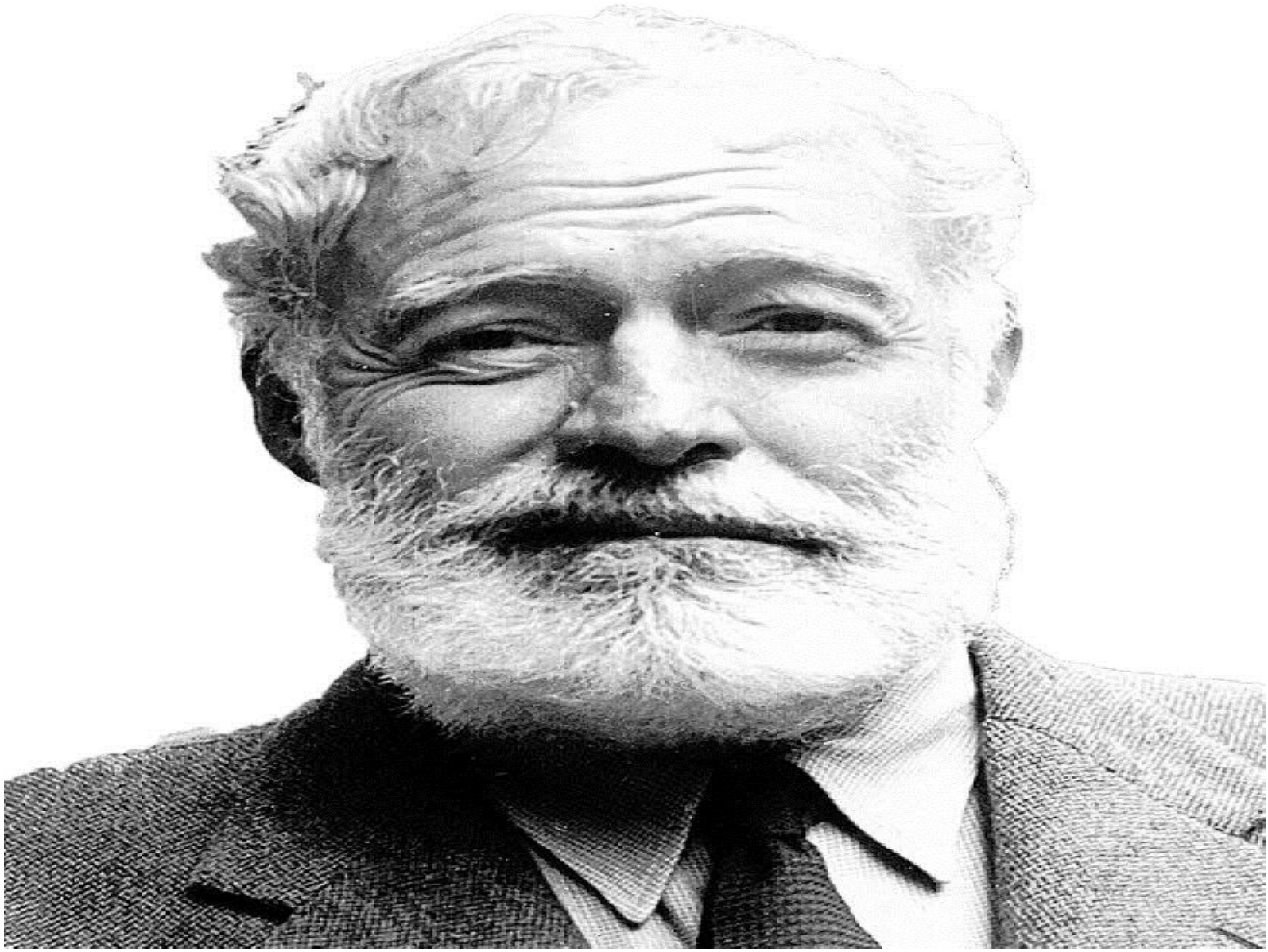
Universidad de las Ciencias Informáticas
Facultad 1

Título: Plataforma de Virtualización para Nova Servidores

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Autor(es): Yosmay Morales Suárez
Tutor(es): Ing. Cecilia Esther Hernández Espinosa
Ing. Gladys Marsi Peñalver Romero

La Habana, 2016



"El secreto de la sabiduría, del poder y del conocimiento es la humildad."

Ernest Hemingway

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____
Autor: Yosmay Morales Suárez.

_____ _____
Tutora: Ing. Cecilia Esther Hernández Espinosa. Tutora: Ing. Gladys Marsi Peñalver Romero.

AGRADECIMIENTOS

En primer lugar quiero agradecer a mi padrastro Mario Báez, por el apoyo que siempre me ha brindado y sobre todo por no dejarme dar un paso atrás en momentos difíciles de la vida.

Agradecer a mi madre y mi abuela Livian, por el amor y comprensión que siempre me ofrecen.

Agradecer a mis tutoras Cecilia y Gladys por todo el esfuerzo, dedicación y apoyo que me brindaron en el desarrollo de la investigación.

Quiero agradecer a todas mis amistades, en especial a mis compañeros de aula que estuvieron presente en los momentos difíciles durante el transcurso de la carrera universitaria.

DEDICATORIA

Dedico los logros obtenidos a toda mi familia principalmente a: mi madre Yosamy, mi abuela Livian, hermana Amelia, mi padrastro Mario, mis tíos Yaisy, Pachy y mi primo hermano Kevin.

RESUMEN

La virtualización es una de las principales tendencias que buscan concretar la mayoría de las empresas en la actualidad, permitiéndoles construir arquitecturas tolerantes a fallos con bajos tiempos de indisponibilidad; así como mitigar la infrautilización del hardware de los servidores, haciendo un uso más eficiente de los recursos del servidor. Muchas de las instituciones cubanas se inclinan al mismo objetivo para darle solución a los problemas relacionados con el uso de las Tecnologías de la Información y la Comunicación. El presente trabajo de diploma tiene el objetivo de establecer un hipervisor para la distribución cubana de GNU/Linux Nova Servidores, variante a ser utilizada para las redes públicas en Órganos de la Administración Central del Estado. Para la selección del hipervisor, al concluir con un análisis detallado sobre la virtualización de servidores, se determinan variables a estar presente en una plataforma de virtualización, métricas que cuantifiquen cada variable y pruebas de desempeño. Cada una de las pruebas otorga un valor a las métricas asociadas y permite comparar las capacidades de los hipervisores. A partir de las mismas, se obtuvo como resultado que el hipervisor LXC es el que le posibilitará a Nova Servidores el máximo aprovechamiento de las prestaciones de los recursos de hardware disponibles.

Palabras clave: GNU/Linux, hipervisor, Nova Servidores, plataforma de virtualización.

ÍNDICE

INTRODUCCIÓN **1**

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA DE LA PLATAFORMA DE VIRTUALIZACIÓN PARA NOVA SERVIDORES **5**

1.1. **DESCRIPCIÓN DE ELEMENTOS SIGNIFICATIVOS DEL DOMINIO** 5

1.2. **ESTADO DEL ARTE** 6

1.3. **ACERCAMIENTO A LA VIRTUALIZACIÓN** 10

1.4. **ANÁLISIS DE LAS TÉCNICAS DE VIRTUALIZACIÓN** 11

1.5. **CONCLUSIONES PARCIALES** 20

CAPÍTULO 2. CRITERIOS Y PRUEBAS PARA LA SELECCIÓN DE PLATAFORMAS DE VIRTUALIZACIÓN **21**

2.1. **JUSTIFICACIÓN DE LOS CRITERIOS, MÉTRICAS Y PRUEBAS UTILIZADAS** 21

2.2. **RELACIÓN ENTRE VARIABLES Y TÉCNICAS DE VIRTUALIZACIÓN** 24

2.3. **MÉTRICAS PARA LA EVALUACIÓN DE LAS VARIABLES DE LAS PLATAFORMAS DE VIRTUALIZACIÓN** 24

2.4. **PRUEBAS PARA LA EVALUACIÓN DE PLATAFORMAS DE VIRTUALIZACIÓN** 26

2.5. **PLATAFORMAS DE VIRTUALIZACIÓN SELECCIONADAS PARA EL ESTUDIO** 32

2.6. **CONCLUSIONES PARCIALES** 39

CAPÍTULO 3. EVALUACIÓN DE LAS VARIABLES A TENER PRESENTE EN UNA PLATAFORMA DE VIRTUALIZACIÓN **40**

3.1. **DESCRIPCIÓN DEL HARDWARE Y SOFTWARE EMPLEADO** 40

3.2. **RESULTADOS OBTENIDOS DE LAS PRUEBAS** 42

3.3. **RESULTADO FINAL** 52

3.4. **CONCLUSIONES PARCIALES** 58

CONCLUSIONES **59**

RECOMENDACIONES **60**

REFERENCIAS BIBLIOGRÁFICAS **61**

INTRODUCCIÓN

La constante evolución de las Tecnologías de la Información y la Comunicación (TIC) conlleva a que las empresas que trabajan con equipos informáticos de grandes prestaciones se enfrenten con frecuencia a problemas como el ahorro de costes debido a la adquisición de equipamiento de cómputos, los gastos en mantenimiento, reparación de los equipos y soporte. Al mismo tiempo, toda empresa se traza la meta de aplicar mecanismos de seguridad y disponibilidad para ofrecer servicios eficientes para beneficio de sus usuarios. Algunas de estas empresas, producto de los propios problemas de ahorro de coste, cuentan con pocos servidores físicos, siendo necesario tener varios servicios en un solo servidor cuando es conveniente tener cada uno en un ordenador físico independiente. Estos elementos traen consigo dificultades que provocan ineficiencias en los servicios brindados, incurriendo en la seguridad, debido a que los servicios no se encuentran aislados y el fallo de uno de ellos pudiera provocar interrupciones en otros.

Por otra parte, varias de las empresas a pesar de presentar problemas de ahorro de coste, alojan un servicio por servidor, incurriendo en una infrautilización del hardware del servidor ya que no aprovechan al máximo sus capacidades. Como vía de solución a los escenarios descritos, desde 1960 las empresas acuden a la virtualización como una técnica que ayuda a reducir tanto el número de servidores físicos, como los gastos directamente asociados a grandes infraestructuras. De esta manera, se les proporciona a los usuarios de las TIC facilidades de administración a partir de la centralización y la optimización de los recursos.

Entre las principales ventajas que brinda la virtualización se incluye el incremento de la seguridad, consolidación de los recursos, tolerancia a fallos, así como un mejor aprovechamiento del hardware disponible [1] [2] [3].

Un entorno virtualizado posibilita que en lugar de tenerse cuatro servidores físicos, alojando un servicio (correo electrónico, web, sistema de nombres de dominio y gestores de bases de datos) en cada servidor, se obtengan dos servidores virtuales por cada físico, alojando un servicio por sistema virtual, comportándose como si estuvieran sobre uno real. Mediante este proceso se consigue dar una respuesta rápida en caso de desastres o fallos, además de permitir realizar una restauración de los ordenadores de forma rápida y segura. También se consigue la reducción de costos de la infraestructura, debido a la disminución de consumo y de recursos, ya que en lugar de utilizar cuatro máquinas se utilizan solamente dos.

Además, la disponibilidad es un factor clave en la infraestructura de los Centros de Procesamiento de Datos (CPD), respaldado por la centralización de los datos, donde toda la información se encuentra en un mismo sitio. La centralización permite almacenar en un servidor independiente toda la información, lo que favorece una adecuada gestión de los datos, así como facilidad de acceso. La virtualización es un proceso que evita interrupciones de un servicio determinado, puesto que si ocurriese un fallo en una aplicación o servidor se podría usar de inmediato un servidor virtual como respaldo [2]. Al mismo tiempo, permite que la administración de los recursos tecnológicos sea simple, teniendo en cuenta que las cargas de trabajo se implementan con mayor rapidez, el rendimiento y la disponibilidad aumentan, y las operaciones se automatizan.

Otra de las ventajas de la virtualización es la integración con infraestructuras de nubes, ya que para llegar al cómputo en la nube, muchas de las empresas a nivel mundial parten de la implementación de la virtualización [4]. Al mismo tiempo que la computación en la nube se vuelve una de las principales tendencias que acogen las empresas actualmente. Generalmente las tecnologías de virtualización permiten construir arquitecturas tolerantes a fallos con bajos tiempos de indisponibilidad, así como mitigar la infrautilización del hardware de los servidores, haciendo un uso más eficiente de los recursos del servidor.

Los Órganos de la Administración Central del Estado (OACE) en Cuba, siguen las tendencias de la virtualización para solucionar los problemas existentes relacionados con el uso de las TIC. A partir de la importancia de la virtualización, en la Universidad de las Ciencias Informáticas (UCI), el Centro de Software Libre (CESOL), trabaja en la mejora de la distribución de GNU/Linux Nova Servidores, variante a utilizar para la administración y gestión de los servidores y servicios telemáticos en las redes públicas de los OACE. El objetivo de la mejora, es utilizar una plataforma de virtualización que les permita a las instituciones del país administrar los servidores y servicios al mismo tiempo que se da un mejor uso a las prestaciones de hardware disponibles. Uno de los principales retos para los administradores es saber cuál es la opción más adecuada para su entorno de trabajo cuando existen tantas soluciones de virtualización diferentes en el mercado y que sean varios los factores a tener en cuenta a la hora de tomar una decisión, puesto que una mala elección representa un decremento considerable en el desempeño de los CPD.

Por lo antes expuesto se define como **problema de investigación**: ¿Cómo determinar una plataforma de virtualización para la distribución cubana de GNU/Linux Nova Servidores que permita un mayor aprovechamiento de las prestaciones de los recursos de hardware disponibles?

Se establece como **objeto de estudio**, la virtualización en distribuciones GNU/Linux y el **campo de acción** lo constituyen las soluciones de virtualización que aplican a Nova Servidores.

El **objetivo general** del presente trabajo consiste en: determinar una plataforma de virtualización para la distribución cubana de GNU/Linux Nova Servidores que permita un mayor aprovechamiento de las prestaciones de los recursos de hardware disponibles.

Del cual se derivan los siguientes **objetivos específicos**:

1. Caracterizar las tendencias actuales asociadas a la virtualización en los CPD.
2. Determinar las plataformas de virtualización que aplican a Nova Servidores.
3. Establecer variables, métricas y pruebas para la selección de la plataforma de virtualización.
4. Comparar los resultados de las pruebas realizadas basados en las métricas y umbrales propuestos.

Para dar cumplimiento a los objetivos específicos se definieron las siguientes **tareas de investigación**.

1. Caracterización de las tendencias actuales asociadas a la virtualización en los CPD.
2. Descripción de las plataformas de virtualización que van a utilizarse para la evaluación.
3. Selección de las variables y umbrales para evaluar las plataformas seleccionadas.
4. Evaluación de las plataformas de virtualización seleccionadas.
5. Análisis de los resultados obtenidos de la evaluación de las plataformas de virtualización.

El cumplimiento satisfactorio de las tareas expuestas anteriormente contribuirá al desarrollo exitoso de la **idea a defender:** la selección de una plataforma de virtualización para Nova Servidores permitirá un mayor aprovechamiento de las prestaciones de los recursos de hardware disponibles.

Con el propósito de dar solución a las tareas antes mencionadas se utilizaron los siguientes métodos:

Métodos Teóricos.

- **Analítico-Sintético:** permitió identificar los principales conceptos, características y funcionalidades empleadas dentro del marco de trabajo de la virtualización, analizando y estudiando documentos, trabajos de diploma, artículos y bibliografía de diferentes autores, para así extraer los elementos más significativos.
- **Histórico lógico:** se utiliza con el objetivo de realizar un estudio sobre los antecedentes, evolución y tendencias actuales de la virtualización sobre soluciones de Software Libre y Código Abierto

(SLCA), permitiendo desarrollar una valoración sobre las plataformas de virtualización y determinar en qué entorno es más factible su uso.

Métodos Empíricos.

- **Observación:** permitió hacer un análisis y evaluación del comportamiento de las plataformas de virtualización que se acoplan al entorno de trabajo, con el objetivo de identificar elementos que hagan de ellas una solución competente.

El presente trabajo está estructurado en introducción, tres capítulos de contenido, conclusiones, recomendaciones y referencias bibliográficas. El orden y contenido de los capítulos es el siguiente:

Capítulo 1. "Fundamentación teórica de la Plataforma de Virtualización para Nova Servidores**"**, donde se definen los principales conceptos a tener en cuenta, resultando la base que sustenta el desarrollo de las tareas de investigación. Se realiza un análisis de las tecnologías de virtualización, con el objetivo de determinar los elementos que engloban a la virtualización como una de las tendencias actuales.

Capítulo 2. "Criterios y pruebas para la selección de plataformas de virtualización**"**, se analizan un conjunto de plataformas de virtualización de SLCA que resultan significativas para el campo de acción. Con el objetivo de realizar un análisis entre ellas, determinando variables a estar presente en una plataforma de virtualización que permitan ejercer una evaluación de cada una. La evaluación se realiza haciendo uso de un conjunto de métricas que miden cada una de las variables. Posteriormente se plasman diversas pruebas que forman parte de la evaluación de las plataformas y tienen el objetivo de evaluar el desempeño de los hipervisores.

Capítulo 3. "Evaluación de las variables a tener presente en una plataforma de virtualización**"**. A partir de las métricas, variables y pruebas establecidas en el capítulo dos, se evalúan cada uno de los hipervisores seleccionados para el estudio, lo que permite validar las variables a tener presente en la selección de una plataforma de virtualización. Una vez finalizada la etapa de validación se realiza un análisis de los resultados obtenidos, con el fin de determinar cuál de las plataformas brinda el mejor aprovechamiento de los recursos de hardware disponibles.

CAPÍTULO 1. **FUNDAMENTACIÓN TEÓRICA DE LA PLATAFORMA DE VIRTUALIZACIÓN PARA NOVA SERVIDORES**

La virtualización es una tecnología que impacta de forma considerable en la eficiencia de los CPD, por lo que se requiere de su uso y buen funcionamiento para un mejor desarrollo de los procesos que se gestionan en ellos. En el presente capítulo se plantean los elementos teóricos relacionados con la virtualización, definiéndose los tipos de hipervisores y técnicas que emplean. De esta forma, se establecen las bases para el proceso de evaluación en los capítulos posteriores y así determinar el entorno de virtualización que brinda mayor beneficio sobre el escenario de trabajo de GNU/Linux Nova Servidores 5.0.

1.1. Descripción de elementos significativos del dominio

El proceso de reunir todos los ficheros y programas necesarios, instalarlos en el sistema y configurarlos con el objetivo de obtener un sistema completo de Linux presenta una alta complejidad. Por ello, muchas empresas realizan dicho proceso para el beneficio de los usuarios. Por esto, surgen las **distribuciones GNU/Linux**, que consisten en el agrupamiento de ficheros y programas, organizados y preparados que posteriormente se instalaran en un ordenador [5].

Nova Servidores clasifica como una distribución GNU/Linux dirigida a ordenadores que tienen la función de atender servicios telemáticos como: web, correo electrónico, directorio activo, servicio DNS[1], *proxy* y otros.

Las distribuciones y los servidores son algunas de las tecnologías de las que dependen de forma considerable muchas de las empresas, que cumplen una determinada función, haciendo posible el desarrollo de diversas tareas. Estas tecnologías son gestionadas por los CPD de las empresas, refiriéndose a las instalaciones donde en ellas radican los recursos necesarios, dígase computadoras, redes de comunicación, entre otras dependencias debidamente acondicionadas, que cumplen tareas como la administración y procesamiento de los datos e información de las entidades en donde ofrecen sus servicios. Los CPD son nombrados también como Centro de Datos (DC), por sus siglas en inglés *Data Center* [6].

[1] Domain Name System (Sistemas de Nombre de Dominio).

