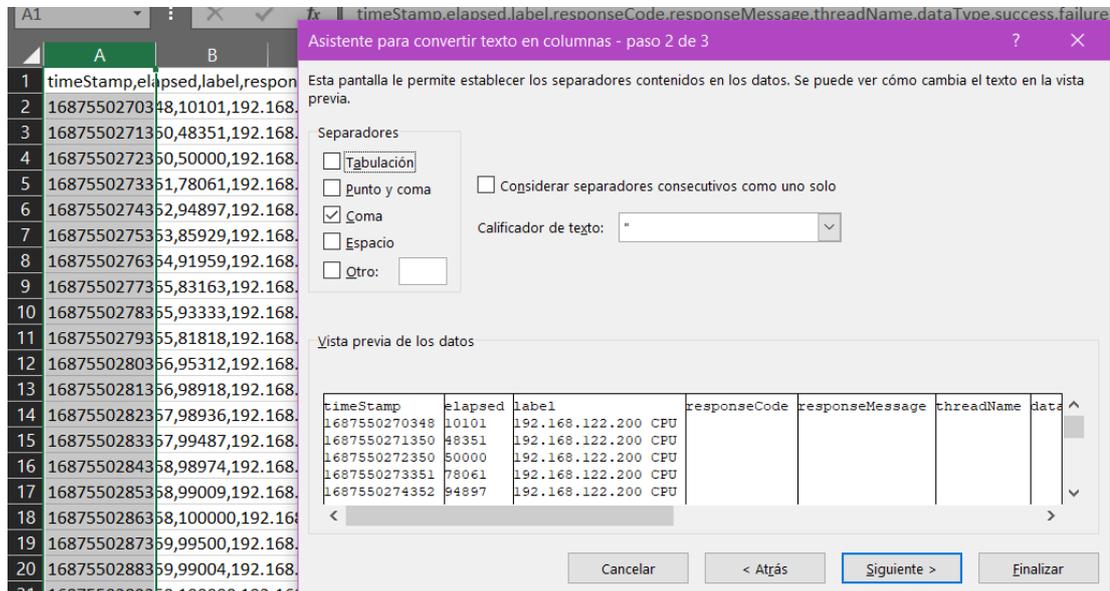


Figura 16. Transformar datos de csv a xlsx

Después de obtener los datos de tiempo transcurrido y el porcentaje, es necesario transformarlos para poder utilizarlos de forma conveniente. Para lograr esto, se emplean dos columnas: “timeStamp” y “elapsed”.

Para transformar la columna de “timeStamp”, es necesario realizar una serie de operaciones. Primero, se debe restar el valor de la celda de la fila 2 a los valores de la fila N. Luego, a estos valores se les debe dividir por una sucesión de tiempo que es 1000 ms por 60 segundos y por 60 minutos. De esta manera se obtiene un resultado de

0 a 11, que corresponden a las 12 horas de ejecución por algoritmo. La fórmula utilizada para este cálculo es “=REDONDEAR.MENOS((A2-A\$2)/(1000*60*60);0)”.

La **figura 17** muestra cómo se usó la fórmula para representar los datos.

	E	F	G	H	I
	timeStamp	TIEMPC	elapsed	PORCENTAJE	timeStamp
01	1,68797E+12	0	42500	42,5	1,68755E+12
00	1,68798E+12	1	99502	99,5	1,68755E+12
00	1,68798E+12	2	99502	99,5	1,68756E+12
00	1,68799E+12	3	100000	100	1,68756E+12
00	1,68799E+12	4	100000	100	1,68756E+12
00	1,68799E+12	5	100000	100	1,68757E+12
12	1,688E+12	6	80693	80,6	1,68757E+12
00	1,688E+12	7	98957	98,9	1,68758E+12
00	1,688E+12	8	99502	99,5	1,68758E+12
00	1,68801E+12	9	100000	100	1,68758E+12
00	1,68801E+12	10	100000	100	1,68759E+12
00	1,68801E+12	11	98941	98,9	1,68759E+12

Figura 17. Fórmula de transformación de timeStamp

La fórmula “=REDONDEAR.MENOS(A2*0,001;1)” se utiliza para calcular el porcentaje de uso de CPU y memoria. El resultado de esta fórmula es un valor que oscila entre 0 y 100, y se utiliza como medida para evaluar el rendimiento de los algoritmos ya que el valor original de “elapsed” se encuentra representado en decenas de miles. Esta fórmula permite obtener un valor numérico que indica el porcentaje de

uso de CPU y memoria, lo cual es útil para medir el rendimiento de los algoritmos. Esta fórmula se representa en la **figura 18**.

=REDONDEAR.MENOS(K2*0,001;1)			
H	J	L	
RCENTAJE	TIEMPO	PORCENTAJE	TIEMPO
42,5	0	31	
58	0	50,2	

Figura 18. Fórmula de transformación de elapsed

Para transformar los datos obtenidos, es necesario realizar el proceso cuatro veces por algoritmo, y los resultados se guardan en la misma hoja de Excel para mantener una estructura ordenada de archivos. Los datos se representan con diferentes colores: celeste para la primera ejecución, anaranjado para la segunda, verde para la tercera y gris para la cuarta. En la columna Q se calcula el promedio de los cuatro porcentajes a lo largo del tiempo, y en la columna R se obtiene el promedio total de cada algoritmo. Es importante realizar este proceso para los cuatro algoritmos restantes (RR-RND-LC-WRR) en cada máquina virtual en las cuatro ejecuciones realizadas. Para mayor

claridad, se puede revisar la **figura 19** como ejemplo de una hoja normalizada de un algoritmo.

	B	D	F	H	J	L	N	P	Q	R
1	TIEMPC	PORCENTAJE	TIEMPC	PORCENTAJE	TIEMPC	PORCENTAJE	TIEMPC	PORCENTAJE	PROM	TOTAL
2	0	10,1	0	42,5	0	31	0	50,5	33,53	99,29
3	0	48,3	0	58	0	50,2	0	99	63,88	
4	0	50	0	71,2	0	71,7	0	98,9	72,95	
5	0	78	0	89,9	0	89,3	0	100	89,30	
6	0	94,8	0	88,4	0	89,1	0	100	93,08	
7	0	85,9	0	94	0	91,4	0	99	92,58	
8	0	91,9	0	95,9	0	93,9	0	99,5	95,30	
9	0	83,1	0	97,9	0	98	0	99,4	94,60	
10	0	93,3	0	98,1	0	98	0	99,5	97,23	
11	0	81,8	0	98,9	0	98	0	99,4	94,53	
12	0	95,3	0	98,5	0	98,9	0	99,5	98,05	
13	0	98,9	0	99,5	0	100	0	99,5	99,48	
14	0	98,9	0	100	0	99	0	99,5	99,35	
15	0	99,4	0	100	0	100	0	99	99,60	

Figura 19. Tiempo y porcentaje de uso por algoritmo

El proceso que se describe en las **figuras 16 a 19** se utiliza para graficar las líneas de tiempo. Para generar los gráficos de barras se necesita seguir algunos pasos adicionales. A partir del documento normalizado mostrado en la **figura 19**, es necesario obtener los promedios de cada columna de porcentaje al final, tal como se muestra en la **figura 20**.

11	100	11	99,5	11	99	11	100
11	100	11	99,5	11	100	11	100
11	99,5	11	99	11	100	11	99,5
11	100	11	100	11	100	11	100
	99,42105666		99,14205764		99,64705323		98,95913499

Figura 20. Promedio de cada columna de porcentaje

Una vez que se ha completado el proceso mostrado en la **figura 20** para cada archivo de datos, es necesario copiar y pegar los datos en otro documento en el orden indicado en la **figura 21**. Este documento consta de cuatro columnas: la primera indica la métrica que se está midiendo (en este caso se trata de la CPU), la segunda muestra la máquina virtual utilizada, la tercera lista cada uno de los cuatro algoritmos (incluyendo

el sin balanceador), y la cuarta columna presenta los promedios por algoritmo obtenidos. Este proceso se lleva a cabo para tener todos los promedios en un solo documento, con el fin de generar los gráficos de barras necesarios.

	A	B	C	D
1	Metric	Machine	Algorithm	Value
2	CPU	VM1	SIN BALANCEADOR	99,6745247
3	CPU	VM1	SIN BALANCEADOR	99,59153139
4	CPU	VM1	SIN BALANCEADOR	99,67277864
5	CPU	VM1	SIN BALANCEADOR	99,58832874
6	CPU	VM1	LEAST CONNECTION	99,42105666
7	CPU	VM1	LEAST CONNECTION	99,14205764
8	CPU	VM1	LEAST CONNECTION	99,64705323
9	CPU	VM1	LEAST CONNECTION	98,95913499
10	CPU	VM1	RANDOM	97,65418869
11	CPU	VM1	RANDOM	96,70998841
12	CPU	VM1	RANDOM	99,36198656
13	CPU	VM1	RANDOM	98,08572554
14	CPU	VM1	ROUND ROBIN	97,23519331
15	CPU	VM1	ROUND ROBIN	98,29231649
16	CPU	VM1	ROUND ROBIN	97,19789877
17	CPU	VM1	ROUND ROBIN	97,65018911
18	CPU	VM1	WEIGHTED ROUND ROBIN	99,52524157
19	CPU	VM1	WEIGHTED ROUND ROBIN	99,54206002
20	CPU	VM1	WEIGHTED ROUND ROBIN	99,59960839
21	CPU	VM1	WEIGHTED ROUND ROBIN	99,56784961
22	CPU	VM2	SIN BALANCEADOR	0,390909091

Figura 21. Tabla de valores de CPU-VM-ALGORITMO

Para generar los gráficos de tiempo de respuesta es necesario seguir un proceso diferente, ya que los datos de origen provienen de una fuente distinta a la de los anteriores. Los datos obtenidos del tiempo de respuesta por página en Jmeter se presentan en un archivo CSV demasiado extenso para ser normalizado en Excel, con alrededor de 3 millones de filas por archivo. Asimismo, otro problema que se presenta es el orden de estas filas, dado que se muestran a medida que las páginas web son cargadas, de manera que cada una aparece en diferentes intervalos de tiempo. Por ejemplo, la marca 1 de la **figura 22** muestra la página de contacto, mientras que las marca 2 corresponde a la página de portafolio, la marca 3 muestra el diseño de la página y la marca 4 muestra nuevamente el portafolio.

```

1687593465837,934,/wordpress/index.php/15c42-web-agency-gb-contact-us/-8,200,OK
1687593465652,1122,/wordpress/index.php/15c42-web-agency-gb-portfolio/-4,200,OK
1687593466774,1,/wordpress/wp-content/uploads/2020/01/neve-web-design-bg-15.jpg
1687593456591,10191,/wordpress/index.php/15c42-web-agency-gb-portfolio-single/-1
1687593465693,1102,/wordpress/index.php/15c42-web-agency-gb-portfolio/-4,200,OK
1687593466795,1,/wordpress/wp-content/uploads/2020/01/neve-web-design-bg-15.jpg
1687593465753,1063,/wordpress/index.php/15c42-web-agency-gb-portfolio/-4,200,OK
1687593466816,0,/wordpress/wp-content/uploads/2020/01/neve-web-design-bg-15.jpg
1687593465700,1140,/wordpress/index.php/15c42-web-agency-gb-about-us/-3,200,OK
1687593465943,953,/wordpress/index.php/15c42-web-agency-gb-contact-us/-8,200,OK
1687593465846,1057,/wordpress/index.php/15c42-web-agency-gb-portfolio-single/-6
1687593465838,1089,/wordpress/index.php/15c42-web-agency-gb-portfolio/-4,200,OK
1687593466927,1,/wordpress/wp-content/uploads/2020/01/neve-web-design-bg-15.jpg
1687593466001,934,/wordpress/index.php/15c42-web-agency-gb-portfolio/-4,200,OK
1687593466935,0,/wordpress/wp-content/uploads/2020/01/neve-web-design-bg-15.jpg
1687593450025,16930,/wordpress/-1,200,OK,Prueba 1-7,text,true,,11728,463,50,50,|

```

contact-us/-8 1
portfolio/-4 2
neve-web-design-bg 3
portfolio/-4 4

Figura 22. Datos crudos del tiempo de respuesta

Teniendo en cuenta el párrafo anterior, se concluye que es impracticable normalizar todos los datos debido al tiempo limitado disponible para completar el proyecto. Por esta razón, se utilizará la tabla de resumen que se muestra en la **figura 12** y que es proporcionada por Jmeter. Estos datos se presentan en un archivo CSV que puede ser separado en columnas utilizando la herramienta de Excel, y el resultado se puede ver en la **figura 23**.

	A	B	C	D	E	F	G	H	I	J	K
1	Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
2	Index	266243	1258	83	51272	790.33	0.017%	616.300	70.60	2.79	11730.7
4	About	266237	1248	86	50435	760.84	0.018%	616.393	74.21	2.77	12327.7
5	Portafolio 1	266230	1135	71	50379	625.95	0.009%	616.391	65.88	3.02	10944.5
7	Portafolio 2	266230	1121	83	50893	641.03	0.011%	616.385	67.90	3.06	11280.3
8	Blog	266229	1273	86	50510	537.93	0.008%	616.386	64.76	3.03	10758.8
9	Contact	532421	1028	69	50375	493.54	0.007%	1.232.693	124.91	7.89	10376.2
10	TOTAL	2396057	900	0	51272	743.94	0.010%	5.546.379	1296.25	27.06	23932.1

Figura 23. Tabla de resumen del tiempo de respuesta

Una vez que se ha realizado este proceso con todas las tablas de datos clasificadas por algoritmo, es necesario calcular el promedio de cada una de ellas en una sola tabla, la cual será la que represente el tiempo de respuesta por algoritmo. Todas las tablas promediadas deben ser enviadas a una sola hoja de Excel, como se muestra en la **figura 24**. En la primera columna se indica a qué algoritmo representa cada cuadro. En la

segunda columna se encuentra el nombre de la página web que se probó. En la tercera columna se muestra el número de interacciones que ha tenido cada página (la página de contacto tendrá el doble, ya que carga y envía datos desde y hacia la base de datos); las columnas average, min, max y standard deviation representan la cantidad de tiempo en milisegundos que ha tomado cada página en mostrarse. La columna de porcentaje de error muestra si las páginas los han sufrido al momento de cargarse, y el Throughput se define como el número de mensajes que se han recibido de manera exitosa. Las últimas tres columnas muestran los kilobytes enviados, recibidos y el promedio de ambos.

