UNIVERSIDAD POLITÉCNICA DE SINALOA
PROGRAMA ACADÉMICO DE
INGENIERÍA EN INFORMÁTICA

“Implementación de *Docker* en la gestión del entorno de desarrollo”

Para cumplir con la acreditación de las estadías profesionales
y contar con los créditos necesarios para
obtener el grado de Ingeniería en Informática

Autor:

Lilia María Tirado Zatarain

Asesor:

M.S.I. Rosa Karina González Trigueros

Asesor OR:

Ing. Alejandro Duarte Sánchez

Mazatlán, Sinaloa a 15 de Diciembre de 2017

intugo

Zapopan, Jal. 25 de septiembre de 2017

M.C. MARÍA GUADALUPE GARCÍA RAMÍREZ
DIRECTORA DE VINCULACIÓN, DIFUSIÓN Y EXT. UNIVERSITARIA.
UNIVERSIDAD POLITÉCNICA DE SINALOA.

PRESENTE

Por este medio, hago de su conocimiento que el alumno(a) el C. Lilia Maria Tirado Zatarain, con número de matrícula 2014030473, de la carrera de Ingeniería en Informática, ha sido aceptado para realizar su estadía práctica , en esta empresa, durante el período que comprende del 4 de septiembre de 2017 al 8 de diciembre de 2017, para cubrir un total de 600 horas.

Dicho alumno realizará actividades dentro del área, Desarrollo bajo la supervisión del C. Martha Iris Vivían Cortez Felix, Coordinadora de Recursos Humanos y Operaciones.

Sin otro particular, le envío un cordial saludo.

Atte.

Martha Iris Vivían Cortez Felix
Coordinadora de Recursos Humanos y Operaciones

intugo
Vangtel Mexico, S.A. de C.V.
Coordinación R.H.
VME050706UUA

www.intugo.co
VANGTEL MÉXICO S.A. DE C.V.
Tel. +52 333 030 7081
Av. Lopez Mateos Sur #2077 Int. Z-7,
Col. Jardines de Plaza del Sol,
C.P. 44510 Guadalajara, Jalisco, MX.
/trabajaenintugo /intugomx

Zapopan, Jal. 08 de Diciembre de 2017

M.C. MARÍA GUADALUPE GARCÍA RAMÍREZ
DIRECTORA DE VINCULACIÓN, DIFUSIÓN Y EXT. UNIVERSITARIA.
UNIVERSIDAD POLITÉCNICA DE SINALOA.

PRESENTE

Por este medio, hago de su conocimiento que el alumno(a) el C. **Lilia María Tirado Zatarain**, con número de matrícula **2014030473**, de la carrera de Ingeniería en Informática, ha cumplido con 600 horas correspondientes a estadía final , en esta empresa/institución, durante el período que comprende del 04/09/2017 al 08/12/2017.

Dicho alumno realizó actividades dentro del área Desarrollo, bajo la supervisión del C. Martha Vivían Cortez Felix, Coordinadora de Recursos Humanos y Operaciones.

Sin otro particular, le envío un cordial saludo.

Atte.

Martha Iris Vivían Félix
Coordinadora de Recursos Humanos y Operaciones

www.intugo.co
VANGTEL MÉXICO S.A. DE C.V.
Tel. +52 333 030 7081
Av. López Mateos Sur #2077 Int. Z-7,
Col. Jardines de Plaza del Sol,
C.P. 44510. Guadalajara, Jalisco, MX.
/trabajaenintugo /intugomx

Vangtel Mexico, S.A. de C.V.
Coordinación R.H.
VME050706UUA

UNIVERSIDAD POLITÉCNICA DE SINALOA

SINALOA
SECRETARÍA DE EDUCACIÓN PÚBLICA Y CULTURA

C. TIRADO ZATARAIN LILIA MARIA **Folio 2014030473-2017-093**
P r e s e n t e .-

Por medio de la presente me permito comunicarle que **es de aceptarse el tema de tesina,** el cuál se ha solicitado bajo el título:

" Implementación de Docker en la gestión del entorno de desarrollo"

mismo que usted desarrollará con objeto de dar lugar a los trámites conducentes para la acreditación de la asignatura de Estadías Profesionales de la Unidad Académica de:

Ingeniería en Informática

Así mismo, le comunico que para el desarrollo de la citada tesina le ha sido asignado como Director Asesor de la misma a la: **M.S.I. Rosa Karina González Trigueros** y como asesores a la **M.C. Gloria Irene Téllez Rodríguez** y a la **M.T.I. Ismaylia Saucedo Ugalde**

Sin otro particular por el momento, aprovecho la ocasión para enviarle un cordial saludo.

Atentamente

Dr. Rodolfo Ostos Robles
Director del Programa Académico de Ingeniería en Informática
Universidad Politécnica de Sinaloa

C.c.p. Interesado

SEP

UTP

UNIVERSIDAD POLITÉCNICA DE SINALOA

SINALOA
SECRETARÍA DE EDUCACIÓN PÚBLICA Y CULTURA

C. TIRADO ZATARAIN LILIA MARIA **Folio 2014030473-2017-093**
P r e s e n t e.-

Por este conducto le envío un cordial saludo y aprovecho la ocasión para notificarle que el jurado que le fue asignado para evaluar la tesina desarrollada en las estadías profesionales denominada **" Implementación de Docker en la gestión del entorno de desarrollo"** y que después de ser revisada en reunión de sinodales, ante la Dirección de la Unidad Académica de Ingeniería en Informática, integrada por:

PRESIDENTE DEL JURADO: M.S.I. Rosa Karina González Trigueros

SINODAL: M.C. Gloria Irene Téllez Rodríguez

SINODAL: M.T.I. Ismaylia Saucedo Ugalde

Ha decidido autorizar y aceptar la digitalización de la misma por el participante, conforme a la normatividad vigente y cumpliendo con los requisitos para tal caso.

Agradeciendo la atención a la presente, le reitero a Usted mi atenta consideración y respeto.

Atentamente

Dr. Rodolfo Ostos Robles
Director del Programa Académico de Ingeniería en Informática
Universidad Politécnica de Sinaloa

SEP

UTyP

A mis padres, por su amor

y solidaridad.

Agradecimientos

A lo largo de mis estudios, han habido personas que me han apoyado incondicionalmente y sin las cuales, no sería la persona que hoy está a un paso de culminar, finalmente, sus estudios universitarios, así que a todas esas personas que estuvieron para mí, que me apoyaron y me dejaron crecer, gracias.

Para mí, esta carrera universitaria significa no estar con las personas que amo en la comodidad de mi casa, dedicarme exclusivamente a la carrera mientras personas que amo luchan día a día para que yo esté aquí, por lo que me gusta sentir que les pago bien, que aprovecho cada una de las oportunidades que los estudios me brindan para crecer como persona, para poder independizarme y algún día poder devolverles lo que me han brindado y no como un sacrificio, sino para satisfacción personal.

A fin de cuentas, es lo que esta tesina representa, la culminación de los estudios profesionales, uno de los finales del camino concernientes a mi profesión.

Y ahora, empecemos con los agradecimientos.

Quiero agradecer eternamente a mi madre, mi modelo a seguir, mi luz cuando me encuentro perdida o sin ganas de continuar, la principal razón de que me levante todos los días con buena cara y esperando lo mejor, pues quién sino ella quien me ha dado herramientas para aprender a cómo vivir mi vida, a definir mis metas y a ir con todo a la hora de realizarlas, gracias madre por estar incondicionalmente para mí, por escucharme aun cuando no entiendas de lo que estoy hablando.

A mí familia en general, porque me hacen sentir parte de ustedes, aún en la distancia, sé que nadie me apoyará jamás como lo hacen ustedes y que nadie es tan digno de mi confianza como lo son ustedes tres, son lo primero y lo que siempre amaré. Amo saber que disfrutan de mis éxitos tanto o más que yo y que signifique

tanto que ahora esté a un paso de obtener mi título universitario en algo que realmente disfruto. No pude pedir mejor familia, quizá mejores genes, pero no mejores personalidades.

También quiero agradecer a mis amigos y compañeros que hicieron que estos años de estudios fueran disfrutables y llevaderos, por hacerme amar ir a la escuela y estar ahí en esos grandes proyectos, por no hacer tediosos y homogéneos los días de la semana, pero sobre todo por enseñarme de ustedes y aprender de mí.

Finalmente, a las personas que me están apoyando con la elaboración de este, no poco importante, documento, por su tiempo y sus ganas de que todo salga sin problemas.

¡Gracias!

Resumen

Las mejoras dentro de los procesos de una empresa siempre están disponibles, independientemente del tiempo que dicha empresa tenga operando, la mejora continua existirá siempre. Aún más si tenemos en cuenta las nuevas e innovadoras tecnologías que día a día salen al mundo. Especialmente cuando hablamos de desarrollo de software, pues demasiadas tecnologías nuevas y funcionales se están dando a conocer continuamente, mientras otras esperan ser descubiertas.

Una de las ventajas de las mejoras es la reducción de recursos y el aumento de eficiencia, lo que brinda una mejor utilidad a las finanzas de una empresa, por lo que es apreciable que estas mejoras se implementen dentro de sus procesos laborales.

En este documento se expone una posible mejora encontrada dentro de los procesos con los que se rige Internet Brands, más concisamente, su vertical llamada LEGAL.

Dicha mejora consiste en la virtualización de los entornos de desarrollo, pues un programador dedica demasiado tiempo a tener que estar gestionando su sistema operativo cada vez que se cambia de proyecto, pero a su vez se busca que la tecnología señalada sea hoy en día sino la mejor, una de las mejores soluciones para la implementación de la mejora.

La finalidad de este documento, es sustentar la tecnología Docker como la estándar a la hora de virtualizar entornos de desarrollo, demostrando todas las ventajas que esta implementación podría ofrecer a la empresa y a los desarrolladores. Así como su comparación con otras tecnologías encontradas a lo largo de la investigación.

Índice

Lista de imágenes

Lista de figuras

Lista de tablas

CAPÍTULO I. ANTECEDENTES Y PLANTEAMIENTO DEL PROBLEMA

Introducción

Las herramientas con las que cuenta el ámbito de tecnologías de la información, son cada vez más y mejores, las necesidades que nuevos proyectos requieren cambian constantemente, y es el deber de una empresa aprovechar cada nueva tecnología y/o herramienta a como mejor le convenga.

Los entornos de desarrollo son muy comunes dentro de las empresas que se dedican al desarrollo de cosas, en este caso y más concretamente, al desarrollo de software, pues las máquinas tienen que tener las mismas configuraciones que los servidores principales para que las aplicaciones funcionen tan bien en desarrollo como en producción. Esto con el fin de evitar problemas y/o errores y gastar más tiempo a la hora de tener el producto final y así aumentar la productividad media de los desarrolladores.

El propósito de esta investigación es determinar si la estandarización en la implementación de utilizar entornos de desarrollo virtualizados con Docker es o no la mejor opción para llevar a cabo en los procesos de la empresa. Puesto que existen diferentes formas de virtualizar un entorno, pero se busca que sea la más óptima en cuanto a tiempo y consumo de recursos.

La propuesta surgió debido a la oportunidad de mejora encontrada dentro de la empresa y a las tecnologías que se encargan de mejorar el rendimiento a la hora de virtualizar los entornos de desarrollo. Así como la estandarización como buena práctica dentro del desarrollo de software.

Se espera que la implementación de la propuesta argumentada a través de este documento, aumente el rendimiento de producción de código, le ahorre dinero en

cuanto al producto necesario para la mejor funcionalidad del proyecto y aproveche mejor los recursos con los que cuenta la empresa, incluyendo recursos humanos.

Organismo receptor

Internet Brands, Inc.

Domicilio

Av. Tizoc 97, Cd del Sol, 45050, Zapopan, Jal.

Ubicación

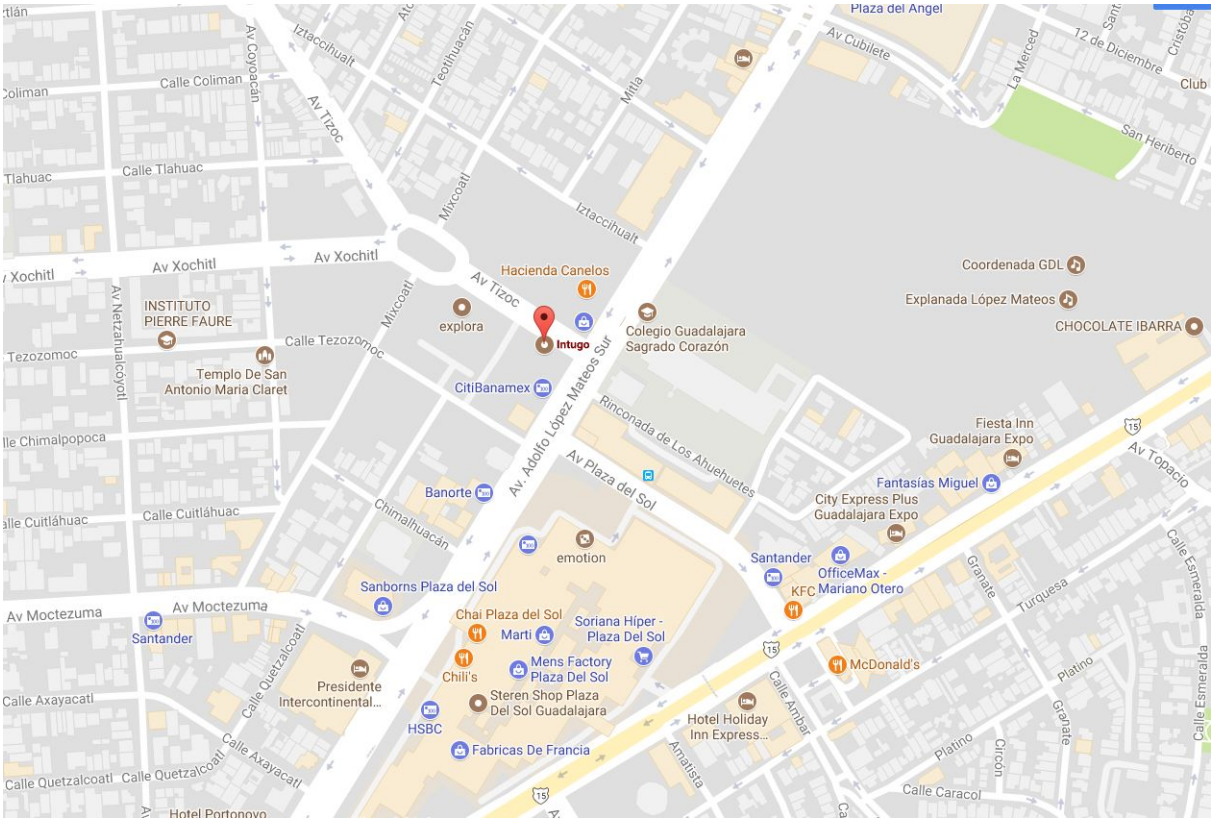

Imagen 1: Localización geográfica

Antecedentes

Desde su lanzamiento en 1998 como CarsDirect.com, Internet Brands ha crecido hasta convertirse en una organización de servicios en línea y de servicios en línea totalmente integrada, enfocada en cuatro categorías verticales de alto valor: Automoción, Salud, Legal y Hogar / Viajes.