Muchos de los CPD alojan servidores con grandes capacidades de cómputo los cuales contienen aplicaciones y servicios individuales. La propia evolución de las tecnologías y los requisitos puntuales de las empresas han hecho que las aplicaciones y servicios se instalen y configuren en los servidores en una relación de uno a uno. Es decir, cada vez que se necesita una nueva aplicación, se compra un nuevo servidor y se instala, en lugar de aprovechar los recursos existentes. Por lo que para alimentar a los servidores, se requiere un alto nivel de consumo de energía, además de los costes de compra, incluyendo también otros dispositivos como *switches y routers*.

La virtualización evoluciona como una técnica que permite reducir tanto el número de servidores físicos, como los gastos directamente asociados a grandes infraestructuras. Existen diversos conceptos vinculados a la virtualización, a continuación se presentan los de mayor relevancia para el presente trabajo de diploma.

Máquina Virtual (MV): Es un ordenador, que contiene los mismos recursos tecnológicos que una computadora física, solo que virtuales. Incluye un SO, diversas aplicaciones y los dispositivos virtuales de las MV poseen las mismas funcionalidades que los de la física, solo que con beneficios adicionales en términos de portabilidad, manejabilidad y seguridad [7].

Sistema Operativo Invitado o Huésped (*Guest*): El invitado es el SO que se instala en la MV que crea el software de virtualización, al cual se le asignan determinados recursos de cómputo para funcionar. [8]

Sistema Operativo Anfitrión (*Host*): El anfitrión es el SO del ordenador físico, sobre el cual se instala el software que realiza el proceso de virtualización [8].

Escenario de trabajo: Referente a la administración de los servidores y servicios por parte de Nova Servidores 5.0 ejecutándose en un ordenador.

1.2. Estado del arte

En la actualidad, a medida que el desarrollo de la informática se eleva, usuarios de todas partes del mundo se vuelven más exigentes en cuanto a servicios telemáticos se trata. Por lo que para satisfacer las demandas existentes, mantener en línea los servicios y ofrecer una administración íntegra se hace necesaria la utilización de servidores que permitan la gestión de los servicios. Actualmente existen diferentes servidores que satisfacen la necesidad de brindar una variedad de servicios, en función del usuario, cada uno realizando una funcionalidad específica, por ejemplo: servidores proxy, de correo electrónico, DNS,

Web, o de bases de datos. Además, es significativo el uso de soluciones de propósito específico para brindar los servicios, siendo el caso de los sistemas operativos orientados a servidores.

1.2.1. La virtualización en los SO para servidores

Los SO para servidores ofrecen seguridad y estabilidad, así como las herramientas necesarias para una adecuada administración. Diversos fabricantes de sistemas incluyen en sus productos soporte para plataformas de virtualización con el objetivo de aprovechar al máximo las funcionalidades de los servidores. Al mismo tiempo, se benefician de las características y ventajas que ofrece una plataforma determinada y la virtualización en general, igualmente reflejado en los servicios. Muchos de estos sistemas ofrecen soporte para varios hipervisores, mientras que algunos cuentan con plataformas de virtualización integradas en el kernel (núcleo del sistema), siendo el caso de GNU/Linux, lo cual ofrece apoyo en el proceso de virtualización. A partir de los sistemas destinados para servidores, se encuentran soluciones como: Windows Server, 2003, 2008, 2012 entre otros, por parte de la compañía *Microsoft*. Existen también otros sistemas basados en Unix como: FreeBSD, OpenBSD, Mac OS y Solaris. Mientras que por parte de GNU/Linux se encuentran: Red Hat, Debian y Ubuntu Server.

A continuación se reflejan las principales plataformas que soportan los sistemas enfocados en el objeto de estudio, lo que permite ver las tecnologías que soportan y exponer como muchos de los sistemas integran alguna tecnología para sus propios beneficios y de los usuarios.

Ubuntu Server al igual que Debian brindan soporte a los siguientes hipervisores: *XEN*, *KVM (Kernel-based Virtual Machine)*, *VMware*, *VirtualBox*, *QEMU*, además de *LXC* (*Linux Containers*), *Docker* y *OpenVZ* para trabajar en modo contenedor [9] [10] [11] [12]. *KVM* y LXC son las tecnologías de virtualización predeterminadas en el kernel de Linux a partir de la versión 2.6.20 y 2.6.29 respectivamente. Uno de los factores por lo que el soporte es significativo, es el gran vínculo con la computación en la nube, dado que forma parte de la base de la misma [4]. Incluso noticias exponen [13] que *Microsoft* recomienda a *Ubuntu Server* como el SO más completo para el uso de tecnologías *cloud* (en la nube), siendo esta una de las tendencias sobre la cual trabajan últimamente las empresas [4].

Actualmente Red Hat utiliza para realizar los procesos de virtualización la variante *Red Hat Enterprise Virtualization* de forma nativa, que se divide entre un gestor de virtualización y el hipervisor de virtualización

[14] [15]. Red Hat soporta tecnologías de virtualización bases como: OpenVZ [16] [17], Docker [18], Hyper-V, KVM, XEN [19], LXC, entre otras.

1.2.2. Tendencias asociadas a la virtualización en los CPD

A medida que el tiempo transcurre, la administración se centra más en temas como escalabilidad, seguridad y rapidez de las tecnologías, con el objetivo de ofrecer mejores servicios y ser más competitivos. Esto conlleva a realizar innovaciones que les permitan alcanzar de forma ágil los temas en cuestión. Innovaciones que se pueden definir como metas o tendencias que muchos de los CPD del mundo se trazan, para hacerle frente a estos temas. Algunas de estas tendencias están centradas hacia la virtualización, la computación en la nube [2] y la seguridad de los datos.

La virtualización es una de las tendencias que los CPD buscan concretar por los beneficios que ofrece, afirma *Steve Hassell,* presidente de soluciones para centros de datos de *Emerson Network Power.* Los servicios de virtualización, en conjunto con los servicios de nubes, son uno de los aspectos que revolucionarán las industrias de las comunicaciones, señaló el presidente de Negocio de Acceso de Redes Fijas de *Alcatel-Lucent*, *Federico Guillén*. Afirmó además, en el marco del Foro Mundial de Banda Ancha 2015, que la virtualización de los recursos tecnológicos es una de las bases para decidir qué tipo de innovaciones se escalan y pueden ir directamente a la nube, y cuáles pueden permanecer en la red.
Según estudios realizados [4], la virtualización es un factor significativo en el patrón de adopción de la mayoría de las empresas para alcanzar la computación en la nube, siendo otra de las tendencias sobre la cual trabajan últimamente las empresas. Las mismas buscan llegar a la nube en una de sus tres formas, según las necesidades y el entorno sobre el cual operan:

- Privada, como servicios que se brindan dentro de las empresas, donde enfatizan la protección de los datos.
- Pública, se ofrecen servicios de forma gratuita, sin complicaciones, que también pueden ser de pago, según el servicio que se ofrezca, el proveedor de la nube asume las responsabilidades de instalación, gestión, provisión y mantenimiento.

[2] Gartner define la computación en la nube como: “Un estilo de informática en el que unos recursos escalables y flexibles relacionados con las TIC, se ofrecen como servicios al consumidor mediante tecnologías de Internet” [134].

- Híbrida, combinando ambos modelos, de tal forma que las responsabilidades se dividen entre el proveedor y la empresa que brinda el servicio, haciendo uso de la estructura privada para la protección de los datos y la pública para otros beneficios.

Esta última es una de las formas por la que optan la mayor cantidad de las empresas. Según el informe titulado, *"Hybrid Cloud: Benefits, roadblocks, favored vendors"* de Tech Pro Research**,** el 93% del personal encuestado, está familiarizado con el tema de la nube híbrida y un tercio de estos, ya han implementado una solución de este tipo, dejando un 37% del personal, que están considerando la implementación de la nube híbrida [20].

Carlos Perea, Vice Presidente de Ventas para Latinoamérica de *Extreme Networks*, afirma que el cómputo en la nube, en todas sus formas (público, privado e híbrido), está cambiando la manera en que las empresas consumen, administran y cotizan las TIC. También plantea que estos servicios ofrecen una amplia gama de beneficios de menores costos totales y menor complejidad. Según *Gartner* en el 2015, el 50% de las grandes empresas mundiales se basarán en servicios externos de la computación en la nube, como uno de sus 10 más importantes generadores de ingreso. Para el año 2016, todas las grandes empresas globales utilizarán algún tipo de servicio en la nube [21].

En el entorno de la virtualización y la nube en los CPD, se pronostica que la cantidad de carga de trabajo (cantidad de procesamiento que lleva a cabo un servidor para ejecutar una aplicación y soportar cierta cantidad de usuarios que interactúan con la aplicación) por servidor tradicional instalado en los centros aumentará a 2% en 2016 de 1,5% en 2011. Por otra parte, también aumentará la carga de trabajo por servidor en la nube en un 8,5% en 2016 de un 4,2% en 2011, por lo que en 2016 se procesarán en la nube casi dos tercios de todas las cargas de trabajo. Concretamente, el 51% del total de cargas de trabajo será procesado en la nube, frente al 49% que se gestionará en centros de datos tradicionales, tasas que en 2017 serán del 63% y del 37% respectivamente [22].

Una vez que la virtualización se encuentre firmemente planteada en los centros, conjuntamente con la ampliación de las redes y el almacenamiento, la atención recaerá sobre la seguridad de estos activos y la de los datos, esto es predecible por la gran cantidad de violaciones cometidas en los últimos años. Las violaciones están destinadas a quebrantar la seguridad cibernética, incluso las cometidas por *hackers* o individuos que simplemente se conectan a una red sin la debida autorización, con el fin de realizar acciones sobre los datos. Por lo que, otra de las tendencias, va encaminada a mejorar las ofertas de seguridad [23].

Se puede evidenciar que productos, como NSX[3] de VMware, aplican conceptos de seguridad de micro-segmentación que les permiten enfrentar las distintas violaciones informáticas. Por otra parte, se encuentra *DataGravity*, la cual incluye un motor de análisis de seguridad y cumplimiento fuera del nodo, demostrando que es posible aplicar seguridad a la infraestructura tradicional [24]. Los gerentes de las distintas instalaciones trabajan en conjunto con los departamentos de seguridad de TIC con el objetivo de realizar un análisis y evaluación del software y las tecnologías, para así garantizar la seguridad y realizar una evaluación de los mecanismos de seguridad que les brindan los proveedores y los contratistas.

Otra alternativa que los administradores emplean para enfrentar las violaciones y para que la seguridad sea efectiva, es realizar un seguimiento a los servidores e identificar que estos no contengan vulnerabilidades. También realizan una supervisión en tiempo real y suministran protección automática, para así, no solo proteger los datos, sino también las aplicaciones. En conclusión, estas son algunas de las tendencias que han tenido seguimiento desde hace algunos años, desarrollándose desde inicios del 2015 y que continuarán en años posteriores. Así se evidencia la importancia del uso de la virtualización para empresas y CPD en la actualidad.

1.3. Acercamiento a la virtualización

La virtualización es un término antiguo, desarrollado por IBM en 1960 [25], que cumplía con el objetivo de incrementar los niveles de compartición y utilización de los costosos recursos informáticos como eran los *mainframe*[4]. La virtualización en ese momento era una forma simple de particionar los equipos en MV independientes. Las particiones permitían a los *mainframe* ejecutar varias aplicaciones y procesos al mismo tiempo, a través del cual se podía aprovechar al máximo la inversión [26].

La virtualización, como una alternativa para el aprovechamiento de los recursos, se puede definir de varias formas según el entorno en que está siendo aplicada. Cada autor la define desde su punto de vista y en relación a su escenario de trabajo. Según *Bob Muglia*, vicepresidente para servidores y herramientas de negocios de *Microsoft Corporation*, el término puede definirse como: "*Una estrategia para desplegar los*

[3] Software-define networking (plataforma virtual de virtualización de redes para el centro de datos definido por el software [135]).

[4] Mainframe: Es una computadora grande, potente y costosa usada principalmente por una gran compañía para el procesamiento de una gran cantidad de datos.

recursos del ordenador en diferentes capas (hardware, software, datos, red, almacenamiento), aisladas unas de las otras" [26]. *Según Alfonso Ramírez, director general de VMware para España y Portugal, "la virtualización es una capa de abstracción que separa el hardware físico del software que se ejecuta en él, y que permite que múltiples entornos se consoliden sobre una única pieza de hardware como* máquinas virtuales*. De esta forma, se consigue una gran economía de material, electricidad y espacio, así como una mayor flexibilidad en los sistemas de información y una mejor utilización de los recursos de las TIC"* [27].

Para un mejor entendimiento del término, enfocado al tema de la presente investigación, se propone la siguiente definición: "La virtualización es el proceso en donde se abstraen recursos tecnológicos, con el fin de dividirlos en múltiples recursos lógicos para un aprovechamiento completo de sus funciones. Donde estos recursos lógicos divididos operan de forma aislada, minimizando la posibilidad de interferir entre ellos. Dígase recurso: a una plataforma de hardware, un dispositivo de almacenamiento, un SO o algún recurso de la red". Pueden existir diferentes formas de virtualización, según lo que se pretenda virtualizar. Es posible virtualizar el hardware de un servidor, su software, virtualizar sesiones de usuario, aplicaciones y también se pueden crear MV en una computadora de escritorio.

1.4. Análisis de las Técnicas de Virtualización

Cuando se abordan temas referidos a la virtualización, se expone su contenido de manera general, refiriéndose a las técnicas generales como virtualización de plataforma o de recurso, sin mencionar las técnicas derivadas. En relación con el desarrollo de la presente investigación se considera que la forma más adecuada para describir los tipos de virtualización es: la virtualización Completa con Traducción Binaria, Asistida por Hardware, la Paravirtualización y la virtualización a nivel de SO o basada en Contenedores. Los cuales son los tipos de virtualización que comprenden las técnicas de plataformas, también conocida como virtualización de hardware. Este término se define como el proceso de llevar a cabo una encapsulación del hardware físico, es decir, que se oculta el mismo de tal forma que se pueda dividir para compartirse entre múltiples SO huéspedes y pueda ser usado por estos de manera transparente [28].

Un factor significativo y necesario para llevar a cabo el proceso de virtualización, es el uso del Hipervisor (*Hypervisor*), elemento de mayor relevancia que hace posible la virtualización. Este software, también conocido como VMM por sus siglas en inglés *Virtual Machine Monitor*, se encuentra entre el hardware y el huésped, separando el SO y las aplicaciones del hardware, realizando la función de intermediario. Es el

responsable de llevar el control y asignar la cantidad de accesos que los sistemas operativos y aplicaciones tienen al procesador, memoria, disco duro, dispositivos de entrada-salida y conexiones de red [28].

1.4.1. Tipos de Hipervisores

Actualmente los hipervisores se clasifican en dos tipos según el ámbito del sistema en que se ejecutan [28] [29].

Hipervisores tipo uno o *bare metal*: los hipervisores de tipo uno se ejecutan directamente sobre el hardware al mismo nivel que un SO. Son más eficientes ya que tienen acceso directo a los recursos, sin tener que pasar por un SO intermediario. Mayormente son empleados por empresas en el proceso de virtualización de servidores [29] (Figura 1).

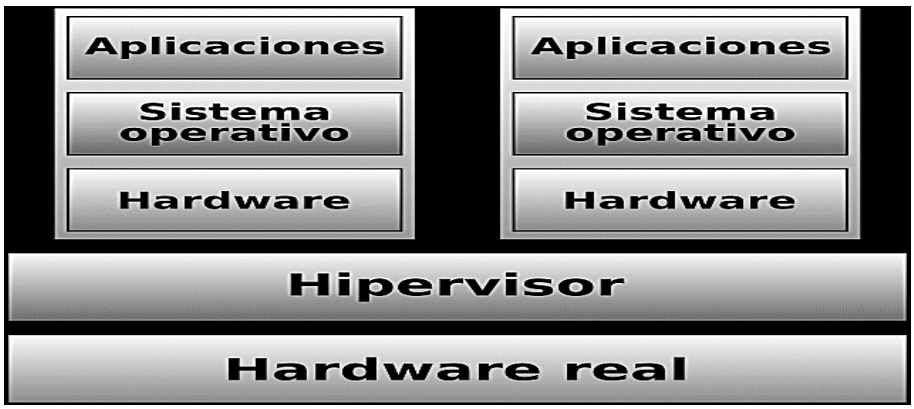


Figura 1 Diagrama de representación conceptual del Hipervisor de tipo uno

Los hipervisores de tipo uno se clasifican en: **monolítico** y ***microkernel*** (Figura 2) [30]**.**

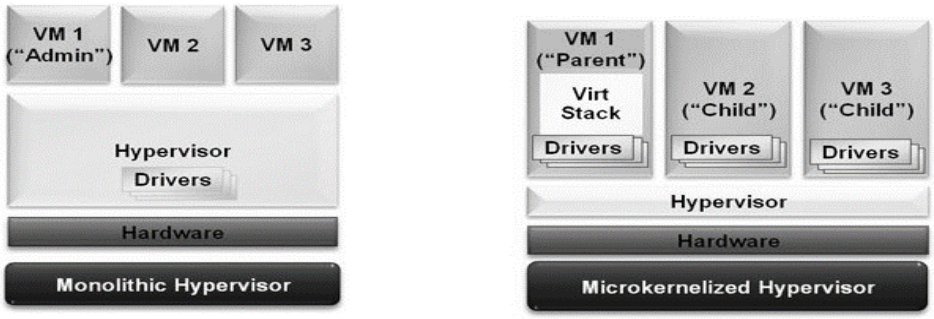


Figura 2 Diagrama de representación conceptual del Hipervisor de tipo uno (Monolítico, Microkernel)

El proceso de virtualización llevado a cabo por un hipervisor monolítico es complejo. Cuando una MV hace una llamada al hardware físico, pasa primero por el hipervisor que es el que tiene el control. El hardware

virtualizado redirige la llamada hacia los *drivers* de dispositivo que operan dentro del hipervisor, donde posteriormente es enrutada (encaminada) por los mismos hacia el recurso real. Por lo que este proceso requiere el desarrollo de *drivers* específicos para cada componente de las máquinas virtuales, además de cambios en el código de las llamadas, lo cual disminuye el rendimiento. También está influido por el exceso de código que se utiliza para realizar los accesos a los recursos físicos, realizados por las máquinas virtuales. Ejemplos de hipervisores monolíticos son los productos *VMware ESX*.

En cambio, los *microkernel* son más estables y con un alto nivel de rendimiento, ya que no utilizan tanto código intermedio, ni tantos cambios en el código de las llamadas. Los *microkernel* son más sencillos, por no necesitar de *drivers* específicos para cada MV, sino que acceden directamente al hardware mediante los mismos *drivers* que utiliza el hipervisor. De este modo, aparece como una capa sencilla dedicada a separar y administrar las diferentes máquinas virtuales. Ejemplo de este tipo es *Hyper-V*.

Algunos de los hipervisores de tipo uno más conocidos son los siguientes:

1. *VMware: ESX/ESXi/ESXi Free.*
2. *XEN.*
3. *KVM*
4. *Citrix XenServer.*
5. *Microsoft Hyper-V Server.*

Hipervisores tipo dos o *hosted*: Los hipervisores de tipo *hosted* se ejecutan, sobre un SO base (Linux, *Windows*, MAC OS), denominado anfitrión, que a su vez se ejecuta sobre el mismo hardware físico. Este tipo de hipervisor como está en una capa más alejada del hardware físico, tiende a tener menos rendimiento que los hipervisores de tipo uno (Figura 3) [29].

Algunos de los Hipervisores de tipo dos más utilizados son los siguientes:

1. *Oracle VirtualBox y VirtualBox OSE.*
2. *VMware: Workstation, Server, Player.*
3. *Microsoft: Virtual PC, Virtual Server.*

Figura 3 Diagrama de representación conceptual del Hipervisor de tipo dos

1.4.2. Técnicas de Virtualización.

A continuación se profundiza sobre los tipos de virtualización más conocidos, evidenciando: plataformas de virtualización, donde sus funcionalidades y métodos de desarrollo se adhieren a las técnicas en cuestión, características y particularidades que las diferencian unas de otras.

