

 UNIVERSIDAD
SEÑOR DE SIPÁN

**FACULTAD DE INGENIERÍA, ARQUITECTURA Y
URBANISMO**

**ESCUELA ACADÉMICO PROFESIONAL DE
INGENIERÍA DE SISTEMAS**

TESIS

**COMPARACIÓN DE ALGORITMOS DE
BALANCEADORES DE CARGA UTILIZANDO CLÚSTER
HOMOGÉNEO EN SERVIDORES WEB**

**PARA OPTAR EL TÍTULO PROFESIONAL DE
INGENIERO DE SISTEMAS**

AUTORES:

BACH. CHUNGA ZULOETA JORGE LUIS

BACH. CHUZÓN SANCHEZ WALTER

ASESOR:

MG. VILLEGAS CUBAS JUAN ELIAS

**LÍNEA DE INVESTIGACIÓN:
TECNOLOGÍAS DE LA INFORMACIÓN**

**PIMENTEL – PERÚ
2017**

COMPARACIÓN DE ALGORITMOS DE BALANCEADORES DE CARGA UTILIZANDO CLÚSTER HOMOGÉNEO EN SERVIDORES WEB

Aprobación de la Tesis

Mg. Villegas Cubas Juan Elias
Asesor de tesis

Mg. Bravo Ruiz Jaime Arturo
Presidente de Jurado de tesis

Ing. Mejia Cabrera Heber Ivan
Secretario del Jurado de tesis

Ing. Villegas Cubas Juan Elias
Vocal de Jurado de tesis



AGRADECIMIENTO

En primera instancia agradecemos a dios, por bendecirnos de salud, amor, bienestar y darnos fuerzas todos los días para seguir adelante.

A nuestros padres, por haberme proporcionado la mejor educación y lecciones de vida.

Al ingeniero Villegas Cubas Juan Elías por su asesoría profesional, orientación y consejos para esta investigación.

“Agradezco a toda mi familia por su apoyo a lo largo de la carrera en especial a mi madre Rosario Zuloeta Rufasto y a todas las personas que contribuyeron con su granito de arena para mi crecimiento personal y profesionalmente solo me queda decirles mil gracias.” - **CHUNGA JORGE**

“En especial a mi padre Walter Chuzon Castro, por haberme enseñado que con esfuerzo, trabajo y constancia todo se consigue, y que la vida es una lucha constante del día a día; a mi madre Idelsa Violeta Sanchez Medina, por cada día hacerme ver la vida de una forma diferente y darme confianza para tomar decisiones importantes; a mis hermanos, por estar siempre ahí en forma de apoyo y confianza. A mi Amigo de Tesis, por el esfuerzo y empeño que otorgo a este proyecto y agradezco por la amistad sincera hacia mi persona. A mi familia CHUZON - SANCHEZ, por haberme aportado enseñanzas, vivencias en cada instante de mi vida y a todos aquellos que siguen estando cerca de mí y que le regalan a mi vida algo de ellos.” - **CHUZON WALTER**



Índice

Resumen.....	10
Abstract.....	11
Introducción.....	12
CAPÍTULO I: PROBLEMA DE INVESTIGACIÓN.....	13
1.1. Situación Problemática.....	13
1.2. Formulación del Problema.....	14
1.3. Delimitación de la investigación	14
1.4. Justificación e importancia de la investigación.....	15
1.5. Limitaciones de la investigación.....	17
1.6. Objetivos de la investigación.....	17
1.6.1. Objetivo general.....	17
1.6.2. Objetivos específicos.....	17
CAPÍTULO II: MARCO TEÓRICO	18
2.1. Antecedentes de Estudios:.....	18
2.2. Estado del arte	23
2.3. Base teórica científicas	24
2.3.1. Balanceo de cargas	24
2.3.2. Métodos de Balanceo de cargas	26
2.3.2.1. Random.....	26
2.3.2.2. Round Robin.....	27
2.3.2.3. Weighted Round Robin.....	27
2.3.2.4. Dynamic Round Robin	27
2.3.2.5. Least connections.....	28
2.3.3. Servidores	28
2.3.3.1. Componentes de Hardware de un servidor.....	30
2.3.3.2. Tipos de servidores.....	31
2.3.3.2.1. Servicios web.....	31
2.3.3.2.2. Servidores de base de datos	31
2.3.3.2.3. Servidores de correo electrónico	32
2.3.3.2.4. Servidores de comunicaciones	32
2.3.3.2.5. Servidores de servidores virtuales:.....	32



2.3.4.	Que es la Virtualización	33
2.3.5.	Clúster	34
2.3.5.1.	Componentes de un Clúster	35
2.3.5.1.1.	Nodos	36
2.3.5.1.2.	Sistemas Operativos	37
2.3.5.1.3.	Conexiones de Red	37
2.3.5.1.4.	Middleware	38
2.3.5.1.5.	Ambiente de programación Paralela	38
2.3.5.2.	Tipos de clúster	38
2.3.5.2.1.	High Performance (Alta Rendimiento)	38
2.3.5.2.2.	High Availability (Alta Disponibilidad)	38
2.3.5.2.3.	High Raliability (Alta Confiabilidad)	39
2.3.5.3.	Sub-tipos de Clúster	39
2.3.5.3.1.	Homogéneos	39
2.3.5.3.2.	Heterogéneos	39
2.3.6.	Que es Linux	39
2.3.6.1.	Que son las distribuciones	40
2.3.6.1.1.	RedHat	40
2.3.6.1.2.	Debian	40
2.3.6.1.3.	SuSe	41
2.4.	Definición de la terminología	41
2.4.1.	Servidores	41
2.4.2.	Balanceo de cargas	42
2.4.3.	Clúster	42
2.4.4.	Software libre	43
2.4.5.	Nodos	43
2.4.6.	Algoritmos	43
2.4.7.	Red	43
2.4.8.	Cuellos de botella	44
2.4.9.	Virtualización	44
2.4.10.	Internet	44
CAPÍTULO III: MARCO METODOLOGICO		46
3.1.	Tipo y diseño de investigación	46
3.1.1.	Tipo de investigación	46



3.1.2.	Diseño de la investigación.....	46
3.2.	Población y muestra.....	47
3.2.1.	Población.....	47
3.2.2.	Muestra	47
3.3.	Hipótesis.....	49
3.4.	Variabes.....	49
3.5.	Operacionalización	49
3.6.	Abordaje metodológico, técnicas e instrumentos de recolección de datos.50	
3.6.1.	Abordaje metodológico	50
3.6.2.	Técnicas de recolección de datos	50
3.6.3.	Instrumentos de recolección de datos	51
3.7.	Procedimiento para la recolección de datos	51
3.8.	Análisis estadístico e interpretación de los datos	52
3.9.	Principios éticos.....	54
3.9.1.	Ética de la aplicación	54
3.9.2.	Ética de los valores.....	54
3.10.	Criterios de rigor científico.....	54
CAPÍTULO IV: ANALISIS E INTERPRETACION DE LOS RESULTADOS.....		56
4.1.	Resultados en tablas y gráficos.....	56
4.2.	Discusión de resultados.....	61
CAPÍTULO V: PROPUESTA DE INVESEGACION.....		65
5.1.	Seleccionar algoritmos para realizar balanceo de cargas en clúster de alto desempeño.	65
5.2.	Modelos de los algoritmos Seleccionados	69
5.3.	Implementar los algoritmos seleccionados de balanceo de cargas en los servidores web.....	71
CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES.....		82
6.1.	Conclusiones.....	82
6.2.	Recomendaciones.....	83



ÍNDICE DE IMÁGENES

Imagen N° 1 Compañía de software Dynatrace url www.dynatrace.com	14
Imagen N° 2 Instituto nacional de estadística e informática url: https://www.inei.gob.pe/estadisticas/indice-tematico/tecnologias-de-la-informacion-y-telecomunicaciones/	16
ΙμαγεΝ N° 3 (Bookman,2003) Βαλανχεαδορ δε χαργα	26
Imagen N° 4 (marchionni, 2003) Balanceador de carga	30
Imagen N° 5 (Gallardo F., 2011) Cluster de computadores	35
Imagen N° 6 (Gallardo, 2011) componentes de un cluster	36
Imagen N° 7 Resultado de tiempo promedio por petición de las 386 peticiones con 100 peticiones simultaneas.....	57
Imagen N° 8 Resultado de promedio de tiempo totales de 100 peticiones simultaneas.....	57
Imagen N° 9 Resultado de tiempos promedio por petición de las 386 peticiones con 193 peticiones simultaneas.....	58
Imagen N° 10 Resultado de promedio de tiempos totales de 193 peticiones simultaneas.	58
Imagen N° 11 Resultado de tiempos promedio por petición de las 386 peticiones con 378 peticiones simultaneas.....	59
Imagen N° 12 Resultado de promedio de tiempos totales de 378 peticiones simultaneas.	59
Imagen N° 13 Resultado del porcentaje de promedio del consumo de memoria de las 386 peticiones con 100 peticiones simultaneas.....	60
Imagen N° 14 Resultado del porcentaje de promedio del consumo de memoria de las 386 peticiones con 193 peticiones simultaneas.....	61
Imagen N° 15 Resultado del porcentaje de promedio del consumo de memoria de las 386 peticiones con 378 peticiones simultaneas.....	61
Imagen N° 16 Modelo algoritmo Round Robin (Elaboración propia).	69
Imagen N° 17 Modelo algoritmo Least Connection (Elaboración propia).	70
Imagen N° 18 Visión General de la investigación.	71
Imagen N° 19 Modelo de despliegue de la investigación.	72
Imagen N° 20 Máquina virtual Linux Server en VirtualBox.	72
Imagen N° 21 Comando de actualización de Linux virtual Server.	73
Imagen N° 22 Comando de instalación de Haproxy.	73
Imagen N° 23 Comando para habilitar el Haproxy.....	73
Imagen N° 24 Configuración de Haproxy.....	74
Imagen N° 25 Preparación del haproxy y especificación del algoritmo de balanceo, direccionamiento de servidores.....	74
Imagen N° 26 Configuración de IP de Haproxy.	75
Imagen N° 27 Comandos de reinicio de red y Haproxy.....	75
Imagen N° 28 Comando para la instalación del servidor web.	76
Imagen N° 29 Comando para la instalación del servidor web mediante comandos de tareas....	76
Imagen N° 30 Comando para la creación del archivo de configuración de Joomla.	76
Imagen N° 31 comando de activación de Joomla y reinicio de Apache2.....	77
Imagen N° 32 Comandos de la creación de la base de datos y usuario.....	77
Imagen N° 33 Comando para dar todos los permisos de la base de datos.	78
Imagen N° 34 comando de permiso de acceso.....	78
Imagen N° 35 Página de inicio para instalar Joomla.	79
Imagen N° 36 Configuración de los IPs en ambos servidores.	79



Imagen N° 37 Servidor Cliente.	80
Imagen N° 38 Comando de instalación de la herramienta ApacheBench.	80
Imagen N° 39 Comando para simular las peticiones hacia el servidor HAproxy.	81
Imagen N° 40 Consumo de memoria de la Tabla 10.	89
Imagen N° 41 Consumo de memoria de la Tabla 11.	90
Imagen N° 42 Consumo de memoria de la Tabla 12.	91
Imagen N° 43 Consumo de memoria de la Tabla 13.	91
Imagen N° 44 Pruebas de 386 peticiones con 100 peticiones concurrentes usando algoritmo Least connection.	92
Imagen N° 45 Pruebas de 386 peticiones con 193 peticiones concurrentes usando algoritmo Least connection.	93
Imagen N° 46 Pruebas de 386 peticiones con 100 peticiones concurrentes usando algoritmo Round robin.	93
Imagen N° 47 Pruebas de 386 peticiones con 193 peticiones concurrentes usando algoritmo Round Robin.	94
Imagen N° 48 Páginas Web consultadas hacia los servidores.	103



ÍNDICE DE TABLAS

Tabla 1 Sistemas Operativos	37
Tabla 2 Nivel de confianza.....	48
Tabla 3 Operacionalización.....	49
Tabla 4 Tabla de Promedios de tiempos de las 386 peticiones	56
Tabla 5 Tabla Porcentaje de promedio del consumo de memoria de las 386 peticiones	60
Tabla 6 Tabla Peticiones simultaneas realizadas para la investigación.....	62
Tabla 7 Tabla de discusión.....	64
Tabla 8 Algoritmos de balanceo de carga	66
Tabla 9 Herramienta para el desarrollo de un clúster.....	67
Tabla 10 Respuesta de 386 peticiones con 100 peticiones concurrentes usando el algoritmo Least Connection.....	89
Tabla 11 Respuesta de 386 peticiones con 193 peticiones concurrentes usando el algoritmo Least Connection.....	90
Tabla 12 Respuesta de 386 peticiones con 100 peticiones concurrentes usando el algoritmo Round Robin	90
Tabla 13 Respuesta de 386 peticiones con 193 peticiones concurrentes usando el algoritmo Round Robin	91
Tabla 14. Cuadro de 100 Pruebas de 386 peticiones con 193 peticiones simultaneas.	95



Resumen

En los últimos años han incrementado los usuarios que ingresan diariamente a sitios web esto genera un gran tráfico en la red ocasionando posibles caídas y demoras en los tiempo de respuesta a la solicitud realizadas por el usuario, hemos visto como paginas gubernamentales o hasta el mismo google deja de funcionar.

El usuario necesita que el sitio web siempre este operativo y pueda realizar sus tareas sin retraso, para esto existe una técnica en informática llamada balanceo de cargas que permite repartir la carga de trabajo entre los recursos de un servidor web.

Dentro del balanceo de cargas existen algoritmos los cuales serán evaluados y comparados entre si bajo dos indicadores tiempo de respuesta y consumo de memoria del servidor.

Nuestra población van hacer las solicitudes emitidas hacia un servidor. Como no se tiene un número específico de solicitudes se trabajara con una población infinita.

Esta investigación mostrará que algoritmo de balanceo de carga permite en la medida de lo posible evitar las caídas y dar un buen tiempo de respuesta realizadas por el usuario, con esto se lograra que las empresas públicas y privadas de un buen servicio a sus clientes por medio de las páginas web.

Palabras claves: Balanceo de carga, alto desempeño, servicios web, homogéneos, clúster, servidor web.



Abstract

In recent years have increased users who enter daily to websites that generates a large network traffic, causing power failures and delays in response time to the request made by the user, we have seen government pages or to the same Google stops working.

The user needs the website provided this operation and to perform its tasks without delay, for this there is a technique called load balancing computer that can distribute the workload among the resources of a web server.

Within the load balancing algorithms exist which are evaluated and compared to each other on two indicators response time and avoid server crash.

Our population will make requests issued to a server. As you do not have a specific number of applications you work with an infinite population.

This research will show that algorithm load balancing allows as far as possible avoid falls and give a good response time by the user, with this being achieved that public and private companies of good service to its customers through web page.

Keywords: load balancing, high performance, web services, homogeneous cluster web server.

Introducción

Los servicios que brindan las empresas utilizando sus sitios web buscan brindar un servicio eficiente y que tenga un desempeño constante lo cual implica que evite la caída del servicio y que la respuesta a los usuarios se de en el menor tiempo posible. En esta investigación que se encuentra en la línea de las tecnologías de información, se presenta las soluciones y herramientas que serán utilizadas para tales problemas.

En un centro de datos encontramos al servidor el cual tiene un rol muy importante dentro de la organización porque es el encargado de responder a todas las solicitudes enviadas por los usuarios que se encuentre adentro o afuera de la empresa esto lo transforma en un servicio clave.

Para esto existe una solución tecnológica implementando un sistema de clúster el cual permita mantener el servicio con un grado de desempeño alto, así como también tener un sistemas de balanceo de carga que permita administrar la carga entre los servidores esto se realiza bajo un algoritmo el cual evita los llamados cuellos de botella que se puede generar por la alta congestión de tráfico.

Los resultados de la investigación es mejorar el rendimiento de los servicios de un servidor web bajo dos aspectos menor consumo de memoria y tiempos de respuesta todo esto estará implementando en tecnología de software libre.

CAPÍTULO I: PROBLEMA DE INVESTIGACIÓN

1.1. Situación Problemática

M. Mercedes, T. Diaz & E. Ruiz (2012). La red de redes llamada comúnmente Internet es una de las tecnologías que se ha convertido en algo predominante para las organizaciones y usuarios, utilizándola principalmente para realizar consultas de información y envío de datos mediante los sitios web. El principal problema de estos sitios web es como administrar las solicitudes de un amplio número de usuarios que la actualidad puede acceder a los sitios web por diversos dispositivos tecnológicos (laptops, teléfonos, móviles, iPad, etc.) causando un incremento considerable que los servidores web sufran caídas o que la respuesta a las solicitudes dadas por el usuario se realicen de una manera más lenta o con errores (error 405, error host unknown, etc.) la caída se puede dar tres o cuatro veces por mes según el sitio web sea consultado.

Esto origina al usuario una incomodidad al ver que la información que él está consultando tarde demasiado o que no cargue, lo que les genera a las empresas que los usuarios se lleven una mala imagen de la empresa y que esta mala imagen sea difundida a los demás usuarios, generándoles pérdidas de clientes y dinero ya que este buscara el mismo servicio, pero en otra parte.

J: Branch, F. Ramos, A. Mesa & R. Jimenez (2009). Con el pasar de los años las organizaciones se han topado con diversos problemas que no han sido fáciles de resolver, estas requieren disminuir al máximo el tiempo de ejecución de las aplicaciones ejecutadas por el usuario. Por este motivo se ha visto en la necesidad de frecuentar herramientas para facilitar su solución.



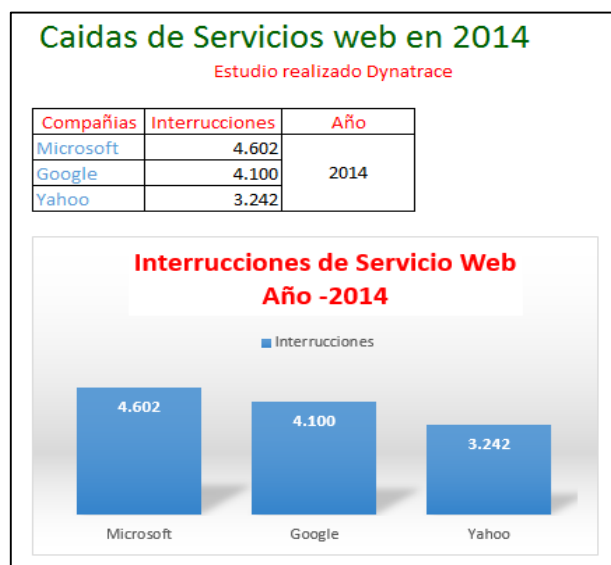


Imagen N° 1 Compañía de software Dynatrace url www.dynatrace.com

1.2. Formulación del Problema

¿Cómo identificar el algoritmo con mejor respuesta de tiempos y menor uso de recursos en los servidores web con balanceo de carga?