Los galardonados sitios web de consumo de la compañía lideran sus categorías y atienden a más de 100 millones de visitantes mensuales, mientras que una amplia gama de ofertas de presencia en la web ha establecido relaciones profundas y de largo plazo con clientes SMB y empresariales.

La poderosa plataforma de propiedad de Internet Brands brinda flexibilidad y escalabilidad para impulsar el crecimiento continuo de la compañía. [1]

Hoy en día, y gracias a la empresa *Intugo* Internet Brands cuenta con equipos remotos en México, uno se encuentra en Monterrey y el otro en Guadalajara. Hace aproximadamente cinco años que el equipo remoto de la ciudad de Guadalajara se encuentra laborando y hoy ya son más de veinte desarrolladores activos los que conforman este equipo.

Actualmente, la virtualización de los entornos de desarrollo ha tenido un gran avance, pero no son muchas las empresas que los explotan en su totalidad, prefiriendo tecnologías, que si bien cumplen con su objetivo, no son tan sencillas de implementar, o consumen demasiados recursos de las máquinas.

En Internet Brands, no se tienen completamente estandarizados los procesos para la virtualización de estos ambientes, lo que es un problema, pues cada vertical cuenta con una gran cantidad de proyectos, los cuales dependen de diferentes ambientes.

El uso de tecnologías como Vagrant, que viene con soporte para muchos nuevos frameworks, es ampliamente utilizado sin tener en cuenta las desventajas que, virtualizar máquinas completas nos provocan.

Además de Vagrant, la instalación de los ambientes localmente, es una de las técnicas que utilizan para poder desarrollar dentro de un proyecto.

Ambas técnicas funcionan, pero existen otras que nos ahorran tiempo y recursos.

Planteamiento del problema

Internet Brands promueve una cultura empresarial y amistosa que aplaude la innovación y los resultados al mismo tiempo que adopta el cambio y la independencia. Sus empleados están intensamente impulsados y constantemente animados a llegar más lejos y usar la creatividad para lograr el éxito.

Su sede corporativa en El Segundo, California, está ubicada centralmente, lo que les permite formar un equipo formado por miembros de los condados de Los Ángeles, Orange y Riverside y ahora equipos remotos en Colombia, India y México. [2]

Como se menciona anteriormente, IB está compuesto por equipos remotos, lo que de complica la tarea de mantenerse actualizado en cuanto a las tecnologías que son utilizadas en cada proyecto, ellos mismos se placen en decir que es una compañía que cambia sus formas de hacer las cosas basados en la innovación de sus desarrolladores y líderes de proyecto. Por lo anterior, es decir, el cambio en sus tecnologías y la dependencia de los líderes de proyecto, en cada proyecto se trabaja de diferente forma, utilizando tecnologías completamente diferentes.

Lo anterior, perjudica principalmente al manejo de los ambientes de desarrollo, pues quienes lo manejan localmente se ven en la tarea de instalar y desinstalar herramientas muy constantemente, pero hay quienes configuran el ambiente de desarrollo en Vagrant, una buena forma de hacerlo, sin embargo, la cantidad de

recursos que consume es considerable, aumentando los requerimientos de las máquinas con las que trabajan los desarrolladores, por ende, los precios de estas.

Cabe mencionar que actualmente, algunos proyectos dentro de Internet Brands implementan docker como gestor de ambientes de desarrollo, sin embargo, existen aquellos que aún siguen con vagrant (aun cuando se consumen entre ellos y tienen al mismo líder de desarrollo), lo que complica la instalación de estos ambientes, principalmente por la falta de una estandarización.

El tiempo que le toma a un desarrollador tener listo su ambiente utilizando Vagrant es de aproximadamente dos días si todo sale como se espera, si hay complicaciones tiene que buscar soporte de las personas encargadas de ellos, las cuales, al estar ocupadas con otras cuestiones, muchas veces carecen de tiempo y alargan el tiempo de espera.

Hipótesis

Comprobar que la correcta gestión de entornos de desarrollo utilizando la herramienta de Docker, así como su estandarización dentro de la empresa, aumenta la productividad de los desarrolladores, además de reducir el costo de los equipos necesarios para la instalación de los entornos de desarrollo al disminuir los requerimientos de dichas máquinas.

Objetivos

- Proponer dentro de una vertical la estandarización en la implementación de los entornos de desarrollo utilizando docker para poder aumentar la productividad del desarrollo y reducir los requerimientos de hardware de las máquinas.

- Disminuir el tiempo de instalación de ambientes de desarrollo dentro de la vertical de LEGAL.
- Optimizar los recursos de las máquinas que se utilizan por los desarrolladores, en las cuales será montado el ambiente.
- Proponer la estandarización en la forma en la que los líderes de desarrollo dentro de la vertical de LEGAL crean los ambientes de desarrollo para cada proyecto.
- Disminuir la complejidad con la que se encuentran tanto desarrolladores como líderes de desarrollo en cuanto a la gestión de ambientes de desarrollo.

Importancia y/o justificación del estudio

El tiempo empleado por un desarrollador es muy costoso y tiende a subir dependiendo de su grado de experiencia; sin embargo, hay factores que consumen tiempo, los cuales no son controlados por el desarrollador. Tal es el caso de la instalación de un ambiente de trabajo. Instalar las herramientas necesarias, configurarlas, y dejar todo completamente funcional para poder empezar a desarrollar consume tiempo, el cual puede restar en la productividad que dicho desarrollador está teniendo, y aumentar los recursos que la empresa invierte en esa persona.

Además, cuando los requerimientos de los entornos de desarrollo consumen muchos recursos, el equipo con el que trabaja el desarrollador debe estar a la altura, y por ende ser más robusto, recordemos que las tecnologías usadas actualmente, como lo es vagrant, crea una máquina virtual dentro de otra, es decir que consume los recursos de la máquina física, lo que resta los recursos de esta última mientras el ambiente esté siendo utilizado.

Docker promete que con sus contenedores no es necesaria la instalación de otra máquina virtual, pues no reserva recursos asignados, sino que utiliza los mismos de la máquina física y los consume cuando los necesita, por lo tanto, el rendimiento del

equipo debería de mantenerse en óptimas condiciones para trabajar, aun cuando el ambiente esté corriendo.

Como una empresa innovadora y vanguardista, es importante conocer los beneficios que las nuevas tecnologías, como lo son Docker y docker-compose, pueden ofrecernos, así como la correcta implementación para su amplio aprovechamiento.

Docker y docker-compose en este caso, nos ayudan a la gestión de los ambientes de desarrollo, facilitando la forma de trabajo y minimizando los recursos que necesitamos para la estandarización de los ambientes de trabajo.

Limitaciones del estudio

La implementación de dicha tecnología se limitará a los proyectos que se gestionan mediante la vertical de LEGAL y el equipo remoto que se encuentra en el municipio de Zapopan, Jalisco. Posteriormente se evaluarán los resultados que arrojó el estudio y se analizará si es conveniente o no estandarizar esta tecnología para toda la vertical antes mencionada. Si resulta ser una buena práctica, se propondrá la utilización de esta tecnología a las verticales hermanas dentro de Internet Brands.

CAPÍTULO II. MARCO REFERENCIAL Y MARCO TEÓRICO

Introducción

Comúnmente, al desarrollar software para proyectos de diversa índole, nos encontramos con que nuestra máquina de desarrollo puede verse envuelta rápidamente en un cierto caos provocado por la coexistencia de varios lenguajes o plataformas, servidores de bases de datos locales, espacios de trabajo de nuestro IDE, proyectos de git y subversion, etc.

En otros casos, puede ocurrir que nos incorporamos a un proyecto ya existente y en el que existe un camino ya prefijado para configurar el entorno. Este camino nos puede venir marcado por distintos grados de formalidad dependiendo del estado y características del proyecto, pero puede perder precisión con el paso del tiempo, ya que vamos haciendo los cambios según van surgiendo las necesidades específicas de cada entorno a lo largo de semanas, meses o incluso años. Así, cuando hay una nueva incorporación o se retoma el proyecto después de un tiempo, configurar el entorno de una forma rápida y sencilla puede ser todo un reto.

Una de las opciones más interesante para solucionar los problemas planteados es el uso de máquinas virtuales, para de esta forma, aislar cada proyecto en un entorno independiente. El problema de esta solución es que el proceso de configuración y distribución sigue siendo una tarea manual, repetitiva y en definitiva, poco conveniente. [3]

Una variante de la idea anterior y que hemos adoptado de forma interna es un flujo de trabajo basado en la virtualización (Virtualbox y Vagrant) que simplifica de gran manera la configuración de los entornos y nos permite resolver rápidamente los problemas expuestos anteriormente.

Incluso, hoy en día existen herramientas como puphpet que nos ayudan con la

gestión de los *Vagrant Files,* lo que facilita la tarea de crear las máquinas virtuales iguales en diferentes máquinas físicas.

Por otro lado, una de las desventajas de este método es el peso de los sistemas virtuales creados, pues para un óptimo rendimiento estos entornos tienden a consumir muchos recursos de la máquina fisica pero, aun así, es menos que lo que consume una máquina virtual completa.

Así es que para dar respuesta a estos problemas nace Docker, cuyo objetivo es permitir la creación de paquetes estándar pensados para despliegue llamados "contenedores" que incluyen todo lo necesario para que una aplicación funcione (dependencias, servicios, etc…) y que se aíslan del sistema subyacente para lograr que siempre funcionen exactamente igual proporcionando optimización a los recursos consumidos. [4]

Por el momento el número de paquetes disponibles para la plataforma Windows es algo limitado a comparación de la mayoría de las distribuciones de Linux, como Ubuntu, Arch Linux, Debian, Fedora y Red Hat. Esto debido a que los desarrolladores se están centrando en hacerla lo más estable y funcional posible antes de la llegada de la versión estable. El problema con la estabilidad de Docker en Windows, es que para funcionar monta Ubuntu sobre el Sistema Operativo, por lo que al lanzar una terminal, el programa busca (y descarga) para después ejecutar el kernel del sistema operativo [5], lo que ocasiona problemas.

En la tabla 1 se puede apreciar una comparativa sobre Docker y Vagrant enfocada en la virtualización de entornos de desarrollo.

Característica	Docker	Vagrant
Tipo de virtualización	VE	VM
Recursos garantizados a nivel de hardware	No	Sí
Plataformas SO compatibles	Linux, Windows, MacOS	Linux, Windows, MacOS
El tiempo de inicio de la máquina creada	Unos segundos	Unos minutos
Nivel de aislamiento para sistemas virtuales creados	Parcial	Completo
Peso de los sistemas virtuales creados	Muy ligero	Pesado, pero aún mejor que máquina virtual completa
Otras ventajas	Rápido, fácil de aprender	Integración con herramientas de CM

Tabla 1: Comparativa Docker y Vagrant

Descripción y análisis de investigaciones relacionadas

Estado del arte

Docker

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de Virtualización a nivel de sistema operativo en Linux.

Es además, la empresa que impulsa el movimiento de contenedores y el único proveedor de plataformas de contenedores que aborda cada aplicación a través de la nube híbrida. Las empresas de hoy en día están bajo presión para transformarse digitalmente, pero están limitadas por aplicaciones e infraestructura existentes mientras racionalizan una cartera cada vez más diversa de nubes, centros de datos y arquitecturas de aplicaciones. Docker permite una verdadera independencia entre las aplicaciones y la infraestructura y los desarrolladores y las operaciones de TI para desbloquear su potencial y crea un modelo para una mejor colaboración e innovación. [6]

Historia de Docker

Salomón Hykes comenzó Docker como un proyecto interno dentro dotCloud, empresa enfocado a una plataforma como un servicio (PaaS), con las contribuciones iniciales de otros ingenieros de dotCloud, incluyendo Andrea Luzzardi

y Francois-Xavier Bourlet. Jeff Lindsay también participó como colaborador independiente. Docker representa una evolución de la tecnología patentada de dotCloud, que es a su vez construida sobre proyectos de código abierto anteriores como Cloudlets.

Docker fue liberado como código abierto en marzo de 2013. El 13 de marzo de 2014, con el lanzamiento de la versión 0.9, Docker dejó de utilizar LXC como el entorno de ejecución por defecto y lo reemplazó con su propia biblioteca, libcontainer, escrito en Go (lenguaje de programación). El 13 de abril de 2015, el proyecto tenía más de 20.700 estrellas de GitHub (haciéndolo uno de los proyectos con más estrellas de GitHub, en 20a posición), más de 4.700 bifurcaciones ("forks"), y casi 900 colaboradores.

Un análisis en mayo de 2015 mostró las siguientes organizaciones como las principales contribuyentes de Docker: Red Hat (mayores contribuyentes, aún más que el equipo de Docker en sí), el equipo de Docker, IBM, Google, Cisco Systems y Amadeus IT Group. [7]

Contenedor

Una imagen de contenedor es un paquete ligero, autónomo y ejecutable de una pieza de software que incluye todo lo necesario para ejecutarlo: código, tiempo de ejecución, herramientas del sistema, bibliotecas del sistema, configuraciones. Disponible para aplicaciones basadas en Linux, MacOS y Windows, el software en contenedor siempre funcionará igual, independientemente del entorno. Los contenedores aíslan el software de su entorno, por ejemplo, las diferencias entre los entornos de desarrollo y de prueba, y ayudan a reducir los conflictos entre los equipos que ejecutan software diferente en la misma infraestructura. [8]

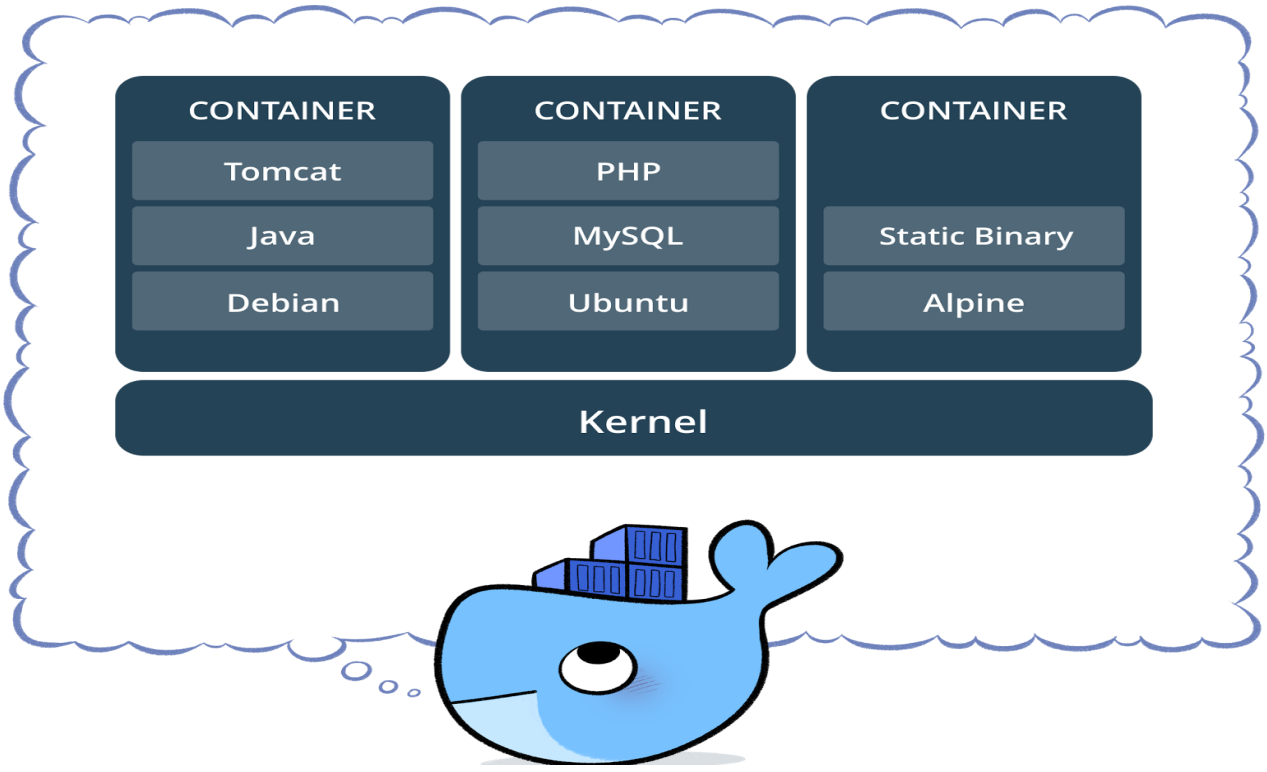

Imagen 2: Contenedores

Contenedor Linux

LXC (Linux Containers) es una tecnología de virtualización en el nivel de sistema operativo (SO) para Linux. LXC permite que un servidor físico ejecute múltiples instancias de sistemas operativos aislados, conocidos como Servidores Privados Virtuales (SPV o VPS en inglés) o Entornos Virtuales (EV). LXC no provee de una máquina virtual, más bien provee un entorno virtual que tiene su propio espacio de procesos y redes.