- **Virtualización Completa con Traducción Binaria.**

La virtualización completa con traducción binaria es una técnica de virtualización que se realiza mediante el hipervisor. Crea un hardware cercano al nativo, para ejecutar un SO completo, refiriéndose a la carga de todas las interfaces y componentes necesarios para que el SO invitado opere como si estuviera instalado sobre un hardware físico. En esta técnica no se requiere modificar el SO invitado puesto que emplea la traducción binaria. La función de la traducción se basa en analizar reiteradamente las instrucciones del sistema invitado, para interceptar las críticas que son las que se quieren virtualizar y reemplazarlas por un nuevo conjunto de instrucciones. El reemplazo tiene el objetivo de que las nuevas instrucciones tengan el efecto deseado en el hardware virtual. El hipervisor es el encargado de examinar el flujo de instrucciones e identificar las críticas para el control y el comportamiento, una vez interceptadas, procede a reemplazarlas.

La virtualización completa basa su funcionamiento tanto en la traducción binaria, como en la ejecución directa, traducción de las instrucciones críticas y ejecución directa en el hardware de las no críticas. De esta manera, se disminuye la probabilidad de que ocurra un nivel de sobrecarga completo insertado por la expansión de las operaciones [31], promoviendo la eficiencia y la seguridad.

El SO invitado está totalmente desacoplado del hardware subyacente, por lo que el sistema se ejecuta como si empleara un hardware real. El desacoplamiento trae consigo un aislamiento que refleja un alto grado de seguridad, debido al reemplazo de las instrucciones críticas por el hipervisor y emuladas por software.

La traducción es un proceso lento, que influye en la disminución del rendimiento y provoca que este no sea el ideal. Para mejorarlo, la traducción utiliza una caché de código para almacenar las instrucciones en caliente traducidas, pero esto trae como consecuencia un gran consumo de memoria [32].

Con la virtualización completa se puede alcanzar un alto grado de flexibilidad, ya que se ejecutan múltiples SO de diferentes proveedores. Es la única técnica de virtualización que no requiere asistencia por hardware o del SO para alcanzar la virtualización completa y brinda una alta compatibilidad al no requerir modificación los SO *huésped* [33]. Ejemplo, *VMware Workstation* y *Oracle VM VirtualBox*.

- **Virtualización Asistida por Hardware**

Antiguamente los a servidores basados en arquitecturas de 32 bits no eran desarrollados con fines de virtualización, por lo que a partir del año 2006, empresas desarrolladoras de microprocesadores como Intel y AMD[5], crearon alternativas para hacerle frente al problema. Las alternativas, se basan en la inserción de un conjunto nuevo de instrucciones en la arquitectura del procesador facilitando el proceso de virtualización. Tanto *Intel* como *AMD*, proporcionaron nuevas tecnologías capaces de fomentar estas alternativas, nombrando cada uno a sus nuevos productos. *Intel* nombró a su nueva generación de tecnologías (microprocesadores) como Intel-VT (*Virtualization Technology*), VT-x para 32 bit y VT-y para arquitecturas de 64 bit, en el caso de *AMD*, con su tecnología AMD-V (*Virtualization*) [33] [34]. En general aunque el soporte para ambas es muy distinto, en principio, no lo es, ya que no son directamente compatibles, pero sí brindan las mismas funcionalidades, solo que de acuerdo al tipo de Unidad Central de Procesamiento (CPU).

Esta técnica tiene similitud con la virtualización completa, ya que los ordenadores antiguos de 32 bits, no se desarrollaban con funciones de virtualización. Por esta razón surgían problemas en el momento de realizar el proceso de virtualización, ya que no se capturaban algunas instrucciones que debían ser manejadas por el hipervisor. Por lo que también se puede encontrar a la virtualización asistida por hardware, como

[5] ADVANCED MICRO DEVICES

virtualización completa o nativa con apoyo del hardware. Los entornos paravirtualizados pueden ejecutar SO sin modificar, brindando por tanto una alta compatibilidad. Esto permite simplificar las técnicas de virtualización, de modo que se eliminaría la necesidad de que el hipervisor realice traducción binaria sobre los sistemas o paravirtualización [34] [35] [8]. De esta forma, se mejora el rendimiento con respecto a la virtualización completa con traducción binaria, aunque se limita la escalabilidad por el uso excesivo del procesador para alcanzar la virtualización ya que para llevar a cabo el proceso de virtualización asistida por hardware, los CPU deben de soportar la extensión Intel-VT o AMD-V.

Para un mejor entendimiento de cómo funciona la técnica asistida por hardware, a continuación se realiza una breve explicación. Las arquitecturas del hardware principalmente de 32 bits, ejecutan las instrucciones por niveles o anillos de privilegios, desde el cero hasta el nivel 3 con menos privilegio (Figura 4). El anillo 0 es destinado para el núcleo del SO, el 2 y 3 para los controladores de dispositivos de entradas y salida (I/O), y por último el nivel 3 destinado para las aplicaciones de usuario. La inserción de las nuevas instrucciones a la CPU, permite la creación de un nuevo nivel denominado anillo -1 en el cual se ejecuta el hipervisor. Esto permite que las máquinas virtuales creadas por este se ejecuten en el nivel cero, obteniendo el control y el máximo privilegio, además de separar los restantes niveles de las operaciones de virtualización [8].

En esta técnica de virtualización se garantiza una alta seguridad y aislamiento ya que el hipervisor asigna de forma única los distintos recursos a los sistemas, decayendo en flexibilidad con respecto a la paravirtualización ya que no pueden aprovechar las ventajas existentes cuando los sistemas tienen control de que están siendo virtualizados [36].

Aunque la mayoría de los CPU actualmente soportan la virtualización asistida por hardware, es de suma importancia saber cuándo este recurso lo permite, ya que no se puede llevar a cabo dicha técnica si no es así. También hay que tener en cuenta que el Sistemas de Entrada y Salida Básico (BIOS) lo permita [37]. Ejemplo *KVM*.

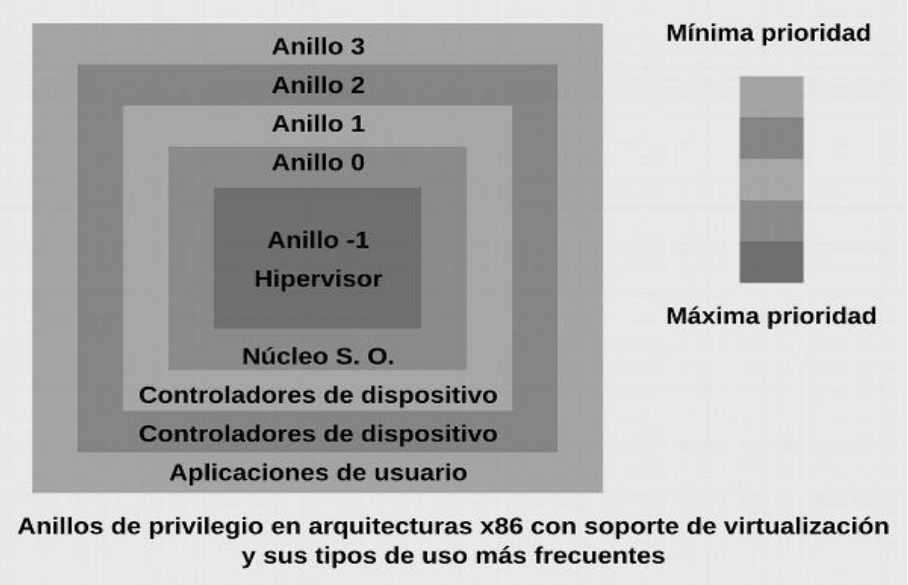


Figura 4 Anillos de privilegio en arquitecturas x86.

- **Paravirtualización**

En los ordenadores, normalmente el software principal es el SO ya que controla el acceso a recursos de hardware y el uso compartido de la CPU entre las aplicaciones. El control se puede realizar debido a que los ordenadores modernos soportan cuatro niveles de privilegios. Desde el nivel cero, donde se ejecuta el SO encargándose de supervisar y controlar los recursos, hasta el nivel 3, siendo el menos privilegiado. La técnica de Paravirtualización se basa en insertar el hipervisor en el nivel cero, con el objetivo de que este pase a tener el máximo control sobre los recursos. Al igual que otros tipos de virtualización, permite la ejecución de múltiples sistemas, solo que estos deben estar modificados. Es decir, implica modificar el kernel de los sistemas huéspedes, para utilizar estructuras denominadas *hypercalls* que se comunican directamente con el hipervisor. Esto brinda seguridad, estabilidad a los sistemas y una mejor comunicación entre el hipervisor y el huésped, al sustituir las *hypercalls* por cada instrucción que no pueden ser virtualizada como las relacionadas con la administración de memoria e interrupciones y preservación de tiempo. Esto conlleva a que los sistemas operativos huéspedes sepan que están siendo virtualizados, por lo que contribuyen a que el rendimiento sea óptimo y brinda una alta escalabilidad.

Al ser necesario que se modifiquen los kernel de los huéspedes, esta técnica de virtualización solo se puede utilizar en un kernel de código abierto, por lo que no se pueden virtualizar sistemas *Windows*, lo cual reduce la compatibilidad y flexibilidad con dichos sistemas [36] [33].

La mayoría de los productos de virtualización tienen una arquitectura de paravirtualización, ya que comparado con otros tipos de virtualización es más sencilla y práctica. Por ejemplo, comparado con la completa con traducción binaria es mejor en rendimiento, debido a que la aceleración de la traducción es un proceso difícil y lento [32]. Ejemplos XEN, *User-mode Linux* (UML).

- **Virtualización a nivel de Sistema Operativo**

La virtualización a nivel de SO o basada en contenedores, es una tecnología de virtualización de servidor que trabaja a nivel de SO. Esta técnica se encarga de abstraer algunos servicios del sistema como pueden ser sistemas de archivos, dispositivos, la red y la seguridad, para así poder proveer un entorno virtualizado a las aplicaciones nativas. El servidor y una instancia del SO son virtualizados en múltiples particiones aisladas. Por ejemplo, podría ejecutarse PostgreSQL en una partición, y un servidor web como Apache en otra instancia, optimizando cada instancia por separado en directa relación a la aplicación que allí se ejecuta. Cada uno de los huéspedes o contenedores tienen espacios de nombre diferentes y comparten el mismo kernel, que también es el mismo sobre el cual se ejecuta el sistema anfitrión.

Este acercamiento brinda un rendimiento muy cerca del nativo, debido a que no es necesario duplicar los recursos del hardware [38] [39]. Los entornos virtuales comparten la misma instancia del kernel con lo que se alcanza una alta escalabilidad, característica que le permite al sistema físico soportar más servidores virtuales que las MV completas que pudiera virtualizar. Los contenedores no son sistemas operativos completos con todas sus aplicaciones, funcionalidades o interfaces; son una instancia del SO anfitrión que internamente contiene la aplicación que se quiere ejecutar y todas las dependencias que son necesarias para la ejecución de la misma. Se puede referir a los contenedores, como máquinas virtuales ligeras, menos exigentes, con los *chips* y memorias de los equipos donde se ejecutarán [40].

Para muchos resulta una interrogante la relación entre el término contenedor y máquina virtual. Puede existir una similitud entre ambos términos ya que estos utilizan un sistema base para poder realizar sus funciones, como otorgarles los recursos necesarios a los sistemas huéspedes para su ejecución. Pero hay que tener presente que ambos términos no significan lo mismo, la gran diferencia radica en los sistemas huéspedes.

La máquina virtual contiene todo un SO, inclusive puede ser diferente que el anfitrión, mientras que el contenedor aprovecha el sistema sobre el cual se está ejecutando, siendo una instancia del mismo.

Ejemplo, para virtualizar una base de datos sobre una máquina virtual se requiere:

- Primeramente el hardware o servidor físico, ejerciendo la función de base.
- Un SO que se ejecute sobre la máquina física.
- Un sistema de virtualización o hipervisor que es el encargado de la asignación de los recursos.
- Un SO invitado bajo el hipervisor, que tiene que ser completo, ya que no puede obtener recurso del kernel de su anfitrión.
- Y por último instalar bajo el sistema invitado el motor de bases de datos y todas sus dependencias.

Mientras que por parte de los contenedores solo se requiere:

- Una máquina física, que también puede ser virtual.
- Un SO sobre la máquina.
- Un motor que permita crear contenedores, ya instalado sobre el sistema.
- Un contenedor basado en una imagen que contenga el motor de base de datos y todas sus dependencias.

Existe un nivel de diferencia entre ambos términos, se puede apreciar que en el sistema de contenedores no necesariamente debe existir una máquina física, pudiendo ser una máquina ya virtualizada. Por lo que no tendría sentido el uso de un hipervisor, ahorrando recursos y aprovechando al máximo los que están en uso, aumentando así el rendimiento. Esto conlleva a que, como la máquina está virtualizada, se puede adaptar rápidamente y con facilidad a diferentes actividades y cambios.

Cada máquina virtual emula todo un SO, en el caso que se deseara poner en marcha una segunda base de datos como réplica, implicaría destinar más recursos hacia un nuevo sistema. Mientras que con los contenedores, solamente se lanzaría otra instancia mucho más ligera, lo que posibilita un mejor aprovechamiento de los recursos.

Un punto de vista importante es la facilidad y capacidad de ejecución de un contenedor. Por ejemplo, el tiempo que se demora en iniciar un contenedor equivale al mismo tiempo que se demora en iniciar un proceso en el sistema. En cambio para que inicie una máquina virtual, se debe esperar a que lo haga también el SO subyacente, por lo que se estaría ganando en agilidad y rapidez.

Expuestas estas razones quedan esclarecidas las diferencias entre los términos: máquina virtual y contenedores [41].

La virtualización a nivel de SO también, permite simular que las aplicaciones se están ejecutando sobre sistemas diferentes, cuando en realidad están compartiendo recursos con otros contenedores. Esto tiene como ventaja que las aplicaciones se ejecuten de forma aislada, lo que evita que interfieran unas con otras. Aunque hay que tener en cuenta que el aislamiento es pobre ya que cuando un huésped consuma demasiados recursos, los otros se verán afectados.

Uno de los inconvenientes que posee esta tecnología, es que solo se pueden ejecutar entornos virtuales para la misma CPU y SO. Esto provoca que si ocurre algún fallo en el procesador, todos los entornos se verían afectados, evidenciándose entonces el bajo nivel de seguridad. También se ve afectada la compatibilidad ya que solo se pueden ejecutar SO Linux que compartan el mismo kernel del sistema principal. Aunque no es necesario ningún requerimiento de hardware en este tipo de virtualización, solo es recomendable para infraestructuras de software homogéneas, limitando de esta forma la flexibilidad del mismo [38]. Ejemplos de algunas plataformas, LXC, *Docker*, *OpenVZ*, Linux-VServer etc.

1.5. Conclusiones parciales

En el presente capítulo se determina que las plataformas de virtualización que aplican a Nova Servidores son las que dependen de un SO anfitrión para su funcionamiento.

A partir del análisis de las técnicas de virtualización se llega a la conclusión de que:

La virtualización completa con traducción binaria propone un bajo índice de rendimiento y escalabilidad, lo que determina que se use en entornos donde las cargas de trabajo no sean elevadas.

La virtualización con soporte de hardware, se identifica como una forma para reducir las carencias de la técnica de virtualización completa, manteniendo las ventajas que esta trae consigo.

La Paravirtualización alcanza rendimientos admisibles con gran escalabilidad y seguridad, mientras que se ve vigente el déficit de compatibilidad y flexibilidad por el uso de SO modificados.

El alto rendimiento y escalabilidad son factores claves de la virtualización a nivel de SO, presentando además bajo grado de seguridad, dado que un problema en el kernel, traería consigo una afectación en los demás entornos.

CAPÍTULO 2. **CRITERIOS Y PRUEBAS PARA LA SELECCIÓN DE PLATAFORMAS DE VIRTUALIZACIÓN**

En el actual capítulo se realiza un análisis sobre la tesis titulada **"Criterios y Pruebas para la Selección de Plataformas de Virtualización para Nubes Privadas con Soporte para IaaS"** y la **ISO 9126-1**, en vista de determinar factores significativos para el desarrollo del objetivo de la tesis. Los distintos elementos permitirán establecer variables a tener presente en las plataformas de virtualización, métricas que cuantifiquen cada una de las variables y pruebas que permitan evaluar el desempeño de las plataformas. Tanto las pruebas como los restantes factores permiten comprender su completo funcionamiento, además del cumplimiento de las tareas efectuadas por estos. Como parte del desarrollo del actual capítulo también se plasman los distintos hipervisores sobre los cuales se realizan las pruebas y una breve descripción, resaltando características, funcionalidades, ventajas y limitaciones de cada uno.

2.1. Justificación de los Criterios, Métricas y Pruebas utilizadas

En el año 2015, en el Instituto Superior Politécnico José Antonio Echeverría de la Habana, se realizó una investigación por el ingeniero Ariel Lee Estévez, la cual tiene como título **"Criterios y Pruebas para la Selección de Plataformas de Virtualización para Nubes Privadas con Soporte para IaaS[6]"** [42]. Como su nombre lo indica, se establecen criterios basados en indicadores y métricas para la selección de hipervisores, que atribuyan al diseño de nubes privadas con soporte para IaaS. Para esto se realizó una propuesta de los Requerimientos No Funcionales (RNF) de las plataformas de virtualización que tributan a los requerimientos de la nube privada. Se realizaron análisis sobre la influencia de los diferentes indicadores sobre los RNF de las plataformas, evaluando cada indicador como requerido, recomendado u opcional, en vista de fomentar la selección. Se evidencia la realización de cada análisis realizado a través de tablas plasmadas durante el desarrollo de la tesis. A continuación de los distintos análisis, se establecen métricas que avalan cuantitativamente al cumplimiento de los RNF por las plataformas.

Como parte de la investigación se realiza una propuesta de pruebas a seguir en aras de evaluar el desempeño de hipervisores. Las diferentes pruebas analizan de forma exhaustiva cuatro subsistemas principales de un sistema de cómputo: CPU, memoria, red y disco.

[6] Infraestructura como Servicio.

A partir del análisis antes expuesto, se considera que en el proceso de desarrollo de la investigación de Ariel Lee Estévez se establecen factores claves, que atribuyen a la obtención de un resultado exitoso, al mismo tiempo que se respalda todo lo expuesto durante la investigación. Se toma en cuenta la utilización de algunos de los factores, que en conjunto con características definidas por el estándar internacional para la evaluación de la calidad del software **ISO 9126-1** [43] y criterios del autor de la presente investigación permiten establecer variables a tener en cuenta en una plataforma de virtualización, métricas y pruebas. Una o varias métricas están asociadas a una variable, lo que permite cuantificarla o cualificarla. Las pruebas por otra parte otorgan una evaluación a las métricas y permiten comparar las capacidades de los hipervisores.

Como variables determinadas a raíz del análisis de la tesis del ingeniero Ariel Lee Estévez se encuentran: escalabilidad, rendimiento, aislamiento, tiempo de respuesta, compatibilidad y fiabilidad. Las variantes restantes no se tienen en cuenta por que están definidas específicamente para evaluar hipervisores en escenarios de nubes privadas con IaaS y no para entornos tradicionales. También se determinaron las variables portabilidad y seguridad procedentes del estándar ISO 9126-1, desechando las restantes por dos principales razones:

- Existen variables del estándar que se solapan con otras ya establecidas en la investigación ejemplo la eficiencia, en donde se puede plantear que una plataforma es eficiente si atribuye un elevado rendimiento, es fiable y los tiempos de respuesta a solicitudes son mínimo.
- Existen variables que su evaluación no ofrecen resultados significativos para el objetivo de la investigación ya que no atribuyen a determinar si una plataforma es la más adecuada o completa para nova servidores 5.0. Ejemplo la usabilidad, enfocada a la capacidad del producto de ser entendido, aprendido, usado y ser atractivo para el usuario.

En la presente investigación se define además como variable a tener en cuenta en un hipervisor, la plataforma de gestión, variante significativa ya que muchos de los hipervisores sin un software para gestionar las MV se limitan a usar solo funciones específicas (crear, eliminar, entre otras). La migración o el balanceo de carga en caliente son funciones que algunos hipervisores soportan, pero sin el uso de una plataforma resulta complicado. Mientras que otros hipervisores no lo soportan y solo por medio de un software de gestión pueden realizar las funciones de migración y balanceo.

2.1.1. Descripción de las variables

Rendimiento: Hace referencia a la capacidad que tenga la plataforma de virtualización para hacer un uso eficiente de los recursos de hardware disponibles de tal forma que permita alcanzar un desempeño igual o cercano al nativo.

Tiempo de Respuesta: Tiempo de demora de una determina operación desde que inicia hasta que se obtiene una respuesta.

Aislamiento: Capacidad de los hipervisores de aislar el acceso a los recursos de los sistemas huésped de tal forma que un huésped no intervenga en el funcionamiento de otros. Se tiene en cuenta además la capacidad del hipervisor de limitar el acceso a objetos lógicos (archivos, direcciones de memoria virtual, números de puertos, identificadores de procesos y otros) de una MV.

Portabilidad: Esta variable está sujeta al despliegue sencillo de aplicaciones virtuales entre múltiples plataformas de virtualización. Es decir la capacidad del producto para ser migrado de un entorno a otro.

Escalabilidad: Capacidad del hipervisor de desplegar la mayor cantidad posible de sistemas huésped.

Seguridad: Compuesto por la inclusión de características de cortafuegos, detección de intrusos y control de acceso como rasgos de seguridad adicional y el monitoreo de la red en busca de paquetes con anomalías. Además de que implemente mecanismos de modo que si ocurre un ataque a una máquina virtual, el sistema anfitrión no quede expuesto.

Compatibilidad: La compatibilidad como atributo se refiere al soporte que tiene la arquitectura para diferentes estándares, tecnologías, y sus evoluciones con el propósito de ser compatible con diferentes proveedores de hardware y soluciones de software. Es decir que la plataforma sea multiplataforma, además de que permita crear máquinas virtuales con SO de diferentes proveedores.

Fiabilidad: Es la habilidad de proporcionar un conjunto de servicios en los que se pueda confiar de manera justificada.

Plataforma de Gestión: Capacidad del hipervisor de poder ser administrado por múltiples herramientas. En cada software de gestión se tiene en cuenta el máximo y la calidad de funcionalidades que ofrece, además de la usabilidad.

2.2. Relación entre variables y técnicas de virtualización

En la tabla 1 se muestran las variables a medir de las plataformas de virtualización y la influencia que tienen sobre las técnicas de virtualización. Para las evaluaciones que cuantifican la influencia de los tipos de virtualización sobre las variables, se tuvo en cuenta la descripción de las técnicas de virtualización y las plataformas en conjunto con criterios del autor de la presente investigación y bibliografías que respaldan los datos mostrados [44]. Muchos de los resultados coinciden con los expuestos en el trabajo de Ariel Lee Estévez [42]. La evaluación va desde el “1” al “3”, siendo “1” el valor más bajo y “3” el más alto en la evaluación. El signo “-” indica que no se emitió ningún criterio al respecto ya que la comparación carecía de sentido en ese caso en particular. De forma general la relación de las variables y las técnicas de virtualización permiten apoyar la selección del hipervisor al exponer el comportamiento de las variables en las técnicas de virtualización.

Tabla 1 Variables a medir de los hipervisores y su influencia en las técnicas de virtualización.

Variables / **Operación**	Escalabilidad	Rendimiento	Tiempo de Respuesta	Compatibilidad	Portabilidad	Fiabilidad	Seguridad	Aislamiento	Plataforma de gestión
Completa con Traducción Binaria	1	1	1	3	3	3	3	3	-
Contenedores	3	3	3	1	1	2	2	2	-
Hardware	2	2	2	3	2	3	3	3	-
PV	3	3	3	2	2	3	3	3	-

2.3. Métricas para la evaluación de las variables de las plataformas de virtualización

Las métricas que se plasman a continuación, permiten evaluar las plataformas seleccionadas en aras de obtener la indicada para Nova Servidores. En la tabla 2 se muestran cada una de las métricas en conjunto con una breve descripción.

Tabla 2 Métricas para la evaluación de las variables

Métrica	Descripción
CUMPLIMIENTO_INDICADORES	Capacidad del hipervisor de cumplir o tener presente una determinada variable.
DEMORA_OPS_CPU	Tiempo que demora la ejecución de una tarea en específico.
VEL_OPS_CPU	Velocidad de ejecución de instrucciones y operaciones relacionadas con el CPU.
FLOPS	Razón de ejecución de operaciones con coma flotante por segundo.
IOPS	Velocidad promedio en Operaciones por Segundo (OPS) de las operaciones de lectura/escritura (L/E). Se define en función del número de operaciones de L/E durante el tiempo de observación.
THROUGHPUT	Cantidad de datos libres de errores transferidos por unidad de tiempo.
VEL_OPS_MEM	Velocidad de operaciones de acceso a memoria por segundo.
LATENCIA	Tiempo que demora la realización de una operación desde que está lista para realizarse hasta su cumplimiento.
JITTER	Variación de la latencia. Cambios en la demora de realización de operaciones.
PACKET_LOSS	Cantidad de paquetes perdidos en un flujo de red determinado.

Para un mejor entendimiento de la relación existente entre las métricas y las variables, se muestra a continuación la tabla 3 que refleja las variables a medir de las plataformas y las métricas asociadas a cada una.

Tabla 3 Métricas asociadas a las variables de las plataformas

Variables	Métricas	Variables	Métricas
Escalabilidad	CUMPLIMIENTO_INDICADORES	Rendimiento	VEL_OPS_MEM, VEL_OPS_CPU, IOPS, THROUGHPUT, FLOPS
Compatibilidad	CUMPLIMIENTO_INDICADORES	Tiempo de Respuesta	IOPS, JITTER DEMORA_OPS_CPU, LATENCIA
Portabilidad	CUMPLIMIENTO_INDICADORES	Seguridad	CUMPLIMIENTO_INDICADORES
Fiabilidad	PACKET_LOSS	Plataforma de Gestión	CUMPLIMIENTO_INDICADORES
Aislamiento	CUMPLIMIENTO_INDICADORES		

2.4. Pruebas para la evaluación de plataformas de virtualización

Como parte de la evaluación de las plataformas y en vista de una adecuada selección se establecen distintas pruebas que evalúan cuatro de los principales subsistema de un hardware, dígase CPU, RAM, red y disco duro. Cada una de estas pruebas es establecida con el objetivo de evaluar el desempeño de los hipervisores. A la vez que permiten otorgarle un valor a las métricas establecidas con anterioridad y que atribuyen a las variables de las plataformas.

2.4.1. Tecnologías a utilizar

Para realización de las pruebas se hace uso de la *Suite Phoronix* (pts) y las herramientas Netperf e Iperf para medir el subsistema de red.

- **Phoronix Test Suite**

Phoronix es diseñado por Michael Larabel, dueño de la empresa *Phoronix Media*, incluyendo el sitio web con el mismo nombre, como una aplicación independiente de la plataforma que soporta sistemas Linux, BSD, Solaris, Microsoft Windows y Mac OS X. Es una plataforma de *Testeo (pruebas) y benchmarking*

(comparación), la cual permite realizar diversas pruebas automáticas, como rendimiento, además de brindar la función de comparativa de resultados entre componentes de software y hardware (tarjetas gráficas, discos duros, unidades de estado sólido, monitores, procesadores, Memoria de Acceso Aleatorio (RAM) y tarjetas de sonido).

El programa se integra con el sitio web *OpenBenchmarking.org,* permitiéndoles a los usuarios subir los resultados al sitio y poder comparar los resultados obtenidos con otras fuentes vigentes en la web. Existen dos formas de poner en funcionamiento la herramienta y realizar las pruebas: haciendo uso de líneas de comandos se pueden realizar las pruebas de forma manual y de forma gráfica. Las pruebas arrojan resultados cuantitativos y otros cualitativos, como imágenes de calidad o fallos ocurridos. También brinda algunos datos de los sistemas como pueden ser: consumo de energía del sistema y uso de disco [45] [46].

- **Netperf**

Netperf es una herramienta de redes informáticas que permite evaluar y comparar el rendimiento en las redes a partir de la transferencia de datos mediante los protocolos TCP y UDP. Es una herramienta de software libre desarrollada originalmente por la compañía Hewlett Packard [47].

- **Iperf**

Iperf es una herramienta que se utiliza para hacer pruebas de rendimiento en red mediante la creación de flujos de datos TCP y UDP. Es un software de código abierto escrito en el lenguaje de programación C++ y desarrollada por el Equipo de Soporte de aplicaciones distribuidas (DAST) en el Laboratorio Nacional de Investigación Aplicada de red (NLANR) [48] [49].

2.4.2. Prueba 1: Subsistema de CPU

El objetivo de la prueba, es la evaluación del poder y capacidad de trabajo del microprocesador en la ejecución de instrucciones en relación con el tiempo de demora. Las pruebas se ejecutan sobre los diferentes hipervisores en vista de determinar el comportamiento y la forma de manejo ante la realización de varios procesos. En la tabla 4 se describen las pruebas utilizadas para la evaluación de la CPU, además de las métricas a la que atribuyen y los criterios de aceptación.

Tabla 4 Herramientas en la sub-suite CPU

Nombre y Versión	Descripción	Métricas [Unidades]	Criterios de Aceptación
GraphicsMagick (pts/graphics-magick-1.6.1)	Esta es la prueba de *GraphicsMagick* con su implementación de OpenMP[7] que realiza varias pruebas con imágenes para estresar el CPU del sistema [50] [51].	**VEL_OPS_CPU** [Iteraciones por minuto]	**Mayor** es mejor
FLAC[8] Audio Encoding (pts/encode-flac-1.5.0)	Esta prueba mide cuanto tiempo es necesario para codificar una muestra en *Waveform Audio File* (WAV) en formato FLAC [52] [53].	**DEMORA_OPS_CPU** [Segundos]	**Menor** es mejor
C-Ray (pts/c-ray-1.1.0)	Esta es una prueba de C-Ray, un simple trazador de líneas designado para medir los FLOPS del CPU. Esta prueba es multiproceso (16 procesos por núcleo), trazará 8 líneas por pixel y generara una imagen con resolución de 1600x1200 [54] [55].	**DEMORA_OPS_CPU** [Segundos]	**Menor** es mejor
Tachyon (pts/tachyon-1.1.1)	Esta es una prueba que ejecuta un trazado de líneas paralelas [56] [57].	**DEMORA_OPS_CPU** [Segundos]	**Menor** es mejor
Himeno Benchmark (pts/himeno-1.1.0)	El *benchmark* Himeno calcula la solución para los sistemas de ecuaciones de presión de Poisson usando el método Point-Jacob [58].	**FLOPS** [MFLOPS[9]]	**Mayor** es mejor

2.4.3. Prueba 2: Subsistema de Disco Duro

La actual prueba presenta como objetivo, la evaluación del rendimiento del disco duro en cada uno de los hipervisores seleccionados. Para esto se hace uso de las herramientas establecidas en la tabla 5.

[7] Open Multi-Processing (Multi-Procesamiento Abierto).
[8] Free Lossless Audio Codec (Códec gratis de audio sin pérdidas).
[9] Mega FLOPS.

Tabla 5 Herramientas en la sub-suite Disk

Nombre y Versión	Descripción	Métrica [Unidades]	Criterios de aceptación
Gzip Compression (pts/compress-gzip-1.1.0)	Mide el tiempo necesario para comprimir un archivo utilizando el formato Gzip [59] [60].	**IOPS** [Segundos]	**Menor** es mejor
PostMark (pts/postmark-1.1.0)	Esta es una prueba para simular el acceso a servidores Web y de correo. El programa ejecuta 25000 transacciones con 500 archivos simultáneos con tamaños variables entre 5 y 512 Kilo Bytes (KB) [61].	**THROUGHPUT** [Transacciones por segundo (TPS)]	**Mayor** es mejor
IOzone (pts/iozone-1.8.0)	Comprobación del rendimiento del disco duro y del sistema de archivos [62] [63].	**IOPS** **THROUGHPUT** [Kb/s]	**Mayor** es mejor
FS-Mark (pts/fs-mark-1.0.0)	Prueba diseñada para medir el rendimiento del sistema de archivos [64].	**IOPS** [Archivos/s]	**Mayor** es mejor
Compile Bench (pts/compilebench-1.0.1)	Simula el envejecimiento del sistema de archivos, mediante la creación de archivos y mide como el sistema mantiene la localización de los directorios a medida que el disco se llena.	**THROUGHPUT** [MB/s]	**Mayor** es mejor

2.4.4. Prueba 3: Subsistema de RAM

El objetivo de las siguientes pruebas es evaluar las capacidades del hipervisor, mediante el acceso a la memoria RAM. La tabla 6 muestra las pruebas y métricas que intervienen en el proceso.

Tabla 6 Herramientas en la sub-suite Memory.

Nombre y Versión	Descripción	Métrica [Unidades]	Criterios de aceptación
RAMspeed SMP[10] (pts/ramspeed-1.4.0)	Prueba del rendimiento de la memoria RAM [65] [66]	**VEL_OPS_MEM** [MB/s]	**Mayor** es mejor
STREAM[11] (pts/stream-1.2.0)	Prueba del rendimiento de la memoria RAM [67] [68].	**VEL_OPS_MEM** [MB/s]	**Mayor** es mejor
CacheBench (pts/cachebench-1.0.0)	Es una prueba de rendimiento designada para probar la memoria y el ancho de banda de la cache [69] [70].	**VEL_OPS_MEM** [MB/s]	**Mayor** es mejor

2.4.5. Prueba 4: Subsistema de Red

La tabla 7 muestra una de las herramientas que se utilizan para evaluar el subsistema de red, una breve descripción de la misma y la métrica asociada

Tabla 7 Herramientas en la sub-suite Red

Nombre y Versión	Descripción	Métrica [Unidades]	Criterios de aceptación
Network loopback **pts/network-loopback-1.0.1**	Esta prueba mide el rendimiento del adaptador de red en bucle utilizando un micro-punto de referencia para medir el rendimiento de TCP [71]	**THROUGHPUT** [Segundos]	**Menor** es mejor

[10] Symmetric Multiprocessing (Multiprocesamiento Simétrico).
[11] Sustainable Memory Bandwidth in High Performance Computers (Ancho de Banda Sostenible en Computadores de Alto Rendimiento).

2.4.6. Prueba 6: Evaluación del subsistema red mediante Netperf-Iperf

Las herramientas descritas en la tabla 8 ofrecen pruebas que tienen el objetivo de evaluar el subsistema red, en vista de analizar su comportamiento cuando se aumentan las cargas de trabajo.

Tabla 8 Evaluación del sub-sistema Red

Nombre y Versión	Descripción	Métrica [Unidades]	Criterios de aceptación
Netperf (netperf 2.6.0)	Es una herramienta de red que puede ser utilizada para la comprobación de ancho de banda y latencia en la red [72] [47] [49].	**THROUGHPUT** [MBits/s]	**Mayor** es mejor
		LATENCIA [Microsegundos]	**Menor** es mejor
Iperf (iperf 2.0.5)	Iperf es una herramienta de SLCA que puede ser utilizada para medir la variación de la demora de red y pérdida de paquetes [73] [48] [74].	**JITTER** [Microsegundos]	**Menor** es mejor
		PACKET_LOSS [% de Paquetes Perdidos]	**Menor** es mejor

2.4.7. Modo de ejecución de las pruebas

La ejecución de las pruebas se realiza de la siguiente forma:

1. Instalar el hipervisor sobre el hardware disponible.
2. Desplegar una MV en el sistema anfitrión.
3. Ejecutar la prueba sobre la MV desplegada utilizando las herramientas antes descritas.
4. Registrar los valores obtenidos.
5. Analizar los resultados.

2.5. Plataformas de virtualización seleccionadas para el estudio

En este epígrafe se realiza un análisis de un conjunto de hipervisores, determinando los que resultan ser los máximos exponentes en el ámbito del software libre a partir de una variedad existente en el mercado y un grupo de indicadores que a continuación se definen.

Indicadores de selección [75] [76]:

Posición en el mercado del fabricante: Posición en el mercado de los fabricantes de tecnologías de virtualización y los líderes (Figura 5, cuadrante de *Gartner*[12]*)* [77]. Este indicador permite determinar el nivel de fortaleza y madurez de los hipervisores en la virtualización a partir de la posición de los fabricantes en el mercado.

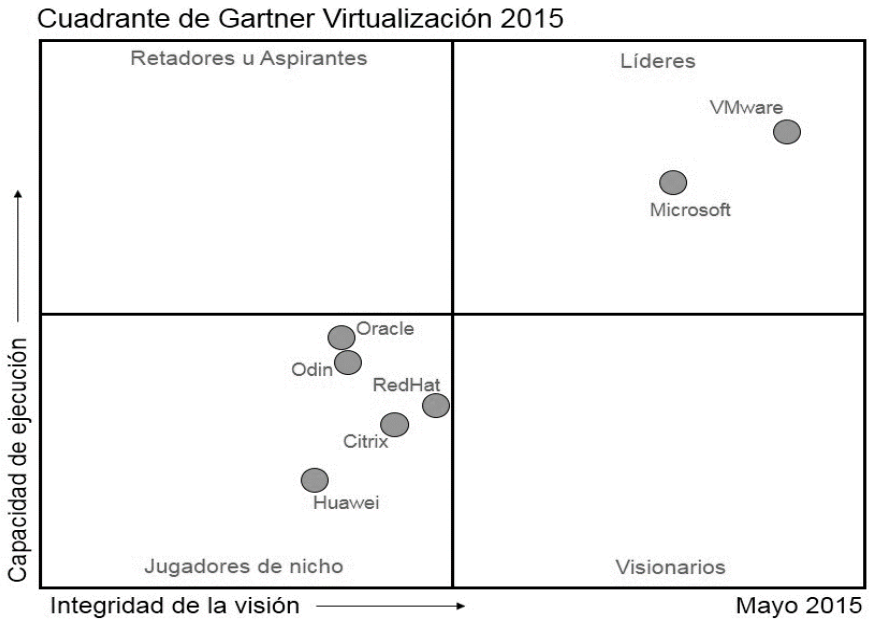


Figura 5 Cuadrante de Gartner 2015

Soporte Comercial: Tiempo de soporte garantizado por el proveedor, comunidades de usuarios y tiempo mínimo de notificación de retiro del mercado.

[12] Gartner: Empresa consultora y de investigación de las tecnologías de la información

Versión de la plataforma: Se evidencia una versión estable de la plataforma, con la que se estará trabajando, soportada por Nova Servidores 5.0 en su versión del kernel 3.16.
Tipo de Licencia de la plataforma: La licencia del hipervisor aporta de forma razonable a la facilidad económica, siendo este uno de los requerimientos sobre el cual se desea trabajar.
Sistema Operativo: El SO es parte de la base en que se desarrolla la tesis en cuestión, ya que se requiere un SO Linux anfitrión. Este parámetro se refiere a la utilización de un hipervisor de tipo dos.

Los diferentes hipervisores seleccionados ofrecen alternativas en SLCA, mientras que su uso está sujeto a la utilización de un SO anfitrión [78]. Unos han sido reconocidos como modelos en la realización de una determinada función, al contrario de otros que llegan a ser completos. De esta forma se puede decir que dentro del amplio conjunto de software de virtualización, los seleccionados son los más reconocidos por la gran variedad de funcionalidades que ofrecen [36]. A continuación se muestra en la tabla 9, los hipervisores seleccionados.

Tabla 9 Hipervisores

Nombre	Método de Operación	SO anfitrión	Licencia	Versión de la plataforma	Soporte Comercial
Linux Containers (LXC)	Contenedores	Linux	GPL [13]versión 2	Versión de LXC 1.0.8 [79], versión mínima recomendable del kernel 3.10	Si
Docker	Contenedores	Linux	Apache 2.0	Versión mínima del kernel 3.10, versión de docker 1.6.2 [80]	Si

[13] Licencia Pública General.

