Universidad de Mendoza

Facultad de Ingeniería

Tesis de Maestría en Teleinformática

GESTOR DE MÁQUINAS VIRTUALES

Ing. Érica .B. González

Director de Tesis:
Mg. Ing. Diego Navarro
17 /12/2010

AGRADECIMIENTOS

Primero y antes que nada quiero dar gracias a Dios por estar conmigo en cada paso que doy, por fortalecer mi corazón e iluminar mi mente y por haber puesto en mi camino a aquellas personas que han sido mi soporte y compañía durante todo el periodo de estudio.

Quiero agradecerle a mi esposo Gastón por estar a mi lado y en especial a los dos solcitos que iluminan mi día a día. Delfina y María Jesús.

A mi papá que desde el cielo me da las fuerzas para llevar a cabo mis metas, a mi mamá por su amor, su apoyo y sobre todo por seguir dedicándome su tiempo incondicionalmente.

A mis hermanos Silvia, Sonia y Ariel por su preocupación y porque son los peldaños más importantes de mi vida.

A Diego Navarro, Carlos Palacio, Pablo Gómez, Laurita Missaglia, Verónica Cisterna, Ceferino Mulet y a toda la dirección que integra la maestría por permitirme desarrollar como persona, alumno y profesional.

ÍNDICE

I. PRÓLOGO

Para poder llevar a cabo el presente trabajo se estudió la problemática de la mayoría de los organismos y organizaciones con respecto a la expansión de la infraestructura de TI (Tecnologías de la Información) que están sufriendo infaliblemente por el incremento de servidores para ejecutar nuevas aplicaciones, lo que a su vez puede dar como resultado numerosos servidores ociosos, heterogéneos e inclusive incompatibles entre ellos, mayores costos de gestión de la red y menor fiabilidad.

Por lo general la tendencia de la mayoría de estas organizaciones para hacer frente a estos desafíos es migrar a un entorno virtualizado. En informática, virtualización se refiere a la abstracción de los recursos de una computadora, esto se logra a través de una capa abstracta llamada hypervisor o VMM (Monitor de máquina virtual) que permite que múltiples máquinas virtuales con sistemas operativos heterogéneos puedan ejecutarse individualmente en la misma máquina física. Cada máquina virtual tiene su propio hardware virtual (por ejemplo, RAM, CPU, DISCO) sobre éste se instala el sistema operativo y las aplicaciones.

En este trabajo de tesis se estudiará, qué es virtualización, las distintas formas de implementarla, ventajas y desventajas que tiene una organización a migrar a un entorno virtualizado y como aporte de implementación se desarrollará un gestor de máquinas virtuales.

El objetivo es facilitar la gestión de entornos virtuales para el cliente y poder administrarlos en forma remota desde una interfaz WEB amigable utilizando OPENVZ como técnica de virtualización a nivel de sistema operativo.

Si bien la virtualización ofrece diversas soluciones y facilidades, el administrador de tecnología debe contar con los planes de contingencia adecuados y con todos los recaudos necesarios adicionales que escapan a la virtualización para que una organización funcione lo mejor posible.

II. INTRODUCCIÓN

Hace algunos años el incremento de implementaciones de servidores y estaciones de trabajo genera nuevos problemas operacionales y de infraestructura en el departamento de tecnología.

Entre estos problemas se incluyen el grado normal de utilización de los servidores es sólo de un 10% a un 15% de la capacidad. El grado de utilización disminuye con el tiempo y por otra parte los requisitos de recursos para muchas cargas de trabajo típicas de servidor como impresión, correo electrónico, servidores Web internos y controladores de dominio no han cambiado mucho.
El bajo grado de utilización de los servidores aumenta la complejidad de la gestión del hardware y disminuye el retorno de la inversión ROI (Rendimiento del Capital Invertido) del cliente. Comprar hardware nuevo cuando el existente no está apenas utilizado, gestionar una infraestructura de TI (Tecnologías de la Información) cada vez más grande y provisionar ese hardware plantea desafíos innecesarios a los clientes.

A medida que los entornos informáticos se hacen más complejos, aumenta el nivel de especialización de la formación y la experiencia que necesita el personal de gestión de infraestructuras y los costos asociados al mismo. Las organizaciones gastan cantidades desproporcionadas de dinero y recursos en tareas manuales ligadas al mantenimiento de los servidores y aumenta la necesidad de personal para realizarlas.

El costo de mantenimiento de las estaciones de trabajo de los usuarios finales es elevadísimo, dado que la gestión y la seguridad de las mismas plantean numerosos desafíos. Controlar un entorno de máquinas distribuidas y aplicar políticas de gestión, acceso y seguridad sin perjudicar la capacidad del usuario de trabajar con

eficacia es complejo y costoso. Se tienen que aplicar continuamente muchos parches y actualizaciones para eliminar las vulnerabilidades de seguridad.

Los planes de contingencia y protección ante desastres resultan insuficientes dado que las organizaciones se ven cada vez más afectadas por las paradas de las aplicaciones en servidor crítico y la falta de acceso a las estaciones de trabajo. La amenaza de ataques a la seguridad o desastres naturales han acentuado la importancia de la planificación de la continuidad de la organización tanto en lo relativo a computadoras como a servidores.

El propósito fundamental de la virtualización es simplificar estos problemas. Los gerentes de informática se han dado cuenta del potencial de la virtualización, que hoy es una realidad y el tiempo preciso para que las organizaciones comiencen a implementarla en su centro de procesamiento de datos.
La consolidación de servidores, que básicamente significa utilizar un sólo servidor en lugar de varios, conlleva a la reducción del uso de cableado, ahorro de energía eléctrica, reducción en requerimientos de refrigeración, menor ocupación de espacio y esto significa un inminente costo total menor para la organización y unos de los factores más importantes para los gerentes de informática y para la toma de decisiones para migrar a un entorno virtualizado.

Teniendo en cuenta los problemas mencionados anteriormente que sobrellevan algunas organizaciones y la virtualización como una solución alternativa a sus necesidades, este trabajo de tesis tiene como:

Objetivo general:

Desarrollar un gestor de servidores virtuales basado en un modelo cliente servidor utilizando sistema operativo Linux y una técnica de virtualización a nivel de sistema operativo.

Objetivos específicos:

- Poder instanciar, encender, apagar, modificar y monitorizar recursos de cada una de las máquinas virtuales en tiempo real según las necesidades del usuario. Tarea destinada al administrador de tecnología.

- Ofrecer al usuario Host virtuales distintos pre-configurados mediante una interfaz Web.

- Brindar al usuario la posibilidad de monitorizar ciertos recursos de su servidor y acceder remotamente desde cualquier lugar.

En la actualidad esta tecnología que según su concepto y características facilita la gestión de datos en menos espacio y por menos costo energético, entre algunas ventajas, estaría impactando de forma significativa en el mercado de servidores y aún más en la forma de cómo se organizan hoy los centros de cómputos.

III. DEFINICIÓN DE MÁQUINA VIRTUAL

El concepto de máquina virtual surge con el sistema VM/370 de IBM en 1972. La idea principal es la de permitir ejecutar varios sistemas operativos simultáneamente sobre el mismo hardware. Para ello separa las dos funciones básicas que realiza un sistema de tiempo compartido: multiprogramación y abstracción del hardware.

Las computadoras modernas son una de las estructuras más avanzadas del diseño humano y sólo son posibles gracias a nuestra habilidad para manejar la complejidad extrema. Las computadoras contienen muchos chips, cada uno con ciento de millones de transistores. Éstos se interconectan y se combinan con dispositivos de entrada/salida de alta velocidad y con una infraestructura de red que forman plataformas con las que el software puede operar. Los sistemas operativos, aplicaciones y bibliotecas, software de gráficos y red cooperan para mantener un ambiente poderoso para la administración de datos, la educación, la comunicación y muchas otras aplicaciones.

La clave para manejar esta complejidad en las computadoras es la división en los niveles de abstracción separados por interfaces bien definidas. Los niveles de abstracción permiten que los detalles de implementación en niveles más bajos de un diseño puedan ser ignorados o simplificados, facilitando por tanto el diseño de componentes en los niveles más altos. Por ejemplo: los detalles del disco duro (que es dividido en sectores y pistas) son abstraídos por el sistema operativo, el disco frente al software de aplicación se muestra como un conjunto de archivos de tamaño variable Una aplicación pueda entonces crear, escribir, y leer los archivos, sin el conocimiento de cómo el disco duro se construye y se organiza.
Los niveles de abstracción son organizados en una jerarquía, lo niveles más bajos se llevan a cabo en el hardware y los niveles más altos en el software. Al hablar de

virtualización estamos interesados en los niveles cercanos a la frontera hardware/software.
En los niveles del hardware, todos los componentes son físicos, tienen propiedades reales y sus interfaces son definidas de manera que varias partes puedan conectarse físicamente.

En los niveles del software, los componentes son lógicos, con menos restricciones basadas en las características físicas.
Existen niveles donde el software está separado de la máquina en la que corre. El software de computadora es ejecutado por una máquina.

Desde la perspectiva del sistema operativo, una máquina está compuesta en su mayoría por el hardware, incluyendo uno o más procesadores que ejecutan un conjunto específico de instrucción, memoria real, y dispositivos de entrada y salida. Sin embargo no restringimos el uso del término de máquina a sólo los componentes de hardware de una computadora. Desde el punto de vista de programas de aplicación por ejemplo, la máquina es una combinación del sistema operativo y las partes del hardware accesibles a través de instrucciones binarias a nivel de usuario.

Otro aspecto a tener en cuenta es el manejo de interfaces bien definidas.
Las interfaces bien definidas permiten que las tareas de diseño de las computadoras estén separadas para que los diseñadores de software y hardware puedan trabajar de manera más o menos independientemente.
El conjunto de instrucciones es una de esas interfaces. Por ejemplo, los diseñadores de Intel y AMD desarrollan microprocesadores que implementan el conjunto de instrucciones Intel IA32, mientras que los desarrolladores de software de Microsoft desarrollan compiladores que mapean lenguajes de alto nivel al mismo conjunto de instrucción.

La interfaces del sistema operativo, definidas como un conjunto de llamadas, es otra interfaz estandarizada importante en el sistema de computación.
Las interfaces bien definidas permiten el desarrollo de subsistemas de computación interactivos en diferentes compañías en tiempos diferentes, a veces en años separados.

Los desarrolladores de software de aplicación no necesitan estar al tanto de los cambios del sistema operativo, y el hardware y software pueden ser actualizados acorde a planificaciones diferentes. El software puede correr en diferentes plataformas implementando el mismo conjunto de instrucción.

A pesar de sus ventajas, las interfaces bien definidas pueden ser limitables. Los subsistemas y componentes diseñados para las especificaciones de una interfaz no funcionarán con el mismo diseño para otra interfaz.

Los programas de aplicación cuando están distribuidos como programas binarios, están sujetos a un conjunto de instrucción específico y a un sistema operativo. Un sistema operativo está sujeto a una computadora que implementa un sistema de memoria específico y un sistema de entrada y salida.

Además de la interfaz hardware/software, las cuestiones sobre la gestión de los recursos de hardware también pueden limitar la flexibilidad del sistema. Los sistemas operativos desarrollados para una arquitectura específica (uniprocesador o multiprocesador de memoria compartida) están diseñados para gestionar recursos directamente. Esto limita la flexibilidad del sistema, no sólo en términos de las aplicaciones de software disponibles sino en términos de seguridad y aislamiento de fallos especialmente cuando el sistema es compartido por múltiples usuarios o grupos de usuarios.

La virtualización minimiza estas restricciones e incrementa la flexibilidad. Cuando un sistema o subsistema como por ejemplo CPU, memoria, dispositivo de entrada/salida es virtualizado, su interfaz y todos los recursos visibles a través de la interfaz son mapeados sobre la interfaz y recursos de un sistema real donde se están implementando. Por consiguiente el sistema real se transforma de manera que parece ser un sistema diferente, un sistema virtual o incluso un conjunto de múltiples sistemas virtuales.

Formalmente la virtualización involucra la construcción de un isomorfismo que mapea un sistema virtual guest a un host real. Este isomorfismo mapea el estado guest al estado del host y para una secuencia de operaciones, e que modifica el estado guest (la función e modifica el estado Si al estado Sj) hay una secuencia de operaciones e' en el host que realiza un modificación equivalente en el estado del host (cambia S'i a S'j).

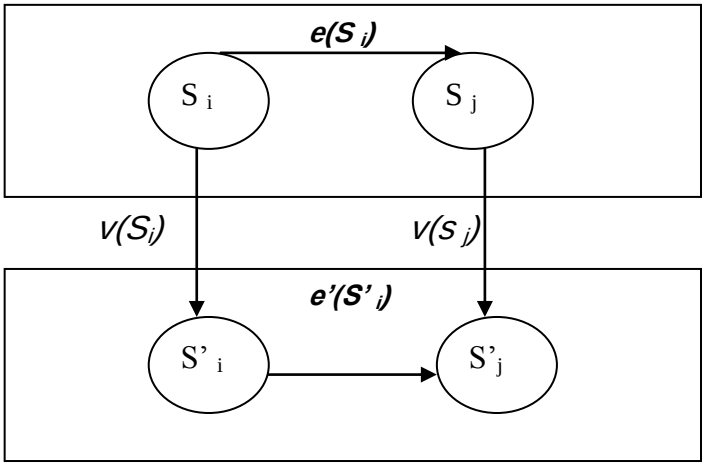


Figura 1. Virtualización. Formalmente, virtualización es la construcción de un isomorfismo entre el sistema guest y Host: e' o V(S j) = V o e(S j)[1].

Aunque tal isomorfismo puede usarse para caracterizar tanto abstracción como virtualización.

1. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5

La diferencia entre la virtualización y la abstracción es que la virtualización no necesariamente oculta detalles, el nivel de detalles del sistema virtual es casi siempre el mismo que el del sistema real.

El concepto de virtualización no sólo puede ser aplicado a subsistemas como discos, memorias, red sino también a máquinas enteras.
Una máquina virtual se implementa agregando una capa de software a la máquina real para soportar el diseño de arquitectura de máquina virtual.
En general, una máquina virtual puede ocultar las exigencias de compatibilidad de la máquina real y de los recursos de hardware para obtener un grado más alto de portabilidad de software y flexibilidad.

Existe una amplia variedad de máquinas virtuales para proporcionar una amplia variedad de beneficios. Se pueden implementar múltiples máquinas virtuales replicadas en una plataforma de hardware para proporcionar a cada usuario o grupo de usuario sus propios entornos de sistemas operativos. Los diferentes entornos de sistema (posiblemente con sistemas operativos distintos) también proporcionar aislamiento y seguridad mejorada. Un gran servidor multiprocesador puede dividirse en servidores virtuales más pequeños manteniendo la capacidad de balancear el uso de los recursos a través del sistema. Las máquinas virtuales también pueden usar técnicas de emulación para soportar compatibilidad multi-plataforma haciendo que el software escrito para una plataforma se ejecute en otra. Además de la emulación las máquinas virtuales pueden proporcionar optimizaciones dinámicas de los binarios de los programas.

Los ejemplos descritos anteriormente se construyes para coincidir con arquitecturas de máquinas reales existentes. Pero también hay máquinas virtuales sin máquina real correspondiente. Es común inventar máquinas virtuales a la medida de nuevos lenguajes de alto nivel. Los programas escritos en estos lenguajes se compilan en binarios dirigidos a la máquina virtual. Entonces

cualquier máquina real para la que exista una implementación de la máquina virtual puede ejecutar el código compilado. Un ejemplo es JAVA y su máquina virtual.

IV. ARQUITECTURA DE COMPUTADORA

El término de arquitectura, cuando se aplica a computadoras se refiere a la funcionalidad y la apariencia de un sistema de computadora o subsistema pero no a los detalles de su implementación. La arquitectura se describe a través de una especificación de una interfaz y el comportamiento lógico de los recursos manipulados vía la interfaz.

Los niveles de abstracción de un sistema de computadora corresponden a la implementación de ambas capas de software y hardware.

En el software hay una interfaz entre las aplicaciones y las bibliotecas (interfaz 2) y otra interfaz está sobre el sistema operativo (interfaz 3).

Las interfaces de hardware incluyen una arquitectura de entrada/salida que describe la información que manejan los drivers de los dispositivos de entrada/salida (interfaz 11), una arquitectura de memoria de hardware que describe la forma que las direcciones se traducen (interfaz 9), una interfaz de memoria que accede a las señales que deja el procesador (interfaz 12), y otra para las señales que alcanzan los chips de DRAM en memoria (interfaz 14). El sistema operativo se comunica con los dispositivos de entrada/salida a través de una sucesión de interfaces: 4, 8, 10, 11, y 13.

La interfaz ISA (Arquitectura del conjunto de instrucciones) marca la división entre hardware y software, está compuesta por las interfaz 7 y 8.
Hay dos partes importantes de ISA en la definición de máquinas virtuales. La primera incluye aspectos de ISA visibles a los programas de aplicación. Esto hace referencia a USER ISA La segunda parte incluye aspectos visibles al software supervisor como sistemas operativos, encargados de administrar los recursos de

hardware. Esto hace referencia a SYSTEM ISA Por supuesto que el software supervisor también puede usar todos los elementos de USER ISA. La interfaz 7 consiste en el USER ISA solamente y la interfaz 8 consiste en el USER ISA y SYSTEM ISA.

ABI (Interfaz binaria de aplicación) consiste en la interfaz 3 y 7 y provee un programa con accesos a los recursos de hardware y servicios disponibles en el sistema. Una interfaz importante relacionada es la API (Interfaz de programación de aplicaciones) que consiste en la interfaz 2 y 7.

ABI tiene dos componentes, la primera es un conjunto de todas instrucciones de usuarios (interfaz 7) no están incluidas las instrucciones de sistema en la ABI. En el nivel de ABI todos los programas de aplicación interactúan con los recursos de hardware compartidos indirectamente, invocando al sistema operativos vía interfaz de llamadas al sistema. (Interfaz 3) con la segunda componente de ABI.

Las llamadas al sistema proveen un conjunto de operaciones específicas que el sistema operativo puede realizar en nombre del programa de usuario.
La interfaz de llamadas al sistema es básicamente implementada por una instrucción que transfiere el control al sistema operativo de manera similar que un procedimiento o llamadas de subrutinas. Excepto una llamada de direccionamiento que es restringida a un direccionamiento especifico en el sistema operativo. Un programa binario compilado en una ABI específica puede correr sin modificaciones en un sistema con la misma ISA y sistema operativo.
ABI es definida en base a HLL (lenguaje de alto nivel). La clave de una API es la biblioteca estándar, de manera que llamadas de una aplicación pueda invocar varios servicios en el sistema, incluyendo todos los provistos por el sistema operativo.
Una API definida en código fuente permite que las aplicaciones escritas para ella puedan ser migradas a cualquier sistema que soporten la misma API.