Es similar a otras tecnologías de virtualización en el nivel de SO como OpenVZ y Linux-VServer, asimismo se asemeja a aquellas de otros sistemas operativos como FreeBSD jail y Solaris Containers.

LXC se basa en la funcionalidad cgroups del Linux que está disponible desde la versión 2.6.29, desarrollada como parte de LXC. También se basa en otras

funcionalidades de aislamiento de espacio de nombres, que fueron desarrolladas e integradas dentro de la línea principal del núcleo de Linux. [9]

Docker Compose

Compose es una herramienta para definir y ejecutar aplicaciones Docker de contenedores múltiples. Con Compose, se usa un archivo YAML para configurar los servicios de la aplicación. Luego, con un solo comando, crea e inicia todos los servicios desde su configuración.

Compose funciona en todos los entornos: producción, desarrollo, prueba, así como flujos de trabajo de elementos de configuración. [10]

Dockerfile (imágenes de Docker)

Las imágenes de docker son el sistema de archivos que usa el proceso o procesos que se arrancan en los contenedores. [11]

```
FROM ubuntu
ENV MYPASSWORD password
# Update ubuntu
RUN apt-get update
RUN apt-get upgrade -y
# Set root password
RUN echo "root:$MYPASSWORD" | chpasswd
# Install Supervisor to start processes
RUN apt-get install -y supervisor
RUN mkdir -p /var/log/supervisor
RUN /bin/echo -e "[supervisord]\nnodaemon=true\n" > /etc/supervisord.conf
# Install OpenSSH
RUN apt-get install -y openssh-client openssh-server
RUN mkdir -p /var/run/sshd
# Setup SSHD to allow root logins
RUN sed -i 's/^PermitRootLogin.*/PermitRootLogin yes/g' /etc/ssh/sshd_config
# Tell Supervisor to start sshd
RUN /bin/echo -e "[program:sshd]\ncommand=/usr/sbin/sshd -D" >> /etc/
supervisord.conf
# Start Supervisor
CMD ["/usr/bin/supervisord"]
```

Imagen 3: Ejemplo Dockerfile

Máquina virtual

Una máquina virtual es un software que emula un ordenador justo como si fuese uno real. Todo esto sucede en una ventana dentro de tu sistema operativo actual como

cualquier otro programa que uses. La idea de este tipo de software es que puedas ejecutar sistemas operativos como si fuesen una aplicación, mientras este cree que está usando el hardware de un ordenador físico común. Cada vez que quieras usar este sistema operativo puedes abrir el software de virtualización y “encender” tu máquina. [12]

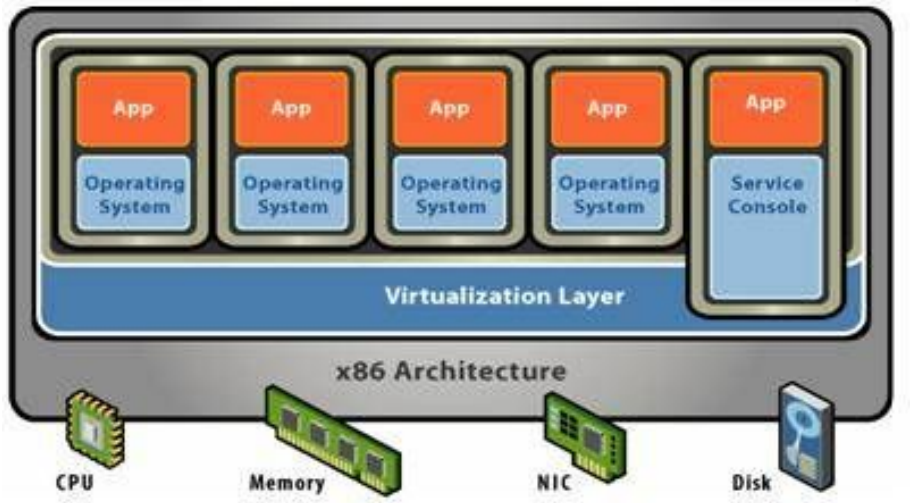

Figura 1: Máquina virtual

Vagrant

Vagrant es una herramienta para crear y administrar entornos de máquinas virtuales en un solo flujo de trabajo. Con un flujo de trabajo fácil de usar y enfoque en la automatización, Vagrant reduce el tiempo de configuración del entorno de desarrollo, aumenta la paridad de producción y hace que los "trabajos en mi máquina" excusen una reliquia del pasado.

Vagrant proporciona entornos de trabajo fáciles de configurar, reproducibles y portátiles, construidos sobre la tecnología estándar de la industria y controlados por un único flujo de trabajo consistente para ayudar a maximizar la productividad y la flexibilidad de usted y su equipo.

Para lograr su magia, Vagrant se para sobre los hombros de los gigantes. Las máquinas se aprovisionan sobre VirtualBox, VMware, AWS o cualquier otro proveedor. Luego, las herramientas de aprovisionamiento estándar de la industria, como las secuencias de comandos del shell, Chef o Puppet, pueden instalar y configurar automáticamente el software en la máquina virtual. [13] [14]

Historia de Vagrant

Vagrant se inició por primera vez como un proyecto paralelo personal de Mitchell Hashimoto en enero de 2010. La primera versión de Vagrant se lanzó en marzo de 2010. En octubre de 2010, Engine Yard declaró que iban a patrocinar el proyecto

Vagrant. La primera versión estable, Vagrant 1.0, se lanzó en marzo de 2012, exactamente dos años después de la publicación de la versión original. En noviembre de 2012, Mitchell formó una organización llamada "HashiCorp" para apoyar el desarrollo a tiempo completo de Vagrant; Vagrant siguió siendo un software de código abierto de licencia liberal. HashiCorp ahora trabaja en la creación de adiciones comerciales y proporciona soporte profesional y capacitación para Vagrant.

Vagrant originalmente estaba vinculado a VirtualBox, pero la versión 1.1 agregó soporte para otro software de virtualización como VMware y KVM, y para entornos de servidores como Amazon EC2. Vagrant está escrito en Ruby, pero puede usarse en proyectos escritos en otros lenguajes de programación como PHP, Python, Java, C # y JavaScript. Desde la versión 1.6, Vagrant admite nativamente contenedores Docker, que en algunos casos pueden servir como un sustituto para un sistema operativo completamente virtualizado. [15]

Vagrantfile

La función principal de Vagrantfile es describir el tipo de máquina requerida para un

proyecto y cómo configurar y aprovisionar estas máquinas. Los archivos Vagrant se llaman archivos Vagrant porque el nombre de archivo literal real para el archivo es Vagrantfile (la cubierta no importa a menos que su sistema de archivos se ejecute en un modo estrictamente sensible a las mayúsculas y minúsculas).

Vagrant está diseñado para ejecutarse con un Vagrantfile por proyecto, y se supone que Vagrantfile está comprometido con el control de versiones. Esto permite a otros desarrolladores involucrados en el proyecto verificar el código, ejecutar vagabundo y estar en camino. Los archivos Vagrant son portátiles en todas las plataformas que admite Vagrant.

La sintaxis de Vagrantfiles es Ruby, pero el conocimiento del lenguaje de programación Ruby no es necesario para realizar modificaciones en el Vagrantfile, ya que es en su mayoría asignación simple de variables. De hecho, Ruby ni siquiera es la comunidad más popular en la que se usa Vagrant, lo que debería ayudar a mostrarte que, a pesar de no tener conocimiento de Ruby, las personas tienen mucho éxito con Vagrant. [16]

```
Vagrant::Config.run do |config|
    config.vm.box = "hashicorp/precise64"

    #vm name
    config.vm.network : hostonly, "192.168.33.13"

    #vagrant IP
    config.vm.forward_port 80,8080
    #all request will be redirect to port 8080

    config.vm.share_folder("v-root", "/var/www", ".", :nfs => true)

    #Path to share local machine with vm
    config.vm.share_folder("v-root", "/var/www", "./files", :nfs => true)

    #virtual memory to VM
    config.vm.customize do |vm|
        vm.memory_size = 512
    end

    config.vm.provision : puppet do |puppet|
        puppet.manifests_path = "manifests"
        puppet.manifest_file = "base.pp"
    end
end
```

Imagen 4: Ejemplo Vagrantfile

Virtualbox

VirtualBox es un potente producto de virtualización x86 y AMD64 / Intel64 para uso empresarial y doméstico. VirtualBox no solo es un producto extremadamente rico en características y alto rendimiento para clientes empresariales, sino que también es la única solución profesional que está disponible libremente como software de código abierto bajo los términos de la Licencia Pública General de GNU (GPL) versión 2.

Actualmente, VirtualBox se ejecuta en servidores Windows, Linux, Macintosh y Solaris y es compatible con una gran cantidad de sistemas operativos invitados, incluidos, entre otros, Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS / Windows 3.x, Linux (2.4, 2.6, 3. xy 4.x), Solaris y

OpenSolaris, OS / 2 y OpenBSD.

VirtualBox se está desarrollando activamente con lanzamientos frecuentes y tiene una lista cada vez mayor de características, sistemas operativos invitados compatibles y plataformas en las que se ejecuta. VirtualBox es un esfuerzo de la comunidad respaldado por una empresa dedicada: se anima a todos a contribuir mientras que Oracle garantiza que el producto siempre cumpla con los criterios de calidad profesional. [17]

Virtualización

En Informática, virtualización es la creación a través de software de una versión virtual de algún recurso tecnológico, como puede ser una plataforma de hardware, un sistema operativo, un dispositivo de almacenamiento u otros recursos de red.

Dicho de otra manera, se refiere a la abstracción de los recursos de una computadora, llamada Hypervisor o VMM (Virtual Machine Monitor) que crea una capa de abstracción entre el hardware de la máquina física (host) y el sistema operativo de la máquina virtual (virtual machine, guest), dividiéndose el recurso en uno o más entornos de ejecución.

Esta capa de software (VMM) maneja, gestiona y arbitra los cuatro recursos principales de una computadora (CPU, Memoria, Dispositivos Periféricos y Conexiones de Red) y así podrá repartir dinámicamente dichos recursos entre todas las máquinas virtuales definidas en el computador central. Esto hace que se puedan tener varios ordenadores virtuales ejecutándose en el mismo ordenador físico.

Tal término es antiguo; se viene usando desde 1960, y ha sido aplicado a diferentes aspectos y ámbitos de la informática, desde sistemas computacionales completos, hasta capacidades o componentes individuales.

La virtualización se encarga de crear una interfaz externa que encapsula una

implementación subyacente mediante la combinación de recursos en localizaciones físicas diferentes, o por medio de la simplificación del sistema de control. Un avanzado desarrollo de nuevas plataformas y tecnologías de virtualización ha hecho que en los últimos años se haya vuelto a prestar atención a este concepto.

La máquina virtual en general simula una plataforma de hardware autónoma incluyendo un sistema operativo completo que se ejecuta como si estuviera instalado. Típicamente varias máquinas virtuales operan en un computador central. Para que el sistema operativo "guest" funcione, la simulación debe ser lo suficientemente grande (siempre dependiendo del tipo de virtualización).

Existen diferentes formas de virtualización: es posible virtualizar el hardware de servidor, el software de servidor, virtualizar sesiones de usuario, virtualizar aplicaciones y también se pueden crear máquinas virtuales en una computadora de escritorio.

Entre los principales proveedores de software que han desarrollado tecnologías de virtualización integrales se encuentran Citrix, VMware y Microsoft. Estas compañías han diseñado soluciones específicas para virtualización. Si bien la virtualización no es un invento reciente, con la consolidación del modelo de la Computación en la nube, la virtualización ha pasado a ser uno de los componentes fundamentales, especialmente en lo que se denomina infraestructura de nube privada. [18]

Diferencias entre virtualizar un Sistema operativo e instalarlo

Virtualizar el sistema operativo es una opción interesante si no queremos instalar dos sistemas operativos en el mismo ordenador, pero si por el contrario lo que hacemos es instalarlo, todos los sistemas operativos que tengamos instalados funcionarán de la misma manera que si estuvieran instalados en distintos ordenadores.