KVM	AMD-V and Intel-VT Soporte de Hardware, Completa	Linux, FreeBSD	GPL versión 2	Nativa del kernel a partir de la versión 2.6.20	Si
XEN	Paravirtualización.	GNU/Linux, Unix-like	GPL versión 2	Versión 4.4 [81] de XEN [79]	Si
VirtualBox	Virtualización Completa	Windows, Linux, Mac OS X x86, Solaris, FreeBSD.	GPL versión 2	Versión 4.3.12 de VirtualBox	Si (Con licencia comercial)

2.5.1. Descripción de los hipervisores

A continuación se realiza una breve descripción de las plataformas de virtualización que serán posteriormente evaluadas, dejando plasmado algunas de las características que tienen en común, rasgos que las diferencia una de las otras, así como principales funcionalidades. Algunos de estos hipervisores poseen la característica en común de poder ser administrados por *libvirt* [82], un conjunto de herramientas para la administración de máquinas virtuales de las versiones recientes de hipervisores de Linux. Y otros casos por la plataforma de gestión *Proxmox* [83], tanto para sus recientes versiones como para las más antiguas.

LXC: *LXC* es una tecnología de virtualización ligera, que entra entre los hipervisores de tipo 2 donde su método de operación es la virtualización a nivel de SO o basada en contenedores como su nombre lo indica [84]. *LXC* permite ejecutar en un único anfitrión múltiples entornos virtuales denominados contenedores o máquinas virtuales ligeras, las cuales se encuentran aisladas unas de otras. El aislamiento es llevado a cabo por el *cgroups* [14] *y namespaces* [15] que son completamente nativos del kernel Linux, evidenciado a partir de la gestión y la independencia de los recursos [78] [85].

[14] Grupos de Control de Procesos.
[15] Espacios de nombre.

Al ejecutar *LXC*, se pueden crear varios contenedores, dentro de los cuales residen otras versiones de GNU/Linux, con sus propias interfaces de red, cuotas de disco duro, CPU, RAM[16], *namespaces* de procesos y de usuarios [86].

LXC cómo se mencionaba con anterioridad no emula todo un SO, más bien solo las dependencias necesarias dígase sistemas de archivos y librerías, para poder ejecutar las aplicaciones que se tengan instaladas en cada contenedor. A partir de la versión del kernel de Linux 2.6.29, *LXC* se encuentra oficialmente integrado [84]. Los contenedores de Linux también pudieran verse como una técnica de *chroot* mejorada. La diferencia es que un entorno *chroot* separa sólo el sistema de archivos, mientras que *LXC* va más allá y ofrece la gestión y control de recursos a través de *cgroups*.

Chroot no es más que una sección del sistema de archivos que se aísla del resto del sistema de archivos. El aislamiento está dado por la función del comando *chroot*, al realizar un cambio de raíz de la sección. El cambio de raíz, tiene el objetivo de que cuando un programa se ejecuta en una "jaula *chroot*" no pueda acceder a los archivos fuera del árbol de directorio designado [85].

Dentro de las principales ventajas que ofrece *LXC* se encuentra, el aislamiento a través de los contenedores mediante de los cgroups como se mencionaba. Alcanza un rendimiento casi nativo producto de la asignación de recurso en tiempo real. Por otra parte, también mediante los *cgroups* proporciona control de las interfaces de red y la aplicación de los recursos dentro de los contenedores.

Al mismo tiempo que ofrece numerosas ventajas, presenta limitaciones, como el uso del mismo kernel por parte de todos los entornos virtualizados. Solamente permite como sistemas invitados a distribuciones Linux. Y total dependencia del sistema anfitrión, en cuanto a seguridad se trata, puesto que los invitados son instancias del anfitrión, un fallo en el mismo traería consecuencias graves [85]. *LXC* se puede ver también como una derivación de *OpenVZ* con nuevas ventajas incluidas como soporte de la librería *LibLXC*[17], una

[16] Memoria de Acceso Aleatorio.
[17] Conjunto de implementaciones de funcionalidad que permiten gestionar los contenedores de Linux.

API[18] para *Ruby*[19], *Python*[20] v2 y v3, *Haskell*[21] y *GO*[22]. No obstante, aún no soporta funcionalidades como garantizar *QoS* en el almacenamiento, la migración en caliente, entre otras [87].

Docker: Al igual que *LXC*, *Docker* hace uso de la virtualización basada en contenedores, siendo un hipervisor de tipo 2, por lo que para su ejecución es necesario un SO anfitrión. Los contenedores son ligeros por no cargar un SO completo, esto se debe a que implementa un alto nivel de API. *Docker* es una plataforma de código abierto con el que fácilmente se puede crear contenedores, siendo de gran ayuda para administradores y desarrolladores, al brindar un entorno seguro para probar, ejecutar aplicaciones y sistemas [88]. Los contenedores son ligeros, ya que el espacio que ocupa con todas las dependencias y código de la aplicación, no es grande comparado con la ejecución de una MV. Otro de los factores que hace único a los contenedores es la portabilidad, porque pueden desplegarse en otro sistema que soporte esta tecnología, logrando reducir los tiempos dedicados a instalación y configuración [89].

Docker inicialmente estuvo basado en *LXC*, actualmente ha desarrollado su propia implementación, *libcontainer*, la que emplea directamente los *namespaces* y el *cgroup* del kernel. *Libcontainer* le permite a *Docker* crear múltiples entornos totalmente aislados entre sí, sobre un mismo sistema. Cada entorno puede tener un nivel de aislamiento de recursos (CPU, red, memoria, E/S[23], etc.) igual que las MV, pero a un nivel más eficiente y portable.

Por el hecho de que un contenedor puede ejecutarse más rápido que una MV, y al usar mucho menos espacio y menos RAM; los administradores de sistemas pueden hacer un mejor uso de los recursos de cómputo [90]. Para completar al máximo dicho aislamiento, como se mencionaba *Docker* hace uso de los *namespaces* para cada aspecto del contenedor, dígase gestión de red, de procesos, entre otros. Esto

[18] Application Programming Interface (Interfaz de Programación de Aplicaciones).
[19] Lenguaje de programación orientada a objeto.
[20] Lenguaje de programación interpretado, multiparadigma ya que soporta orientación a objeto, programación imperativa y programación funcional en menor medida.
[21] Lenguaje de programación funcional.
[22] Go es un lenguaje de programación compilado, concurrente, imperativo, estructurado y orientado a objetos soportado por diferentes tipos de sistemas UNIX (Linux, MAC OS X, FreeBSD).
[23] Entrada/Salida

asegura que cada aspecto se ejecute en su propio *namespaces*, sin tener acceso a ningún otro, solo a su contexto [90].

Para el uso de *Docker* es necesario que el procesador soporte la virtualización por hardware, es decir, que la CPU soporte tecnologías de virtualización AMD-V o Intel-VT. Además de funcionar solamente sobre distribuciones con versiones del kernel iguales o superiores a la 3.8. Solo funciona sobre distribuciones Linux, aunque para otros sistemas como *Windows*, se cuenta con un archivo denominado *boot2docker*. Este archivo se puede ejecutar con una plataforma de virtualización como VirtualBox, en *Windows* u otro sistema, y así poder ejecutar el hipervisor *Docker* [89]. Algunas insuficiencias de *Docker* parten de que no fue diseñado para el soporte de múltiples aplicaciones, procesos y servicios como *init* [91], *cron* [92], *syslog* [93] y *ssh* [94]. Lo que introduce una alta complejidad ya que las arquitecturas, aplicaciones y servicios son diseñados para SO multiprocesos [95] [96].

KVM: Es una solución de virtualización completa que permite ejecutar sistemas operativos basados en arquitecturas por 32 bits y 32-64 bits sin tener que haberlo modificado con anterioridad [36]. Para ello se basa en las nuevas instrucciones Intel-VT y AMD-V, por lo que será necesario disponer de un equipo con un procesador que las soporte para poder utilizarlo [97] [98]. Forma parte del kernel de Linux desde 2007, como un módulo desde la versión 2.6.20, permitiendo que los programas en espacio de usuario utilicen las tecnologías anteriores [99]. Permite ejecutar máquinas virtuales utilizando imágenes de disco que contienen sistemas operativos sin modificar. Cada máquina virtual tiene su propio hardware virtualizado: una tarjeta de red, discos duros, tarjeta gráfica, etcétera. Las MV tendrán tres modos de ejecución (invitado, usuario y núcleo), siendo procesos normales del sistema, pero con la peculiaridad de que se le agrega el modo invitado, utilizado para ejecutar todo el código de los huéspedes que no sea de E/S [78].

KVM fue creado por compañía *Qumranet* y en el 2008 adquirido por *RedHat*, el cual ha puesto todo su empeño en definirlo como la mejor solución de virtualización en el mundo SLCA [97] [99]. Al integrase al kernel, le permite al SO convertirse en un hipervisor de tipo uno y ejecutar de varias MV aisladas. Utiliza *QEMU*[24] para realizar la virtualización, pero no como emulación, sino únicamente como herramienta para realizar la carga de las MV [99] [35].

[24] Emulador de arquitecturas, como x86, x86_64, ARM, SPARC, PowerPC y MIPS.

La principal ventaja de *KVM* con respecto a sus contrarios como XEN, es que viene incluido como módulo del kernel, por lo que no es necesario instalar ningún software específico. Si se dispone de una máquina con el procesador adecuado y se instala un Linux basado en este kernel, se dispondrá automáticamente de los servicios de virtualización. Es de esperar por tanto, que esta tecnología evolucione bastante rápido y que aparezcan herramientas que permitan sacarle mayor provecho [36].

XEN: Es una solución de código abierto que puede trabajar tanto para obtener Paravirtualización a nivel de SO como para brindar asistencia por hardware [98], lo que le permite trabajar con sistemas invitados modificados y sin modificar. Mediante la técnica de paravirtualización, XEN brinda comunicación entre el hipervisor y el SO invitado, con el objetivo de que exista una colaboración de los sistemas huésped, capaz de gestionar de forma simple el proceso de virtualización. Para esto se requieren cambios en el sistema, pero se obtiene un alto nivel de rendimiento [35] producto de la comunicación entre el hipervisor y los sistemas virtuales. El código de *XEN* es sencillo y ligero ya que sólo cuenta con el necesario para repartir los recursos entre varias máquinas virtuales. En el código no se alojan drivers, por lo que para trabajar es necesario un SO que proporcione los drivers y una forma de comunicarse con *XEN*, entendiéndose como la administración del sistema. La administración se basa en hacerse cargo de la planificación de las tareas y de la gestión de memoria, delegando la gestión E/S en una MV especial denominada dominio 0 (dom0). Para realizar toda la administración el hipervisor se ejecuta en el nivel más privilegiado [36] [100].

De esta manera es como trabaja *XEN*, dividiendo toda la administración en un dominio principal (dom0) y varios subdominios denominados, dominios U (domsU). Cada uno de estos dominios se puede interpretar como MV, en donde el dom0 es el único que tiene control sobre el hipervisor y los demás dominios. El kernel de este dominio principal se ejecuta en el mismo nivel del microprocesador, así como *XEN*, mientras que las otras MV se ejecutan en el nivel 1, dejando el nivel 3 para sus aplicaciones. Cuando una MV (domsU) solicita acceso a un dispositivo, pasa a través del dom0, asegurando el aislamiento.

VirtualBox: Es un hipervisor de tipo dos que funciona sobre múltiples SO dígase GNU/Linux, Mac SO X, Solaris/OpenSolaris y Windows, siendo un potente software para realizar pruebas, desarrollar y desplegar múltiples soluciones a través de múltiples plataformas en una misma máquina. Permite virtualizar los SO FreeBSD, GNU/Linux, OpenBSD, OS/2 Warp, *Windows*, Solaris, MS-DOS entre otros. Para la ejecución se requiere de un hardware con un procesador compatible con 32 bits de Intel o AMD, aunque no es necesario que soporte virtualización por hardware Intel-VT o AMD-V [101].

Permite crear varias MV de 32 y 64 bits sobre las cuales puede instalarse cualquiera de los SO mencionados [36], en donde cada MV puede arrancarse, suspenderse y pararse de modo independiente. También pueden moverse de un equipo a otro ya que el funcionamiento del software y los ficheros son los mismos en cada MV. Se cuenta con versiones de VirtualBox operando tanto bajo licencia privada como de código abierto, que está sujeta a la licencia GPL [102]. Ofrece funcionalidades como la ejecución de máquinas virtuales de forma remota, por medio del *Remote Desktop Protocol* (RDP, Protocolo de Escritorio Remoto), además de montar imágenes ISO como unidades virtuales de CD o DVD, o como un disco *floppy*.

En conclusión *VirtualBox* implementa un parche en el código del sistema anfitrión, con el objetivo de que el código que se ejecuta en el anillo 0, lo haga en el anillo 1, dejando el anillo 0 para que se ejecute el hipervisor y tenga el máximo control de acceso. En ocasiones la creación de parches trae consigo problemas y para resolverlos, además de no tener que volver a compilar, se hace uso de un recompilador dinámico basado en *QEMU* [78].

2.6. Conclusiones parciales

LXC, Docker, KVM, Xen y VirtualBox fueron establecidos como hipervisores candidatos para la selección, teniendo en cuenta la posición en el mercado del proveedor, el tipo de licencia, la compatibilidad con Nova Servidores 5.0 y el soporte ofrecido.

La selección del candidato más adecuado se corresponde con una evaluación a partir de métricas asociadas a las variables: escalabilidad, rendimiento, seguridad, aislamiento, tiempo de respuesta, compatibilidad, portabilidad, fiabilidad y la plataforma de gestión.

CAPÍTULO 3. EVALUACIÓN DE LAS VARIABLES A TENER PRESENTE EN UNA PLATAFORMA DE VIRTUALIZACIÓN

En el presente capítulo se describen las capacidades del hardware a utilizar en el proceso de pruebas y la cantidad de recursos a los que pueden acceder los sistemas virtuales. Con la culminación de las pruebas se expondrán los resultados obtenidos, en donde se resaltan resultados cualitativos, teniendo en cuenta la descripción de diversos autores referidos a las plataformas de virtualización y los tipos de virtualización que realizan. Además se exponen los cuantitativos asociados a las métricas de las variables, lo que permite la determinación del resultado final, siendo el hipervisor que aplica a Nova Servidores 5.0.

3.1. Descripción del hardware y software empleado

El entorno de prueba parte de la base de un hardware con las características planteadas en la tabla 10, un sistema anfitrión (Nova Servidores 5.0) y la ejecución de MV con servicios telemáticos para simular lo más real posible el escenario de trabajo.

Tabla 10 Características del hardware físico.

Recurso de Hardware	Principales Características
Placa Madre	H8IM
Procesador	Intel(R) Core(TM) i3-4160 3.60 GHz
Memoria del Sistema	4 GB DDR3
Interfaz de Red	RTL8111/8168/8411 PCI Express Gigabit Ethernet Controller-Vendor(Realtek Semiconductor Co.Ltd) - 1 Gbits/s-100Mbits/s
Almacenamiento	ST1000DM003-IER162 Disk Device 1000.2 GB

3.1.1. Capacidades de las Máquinas Virtuales

Para cada una de las MV desplegadas por hipervisor se limitaron los recursos a consumir de la siguiente forma (tabla 11):

Tabla 11 Características de los recursos por MV

Recurso de hardware	Límite de acceso
Procesador	1 procesador
Disco Duro	50 GB
Memoria	1024 MB
Red	100 Mbits/s

3.1.2. Propiedades de la Máquina Cliente

En las pruebas de red se utiliza un ordenador externo que realiza solicitudes a las MV y viceversa para evaluar el desempeño del hipervisor en cuanto al manejo de transferencia y solicitudes. A continuación se establecen las propiedades del cómputo (tabla 12).

Tabla 12 Propiedades de la máquina cliente

Tipo	Propiedad
Placa Madre	Toshiba Satellite A665
Procesador	Intel Core i3-350M, 2266 MHz (17 x 133)
Memoria del sistema	3890 MB (DDR3-1066 DDR3 SDRAM)
Interfaz de Red	Intel(R) Centrino(R) Advanced-N 6200 AGN
Almacenamiento	ST950042 0AS SCSI Disk Device (500 GB, 7200 RPM, SATA-II)

3.2. Resultados obtenidos de las pruebas

En el actual epígrafe se plantean los parámetros a tener presente para el empleo de las herramientas. Los valores obtenidos pueden variar según las cargas de trabajo, manteniéndose las capacidades de los hipervisores.

3.2.1. Evaluación del subsistema red con las herramientas Netperf e Iperf

Las siguientes pruebas se basan en la evaluación de parámetros significativos en las conexiones de red como el desempeño, demora en la propagación y transmisión de paquetes, haciendo uso de la herramienta *Netperf*. Las pruebas de red permiten determinar cómo el hipervisor maneja las entradas y salidas de los datos, que interactúan con las MV. La herramienta *Netperf* para el cálculo de las variables antes mencionadas no tiene en cuenta el tiempo de conexión, solo el tiempo de transferencia.

La primera prueba de este grupo evalúa el desempeño, visto como la velocidad o ancho de banda de transferencia. Para ello se realiza la simulación de una transferencia de paquetes con tamaños de 1 y 17 Kbyte respectivamente, en donde la salida mostrada se encuentra dada en Mbps. Un dato significativo en algunos de los hipervisores es la capacidad de cambiar el modo de conexión de NAT[25] a reenvió de puerto. Principalmente *Docker* realiza este proceso ya que su funcionamiento mediante NAT genera sobrecarga cuando el flujo de carga incluye grandes cantidades de paquetes, según un informe de investigación de la empresa IBM [103]. En la figura 6 del anexo D se encuentran los resultados obtenidos para cada hipervisor.

Otro de los parámetros evaluados como se mencionaba es la latencia, para esto se utilizaron las pruebas TCP-UDP solicitud y respuesta, denominada TCP_RR y UDP_RR respectivamente de *Netperf*, las cuales simulan el envío de paquetes con tamaños de 1 y 17 Kbyte respectivamente, lo que determina la demora en microsegundos de la transferencia. La aplicación de la prueba arrojó resultados que sitúan a los hipervisores basados en la técnica de contenedores con los mejores valores de desempeños, los cuales se muestran en la figura 7 y 8 del anexo D.

Uno de los requisitos a tener en cuenta en las pruebas de redes, es el tráfico de conexiones, así como las propiedades de red ejemplo capacidad máxima de velocidad de transferencia que admite los recursos de red. Un entorno saturado de conexiones puede influir en los resultados obtenidos por las pruebas de forma

[25] Network Address Translation (Traducción de direcciones de red).

diferente, restando valor o adicionándolo según el estado inicial. Por lo que el autor de la tesis considera que en vista de obtener resultados equitativos, se realicen las pruebas relacionadas con el subsistema de red, bajo las mismas condiciones en que se desarrolle un 1er ciclo para un 1er hipervisor. En otro caso es recomendable crear un entorno controlado, definido como la conexión de un cliente a un servidor a través de un *switch*.

Centrándose en la herramienta *Iperf*, la misma permite medir el desempeño de un enlace local o hacia Internet, mediante la simulación de transferencia de datos, en donde el hipervisor controla el acceso a los recursos de red. Se determina igual la variación de la latencia (*jitter*), es decir: mide las variaciones del tiempo de demora por envío de datos. El *jitter* no depende de la latencia ya que puede tener tiempos de respuesta altos y una muy baja fluctuación. Esta variable es particularmente importante en los enlaces de la red de soporte de voz sobre IP (*VoIP*), ya que una alta fluctuación puede romper una conexión de llamada. Otras de las métricas a medir es la pérdida de paquetes, teniendo como parámetros: anchos de banda de 95 Mbps y 100 Mbps y un tamaño de paquete de 500 MB.