1	Algorithm	Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
2	LC	Index	235853	1157,25	61,5	43145,25	738,17	0,009585	685737,25	72,475	2,79	11019,35
3	LC	About	235840,75	1145,75	61	42667,5	635,48	0,0001575	685755	76,0725	2,7725	11568,975
4	LC	Portafolio1	235833,25	1040,5	53	42967,5	531,345	0,00008	685743	67,74	3,02	10294,75
5	LC	Portafolio2	235829,75	1028	57	43105,75	557,7	0,0000975	685730	69,7625	3,065	10604,175
6	LC	Blog	235825,25	1168,25	61,25	42431,25	457,1575	0,0000625	685723,5	66,62	3,0325	10123,5
7	LC	Contact	471630	942,75	50,25	42164,25	409,93	0,000055	1371389,75	128,6225	7,8925	9771,175
8	RND	Index	256299,5	1340	80	54001,75	1472,2375	0,0006075	599292,75	68,625	2,71	11725,975
9	RND	About	256298,25	1290,25	76,5	52523,25	1274,945	0,0004875	600149,25	72,2275	2,7	12323,825
10	RND	Portafolio1	256286	1163,5	70,5	51643,75	1185,3725	0,0004225	600354,5	64,145	2,9375	10941
11	RND	Portafolio2	256273,5	1142,25	70,5	52313,75	1078,515	0,00035	600545	66,14	2,9875	11277,675
12	RND	Blog	256256,5	1299	78	52129	951,94	0,0002925	600615	63,09	2,9575	10756,125
13	RND	Contact	512502,5	1038	45,5	52363,75	920,17	0,000265	1201340	121,705	7,6875	10374,05
14	RR	Index	254195,50	1351,25	79,25	52664,75	1630,08	0,00	589194,50	67,48	2,66	11727,95
15	RR	About	254188,25	1315	76,25	50633,75	1569,195	0,00044	589400	70,94	2,6525	12324,3
16	RR	Portafolio1	254181	1188,5	70,75	53282,25	1385,7125	0,00034	589429,25	62,985	2,885	10942,025
17	RR	Portafolio2	254173,75	1170,75	70,75	53507,75	1385,955	0,00035	589497,25	64,9225	2,9275	11277,675
18	RR	Blog	254167,25	1329,5	78,25	50727,5	1446,695	0,0003375	589600	61,93	2,9025	10756
19	RR	Contact	508310,5	1058,75	63,25	52533,25	1237,715	0,0003	1179824	119,525	7,5525	10373,85
20	SIN	Index	147496,25	2281,25	149,25	22170	317,77	0	341430,25	39,1075	1,5425	11728,975

Figura 24. Promedio de cuadros de tiempo de respuesta

3.2.4 Graficar las líneas de tiempo en Power BI

Como primer paso, se deben importar todos los archivos de Excel relacionados con el tiempo, los cuales se presentan en la **figura 19**. Todos los archivos cargados en Power BI se muestran en la **figura 25**.

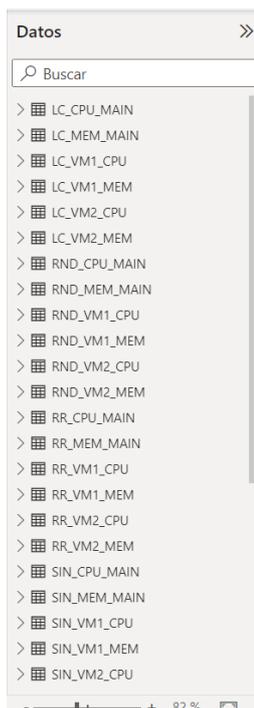


Figura 25. Columna de datos importados a Power BI

Para sincronizar los tiempos de cada uno de los archivos en Power BI, se debe crear relaciones de “muchos a muchos” entre las tablas del mismo grupo, es decir, cada categoría de CPU y memoria de cada máquina. Esto se puede hacer en la pestaña “Vista de modelo” en Power BI y está representado en la **figura 26**.

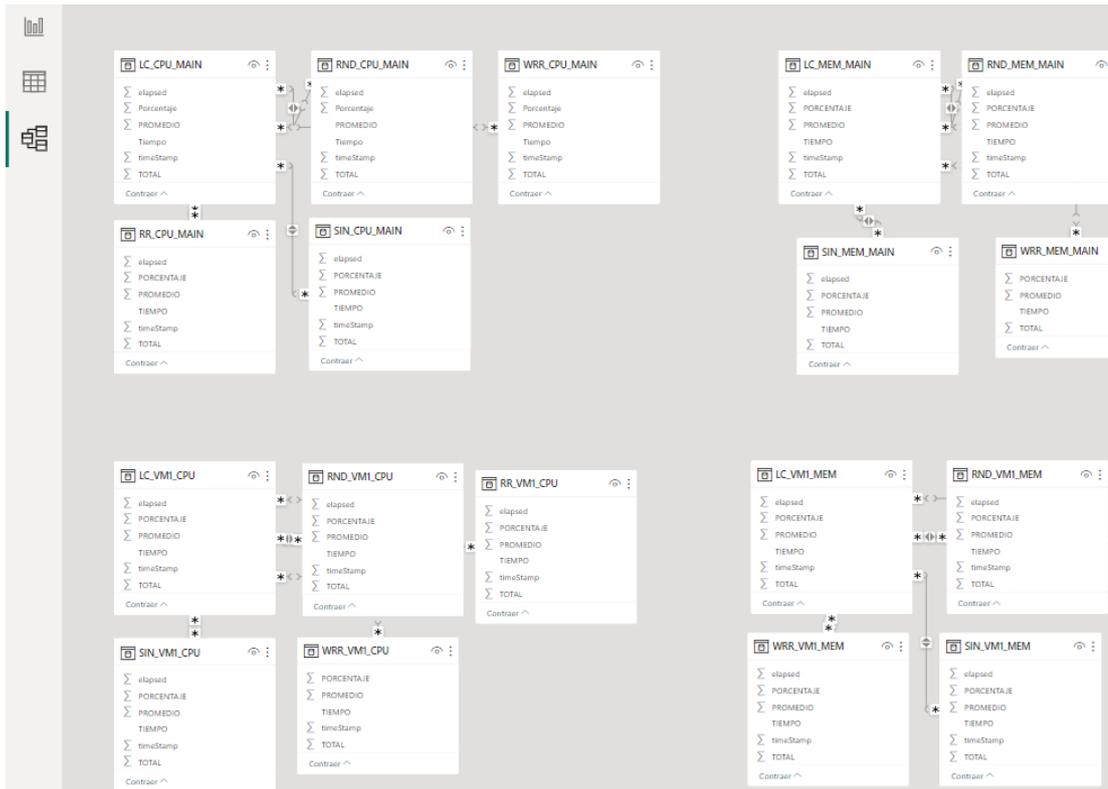


Figura 26. Relación de datos en Power BI

Con todas las relaciones establecidas, se pueden generar los gráficos correspondientes. En la **figura 27** se muestra la forma en que deben colocarse los elementos en los ejes x e y. En el eje x se debe colocar el tiempo, que debe ser representado en una línea de 0 a 11 horas, y en el eje y se deben colocar cada uno de los algoritmos para que puedan ser representados en esa línea de tiempo.

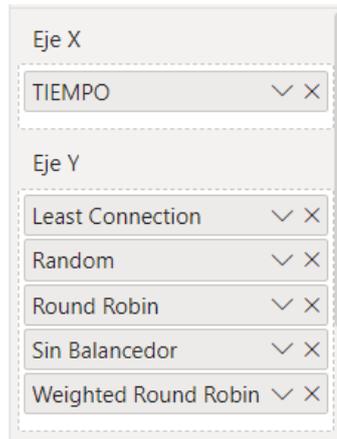
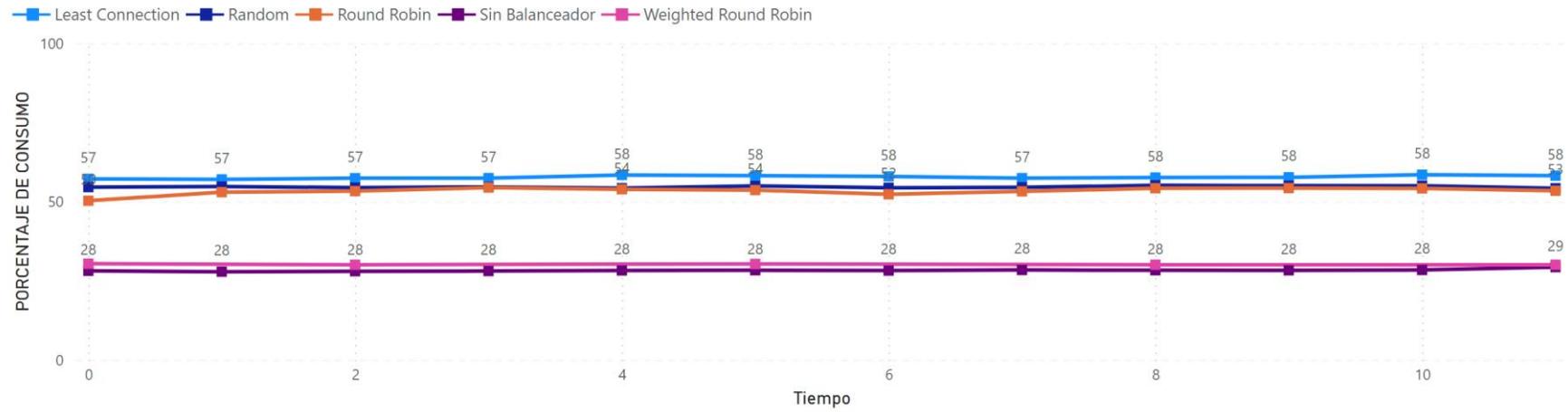


Figura 27. Ítems del eje x; eje y

En este sentido, se observa en la **figura 28** el gráfico que representa el porcentaje promedio de consumo de CPU y memoria que ha tenido la máquina principal (Main) en un intervalo de 12 horas para cada algoritmo balanceador de carga. En la **figura 29** y **30** se presenta la escala de consumo por tiempo de VM1 y VM2 respectivamente. Por último, la **figura 31** muestra el consumo de disco duro de Main, pero como se puede observar, no presenta una gráfica determinante para tomar una decisión en rendimiento, por lo cual no se la tomará en cuenta.

CPU PRINCIPAL



MEMORIA PRINCIPAL

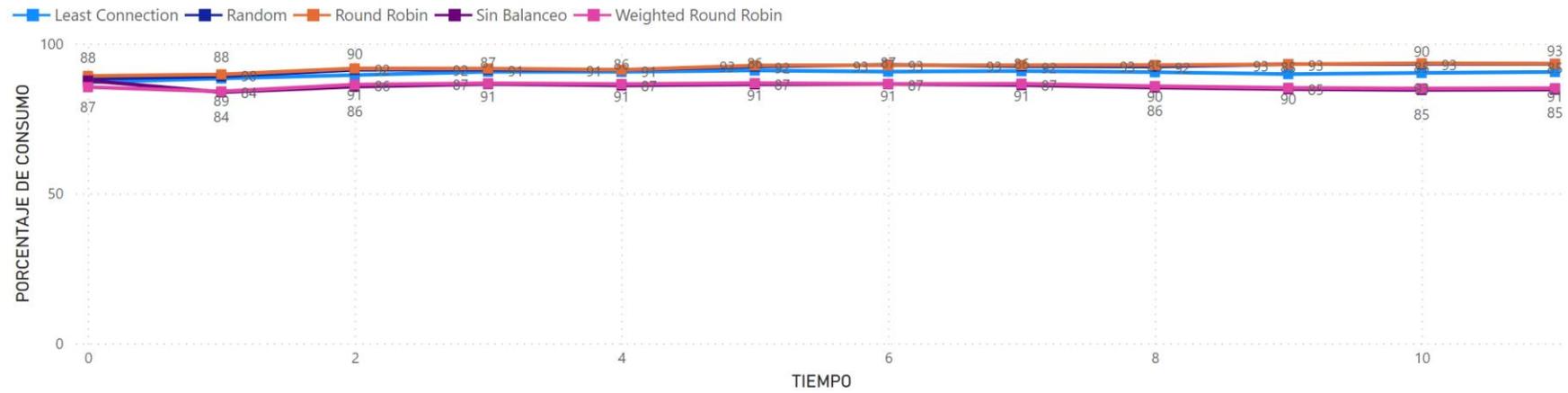
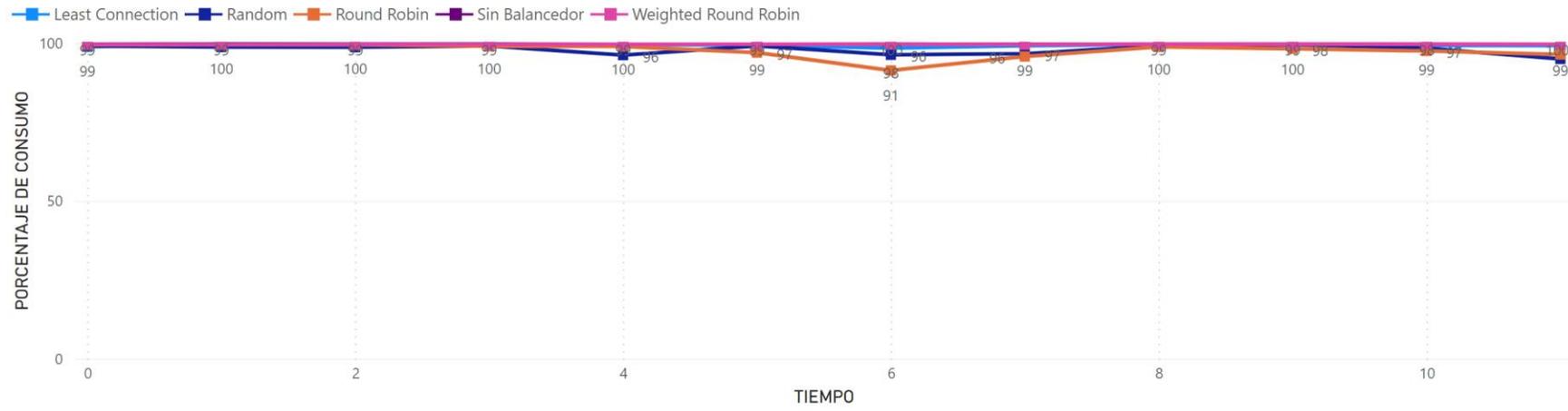