El único y pequeño inconveniente es que necesitamos un gestor de arranque que al encender nuestro ordenador nos dé la opción de elegir qué sistema operativo queremos utilizar, lo que conlleva que si por ejemplo estamos en Windows y queremos cambiar a GNU/Linux deberíamos reiniciar nuestro ordenador. La virtualización por el contrario permite cambiar de sistema operativo como si se tratase de cualquier otro programa, sin embargo, esta agilidad tiene la desventaja de que un sistema operativo virtualizado no es tan potente como uno que ya estuviera instalado. [19]

Retos de la Virtualización

- Índices de utilización más altos como — Antes de la virtualización, los índices de utilización del servidor y almacenamiento en los centros de datos de la empresa rondaban menos del 50% (de hecho, del 10% al 15% de los índices de utilización fueron los más comunes). A través de la virtualización, las cargas de trabajo pueden ser encapsuladas y transferidas a los sistemas inactivos o sin uso — lo cual significa que los sistemas existentes pueden ser consolidados, así que las compras de capacidad adicional del servidor pueden ser retrasadas o evitadas. [20]

- Consolidación de Recursos — La virtualización permite la consolidación de múltiples recursos de TI. Más allá de la consolidación de almacenamiento, la virtualización proporciona una oportunidad para consolidar la arquitectura de sistemas, infraestructura de aplicación, datos y base de datos, interfaces, redes, escritorios, e incluso procesos de negocios, resultando en ahorros de costo y mayor eficiencia. [20]

- Uso/costo menor energía — La electricidad requerida para que funcionen los centros de datos de clase empresarial ya no está disponible en suministros ilimitados, y el costo está en una espiral ascendente. Por cada dólar gastado

en un servidor hardware, un dólar adicional es gastado en energía (incluyendo el costo de los servidores en función y los enfriadores). Utilizando virtualización para consolidar hace posible cortar el consumo total de energía y ahorrar dinero de una manera significativa. [20]

- Ahorros de espacio — La extensión del servidor permanece como un serio problema en la mayoría de los centros de datos empresariales, pero la expansión del centro de datos no es siempre una opción, con los costos de construcción promediando miles de dólares por pie cuadrado. La virtualización puede aliviar la tensión mediante la consolidación de muchos sistemas virtuales en menos sistemas físicos. [20]

- Recuperación de desastre/continuidad del negocio — La virtualización puede incrementar la disponibilidad de los índices del nivel de servicio en general y proporcionar nuevas opciones de soluciones para la recuperación de desastre. [20]

- Costos de operación reducidos — La empresa promedio gasta $8 dólares en mantenimiento por cada $1 dólar invertido en nueva infraestructura. La virtualización puede cambiar el radio de servicio-a administración reducir la carga total de trabajo administrativo, y cortar el total de costos de operación. [20]

Ventajas de la virtualización

- Reutilización de hardware existente (para utilizar software más moderno) y optimizar el aprovechamiento de todos los recursos de hardware. [21]

- Rápida incorporación de nuevos recursos para los servidores virtualizados. [21]

- Reducción de los costes de espacio y consumo necesario de forma proporcional al índice de consolidación logrado (Estimación media 10:1). [21]

- Administración global centralizada y simplificada. [21]
- Nos permite gestionar nuestro CPD como un pool de recursos o agrupación de toda la capacidad de procesamiento, memoria, red y almacenamiento disponible en nuestra infraestructura. [21]
- Mejora en los procesos de clonación y copia de sistemas: Mayor facilidad para la creación de entornos de test que permiten poner en marcha nuevas aplicaciones sin impactar a la producción, agilizando el proceso de las pruebas. [21]
- Aislamiento: un fallo general de sistema de una máquina virtual no afecta al resto de máquinas virtuales. [21]
- No sólo aporta el beneficio directo en la reducción del hardware necesario, sino también los costes asociados. [21]
- Reduce los tiempos de parada. [21]
- Migración en caliente de máquinas virtuales (sin pérdida de servicio) de un servidor físico a otro, eliminando la necesidad de paradas planificadas por mantenimiento de los servidores físicos. [21]
- Balanceo dinámico de máquinas virtuales entre los servidores físicos que componen el pool de recursos, garantizando que cada máquina virtual ejecute en el servidor físico más adecuado y proporcionando un consumo de recursos homogéneo y óptimo en toda la infraestructura. [21]
- Contribución al medio ambiente -Green IT- por menor consumo de energía en servidores físicos. [21]
- Alta disponibilidad. [21]

¿Es la tecnología de Docker sólo virtualización?

Sí y no. Aquí hay una manera fácil de pensar sobre los dos:

- La virtualización permite que muchos sistemas operativos se ejecuten simultáneamente en un solo sistema.
- Los contenedores comparten el mismo kernel del sistema operativo y aíslan los procesos de la aplicación del resto del sistema. [22]

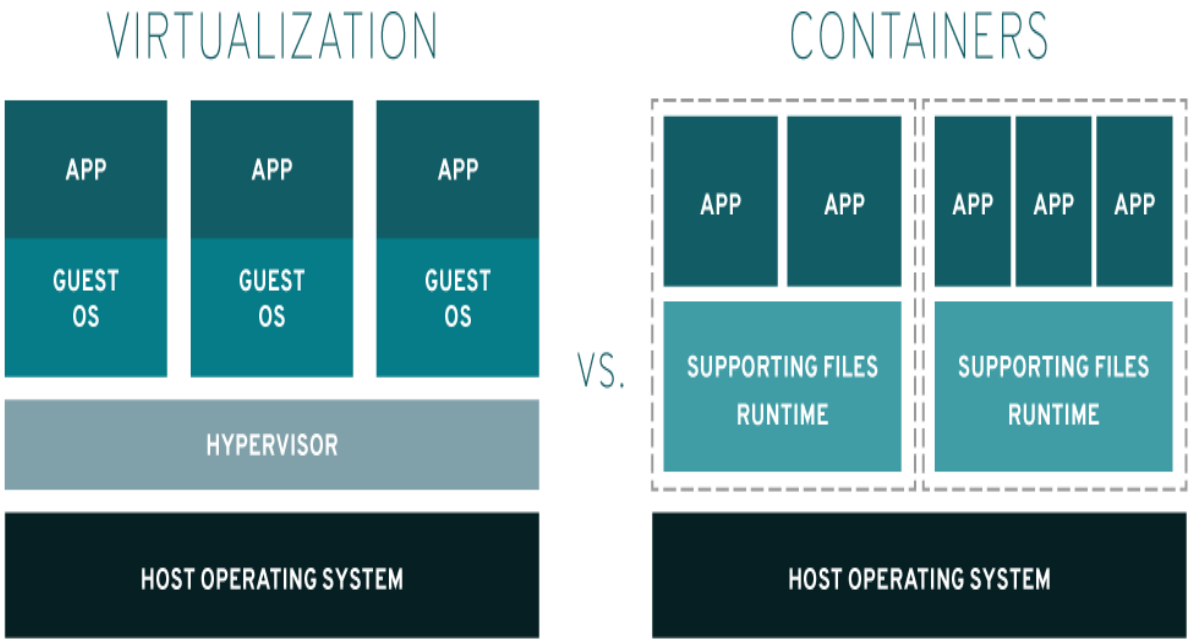

Figura 2: Virtualización vs. Contenedores

Software y servicios

Linux

GNU/Linux, también conocido como Linux, es un sistema operativo libre tipo Unix; multiplataforma, multi-usuario y multi-tarea. El sistema es la combinación de varios proyectos, entre los cuales destacan GNU (encabezado por Richard Stallman y la Free Software Foundation) y el núcleo Linux (encabezado por Linus Torvalds). Su desarrollo es uno de los ejemplos más prominentes de software libre: todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera, bajo los términos de la GPL (Licencia Pública General de GNU) y otra serie de licencias libres.

Imagen 5: Logo de Linux

A pesar de que «Linux» denomina en la jerga cotidiana al sistema operativo, éste es en realidad solo el kernel (núcleo) del sistema. La idea tiene origen en el proyecto GNU a mediados de la década de 1980, así como una gran cantidad del resto de los componentes que van desde los compiladores de GNU hasta entornos de escritorio. Sin embargo, tras la aparición de Linux en la década de 1990 una parte significativa de los medios generales y especializados han utilizado el término Linux para referirse al todo. Esto ha sido motivo de polémicas.

Además, existen derivados de Linux que no tienen componentes GNU (por ejemplo Android), y distribuciones de GNU donde Linux está ausente (por ejemplo Debian GNU/Hurd).

A GNU/Linux se le encuentra normalmente en forma de compendios conocidos como distribuciones o distros, a las cuales se les han adicionado selecciones de aplicaciones y programas para descargar e instalar las mismas. El propósito de una

distribución es ofrecer un producto final que el usuario pueda instalar, así como cumplir con las necesidades de un determinado grupo de usuarios.

Algunas de ellas son especialmente conocidas por su uso en servidores y supercomputadoras, donde GNU/Linux tiene la cuota más importante del mercado. Según un informe de IDC, GNU/Linux es utilizado por el 78% de los principales 500 servidores del mundo. Otro informe le da una cuota de mercado de 89% en los 500 mayores superordenadores. Con menor participación, el sistema GNU/Linux también se usa en el segmento de las computadoras de escritorio, portátiles, computadoras de bolsillo, teléfonos móviles, sistemas embebidos, videoconsolas y otros dispositivos. [23]

Ubuntu

Ubuntu es una distribución del sistema operativo GNU/Linux y que se distribuye como software libre, la cual durante un tiempo incluyó su propio entorno de escritorio denominado Unity, actualmente utiliza GNOME, como en sus orígenes. Su nombre proviene de la ética homónima, en la que se habla de la existencia de uno mismo como cooperación de los demás.

Está orientado al usuario promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia del usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto. Estadísticas web sugieren que la cuota de mercado de Ubuntu dentro de las distribuciones Linux es, aproximadamente, del 49%, y con una tendencia a aumentar como servidor web.

Su patrocinador, Canonical, es una compañía británica propiedad del empresario sudafricano Mark Shuttleworth. Ofrece el sistema de manera gratuita, y se financia por medio de servicios vinculados al sistema operativo y vendiendo soporte técnico. Además, al mantenerlo libre y gratuito, la empresa es capaz de aprovechar los desarrolladores de la comunidad para mejorar los componentes de su sistema

operativo. Extraoficialmente, la comunidad de desarrolladores proporciona soporte para otras derivaciones de Ubuntu, con otros entornos gráficos, como Kubuntu, Xubuntu, Ubuntu MATE, Edubuntu, Ubuntu Studio, Mythbuntu y Ubuntu GNOME.

Canonical, además de mantener Ubuntu, también provee una versión orientada a servidores, Ubuntu Server, una versión para empresas, Ubuntu Business Desktop Remix, una para televisores, Ubuntu TV, otra versión para tabletas Ubuntu Tablet, también Ubuntu Phone y una para usar el escritorio desde teléfonos inteligentes, Ubuntu for Android.

Cada seis meses se publica una nueva versión de Ubuntu. Esta recibe soporte por parte de Canonical durante nueve meses por medio de actualizaciones de seguridad, parches para bugs críticos y actualizaciones menores de programas. Las versiones LTS (Long Term Support), que se liberan cada dos años, reciben soporte durante cinco años en los sistemas de escritorio y de servidor. [24]

Debian

El Proyecto Debian es una asociación de individuos que han hecho una causa común para crear un sistema operativo libre. Este sistema operativo se llama Debian.

Un sistema operativo es el conjunto de programas básicos y utilidades que hacen funcionar su computadora. En el núcleo de un sistema operativo está el kernel. El kernel es el programa más fundamental en la computadora y hace todo el mantenimiento básico y le permite iniciar otros programas.

Los sistemas Debian actualmente usan el kernel de Linux o el núcleo de FreeBSD. Linux es una pieza de software iniciada por Linus Torvalds y respaldada por miles de programadores de todo el mundo. FreeBSD es un sistema operativo que incluye un kernel y otro software.

Sin embargo, se está trabajando para proporcionar Debian para otros kernels, principalmente para Hurd. Hurd es una colección de servidores que se ejecuta en la parte superior de un microkernel (como Mach) para implementar diferentes funciones. The Hurd es un software gratuito producido por el proyecto GNU.

Una gran parte de las herramientas básicas que completan el sistema operativo provienen del proyecto GNU; de ahí los nombres: GNU / Linux, GNU / kFreeBSD y GNU / Hurd. Estas herramientas también son gratuitas.

Por supuesto, lo que la gente quiere es software de aplicación: programas para ayudarlos a lograr lo que quieren hacer, desde editar documentos hasta llevar una empresa, jugar juegos o escribir más software. Debian incluye más de 51,000 paquetes (software precompilado que se incluye en un formato agradable para una fácil instalación en su máquina), un administrador de paquetes (APT) y otras utilidades que hacen posible administrar miles de paquetes en miles de computadoras con la misma facilidad como instalar una sola aplicación. Todo gratis

Es un poco como una torre. En la base está el kernel. Además de eso, están todas las herramientas básicas. Lo siguiente es todo el software que ejecuta en la computadora. En la parte superior de la torre se encuentra Debian, organizando y ajustando cuidadosamente todo para que todo funcione en conjunto.

Debian se ejecuta en casi todas las computadoras personales, incluida la mayoría de los modelos más antiguos. Cada nueva versión de Debian generalmente admite una mayor cantidad de arquitecturas de computadora. [25]

CentOS

CentOS (Community ENTerprise Operating System) es una bifurcación a nivel binario de la distribución Linux Red Hat Enterprise Linux RHEL, compilado por

voluntarios a partir del código fuente publicado por Red Hat, siendo la principal diferencia con este la remoción de todas las referencias a las marcas y logos propiedad de Red Hat.

Es un sistema operativo de código abierto, basado en la distribución Red Hat Enterprise Linux, operándose de manera similar, y cuyo objetivo es ofrecer al usuario un software de "clase empresarial" gratuito. Se define como robusto, estable y fácil de instalar y utilizar. Desde la versión 5, cada lanzamiento recibe soporte durante diez años, por lo que la actual versión 7 recibirá actualizaciones de seguridad hasta el 30 de junio de 2024.

La distribución de CentOS Linux es una plataforma estable, predecible, manejable y reproducible derivada de las fuentes de Red Hat Enterprise Linux (RHEL). Ahora estamos buscando expandirnos al crear los recursos necesarios para que otras comunidades se unan y puedan construir sobre la plataforma CentOS Linux. Y hoy comenzamos el proceso ofreciendo un modelo de gobernanza claro, mayor transparencia y acceso. En las próximas semanas, nuestro objetivo es publicar nuestro propio mapa de ruta que incluya variantes del núcleo de CentOS Linux.

Desde marzo de 2004, CentOS Linux ha sido una distribución respaldada por la comunidad derivada de fuentes proporcionadas libremente al público por Red Hat. Como tal, CentOS Linux pretende ser funcionalmente compatible con RHEL. Principalmente cambiamos los paquetes para eliminar la marca y la obra de arte de los proveedores en la etapa inicial. CentOS Linux no tiene costo y se puede redistribuir libremente.

CentOS Linux está desarrollado por un pequeño pero creciente equipo de desarrolladores principales. A su vez, los desarrolladores principales cuentan con el respaldo de una comunidad de usuarios activa que incluye administradores de sistemas, administradores de redes, administradores, colaboradores clave de Linux y entusiastas de Linux de todo el mundo.

Durante el próximo año, el Proyecto CentOS ampliará su misión de establecer

CentOS Linux como una plataforma comunitaria líder para tecnologías emergentes de código abierto provenientes de otros proyectos como OpenStack. Estas tecnologías estarán en el centro de múltiples variaciones de CentOS, como descargas individuales o acceso desde un instalador personalizado. Lea más sobre las variantes y grupos de interés especial que las producen. [26]

Red Hat

Red Hat es una distribución Linux creada por Red Hat, que llegó a ser una de las más populares en los entornos de usuarios domésticos hasta el 22 de septiembre de 2003 cuando los proyectos Fedora y Red Hat se fusionaron.