El manejo de las conexiones por parte de los hipervisores es fundamental y uno de los puntos significativos dentro de las conexiones es la variación del tiempo de espera ya sea de solicitud o respuesta. Puesto que variaciones altas traen consigo pérdidas de paquetes, resultando ineficiente en las conexiones de red. Por lo que se evalúa este parámetro con el objetivo de ver el porciento de variación y así mismo de pérdida de paquetes, cuando una plataforma controla y maneja el acceso a los recursos físicos de hardware. Los resultados se muestran en milisegundos (ms) en la figura 9 del anexo D.

Un caso notable para *jitter* es cuando el valor se encuentra sobre los 200ms [104], lo que no se pone de evidencia en ninguna de las plataformas evaluadas. Los hipervisores con mejores resultados obtenidos son *LXC* y *Docker* tanto para los 95Mbps como para los 100Mbps, dejando en última posición y con los resultados más bajos según el criterio de aceptación a *XEN*. En la figura 10 del anexo D se muestran los resultados adquiridos en la prueba de pérdida de paquetes, no siendo críticos para ninguna de las plataformas.

3.2.2. Prueba 1: Subsistema CPU

Las siguientes pruebas además de evaluar el poder de la CPU al realizar un sin números de operaciones, da a conocer como maneja el hipervisor el acceso a los recursos por parte de las MV hospedadas. En particular se evalúan métricas como la velocidad, demora y los MFLOPS que son vigentes en la ejecución de operaciones relacionadas con la CPU.

- **GraphicsMagick**

La primera prueba denominada *GraphicsMagick* tiene el objetivo de realizar la transformación de una imagen. Para eso emplea *OpenMP*, lo que permite aumentar la velocidad de las operaciones relacionadas con la CPU. Así mismo el procesamiento de una imagen esta compuestos por bucles que iteran a través de atributos de una imagen como pixeles, filas o áreas [105]. Por lo que el resultado de la prueba viene dado en iteraciones, es decir cuántas iteraciones bajo unidad de tiempo (minutos) se emplean para la transformación de una imagen. La prueba fue ejecutada con algunos de los parámetros por defectos establecidos en PTS, siendo el caso de una duración de 60 segundos (**-duration 60**), *Color Space RGB* [26] (**-colorspace RGB**) para la reducción de color de una imagen en el formato RGB. *Resize* (-**resize 50%**) para la reducción de imagen en este caso al 50% del estado normal y por último *Sharpen* (**-sharpen 0x1.0**) que aumenta el enfoque de una imagen haciendo uso de un operador *Gaussiano* de radio 0 y desviación estándar 1, en este caso [106]. La figura 11 del anexo D muestra los resultados obtenidos por parámetros.

- **C-ray**

La prueba *C-ray* genera una imagen de entrada con una resolución de 1600 x 1200 píxeles, mediante un trazado de 8 líneas por píxeles y un uso de 16 hilos (*threads*) por núcleo. Para esto se emplea el comando por defecto de pts, siendo: **c-ray-mt -t 64 -s 1600x1200 -r 8 -i sphfract -o output.ppm**. La opción ***–t*** es el número de hilos, ***-s*** la resolución, ***-r*** cantidad de líneas y ***–i*** e ***–o*** imagen de entrada y salida respectivamente. *C-ray* permite medir la capacidad de la CPU en FLOPS mediante el trazado de líneas por píxel y muestra como resultado la demora en segundos que fueron necesarios para el completamiento de la prueba. La figura 12 del anexo D muestra los resultados obtenidos.

[26] Red Green Blue (Rojo Verde Azul)

- **Himeno Benchmark**

Himeno es una prueba para medir la capacidad del cómputo, al medir las operaciones con puntos flotantes, mostrando así mismo el valor en MFLOPS para resolver los sistemas de ecuaciones de presión de *Poisson* haciendo uso del método *Point-Jacob*. La prueba está compuesta por un archivo en lenguaje de programación C, el cual incluye los métodos necesarios para solucionar las ecuaciones, además de la salida de puntos flotantes de las operaciones. Los resultados arrojados se evidencian en la figura 13 del anexo D.

- **FLAC Audio Encoding**

FLAC Audio Encoding es una prueba que determina la demora en segundos para convertir un archivo de audio de formato WAV en FLAC en las CPU. FLAC es un códec de audio que permite comprimir un archivo de audio sin que se pierda información. Para esta prueba se emplea un archivo de audio en formato WAV con un tamaño de 51 MB. Los resultados de la actual prueba son reflejados en la figura 14 del anexo D.

- **Tachyon**

Tachyon es una prueba de trazado de líneas paralelas que determina el tiempo de demora en la generación de imágenes aleatorias, para una escena determinada. La escena seleccionada cuenta con una cantidad de objetos igual a 2330. En la figura 15 del Anexo D se muestran los resultados obtenidos.

3.2.3. Prueba 2: Subsistema de disco duro

Las pruebas referentes al subsistema disco duro permiten determinar la velocidad de las operaciones de entradas y salidas así como el rendimiento de estas operaciones, lo que da a conocer como los hipervisores manejan las operaciones de acceso a disco, escritura, lectura y sistema de ficheros de las MV.

- **Gzip Compression**

Esta primera prueba tiene el objetivo de comprimir un archivo de 2 GB en el formato Gzip, lo que da como resultado el tiempo de duración en segundos para la realización del proceso. Ver figura 16 del anexo D para detalles de los resultados obtenidos.

- **Compile Bench**

Esta prueba simula el envejecimiento del sistema de archivo, es decir la ocupación total de espacio del disco duro y mide el rendimiento al analizar como el sistema mantiene la localización de los directorios a medida que el disco se llena. Para esto hace uso de procesos de lectura y escritura como: Creación inicial (***INITIAL_CREATE***), Compilación (***COMPILE***) y Lectura de árbol compilado (***READ_COMPILED_TREE***).

La prueba mide la velocidad en MB/s de las diferentes instrucciones e inicia con 10 directorios. Los resultados obtenidos se reflejan en la figura 17 del anexo D.

- **IOzone**

IOzone permite comprobar el rendimiento del disco mediante operaciones de L/E de un archivo de 1GB, para esto se utilizaron los parámetros **"-r 4k -s 1024M -i0"** y **"-r 4k -s 1024M -i0 -i1"** en donde -r es el tamaño de registros, ***-s*** tamaño del archivo y ***–i1*** e ***–i0***, hacen referencia a los procesos de escritura y lectura respectivamente. La salida de la prueba es en Kb/s determinando así el rendimiento para L/E. Los resultados obtenidos por operación se muestran en la figura 18 del anexo D.

- **fs-mark**

La actual prueba está diseñada para determinar el rendimiento del sistema de archivo al ver la cantidad de ficheros por segundos a los que se pueden acceder. Para esto fueron utilizados los parámetros **"-s 1048576 –n 1000 –t 4"** en donde –s es la cantidad de ficheros total a acceder y –n es el tamaño de los ficheros. Viéndose: cuantos ficheros en un rango de [0-1000] de 1 MB se puede acceder por segundos. Lo que trae como resultado lo reflejado en la figura 19 del anexo D.

- **PostMark**

PostMark es una prueba que simula el acceso a servidores web y de correo, lo que permite medir el rendimiento en transacciones de disco de las plataformas, al ver cuántas transacciones por segundos de 250 000 con 500 archivos simultáneos de tamaños variables entre 5 y 512 KB se pueden realizar. En el proceso de desarrollo de la prueba se modifican los valores por defecto de la herramienta con los comandos, ***set transactions*** **250000**, ***set size*** **5120 524288** y por último ***set number*** **500**. Los resultados obtenidos se muestran en la figura 20 del anexo D.

3.2.4. Prueba 3: Subsistema de memoria RAM

El siguiente grupo de pruebas están destinadas para medir el subsistema de memoria RAM y determinar cómo los hipervisores llevan el control de acceso al recurso de memoria.

- **RAMspeed SMP**

Esta primera prueba evalúa el rendimiento de la RAM mediante operaciones de suma, copia y multiplicación por un escalar de números enteros y suma en operaciones con punto flotante. Se hizo uso de los parámetros ***"COPY SCALE ADD TRIAD -b6"***, que no son más que operaciones básicas que se realizan sobre la

memoria RAM. En el caso del atributo ***–b6*** hacen referencia a las operaciones de suma, copia, multiplicación y unión de punto flotante. La salida de la prueba es en MB/s lo que refleja el rendimiento por operación. En la figura 21 del anexo D se muestran los resultados obtenidos por hipervisor.

- **STREAM**

STREAM es una prueba semejante a RAMspeed que mide el rendimiento de la RAM mediante operaciones y argumentos ***"Add***, ***Copy***, ***Scale*** y ***Triad"*** [107] y la ocupación de parte de la memoria con archivos de 2KB. Se emplea para el completamiento de la misma un tiempo de 10 segundos, en donde los resultados son expuestos en la figura 22 del anexo D.

- **CacheBench**

Esta prueba permite medir el rendimiento y ancho de banda de la memoria para operaciones de L/E. En el funcionamiento de la misma se utilizaron los argumentos –r y –w para lectura y escritura respectivamente. Su funcionamiento se basa en realizar operaciones de escritura y lectura sobre espacio de memoria de 256 a 65536 Kbyte, reflejando la capacidad de operación en MB/s por tamaño de archivo. Cuando la prueba concluye se evidencia el rendimiento de memoria promedio para realizar las operaciones de L/E sobre la memoria. Los resultados de esta prueba se evidencian en la figura 23 del anexo D.

3.2.5. Prueba 4: Subsistema de red

La prueba denominada ***Network Loopback*** mide el rendimiento de la red, simulando la transferencia de un paquete de datos de 10GB, dando como resultado el tiempo en segundo que demora la transferencia, reflejado en la figura 24 del anexo D.

3.2.6. Análisis de la variable Escalabilidad

La escalabilidad es una variable que permite determinar las limitaciones del hipervisor mediante el despliegue de sistemas huésped. En este epígrafe se analiza la escalabilidad de manera general entre *LXC*, *Docker* y las máquinas virtuales, refiriéndose a la mayor cantidad de despliegues que se pueden realizar por hipervisor. En el caso de los hipervisores que crean máquinas virtuales (*XEN*, *KVM*, *VirtualBox*), un factor que lo limita es la capacidad del disco duro, en donde el resultado máximo es la división del tamaño del disco duro entre la capacidad necesaria para hospedar un SO. Por parte de LXC el límite está dado solamente por el peso de la imagen base y las dependencia de la aplicación que ejecute el contenedor. Como último destacar a *Docker* ya que se pueden desplegar gran cantidad de instancias, puesto que los contenedores pueden compartir el mismo espacio de disco e imagen, aislando las operaciones de lectura y

escritura de cada contenedor. En general se puede decir que por parte de una creación de 10 a 100 MV se pueden crear de 100 a 1000 contenedores [108].

3.2.7. Análisis de la variable Aislamiento

El aislamiento es un factor importante que explica por qué la disponibilidad y la protección de las aplicaciones que se ejecutan en un entorno virtual, es superior, al de las aplicaciones que se ejecutan en un sistema tradicional no virtualizado. El aislamiento es una variable que radica en todos los hipervisores, siendo más eficiente en unos que en otros. En este epígrafe se realiza un análisis del aislamiento entre los procesos y la capacidad de abstracción del hipervisor al separar las MV unas de otras. En relación a los contenedores para lograr un aislamiento entre los procesos hacen uso de los *namespaces* y *cgroups*. Los *namespaces* aislando los recursos del sistema al nivel del proceso, lo que asegura que el contenedor solo vea su propio entorno; mientras que los *cgroups* ayudan a controlar, justificar y aislar los recursos que una colección de procesos puede utilizar [109]. En los contenedores en general el aislamiento es pobre, debido a que comparten el mismo kernel del *host* y un fallo se reflejaría en los contenedores.

Las MV proveen un alto nivel de aislamiento, un error en un proceso o un fallo de funcionamiento en una MV no incurre en el funcionamiento de otras bajo el mismo hipervisor. Esto es posible por la encapsulación ofrecida por el hipervisor en donde un conjunto de recursos del hardware virtual, así como un SO y todas sus aplicaciones, se encuentran dentro de un paquete de software. Al estar empaquetado elimina la posibilidad de acceder al sistema *host*, debido a que el sistema huésped y sus aplicaciones no están conscientes que están siendo virtualizados. Por lo que, a vista del usuario o atacante remoto, el sistema se ejecuta de forma habitual, como si se ejecutara sobre un hardware real.

3.2.8. Análisis de la variable Compatibilidad

La evaluación de la compatibilidad en los hipervisores, es la capacidad de ser tanto multiplataforma como que permita desplegar sistemas virtuales de múltiples fabricantes. Como hipervisores multiplataforma se encuentra *VirtualBox*, que permite desplegar MV de varios fabricantes. En cambio *XEN* y *KVM* solo se ejecutan sobre sistemas Linux y permiten desplegar MV de varios fabricantes. En el caso de *XEN* para desplegar las MV que no son de Linux, el proceso de virtualización necesita apoyo del hardware. Con un nivel de compatibilidad mínimo se encuentra *LXC*, solamente puede ejecutar sobre sistemas Linux y desplegar contenedores de Linux, producto de compartir el mismo kernel que el *host*. En el caso de *Docker* solo funciona de forma nativa en Linux, pero dispone de una herramienta *bott2docker* [110] que permite su

ejecución de forma sencilla tanto en *Windows* como en MAC OS X. *Boot2docker* instala *Docker* en una máquina virtual en memoria usando *VirtualBox*, permitiendo obtener información de la máquina virtual a través de diversos comandos.

3.2.9. Análisis de la variable Plataforma de Gestión

Las plataformas de gestión permiten administrar, gestionar y monitorear el funcionamiento de las máquinas virtuales, aportando ventajas significativas para los administradores. Existen varias de estas herramientas que permiten gestionar las máquinas virtuales a comandos, mediante interfaces gráficas y web. *VirtualBox* puede ser manejado por la interfaz común que trae por defecto, mediante comando por consola [111] [112] y web mediante la interfaz de administración para un uso en intranet *PhpvirtualBox* [113] [114]. Para la gestión y administración de la virtualización de plataformas como *KVM*, *XEN* y *LXC* se puede hacer uso de la API *libvirt-bin*, el administrador *virt-manager* y el paquete *virtinst* [115]. Entre las ventajas más destacadas por *libvirt* se encuentran además de la gestión de máquinas virtuales, la gestión de redes virtuales y de almacenamiento. Gestión remota usando cifrado TLS y control de acceso local usando *PolicyKit* [116]. Una de las herramientas más usadas por administradores para la gestión es *Proxmox*, brindando soporte para *KVM* y LXC. *Proxmox* trae consigo numerosas ventajas como el soporte para IPv6, políticas de *backup* y la herramienta clúster de alta disponibilidad entre otras [117] [118].

En cuanto a la gestión de *Docker* se realiza a través de comando por consola y en otros casos mediante una tecnología denominada Kurbenetes lanzada por la empresa de *Google*. Es una herramienta de código abierto destinada para la gestión de máquinas virtuales en centros de datos para la nube. Entre las características que ofrece se puede mencionar el desarrollo y gestión de entornos de trabajo en producción, como DNS, balance de carga, escalado, chequeo de estado a nivel de aplicación, entre otras [119].

3.2.10. Análisis de la variable Seguridad

Actualmente la alta seguridad es otra de las principales tendencias que buscan concretar y perfeccionar los administradores por el aumento de ataques informáticos por año [120]. Por lo que su evaluación determina las capacidades del hipervisor al implementar características de seguridad.

- *LXC-Docker*

Los contenedores, son capaces de brindar soporte de políticas de seguridad para el control de acceso y capacidades de procesos, gestionado por los módulos de seguridad *SELinux* y *AppArmor* [121] [122].
En iniciales versiones de *Docker* hasta la 1.3.1 se recurre a HTTP[27] cuando HTTPS [28] falla, permitiendo realizar ataques con el fin de obtener datos de autenticación e imagen [123] de los contenedores. Otra de las vulnerabilidades se encuentra en la opción **–v** insertada cuando se crea un contenedor, que permite compartir espacios de disco entre los contenedores y el *host*, por lo que se puede obtener acceso al *host* mediante un ataque al contenedor.

Si se accede a la API del contenedor, se tendrá acceso de superusuario en el *host*, permitiendo realizar cambios, crear cuentas de usuarios, cierre de la máquina, correr procesos maliciosos e insertar códigos especiales en el kernel para ocultarlos. En versiones a partir de la 1.6.1, *Docker* implementa mecanismos que erradican las vulnerabilidades presente en versiones anteriores. Por ejemplo antes de exponer la API de *Docker*, fijarlos con certificados TLS[29] [124]. Como práctica de seguridad, *Docker* contrata personal capacitado en términos de seguridad, los que exponen mecanismos a tener en cuenta para mitigar las vulnerabilidades existentes [125].

Con respecto a LXC en su versión 1.1.2 y anteriores presentan vulnerabilidades en *lxclock.c*, que le permiten a los usuarios locales crear archivos arbitrarios a través de un ataque *symlink*[30] en la dirección /run/lock/lxc/*. Los archivos arbitrarios eliminan o modifican un conjunto de ficheros, como las dependencias de un servicio y el mal manejo de estos archivos implicaría una caída completa del servicio [126]. Con una alta complejidad se encuentra la vulnerabilidad de *lxc-start* en versiones anteriores a la 1.1.14, que les permite a los administradores locales del contenedor escapar del confinamiento *AppArmor* a través de un ataque de enlace simbólico. Lo que da lugar a que no tengan restricciones de acceso y puedan atentar contra la seguridad del contenedor y el sistema *host* [127].

[27] Hypertext Transfer Protocol (Protocolo de transferencia de hipertexto).
[28] HTTP+SSL (Seguridad de la capa de transporte).
[29] Transport Layer Security (Seguridad en la Capa de Transporte).
[30] Método para hacer referencia a otros archivos y carpetas en Linux (Acceso directo).

Dos de los principales factores que degrada la seguridad en los contenedores está dado por los límites que impone el hipervisor en cuanto al acceso de objetos lógicos de una MV, como los archivos, direcciones de memoria virtual, números de puertos e identificadores de procesos. La técnica de contenedores sacrifica la seguridad mediante la ubicación de la lista de proceso en el *host* y no encapsulada en el contenedor, ganando en rendimiento. El segundo factor es la seguridad en cuanto al kernel, ya que los contenedores comparten el mismo que el sistema *host*, por lo que la ocurrencia de un fallo también se reflejaría en los contenedores.

- *XEN*

Todos los hipervisores presentan vulnerabilidades que a medida que el tiempo pasa sus proveedores han ido mitigando. *XEN* es uno de los casos, principalmente para las versiones entre la 3.2.x y la 4.6.x las cuales integran vulnerabilidades debido a un tratamiento inadecuado de errores. La función de intercambio de memoria (*memory.c*) en XEN, en ocasiones no entrega correctamente las páginas a un dominio, permitiéndole a los administradores de SO invitados provocar una Denegación de Servicio (DoS) en el sistema anfitrión. Para versiones entre la 4.1.x y 4.6.x cuando el sistema huésped está configurado para usar la técnica de paravirtualización que se ejecute como un proceso en el espacio del usuario en el dominio *toolstack* [128] puede producir una pérdida del espacio de direcciones virtuales. Esto provoca que los archivos que contienen el kernel y el disco RAM inicial sean irrecuperables y que los atacantes puedan provocar una denegación de consumo de memoria y de disco. Actualmente *XEN* implementa actualizaciones para sus vulnerabilidades y recomienda consultar los avisos de seguridad. [129] [130] [131].

- *KVM*

El subsistema *KVM* presenta una vulnerabilidad con gravedad media en el kernel de Linux hasta la versión 4.2.6, permitiendo a usuarios del SO invitado causar una DoS en el *host* del SO, desencadenando excepciones #AC (comprobación de alineación), relacionadas con *svm.c* y *vmx.c.* Otro de los riesgos que presenta el uso de *KVM*, es mediante la vulnerabilidad descubierta por *Jasson Geffner* denominada VENOM[31], que permite acceder al entorno de ejecución del hipervisor desde una MV, lo que significa acceder por completo al entorno de ejecución del resto de MV que tiene el *host* en ejecución. Esta

[31] Virtualized Environment Neglected Operations Manipulation (Manipulación de Operaciones Descuidadas en un Entorno Virtualizado).

vulnerabilidad se encuentra alojada de igual forma para aquellas MV que estén ejecutándose sobre sistemas *QEMU* y *XEN*. Por lo que se recomienda consultar sobre parches y actualizaciones que erradiquen este tipo de vulnerabilidades [132]. Para muchas de las vulnerabilidades por parte de *KVM* existen actualizaciones que las eliminan, por lo que es significativo en el uso de *KVM*, consultar las descripciones de las actualizaciones.

- *VirtualBox*

En el caso de *VirtualBox* al igual que todo los hipervisores presentan vulnerabilidades, unas no tan graves, mientras que otras con un alto nivel capaz de comprometer la integridad de los sistemas huésped. *Oracle,* compañía fabricante y proveedora de *VirtualBox*, implementa constantemente parches y actualizaciones erradicando la vulnerabilidades presentes. Además de la integridad, las vulnerabilidades presentes les permiten a usuarios locales y atacantes remotos afectar la disponibilidad y confidencialidad de los sistemas [133].

3.2.11. Análisis de la variable Portabilidad

La portabilidad refleja la capacidad de poder mover de un sistema a otro las MV de los hipervisores, ahorrando el tener que crear nuevamente la MV con todas las aplicaciones en un nuevo sistema. En el caso de los contenedores, *LXC* de forma nativa, no favorece la portabilidad puesto que no implementa ningún mecanismo que lo facilite. Por otra parte *Docker* brinda la característica de guardar los cambios realizados sobre un contenedor como una imagen, permitiendo desplegarlo en cualquier otro sistema que soporte esta tecnología. Las MV por otra parte dependen de la plataforma de gestión que las cree y las gestione. En el caso de *VirtualBox* implementa mecanismos que permiten exportar las MV al estándar abierto OVF[32], pudiendo consumirlo en otros sistemas.

3.3. Resultado final

En el actual epígrafe dado que las unidades de medidas de las salidas de las pruebas no son homogéneas se decide llevar todos los resultados a una misma escala mediante la normalización de los resultados obtenidos en las pruebas efectuadas sobre los subsistemas del hardware (Anexo D) y de las variables que fueron evaluadas de forma cualitativa. Con los valores de puntuación de 1 a 5 siendo el (5) el valor para el

[32] Open Virtualization Format (Formato Abierto de Virtualización).

hipervisor con mejor resultado en una determinada prueba y el (1) para el de más bajo resultado. Esto permite que se realice una comparación equitativa de las variables y que se defina la plataforma con mejores resultados. El proceso se ejerce de la siguiente forma:

1. Se normalizan los resultados obtenidos por pruebas, que a su vez atribuyen a una o varias métricas.
2. En caso de que la prueba posea más de una opción, se promedian los valores normalizados por opción y se vuelve a normalizar.
   - Ejemplo:
     - R: Resultados.
     - VN: Valor normalizado.
     - P: Promedio de los valores normalizados.
     - PN: Promedio normalizado.

Tabla 13 Resultados obtenidos de la prueba GraphicsMagick.

	LXC		Docker		KVM		XEN		Virtual Box	
Opciones	R	VN	R	VN	R	VN	R	VN	R	VN
Colorspace	2876	4	2878	5	1186	1	1669	3	1479	2
Resize 50%	1731	5	739	3	609	1	775	4	724	2
Sharpen0x1.0	547	5	165	1	267	3	197	2	319	4
	P	4,66	**P**	3	**P**	1,6	**P**	3	**P**	2,6
	PN	5	**PN**	4	**PN**	2	**PN**	4	**PN**	3

3. El resultado normalizado atribuye a VEL_OPS_CPU, dado que esta métrica está asociada a la prueba *GraphicsMagick*.
4. Se vuelve a realizar el procedimiento por prueba y métrica asociada.
5. Se promedian todos los valores normalizados de las métricas que atribuyen a una determinada variable. Se normalizan estos datos resultantes, arrojando un nivel de capacidad del hipervisor por métrica.

A continuación en las tablas 14, 15, 16, 17, 18, 19, 20, 21 y 22 se dejan plasmados los resultados por variable del procedimiento anterior descrito, mostrando los valores normalizados por pruebas y métricas en el Anexo F.

Tabla 14 Valores de rendimiento normalizados por hipervisor

Hipervisor	Variable Rendimiento					Promedio	Valor Normalizado
	VEL_OPS_MEM	VEL_OPS_CPU	IOPS	THROUGHPUT	FLOPS		
LXC	5	5	4	5	4	4,6	5
DOCKER	5	4	5	4	5	4,6	5
KVM	2	2	2	2	1	1,8	2
XEN	3	4	3	3	2	3	4
VirtualBox	4	3	3	1	3	2,8	3

Tabla 15 Valores de fiabilidad normalizados por hipervisor

Hipervisor	Variable Fiabilidad	Promedio	Valor Normalizado
	PACKET_LOSS		
LXC	5	5	5
DOCKER	5	5	5
KVM	4	4	4
XEN	3	3	3
VirtualBox	2	2	2

Tabla 16 Valores de tiempo de respuesta normalizados por hipervisor

Hipervisor	Tiempo de Respuesta				Promedio	Valor Normalizado
	DEMORA_OPS_CPU	IOPS	JITTER	LATENCIA		
LXC	5	4	4	5	4,5	4

DOCKER	4	5	5	5	4,75	5
KVM	1	2	2	3	2	1
XEN	2	3	1	4	2,5	2
VirtualBox	3	3	3	2	2,75	3

Tabla 17 Valores de escalabilidad normalizados por hipervisor

Hipervisor	Variable Escalabilidad	Promedio	Valor Normalizado
	CUMPLIMIENTO_INDICADORES		
LXC	5	5	5
DOCKER	5	5	5
KVM	2	2	2
XEN	2	2	2
VirtualBox	2	2	2

Tabla 18 Valores de compatibilidad normalizados por hipervisor

Hipervisor	Variable Compatibilidad	Promedio	Valor Normalizado
	CUMPLIMIENTO_INDICADORES		
LXC	3	3	3
DOCKER	3	3	3
KVM	4	4	4
XEN	4	4	4
VirtualBox	5	5	5

Tabla 19 Valores de seguridad normalizados por hipervisor

Hipervisor	Variable Seguridad	Promedio	Valor Normalizado
	CUMPLIMIENTO_INDICADORES		
LXC	2	2	2
DOCKER	2	2	2
KVM	4	4	4
XEN	4	4	4
VirtualBox	4	4	4

Tabla 20 Valores de aislamiento normalizados por hipervisor

Hipervisor	Variable Aislamiento	Promedio	Valor Normalizado
	CUMPLIMIENTO_INDICADORES		
LXC	2	2	2
DOCKER	2	2	2
KVM	4	4	4
XEN	4	4	4
VirtualBox	4	4	4

Tabla 21 Valores de portabilidad normalizados por hipervisor

Hipervisor	Variable Portabilidad	Promedio	Valor Normalizado
	CUMPLIMIENTO_INDICADORES		
LXC	3	3	3
DOCKER	4	4	4

KVM	4	4	4
XEN	4	4	4
VirtualBox	4	4	4

Tabla 22 Valores de la variable plataformas de gestión normalizados por hipervisor

Hipervisor	Variable Plataformas de Gestión	Promedio	Valor Normalizado
	CUMPLIMIENTO_INDICADORES		
LXC	5	5	5
DOCKER	2	2	2
KVM	5	5	5
XEN	4	4	4
VirtualBox	3	3	3

La tabla 23 muestra al hipervisor más completo con el valor 5 teniendo en cuenta la evaluación de las variables definidas, dejando planteado por igual la posición de los hipervisores restantes según sus capacidades.

Tabla 23 Posición de los hipervisores según su capacidad total (Resultado Final)

Hipervisor	Promedio General	Valor General Normalizado
LXC	3,87	5
DOCKER	3,75	4
KVM	3,25	2
XEN	3,37	3
VirtualBox	3,12	1

Una vez planteado todos los resultados generales teniendo en cuenta cada una de las variables definidas en el estudio, se determina a *Linux Container* como la plataforma seleccionada para Nova Servidores, la cual posibilitará un aprovechamiento máximo de las capacidades del hardware.

3.4. Conclusiones parciales

Las técnicas de pruebas realizadas a los subsistemas RAM, CPU, disco duro y red, mediante las *Suite Phoronix Test*, las herramientas *Netperf* e *Iperf*, arrojaron como resultado que LXC es el hipervisor más completo para incorporar en Nova Servidores 5.0.

CONCLUSIONES

Una vez concluida la presente investigación se arriba a las siguientes conclusiones:

1. La incorporación de un hipervisor de tipo 2 en Nova Servidores, se adecua a la tendencia mundial en el uso de la virtualización e incide como factor significativo en el patrón de adopción de la mayoría de las empresas para alcanzar la computación en la nube.
2. LXC, Docker, KVM, Xen y VirtualBox fueron seleccionados como hipervisores candidatos, teniendo en cuenta la posición en el mercado del proveedor, el tipo de licencia, la compatibilidad con Nova Servidores 5.0 y el soporte ofrecido.
3. La selección del candidato más adecuado se corresponde con una evaluación a partir de métricas asociadas a la escalabilidad, el rendimiento, la seguridad, el aislamiento, el tiempo de respuesta, la compatibilidad, la portabilidad, la fiabilidad y la plataforma de gestión del hipervisor.
4. Las técnicas de pruebas realizadas a los subsistemas RAM, CPU, disco duro y red, mediante las suite *Phoronix Test*, las herramientas *Netperf* e *Iperf*, arrojaron como resultado que LXC es el hipervisor más completo para incorporar en Nova Servidores 5.0.

RECOMENDACIONES

- Para el despliegue de máquinas virtuales completas, que posibilite la instalación de software que no sea del tipo GNU/Linux se recomienda el uso del hipervisor KVM.
- Para actualizaciones posteriores del kernel de Nova Servidores tener en cuenta la actualización de LXC a LXD integrado ya en la versión 16.04 de Ubuntu. La cual ofrece una seguridad semejante a las máquinas virtuales, con la rapidez de despliegue de los contenedores.

REFERENCIAS BIBLIOGRÁFICAS

1. RUBIO SAPIÑA, A. Alta disponibilidad en servidores y optimización de recursos hardware a bajo coste. [En línea] 2012. [Consultado el: 9 de 2 de 2016.] Disponible en: http://riunet.upv.es/bitstream/handle/10251/30138/tesis_master.pdf?sequence=1.

2. CISCO SYSTEAMS, Inc. El ROI de la virtualización. [En línea] 2011. [Consultado el: 9 de 2 de 2016.] Disponible en: http://www.cisco.com/web/LA/soluciones/comercial/centro_recursos/pdf/El_ROI_de_la_virtualizacion.pdf.

3. VMWARE, Inc. VMware Case Study. [En línea] 2015. [Consultado el: 9 de 2 de 2016.] Disponible en: http://www.vmware.com/files/ve/pdf/customers/VMware-Inter-CaseStudy-15Q1-VE.pdf?src=WWW_customers_VMware-Inter-CaseStudy-15Q1-VE.pdf.

4. SYMANTEC APPLIED RESEARCH. Encuesta-sobre-Virtualizacion-y-Evolucion-hacia-la-Nube-SOLA. [En línea] 2011. [Consultado el: 20 de 10 de 2015.] Disponible en: http://www.symantec.com/content/es/mx/enterprise/images/theme/evolution/symantec-Encuesta-sobre-Virtualizacion-y-Evolucion-hacia-la-Nube-SOLA.pdf.

5. MARTINEZ, R . Distribuciones de Linux | El rincón de Linux. [En línea] 2014. [Consultado el: 15 de 10 de 2015.] Disponible en: http://www.linux-es.org/distribuciones.

6. FERRER QUINTANA, J . Centro de precesamiento de datos:El Cerebro de nuestra sociedad. [En línea] 2009. [Consultado el: 30 de 11 de 2015.] Disponible en: http://www.academiadelanzarote.es/Discursos/Discurso%2034.pdf.

7. TWEAKHOUND. Stop Dual-Booting | TweakHound. [En línea] [Consultado el: 20 de 11 de 2015.] Disponible en: http://tweakhound.com/2015/09/23/stop-dual-booting/.

8. CADASCU. Tema 3:Virtualización. [En línea] [Consultado el: 27 de 10 de 2015.] Disponible en: http://cadascu.files.wordpress.com/2013/06/tema3_iso_virtualizacion.pdf.

9. MEDINA, J . Guía de Instalación de Servidor Linux basado en Ubuntu Server 12.04 LTS. [En línea] 2014. [Consultado el: 2 de 12 de 2015.] Disponible en: http://docs.tuxjm.net/guia-de-instalacion-de-servidor-linux-basado-en-ubuntu-server-1204-lts/pdf/guia-de-instalacion-de-servidor-linux-basado-en-ubuntu-server-1204-lts.pdf.

10. —.El manual del Administrador de Debian. [En línea] [Consultado el: 6 de 5 de 2016.] Disponible en: http://debian-handbook.info/browse/es-ES/stable/sect.virtualization.html.

11. BENHUTCHINGS. VirtualBox. [En línea] 10 de 12 de 2015. [Consultado el: 6 de 5 de 2016.] Disponible en: http://wiki.debian.org/VirtualBox.

12. —.Installation on Debian. [En línea] [Consultado el: 6 de 5 de 2016.] Disponible en: http://docs.docker.com/engine/installation/linux/debian/.

13. RANCHAL, Juan . Microsoft recomienda Ubuntu como el mejor Linux para la nube. [En línea] 6 de 10 de 2015. [Consultado el: 30 de 11 de 2015.] Disponible en: http://www.muycomputer.com/2015/10/06/microsoft-recomienda-ubuntu.

14. RED HAT, Inc. Solución de virtualización completa para servidores y escritorios. [En línea] 2016. [Consultado el: 10 de 5 de 2016.] Disponible en: http://www.redhat.com/es/technologies/virtualization.

15. —. Capítulo 2. Instalación de Red Hat Enterprise Virtualization. [En línea] 2016. [Consultado el: 10 de 5 de 2016.] Disponible en: http://access.redhat.com/documentation/es-ES/Red_Hat_Enterprise_Virtualization/3.0/html/Quick_Start_Guide/Quick_Start_Guide-Install_RHEV.html.

16. OPENVZ. Quick installation. [En línea] 2016. [Consultado el: 10 de 5 de 2016.] Disponible en: http://openvz.org/Quick_installation.

17. —.Install OpenVZ. [En línea] 2016. [Consultado el: 10 de 5 de 2016.] Disponible en: http://www.virtualizor.com/wiki/Install_OpenVZ.

18. DOCKER. Red Hat Enterprise Linux. [En línea] [Consultado el: 10 de 5 de 2016.] Disponible en: http://docs.docker.com/engine/installation/linux/rhel/.

19. RADVAN, S.; PARKER, D. Red Hat Enterprise Linux 5 Virtualization Guide. [En línea] 2014. [Consultado el: 10 de 5 de 2016.] Disponible en: http://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/5/pdf/Virtualization/Red_Hat_Enterprise_Linux-5-Virtualization-en-US.pdf.

20. MRKALJ, M . Hybrid Cloud: Benefits, roadblocks, favored vendors. report by Tech Pro Research . [En línea] Disponible en: http://www.bizcloudnetwork.com/study-hybrid-cloud-solutions/.

21. CEARLEY, D ;HILGENDORF, K . [En línea] 24 de 4 de 2014. Disponible en: http://www.gartner.com/doc/2718918/cloud-computing-innovation-key-initiative.

22. CISCO. Índice global de la nube de Cisco: Pronóstico y metodología, 2011-2016. 2012. Informe técnico.

23. PAREA, C. 5 tendencias que modelarán la inversión de TI - Revista ITNow. [En línea] [Consultado el: 19 de 11 de 2015.] Disponible en: http://revistaitnow.com/it_connect/5-tendencias-que-modelaran-la-inversion-de-ti.

24. KIRSCH, B. Tendencias y predicciones de virtualización de servidores para 2015. [En línea] [Consultado el: 18 de 11 de 2015.] Disponible en: http://searchdatacenter.techtarget.com/es/reporte/Tendencias-y-predicciones-de-virtualizacion-de-servidores-para-2015.

25. —. History of Virtualization . [En línea] 14 de 5 de 2014. [Consultado el: 15 de 11 de 2015.] Disponible en: http://www.infobarrel.com/History_of_Virtualization.

26. FERNÁNDEZ ROMERO, Y; GARCÍA POMBO, K . Virtualización ,Revista digital de las Tecnologías de la Información y las Comunicaciones (Telemática)., 2011,3, págs. 61-73.

27. JIMÉNEZ, A. ¿Qué es la virtualización del Centro de Datos? [En línea] 18 de 3 de 2009. [Consultado el: 8 de 10 de 2015.] Disponible en: http://www.itcio.es/virtualizacion-centro-datos/informes/1005213011402/virtualizacion-centro-datos.1.html.

28. TIM JONES, M. La anatomía de un Hipervisor en Linux. [En línea] [Consultado el: 15 de 10 de 2015.] Disponible en: http://www.ibm.com/developerworks/ssa/library/l-hypervisor/.

29. LÓPEZ MEDINA, A . Análisis de la virtualización de Sistemas Operativos. [En línea] 2010. [Consultado el: 20 de 10 de 2015.] Disponible en: http://www.maia.ub.es/~sergio/linked/an_lisis_de_la_virtualizaci_n_de_sistemas_operativos.pdf.

30. —. ¿Qué son los Hipervisores? [En línea] [Consultado el: 27 de 10 de 2015.] Disponible en: http://www.datakeeper.es/?p=716.

31. GARCÍA GARCÍA, A . Cloud Computing PaaS Platform Cloud ComPaaS. [En línea] 2010 [Consultado el: 4 de 11 de 2015.] Disponible en: http://riunet.upv.es/bitstream/handle/10251/14076/memoria.pdf?sequence=1

32. HWANG, K; DONGARRA, J; FOX, G. Cloud Computing: Virtualization Classes | TechNet Magazine. [En línea] 2012. [Consultado el: 4 de 11 de 2015.] Disponible en: http://technet.microsoft.com/en-us/magazine/hh802393.aspx.

33. VMWARE, Inc. VMware Understanding Full Virtualization, Paravirtualization, and Hardware Assist. [En línea] [Consultado el: 4 de 11 de 2015.] Disponible en: https://www.vmware.com/files/pdf/VMware_paravirtualization.pdf

34. MEDINA, J. Virtualización asistida por hardware. [En línea] [Consultado el: 5 de 11 de 2015.] Disponible en: http://bevirtual.blogspot.com/2007/04/nota-tcnica-virtualizacin-asistida-por.html.

35. TIM JONES,M. Virtual Linux — Institut Puig Castellar. [En línea] [Consultado el: 20 de 10 de 2015.] Disponible en: http://elpuig.xeill.net/Members/vcarceler/articulos/virtual-linux.

36. VON HAGEN, W. Professional Xen Virtualization. [En línea] 2008 [Consultado el: 20 de 10 de 2015.] Disponible en: http://pdf.th7.cn/down/files/1312/Professional%20Xen%20Virtualization.pdf

37. CERVIGON, D . ¿Soporta mi procesador la Virtualización Asistida por Hardware?. [En línea] [Consultado el: 20 de 10 de 2015.] Disponible en: http://blogs.technet.com/b/davidcervigon/archive/2007/04/27/soporta-mi-procesador-la-virtualizaci-n-asistida-por-hardware.aspx.

38. MATTHEWS, J. Running Xen: A Hands-On Guide to the Art of Virtualization. [En línea] [Consultado el: 6 de 11 de 2015.] Disponible en: http://old-list-archives.xenproject.org/archives/html/xen-users/2009-10/pdfW70n7sQyIr.pdf

39. ALEGSA, L . Definición de Virtualización a nivel de Sistema Operativo. [En línea] [Consultado el: 7 de 11 de 2015.] Disponible en: http://www.alegsa.com.ar/Dic/virtualizacion%20a%20nivel%20sistema%20operativo.php.