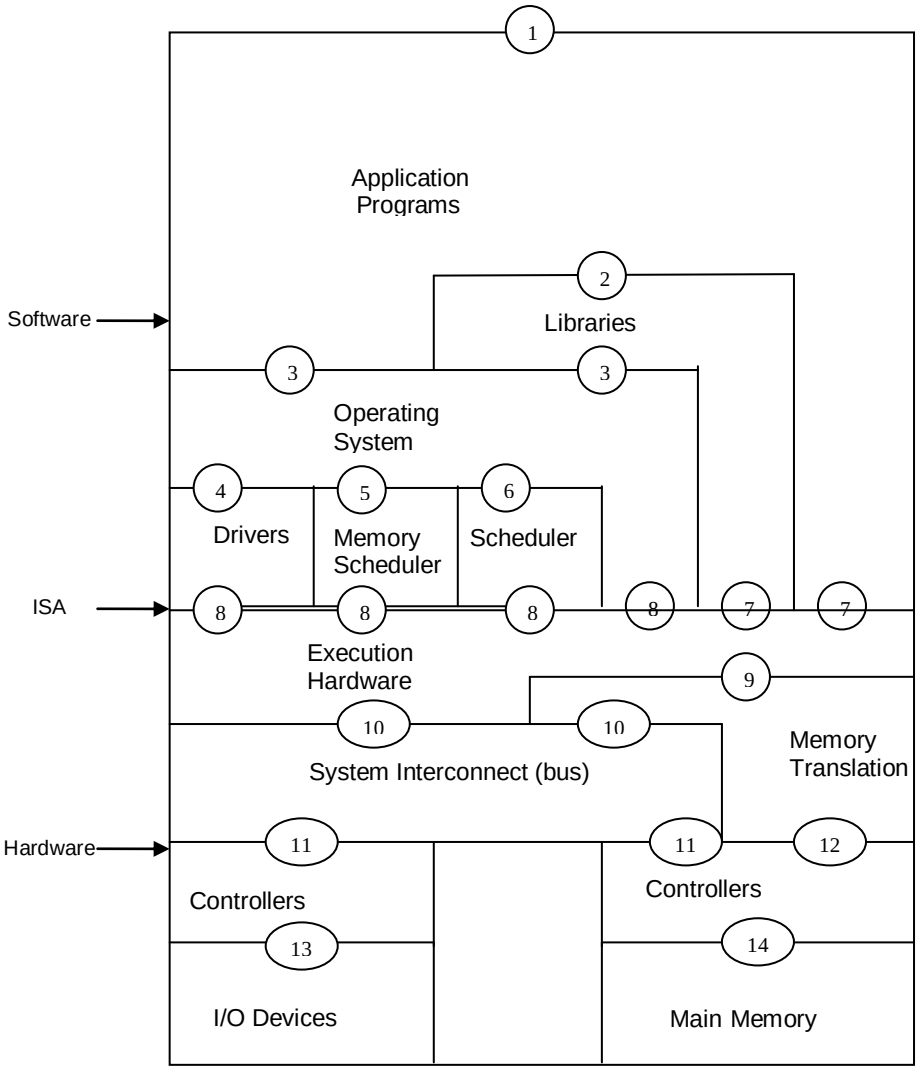


Figura 2: Implementación de cómo se comunican las capas verticalmente mediante las interfaces.[2]

Una API es una abstracción de los detalles de los servicios de implementación, especialmente sobre todo aquellos que involucran hardware privilegiado.

2. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5.

I. TIPOS DE ARQUITECTURAS DE COMPUTADORA

Una máquina virtual desde la perspectiva de un proceso que ejecuta un programa usuario, consiste en un espacio de direcciones lógicas de memoria al que se le ha asignado un proceso junto con los registros de usuarios y las instrucciones que permiten la ejecución del código que pertenece al proceso.

El sistema entrada/salida es visible sólo a través del sistema operativo y la única manera que el proceso puede interactuar con él es a través de las llamadas de sistema operativo (algunas veces a través de las bibliotecas) que se ejecutan como parte de los procesos.
Por lo tanto podemos decir que una máquina virtual de procesos se ejecuta como un proceso normal dentro de un sistema operativo y soporta un sólo proceso. La máquina se inicia automáticamente cuando se lanza el proceso que se desea ejecutar y se para cuando éste finaliza. Su objetivo es el de proporcionar un entorno de ejecución independiente de la plataforma de hardware y del sistema operativo, que oculte los detalles de la plataforma subyacente y permita que un programa se ejecute siempre de la misma forma sobre cualquier plataforma.

El ejemplo más conocido actualmente de este tipo de máquina virtual es la máquina virtual de Java. Otra máquina virtual muy conocida es la del entorno .Net de Microsoft que se llama "Common Language Runtime".
Los procesos son normalmente transitorios, son creados por un período de tiempo, quizás creen otros procesos (fork) y terminen.

Una máquina virtual desde la perspectiva de un sistema operativo, un sistema completo es soportado por la máquina real. El sistema es un ambiente de ejecución completo que soporta un número de procesos simultáneos que pertenecen a usuarios diferentes. Todos los procesos comparten el sistema de archivos y otros recursos de entrada/salida.

El sistema es persistente, excepto que se reinicie, le almacena a los procesos memoria física y recursos de entrada/salida y permite que éstos interactúen con otros recursos a través del sistema operativo que es parte del sistema.
Desde la perspectiva de un sistema, la máquina es implementada por el hardware solo y la ISA provee la interfaz entre el sistema y la máquina.

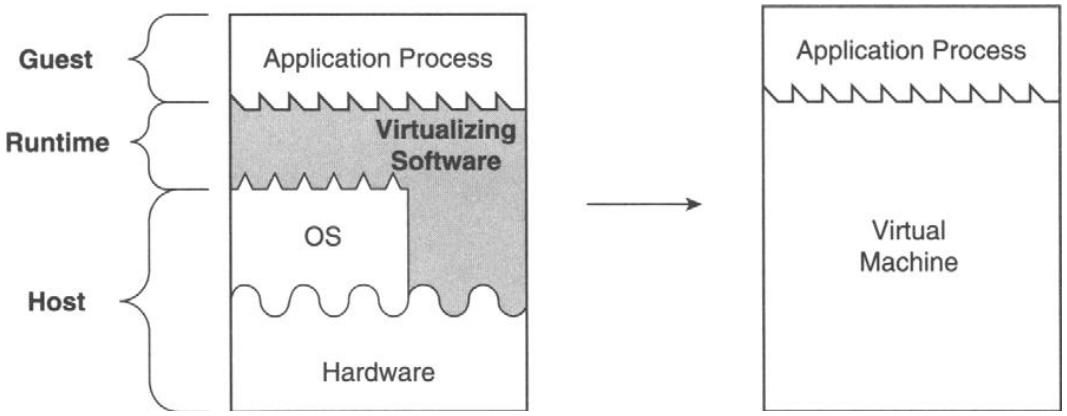


Figura 3: Máquina virtual de sistema. El software de virtualización traduce ISA usada por una plataforma de hardware a otra, formando una máquina virtual de sistema, capaz de desarrollar un software para diferentes conjuntos de hardware.[3]

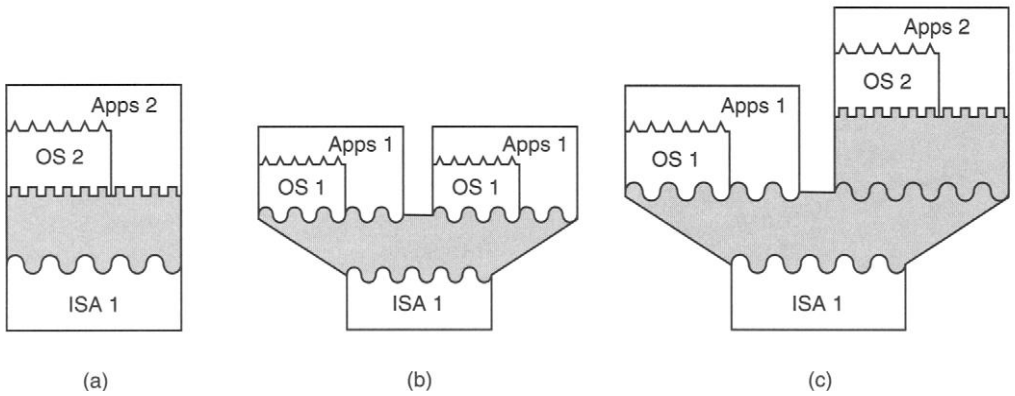


Figura 4: Máquina virtual de proceso. A) emula un conjunto de instrucciones a otra. B) Replica una máquina a virtual a múltiples sistemas operativos que pueden ser soportados simultáneamente. C) composición de un software de virtualización para dar origen a un sistema más complejo.[4]

3 y 4. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5

En la práctica una máquina virtual ejecuta un software (que puede ser o un proceso o un sistema completo) dependiendo del tipo de máquina.
La máquina virtual se implementa como una combinación de máquina real y de un software de virtualización.
Como característica del isomorfismo descrita anteriormente el proceso de virtualización consiste en dos partes:

I. Mapeo de recursos virtuales. Puede ser registro de memoria o archivos o recursos reales de computadoras.
II. El uso de instrucciones y/o llamadas del sistema de la máquina real para llevar a cabo las acciones definidas por las instrucciones y/o llamadas del sistema de la máquina virtual. Ejemplo: emulación de una máquina virtual ABI o ISA.

En una máquina virtual de proceso, el software de virtualización está en la interfaz ABI encima de la combinación de hardware y sistema operativo. El software de virtualización emula tanto instrucciones de nivel de usuario como las llamadas al sistema operativo.

Con respecto a la terminología, normalmente se le llama a la máquina real HOST y al software que corre en la máquina virtual GUEST.
La plataforma real corresponde a la máquina real y el nombre del software de virtualización depende del tipo de máquina virtual que está siendo implementada.
En el caso de MV proceso, el software de virtualización a menudo se lo denomina RUNTIME SOFTWARE.
El RUNTIME es creado para soportar que corra un proceso guest y que éste corra sobre un sistema operativo.

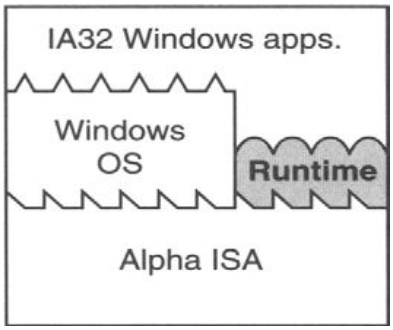


Figura 5: Proceso de virtualización emula aplicaciones guest. El sistema Digital FX!32 permite que las aplicaciones IA/32 Windows puedan correr en plataformas ALPHA.[5]

En una máquina virtual de sistema, el software de virtualización se encuentra entre el hardware de la máquina y el software convencional. En este ejemplo en particular el software de virtualización emula el hardware ISA para que el software convencional vea un ISA diferente al soportado por el hardware. En muchos sistemas el guest y el host ejecutan el mismo ISA.

Al software de virtualización se lo denomina VMM (virtual machine monitor) término desarrollado en 1960.

II. CARACTERÍSTICAS DE MÁQUINA VIRTUAL DE PROCESO

Las máquinas virtuales a nivel se proceso proporcionan a las aplicaciones de usuario una interfaz ABI. En sus varias implementaciones pueden proporcionar replicación, emulación y optimización.

MULTIPROGRAMACIÓN:

La combinación de la interfaz de las llamadas del sistema operativo y del conjunto de instrucciones de usuario forma la máquina que ejecuta un proceso usuario. La mayoría de los sistemas operativos pueden soportar múltiples procesos usuarios corriendo simultáneamente debido a la característica de multiprogramación donde

5. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5.

cada proceso usuario tiene la ilusión de tener una máquina completa para él sólo. Cada proceso tiene su propio espacio de direcciones y acceso a la estructura de archivos.

El sistema operativo es quien comparte el hardware y administra los recursos para hacer posible esto. En efecto el sistema operativo proporciona una máquina virtual o nivel de proceso para cada una de las aplicaciones que se ejecutan concurrentemente.

EMULADORES Y TRADUCTORES BINARIOS:

Un problema desafiante para una máquina virtual de proceso es soportar programas binarios compilados para un conjunto de instrucciones distintas que las que son ejecutadas por el hardware del Host. El sistema operativo sobre el que se ejecuta el programa binario puede ser el mismo que aquel sobre el que se ejecuta normalmente aunque compilado para una arquitectura distinta.

El método de emulación más real es la interpretación. Un programa intérprete ejecuta las ISA destino, descodifica y emula la ejecución de instrucciones fuentes. Esto puede ser un proceso relativamente lento si se necesita que cada instrucción destino sea interpretada.

Para mejorar el rendimiento se usa la traducción binaria, se toma un bloque de instrucciones fuente y se convierte a las instrucciones destinos que ejecutan funciones equivalentes. Puede haber una carga alta en el proceso de traducción pero una vez que el bloque de instrucciones es traducido, las instrucciones traducidas son almacenadas y ejecutadas repetidamente mucho más rápido de las que son interpretadas. Así funcionan los traductores binarios y es una característica importante de máquina virtual de proceso.

La interpretación y la traducción binaria tienen características de rendimiento distinto.
La interpretación tiene una baja sobrecarga inicial pero consume tiempo debido a que la instrucción es emulada.
La traducción binaria tiene una alta sobrecarga inicial cuando realiza las traducciones pero es más rápida para cada una de las ejecuciones repetidas.

En consecuencia algunas máquinas virtuales usan estrategia de emulación combinando con la técnica profiling. Inicialmente un bloque de instrucciones fuentes son interpretadas y profiling es usada para determinar que secuencia de instrucciones son ejecutadas frecuentemente. Luego el bloque ejecutado frecuentemente puede ser traducido. Algunos sistemas usan estos códigos optimizadores en el código traducido si la frecuencia de ejecución es alta.

En la mayoría de las máquinas virtuales las fases de interpretación y traducción binaria ocurren por encima del curso de ejecución de un programa.

OPTIMIZADORES BINARIOS PARA ISA IGUALES:
La mayoría de los optimizadores binarios dinámicos no sólo traducen código fuente a destino sino también realizan optimizaciones en el código. Esto es naturalmente en máquinas donde el conjunto de instrucciones usadas por el Host y el Guest es el mismo y la optimización de un programa binario es el propósito de la máquina virtual. Los optimizadores binarios para ISA iguales son implementados de una manera similar a emular máquinas virtuales, incluyendo optimización y software de código optimizado almacenado. Un ejemplo de un optimizador binario dinámico es Dynamo System, de Hewlett Packard.

MÁQUINA VIRTUAL DE LENGUAJE DE ALTO NIVEL. INDEPENDENCIA DE LA PLATAFORMA:

Para VM de proceso descrita anteriormente, la portabilidad multiplataforma es un objetivo muy importante (poder ejecutar programas que usan un ISA sobre una máquina con otro ISA diferente). El problema se presenta si la plataforma del host corre un sistema operativo diferente y algún programa binario compilado originalmente se quiera ejecutar en él. Por ejemplo si se quiere ejecutar binarios IA-32 en plataformas SPARC, PowerPC y MIPS, se tendría que desarrollar una máquina virtual para cada una de ellas.

La compatibilidad multiplataforma está diseñada para facilitar la portabilidad y coincidir con las características de un lenguaje de alto nivel usado para el desarrollo de las aplicaciones. Estas máquinas virtuales de alto nivel son similares a las máquinas virtuales de proceso descritas antes. Sin embargo minimizan las características específicas del hardware y del sistema operativo ya que éstas comprometen la independencia de plataforma.

En un sistema convencional, un compilador realiza un análisis léxico, sintáctico y semántico para generar código intermedio, similar al código de máquina pero más abstracto. Este código intermedio no contiene registros asignados. Por ejemplo, el generador de código toma el código binario y genera un binario que contiene el código de máquina para un ISA específico y sistema operativo. El archivo binario es ejecutado y distribuido en máquinas que soporten esa combinación de ISA y sistema operativo. Para ejecutar el programa en una plataforma diferente debe ser recompilado para esa plataforma.

En una máquina virtual este proceso cambia, el compilador genera código de máquina abstracto similar al código intermedio. El código es distribuido para diferentes plataformas. En este proceso la máquina virtual contiene un intérprete

que toma cada instrucción, la decodifica y luego realiza la transformación de estado requerida (incluyendo memoria y stack).

Básicamente, existen dos grandes formas de ejecutar programas: programas compilados (previamente pasados por un compilador) y programas interpretados (necesitan pasar por un intérprete para ejecutarse en tiempo real).

La ventaja de un HLL (Lenguaje de alto nivel) es que se puede portar fácilmente una vez que la máquina virtual se ha implementado en la plataforma destino.
Además es mucho más simple desarrollar este lenguaje que desarrollar un compilador para cada plataforma y recompilarlo para cada aplicación cuando esta sea portada.

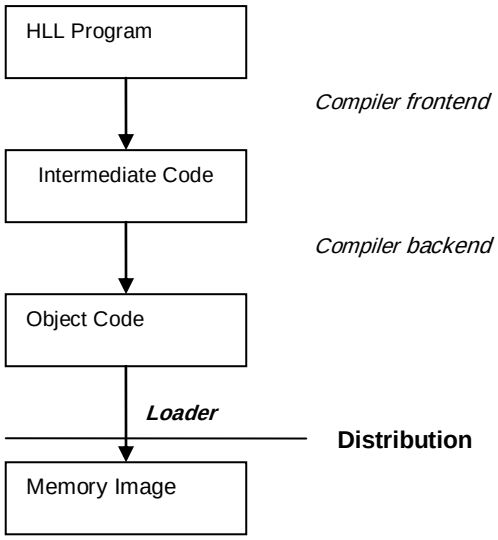


Figura 6: En un sistema convencional el código objeto es distribuido.[6]

6. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5.

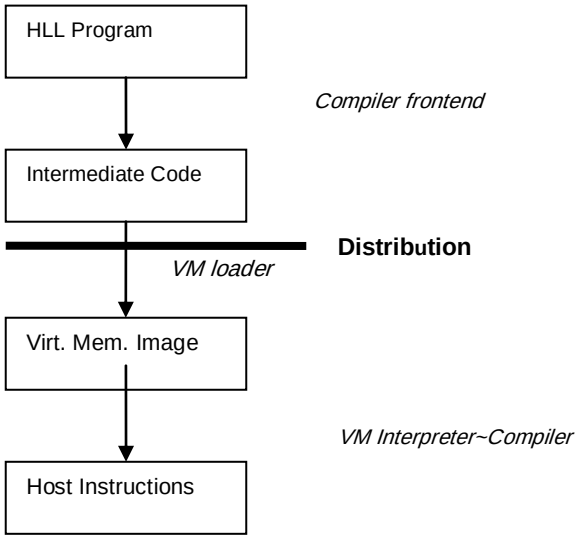


Figura 7: En un sistema virtual el código intermedio es ejecutado por la VM.[7]

III. MÁQUINA VIRTUAL DE SISTEMA

Una máquina virtual de sistema consiste en un sistema compilado en el cual algunos procesos pueden pertenecer a múltiples usuarios y coexistir.
Estos tipos de máquinas fueron desarrollados en los años 60 - 70 y fueron las originarias de las máquinas virtuales.
En un sistema de virtualización un simple host puede soportar múltiples sistemas operativos guest simultáneamente y distintos.