La versión 1.0 fue presentada el 3 de noviembre de 1994. Y aunque no es tan antigua como la legendaria distribución Slackware, sí que ostenta el título de una de las más clásicas y robustas.

Fue la primera distribución en usar RPM como su formato de paquete, y fue la que sirvió de punto de partida para otras distribuciones, tales como la orientada hacia PC de escritorio Mandrake Linux (originalmente Red Hat Linux con KDE), Yellow Dog Linux, la cual se inició desde Red Hat Linux con soporte para PowerPC, y ASPLinux (Red Hat Linux con mejor soporte para caracteres no-Latinos).

Desde el 2003, Red Hat ha desplazado su enfoque hacia el mercado de los negocios con la distribución Red Hat Enterprise Linux y la versión no comercial Fedora Core. Red Hat Linux, la versión final, llegó oficialmente al final de su vida útil el 30 de abril de 2004, aunque el proyecto Fedora Legacy continuó publicando actualizaciones, hasta ser abandonado dicho proyecto a finales del año 2006.

Red Hat Software Inc. fue fundada en el año 1994 por Bob Young y Marc Ewing. En agosto de 1999, Red Hat salió a bolsa y sus acciones se situaron en octava posición en ganancias en Wall Street. Cuatro años más tarde, el valor de las acciones de Red Hat llegó entorno a una centésima parte del máximo valor que llegara a

alcanzar antes de la crisis de las puntocom. Aun así, sus comienzos exitosos en el mercado de valores sirvieron para que Red Hat fuera portada en periódicos y revistas no directamente relacionadas con temas informáticos.

En noviembre de 1999 la compañía adquirió Cygnus Solutions, una empresa fundada una década antes y que ya había demostrado cómo con una estrategia integral basada en software libre se puede ganar dinero.

En septiembre de 2003, Red Hat decidió concentrar sus esfuerzos de desarrollo en la versión corporativa de su distribución, Red Hat Enterprise Linux y delegó la versión común a Fedora Core, un proyecto abierto independiente de Red Hat.

El logo de Red Hat muestra un sombrero rojo portado por un hombre misterioso, destacado sobre el resto de la imagen en blanco y negro. [27]

Base de datos

Se define una base de datos como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada o estructurada.

Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una

parte de la información sobre cada elemento que queramos guardar en la tabla, cada fila de la tabla conforma un registro. [28]

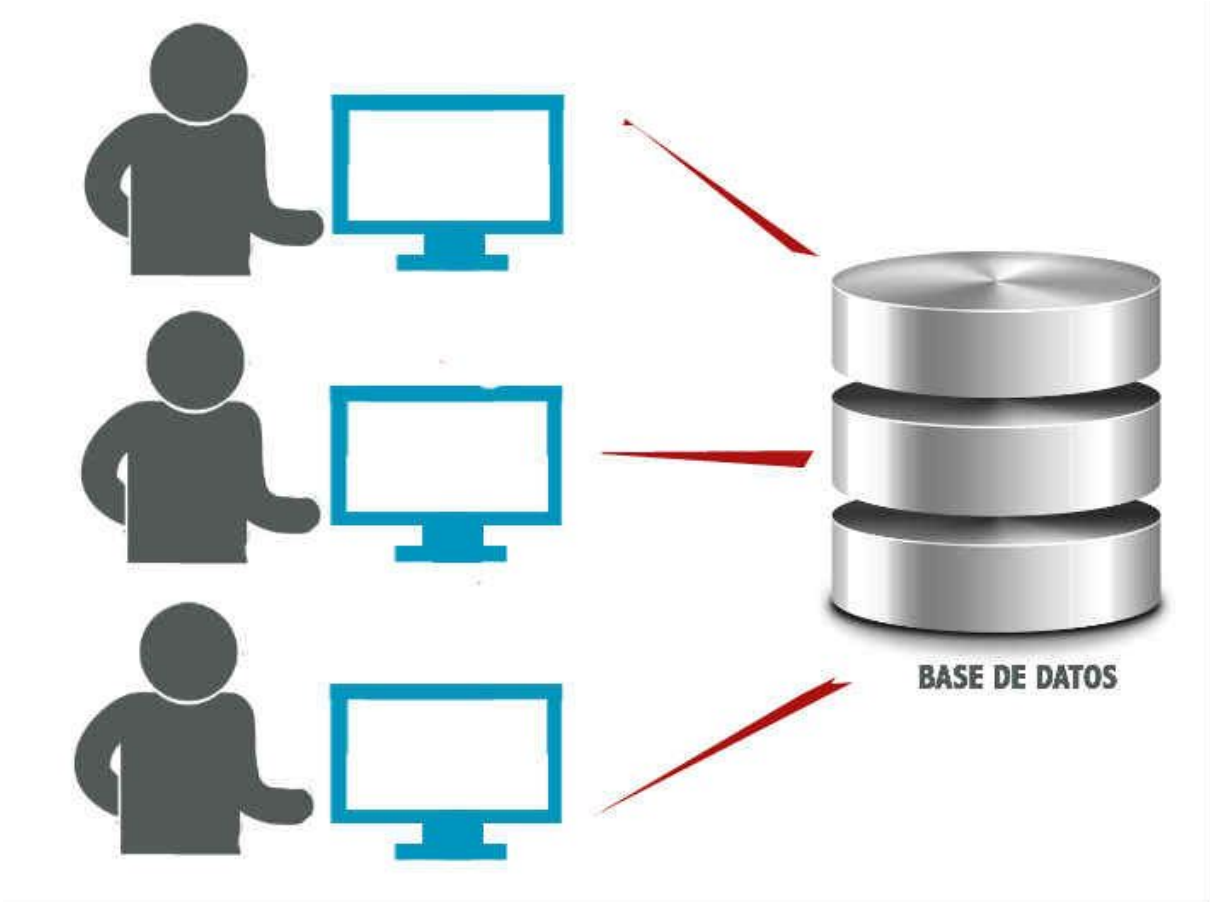

Imagen 6: Representación de una base de datos

MySQL

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation y está considerada como la base datos open source más popular del mundo y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web.

MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David

Axmark, Allan Larsson y Michael Widenius). MySQL A.B. fue adquirida por Sun Microsystems en 2008, y ésta a su vez fue comprada por Oracle Corporation en 2010, la cual ya era dueña desde 2005 de Innobase Oy, empresa finlandesa desarrolladora del motor InnoDB para MySQL.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de doble licenciamiento anteriormente mencionado. La base de datos se distribuye en varias versiones, una Community, distribuida bajo la Licencia pública general de GNU versión 2 y varias versiones Enterprise, para aquellas empresas que quieran incorporarlo en productos privativos. Las versiones Enterprise incluyen productos o servicios adicionales tales como herramientas de monitorización y soporte oficial. En 2009 se creó un fork denominado MariaDB por algunos desarrolladores (incluido algunos desarrolladores originales de MySQL) descontentos con el modelo de desarrollo y el hecho de que una misma empresa controle a la vez los productos MySQL y Oracle Database.

Está desarrollado en su mayor parte en ANSI C y C++. Tradicionalmente se considera uno de los cuatro componentes de la pila de desarrollo LAMP y WAMP.

MySQL es usado por muchos sitios web grandes y populares, como Wikipedia, Google (aunque no para búsquedas), Facebook, Twitter, Flickr y YouTube. [29]

Percona

Percona Server para MySQL® es un reemplazo gratuito, completamente compatible, mejorado, de código abierto y de reemplazo directo para MySQL que brinda rendimiento, escalabilidad e instrumentación superiores. Con más de 3.000.000 de descargas, Percona Server para los algoritmos de autoajuste de MySQL y la

compatibilidad con hardware de alto rendimiento ofrece un rendimiento y una fiabilidad excelentes.

Percona Server para MySQL cuenta con la confianza de miles de empresas para proporcionar un mejor rendimiento y concurrencia para sus cargas de trabajo más exigentes que otros servidores MySQL, y ofrece un mayor valor a los usuarios del servidor MySQL con rendimiento optimizado, mayor escalabilidad y disponibilidad, copias de seguridad mejoradas y mayor visibilidad.

Percona Server para MySQL ya está disponible en Debian 9 (stretch). El soporte solo cubre la arquitectura amd64. [30]

PostgreSQL

PostgreSQL es un poderoso sistema de base de datos relacional de objetos de código abierto. Tiene más de 15 años de desarrollo activo y una arquitectura comprobada que le ha valido una sólida reputación de fiabilidad, integridad de datos y corrección. Se ejecuta en todos los principales sistemas operativos, incluidos Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, macOS, Solaris, Tru64) y Windows. Es totalmente compatible con ACID, tiene soporte completo para claves externas, combinaciones, vistas, disparadores y procedimientos almacenados (en múltiples idiomas). Incluye la mayoría de los tipos de datos SQL: 2008, incluidos INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL y TIMESTAMP.

También es compatible con el almacenamiento de objetos grandes binarios, incluyendo imágenes, sonidos o video. Tiene interfaces de programación nativas para C / C ++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros, y documentación excepcional.

Una base de datos de clase empresarial, PostgreSQL cuenta con funciones sofisticadas como Multi-Version Concurrency Control (MVCC), recuperación puntual, tablespaces, replicación asincrónica, transacciones anidadas (puntos de guardado),

copias de seguridad en línea / calientes, un sofisticado planificador / optimizador de consultas y escritura registro anticipado para la tolerancia a fallas. Admite conjuntos de caracteres internacionales, codificaciones de caracteres multibyte, Unicode, y es compatible con la configuración regional para la clasificación, la distinción entre mayúsculas y minúsculas y el formateo.

Es altamente escalable tanto en la gran cantidad de datos que puede administrar como en la cantidad de usuarios simultáneos que puede acomodar. Hay sistemas PostgreSQL activos en entornos de producción que administran más de 4 terabytes de datos. [31]

MariaDB

MariaDB Server es uno de los servidores de bases de datos más populares del mundo. Está hecho por los desarrolladores originales de MySQL y está garantizado para ser de código abierto. Los usuarios notables incluyen Wikipedia, WordPress.com y Google.

MariaDB convierte los datos en información estructurada en una amplia gama de aplicaciones, desde bancos hasta sitios web. Es un reemplazo mejorado y de reemplazo directo para MySQL. MariaDB se usa porque es rápido, escalable y robusto, con un rico ecosistema de motores de almacenamiento, complementos y muchas otras herramientas que lo hacen muy versátil para una amplia variedad de casos de uso.

MariaDB se desarrolla como software de código abierto y como base de datos relacional, proporciona una interfaz SQL para acceder a los datos. Las últimas versiones de MariaDB también incluyen características GIS y JSON. [32]

MongoDB

MongoDB (que proviene de «humongous») es la base de datos NoSQL líder y permite a las empresas ser más ágiles y escalables. Organizaciones de todos los tamaños están usando MongoDB para crear nuevos tipos de aplicaciones, mejorar la experiencia del cliente, acelerar el tiempo de comercialización y reducir costes.

Es una base de datos ágil que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan, proporcionando siempre la funcionalidad que los desarrolladores esperan de las bases de datos tradicionales, tales como índices secundarios, un lenguaje completo de búsquedas y consistencia estricta.

MongoDB ha sido creado para brindar escalabilidad, rendimiento y gran disponibilidad, escalando de una implantación de servidor único a grandes arquitecturas complejas de centros multidatos. MongoDB brinda un elevado rendimiento, tanto para lectura como para escritura, potenciando la computación en memoria (in-memory). La replicación nativa de MongoDB y la tolerancia a fallos automática ofrece fiabilidad a nivel empresarial y flexibilidad operativa.

Las suscripciones de MongoDB ofrecen un servicio de asistencia técnica profesional, licencias comerciales y acceso a características de software de MongoDB Enterprise.

Las suscripciones no solo ayudan a los clientes a lograr una infraestructura de TI estable, escalable y segura, sino también a alcanzar sus objetivos empresariales más amplios, tales como reducir los costes, acelerar el tiempo de comercialización y disminuir los riesgos.

MongoDB Enterprise ofrece seguridad avanzada, monitorización on-premises, soporte SNMP, certificaciones de SO y mucho más. El servicio de gestión de MongoDB (MMS) ofrece funcionalidad de monitorización y respaldo en la nube o bien on-premises como parte de MongoDB Enterprise. [33]

Redis

Redis es un almacén de estructura de datos en memoria de código abierto (con licencia de BSD), que se utiliza como base de datos, caché y agente de mensajes. Admite estructuras de datos tales como cadenas, hashes, listas, conjuntos, conjuntos ordenados con consultas de rango, mapas de bits, hiperlogálogos e índices geoespaciales con consultas radiales. Redis tiene una replicación incorporada, secuencias de comandos Lua, desalojo LRU, transacciones y diferentes niveles de persistencia en disco y proporciona alta disponibilidad a través de Redis Sentinel y particiones automáticas con Redis Cluster.

Puede ejecutar operaciones atómicas en estos tipos, como agregar a una cadena; incrementando el valor en un hash; empujando un elemento a una lista; computación establecer intersección, unión y diferencia; o conseguir el miembro con la clasificación más alta en un conjunto ordenado.

Para lograr su rendimiento sobresaliente, Redis trabaja con un conjunto de datos en memoria. Dependiendo de su caso de uso, puede persistir ya sea volcando el conjunto de datos en el disco de vez en cuando, o agregando cada comando a un registro. La persistencia se puede inhabilitar opcionalmente, si solo necesita una memoria caché en la memoria, rica en funciones y en red.

Redis también es compatible con la replicación asincrónica maestro-esclavo trivial a la configuración, con primera sincronización rápida sin bloqueo, autoconexión con resincronización parcial en la división de red. [34]

Servidor web

Un servidor Web es un programa que utiliza el protocolo de transferencia de hipertexto, HTTP (Hypertext Transfer Protocol), para servir los archivos que forman páginas Web a los usuarios, en respuesta a sus solicitudes, que son reenviados por los clientes HTTP de sus computadoras. Las computadoras y los dispositivos dedicados también pueden denominarse servidores Web.

El proceso es un ejemplo del modelo cliente/servidor. Todos los equipos que alojan sitios Web deben tener programas de servidor Web. Los principales servidores Web incluyen Apache (el servidor Web más ampliamente instalado), Internet Information Server (IIS) de Microsoft y nginx (que se pronuncia engine X) de NGNIX. Otros servidores Web incluyen el servidor NetWare de Novell, el servidor Web de Google (GWS) y la familia de servidores Domino de IBM.

Los servidores Web a menudo forman parte de un paquete más amplio de programas relacionados con internet e intranet para servir correo electrónico, descargar solicitudes de archivos de protocolo de transferencia de archivos (FTP) y crear y publicar páginas Web. Las consideraciones al elegir un servidor Web incluyen cuán bien funciona con el sistema operativo y otros servidores, su capacidad para manejar la programación del servidor, las características de seguridad y las herramientas particulares de publicación, motor de búsqueda y creación de sitios que vienen con él.