Figura 28. Gráfica de tiempo de Main

CPU VM1



MEMORIA VM1

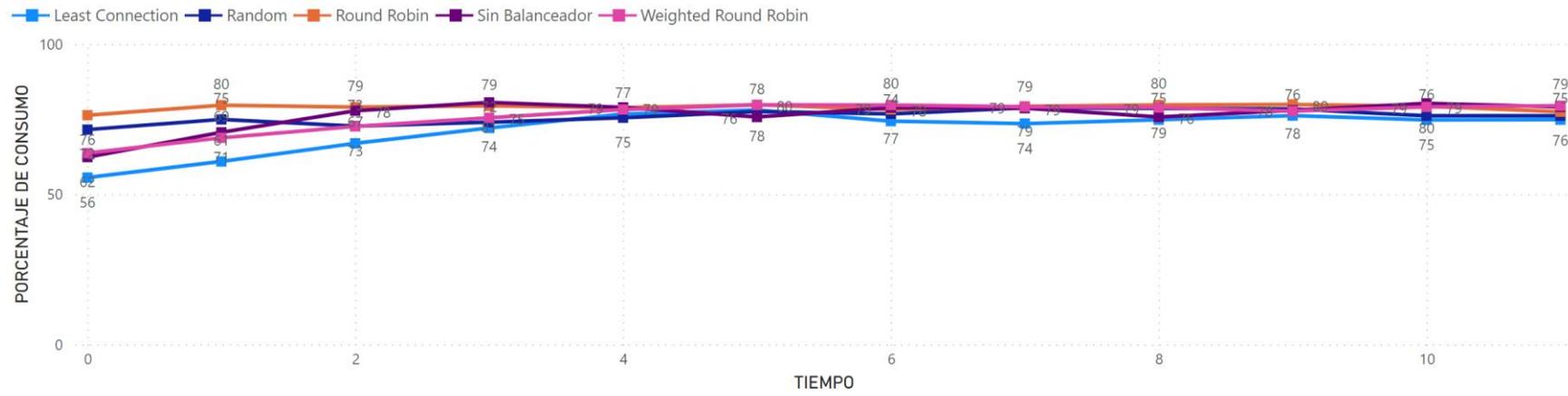
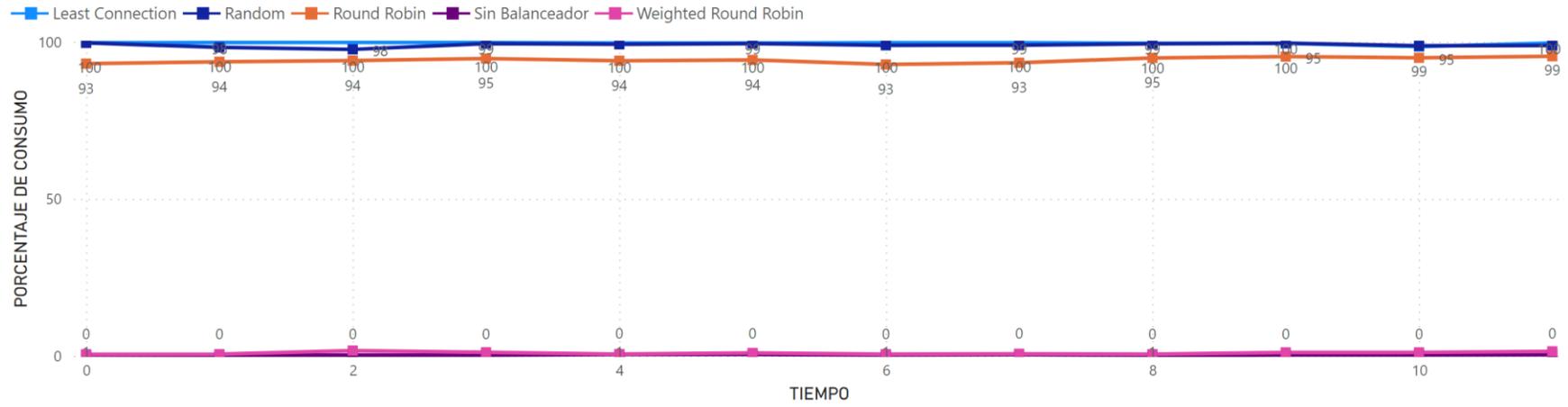


Figura 29. Gráfica de tiempo de VM1

CPU VM2



MEMORIA VM2

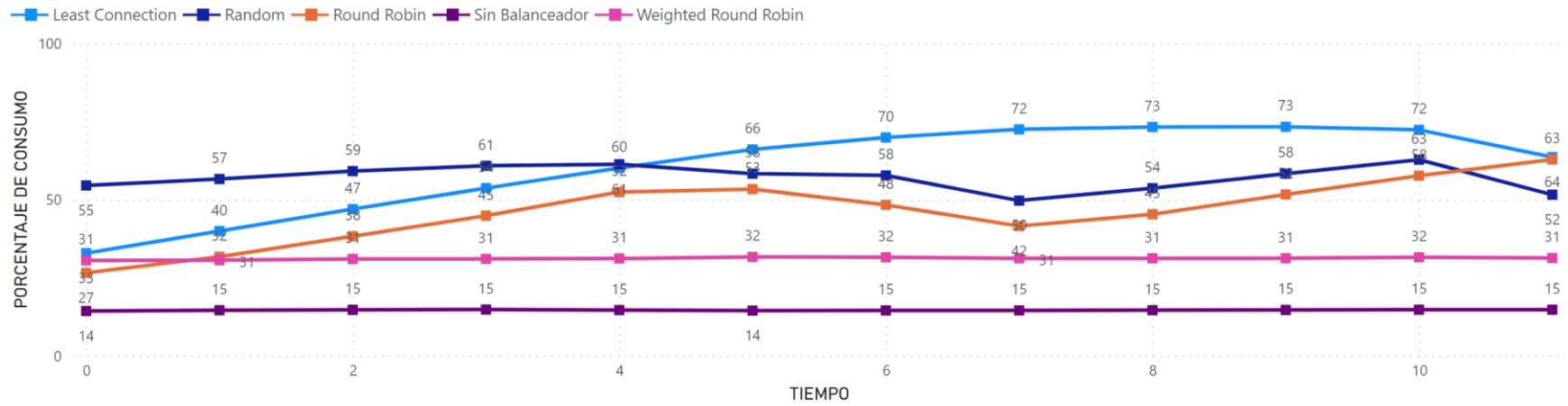


Figura 30. Gráfica de tiempo de VM2

1ra EJECUCION, 2da EJECUCION, 3ra EJECUCION, 4ta EJECUCION y TOTAL por TIEMPO

1ra EJECUCION 2da EJECUCION 3ra EJECUCION 4ta EJECUCION TOTAL

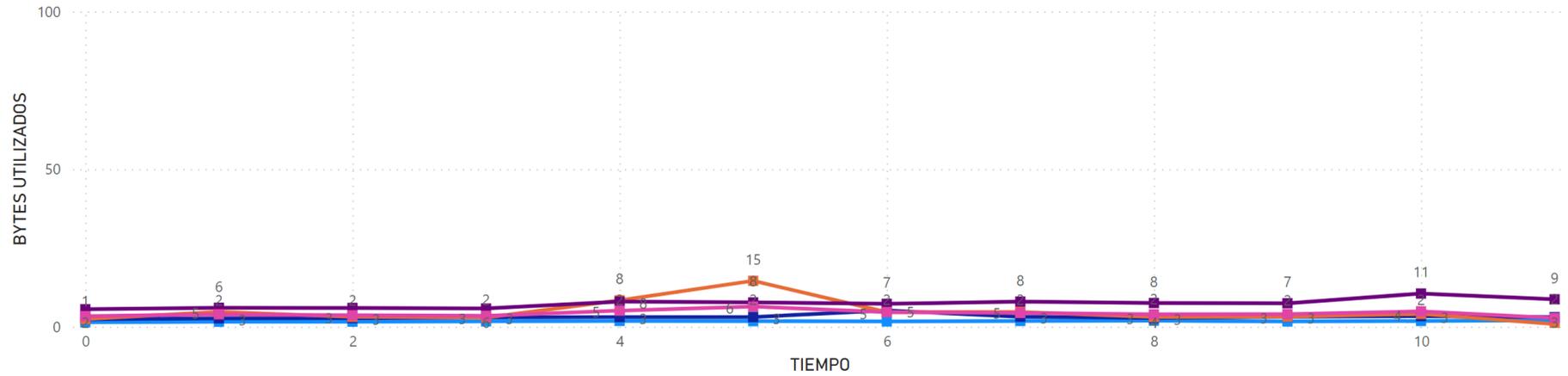


Figura 31. Gráfica de tiempo del disco duro de la máquina física

3.2.5 Realizar los gráficos de barras comparativos en Tableau

Para realizar los gráficos de barras en Tableau, primero se debe importar la hoja de Excel como un origen de datos que contenga los resultados promediados mostrados en la **figura 21**. Una vez hecho esto, los ítems aparecerán en una nueva hoja de gráficos, mostrando las filas por números. La primera fila (F1) representa la métrica de CPU, la segunda fila (F2) representa la máquina virtual, la tercera fila (F3) representa el algoritmo con sus cuatro ejecuciones, y la cuarta fila (F4) representa el promedio de cada ejecución. La **figura 32** muestra la representación de cada una de las filas importadas en Tableau.

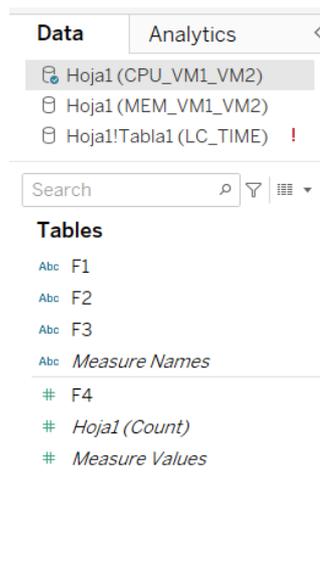


Figura 32. Ítems para las gráficas en Tableau

Ahora que se han importado estos valores, se puede realizar la gráfica deseada en Tableau. Las tres primeras filas que son categóricas se colocan en la parte de columnas, y las filas serán los valores que se tienen en la F4. En el apartado de métricas, se indica que los colores de cada columna serán definidos por el algoritmo al que pertenecen, y los valores de cada columna se verán representados por el valor F4. El resultado es el

gráfico mostrado en la **figura 33** que representa el consumo de CPU dividido en cada máquina virtual de acuerdo con cada algoritmo.

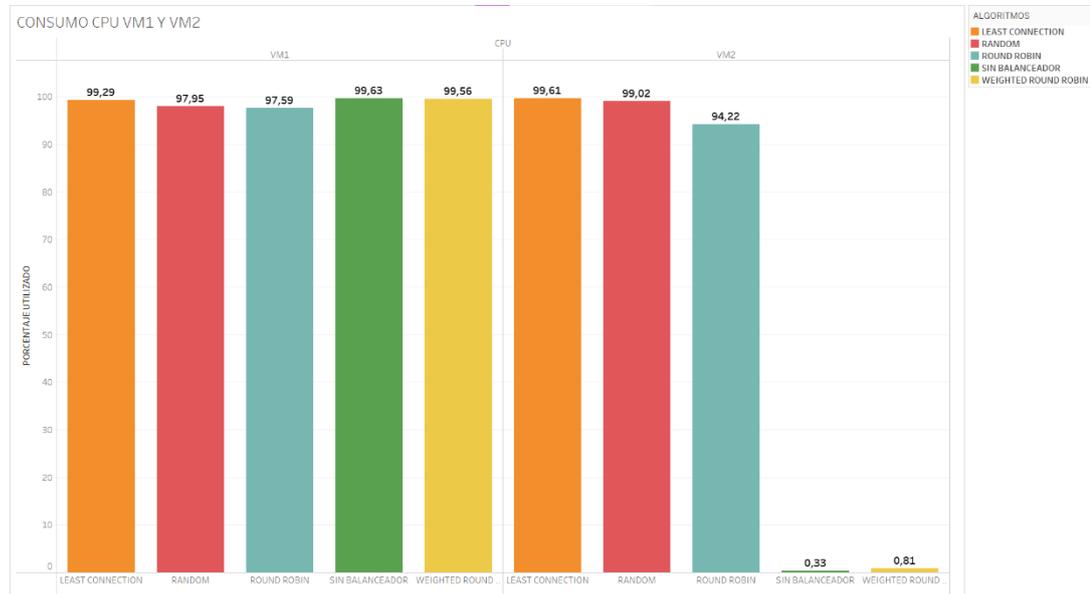


Figura 33. Gráfica del consumo de CPU en VM1 y VM2

Ahora con respecto al uso de memoria en la **figura 34**, se deben seguir los mismos pasos que se hicieron para el consumo de CPU en Tableau de la **figura 33**. La diferencia es que esta vez se utiliza un archivo diferente, ya que la métrica que contiene es la de memoria (RAM) en lugar de CPU.

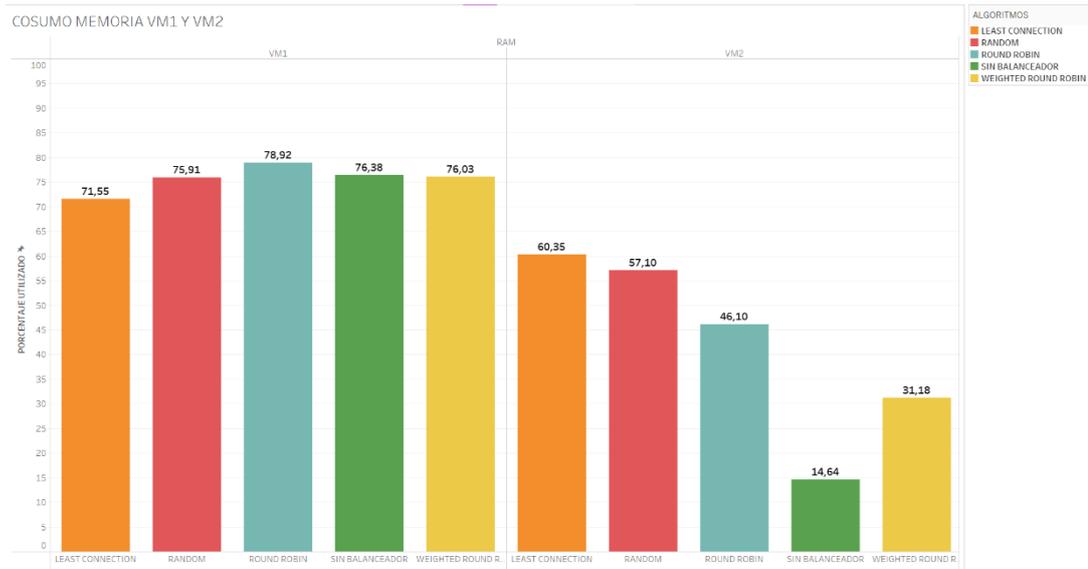


Figura 34. Gráfica del consumo de memoria en VM1 y VM2

Para graficar el tiempo de respuesta que ha tenido cada página en Tableau, se debe importar el último archivo normalizado de Excel que se presenta en la **figura 24**. Una vez hecho esto, los ítems para seleccionar son los que se muestran en la **figura 35**. En la parte de columnas, se debe escoger el nombre de la página (label), mientras que en la parte de filas se escogerá el tiempo de respuesta (average). El color de cada columna será representado por el algoritmo al que pertenece.

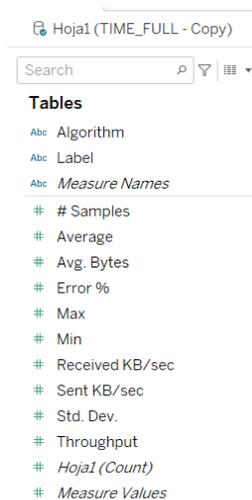


Figura 35. Ítems para la gráfica de tiempos de respuesta

En la **figura 36** se presenta el gráfico de tiempo de respuesta acumulada en milisegundos de cada página, está determinada por la suma de los tiempos de respuesta de cada algoritmo.

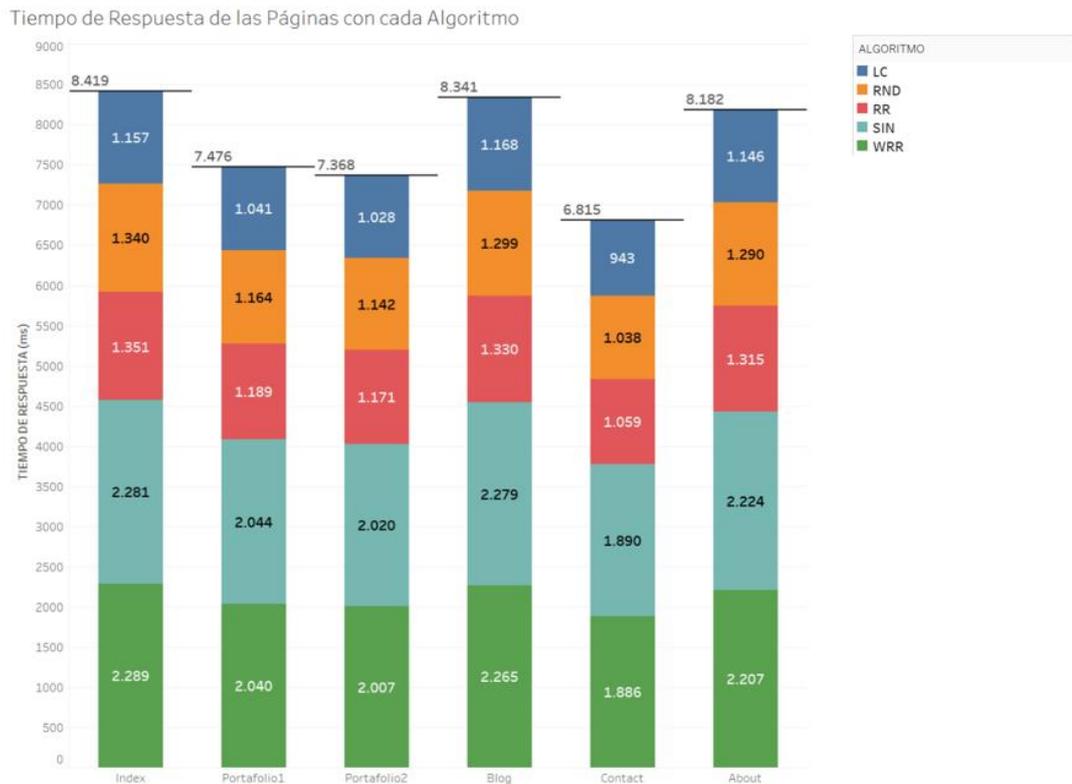


Figura 36. Gráfica de tiempo de respuesta de cada página

En la **figura 37** se presenta un gráfico de barras que tiene como objetivo mostrar el tiempo en milisegundos que ha tomado cada algoritmo balanceador de carga para satisfacer la presentación de todas las páginas por usuario que se ha conectado a las mismas.

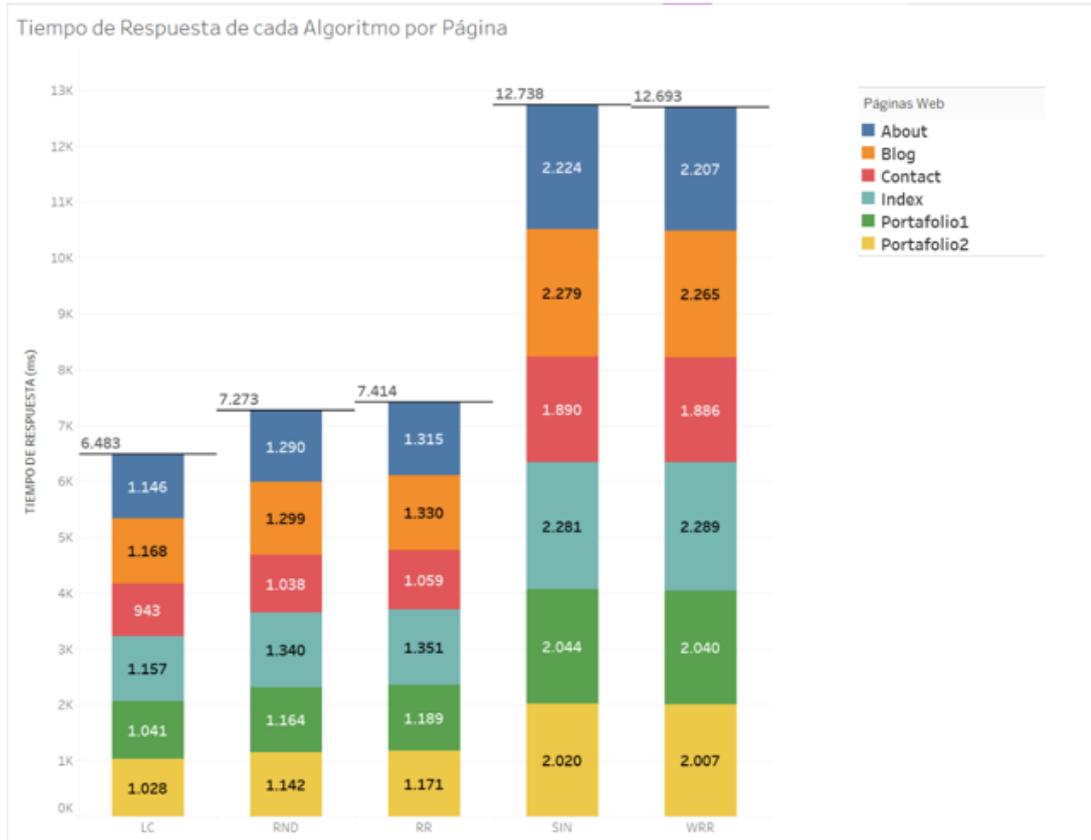


Figura 37. Gráfica de tiempo de respuesta por algoritmo

Como gráfica final, en la **figura 38** se presenta el número de instancias procesadas por algoritmo. Para esto se usó el mismo archivo de datos y se cambió el parámetro de (average) por el de número de interacciones (#samples) las cuales se representan en una escala de millones en el mismo tiempo de 12 horas.

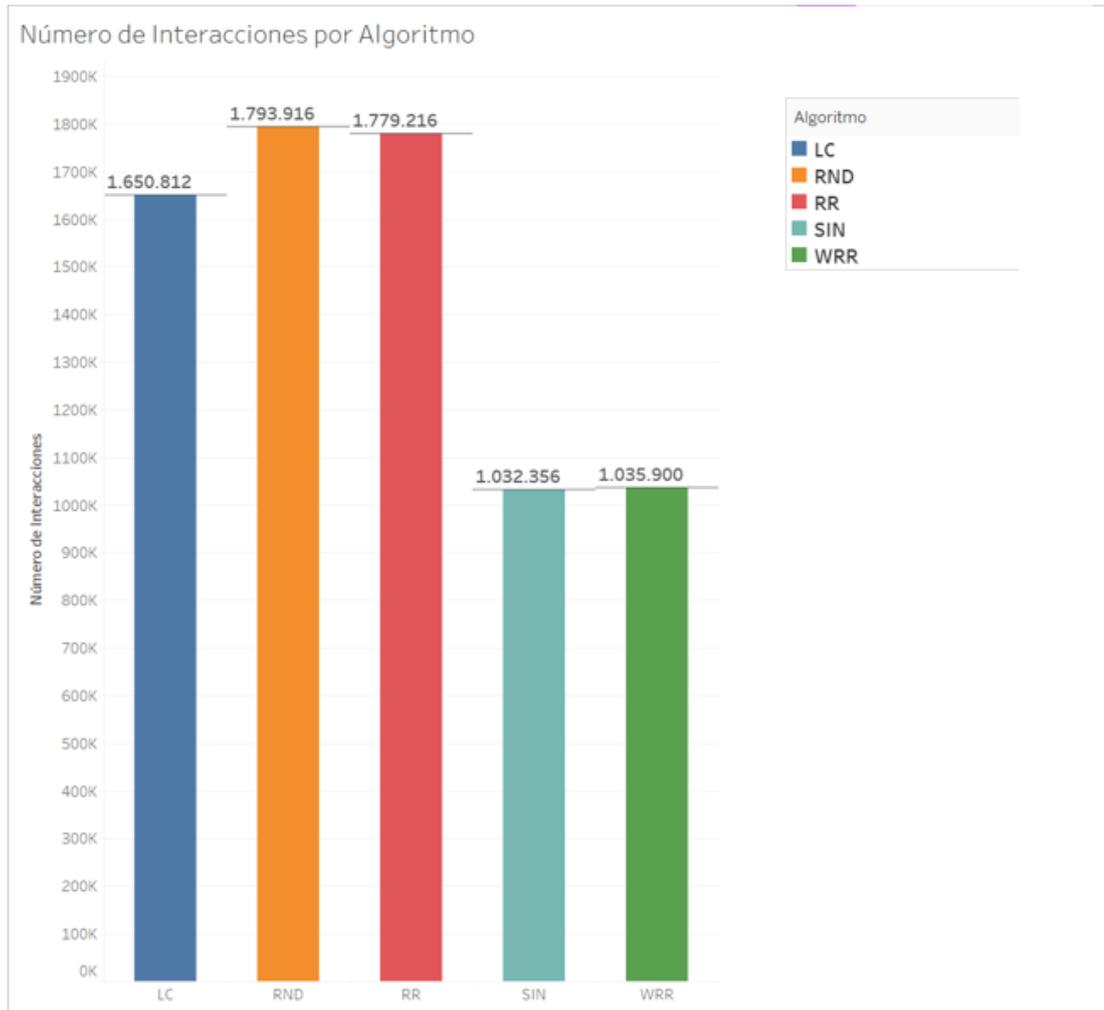


Figura 38. Gráfica de la cantidad de interacciones procesadas por algoritmo

3.2.6 Obtención de ANOVA en R Studio

Para poder realizar el cálculo de correlaciones se usa el mismo archivo de promedios por porcentaje de uso en cada algoritmo presentado en la **figura 21**. Teniendo en cuenta esto, se puede obtener cuadros de ANOVA, comparando cada uno de los algoritmos por métrica (CPU y Memoria) para poder entender mejor su comportamiento. El código para obtener estos ANOVA se encuentra explicado en detalle en el **anexo 32**.

En la **figura 39, 40, 41 y 42** se presentan cuadros que tienen los siguientes componentes de un ANOVA: diff (diferencia entre las medias de los dos grupos), lwr

(punto final inferior del intervalo de confianza al 95% por defecto), upr (punto final superior del intervalo de confianza al 95% por defecto) y p adj (valor p después del ajuste para las comparaciones múltiples).

La correlación de los algoritmos balanceadores de carga de la VM1 con la métrica de CPU se presenta en la **figura 39**.

```
Fit: aov(formula = Value ~ Algorithm, data = cpuvm, subset = cpuvm$Machine == "VM1")

$Algorithm
              diff          lwr          upr      p adj
RANDOM-LEAST CONNECTION    -1.33935333 -2.5615677 -0.1171389 0.0286017
ROUND ROBIN-LEAST CONNECTION -1.69842621 -2.9206406 -0.4762118 0.0049492
SIN BALANCEADOR-LEAST CONNECTION  0.33946524 -0.8827492  1.5616796 0.9078068
WEIGHTED ROUND ROBIN-LEAST CONNECTION  0.26636427 -0.9558501  1.4885787 0.9593155
ROUND ROBIN-RANDOM        -0.35907288 -1.5812873  0.8631415 0.8897424
SIN BALANCEADOR-RANDOM    1.67881857  0.4566042  2.9010330 0.0054477
WEIGHTED ROUND ROBIN-RANDOM  1.60571760  0.3835032  2.8279320 0.0077949
SIN BALANCEADOR-ROUND ROBIN  2.03789145  0.8156770  3.2601059 0.0009621
WEIGHTED ROUND ROBIN-ROUND ROBIN  1.96479048  0.7425761  3.1870049 0.0013623
WEIGHTED ROUND ROBIN-SIN BALANCEADOR -0.07310097 -1.2953154  1.1491134 0.9997072
```

Figura 39. ANOVA VM1-CPU

La correlación de algoritmos de VM2 usando CPU se muestran en la **figura 40**.

```
Fit: aov(formula = Value ~ Algorithm, data = cpuvm, subset = cpuvm$Machine == "VM2")

$Algorithm
              diff          lwr          upr      p adj
RANDOM-LEAST CONNECTION    -0.5825604 -1.2195615  0.05444067 0.0811024
ROUND ROBIN-LEAST CONNECTION -5.3869012  -6.0239022  -4.74990006 0.0000000
SIN BALANCEADOR-LEAST CONNECTION -99.2772208 -99.9142219 -98.64021974 0.0000000
WEIGHTED ROUND ROBIN-LEAST CONNECTION -98.7961621 -99.4331632 -98.15916100 0.0000000
ROUND ROBIN-RANDOM        -4.8043407  -5.4413418  -4.16733963 0.0000000
SIN BALANCEADOR-RANDOM    -98.6946604 -99.3316615 -98.05765931 0.0000000
WEIGHTED ROUND ROBIN-RANDOM -98.2136017 -98.8506028 -97.57660057 0.0000000
SIN BALANCEADOR-ROUND ROBIN -93.8903197 -94.5273208 -93.25331858 0.0000000
WEIGHTED ROUND ROBIN-ROUND ROBIN -93.4092609 -94.0462620 -92.77225984 0.0000000
WEIGHTED ROUND ROBIN-SIN BALANCEADOR  0.4810587  -0.1559424  1.11805984 0.1884655
```

Figura 40. ANOVA VM2-CPU

En la **figura 41** se puede observar el anova de VM1 con la métrica de Memoria.

```

Fit: aov(formula = Value ~ Algorithm, data = memvm, subset = memvm$Machine == "VM1")

$Algorithm
              diff          lwr          upr          p adj
RANDOM-LEAST CONNECTION      4.3578125 -5.424113 14.139737 0.6511932
ROUND ROBIN-LEAST CONNECTION  7.3718160 -2.410109 17.153741 0.1899486
SIN BALANCEADOR-LEAST CONNECTION 4.8274905 -4.954434 14.609415 0.5636850
WEIGHTED ROUND ROBIN-LEAST CONNECTION 4.4835248 -5.298400 14.265450 0.6278061
ROUND ROBIN-RANDOM          3.0140035 -6.767921 12.795928 0.8721444
SIN BALANCEADOR-RANDOM      0.4696780 -9.312247 10.251603 0.9998775
WEIGHTED ROUND ROBIN-RANDOM  0.1257123 -9.656213  9.907637 0.9999994
SIN BALANCEADOR-ROUND ROBIN -2.5443255 -12.326250  7.237600 0.9256103
WEIGHTED ROUND ROBIN-ROUND ROBIN -2.8882912 -12.670216  6.893634 0.8879885
WEIGHTED ROUND ROBIN-SIN BALANCEADOR -0.3439658 -10.125891  9.437959 0.9999646

```

Figura 41. ANOVA VM1-Memoria

Finalmente se muestra la **figura 42** en la cual está presente el anova de VM2 con la métrica de Memoria.

```

Fit: aov(formula = Value ~ Algorithm, data = memvm, subset = memvm$Machine == "VM2")

$Algorithm
              diff          lwr          upr          p adj
RANDOM-LEAST CONNECTION      -3.245622 -23.902607 17.411364 0.9875877
ROUND ROBIN-LEAST CONNECTION -14.251348 -34.908334  6.405637 0.2580140
SIN BALANCEADOR-LEAST CONNECTION -45.707742 -66.364727 -25.050756 0.0000483
WEIGHTED ROUND ROBIN-LEAST CONNECTION -29.169537 -49.826522  -8.512552 0.0043277
ROUND ROBIN-RANDOM          -11.005727 -31.662712  9.651259 0.4934659
SIN BALANCEADOR-RANDOM      -42.462120 -63.119105 -21.805135 0.0001103
WEIGHTED ROUND ROBIN-RANDOM -25.923916 -46.580901  -5.266930 0.0110891
SIN BALANCEADOR-ROUND ROBIN -31.456393 -52.113379 -10.799408 0.0022418
WEIGHTED ROUND ROBIN-ROUND ROBIN -14.918189 -35.575174  5.738796 0.2215090
WEIGHTED ROUND ROBIN-SIN BALANCEADOR 16.538205  -4.118781 37.195190 0.1496600

```

Figura 42. ANOVA VM2-Memoria

3.2.7 Definición del algoritmo optimo

Una vez obtenidos los gráficos y los resultados de las correlaciones de estos, se procederá a estudiarlos para entender cuál tiene el mayor beneficio con respecto a la optimización de tiempo de respuesta y uso de recursos.

En la **figura 28** la única diferencia aparente es en la parte de CPU, en la que el algoritmo "Weighted Round Robin" (WRR) presenta el menor consumo a lo largo de toda la línea de tiempo. En cuanto a la memoria, no se observan diferencias.

En la **figura 29** no existen diferencias significativas en ninguna de las gráficas tanto de CPU como en memoria.

En la **figura 30** se observa una diferencia significativa nuevamente con el algoritmo WRR, ya que se le brinda poco uso. Esto se explica por el peso asignado a cada servidor en el momento de la configuración. El peso se asignó al servidor de VM1, y por esta razón, a no ser que VM1 colapse, entonces VM2 no entra en funcionamiento, lo que explica su casi nula actividad. En segundo lugar, con una pequeña diferencia, se puede observar que el algoritmo Round Robin (RR) presenta un consumo ligeramente menor en comparación con los otros dos algoritmos: Least Connection (LC) y Random (RND).

La representación de la **figura 31** corresponde al disco duro en la máquina principal, sin embargo, no se tomará en cuenta ya que su uso es casi nulo a lo largo de toda la línea de tiempo, lo que significa que no es determinante al momento de tomar una decisión.

Al analizar la gráfica de la **figura 33** se puede observar que en las primeras cinco columnas que corresponden a VM1, no existe una diferencia significativa entre los algoritmos. Sin embargo, en la segunda porción correspondiente a VM2, claramente se puede apreciar que el consumo del algoritmo WRR es casi nulo en comparación con los demás algoritmos. Además, se puede observar una pequeña diferencia con el algoritmo RR.

En la **figura 34** se pueden encontrar diferencias. En VM1, el algoritmo LC presenta una ligera mejora de optimización con respecto al consumo de memoria. En cuanto a VM2, nuevamente el algoritmo WRR obtiene el menor consumo, seguido del algoritmo RR. Sin embargo, a partir de esta gráfica se puede inferir que el algoritmo

LC es el que mejor equilibra el consumo de memoria, ya que en ambas máquinas presenta un porcentaje equivalente.

En el gráfico de tiempos de respuesta por página que se muestra en la **figura 36** no se observa una diferencia importante en el tiempo de presentación de las páginas. La página de "contact" es la que presenta la menor tardanza en responder, ya que solo se ingresan datos y no carga imágenes. En contraste, la página "index" tarda más en cargar debido a que contiene imágenes y animaciones al ser visitada, pero esto no genera una tardanza alarmante.

En cuanto a los tiempos de respuesta de los algoritmos representados en la **figura 37** se observan diferencias significativas. El algoritmo LC presenta el menor tiempo de respuesta total, con un total de 6,4 segundos. En contraste, el algoritmo WRR registra un tiempo de 12,6 segundos, casi el doble de tiempo por página. Esto puede deberse a la inactividad de VM2 con este algoritmo. El algoritmo LC mantuvo sus cargas en servidores casi similares en todo momento, por lo que dichas cargas se distribuyeron equitativamente y se refleja en una mejora de casi la mitad del tiempo que tomaría la carga sin un balanceador.

En el **gráfico 38** se puede observar una diferencia significativa entre los algoritmos WRR y LC, ya que WRR procesó 1035900 interacciones, mientras que LC pudo procesar 1650812 interacciones en el mismo tiempo. Por otro lado, los algoritmos RR y RND han mantenido una constante de casi 1700000 interacciones cada uno. Aunque hay un debate entre los algoritmos WRR y LC, ya que mientras el primero mejora la optimización de los recursos del sistema, el segundo genera un menor tiempo de respuesta.

Es importante tener en cuenta que en el capítulo **2.2.2 población y muestra** se demostró que una pequeña empresa no tendrá muchas más interacciones que unas 12000 en un período de 24 horas, por lo que ambos algoritmos (LC – WRR) pueden satisfacer este número sin problemas. Además, en [10] planteado en el capítulo **1.1.1 Planteamiento del problema**, un usuario Usualmente espera un tiempo inferior a los 2 segundos a que cargue una página web. Este es un fuerte punto a favor del algoritmo LC ya que en todas las páginas muestra un tiempo inferior a 2 segundos, de hecho, sus valores están cerca de 1 segundo por página. Por contraparte el algoritmo WRR solo en una página está dentro de ese intervalo de tiempo.

Para tomar en cuenta las diferencias significativas, se debe observar el p-valor de los cuadros en las **figuras 39-42**. Si el valor es menor a 0,05, sí existe diferencia significativa; de lo contrario, no la hay. Se discriminará la **figura 41** de Memoria en VM1 ya que no presenta diferencias significativas en ningún algoritmo. Con respecto a los demás, se tomarán únicamente los algoritmos que presenten esa diferencia significativa.

En la **figura 39**, se puede ver que estas diferencias significativas no se presentan entre los dos algoritmos LC y WRR por lo que quiere decir que su rendimiento es parecido. En este sentido, los algoritmos que presentan diferencia significativa con respecto a estos son RND y RR. El promedio de LC y WRR (representado en la columna lwr y upr) es superior a estos dos algoritmos por lo que se infiere que su consumo de recursos es mayor.

En la figura 40 todos los algoritmos tienen diferencias significativas, el único caso excepcional es del algoritmo WRR comparado a SIN los cuales presentan un rendimiento similar por eso no tienen una diferencia significativa.

Por último, en la **figura 41** el algoritmo WRR presenta 2 diferencias significativas con respecto a los algoritmos LC y RND, esto indica que tiene un consumo de memoria significativamente menor.

Teniendo en cuenta toda la información extraída de los gráficos y tablas, se puede inferir que el algoritmo WRR es una buena opción; sin embargo, al no ser usado prácticamente en VM2 en ninguna de las ejecuciones, se deduce que no equilibra las cargas de manera equitativa. Por otro lado, el algoritmo LC demuestra equilibrar de manera casi equitativa las cargas, lo que se refleja en la reducción de tiempo de respuesta en casi la mitad a comparación de un entorno sin balanceador o con el algoritmo WRR implantado. Además, muestra un rendimiento casi similar en comparación con RND y RR para procesar interacciones con las páginas. Si bien sacrifica la cantidad total de interacciones, logra una mejora significativa en la reducción del tiempo de respuesta. En resumen, se puede concluir que el algoritmo LC es una opción más efectiva y equitativa en comparación con la opción WRR.

Basado en todo lo revisado, se concluye que para este proyecto y con las configuraciones detalladas en el mismo de las máquinas virtuales, el algoritmo LC (Least Connection) es el que presenta el mejor performance en términos de tiempo de respuesta y optimización de recursos del sistema.

CAPÍTULO IV.- CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Los problemas de rendimiento, escalabilidad y disponibilidad en servidores virtuales de pequeñas empresas en el Ecuador son consecuencia de la creciente demanda de servicios web y la limitación de los recursos físicos disponibles.
- Los algoritmos balanceadores de carga evaluados presentan ventajas y desventajas en cuanto al tiempo de respuesta y la carga del sistema, lo que demuestra la importancia de seleccionar cuidadosamente el algoritmo que mejor se adapte a las necesidades específicas del servidor web y de las máquinas virtuales involucradas.
- El algoritmo Least Connection (LC) presenta el mejor performance entre los algoritmos evaluados, ya que asigna las solicitudes equitativamente en cada máquina virtual, logrando un balanceo más justo y eficiente, mejorando así los tiempos de respuesta de cada página web.

4.2 Recomendaciones

- Realizar más pruebas con diferentes escenarios y configuraciones de máquinas virtuales para validar los resultados obtenidos y compararlos con otros algoritmos balanceadores de carga.
- Implementar el algoritmo Least Connection en un servidor web real para comprobar su efectividad y beneficios en un entorno productivo.
- Estudiar otras técnicas y estrategias para optimizar el tiempo y los recursos en servidores web, como el uso de caché, compresión, seguridad, etc.

REFERENCIAS BIBLIOGRÁFICAS

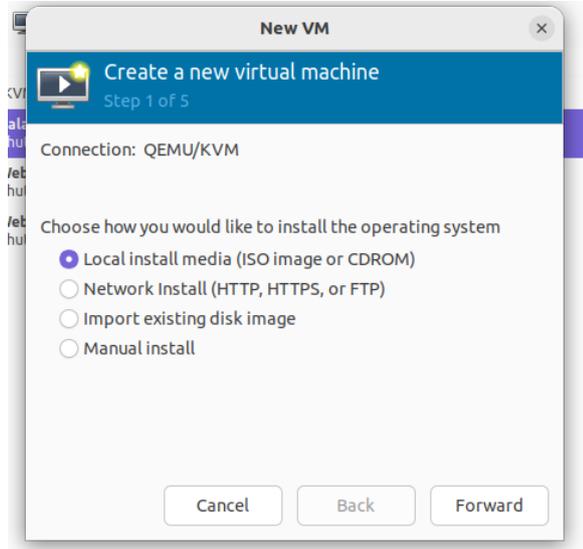
- [1] A.-P. Barzu, M. Carabas, and N. Tapus, “Scalability of a Web Server: How Does Vertical Scalability Improve the Performance of a Server?,” in *2017 21st International Conference on Control Systems and Computer Science (CSCS)*, May 2017, pp. 115–122. doi: 10.1109/CSCS.2017.22.
- [2] J. Martínez Martín, “Dispositivos de bajo rendimiento sobre servidores de aplicación en nube privada,” Bachelor thesis, Universitat Politècnica de Catalunya, 2012. Accessed: Dec. 01, 2022. [Online]. Available: <https://upcommons.upc.edu/handle/2099.1/14907>
- [3] “Google Trends,” *Google Trends*. https://trends.google.es/trends/explore?date=today%205-y&geo=EC&q=%2Fm%2F02_7vm,%2Fm%2F025yt07&hl=es (accessed May 04, 2023).
- [4] W. Chen, Z. Shang, X. Tian, and H. Li, “Dynamic Server Cluster Load Balancing in Virtualization Environment with OpenFlow,” *International Journal of Distributed Sensor Networks*, vol. 11, no. 7, p. 531538, Jul. 2015, doi: 10.1155/2015/531538.
- [5] “Google Trends,” *Google Trends*. <https://trends.google.es/trends/explore?date=today%205-y&geo=EC&q=%2Fm%2F0gl7w,%2Fg%2F120x6v6f&hl=es> (accessed May 04, 2023).
- [6] “Caso de Estudio - Final.pdf.” Accessed: May 11, 2023. [Online]. Available: <http://repositorio.puce.edu.ec/bitstream/handle/22000/14419/Caso%20de%20Estudio%20-%20%20Final.pdf?sequence=1&isAllowed=y>
- [7] J. L. Chunga Zuloeta and W. Chuzón Sanchez, “Comparación de algoritmos de balanceadores de carga utilizando clúster homogéneo en servidores web,” *Repositorio Institucional - USS*, 2017, Accessed: Nov. 15, 2022. [Online]. Available: <http://repositorio.uss.edu.pe/handle/20.500.12802/4667>
- [8] D. C. Sánchez Rodríguez, E. M. Macías López, and Á. Suárez Sarmiento, *Effective load balancing on a LAN-WLAN cluster*. 2003. Accessed: Dec. 01, 2022. [Online]. Available: <https://accedacris.ulpgc.es/jspui/handle/10553/52881>
- [9] “Implementación y configuración de VMware Unified Access Gateway - Unified Access Gateway 3.0”.
- [10] “Top 12 Most Common Website Problems and How to Fix Them,” *Paradox Marketing*. <https://paradoxmarketing.io/capabilities/search-engine-optimization/insights/top-12-most-common-website-problems-and-how-to-fix-them/> (accessed Nov. 13, 2022).
- [11] “Performance of Hybrid Load Balancing Algorithm in Distributed Web Server System | SpringerLink.” <https://link.springer.com/article/10.1007/s11277-018-5758-6> (accessed Apr. 27, 2023).
- [12] H. Babbar, S. Parthiban, G. Radhakrishnan, and S. Rani, “A genetic load balancing algorithm to improve the QoS metrics for software defined networking for multimedia applications,” *Multimed Tools Appl*, vol. 81, no. 7, pp. 9111–9129, Mar. 2022, doi: 10.1007/s11042-021-11467-x.
- [13] A. Jimenez Herrera and C. E. Arreola González, “Implementación de un balanceador de cargas en una red SDN mediante el controlador Ryu,” Dec. 2022, Accessed: Apr. 27, 2023. [Online]. Available: <http://www.ptolomeo.unam.mx:8080/xmlui/handle/RepoFi/18172>

- [14] D. C. Devi and V. R. Uthariaraj, "Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Tasks," *The Scientific World Journal*, vol. 2016, p. e3896065, Feb. 2016, doi: 10.1155/2016/3896065.
- [15] L. F. Cubas Fernández, "ANÁLISIS COMPARATIVO DEL RENDIMIENTO Y EL ESFUERZO MEDIANTE PRUEBAS DE CARGA EN SERVIDORES WEB," *Repositorio Institucional - USS*, 2019, Accessed: Apr. 16, 2023. [Online]. Available: <http://repositorio.uss.edu.pe//handle/20.500.12802/6264>
- [16] P. Kumar and R. Kumar, "Issues and Challenges of Load Balancing Techniques in Cloud Computing: A Survey," *ACM Comput. Surv.*, vol. 51, no. 6, p. 120:1-120:35, Feb. 2019, doi: 10.1145/3281010.
- [17] A. Jyoti, M. Shrimali, S. Tiwari, and H. P. Singh, "Cloud computing using load balancing and service broker policy for IT service: a taxonomy and survey," *J Ambient Intell Human Comput*, vol. 11, no. 11, pp. 4785–4814, Nov. 2020, doi: 10.1007/s12652-020-01747-z.
- [18] M.-L. Chiang, H.-S. Cheng, H.-Y. Liu, and C.-Y. Chiang, "SDN-based server clusters with dynamic load balancing and performance improvement," *Cluster Comput*, vol. 24, no. 1, pp. 537–558, Mar. 2021, doi: 10.1007/s10586-020-03135-w.
- [19] S. D. Canto, *Procesamiento paralelo: teoría y programación*. Sanz y Torres, 2003.
- [20] N. Shah and M. Farik, "Static Load Balancing Algorithms In Cloud Computing: Challenges & Solutions," vol. 4, no. 10, 2015.
- [21] J. W. B. Bedoya, F. F. R. Corchado, A. M. Múnera, and R. E. J. Mejía, "PROPOSICIÓN DE UN MÉTODO PARA BALANCEO DE CARGA EN UN CLUSTER HETEROGÉNEO SIMULADO EN NS2," *Avances en Sistemas e Informática*, vol. 6, no. 1, Art. no. 1, Jan. 2009.
- [22] K. C. Jiménez and R. J. F. Gómez, "LOAD BALANCING: BALANCEO DE CARGA. CONCEPTO, ESTADO DEL ARTE Y APLICABILIDAD EN LINUX Y WINDOWS.," *LOAD BALANCING*, p. 106, 2004.
- [23] "Elastic Load Balancing - Application Load Balancers," p. 152.
- [24] "Elastic Load Balancing - Network Load Balancers," p. 99.
- [25] "Elastic Load Balancing - User Guide," p. 33.
- [26] "Intro to Load Balancing for Developers – The Algorithms," *DevCentral - an F5 Community*, Apr. 01, 2009. <https://community.f5.com/t5/technical-articles/intro-to-load-balancing-for-developers-ndash-the-algorithms/ta-p/273759> (accessed Dec. 01, 2022).
- [27] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Load balancing techniques in cloud computing environment: A review," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 7, pp. 3910–3933, Jul. 2022, doi: 10.1016/j.jksuci.2021.02.007.
- [28] M. D. G. Río, *Tecnologías de Virtualización*. IT Campus Academy, 2014.
- [29] D. Ruest and N. Ruest, *Virtualization: a beginner's guide*. in Network professional's library. New York: McGraw Hill, 2009.
- [30] P. N. M. Pérez, "TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN," p. 44.
- [31] L. Vokorokos, A. Baláž, and N. Ádám, "Secure Web Server System Resources Utilization," *Acta Polytechnica Hungarica*, vol. 12, no. 2, p. 15, 2015.

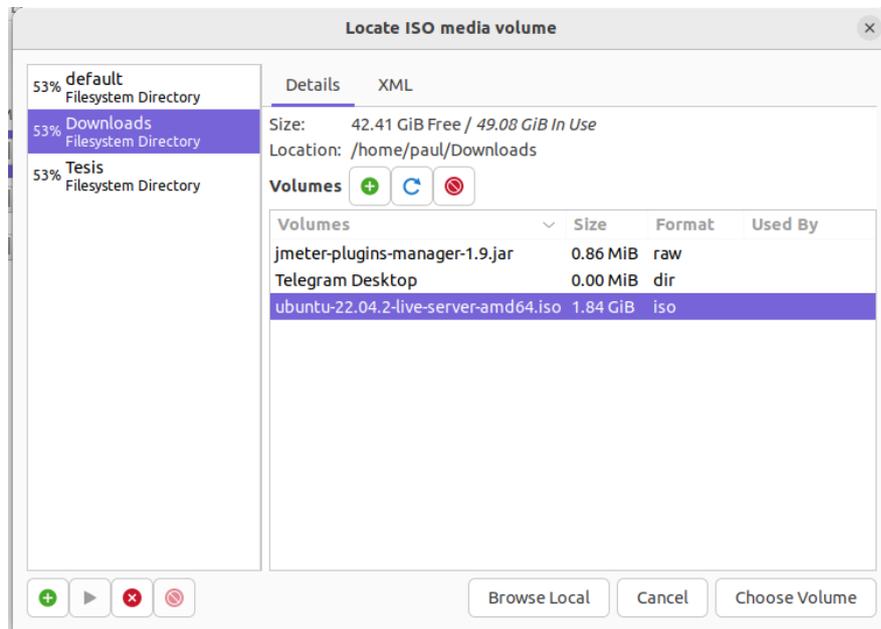
- [32] “Valve Corporation,” *Valve Corporation*. <https://www.valvesoftware.com/es/> (accessed May 07, 2023).
- [33] “Tablas de Steam.” <https://store.steampowered.com/charts/> (accessed May 04, 2023).
- [34] “Encuesta sobre hardware y software de Steam.” <https://store.steampowered.com/hwsurvey/> (accessed May 04, 2023).
- [35] “Mipymes y Organizaciones de Economía Popular y Solidaria son una pieza clave para la economía del país – Servicio Ecuatoriano de Normalización INEN.” <https://www.normalizacion.gob.ec/mipymes-y-organizaciones-de-economia-popular-y-solidaria-son-una-pieza-clave-para-la-economia-del-pais/> (accessed Jul. 17, 2023).
- [36] “¿Qué es Kanban y cómo funciona este método de gestión?” <https://josefacchin.com/metodo-kanban/> (accessed May 15, 2023).

ANEXOS

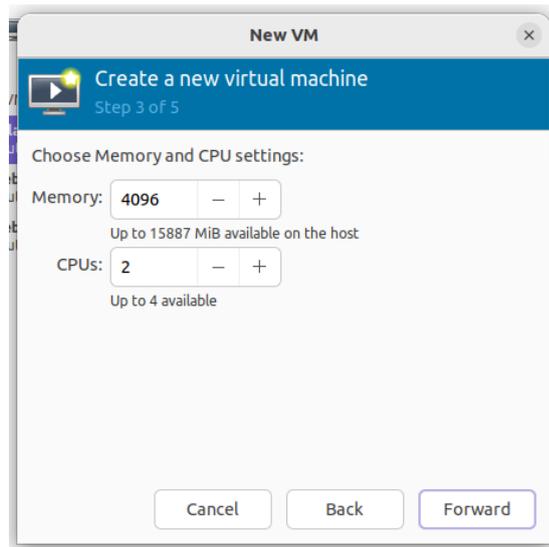
CREAR UNA MÁQUINA VIRTUAL:



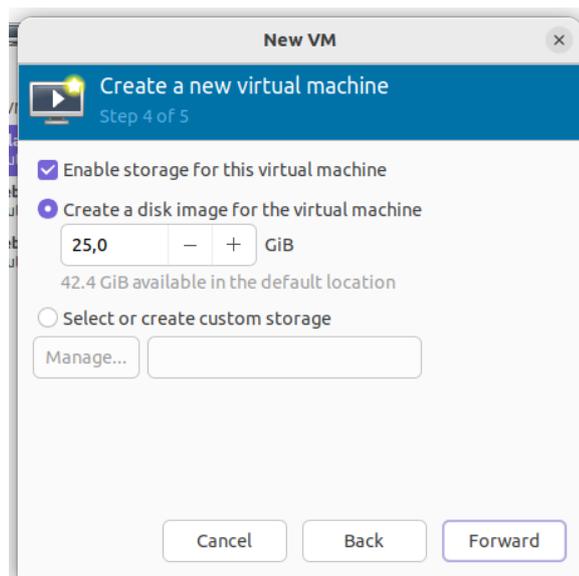
Anexo 1. Crear nueva máquina virtual



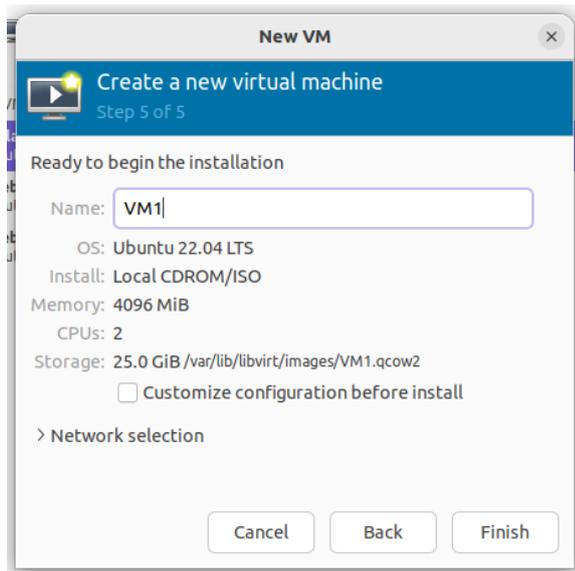
Anexo 2. Selección de ISO



Anexo 3. Selección de cantidad de Memoria y CPUs

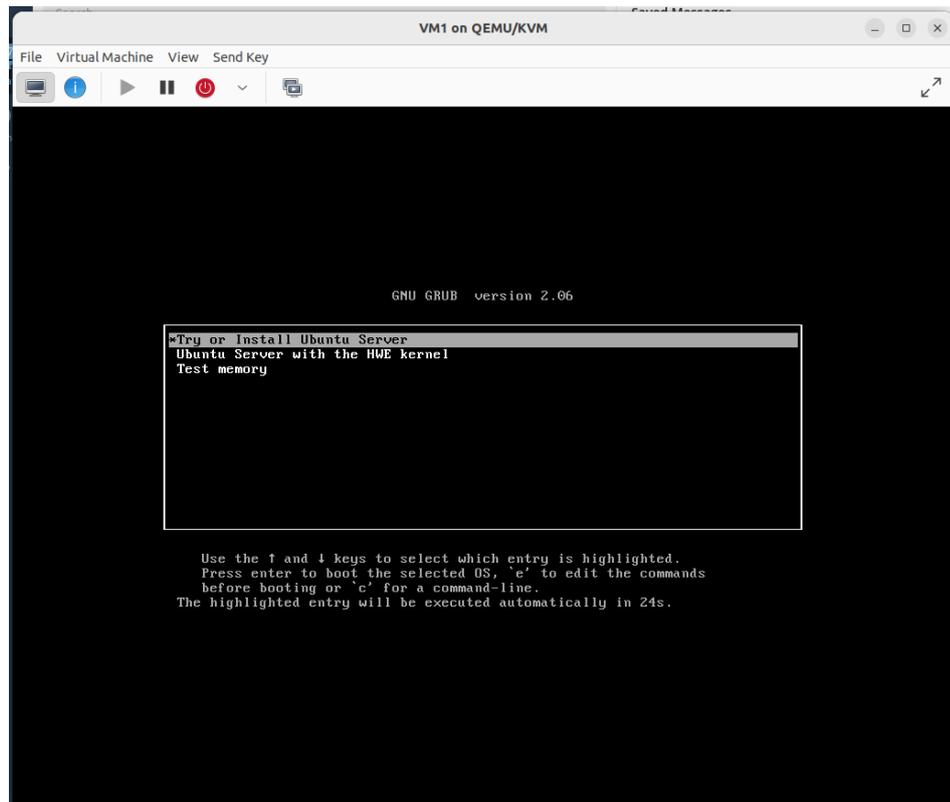


Anexo 4. Crear espacio de Disco

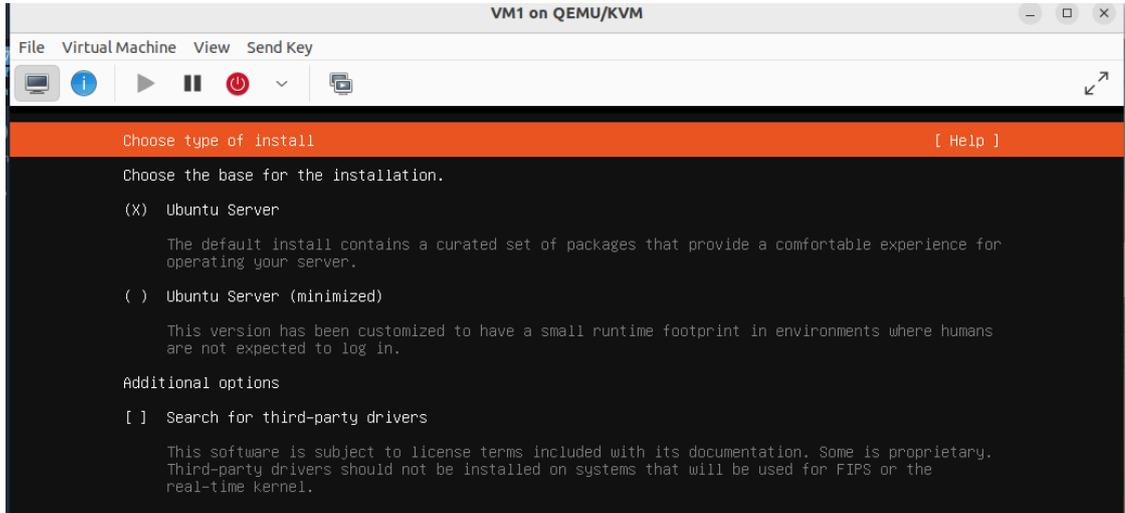


Anexo 5. Nombre de la Máquina virtual y revisión de configuración

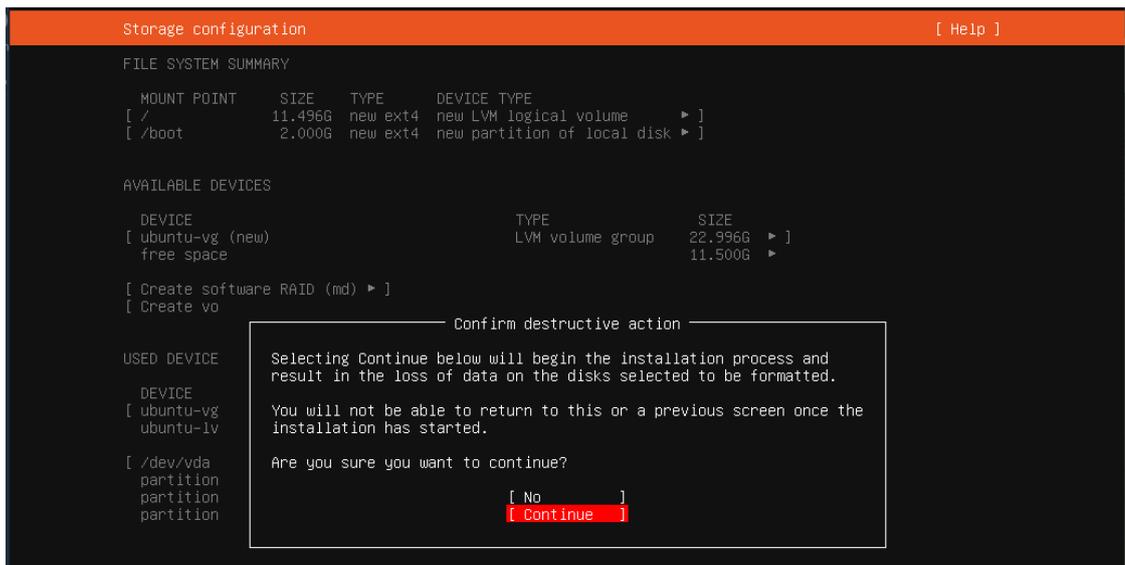
INSTALACIÓN DE UBUNTU SERVER:



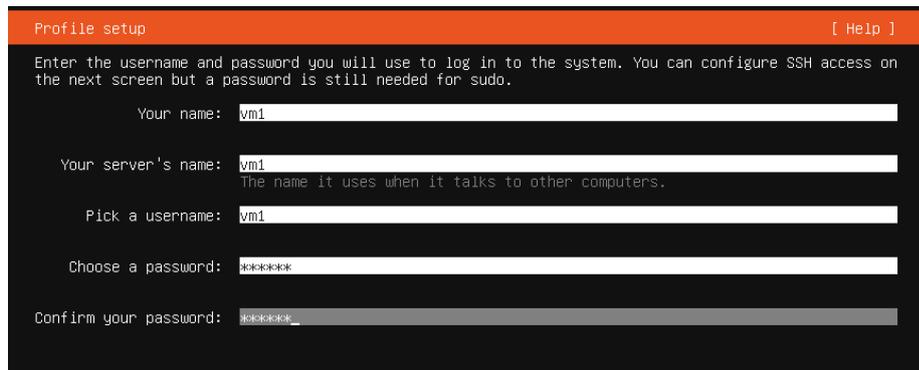
Anexo 6. Selección de instalación de Ubuntu



Anexo 7. Selección de tipo de instalación



Anexo 8. Configuración de almacenamiento



Anexo 9. Configuración de Usuario y Contraseña

CONFIGURACIÓN DE VM1:

Cambiar la configuración de red en esta ruta: `/etc/netplan/00-installer-config.yaml`

Y aplicar los cambios: `netplan apply`

```
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp1s0:
      dhcp4: no
      addresses: [192.168.122.200/24]
      gateway4: 192.168.122.1
      nameservers:
        addresses: [8.8.8.8,192.168.122.1]
  version: 2
~
~
~
~
```

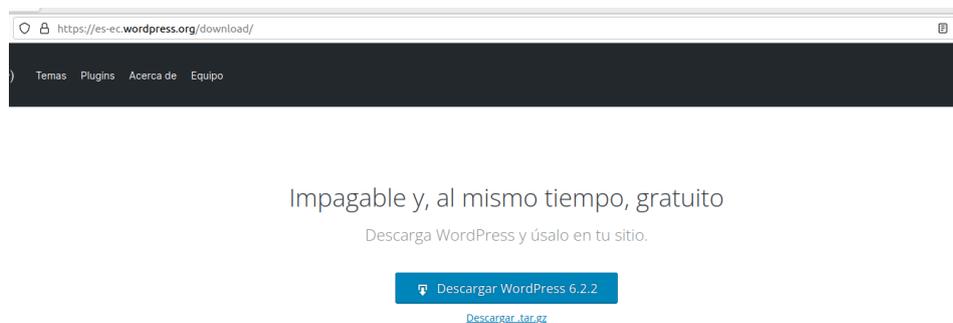
Anexo 10. Configuración de IP estática

Instalar LAMP con el siguiente comando:

```
apt install apache2 mariadb-server php php-curl php-gd php-mbstring php-xml php-xmlrpc php-soap libapache2-mod-php php-mysql -y
```

Realizar la instalación segura de MySQL y configurar un usuario y contraseña para WordPress.

INSTALACIÓN Y CONFIGURACIÓN DE WORDPRESS:

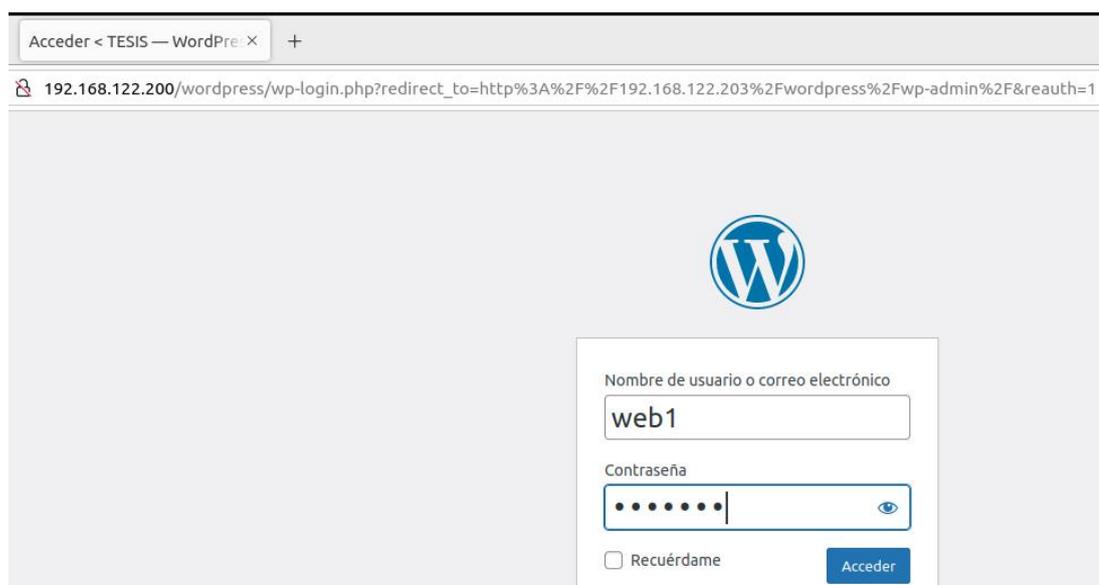


Anexo 11. Descargar WordPress desde la Página oficial

Seguir los pasos de la página oficiales para la instalación:

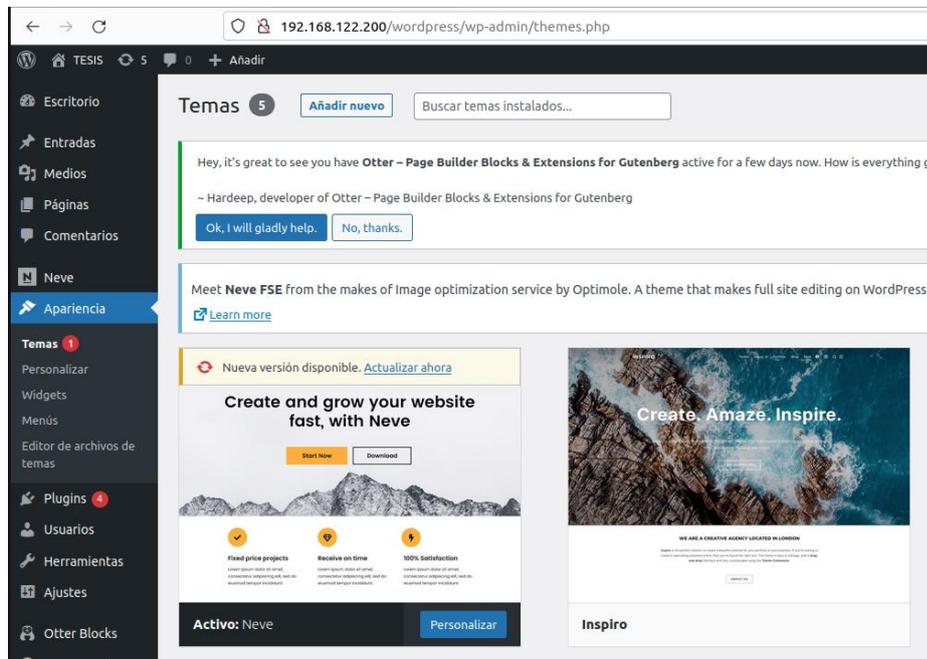
[How to install WordPress | Adv-admin Handbook Handbook | WordPress Developer Resources](#)

Acceder como administrador:



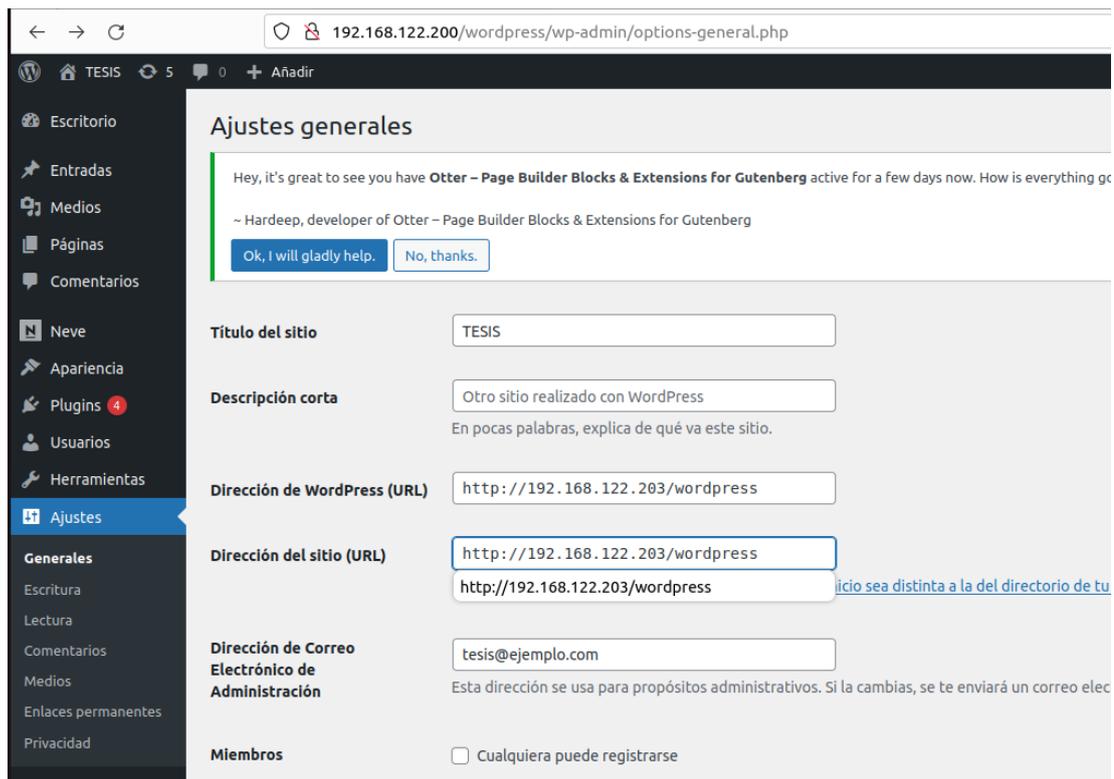
Anexo 12. Ingresar a WordPress con las credenciales

En el apartado de “Apariencia” se puede cambiar el tema de WordPress. Para esta práctica se usó un tema y páginas de uso libre. Las páginas son sencillas e informativas, y se generó una página para subir datos personales en la página de contacto.



Anexo 13. Instalación de un Tema en WordPress

Ya configurado el entorno de pruebas que serán las páginas web, se procede a cambiar la IP de WordPress para que se pueda abrir con la IP del servidor con HAProxy, en este caso será: 192.168.122.203



Anexo 14. Ajustar la IP de WordPress desde la interfaz gráfica

PARA PRUEBAS DE JMETER:

Desde este link se debe descargar y luego descomprimir el agente de servicio para Jmeter en cada una de las máquinas en las que se hará la prueba de rendimiento.

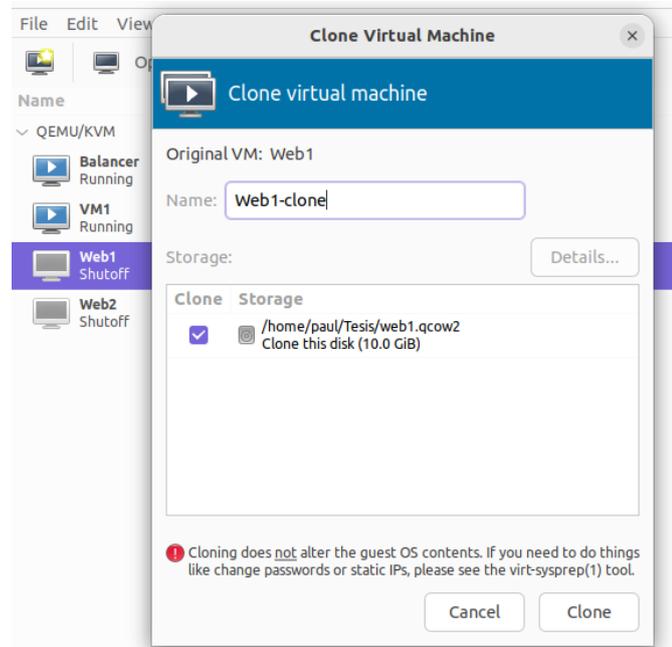
Wget <https://github.com/undera/perfmon-agent/releases/download/2.2.3/ServerAgent-2.2.3.zip>

```
root@web1:~# ls
ServerAgent-2.2.3 ServerAgent-2.2.3.zip snap
root@web1:~# cd ServerAgent-2.2.3/
root@web1:~/ServerAgent-2.2.3# ls
CMDRunner.jar LICENSE ServerAgent.jar lib startAgent.bat startAgent.sh
root@web1:~/ServerAgent-2.2.3#
```

Anexo 15. Agente del Servicio Jmeter

CONFIGURAR VM2:

Se debe clonar la máquina VM1



Anexo 16. Clonar VM1

Cambiar la IP de la máquina y esta nueva IP se la debe cambiar en WordPress. Como no se permite ingresar por medio de la interfaz gráfica, se debe efectuar el cambio en la base de datos directamente.

```
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.6.12-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [wordpress]> SELECT * FROM wp_options WHERE option_name IN('siteurl','home');
+-----+-----+-----+-----+
| option_id | option_name | option_value | autoload |
+-----+-----+-----+-----+
| 2 | home | http://192.168.122.200/wordpress | yes |
| 1 | siteurl | http://192.168.122.200/wordpress | yes |
+-----+-----+-----+-----+
2 rows in set (0.000 sec)

MariaDB [wordpress]> UPDATE wp_options SET option_value='http://192.168.122.201/wordpress' WHERE option_name = 'siteurl';
```

Anexo 17. Cambiar la IP de WordPress directamente en la Base de Datos

```
MariaDB [wordpress]> SELECT * FROM wp_options WHERE option_name IN('siteurl','home');
+-----+-----+-----+-----+
| option_id | option_name | option_value | autoload |
+-----+-----+-----+-----+
|          2 | home       | http://192.168.122.203/wordpress | yes      |
|          1 | siteurl    | http://192.168.122.200/wordpress | yes      |
+-----+-----+-----+-----+
2 rows in set (0.000 sec)
```

Anexo 18. IP del sitio de WordPress

INSTALAR Y CONFIGURAR HAProxy:

Como primer paso se deberá cambiar la IP de la máquina para luego instalar HAProxy.

Los comandos de instalación se pueden encontrar en la página oficial:

[How to Install HAProxy on Ubuntu \(In a Few Simple Steps\)](#)

```
4 vi /etc/netplan/00-installer-config.yaml
5 netplan apply
6 ip add
7 apt-get install --no-install-recommends software-properties-common
8 add-apt-repository ppa:vbernat/haproxy-2.8
9 apt-get install haproxy=2.8.*
```

Anexo 19. Cambiar la IP e instalar HAProxy

Dentro de este directorio /etc/haproxy/haproxy.cfg, se configura las IP de las máquinas virtuales a balancear y se indica el tipo de balanceo que se va a efectuar.

Una vez hechos los cambios se debe reiniciar el servicio con el siguiente comando:

```
systemctl restart haproxy
```

```
frontend web_frontend
  bind *:80
  mode http
  default_backend web_backend

backend web_backend
  mode http
  balance roundrobin
  server web1 192.168.122.200:80 check
  server web2 192.168.122.201:80 check
#   server web3 192.168.122.202:80 check

frontend stats
  mode http
  bind *:8404
  stats enable
  stats uri /stats
  stats refresh 10s
  stats admin if LOCALHOST
```

Anexo 20. Configuración de HAProxy para realizar el balanceo de carga

Para la máquina principal se instala Jmeter desde la página oficial:

[Apache Jmeter – Download Apache Jmeter](#)

Apache JMeter 5.6.2 (Requires Java 8+)

Binaries

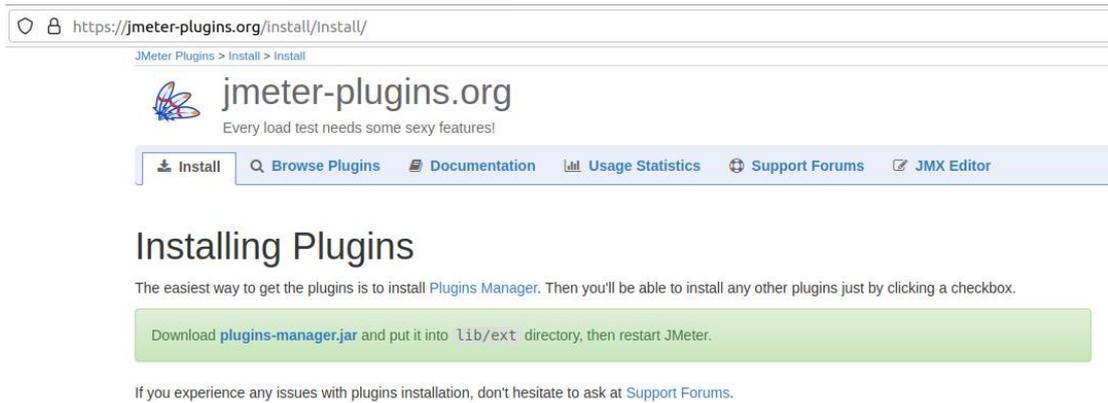
[apache-jmeter-5.6.2.tgz](#) [sha512](#) [pgp](#)

[apache-jmeter-5.6.2.zip](#) [sha512](#) [pgp](#)

Anexo 21. Descargar Jmeter desde la página oficial

También se debe descargar el administrador de plugins de Jmeter:

[Install :: Jmeter-Plugins.org](#)



Anexo 22. Descargar los plugins de Jmeter

Se debe pegar este administrador de plugins en el directorio de instalación de Jmeter:

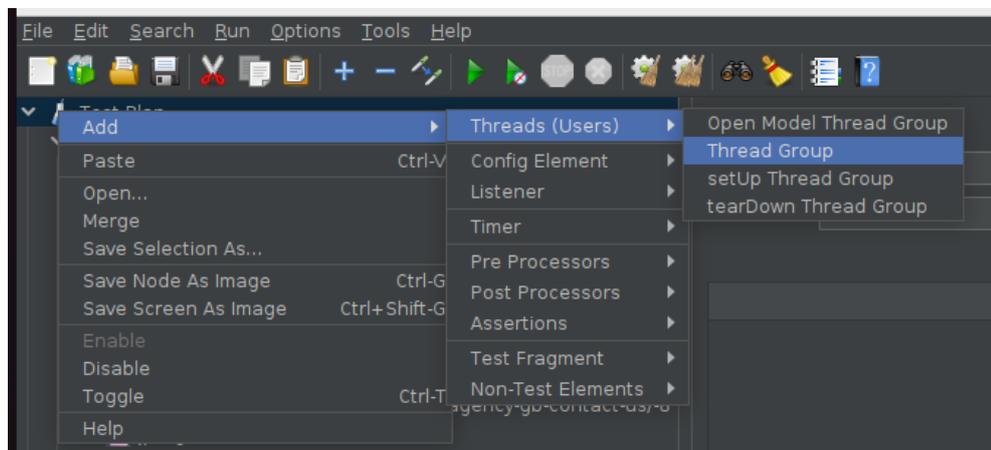
`/usr/share/Jmeter/lib/ext/`

```
root@paul-pc:/lib# cd /usr/share/jmeter/lib/ext/
root@paul-pc:/usr/share/jmeter/lib/ext# ls
ApacheJMeter_bolt.jar      ApacheJMeter_ftp.jar      ApacheJMeter_java.jar     ApacheJMeter_junit.jar
ApacheJMeter_components.jar ApacheJMeter_functions.jar ApacheJMeter_jdbc.jar     ApacheJMeter_ldap.jar
ApacheJMeter_core.jar     ApacheJMeter_http.jar     ApacheJMeter_jms.jar     ApacheJMeter_mail.jar
root@paul-pc:/usr/share/jmeter/lib/ext#
```

Anexo 23. Pegar los plugins en la carpeta de Jmeter

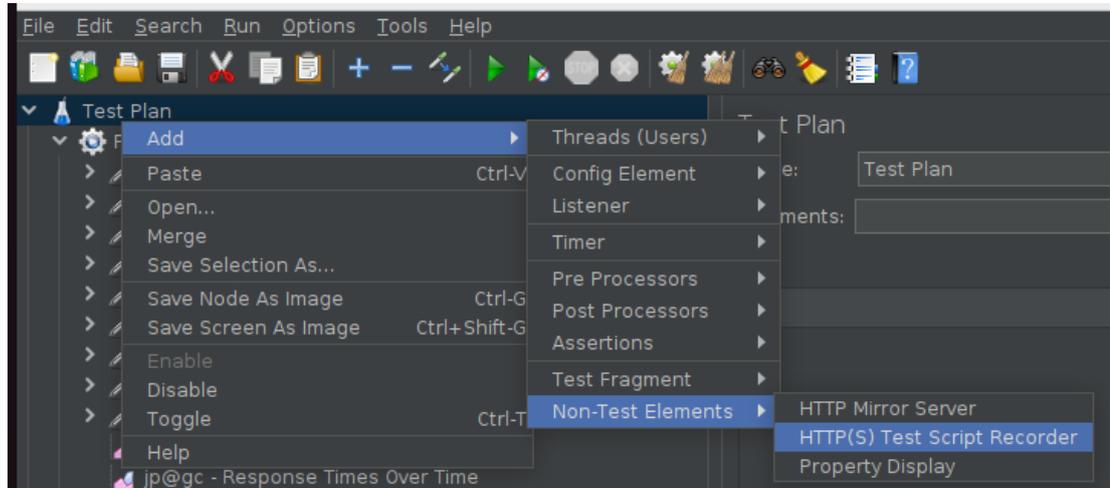
CONFIGURACIÓN DE JMETER:

Crear un nuevo grupo de hilos



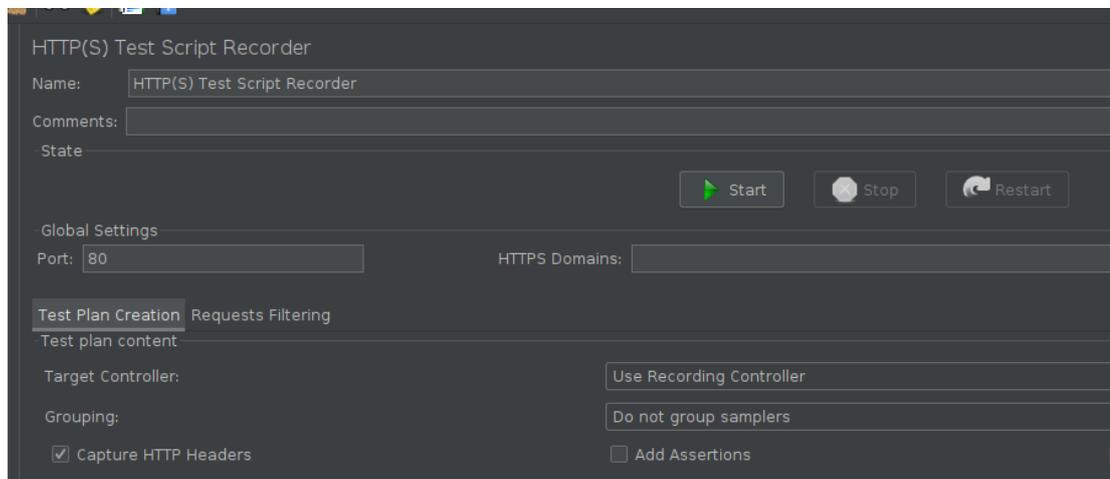
Anexo 24. Crear un nuevo grupo de hilos

Crear una nueva herramienta de testeo



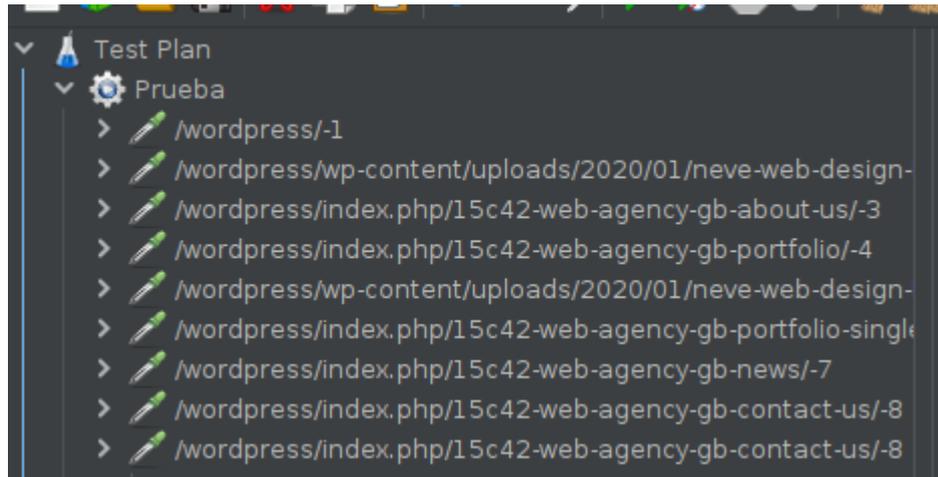
Anexo 25. Insertar un Test de Http

Se inicia la herramienta de testeo y manualmente se debe ingresar al sitio de WordPress y navegar por cada página para que Jmeter lo registre



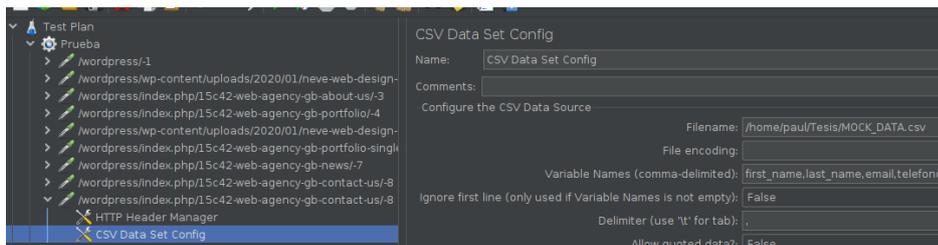
Anexo 26. Configuración del Test

Una vez se han navegado por todas las páginas se debe detener la prueba y verificar que las interacciones que se realizaron en la página web ahora se encuentran descritas en Jmeter como actividades

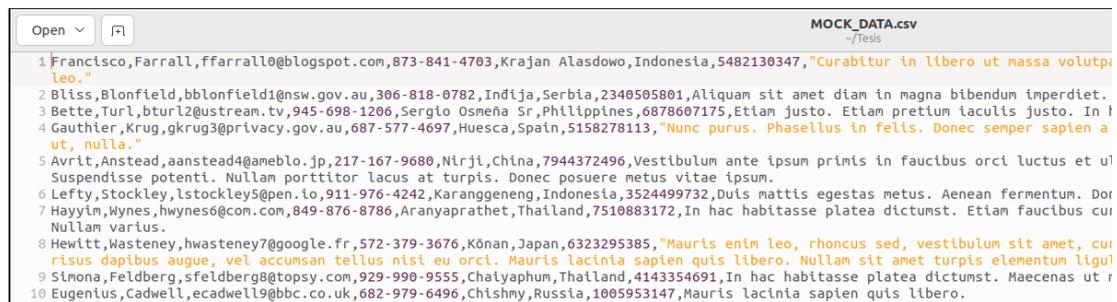


Anexo 27. Páginas recolectadas por el Test

En la última página, al ser de contacto, se ingresan datos para que entren a la base de datos. Estos datos que ingresan se pueden configurar por medio de un archivo csv.



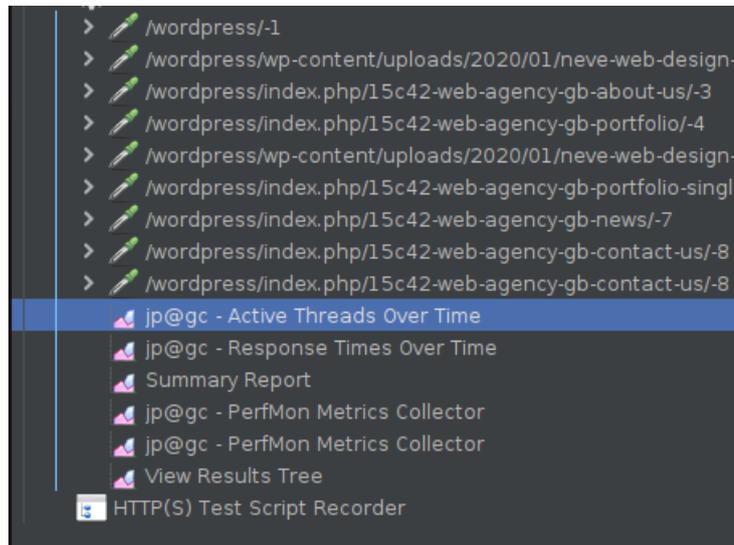
Anexo 28. Configuración de la prueba de ingreso de datos



Anexo 29. CSV con el formato para llenado de datos

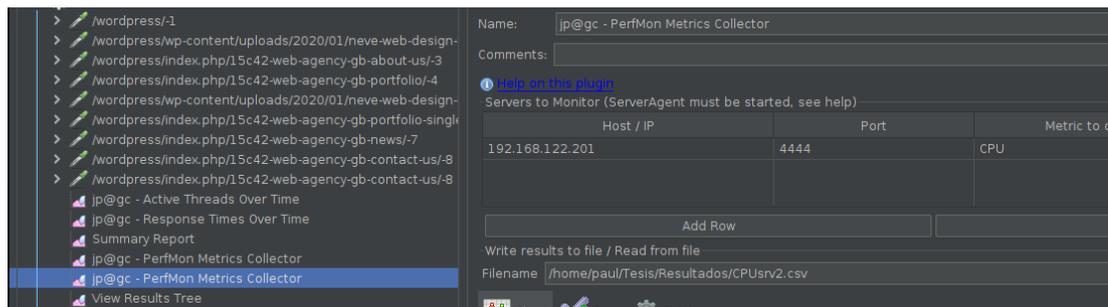
AÑADIR LAS PRUEBAS:

Las pruebas son fáciles de añadir solo escogiéndolas en el panel de pruebas y se listarán en el grupo de hilos automáticamente sin necesidad de configurar la mayoría.



Anexo 30. Pruebas añadidas en Jmeter

Para la recolección de los datos de consumo de CPU y Memoria se debe ingresar únicamente la IP de la máquina a la que se le realizará las pruebas.



Anexo 31. Configuración de la prueba en Jmeter

R STUDIO CODE:

La línea 1 sirve para limpiar las variables.

Las líneas de 2 - 6 sirven para importar los archivos de datos y asignarlos a variables.

Las líneas 11 – 13 sirve para obtener un anova de esa variable de datos. Obtener el resumen de ese anova y, por último, que imprima el anova en los resultados.

El resto de las porciones de código realizan la misma actividad, pero con diferentes métricas, CPU, Memoria en VM1 y VM2.

```
1 rm( list = ls() )
2 #setwd("/Users/victormorales/Downloads")
3 setwd("C:/Users/paul_/OneDrive - UNIVERSIDAD TÉCNICA DE AMBATO/Tesis/Procesar datos")
4
5 memvm <- rio::import("MEM_VM1_VM2.xlsx")
6 cpuvvm <- rio::import("CPU_VM1_VM2.xlsx")
7
8
9
10 # ***** RAM *****
11 m1_mem <- aov(Value~Algorithm,data =memvm,subset = memvm$Machine=="VM1")
12 summary(m1_mem)
13 TukeyHSD(m1_mem)# comparacion en pares, uno a uno
14
15
16
17 m2_mem <- aov(Value~Algorithm,data =memvm,subset = memvm$Machine=="VM2")
18 summary(m2_mem)
19 TukeyHSD(m2_mem)# comparacion en pares, uno a uno
20
21
22
23 # ***** CPU *****
24 m1_cpu <- aov(Value~Algorithm,data =cpuvvm,subset = cpuvvm$Machine=="VM1")
25 summary(m1_cpu)
26 TukeyHSD(m1_cpu)# comparacion en pares, uno a uno
27
28
29
30 m2_cpu <- aov(Value~Algorithm,data =cpuvvm,subset = cpuvvm$Machine=="VM2")
31 summary(m2_cpu)
32 TukeyHSD(m2_cpu)# comparacion en pares, uno a uno
33
34
```

Anexo 32. R Studio Code