40. ESAÚ, A. Docker que es y sus principales características. [En línea] 5 de 5 de 2014. [Consultado el: 15 de 10 de 2015.] Disponible en: http://openwebinars.net/docker-que-es-sus-principales-caracteristicas/.

41. VOSIVANCOS, G . Docker and the era of the Containers. [En línea] 17 de 2 de 2015. [Consultado el: 28 de 10 de 2015.] Disponible en: http://blog.celingest.com/en/2015/02/17/docker-and-the-era-of-the-container.

42. LEE, A . Criterios y Pruebas para la Selección de Plataformas de Virtualización para Nubes Privadas con Soporte para IaaS. Facultad de Ingeniería Eléctrica, Instituto Superior Politécnico José Antonio Echeverría. La Habana : s.n., 2015. Trabajo de Diploma.

43. ISO 9126-1. Infromation Technology-Software Product Quality: Part 1,Quality Model. [En línea] 2000. [Consultado el: 6 de 2 de 2016.] Disponible en: http://www.cse.unsw.edu.au/~cs3710/PMmaterials/Resources/9126-1%20Standard.pdf. ISO/IE FDIS 9126-1:2000.

44. VEGA, S; et al. Desarrollo de un driver LXC para Opennebula. [En línea] 2016. [Consultado el: 1 de 5 de 2016.] Disponible en: http://www.informaticahabana.cu/sites/default/files/ponencias/TEL40.pdf.

45. HAMMERFEST. Phoronix Test Suite v6.0.0. [En línea] Disponible en: http://www.phoronix-test-suite.com/documentation/phoronix-test-suite.pdf.

46. PHORONIX, T. [En línea] 17 de 7 de 2015. Disponible en: http://es.softoware.net/system-utilities/download-phoronix-test-suite-for-linux.html.

47. JONES, R . netperf(1) - Linux man page. [En línea] [Consultado el: 1 de 3 de 2016.] Disponible en: http://linux.die.net/man/1/netperf.

48. IPERF.fr. Iperf - The TCP/UDP Bandwidth Measurement Tool. [En línea] [Consultado el: 8 de 2 de 2016.] Disponible en: http://iperf.fr.

49. JONES, R . Care and Feeding of Netperf. [En línea] 2012. [Consultado el: 15 de 3 de 2016.] Disponible en: http://www.netperf.org/svn/netperf2/trunk/doc/netperf.pdf.

50. OPENBENCHMARKING. GraphicsMagick Test Profile. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/graphics-magick.

51. GRAPHICSMAGICK. GraphicsMagick Image Processing System. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] www.graphicsmagick.org/GraphicsMagick.html.

52. OPENBENCHMARKING. FLAC Audio Encoding Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/encode-flac.

53. COALSON, J. FLAC - Free Lossless Audio Codec. [En línea] 2016. [Consultado el: 2016.] Disponible en: http://xiph.org/flac.

54. OPENBENCHMARKING. C-Ray Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/c-ray.

55. TSIOMBIKAS, J. futuretech. C-Ray Raytracing Benchmark. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://www.futuretech.blinkenlights.nl/c-ray.html.

56. OPENBENCHMARKING. Tachyon Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/tachyon.

57. TACHYON, P. Multiprocessor Ray Tracing System. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://jedi.ks.uiuc.edu/~johns/raytracer.

58. OPENBENCHMARKING. Himeno Benchmark Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/himeno.

59. —. Gzip Compression Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/compress-gzip.

60. GZIP. The gzip home page. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://www.gzip.org.

61. OPENBENCHMARKING. PostMark Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/postmark.

62. —. IOzone Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/iozone.

63. IOZONE. Iozone Filesystem Benchmark. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://www.iozone.org.

64. OPENBENCHMARKING. FS-Mark Test Profile. [En línea] 2016. [Consultado el: 6 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/fs-mark.

65. —. RAMspeed SMP Test Profile. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/ramspeed.

66. ALASIR. RAMspeed, a cache and memory benchmark. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://alasir.com/software/.

67. OPENBENCHMARKING. Stream Test Profile. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/stream.

68. MCCALPIN, J. Memory Bandwidth: Stream Benchmark Performance Results. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://www.cs.virginia.edu/stream.

69. OpenBenchmarking. CacheBench Test Profile. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/cachebench.

70. LLCbench Home Page. [En línea] 2016. [Consultado el: 7 de 2 de 2016.] Disponible en: http://icl.cs.utk.edu/projects/llcbench/index.html.

71. OpenBenchmarking. Loopback TCP Network Performance . [En línea] 2016. [Consultado el: 9 de 2 de 2016.] Disponible en: http://openbenchmarking.org/test/pts/network-loopback.

72. —. The Netperf Homepage. [En línea] [Consultado el: 8 de 2 de 2016.] Disponible en: http://www.netperf.org/netperf.

73. —. IPERF: How to test network Speed,Performance,Bandwidth. [En línea] [Consultado el: 8 de 2 de 2016.] Disponible en: http://www.slashroot.in/iperf-how-test-network-speedperformancebandwidth.

74. CANONICAL Ltd. iperf. [En línea] 2010. [Consultado el: 1 de 3 de 2016.] Disponible en: http://manpages.ubuntu.com/manpages/precise/en/man1/iperf.1.html.

75. VIRTUALIZATION Matrix: Compare VMware, Red Hat, Citrix & Microsoft. [En línea] [Consultado el: 2016 de 1 de 1.] www.virtualizationmatrix.com.

76. NIÑO, Jesús . Sistemas operativos monopuesto. [En línea] 2011. [Consultado el: 2 de 2 de 2016.] Disponible en: http://books.google.com.cu/.

77. INCAREDLIMA - Perú. Cuadrante mágico Gartner Virtualización de servidores 2015. [En línea] 2015. [Consultado el: 6 de febrero de 2016.] Disponible en: http://incared.net/noticias/cuadrante-magico-gartner-virtualizacion-2015/.

78. TALENS-OLIAG, S. Instituto Tecnológico de Informática (ITI). Herramientas de virtualización libres para sistemas GNU/Linux. [En línea] 25 de 9 de 2010. [Consultado el: octubre de 15.] Disponible en: http://www.uv.es/sto/charlas/2010_CIM/hvl-cim-2010.html/index.html.

79. UBUNTU Wiki. What's new in Ubuntu Server 14.04 LTS? | Ubuntu Insights. [En línea] 7 de 8 de 2015. [Consultado el: 9 de 2 de 2016.] Disponible en: http://insights.ubuntu.com/2014/04/17/whats-new-in-ubuntu-server-14-04-lts/.

80. DOCS.docker. Installation on Ubuntu,Docker. [En línea] [Consultado el: 9 de 2 de 2016.] Disponible en: http://docs.docker.com/engine/installation/linux/ubuntulinux/.

81. —. Disponible en: http://www.vmware.com/mx/virtualization/overview. [En línea] 17 de 4 de 2015. [Consultado el: 9 de 2 de 2016.] Disponible en: http://help.ubuntu.com/community/Xen.

82. LIBVIRT. libvirt: The virtualization API. [En línea] [Consultado el: 11 de 11 de 2015.] Disponible en: http://libvirt.org/.

83. PROXMOX Server Solutions GmbH. Proxmox Virtual Environment. [En línea] 2004 - 2016. [Consultado el: 4 de 4 de 2016.] Disponible en: http://www.proxmox.com/en/proxmox-ve.

84. UBUNTU Documentation Team wiki page. LXC. [En línea] [Consultado el: 2015 de 12 de 15.] Disponible en: http://help.ubuntu.com/lts/serverguide/lxc.html.

85. SUSE Documentation Team,SUSE Linux Enterprise Server 11 SP4. Virtualization with Linux Containers (LXC). [En línea] 14 de 7 de 2015. [Consultado el: 1 de 12 de 2015.] Disponible en: http://www.suse.com/documentation/sles11/singlehtml/lxc_quickstart/lxc_quickstart.html.

86. FUNTOO. Linux Containers. [En línea] 30 de 10 de 2015. [Consultado el: 15 de 12 de 2015.] Disponible en: http://www.funtoo.org/Linux_Containers.

87. OPENVZ. Comparison. [En línea] 23 de 12 de 2015 . [Consultado el: 4 de 4 de 2016.] Disponible en: http://openvz.org/Comparison.

88. DOCKER Documentation. [En línea] 2015. [Consultado el: 16 de 1 de 2016.] Disponible en: http://docs.docker.com.

89. SÁNCHEZ, L . Contenedores GNU-Linux Docker ASIR 14-15. [En línea] 23 de 6 de 2015. [Consultado el: 4 de 2 de 2016.]Disponible en: http://www.iesjacaranda-brenes.org/redmine/projects/docker/activity?from=2015-06-27

90. BERNSTEIN, D; Cloud Strategy Partners. Containers and Cloud: From LXC to Docker to Kubernetes,IEEE CLOUD COMPUTING. [En línea] 9 de 2014. [Consultado el: 25 de 1 de 2016.] Disponible en: http://ieeexplore.ieee.org/Xplore/home.jsp?reload=true.

91. NODO50. El proceso init. [En línea] [Consultado el: 1 de 5 de 2016.] Disponible en: http://www.nodo50.org/utlai/linux/27010714.htm.

92. Dirección de Informática y Comunicaciones | MINSAP. Cómo añadir tareas al Cron de Linux. [En línea] 2015. [Consultado el: 1 de 5 de 2016.] Disponible en: http://www.di.sld.cu/?p=9599.

93. CAROLINA, D; SIERRA, A. GUÍA METODOLÓGICA PARA LA GESTIÓN CENTRALIZADA DE REGISTRO DE EVENTOS DE SEGURIDAD EN PYMES. [En línea] 2007. [Consultado el: 1 de 5 de 2016.] Disponible en: http://pegasus.javeriana.edu.co/~regisegu/Docs/documento.pdf

94. UBUNTU. SSH ("Secure SHell"). [En línea] 2015. [Consultado el: 1 de 5 de 2016.] Disponible en: http://help.ubuntu.com/community/SSH.

95. BAUKES, M . Docker vs LXC. [En línea] 2016. [Consultado el: 4 de 4 de 2016.] Disponible en: http://www.upguard.com/articles/docker-vs-lxc.

96. DOCKER. Docker. [En línea] Disponible en: http://docs.docker.com/engine/reference/builder/.

97. CIO. 5 open source virtualisation technologies to watch - CIO. [En línea] 2015. [Consultado el: 27 de 1 de 2015.] Disponible en: http://www.cio.com.au/article/373314/5.

98. JAYASINGHE, D; MALKOWSKI, S; et al. IEEE TRANSACTIONS ON SERVICES COMPUTING, Variations in Performance and Scalability: An Experimental Study in IaaS Clouds Using Multi-Tier Workloads. [En línea] Abril-Junio de 2014. [Consultado el: 30 de 1 de 2016.] Disponible en: http://ieeexplore.ieee.org/Xplore/home.jsp?reload=true. VOL. 7, NO.2.

99. NAZARENO, G. IES Ingeniero de la Cierva. Proyecto de Innovación: Implantación y puesta a punto de la infraestructura (Virtualización de Servidores: KVM: Kernel-based Virtual Machine). [En línea] 2015. [Consultado el: 30 de 1 de 2016.]Disponible en: http://www.gonzalonazareno.org/cloud/material/KVM.pdf

100. GONZÁLEZ, E. Tesis de Maestría en Teleinformática, GESTOR DE MÁQUINAS VIRTUALES, Universidad de Mendoza. [En línea] 17 de 12 de 2010. [Consultado el: 30 de 1 de 2016.]Disponible en: http://www.um.edu.ar/web/documentos/UM-MTI-GonzalezE.pdf

101. ORACLE. Oracle VM VirtualBox. [En línea] 2015. [Consultado el: 5 de 2 de 2015.] Disponible en: http://www.oracle.com/us/technologies/virtualization/oraclevm/oracle-vm-virtualbox-ds-1655169.pdf.

102. —. Welcome to VirtualBox.org. [En línea] [Consultado el: 5 de 2 de 2016.] Disponible en: http://www.virtualbox.org/.

103. DATAMATION. Docker como alternativa en virtualización de servidores | Datamation. [En línea] 25 de 8 de 2014. [Consultado el: 15 de 4 de 2016.] Disponible en: http://www.datamation.com.ar/docker-como-alternativa-en-virtualizacion-de-servidores-5546.

104. SCHRODER, C . Linux Networking Cookbook . [En línea] 11 de 2008. [Consultado el: 4 de 4 de 2016.] Disponible en: http://sunshine.prod.uci.cu/gridfs/sunshine/books/linux_networking_cookbook.pdf.

105. GRAPHICSMAGICK Group. OpenMP in GraphicsMagick. [En línea] 2015. [Consultado el: 8 de 3 de 2016.] Disponible en: http://www.graphicsmagick.org/OpenMP.html.

106. GRAPHICSMAGICK. GraphicsMagick GM Utility. [En línea] 2015. [Consultado el: 30 de 3 de 2016.]

107. Hwang, J.; Zeng, S.; et al. A Component-Based Performance Comparison of Four Hypervisors. [En línea] [Consultado el: 1 de 4 de 2016.] Disponible en: http://faculty.cs.gwu.edu/~timwood/papers/im2013_tech.pdf.

108. ROSE, R. Linux Containers and the Future Cloud. [En línea] 2015. [Consultado el: 5 de 5 de 2016.] Disponible en: http://www.yumpu.com/en/document/view/30470818/netlec8-final.

109. DE JESÚS, J. . Cómo, Dónde y Por Qué IBM PureApplication se Adapta a su Nube. [En línea] 17 de 6 de 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://www.ibm.com/developerworks/ssa/websphere/techjournal/1506_dejesus/1506_dejesus-trs.html.

110. GITHUB.io. Using boot2docker to run Docker on a Mac or Windows. [En línea] [Consultado el: 2 de 5 de 2016.] Disponible en: http://odewahn.github.io/docker-jumpstart/boot2docker.html.

111. UBUNTU-es. Virtualbox en linea de comandos. [En línea] 2 de 7 de 2013. [Consultado el: 2 de 5 de 2016.] Disponible en: http://www.ubuntu-es.org/node/176583.

112. GARRON, G. Controla VirtuaBox desde la consola. [En línea] 17 de 5 de 2013. [Consultado el: 2 de 5 de 2016.] Disponible en: http://www.garron.me/es/gnu-linux/controla-maquinas-virtuales-virtualbox.html.

113. KIOSKEA.net. Instalar PhpVirtualBox bajo Ubuntu. [En línea] Comunidad de Infomación, 2014. [Consultado el: 2 de mayo de 2016.] Disponible en: http://es.ccm.net/faq/7022-instalar-phpvirtualbox-bajo-ubuntu

114. ARCHLINUX. PhpVirtualBox. [En línea] 28 de 12 de 2015. [Consultado el: 2 de mayo de 2016.] Disponible en: http://wiki.archlinux.org/index.php/PhpVirtualBox.

115. UBUNTU Documentation Team. libvirt. [En línea] [Consultado el: 2 de mayo de 2016.] Disponible en: http://help.ubuntu.com/lts/serverguide/libvirt.html#libvirt-management.

116. DEBIAN.org. PolicyKit. [En línea] 2010. [Consultado el: 2 de mayo de 2015.] Disponible en: http://wiki.debian.org/PolicyKit.

117. OCHOBITSHACENUNBYTE. PROXMOX lanza su versión 4.0 con importantes mejoras. [En línea] 9 de 10 de 2015. [Consultado el: 2 de mayo de 2016.] Disponible en: http://www.ochobitshacenunbyte.com/2015/10/09/proxmox-lanza-version-4-0-importantes-mejoras/.

118. PROXMOX. Proxmox Virtual Environment. [En línea] 5 de 2016. [Consultado el: 2 de 5 de 2016.] Disponible en: http://pve.proxmox.com/wiki/Main_Page.

119. HIDALGO, J. Google lanza Kubernetes. [En línea] 2014. [Consultado el: 2 de 5 de 2016.] Disponible en: http://wwwhatsnew.com/2014/06/11/google-lanza-kubernetes-software-de-codigo-abierto-para-la-gestion-de-maquinas-en-centros-de-datos-para-la-nube/.

120. GUERRA , G. Tendencias en seguridad informática para 2016. [En línea] 2015. [Consultado el: 1 de 5 de 2016.] Disponible en: http://www.netmedia.mx/b-secure/tendencias-en-seguridad-informatica-para-2016/.

121. DEBIAN. AppArmor. [En línea] 2015. [Consultado el: 1 de 5 de 2016.] Disponible en: http://wiki.debian.org/AppArmor.

122. DEBIAN-handbook.info. Introduction to SELinux. [En línea] [Consultado el: 1 de 5 de 2016.] Disponible en: http://debian-handbook.info/browse/stable/sect.selinux.html.

123. INCIBE. Vulnerabilidad en Docker y docker-py. (CVE-2014-5277). [En línea] 2014. [Consultado el: 1 de 5 de 2016.] Disponible en: http://www.incibe.es/vulnDetail/CERT/Alerta_Temprana/Actualidad_Vulnerabilidades/detalle_vulnerabilidad/CVE-2014-5277.

124. DOCKER Inc. Introduction to Docker. [En línea] 2014. [Consultado el: 2 de 5 de 2016.] Disponible en: http://wiki.zenoss.org/download/core/drich_slides/DockerSlides.pdf.

125. CIS, D. CIS Docker 1.6 Benchmark. [En línea] 22 de 4 de 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://benchmarks.cisecurity.org/tools2/docker/CIS_Docker_1.6_Benchmark_v1.0.0.pdf.

126. CERT de Seguridad e Industria. Vulnerabilidad en lxclock.c en LXC (CVE-2015-1331). [En línea] 22 de 10 de 2015. [Consultado el: 1 de 5 de 2016.] Disponible en: http://www.incibe.es/vulnDetail/CERT/Alerta_Temprana/Actualidad_Vulnerabilidades/detalle_vulnerabilidad/CVE-2015-1331.

127. —. Vulnerabilidad en lxc-start en lxc (CVE-2015-1335). [En línea] 1 de 10 de 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://www.incibe.es/vulnDetail/CERT/Alerta_Temprana/Actualidad_Vulnerabilidades/detalle_vulnerabilidad/CVE-2015-1335.

128. OPENSUSE. Guía-Instalación de XEN en opensuse. [En línea] [Consultado el: 2 de 5 de 2016.] Disponible en: http://cuchillac.net/archivos/pre_virtual_cloud_computing/1_cloud_privadas/guias/guia_instalar_xen.pdf.

129. MITRE. CVE-2015-8341. [En línea] 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-8341.

130. —. CVE-2015-8339. [En línea] 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-8339.

131. ROPERO, A. Vulnerabilidades de denegación de servicio en Xen. [En línea] 5 de 12 de 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://unaaldia.hispasec.com/2015/12/vulnerabilidades-de-denegacion-de.html.

132. GEFFNER, J. Virtualized Environment Neglected Operations Manipulation. [En línea] 21 de 5 de 2015. [Consultado el: 2 de 5 de 2016.] Disponible en: http://venom.crowdstrike.com/.

133. MITRE, C. CVE Details. [En línea] 2015-2016. [Fecha de Consulta: 2 de mayo de 2016.] Disponible en: http://www.cvedetails.com/vulnerability-list/vendor_id-93/product_id-20406/Oracle-Vm-Virtualbox.html.

134. VMWARE. NSX. [En línea] 2016. [Consultado el: 3 de 5 de 2016.] Disponible en: http://www.vmware.com/latam/products/nsx.

135. MANAGEMENT, S. La nube: oportunidades y retos para los integrantes de la cadena de valor. [En línea] 2012. [Consultado el: 26 de 11 de 2015.] Disponible en: http://www.managementsolutions.com/PDF/ESP/La-nube.pdf.