En la imagen 7 se puede apreciar el diagrama de cuál es el proceso que opera un servidor web o servidor HTTP. [35]

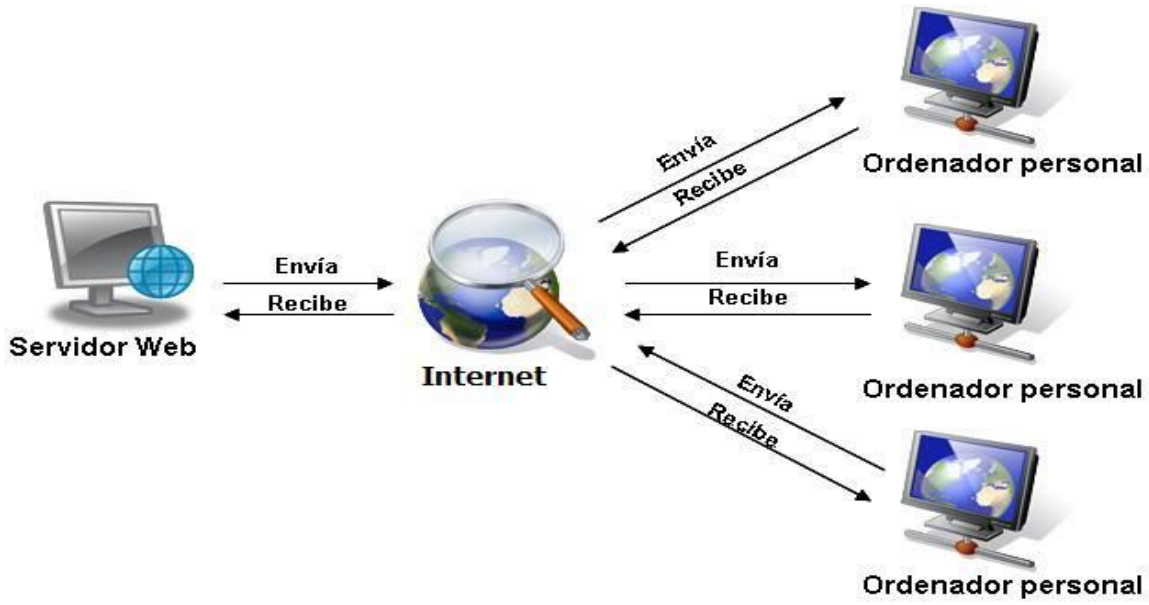

Imagen 7: Servidor web

Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1, y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que alguien quería que tuviese la connotación de algo que es firme y enérgico, pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet. Además, Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. En inglés, a patchy server (un servidor "parcheado") suena igual que Apache Server.

El servidor Apache es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server (httpd).

Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Jugó un papel fundamental en el desarrollo fundamental de la World Wide Web y alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo, ha sufrido un descenso en su cuota de mercado en los últimos años. (Estadísticas históricas y de uso diario proporcionadas por Netcraft3). En 2009 se convirtió en el primer servidor web que alojó más de 100 millones de sitios web.

La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, algunas se pueden accionar remotamente en ciertas situaciones, o explotar por los usuarios locales malévolos en las disposiciones de recibimiento compartidas que utilizan PHP como módulo de Apache. [36]

Sumario

Existen varias formas de virtualizar ambientes de desarrollo, las más populares son las máquinas virtuales, ahora gestionadas con vagrant, y los contenedores de Linux, gestionados con Docker. Ambas funcionan perfectamente para lo que están diseñadas, aun cuando Docker es una tecnología más nueva, pero no por ellos más o menos estable que las máquinas virtuales.

La virtualización genera grandes ventajas cuando se trabaja en equipo, pues si se hace de forma adecuada, o compartiendo los archivos de configuración (vagrantfile o dockerfile) se puede garantizar que los ambientes de desarrollo sean similares, lo que ahorra errores a la hora de compartir código y funcionamiento.

Una de las principales diferencias entre Docker y las máquinas virtuales es que las máquinas virtuales están sobre sistemas operativos virtuales, es decir, el sistema operativo se instala, mientras que los contenedores de Docker operan sobre el kernel del sistema operativo de la máquina.

Sin embargo, al hablar de servicios, son instalables en ambas tecnologías. Los más comunes, y para los fines de esta investigación, son las bases de datos, y los servidores HTTP.

Cuando hablamos de bases de datos, lo más común es que se nos vengan a la mente las bases de datos relacionales, aun cuando existen diversos tipos, uno de ellos son las bases de datos NoSQL y orientada a documentos, como lo es MongoDB. Y como base de datos relacionales y SQL, tenemos MySQL, la cual es una de las más populares, así como Percona, MariaDB y PostgreSQL, todas forks de MySQL que prometen ser mejores y tener mejor rendimiento.

Finalmente, al hablar de servidores web o HTTP, el más popular (y usado en la empresa) es Apache, este servidor web es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server.

CAPÍTULO III. METODOLOGÍA

Sujetos

Debido al hecho de que todos los equipos de cómputo dentro de Internet Brands Guadalajara cuentan con las mismas características, la investigación sobre el mejor rendimiento entre las tecnologías existentes y más comunes para levantar entornos de desarrollo se llevará a cabo en la máquina número 37 dentro de estas instalaciones.

Características

Modelo: DELL OptiPlex 3050 SFF Factor de forma pequeño.

Procesador: Séptima generación del Procesador Intel® Core™ i5-7500 (6MB/4T/3.4GHz).

Memoria RAM: 16GB de Memoria DDR4 a 2400MHz.

Sistema Operativo: Linux Ubuntu 16.4 LTS.

Almacenamiento: Samsung SSD 250GB.

Tarjeta de video: Gráficos Integrados Intel®.

Unidad óptica: Unidad de 8x, 9.5mm, con Bandeja de carga automática (DVD +/- RW), lectura y escritura de CD/DVD.

Puertos:

- 4 USB 3.0 (2 frontales/2 traseros)
- 4 USB 2.0 (2 frontales/2 traseros)
- 2 USB 2.0 internos
- 1 RJ-45
- 1 Display Port 1.2

- 1 puerto HDMI 1.4
- 1 UAJ
- 1 línea de salida
- 1 VGA
- 1 puerto serial + PS/2

Ranuras:

- 1 conector M.2 (tarjeta Wi-Fi)
- 1 PCIe x16 de altura media
- 1 PCIe x1 de altura media

Dimensiones:

- Altura: 29 cm (11,4") x Ancho: 9,26 cm (3,7") x Profundidad: 29,2 cm (11,5")
- Peso: 5,14 kg (11,31 lb)

Conexiones inalámbricas: Sin Tarjeta Inalámbrica.

Alimentación:

- Unidad de Fuente de Alimentación interna (de 180 W para el factor de forma pequeño y 240 W para torres); 80 PLUS.
- EPA Bronze y EPA Platinum.
- Cumple con la norma ENERGY STAR.

Imagen 8: DELL OptiPlex 3050

Material

Proyecto

Affiliates.

Software

Sistema operativo usado en los servidores de la empresa: Ubuntu 16.4 LTS

Docker

Docker Compose

Vagrant

Virtual Box

OpenStack para benchmarks.

Procedimientos

1. Detección de oportunidad de mejoramiento dentro de la empresa.

Se realizó un análisis a los procedimientos de la empresa para detectar posibles mejoras a realizar, determinando cuál sería la que tuviera mayor repercusión en cuanto a los recursos que opera la empresa.

2. Investigación de las tecnologías para la implementación de la mejora.

Después de detectar la mejor mejora, se llevó a cabo una investigación sobre las posibles formas en las que la mejora se podía llevar a cabo, detectando todas las posibles soluciones al mismo problema.

3. Elección de una tecnología.

Después de evaluar (sin pruebas) las tecnologías que se pueden utilizar para la implementación de la mejora encontrada en el punto 1, se eligió una de ellas para la realización de la investigación.

4. Comparación conceptual de la tecnología seleccionada con otras tecnologías encontradas.

Posteriormente, se realizó la pertinente búsqueda de características de dichas soluciones, a manera conceptual, es decir, teóricamente.

5. Implementación de pruebas de rendimiento.

Después de la teoría, viene la práctica. Las pruebas de rendimiento se realizaron para medir el rendimiento de las tecnologías en los equipos.

6. Análisis de resultados.

Después de la investigación teórica y de rendimiento, viene el análisis de los datos arrojados por dicha investigación. Se analizaron los datos para poder llegar a una solución sobre si estábamos eligiendo la mejor forma de hacerlo, o no.

7. Conclusión.

Finalmente, y después de evaluar todos los resultados, se tomó la decisión fundamentada de si estábamos eligiendo la mejor manera de llevar a cabo el proceso o no. Si era un sí, se planteará en un futuro la implementación de la mejora, si no, se realizarán más investigaciones sobre las otras tecnologías detectadas.

CAPÍTULO IV. RESULTADOS

Introducción

El rendimiento de una tecnología tiene múltiples aspectos a evaluar. En este caso, lo que nos importaba era medir el rendimiento en la virtualización de entornos de desarrollo utilizando Docker comparado con la otra forma de levantar un entorno de desarrollo con las que contamos, las cuales son las KVM y este caso usamos Vagrant, midiendo ambos rendimientos contra la forma más eficiente de levantar un entorno, pero no más recomendada cuando se trabaja en varios proyectos y todos tienen diferentes configuraciones y servicios, la cual es nativamente, es decir, no virtualizada.

Con este análisis se pretende aislar y comprender rendimiento producido por las máquinas virtuales (VirtualBox con Vagrant) y los contenedores (Docker) basándonos principalmente en la no virtualización y obtener cuál de las formas virtualizadas es la que se acerca más a esta última para poder determinar si Docker es hoy en día una mejor tecnología a implementar sobre las máquinas virtuales.

De esta forma se pretende dar respuesta a si la hipótesis planteada al inicio de la investigación es o no es aprobada, esto dependiendo de los resultados que arroje el benchmark, pero además se tomará en cuenta la teoría de contenedores y la de máquinas virtuales, pues aparte del rendimiento, la facilidad de operación y su fácil estandarización es también importante para la resolución de esta mejora para la empresa Internet Brands.

Cabe destacar que todas las pruebas se realizaron a través de OpenStack, una plataforma que combina herramientas de open source que utilizan conjuntos de recursos virtuales para crear y gestionar nubes privadas y públicas [37], esta herramienta facilitó la tarea de levantar un benchmark con las configuraciones adecuadas, un benchmark es una técnica utilizada para medir el rendimiento de un sistema o uno de sus componentes.

Análisis de datos

Se tomó como base un benchmark realizado por IBM para el análisis de los resultados obtenidos en las pruebas de rendimiento que se presentan a continuación, esto por el coste que podría generar el realizar las pruebas en un equipo con demasiado equipamiento, como era requerido. [39]

Ancho de banda de la red

Se utilizó nuttcp, que es una herramienta de medición de rendimiento de red diseñada para ser utilizada por los administradores de redes y sistemas, para medir el rendimiento de la transferencia de datos unidireccional a través de una única dirección TCP con una unidad máxima de transferencia (UMT).

Las tres configuraciones alcanzan 9.3 Gbps tanto en la dirección de transmisión como de recepción, muy cerca del límite teórico de 9.41 Gbps debido a los encabezados de paquetes. Debido a la descarga de segmentación, la transferencia masiva de datos es muy eficiente incluso teniendo en cuenta las capas adicionales creadas por las diferentes formas de virtualización.

La Figura 3 muestra la utilización de la CPU en todo el sistema para esta prueba, medida usando el comando perf stat -a.

El uso de Docker de puentes y NAT aumenta notablemente la longitud del camino de transmisión; vhost-net es bastante eficiente en la transmisión, pero tiene una gran sobrecarga en el lado de recepción. Los contenedores que no usan NAT tienen un rendimiento idéntico al Linux nativo.

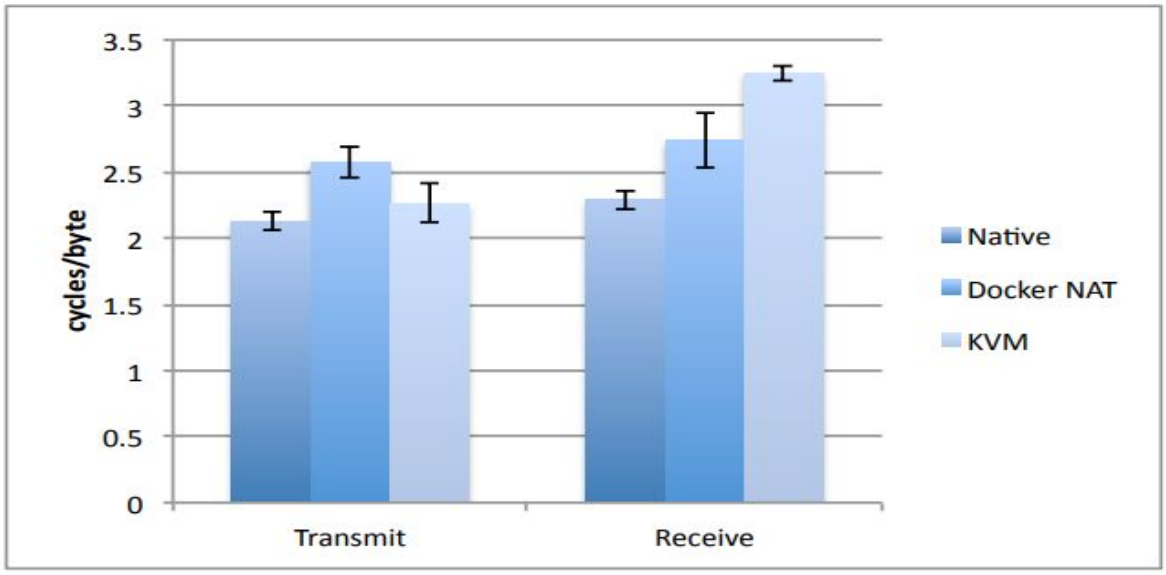

Figura 3: Eficiencia de la transferencia TCP.

Latencia de conexión

Se utilizó el netperf, un benchmark que se puede usar para medir diversos aspectos del rendimiento de la red, utilizando el rendimiento de request/response para medir la latencia de red de ida y vuelta.