Los sistemas mainframes grandes y caros del pasado son en la actualidad servidores o granjas de servidores los cuales pueden ser compartidos por un número de usuarios o grupos de usuarios.

Quizás una de las características de los sistemas de virtualización actuales es que proporcionan una manera segura de dividir el/los software que corren simultáneamente en la misma plataforma de hardware.

7. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5.

El software que corre en el sistema guest es aislado de los softwares que corren en otros guest. Además si la seguridad de un guest se ve comprometida o si sufre alguna falla, los sistemas de los otros guest no se ven afectados.

La característica de replicación de plataforma es una característica importante de la virtualización.
El problema está en dividir los recursos de hardware entre ambientes de sistemas operativos invitados múltiples.

El VMM tiene acceso y administra los recursos de hardware. El sistema operativo invitado y las aplicaciones compiladas por este sistema operativo son administrados y están bajo el control del hypervisor.
Cuando un sistema operativo guest realiza ciertas operaciones como instrucciones con privilegios que directamente invocan a los recursos de hardware compartidos, esta operación es interceptada por el hypervisor, verificada y realizada por el hypervisor del guest.

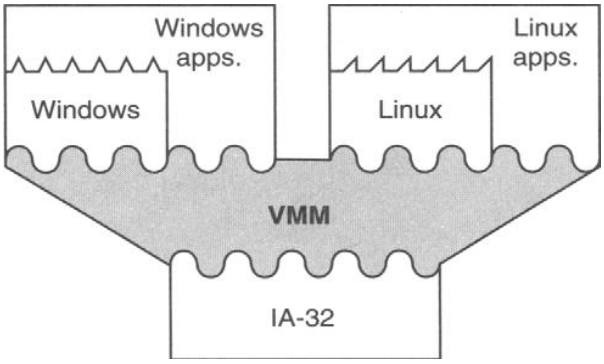


Figura 8: Una plataforma de hardware que soporta máquinas con distintos SO.[8]

8. James. E. Smith. Ravi Nair. Virtual Machines. Elsevier. Inc. 2005. ISBN 1-55860-910-5.

EMULACION.UNA MÁQUINA VIRTUAL DE SISTEMA COMPLETO:

En las máquinas virtuales definidas anteriormente, todos los softwares de sistemas (guest y host) y softwares de aplicación usan la misma ISA que el hardware principal. Para poder correr estos softwares en ISA diferentes el software de máquina virtual debe emular el hardware entero. Tiene que llevar a cabo la emulación de todas las instrucciones y debe convertir las operaciones del ISA del sistema invitado a llamadas al sistema operativo equivalentes hechas en el sistema operativo anfitrión.

IMPLEMENTACIONES DE MÁQUINA VIRTUAL DE SISTEMA:

Desde el punto de vista del usuario los sistemas de virtualización proporcionan más o menos la misma funcionalidad, la diferencia es la manera en la que se llevan a cabo.

Según la arquitectura de VMM (Monitor de máquina virtual) clásico, una alternativa es cuando el hypervisor se ejecuta sobre el hardware y las máquinas virtuales se ubican sobre este. El hypervisor corre en el modo más privilegiado y mientras el resto de los sistemas operativos guest corren en un modo con menor privilegio. Esta es la manera más transparente y el hypervisor puede interceptar e implementar las acciones específicas de los sistemas operativos invitados que interactúan con los recursos de hardware.

La arquitectura VMM (Monitor de máquina virtual) es muy eficiente y provee servicios a todos los sistemas invitados más o menos de manera equivalente.

Una desventaja de esta técnica es que requiere que el sistema este limpio y se instala primero el VMM y luego los sistemas operativos invitados encima.

Otra desventaja es que los controladores deben estar disponibles en el momento de la instalación del VMM, porque éste interactúa directamente con los dispositivos de entrada/salida.

La segunda alternativa de la implementación del VMM, es cuando éste se ejecuta sobre el sistema operativo anfitrión. La instalación del VMM es similar a la de cualquier programa de aplicación. La desventaja de esta alternativa es que se incluye una capa intermedia más, lo que hace menos eficiente el proceso de virtualización cuando un servicio de sistema operativo es solicitado.

En este esquema cada máquina virtual puede ejecutar cualquier sistema operativo soportado por el hardware subyacente. Así los usuarios pueden ejecutar dos o más sistemas operativos distintos simultáneamente en máquinas "privadas" virtuales.

Para que el hypervisor se ejecute en el modo más privilegiado o menos privilegios se utiliza un mecanismo de seguridad (que permite proteger datos y funcionalidades frente a fallas y comportamientos maliciosos) denominado "Anillos de protección o también conocido como Dominios de protección jerárquicos".
Este sistema se basa en cuatro niveles de prioridad que a menudo se representan como anillos concéntricos donde quien está más al centro tiene poder sobre los que están más a la periferia.

Nivel 0: kernel.
Nivel 1: servicios del sistema
Nivel 2: extensiones al sistema operativo
Nivel 3: aplicaciones del usuario

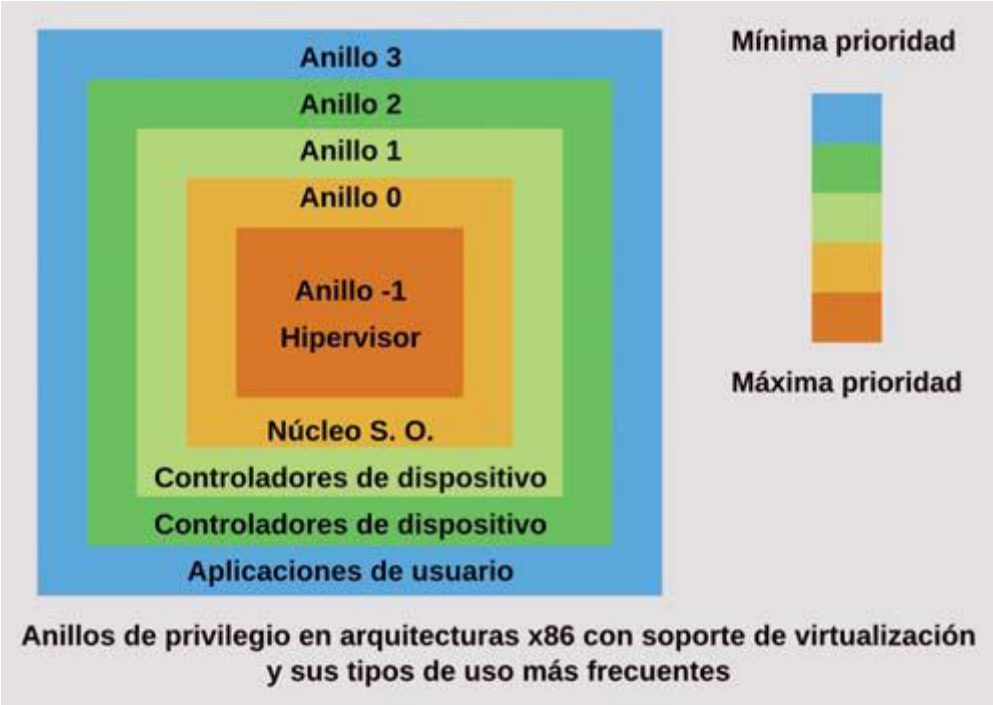


Figura 9: Anillos de protección.[9]

Existen cuatro niveles o anillos (0-3) de privilegio en modo protegido.

El núcleo del sistema operativo que necesita ejecutar las instrucciones más privilegiadas, corre en el anillo 0 en cambio las aplicaciones de usuario corren en el anillo 3.

El sistema operativo puede utilizar todas las instrucciones disponibles porque se ejecuta en modo supervisor, concretamente en anillo 0. Como el anillo 3 depende del anillo 0, cualquier inestabilidad en el anillo 0 repercute en el modo usuario. Para aislar el anillo 0 de cada máquina virtual es necesario mover el anillo 0 lo más cerca posible de la VM. De esta forma cualquier fallo en el anillo 0 desplazado de una VM no repercute en el anillo 0 nativo y por lo tanto tampoco repercute sobre otras máquinas virtuales El anillo 0 desplazado puede estar ubicado entre los anillo 1 y anillo 3 nativos.

9. http://www.intel.com

Cuanto más lejos esté ubicado el anillo 0 desplazado del anillo 0 nativo, menos rendimiento se obtiene.
La virtualización ubica el VMM en el anillo 0 nativo y desplaza hacia arriba el anillo 0 de las máquinas virtuales.

Existen dos tipos de hypervisor:
Tipo 1 - Hardware / Hypervisor VMM / Maquina Virtual
También conocidos como no hosted o sobre el hardware Se ejecutan en anillo 0 nativo directamente sobre el hardware subyacente. Los sistemas operativos virtualizados se ejecutan en los anillos superiores.
Tipo 2 - Hardware / Sistema Operativo / Hypervisor VMM / Maquina Virtual
También conocidos como hosted. Se ejecutan habitualmente en el anillo 3 sobre un sistema operativo. Los VMM de tipo-2 se ejecutan en el anillo más lejano del anillo 0 nativo. Tiene como principal inconveniente el bajo rendimiento debido a las múltiples capas que deben ser atravesadas

DESAFIOS ACTUALES PARA UN SOFTWARE DE VIRTUALIZACION.

Uso de memoria privada para uso exclusivo de VMM:

Para almacenar información del sistema, el VMM debe usar bloques privados de memoria que sólo ellos pueden acceder. El problema es cómo asignar esta memoria de tal manera que el sistema operativo invitado no tenga acceso a ella (ya sea sin querer o a propósito). La solución principal es que el VMM pueda interceptar los accesos a estas áreas de memoria y emular el resultado esperado del acceso inicial. Este engorroso proceso es requerido por el hardware que no soporta virtualización. En procesadores que si soportan virtualización, ciertas páginas de memoria usadas por VMM pueden ser accesibles sólo por el software de virtualización (VMM), ya que tiene el mayor nivel de privilegio. Este paso hace que las áreas inaccesibles y más importantes sean invisibles para todos los

demás programas.

Uso del manejo de interrupciones VMM:

Las interrupciones que requieren inmediata atención del sistema deben ser manejadas por el VMM. El problema es que los sistemas operativos tienen la capacidad de impedir la entrega de las interrupciones. Este mecanismo se utiliza para bloquear las interrupciones de algunas actividades que deben realizarse sin la interferencia de un evento externo. VMM puede gestionar el flujo de las interrupciones de los sistemas operativos invitados, pero para ello tienen que monitorizar este flujo para poder bloquear y permitir dichas interrupciones. Algunos sistemas operativos hacen uso intensivo de esta característica, lo que provoca penalidades significativas en el rendimiento de VMM.

La virtualización asistida por hardware, aborda la problemática de soluciones de software mencionadas anteriormente y le permite al VMM correr fuera de la plataforma de sistemas operativos y aplicaciones sin tener que recurrir a la traducción binaria o paravirtualización. Esta capacidad facilita en gran medida el despliegue de VMM y proporciona una mayor fiabilidad y manejabilidad de los sistemas operativos invitados y las aplicaciones.

CÓMO FUNCIONA LA TECNOLOGÍA DE VIRTUALIZACIÓN ASISTIDA POR HARDWARE:

El hipervisor debe hacer dos cosas:

- Emular el entorno de hardware completo hasta el punto que el sistema operativo anfitrión no pueda decir que no posee la plataforma de hardware completo.

- Debe controlar todas las circunstancias inusuales que pueden surgir ya sea en el sistema operativo o en la aplicación.

Ambas tareas se deben realizar con altos niveles de fiabilidad y bajo rendimiento
El hardware que no es compatible con virtualización basada en hardware hace que sea difícil para el VMM cumplir estos objetivos, ya que los procesadores tradicionales fueron diseñados principalmente para ejecutar una sola instancia de un sólo sistema operativo.

Actualmente existen procesadores y sistemas operativos modernos que aplican el concepto de niveles de privilegio, que define qué acciones pueden ser realizadas por procesos específicos y permite que el VMM funcione correctamente.
A diferencia de la virtualización por software, la virtualización asistida por hardware permite a la capa de virtualización ejecutarse en un "anillo -1", es decir, con mayor prioridad que el 0, de forma que ya no es necesario engañar al sistema operativo.

En un sistema que esté funcionando usando la arquitectura de máquina virtual que se define en esta tecnología, se distinguen dos tipos diferentes de software. Por un lado encontramos al VMM (Monitor de máquina virtual), que actúa como host y tiene un control absoluto sobre el microprocesador y el resto del hardware del sistema. Además el VMM es capaz de mantener el control de forma selectiva sobre determinados recursos del procesador, de la memoria física, de la gestión de interrupciones y de los accesos de entrada/salida. Por otro lado se distingue al software invitado, término bajo el que se engloban a todas las máquinas virtuales presentes en el sistema. El software invitado suele estar formado por un sistema operativo completo y las aplicaciones de modo usuario que funcionan sobre dicho sistema. Dicho conjunto funciona de forma totalmente independiente a otras máquinas virtuales que estén funcionando en el equipo, si bien el sistema operativo de las máquinas virtuales trabaja con un nivel de privilegio reducido ya

que el VMM retiene el control de los recursos del procesador y del resto del hardware.

Los dos principales fabricantes de procesadores (INTEL Y AMD) llegan a lo mismo usando diferentes tecnologías, con similitudes pero incompatibles entre sí. Intel, con su Intel Virtualization Technology (o Intel-VT), y AMD con su AMD Virtualization (o AMD-V o también AMD "Pacífica").

En el caso de Intel, el soporte para la virtualización se proporciona mediante un nuevo modo de funcionamiento del micro. Dentro de dicho modo existen dos tipos: root y no root. De forma general el monitor VMM funciona en modo root, mientras que el software invitado (las máquinas virtuales) trabaja en modo no root. El monitor VMM puede hacer que el procesador pase a modo no root mediante una transición denominada entrada VM ("VM entry") mientras que las transiciones del micro a modo root se efectúan mediante salidas VM ("VM exit"). La implementación del modo de funcionamiento VMX se realiza mediante una serie de nuevas instrucciones. Además la ejecución de determinadas instrucciones así como que se produzcan ciertos eventos y situaciones, hace que se produzcan salidas VM que hacen pasar al procesador a modo root. Todas estas restricciones se aplican incluso al software que corre en el anillo cero del micro cuando éste se encuentra en modo protegido, por lo que cualquier aplicación o sistema operativo funciona en un equipo que esté trabajando en modo VMX.

SECUENCIAS DE WM-ENTRY Y WM-EXIT PARA LA ENTRADA Y SALIDA DE MÁQUINAS VIRTUALES.

VM Entry

- Transición de MMV a huésped.
- Entra en modo non-root.

- Carga el estado del huésped.
- VMLAUNCH instrucción usada en entrada incial. VMRESUME instrucción usada en llamadas siguientes.

VM Exit

- VMEXIT instrucción usada para pasar a MMV
- Entra en modo root
- Salva el estado del huésped
- Carga el estado de MMV

Al hacer uso de esta tecnología de virtualización, los productos nuevos VMM tienen las siguientes características:

Robusto: VMM ya no necesita utilizar la paravirtualización o la traducción binaria.
Esto significa que será capaz de ejecutar ciertos sistemas operativos disponibles en el mercado y las aplicaciones sin ningún paso especial.

Mejorado: permite ejecutar VMM en sistemas operativos invitados de 64 bits, inicialmente era en procesadores x86.
Fiabilidad: Debido al soporte de hardware, VMMs ahora puede ser más pequeño, menos complejo y más eficiente. Esto mejora la fiabilidad y la disponibilidad y reduce el potencial de conflictos de software.

Seguro: El uso de las transiciones de hardware en el VMM refuerza el aislamiento de máquinas virtuales y además previene la corrupción de una máquina virtual de afectar a los otros en el mismo sistema.

V. BENEFICIOS DE MIGRAR A UN ENORNO VIRTUALIZADO

La virtualización es uno de los factores importantes en la optimización de sistemas en cualquier organización dado que puede ser simplemente un medio para reducir y simplificar la infraestructura de servidores o puede llegar a ser una herramienta para transformar la visión global del centro de datos.

Es importante nombrar la consolidación física como la base los otros dos pasos de optimización (consolidación lógica y racionalización de sistemas y aplicaciones).

La consolidación física permite reducir el crecimiento horizontal, la consolidación lógica permite compartir las cargas de trabajo entre varias máquinas virtuales situadas en diferentes máquinas físicas y la racionalización permite identificar aplicaciones innecesarias o redundantes que pueden ser eliminadas.

La consolidación, la fiabilidad y la seguridad son beneficios que ayudan a justificar el movimiento hacia una infraestructura virtual.

Otra ventaja importante de la virtualización es el aislamiento, dado que cada máquina virtual es aislada de las demás máquinas virtuales del mismo servidor. Cada una tiene un sistema operativo entero, con su configuración, datos, red.
El sistema operativo no puede establecer una diferencia entre una máquina virtual y una máquina física, ni tampoco lo pueden hacer las aplicaciones u otras máquinas de una red. La propia máquina virtual se cree que es una máquina "real", pero esta sólo se compone de software y no contiene ninguna clase de componente de hardware.

Este nivel de aislamiento también facilita el mismo nivel de flexibilidad que proporciona un entorno físico no virtualizado. Imaginemos un servidor que funcione como servidor Web durante la semana laboral de 8:00 hs a 17:00 hs, un servidor que ejecute procesos batch todas las noches de la semana laboral y otro servidor que realice copias de seguridad durante el fin de semana. En un entorno no virtualizado, se requieren tres servidores dedicados, cada uno de ellos "altamente utilizado" durante sus respectivos horarios pero que permanecen ociosos el resto del tiempo.

Este escenario desperdicia tiempo de computación y dinero en infraestructura no utilizada. La virtualización permite solucionar este problema creando tres máquinas virtuales sobre un mismo servidor.

Durante la semana laboral se enciende la máquina virtual que funciona como servidor Web a las 8:00 hs y se apaga a las 17:00 hs. Cada día se encienda y apaga por la noche la máquina virtual que ejecuta los procesos bacth. Finalmente los fines de semana se encienda y apaga la máquina virtual que realiza las copias de seguridad.

Esta flexibilidad permite ejecutar las misma funciones que realizaría un entorno no virtualizado pero ahorrando dinero y manteniendo los mismo niveles de fiabilidad.

Los servidores virtuales en estado pasivo tienen las mismas características físicas que los servidores activos que monitorizan y por lo tanto el mismo precio y lo más importante es que garantizan la continuidad de producción frente interrupciones planificadas o no planificadas manteniendo un alto grado de producción.