1.3. Delimitación de la investigación

El proyecto de investigación se realiza en los ambientes del laboratorio de investigación en sistemas inteligentes y seguridad informática perteneciente a la escuela profesional de ingeniería de sistemas, también se utilizó nuestros ambientes domiciliarios para esta investigación.

La investigación está dirigida al campo de balanceadores de cargas utilizados en servidores web; tuvo como colaboradores a nuestro Asesores.



El periodo de desarrollo del tema de investigación se lleva a cabo desde marzo hasta Diciembre del 2016.

1.4. Justificación e importancia de la investigación

En la actualidad el incremento del comercio y el flujo de datos de información en Internet, ha puesto en una situación en que las empresas no les de la importancia debida a las tecnologías de información para así sus servicios web puedan funcionar con un alto nivel de desempeño que optimice los tiempo de respuesta, de igual forma evitar la sobrecarga existente en los servidores web.

Debido al gran número de usuarios que realice solicitudes diarias para esto se realizara el proceso de balanceo de carga que permite repartir las solicitudes enviadas por el usuario y esta sean repartidas a distintos servidores, para que así las respuestas a las solicitudes sean más rápida y se evite la sobrecarga a los servidores.

El usuario será beneficiado al ver que su solicitud se realiza de una manera más rápida y sin retrasos también será beneficiados los trabajadores que este laborando ya que ellos pondrá agilizar los trámites y puedan cumplir con las fechas establecidas para cada tramite sin tener el temor a que haya caídas de sistema



Para el aporte científico relacionados a los sistemas de clúster y sistemas de balanceo de carga brindaremos un estudio confiable dentro del área de las TI esto servirá para futuras investigaciones que este ligados con estos temas.

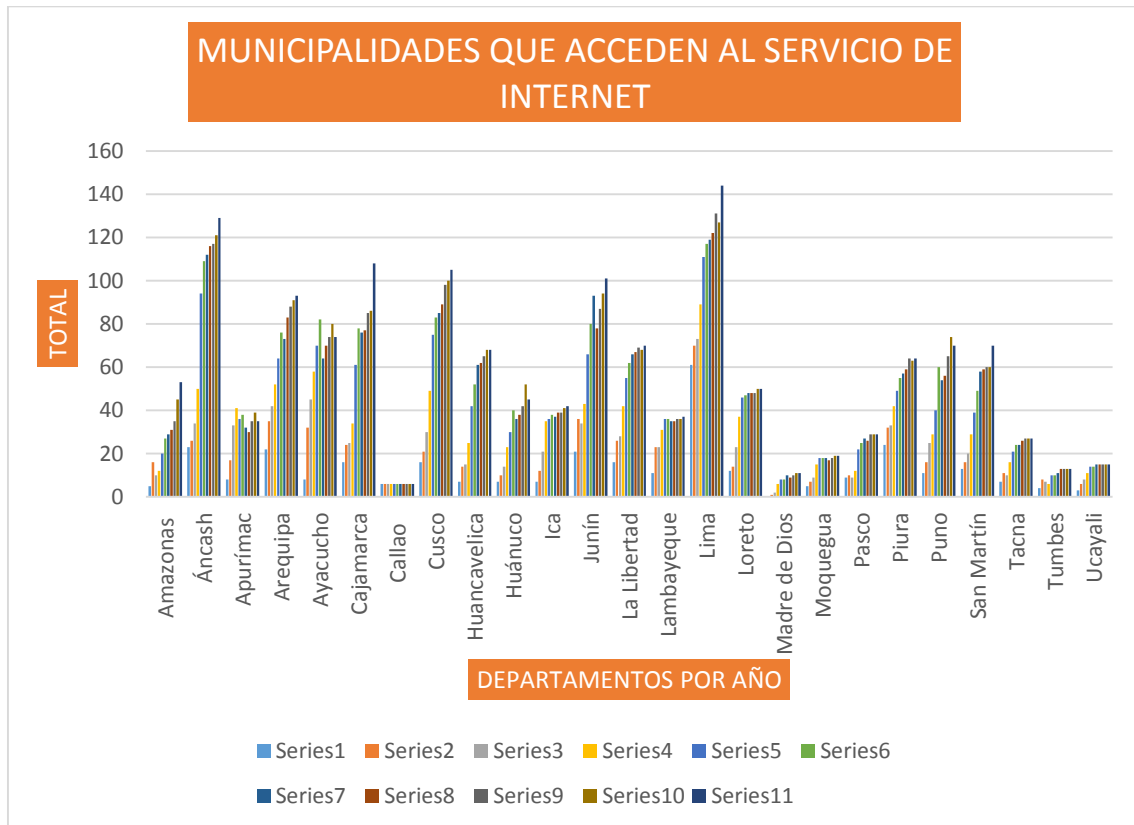


Imagen N° 2 Instituto nacional de estadística e informática
 url: <https://www.inei.gob.pe/estadisticas/indice-tematico/tecnologias-de-la-informacion-y-telecomunicaciones/>



1.5. Limitaciones de la investigación

La principal limitación que se nos presenta es contar con una entidad pública o privada que tenga más de un servidor web para realizar las pruebas respectivas.

1.6. Objetivos de la investigación

1.6.1. Objetivo general

Comparar algoritmos de balanceadores de carga en clúster homogéneo para el alto desempeño en los servidores web.

1.6.2. Objetivos específicos

- a) Seleccionar algoritmos para realizar balanceo de cargas en clúster de alto desempeño.
- b) Implementar los algoritmos seleccionados de balanceo de cargas en los servidores web.
- c) Evaluar los algoritmos preseleccionados, estos serán analizados de acuerdo al tiempo de respuestas y a evitar en la medida de lo posible la sobre carga a los servidores web.
- d) Realizar las pruebas de algoritmos de balanceo de cargas bajo los criterios de evaluación anteriormente mencionados



CAPÍTULO II: MARCO TEÓRICO

2.1. Antecedentes de Estudios:

A. Paulin & v. Shanthi (2014, India) en su investigación “*A load balancing model using firefly algorithm in cloud computing*”. Propusieron el algoritmo de planificación que se basaba en la optimización del programa para procesar solicitudes a la red de la nube. Su principal objetivo es maximizar la utilización de recursos y proporcionar un balanceo de carga entre los recursos en los servidores en la nube.

Detallan un modelo de carga de todos los recursos, se obtiene en cuenta a los factores como: tiempos de transformación y velocidad de acceso. Una vez iniciada la asignación de los recursos a los nodos apropiados es un problema de la distribución óptima, ya que existen bastantes algoritmos de optimización, así como los algoritmos genéticos y unos algoritmos genéticamente modificados usados para el balanceo de cargas.

Desarrollaron el algoritmo de luciérnaga para realizar el balanceo de carga con el cual obtuvieron un análisis del rendimiento producido donde les da unos resultados esperados de optimización de tiempos mediante el balanceo de carga.



J. Vega (2015 – Mexico), en su tesis “*Implementación de un balanceador de carga y alta disponibilidad para servicios web*”. El objetivo de este trabajo es la implementación de un Balanceo de carga y alta disponibilidad para servicios web, que minimizara el problema de la caída de los servicios web en la Facultad de Medicina Veterinaria y Zootecnia, ya que esta página permite a los alumnos y profesores a comunicarse entre ellos y así mejorar la experiencia de aprendizaje.

La herramienta que se utilizó para la implementación de balanceo de carga es Ha Proxy ya que permitió realizar dos funciones el balanceo de carga propiamente y la alta disponibilidad permitiendo el ahorro de procesamiento de los equipos.

La conclusión que se llegó al momento de la implementación del balanceo es que ayudo a disminuir los tiempos de una manera considerable pasando de segundos a milisegundos, estos resultados superaron las expectativas para que el proyecto fuera un éxito.

Se está presentando una implementación de un balanceo de cargas para el servidor que se encuentra en la facultad de veterinaria y medicina que se encuentra en la Universidad Autónoma de México lo cual su principal objetivo era reducir los tiempos de respuesta ante las solicitudes de los usuarios lo cual lo ha logrado.



C: Figuereo & H. Simbaña (2013 - Quito) en su tesis *“Análisis, comparación e implementación de una infraestructura virtual open source con alta disponibilidad basada en clusters, para servidores y escritorios dentro de las instalaciones de la empresa sinergyhard CIA. LTDA.”* Cuyo objetivo es desarrollar actividades de forma eficiente y eficaz, sin realizar gastos exagerados para su economía interna.

Haciendo uso de herramientas de virtualización como Red Hat enterprise virtualización la cual está diseñada para empresarios para asegurar la existencia de al menos una red San, hypre-v, citrix xenserver. Según los resultados respecto a otras herramientas esta es la mejor herramienta que se orienta a los requerimientos de alta disponibilidad de máquinas virtuales, servidores y entre dispositivos de almacenamiento de máquinas virtuales.

Haciendo uso de las distintas herramientas que se muestrean en la investigación se busca un modo eficiente y eficaz para la economía interna de una empresa, para ello proponen que la herramienta de Red Hat Enterprise virtualización es la que se adecua más a los requerimientos de la organización orientados a las máquinas virtuales como también a servidores.



J. Santana (2010, Mexico) en su investigación de “*Algoritmos de balance de carga con manejo de información parcial*”, su objetivo principal es diseñar un algoritmos de balance de carga con manejo de información parcial.

Para realizar la distribución de carga se hace la comparación de distintos algoritmos como toroide, seguido del árbol binario y el algoritmo global, se usó la aplicación de N-Reinas y uso de una plataforma de ejecución de clúster homogéneo pacifico, además se percibió que los cores que tiene mayor capacidad de procesamiento son los que han procesado más carga.

Mediante la utilización de los algoritmos utilizados entre los cuales están: el algoritmo toroide, el árbol binario y el algoritmo global; se puede realizar el balanceo de carga con manejo de información parcial.

M. Gallardo (2011, Ecuador). En su tesis "*Integración de aplicaciones de software libre aplicado a un clúster de alta disponibilidad y balanceo de carga de servidores proxy*". Cuyo objetivo Diseñar una solución de alta disponibilidad y balanceo de carga, para proveer de servicios que permiten dar prolongación al negocio en las operaciones realizadas por las empresas.



Se obtiene como prueba de utilizar configuraciones generales en los clúster que incluye los nodos, los algoritmos para balanceo de carga, instalaciones y configuraciones de servidores de páginas web Apache.

Mediante el clúster que se plantea en esta tesis se pretende evitar la caída del servicio a brindar mediante el uso de software libre e implementando clúster de alta disponibilidad aplicándolas en pequeñas empresas las cuales hace factible por su fácil instalación, mantenimiento de los sistemas y porque hace uso de software libre (open source).

A. Gutierrez (2012, España). En su tesis "*Clúster de alta disponibilidad y balanceo de carga sobre un servidor web*". Tiene como objetivo principal tener un clúster que se mantenga activo, estable el servicio web y disponer de un método el cual permita administrar la carga de peticiones de visitas a la web entre sus distintos servidores disponibles.

El desarrollo de este proyecto tiene como punto principal cumplir de forma precisa los objetivos expuestos en el estudio de viabilidad, los cuales se alcanzaron en su totalidad, por lo que el clúster logra ofrecer alta disponibilidad en el servicio ofrecido y un balanceo de carga de peticiones web entre los nodos existentes. Teniendo como resultado se dispone de un clúster de dos nodos operativos para su utilización en entornos de producción real en una organización.



Se aprecia como el investigador implementó un clúster de alta disponibilidad con el uso de algunas herramientas teniendo como resultado el desarrollo de este con el propósito de brindar una alta disponibilidad de los servicios y balancear la carga de trabajo en sus peticiones web.

2.2. Estado del arte

G. Gopinath & K. Shriram (2014) en su tesis "*An in-depth analysis and study of Load balancing techniques in the cloud computing environment*" realizaron un exhaustivo estudio de los algoritmos Max-Min y Min-Min guiándose de una población seleccionada del entorno de la nube dando algunos resultados como que el algoritmos Max-Min era mejor que el algoritmo Min-Min en términos de respuesta en corto tiempo.

Por otro lado el algoritmo min-min tiene también sus ventajas estos resultados depende no del algoritmo si no del entorno al que se enfoca a la nube.

M. Talavera & C. Albea (2014, España) en su investigación "*Control de balanceo de carga de un grupo de servidores de red*" cuyo objetivo es proponer un adecuado control distribuido de carga de N servidores. El análisis se dio mediante la ley de control que está basado en la teoría



consensus que reduce el tamaño de la cola de cada servidor con el propósito de realizar un balanceo de carga.

La carga computacional de sistema es menor en cuanto a los distintos métodos de balanceo de carga lo que hace que se redujeran las pérdidas de datos y así mismo hacen que el sistema sea robusto y escalable.

Se presentó un método para estimar la cuenca de atracción que tiene en cuenta las restricciones físicas del sistema que está basado en estimar una superficie de Lyapunov donde se cumplen las restricciones.

2.3. Base teórica científicas

2.3.1. Balanceo de cargas

Según (Bookman, 2003). El equilibrio de carga se refiere al método en el que los datos se distribuyen a través de uno o más servidores, casi todas las aplicaciones paralelas o distribuidas pueden beneficiarse de equilibrio de carga. Esto es manejado por un nodo maestro. Los datos son gestionados por el nodo maestro y se distribuye en dos o más máquinas dependiendo del tráfico.

Los datos no tienen que ser distribuido por igual, por supuesto. Si usted tiene un servidor en Gigabit Ethernet, ese servidor, obviamente, puede absorber más tráfico que un simple nodo fast ethernet. Una ventaja del balanceo de carga es que los servidores no tienen que ser local. Muy a



menudo, las peticiones web de una parte del país se encaminan a un lugar más conveniente en lugar de un repositorio centralizado.

Las solicitudes hechas de un usuario generalmente se encapsulan en una sesión de usuario, lo que significa que todos los datos serán redirigidos al mismo servidor y no se redirigen a otros dependiendo de la carga.

El equilibrio de carga también suelen manejar la conmutación por error mediante la redirección del tráfico desde el nodo caído y la difusión de los datos a través de los nodos restantes.

La principal desventaja de equilibrio de carga es que los datos tienen que permanecer consistente y disponible en todos los servidores, si se pudiera utilizar un método tal como rsync para mantener la integridad de los datos. Aunque el equilibrio de carga se realiza normalmente en el ISP de mayor tamaño por hardware.



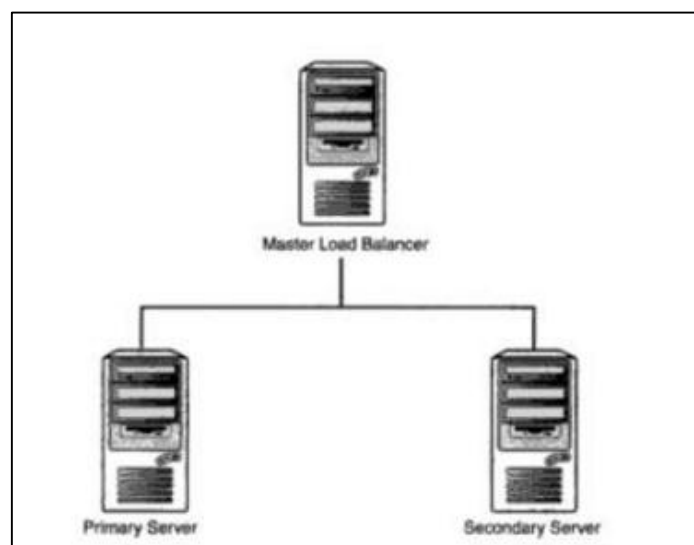


Imagen N° 3 (Βοοκμαν,2003) Βαλανχεαδορ δε χαργα

2.3.2. Métodos de Balanceo de cargas

2.3.2.1. Random

“Es un método que distribuye de forma alternada la carga entre los servidores disponibles, recolectando una a la vez de la generación de números aleatorios y él envió de la conexión actual a la misma” (Don MacVittie, 2010).

Formula:

$$x_{n+1} = (ax_{n+1} + c) * (mod m)$$

x_0 : Valor inicial

a : Contante multiplicativa

c : Constante aditiva

m : Numero cálculo de los restos

2.3.2.2. Round Robin

“Este método pasa cada una de las solicitudes de conexión al siguiente servidor en línea”(S. Mittal ,2013). Es decir distribuye las conexiones a cada uno de los servidores activos una por una, así el tiempo de ejecución de la distribución de conexión es forma homogénea.

Formula:

Tp: tiempo de proceso

Tpu: tiempo del ultimo proceso

Tpp: tiempo del próximo proceso

$$tp = tp + [(tProcesoUltimo - tProceso) - (tProcesoUltimo - tProcesoProximo)] + tp$$

2.3.2.3. Weighted Round Robin

“Con este método, el número de conexiones que cada nodo recibe en el tiempo es proporcional a un peso ponderado que se definida para cada máquina” (Don MacVittie, 2010).

Formula:

$$numero = normalizado * (peso/tamañoPaquete)$$

2.3.2.4. Dynamic Round Robin

“Este método es similar al anterior salgo que este las ponderaciones están basadas en el continuo monitoreo de los servidores ya que van cambiando constantemente” (Don MacVittie, 2010).



Este método distribuye las conexiones basado en algunos aspectos derivados del análisis de desempeño del servidor en tiempo real como es el número actual de conexiones o la velocidad en el tiempo de respuesta de un nodo.

TQ= Cantidad de tiempo.

Formula:

$$TQ = (Mediana * MayorTiempoRafaga) + (Mediana * MenorTiempoRafaga)$$

2.3.2.5. Least connections

“Aquí el método entrega una nueva conexión al servidor que tiene el menor número de conexiones. Least Connections es usado en ambientes donde los servidores y distintos equipos tienen capacidades similares” (Don MacVittie, 2010).

Formula:

$$(conexiones\ activas \leq conexiones\ totales) + conexiones\ inactivas$$

2.3.3. Servidores

Es un software que suministra a usuarios finales, que solicitan peticiones a distintos servicios como servidor de correo, servidor de fax, servidor web, etc. Según (Marchionni, 2011) “Los servidores son equipos informáticos que brindan un servicio en la red; dando información a otros

servidores y a los usuarios. Son equipos de mayores prestaciones y dimensiones que una PC de escritorio”. [p. 23]

(Marchionni, 2011) afirma que “Un servidor puede tener muchos procesadores con varios núcleos cada uno, que contienen grandes cantidades de memoria RAM, entre 16 GB a 1 TB a más; así el almacenamiento ya no se limitaría a un disco duro, ya que puede tener varios de ellos, con una capacidad del orden del TB. De acuerdo a sus capacidades, un servidor puede dar un solo servicio o más de uno”. [p. 23].

2.3.3.1. Componentes de Hardware de un servidor

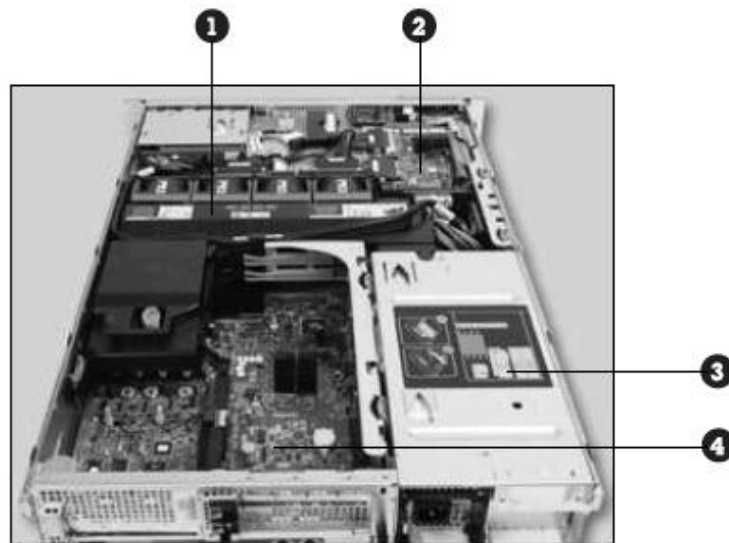


Imagen N° 4 (marchionni, 2003) Balanceador de carga

1. **Refrigeración:** Es un sistema de mayor importancia, porque si deja de funcionar, el servidor tendrá problemas como dejar de realizar su funcionamiento.
2. **HDs:** Es la parte reservada para los discos rígidos. según (Marchionni, 2011) “Éstos se pueden sacar y volver a colocar estando el servidor encendido, sin que haya pérdida de datos; se reconstruye el disco nuevo y sigue funcionando”. [p. 27] Esto se conoce como RAID esto se puede tener en cuenta por razones de seguridad. Evitando pérdidas de datos.
3. **Fuentes:** Ubicación de las fuentes de energía. según (Marchionni, 2011) afirma que “es posible apreciar el espacio para la ventilación con los ventiladores propios del servidor”. [p. 27].

4. Placas y micro: según (Marchionni, 2011) “Es la ubicación de la memoria RAM, la CPU y las placas de expansión, así como también de disipadores, cables y conectores” [p. 27].

2.3.3.2. Tipos de servidores

Hay unas diferentes tipos de servidores, estos pueden ser virtuales o físicos entre algunos de ellos tenemos:

2.3.3.2.1. Servicios web

Un servidor web ejecutado en un equipo se mantiene en la espera de peticiones por parte de un usuario (cliente) quien hace una petición mediante una página web hacia un servidor web. Este actúa cuando el servidor recibe una petición, este responde mediante una página web que se mostrara en la página web y si este no procesa la solicitud de la petición hecha este puede mostrar mensajes de errores correspondientes.

Según (E. Marchionni, 2011). Afirma que “un servidor web puede publicar cualquier aplicación web, brindarle la seguridad correspondiente y administrarla por completo” [p.25].

2.3.3.2.2. Servidores de base de datos

Según (E. Marchionni, 2011). “Este tipo de servidor tiene como principal característica de controlar grandes cantidades de datos y generar

información” [p. 25]. Para poder administrar los datos se conectan a un storage.

2.3.3.2.3. Servidores de correo electrónico

Según (E. Marchionni, 2011) afirma que “Estos servidores son capaces de controlar todos los correos de una organización en un solo lugar. También trabajan con un storage, esto se debe a la gran cantidad de datos que controlan, allí se almacenan los correos, donde se los redireccionan a clientes y servidores de seguridad, analizadores y replicadores” [p.26].

2.3.3.2.4. Servidores de comunicaciones

Según (E. Marchionni, 2011) “Este tipo de servidor brinda servicios de chat, telefonía IP, teleconferencia, video, etc. También son capaces de entregar servicios de pre-atendedor si se conecta a una consola telefónica” [p. 27].

2.3.3.2.5. Servidores de servidores virtuales:

Según (E. Marchionni, 2011) afirma que “un solo servidor físico puede contener varios servidores virtuales, pero el usuario final no distinguirá las diferencias. Sólo desde su administración podremos explotar todas sus características” [p.27]. Una de las desventajas de este tipo de servidor es que si se cae el servidor se caen todos sus servicios.



2.3.4. Que es la Virtualización

Durante el avance de los años, el aumento tecnológico del hardware avanza más que el del software. El hardware crece en capacidad y velocidad de procesamiento; y no existía un software que tomara ese crecimiento de hardware y lo aprovechaba al máximo. El autor (Marchionni, 2011) pensó si “Imaginemos un caso, del servidor de una aplicación que contenga tan sólo los datos de facturación. El uso que puede llegar a tener durante el mes es muy poco, sólo deberemos actualizar los datos con cierta frecuencia y, tal vez, efectuar alguna que otra consulta” [p.108].

El uso de la CPU, en promedio, en esos tiempos era del 10% del total. Por el contrario, a fin de mes donde se presentan todos los informes y se hace la facturación, el uso será casi del 100% o en ciertas ocasiones del 130%, es decir, el hardware se está desaprovechando por otra parte hasta el momento no hay forma disminuir en costos de los equipos. Según (Marchionni, 2011) era “Indefectiblemente, el hardware era necesario a fin de mes, aunque durante el resto de los días no se aprovechara su poder de cálculo” [p.108].

La virtualización de servidores ofrece una solución para este problema brindándonos ahorro, agilidad en la administración y aprovechamiento del hardware.



Según (Marchionni, 2011) afirma que “La virtualización nos permite usar toda la capacidad de nuestros servidores durante el mayor tiempo posible. Así podemos exprimir todos nuestros recursos de hardware sin gastar de más. También nos da la posibilidad de tener varios servidores en uno solo y de este modo compartir todos los recursos” [p.108].

La virtualización pone a disposición una capa o cubierta de tecnología entre el hardware y el sistema operativo esto hace tener la idea al sistema operativo que está instalado en un equipo físico cuando en verdad está virtualizado. El autor (Marchionni, 2011) afirma que “un servidor virtualizado se comportará de la misma forma o mejor que uno físico” [p.109].

2.3.5. Clúster

Según (Buyya, 1999) “Los clúster se forman de la unión de máquinas conectadas entre sí para realizar tareas complejas que ha sido un problema en el mundo de la informática. Los sistemas de información no podían realizar tareas complejas debidas a la ineficacia de los equipos, y se tenía que ser uso de las llamadas supercomputadoras pero las cuales era muy robustas en su arquitectura y no era accesibles para cualquier empresa ya que estas tenían un costo elevado” [p.10].



Los clúster se comportan como un gran recurso conformado por nodos; esta tecnología ha evolucionado para desarrollar, depurar aplicaciones que necesitan un buen número de procesadores de alto rendimiento, alta disponibilidad y alto desempeño estas son los tipos de los clúster para elegir uno de estos tipos va depender del flujo de datos que se maneja en la empresa.

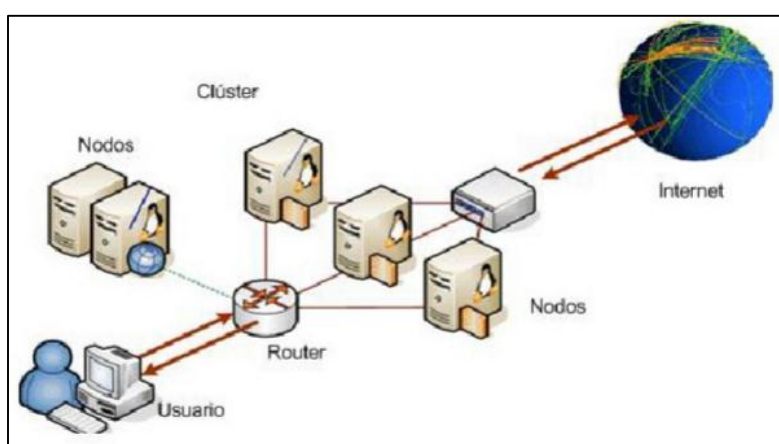


Imagen N° 5(Gallardo F., 2011) Cluster de computadores

2.3.5.1. Componentes de un Clúster

Los clúster están conformados por varios componentes los cuales pertenece a hardware y software. “Estos componentes son utilizados por el clúster para que pueda funcionar correctamente”. (Gallardo, 2011).



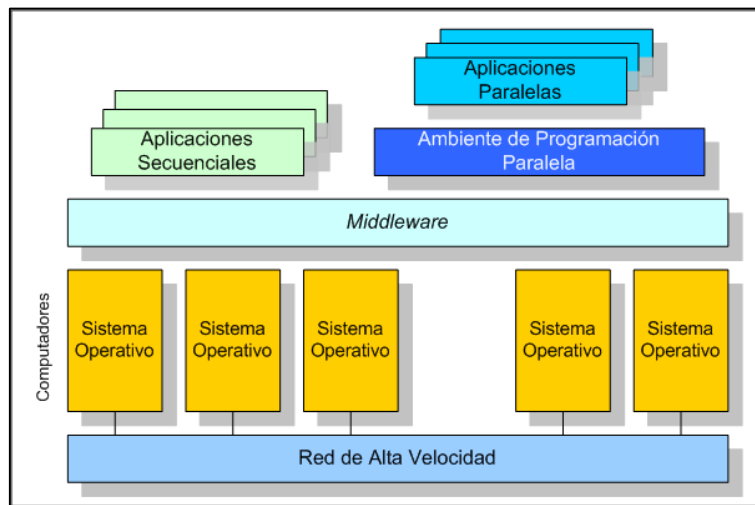


Imagen N° 6 (Gallardo, 2011) componentes de un cluster

2.3.5.1.1. Nodos

“Los nodos puede ser PCs, estaciones de trabajo o SMPS, los cuales pueden encontrarse encapsulados en un solo contenedor o estar físicamente diferenciados al momento de crear un clúster los nodos deben tener la misma semejanza en hardware para que así funcione correctamente el clúster sin sobrecargar a un nodo por su diferencia en hardware” (M. Castells, 2000).



2.3.5.1.2. Sistemas Operativos

El sistema operativo debe ser multiproceso y multiusuario los que van a gestionar los recursos de la computadora, también debe ser fácil en el uso y acceso para que así el usuario tenga facilidades en el manejo.

(C. Santiago, R.Garcia & J. Santos, 2007). Ejemplos:

Tabla 1 *Sistemas Operativos*

Sistemas Operativos	
GNU/Linux	ABC GNU/Linux1
OpenMosix	Rocks2
Sun Grid Engine	Unix
Solaris	Windows
Server 2008	Mac OS X
Xgrid	Solaris

2.3.5.1.3. Conexiones de Red

Las conexiones de red son utilizadas por los propios nodos los cuales puede conectarse a una red Ethernet o utilizar otras tecnologías de altas velocidades fast Ethernet, Gigabit Ethernet, etc. (Stallings & William ,2004).



2.3.5.1.4. Middleware

(Hailperin M, 2011-2016). “Es una interfaz que se encuentra entre los programas de aplicaciones y el sistema operativo. Middleware es un medio de interacción entre distintos programas de aplicaciones dado que proporciona mecanismos de apoyo en la interacción controlada a través de la coordinación, almacenamiento y comunicación” [p.26].

2.3.5.1.5. Ambiente de programación Paralela

Estos ambientes permiten implementar métodos o algoritmos los cuales hagan uso de recursos compartidos de una CPU.

2.3.5.2. Tipos de clúster

Dependiendo del flujo de datos que deseamos atender estos se dividen en:

2.3.5.2.1. High Performance (Alta Rendimiento)

Este tipo de clúster es para realizar un mejor tiempo de proceso al momento de recibir las solicitudes por el usuario final. (Oñate & ortega, 2010).

2.3.5.2.2. High Availability (Alta Disponibilidad)

Estos clúster son creados para mantener el servicio disponible los siete días de las semanas y las 24 horas del día para que estos servicios se encuentren activos el mayor tiempo posible. (Oñate & ortega, 2010).



2.3.5.2.3. High Raliability (Alta Confiabilidad)

Estos clúster son creados para aportar la confiabilidad al momento que se ejecute los procesos de una determinada empresa. Estos clústeres son los más difíciles de implementar ya que tiene como prioridad la seguridad de los procesos que estos se ejecute sin que la información sea corrompida. (Oñate & ortega, 2010).

2.3.5.3. Sub-tipos de Clúster

También cabe destacar que hay subtipos los cuales son:

2.3.5.3.1. Homogéneos

Lo clúster homogéneos se conforma con los nodos de las mismas características iguales en Hardware y Software.

2.3.5.3.2. Heterogéneos

Los clúster de este subtipo son conformados por nodos los cuales tienen características diferentes en cuanto a software y Hardware.

2.3.6. Que es Linux

(J. Garcia, I. Aguinaga & A.Mora ,2000) afirman que “Es un sistema operativo libre basado en Unix el cual otorga a Linux robustez y fiabilidad. En el que no hay que pagar licencia” [p.6]. Está distribuido bajo los términos de licencia GPL (Licencia Publica General), es decir cualquier persona puede usarlo.



2.3.6.1. Que son las distribuciones

(J. Garcia, I. Aguinaga & A.Mora ,2000) afirman que una “distribución es un conjunto de programas y ficheros (incluye la versión final estable del núcleo), organizados y listos para su instalación las cuales se obtienen a través del internet o comprando los CDs de las mismas” [p.7].

Una distribución es un argumento del núcleo del sistema operativo de Linux y otra sucesión de aplicaciones. Entre las distribuciones más conocidas tenemos RedHat, Debian, Slackware, SuSE, Ubuntu entre otros.

2.3.6.1.1. RedHat

(M. Santos M, 2004) “RedHat es una de las mas importantes distribuciones ya que personaliza Linux, liderando en los proyectos Open Source. Una de las características es el formato de paquetes de software llamado RPM para gestionar paquetes”. [p.9].

2.3.6.1.2. Debian

Esta distribución GNU/Linux es la única no comercial. Conformado por distintos colaboradores con la finalidad de desarrollar un sistema operativo basado en software libre. Según (R. Hertzog & R. Mas ,2015) afirman que es un “sistema operativo completo, incluyen el software y los sistemas para su instalación y gestión, basado en el núcleo de Linux y



software Libre” [p.34]. Esta distribución está formado por más de 3000 paquetes. Esta distribución gestiona su propio formato

2.3.6.1.3. SuSe

(M. Santos M, 2004) “Es una distribución alemana concentrada a los negocios ya que contiene una gran cantidad de paquetes y manuales” [p.]. Esta distribución al igual que RedHat tiene varias versiones una de las cuales está enfocada en usuarios normales y otra enfocada a las organizaciones para levantar servidores” [p.10].

2.4. Definición de la terminología

2.4.1. Servidores

Un servidor es un equipo que atiende las peticiones de un cliente y devuelve una respuesta en conjunto. Los servidores se pueden ejecutar en cualquier tipo de equipo, inclusive en computadoras dedicadas conocidas como servidor. “En la mayoría de los casos una misma computadora puede proveer múltiples servicios y tener varios servidores en funcionamiento” (Marchionni, 2011).

Una de las ventajas de montar un servidor en servidores o computadoras dedicadas es la seguridad. Por ello la mayor parte de los servidores son procesos diseñados de forma que puedan funcionar en computadoras con un fin específico.



2.4.2. Balanceo de cargas

El balance o balanceo de carga es un método usado para dividir el trabajo a realizar entre distintos procesos, ordenadores, etc. Está ligado a los sistemas de multiprocesamiento, que usan varias unidades de procesamiento para realizar distintos trabajos útiles.

El balance de carga se lleva gracias a un método o algoritmo que distribuye de la manera más equitativa el trabajo, para evitar el tráfico de datos, es decir evitar los cuellos de botella.

2.4.3. Clúster

Un clúster es un conjunto de computadoras que utilizan componentes comunes y actúan como si se tratase de un solo sistema u ordenador.

Se les llama clúster a los conglomerados informáticos que se agrupan a partir de partes de hardware compartidas que imparten el comportamiento de todas las unidades como una sola. Estos tipos de clúster se utilizan en áreas científicas, de ingeniería y de diverso tipo de investigación para operaciones de gran escala. Por otra parte, también se utilizan clúster en servidores y bases de datos de alto rendimiento, como se ve en empresas de gran tamaño.



2.4.4. Software libre

“La definición de software libre se establece para cumplir que un programa sea considerado libre. Cada un cierto tiempo se modifica definición para resolver problemas sobre cuestiones delicadas.” (Free Software Foundation, 2001).

2.4.5. Nodos

Un nodo es un punto o espacios reales o abstractos en el cual se convergen las conexiones de otros puntos, compartiendo sus características y siendo estos nodos.

Según (Castells M., 2000) “Un nodo es el punto en el que una curva se interseca consigo misma. Lo que un nodo es concretamente, depende del tipo de redes a que nos refiramos” [p.549].

2.4.6. Algoritmos

(J. Lopez, 2007). “Un algoritmo es una agrupación de normas o reglas bien definidas, ordenadas y finitas que ejecutan una tarea por medio de pasos para realizar dicha actividad” [p.21].

2.4.7. Red

“Una red se utiliza para definir a una estructura que cuenta con un patrón característico” (A. Philippe & J. Dordogne, 2006).



2.4.8. Cuellos de botella

En ingeniería, un cuello de botella es un problema donde la capacidad de un sistema completo se ve limitado por un único componente. El componente es conocido como punto del cuello de botella. “El término es una derivación metafórica que hace referencia al cuello de una botella, donde la velocidad del flujo de un líquido es limitado por este cuello angosto.” (Alegsa L, 2010).

2.4.9. Virtualización

(Marchionni, 2011). “La virtualización es una máquina virtual de un dispositivo, como un servidor, un dispositivo de almacenamiento, una red o incluso un sistema operativo, donde se distribuye los trabajos en uno o más entornos de ejecución”. [p.108].

2.4.10. Internet

Según (A. Portilla, J. Perez & D. Hoz, 2000). Internet es una red de redes que realiza las interconexiones descentralizadas de ordenadores a través de un grupo de protocolos TCP/IP. Tienes sus inicios desde 1969, cuando una agencia del Departamento de Defensa de los Estados Unidos realizo



búsquedas alternativas ante una guerra atómica que trae consigo el problema de las personas.

Luego de tres años se realizó la primera demostración pública del sistema ideado, esto debió gracias a que tres universidades de California y una de Utah realizaron una conexión exitosa conocida como ARPANET (Advanced Research Projects Agency Network).

CAPÍTULO III: MARCO METODOLOGICO

3.1. Tipo y diseño de investigación

3.1.1. Tipo de investigación

Para el estudio de esta Tesis se realizara una investigación descriptiva y analítica que parten de la investigación cuantitativa. Descriptiva por que se someterá a un análisis en el que se mide y evalúa diversos componentes como la distribución del número de atenciones y disminuir el tiempo de respuesta a las distintas solicitudes hechas a los servidores. Analítica ya que se establecerá la comparación de distintos algoritmos o métodos en las cuales se pueda reducir la saturación de las distintas solicitudes hacia los servidores y así obtener cual es un algoritmo más óptimo para realizar el balanceo de cargas.

Debido a que el objetivo de esta tesis es “Implementar y Comparar algoritmos de balanceadores de carga en Clúster homogéneo para el alto desempeño en los servicios web” y así obtener como resultado un balanceador de carga que como la distribución del número de atenciones y disminuir el tiempo de respuesta a las distintas solicitudes hechas a los servidores.

3.1.2. Diseño de la investigación

El enfoque de esta investigación es cuantitativa. Por esta razón es necesario realizar una investigación cuasi experimental, ya que esta tesis



tiene como objetivo realizar comparación de los distintos métodos o algoritmos para balanceo de carga utilizando clúster homogéneos en servidores web y así dependiendo los resultados de la estadísticas realizadas por cada algoritmo obtener un buen resultado donde como la distribución del número de atenciones y disminuir el tiempo de respuesta a las distintas solicitudes hechas a los servidores.

3.2. Población y muestra

3.2.1. Población

La población estará formada por las peticiones emitidas hacia un servidor web, estas son variables de acuerdo a los accesos que se puede dar a un determinado sitio web, para ellos Se usa la fórmula de muestra infinita ya que no se tiene un número exacto de peticiones hacia un servidor web, por esta razón podremos resaltar nuestra población con las 384 peticiones que nos da como resultado la formula.

3.2.2. Muestra

Para nuestra muestra se toma un porcentaje de las solicitudes que se envía al servidor web y está determinada en la siguiente formula.

Determinación de la muestra:

$$n = \frac{z_{\infty}^2 * p * q}{e^2}$$

En donde:

z_{∞} : Nivel de confianza.



p: Probabilidad de éxito o proporción esperada.

q: probabilidad de fracaso.

e: Precisión (Error máximo admisible en términos de proporción).

Descripción:

p: 0.5 (Probabilidad de éxito)

q: 0.5 (1-p)

e: 0.05 (Máximo de error permisible).

z_{∞} : 1.96 (Valor tabla al 95%).

Tabla 2 Nivel de confianza

z_{∞}	1.15	1.28	1.44	1.65	1.96	2	2.58
Nivel de confianza	75%	80%	85%	90%	95%	95.5%	99%

$$n = \frac{(1.96)^2 * 0.5 * 0.5}{(0.05)^2}$$

$$n = \frac{0.9604}{0.0025}$$

$$n = 384.16$$



De acuerdo a los resultado obtenidos de la formula se tiene que la muestra es 384 peticiones realizadas al servidor.

3.3. Hipótesis

Con la implementación de servidores web con balanceo de carga permitirá identificar el algoritmo con mejor respuesta de tiempo y menor uso de recursos.

3.4. Variables

3.4.1. Variables independiente

Comparación de algoritmos de balanceo de cargas.

3.4.2. Variable dependiente

Desempeño en los servidores web.

3.5. Operacionalización

Tabla 3 Operacionalización

Variable dependiente	Dimensiones	Indicadores	Formula
Desempeño de los servidores web.	Rendimiento del servidor	Recurso de memoria	$Memoria = Ma - Mporpeticion$
		Tiempo de respuesta	$SumatoriaT(i) = Peticionuna auna + PeticionConcurrente$



3.6. Abordaje metodológico, técnicas e instrumentos de recolección de datos.

3.6.1. Abordaje metodológico

De acuerdo a esta investigación responde a las preguntas porque es de tipo descriptivo y analítica. Ya que el tipo descriptivo explica y ordena los diferentes resultados que se pueden tener al seleccionar los diferentes algoritmos de balanceadores de carga desde distintos puntos como son sus características y procedimientos que pueden tener. Analítica porque que se realizara una selección de distintos algoritmos de balanceo de carga y así disminuir el tráfico de los datos dando un mejor desempeño al brindar los servicios.

3.6.2. Técnicas de recolección de datos

- a) **Observación:** Porque empleando esta técnica nos permite, describir, conocer datos de cómo se maneja los procesos en el campo.
- b) **Entrevista:** Por que usar esta técnica nos permite obtener la información mediante el dialogo.
- c) **Recolección de Información:** Porque empleando esta técnica se pudo obtener la información necesaria para realizar la investigación.
- d) **Revisión Bibliográfica:** Esta técnica nos permitirá la localización y recuperación de información sobre el tema a investigar.



3.6.3. Instrumentos de recolección de datos

- a) Ficha de entrevista
- b) Fichas de observación
- c) Ficha de verificación
- d) Diagrama de estado

3.7. Procedimiento para la recolección de datos

a) Observación

Mediante este procedimiento se podrá apreciar el proceso de solicitudes hacia un servidor y también su tiempo de respuesta en cuanto a las peticiones enviadas por el usuario.

b) Entrevista

La entrevista se va realizar a los encargados del área de tecnologías de la información de las municipalidades provinciales del departamento Lambayeque con el fin de analizar el desempeño que tiene los servidores al momento dar respuestas a las solicitudes enviadas por el usuario.

Pasos:

- a. Estará a cargo de los autores de esta investigación
- b. Se realizara a los encargados de TI de cada una de las municipalidades que serán parte del proyecto.
- c. Se efectuara en el mismo local de las municipalidades



d. La entrevista se realizara en el horario laboral tomando como prioridad el horario de la tarde.

c) Recolección de Datos

Para la recolección de datos del proyecto Comparación de algoritmos de balanceadores de carga utilizando clúster homogéneo en servidores web.

Se realizara la recolección de datos en el proceso de envió de solicitudes.

d) Revisión bibliográfica

Se leerá información de las investigaciones y trabajos que se han venido realizando desde el inicio de balanceadores de carga hasta la actualidad permitiéndonos conocer más a detalle el tema a investigar.

3.8. Análisis estadístico e interpretación de los datos

Los datos serán analizados con las siguientes formulas estadísticas.

a) Media aritmética

La media aritmética se obtiene al sumar todos los datos y dividir el resultado entre el número total de datos.

$$\bar{x} = \frac{x_1 + x_2 + x_3 + \dots + x_n}{N}$$

$$x = \frac{\sum_{i=1}^n X_1}{N}$$



b) Media aritmética ponderada \bar{x}

Se agrupa a los números X_1, X_2, \dots, X_n que se promediarán, ciertos factores o pesos W_1, W_2, \dots, W_n que dependerán de la importancia de cada uno de los números. Por ello se genera una media aritmética ponderada, que también se representa con equis testada.

$$\bar{x} = \frac{w_1x_1 + w_2x_2 + \dots + w_nx_n}{w_1 + w_2 + \dots + w_n} = \frac{\sum_{i=1}^n w_i x_i}{\sum_{i=1}^n w_i}$$

c) La moda

La moda es el dato que se repite frecuentemente en la sucesión. Ejemplo:

A continuación se muestra una sucesión demostrando la moda, se tiene:

3, 5, 6, 6, 7, 7, **8, 8, 8**, 9, 9, 10

d) Formula Porcentual

Sea P la cantidad cuyo porcentaje quiero calcular en relación a un total T, dicho porcentaje es:

$$p (\%) = (P / T) \times 100$$



3.9. Principios éticos

3.9.1. Ética de la aplicación

Se obtendrá beneficios a futuro para otros campos donde puede ser tomado en cuenta el balanceo de carga para así poder agilizar aún más los tiempos de respuesta de los servidores

3.9.2. Ética de los valores

Esta ética te permite o impide ciertas situaciones las cuales podemos abusar de una cierta manera pero empleando la responsabilidad debida. Al momento de la realización del proyecto emplearemos esta ética de valores cuando recopilemos información de otras tesis y cuando presentemos los resultados sin alteraciones.

3.10. Criterios de rigor científico

3.10.1. Replicabilidad

La probabilidad que se repita la investigación de este tipo orientada a otras instituciones o los servicios en la nube.

3.10.2. Credibilidad o valor de la verdad

La credibilidad implica la valoración de los argumentos como creíbles estos se verá reflejados en los resultados. En la presente investigación se apoyara en los siguientes aspectos.



- a) Valoración por juicios de expertos de los instrumentos de investigación.
- b) Estimación valorativa de los datos y/o información derivada de los instrumentos aplicados.
- c) Valoración de los resultados que se obtuvieron en el proceso de desarrollar el tema de balanceador de carga.

CAPÍTULO IV: ANALISIS E INTERPRETACION DE LOS RESULTADOS

4.1. Resultados en tablas y gráficos

En esta investigación se realizó la evaluación de los algoritmos de balanceo de carga seleccionados, basándonos en el tiempo de respuesta y en el consumo de memoria de los servidores web. En distintas pruebas realizadas con 386 peticiones y n peticiones simultáneas como resultado obtuvimos:

Tabla 4 *Tabla de Promedios de tiempos de las 386 peticiones*

Algoritmos	Peticiones Simultaneas	Promedio de Tiempo por petición	Promedio de Tiempo total de peticiones simultaneas
Round Robin	100	0,9555	95,5995
Least Connection		0,9856	98,5645
Round Robin	193	1,0127	194,5118
Least Connection		1,3073	250,4840
Round Robin	378	1,6669	629,4282
Least Connection		1,8939	708,9889



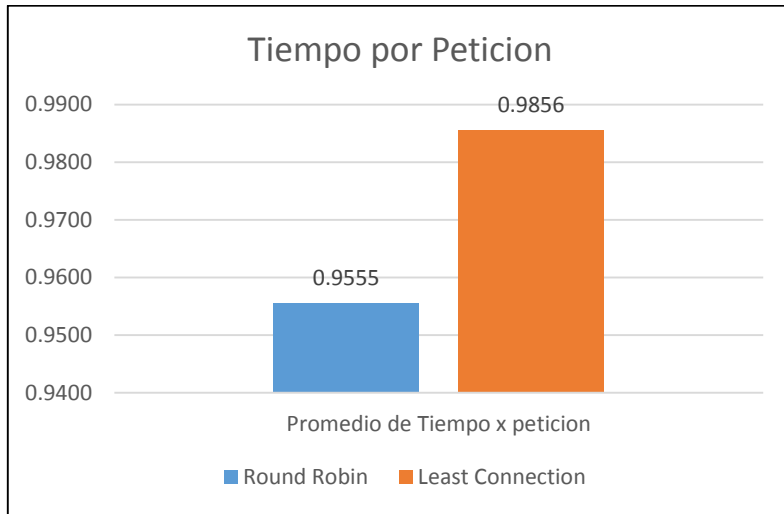


Imagen N° 7 Resultado de tiempo promedio por petición de las 386 peticiones con 100 peticiones simultaneas.

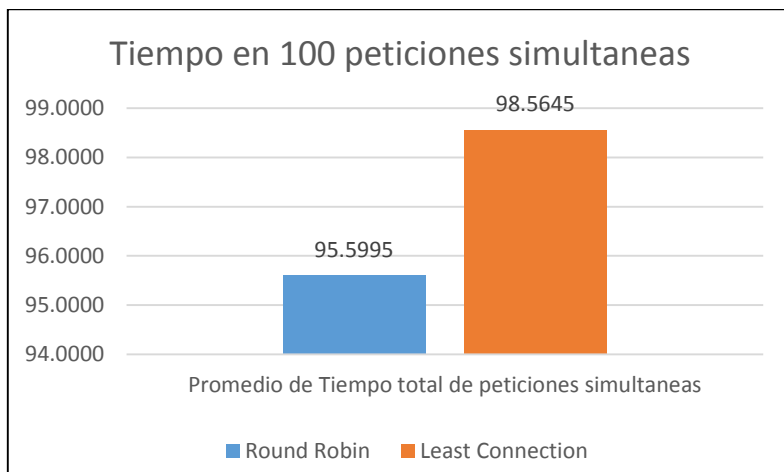


Imagen N° 8 Resultado de promedio de tiempo totales de 100 peticiones simultaneas.



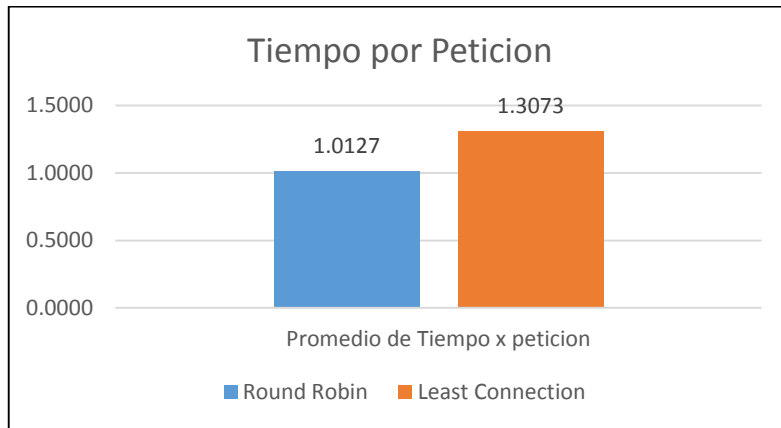


Imagen N° 9 Resultado de tiempos promedio po petición de las 386 peticiones con 193 peticiones simultaneas.

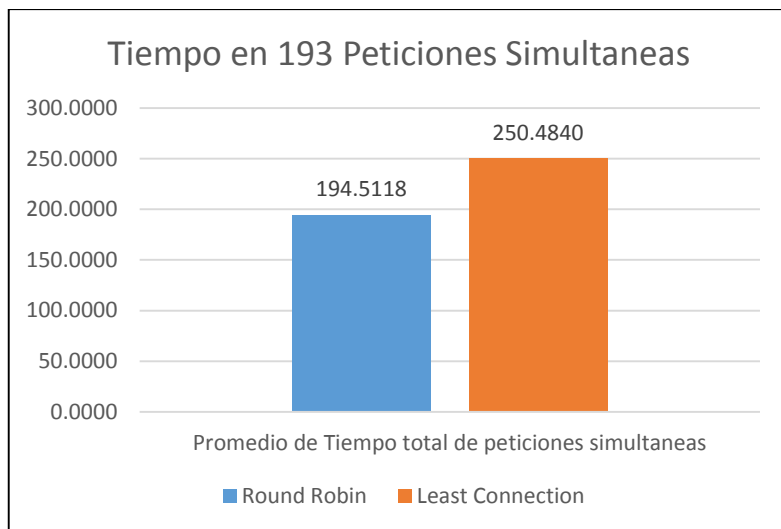


Imagen N° 10 Resultado de promedio de tiempos totales de 193 peticiones simultaneas.



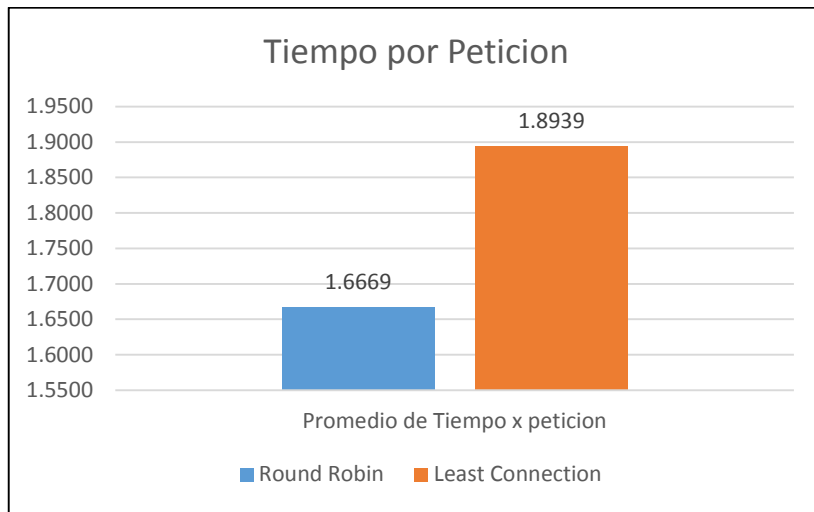


Imagen N° 11 Resultado de tiempos promedio por petición de las 386 peticiones con 378 peticiones simultaneas.

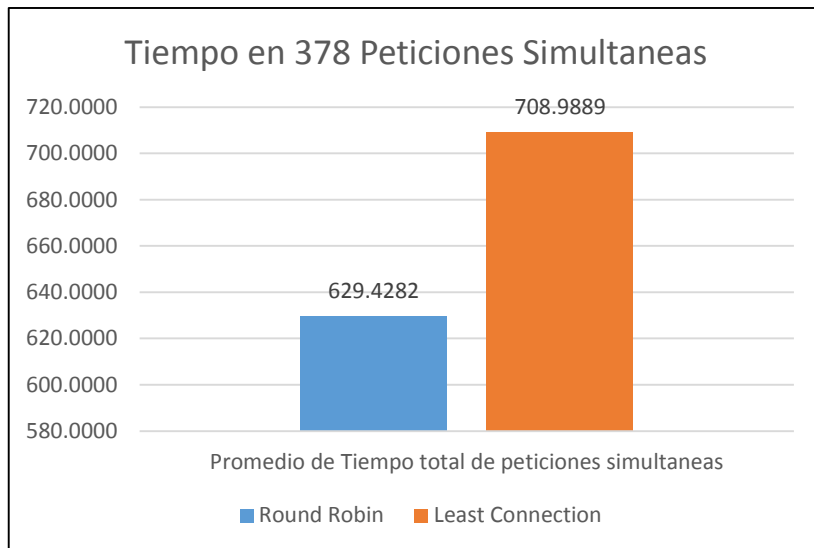


Imagen N° 12 Resultado de promedio de tiempos totales de 378 peticiones simultaneas.



Tabla 5 Tabla Porcentaje de promedio del consumo de memoria de las 386 peticiones

Algoritmo	Peticiones simultaneas	% de Memoria usado
Round Robin	100	9.3137%
Least Connection		9.3234%
Round Robin	193	9.4500%
Least Connection		9.4497%
Round Robin	378	9.7132%
Least Connection		9.8131%

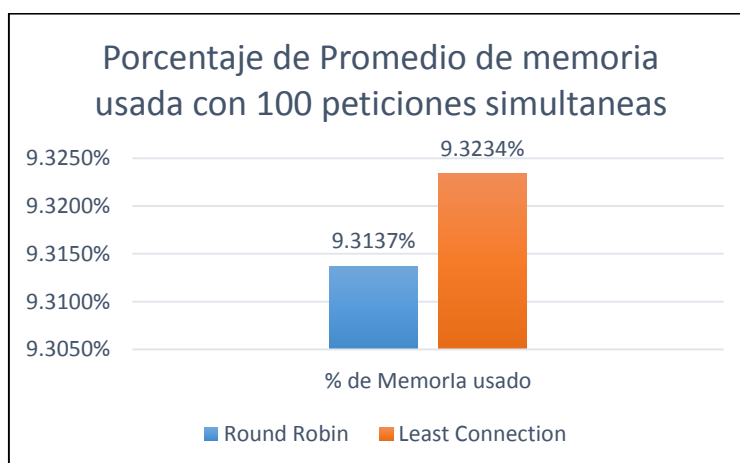


Imagen N° 13 Resultado del porcentaje de promedio del consumo de memoria de las 386 peticiones con 100 peticiones simultaneas



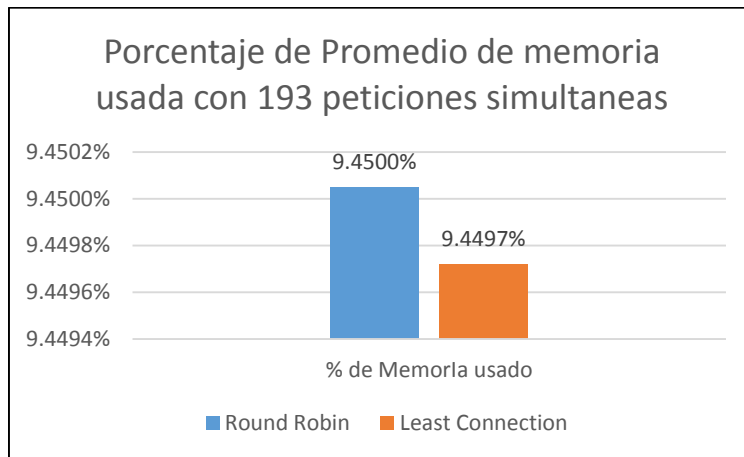


Imagen N° 14 Resultado del porcentaje de promedio del consumo de memoria de las 386 peticiones con 193 peticiones simultaneas.

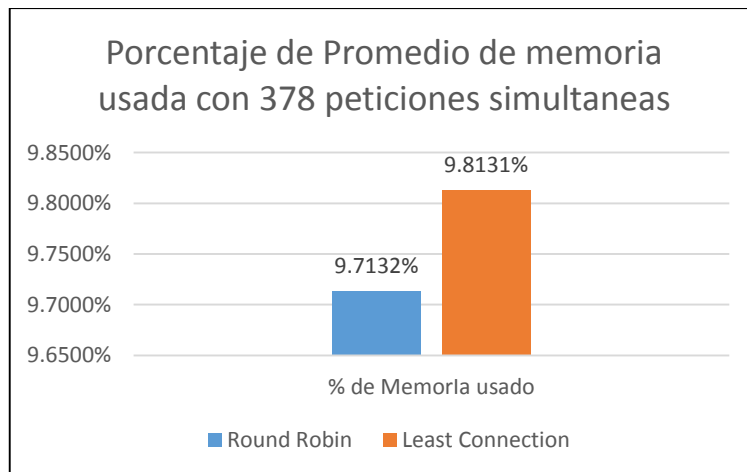


Imagen N° 15 Resultado del porcentaje de promedio del consumo de memoria de las 386 peticiones con 378 peticiones simultaneas.

4.2. Discusión de resultados

En esta investigación se trabaja con una población infinita de peticiones web de las cuales a juicio personal de los encargados a desarrollar. Este proyecto se decidió utilizar 100, 193 y 378 peticiones simultánea; estos números de peticiones se eligió con la finalidad de visualizar la diferencia



entre estos dos algoritmos para ver quién es el mejor en tiempo de respuesta y consumo de memoria.

Tabla 6 *Tabla Peticiones simultaneas realizadas para la investigación*

Peticiones simultaneas		
100	193	378

Con esta investigación se pretende evitar el llamado cuello de botella que hoy en día se da debido a las múltiples solicitudes o peticiones hechas a un servidor para ello se plantea la comparación de algoritmos de balanceo de cargas en clúster homogéneos con el propósito de proponer uno que sea adecuado para reducir el tiempo de respuestas hacia el servidor y que tenga menor consumo de memoria.

Para esto se usara tecnología de software libre como son Linux Virtual Server, Apache, HaProxy e algoritmos de balanceo de cargas como round robin donde las peticiones se van entregando de uno a uno a los servidores activos; y least connection con este algoritmo las peticiones se hacen dependiendo del número de conexiones que tenga cada servidor.



Como podemos observar en la tabla 4. Como actúan cada algoritmo como es el caso de Round Robin teniendo 386 peticiones con 100 peticiones concurrentes realizadas de 100 (verificar en la Tabla 4) pruebas para ello podemos observar que el tiempo de respuesta promedio de cada petición equivale a 0.9555 ms, Total de Tiempo peticiones concurrente es $0.9555 \times 100 = 95.5995$ ms. Verificar en la Imagen N° 7,8, 9, 10,11 y 12 dependiendo de las n peticiones simultaneas.

Así mismo también se puede calcular de la misma manera para el algoritmo least connection. Con cada prueba se puede visualizar que el algoritmo Round robin puede ser más útil y sencillo para reducir el tráfico de datos en estos dos aspectos evaluados.(Se puede verificar en el anexo N°2 en su Tabla 10 y 12) que describe una prueba con cada algoritmo.

Como podemos apreciar en el siguiente cuadro de las 600 pruebas realizadas en esta investigación se concluyó lo siguiente:



Tabla 7 Tabla de discusión

Algoritmos	Pruebas Peticiones		Resultado		Discusión
	Total de Peticiones	Peticiones Simultaneas	Promedio de Tiempo	Promedio Memoria	
Round Robin	386	100	0,9555 ms	9,3137%	Podemos observar que en la primera prueba el algoritmo Round Robin consume menos memoria y da un mejor tiempo de respuesta al usuario final
Least Connection			0,9856 ms	9,3234%	
Round Robin		193	1,0127 ms	9,4500%	En esta segunda prueba también podemos observar que el algoritmo Round Robin es mejor en estos dos aspectos en tiempo y memoria.
Least Connection			1.3073 ms	9,4497%	
Round Robin		378	1,6669 ms	9,7132%	La tercera prueba realiza también pasa lo mismo que el algoritmo Round Robin es mejor en tiempo de respuestas y en consumo de memoria.
Least Connection			1,8939 ms	9,8131%	

Concluimos que a través de la tabla 7 podemos ver que el algoritmo Round Robin es mejor en tiempo de respuesta y en consumo memoria, pero el algoritmo Least connection responde mejor cuando las conexiones que realice el usuario al servidor toman más tiempo como por ejemplo un examen en línea o una video llamada.



CAPÍTULO V: PROPUESTA DE INVESTIGACION

En esta investigación se propone un balanceo de carga para el alto desempeño y esto consiste en repartir la carga de datos entre los distintos nodos esclavos y maestro, donde se va acumulando la información de las tareas o peticiones que estarán llegando a medida que va transcurriendo el tiempo. El reparto de las tareas a los nodos dependerá de algunos aspectos como el tamaño de las tareas, tiempo de respuesta y capacidad de enlace por ellos se hace usos de distintos algoritmos de balanceo para distribuir las carga de datos.

5.1. Seleccionar algoritmos para realizar balanceo de cargas en clúster de alto desempeño.

En esta investigación se propone trabajar con algoritmos de balanceo de carga los cuales consisten en distribuir las cargas de datos a los servidores para así evitar el sobrecargo de información y evitar el congestionamiento en la red o los llamados cuellos de botella. Por esta razón se seleccionan un grupo de algoritmos para repartir la carga de datos. En el siguiente cuadro se aprecian las ventajas y el motivo de porque la implementación del algoritmo a utilizar.

Tabla 8 Algoritmos de balanceo de carga

Algoritmos	Ventajas	¿Por qué usaría este algoritmo?
Weighted Round Robin	<ul style="list-style-type: none"> ✓ Reparte la carga de datos en los servidores reales uno a uno dependiendo al peso de la solicitud. 	<ul style="list-style-type: none"> ✓ Fácil de implementar.
Round robin	<ul style="list-style-type: none"> ✓ Es equitativo. ✓ Las prioridades no cambian. ✓ se asigna un quantum (10 a 100ms) de tiempo de igual duración para todos los procesos. 	<ul style="list-style-type: none"> ✓ Fácil de implementar pero entre sus desventajas si el tiempo es corto disminuye el rendimiento del cpu y si es muy largo empobrece los tiempos de respuesta.
Least connections	<ul style="list-style-type: none"> ✓ Este algoritmo es recomendable para clúster homogéneos. ✓ Se reparte la petición entrantes al servidor con menos conexiones activas para que sean atendidas. 	<ul style="list-style-type: none"> ✓ Una de las desventajas de este algoritmo es que no tiene en cuenta la cantidad de conexiones que debe de mantener un servidor. Se tendría en cuenta las conexiones activas para así distribuir la carga en los servidores.
Random	<ul style="list-style-type: none"> ✓ Es un algoritmo más simple para balanceo de carga ya que asigna las tareas al azar en los servidores disponibles. ✓ Bajo costo alta escalabilidad. 	<ul style="list-style-type: none"> ✓ No se escogió este algoritmo ya que para que realice un balanceo de carga aceptable se debe tener una cantidad de tareas muy alta a la cantidad de servidores, ya que no se tienen en cuenta la comunicación entre tareas.

También se plantean alternativas y selección de herramienta clúster.

Existen un gran número de herramientas para el desarrollo de un clúster, la elección de uno de ellos dependerá del tipo de algoritmo, de la complejidad de la instalación, uso específico y de la eficacia de atención a las solicitudes enviadas por el usuario. Entre las opciones que existen son:



- ✓ Heartbeat
- ✓ Haproxy
- ✓ Piranha
- ✓ Ultramonkey
- ✓ Linux Virtual Server

Tabla 9 Herramienta para el desarrollo de un clúster

Herramienta	Características	Ventajas	Desventajas
Heartbeat	<ul style="list-style-type: none"> ✓ Envía las solicitudes para verificar si el servidor principal está activo o no. ✓ Monitorea el software que tiene bajo su disposición con el fin de detectar fallos. ✓ Intercambia la solicitud del usuario previa verificación del nodo. 	<ul style="list-style-type: none"> ✓ Identifica fallos de sus recursos. ✓ Trabaja con una IP virtual general. ✓ Proporciona una infraestructura de clúster con capacidades y mensajes a servidores de cliente. 	<ul style="list-style-type: none"> ✓ Es un componente crítico en servidores de alta disponibilidad.
Haproxy	<ul style="list-style-type: none"> ✓ Es un balanceo de carga de código abierto. ✓ Realiza el balanceo a través de HTTP ✓ Esta particularmente bien adaptado para los sitios web que contempla un alto trafico 	<ul style="list-style-type: none"> ✓ Proporciona un cuadro estadístico para monitoreo de nodos. ✓ Proporciona algoritmos de balanceo de cargas. ✓ Robusto para realizar balanceo de carga en servidores web 	<ul style="list-style-type: none"> ✓ Su conducta es impredecible para los sitios web con alta carga de proceso o que genera un gran trafico
Piranha	<ul style="list-style-type: none"> ✓ Es un paquete incluido en la familia red Hat. ✓ Permite administrar los servicios mediante una página web. ✓ Balanceo mediante direcciones IP. 	<ul style="list-style-type: none"> ✓ Contiene algoritmos de balanceo de cargas. ✓ Está compuesto por un servidor LVS y un gestor que permite ver los servicios por medio de una interfaz gráfica. ✓ Está elaborado para hacerse más grande sin perder calidad en los servicios ofrecidos. 	<ul style="list-style-type: none"> ✓ En la actualidad disponible solo para versiones piranha, las cuales tiene un pago de por medio para utilizar la herramienta.
Ultramonkey	<ul style="list-style-type: none"> ✓ Crea servicios de red con balanceo de carga 	<ul style="list-style-type: none"> ✓ Proporciona una solución flexible que se puede 	<ul style="list-style-type: none"> ✓ Su configuración es compleja y



	<ul style="list-style-type: none"> ✓ Realiza balanceo de alta disponibilidad ✓ Monitorización a nivel de servicio con Idirectord. ✓ Código Open Source. 	<p>adaptar a una amplia gama de necesidades.</p> <ul style="list-style-type: none"> ✓ Se apoya en el proyecto LVS para algunos componentes. 	<p>existe poca documentación.</p>
<p>Linux Virtual Server</p>	<ul style="list-style-type: none"> ✓ Realiza balanceo de alta disponibilidad. ✓ Ofrece tres formas de balanceo mediante Nat, Ip Tunneling y direct Routing. ✓ Reúne varias herramientas para realizar el balanceo de cargas. 	<ul style="list-style-type: none"> ✓ Emplea distintos protocolos para realizar el Balanceo de carga. ✓ Los servidores puede estar en una misma red o en redes diferentes. 	<ul style="list-style-type: none"> ✓ Los nodos de dirección ejecuta el sistema operativo GNU/LINUX. ✓ Genera mucho tráfico cuando el número de servidores reales es elevado.



5.2. Modelos de los algoritmos Seleccionados

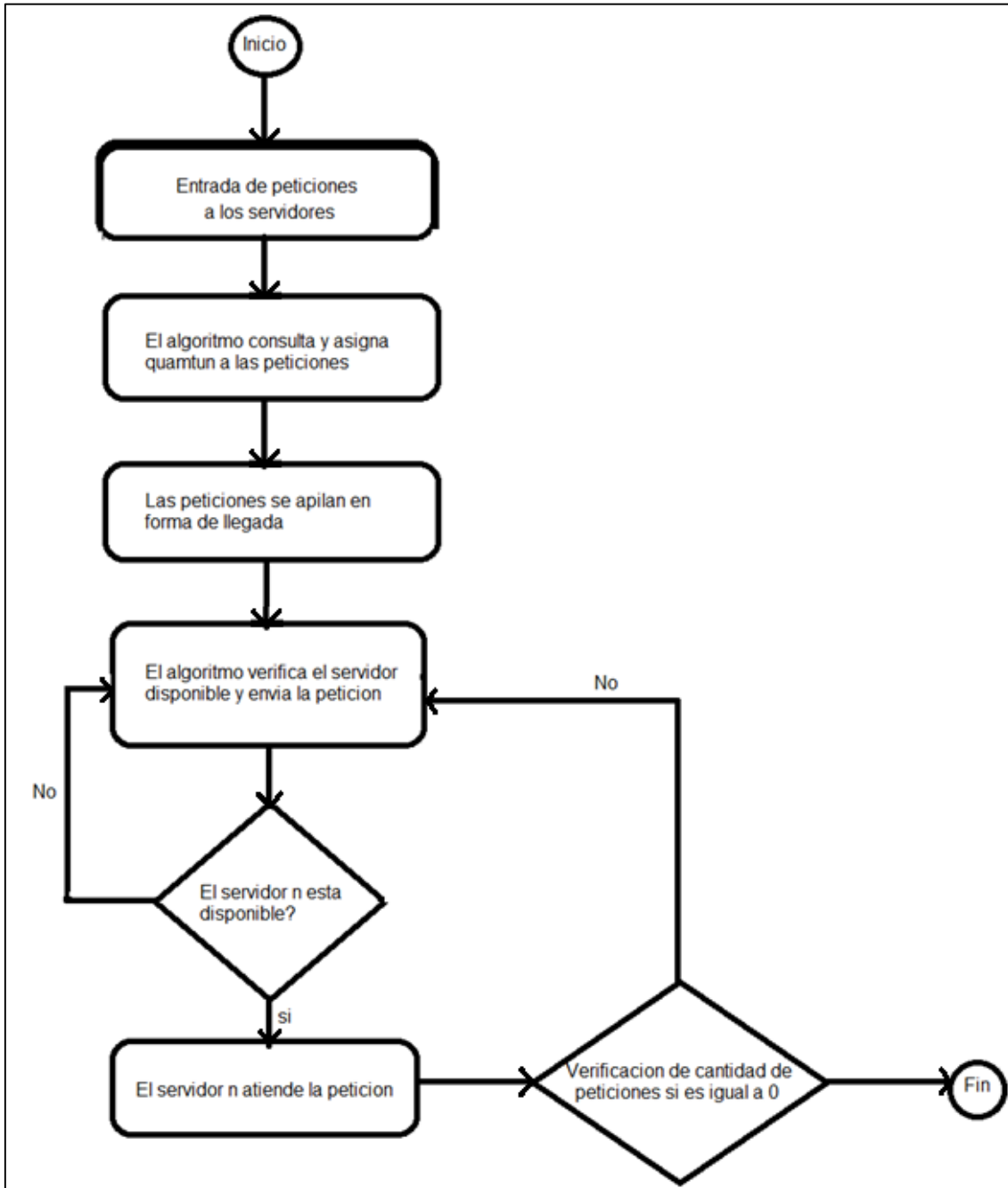


Imagen N° 16 Modelo algoritmo Raund Robín (Elaboración propia).

$$tp = tp + [(tProcesoUltimo - tProceso) - (tProcesoUltimo - tProcesoProximo)] + tp$$



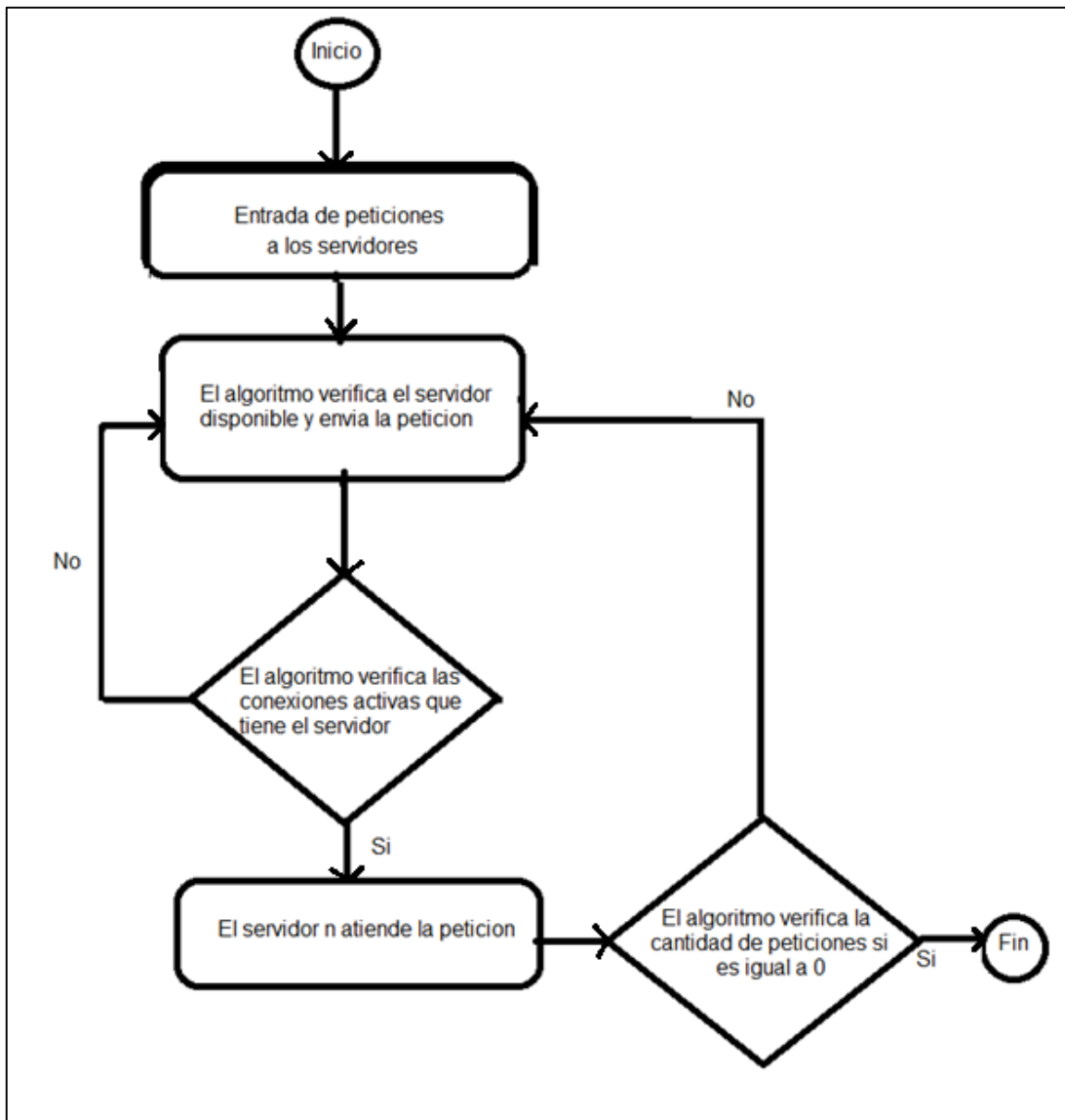


Imagen N° 17 Modelo algoritmo Least Connection (Elaboracion propia).
 (conexiones activas ≤ conexiones totales) + conexiones inactivas



5.3. Implementar los algoritmos seleccionados de balanceo de cargas en los servidores web.

Para esta investigación se realizó la comparación de dos algoritmos round robin y least connetion con ayuda de distintas herramientas y sistemas operativos como los son: Ubuntu server, apache, haproxy, virtualbox, joomla etc. para así tratar de reducir el tiempo de respuesta de las peticiones y la sobrecarga de los servidores web. A continuación, se muestra una visión general de la investigación.

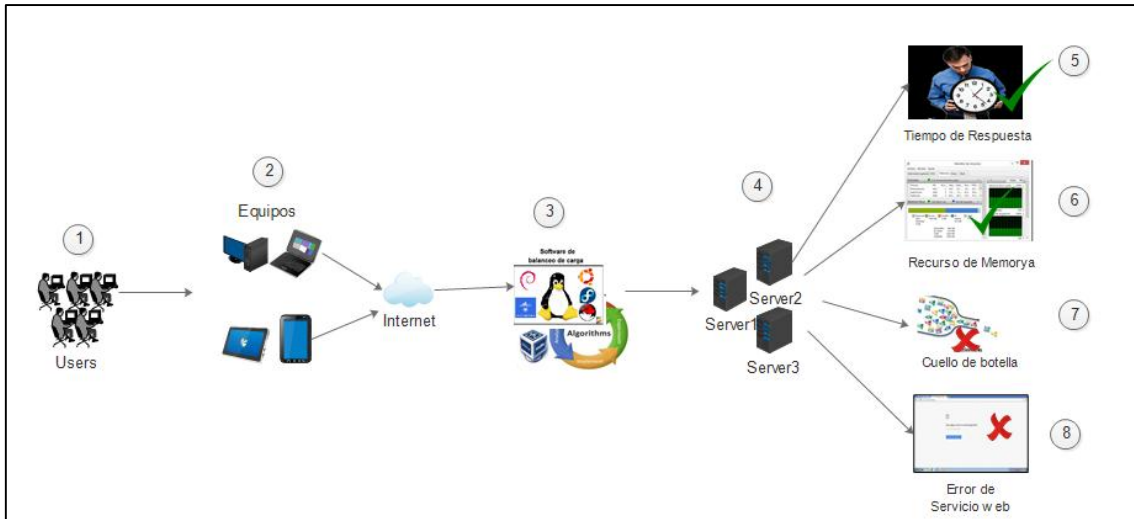


Imagen N° 18 Visión General de la investigación.

En la siguiente imagen se puede observar el modelo de despliegue de esta investigación:



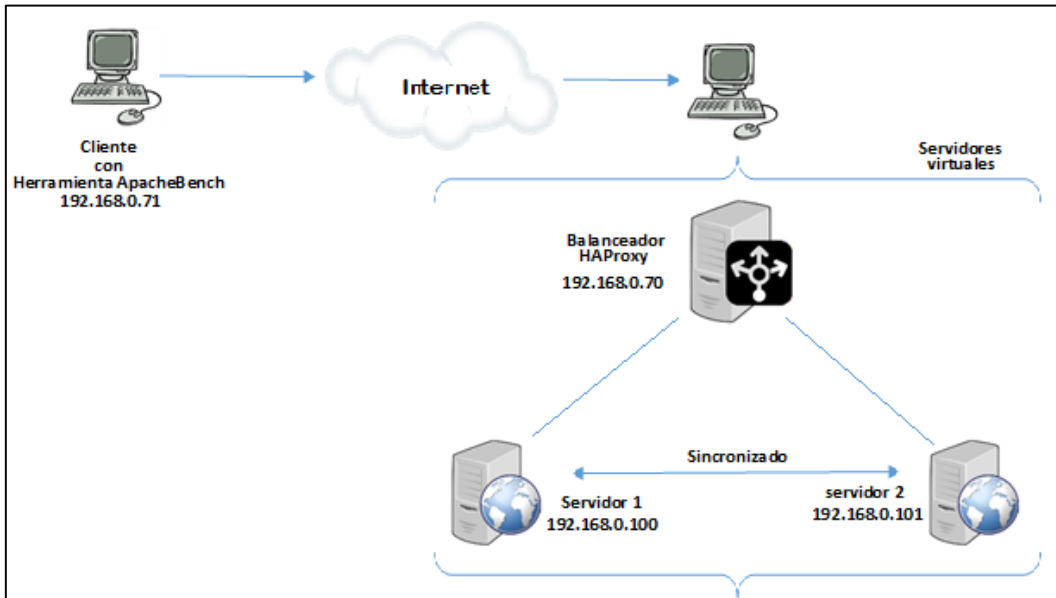


Imagen N° 19 Modelo de despliegue de la investigación.

En primera instancia se instala virtualBox donde se instalaran las máquinas virtuales, donde estarán nuestros servidores para ello utilizaremos Linux Virtual Server.

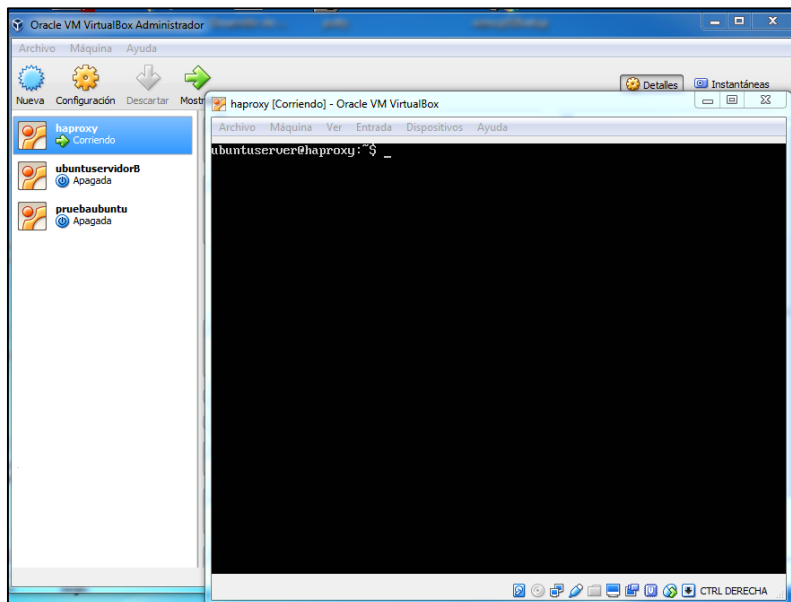


Imagen N° 20 Máquina virtual Linux Server en VirtualBox.



Se prepara las máquinas virtuales para ello se actualiza el sistema y se procede con la instalación de las herramientas Haproxy que es un software de código abierto usado para mejorar el rendimiento y la fiabilidad de un entorno del servidor mediante la distribución de carga de trabajo a través de los servidores web; Apache que es un servidor web y Joomla que es un gestor de contenidos que permite hacer sitios web dinámicos.

```
Archivo Máquina Ver Entrada Dispositivos Ayuda
ubuntu@prueba:~$ sudo apt-get update
```

Imagen N° 21 Comando de actualización de Linux virtual Server.

```
Archivo Máquina Ver Entrada Dispositivos Ayuda
root@haproxy:~# sudo apt-get install haproxy_
```

Imagen N° 22 Comando de instalación de Haproxy.

Se habilita el haproxy en el archivo ubicado en **/etc/default/haproxy** luego se comienza con la configuración del fichero **/etc/haproxy/haproxy.cfg**.

```
haproxy [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
root@haproxy:~# cd /etc/default/
root@haproxy:~# nano haproxy_
debian_version deluser.conf
GNU nano 2.2.6 Archivo: haproxy
# Set ENABLED to 1 if you want the init script to start haproxy.
#se habilita en haproxy
ENABLED=1
# Add extra flags here.
#EXTRA_OPTS="-de -m 16"
```

Imagen N° 23 Comando para habilitar el Haproxy.



```
root@haproxy:/etc/default# cd
root@haproxy:~# cd haproxy
bash: cd: haproxy: No existe el archivo o el directorio
root@haproxy:~# cd /etc/haproxy/
root@haproxy:/etc/haproxy# nano haproxy.cfg_
```

Imagen N° 24 Configuración de Haproxy.

Se

```
GNU nano 2.2.6 Archivo: haproxy.cfg
#esta listen de aqui lo agrego para el puerto 80
listen servidoresweb 0.0.0.0:80
    mode http
    stats enable
    stats uri /haproxy?stats
    #balance leastconn
    balance roundrobin
    option httpclose
    option forwardfor
    server servidorB 192.168.0.101:80 check
    server prueba 192.168.0.100:80 check
```

Imagen N° 25 Preparación del haproxy y especificación del algoritmo de balanceo, direccionamiento de servidores.

realiza la configuración que tendrá nuestro haproxy para esta investigación se le asignara el Ip 172.168.0.70 en la **ruta /etc/network/interfaces**; y los IPs que tendrán nuestros servidores en este caso 192.168.0.101 y 192.168.0.100 (Se puede observar en la figura numero 19).



```

haproxy [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@haproxy:/etc# nano network/interfaces_
GNU nano 2.2.6      Archivo: network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
#iface eth0 inet dhcp
iface eth0 inet static
address 192.168.0.70
netmask 255.255.255.0
gateway 192.168.0.1
    
```

Imagen N° 26 Configuración de IP de Haproxy.

```

haproxy [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@haproxy:/# cd etc/init.d/
root@haproxy:/etc/init.d# ./haproxy restart
* Restarting haproxy haproxy                                [ OK ]
root@haproxy:/etc/init.d# ./networking restart
* Running ./networking restart is deprecated because it may not enable again so
me interfaces
* Reconfiguring network interfaces...                        [ OK ]
root@haproxy:/etc/init.d# _
    
```

Imagen N° 27 Comandos de reinicio de red y Haproxy.

En los servidores instalaremos el servidor web con el siguiente comando **sudo apt-get install lamp-server^** o se puede instalar por medios de tareas de comandos las cuales primero se debe de instalar con el comando **sudo apt-get install tasksel** ; ejecutar **sudo tasksel** y seleccionar el servidor lamp.



```

ServidorA [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@servidorA:/# sudo apt-get install lamp-server^
    
```

Imagen N° 28 Comando para la instalación del servidor web.

```

ServidorA [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@servidorA:/# sudo apt-get install tasksel
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
tasksel ya está en su versión más reciente.
0 actualizados, 0 se instalarán, 0 para eliminar y 191 no actualizados.
root@servidorA:/# sudo tasksel_
    
```

Imagen N° 29 Comando para la instalación del servidor web mediante comandos de tareas..

Creamos la apache conf para joomla en el directorio apache webserver con los siguientes comandos. **Sudo cp /etc/apache2/sites-available/default /etc/apache2/sites-available/joomla.**

```

ServidorA [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@servidorA:/# sudo cp //etc/apache2/sites-available/default /etc/apache2/sites-available/joomla
    
```

Imagen N° 30 Comando para la creación del archivo de configuración de joomla.

Una vez esto activamos joomla y reiniciamos el servidor de apache de la siguiente manera.




```

ServidorA [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
root@servidorA:~# sudo cp //etc/apache2/sites-available/default /etc/apache2/sites-available/joomla
root@servidorA:~# sudo a2ensite joomla
Enabling site joomla.
To activate the new configuration, you need to run:
  service apache2 reload
root@servidorA:~# sudo service apache2 restart
 * Restarting web server apache2
apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1 for ServerName
... waiting apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1 for ServerName
[ OK ]
    
```

Imagen N° 31 comando de activación de joomla y reinicio de apache2.

Iniciamos sesión en el servidor mysql como usuario root con el siguiente comando **mysql -u root -p** ingresamos la contraseña y comenzamos a crear la base de datos, usuario y contraseña; le concedemos todos los permisos en la base de datos.

```

ServidorA [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
mysql> CREATE DATABASE dbjoomla
-> ;
Query OK, 1 row affected (0.00 sec)

mysql> CREATE USER joomlauser;
Query OK, 0 rows affected (0.00 sec)

mysql> SET PASSWORD FOR joomlauser = PASSWORD("joomla");
Query OK, 0 rows affected (0.00 sec)

mysql>
    
```

Imagen N° 32 Comandos de la creación de la base de datos y usuario.



```

mysql> GRANT ALL PRIVILEGES ON dbjoomla.* TO joomlauser@localhost IDENTIFIED BY
'joomla';
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)

mysql> EXIT
Bye
root@servidorA:~#
    
```

Imagen N° 33 Comando para dar todos los permisos de la base de datos.

Descargamos **joomla** y creamos la carpeta **joomla** en la dirección **/var/www/** con el siguiente comando **mkdir /var/www/joolma** luego cambiamos el acceso **/var/www/joomla** a apache user y configuramos los permisos.

```

root@servidorA:~# sudo chown -R www-data.www-data /var/www/joomla/
root@servidorA:~# sudo chmod -R 777 /var/www/joomla/
root@servidorA:~# _
    
```

Imagen N° 34 comando de permiso de acceso.

Para posteriormente descomprimirlos en la carpeta **joomla** de la siguiente manera: **sudo unzip Joomla_3.8.8-Stable-Full_Package.zip**. Esperamos mientras se prepara **joomla** y ya podemos continuar desde nuestro navegador de Windows para instalar con la ip del servidor en este caso **192.168.0.100/joomla** y crear a nuestro gusto la página web.





Imagen N° 35 Página de inicio para instalar joomla.

Se configura los Ips de ambos servidores y se procede a reiniciar los mismos.

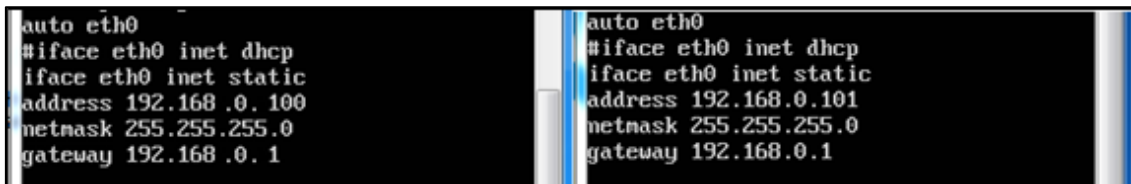


Imagen N° 36 Configuración de los IPs en ambos servidores.

Para realizar las peticiones vamos a usar la herramienta ApacheBench que nos permite simular un grupo de peticiones hacia un servidor, para esta investigación se realizara desde un servidor cliente.



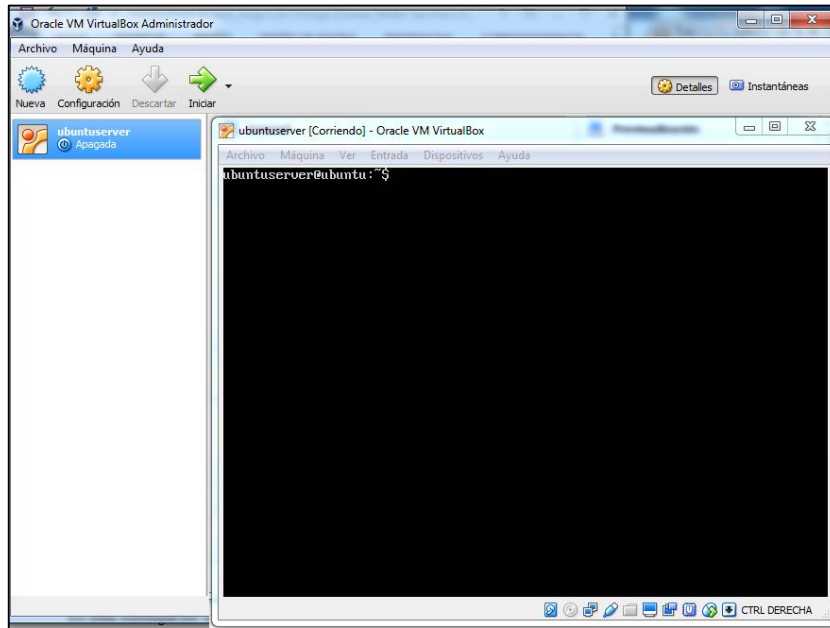


Imagen N° 37 Servidor Cliente.

Procedemos a instalar la herramienta con el siguiente comando:

sudo apt-get install apache2-utils.

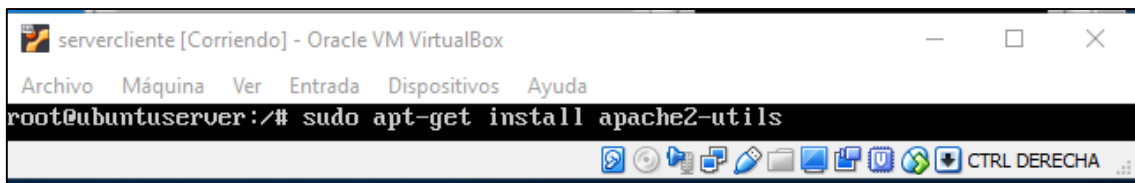


Imagen N° 38 Comando de instalación de la herramienta ApacheBench.

Una vez hecho esto ahora procedemos a simular las peticiones hacia el servidor

con el siguiente comando:

ab -n 386 -c 100 -g grafica.data 192.168.0.70/joomla



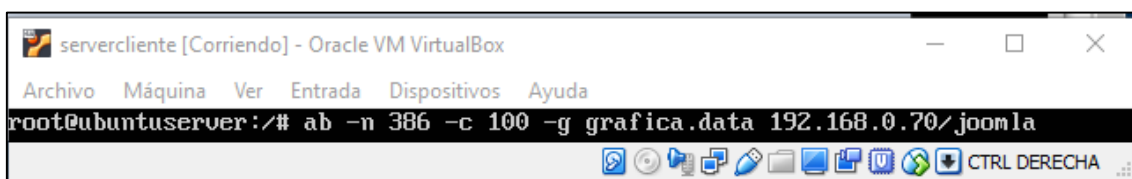


Imagen N° 39 Comando para simular las peticiones hacia el servidor HAProxy.

Con este comando especificamos una simulación donde **ab** nos permite hacer uso de la herramienta, **-n** especifica la cantidad de peticiones a enviar, **-c** especifica las peticiones simultaneas y especificamos la ruta del servidor en este caso seria **192.168.0.70/joomla** que es el Ip de nuestro servidor HAProxy. (Se puede verificar en el anexo N°3 en la imagen N° 15).

CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

En esta investigación se hizo uso de la herramienta haproxy la cual nos permite realizar balanceo de carga permitiendo dar un buen tiempo de respuestas a las solicitudes realizadas por el usuario y también poder evitar caída del servidor web.

El balanceador de carga implementado se le realizó pruebas de rendimiento dando como resultados un buen tiempo de respuestas a las solicitudes una a una y a las solicitudes simultáneas como también consume una cantidad mínima de recursos del servidor para atender las peticiones de los usuarios.

La investigación del balanceo de cargas se comparó dos algoritmos teniendo más eficacia en el algoritmo round robin, obteniendo un mejor tiempo de respuesta y un consumo mínimo de recurso del servidor a las solicitudes realizadas por el usuario.

Actualmente en el mercado existe varias opciones de clúster (balanceadores de carga, alto desempeño y alta disponibilidad) estas opciones son caras a la vista de las empresas pequeñas. La opción de clúster que se plantea en la investigación se maneja bien en tiempo de respuesta y rendimiento para las solicitudes que pueda recibir las páginas web de estas empresas.



6.2. Recomendaciones

La investigación en balanceo de cargas es muy amplia buscando la mejora continua de esta investigación, se recomienda a futuros investigadores en este campo implementar nuevas herramientas, algoritmos para mejorar los tiempos de respuesta y la optimización de los recursos de un servidor.

Las herramientas que se emplearon en la investigación se tienen que actualizar en la medida que sea requieran así también buscar herramientas de seguridad para posibles ataque informáticos a los servidores web.

Se recomienda realizar un caso de estudio en este campo para realizar una clasificación de las peticiones hechas a un servidor implementadas en las pymes.

Referencias

- Alegsa L. (2010). *Definición de cuello de botella*. Recuperado el 25/05/2016 de:
<http://www.alegsa.com.ar/Dic/cuello%20de%20botella.php>
- Bookman C. (2003). *Linux Clustering: Building and Maintaining Linux Clusters*. E.E.U.U .
- Branch J., Ramos F. y Mesa A. (2009). *Proposed method load balancing in heterogeneous cluster simulated in NS2*. Mexico.
- Buyya, (1999). "High performance cluster computing/ 1, Architectures and systems". (Australia).
- Castells M.(2000). *La era de la información: economía, sociedad y cultura* - 2a edicion. E.E.U.U.
- Don MacVittie. (2010). *Intro to Load Balancing for Developers – The Algorithms*. Uninted States. Recuperado el 5/05/2016 de:
<https://devcentral.f5.com/articles/intro-to-load-balancing-for-developers-ndash-the-algorithms>.
- EVIUXA (2004). "Superordenadores". Trabajo universitario. México.
Disponible en:
<http://html.rincondelvago.com/superordenadores.html>



Figuerero C. y Simbaña H (2013). *Análisis, comparación e implementación de una infraestructura virtual open source con alta disponibilidad basada en clusters, para servidores y escritorios dentro de las instalaciones de la empresa synergyhard CIA. LTDA.* Ecuador

Free Software Foundation(2001). *Que es el software Libre.* Recuperado el 20/06/2016 de: <https://www.gnu.org/philosophy/free-sw.es.html>

Gallardo M.(2011). *Integración de aplicaciones de software libre aplicado a un clúster de alta disponibilidad y balanceo de carga de servidores proxy.* Ecuador

García J., Aguinaga I. y Mora A. (2000). *Aprenda Linux como si estuviera en primero.* España

Gutierrez A. (2012). *Clúster de alta disponibilidad y balanceo de carga sobre un servidor web.* España.

Gopinath G. y Shriram K (2014). *An in-depth analysis and study of Load balancing techniques in the cloud computing environment.* Ettimadai, Coimbatore.

Hailperin M. (2011). *Operatin Systems and Middleware* .1a edición. USA.



- Hertzog R. y Mas R. (2015). *El manual de administrador de Debian*.
- Lopez J. (2007). *Algoritmos y programación- 2da edition*.
- Marchionni E. (2011). *Administrador de servidores - 1a edicion*. - Buenos Aires: Fox Andina; Banfield – Lomas de Zamora: Dradi
- Mercedes M., Diaz T. y Ruiz E.(2012) *Cluster de balanceo de carga y alta disponibilidad para servicios web y mail* . Colombia
- Mittal s. (2013). *Java Hungry*. Boston,USA. Recuperado el 20/05/2016 de:
<http://javahungry.blogspot.com/2013/09/round-robin-scheduling-algorithm-with-example-java-program-code.html>
- Philippe A. y Dordoigne J. (2006). *Redes informáticas*. España
- Paulin a. y shanthi v. (2014). *A load balancing model using firefly algorithm in cloud computing*. chennai, india.
- Portilla a., Perez J., Hoz D., (2000).*e-logistics(I). Nuevas tecnologías de la información*. España.
- Red Hat (2003). *Manual de referencia de Red Hat Linux*.
- Santana, J.(2010). *Algoritmos de balance de carga con manejo de información parcial*. México – D.F.
- Santiago c., Garcia R. y Santos J.. (2007). *Fundamentos de sistemas operativos*. Mexico



Santos M. (2004). *Curso de Linux*. España.

SuSE Linux AG (2004). *MANUAL DE ADMINISTRACIÓN*

Stallings y William (2004). *Comunicaciones y redes de computadores*.
España

Talavera M. y Albea C. (2014). *Control de balanceo de carga de un grupo
de servidores de red*. España

Vega J. (2015). *Implementación de un balanceador de carga y alta
disponibilidad para servicios web*. México.

Anexos:

Anexo N° 1

Las tablas y cuadros estadísticos hacen referencia a las tareas ejecutadas por el servidor y ver cuánto es la demora en la respuesta. También se muestra un cuadro para visualizarlo gráficamente (Branch J., Ramos F. y Mesa A., 2009).

TABLA I. RESULTADOS DE LA EJECUCIÓN DE LA CARGA DE TRABAJO CON 3 NODOS ESCLAVOS

Cantidad de tareas totales ejecutadas	Tiempo de ejecución para cada método [seg]		
	Aleatorio	Uniforme	Propuesto
99	6,361607	6,449856	5,712228
141	8,585431	8,912146	7,595445
149	8,767952	9,413576	8,042756
250	14,42972	15,71695	13,50117
396	23,14886	24,54513	21,01382
479	27,21146	29,05732	25,03099
553	31,74227	34,24748	29,50111
1179	66,19169	71,70027	61,83991

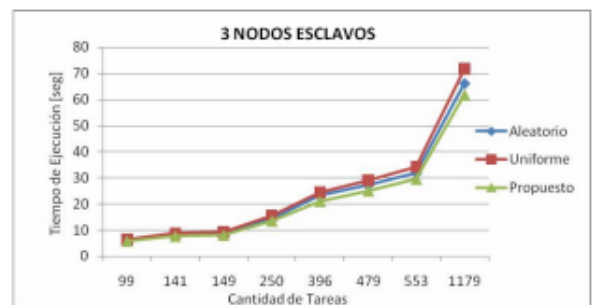


Figura 1. Resultados de la ejecución de carga de trabajo con 3 nodos esclavos.

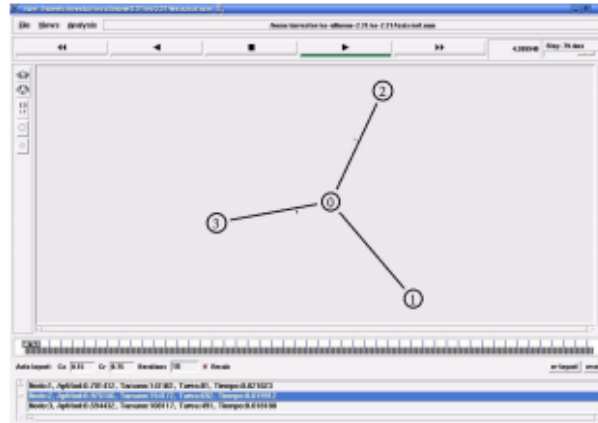


Figura 2. Simulación en el Network Simulator 2 (NS-2) con 3 nodos esclavos.



Anexo N° 2

A continuación se muestra las tablas con los valores de las peticiones realizadas con ambos algoritmos con la herramienta Haproxy.

Tabla 10 Respuesta de 386 peticiones con 100 peticiones concurrentes usando el algoritmo Least Connection

HAProxy version 1.4.18, released 2011/09/16
 Statistics Report for pid 2145

> General process information

pid = 2145 (process #1, sbproc = 1)
 uptime = 0d 0h22m24s
 system limits: memmax = unlimited; ulimitn = 785
 maxsock = 785; maxconn = 386; maxpipes = 0
 current conn = 2; current pipes = 0/0
 Running tasks: 14

Queue		Session rate		Sessions			Bytes		Denied		Errors		Warnings		Server															
Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Req	Reqs	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Downtm	Thrtle		
Frontend																														
			2	400	-	2	101	386	472		31 698	198 887	0	0	82	0	0	0	0	0	2m24s UP	OPEN								
servidorB																														
0	0	-	0	192		0	50	-	192	192	15 300	82 752	0	0	0	0	0	0	0	0	2m24s UP	L4OK in 0ms	1	Y	-	0	0	0s	-	
prueba																														
0	0	-	0	194		0	50	-	194	194	15 520	83 420	0	0	0	0	0	0	0	0	2m24s UP	L4OK in 0ms	1	Y	-	0	0	0s	-	
Backend																														
0	0		0	386		0	100	386	386	386	31 698	198 887	0	0	0	0	0	0	0	0	2m24s UP		2	2	0	0	0	0s	-	

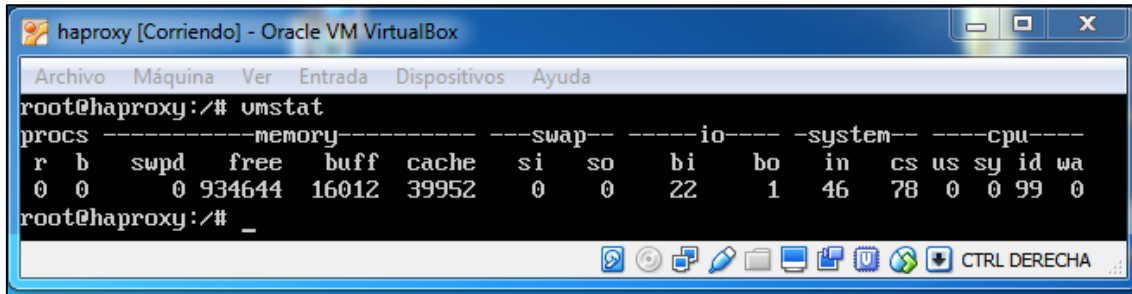


Imagen N° 40 Consumo de memoria de la Tabla 10.



Tabla 11 Respuesta de 386 peticiones con 193 peticiones concurrentes usando el algoritmo Least Connection.

HAProxy version 1.4.18, released 2011/09/16
 Statistics Report for pid 1789

> General process information

pid = 1789 (process #1, nproc = 1)
 uptime = 0d 0h00m13s
 system limits: memmax = unlimited; ulimit = 785
 maxsock = 785; maxconn = 386; maxpipes = 0
 current conn = 2; current pipes = 0/0
 Running tasks: 1/4

servidoresweb	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle		
Frontend	0	0	-	2	209	-	2	193	386	406		31 289	177 978	0	0	17					OPEN										
servidorB	0	0	-	0	90		0	97	-	187	187	14 900	80 597	0	0	0	0	0	0	0	13s UP	L40k in 0ms	1	Y	-	0	0	0s	-		
prueba	0	0	-	0	103		0	96	-	199	199	15 920	85 970	0	0	0	0	0	0	0	13s UP	L40k in 0ms	1	Y	-	0	0	0s	-		
Backend	0	0	-	0	193		0	193	386	386	386	31 289	177 978	0	0	0	0	0	0	0	13s UP		2	2	0	0	0	0s	-		

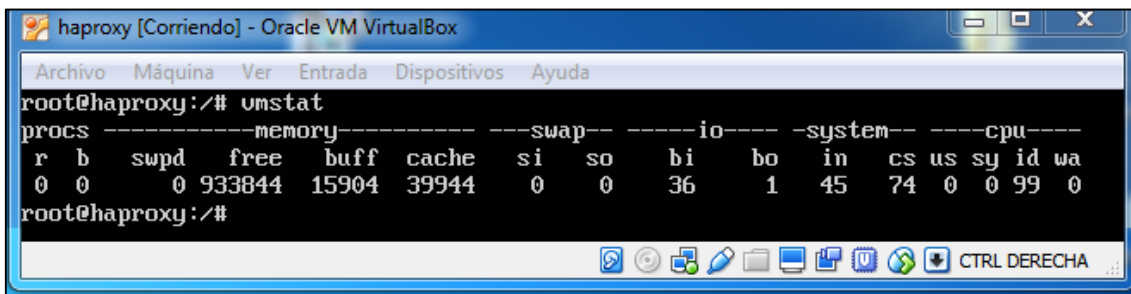


Imagen N° 41 Consumo de memoria de la Tabla 11.

Tabla 12 Respuesta de 386 peticiones con 100 peticiones concurrentes usando el algoritmo Round Robin

HAProxy version 1.4.18, released 2011/09/16
 Statistics Report for pid 2343

> General process information

pid = 2343 (process #1, nproc = 1)
 uptime = 0d 0h00m20s
 system limits: memmax = unlimited; ulimit = 785
 maxsock = 785; maxconn = 386; maxpipes = 0
 current conn = 2; current pipes = 0/0
 Running tasks: 1/4

servidoresweb	Queue			Session rate			Sessions				Bytes		Denied		Errors		Warnings		Server												
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle		
Frontend	0	0	-	2	259	-	2	100	386	424		31 289	181 398	0	0	35					OPEN										
servidorB	0	0	-	0	112		0	58	-	192	192	15 440	83 183	0	0	0	0	0	0	0	25s UP	L40k in 0ms	1	Y	-	0	0	0s	-		
prueba	0	0	-	0	113		0	50	-	193	193	15 440	82 990	0	0	0	0	0	0	0	25s UP	L40k in 0ms	1	Y	-	0	0	0s	-		
Backend	0	0	-	0	225		0	100	386	386	386	31 289	181 398	0	0	0	0	0	0	0	25s UP		2	2	0	0	0	0s	-		



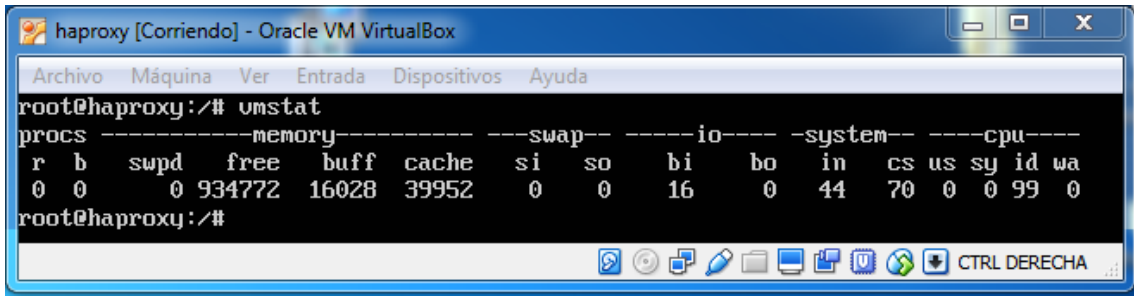


Imagen N° 42 Consumo de memoria de la Tabla 12.

Tabla 13 Respuesta de 386 peticiones con 193 peticiones concurrentes usando el algoritmo Round Robin

HAProxy version 1.4.18, released 2011/09/16
 Statistics Report for pid 2372

> General process information

pid = 2372 (process #1, nbproc = 1)
 uptime = 0s 0m0.1m54s
 system limits: memmax = unlimited; ulimit-n = 785
 maxsock = 785; maxconn = 386; maxpipes = 0
 queue conns = 1; current pipes = 0/0
 Running tasks: 1/3

servidoresweb	Queue			Session rate			Sessions				Bytes		Denied			Errors			Warnings		Status	LastChk	Server						
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis			Wght	Act	Bck	Chk	Dwn	Downtm	Thrtm
Frontend	0	0	-	1	360	-	1	193	386	447		31 289	185 880	0	0	56					OPEN								
sendorB	0	0	-	0	175	-	0	92	-	123	193	15 440	\$3 193	0	0	0	0	0	0	0	1m54s UP	L4OK: in 0ms	1	Y	-	0	0	0s	-
prueba	0	0	-	0	175	-	0	91	-	193	193	15 440	\$2 990	0	0	0	0	0	0	0	1m54s UP	L4OK: in 0ms	1	Y	-	0	0	0s	-
Backend	0	0	-	0	350	-	0	183	386	386	386	31 289	185 880	0	0	0	0	0	0	0	1m54s UP		2	2	0	0	0	0s	-

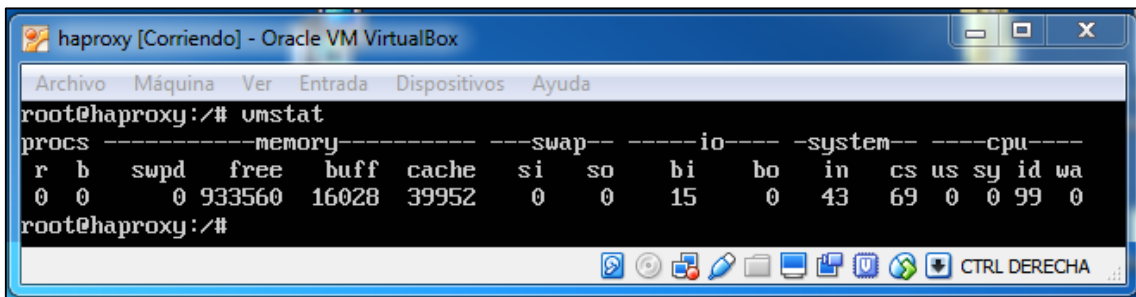


Imagen N° 43 Consumo de memoria de la Tabla 13.



Anexo N° 3

Pruebas de solicitudes enviadas al clúster para medir el tiempo de respuesta con la herramienta de Apache bench ejecutadas con el siguiente comando: `ab -n 386 -c 50 -g grafica.data /192.168.1.70/joomla` que realiza las peticiones al servidor.

```

Server Hostname: 192.168.1.70
Server Port: 80

Document Path: /
Document Length: 154 bytes

Concurrency Level: 100
Time taken for tests: 0.407 seconds
Complete requests: 386
Failed requests: 192
  (Connect: 0, Receive: 0, Length: 192, Exceptions: 0)
Total transferred: 166172 bytes
HTML transferred: 59636 bytes
Requests per second: 949.02 [#./sec] (mean)
Time per request: 105.372 [ms] (mean)
Time per request: 1.054 [ms] (mean, across all concurrent requests)
Transfer rate: 398.96 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:  10   37  10.3   36   60
Processing:  22   57  17.9   55  114
Waiting:  22   56  17.8   54  108
Total:  43   93  20.6   93  150

Percentage of the requests served within a certain time (ms)
 50%    93
 66%    99
 75%   105
 80%   110
 90%   121
 95%   126
 98%   148
 99%   149
100%   150 (longest request)

```

Imagen N° 44 Pruebas de 386 peticiones con 100 peticiones concurrentes usando algoritmo Least connection.



```

Server Hostname: 192.168.1.70
Server Port: 80

Document Path: /
Document Length: 155 bytes

Concurrency Level: 193
Time taken for tests: 3.451 seconds
Complete requests: 386
Failed requests: 199
  (Connect: 0, Receive: 0, Length: 199, Exceptions: 0)
Total transferred: 166167 bytes
HTML transferred: 59631 bytes
Requests per second: 111.84 [#./sec] (mean)
Time per request: 1725.696 [ms] (mean)
Time per request: 8.941 [ms] (mean, across all concurrent requests)
Transfer rate: 47.02 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    18   719 500.5   1062   1460
Processing:  435   817 207.6    800   1300
Waiting:    434   816 206.5    799   1298
Total:      460  1536 539.7   1709   2272

Percentage of the requests served within a certain time (ms)
 50%: 1709
 66%: 1923
 75%: 1983
 80%: 2016
 90%: 2206
 95%: 2232
 98%: 2249
 99%: 2261
100%: 2272 (longest request)
    
```

Imagen N° 45 Pruebas de 386 peticiones con 193 peticiones concurrentes usando algoritmo Least connection.

```

Server Hostname: 192.168.1.70
Server Port: 80

Document Path: /
Document Length: 154 bytes

Concurrency Level: 100
Time taken for tests: 0.353 seconds
Complete requests: 386
Failed requests: 193
  (Connect: 0, Receive: 0, Length: 193, Exceptions: 0)
Total transferred: 166173 bytes
HTML transferred: 59637 bytes
Requests per second: 1093.77 [#./sec] (mean)
Time per request: 91.427 [ms] (mean)
Time per request: 0.914 [ms] (mean, across all concurrent requests)
Transfer rate: 459.83 [Kbytes/sec] received

Connection Times (ms)
  min  mean[+/-sd] median  max
Connect:    12    30  8.4    34    46
Processing:  25    54 17.9    51   116
Waiting:    25    53 17.6    50   102
Total:      43    84 19.0    84   136

Percentage of the requests served within a certain time (ms)
 50%: 84
 66%: 88
 75%: 92
 80%: 97
 90%: 109
 95%: 123
 98%: 134
 99%: 135
100%: 136 (longest request)
    
```

Imagen N° 46 Pruebas de 386 peticiones con 100 peticiones concurrentes usando algoritmo Round robin.




```

Server Hostname: 192.168.1.70
Server Port: 80

Document Path: /
Document Length: 155 bytes

Concurrency Level: 193
Time taken for tests: 0.491 seconds
Complete requests: 386
Failed requests: 193
  (Connect: 0, Receive: 0, Length: 193, Exceptions: 0)
Total transferred: 166173 bytes
HTML transferred: 59637 bytes
Requests per second: 785.38 [#/sec] (mean)
Time per request: 245.742 [ms] (mean)
Time per request: 1.273 [ms] (mean, across all concurrent requests)
Transfer rate: 330.18 [Kbytes/sec] received

Connection Times (ms)
      min  mean[+/-sd] median  max
Connect:    15   110  43.0   84   177
Processing:  49   102  22.2   95   165
Waiting:    49   101  22.2   95   165
Total:     137   212  54.0  192   319

Percentage of the requests served within a certain time (ms)
 50%    192
 66%    237
 75%    252
 80%    282
 90%    293
 95%    304
 98%    318
 99%    319
100%    319 (longest request)
    
```

Imagen N° 47 Pruebas de 386 peticiones con 193 peticiones concurrentes usando algoritmo Round Robin..



Anexo N° 4

A continuación se muestra una tabla con las 100 pruebas realizadas con cada algoritmo y el cálculo de su promedio de tiempos de respuesta.

Tabla 14. Cuadro de 100 Pruebas de 386 peticiones con 193 peticiones simultaneas.

Peticiones 386 - Peticiones en simultaneo 193										
N°	Algoritmos	Numero de peticiones	Tiempo de prueba	Tiempo x peticion	Tiempo total de peticiones simultaneas	Algoritmos	Numero de peticiones	Tiempo de prueba	Tiempo x peticion	Tiempo total de peticiones simultaneas
1	Round Robin	386	0,478	1,2380	238,8690	Least Connection	386	0,443	1,1480	221,5690
2	Round Robin	386	0,361	0,9340	180,2660	Least Connection	386	0,374	0,9690	186,9850
3	Round Robin	386	0,356	0,9210	177,7810	Least Connection	386	0,373	0,9660	186,4480
4	Round Robin	386	0,38	0,9840	189,8780	Least Connection	386	0,36	0,9330	180,1630
5	Round Robin	386	0,419	1,0860	209,5680	Least Connection	386	0,386	0,9990	192,8910
6	Round Robin	386	0,356	0,9220	177,9970	Least Connection	386	0,364	0,9430	181,9210
7	Round Robin	386	0,379	0,9810	189,2590	Least Connection	386	0,38	0,9850	190,0580
8	Round Robin	386	0,398	1,0310	190,9500	Least Connection	386	0,364	0,9420	181,8590



9	Round Robin	386	0,365	0,9460	182,5040	Least Connection	386	0,366	0,9490	183,1720
10	Round Robin	386	0,396	1,0270	198,2140	Least Connection	386	0,369	0,9550	184,3620
11	Round Robin	386	0,639	1,6550	319,4770	Least Connection	386	0,368	0,9530	183,8930
12	Round Robin	386	0,38	0,9840	189,8850	Least Connection	386	0,353	0,9160	176,7240
13	Round Robin	386	0,369	0,9560	184,5570	Least Connection	386	0,409	1,0590	204,4720
14	Round Robin	386	0,35	0,9080	175,1600	Least Connection	386	0,384	0,9550	192,0600
15	Round Robin	386	0,361	0,9360	180,5810	Least Connection	386	0,391	1,0130	195,5370
16	Round Robin	386	0,377	0,9770	188,5640	Least Connection	386	0,998	0,9980	192,5240
17	Round Robin	386	0,368	0,9540	184,2040	Least Connection	386	0,379	0,9830	189,6250
18	Round Robin	386	0,353	0,9150	176,5690	Least Connection	386	0,371	0,9620	185,7410
19	Round Robin	386	0,375	0,9710	187,4140	Least Connection	386	0,392	1,0150	195,9460
20	Round Robin	386	0,383	0,9330	191,6330	Least Connection	386	0,364	0,9440	182,1910
21	Round Robin	386	0,37	0,9580	104,9650	Least Connection	386	0,403	1,0440	201,5040



22	Round Robin	386	0,366	0,9490	183,1420	Least Connection	386	0,36	0,9330	180,0490
23	Round Robin	386	0,348	0,9010	173,8930	Least Connection	386	0,379	0,9810	189,4210
24	Round Robin	386	0,375	0,9710	187,3620	Least Connection	386	0,366	0,9490	183,2310
25	Round Robin	386	0,364	0,9440	182,1140	Least Connection	386	0,349	0,9030	174,2730
26	Round Robin	386	0,329	0,8510	164,3270	Least Connection	386	0,388	1,0050	193,9650
27	Round Robin	386	0,362	0,9390	181,2200	Least Connection	386	0,365	0,9440	182,2820
28	Round Robin	386	0,36	0,9330	179,9760	Least Connection	386	0,346	0,8970	173,0300
29	Round Robin	386	0,361	0,9360	180,7630	Least Connection	386	0,413	1,0710	206,6400
30	Round Robin	386	0,369	0,9560	184,5550	Least Connection	386	0,377	0,9760	188,4560
31	Round Robin	386	0,634	1,6440	317,2330	Least Connection	386	0,439	1,1370	219,4070
32	Round Robin	386	0,405	1,0500	202,6370	Least Connection	386	0,352	0,9120	175,9770
33	Round Robin	386	0,372	0,9640	185,9830	Least Connection	386	0,366	0,9490	183,0650
34	Round Robin	386	0,389	1,0080	194,6080	Least Connection	386	0,387	1,0020	193,2960
35	Round Robin	386	0,389	1,0070	194,4190	Least Connection	386	0,388	1,0040	193,8290



36	Round Robin	386	0,407	1,0550	203,6470	Least Connection	386	0,395	197,5930	976,7600
37	Round Robin	386	0,548	1,4190	273,8460	Least Connection	386	0,399	1,0330	199,3060
38	Round Robin	386	0,381	0,9870	190,4990	Least Connection	386	0,51	1,3210	254,9560
39	Round Robin	386	0,396	1,0260	197,9300	Least Connection	386	0,372	0,9630	185,8830
40	Round Robin	386	0,428	1,1080	213,8600	Least Connection	386	0,387	1,0030	193,6630
41	Round Robin	386	0,407	1,0550	203,5590	Least Connection	386	0,369	0,9570	184,7490
42	Round Robin	386	0,451	1,1680	225,5200	Least Connection	386	0,379	0,9820	189,5180
43	Round Robin	386	0,38	0,9830	189,7930	Least Connection	386	0,419	1,0840	209,3030
44	Round Robin	386	0,38	0,9850	190,1510	Least Connection	386	0,385	0,9990	192,7190
45	Round Robin	386	0,423	1,0950	211,3340	Least Connection	386	0,498	1,2960	249,0060
46	Round Robin	386	0,372	0,9630	185,8780	Least Connection	386	0,349	0,9030	174,3080
47	Round Robin	386	0,362	0,9380	181,0630	Least Connection	386	0,373	0,9670	186,5690
48	Round Robin	386	0,334	0,8650	167,0180	Least Connection	386	0,501	1,2990	250,6370



49	Round Robin	386	0,349	0,9050	174,5690	Least Connection	386	0,497	1,2870	248,2980
50	Round Robin	386	0,379	0,9830	189,6560	Least Connection	386	0,469	1,2150	234,4690
51	Round Robin	386	0,342	0,8870	171,1870	Least Connection	386	0,373	0,9650	186,2920
52	Round Robin	386	0,357	0,9240	178,3740	Least Connection	386	0,428	1,1080	213,7800
53	Round Robin	386	0,368	0,9540	184,0610	Least Connection	386	0,328	1,8500	168,9570
54	Round Robin	386	0,349	0,9050	174,5870	Least Connection	386	0,361	0,9360	180,6090
55	Round Robin	386	0,327	0,8470	163,4620	Least Connection	386	0,393	1,0180	196,4600
56	Round Robin	386	0,37	0,9580	184,9150	Least Connection	386	0,377	0,9750	188,2690
57	Round Robin	386	0,372	0,9630	185,7900	Least Connection	386	0,46	1,1930	230,1840
58	Round Robin	386	0,357	0,9260	178,7450	Least Connection	386	0,382	0,9890	190,7830
59	Round Robin	386	0,36	0,9340	180,2490	Least Connection	386	0,465	1,2050	232,5900
60	Round Robin	386	0,349	0,9050	174,7350	Least Connection	386	0,371	0,9620	185,6380
61	Round Robin	386	0,378	0,9790	188,8950	Least Connection	386	0,372	0,9630	185,9420
62	Round Robin	386	0,378	0,9790	189,0230	Least Connection	386	0,41	1,0620	204,9870



63	Round Robin	386	0,353	0,9150	176,5440	Least Connection	386	4,126	10,6900	2063,2330
64	Round Robin	386	0,337	0,8740	168,7220	Least Connection	386	0,65	1,6830	324,8070
65	Round Robin	386	0,331	0,8580	165,6110	Least Connection	386	0,54	1,3990	269,9120
66	Round Robin	386	0,365	0,9470	182,6910	Least Connection	386	0,555	1,4370	277,2900
67	Round Robin	386	0,425	1,1000	212,3410	Least Connection	386	0,37	0,9580	184,9540
68	Round Robin	386	0,348	0,9020	174,0880	Least Connection	386	0,573	1,4850	286,5910
69	Round Robin	386	0,359	0,9310	179,7450	Least Connection	386	0,471	1,2200	235,4150
70	Round Robin	386	0,35	0,9060	174,9200	Least Connection	386	0,433	1,1210	216,3380
71	Round Robin	386	0,708	1,8350	354,2040	Least Connection	386	0,457	1,1850	228,6650
72	Round Robin	386	0,505	1,3080	252,4280	Least Connection	386	0,483	1,2520	241,5880
73	Round Robin	386	0,464	1,2010	231,8280	Least Connection	386	0,685	1,7750	342,5560
74	Round Robin	386	0,441	1,1430	220,5660	Least Connection	386	0,423	1,0960	211,5800
75	Round Robin	386	0,359	0,9300	179,4820	Least Connection	386	0,478	1,2390	239,2170



76	Round Robin	386	0,409	1,0580	204,2770	Least Connection	386	0,51	1,3220	255,0650
77	Round Robin	386	0,36	0,9330	179,9790	Least Connection	386	0,465	1,2060	232,7480
78	Round Robin	386	0,39	1,0990	194,7740	Least Connection	386	0,662	1,7140	330,7840
79	Round Robin	386	0,427	1,1050	213,2520	Least Connection	386	0,536	1,3890	268,1350
80	Round Robin	386	0,359	0,9300	179,4840	Least Connection	386	0,608	1,5750	304,0220
81	Round Robin	386	0,343	0,8890	171,6710	Least Connection	386	0,638	1,6530	318,9330
82	Round Robin	386	0,379	0,9810	189,2870	Least Connection	386	0,51	1,3220	255,0760
83	Round Robin	386	0,384	0,9940	191,9240	Least Connection	386	0,724	1,8750	361,8510
84	Round Robin	386	0,365	0,9460	182,5060	Least Connection	386	0,56	1,4510	279,9500
85	Round Robin	386	0,417	1,0800	208,3460	Least Connection	386	0,495	1,2830	247,5700
86	Round Robin	386	0,378	0,9780	188,8240	Least Connection	386	0,707	1,8310	353,3090
87	Round Robin	386	0,371	0,9610	185,5170	Least Connection	386	0,69	1,7880	345,1140
88	Round Robin	386	0,398	1,0320	199,0790	Least Connection	386	0,676	1,7500	337,8250
89	Round Robin	386	0,356	0,9230	178,1680	Least Connection	386	0,697	1,8050	348,2890



90	Round Robin	386	0,355	0,9210	177,7080	Least Connection	386	0,563	1,4580	281,2980
91	Round Robin	386	0,346	0,8970	173,1520	Least Connection	386	0,582	1,5070	290,8260
92	Round Robin	386	0,587	1,5190	293,2520	Least Connection	386	0,459	1,1900	229,7390
93	Round Robin	386	0,471	1,2190	235,2540	Least Connection	386	0,424	1,0990	212,1670
94	Round Robin	386	0,396	1,0260	198,0110	Least Connection	386	0,571	1,4800	285,6190
95	Round Robin	386	0,434	1,1240	216,9930	Least Connection	386	0,687	1,7790	343,3340
96	Round Robin	386	0,414	1,0710	206,7560	Least Connection	386	0,468	1,2120	234,0040
97	Round Robin	386	0,367	0,9510	183,4990	Least Connection	386	0,481	1,2470	240,6690
98	Round Robin	386	0,343	0,8890	171,4860	Least Connection	386	0,394	1,0200	196,7680
99	Round Robin	386	0,373	0,9670	186,6630	Least Connection	386	0,458	1,1870	229,0950
100	Round Robin	386	0,371	0,9600	185,2710	Least Connection	386	0,446	1,1550	222,8580
			Promedio	1,0127	194,5118			Promedio	3,2326	250,4840



Anexo N° 5

En la imagen que se muestra a continuación se puede observar las páginas las cuales se están consultando los dos servidores.

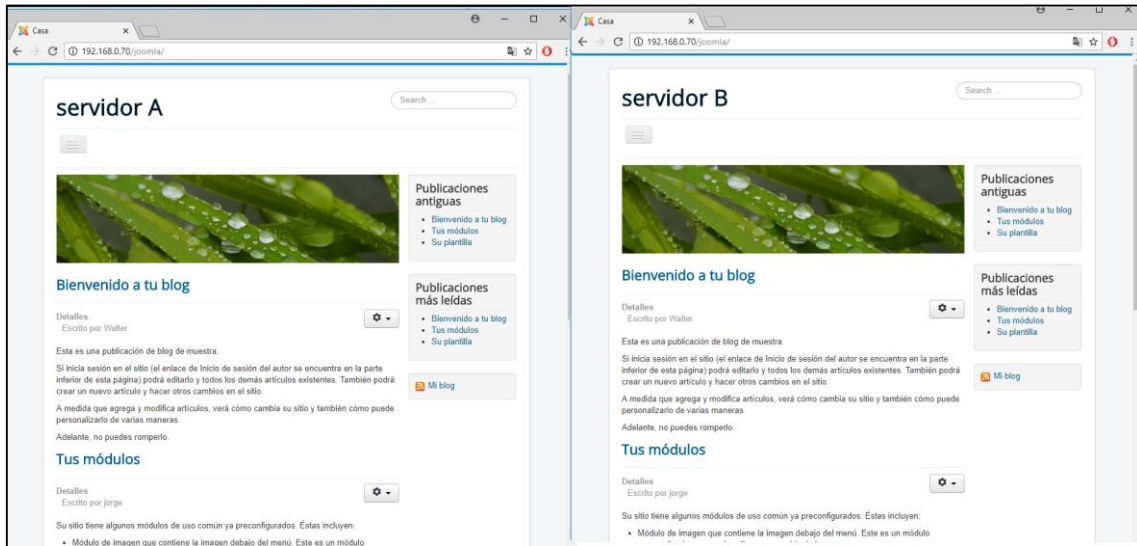


Imagen N° 48 Páginas Web consultadas hacia los servidores.