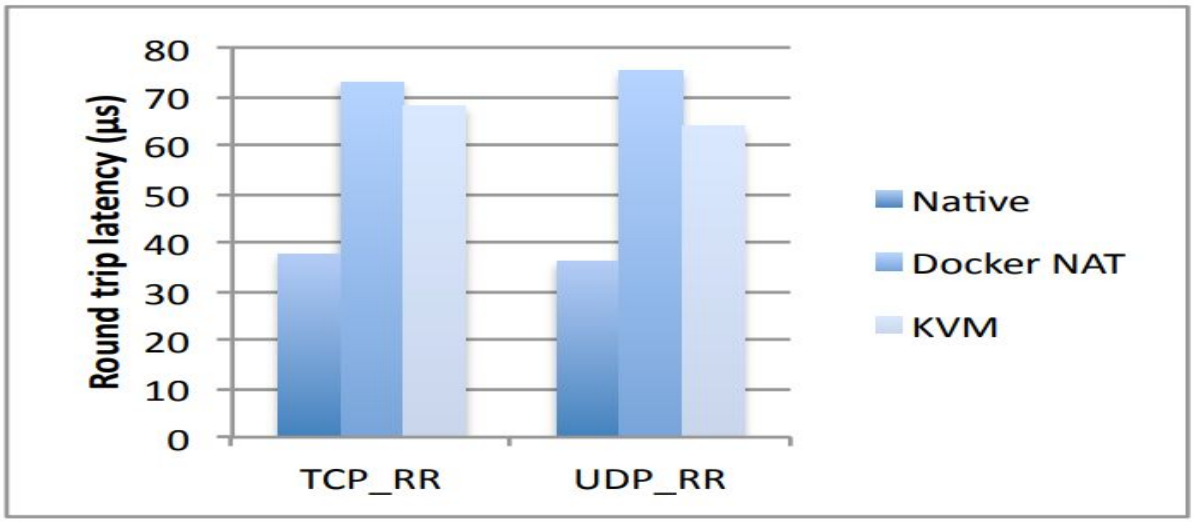

Figura 4: Latencia de ida y vuelta de red (µs).

La Figura 4 muestra la latencia de transacción medida para las variantes TCP y

UDP del benchmark. NAT, como se usa en Docker, duplica la latencia en esta prueba. KVM agrega 30 µs de sobrecarga a cada transacción en comparación con la pila de red no virtualizada, un aumento del 80%. TCP y UDP tienen latencia muy similar porque en ambos casos una transacción consiste en un solo paquete en cada dirección.

Redis

La prueba consiste en una cantidad de clientes que emiten solicitudes al servidor. Se utilizó una mezcla de 50% de lectura y 50% de escritura. Cada cliente mantiene una conexión TCP persistente con el servidor y puede canalizar hasta 10 solicitudes simultáneas a través de esa conexión. Por lo tanto, el número total de solicitudes en vuelo es 10 veces el número de clientes.

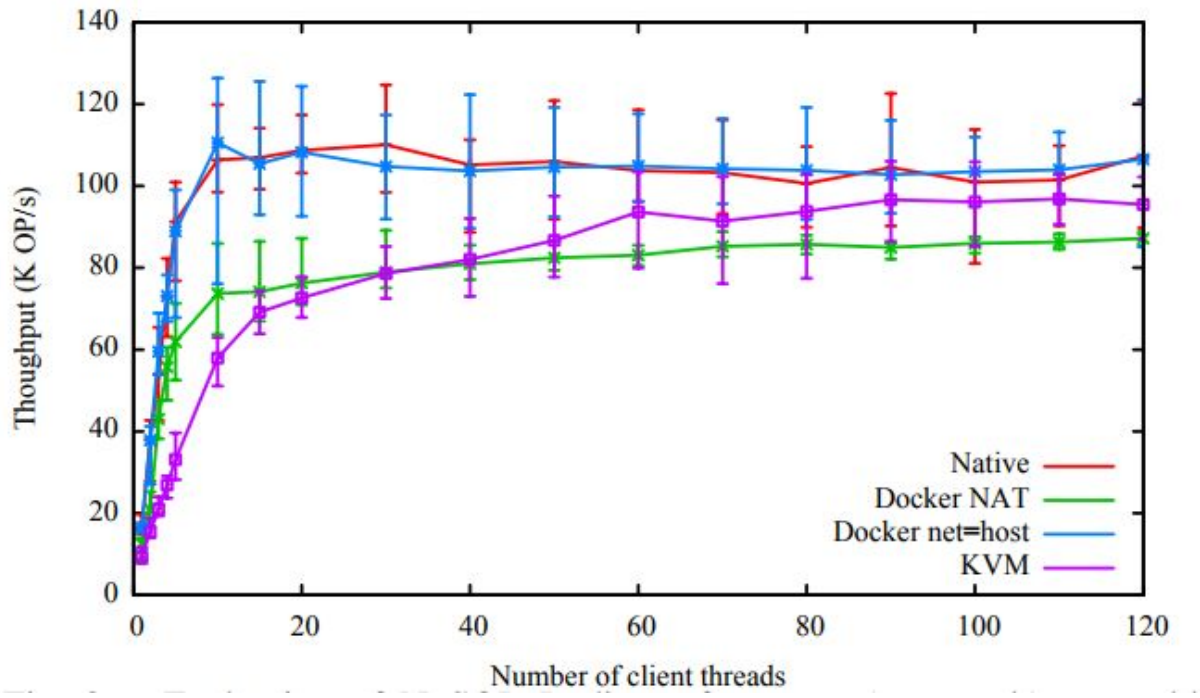

Figura 5: Evaluación del rendimiento de Redis NoSQL en múltiples escenarios de implementación. Cada punto de datos es la media aritmética obtenida de 10 corridas.

La Figura 5 muestra el rendimiento (en solicitudes por segundo) con respecto al número de conexiones de clientes para los diferentes modelos de implementación.

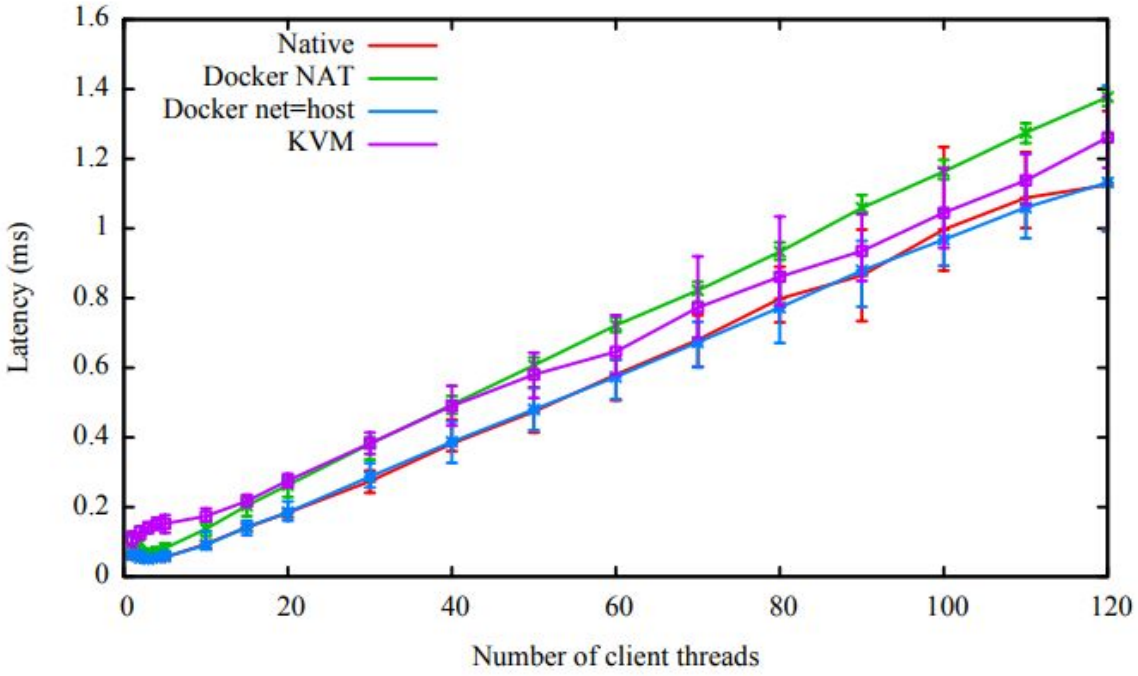

Figura 6: Latencia promedio (en ms) de operaciones en diferentes implementaciones de Redis. Cada punto de datos es la media aritmética obtenida de 10 corrida.

La Figura 6 muestra las latencias promedio correspondientes (en µs) para cada uno de los experimentos. En la implementación nativa, el subsistema de red es bastante suficiente para manejar la carga.

Un servidor Redis ejerce mucha presión en los subsistemas de red y memoria. Al utilizar Docker con la pila de red del host, podemos ver que tanto el rendimiento como la latencia son prácticamente los mismos que en el caso nativo.

MySQL

Se ejecutó el benchmark sysBench, que es una herramienta para benchmarks de secuencias de comandos basada en LuaJIT [38], contra una única instancia de MySQL 5.5.37.

MySQL fue configurado para usar InnoDB como motor de base de datos y se habilitó una memoria caché de 3GB; este caché es suficiente para almacenar en caché todas las lecturas durante las ejecuciones de las pruebas. El benchmark utiliza una base de datos precargada con 2 millones de registros y ejecuta un conjunto fijo de transacciones de lectura / escritura eligiendo entre cinco consultas SELECT, dos consultas UPDATE, una consulta DELETE y un INSERT.

Se midieron cinco configuraciones diferentes: MySQL ejecutando normalmente en Linux (nativo), MySQL bajo Docker utilizando redes de host y un volumen (Docker net = volumen de host), usando un volumen, pero una red Docker normal (volumen Docker NAT), almacenando la base de datos dentro del sistema de archivos contenedor (Docker NAT AUFS) y MySQL ejecutándose bajo KVM.

Configuration	Network	Storage
Native	Native	Native
Docker net=host Volume	Native	Native
Docker NAT Volume	NAT	Native
Docker Nat AUFS	NAT	AUFS
KVM	vhost-net	virtio + qcow

Tabla 2: Configuración de MySQL

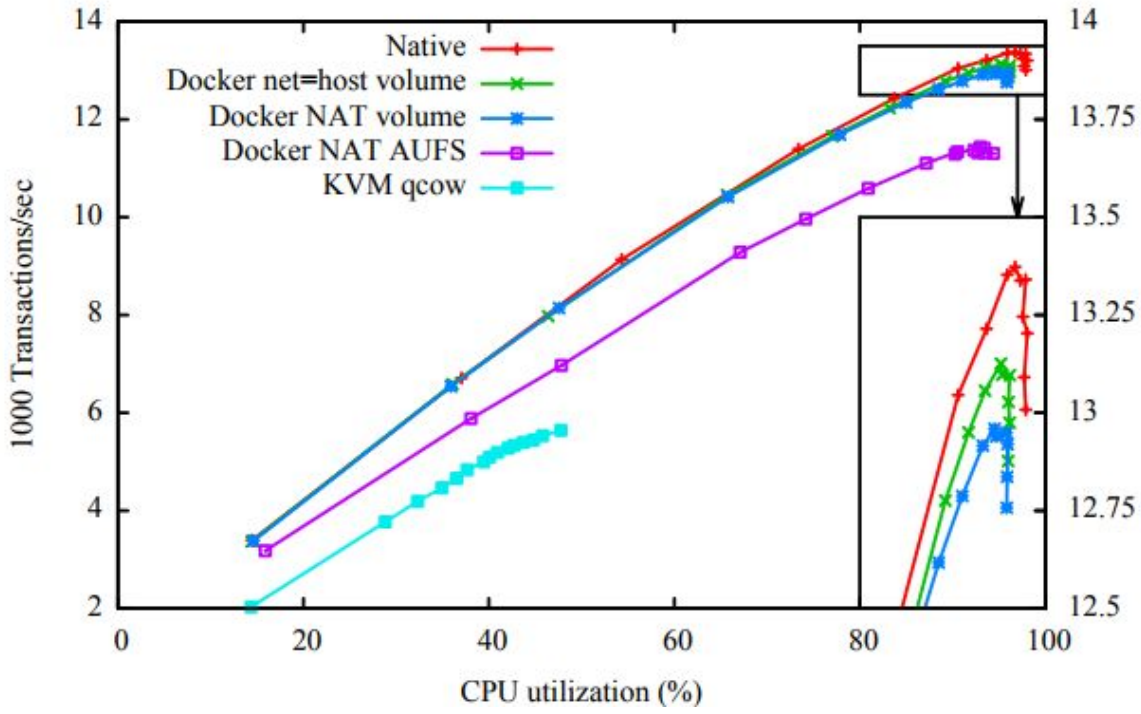

Figura 7: Rendimiento de MySQL (transacciones / s) vs. utilización de la CPU.

En la figura 7 el eje Y derecho muestra la pérdida en el rendimiento en comparación con nativo.

La forma general de esta curva es la que esperaríamos: el rendimiento aumenta con la carga hasta que la máquina se sature, luego se nivela con una pequeña pérdida debido a la contención cuando se sobrecarga. Docker tiene un rendimiento similar al nativo, con una diferencia asintóticamente cercana al 2% a mayor concurrencia. KVM tiene una sobrecarga mucho más alta, más del 40% en todos los casos medidos.

El benchmark genera una gran cantidad de paquetes pequeños, por lo que, aunque el ancho de banda de la red es pequeño, la pila de red no puede mantener la cantidad de paquetes por segundo necesarios. Dado que el punto de referencia utiliza solicitudes sincrónicas, un aumento en la latencia también reduce el rendimiento en un nivel de concurrencia dado.

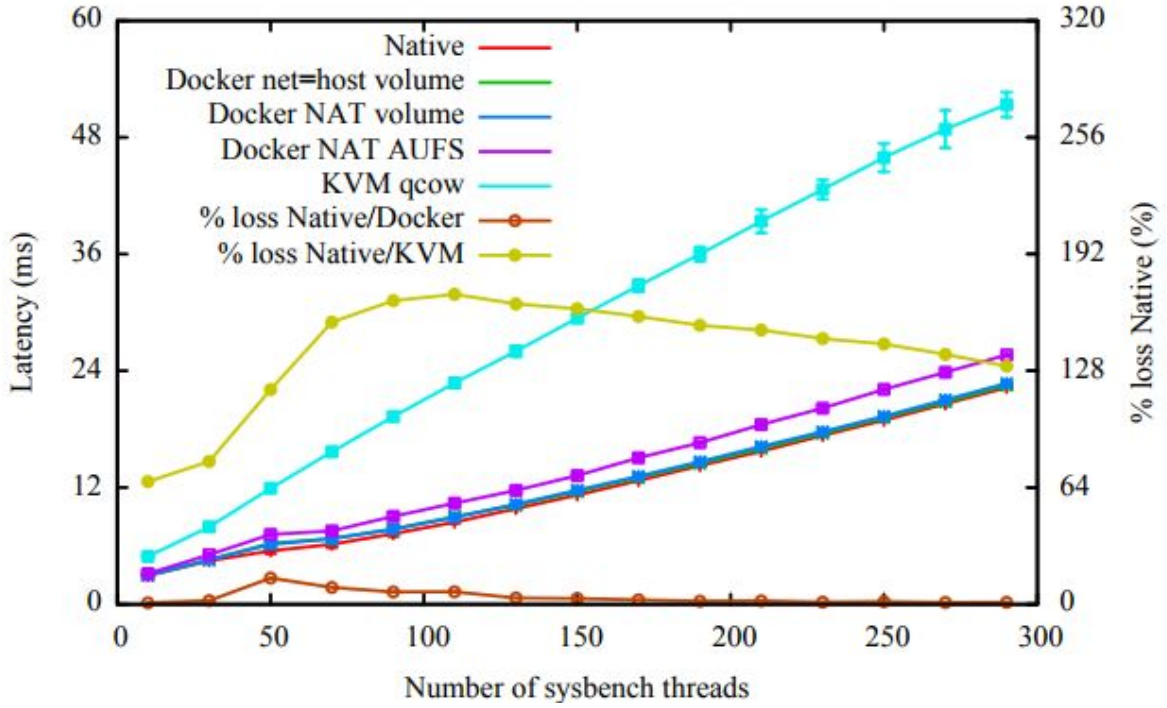

Figura 8: Latencia de MySQL (en ms) vs. concurrencia.