La migración es otra ventaja importante dado que en algunos sistemas de máquinas virtuales se puede mover una máquina virtual de un servidor a otro, en frío o caliente sin cambiar configuración.
Existen sistemas de máquinas virtuales que realizan el proceso de migración de entornos virtuales en caliente entre equipos físicos sin pararlos, la memoria de la máquina virtual es copiada al destino sin detener su ejecución. Generalmente una parada muy breve de alrededor de 60 a 300 ms es necesaria para realizar la sincronización final antes de que la máquina virtual comience a ejecutarse en su destino final.
Un servidor físico es esencialmente una pieza de hardware, una máquina virtual se debe considerar más bien una colección de software que se ha convertido en archivos. Esos archivos están encapsulados, es decir, están recopilados y organizados en contenedores.

Al igual que los archivos, las máquinas virtuales se pueden copiar, mover, distribuir o enviar por correo electrónico y los archivos de máquina virtual se pueden distribuir en cualquier medio que sea lo bastante grande para almacenarlos. Esto incluye todo, desde un tarjeta de memoria, DVD o unidad de disco duro, hasta una SAN (Red de área de almacenamiento).

Por el contrario mover o copiar máquinas físicas resulta mucho más difícil. Para los usuarios que las utilizan por primera vez, las aplicaciones de una máquina física normalmente se instalan en lugar de copiarse simplemente. Gracias a la capacidad de encapsularse como un archivo, las máquinas virtuales son mucho más portables que las máquinas físicas.

La portabilidad de las máquinas virtuales como archivos aumenta enormemente su facilidad de gestión y representa una gran ventaja para los clientes. Pueden moverse en segundos sistemas completos, aplicaciones, sistemas operativos, BIOS y hardware virtual totalmente configurados, desde un servidor físico a otro,

sin paradas por mantenimiento y con una consolidación continua de la carga de trabajo.

La segunda diferencia clave entre las máquinas virtuales y las físicas es que las primeras son completamente independientes del hardware físico. Una máquina virtual puede tener una tarjeta de red, una tarjeta VGA o un controlador SCSI, pero estos componentes no interactúan con el hardware subyacente de la máquina física en la que residen.

Una máquina virtual puede ejecutarse en un servidor físico que tiene una tarjeta de red de la marca "nic", siempre verá una tarjeta de red virtual, la máquina virtual no ve la tarjeta de red de la marca "nic".

Esto significa que una máquina virtual puede moverse de un servidor físico a otro sin realizar cambios en los controladores de dispositivos, el sistema operativo o las aplicaciones, aunque los dos servidores físicos sean de dos fabricantes completamente diferentes. Varias máquinas virtuales instaladas en el mismo servidor físico pueden incluso ejecutar sistemas operativos diferentes.

Una aplicación que se ejecuta en una portátil puede moverse a un PC de escritorio, o incluso a un servidor y ejecutarse exactamente igual. La independencia del hardware proporciona más portabilidad y flexibilidad para la gestión y provisionamiento de servidores.
Podemos decir también que los entornos virtuales se pueden utilizar para montar entornos de prueba y desarrollo. Se crea la máquina virtual el tiempo necesario y luego se elimina.

La fiabilidad siempre ha sido muy importante en las organizaciones. Está muy ligada a la no interrupción de los servicios que proporciona un sistema o una aplicación. Con la virtualización la fiabilidad se obtiene gracias al aislamiento de

los errores, software en cada máquina virtual, la posibilidad de reubicar máquinas virtuales en otras máquinas físicas cuando se producen errores de hardware y la posibilidad de crear máquinas virtuales redundantes (o en failover) para garantizar la continuidad de los servicios.

Finalmente con la virtualización la seguridad se obtiene al poder contener los ataques digitales en una máquina virtual, sin que afecten a otras máquinas virtuales y a la posibilidad de aplicar diferentes niveles de seguridad a cada máquina virtual.

Actualmente la reducción de costos, este es el punto donde la virtualización aporta más a la infraestructura de TI por la reducción de utilización de espacio físico, reducción en la cantidad de cables, reducción en el consumo de energía eléctrica, reducción en la generación de calor, incremento en la disponibilidad de la infraestructura, reducción de costos en licenciamiento entre otros.

La disminución de costo es una de las ventajas más atractivas para los organismos y organizaciones. Una máquina virtual es compatible con sistemas operativos estándar como Windows y Linux y con los controladores de hardware y aplicaciones creados para esos sistemas operativos. Una máquina virtual tiene placa base, tarjeta VGA, controlador de tarjeta de red, todos los componentes que se encuentran en un servidor físico.

Habitualmente las aplicaciones desarrolladas para cualquier sistema operativo como Windows, Linux, Netware o Solaris pueden también ejecutarse en una máquina virtual sin ninguna modificación o requisito especial que el requerido por la propia aplicación.

VI. ASPECTOS IMPORTANTES A TENER EN CUENTA

La virtualización tiene sus riesgos, ya que somete los recursos físicos a un mayor nivel de estrés. Aunque la infrautilización de la CPU puede ser un factor que anime a la virtualización de los servidores, otros recursos de hardware pueden sufrir una sobre explotación. Dado que un sistema host tiene una capacidad limitada (que depende de la aplicación) para paginar la memoria utilizada por los sistemas huéspedes, el cuello de botella más reconocible es la memoria física (la RAM). Las opciones que permiten mitigar los embotellamientos de la memoria de manera programática tienen como contrapartida efectos negativos en el rendimiento.

El objetivo perseguido mediante la migración a un entorno virtualizado, de correr más cargas de trabajo virtuales sobre un menor número de sistemas físico implica un cierto costo dado que no significa que el hardware tenga que bajar posiciones en la lista de prioridades de los administradores de tecnología ni en sus presupuestos. Aunque se rompa el vínculo entre aplicaciones o sistemas operativos individuales y sistemas físicos individuales, resulta imprescindible contar con el hardware suficiente para soportar la carga en su conjunto.

El problema de consolidar todas o varias aplicaciones de una organización en un sólo equipo aumenta el impacto en caso de indisponibilidad del hardware lo que representa además un alto costo de recuperación.

Aunque el despliegue de la virtualización es cada vez más amplio en las organizaciones, muchas aplicaciones todavía no están preparadas para funcionar en entornos virtuales y otras resultan más lentas.
A medida que la consolidación del servidor continúa creciendo, aumentar las cantidades de máquinas virtuales invitadas en un sólo entorno virtualizado implica

planificar respaldos, restauraciones y recuperaciones ante desastres del entorno virtual .Lo que lleva a contar con un fuerte esquema de respaldo (Físico y lógico).

Es importante contar con planes de contingencia y sistemas redundantes, doble red, doble disco (implementar RAID, ofrecen cierta tolerancia a fallos mediante la recuperación de información, doble fuente de corriente). Como así también realizar copias de seguridad de las imágenes de los sistemas operativos virtualizados.

El funcionamiento eficiente de un centro de datos virtualizado o no depende de la administración o gestión del administrador de tecnología, es decir debe contar con planes anteriormente mencionados. En el caso de no ser así se puede decir que el impacto que ocasiona un problema físico o lógico es similar en los dos esquemas.

VII. VIRTUALIZACIÓN DE SERVIDORES

Nos interesa estudiar en este trabajo la virtualización de servidores que es una forma de crear una máquina virtual para que dentro de ella corra uno o varios sistemas operativos

Esta máquina virtual puede o no recibir ayuda por parte del hardware en que la corremos. A continuación se describen las distintas formas de implementar la virtualización de servidores.

I. EMULACIÓN DE HARDWARE

La emulación de hardware se basa en crear máquinas virtuales que emulan el hardware de una o varias plataformas distintas de hardware.

Primero se lleva a cabo la instalación de software de virtualización (hypervisor) y luego la instalación de cualquier otro SO: por ejemplo se puede instalar un Windows Xp en un entorno virtual Qemu en Debían.

Esta aplicación que simula el hardware completo permitiendo la ejecución de los sistemas operativos sin modificar, lo hace bajo el control del emulador que simula el sistema completo. Incluyendo la ejecución de las instrucciones a nivel de CPU. El emulador simula la ejecución de código binario para una CPU concreta, en un sistema real que usa un procesador y un juego de instrucciones diferente al del sistema emulado.

Por ejemplo es posible ejecutar un sistema operativo sin modificar diseñado para un PowerPC sobre una máquina anfitrión con procesador ARM.

Otro caso sería tener dentro de un servidor linux varios equipos x86 o PowerPC corriendo diferentes versiones de Linux.

Incluso es posible ejecutar múltiples máquinas virtuales, cada una simulando un procesador diferente

Las aplicaciones en este enfoque se ejecutan en un sistema operativo guest verdaderamente aislado, uno por cada VMM, permitiendo que distintos sistemas operativos se ejecuten al mismo tiempo.

El inconveniente de este modelo de virtualización es que la simulación es muy lenta (para cada instrucción del sistema emulado puede ser necesario ejecutar entre 100 y 1000 instrucciones a la CPU real), la más costosa y menos eficiente.

Una de las ventajas es que se pueden ejecutar sistemas operativos completamente diferentes del anfitrión. Por ejemplo es posible ejecutar Windows sobre Linux. Además al permitir diferentes sistemas operativos ejecutándose simultáneamente, se usa para ambientes de desarrollo y prueba. Se usa también este tipo de virtualización cuando se quiere mover un ambiente con sistema operativo y aplicaciones de un servidor físico a un servidor con virtualización por software.

Podemos citar emuladores como Qemu, Bochs, Name entre otros.

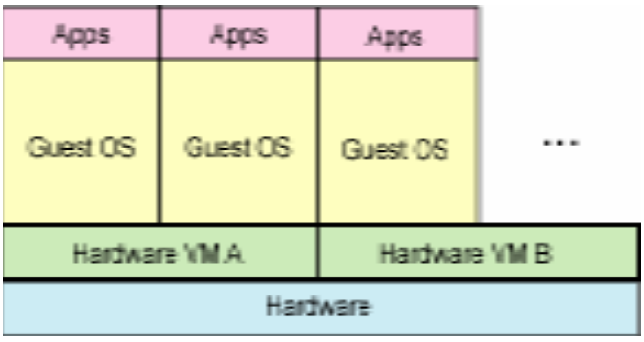


Figura 10: Técnica de Emulación[10].

10. http://www.microsoft.com/virtualization

II. PARAVIRTUALIZACION

Un emulador funciona como un programa normal en un sistema operativo como entre otros (Linux o Windows) pero teóricamente se puede eliminar el sistema operativo anfitrión. Para que esto pueda ocurrir se necesita un software especial que funcione directamente sobre el hardware. Los programas de este tipo son conocidos como hypervisores o monitores de máquinas virtuales.

Este hypervisor gestiona los sistemas operativos huéspedes independientemente del sistema operativo, lo que asegura que haya operaciones paralelas sin problemas.

En este tipo de medios, los sistemas huéspedes no trabajan directamente con el hardware, sino que envían sus peticiones al hypervisor. Por ejemplo, si uno de los sistemas Linux huésped funciona en paralelo necesita acceso al disco, envía una petición de acceso al hypervisor. Éste maneja entonces el acceso físico y devuelve los resultados al sistema.

Para permitir que todos los invitados hablen con el hypervisor, éste les ofrece una interfaz estandarizada para el hardware físico, que la pueden usar otros programas y sistemas. Esta técnica conocida como paravirtualización tiene la ventaja de una ejecución sorprendentemente rápida comparada con otras soluciones.

El producto gratuito Xen, Virtual Iron, User Mode Linux, KVM (Kernel Virtual Machine) de linux y VMware Workstation 6, ESX Server de VMware como productos comercializados que soportan paravirtualización son ejemplos más populares de esta tecnología.

Los requisitos del sistema huésped para soportar el hypervisor son un obstáculo para la paravirtualización, ya que implican modificar el sistema operativo (para incluir instrucciones relacionadas con la virtualización). Con un sistema tan blindado como Microsoft Windows, esta tarea es obviamente difícil. Otra complicación es que el hypervisor tiene que manejar personalmente numerosas

tareas del sistema operativo. Por ejemplo, tiene que saber qué tipo de adaptador de gráficos tiene el sistema y como dirigirse a él.

Para evitar las tediosas modificaciones de drivers se utiliza algunos de los métodos siguientes:

I. El hypervisor Xen simplemente elige uno de los sistemas operativos paralelos como su huésped favorito. Una vez que se ha establecido esta relación emplea los drivers de ese sistema.
   En otras palabras si algún otro sistema operativo funcionando en la máquina accede a la interfaz USB, el hypervisor le pasa la petición a su favorito.

II. El segundo método transforma un kernel existente de Linux para producir un hypervisor). KVM por ejemplo, emplea la técnica de la virtualización; provee de un modulo kernel que convierte al del Linux en uso en un hypervisor, que a su vez emplea un emulador modificado a partir de QEMU para lanzar otros sistemas operativos. A este método de virtualización se le llama basado en kernel.

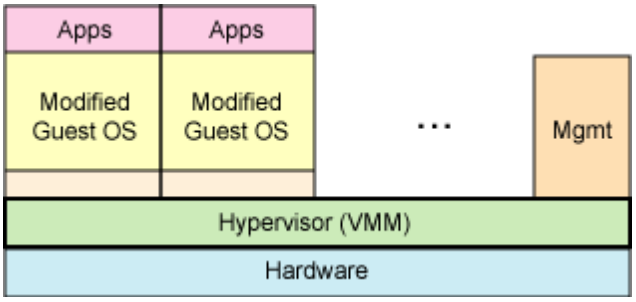


Figura 11: Técnica de Paravirtualización.[11]

11. http://www.microsoft.com/virtualization

III. VIRTUALIZACION A NIVEL DE SISTEMA OPERATIVO

La Virtualización a nivel de sistema operativo consiste en modificar el núcleo del sistema operativo o Kernel para incluir los mecanismos de virtualización, es eficiencia nativa.

El servidor físico y una única instancia del sistema operativo son virtualizadas en múltiples particiones aisladas, donde cada partición duplica un servidor real. El kernel se ejecutará en un único sistema operativo y proveerá esa funcionalidad del sistema operativo para cada una de las particiones.

Los guests comparten el mismo sistema operativo que el anfitrión y todos utilizan el mismo kernel y es él kernel el que se ocupa de determinar para quién trabaja en un momento determinado.

No permite ejecutar dos sistemas operativos simultáneamente, sino SVP (Servidores privados virtuales) dentro de un único servidor, es decir, es un único kernel pero que permite aislar los servidores. Cada servidor tendrá su propia red, espacio de disco, de memoria, se podrá reiniciar así mismo tendrá limitación de uso de CPU con el fin de evitar que un servidor virtual consuma recursos de los otros. También esta tecnología se denomina como Jail, pues es extender el concepto de chroot.

Algunos ejemplos: Virtuozzo, OpenVZ, Linux VServer, Solaris Containers y FreeBSD Jails.

Las ventajas de este sistema son la baja carga y el uso eficiente de los recursos físicos de la máquina anfitriona. Para poder compartir los recursos del mismo

núcleo, los sistemas operativos de las máquinas virtuales deben ser compatibles con los de la máquina anfitriona. La desventaja es que requiere cambios el kernel de dicho sistema.

Otra desventaja es que limita la elección del sistema operativo, lo que significa que ofrece el mismo sistema operativo que el host. En su versión para Linux permite crear servidores virtuales con distintas distribuciones de Linux sobre un anfitrión Linux. En su versión para Windows permite crear servidores virtuales Windows sobre un anfitrión Windows.

Otro inconveniente es que sólo permite ejecutar entornos virtuales para la misma CPU y sistema operativo y en realidad sólo hay un núcleo, de manera que si ese núcleo tiene un problema, todas las máquinas virtuales se ven afectadas.

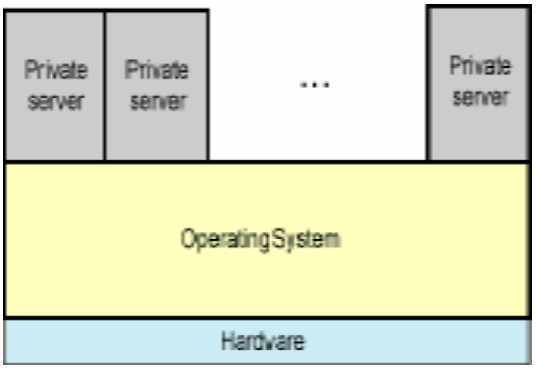


Figura 12: Técnica de Virtualización a Nivel de Sistema Operativo.[12]

IV. VIRTUALIZACION COMPLETA O NATIVA

La virtualización completa usa una máquina virtual que hace entre intermedia entre el sistema invitado y el hardware real.

El software de virtualización es conocido como VMM (Monitor de máquina virtual) o hypervisor.

En este tipo de sistemas, el hypervisor se encarga de emular un sistema operativo y analiza dinámicamente el código que quiere ejecutar el sistema invitado,

12. http://www.microsoft.com/virtualization

reemplazando las instrucciones críticas (las que hace falta virtualizar) por nuevas secuencias de instrucciones que tiene el efecto deseado en el hardware virtual mientras que las instrucciones no críticas se ejecutan tal cual en la CPU real

Se puede decir que el host emula lo suficientemente bien el hardware como para que los guests puedan ser ejecutados de forma nativa, es decir sin cambios en el kernel y además de forma completamente aislada.
Se pueden ejecutar varios guests en la misma máquina y compartir eficientemente sus recursos.

Algunos ejemplos de virtualización nativa KVM, VMware Workstation, VMware Server, Parallels Desktop, Adeos, Mac-on-Linux, Win4BSD, Win4Lin Pro y z/VM.

.

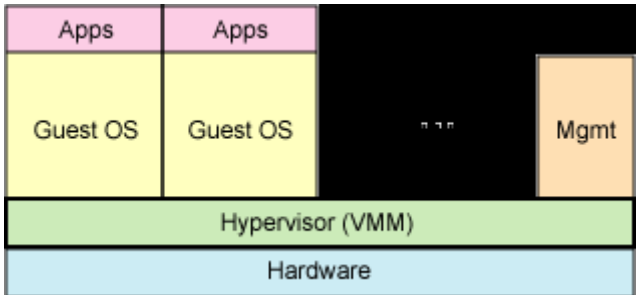


Figura 13: Técnica de Virtualización Completa.[13]

13. http://www.microsoft.com/virtualization

VIII. HERRAMIENTAS DE VIRTUALIZACIÓN GNU

En este punto analizaremos algunas de las características más importantes de algunos sistemas de virtualización disponibles para GNU/Linux. Dado que sólo nos interesa en este trabajo, analizar las herramientas de virtualización GNU, para llevar a cabo la implementación del sistema de gestión de máquinas virtuales.

I. BOCHS

Es un emulador código abierto de arquitecturas basadas en x86 y AMD64 con licencia de software abierto, que funciona en múltiples plataformas capaz de emular una máquina completa incluyendo periféricos y funciona en prácticamente cualquier sistema anfitrión (se puede usar para emular una computadora en un Linux, que se ejecuta en una arquitectura PowerPc, Sparc, Mips, etc.) y además permite simular varios sistemas operativos como Linux, Windows, DOS.

El problema de este sistema es que es muy lento (se puede ejecutar un programa instrucción por instrucción viendo el contenido de los registros y de la memoria en todo momento, lo que constituye una valiosa información sobre el estado de la máquina para depuración.) a pesar de que las últimas versiones van mejorando la velocidad de emulación empleando técnicas de optimización.

Es muchísimo más lento que VMWare o QEMU.

II. QEMU

QEMU es un emulador de software libre, similar a Bochs que soporta dos modos de ejecución.

Emulación del sistema completo: emula el sistema de computadora completo, incluyendo el procesador y periféricos. Este modo permite emular distintas

arquitecturas (x86, x86_64, ARM, SPARC, PowerPC y MIPS) utilizando traducción dinámica de instrucciones. Se pueden emular Windows o Linux en Linux, Solaris o FreeBSD.

Emulación del modo usuario: Puede ejecutar programas compilados para un tipo de CPU en otro CPU. Este modo permite que un programa compilado para MIPS pueda ser ejecutado en GNU/Linux x86. Este modo sólo funciona en entornos GNU/Linux.

Para arquitecturas x86 QEMU soporta el uso de un módulo de aceleración para sistemas anfitriones Linux y Windows que permite que parte del código que se ejecuta en sistemas invitados sea ejecutado directamente por la CPU real haciendo que Qemu funciones como sistema de virtualización nativa en lugar de cómo un emulador.

Fue desestimado su uso para la implementación del sistema de gestión de entornos virtuales por lo poco desarrollado que se encuentra en la actualidad.

III. KVM (Kernel Virtual Machine)

Es una solución de virtualización completa en la que se utiliza el núcleo de Linux como hypervisor, de manera que tanto el control de los dispositivos reales como la planificación de tareas y la gestión de memoria del sistema anfitrión los hace el núcleo de Linux. La instalación es muy sencilla y tiene un muy buen rendimiento en operaciones de CPU.

En este modelo las máquinas virtuales son procesos normales del sistema por esto la gestión de memoria y la planificación de procesos son los estándar de Linux (usuario y núcleo).

Una máquina virtual tendrá 3 modos de ejecución:

Modo invitado: será el modo de ejecución normal para el código del sistema invitado siempre que no tenga operaciones de entrada-salida.

Modo usuario: se usa para ejecutar operaciones de entrada-salida del sistema invitado, gestiona dispositivos virtuales a nivel de usuario.

Modo núcleo: se usa para trabajar en modo invitado y para gestionar las salidas desde modo usuario causadas por operaciones especiales de entrada-salida

IV. XEN

Xen es un paravirtualizador de código abierto desarrollada por la Universidad de Cambridge, que permite ejecutar instancias de sistemas operativos con todas sus características, proporcionando aislamiento seguro, control de recursos, garantías de calidad de servicio y migración de máquinas virtuales en vivo.

El hypervisor se ejecuta en el nivel más privilegiado de la máquina y básicamente se hace cargo de la planificación de tareas y de la gestión de memoria, delegando la gestión de la entrada-salida en un invitado privilegiado (llamado domain 0 en Xen) que arranca con el hypervisor.
En este modelo el código del hypervisor es más sencillo y ligero a pesar de que actualmente y dado la complejidad de las CPU (multithreading, multicore) y de la gestión de memoria, cada vez el tema de la simplicidad es evidente.
Cuando XEN se emplea en un CPU que no soporta virtualización a nivel de hardware es necesario modificar el código del sistema operativo que se va a ejecutar sobre él.
Si la CPU soporta virtualización el hypervisor de Xen se ejecuta en el anillo de máxima prioridad y en este caso se ejecutan sistemas operativos

Es importante tener en cuenta que mientras mayor sea el aislamiento entre máquinas virtuales menor será el rendimiento.

Xen en un entorno paravirtualizado obtiene un perfecto equilibrio entre rendimiento y aislamiento.

V. LINUX VSERVER

Es un sistema de virtualización a nivel de sistema operativo que se implementa como una serie de parches sobre el núcleo de Linux.

Lo que hace este sistema es incluir el apoyo en el núcleo para crear y mantener múltiples entornos de usuario independientes VPS (Sistemas privados virtuales) sin que tengan ninguna interferencia entre ellos.

Para independizar los espacios de usuario se define el concepto de contexto, que no es más que un contenedor de procesos relacionados con un único VPS. Cuando el sistema arranca, define un contexto por defecto que es el que emplean todos los procesos que pertenecen al sistema anfitrión.

A parte de los contextos también emplea una llamada chroot para redefinir el directorio raíz de los procesos que se ejecutan dentro de un contexto determinado y evita que puedan acceder a los directorios que hay por debajo de la raíz.

El sistema está disponible para múltiples familias de microprocesadores (x86, X86-64, PowerPC, ARM).

El problema de este sistema es que no gestiona adecuadamente la utilización compartida de recursos virtuales como las tarjetas de red (o mejor dicho dispositivos virtuales de red) puesto que lo que hace es usar los recursos del anfitrión sin aislarlos de los que usa la máquina virtual (si se lanza una operación bind() contra un puerto dentro de una máquina virtual y el puerto está ocupado por un proceso que se ejecuta en el anfitrión y que no especificó una dirección IP el bind() falla, cosa que no pasaría si el aislamiento entre dispositivos virtuales de red fuera total.

VI. OPENVZ

Es un sistema similar al LinuxVServer que incluye capacidades y herramientas de administración más adelantadas que las de éste último.

En concreto OpenVz añade virtualización (múltiples entornos virtuales aislados dentro del mismo núcleo), gestión de recursos (proporciona mecanismos para limitar y en ocasiones garantizar disponibilidad de recursos como la CPU, la memoria o el espacio de disco para cada entorno virtual) y capacidad de checkpointing (posibilidad de congelar un entorno virtual, almacenar su estado completo en un fichero que se usará en caso de descongelar el entorno virtual, en la misma máquina o en otra real, dejándolo en el mismo estado que tenía antes de la congelación).

Funciona en múltiples plataformas x86, X86-64, PowerPC.

IX ¿POR QUÉ SE ELIGIÓ OPENVZ?

Se prefirió OPENVZ como herramienta de virtualización GNU, para poder desarrollar el sistema de gestión de servidores virtuales basado en un modelo cliente servidor utilizando sistema operativo Linux, porque en el momento que se presentó el tema de tesis, no se encontró sistema alguno que utilizara esta herramienta de virtualización open source a nivel de sistema operativo para gestionar entornos virtuales desde una interfaz Web.

Además si se lo compara a máquinas virtuales tales como VMware, VirtualBox y las tecnologías de virtualización tales como Xen, OpenVZ ofrece menor flexibilidad en la elección del sistema operativo: tanto los huéspedes como los anfitriones deben ser Linux (aunque las distribuciones de Linux pueden ser diferentes en diferentes entornos virtuales). Sin embargo, la virtualización en el nivel de sistema operativo de OpenVZ proporciona mejor rendimiento, escalabilidad, densidad, administración de recursos dinámicos, y facilidad de administración que las alternativas.

Otra ventaja importante es que se distribuye con un conjunto de utilidades que simplifican mucho la creación y mantenimiento de los entornos virtuales, incluyendo la posibilidad de trabajar en plantillas de entornos virtuales pre instaladas que contienen una imagen del sistema de archivos raíz de una máquina virtual.

X. TÉCNICA DE VIRTUALIZACION DE OPENVZ

Analizadas cada una de las técnicas de virtualización con sus respectivas características y ejemplos de cada una de ella, se detallará en particular el funcionamiento de OPENVZ, dado que se utilizará la misma para la implementación del sistema de gestión de servidores virtuales.

OpenVZ es una tecnología de virtualización en el nivel de sistema operativo basada en el núcleo y sistema operativo Linux. OpenVZ es la base de Parallels Virtuozzo una solución de virtualización comercial ofrecido por Parallels.

OpenVZ permite que un servidor físico ejecute múltiples instancias de sistemas operativos aislados, conocidos como SPV (Servidores Privados Virtuales) o EV (Entornos Virtuales).

I. APLICACIONES DE OPENVZ

OpenVZ brinda solución a los Proveedores de Servicio de Hosting permitiéndoles:

- Tener centenares de clientes cada uno con sus servidores privados virtuales dedicados pero compartiendo un sólo servidor físico.
- Calidad de Servicio garantizada a cada cliente.
- Migrar los clientes y sus entornos entre servidores de manera transparente sin ninguna reconfiguración manual.

Si el administrador tiene varios servidores Linux dedicados dentro de una organización y cada uno de los cuales ejecuta un servicio específico al usar OpenVZ se puede consolidar todos estos servidores en una sola computadora sin perder ni un poco de información valiosa y sin afectar el rendimiento.

Los Servidores Privados virtuales simplemente se comportan como un servidor autosuficiente aislado:

I. Cada VPS tiene sus propios procesos, usuarios, archivos y proporciona acceso con privilegios de root en el shell.
II. Cada VPS tiene su propio IP, números del puerto, reglas de ruteos y filtrados de paquetes.
III. Cada VPS tiene su propia configuración de sistema y software de la aplicación, así como sus propias versiones de bibliotecas de sistema. Es posible instalar o personalizar los paquetes del software independientemente dentro de un VPS de otro VPSs o del sistema Host. Pueden ejecutarse múltiples distribuciones de un paquete y en el mismo Linux

De hecho, ciento de servidores pueden agruparse de esta manera. Además de las ventajas evidentes de tal consolidación aumentó facilidad de administración.
Otro campo importante donde se puede implementar OpenVZ es en las instituciones educativas, proporcionándole a cada estudiante un servidor personal en Linux que pueda ser monitoreado y administrado remotamente.
Las compañías de desarrollo de software pueden usar ambientes virtuales para pruebas y desarrollo.
OpenVZ puede aplicarse eficazmente en una amplia gama de áreas: Web hosting, desarrollo y testing de software, capacitación o entrenamiento a usuarios.

II. CARACTERÍSTICAS DE OPENVZ

El núcleo de OpenVZ es un núcleo Linux modificado, que agrega una noción de entorno virtual. El mismo proporciona virtualización, aislamiento, administración de recursos y punto de comprobación.

Administración de recursos:

Como todos los entornos virtuales usan el mismo kernel, la administración de recursos es de suprema importancia. Realmente cada entorno virtual debería permanecer dentro de sus límites y no afectar a otros entornos virtuales de ninguna manera y esto es lo que la administración de recursos hace.

La administración de recursos de OpenVZ consiste de tres componentes: cuota de disco de dos niveles, planificador de CPU razonable, y monitor de usuarios. Debe notarse que todos esos recursos se pueden cambiar durante el tiempo de ejecución de un entorno virtual, no hay necesidad de reiniciar el sistema. Es decir, si se desea dar a un entorno virtual menos memoria, sencillamente hay que cambiar los parámetros apropiados al vuelo. Esto es o muy difícil de hacer o imposible con otras maneras de virtualización tales como VM o hypervisor

Punto de comprobación y migración en vivo:

La característica de migración en vivo y punto de comprobación se liberó para OpenVZ a mediados de abril de 2006. Esta permite migrar un entorno virtual desde un servidor físico a otro sin necesidad de apagar/reiniciar un entorno virtual. El proceso se conoce como punto de comprobación: un entorno virtual se congela y todo su estado se guarda en un archivo en disco. Este archivo puede ser transferido a otra máquina y el entorno virtual puede descongelarse (restaurarse) allí.

La demora es alrededor de unos pocos segundos, dado que cada pieza de estado de entorno virtual, incluyendo conexiones de red abiertas se guarda, desde la perspectiva del usuario parece una demora en la respuesta: una transacción de

base de datos toma más tiempo que el usual, cuando continúa como normal el usuario no nota que su base de datos está ya corriendo en otra máquina.

Esta característica hace posible escenarios tales como actualizar un servidor sin necesidad de reiniciarlo: si la base de datos necesita más memoria o recursos de CPU, sencillamente se debe comprar un servidor mejor y más nuevo y migrar en vivo el entorno virtual a éste, migrar todos los entornos virtuales a otro, apagarlo, agregar memoria, arrancarlo de nuevo y migrar todos los entornos virtuales de nuevo.

Monitor de usuarios:

El monitor de usuarios es un grupo de contadores por entorno virtual, límites, y garantías. Hay un conjunto de alrededor de 20 parámetros que se eligen cuidadosamente para cubrir todos los aspectos de la operación de entorno virtual, de manera que ningún entorno virtual por sí sólo pueda abusar de cualquier recurso el cual es limitado por todo el nodo y así hacer daño a otros entornos virtuales.

Los recursos contabilizados y controlados son principalmente memoria y objetos en el kernel tales como los segmentos de memoria compartidos IPC, buffers de red. Cada recurso puede verse desde /proc/user_beancounters y tiene cinco valores asociados con éste: uso actual, uso máximo (por el tiempo de vida de un entorno virtual) contador de barrera, de límite y de falla. El significado de barrera y límite es dependiente del parámetro; en breve, aquellos que se pueden pensar como un límite soft y un límite hard. Si cualquier recurso alcanza el contador de límite o de falla, entonces el dueño del entorno virtual puede ver si algo malo está pasando analizando la salida de /proc/user_beancounters en su entorno virtual.

Cuota de disco de dos niveles:

El dueño (root) del sistema anfitrión (OpenVZ) puede configurar cuota de disco por entorno virtual, en términos de bloques de disco e inodos (aproximadamente igual al número de archivos). Éste es el primer nivel de cuota de disco. Además de esto, un dueño de entorno virtual (root) puede usar las herramientas usuales de cuotas dentro de su propio entorno virtual para definir cuotas de disco estándar de UNIX por usuario y por grupo.

Si se desea dar a un entorno virtual más espacio, sencillamente hay que incrementar la cuota de disco. No se necesita redimensionar las particiones de disco.

Virtualización y aislamiento:
Cada entorno virtual es una entidad separada y desde el punto de vista de su dueño se muestra como un servidor físico real. De manera que tiene sus propios:

Archivos:
Bibliotecas del sistema, aplicaciones, /proc y /sys virtualizado, locks virtualizados.

Usuarios y grupos:
Cada entorno virtual tiene sus propios usuarios root, así como también otros usuarios y grupos.

Árbol de procesos:
Un entorno virtual solamente ve sus propios procesos (comenzando a partir de init). Los PIDs son virtualizados, de manera que el PID de init es 1 como debe ser.

Red:

Dispositivo de red virtual, que permite a un entorno virtual tener sus propias direcciones IP, así como también un conjunto de reglas de netfilter (iptables) y reglas de encaminamiento.

Dispositivos:

Si es necesario solamente al entorno virtual puede otorgarse acceso a los dispositivos reales como interfaces de red, puertos seriales, particiones de disco, etc.

Objetos IPC

Permite implementar memoria compartida, semáforos, mensajes.

XI. DESARROLLO DE GESTOR DE MÁQUINAS VIRTUALES

Como hemos vistos en los capítulos anteriores la situación actual que viven los responsables de tecnología dado que dedican un servidor físico a cada tipo de aplicación (Exchange, servidores Web, servidor de aplicaciones, bases de datos), produciendo un ¨server sprawl¨ (despliegue desbordado de servidores o proliferación de servidores) y sus demás desventajas que esto ocasiona.

Actualmente podemos evitar este problema con las nuevas tecnologías de hardware que hacen inadecuada esta metodología de servidores dedicados. A su vez se tiende cada día a la consolidación y optimización de recursos aunque aun se tiene en cuenta obviamente el costo de hardware debido a que es un factor relevante.

Para enfocar la solución a esta problemática se utilizó una técnica de virtualización a nivel de sistema operativo, para aprovechar al máximo los recursos de hardware brindados, utilizando poderosas herramientas y sistema operativo Linux como base y punto de partida de este trabajo de tesis.

Los requerimientos iniciales para llevar a cabo el desarrollo del sistema fueron los siguientes:

I. Utilizar una computadora como equipo base denominado host a virtualizar y software de virtualización de código abierto que permita crear distintos entornos virtuales con sistema operativo Linux con características de memoria, CPU, disco, red y servicios distintos,
II. Modificar distintos recursos de las máquinas para probar y poner en funcionamiento la herramienta de virtualización “openvz”.
III. Diferenciar las tareas de virtualización que puede llevar a cabo el administrador del gestor de máquinas virtuales y las tareas que el cliente puede implementar.
IV. Desarrollar una interfaz Web para que el administrador o cliente administre las distintas máquinas virtuales.
V. Desarrollar un protocolo de aplicación para que reciba las instrucciones desde la interfaz Web, y el sistema operativo de la máquina host las lleve a cabo.

Para llevar a cabo los requerimientos iniciales se instalo en el computador principal el sistema operativo Linux con distribución debian 5.0 y la herramienta de virtualización OPENVZ porque el modelo por sus características se adaptaba mejor a la implementación del gestor de servidores virtuales a nivel de sistema operativo.

Se mencionaron las características OPENVZ en el capítulo anterior pero además es importante mencionar otras como que tiene soporte para checkpointing y para migraciones a tiempo real para procesadores IA64, una característica que no ofrece ningún otro software open source de virtualización del sistema operativo y que permite a los administradores de sistemas mover servidores virtuales entre los servidores físicos sin la intervención del usuario final y sin necesidad de disponer de costosos sistemas de almacenamiento. Soporte para NFS (Sistema de archivos de red) que permite el acceso a los archivos de disco de red desde los entornos virtuales de OpenVZ. Soporte estándar para redes VLAN (IEEE802.1Q) en entornos virtuales, lo que permite que cada paquete de red pueda ser etiquetado en redes diferentes. Soporte para FUSE (Sistema de archivo en el espacio de usuario), que posibilita que por ejemplo: un servidor FTP o SSH se muestre como un sistema de archivos dentro de un entorno virtual.

Actualmente se está trabajando para que tenga la capacidad de permitir programar el I/O por entornos virtuales así como el recuento de I/O de cada entorno virtual. Esto permitirá mejorar la distribución de las tasas de transferencia de entradas/salidas (evitando así posibles saturaciones) a través de todos los entornos virtuales. Asimismo, la función de programación permitirá establecer prioridades para que así algunas máquinas virtuales tengan “mayor prioridad” a lo hora de acceder al disco mientras otros tengan una “prioridad menor”.

El gestor de entornos virtuales desarrollado tiene la capacidad de configurar entornos virtuales con sus respectivos sistemas operativos y parámetros de memoria, disco, CPU, red. Admitiendo también asignar valores mínimos y máximos a cada uno de los parámetros definidos en las máquinas virtuales. Cada una debería permanecer dentro de sus límites y no afectar a otros entornos virtuales de ninguna manera.

Para poder concluir los requerimientos iniciales se diseño una estructura cliente servidor de manera que el usuario pueda ingresar desde cualquier lugar a su entorno virtual como sus datos personales y gestionarlo según los servicios contratados.

De la misma manera el administrador de las máquinas puede monitorizar y gestionar los recursos de todas las máquinas virtuales de igual forma que en modo consola (shell).

La interfaz gráfica Web, le permite al administrador y a los distintos usuarios poder manipular las máquinas virtuales de una manera mucho más amigable que la que ofrece el lenguaje de virtualización Openvz a través de la consola (shell).

I. DISEÑO DEL SISTEMA

El diseño del gestor de entornos virtuales se basa en la siguiente estructura tipo cliente servidor.

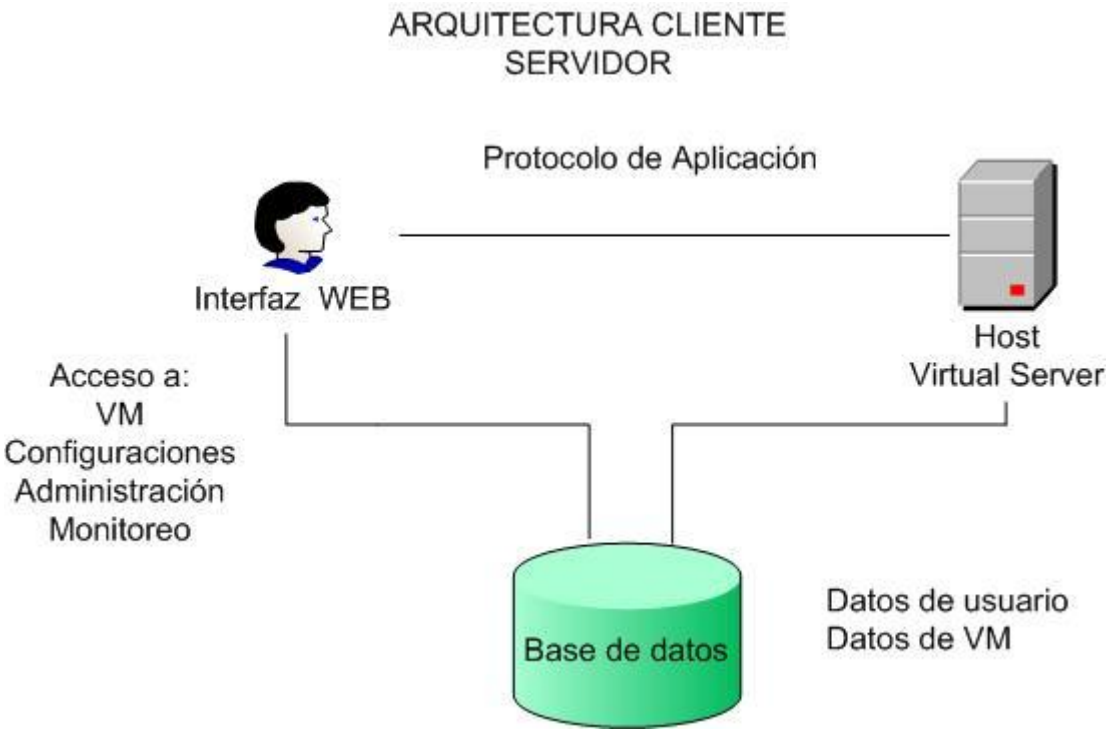


Figura 14: Arquitectura del gestor de entornos virtuales (Entornos Virtuales)

El software está compuesto como se muestra en la figura 14, por una interfaz Web que se comunica con un demonio ubicado en el servidor Host que recibe las acciones solicitadas por el usuario que debe realizar, interactuando con el sistema operativo de la máquina anfitriona.
El dialogo entre la aplicación Web y el demonio queda establecida mediante un protocolo de Aplicación llamado “virtual-server”

La interfaz de administración Web está desarrollada con el lenguaje de programación PHP y MySQL:

- PHP (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje "open source" interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Una característica importante para el desarrollo e implementación del

sistema es que permite soporte de sockets.

- MySQL es un sistema de gestión de bases de datos (SGBD) multiusuario, multiplataforma. Está escrito en C y C++, emplea el lenguaje SQL para consultas a la base de datos y está disponible como freeware bajo licencia GPL. Se utilizo una herramienta escrita en PHP llamada PHPMYADMIN con la intención de manejar la administración de MySQL a través de páginas Web, utilizando Internet. Permite crear y eliminar bases de Datos, crear, eliminar y modificar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 50 idiomas. Se encuentra disponible bajo la licencia GPL.

Este tipo de estructura permite una administración de entornos virtuales bastante intuitiva tanto para el cliente como para el administrador. Los datos personales del cliente y de las máquinas virtuales son almacenados en una base de datos.

Las funcionalidades del sistema teniendo en cuenta la estructura mencionada anteriormente (base de datos, interfaz Web, protocolo de aplicación y daemon) se diseñaron mediante los siguientes casos de usos.

INGRESO AL SISTEMA

Para contratar un plan básico, completo o profesional se debe ingresar al domino de DatacenterEG y seleccionar el plan en la pestaña de la derecha llamada “Servidor Virtual”.

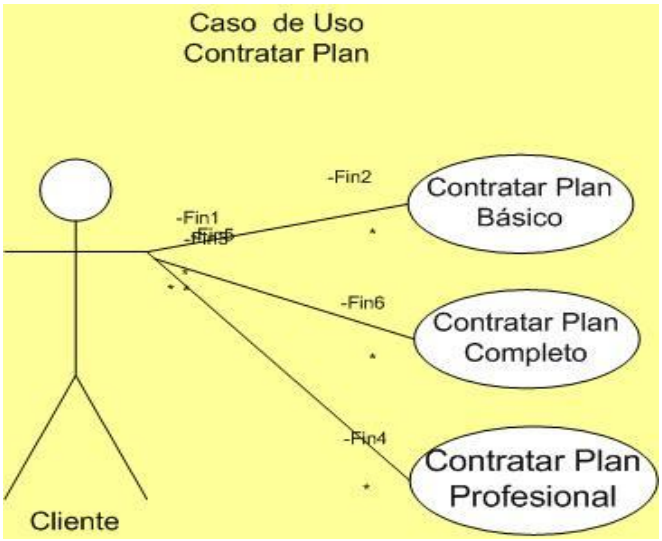
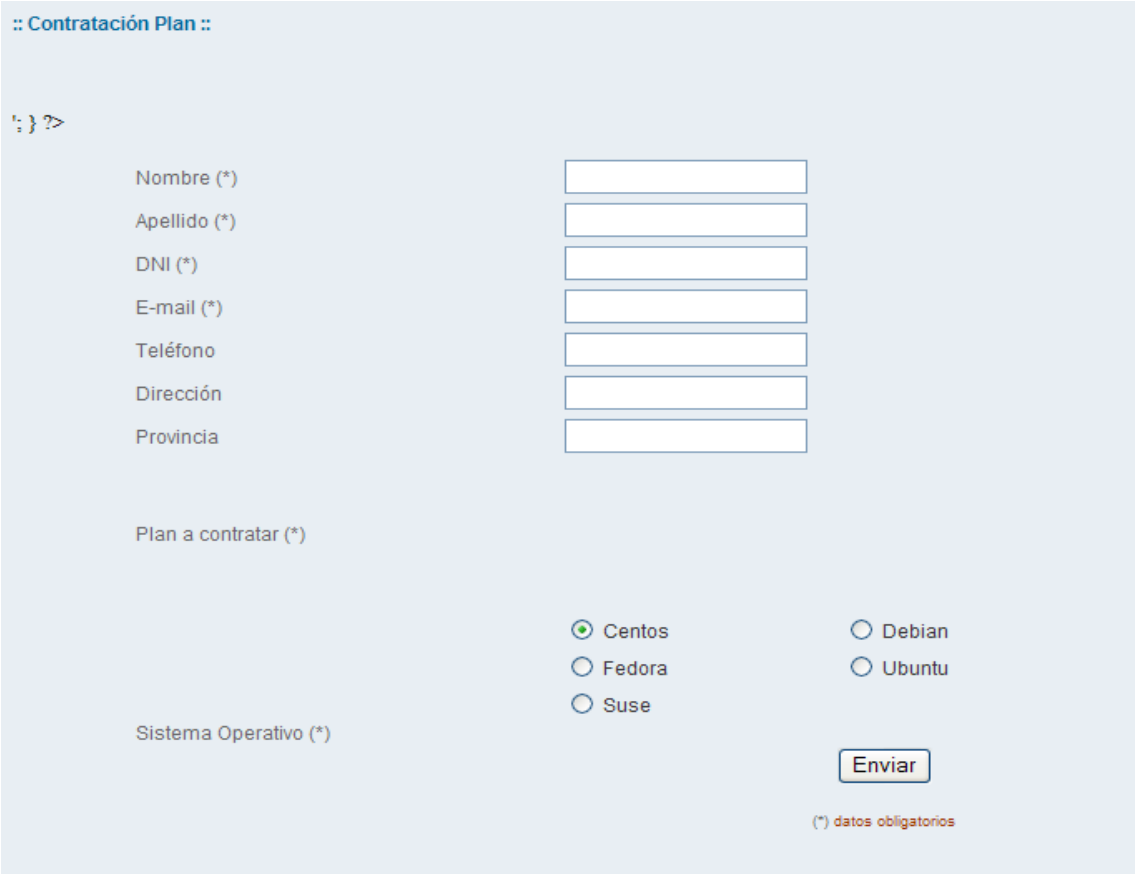


Figura 15: Caso de Uso: Contratar Plan

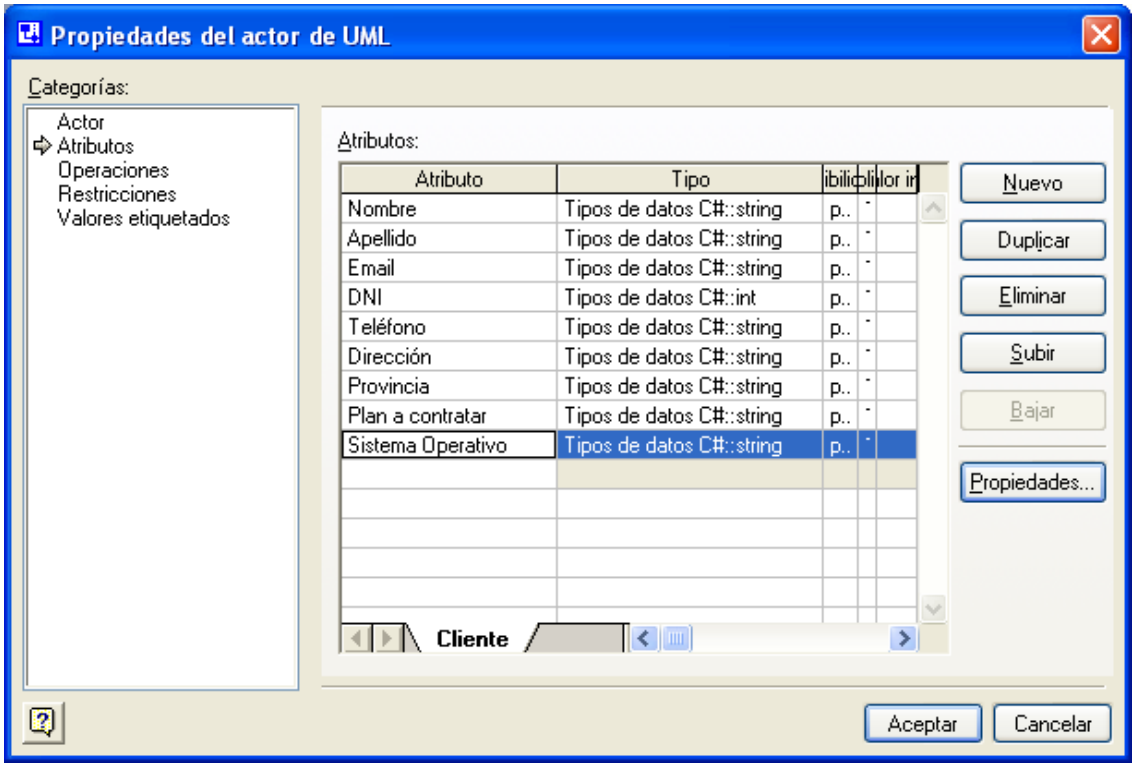
El sistema permite la contratación de distintos planes de máquinas virtuales (mediante el botón "Contratar" ubicado por debajo de la foto de cada plan o por el botón "Contratar" ubicado arriba a la izquierda en la página Web). Estos planes se diferencian por los recursos de memoria, disco y distribución del sistema operativo (Centos, Ubuntu, Fedora, Suse y Debian) de los entornos virtuales.

En el momento de que elige los entornos virtuales mediante el botón “Contratar” el sistema facilita el alta al usuario y a la/las máquinas virtuales correspondientes asociadas al mismo, almacenando los datos en las distintas tablas de la base de datos involucradas en el proceso de contratación.

Para registrar el usuario se le solicitan los siguientes datos mostrados en el formulario a continuación.



Los atributos del cliente requeridos para dar de alta al plan elegido, son los siguientes:



El sistema valida el D.N.I ingresado. En el caso que exista en la base de datos notifica al cliente "Usuario existente" y sino automáticamente se le envía un mail al cliente con el nombre de cuenta, contraseña y dirección de red asignada. Estos datos deberá ingresar en la sección de "Acceso al cliente" para monitorizar la/las máquinas virtuales contratadas.

Es necesario tener instalado un smtp en el host anfitrión que envíe el mail generado al usuario con los datos de la contratación del servicio.
En el momento que el cliente recibe el correo con los datos respectivos ya está en condiciones de hacer uso del servicio.

El sistema tiene la opción de crear máquinas virtuales en el momento que el cliente elige y compra un plan o desde la consola de administración cuando el administrador lo crea pertinente.

VALIDACIÓN Y REGISTRACIÓN DE USUARIO.

El sistema valida dos tipos de usuarios, uno es el administrador de sistema que le permite gestionar y modificar los atributos de distintos entornos y de más

funcionalidades que el usuario común que es el otro tipo de usuario (cliente que contrata los planes de virtualización). Cada uno tiene permisos y funciones específicas individuales.

Para registrarse e identificarse se debe acceder a la ventana de "Acceso Clientes" que está en la parte superior derecha de la página web de DATACENTER.EG e ingresar los datos personales de usuario y clave.

Una vez que la validación del cliente (usuario y password) fue exitosa. El sistema habilita las ventanas correspondientes.

FUNCIONALIDADES DEL SISTEMA.

Como se mencionó anteriormente el sistema diferencia dos tipos de usuario (usuario cliente y usuario administrador).

El usuario común puede monitorizar las máquinas virtuales personales mediante la web o ingresar a ellas mediante ssh con usuario y contraseña. El sistema habilita la ventana de monitoreo denominada "Status"

El usuario administrador puede monitorizar todas las máquinas virtuales creadas y modificar los atributos de las mismas. También puede utilizar el protocolo ssh para acceder a los entornos virtuales. El sistema habilita la ventana de monitoreo "Status" y de "Hardware".

MONITORIZAR MÁQUINAS VIRTUALES

El sistema identifica al cliente y habilita la ventana de monitoreo de la/las máquinas virtuales creadas.

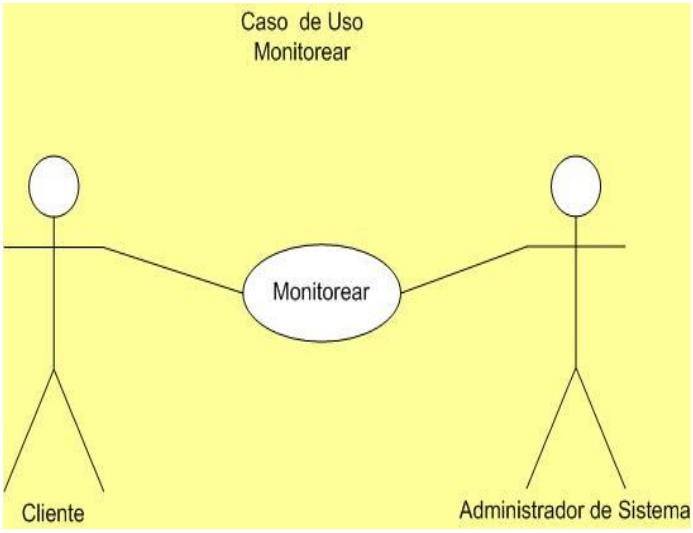
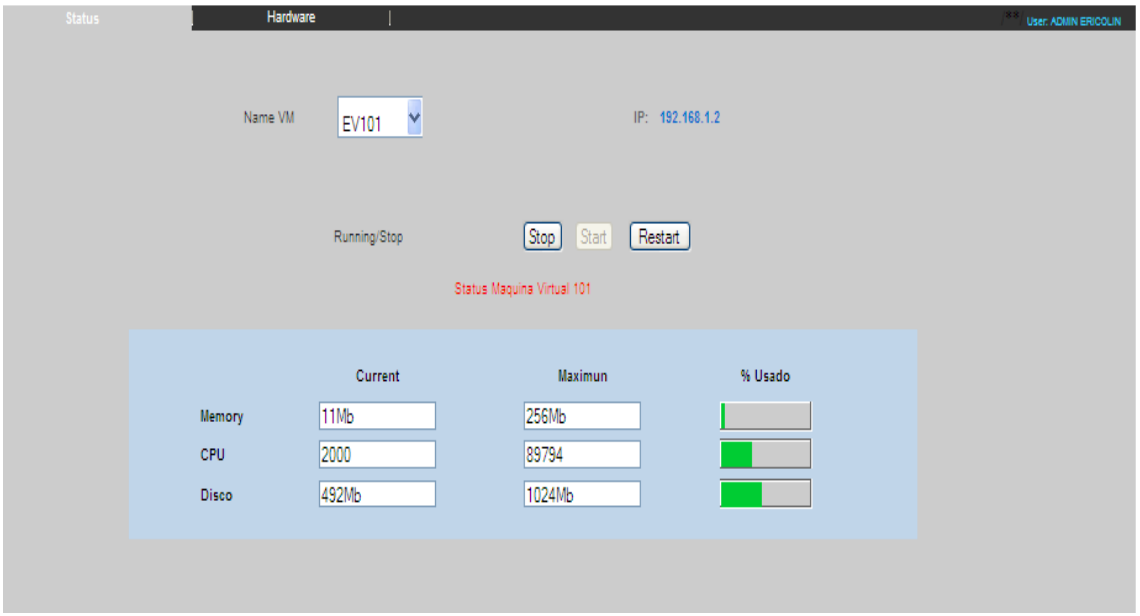


Figura 16: Caso de Uso. Monitorizar

Este caso de uso "Monitorizar" le permite al administrador y al cliente sólo ver el estado de las máquinas (Running y Stop) y los recursos utilizados de las mismas.
El administrador como tiene permisos especiales puede ver todas las máquinas virtuales creadas en el sistema en cambio el cliente sólo chequeará el estado de los entornos virtuales que contrató. Los entornos virtuales se despliegan y se van seleccionado de a uno a la vez en el combo llamado Name VM ubicado en el centro de la ventana llamada "Status".

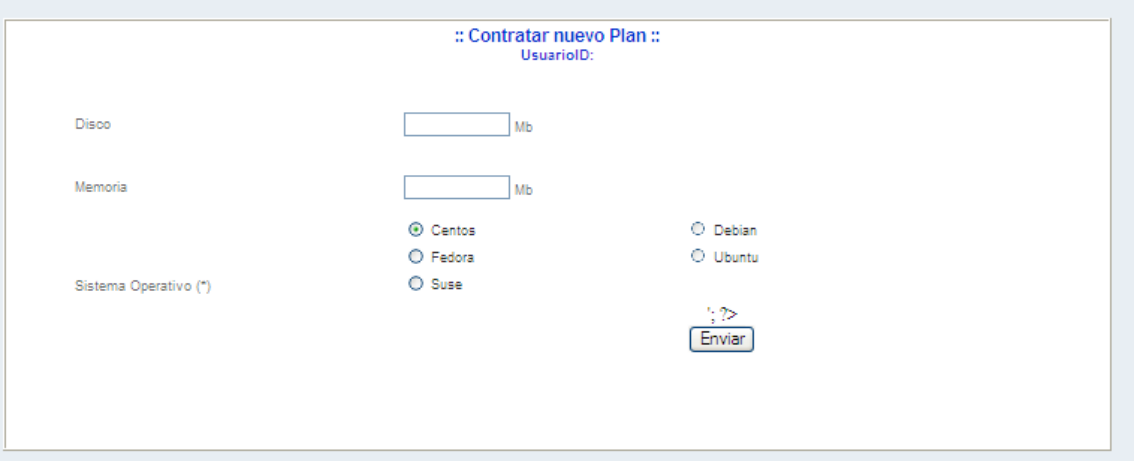


El sistema da un breve detalle del nombre de la máquina y la dirección de red asignada. Como el valor máximo, valor actual de memoria y un porcentaje en un diagrama de barras del disco, memoria y CPU.

El caso de uso MONITORIZAR sólo permite ver los atributos asignados y el estado de los entornos creados. Da la posibilidad de reiniciar, arrancar o apagar los entornos virtuales pero no se puede realizar ningún cambio de memoria, disco o CPU.
Esta tarea está destinada sólo al administrador de sistema y se hace en otra pantalla.

MODIFICAR RECURSOS DE ENTORNOS VIRTUALES

Cuando el usuario es administrador se habilita la ventana denominada “Hardware”.
Donde se puede configurar, modificar y/o actualizar algunos parámetros de las máquinas virtuales que se están ejecutando, asignar máquinas nuevas a algún usuario existente del botón “Contratar” y eliminar alguna ya creada del botón “Eliminar” el cual muestra una pantalla con la información del identificador y nombre de la máquina virtual a eliminar.
El botón “Contratar” despliega la siguiente ventana.



El sistema permite crear un nuevo entorno virtual a un usuario existente asignándole atributos como memoria, disco y distribución del sistema operativo como se muestra en la pantalla anterior.

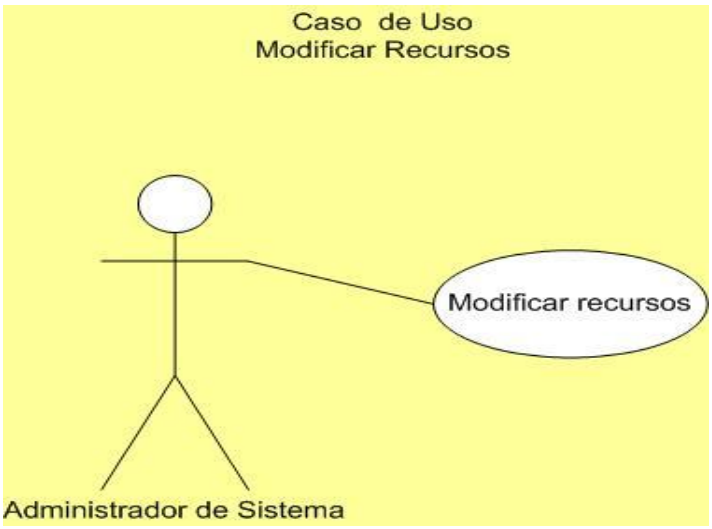


Figura 17: Caso de Uso. Modificar Recursos.

En la pantalla “Hardware” se pueden modificar los valores mínimos y máximos de memoria, disco y CPU como también el hostname, DNS, puerta de enlace y dirección de red.

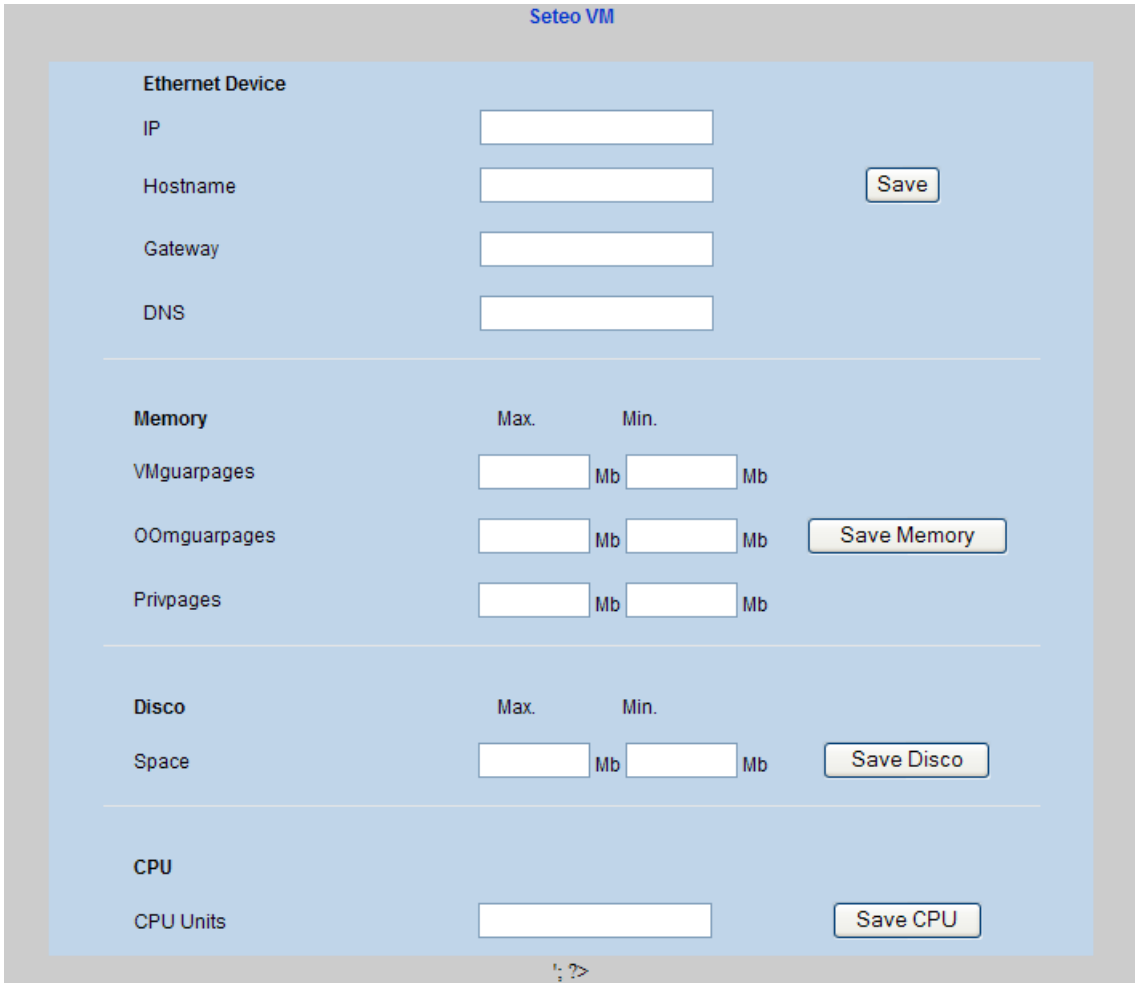
Con el botón “Save” se modifican y/o actualizan uno, algunos o todos los parámetros de red de la máquina seleccionada en el combo ubicado arriba de la ventana “Hardware”.

Con el botón “Save Memory” se modifican y/o actualizan uno, algunos o todos los parámetros de memoria de la máquina seleccionada en el combo ubicado arriba de la

ventana “Hardware”. Es importante aclarar algunos conceptos de los tipos de memoria que openvz maneja:

- **Privvmpages:** permite el control del total de memoria asignada (utilizada) por aplicaciones. Hay que tener en cuenta que privvmpages no garantiza al entorno el total de la cantidad de memoria especificada, ni que los otros entornos virtuales tendrán la capacidad de utilizar su memoria. El mecanismo más útil para controlar la asignación de memoria es vmguarpages.
- **Oomguarpages:** cantidad de memoria reservada para el entorno virtual en situaciones de falta de memoria. Si una aplicación empieza a consumir más memoria que la que tiene la máquina, el sistema entra en condición de “sin-memoria”. En ese caso el sistema operativo comenzará a matar los procesos de los contenedores para liberar algo de espacio y prevenir la muerte total del sistema. Oomguarpages cuenta el total de memoria y espacio swap utilizado por los procesos de un entorno virtual concreto. El “min” del parámetro oomguarpages es la cantidad de memoria garantizada(en condiciones de “sin-memoria”).
- **Vmguarpages:** asignación de memoria garantizada. La cantidad total de memoria garantizada que un contenedor puede asignar es el “min” de vmguarpages, mientras que la cantidad actual de memoria asignada es contado por el parámetro privvmpages.Si el total de memoria asignada actual no execede el “min” de vmguarpages, las asignaciones del contenedor siempre tendrán éxito. Si el total de memoria asignada actual excede del garantizado pero está por debajo de la barrera de privvmpages, dependerá del total de recursos que tenga el sistema.

Con el botón “Save Disco” se modifica el espacio en disco de la máquina seleccionada en el combo ubicado arriba de la ventana “Hardware”.
Con el botón “Save CPU” se modifica las CPU units de la máquina seleccionada en el combo ubicado arriba de la ventana “Hardware”.



Se recomienda guardar cada vez que se carga información en el sistema, dado que se está trabajando en una base remota a la que se tiene acceso a través de la red. De esta forma se evitará perder información por inestabilidad del sistema de conexión.

Es importante aclarar que los atributos que se quieren modificar y/o actualizar requieren un cierto tiempo ya que se modifican los parámetros con las máquinas virtuales en funcionamiento.

ADMINISTRACIÓN DE TEMPLATES PRECONFIGURADAS Y CLAVE DEL USUARIO ADMINISTRADOR.

El sistema permite la administración de las plantillas preconfiguradas de OPENVZ, es decir una vez que ha sido instalada correspondientemente en el sistema operativo se puede insertar en la base de datos a través del entorno web con el mismo nombre que se instalo en la máquina host.

Una vez realizada esta acción el sistema actualizará automáticamente las opciones de las distribuciones de sistema operativo que el sistema le brinda al usuario o administrador mediante La interfaz Web en el momento que se requiera instanciar una máquina virtual. . Esto se realiza ingresando al sistema como usuario administrador y seleccionando la pestaña "Administration en la opción de "Templates".

Por defecto el sistema tiene una clave del usuario root o administrador, la cual puede ser modificada desde la interfaz web. Esto se realiza ingresando al sistema como usuario administrador y seleccionando la pestaña "Administration". En esta ventana el usuario tiene la posibilidad de modificarla.

ELIMINAR ENTORNO VIRTUAL

El sistema admite la eliminación de cualquier entorno virtual creado por el administrador con anterioridad desde la pantalla "Hardware".
Se realiza mediante el botón "Eliminar" y seleccionando la máquina a eliminar en el combo cargado con la asociación del identificador y dueño de la misma.

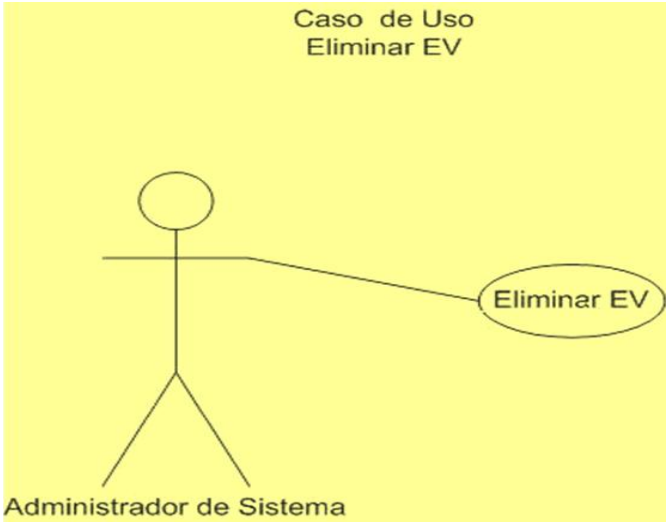


Figura 18: Caso de Uso. Eliminar Entorno Virtual.

Cada una de las funciones que se realizan en el sistema de virtualización están definidas en distintas clases y almacenadas en una base de datos de tal forma que cuando se invoca a una de ellas desde la interfaz Web instantáneamente se realiza una conexión al

servicio (daemon) corriendo en el puerto 9006 tcp en la máquina host. El cual recibe distintos parámetros que debe interpretar para ejecutar las funciones de virtualización con Openvz requerida desde la interfaz por el administrador o cliente.

Este servicio será el responsable de recibir las instrucciones desde la interfaz Web y ejecutar las siguientes funciones y parámetros de virtualización detallas a continuación.

Las respuestas exitosas o de fracasos de estas funciones son mostradas en la interfaz Web de manera que sea mucho más manipulable para los usuarios.

XII. IMPLEMENTACIÓN DEL SISTEMA

Para llevar a cabo la implementación, configuración y desarrollo del gestor de máquinas virtuales se tuvo en cuenta lo siguiente:

I. Virtualizador OpenVz.
II. Interfaz de Administración Web.
III. Daemon (virtual-server) y Protocolo de Aplicación.
IV. Base de datos.

I. VIRTUALIZADOR OPENVZ

Una vez instalado el paquete correspondiente a openvz se puso en marcha la ejecución de los comandos de virtualización que se utilizaron en la programación del sistema. La sintaxis de los comandos es la siguiente:

- CREAR MÁQUINA VIRTUAL

vzctl create ID --ostemplate PLANTILLA

Parámetros
ID: identificador de la máquina virtual
PLANTILLA: sistema operativo a instalar

Ejemplo
Un ejemplo de creación:
vzctl create 101 -–osstemplate debian-3.1-i386-minimal
creating VE private area (debian-3.1-i386-minimal)
performing postcreate actions
VE private area was created

Genérica
Vzctl create (parámetro) -- ostemplate (parámetro)

- ARRANCAR MÁQUINA VIRTUAL

Por defecto las máquinas son creadas pero no arrancadas, para ello deberemos realizar:
vzctl start ID

Ejemplo
vzctl start 101

Genérica
vzctl start (parámetro)

- ELIMINAR UNA MÁQUINA VIRTUAL

Para parar y borrar una máquina virtual, realizaremos lo siguiente:
Primero paramos la máquina.

vzctl stop ID

Y luego la eliminamos

$ vzctl destroy ID
rm /etc/vz/conf/ID.conf.destroyed
umount /vz/private/ID
rm -rf /vz/private/ID

Ejemplo
vzctl stop 101

Genérica
vzctl stop (parámetro)

- RESETEAR UNA MAQUINA VIRTUAL

vzctl restart ID

Ejemplo
vzctl restart 101

Genérica
vzctl restárt (parámetro)

- MODIFICAR RECURSOS DE MAQUINA VIRTUAL

Administración de memoria
vzctl set ID --vmguarpages xxxM --save
vzctl set ID --oomguarpages xxxM --save
vzctl set ID --privvmpages xxxxx:xxxxx --save

Ejemplo para una maquina de 256 Mb
vzctl set 101 --vmguarpages 256M --save
vzctl set 101 --oomguarpages 256M --save
vzctl set 101 --privvmpages 49152:53575 --save

Genérica
Vzctl set (parametro) –vmguarpages/oomguarpages/privvmpages (parametro) --save

Administración de disco
vzctl stop ID
vzctl set ID --diskspace 20000000:22000000 --save
vzctl start ID

Ejemplo para una maquina de 20 Gb
vzctl stop 101
vzctl set 101 --diskspace 20000000:22000000 --save
vzctl start 101

Genérica

Vzctl set (parámetro) –diskspace (parámetro) -- save

Administración de CPU

vzctl set ID --cpuunits 1500 --cpulimit 4 --save

Ejemplo

vzctl set 101 --cpuunits 1500 --cpulimit 4 --save

Genérica

vzctl set (parametro) --cpuunits (parametro) --cpulimit (parametro) --save

Configuración de parámetros de red

vzctl set ID --hostname NAME --save

vzctl set ID --ipadd X.X.X.X --save

vzctl set ID --nameserver X.X.X.X --save

Ejemplo

vzctl set 101 --hostname virtual1 --save

vzctl set 101 --ipadd 192.168.0.101 --save

vzctl set 101 --nameserver 192.168.0.254 --save

La sintaxis que utiliza este virtualizador es la que se utilizó para definir las clases en el código de programación.

II. INTERFAZ DE ADMINISTRACIÒN WEB

En el capitulo anterior se definieron las funcionalidades del sistema y cómo interactúa el administrador que tiene privilegios para monitorizar todas las máquinas virtuales creadas, configurar los recursos de cada uno (dirección de red, hostname, puerta de enlace, DNS, memoria, CPU, disco) y también reiniciar, arrancar, parar, apagar o eliminar WM.

El cliente sólo puede monitorizar los recursos de cada WM, ver el estado de las máquinas virtuales y ejecutar los comandos como reiniciar, arrancar, apagar los entornos virtuales personales.

Cada una de estas acciones se definió como clases.

Función que crea una VM.

```
function createVM($so) {
        $this->command = 'create '. $this->_id  .' '. $so;
   }
```

Función que para una VM.

```
 function stopVM() {
      $this->command = 'stop '. $this->_id;
   }
```

Función que inicia una MV.

```
function startVM() {
      $this->command = 'start '. $this->_id;
   }
```

Función que reinicia una MV.

```
function restartVM() {
      $this->command = 'restart '. $this->_id;
   }
```

Función que muestra la memoria asignada en la MV.

```
function getMemory() {
      $this->command = 'getmemory '. $this->_id;
   }
```

Función que muestra la cantidad de disco asignada en la MV.

```
function getDisk() {
      $this->command = 'getdisk '. $this->_id;
   }
```

Función que muestra el CPU utilizado de la MV.

```
function getCPU() {
        $this->command = 'getcpu';
    }
```

Función que setea la memoria de la MV.

```
 function setMemoryVM($memory1, $memory2, $unitsMemory = 'M') {
        $this->command = 'setmemoryvm '. $this->_id .' '.
$memory1.$unitsMemory.':'.$memory2.$unitsMemory;
    }
```

Función que setea la memoria de la MV.

```
 function setMemoryOO($memory1, $memory2, $unitsMemory = 'M') {
         $this->command = 'setmemoryoo '. $this->_id .' '.
$memory1.$unitsMemory.':'.$memory2.$unitsMemory;
    }
```

Función que setea la memoria de la MV.

```
function setMemoryP($memory1, $memory2,$unitsMemory = 'M') {
        $this->command = 'setmemoryp '. $this->_id .' '.
$memory1.$unitsMemory.':'.$memory2.$unitsMemory;
    }
```

Función que setea el disco de la MV.

```
function setDisk($space1, $space2, $unitsSpace = 'M') {
        $this->command = 'setdisk '. $this->_id .' '. $space1.$unitsSpace.':'.
$space2.$unitsSpace;
    }
```

Función que setea el nombre de la MV.

```
function setName($name) {
        $this->command = 'setname '. $this->_id .' '. $name;    }
```

Función que setea el CPU de la MV.

```
function setCPU($cpu) {
        $this->command = 'setcpu '. $this->_id  .' '. $cpu;
    }
```

Función que setea el DNS de la MV.

```
function setDNS($dns) {
         $this->command = 'setdns '. $this->_id  .' '. $dns;
    }
```

Función que setea el IP de la MV.

```
function setIP($ip) {
        $this->command = 'setip '. $this->_id  .' '. $ip;
    }
```

Función que setea el Gateway de la MV.

```
      $this->command = 'setgateway '. $this->_id  .' '. $gateway;
    }
```

Función que elimina una MV.

```
function destroyVM() {
        $this->command = 'destroy '. $this->_id;
    }
```

Función que elimina la IP de la MV.

```
 function deleteIP() {
        $this->command = 'deleteip '. $this->_id;
    }
```

Todas estas actividades que realizan cambios sobre el sistema operativo requieren de privilegios de root (crear, reiniciar, modificar atributos de máquinas virtuales) y dado a que la administración de los distintos entornos virtuales es a través de la interfaz Web, hubiera sido necesario correr el servidor de Web con privilegios de root lo cual no está

recomendado. Para poder realizar estas tareas de virtualización y tener una división de privilegios.

Por este inconveniente se desarrollo un protocolo de aplicación entre el host en donde corre el daemon y la interfaz Web logrando una comunicación entre ellos utilizando un socket. El daemon realiza un bind sobre una interfaz local (127.0.0.1) permitiendo que solamente sea accedido desde la misma máquina donde corre la administración, pero en ningún caso es posible ejecutar un comando del sistema operativo desde la interfaz Web.

III. DAEMON (VIRTUAL-SERVER) Y PROTOCOLO DE APLICACIÓN

El daemon interactúa con la interfaz a través de un socket tcp utilizando un protocolo de aplicación.
Este protocolo de aplicación representa el conjunto de acciones que la interfaz puede solicitar a la máquina host.
Está compuesto por verbos (acciones) y argumentos para los mensajes desde la interfaz a la máquina host y las respuestas devuelven la acción solicitada.
Este protocolo de aplicación tiene la capacidad de recibir las instrucciones desde la Web realizadas por el usuario ubicado físicamente en cualquier lugar, recibirlas y ejecutarlas en el Host virtualizado.

Este protocolo recibe los parámetros que les envía las clases definidas anteriormente compuestas por el comando a ejecutar seguidas de los parámetros.

A continuación se detallan los mensajes del protocolo:

Mensaje de interfaz	Función a ejecutar	Atributos de la función	Descripción
Create	Vzctl create ()	ID: identificador de la máquina virtual. PLANTILLA: sistema operativo a instalar. Ej: vzctl create 101 –ostemplate debian-3.1-i386-minimal	Crea un entorno virtualizado asociado a un identificador único y con su respectivo sistema operativo.
Start	Vzctl Start ()	ID: identificador de la máquina virtual. Ej: vzctl start ID	Por defecto las máquinas son creadas pero no arrancadas, para ello debemos ejecutar dicha función.
Stop	Vzctl stop ()	ID: identificador de la máquina virtual. Ej: vzctl stop ID	Detiene la maquina virtual.

DeleteVM	Vzctl destroy ()	ID: identificador de la máquina virtual. Ej: vzctl stop ID $ vzctl destroy ID rm etc/vz/conf/ID.conf.destroyed umount /vz/private/ID rm -rf /vz/private/ID	Primero paramos la máquina y luego la eliminamos.
Restart	vzctl restart ()	ID: identificador de la máquina virtual. Ej: vzctl restart 101	Función reiniciar la maquina virtual.

SetMemoryVM SetMemoryOC SetMemoryP	vzctl set ()	ID: identificador de la máquina virtual VMGUARPAGES: memoria disponible para el entorno virtual. OOMGUARPAGES: garantía de memoria cuando se excede el valor provvmpages PRIVVMPAGES: cantidad de memoria asignada por aplicación. Ej: para una maquina de 256 Mb vzctl set 101 --vmguarpages 256M --save vzctl set 101 --oomguarpages 256M --save vzctl set 101 --privvmpages 256M –save	Configurar y/o modificar los parámetros de memoria por entorno virtual.

SetDisk	vzctl set ()	ID: identificador de la máquina virtual DISKPACE: tamaño de disco. Ej: para una maquina de 20 Gb vzctl stop 101 vzctl set 101 --diskspace 20000000:22000000 --save vzctl start 101	Configurar y/o modificar capacidad de disco.
SetName	vzctl set ()	ID: identificador de la máquina virtual HOSTNAME: nombre del entorno virtual. Ej: vzctl set 101 --hostname virtual1 --save	Asigna nombre individual a cada entorno virtual.
GetMemory	vzctl exec 101 free –m	ID: identificador de la máquina virtual. Ej vzctl exec 101 free –m	Pregunta por la cantidad de memoria, disponible, usada y libre de cada entorno virtual.

GetDisk	vzctl exec ()	ID: identificador de la máquina virtual. Ej: vzctl exec 101 df –m	Pregunta por la capacidad de disco disponible, usado y libre de cada entorno virtual.
GetCPU	vzcpucheck –v	Sin atributos.	Pregunta por la cantidad de CPU usado.
SetIp	vzctl set ()	ID: identificador de la máquina virtual IPADD: dirección IP. vzctl set 101 --ipadd 192.168.0.101 --save	Configurar y/o modificar distintos atributos de la red de cada entorno virtual.
SetDns	vzctl set ()	ID: identificador de la máquina virtual. NAMESERVER: configuración de dns vzctl set 101 --nameserver 192.168.0.254 –sabe	Configurar y/o modificar distintos atributos de la red de cada entorno virtual.

El daemon (demonio) está desarrollado en lenguaje de scripting del shell bash de Linux. Es un script que puede ser invocado desde la línea de comando de la máquina host. Para poder transformarlo en un servicio tcp se utilizo xinetd, que se encarga de

realizar toda la gestión de tcp para la comunicación. Es decir, xinetd permite que la entrada estándar y la salida estándar del script sea un socket tcp.

La configuración y la implementación del daemon y servicio de xinetd es la siguiente:
En el servicio de xinetd en /etc/xinetd.d/virtual se creó un servicio con el nombre virtual

```
service virtual
{
disable = no
socket_type = stream
wait = no
user = root
server = /usr/local/virtual/virtual-server.sh
log_on_success += DURATION USERID
log_on_failure += USERID
bind = 127.0.0.1
}
```

El archivo /etc/services también se agrega una línea que asocia el servicio al puerto tcp donde se va a realizar el bind.

```
virtual         9006/tcp
```

El archivo de configuración /etc/xinetd.d/virtual con la línea de configuración en /etc/services le dicen a xinetd que debe realizar un bind del ejecutable usr/local/virtual/virtual-server.sh en el puerto tcp 9006 sobre la dirección IP 127.0.0.1 y que este ejecutable correrá con privilegios de root.

De esta manera se logra “prender” el daemon virtual-server y dejarlo asociado a la máquina local.

IV. BASE DE DATOS

La interfaz de administración Web se implementó utilizando el lenguaje de programación PHP, que posee características óptimas para el acceso a base de datos y buen soporte de la API de sockets.

La interfaz debe comunicarse con el servidor virtual y permitir que el administrador y/o cliente realice las acciones necesarias.
La información de alto nivel se almacena en la base de datos utilizando la siguiente estructura de datos

```
CREATE DATABASE `tesis` ;

CREATE TABLE `usuario`
(
        `usuarioID` bigint (20) NOT NULL AUTO_INCREMENT ,
        `nombre` varchar (20) NOT NULL,
        `apellido` varchar (20) NOT NULL,
        `email` varchar (20) NOT NULL,
        `dni` integer (10) NOT NULL,
        `pass` varchar (20) NOT NULL,
        `telefono` varchar (20),
        `direccion` varchar (40),
        `provincia` varchar (15),
        `usuarioTipoID` bigint (20) NOT NULL,
        PRIMARY KEY (`usuarioID`)
) TYPE=InnoDB CHARACTER SET latin1 COLLATE latin1_swedish_ci;

CREATE TABLE `usuarioTipo`
(
        `usuarioTipoID` bigint (20) NOT NULL AUTO_INCREMENT ,
        `desc` varchar (20) NOT NULL,
        PRIMARY KEY (`usuarioTipoID`)
) TYPE=InnoDB CHARACTER SET latin1 COLLATE latin1_swedish_ci;
```

```
INSERT INTO `usuarioTipo` VALUES(0,'Administrador');
INSERT INTO `usuarioTipo` VALUES(0,'Comun');

CREATE TABLE `SO`
(
	`SOID` bigint (20) NOT NULL AUTO_INCREMENT ,
	`desc` varchar (20) NOT NULL,
	PRIMARY KEY (`SOID`)
) TYPE=InnoDB CHARACTER SET latin1 COLLATE latin1_swedish_ci;

INSERT INTO `SO` VALUES(0,'Centos');
INSERT INTO `SO` VALUES(0,'Fedora');
INSERT INTO `SO` VALUES(0,'Gentoo');
INSERT INTO `SO` VALUES(0,'OpenSuse');
INSERT INTO `SO` VALUES(0,'Ubuntu');
INSERT INTO `SO` VALUES(0,'Slackware');
INSERT INTO `SO` VALUES(0,'Mandrake');
INSERT INTO `SO` VALUES(0,'Fedora');
INSERT INTO `SO` VALUES(0,'Red Had');
INSERT INTO `SO` VALUES(0,'Suse');

CREATE TABLE `usuarioVM`
(
	`usuarioVMID` bigint (20) NOT NULL AUTO_INCREMENT ,
	`usuarioID` bigint (20) NOT NULL ,
	`VMID` integer (3) NOT NULL ,
	`nameVM` varchar (20) NOT NULL,
	`kernelVersion` varchar (20) NOT NULL,
	`soID` integer (2) NOT NULL,
	`planID` integer (2) NOT NULL,
	PRIMARY KEY (`usuarioVMID`)
) TYPE=InnoDB CHARACTER SET latin1 COLLATE latin1_swedish_ci;
```

```
ALTER TABLE `usuarioVM` ADD CONSTRAINT FK_usuarioVM_1 FOREIGN KEY (usuarioID) REFERENCES `usuario`(usuarioID);

CREATE TABLE `plan`
(
        `planID` bigint (20) NOT NULL AUTO_INCREMENT ,
        `ram` integer (3) NOT NULL,
        `disco` integer (3) NOT NULL,
        PRIMARY KEY (`planID`)
) TYPE=InnoDB CHARACTER SET latin1 COLLATE latin1_swedish_ci;

INSERT INTO `plan` VALUES(1,64,1);
INSERT INTO `plan` VALUES(2,128,2);
INSERT INTO `plan` VALUES(3,256,4);
```

XIII. CONCLUSIONES

El sistema desarrollado está basado en una arquitectura cliente servidor que demostró ser confiable en la misma, dado que permitió administrar y realizar las funcionalidades deseadas por el usuario y administrador sobre los entornos virtuales, según las opciones propuestas por el software de manera amigable y efectiva.

Como relación costo/beneficio el sistema permite que la organización que lo implemente tenga un ahorro significativo en equipamiento y en gestión en cuanto a las máquinas virtuales y a los recursos asignados a la misma de manera más controlada y racionalizada.
El sistema se puede instalar fácilmente en un hardware convencional de PC con un sistema operativo Linux y una herramienta de virtualización open-source (OPENVZ).
Con respecto a las características y funcionalidades del sistema de gestión de entornos virtuales mencionadas anteriormente, podemos concluir que las particularidades de OPENVZ y en particular Linux como sistema operativo permitieron que la implementación de software de gestión se llevara a cabo sin problemas logrando los objetivos planteados en el trabajo.
El sistema permite hacer frente al continuo crecimiento de los servicios de las organizaciones, por tal motivo y por todas las funcionalidades de la misma, el presente trabajo deja abierto para futuros estudios algunos aspectos de OPENVZ que no han sido cubiertos como por ejemplo migración de máquinas en vivo, backup, NFS, FUSE y vlan.

Dicho sistema se adapta cabalmente a la situación actual que viven las organizaciones como por ejemplo: bancos, hospitales, colegios, proveedores de servicios, organismos públicos o privados entre otros, que tienden a una infraestructura basada en sistemas de máquinas virtuales y cuentan con tecnologías enfocadas a la web.
Dado que les ofrece entornos seguros donde experimentar docencia, probar aplicaciones de desarrollo, aplicaciones web, hosting, seguridad, portabilidad, backup de máquinas enteras no sólo datos, seguridad (corta fuego, firewall) e independencia del hardware.

El usuario puede manipular las máquinas como si fueran sus propios servidores (apagar, reiniciar, acceder mediante ssh para la instalación de servicios u programas necesarios

y monitorizar el estado de las mismas). Se puede decir que "***es una nueva modalidad de alojamiento actual"*** que le da al usuario una cierta libertad sobre el servidor mediante una interfaz Web.

Generalmente este tipo de herramienta se utiliza para implementar ambientes de desarrollo y pruebas de software, instalar aplicaciones que requieran alta disponibilidad y brindar fiabilidad en los servicios instalados, consolidación y contención de los entornos virtuales, simplificación de la infraestructura y una sencilla protección y gestión de las máquinas virtuales ofrecidas mediante una interfaz Web.

XIV. BIBLIOGRAFÍA

- Dummies, Sun and AMD . Virtualization. 2008. Special Edition. Wiley Publishing ISBN 07030-577.
- James. E. Smith. Ravi Nair. Virtual Machines. 2005. ISBN 1-55860-910-5. Elsevier.
- Josep Ros. Virtualización Corporativa con Wmware. 2009 Ncora Tecnology S.L.
- James E. Smith .The Architecture of Virtual Machines. 2005.Computer Magazine.
- Victor Moreno CCIE. Kumar Reddy. Network Virtualization.2006. Cisco Press, ISBN-10: 1-58705-248-2.
- Jason R. Fink y Matthew D. Sherer Sams. Linux Performance Tuning and Capacity Planning. IBM Readpaper publication.2008. ISBN-10: 0672320819.
- David Marshall Wade A. Reynolds and Dave McCrory. Advanced server. Virtualization VMware and Microsoft Platforms in the Virtual Data Center.2006. Auerbanch Publication .Taylor & Francis Group.
- Jane E. Klobas. Paul D. Jackson. Becoming Virtual Knowledge Management and Transformation of the Distributed Organization. 2008 Physica-Verlag..

- PowerVM Virtualization on IBM System p. Introduction and Configuration. 2008. IBM Readbooks publication.
- PowerVM Virtualization on IBM System p.Managing and Monitoring. 2008. IBM Readbooks publication.
- IBM Virtualization Engine TS7500. Planning. Implementation, and Usage Guide. 2008. IBM Readbooks publication.
- Tulloch, M. Understanding Microsoft Virtualization Solutions. Microsoft Press.2009 ISBN 9780735693371.
- Kevin Crowston. Sandra Sieber. Eleanor Wynn. Virtuality and Virtualization. 2007. Proceedings of the International Federation of Information Processing Working Groups 8.2.USA.
- White Paper IBM: Improving Systems Management and Availability with x86 Virtualization. 2009.
- White Paper DELL. Centros de datos del Futuro.2003.
- White Paper IBM. Creating a Dynamic Infrastructure through Virtualization. 2009.
- White Paper IBM .Virtualization Services. 2008.
- Conferencia de Sergio Talens Oliag en Barcelona. Herramientas de virtualización libres para sistemas GNU/LINUX. 2008.
- Openvz How-to (http://openvz.org/).
- http://www.linux-vserver.org.
- http://www.xen.org.
- http://www.wmware.com
- http://www.virtualbox.org.
- http://namedev.org.
- http://bochs.sourceforge.net.

XV. INDICE DE FIGURAS

XVI APÉNDICE

IDC (International Data Corporation): compañía establecida para la investigación de información de tendencias tecnológicas. Desde 1964, es proveedor líder de inteligencia de mercado y de servicios de asesoría de la industria de tecnologías de información.

ROI (Rendimiento del Capital Invertido): Es la relación que permite determinar la rentabilidad de todos los capitales invertidos en una empresa.

Fórmula

$$ROI = \frac{Beneficio\ neto}{Capitales\ invertidos}$$

Su medida es un número relacionado con la razón Costo/Beneficio.

- El costo es más sencillo de medir: se sabe cuánto se gasta, lo complicado es calcular el beneficio de factores como (el cambio tecnológico, el desorden al controlar y medir finanzas durante un proyecto, factores intangibles como satisfacción de usuarios, mejoras o comunicación.

TCO (Costo Total de Propiedad): proveniente del término anglosajón (Total Cost of Ownership o TCO): es un método de cálculo diseñado para ayudar a los usuarios y a los gestores empresariales a determinar los costes directos e indirectos, así como los beneficios, relacionados con la compra de equipos o programas informáticos.
TCO también incluye los costos relacionados con el uso y mantención de los equipos y sistemas., costos de implementación, capacitación de los usuarios, costos asociados con las fallas o períodos fuera de servicio (planeados o no planeados), incidentes de pérdida de performance, costos por incumplimientos (pérdida de reputación y pérdidas por la recuperación de la falla), espacio, energía, desarrollo, control de calidad, y muchos más. A veces el TCO es llamado Costo Total de Operación.

TCO=suma (costos Directos + costos indirectos)

Por ejemplo, la compra de un servidor puede incluir la compra en sí misma, reparaciones, mantenimiento, actualizaciones, servicios y soporte, redes, seguridad, formación de usuarios y costos de licencias.

Un buen análisis TCO permite descubrir los costos "ocultos" o los costos no obvios que podrían quedar fuera del proceso de compra o del proceso de planificación.

- El análisis comienza con el diseño de modelo adecuado, tal que cobra completamente los costos relacionados con el sistema a evaluar.
- Se deben incluir los gastos de la adquisición, de la operación y los gastos de mejoras, en otras palabras incluye las variables de la operación.

Recursos	Ciclos de vida del sistema		
	Adquisición	Operación	Ampliaciones y cambios
Software			
Hardware			
Personal			
Redes			
Instalaciones			

Con respecto a la virtualización, el TCO disminuye como la consolidación reduce el costo de mantenimiento de los sistemas y existe una mejor gestión del espacio y la energía eléctrica dentro del centro de datos. Las organizaciones deben sin embargo, considerar los costos de licencias de software que puedan surgir de la virtualización en caso que se requieran instancias adicionales de sistemas operativos y software de aplicación dependiendo de cómo la solución ha sido diseñada.