La Figura 8 muestra la latencia en función de la cantidad de usuarios simulados por SysBench. Como era de esperar, la latencia aumenta con la carga, pero curiosamente, Docker aumenta la latencia más rápido para niveles moderados de carga, lo que explica el menor rendimiento a niveles de concurrencia bajos. La porción expandida del gráfico muestra que Linux nativo es capaz de alcanzar una utilización de CPU máxima más alta y Docker no puede alcanzar ese mismo nivel, una diferencia de alrededor del 1.5%. Este resultado muestra que Docker tiene un impacto pequeño pero mensurable.

Las curvas de latencia de rendimiento en la Figura 9 facilitan la comparación de las alternativas dadas una latencia o rendimiento objetivo. Un aspecto interesante de esta curva es la reducción del rendimiento causada por un mayor cambio de contexto en varios casos cuando se introducen más clientes después de la saturación. Dado que hay más tiempo de inactividad en el caso de Docker, una sobrecarga más alta no afecta el rendimiento del número de clientes utilizados en el

índice de referencia.

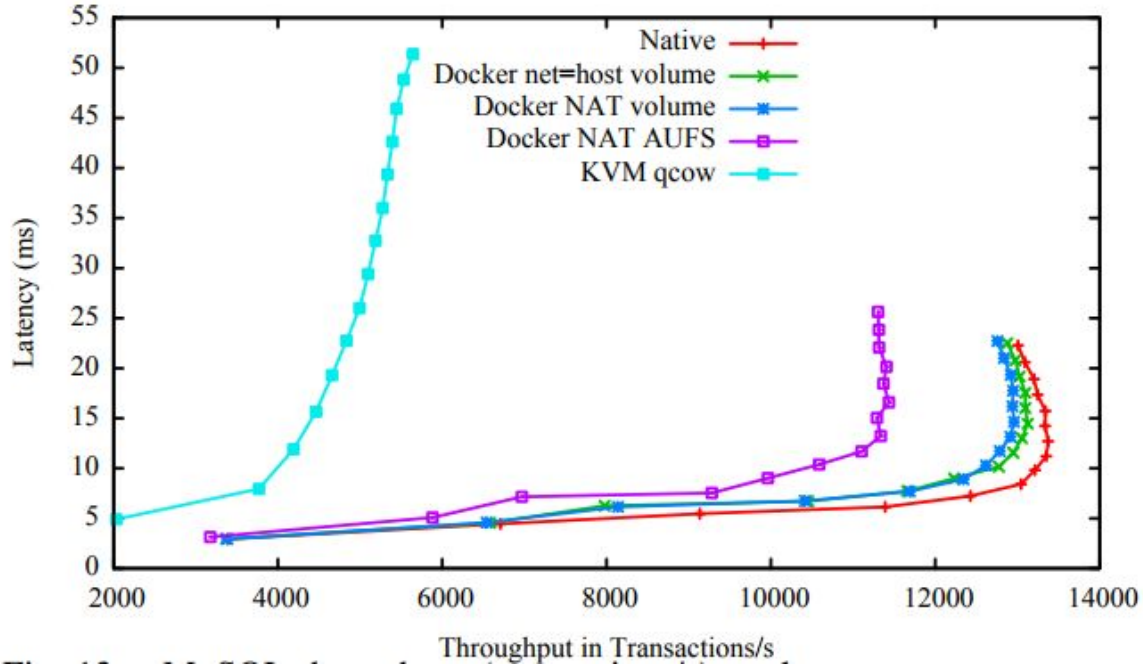

Figura 9: Rendimiento de MySQL (transacciones / s) vs. latencia.

CAPÍTULO V. CONCLUSIONES

Conclusiones

Como es apreciable en los resultados expuestos, ambas tecnologías, las máquinas virtuales con vagrant (KVM) tanto como los contenedores de Docker, son buenas tecnologías a la hora de virtualizar un ambiente, de hecho, se puede apreciar como en rendimiento tienen casos en los que son muy similares, sin embargo, en algunos casos Docker excede el rendimiento de las KVM.

Ambas tecnologías introducen una sobrecarga insignificante para el rendimiento del CPU, esto debido al gran equipo en el que estamos probando, sin embargo, en equipos más pequeños, es una cantidad que significa.

Para finalizar y fusionando la investigación teórica con investigación de rendimiento, se llega a la conclusión de que, en efecto, al utilizar Docker se ahorran recursos a la hora de levantar un entorno de desarrollo, y que esta, además, es una mejor opción a la hora de desarrollar en máquinas no tan potentes, pues utiliza menos recursos que una máquina virtual.

El único inconveniente es el sistema operativo, pues al no poder ser virtualizable, se tiene que tener en consideración que este puede variar de máquina en máquina. Esto no tiende a afectar mucho el correcto levantamiento del entorno, pero es bueno tomarlo en cuenta.

Por lo que, en resumen, la hipótesis planteada al inicio de este documento se cumple sustentablemente,

Dado lo anterior, la propuesta de la estandarización de los entornos de desarrollo mediante el uso de Docker será expuesta a las personas responsables de la parte técnica de cada proyecto, esperando así, mejorar uno de los procesos de la empresa.

Recomendaciones o sugerencias

La estandarización de los procesos dentro de una empresa es importante para mejorar la eficiencia con la que se llevan a cabo, pues una de sus ventajas es la reducción de errores. De esta forma se evitan conflictos de malentendidos, lo que reduce el costo de tiempo que les toma a las personas realizar dichas tareas.

Hablando de eficiencia, la virtualización de los entornos de desarrollo reduce el tiempo que le toma a un desarrollador prepararse para empezar a trabajar en un proyecto, por lo que dedica más tiempo a la codificación, especialmente si cambia mucho de proyecto.

Por ende, aplicar la estandarización a los procesos de virtualización utilizando Docker, que fue la herramienta de estudio, hará que la empresa se vea afectada positivamente ahorrando tiempo de desarrollo y reduciendo costos de equipo.

DEFINICIÓN DE TÉRMINOS

Docker

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de Virtualización a nivel de sistema operativo en Linux.

Entorno de desarrollo

Conjunto de herramientas o programas que conforman una aplicación completa, desde los servicios que se necesitan hasta la aplicación en general, todo para el correcto funcionamiento de la aplicación.

LEGAL/NOLO

Vertical dentro de Internet Brands que se encarga de todos los proyectos que tiene que ver con cuestiones legales, el nombre de NOLO viene de su principal servicio que se encarga de conectar a los abogados de Estados Unidos a través de los servicios ofrecidos en nolo.com

Máquina virtual

Una máquina virtual es un software que emula un ordenador justo como si fuese uno real.

Vagrant

Vagrant es una herramienta para crear y administrar entornos de máquinas virtuales en un solo flujo de trabajo.

REFERENCIAS

[1] INTERNET BRANDS, Internet Brands. About us [En línea] [consultado 10 octubre 2017]. Disponible en: <https://www.internetbrands.com/about-us/>

[2] INTERNET BRANDS, Internet Brands. Work with us [En línea] [consultado 10 octubre 2017]. Disponible en:<https://www.internetbrands.com/work-with-us/>

[3] EAM, Information Technologies. Desarrollo de aplicaciones. En: Virtualización de entornos de desarrollo – Parte 1 [En línea] (2016) [Consultado 12 octubre 2017]. Disponible en <http://www.eam.es/blog/virtualizacion-en-entornos-de-desarrollo-parte-1/>

[4] FIALLOS, Jack. Qué es Docker Container. [En línea] (2015) [Consultado 12 octubre 2017]. Disponible en <http://qbit.com.mx/blog/2015/11/26/que-es-docker-container/>

[5] VELASCO, Rubén. Probamos la virtualización por contenedores de Docker en Windows. En: Cómo funciona Docker con Windows. [En línea] (2016) [Consultado 12 octubre 2017]. Disponible en <https://www.redeszone.net/2016/05/12/probamos-la-virtualizacion-contenedores-docker-windows/>

[6] DOCKER, Docker. What is Docker. [En línea] [Consultado 3 noviembre 2017]. Disponible en <https://www.docker.com/what-docker>

[7] HOWTOBRAIN, howtobrain. Contenedores 101: Introducción a Docker. En: Historia de Docker. [En línea] (2016) [Consultado 3 noviembre 2017]. Disponible en <http://howtobrain.org/contenedores-101-introduccion-a-docker/>

[8] DOCKER, Docker. What is a container. [En línea] [Consultado 3 noviembre 2017]. Disponible en <https://www.docker.com/what-container>

[9] ROKITOH, rokitoh. Instalar servidor LXC en GNU/Linux. [En línea] (2015) [Consultado 3 noviembre 2017]. Disponible en <http://red-orbita.com/?p=6824>

[10] DOCKER, Docker. Docker Compose. En: Compose Overview. [En línea] [Consultado 3 noviembre 2017]. Disponible en <https://docs.docker.com/compose/>

[11] DOCKER, Docker. Dockerfile Reference. En: Dockerfile Reference. [En línea]

[Consultado 3 noviembre 2017]. Disponible en <https://docs.docker.com/engine/reference/builder/>

[12] ROUSE, Margaret. Virtual Machine (VM). En: Definition. [En línea] [Consultado 3 noviembre 2017]. Disponible en <http://searchservervirtualization.techtarget.com/definition/virtual-machine>

[13] HASHICORP, Vagrant. What is Vagrant. En: Introduction to Vagrant. [En línea] [Consultado 3 noviembre 2017]. Disponible en <https://www.vagrantup.com/intro/index.html>

[14] PALAT, Jay. Introducing Vagrant. En: What is Vagrant. [En línea] (2012) [Consultado 3 noviembre 2017]. Disponible en <http://www.linuxjournal.com/content/introducing-vagrant>

[15] MOLINA, Alberto. ¿Qué es Vagrant? [En línea] (2017) [Consultado 3 noviembre 2017]. Disponible en <https://openwebinars.net/blog/que-es-vagrant-videotutorial/>

[16] HASHICORP, Vagrant. Vagrantfile [Consultado 3 noviembre 2017]. Disponible en <https://www.vagrantup.com/docs/vagrantfile/>

[17] VIRTUALBOX, VirtualBox. VirtualBox. En: Welcome to VirtualBox.org! [En línea] [Consultado 3 noviembre 2017]. Disponible en <https://www.virtualbox.org/>

[18] EVALUANDO SOFTWARE, Evaluando Software. Qué es la virtualización. [En línea] (2015) [Consultado 3 noviembre 2017]. Disponible en <http://www.evaluandosoftware.com/que-es-la-virtualizacion/>

[19] ITSOFTIN, Itsoftin. Diferencias entre virtualizar un Sistema operativo e instalarlo. [En línea] [Consultado 3 noviembre 2017]. Disponible en <http://itsoftindustrial.com/index.php/13-notas/22-diferencias-entre-virtualizar-un-sistema-operativo-e-instalarlo>

[20] SARASTI, Katherine. Vitualización Libro Dígital. En: Retos de la virtualización. [En línea] (2015) [Consultado 3 noviembre 2017]. Disponible en <http://es.calameo.com/books/00434742881dc41d4d63b>

[21] INSINEC, Insinec. Virtualización de servidores. [En línea] [Consultado 3 noviembre 2017]. Disponible en <http://insinec.es/sistemas-informaticos/virtualizacion-de-servidores>

[22] RED HAT, Red Hat. ¿Qué es un contenedor de Linux? En: ¿No sería esto solo virtualización? [En línea] [Consultado 3 noviembre 2017]. Disponible en

<https://www.redhat.com/es/topics/containers/whats-a-linux-container>

[23] PRADA, Elisa. Tecnología LINUX, ERP y APLICACIÓN. En: GNU/Linux. [En línea] (2017) [Consultado 18 noviembre 2017]. Disponible en <https://issuu.com/elisaprada7/docs/revista_20elisa_20m_20prada>

[24] UBUNTU, Ubuntu. Welcome to Ubunto. En: What is Ubuntu? [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://help.ubuntu.com/lts/installation-guide/s390x/ch01s01.html>

[25] DEBIAN, Debian. Acerca de Debian. En: ¿Qué es Debian? [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.debian.org/intro/about#what>

[26] CENTOS, CentOS. About. En: CentOS Linux. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.centos.org/about/>

[27] RED HAT, Red Hat. Plataformas Linux. En: Red Hat Enterprise Linux. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.redhat.com/es/technologies/linux-platforms/enterprise-linux>

[28] PEREZ VALDES, Damian. ¿Qué son las bases de datos? [En línea] (2007) [Consultado 18 noviembre 2017]. Disponible en <http://www.maestrosdelweb.com/que-son-las-bases-de-datos>

[29] ORACLE, Oracle. Overview. En: Performant, Reliable and Easy to use. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.oracle.com/mysql/index.html>

[30] PERCONA, Percona. Percona Server for MySQL. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.percona.com/software/mysql-database/percona-server>

[31] POSTGRESQL, PostgreSQL. About. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.postgresql.org/about/>

[32] MARIADB, MariaDB. About MariaDB. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://mariadb.org/about/>

[33] MONGODB, mongoDB. Renventando la gestión de datos. En: Reinventando la gestión de datos. [En línea] [Consultado 18 noviembre 2017]. Disponible en <https://www.mongodb.com/es>

[34] REDIS, redis. Inroduction to Redis. [En línea] [Consultado 18 noviembre 2017].

Disponible en <https://redis.io/topics/introduction>

[35] ROUSE, Margarete. Servidor Web. En: Definition. [En línea] [Consultado 26 noviembre 2017]. Disponible en <http://searchdatacenter.techtarget.com/es/definicion/Servidor-Web>

[36] CASTELLANOS, Luis. Sistemas Operativos para servidores Web. En: Apache. [En línea] (2014) [Consultado 26 noviembre 2017]. Disponible en <https://lcsistemasoperativos.wordpress.com/tag/apache/>

[37] RED HAT, Red Hat. El concepto de OpenStack. En: ¿Qué es? [En línea] [Consultado 26 noviembre 2017]. Disponible en <https://www.redhat.com/es/topics/openstack>

[38] AKOPITOV, akopitov. Sysbench. [En línea] [Consultado 26 noviembre 2017]. Disponible en <https://github.com/akopytov/sysbench>

[39] FELTER, Wes. FERREIRA, Alexandre. RAJAMONI, Ram. RUBIO, Juan. An Updated Performance Comparison of Virtual Machines and Linux Containers. En: Evaluation. [En línea] (2014) [Consultado 30 noviembre 2017]. Disponible en <http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf>