ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA INGENIERÍA EN SISTEMAS

"ESTUDIO COMPARATIVO DE SISTEMAS DE VIRTUALIZACIÓN DE ORDENADORES, POR SOFTWARE, DE DISTRIBUCIÓN LIBRE, PARA DESARROLLAR UNA INFRAESTRUCTURA DE SERVIDORES VIRTUALES EN LA EIS-ESPOCH"

TESIS DE GRADO

Previa la obtención del Título de

INGENIERA EN SISTEMAS INFORMÁTICOS

Presentado Por:

SANTOS VIDAL MARÍA DOLORES

Riobamba – Ecuador

2010

Esta investigación ha sido el fruto de mucho esfuerzo y en su desarrollo se presentaron obstáculos que ponen a prueba mi tolerancia, sabiduría y paciencia. Por ello quiero agradecer sobre todas las cosas a Dios, porque supo guiarme e iluminarme cuando el camino se ponía cuesta arriba.

A mis padres y hermanos, por el apoyo incondicional, por sus enseñanzas y consejos que me alentaron en todo momento a seguir y mirar hacia delante. A los ingenieros Danilo Pastor y Mario Paguay, por el entusiasmo, que me transmitieron en el desarrollo de esta investigación, y por haberne brindado su sincera amistad.

A mis mejores amigos, con quienes hemos compartido experiencia, conocimientos, penas, alegrías, logros, desilusiones, risas y llantos. Amigos que también han sido participes para conseguir esta meta.

Y de manera especial, a todas aquellas personas que de una u otra forma me brindaron su ayuda desinteresada con conocimientos paciencia y generosidad.

El esfuerzo de toda mi vida estudiantil; el apoyo brindado incondicional sin interés de por medio, el empeño en entregarne día a día su sabiduría, su amor y tolerancia; por la fortaleza que me transmitieron; por enseñarne a tener presente que nunca debemos renunciar ante nada; y por la confianza que siempre estuvo presente en todo momento, dedico este trabajo

A mis padres.

María Dolores

Yo, María Dolores Santos Vidal soy responsable de las ideas, doctrinas y resultados expuestos en esta tesis; y, el patrimonio intelectual de la Tesis de Grado pertenece a la ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO"

_____

María Dolores Santos Vidal

FIRMAS RESPONSABLES Y NOTA

NOMBRE	**FIRMA**	**FECHA**
Dr. Romeo Rodríguez **DECANO FACULTAD DE INFORMÀTICA Y ELECTRÓNICA**	_____	_____
Ing. Iván Menes **DIRECTOR DE ESCUELA INGENIERÍA EN SISTEMAS**	_____	_____
Ing. Danilo Pastor **DIRECTOR DE TESIS**	_____	_____
Dr. Mario Paguay **MIEMBRO DEL TRIBUNAL**	_____	_____
Ldo. Carlos Rodríguez **DIRECTOR CENTRO DE DOCUMENTACIÓN**	_____	_____

NOTA DE LA TESIS: _____

Índice de Abreviaturas

TI	Tecnologías de la Información
SO	Sistema Operativo
HW+SO+APP	Hardware, Sistema Operativo, Aplicaciones
FSF	(Free Software Fundation), Fundación para el Software Libre
ESPOCH	Escuela Superior Politécnica de Chimborazo
DESITEL	Departamento de Sistemas y Telemática
EIS	Escuela en Ingeniería en Sistemas
DHCP	(Dynamic Host Configuration Protocol), Protocolo de Configuración Dinámica de Host
DNS	(Domain Name System), Sistema de Nombres de Dominio
FTP	(File Transfer Protocol), Protocolo de Transferencia de Archivos
SACV	Sistema Administrador Central de Virtualización
XP	(eXtreme Programming), Programación Extrema
GPL	(General Public License), Licencia Pública General
PC	(Personal Computer), Computador Personal
RAM	(Random Access Memory), Memoria de Acceso Aleatorio
CPU	(Central Procesing Unit), Unidad Central de Procesamiento
NIC	(Network Interface Card), Tarjeta de Interfaz de Red
API	(Programación de Aplicaciones), Application Programming Interface
MIPS	Microprocessor without Interlocked Pipeline Stages
CP	Control Program
VPS	(Virtual Private Servers), Servidores Privados Virtuales
OMPI	Organización Mundial de la Propiedad Intelectual
RAID	(Conjunto Redundante de Discos Independientes), Redundant Array of Inexpensive Disks
SAN	(Storage Area Network), Red de Área de Almacenamiento
VPN	(Virtual Private Network), Red Privada Virtual
NAT	(Network Address Translation), Traducción de Direcciones de Red
CMS	(Sistema de Monitor Convencional), Conversational Monitor System
IEEE	Instituto de Ingenieros Eléctricos y Electrónicos
OMPI	Organización Mundial de la Propiedad Intelectual
TDU	Términos de Uso

CLUF	Contrato de Licencia de usuario Final
OEM	Original Equipment Manuafacturer
TCO	(Total Cost of Ownership), Coste total de la propiedad
VMM	(Virtual Machine Monitor), Monitor de Máquina Virtual
VM	(Virtual Machine), Máquina Virtual
IDC	(International Data Corporation), Corporación Internacional de Data
VT	(Virtualization Technology), Tecnología de virtualización
AMD-V	(Advanced Micro Devices – Virtualization)
LVM	(Logical Volume Manager), Gestor de volúmenes lógicos
PV	(Physical Volumen), Volumen Físico
VG	(Volume Group), Grupo de Volumen
LV	(Logical Volumen), Volumen Lógico
FS	(File System), Sistema de ficheros
RHEL	Red Hat Enterprise Linux
POSIX	Portable Operating System Interface; la X viene de UNIX
OVF	(Open Virtualization Format), Formato Abierto de Virtualización.
VDI	(Virtual Disk Image), Imagen de Disco Virtual
RDP	Remote Desktop Protocol

Índice

Índice de Tablas

Índice de Figuras

Índice de Gráficos

Índice de Anexos

CAPÍTULO I

MARCO REFERENCIAL

1.1. ANÁLISIS DE LA INVESTIGACIÓN

1.1.1. Problematización

Los responsables de las tecnologías de la información (TI) están acostumbrados a dedicar un servidor físico a cada tipo de aplicación (Exchange, servidores Web, servidor de aplicaciones, bases de datos, Proxy, etc.) para evitar cualquier conflicto entre las distintas aplicaciones y para asegurar la **escalabilidad** de las mismas. Las nuevas plataformas de hardware de altas capacidades que han ido surgiendo hacen inadecuada la tradicional metodología de servidores dedicados y el consecuente aumento del coste de consumo energético.

Como se muestra en la Figura I.1., Servidores en Racks, trabajando con servidores físicos, se deben colocar en gabinetes rackeables, es indispensable una cantidad considerable de racks y costos asociados para: servidores rackeables, racks adicionales,

acondicionamiento para nuevos racks, cableados, potencia eléctrica y UPS, medidas de contingencia de hardware y software para cada servidor.

Figura I.1. Servidores en Racks

Virtualización es una técnica para esconder las características físicas de los recursos de la computadora de la forma en que otros sistemas operativos (SO), aplicaciones o usuarios finales interactúan con éstos recursos. La virtualización a nivel de SO es una tecnología que virtualiza servidores sobre el canal del sistema operativo (kernel); simula la descomposición de un servidor físico en varias porciones pequeñas. Un servidor virtual es una máquina que crea un entorno virtualizado sobre la plataforma de computadora para que el usuario final pueda operar software en un ambiente controlado. La virtualización de servidores empaqueta el Hardware, Sistema Operativo, Aplicaciones (HW+SO+APP) en un paquete de servidor virtual portable.

El proyecto GNU fue iniciado por Richard Stallman con el objetivo de crear un sistema operativo completamente libre: el sistema GNU. La Fundación para el Software Libre (FSF) es la principal organización que patrocina el proyecto GNU. [1]

[1] http://www.gnu.org/home.es.html

El Gobierno ecuatoriano, el 26 de Abril de 2007, en su mensaje dirigido a 17 países del continente con motivo del Festival Latinoamericano de Instalación de Software Libre (FLISOL), señala: “De esa manera garantizaremos la soberanía de nuestros estados, dependeremos de nuestras propias fuerzas, no de fuerzas externas a la región; seremos productores de tecnología, no simples consumidores; seremos dueños de los códigos fuentes; y podremos desarrollar muchos productos que, incluso, con una adecuada articulación de nuestros esfuerzos, puede ser de suma utilidad para las empresas públicas y privadas de la región.”[2]

La Escuela Superior Politécnica de Chimborazo (ESPOCH) al ser una Institución de Educación Superior pública para la realización de sus actividades y tareas cotidianas conjuntamente con el Departamento de Sistemas y Telemática (DESITEL), mismo que brinda una diversidad de servicios integrales de calidad en las áreas de desarrollo organizacional y sistemas de información, con una infraestructura tecnológica que le permita producir y recibir información actualizada de instituciones relacionadas al que hacer universitario, adopta, el software libre, para sus actividades administrativas, académicas, investigativas y de vinculación con la sociedad.

La Escuela en Ingeniería en Sistemas (EIS), siendo una dependencia de la ESPOCH, se acoge a la utilización del software libre como política propuesta por el DESITEL, para el cumplimiento de sus funciones institucionales.

Existe un conjunto de materias que tienen como objetivo instalar, configurar y administrar servidores, tales como DHCP, Telnet, Correo Electrónico, DNS, FTP, Web, Proxy, en plataformas Linux y Windows.

[2] http://www.presidencia.gov.ec/noticas.asp?noid=9267

Los usuarios para el desarrollo de sus actividades fortaleciendo habilidades y destrezas en el trabajando en plataformas cliente/servidor, podrían contar con prácticas en las que se pueda adquirir una experiencia en entorno real, solicitando servicios a servidores reales sin afectar el buen funcionamiento de los mismos.

Los proyectos de investigación elaborados por los usuarios, algunos son publicados en un servidor de producción, el cual por estar instalado y configurado en un mismo servidor físico son generalmente desconfigurados los servicios que éste brinda, lo que produce la suspensión de sus servicios y empleando horas adicionales para reinstalar y reconfigurar los mismos.

Una solución para estos inconvenientes, sería la adquisición de varios servidores, la cual implicaría una alta inversión económica por el costo de los equipos; la infraestructura física incluyendo los dispositivos de red, consumo de la electricidad; así como la capacitación que deben recibir el personal encargado de los mismos, siendo, estos gastos sólo una parte pequeña del coste total. Y es por esta razón que la virtualización y optimización de recursos es tan importante.

Por investigaciones previas, otra alternativa para solucionar estos inconvenientes es trabajar en ambientes de servidores virtuales, es decir, enfocarse en la virtualización de ordenadores, para crear una infraestructura de servidores virtuales que permita a los usuarios realizar sus actividades, además de crear servidores de producción, respaldos y de investigación, disminuyendo enormemente los costos adquisitivos de servidores reales, reutilizando los existentes.

Esta infraestructura de servidores virtuales será gestionada mediante una aplicación Web denominada “Sistema Administrador Central de Virtualización” (SACV) desarrollada a

través del paradigma orientado a objetos utilizando la metodología ágil Programación Extrema (eXtreme Programming o XP).

1.1.2. Justificación del Proyecto de Tesis

La Virtualización es sin duda un tema muy utilizado en estos momentos y que mejor opción que usar un software de virtualización de ordenadores para realizar pruebas y hasta gestionar directamente servidores sin tener que contar con mayor cantidad de servidores físicos en una misma red, y es una magnifica oportunidad de romper con los mitos y demostrar que es simple configurar servidores virtuales siempre y cuando tengamos las herramientas necesarias y por supuesto el hardware suficiente como para ejecutarlo. Un servidor virtual permite instalar un sistema operativo, dentro de otro sistema operativo.

La EIS para contribuir en el proceso de enseñanza-aprendizaje podría trabajar con herramientas de apoyo académico en plataformas de distribución libre, es decir de uso exclusivamente pedagógico, más no de producción, promoviendo en los usuarios el uso de plataformas bajo Licencia Pública General (General Public License-GPL) porque brindan mucha flexibilidad y proyección a futuro para el desarrollo de productos software en la que se permitan compartir código fuente.

Aplicando la tecnología de la virtualización se aprovechará los servidores existentes, que son de buenas características a nivel hardware, mejorando las medidas de contingencia de hardware y software, es decir dejar de trabajar en ambientes sin virtualización para llegar a funcionar en un ambiente virtual, como se ilustra en la Figura I.2., Virtualización, al funcionar en un mismo servidor físico el software se encuentra atado al hardware, se

tiene una sola imagen de SO por servidor y una aplicación por SO. Mientras que al utilizar virtualización donde el SO y las aplicaciones son independientes del hardware, cada servidor virtual puede proveer sistemas operativos, donde su administración y aplicaciones se maneja como una sola unidad.

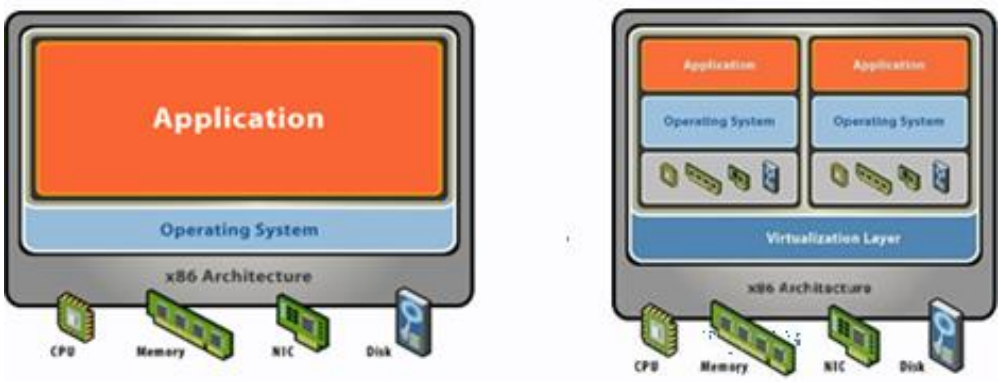

Figura I.2. Virtualización

Para la realización de este proyecto de tesis se plantea efectuar un estudio comparativo sobre la instalación, escalabilidad, alta disponibilidad, flexibilidad, usabilidad, estabilidad, rendimiento, velocidad y extensibilidad de sistemas de virtualización de ordenadores, por software de distribución libre, tales como: Xen que provee el Centos 5.0, VMWare Server y Virtual Box, que trabajan bajo la plataforma Linux, como una solución práctica, efectiva y económica, optimizando los recursos económicos: hardware, humanos, consumo energético y servicios, con los que cuenta la institución, logrando aprovechar en gran magnitud los servidores disponibles en la sala de servidores de la EIS, considerando la filosofía de código abierto, para la no obtención de licencia y acogiéndonos a la política planteada por el DESITEL.

Se ha escogido como sistemas de virtualización a Xen porque cuenta con la licencia GPL de código abierto y está disponible para la descarga de forma gratuita.[3] La decisión de usar la distribución VMWare Server es porque la compañía de Virtualización de servidores Vmware está haciendo una promoción donde ofrece su producto Vmware server gratis. En donde el producto es "Generally Available" y está disponible sin costo para Linux y para Windows.[4] Y VirtualBox porque es una herramienta de virtualización de código abierto con la que se puede ejecutar Linux bajo Windows y viceversa.[5] Existen otros en el mercado pero no es óptimo en su rendimiento y no son de distribución libre.

Uno de los principales argumentos que tiene la EIS para la adopción de software libre es la disminución en el gasto y soberanía tecnológica.

Una vez seleccionado el sistema de virtualización por software de distribución libre que cumpla con los requerimientos en base a los parámetros de optimización de recursos, se procederá a realizar una instalación definitiva bajo la plataforma Linux, para posteriormente crear una infraestructura de servidores virtuales, tal como se muestra en la Figura I.3., Infraestructura de Servidores Virtuales, fortaleciendo el trabajo y gestionando su funcionamiento, mediante la aplicación Web SACV con un entorno amigable.

Reduciendo recursos como networking, electricidad, cableado, simplicidad en su composición física, asignación dinámica de memoria, discos, logrando obtener un respaldo completo a nivel de servidor Host y virtuales adquiriendo un tiempo de recovery mínimo con un control remoto sobre el servidor.

[3] http://www.novell.com/es-es/virtualization/xen_opensource_hypervisor.html
[4] http://sistema01.emnhome.com/blogger/2006/07/vmware-server-gratis.html
[5] http://www.freewarexp.com/descarga_gratis-5455-Virtual_Box_1_3_2.html

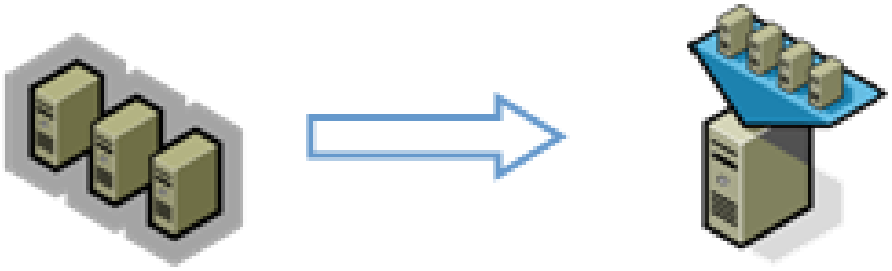

Figura I.3. Infraestructura de Servidores Virtuales

La aplicación Web SACV se convertirá en un gestor de la infraestructura de servidores virtuales y en un instrumento global que brindará servicios de gestión remota de los ordenadores virtuales tales como:

- Inicio y detención de servicio virtual
- Monitoreo de recursos hardware virtual
- Manejo de servidores virtuales
- Funciones especiales

Rompiendo las barreras del espacio, distancia, tiempo, convirtiéndose en una herramienta fundamental para la continuidad de las actividades académicas aportando significativamente en el proceso enseñanza-aprendizaje, permitiendo la comunicación de servidores vía red.

Para el desarrollo de la aplicación Web SACV se utilizará el paradigma orientado a objetos a través de la metodología ligera de desarrollo de software, Programación Extrema (eXtreme Programming o XP), que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo con equipos pequeños de desarrollo. La metodología consiste en una programación

rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

1.1.2.1. Justificación Social

El desarrollo de una infraestructura de servidores virtuales propio de la institución permitirá:

- Realzar la imagen de la Escuela de Ingeniería en Sistemas.
- Proveer a los docentes y estudiantes de la EIS la posibilidad de tener servidores virtuales dinámicos, óptimos, actuales, útiles y de gran interés, para la producción y explotación de aplicaciones.
- Formación de grupos de interés en temas científicos, académicos, culturales, etc.

1.1.2.2. Justificación Técnica

El desarrollo de una infraestrura de servidores virtuales progresa de manera acelerada día a día, así existe gran cantidad de tecnologías que resultan abrumadoras para el administrador de servidores al momento de elegir cuál usar y cómo hacerlo. Con el uso de herramientas de software libre, se obtienen muchos beneficios, uno de ellos la no preocupación por la adquisición de las licencias de uso, es decir se tiene la libertad y flexibilidad de la utilización de software libre.

El estudio comparativo de los sistemas de virtualización de ordenadores, va a proyectar resultados que nos ayudarán a obtener las características acordes a los recursos tecnológicos con que cuenta la EIS. Estas características se verán reflejadas en el desarrollo de la infraestructura de servidores virtuales, así como también en los servicios que prestará el SACV integrado sobre la Plataforma Linux.

La aplicación web se desarrollará en la Escuela de Ingeniería en Sistemas de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo de la ciudad de Riobamba y los beneficios que se obtienen con la implementación de esta aplicación, van dirigidos a sumar prestaciones de administración de servicio.

El Sistema Administrador Central de Virtualización cubrirá las siguientes características:

- Servicios básicos de la aplicación Web SACV de Sistema Administrador Central de Virtualización: Creación de Cuentas de administración, Envío y Recepción de los detalles de los servidores virtuales, Métodos de revisión de los servidores virtuales, Configuraciones Personalizadas.
- Seguridad (mediante el ingreso de claves y usuario).
- Posibilidad de manipulación de cambio de contraseñas como aporte al nivel de seguridad.

En fin, una solución de administración y colaboración muy sólida, accesible y amigable, mediante soporte técnico con formato http.

1.1.3. Objetivos

1.1.3.1. Objetivo General

Estudiar comparativamente sistemas de virtualización de ordenadores por software, de distribución libre, para desarrollar una infraestructura de servidores virtuales gestionados mediante una aplicación Web en la EIS.

1.1.3.2. Objetivos Específicos

- Investigar conceptos y teoría de sistemas de virtualización de ordenadores.
- Comparar sistemas de virtualización de ordenadores, por software de distribución libre, para seleccionar el que mejor se ajuste a los requerimientos de la EIS.
- Desarrollar una infraestructura de servidores virtuales, con fines académicos, en la EIS que permita optimizar los recursos disponibles.
- Implementar una aplicación Web que permita gestionar el funcionamiento de Servidores Virtuales.

1.1.4. Hipótesis

El estudio comparativo entre sistemas de virtualización de ordenadores, por software de distribución libre, permitirá seleccionar el mejor para optimizar los recursos hardware.

CAPÍTULO II

MARCO TEÓRICO

2.1. INTRODUCCIÓN

En este capítulo se explicará los conceptos básicos y términos explicativos fundamentales para el desarrollo de este tema investigativo. Su finalidad es tener en claro los fundamentos teóricos que se necesita conocer sobre la virtualización de ordenadores.

Veremos entre otros, el enfoque, propiedades importantes, consolidación, contención de la infraestructura virtual que se usa para virtualizar ordenadores, así como también gráficos que ilustraran su funcionamiento. Sin duda, un elemento clave en esta sociedad de la informática ha sido la virtualización de ordenadores que, aprovechándose de los avances ofrecidos por la TI y las comunicaciones, permite que en un sólo ordenador físico se pueda tener varios servidores virtuales, a unos costes despreciables y que

cualquier persona o grupo de personas puedan administrar los servicios que éstos brinden.

Las empresas fueron pioneras en el uso de la virtualización de ordenadores, por lo que hoy en día la gran mayoría de las universidades, sobre todo el personal docente, estudiantil e investigador, lo usan para optimizar todo tipo de hardware. Así, por ejemplo, administradores que trabajan en dataceneters con muchos servidores con diferentes aplicativos pueden gestinar sus servicios y optimizar procesos, sin necesidad de desplazarse al sitio donde se alojan los servidores físicos o de utilizar otros métodos tradicionales mucho más costosos.

2.2. GENERALIDADES

2.2.1. Definición de la Virtualización

La virtualización permite que múltiples máquinas virtuales con sistemas operativos heterogéneos puedan ejecutarse individualmente, aunque en la misma máquina. Cada máquina virtual tiene su propio hardware virtual (por ejemplo, RAM, CPU, NIC, etc.) a través del cual se cargan el sistema operativo y las aplicaciones. El sistema operativo distingue al hardware como un conjunto normalizado y consistente, independientemente de los componentes físicos que realmente formen parte del mismo.

Virtualización también puede significar conseguir que varios ordenadores parezcan uno solo. A este concepto se le suele denominar agregación de servidores (server aggregation) o grid computing.

La virtualización del sistema operativo es el uso de software para permitir que un mismo sistema maneje varias imágenes de los sistemas operativos a la misma vez. Esta tecnología permite la separación del hardware y el software, lo cual posibilita a su vez que múltiples sistemas operativos se ejecuten simultáneamente en una sola computadora.

La virtualización, desde un punto de vista muy simple es un programa que se instala en un sistema operativo (llamado anfitrión) que permite instalar y ejecutar otro sistema operativo como si fuera otro ordenador completamente diferente, llamado **servidor virtual**.

Este término es bastante antiguo: viene siendo usado desde antes de 1960, para permitir la división de grandes unidades de hardware mainframe, un recurso costoso y escaso; y ha sido aplicado a diferentes aspectos y ámbitos de la informática, desde sistemas computacionales completos hasta capacidades o componentes individuales. Con el tiempo, las minicomputadoras y computadores personales (PCs) proporcionaron una manera más eficiente y asequible de distribuir el poder de procesamiento, por lo que en los años 80, la virtualización ya casi no se utilizó más. En los años 90, los investigadores comenzaron a ver cómo la virtualización podía solucionar algunos de los problemas relacionados con la proliferación de hardware menos costoso, incluyendo su subutilización, crecientes costos de administración y vulnerabilidad.

La virtualización no es un tema nuevo, de hecho ronda desde hace 40 años. Los primeros usos de la virtualización incluyen el IBM 7044, el Sistema de Tiempo Compartido (CTSS - Compatible Time Sharing System) desarrollado en el Instituto Tecnológico de Massachusetts (MIT - Massachussets Institute of Technology) en el IBM 704. Y el proyecto Atlas de la Universidad de Manchester (uno de los primeros superordenadores

del mundo), que fué pionero en el uso de memoria virtual con paginación y llamadas de supervisor.

Hoy en día, la virtualización está a la vanguardia, ayudando a los negocios con la escalabilidad, seguridad y administración de sus infraestructuras globales de TI.

2.2.1.1. La importancia de la virtualización

Desde una perspectiva de negocio, hay muchas razones para utilizar virtualización. La mayoría están relacionadas con la consolidación de servidores. Simple, si podemos virtualizar un número de sistemas infrautilizados en un solo servidor, ahorrando energía, espacio, capacidad de refrigeración y administración ya que se tiene menos servidores.

Como puede ser difícil determinar el grado de utilización de un servidor, las tecnologías de virtualización soportan la migración en directo. La migración en directo permite que un sistema operativo y sus aplicaciones se muevan a un nuevo servidor para balancear la carga sobre el hardware disponible.

La virtualización también es importante para los desarrolladores. El núcleo Linux ocupa un solo espacio de direcciones, lo que significa que un fallo en el núcleo o en cualquier driver provoca la caída del sistema operativo completo. La virtualización supone que puedes ejecutar varios sistemas operativos, y si uno cae debido a un fallo, el hipervisor y el resto de sistemas operativos continuarán funcionando. Esto puede hacer que depurar el núcleo sea una tarea más parecida a depurar aplicaciones en el espacio del usuario.

- **División**

Se pueden ejecutar múltiples aplicaciones y sistemas operativos en un mismo sistema físico. Los servidores se pueden consolidar en ordenadores virtuales con una arquitectura de escalabilidad vertical (scale-up) u horizontal (scale-out).

Los recursos computacionales se tratan como un conjunto uniforme que se distribuye entre los ordenadores virtuales de manera controlada.

- **Aislamiento**

Los ordenadores virtuales están completamente aislados entre sí y del ordenador host. Si existen fallas en un ordeandor virtual, las demás no se ven afectados. Los datos no se filtran a través de los ordenadores virtuales y las aplicaciones sólo se pueden comunicar a través de conexiones de red configuradas.

- **Encapsulación**

El entorno completo del servidor virtual se guarda en un solo archivo, fácil de mover, copiar y resguardar. La aplicación reconoce el hardware virtual estandarizado de manera que se garantiza su compatibilidad.

2.2.1.2. Factores a considerar para la virtrualización

En las tecnologías de virtualización se consideran como factores a: reducción de costes, mejora el retorno de las inversiones de las TI casi inmediato, uso racional del hardware,

mayor flexibilidad, reducción de gastos operativos, reducción en el consumo de energía, mayor eficiencia de los recursos informáticos, una gestión y administración de los recursos más ágil y centralizada, aumenta la capacidad de los servidores entre un 16 y un 80 por ciento dependiendo de las caracteríticas técnicas de hardware.

2.2.1.3. Ventajas y desventajas de la virtualización

2.2.1.3.1. Ventas de la Virtualización

Razones más importantes para adoptar software de virtualización

- **Consolidación de servidores y optimización de infraestructuras:** la virtualización permite lograr una utilización de los recursos significativamente mayor mediante la agrupación de recursos de infraestructura comunes y la superación del modelo heredado de “una aplicación para un servidor”.

- **Reducción de costes de infraestructura física**: con la virtualización, podemos reducir la cantidad de servidores y hardware inherente al datacenter. Esto lleva a disminuir los requisitos inmobiliarios, de alimentación y refrigeración, con la consiguiente e importante disminución de los costes de TI.

- **Flexibilidad operativa mejorada y capacidad de respuesta:** la virtualización brinda una nueva forma de gestionar la infraestructura de TI y ayuda a los administradores de TI a dedicarle menos tiempo a tareas repetitivas tales como provisioning, configuración, supervisión y mantenimiento.

- **Mayor disponibilidad de aplicaciones y continuidad del negocio mejorada:** elimina las paradas planificadas y efectúa una recuperación rápida de los cortes imprevistos de suministro eléctrico con la capacidad de realizar backup de forma segura y migrar la totalidad de los entornos virtuales sin interrupción del servicio.

- **Capacidad de gestión y seguridad mejorada:** implementar, administrar y supervisar entornos de escritorio protegidos a los que los usuarios puedan acceder localmente o de forma remota, con o sin conexión a red, desde casi cualquier ordenador de escritorio, portátil o tablet PC.

2.2.1.3.2. Desventajas de la Virtualización

No todo son ventajas, también hay que tener en cuenta algunos detalles que pueden ser vistos negativamente:

1. Si se daña el disco duro, se nos dañarán todas las máquinas. Sugirimos uso del RAID, los discos no se dañan siempre, pero a veces pasa.

2. Si nos roban la máquina, nos roban todas las máquinas virtuales. Sugerimos realizar respaldos.

3. En fin, cualquier evento que ocurra con el hardware, afectará a todas las máquinas virtuales (corriente, red, etc) asi que necesitamos un sistema bien redundante (doble red, doble disco, doble fuente de corriente, etc).

Más que contrar son elementos que deben dimensionar adecuadamente para evitar que nos suceda. Si nos sucede es porque no pensamos en el antes de instalarlo, no es culpa de la máquina virtual.

2.2.1.3.3. Infraestructura sin Virtualización

- **Baja utilización de la infraestructura:** Las implementaciones típicas de servidores x86 logran una utilización media de entre un 10% y un 15% de la capacidad total, según señala International Data Corporation (IDC). Normalmente, las organizaciones ejecutan una aplicación por servidor para evitar el riesgo de que las vulnerabilidades de una aplicación afecten a la disponibilidad de otra aplicación en el mismo servidor.

- **Incremento de los costes de infraestructura física:** Los costes operativos para dar soporte al crecimiento de infraestructuras físicas han aumentado a ritmo constante. La mayor parte de las infraestructuras informáticas deben permanecer operativas en todo momento, lo que genera gastos en consumo energético, refrigeración e instalaciones que no varían con los niveles de utilización.

- **Incremento de los costes de gestión de TI:** 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 costes 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.

- **Insuficiente failover y protección ante desastres:** Las empresas se ven cada vez más afectadas por las paradas de las aplicaciones de servidor crítico y la falta de

acceso a escritorios de usuario final. La amenaza de ataques a la seguridad o desastres naturales, han acentuado la importancia de la planificación de la continuidad del negocio tanto en lo relativo a escritorios como a servidores.

- **Escritorios de usuario final de mantenimiento elevado:** La gestión y la seguridad de los escritorios corporativos plantean numerosos desafíos. Controlar un entorno de escritorio distribuido 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 en el entorno del escritorio para eliminar las vulnerabilidades de seguridad.

2.2.2. Técnicas de virtualización

Para la realización de este proyecto de tesis se trabajó con la virtualización de plataforma que es una forma de crear servidores virtuales para que dentro de ella corra un determinado sistema operativo.

Ésta máquina virtual puede o no recibir ayuda por parte del hardware en que la corremos.

La virtualización se divide en sí en dos formas o tipos:

- **Virtualización de plataforma** que involucra la simulación de máquinas virtuales.
- **Virtualización de recursos** que involucra la simulación de recursos combinados, fragmentados o simples.

En el caso específico nuestro veremos la virtualización basada en los procesadores i386 (Intel o AMD).

La virtualización de la plataforma, se realiza mediante la intervención de un sistema operativo hospedero (host) el cuál se hará cargo de crear un sistema simulado para los huéspedes (Guests). Estos guests correrán tal y como si lo estuvieran haciendo sobre un hardware verdadero. Normalmente se les simula por parte del host una máquina de forma tal que los guests puedan correr. Sin embargo a veces no se les simula la máquina sino que se les modifica el kernel para que puedan correr de forma compartida en el host.

2.2.2.1. Virtualización de plataforma

El sentido original del término virtualización, nacido en 1960, es el de la creación de una máquina virtual utilizando una combinación de hardware y software llamado virtualización de plataforma. El término máquina virtual aparentemente tiene su origen en el experimento del sistema de paginación (paging system) de IBM. La creación y administración de las máquinas virtuales también se refiere a la creación de seudo máquinas y de virtualización de servidores más recientemente. Los términos virtualización y máquina virtual han adquirido, a través de los años, significados adicionales.

La virtualización de plataforma es llevada a cabo en una plataforma de hardware mediante un software “host” (“anfitrión”, un programa de control) que simula un entorno computacional (máquina virtual) para su software “guest”. Este software “guest”, que generalmente es un sistema operativo completo, corre como si estuviera instalado en una plataforma de hardware autónoma. Típicamente muchas máquinas virtuales son simuladas en una máquina física dada. Para que el sistema operativo “guest” funcione, la simulación debe ser lo suficientemente robusta como para soportar todas las interfaces

externas de los sistemas guest, las cuales pueden incluir (dependiendo del tipo de virtualización) los drivers de hardware.

IBM reconoció la importancia de la virtualización en la década de 1960 con el desarrollo del mainframe System/360 Model 67. El Model 67 virtualizó todas las interfaces hardware a través del Monitor de Máquina Virtual (VMM - Virtual Machine Monitor). En los primeros días de la computación, el sistema operativo se llamó supervisor. Con la habilidad de ejecutar sistemas operativos sobre otro sistema operativo, apareció el termino hypervisor (termino acuñado en la década de 1970).

El VMM se ejecutaba directamente sobre el hardware subyacente, permitiendo múltiples máquinas virtuales (VMs). Cada VM podía ejecutar una instancia de su propio sistema operativo privado al comienzo este era CMS, o Conversational Monitor System. Las máquinas virtuales han continuado avanzando, y hoy se pueden encontrar ejecutándose en el mainframe System z9. Lo que proporciona compatibilidad hacia atrás, incluso hasta la línea System/360.

2.2.2.2. Virtualización de los recursos

Se extendió a la virtualización de recursos específicos del sistema como la capacidad de almacenamiento, nombre de los espacios y recursos de la red.

Los términos resource aggregation, spanning o concatenation (name spaces) se utiliza cuando se combinan componentes individuales en un mayor recurso o en un recurso de uso común (resource pools). Por ejemplo:

- Conjunto Redundante de Discos Independientes (**RAID** - Redundant Array of Inexpensive Disks) y volume managers combinan muchos discos en un gran disco lógico.

- **La virtualización de almacenamiento** (Storage virtualization) refiere al proceso de abstraer el almacenamiento lógico del almacenamiento físico, y es comúnmente usado en SANs (Storage Area Network).

- Channel bonding y el equipamiento de red utilizan para trabajar múltiples enlaces combinados mientras ofrecen un enlace único y con mayor amplitud de banda.

- **Red privada virtual** (en inglés Virtual Private Network, VPN), Traducción de dirección de red (en inglés Network Address Translation, NAT) y tecnologías de red similares crean una red virtual dentro o a través de subredes.

- Sistemas de computación multiprocessor y multi-core muchas veces presentan lo que aparece como un procesador único, rápido e independiente.

- **Cluster, grid** computing y servidores virtuales usan las tecnologías anteriormente mencionadas para combinar múltiples y diferentes computadoras en una gran metacomputadora.

- Particionamiento es la división de un solo recurso (generalmente grande), como en espacio de disco o ancho de banda de la red, en un número más pequeño y con recursos del mismo tipo más fáciles de utilizar.

- Encapsulación es el ocultamiento de los recursos complejos mediante la creación de un interfaz simple. Por ejemplo, muchas veces CPUs incorporan memoria caché o segmentación (pipeline) para mejorar el rendimiento, pero estos elementos no son reflejados en su interfaz virtual externa. Interfaces virtuales similares que ocultan implementaciones complejas se encuentran en los discos, módems, routers y otros dispositivos “inteligentes” (smart).

2.2.2.3. Aplicación de la Virtualización por plataforma

Existen muchos enfoques a la virtualización de plataformas, aquí se listan con base en cuan completamente es implementada una simulación de hardware:

- **Emulación o simulación:** la máquina virtual simula un hardware completo, admitiendo un sistema operativo “guest” sin modificar para una CPU completamente diferente. Este enfoque fue muy utilizado para permitir la creación de software para nuevos procesadores antes de que estuvieran físicamente disponibles. Por ejemplo: Qemu, Virtual PC, Bochs y PearPC.

  La virtualización más compleja consiste en la emulación de hardware. Con esta técnica, en el sistema anfitrión se utiliza una máquina virtual que emula el hardware, como muestra la Figura II.4.

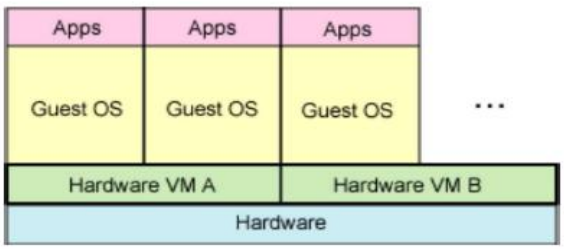

Figura II.4. La emulación de Hardware utiliza un servidor virtual para simular el hardware

- **Virtualización nativa y virtualización completa**: la máquina virtual simula un hardware suficiente para permitir un sistema operativo “guest” sin modificar (uno diseñado para la misma CPU) para correr de forma aislada. Típicamente, muchas instancias pueden correr al mismo tiempo. Este enfoque fue el pionero en 1966, predecesores de la familia de máquinas virtuales de IBM. Algunos ejemplos: VMware Workstation, VMware Server, Parallels Desktop, Adeos, Mac-on-Linux, Win4BSD, Win4Lin Pro y z/VM.

  El host emula lo suficientemente bien el hardware como para que los guests puedan correr de forma nativa (sin cambios en el kernel) y además de forma completamente aislada.

  La virtualización completa, también llamada virtualización nativa, es otra interesante técnica de virtualización. Este modelo utiliza una máquina virtual que media entre el sistema operativo invitado y el hardware nativo (ver Figura II.5). "Mediar" es la palabra clave aquí porque la VMM está entre el sistema, el sistema operativo invitado y el hardware real. Algunas instrucciones protegidas deben capturarse y manejarse dentro del hipervisor ya que el hardware subyacente no es propiedad de un sistema operativo sino que es compartido a través del hipervisor.

  Estos guests normalmente pueden correr varios en la misma máquina y compartir eficientemente sus recursos. Se considera una emulación un poco más avanzado. El caso más conocido es el VMWare.

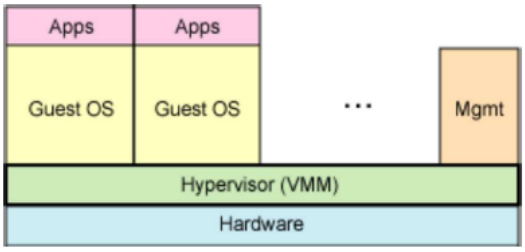

Figura II.5. La virtualiazación completa utiliza un hipervisor para compartir el hardware subyacente

- **Paravirtualización:** la máquina virtual no necesariamente simula un hardware, en cambio ofrece un Interfaz de Programación de Aplicaciones (API - Application Programming Interface) especial que solo puede usarse mediante la modificación del sistema operativo “guest”. La llamada del sistema al hypervisor tiene el nombre de “hypercall” en Xen y Parallels Workstation.

  La paravirtualización es otra técnica popular que cuenta con algunas similitudes con la virtualización completa. Este método utiliza un hipervisor para compartir el acceso al hardware subyacente pero integra código que está al tanto de la virtualización en el propio sistema operativo (ver Figura II.6). Esta aproximación evita la necesidad de recompilar y capturar ya que los propios sistemas operativos cooperan en el proceso de virtualización.

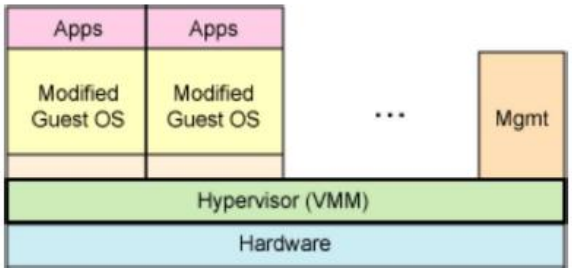

Figura II.6. La paravirtualización comparte el proceso con el SO alojado (Guest OS)

La paravirtualización precisa que los sistemas operativos alojados sean modificados por el hipervisor, lo que es una desventaja. Pero la paravirtualización ofrece un rendimiento próximo al de un sistema no virtualizado. Del mismo modo que con la virtualización completa, es posible soportar varios sistemas operativos diferentes de manera concurrente.

El kernel de los guests tiene que ser modificado para permitir acceder al API del host y poder manejar y acceder a los recursos del host (disco, red, usb, etc).

Ejemplos de paravirtualización incluyen el VMWare ESX Server y el XEN.

- **Virtualización en el nivel del sistema operativo:** La última técnica que exploraremos, la virtualización en el nivel del sistema operativo, utiliza una técnica diferente a las que hemos visto. Esta técnica virtualiza los servidores encima del propio sistema operativo. Este método soporta un solo sistema operativo y símplemente aisla los servidores independientes (ver Figura II.7).

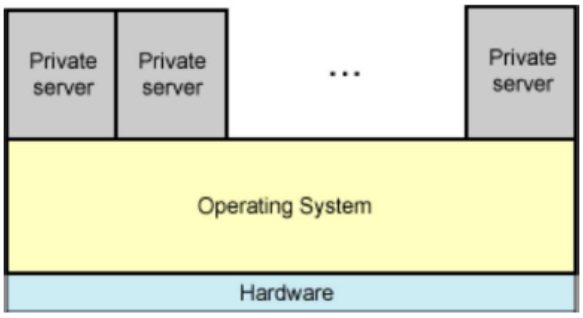

Figura II.7. La virtualización en el nivel del sistema operativo aisla a los servidores

La virtualización en el nivel del sistema operativo requiere cambios en el núcleo del sistema operativo, la ventaja es un rendimiento igual a la ejecución nativa.

2.2.2.4. Virtualización relacionada con Linux

La Tabla muestra diferentes posibilidades de virtualización en Linux, centrándose en aquellas soluciones de código abierto.

Tabla II.1. Proyectos de virtualización relacionados con Linux

Proyecto	Tipo	Licencia
Bochs	Emulación	LGPL
QEMU	Emulación	LGPL/GPL
VMware	Virtualización completa	Privativa
z/VM	Virtualización completa	Privativa
Xen	Paravirtualización	GPL
UML	Paravirtualización	GPL
Linux-VServer	Virtualización en el nivel del sistema operativo	GPL

- **Bochs (emulación): Virtualización a nivel de biblioteca**

Aunque aquí no se haya tratado, otro método de virtualización que emula porciones de un sistema operativo a través de una biblioteca es la virtualización a nivel de biblioteca. Algunos ejemplos son Wine (parte de la API Win32 para Linux) y LxRun (parte de la API Linux para Solaris).

Bochs simula un ordenador x86, es portable y se ejecuta sobre diferentes plataformas, incluyendo x86, PowerPC, Alpha, SPARC y MIPS. El interés de Bochs es que no solo emula el procesador sino el ordenador entero, incluyendo los periféricos, como el teclado, ratón, hardware gráfico, adaptadores de red, etc.

Bochs puede configurarse como un antiguo Intel 386, o sucesores como el 486, Pentium, Pentium Pro, o una variante de 64 bits. Incluso emula instrucciones gráficas opcionales como MMX y 3DNow.

Utilizando el emulador Bochs, podemos ejecutar cualquier distribución Linux en Linux, Microsoft Windows 95/98/NT/2000 (y una variedad de aplicaciones) en Linux, incluso los sistemas operativos BSD (FreeBSD, OpenBSD, etc...) sobre Linux.

- **QEMU (emulación)**

QEMU es otro emulador, como Bochs, pero tiene algunas diferencias que son bienvenidas. QEMU soporta dos modos de operación. El primero es el modo de emulación de sistema completo. Este modo es similar a Bochs ya que emula todo un ordenador personal (PC) con su procesador y periféricos. Este modo emula varias arquitecturas, como x86, x86_64, ARM, SPARC, PowerPC y MIPS, con velocidad razonable utilizando traducción dinámica. Utilizando este modo es posible emular los sistemas operativos Windows (incluyendo XP) y Linux sobre Linux, Solaris y FreeBSD. También se soportan otras combinaciones de sistemas operativos.

QEMU también soporta un segundo modo llamado User Mode Emulation. En este modo, que sólo puede ser alojado en Linux, puede lanzarse un binario para una arquitectura diferente. Esto permite, por ejemplo, que se ejecute en Linux sobre x86 un binario compilado para la arquitectura MIPS (Microprocessor without Interlocked Pipeline Stages). Entre las arquitecturas soportadas por este modo se encuentran ARM, SPARC, PowerPC y otras que están en desarrollo.

- **VMware (virtualización completa)**

VMware es una solución comercial para la virtualización completa. Entre los sistemas operativos alojados y el hardware existe un hipervisor funcionando como capa de

abstracción. Esta capa de abstracción permite que cualquier sistema operativo se ejecute sobre el hardware sin ningún conocimiento de cualquier otro sistema operativo alojado.

VMware también virtualiza el hardware de entrada/salida disponible y ubica drivers para dispositivos de alto rendimiento en el hipervisor.

El entorno virtualizado completo se respalda en un fichero, lo que significa que un sistema completo (incluyendo el sistema operativo alojado, la máquina virtual y el hardware virtual) puede migrarse con facilidad y rapidez a una nueva máquina anfitrión para balancear la carga.

- **z/VM (virtualización completa)**

Aunque el IBM System z estrena nombre, realmente tiene una larga historia que se origina en la decada de 1960. El System/360 ya soportaba virtualización utilizando máquinas virtuales en 1965.

Es interesante observar que el System z mantiene la retrocompatibilidad hasta la antigua línea System/360.

En el System z, se utiliza como hipervisor del sistema operativo a z/VM. En su interior está el Programa de Control (CP - Control Program), que proporciona la virtualización de los recursos físicos a los sistemas operativos alojados, incluyendo Linux (ver Figura II.8).

Esto permite que varios procesadores y otros recursos sean virtualizados para un número de sistemas operativos alojado.

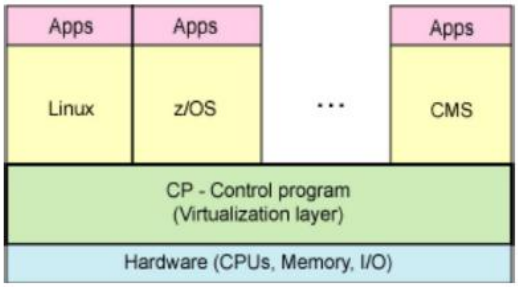

Figura II.8. Virtualización a nivel de SO utilizando z/VM

z/VM también puede emular una LAN virtual para aquellos sistemas operativos hospedados que quieren comunicarse entre sí. La emulación se realiza por completo en el hipervisor, con lo que se obtiene una gran seguridad.

- **Xen (paravirtualización)**

Xen es la solución de fuente abierta proporcionada por XenSource para obtener paravirtualización a nivel de sistema operativo. Recuerde que en la paravirtualización el hipervisor y el sistema operativo colaboran en la virtualización, se requieren cambios en el sistema operativo pero se obtiene un rendimiento próximo a la ejecución nativa.

Como Xen precisa colaboración (modificaciones en el sistema operativo alojado), solo pueden virtualizarse en Xen sistemas operativos parcheados. Desde el punto de vista de Linux, que es de fuente abierta, se trata de un compromiso razonable porque se consigue un mejor rendimiento que con la virtualización completa. Pero desde el punto de vista de un soporte amplio (que incluya otros sistemas operativos que no sean de fuente abierta), se trata de un claro inconveniente.

Es posible ejecutar Windows como SO alojado en Xen, pero solo en sistemas hardware que soporten la tecnología Vanderpool de Intel o Pacifica de AMD. Otros sistemas operativos soportados por Xen son: Minix, Plan 9, NetBSD, FreeBSD y OpenSolaris.

- **User-mode Linux (paravirtualización)**

User-mode Linux (UML) permite que un sistema operativo Linux ejecute otros sistemas operativos Linux en el espacio del usuario. Cada sistema operativo Linux alojado existe como un proceso en el sistema operativo Linux anfitrión (ver Figura II.9). Lo que permite a varios núcleos Linux (con sus propios espacios de usuario asociados) ejecutarse en el contexto de un solo núcleo Linux.

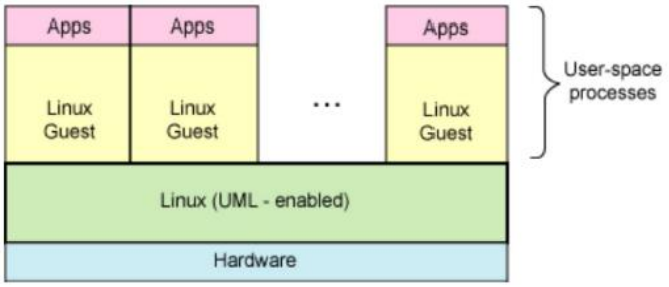

Figura II.9. Alojamiento de Linux en User-mode Linux

Desde el núcleo Linux 2.6, UML se encuentra en la rama principal del núcleo, pero debe ser activado y recompilado antes de utilizarse. Estos cambios proporcionan, entre otras cosas, virtualización de dispositivos. Lo que permite a los sistemas operativos alojados compartir los dispositivos físicos disponibles, como los dispositivos de bloques (floppy, CD-ROM, y sistemas de ficheros), consolas, dispositivos NIC, hardware de sonido y otros.

Puesto que los núcleos alojados se ejecutan en el espacio del usuario deben estar compilados para este uso (aunque puede tratarse de diferentes versiones del núcleo).

Existirá un núcleo anfitrión (que se ejecutará sobre el hardware) y uno o varios núcleos alojados (que se ejecutarán en el espacio de usuario del núcleo anfitrión). Es posible anidar estos núcleos, de manera que un núcleo alojado actúe como anfitrión de otro.

- **Linux-VServer (virtualización a nivel de sistema operativo)**

Linux-VServer es una solución de virtualización a nivel de sistema operativo. Linux-VServer virtualiza el núcleo Linux de manera que varios entornos de espacio de usuario, también llamados Virtual Private Servers (VPS), se ejecutan de forma independiente sin tener conocimiento del resto.

El aislamiento del espacio de usuario se consigue gracias a diferentes modificaciones del núcleo Linux.

Para aislar cada uno de los espacios de usuario del resto hay que estudiar el concepto de un contexto. Un contexto es un contenedor para los procesos de un VPS, de manera que herramientas como ps solo muestran información sobre los procesos del VPS. Para el arranque inicial el núcleo define un contexto por defecto. También existe un contexto espectador para la administración. Como puede suponer, tanto el núcleo como las estructuras internas de datos se han modificado para dar soporte a esta técnica de virtualización.

Con Linux-VServer también se utiliza un tipo de chroot para aislar el directorio raíz de cada VPS. Recordemos que una chroot permite que se especifique un nuevo directorio raíz, además se utilizan otras funciones (llamadas Chroot-Barrier) para que un VPS no pueda escapar desde su confinamiento en el directorio raíz. Cada VPS cuenta con su propia raíz y lista de usuarios y contraseñas.

Linux-VServer está soportado en los núcleos Linux v2.4 y v2.6, pudiendo funcionar sobre diferentes plataformas: x86, x86-64, SPARC, MIPS, ARM y PowerPC.

- **OpenVZ (virtualización a nivel de sistema operativo)**

OpenVZ es otra solución de virtualización a nivel de sistema operativo, como Linux-VServer, pero tiene algunas diferencias interesantes. OpenVZ es un núcleo modificado para la virtualización que soporta espacios de usuario aislados, VPS, con un conjunto de herramientas de usuario para la administración.

Para planificar los procesos, OpenVZ utiliza un planificador de dos niveles. Primero se determina qué VPS debe obtener la CPU. Después, el segundo nivel del planificador escoge el proceso a ejecutar basándose en las prioridades standard de Linux.

OpenVZ también incluye los llamados beancounters. Un beancounter consiste en un número de parámetros que definen la distribución de recursos para un VPS. Esto proporciona cierto nivel de control sobre un VPS, definiendo la cantidad de memoria y el número de objetos para la comunicación entre procesos (IPC) disponibles.

Una característica única de OpenVZ es la habilidad de establecer un punto de control y migrar un VPS desde un servidor físico a otro. Establecer un punto de control significa que el estado de un VPS en ejecución se congela y se guarda en un fichero. Este fichero puede llevarse a un nuevo servidor para restaurar la ejecución del VPS.

Entre las arquitecturas soportadas por OpenVZ se encuentran: x86, x86-64 y PowerPC.

2.2.2.5. Utilidades de los servidores virtuales

- **Soporte hardware para la virtualización completa y la paravirtualización**

La arquitectura IA-32 (x86) crea ciertos problemas cuando se intenta virtualizar.
Algunas instrucciones del modo privilegiado no se pueden capturar y pueden devolver diferentes resultados en función del modo. Por ejemplo, la instrucción STR recupera el estado de seguridad, pero el valor que retorna depende del nivel de privilegios de quien realizó la ejecución. Lo que es problemático cuando se intenta virtualizar diferentes sistemas operativos en diferentes niveles.

Por ejemplo, la arquitectura x86 soporta cuatro anillos de protección, el nivel 0 (mayor privilegio) normalmente ejecuta el sistema operativo, los niveles 1 y 2 dan soporte a los servicios del sistema operativo, y el nivel 3 (el menor de los privilegios) soporta las aplicaciones. Los fabricantes de hardware han detectado este defecto (y otros), y han producido nuevos diseños que soportan y aceleran la virtualización.

Intel está produciendo una nueva tecnología de virtualización que soportará hipervisores en dos de sus arquitecturas, tanto en x86 (VT-x) como en Itanium (VT-i). VT-x soporta dos nuevos modos de operación, uno para la VMM (root) y otro para los sistemas operativos hospedados (no root). En el modo root se cuentan con todos los privilegios, mientras que en el modo no root no se tienen privilegios (incluso para el nivel 0). La arquitectura también permite cierta flexibilidad al definir las instrucciones que provocan que una VM (sistema operativo hospedado) retorne al VMM y almacene el estado del procesador. También se han añadido otras capacidades.

AMD está produciendo la tecnología Pacifica en la que el hardware asiste a la virtualización. Entre otras cosas, Pacifica mantiene un bloque de control para los sistemas operativos hospedados que se guarda con la ejecución de instrucciones especiales. La instrucción VMRUN permite a una máquina virtual (y sus sistema operativo hospedado asociado) ejecutarse hasta que el VMM recupere el control (lo que también es configurable). Las opciones de configuración permiten que el VMM adapte los privilegios de cada uno de los huéspedes. Pacifica también compensa la traducción de direcciones con unidades de gestión de memoria (MMU) para el anfitrión y los huéspedes.

Estas nuevas tecnologías pueden utilizarse en varias de las técnicas de virtualización que se han discutido, como Xen, VMware, User-mode Linux y otras.

- **Linux KVM (Kernel Virtual Machine)**

Las noticias más recientes que provienen de Linux son la incorporación de KVM en el núcleo (2.6.20). KVM es una completa solución de virtualización única al convertir al núcleo Linux en un hipervisor utilizando un módulo del núcleo. Este módulo permite a otros sistemas operativos alojados ejecutarse en el espacio de usuario del núcleo Linux anfitrión (ver Figura II.10). El módulo KVM en el núcleo expone el hardware virtualizado a través del dispositivo de carácteres /dev/kvm. El sistema operativo alojado se comunica con el módulo KVM utilizando un proceso que ejecuta un QEMU modificado para obtener la emulación de hardware.

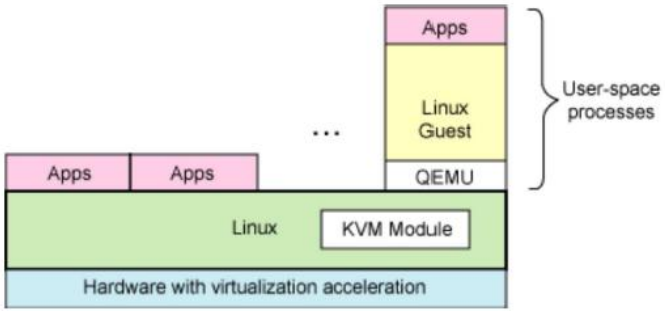

Figura II.10. Virtualización con Kernel Virtual Machine (KVM)

El módulo KVM introduce un nuevo modo de ejecución en el núcleo. Donde el kernel vanilla (standard) aporta el modo kernel y el modo user, KVM aporta el modo guest. Este modo es utilizado para ejecutar todo el código del huésped en el que no se utiliza entrada/salida, y el modo normal de usuario proporciona la entrada/salida para los huéspedes.

La presentación de KVM es una interesante evolución de Linux, ya que es la primera tecnología de virtualización que pasa a formar parte del propio núcleo Linux. Existe en la rama 2.6.20, pero puede utilizarse como un módulo del núcleo en la versión 2.6.19.

Cuando se ejecuta en hardware que soporta la virtualización es posible hospedar a Linux (32 y 64 bits) y Windows (32 bits).

2.2.3. Conceptos Fundamentales sobre Software

2.2.3.1. Definición de software

Probablemente la definición más formal de software es la atribuida a la IEEE (Instituto de Ingenieros Eléctricos y Electrónicos), en su estándar 729: *la suma total de los programas*

de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo. Bajo esta definición el concepto de software va más allá de los programas de cómputo en sus distintas formas: código fuente, binario o código ejecutable, además de su documentación. Es decir, *el software es todo lo intangible.*

2.2.3.2. Definición de software libre

El software libre es aquel que puede ser *distribuido, modificado, copiado y usado*; por lo tanto, debe venir acompañado del código fuente para hacer efectivas las libertades que lo caracterizan. Dentro de software libre hay, a su vez, matices que es necesario tener en cuenta. Por ejemplo, el software de dominio público significa que no está protegido por el copyright, por lo tanto, podrían generarse versiones no libres del mismo, en cambio el software libre protegido con copyleft impide a los redistribuidores incluir algún tipo de restricción a las libertades propias del software así concebido, es decir, garantiza que las modificaciones seguirán siendo software libre.

También es conveniente no confundir el software libre con el software gratuito, éste no cuesta nada, hecho que no lo convierte en software libre, porque no es una cuestión de precio, sino de libertad. Para Richard Stallman el software libre es una cuestión de libertad, no de precio. Para comprender este concepto, debemos pensar en la acepción de libre como en “libertad de expresión”.

En términos del citado autor el software libre se refiere a la libertad de los usuarios para *ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software*. Y se refiere especialmente a cuatro clases de libertad para los usuarios de software:

1. **Libertad 0:** la libertad para ejecutar el programa sea cual sea nuestro propósito. 2. **Libertad 1:** la libertad para estudiar el funcionamiento del programa y adaptarlo a nuestras necesidades -el acceso al código fuente es condición indispensable para esto-. 3. **Libertad 2:** la libertad para redistribuir copias y ayudar así a nuestro compañero. 4. **Libertad 3:** la libertad para mejorar el programa y luego publicarlo para el bien de toda la comunidad el acceso al código fuente es condición indispensable para esto.

Software libre es cualquier programa cuyos usuarios gocen de estas libertades. De modo que deberíamos ser libres de redistribuir copias con o sin modificaciones, de forma gratuita o cobrando por su distribución, a cualquiera y en cualquier lugar. Gozar de esta libertad significa, entre otras cosas, no tener que pedir permiso ni pagar para ello.

2.2.3.3. Definición de software propietario

El software no libre también es llamado software propietario, software privativo, software privado o software con propietario. Se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o que su código fuente no está disponible o el acceso a éste se encuentra restringido.

CAPÍTULO III

ANÁLISIS COMPARATIVO DE SOFTWARE DE VIRTUALIZACIÓN DE ORDENADORES

3.1. INTRODUCCIÓN

En este capítulo se presentará los sistemas de virtualización de ordenadores para la implementación en la Institución según los parámetros o métricas. Estas métricas nos servirán para comparar a los diferentes sistemas de virtualización de ordenadores que existen para Linux; entre ellos tenemos: Bochs, QEMU, VMWare Server, z/VM, Xen, UML, Linux-VServer, OpenVZ, Virtual Box, etc.

De entre estos sistemas de virtualización de ordenadores los más importantes son Xen que provee el Centos, VMWare Server y Virtual Box y se ha escogido estos debido a que

son los más utilizados y también porque la institución cuenta con un servidor que trabaja bajo la plataforma Linux. El resultado de este estudio nos servirá para escoger el sistema de Virtualización de ordenadores que mejor se adapte a la tecnología con que cuenta EIS, y por supuesto, sin dejar a un lado la propuesta que es desarrollar una infraestructura de servidores virtuales que no represente altos costos.

De los sistemas de virtualización de ordenadores antes mencionados se estudiarán las características más importantes que nos ayudarán a definir los parámetros necesarios, para establecer si el sistema de virtualización de ordenadores seleccionado cumple y se ajusta con las exigencias de la infraestructura tecnológica de la institución.

3.2. GENERALIDADES

3.2.1. Determinación de los software a comparar

Como todos conocemos existen dos tipos de programas: los que son pagos y los que no. Dentro de los programas pagos encontramos uno de los más famosos: el VMware Server ESX, que es uno de los referentes en el mercado. A pesar de ser pagado también existe una versión más básica que es gratuita, VMware Server.

Parallels Virtuozzo Containers, es otro de los programas pagos más famosos, que permite la virtualización a nivel de sistema operativo o hardware Parallesl Bare Metal. Típicamente suele emplearse para virtualizar Windows y, en menor medida, GNU/Linux.

Dentro de los programas gratuitos tenemos el Virtual PC de Microsoft, que es un producto de Windows, compatible con versiones avanzadas de XP y Vista.

Dentro de los programas de código libre están el Xen, OpenVZ y VirtualBox, que funcionan tanto en Windows como en GNU/Linux y ambos permiten virtualizar los tres sistemas operativos más famosos, Linux, Windows y Mac OS X.

Existen otros en el mercado pero no es óptimo en su rendimiento y no son de distribución libre. Para el desarrollo de la infraestructura de servidores virtuales, se han seleccionado 3 de ellos por las razones que se explican a continuación:

- El sistema de virtualización Xen cuenta con la licencia GPL de código abierto y está disponible para la descarga de forma gratuita.

- La virtualización Xen ayuda a reducir costes en las organizaciones, Xen es un software de código abierto que permite implementar servidores virtuales.

- La tecnología Xen es utilizada principalmente en el sector corporativo donde grandes cluster son completamente virtualizados. Prueba de ello es que las grandes compañías como Google, Ebay, Yahoo y Amazon utilizan Xen como su plataforma de virtualización sobre la cual corren sus servicios. Es libre y viene incorporado directamente sobre el hardware en marcas como Dell y HP. Las empresas en todo el mundo están rápidamente migrando a esta nueva tecnología de código libre, que es rápida, segura y obtiene un rendimiento similar a correr el servidor directamente sobre el equipo físico. Xen provee un conjunto de aplicaciones y servicios para implementar la virtualización con incluso mayor estabilidad para sistemas Windows y Linux.

- La decisión de usar la distribución VMWare Server es porque la compañía de Virtualización de servidores Vmware está haciendo una promoción donde ofrece su producto Vmware server gratis. En donde el producto es "Generally Available" y está disponible sin costo para Linux y para Windows.

- VirtualBox porque es una herramienta de virtualización de código abierto con la que se puede ejecutar Linux bajo Windows y viceversa.

- En el Ministerio de Educación Ecuador todas sus aplicaciones y servicios están siendo montados en servidores virtualizados con Xen.

- La empresa Telefónica Ecuador dentro de sus objetivos de optimización de la plataforma de monitoreo y gestión de su core, para sus actividades cuenta con la existencia de recursos hardware suficientes para el soporte de más de una aplicación. Se configuró 4 aplicaciones las mismas que deberían manejarse en ambientes independientes por lo tanto se decidió la virtualización utilizando xen.

  La infraestructura se montó en un servidor HP Proliant DL360 con 16 GB RAM y 2 CPU Qual Core donde se crearon 4 servidores virtuales en los que se ha instalado las aplicaciones **hobbit** (monitoreo de todo la plataforma), **oracle grid** (monitoreo de toda la base de datos), **cacti** (monitoreo de la red) y **Gateway de acceso** (permite conexión al sistema).

Es por esto que para la realización de este estudio comparativo se han seleccionado los software de virtualización de software WMware, Xen y VirtualBox por los siguientes motivos:

- Los tres son software de virtualización de ordenadores que gozan de una gran acogida entre los administradores de infraestructura de servidores virtuales, además que cuentan con una amplia gama de recursos disponibles para su utilización.

- Son software de virtualización de ordenadores de código abierto u open source, por lo que tanto su distribución, como el soporte para las mismas además de herramientas adicionales que brinden más y mejores opciones a los administradores de servidores, son gratuitas y mucho más fáciles de conseguir que las aplicaciones que son de pago.

- Está establecido como una política de Gobierno y de Estado la utilización del software libre como medio para garantizar la soberanía y como paso para la integración y liberación de América Latina, apoyando con disminución en el gasto y soberanía tecnológica. Para lograr utiilizar los programas informáticos que pueden ser distribuidos, copiados, estudiados y modificados libremente.

3.2.2. Análisis de los sistemas de virtualización seleccionados

A continuación analizaremos más a cerca de cada una de los software definidos anteriormente:

3.2.2.1. VMware Server

VMware es un sistema de virtualización por software. Un sistema virtual por software es un programa que simula un sistema físico (un ordenador, un hardware) con unas características de hardware determinadas. Cuando se ejecuta el programa (simulador), proporciona un ambiente de ejecución similar a todos los efectos a un ordenador físico (excepto en el puro acceso físico al hardware simulado), con CPU (puede ser más de

una), BIOS, tarjeta gráfica, memoria RAM, tarjeta de red, sistema de sonido, conexión USB, disco duro (pueden ser más de uno), etc.

Un virtualizador por software permite ejecutar (simular) varios ordenadores (sistemas operativos) dentro de un mismo hardware de manera simultánea, permitiendo así el mayor aprovechamiento de recursos. No obstante, y al ser una capa intermedia entre el sistema físico y el sistema operativo que funciona en el hardware emulado, la velocidad de ejecución de este último es menor, pero en la mayoría de los casos suficiente para usarse en entornos de producción.

VMware inserta directamente una capa de software en el hardware del ordenador o en el sistema operativo host. Esta capa de software crea servidores virtuales y contiene un monitor de máquina virtual o “hipervisor” que asigna recursos de hardware de forma dinámica y transparente, para poder ejecutar varios sistemas operativos de forma simultánea en un único ordenador físico sin ni siquiera darse cuenta.

No obstante, la virtualización de un ordenador físico único es sólo el principio. VMware ofrece una sólida plataforma de virtualización que puede ampliarse por cientos de dispositivos de almacenamiento y ordenadores físicos interconectados para formar una infraestructura virtual completa.

VMware fue uno de los primeros software de virtualización para PC. Con capacidad de correr casi en cualquier sistema operativo basado en x86 como sistema operativo invitado por encima de otros, este ofrece nuevas posibilidades para la utilización del hardware.

Tanto para desarrollo, prueba, demostración y producción en ambiente de servidores, VMware provee una plataforma sólida. Además, los nuevos sistemas operativos invitados pueden salvarse como imágenes, compartirse y utilizarse gratuitamente.

3.2.2.1.1. Historia VMware Server

VMware es una compañía pionera en el campo de la virtualización para x86. Los productos VMware existen desde hace tiempo: primero para Windows, pero luego también para Linux. Aunque al principio se pensó como una solución cómoda para desarrolladores, con la que podían probar nuevos programas sin riesgo para sus PCs, ahora existen productos dirigidos a las grandes empresas, para la consolidación de granjas de servidores, etc.

La virtualización es un concepto reconocido que comenzó a desarrollarse en la década de 1960 para particionar el hardware de mainframe de gran tamaño. Hoy en día, los ordenadores basados en arquitectura x86 se enfrentan a los mismos problemas de rigidez e infrautilización a los que se enfrentaban los mainframes en la década de 1960.

VMware inventó en la década de los 90 la virtualización de la plataforma x86 para solucionar dicha infrautilización, superando de paso muchos otros problemas.

Actualmente, VMware es el líder mundial en virtualización x86 y ha logrado aumentar el impulso de la virtualización en este mercado.

Fue IBM quien empezó a implementar la virtualización hace más de 30 años como una manera de lógica de particionar ordenadores mainframe en máquinas virtuales independientes. Estas particiones permitían a los mainframes realizar varias tareas:

ejecutar varias aplicaciones y procesos al mismo tiempo. Dado que en aquella época los mainframes eran recursos caros, se diseñaron para ser particionados para así poder aprovechar al máximo la inversión.

En 1999, VMware introdujo la virtualización en los sistemas x86 como un medio para solucionar de manera eficiente muchos de estos problemas y transformar los sistemas x86 en sistemas para uso general, en infraestructuras de hardware compartido que ofrecen un aislamiento completo, movilidad y opciones de elección del sistema operativo de los entornos de aplicación.

A diferencia de los mainframes, las máquinas x86 no fueron diseñadas para admitir una virtualización completa, por lo que VMware tuvo que superar muchos desafíos para crear máquinas virtuales en ordenadores x86.

La función básica de la mayoría de las CPU, tanto en mainframes como en PC, es ejecutar una secuencia de instrucciones almacenadas (por ejemplo, un programa de software). En los procesadores x86, hay 17 instrucciones específicas que generan problemas a la hora de virtualizar, y provocan que el sistema operativo muestre un aviso, que se cierre la aplicación o simplemente que falle completamente. Como resultado de ello, estas 17 instrucciones constituían un obstáculo importante a la implementación inicial de la virtualización de ordenadores x86.

Para hacer frente a las instrucciones problemáticas de una arquitectura x86, VMware desarrolló una técnica de virtualización adaptable que las “atrapa” cuando se generan y las convierte en instrucciones seguras que se pueden virtualizar, al tiempo que permite al resto de instrucciones ejecutarse sin intervención. El resultado es un servidor virtual de

alto rendimiento que se adapta al hardware host y mantiene una total compatibilidad de software.

3.2.2.1.2. Plataforma VMware Server

VMware Server es, hoy en día, la plataforma líder en sistemas virtualizados y tiene una gran experiencia tanto a nivel empresarial como a nivel doméstico, constituido en su inferior la capa de virtualización, el hipervisor, aunque posee herramientas y servicios de gestión autónomos e independientes.

Está compuesto de un sistema operativo autónomo que proporciona el entorno de gestión, administración y ejecución al software hipervisor, y los servicios y servidores que permiten la interactuación con el software de gestión y administración y los servidores virtuales.

VMware Server trabaja directamente el sistema operativo host ya instalado Linux o Windows. A partir de allí crea servidores virtuales y contiene un monitor de servidor virtual que asigna recursos de hardware de forma dinámica y transparente, para poder ejecutar varios sistemas operativos de forma simultánea en un único ordenador físico sin ni siquiera darse cuenta.

No obstante, la virtualización de un ordenador físico único es sólo el principio. VMware ofrece una sólida plataforma de virtualización que puede ampliarse por cientos de dispositivos de almacenamiento y ordenadores físicos interconectados para formar una infraestructura virtual completa.

3.2.2.1.3. Arquitectura de la Plataforna VMware Server

VMware Server se instala y se ejecuta como una aplicación en la parte superior de un gran Sistema operativo Windows o Linux. Una fina capa de virtualización de particiones de un servidor físico para que múltiples servidores virtuales puedan se ejecutados simultáneamente en un único servidor. Recursos informáticos de un servidor físico se tratan como un conjunto uniforme de los recursos que se pueden asignar a los servidores virtuales de manera controlada.

VMware Server aísla cada servidor virtual de su anfitrión y de otros servidores virtuales, sin afectar si otro en el caso de un servidor virtual se cuelga. Los datos no se filtran a través de servidores virtuales y de las aplicaciones sólo pueden comunicarse a través de la configuración de red. VMware Server encapsula un servidor virtual en un ambiente como un conjunto de archivos, que son fáciles de obtener un back-up, mover y copiar.

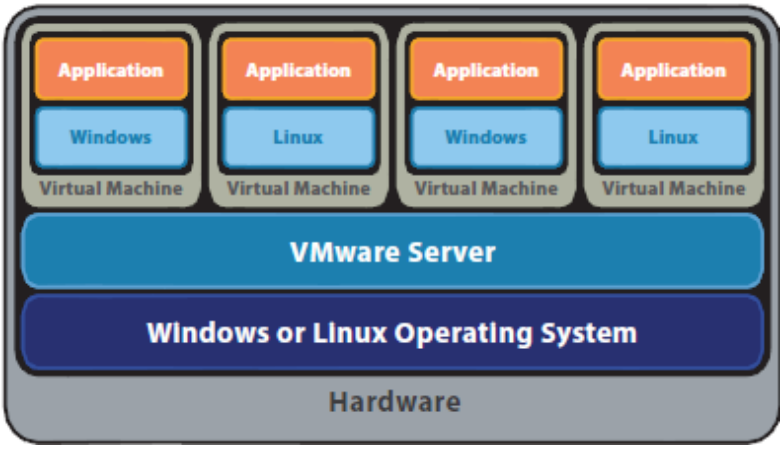

Figura III.11. Arquitectura del VMWare Server

3.2.2.1.4. VMware Server como software de virtualización de ordenadores

La estrategia de VMware parece ser lograr que las que empresas se adapten al uso de vmware con las versiones gratuitas, para que más usuarios y empresas se familiaricen con ellas y luego ayude a adoptar en sus empresas las soluciones “corporativas” como VMware ESX Server (un sistema Red Hat Linux modificado con muchas herramientas de gestión, sobre el que se lanzan los servidores virtuales).

VMware Server es el producto más potente de los que ofrece gratuitamente VMware. Permite crear, modificar y ejecutar máquinas virtuales desde una interfaz gráfica, usando una arquitectura cliente-servidor.

3.2.2.2. Xen

Xen es un motor de servidores virtuales de código abierto, la meta del diseño es poder ejecutar instancias de sistemas operativos con todas sus características, de forma completamente funcional en un equipo sencillo. El software de virtualización Xen es una solución con fuente abierta desarrollada por la comunidad Linux.

Xen proporciona aislamiento seguro, control de recursos, garantías de calidad de servicio y migración de servidores virtuales en vivo. Los sistemas operativos deben ser modificados explícitamente para correr Xen (aunque manteniendo la compatibilidad con aplicaciones de usuario). Esto permite a Xen alcanzar virtualización de alto rendimiento sin un soporte especial de hardware.

Los servidores virtuales son usados a menudo por IBM y otras compañías en sus servidores y ordenadores centrales para abstraer la mayor cantidad de aplicaciones posibles y proteger las aplicaciones poniéndolas en seridores virtuales diferentes (semejante a una jaula chroot). Puede también ser utilizada, no solo por razones de seguridad o funcionamiento, sino también para poder tener arrancados diferentes sistemas operativos en el mismo ordenador. Con la migración de máquinas virtuales en vivo de Xen se puede conseguir hacer balance de cargas sin tiempos muertos.

Las arquitecturas soportadas actualmente por Xen son x86/32 y x86/64 y la última versión desarrollada es la 3.0.3. La técnica utilizada por Xen se denomina para-virtualización (VMM) (virtual machine monitor) o hipervisor, lo que consigue comportamientos de las máquinas virtuales cercanos al de un servidor real.

Para realizar dicha para-virtualización, es necesario cargar en modo núcleo (kernel space) el denominado “hypervisor” que se encarga de la gestión de recursos para los diferentes sistemas operativos de un mismo servidor. La contrapartida en el caso de Xen es que el sistema operativo huésped (guest) debe modificarse para trabajar con el hypervisor en lugar de con el hardware directamente. A día de hoy sólo pueden trabajar con Xen sistemas operativos libres, ya que en estos casos es factible modificar el núcleo (kernel) de forma apropida. Los sistemas operativos “propietarios” deben sacar versiones específicas para trabajar con Xen.

Existe la posibilidad de trabajar con Xen sin necesidad de modificar el sistema operativo huésped, pero para ello es necesario utilizar un microprocesador con tecnología de virtualización. Algunos microprocesadores salidos recientemente al mercado incluyen esta característica, son los conocidos como Intel VT (Virtualization Technology or VanderPool) y AMD-V (Advanced Micro Devices - Virtualization) (Pacifica).

Xen facilita varias funciones empresariales, incluyendo:

- Máquinas virtuales con el rendimiento cercano a la ejecución de manera nativa.
- Migración viva de servidores virtuales ejecutándose entre diferentes equipos físicos.
- Soporta hasta 32 CPUs virtuales por máquina virtual, con conexión en caliente de CPUs virtuales.
- Soporte de la Tecnología de Virtualización de Intel (VT-x) para sistemas operativos sin modificar (incluyendo Microsoft Windows).
- Excelente soporte de hardware (casi todos los dispositivos soportados en Linux.)
- Se trata de una excelente herramienta didáctica para el aprendizaje de redes de computadoras, ya que permite al administrador configurar de forma virtual una red completa en un solo equipo.
- Es una alternativa cada vez más real para la utilización de servidores dedicados virtuales para empresas que ofrezcan servicios a Internet y quieran ahorrar costes sin perder seguridad.

Los diferentes servidores virtuales que se ejecutan en un servidor físico reciben el nombre de dominios en la terminología de Xen. Existe un dominio privilegiado que es sobre el que se instala el “hypervisor” de Xen y que equivale al sistema operativo anfitrión (host) de otros monitores de servidores virtuales como los de la empresa VMware. Este dominio privilegiado recibe el nombre de dom0 y el resto de dominios reciben el nombre genérico de domU.

El hardware que actualmente puede utilizar un domU en Xen es:

- Interfaz de red
- Disco duro

- Memoria RAM
- Dispositivos de E/S

Para el resto de dispositivos: tarjeta gráfica, disquete, CD-ROM, USB, ACPI, hay que elegir el dominio que va a utilizarlo, pero no pueden “compatirse”. Esto puede parecer una gran limitación para determinados usos de las máquinas virtuales, pero no para los aquí mencionados.

3.2.2.2.1. Historia Xen

Xen fue inicialmente un proyecto de investigación de la Universidad de Cambridge (la primer versión del software fue publicada a fines de 2003). Este proyecto de investigación fue liderado por Ian Pratt, quien luego formó una empresa -junto con otras personas- para dar servicios de valor agregado como soporte, mantenimiento y capacitación sobre Xen en Enero de 2005. Esta empresa es Xensource Inc., recibió fondos por millones de dolares de diferentes inversores y actualmente mantiene Xen (junto con otras empresas y la comunidad), también se dedica a programar aplicaciones adicionales no libres para facilitar el uso, instalación y mantenimiento de Xen.

Dado que Xen está licenciado bajo GPL el código no puede cerrarse, y no es solo Xensource quien mantiene el código, sino que varias empresas importantes como IBM, Sun, HP, Intel, AMD, RedHat, Novell están sumamente involucradas en el desarrollo asignando programadores al mantenimiento de este software.

3.2.2.2.2. Plataforma Xen

- Xen es estable y muy manejable.

- Plataforma Windows: hubo una versión modificada de Windows XP funcionando durante los primeros tests. Dicha versión no ha podido comercializarse debido a las restrictivas licencias y contratos que Microsoft aplica a sus productos.
- Para portátiles: Xen no soporta ACPI o APM, por lo tanto funcionará pero no con todas las funcionalidades de un portátil, aunque los desarrolladores esperan poder soportar estas tecnologías de portátiles próximamente.
- Los servidores virtuales Xen pueden ser migrados en vivo entre equipos físicos sin pararlos. Durante este proceso, la memoria del servidor virtual es copiada iterativamente al destino sin detener su ejecución. Una parada muy breve alrededor de 60 a 300 ms es necesaria para realizar la sincronización final antes de que el servidor virtual comience a ejecutarse en su destino final. Una tecnología similar es utilizada para suspender los servidores virtuales a disco y cambiar a otra máquina virtual.

3.2.2.2.3. Características Xen

- En primer lugar, es de código abierto.
- En segundo lugar, es relativamente liviano, de modo que no consume una cantidad excesiva de recursos del procesador.
- En tercer lugar, alcanza un alto nivel de aislamiento entre tecnologías de servidor virtual.
- Por último, al igual que otras tecnologías de servidor virtual, Xen ofrece soporte a sistemas operativos y versiones combinados y permite a los administradores definir y ejecutar una instancia del sistema operativo, en forma dinámica, sin afectar el servicio.

- **Consolidación de Servidores**

Mover multiples servidores en un sólo equipo con el rendimiento y aislamiento de fallas proveído por Xen.

- **Independencia de Hardware**

Permite que aplicaciones semi-obsoletas y sistemas operativos explotar hardware nuevo.

- **Configuraciones multiples de Sistemas Operativos**

Permite correr multiples sistemas operativos simultáneamente, para propósitos de desarrollo y pruebas

- **Desarrollo en kernel**

Permite depurar y probar modificaciones en kernel en un servidor virtual aislada, no se necesita un equipo de pruebas separado.

- **Supercómputo**

El manejo granular de servidores virtuales provee más flexibilidad que el manejo de equipos por separado, también un mejor control y aislamiento que en soluciones de sistemas simples, particularmente para migraciones en vivo y balanceo de cargas.

- **Soporte de Hardware para Sistemas Operativos personalizados**

Permite el desarrollo de nuevos Sistemas Operativos con el beneficio del amplio soporte de hardware en Sistemas Operativos existentes como Linux.

3.2.2.2.4. Arquitectura de la Plataforma Xen

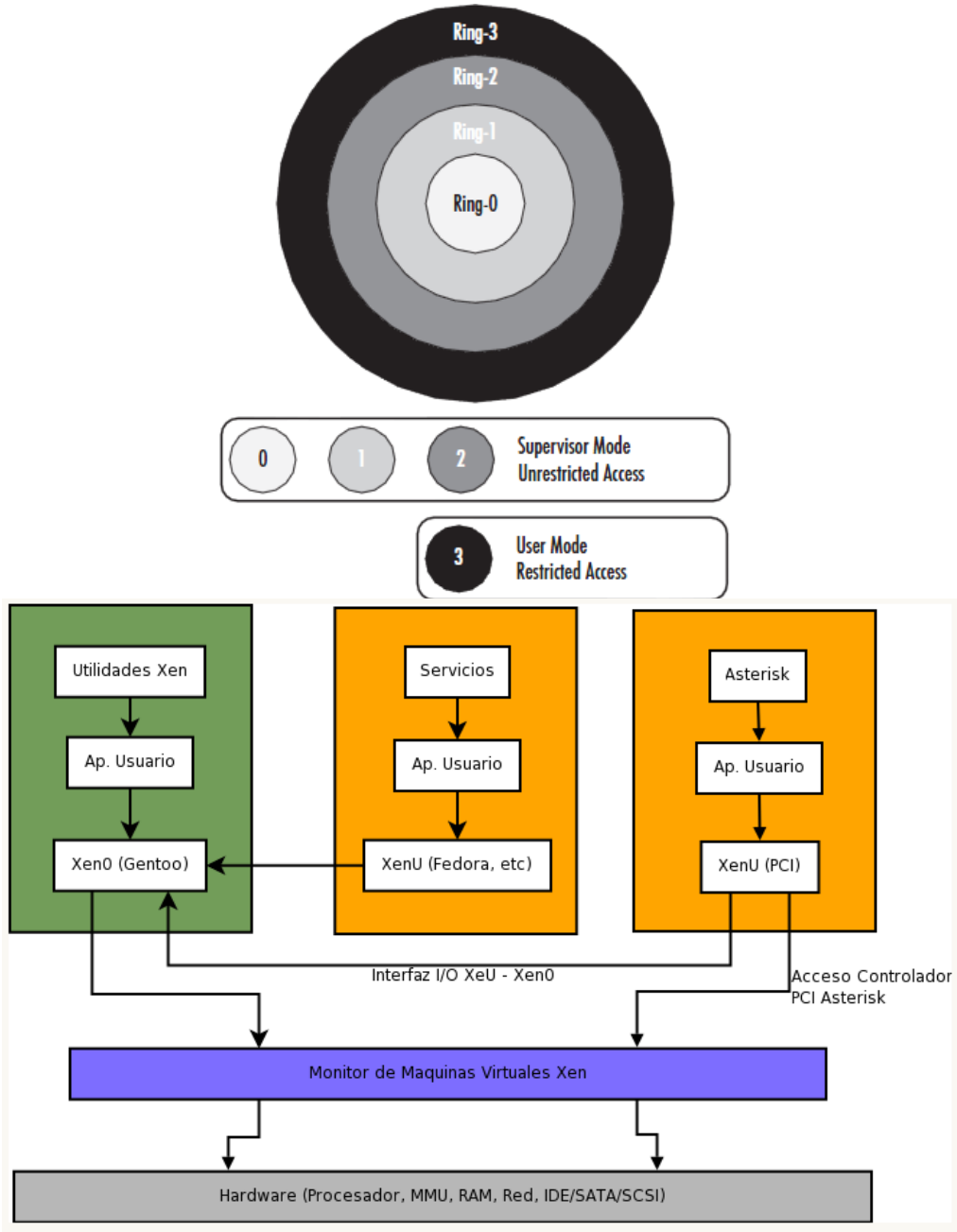

Figura III.12. Arquitectura Xen

Xen hace uso del modo protegido del i386, donde la CPU está compuesta por 4 anillos, el ring 0 se usa normalmente para el kernel del sistema operativo y el ring 3 se usa para las aplicaciones de usuario.

Xen “hypervisor” corre en ring 0, los Sistemas Operativos invitados corren en ring 1 y las aplicaciones en ring 3. Con esto aprovechamos el ring 1 como una capa mas de protección, algo que nadie hasta ahora había hecho. En tiempo de arranque del sistema operativo anfitrion Xen se carga en memoria y ejecuta un kernel parcheado en Ring 1 que se llama domain0. Desde este domino será desde el cual se podrá crear, destruir, migrar o detener el resto de dominios. Estos dominios creados también funcionarán en ring1, mientras que sus aplicaciones lo harán en ring3. Para poder acceder a los dispositivos físicos de una forma segura Xen utiliza el domain0 que es el único que puede acceder a ellos, de modo que los sistemas operativos que corran en dominios, como ya hemos dicho deberán ser parcheados para acceder a los dispositivos físicos. Este es el mayor inconveniente de Xen, pero como ya también se ha dicho en varias ocasiones cuando dispongamos de la tecnologia de virtualización por hardware tanto de AMD como Intel este problema no existirá.

A continuación la explicación de la Figura III.12.:

- La primera capa (gris) es el hardware del servidor.
- Sobre el hardware corre Xen (lila), el cual restringe las direcciones de memoria y el acceso a los dispositivos.
- Xen ejecuta Xen0 (verde), el cual es el sistema operativo que tiene acceso a todos los dispositivos encontrados en la computadora, ejecutandose con RAM restringida.

- Dentro de Xen0, se inicia el arranque de los servidores virtuales XenU (naranja), de las cuales una es un prestador de servicios (web, correo, etc.) y la otra controla un dispositivo PCI.
- El servidor virtual XenU que requiere de acceso a un dispositivo PCI, se comunica al monitor Xen (lila) y para sus servicios de ejecución, se comunica con Xen0 (verde).

3.2.2.2.5. Xen como software de virtualización de ordenadores

Un servidor virtual es un software que crea una plataforma 'puente' entre el administrador y el ordenador, permitiendo que este ejecute determinado software que originalmente no podría funcionar. Los servidores virtuales no son algo nuevo, llevan usándose desde principios de los años 70 y principalmente se idearon para correr varios sistemas operativos diferentes y separados en un mismo servidor físico. Los servidores virtuales también se usan en algunos lenguajes de programación, siendo en la actualidad la más popular la máquina de Java desarrollada por Sun.

Java, al compilarse, genera un bytecode que solo puede ser ejecutado por su propia máquina virtual. Con esto se consigue la portabilidad de los binarios generados con el compilador entre sistemas operativos. Cuando hablamos de virtualizar hoy en día nos referimos normalmente a ejecutar un sistema operativo dentro de otro. El Xen inicialmente necesitaba que los sistemas operativos invitados fueran modificados para ejecutarse exitosamente (esto se conoce como paravirtualización). Sin embargo, la última release (Xen 3.0) incluye soporte para la Tecnología de Virtualización Intel®, lo que permite que el Xen soporte sistemas operacionales invitados no modificados, incluyendo el Windows*7.

El Xen 3.0 soporta hasta 32 procesadores por servidor virtual y transferencias activas de las aplicaciones que están ejecutándose. También se ejecuta tanto en los sistemas basados en los procesadores Intel® Xeon® como en los procesadores Intel Itanium® 2.

Dada la escalabilidad y la disponibilidad moderna de los servidores basados en los procesadores Itanium® 2, la virtualización basada en Linux podrá ir aún más lejos en los datacenter, ofreciendo la escalabilidad y la disponibilidad necesaria para consolidar las aplicaciones más críticas de las empresas.

Xen es una tecnología relativamente nueva, pero el soporte de la industria y el interés de los administradores es grande, y las soluciones listas para ser desarrolladas están comenzando a surgir, y avances rápidos pueden esperarse.

- **Aplicación**

- Dividir cada servidor físico en hasta 30 servidores virtuales, cada uno siendo capaz de hospedar su propio SO (Sistema Operativo) y su pila de aplicaciones.
- Desarrollar y administrar servidores físicos y virtuales de forma eficiente a partir de una interfaz común.
- Destinar los recursos del servidor (procesador, memoria e I/O) dinámicamente y mover aplicaciones y cargas de trabajo en ejecución, y cambiar sesiones muy rápidamente de un servidor virtual para otro. Inicialmente esta capacidad era usada para manutención del tiempo de inactividad cero. Ahora ella comienza a usarse como una manera de proveer automáticamente la nueva capacidad cuando se presentan fallas en un sistema o cuando las cargas de trabajo amenazan con exceder los recursos existentes. Obviamente, la planeación es importante en un proyecto de

consolidación bien hecho. Para la mayoría de las empresas, la cuestión no es si deben o no virtualizar sus infraestructuras para los servidores sino cuál solución usar.

Las políticas para las tomas de decisiones normalmente necesitan cambiar, ya que los servidores físicos individuales podrán ser compartidos entre múltiples unidades de negocios.

- **Ventajas**

- Aislamiento e independencia de servicios y contenidos.
- Se puede utilizar un servidor virtual diferente para ejecutar servicios web, ftp, correo y otros.
- Un fallo en el sistema operativo no repercute en los demás.
- Adecuado cuando se heredan sistemas más antiguos y se pueden producir conflictos entre diferentes versiones de librerías.
- Se consigue alto grado de seguridad y facilidad de migración.
- Ahorro de hardware.
- Se consigue utilizar más los recursos del sistema y procesadores.

3.2.2.3. VirtualBox

3.2.2.3.1. Historia Virtual Box

Sun xVM VirtualBox es un software de virtualización para arquitecturas x86 que fue desarrollado originalmente por la empresa alemana Innotek GmbH, pero que pasó a ser propiedad de la empresa Sun Microsystems en febrero de 2008 cuando ésta compró a innotek. Por medio de esta aplicación es posible instalar sistemas operativos adicionales,

conocidos como “sistemas invitados”, dentro de otro sistema operativo “anfitrión”, cada uno con su propio ambiente virtual. Por ejemplo, se podrían instalar diferentes distribuciones de GNU/Linux en VirtualBox instalado en Windows XP o viceversa.

La aplicación fue inicialmente ofrecida bajo una licencia de software privado, pero en enero de 2007, después de años de desarrollo, surgió VirtualBox OSE (Open Source Edition) bajo la licencia GPL 2. Actualmente existe la versión privada, VirtualBox, que es gratuita únicamente bajo uso personal o de evaluación, y esta sujeta a la licencia de “Uso Personal y de Evaluación VirtualBox” (VirtualBox Personal Use and Evaluation License o PUEL) y la versión Open Source, VirtualBox OSE, que es software libre, sujeta a la licencia GPL.

3.2.2.3.2. Plataforma Virtual Box

VirtualBox está disponible para su ejecución en sistemas Windows y Linux de 32-bits (aunque hay también una versión beta para MacOS X) y es capaz de virtualizar Windows, Linux (versión del núcleo 2.x), OS/2 Warp, OpenBSD y FreeBSD.

Comparado con otros programas de virtualización como VMware o Virtual PC, VirtualBox carece de algunas funcionalidades, pero a cambio aporta otras como: ejecución remota de máquinas virtuales utilizando Remote Desktop Protocol (RDP), soporte para iSCSI y soporte para USB con dispositivos remotos sobre RDP.

VirtualBox soporta virtualización VT-x para el hardware de Intel, y (de manera experimental) virtualización AMD-V para el hardware de AMD.

3.2.2.3.3. Características VirtualBox

El software VirtulaBox incluye una serie de características que enriquecen sus posibilidades de utilización:

- **Integración del ratón**

No tendremos que buscar el ratón una vez que ejecutemos el sistema operativo virtual, que suele ser incómodo al haber “dos ratones”.

- **Compartir carpetas**

Carece de distinción entre sistemas operativos, es decir, puede funcionar completamente con Linux teniendo Windows o Mac y viceversa. Cosa que suele ser asunto de otros programas auxiliares en estos casos y en VirtualBox ya viene integrado.

- **Software ligero**

Es de las aplicaciones de virtualización de servidores que menos ocupa.

- **Gratuita**

A diferencia de VMware que tiene dos versiones, la libre y que es necesaria la compra de una licencia (Professional edition), y que difieren en la velocidad a la que trabajan. VirtualBox y VirtualPC sólo tienen versión gratuita.

- **Fácil y rápido**

El tiempo estimado en descargar la aplicación, instalarla y ponerla en funcionamiento no sobrepasa el dolor de cabeza.

3.2.2.3.4. Arquitectura de Virtual Box

VirtualBox ofrece una capa extensible, diseño modular que incluye amplias herramientas de desarrollo e interfaces. Una capa de hypervisor que viene a ser el núcleo del motor de virtualización que controla la ejecución VM que es la base de la arquitectura de funcionamiento.

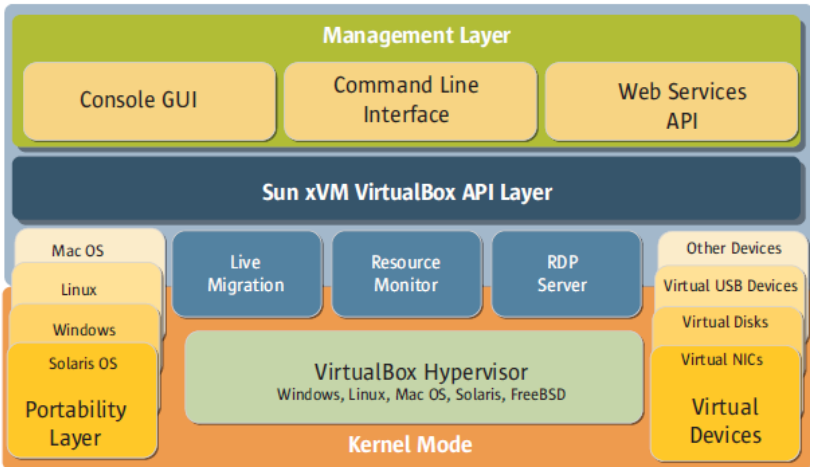

Figura III.13. Arquitectura de Virtual Box

3.2.2.3.5. VirtualBox como software de virtualización de ordenadores

VirtualBox es un programa que crea un ordenador virtual dentro de nuestro sistema operativo donde se instalar cualquier sistema operativo como Linux dentro de Windows

Vista, Windows XP en Linux, es posible con este software de virtualización de ordenadores de la forma más fácil.

Con VirtualBox tendremos un sistema operativo dentro de nuestro sistema operativo pero independiente; es decir, éste no afectará de ninguna forma al sistema operativo host.

VirtualBox está diseñado para trabajar tanto en Linux como en Windows y es capaz de virtualizar los sistemas operativos, entre otros: Windows Vista, XP, 98, 95, Linux, FreeBSD, OpenBSD.

Un detalle a destacar que no podemos dejar de pasar es que, si bien VirtualBox no ocupa mucho espacio, pero los servidores virtuales sí que ocupan espacio.
En cuanto a la emulación de hardware, los discos duros de los sistemas invitados son almacenados en los sistemas host como archivos individuales en un contenedor llamado Virtual Disk Image, incompatible con los demás software de virtualización.

Otra de las funciones que presenta es la de montar imágenes ISO como unidades virtuales de CD o DVD, o como un disco floppy.

3.2.3. Determinación de los parámetros de comparación

Para la realización del estudio comparativo, se debe considerar una serie de aspectos, con los que podremos determinar cuál de los software de virtualización de ordenadores presenta una solución práctica, efectiva y económica para optimizar los recursos: económicos, hardware, humanos, consumo energético y servicios, con los que cuenta la

EIS, este estudio comparativo se realizará de cada uno de los parámetros detallados a continuación:

3.2.3.1. Instalación

La definición de los pasos a realizar en la ejecución de la instalación de los software de virtualización de ordenadores es uno de los aspectos más importantes en la virtualización de servidores porque es la forma de comunicación que tiene el administrador con el software de virtualización de servidores virtuales a instalar.

3.2.3.2. Escalabilidad

La capacidad de que un hardware crezca, adaptándose a nuevos requisitos conforme cambian las necesidades del negocio. La virtualización ofrece capacidad de desarrollo y ventajas competitivas para obtener una mayor escalabilidad, seguridad y acceso a toda la información desde cualquier lugar, lo que hoy en día representa una herramienta que podría marcar la diferencia entre desaparecer y seguir creciendo.

3.2.3.3. Alta disponibilidad

Alta disponibilidad se refiere a la habilidad de los administradores de un software de virtualización de ordenadores para poder acceder al mismo y trabajar con él de modo correcto en cualquier momento. La máxima disponibilidad se consigue cuando no existe tiempo de inactividad no programada. Los fallos del hardware, el software, de acceso a datos, comunicaciones, factores ambientales adversos y desastres (naturales o artificiales) comprometen este factor. Para resolver estos problemas se dota a los sistemas de arquitecturas redundantes que permitan la continuidad en caso de fallo de

elementos simples o de determinados conjuntos de elementos que tomen el relevo en la operación de los elementos susceptibles de fallo.

3.2.3.4. Flexibilidad

Permite mayor flexibilidad para administrar el hardware subyacente y proporciona un provisionamiento dinámico de los recursos computacionales, lo que aumenta la flexibilidad y la utilización en entornos de servidor, integrando recursos que pueden ser manejados centralmente por un eje de la infraestructura para un mejor soporte al escoger los sistemas operativos, el software y proporcionando abstracción de servidores para que la administración de aplicaciones y servicios se desacople de los servidores físicos cambiando dinámicamente los requisitos del negocio.

3.2.3.5. Usabilidad

La usabilidad, hace referencia, a la rapidez y facilidad con que los usuarios llevan cabo sus tareas propias a través del uso de un software de virtualización de ordenadores, idea que descansa en cuatro puntos:

- **Una aproximación al usuario:** Usabilidad significa enfocarse en los usuarios. Para desarrollar un producto usable, se tienen que conocer, entender y trabajar con los administradores que representan a los usuarios actuales o potenciales del software de virtualización de ordenadores.

- Un amplio conocimiento del contexto de uso: Los administradores utilizan los software de virtualización de ordenadores para incrementar su propia productividad. Un software de virtualización de ordenadores se considera fácil de aprender y usar en términos del tiempo que toma el administrador para llevar a cabo su objetivo, el número de pasos que tiene que realizar para ello, y el éxito que tiene en predecir la acción apropiada para llevar a cabo.

- **El software de virtualización de ordenadores a de satisfacer las necesidades del administrador:** Los administrador son gente ocupada intentando llevar a cabo una tarea. Se va a relacionar usabilidad con productividad y calidad.

3.2.3.6. Estabilidad

La estabilidad es una propiedad cualitativa del software de virtualización de ordenadores a la que cabe considerar como la más importante de todas. Ello es debido a que, en la práctica, todo software de virtualización de ordenadores debe ser estable. Si un sistema no es estable, normalmente carece de todo interés y utilidad.

3.2.3.7. Rendimiento

Medida o cuantificación de la velocidad/resultado con que el software de virtualización de ordenadores realiza una tarea o proceso.

El rendimiento son las pruebas que se realizan, desde una perspectiva, para determinar lo rápido que realiza una tarea un software de virtualización de ordenadores en condiciones particulares de trabajo.

3.2.3.8. Velocidad

Es el tiempo de ejecución que toma un software de virtualización de ordenadores para gestionar los servidores virtuales.

3.2.3.9. Extensibilidad

Un software de virtualización de ordenadores ha sido diseñado desde la base para manejar la gran mayoría de entornos de TI. Una sola instancia de virtualización gestionaría muchos servidores virtuales de modo que el administrator de la infraestrura pueda gestionar varios anfitriones y servidores virtuales a través de una única consola.

3.2.4. Descripción de los Módulos de Prueba

3.2.4.1. Módulo 1

El módulo 1 será desarrollado para probar el parámetro: Instalación.

En este módulo se va a realizar la instalación de cada uno de los software de virtualización de ordenadores, para evaluar las formas en las que se puede realizar, además de la facilidad que brindan cada uno de ellos para hacerlo.

3.2.4.2. Módulo 2

El módulo 2 será desarrollado para probar el parámetro: Escalabilidad.

En este módulo se va a definir la escalabilidad como la medida de la capacidad de crecimiento de un software de virtualization de ordenadores para satisfacer demandas de rendimiento cada vez mayores superando las capacidades del mismo para ofrecer un rendimiento adecuado. Esta Información es recopilada por la documentación del fabricante.

3.2.4.3. Módulo 3

El módulo 3 será desarrollado para probar el parámetro: Alta disponibilidad.

Este módulo consta en evaluar la capacidad que tiene un software de virtualización de ordenadores para que funcione adecuadamente en cualquier momento, sea seguro, es la característica por la que se mide el tiempo de funcionamiento sin fallos o desarrolle una cierta función, bajo condiciones fijadas y durante un período de tiempo determinado.
De acuerdo a la información provenida por el fabricante es factible determinar este parámetro.

3.2.4.4. Módulo 4

El módulo 4 será desarrollado para probar el parámetro: Flexibilidad.

En este módulo se va a validar la flexibilidad que tiene un software de virtualización de ordenadores para gestionar la administración de los recursos hardware en los servidores virtuales existentes.

3.2.4.5. Módulo 5

El módulo 5 será desarrollado para probar el parámetro: Usabilidad.

Este módulo consta en la facilidad con que los administradores pueden utilizar un software de virtualización de ordenadores con el fin de alcanzar un objetivo concreto, el grado de usabilidad de un sistema es, por una parte, una medida empírica y relativa de la usabilidad del mismo.

De acuerdo a la información provenida por el fabricante es factible determinar este parámetro.

3.2.4.6. Módulo 6

El módulo 6 será desarrollado para probar el parámetro: Estabilidad.

En este módulo se va a realizar la instalación de software utilitario, llamado streess, es utilizado para determinar la estabilidad de un sistema de virtualización de ordenadores.

Se trata de realizar pruebas más allá de la capacidad operativa normal, a menudo a un punto de ruptura, con el fin de observar los resultados. Las pruebas de estrés puede tener un significado más específico en ciertas infraestructuras, tales como evaluar la capacidad de funcionamiento de los software de virtualización de ordendadores.

El estrés es un generador de la carga de trabajo deliberadamente sencilla para sistemas POSIX. Se impone una cantidad configurable de la CPU, memoria, E/S, y el estrés de disco en el sistema. Está escrito en C, y es software libre bajo la licencia GPLv2.

Serán analizados los parámetros load average, tasks running, task sleeping, cpu, memoria y swap.

- **LOAD AVERAGE**

El promedio de carga trata de medir el número de procesos activos en cualquier momento. Como una medida de la utilización de la CPU, la carga media es simplista, mal definida, pero lejos de ser inútil.

El load average muestra los tres promedios de carga para el sistema. Los promedios de carga son el número medio de procesos listos para ejecutarse durante los últimos 1, 5 y 15 minutos. Para nuestro analices será considerado el primer valor de los tres.

- **TASKS**

El número total de procesos que se ejecutan en el momento de la última actualización, son desglosados en el número de tareas que están running, sleeping, stopped o zombie.

- **CPU**

Validaremos mediante estadísticas el consumo de CPU utilizado por cada uno de los servidores de virtualización de software.

- **MEMORIA**

Estadísticas sobre el uso de memoria, incluyendo la memoria total disponible, la memoria libre, uso de memoria, la memoria compartida, y la memoria utilizada para buffers.

- **SWAP**

Las estadísticas reflejan sobre el espacio de intercambio, como espacio de intercambio total, el espacio de intercambio disponible, y espacio de intercambio utilizado.

3.2.4.7. Módulo 7

El módulo 7 será desarrollado para probar el parámetro: Rendimiento.
En este módulo se va a realizar la instalación de software utilitario, tales como: unixbench, netperf, para evaluar la capacidad de funcionamiento de los software de virtualización de ordenadores.

Unixbench es un benchmark diseñado para proporcionar una evaluación básica de la ejecución de un sistema operativo tipo Unix. Se ejecuta un conjunto de pruebas para evaluar diversos aspectos del rendimiento del sistema y, a continuación, genera un conjunto de resultados. Los componentes a ser testiados son: CPU, Memoria, Mainboard.

Netperf es un benchmark que se puede utilizar para medir diversos aspectos del rendimiento de redes. Se centra en la transferencia masiva de datos y solicitud/respuesta de rendimiento utilizando TCP o UDP. Con la opción –H se establece el nombre de host (o dirección IP) a utilizar para establecer la conexión con el sistema de control remoto. Con la opción –t especificamos la prueba a realizar: UDP_RR, UDP_STREAM -- -m 1024, TCP_RR

3.2.4.8. Módulo 8

El módulo 8 será desarrollado para probar el parámetro: Velocidad.

En este módulo se va a realizar la instalación del software utilitario, bonnie para evaluar la capacidad de funcionamiento de los software de virtualización de ordenadores.

Bonnie es una herramienta sencilla pero útil para determinar la velocidad de respuesta sistema de ficheros y almacenamiento en caché de un sistema operativo.

3.2.4.9. Módulo 9

El módulo 9 será investigado para probar el parámetro: Extensibilidad.

De acuerdo a la información provenida por el fabricante es factible determinar este parámetro.

3.2.5. Desarrollo de los Módulos de Prueba

3.2.5.1. Desarrollo de los Módulos en el software de virtualización de ordenadores WMware Server

- **Módulo 1**

Para la construcción del módulo 1 se va a realizar la instalación del sistema operativo base CentOs 5.3, como software base, y luego continuar con la instalación del software

VMware Server, se seguirá los pasos consultados en la documentación del software que a continuación se describen:

- **Instalación de VMWare Server - Números de Serie**

Para poder usar VMware Server necesitamos un número de licencia que se obtiene registrándose gratis en la página de VMware. Para ello tenemos que acceder en el enlace "Register" (en la frase Register for your free serial number(s) to start using VMware Server) que aparece en la página de descargas.

Podemos pedir varios números de licencia de una vez. En la página de registro ingresaremos nuestros datos, así como: empresa, número de empleados, sistema operativo que usamos, si hemos usado otros productos de VMware (y cuáles), etc. Solo nos podremos registrarnos una vez (salvo que lo hagamos varias veces con datos falsos), lo que se aconsejaría pedir un número alto de licencias para cubrir todas nuestras necesidades actuales y futuras. Sin embargo, en nuestras pruebas no notamos que VMware Server hiciera algún tipo de control de reutilización de licencias. Es posible (aunque quizás no esté del todo de acuerdo con los acuerdos de uso) pedir solo una licencia y luego usarla en varias instalaciones.

Lo que no llegamos a probar, y quizás sea la limitación por la que hacen falta varios números de licencias, es a conectarnos con el mismo cliente de VMware a diferentes instalaciones del servidor que compartieran la misma licencia. Suponemos que en ese caso sí sería posible encontrar algún problema.

Firgua III.14. Registro VMware

Normalmente introduciremos el número de serie al final de la instalación, pero el instalador nos permite no hacerlo en ese momento y e introducir el número luego, usando el interfaz gráfico. Para ello tenemos que ir a la opción Help del menú, y escoger la opción Enter serial number.

- **Instalación**

Podemos descargar VMware Server yendo a la página de VMware (http://www.vmware.com), accediendo en el menú Products, seleccionando Free virtualization products, y luego seleccionando VMware Server. La versión usada para esta comparativa fue la 2.0.2 del 26 de Octubre de 2009.

Para la descarga tenemos que registrarnos en caso de no tener ya creado una cuenta.

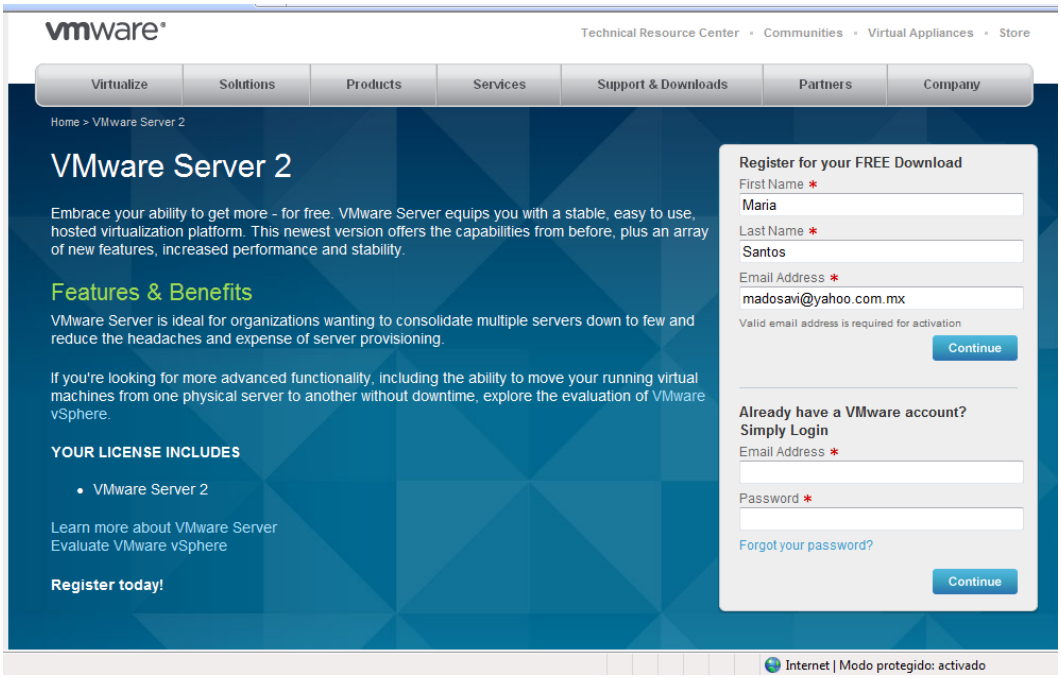

Figura III.15. Creación de cuenta para Registro

- **Instalando vmware server**

Una vez descargado usamos el comando rpm para instalar el vmware server:

```
# rpm –ivh VMware-server-2.0.2-203138.i386.rpm
```

- **Salida**

```
Preparing...                ########################################### [100%]
   1:VMware-server          ########################################### [100%]
```

Se requeriere instalar archivos y paquetes en el servidor

✓ **libXtst-devel:** Paquete de desarrollo X.Org X11 libXtst

```
[root@vmwaretest tmp]# rpm -ivh libXtst-devel-1.0.1-3.1.i386.rpm
warning: libXtst-devel-1.0.1-3.1.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
   1:libXtst-devel          ########################################### [100%]
```

✓ **libXrender-devel**: Paquete de desarrollo X.Org X11 libXrender

```
[root@vmwaretest tmp]# rpm -ivh libXrender-devel-0.9.1-3.1.i386.rpm
warning: libXrender-devel-0.9.1-3.1.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
   1:libXrender-devel       ########################################### [100%]
```

```
[root@vmwaretest tmp]# rpm -ivh mesa-libGL-6.5.1-7.7.el5.i386.rpm
warning: mesa-libGL-6.5.1-7.7.el5.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
        package mesa-libGL-6.5.1-7.7.el5.i386 is already installed
```

- **Instalar xinetd**

Se requiere el servicio/demonio xinetd de internet extendido para utilizar la consola de VMware desde un computador remoto.

```
[root@vmwaretest tmp]# rpm -ivh xinetd-2.3.14-10.el5.i386.rpm
warning: xinetd-2.3.14-10.el5.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
   1:xinetd                 ########################################### [100%]
```

- **Configuración de VMware Server**

Usaremos el script vmware-config.pl para configurar todos los aspectos de VMware.

```
[root@vmwaretest .vnc]# updatedb
[root@vmwaretest .vnc]# locate vmware-config.pl
/usr/bin/vmware-config.pl
Making sure services for VMware Server are stopped.

Stopping VMware autostart virtual machines:
   Virtual machines[FAILED]
Stopping VMware management services:
   VMware Virtual Infrastructure Web Access
   VMware Server Host Agent[FAILED]
Stopping VMware services:
   VMware Authentication Daemon[  OK  ]
   Virtual machine monitor[  OK  ]
You must read and accept the End User License Agreement to continue.
Press enter to display it.

Do you accept? (yes/no)

The answer "" is invalid.  It must be one of "y" or "n".

Do you accept? (yes/no) y

Thank you.

Do you want networking for your virtual machines? (yes/no/help) [yes]

Configuring a bridged network for vmnet0.
Do you want to be able to use NAT networking in your virtual machines? (yes/no)
[yes] no
```

```
Do you want to be able to use host-only networking in your virtual machines?
[no]

Please specify a port for remote connections to use [902]

Please specify a port for standard http connections to use [8222]

Please specify a port for secure http (https) connections to use [8333]

The current administrative user for VMware Server  is ''.  Would you like to
specify a different administrator? [no]

Using root as the VMware Server administrator.

In which directory do you want to keep your virtual machine files?
[/var/lib/vmware/Virtual Machines]

Please enter your 20-character serial number.

Type XXXXX-XXXXX-XXXXX-XXXXX or 'Enter' to cancel:  A2N4M-F216K-U257H-4RPTX

Creating a new VMware VIX API installer database using the tar4 format.

Installing VMware VIX API.

The configuration of VMware Server 2.0.2 build-203138 for Linux for this
running kernel completed successfully.
```

Para terminar la instalación tenemos que introducir un número de serie válido, que habremos conseguido de la página web tal y como describimos anteriormente. Los servicios necesarios para ejecutar VMware se habrán lanzado durante la instalación, por lo que no tenemos que hacer nada más.

- **Módulo 2**

- **Soporte Sistema Operativo de 64 bits**

El uso de sistemas operativos de 64 bits sobre servidores hardware de 64 bits permiten más escalablilidad y soluciones informáticas de alto rendimiento. Además, VMware Server se ejecutará de forma nativa en sistemas operativos Linux host de 64 bits.

Tabla III.2. Arquitectura Soportadas VMware

CPU	Host OS	32-bit Guest OS	64-bit Guest OS
32-bit CPU	32-bit Host OS	Supported	Unsupported
	64-bit Host OS	Unsupported	Unsupported
64-bit CPU	32-bit Host OS	Supported	Supported
	64-bit Host OS	Supported	Supported

- **Compatibilidad**

Al igual que un ordenador físico, un servidor virtual aloja su propio sistema operativo y aplicaciones guest, y dispone de los mismos componentes (placa base, tarjeta VGA, controlador de tarjeta de red, etc.). El resultado de ello es que los servidores virtuales son totalmente compatibles con la totalidad de sistemas operativos x86, aplicaciones y controladores de dispositivos estándar, de modo que se puede utilizar un servidor virtual para ejecutar el mismo software que se puede ejecutar en un ordenador x86 físico.

- **Mayor escalabilidad y flexibilidad**

Podemos utilizar un máximo de 8 GB de RAM por servidor virtual, hasta 10 tarjetas de interfaz de red virtuales, por servidor virtual, la transferencia de datos a mayores velocidades para dispositivos USB 2.0, agregar más discos duros SCSI y controladores a los servidores virtuales en ejecución.

- **Módulo 3**

- **Volume Shadow Copy Service (VSS)**

Tecnología Snapshot utilizado en copias de seguridad del estado de los servidores virtuales para mantener la integridad de los datos de las aplicaciones que se ejecutan dentro de estos.

El soporte de servidor aloja una sola imagen de snapshot por servidor virtual. Para tomar una nueva imagen de snapshot, la imagen de snapshot anterior debe ser sobrescrita.

- **Aislamiento**

Aunque los servidores virtuales pueden compartir los recursos físicos de un único ordenador, permanecen completamente aisladas unas de otras, como si se tratara de máquinas independientes. Si, por ejemplo, hay cuatro máquinas virtuales en un único servidor físico y falla una de ellas, las otras tres siguen estando disponibles. El aislamiento es un factor importante que explica por qué la disponibilidad y protección de las aplicaciones que se ejecutan en un entorno virtual es muy superior a las aplicaciones que se ejecutan en un sistema tradicional no virtualizado.

- **Encapsulamiento**

Un servidor virtual es básicamente un contenedor de software que ata o “encapsula” un conjunto completo de recursos de hardware virtuales, así como un sistema operativo y todas sus aplicaciones, dentro de un paquete de software. El encapsulamiento hace a los servidores virtuales extraordinariamente portables y fáciles de gestionar. Por ejemplo, podemos mover y copiar un servidor virtual de un lugar a otro como lo haría con cualquier otro archivo de software, o guardar un servidor virtual en cualquier medio de almacenamiento de datos estándar, desde una memoria USB de tamaño de bolsillo hasta las redes de área de almacenamiento (SAN) de una empresa.

- **Módulo 4**

- **Administración de recursos hardware**

Usamos el ***Add Hardware wizard*** para añadir nuevo hardware al servidor virtual, el mismo deber ser apagado para añadir cualquier tipo de hardware.

Para iniciar el Add Hardware wizard, se detallan los siguientes pasos:

1.- Seleccionar el servidor virtual del panel Inventory.

2.- Si se requiere cambiar los valores debemos asegurarnos que el servidor virtual esté apagado.

3.- En la sección de Commands de la pestaña Summary, hacer click en Add Hardware.

El wizard Add Hardware.

4.- Para añadir hardware a un servidor virtual existente:

- ✓ Hard Disks — See “Adding a Hard Disk to a Virtual Machine”.
- ✓ Network Adapters — See “Adding a Network Adapter to a Virtual Machine”.
- ✓ CD/DVD Drives — See “Adding a CD/DVD Drive to a Virtual Machine”.
- ✓ Floppy Drives — See “Adding a Floppy Drive to a Virtual Machine”.
- ✓ Passthrough (Generic) SCSI Devices — See “Adding a Passthrough (Generic) SCSI Device to a Virtual Machine”.
- ✓ USB Controller — See “Adding a USB Controller to a Virtual Machine”.
- ✓ Sound Adapter — See “Adding a Sound Adapter to a Virtual Machine”.
- ✓ Serial Ports — See “Adding a Serial Port to a Virtual Machine”.

✓ Parallel Ports — See “Adding a Parallel Port to a Virtual Machine”.

- **Discos duros virtuales**

Soporte para añadir, editar, y remover discos duros virtuales y configuración de valores.

a) Para añadir discos virtuales: Los discos virtuales son almacenados como archivos en un almacenamiento de datos (datastore). La ubicación del datastore puede ser un archivo local del sistema, un almacenamiento CIFS (Windows) o un archivo del sistema montado NFS (Linux).

1.- Desde Add Hardware wizard, hacer clic en Hard Disk.

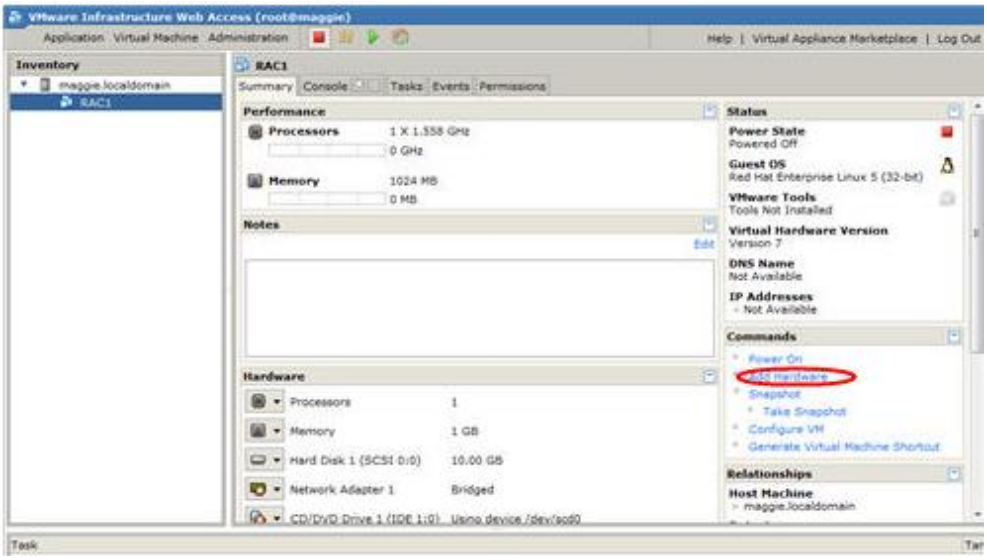

Figura III.16. Vmware Server: Add Hardware

2.- Seguidamente en la página de Hard Disk, seleccionamos una de las siguientes opciones:

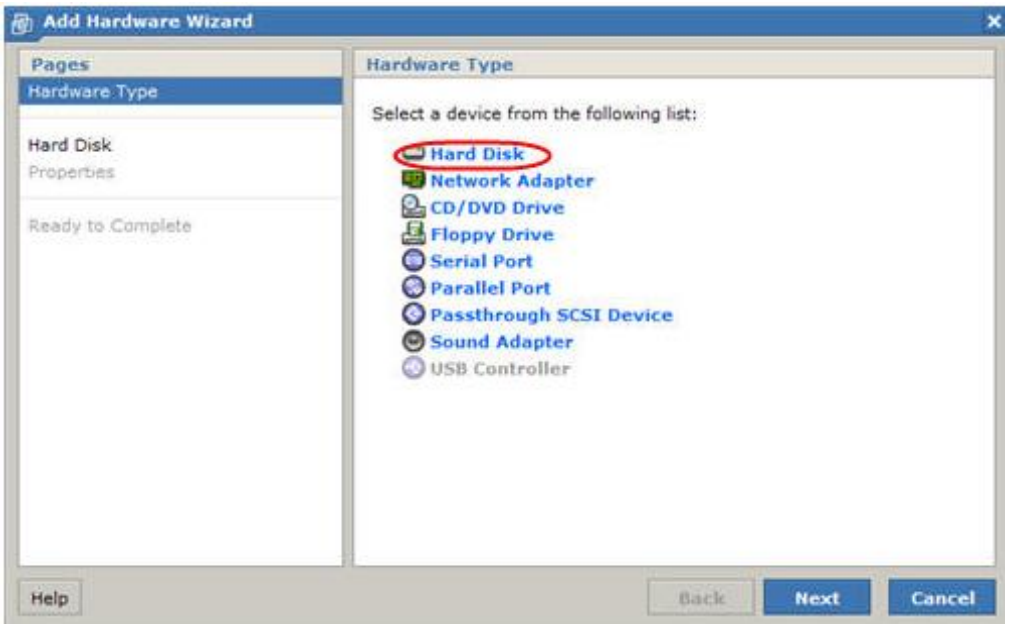

Figura III.17. Vmware Server: Hardware Type

a. Create a New Virtual Disk, Seleccionamos para añadir un disco duro virtual en blanco al servidor virtual.

b. Use an Existing Virtual Disk, Seleccionamos si deseamos reutilizar o compartir un disco duro virtual que ya ha sido creado.

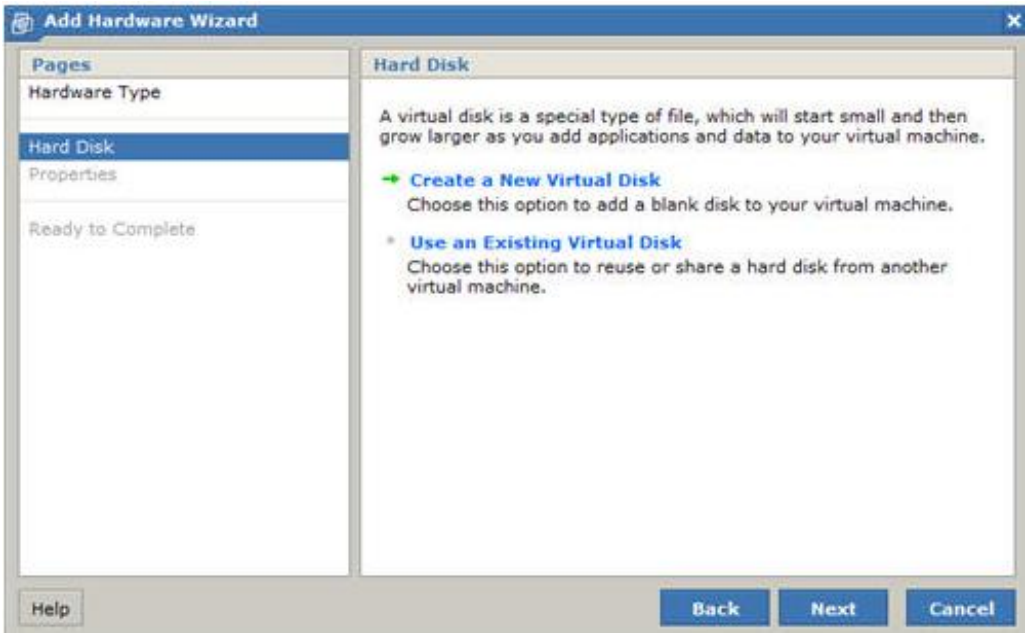

Figura III.18. Vmware Server: Hard Disk

3.- Realizar cualquier cambio en los valores por defecto en la pantalla de Properties, y pulsar en Next.

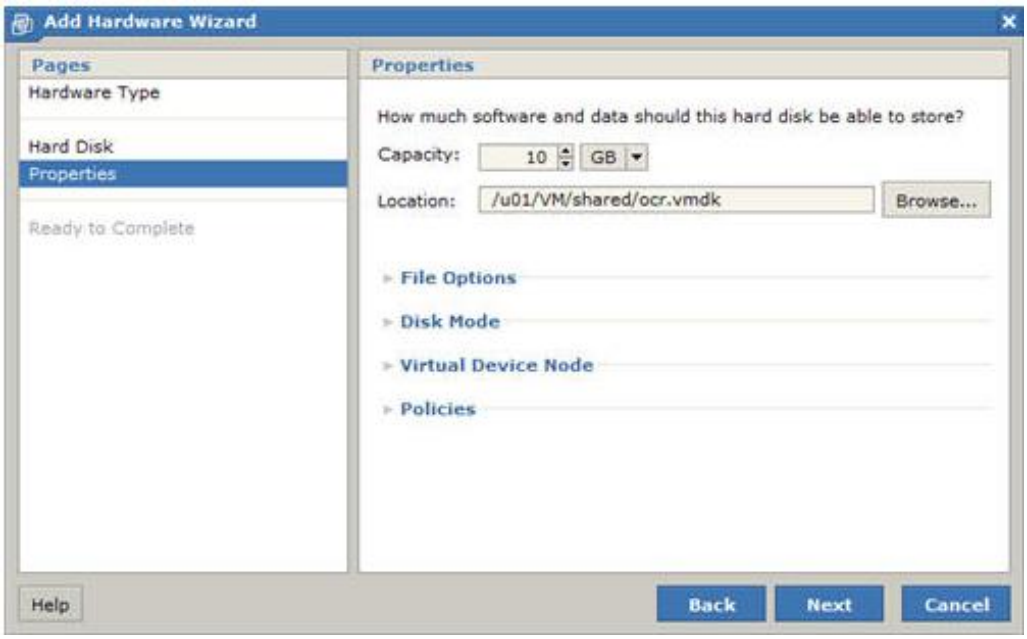

Figura III.19. Vmware Server: Hard Disk Properties

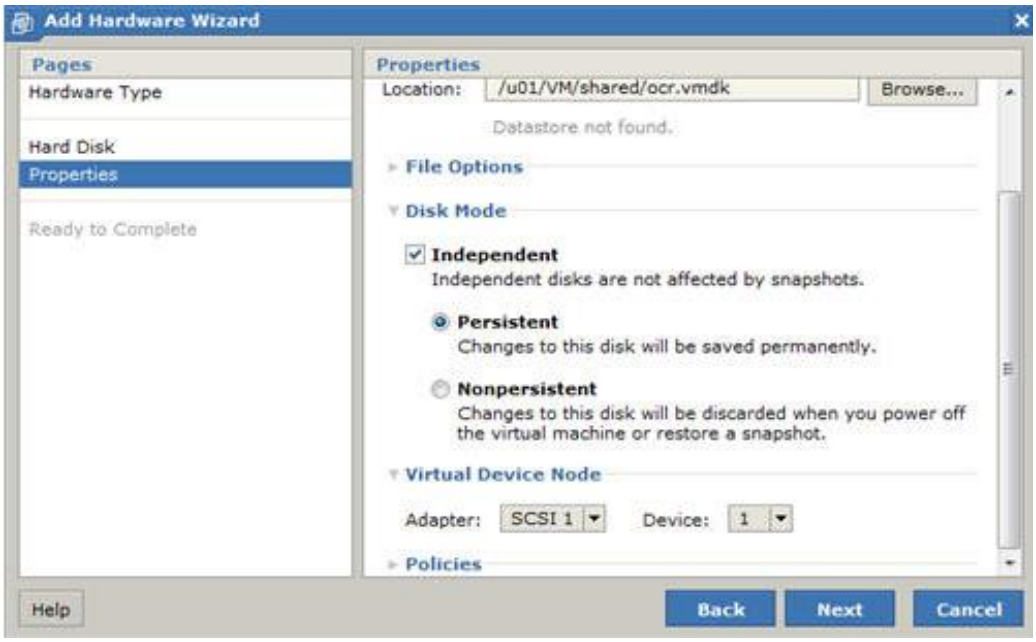

Figura III.20. Vmware Server: Hard Disk Properties Disk Mode

4.- Revisamos el resumen de la configuración y pulsamos en Finish para completar con el wizard.

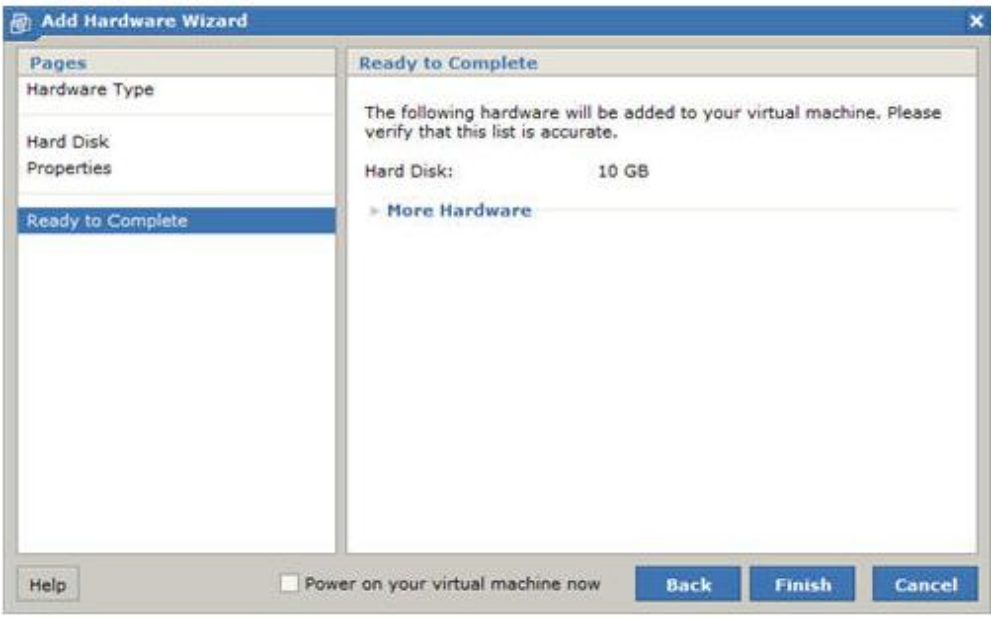

Figura III.21. Vmware Server: Añadido Disco Duro Completado

El wizard crea el nuevo disco virtual, luego el disco virtual es visualizado en el servidor virtual del guest como un nuevo disco duro en blanco. El disco virtual se particionará y formaterá con cualquier herramienta utilitaria del sistema operativo del guest.

b) Para editar discos virtuales: A los discos virtuales existentes podemos cambiar sus valores, tal como se indica a continuación:

1.- Seleccionar el servidor virtual del panel Inventory.

2.- Si se requiere cambiar los valores debemos asegurarnos que el servidor virtual esté apagado.

3.- En la sección de Hardware de la pestaña de Summary, click en el disco duro para modificar y seleccionamos Edit.

c) Para remover discos virtuales: A los discos virtuales existentes podemos removerlos de un servidor virtual:

1.- Seleccionar el servidor virtual del panel Inventory.

2.- Si se requiere cambiar los valores debemos asegurarnos que el servidor virtual esté apagado.

3.- En la sección de Hardware de la pestaña de Summary, click en el disco duro que deseamos remover y seleccionamos una de las siguientes opciones:

Remover, Remueve el disco duro del servidor virtual.

Delete from Disk, Remueve el disco duro desde el servidor virtual y elimina los archivos de disco asociados desde el host del sistema.

4.- Una caja de diálogo nos preguntará la confirmación que deseamos remover el disco. Si en verdad deseamos remover éste, pulsamos en Yes.

El disco virtual será removido.

El procedimiento es similar el cualquier hardware que se desee añadir.

- **Módulo 5**

- **Interfaz de gestión Web**

Una nueva interfaz web de usuario proporciona una interfaz simple, intuitiva y productiva de tal manera para se pueda administrar los servidores virtuales. Con la nueva consola remota VMware, acceder a las consolas independientes de los servidores virtuales mediante la gestión basada en la interfez Web.

✓ Creación de dominios guest

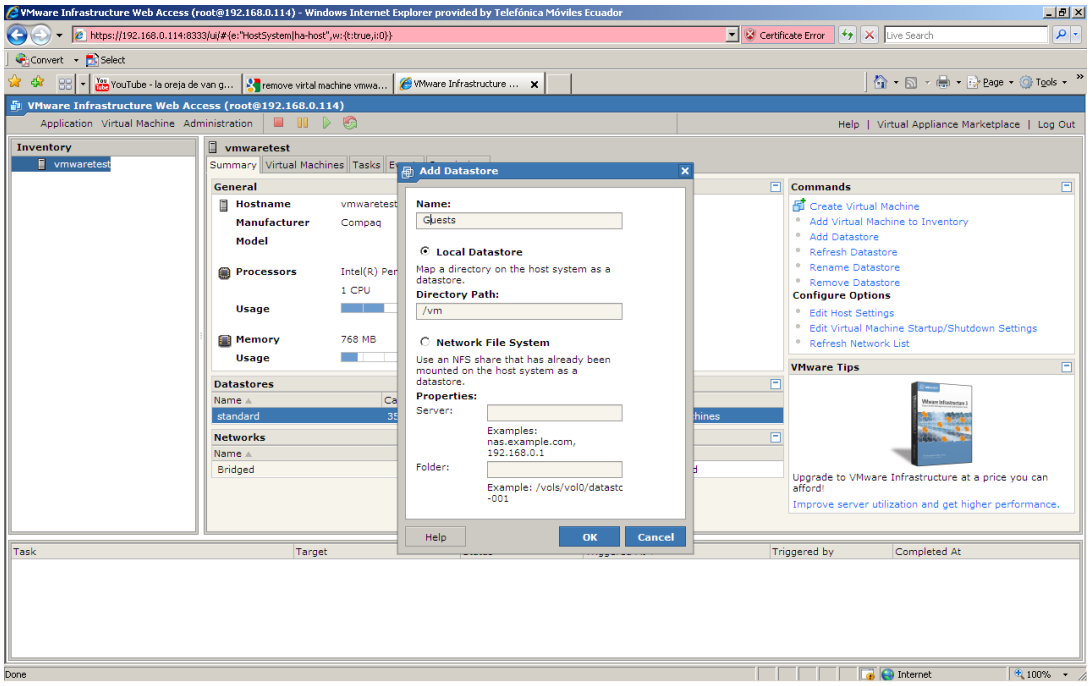

Figura III.22. Vmware Server: Añadir un datastore

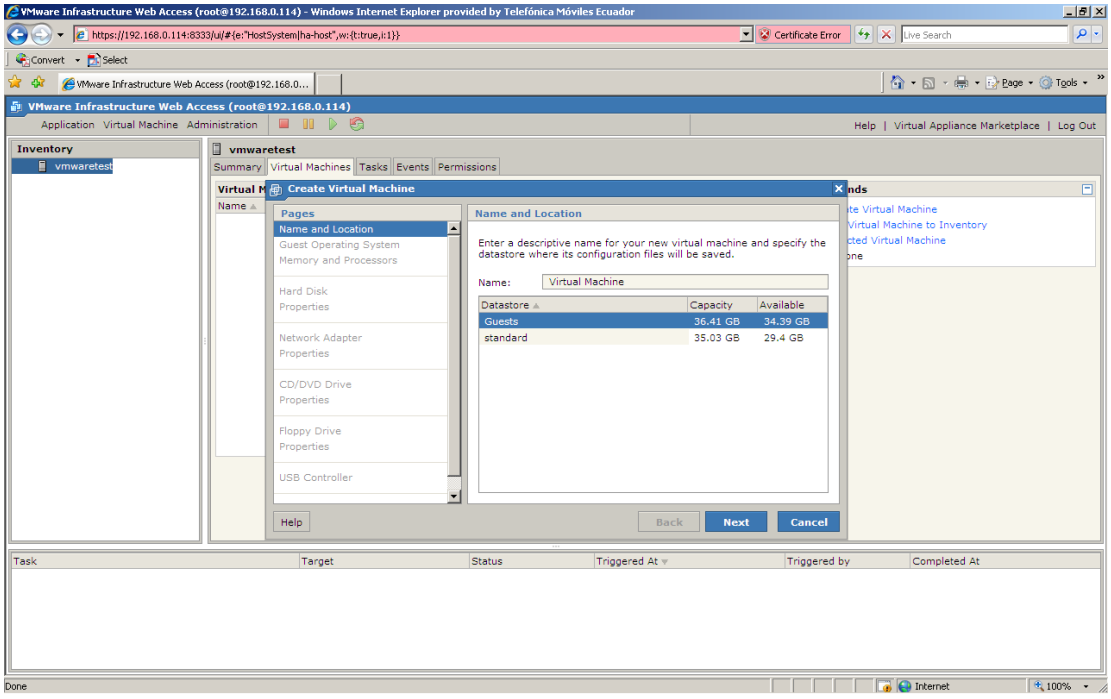

Figura III.23. Vmware Server: Nombre y Ubicación del Servidor Virtual

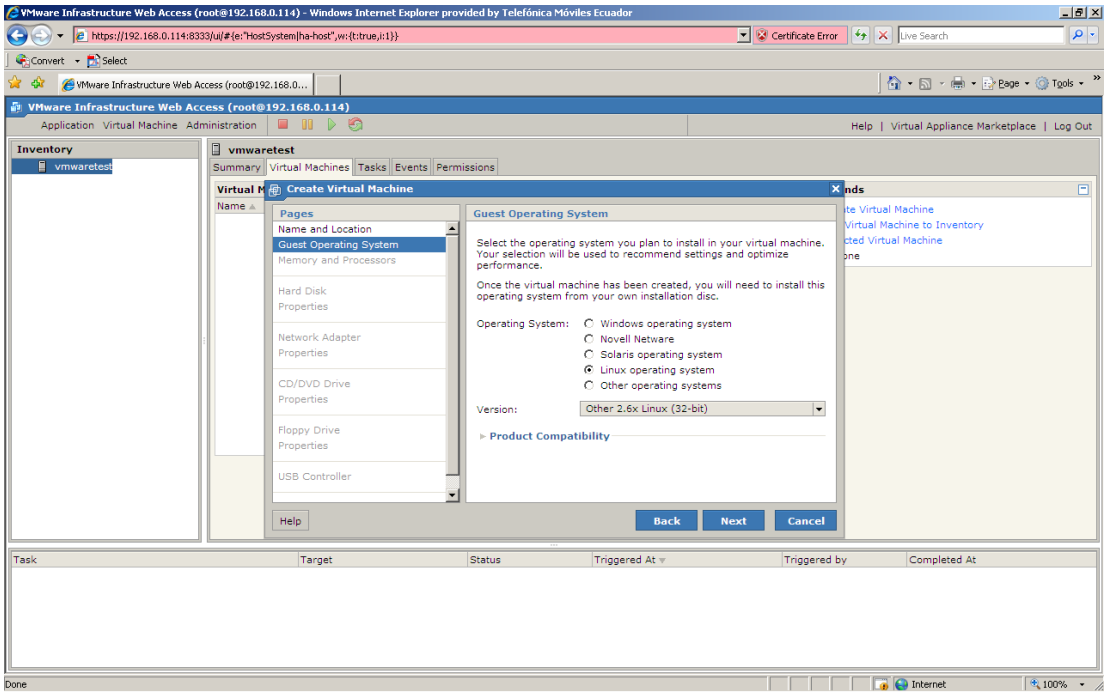

Figura III.24. Vmware Server: Sistema Operativo para el Guest

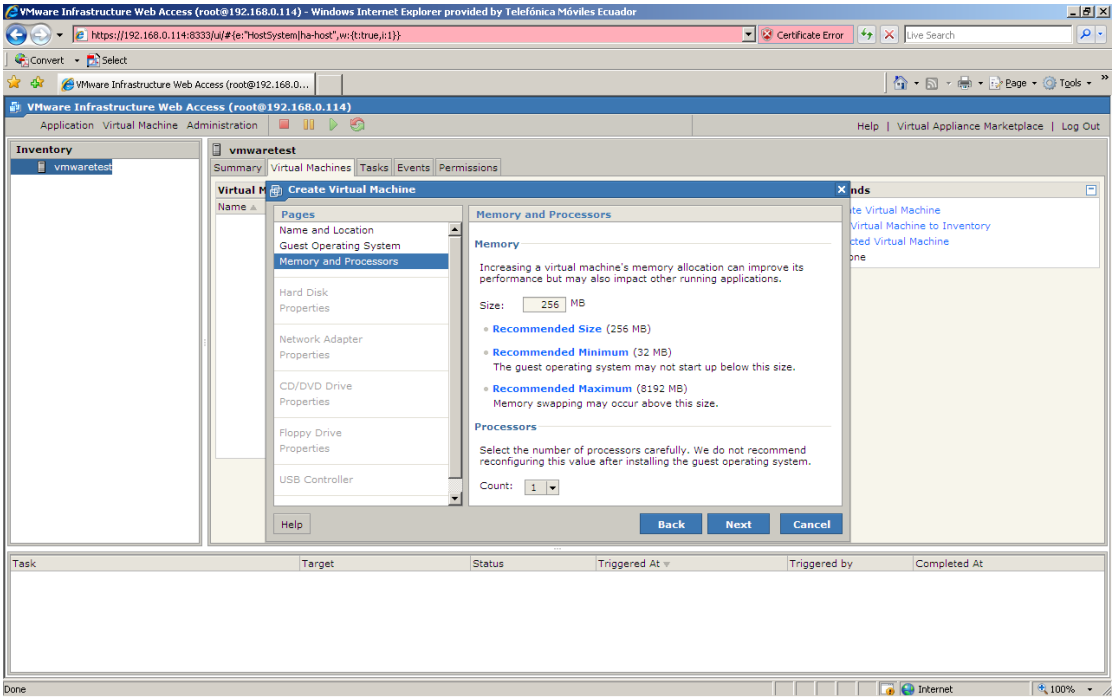

Figura III.25. Vmware Server: Asignación de Memoria y Procesador

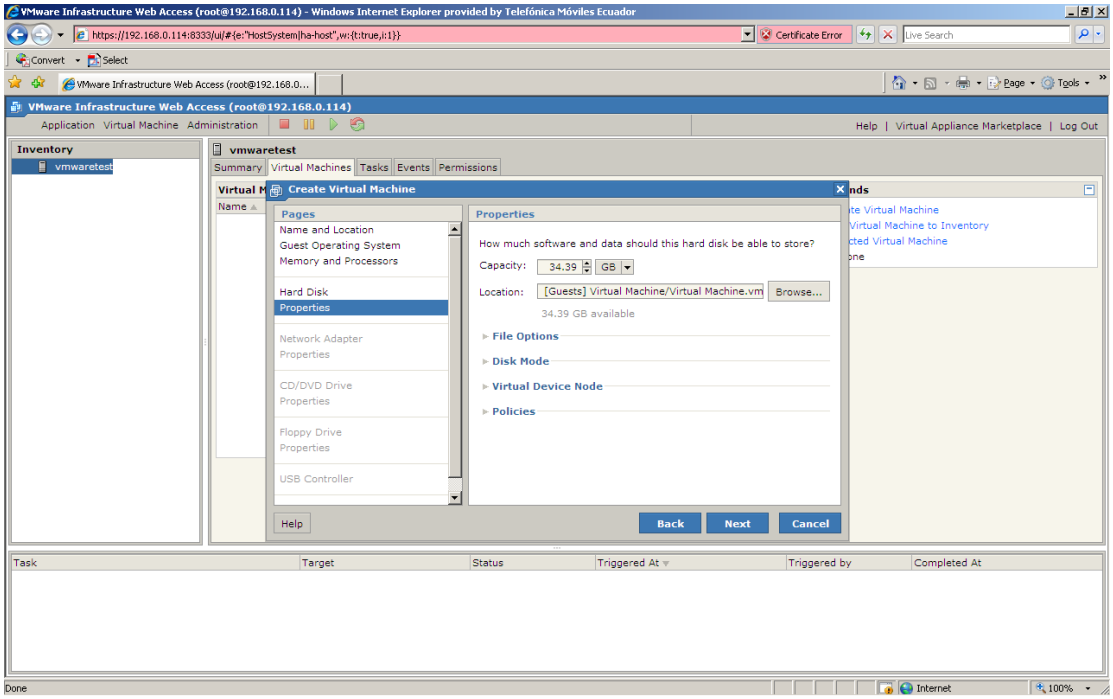

Figura III.26. Vmware Server: Creación de Disco Virtual Propiedades

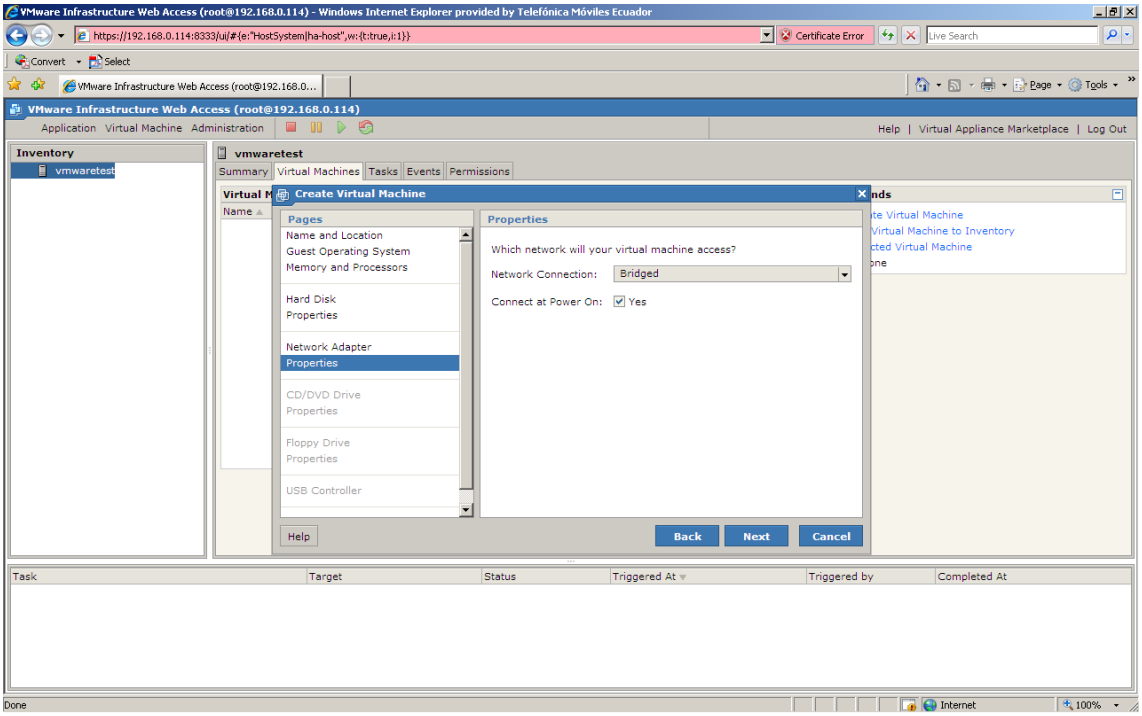

Figura III.27. Vmware Server: Asignación de Adptador de Red Propiedades

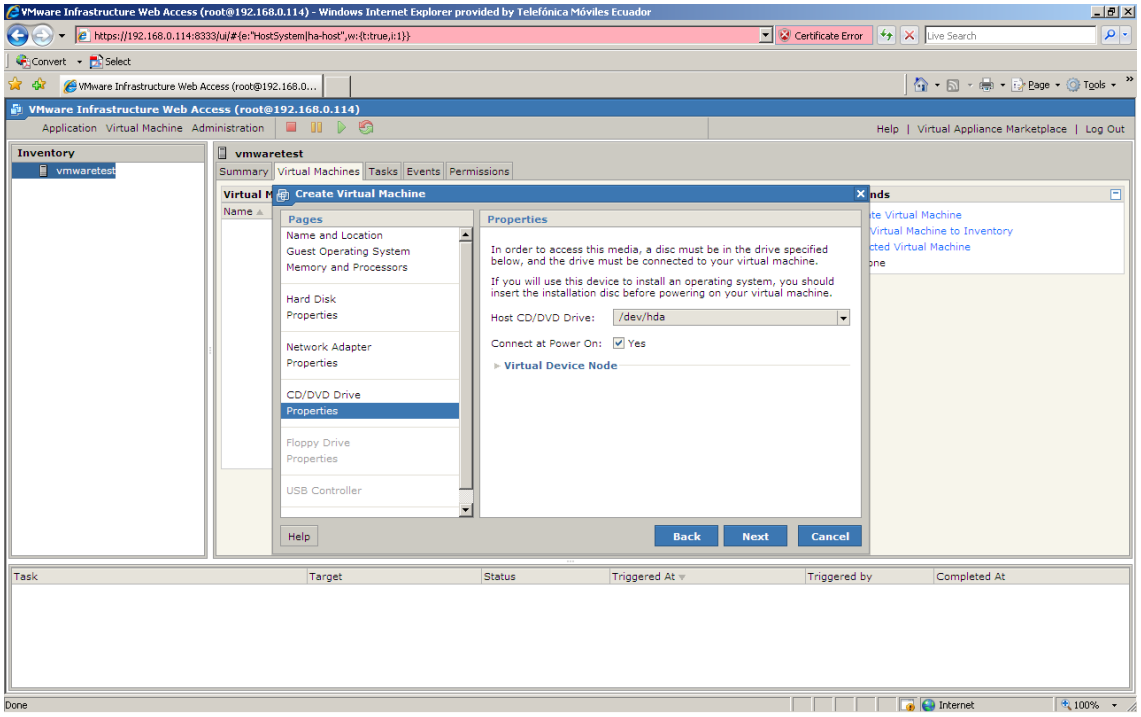

Figura III.28. Vmware Server: Asignación de Drive CD/DVD Propiedades

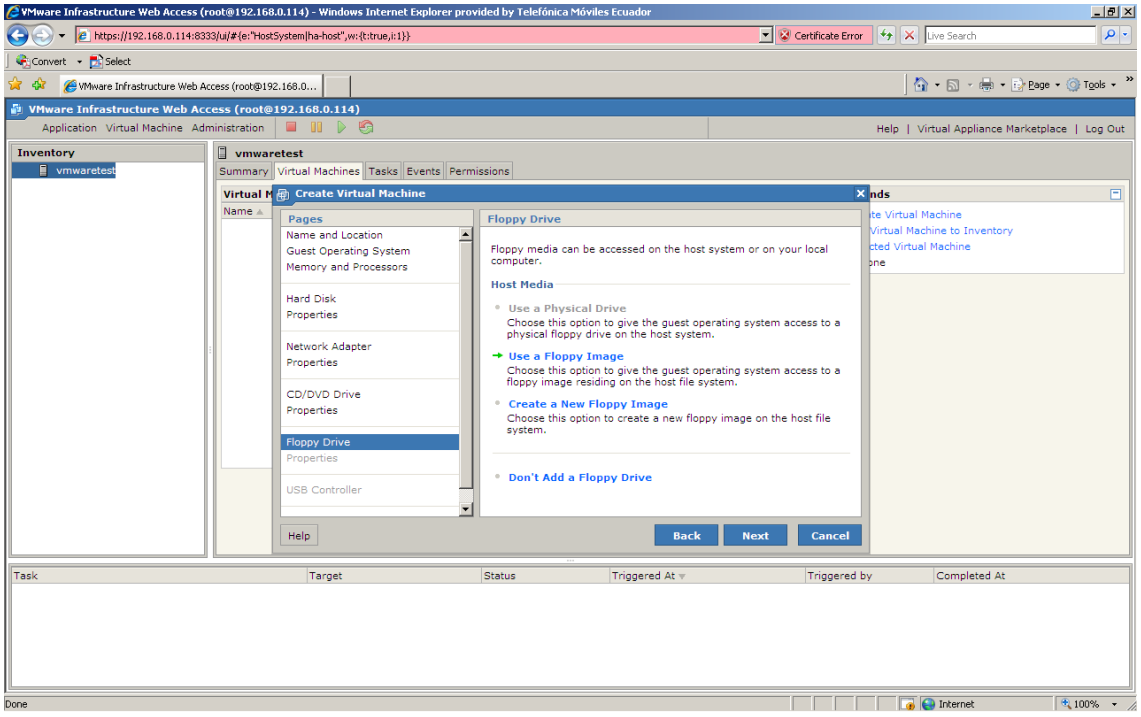

Figura III.29. Vmware Server: Asignación de Drive Floppy

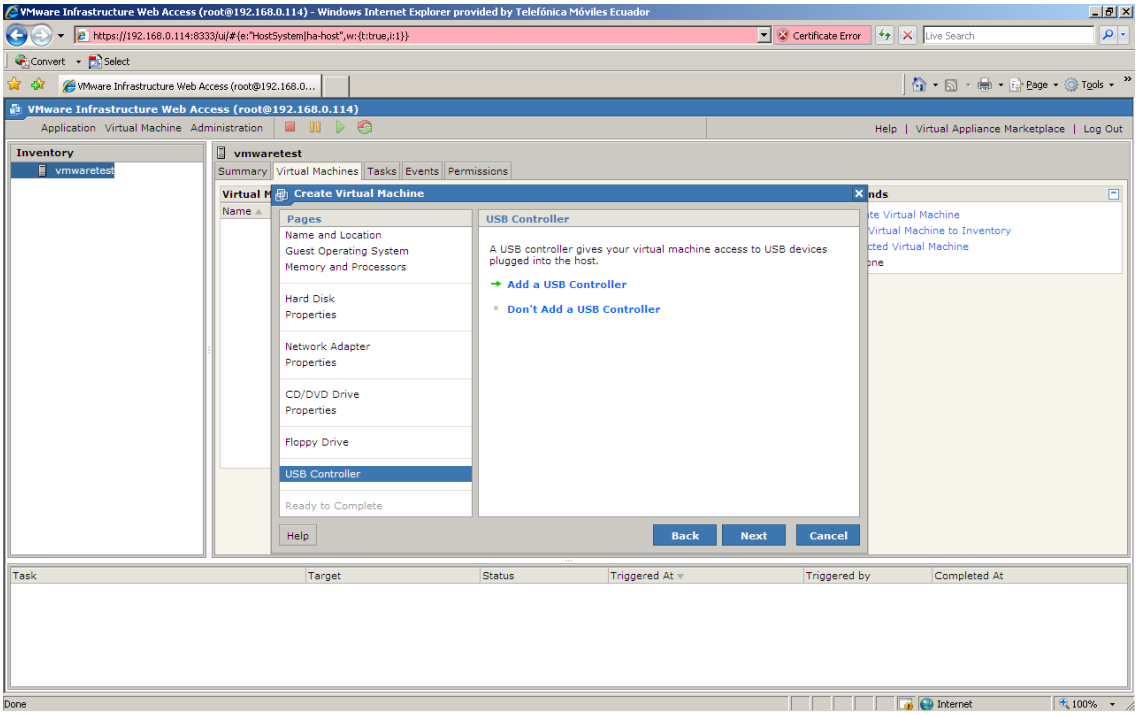

Figura III.30. Vmware Server: Contraladores USB

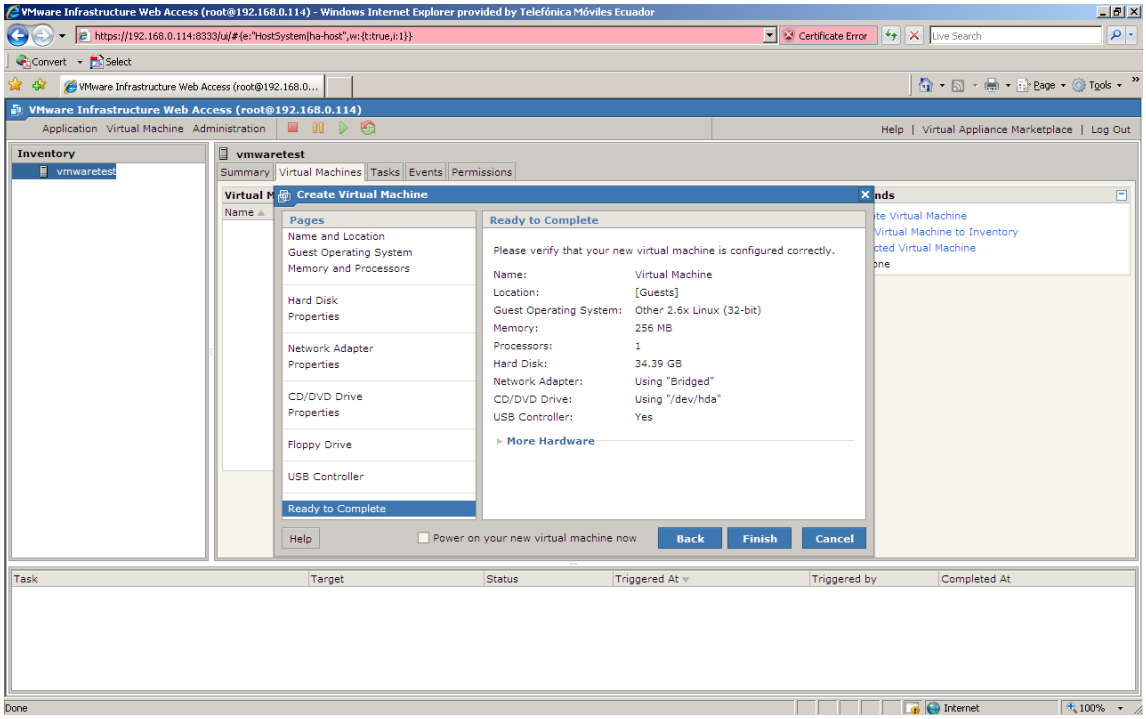

Figura III.31. Vmware Server: Creación del Servidor Virtual Completado

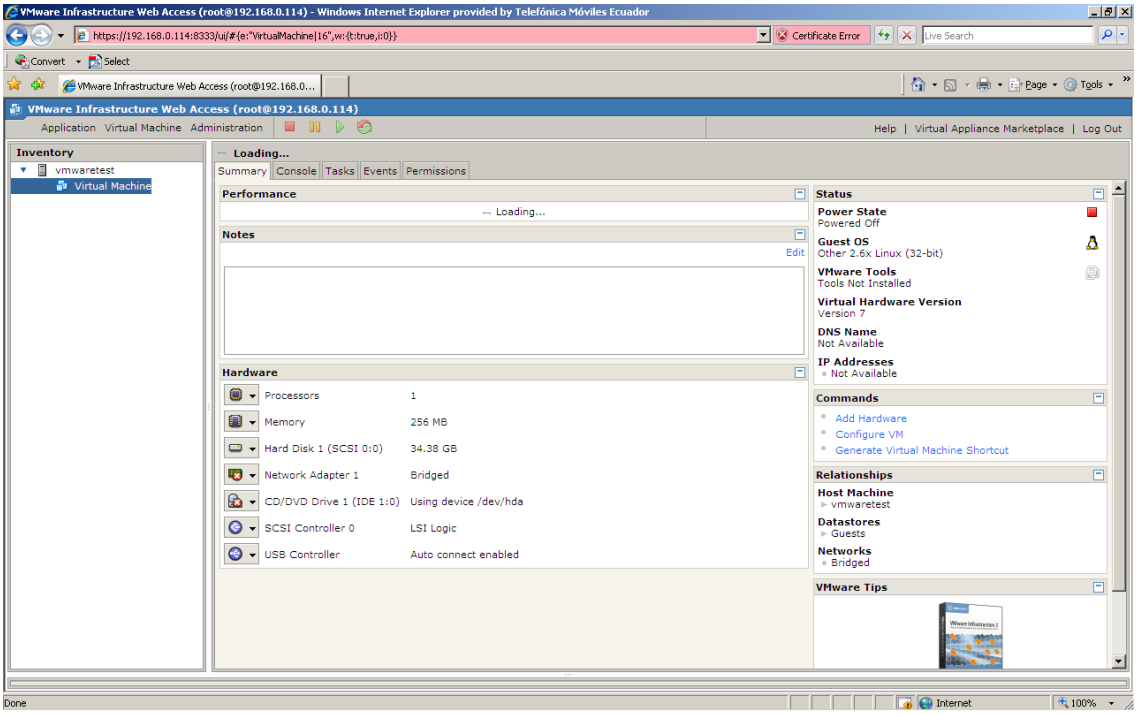

Figura III.32. Vmware Server: Resumen Hardware Servidor Virtual

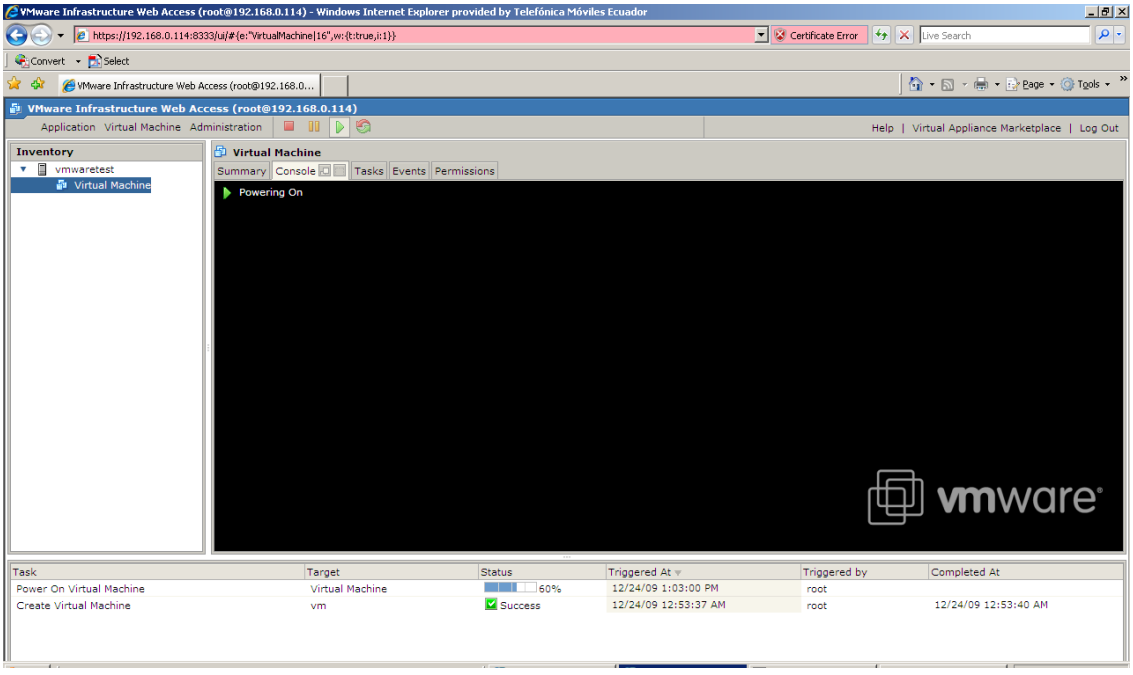

Figura III.33. Vmware Server: Encendido de Servidor Virtual

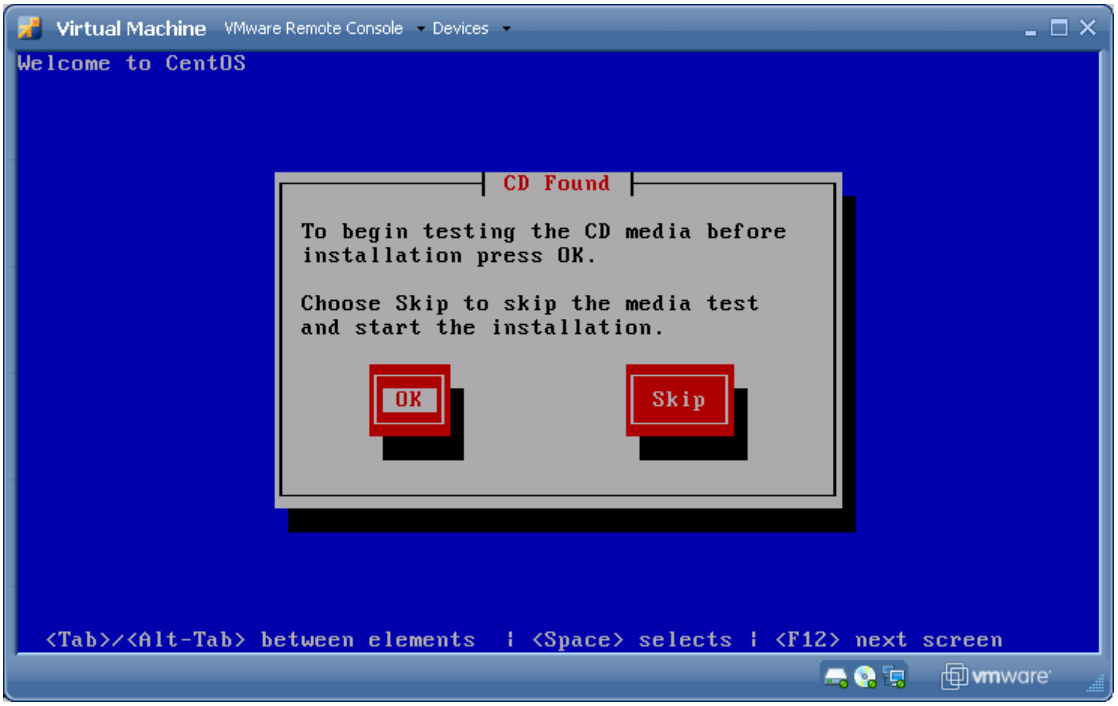

Figura III.34. Vmware Server: Instalación Sistema Operativo en Servidor Virtual

- **Navegadores son compatibles con el acceso a la administración Web de VMware Infraestructure**

Tanto en Windows como Linux, Mozilla Firefox 2 y 3 son compatibles. Además, en Windows, Microsoft Internet Explorer 6 y 7, también son compatibles, pero Internet Explorer 7 recomendado.

- **Consola independiente del servidor virtual**

El acceso a las consolas es independiente de la interfaz VMware Infrastructure Web Access Management del servidor virtual.

La consola remota de VMware permite el acceso a las consolas de los servidores virtuales independiente de la interfaz de gestión de acceso Web de VMware Infrastructure. Una ventana separada es abierta para la consola del servidor virtual que es inicializada y la ventana se puede cambiar el tamaño como sea requerido. Por último, consola remota VMware puede ser acceder directamente desde un acceso directo del escritorio para proporcionar acceso independiente e instantáneo al servidor virtual de la consola.

- **VMware Server es disponible para los clientes de todo el mundo**

Sí. Sin embargo, el instalador, la interfaz de gestión de acceso Web del VMware Infrastructure, y documentación para VMware Server sólo están disponibles en Inglés.

- **Módulo 6**

Procedemos con la instalación del utilitario stress tal como se indica a continuación:

```
[root@cosvmtest tmp]# tar xvf stress-1.0.2.tar.tar
[root@cosvmtest tmp]# cd stress-1.0.
[root@cosvmtest stress-1.0.2]#   ./configure && make && sudo make install
stress: info: [5705] successful run completed in 305s
sytess
```

Los parámetros con los que vamos a evaluar será un promedio de cuatro paramétros que se impone en el sistema especificando -c procesos limitados por CPU, -i de I/O-bound procesos, -m procesos asignados de memoria y –d procesos de disco en un tiempo de ejecución de 5 minutos. A continuación la ejecución del mismo:

```
 [root@cosvmtest stress-1.0.2]# date;stress -c 1 -i 1 -m 1 -d 1 --verbose --timeout 5m;date
Thu Dec 24 10:32:00 ECT 2009
stress: info: [5705] dispatching hogs: 1 cpu, 1 io, 1 vm, 1 hdd
stress: dbug: [5705] using backoff sleep of 12000us
stress: dbug: [5705] setting timeout to 300s
stress: dbug: [5705] --> hogcpu worker 1 [5706] forked
stress: dbug: [5705] --> hogio worker 1 [5707] forked
stress: dbug: [5705] --> hogvm worker 1 [5708] forked
stress: dbug: [5708] allocating 268435456 bytes ...
stress: dbug: [5708] touching bytes in strides of 4096 bytes ...
```

```
stress: dbug: [5705] --> hoghdd worker 1 [5709] forked
stress: dbug: [5709] seeding 1048575 byte buffer with random data
stress: dbug: [5709] opened ./stress.9RWIvT for writing 1073741824 bytes
stress: dbug: [5709] unlinking ./stress.9RWIvT
stress: dbug: [5709] fast writing to ./stress.9RWIvT
stress: dbug: [5709] slow writing to ./stress.9RWIvT
stress: dbug: [5709] closing ./stress.9RWIvT after 1073741824 bytes
stress: dbug: [5709] opened ./stress.TkZkt6 for writing 1073741824 bytes
stress: dbug: [5709] unlinking ./stress.TkZkt6
stress: dbug: [5709] fast writing to ./stress.TkZkt6
stress: dbug: [5709] slow writing to ./stress.TkZkt6
stress: dbug: [5709] closing ./stress.TkZkt6 after 1073741824 bytes
stress: dbug: [5709] opened ./stress.N4SpPH for writing 1073741824 bytes
stress: dbug: [5709] unlinking ./stress.N4SpPH
stress: dbug: [5709] fast writing to ./stress.N4SpPH
stress: dbug: [5705] <-- worker 5706 signalled normally
stress: dbug: [5705] <-- worker 5708 signalled normally
stress: dbug: [5705] <-- worker 5709 signalled normally
stress: dbug: [5705] <-- worker 5707 signalled normally
stress: info: [5705] successful run completed in 305s
Thu Dec 24 10:37:08 ECT 2009
```

```
[root@cosvmtest stress-1.0.2]#  date;stress -c 2 -i 2 -m 2 -d 2 --verbose --timeout 5m;date
Thu Dec 24 10:38:06 ECT 2009
stress: info: [5916] dispatching hogs: 2 cpu, 2 io, 2 vm, 2 hdd
stress: dbug: [5916] using backoff sleep of 18000us
stress: dbug: [5916] setting timeout to 300s
stress: dbug: [5916] --> hogcpu worker 2 [5917] forked
stress: dbug: [5916] --> hogio worker 2 [5918] forked
stress: dbug: [5916] --> hogvm worker 1 [5919] forked
stress: dbug: [5916] --> hoghdd worker 1 [5920] forked
stress: dbug: [5916] using backoff sleep of 6000us
stress: dbug: [5916] setting timeout to 300s
stress: dbug: [5916] --> hogcpu worker 1 [5921] forked
stress: dbug: [5916] --> hogio worker 1 [5922] forked
stress: dbug: [5919] allocating 268435456 bytes ...
stress: dbug: [5919] touching bytes in strides of 4096 bytes ...
stress: dbug: [5920] seeding 1048575 byte buffer with random data
stress: dbug: [5920] opened ./stress.UEbMth for writing 1073741824 bytes
stress: dbug: [5920] unlinking ./stress.UEbMth
stress: dbug: [5920] fast writing to ./stress.UEbMth
stress: dbug: [5920] slow writing to ./stress.UEbMth
stress: dbug: [5920] closing ./stress.UEbMth after 1073741824 bytes
stress: dbug: [5920] opened ./stress.BLZIdu for writing 1073741824 bytes
stress: dbug: [5920] unlinking ./stress.BLZIdu
stress: dbug: [5920] fast writing to ./stress.BLZIdu
stress: dbug: [5916] <-- worker 5919 signalled normally
stress: dbug: [5916] <-- worker 5917 signalled normally
stress: dbug: [5916] <-- worker 5921 signalled normally
stress: dbug: [5916] <-- worker 5920 signalled normally
stress: dbug: [5916] <-- worker 5918 signalled normally
stress: dbug: [5916] <-- worker 5922 signalled normally
stress: info: [5916] successful run completed in 306s
Thu Dec 24 10:43:13 ECT 2009
```

```
[root@cosvmtest ~]# date;stress -c 3 -i 3 -m 3 -d 3 --verbose --timeout 5m;date
Thu Dec 24 11:16:51 ECT 2009
stress: info: [13911] dispatching hogs: 3 cpu, 3 io, 3 vm, 3 hdd
stress: dbug: [13911] using backoff sleep of 36000us
stress: dbug: [13911] setting timeout to 300s
stress: dbug: [13911] --> hogcpu worker 3 [13912] forked
stress: dbug: [13911] --> hogio worker 3 [13913] forked
stress: dbug: [13911] --> hogvm worker 3 [13914] forked
stress: dbug: [13911] --> hoghdd worker 3 [13915] forked
stress: dbug: [13911] using backoff sleep of 24000us
stress: dbug: [13911] setting timeout to 300s
stress: dbug: [13911] --> hogcpu worker 2 [13916] forked
stress: dbug: [13911] --> hogio worker 2 [13917] forked
stress: dbug: [13911] --> hogvm worker 2 [13918] forked
```

```
stress: dbug: [13911] --> hoghdd worker 2 [13919] forked
stress: dbug: [13914] allocating 268435456 bytes ...
stress: dbug: [13914] touching bytes in strides of 4096 bytes ...
stress: dbug: [13915] seeding 1048575 byte buffer with random data
stress: dbug: [13911] using backoff sleep of 12000us
stress: dbug: [13911] setting timeout to 300s
stress: dbug: [13911] --> hogcpu worker 1 [13920] forked
stress: dbug: [13911] --> hogio worker 1 [13921] forked
stress: dbug: [13911] --> hogvm worker 1 [13922] forked
stress: dbug: [13918] allocating 268435456 bytes ...
stress: dbug: [13918] touching bytes in strides of 4096 bytes ...
stress: dbug: [13919] seeding 1048575 byte buffer with random data
stress: dbug: [13911] --> hoghdd worker 1 [13923] forked
stress: dbug: [13919] opened ./stress.Utj6xp for writing 1073741824 bytes
stress: dbug: [13919] unlinking ./stress.Utj6xp
stress: dbug: [13919] fast writing to ./stress.Utj6xp
stress: dbug: [13922] allocating 268435456 bytes ...
stress: dbug: [13922] touching bytes in strides of 4096 bytes ...
stress: dbug: [13923] seeding 1048575 byte buffer with random data
stress: dbug: [13923] opened ./stress.huGRKD for writing 1073741824 bytes
stress: dbug: [13923] unlinking ./stress.huGRKD
stress: dbug: [13923] fast writing to ./stress.huGRKD
stress: dbug: [13915] opened ./stress.4nMaKX for writing 1073741824 bytes
stress: dbug: [13915] unlinking ./stress.4nMaKX
stress: dbug: [13915] fast writing to ./stress.4nMaKX
stress: FAIL: [13911] (420) <-- worker 13912 got signal 9
stress: WARN: [13911] (422) now reaping child worker processes
stress: FAIL: [13911] (420) <-- worker 13913 got signal 9
stress: WARN: [13911] (422) now reaping child worker processes
stress: FAIL: [13911] (420) <-- worker 13914 got signal 9
stress: WARN: [13911] (422) now reaping child worker processes
stress: dbug: [13911] <-- worker 13916 reaped
stress: dbug: [13911] <-- worker 13922 reaped
stress: dbug: [13911] <-- worker 13918 reaped
stress: dbug: [13911] <-- worker 13921 reaped
stress: dbug: [13911] <-- worker 13920 reaped
stress: dbug: [13911] <-- worker 13919 reaped
stress: dbug: [13911] <-- worker 13923 reaped
stress: dbug: [13911] <-- worker 13915 reaped
stress: dbug: [13911] <-- worker 13917 reaped
stress: FAIL: [13911] (456) failed run completed in 209s
Thu Dec 24 11:20:21 ECT 2009
```

Después de la ejecuación del stress se tiene el siguiente análisis respecto al consumo de: load average, tasks running, task sleeping, cpu, memoria y swap:

- **LOAD AVERAGE**

Con una carga inicial de 1 todas las tareas de stress finaliza bien de igual manera para la carga 2; cuando se aumenta la carga a 3, los procesos empiezan a encolarse y no logra finalizar la prueba de stress, empezando a saturarse y colapsar la ejecución.

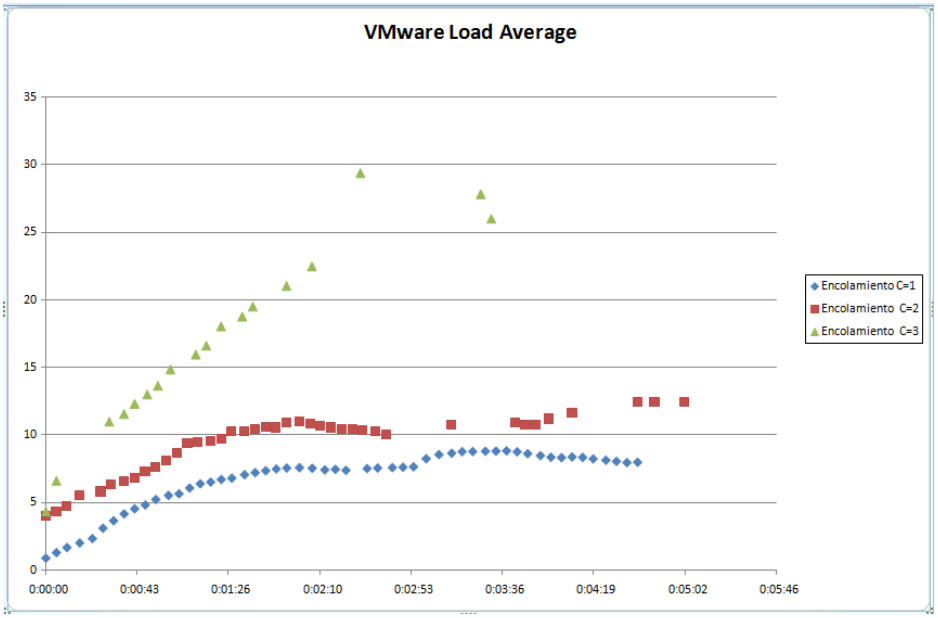

Gráfico III.1. VMware: Load Average

TASKS - RUNNING

Con una carga final de hasta 3 en la utilización del utilitario stress para los procesos CPU, I/O, memoria y disco, el número máximo de procesos que llega a alcanzar es pasado de 25 en el tiempo de ejecución y colapsa inmediantamente.

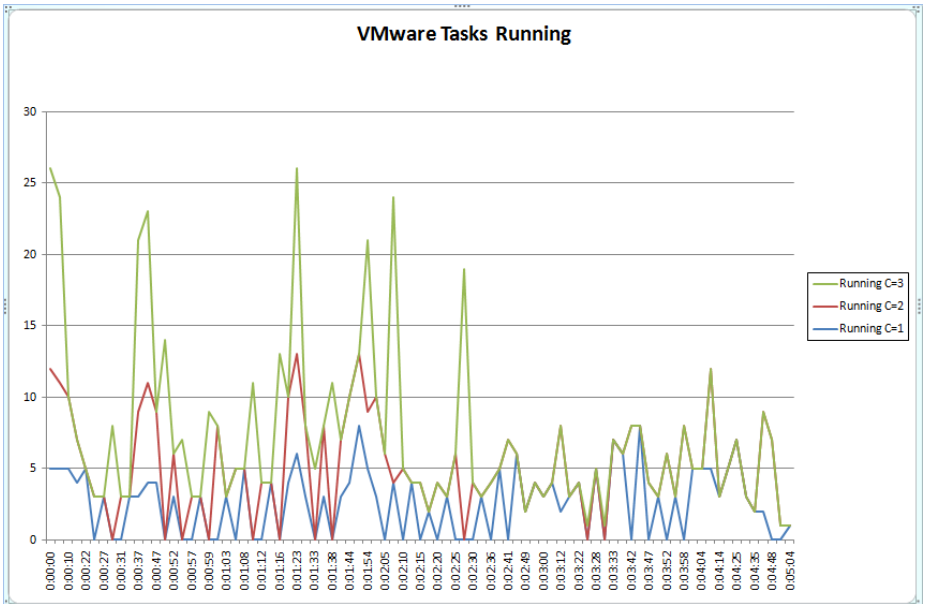

Gráfico III.2. VMware: Tasks - Running

- **TASKS - SLEEPING**

Los procesos que están esperando para ser ejecutados se encuentran en un rango de pasado 50 a 250 debido al fallo en la ejecución del utilitario stress con una carga final de hasta 3 en utilización de procesos CPU, I/O, memoria y disco.

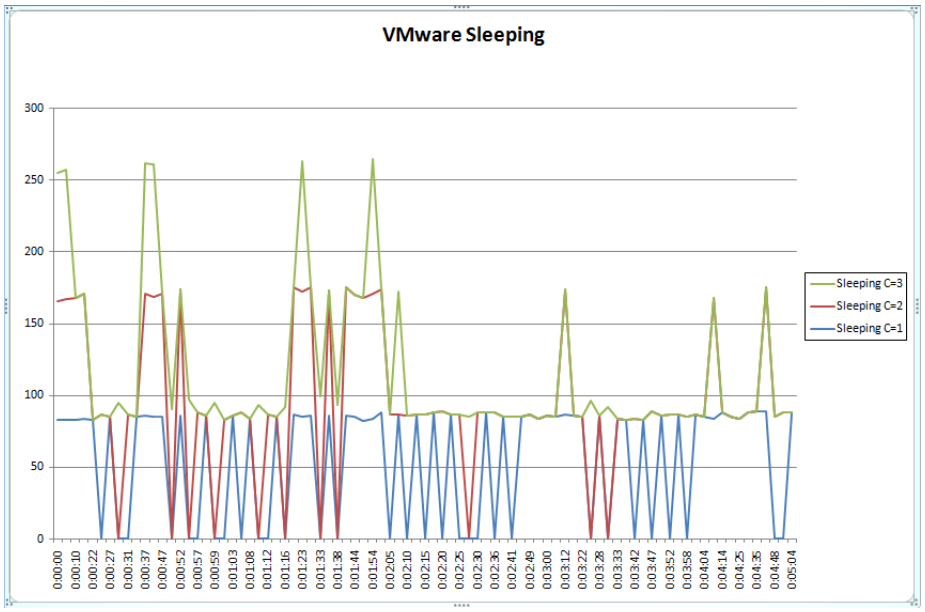

Gráfico III.3. VMware: Tasks - Sleeping

- **CPU**

El consumo del recurso CPU llega a una utilización del 100% con los tres valores de parámetros CPU, I/O, memoria y disco y con el parámetro 3 no logra finalizar la ejecución del utilitario stress.

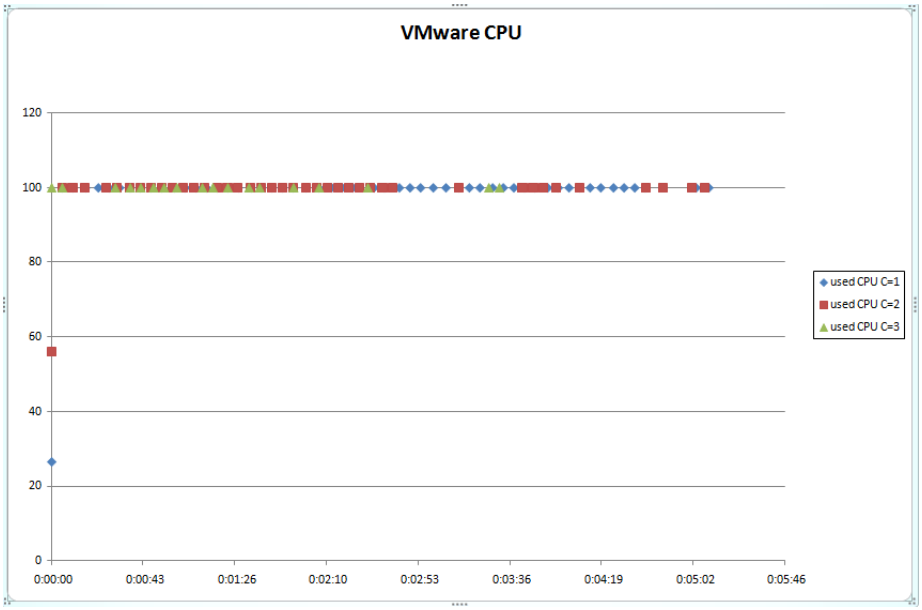

Gráfico III.4. VMware: CPU

- **MEMORIA**

La utilización del recurso memoria llega a su utilización máxima del total de la memoria asignada en kilobytes (kb) 255548k en tiempo en el que falla la ejecución del utilitario stress.

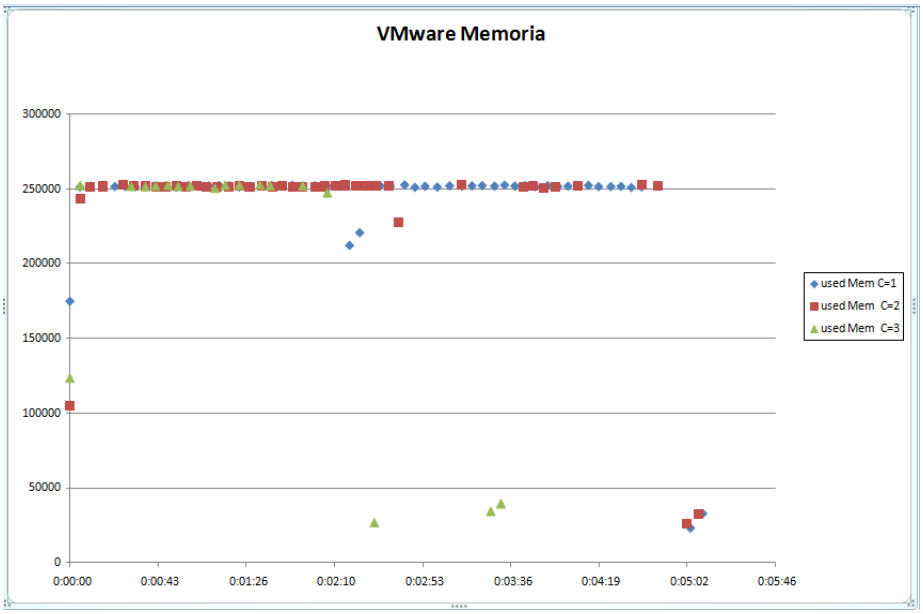

Gráfico III.5. VMware: Memoria

- **SWAP**

La utilización del recurso swap se presencia a partir del parámetro 3, donde el servidor virtual requiere la utilización del swap disponible hasta cuando deja de responder el máxima del total de la memoria asignanda en kilobytes (kb) 255548k. Teniendo un swap de 522104k es donde falla la ejecución del utilitario stress.

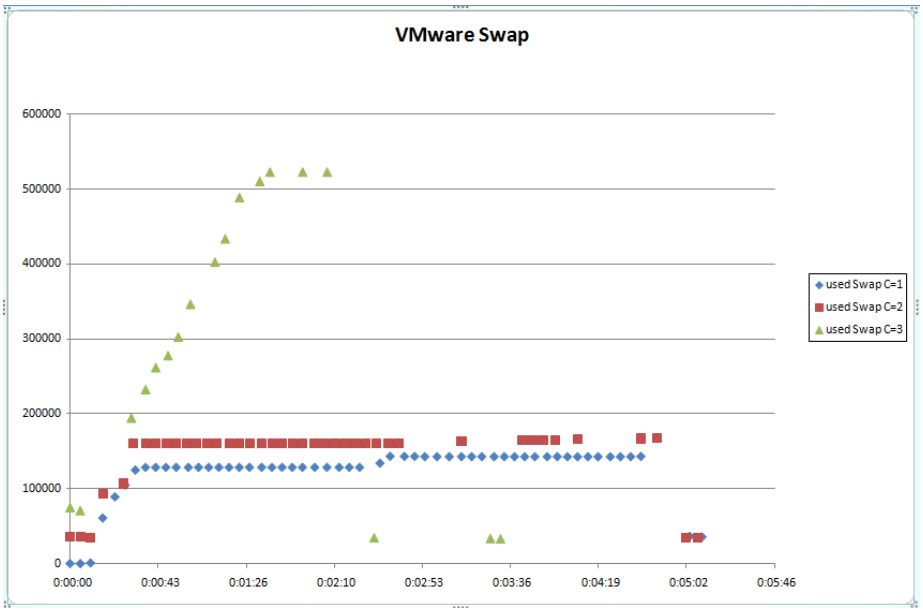

Gráfico III.6. VMware: Swap

➢ **Módulo 7**

- **Interfaz de comunicación de máquina virtual (Virtual Machine Communication Interface VMCI)**

Soporte a la rápida y eficiente comunicación entre un servidor virtual y el sistema operativo host, y entre dos o más servidores virtuales en el mismo host.

También se considera la utilización del utilitario Unixbench, a continuación se detalla su desarrollo:

La instalación del utilitario Unixbench consta de siguientes pasos:

```
[root@cosvmtest tmp]# tar zxvf unixbench-4.1.0.gz
[root@cosvmtest tmp]# cd unixbench-4.1.0
[root@cosvmtest unixbench-4.1.0]# make
```

Para su ejecución se realiza de la siguiente forma :

```
[root@cosvmtest unixbench-4.1.0]# ./Run
========================================================
  BYTE UNIX Benchmarks (Version 4.1.0)
  System -- Linux cosvmtest 2.6.18-128.el5 #1 SMP Wed Jan 21 10:44:23 EST 2009 i686 i686 i386 GNU/Linux
  Start Benchmark Run: Thu Dec 24 13:00:44 ECT 2009
   3 interactive users.
   13:00:44 up 56 min,  3 users,  load average: 0.61, 1.22, 2.51
  lrwxrwxrwx 1 root root 4 Dec 24 08:50 /bin/sh -> bash
  /bin/sh: symblic link to `bash'
  /dev/sda3 1          34321148   3270660  29278924  11% /
Dhrystone 2 using register variables     1630173.4 lps   (10.0 secs, 10 samples)
Double-Precision Whetstone                 439.8 MWIPS (10.5 secs, 10 samples)
System Call Overhead                     45114.5 lps   (10.0 secs, 10 samples)
Pipe Throughput                          58107.2 lps   (10.0 secs, 10 samples)
Pipe-based Context Switching              7728.3 lps   (10.0 secs, 10 samples)
Process Creation                           249.1 lps   (30.0 secs, 3 samples)
Execl Throughput                           138.9 lps   (29.5 secs, 3 samples)
File Read 1024 bufsize 2000 maxblocks     85738.0 KBps  (30.0 secs, 3 samples)
File Write 1024 bufsize 2000 maxblocks    61693.0 KBps  (30.0 secs, 3 samples)
File Copy 1024 bufsize 2000 maxblocks     31438.0 KBps  (30.0 secs, 3 samples)
File Read 256 bufsize 500 maxblocks       25608.0 KBps  (30.0 secs, 3 samples)
File Write 256 bufsize 500 maxblocks      16838.0 KBps  (30.0 secs, 3 samples)
File Copy 256 bufsize 500 maxblocks        9077.0 KBps  (30.0 secs, 3 samples)
File Read 4096 bufsize 8000 maxblocks    215466.0 KBps  (30.0 secs, 3 samples)
File Write 4096 bufsize 8000 maxblocks   173510.0 KBps  (30.0 secs, 3 samples)
File Copy 4096 bufsize 8000 maxblocks     80870.0 KBps  (30.0 secs, 3 samples)
Shell Scripts (1 concurrent)               239.5 lpm   (60.1 secs, 3 samples)
Shell Scripts (8 concurrent)                34.3 lpm   (60.0 secs, 3 samples)
Shell Scripts (16 concurrent)               17.0 lpm   (60.0 secs, 3 samples)
Arithmetic Test (type = short)          263626.9 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = int)            283972.2 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = long)           284902.6 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = float)          249659.2 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = double)         255214.2 lps   (10.0 secs, 3 samples)
Arithoh                              105205679.3 lps   (10.0 secs, 3 samples)
C Compiler Throughput                      117.0 lpm   (60.0 secs, 3 samples)
Dc: sqrt(2) to 99 decimal places          4256.1 lpm   (30.0 secs, 3 samples)
Recursion Test--Tower of Hanoi           39216.3 lps   (20.0 secs, 3 samples)
```

Resultados de la ejecución:

Tabla III.3. Vmware – UnixBench - Index Values

INDEX VALUES			
TEST	**BASELINE**	**RESULT**	**INDEX**
Dhrystone 2 using register variables	116700.0	1630173.4	139.7
Double-Precision Whetstone	55.0	439.8	80.0
Execl Throughput	43.0	138.9	32.3
File Copy 1024 bufsize 2000 maxblocks	3960.0	31438.0	79.4
File Copy 256 bufsize 500 maxblocks	1655.0	9077.0	54.8
File Copy 4096 bufsize 8000 maxblocks	5800.0	80870.0	139.4
Pipe Throughput	12440.0	58107.2	46.7
Process Creation	126.0	249.1	19.8
Shell Scripts (8 concurrent)	6.0	34.3	57.2
System Call Overhead	15000.0	45114.5	30.1
			=========
FINAL SCORE			56.8

El rendimiento obtenido con el utilitario de Unixbench es de 56.8 puntos.

La utilización del benchmark Netperf en el servidor host para el análisis de redes se detalla a continuación:

```
[root@vmwaretest tmp]# tar zxvf netperf-2.4.5.tar.gz
[root@vmwaretest tmp]# cd netperf-2.4.5
[root@vmwaretest netperf-2.4.5]# ./configure
[root@vmwaretest netperf-2.4.5]# make
[root@vmwaretest netperf-2.4.5]# make install
[root@vmwaretest netperf-2.4.5]# vi /etc/services
netperf 12865/tcp
[root@vmwaretest netperf-2.4.5]# netserver
Starting netserver at port 12865
Starting netserver at hostname 0.0.0.0 port 12865 and family AF_UNSPEC
```

De forma similar se instala en el servidor guest:

```
[root@cosvmtest tmp]# tar zxvf netperf-2.4.5.tar.gz
[root@cosvmtest tmp]# cd netperf-2.4.5
[root@cosvmtest netperf-2.4.5]# ./configure
[root@cosvmtest netperf-2.4.5]# make install
[root@cosvmtest netperf-2.4.5]# vi /etc/services
[root@cosvmtest netperf-2.4.5]# netserver
Starting netserver at port 12865
Starting netserver at hostname 0.0.0.0 port 12865 and family AF_UNSPEC
```

Ejecución del benchmark NetPerf:

```
[root@vmwaretest netperf-2.4.5]# netperf -H cosvmtest
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvmtest (192.168.0.151) port 0 AF_INET
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
```

```
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.01      16.45
```

```
[root@vmwaretest netperf-2.4.5]# netperf -H cosvmtest -t UDP_RR
UDP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvmtest (192.168.0.151) port 0 AF_INET
Local /Remote
Socket Size   Request  Resp.   Elapsed  Trans.
Send   Recv   Size     Size    Time     Rate
bytes  Bytes  bytes    bytes   secs.    per sec

109568 109568 1        1       10.00    1186.45
109568 109568
```

```
[root@vmwaretest netperf-2.4.5]# netperf -H cosvmtest -t UDP_STREAM -- -m 1024
UDP UNIDIRECTIONAL SEND TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvmtest (192.168.0.151) port 0 AF_INET
Socket  Message  Elapsed      Messages
Size    Size     Time         Okay Errors   Throughput
bytes   bytes    secs            #      #   10^6bits/sec

109568    1024   10.00       18402      0      15.07
109568           10.00       18364             15.04
```

```
[root@vmwaretest netperf-2.4.5]# netperf -H cosvmtest -t TCP_RR
TCP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvmtest (192.168.0.151) port 0 AF_INET
Local /Remote
Socket Size   Request  Resp.   Elapsed  Trans.
Send   Recv   Size     Size    Time     Rate
bytes  Bytes  bytes    bytes   secs.    per sec

16384  87380  1        1       10.00    1550.19
16384  87380
```

- **Módulo 8**

Se procede con la instalación del utilitario Bonnie:

```
[root@cosvmtest tmp]# cd bonnie/
[root@cosvmtest bonnie]# cd bonnie++-1.03e/
[root@cosvmtest bonnie++-1.03e]# ./configure
[root@cosvmtest bonnie++-1.03e]# make install
```

```
[root@cosvmtest ~]# bonnie++ -d /tmp/bonnie -m costest -u root
Using uid:0, gid:0.
Writing with putc()...done
Writing intelligently...done
Rewriting...
done
Reading with getc()...
done
Reading intelligently...done
start 'em...done...done...done...
Create files in sequential order...done.
Stat files in sequential order...done.
Delete files in sequential order...done.
Create files in random order...done.
Stat files in random order...do
Delete files in random order...done.
```

```
Version 1.03e       ------Sequential Output------ --Sequential Input- --Random-
                    -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
costest     496M  5506  72 10684  51  1072   4   711   4   913   1  75.2   5
                    ------Sequential Create------ --------Random Create--------
                    -Create-- --Read--- -Delete-- -Create-- --Read--- -Delete--
              files  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP
                 16  7831  86 +++++ +++ 12028  99  8537  98 +++++ +++  9495  79
costest,496M,5506,72,10684,51,1072,4,711,4,913,1,75.2,5,16,7831,86,+++++,+++,12028,99,8537,98,+++++,+++,9495,79
```

- **Módulo 9**

- **Extenso los sistema operativos de soporte**

VMware Server ahora es compatible con Windows Server 2008, Windows Vista® Business Edition y Ultimate Edition (sólo guest), Red Hat® Enterprise Linux 5 y Ubuntu 8.04, incluyendo compatibilidad con el paravirtualizado en ciertas distribuciones de Linux.

- **Soporte de Virtual Machine Interface (VMI)**

Transparente para la paravirtualización, en la que una única versión binaria del sistema operativo puede funcionar tanto en hardware nativo o en modo paravirtualizado para mejorar el rendimiento en los ambientes Linux.

- **Independencia del hardware**

Los servidores virtuales son completamente independientes de su hardware físico subyacente. Por ejemplo, se puede configurar un servidor virtual con componentes virtuales (CPU, tarjeta de red, controlador SCSI) que difieren totalmente de los componentes físicos presentes en el hardware subyacente. Los servidores virtuales del

mismo servidor físico pueden incluso ejecutar distintos tipos de sistema operativo (Windows, Linux, etc.).

Si se combina con las propiedades de alta disponibilidad y escalabilidad, la independencia del hardware proporciona la libertad para mover un servidor virtual de un tipo de ordenador x86 a otro sin necesidad de efectuar ningún cambio en los controladores de dispositivo, en el sistema operativo o en las aplicaciones. La independencia del hardware también significa que se puede ejecutar una mezcla heterogénea de sistemas operativos y aplicaciones en un único ordenador físico.

3.2.5.2. Desarrollo de los Módulos en el software de virtualización de ordenadores Xen

- **Módulo 1**

Para la construcción del módulo 1 se va a utilizar el instalador del CentOs, la cual brinda algunas facilidades para la obtención del software de virtualización, los pasos para el desarrollo de este módulo se describen con más detalle en el Capítulo III.

La instalación del monitor de servidores virtuales Xen (dom0) comprende tres elementos principalmente:

- El hypervisor de Xen, que realiza la paravirtualización.
- Núcleo (kernel) modificado que pueda trabajar sobre un hypervisor.
- Herramientas para el manejo de los seridores virtuales: para crearlas, iniciarlas, etc.

La situación ideal es utilizar una distribución de GNU/Linux que incluya todos los componentes empaquetados, lo que nos garantiza la consistencia de nuestro sistema.

Con más detalle este parámetro se desarrolló ampliamente en el capítulo IV.

- **Módulo 2**

- **Soporte Sistema Operativo de 64 bits**

Virtualization Technology (VT) como un conjunto de mejoras de las arquitecturas 64/32 bits que permite crecer la funcionalidad de un servidor virtual, a continuación se detalla las arquitecturas soportadas:

Tabla III.4. Arquitectura Soportadas Xen

CPU	Host OS	32-bit Guest OS	64-bit Guest OS
32-bit CPU	32-bit Host OS	Supported	Unsupported
	64-bit Host OS	Unsupported	Unsupported
64-bit CPU	32-bit Host OS	Supported	Supported
	64-bit Host OS	Supported	Supported

Xen funciona actualmente en sistemas basados en x86. Actualmente se están portando las plataformas x86_64, IA64 y PPC. Los ports de otras plataformas son técnicamente posibles y podrán estar disponibles en el futuro.

- **Módulo 3**

- **Migración de Servidores Virtuales**

Podemos migrar fácilmente de hardware, cuando necesitamos que nuestro (s) servidores corran en un mejor hardware, podemos realizar una fácil y rápida migración sin tener que reinstalar, se mueve toda la información al nuevo hardware y se arranca el servidor. Xen incluye la capacidad para soportar la migración de servidores virtuales hacia servidores físicos. Existen dos maneras para realizar esta actividad:

- ✓ **Offline Mode,** usando el comando *xm migrate VirtualServerName HostName*.
  En este modo el servidor virtual deberá ser detenido en el host original para luego ser reiniciado en el nuevo servidor.

- ✓ **Live Modde,** usando la opción - - live para el comando *xm migrate VirtualServerName HostName.*

- **El servidor virtual arranca más rápido**

No hay que esperar que cuente la memoria o cargue el bios, ya el hardware lo tenía cargado desde tiempo atrás. Si hay un crash en una máquina, el sistema en sí que virtualiza sigue trabajando.

- **Aislamiento**

Aunque los servidores virtuales pueden compartir los recursos físicos de un único ordenador, permanecen completamente aisladas unas de otras, como si se tratara de servidores independientes. Si, por ejemplo, hay cuatro servidores virtuales en un único servidor físico y falla una de ellas, las otras tres siguen estando disponibles. El aislamiento es un facto importante que explica por qué la disponibilidad y protección de

las aplicaciones que se ejecutan en un entorno virtual es muy superior a las aplicaciones que se ejecutan en un sistema tradicional no virtualizado.

- **Herramientas para Virtualización Xen**

La siguiente es una lista de herramientas para la administración de la virtualización, debugging y networking que son útiles para los sistemas que ejecutan Xen:

- ✓ **Herramientas de la Administración de Sistemas,** xentop, xm dmesg, xm log, vmstat, iostat, lsof.
- ✓ **Herramientas Avanzadas de depuración,** XenOprofile, systemTap, crash, xen-gdbserver, sysrq, sysrq t, sysrq w, sysrq c.
- ✓ **Networking,** brtcl, ifconfig, tcpdump.

➢ **Módulo 4**

- **Administración de recursos hardware**

Para la administración de los recursos hardware lo podemos llevar a cabo mediante la utilización de la interfaz gráfica o la consola de línea de comandos.

- ✓ **Interfaz Gráfica de Usuario,** Usamos el Add new virtual hardware wizard para añadir nuevo hardware al servidor virtual, el mismo deber ser apagado para añadir cualquier tipo de hardware.

Figura III.35. Xen: Asignando nuevo hardware

Figura III.36. Xen: Asignando espacio de disco

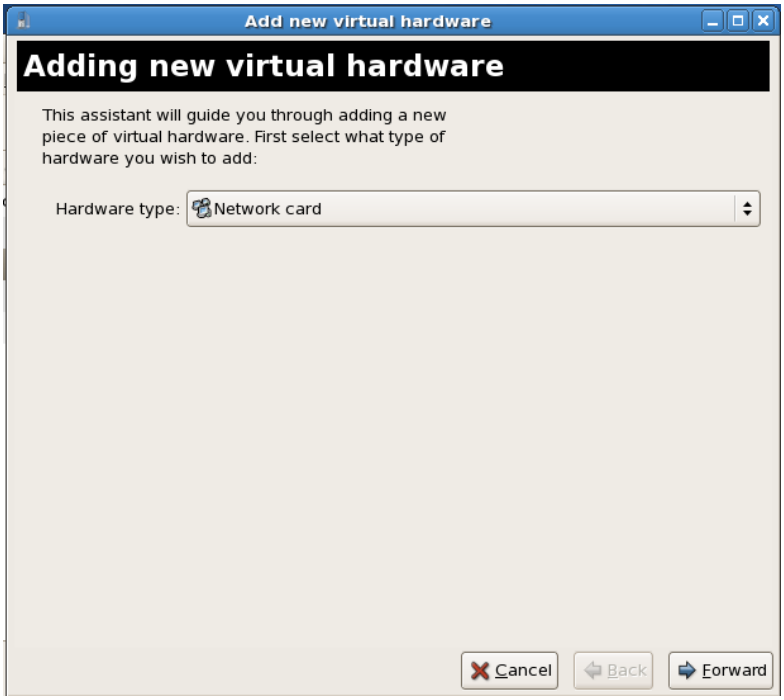

Figura III.37. Xen: Asignando Tarjeta de Red

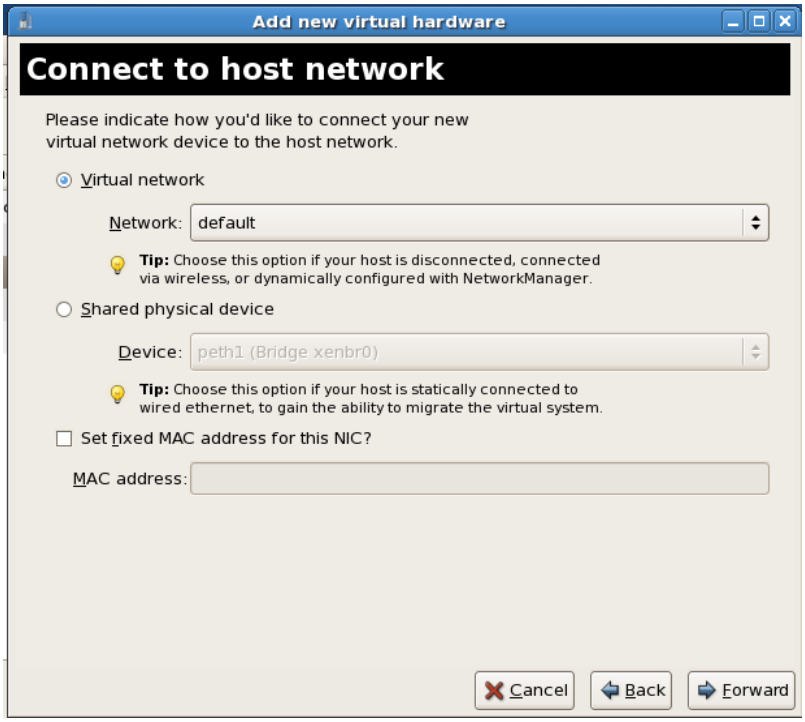

Figura III.38. Xen: Conectar el host de red

- ✓ **Consola de Línea de Comandos,** Usamos los siguientes comandos xm para la administración de recursos hardware:

xm mem-set, setea un valor para la memoria Ram del servidor virtual.

xm vcpu-set, setea la cantidad de procesadores que tendrá el servidor virtual.

- ➢ **Módulo 5**

Dentro del parámetro de la usabilidad se define la creación servidores virtuales, para lo cual en Xen existen las siguientes formas de acrearlas:

- ✓ **Método Paravirtualización,** para la creación de servidores virtuales como guest, utilizando el comando *virt-install* o la herramienta de *Virtual Machine Manager*.

- ✓ **Método Full Virtualización,** para la creación de servidores virtuales como guest, utilizando el comando *virt-install* o la herramienta de *Virtual Machine Manager.*

Con más detalle este parámetro será desarrollo amplimente en el capítulo IV.

- ➢ **Módulo 6**

Procedemos con la instalación del utilitario stress tal como se indica a continuación:

```
[root@cosxentest tmp]# tar xvf stress-1.0.2.tar.tar
[root@cosxentest tmp]# cd stress-1.0.2
[root@cosxentest stress-1.0.2]#   ./configure && make && sudo make install
stress: info: [5705] successful run completed in 305s
```

Los parámetros con los que vamos a evaluar será un promedio de cuatro que se impone en el sistema especificando -c procesos limitados por CPU, -i de I/O-bound procesos, -m

procesos asignados de memoria y –d procesos de disco. A continuación la ejecución del mismo:

Procedemos a ejecutar de la siguiente manera:

```
[root@cosxentest ~]# date;stress -c 1 -i 1 -m 1 -d 1 --verbose --timeout 5m;date
Sun Dec 20 13:13:24 ECT 2009
stress: info: [2533] dispatching hogs: 1 cpu, 1 io, 1 vm, 1 hdd
stress: dbug: [2533] using backoff sleep of 12000us
stress: dbug: [2533] setting timeout to 300s
stress: dbug: [2533] --> hogcpu worker 1 [2534] forked
stress: dbug: [2533] --> hogio worker 1 [2535] forked
stress: dbug: [2533] --> hogvm worker 1 [2536] forked
stress: dbug: [2533] --> hoghdd worker 1 [2537] forked
stress: dbug: [2536] allocating 268435456 bytes ...
stress: dbug: [2536] touching bytes in strides of 4096 bytes ...
stress: dbug: [2537] seeding 1048575 byte buffer with random data
stress: dbug: [2537] opened ./stress.A5dVIb for writing 1073741824 bytes
stress: dbug: [2537] unlinking ./stress.A5dVIb
stress: dbug: [2537] fast writing to ./stress.A5dVIb
stress: dbug: [2537] slow writing to ./stress.A5dVIb
stress: dbug: [2537] closing ./stress.A5dVIb after 1073741824 bytes
stress: dbug: [2537] opened ./stress.f9V8dv for writing 1073741824 bytes
stress: dbug: [2537] unlinking ./stress.f9V8dv
stress: dbug: [2537] fast writing to ./stress.f9V8dv
stress: dbug: [2537] slow writing to ./stress.f9V8dv
stress: dbug: [2537] closing ./stress.f9V8dv after 1073741824 bytes
stress: dbug: [2537] opened ./stress.wyBiyR for writing 1073741824 bytes
stress: dbug: [2537] unlinking ./stress.wyBiyR
stress: dbug: [2537] fast writing to ./stress.wyBiyR
stress: dbug: [2537] slow writing to ./stress.wyBiyR
stress: dbug: [2537] closing ./stress.wyBiyR after 1073741824 bytes
stress: dbug: [2537] opened ./stress.RnPfcj for writing 1073741824 bytes
stress: dbug: [2537] unlinking ./stress.RnPfcj
stress: dbug: [2537] fast writing to ./stress.RnPfcj
stress: dbug: [2533] <-- worker 2534 signalled normally
stress: dbug: [2533] <-- worker 2536 signalled normally
stress: dbug: [2533] <-- worker 2537 signalled normally
stress: dbug: [2533] <-- worker 2535 signalled normally
stress: info: [2533] successful run completed in 304s
Sun Dec 20 13:18:30 ECT 2009
```

```
[root@cosxentest ~]# date;stress -c 2 -i 2 -m 2 -d 2 --verbose --timeout 5m;date
Sun Dec 20 13:25:53 ECT 2009
stress: info: [3202] dispatching hogs: 2 cpu, 2 io, 2 vm, 2 hdd
stress: dbug: [3202] using backoff sleep of 24000us
stress: dbug: [3202] setting timeout to 300s
stress: dbug: [3202] --> hogcpu worker 2 [3203] forked
stress: dbug: [3202] --> hogio worker 2 [3204] forked
stress: dbug: [3202] --> hogvm worker 2 [3205] forked
stress: dbug: [3202] --> hoghdd worker 2 [3206] forked
stress: dbug: [3202] using backoff sleep of 12000us
stress: dbug: [3202] setting timeout to 300s
stress: dbug: [3202] --> hogcpu worker 1 [3207] forked
stress: dbug: [3202] --> hogio worker 1 [3208] forked
stress: dbug: [3202] --> hogvm worker 1 [3209] forked
stress: dbug: [3202] --> hoghdd worker 1 [3210] forked
stress: dbug: [3205] allocating 268435456 bytes ...
stress: dbug: [3205] touching bytes in strides of 4096 bytes ...
stress: dbug: [3206] seeding 1048575 byte buffer with random data
stress: dbug: [3209] allocating 268435456 bytes ...
stress: dbug: [3209] touching bytes in strides of 4096 bytes ...
stress: dbug: [3210] seeding 1048575 byte buffer with random data
stress: dbug: [3206] opened ./stress.GpYjZ4 for writing 1073741824 bytes
stress: dbug: [3206] unlinking ./stress.GpYjZ4
stress: dbug: [3206] fast writing to ./stress.GpYjZ4
stress: dbug: [3210] opened ./stress.M07EWC for writing 1073741824 bytes
stress: dbug: [3210] unlinking ./stress.M07EWC
```

```
stress: dbug: [3210] fast writing to ./stress.M07EWC
stress: dbug: [3206] slow writing to ./stress.GpYjZ4
stress: dbug: [3206] closing ./stress.GpYjZ4 after 1073741824 bytes
stress: dbug: [3210] slow writing to ./stress.M07EWC
stress: dbug: [3210] closing ./stress.M07EWC after 1073741824 bytes
stress: dbug: [3210] opened ./stress.K10MgD for writing 1073741824 bytes
stress: dbug: [3210] unlinking ./stress.K10MgD
stress: dbug: [3210] fast writing to ./stress.K10MgD
stress: dbug: [3206] opened ./stress.yp8d1s for writing 1073741824 bytes
stress: dbug: [3206] unlinking ./stress.yp8d1s
stress: dbug: [3206] fast writing to ./stress.yp8d1s
stress: dbug: [3202] <-- worker 3203 signalled normally
stress: dbug: [3202] <-- worker 3205 signalled normally
stress: dbug: [3202] <-- worker 3207 signalled normally
stress: dbug: [3202] <-- worker 3209 signalled normally
stress: dbug: [3202] <-- worker 3210 signalled normally
stress: dbug: [3202] <-- worker 3206 signalled normally
stress: dbug: [3202] <-- worker 3208 signalled normally
stress: dbug: [3202] <-- worker 3204 signalled normally
stress: info: [3202] successful run completed in 305s
Sun Dec 20 13:31:02 ECT 2009
[root@cosxentest ~]# date;stress -c 3 -i 3 -m 3 -d 3 --verbose --timeout 5m;date
Sun Dec 20 13:31:36 ECT 2009
stress: info: [3403] dispatching hogs: 3 cpu, 3 io, 3 vm, 3 hdd
stress: dbug: [3403] using backoff sleep of 36000us
stress: dbug: [3403] setting timeout to 300s
stress: dbug: [3403] --> hogcpu worker 3 [3404] forked
stress: dbug: [3403] --> hogio worker 3 [3405] forked
stress: dbug: [3403] --> hogvm worker 3 [3406] forked
stress: dbug: [3403] --> hoghdd worker 3 [3407] forked
stress: dbug: [3403] using backoff sleep of 24000us
stress: dbug: [3403] setting timeout to 300s
stress: dbug: [3403] --> hogcpu worker 2 [3408] forked
stress: dbug: [3403] --> hogio worker 2 [3409] forked
stress: dbug: [3403] --> hogvm worker 2 [3410] forked
stress: dbug: [3403] --> hoghdd worker 2 [3411] forked
stress: dbug: [3403] using backoff sleep of 12000us
stress: dbug: [3403] setting timeout to 300s
stress: dbug: [3403] --> hogcpu worker 1 [3412] forked
stress: dbug: [3403] --> hogio worker 1 [3413] forked
stress: dbug: [3403] --> hogvm worker 1 [3414] forked
stress: dbug: [3403] --> hoghdd worker 1 [3415] forked
stress: dbug: [3406] allocating 268435456 bytes ...
stress: dbug: [3406] touching bytes in strides of 4096 bytes ...
stress: dbug: [3407] seeding 1048575 byte buffer with random data
stress: dbug: [3414] allocating 268435456 bytes ...
stress: dbug: [3414] touching bytes in strides of 4096 bytes ...
stress: dbug: [3415] seeding 1048575 byte buffer with random data
stress: dbug: [3410] allocating 268435456 bytes ...
stress: dbug: [3410] touching bytes in strides of 4096 bytes ...
stress: dbug: [3415] opened ./stress.fkpzfl for writing 1073741824 bytes
stress: dbug: [3415] unlinking ./stress.fkpzfl
stress: dbug: [3415] fast writing to ./stress.fkpzfl
stress: dbug: [3411] seeding 1048575 byte buffer with random data
stress: dbug: [3407] opened ./stress.xzPpOQ for writing 1073741824 bytes
stress: dbug: [3407] unlinking ./stress.xzPpOQ
stress: dbug: [3407] fast writing to ./stress.xzPpOQ
stress: dbug: [3411] opened ./stress.iNbhVL for writing 1073741824 bytes
stress: dbug: [3411] unlinking ./stress.iNbhVL
stress: dbug: [3411] fast writing to ./stress.iNbhVL
stress: dbug: [3403] <-- worker 3854 signalled normally
stress: dbug: [3403] <-- worker 3852 signalled normally
stress: dbug: [3403] <-- worker 3858 signalled normally
stress: dbug: [3403] <-- worker 3856 signalled normally
stress: dbug: [3403] <-- worker 3860 signalled normally
stress: dbug: [3403] <-- worker 3862 signalled normally
stress: dbug: [3403] <-- worker 3855 signalled normally
stress: dbug: [3403] <-- worker 3859 signalled normally
stress: dbug: [3403] <-- worker 3861 signalled normally
stress: dbug: [3403] <-- worker 3857 signalled normally
stress: dbug: [3403] <-- worker 3853 signalled normally
stress: dbug: [3403] <-- worker 3863 signalled normally
stress: info: [3403] successful run completed in 309s
```

Después de la ejecuación del stress se tiene el siguiente análisis respecto al consumo de: load average, tasks running, tasks sleeping, cpu, memoria y swap:

- **LOAD AVERAGE**

Con una carga inicial de 1 todas las tareas de stress finalizan bien de igual manera para la carga 2; cuando se aumenta la carga a 3, no existe encolamiento porque los procesos empieza a despachar rápidamente durante todo el tiempo de ejecución de la prueba de stress.

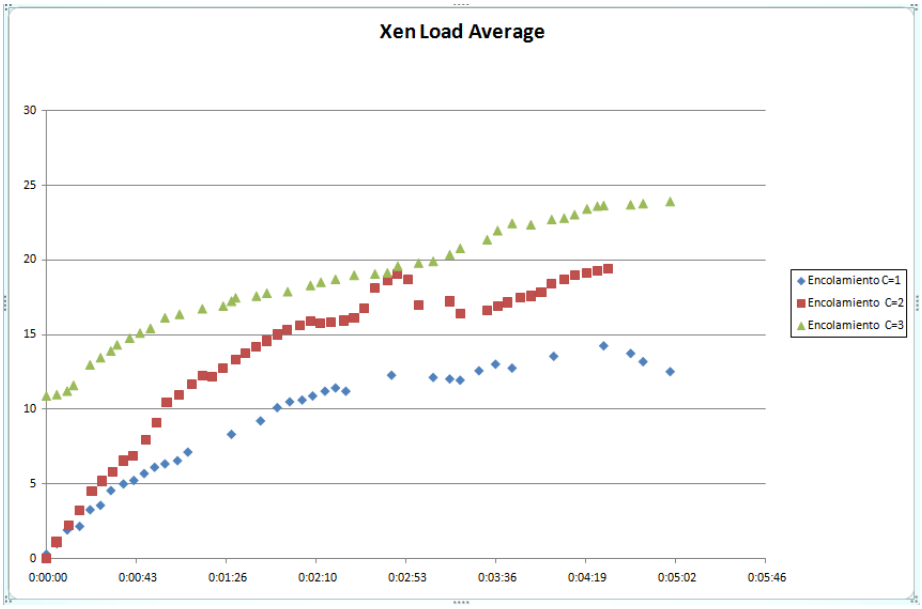

Gráfico III.7. Xen: Load Average

- **TASKS - RUNNING**

Con una carga final de hasta 3 en utilización de procesos CPU, I/O, memoria y disco, el número máximo de procesos que llega a alcanzar es superior a 25 en el tiempo de ejecución del utilitario stress.

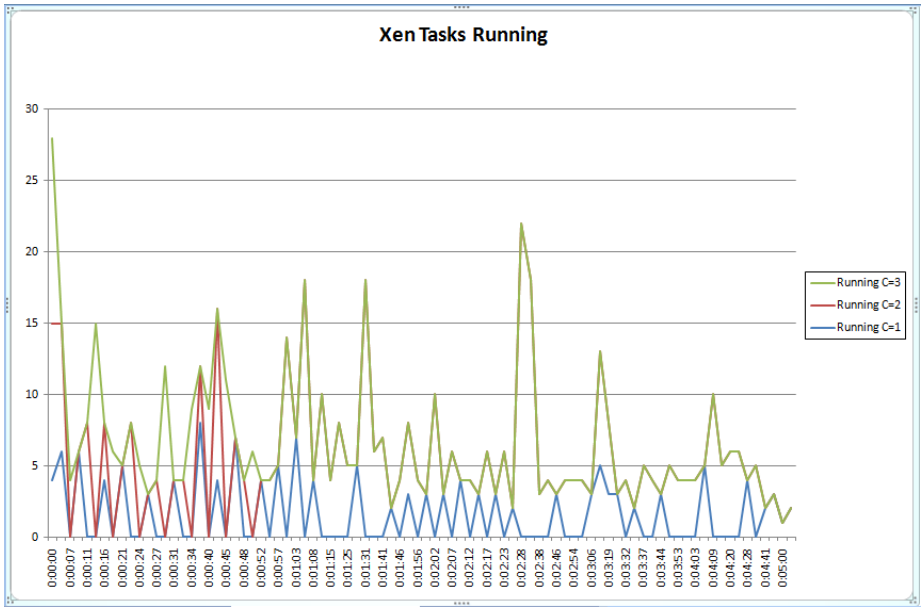

Gráfico III.8. Xen: Tasks - Running

- **SLEEPING**

Los procesos que están esperando para ser ejecutados se encuentran en un rango de pasado 50 a 200 debido a que despacha pronto todos los procesos existentes, esto se evidencia con una carga final de hasta 3 en utilización de procesos CPU, I/O, memoria y disco.

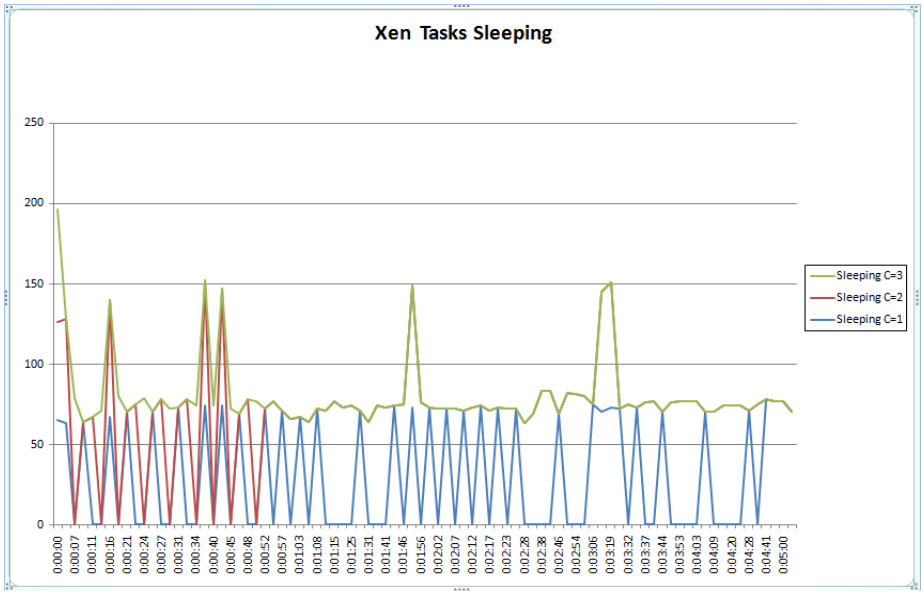

Gráfico III.9. Xen: Tasks – Sleeping

- **CPU**

El consumo del recurso CPU llega a una utilización del 100% con los tres valores de parámetros CPU, I/O, memoria y disco.

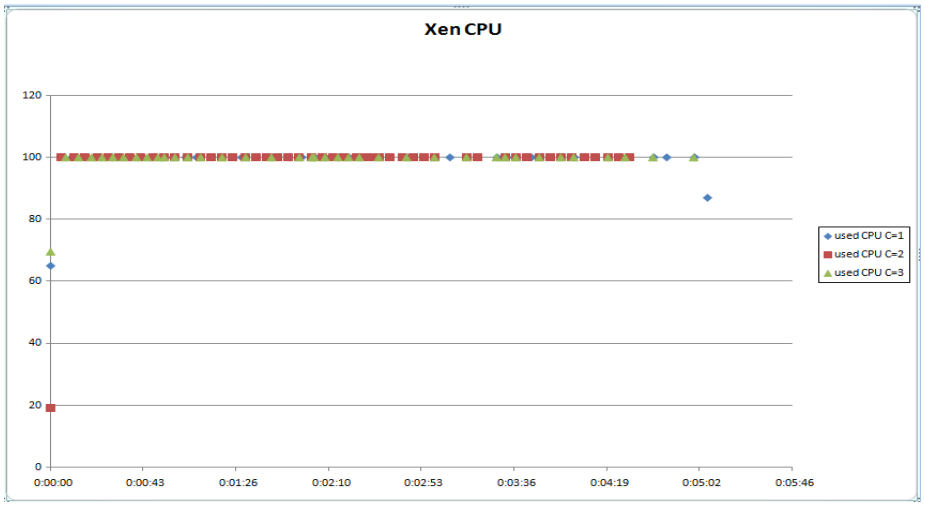

Gráfico III.10. Xen: CPU

- **MEMORIA**

La utilización del recurso memoria llega a su utilización máxima del total de la memoria asignanda en kilobytes (kb) 262320k.

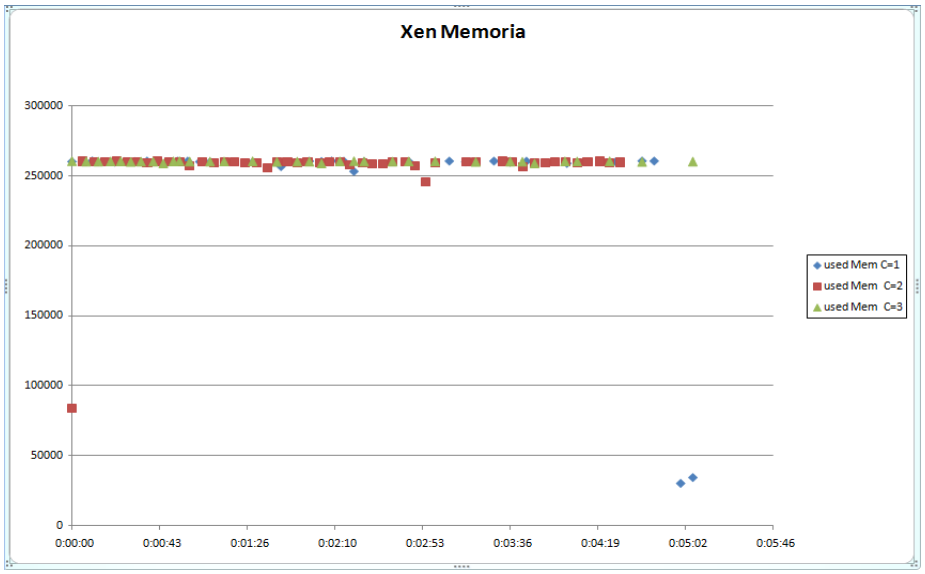

Gráfico III.11. Xen: Memoria

- **SWAP**

La utilización del recurso swap se presencia a partir del parámetro 3, donde el servidor virtual requiere la utilización del swap disponible llegando a obtener 561828k, de un total de 557048k.

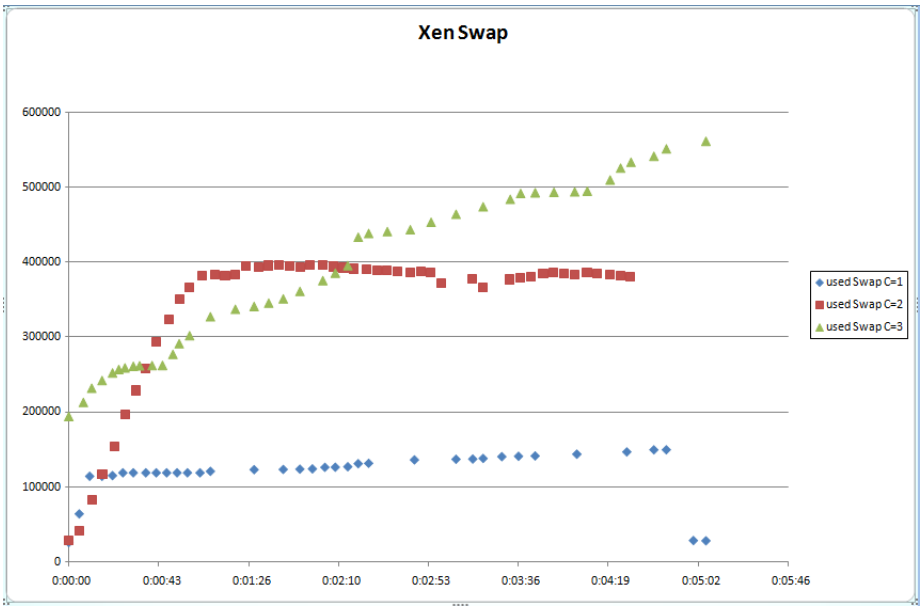

Gráfico III.12. Xen: Swap

- **Módulo 7**

Se considera la utilización del utilitario Unixbench, a continuación se detalla su desarrollo:

La instalación del utilitario Unixbench consta de siguientes pasos:

```
[root@cosxentest tmp]# tar zxvf unixbench-4.1.0.gz
[root@cosxentest tmp]# cd unixbench-4.1.0
[root@cosxentest unixbench-4.1.0]# make
[root@cosxentest unixbench-4.1.0]# ./Run
============================================================
BYTE UNIX Benchmarks (Version 4.1.0)
  System -- Linux cosxentest 2.6.18-128.el5xen #1 SMP Wed Jan 21 11:55:02 EST 2009 i686 i686 i386 GNU/Linux
  Start Benchmark Run: Sun Dec 20 17:27:33 ECT 2009
   1 interactive users.
   17:27:34 up  3:42,  1 user,  load average: 0.25, 0.33, 0.58
  lrwxrwxrwx 1 root root 4 Dec 19 12:11 /bin/sh -> bash
  /bin/sh: symbolic link to `bash'
                     36914728   2067744  32941528   6% /
Dhrystone 2 using register variables     2150603.3 lps   (10.0 secs, 10 samples)
```

```
Double-Precision Whetstone                589.6 MWIPS (10.5 secs, 10 samples)
System Call Overhead                   173588.8 lps   (10.0 secs, 10 samples)
Pipe Throughput                    187762.2 lps   (10.0 secs, 10 samples)
Pipe-based Context Switching           27749.9 lps   (10.0 secs, 10 samples)
Process Creation                     841.5 lps   (30.0 secs, 3 samples)
Execl Throughput                     386.5 lps   (29.7 secs, 3 samples)
File Read 1024 bufsize 2000 maxblocks    274602.0 KBps  (30.0 secs, 3 samples)
File Write 1024 bufsize 2000 maxblocks   211575.0 KBps  (30.0 secs, 3 samples)
File Copy 1024 bufsize 2000 maxblocks    111580.0 KBps  (30.0 secs, 3 samples)
File Read 256 bufsize 500 maxblocks       90116.0 KBps  (30.0 secs, 3 samples)
File Write 256 bufsize 500 maxblocks      60180.0 KBps  (30.0 secs, 3 samples)
File Copy 256 bufsize 500 maxblocks       34419.0 KBps  (30.0 secs, 3 samples)
File Read 4096 bufsize 8000 maxblocks    527786.0 KBps  (30.0 secs, 3 samples)
File Write 4096 bufsize 8000 maxblocks   477649.0 KBps  (30.0 secs, 3 samples)
File Copy 4096 bufsize 8000 maxblocks    229673.0 KBps  (30.0 secs, 3 samples)
Shell Scripts (1 concurrent)             758.7 lpm   (60.0 secs, 3 samples)
Shell Scripts (8 concurrent)             105.0 lpm   (60.0 secs, 3 samples)
Shell Scripts (16 concurrent)             52.3 lpm   (60.0 secs, 3 samples)
Arithmetic Test (type = short)         360975.4 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = int)           391600.8 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = long)          389216.7 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = float)         338480.3 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = double)        338725.2 lps   (10.0 secs, 3 samples)
Arithoh                                132443730.4 lps   (10.0 secs, 3 samples)
C Compiler Throughput                    282.9 lpm   (60.0 secs, 3 samples)
Dc: sqrt(2) to 99 decimal places         14961.4 lpm   (30.0 secs, 3 samples)
Recursion Test--Tower of Hanoi           52649.2 lps   (20.0 secs, 3 samples)
```

Resultados de la ejecución:

Tabla III.5. Xen – UnixBench - Index Values

INDEX VALUES			
TEST	**BASELINE**	**RESULT**	**INDEX**
Dhrystone 2 using register variables	116700.0	2150603.3	184.3
Double-Precision Whetstone	55.0	589.6	107.2
Execl Throughput	43.0	386.5	89.9
File Copy 1024 bufsize 2000 maxblocks	3960.0	111580.0	281.8
File Copy 256 bufsize 500 maxblocks	1655.0	34419.0	208.0
File Copy 4096 bufsize 8000 maxblocks	5800.0	229673.0	396.0
Pipe Throughput	12440.0	187762.2	150.9
Process Creation	126.0	841.5	66.8
Shell Scripts (8 concurrent)	6.0	105.0	175.0
System Call Overhead	15000.0	173588.8	115.7
			========
FINAL SCORE			155.8

El rendimiento obtenido con el utilitario de Unixbench es de 155.8 puntos.

La utilización del benchmark Netperf en el servidor host para el análisis de redes se detalla a continuación:

```
[root@xentest tmp]# tar zxvf netperf-2.4.5.tar.gz
[root@xentest tmp]# cd netperf-2.4.5
[root@xentest netperf-2.4.5]# ./configure
[root@xentest netperf-2.4.5]# make
[root@xentest netperf-2.4.5]# make install
[root@xentest netperf-2.4.5]# vi /etc/services
netperf 12865/tcp
[root@xentest netperf-2.4.5]# netserver
Starting netserver at port 12865
Starting netserver at hostname 0.0.0.0 port 12865 and family AF_UNSPEC
```

De configuración similar se instala en el servidor guest:

```
[root@cosxentest tmp]# tar zxvf netperf-2.4.5.tar.gz
[root@cosxentest tmp]# cd netperf-2.4.5
[root@cosxentest netperf-2.4.5]# ./configure
[root@cosxentest netperf-2.4.5]# make install
[root@cosxentest netperf-2.4.5]# vi /etc/services
[root@cosxentest netperf-2.4.5]# netserver
Starting netserver at port 12865
Starting netserver at hostname 0.0.0.0 port 12865 and family AF_UNSPEC
```

Ejecución del benchmark NetPerf:

```
[root@xentest ~]# netperf -H cosxentest
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosxentest (192.168.0.175) port 0 AF_INET
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.01     159.86
```

```
[root@xentest ~]# netperf -H cosxentest -t UDP_RR
UDP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosxentest (192.168.0.175) port 0 AF_INET
Local /Remote
Socket Size   Request  Resp.   Elapsed  Trans.
Send   Recv   Size     Size    Time     Rate
bytes  Bytes  bytes    bytes   secs.    per sec

109568 109568 1        1       10.00    2957.01
109568 109568
```

```
[root@xentest ~]# netperf -H cosxentest -t UDP_STREAM -- -m 1024
UDP UNIDIRECTIONAL SEND TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosxentest (192.168.0.175) port 0 AF_INET
Socket  Message  Elapsed      Messages
Size    Size     Time         Okay Errors   Throughput
bytes   bytes    secs            #      #   10^6bits/sec

109568    1024   10.01      115556      0      94.54
109568           10.01       48933             40.03
```

```
 [root@xentest ~]# netperf -H cosxentest -t TCP_RR
TCP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosxentest (192.168.0.175) port 0 AF_INET
Local /Remote
Socket Size   Request  Resp.   Elapsed  Trans.
Send   Recv   Size     Size    Time     Rate
```

```
bytes  Bytes  bytes    bytes   secs.    per sec

16384  87380  1        1       10.01    1993.60
16384  87380
```

- **Módulo 8**

Se procede con la instalación del utilitario Bonnie:

```
[root@cosxentest tmp]# cd bonnie/
[root@cosxentest bonnie]# cd bonnie++-1.03e/
[root@cosxentest bonnie++-1.03e]# ./configure
[root@cosxentest bonnie++-1.03e]# make install
```

```
[root@cosxentest tmp]# bonnie++ -d /tmp/bonnie -m costest -u root
Using uid:0, gid:0.
Writing with putc()...done
Writing intelligently...done
Rewriting...done
Reading with getc()...done
Reading intelligently...done
start 'em...done...done...done...
Create files in sequential order...done.
Stat files in sequential order...done.
Delete files in sequential order...done.
Create files in random order...done.
Stat files in random order...done.
Delete files in random order...done.
Version 1.03e       ------Sequential Output------ --Sequential Input- --Random-
                    -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
cosxentest     512M 17788  89 28400  14 10452   1 15115  52 24507   1 141.0   0
                    ------Sequential Create------ --------Random Create--------
                    -Create-- --Read--- -Delete-- -Create-- --Read--- -Delete--
              files  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP
                 16 10217  96 +++++ +++ 25846  98 10043  93 +++++ +++ 19461  77
cosxentest,512M,17788,89,28400,14,10452,1,15115,52,24507,1,141.0,0,16,10217,96,+++++,+++,25846,98,10043,93,++++
+,+++,19461,77
```

- **Módulo 9**

- **Combinación de la compatibilidad de virtualización de servidores host y guest**

Xen soporta varias combinaciones de servidores host y guest.

- ✓ **ARQUITECTURA X86**

Xen en una plataforma x86 es limitada a 16 CPU como núcleos de procesamiento.

Soporte de full virtualización en servidores guest:

Tabla III.6. Arquitectura x86 Soportadas Xen Full Virtualización

Operating system	Support level
Red Hat Enterprise Linux 3 x86	Optimized
Red Hat Enterprise Linux 4 x86	Optimized
Red Hat Enterprise Linux 5 x86	Optimized
Windows Server 2000 32-Bit	Supported
Windows Server 2003 32-Bit	Supported
Windows XP 32-Bit	Supported
Windows Vista 32-Bit	Supported

Utilizar fullvirtualización en Xen el procesador debe tener habilitado Intel-VT o AMD-V como tecnologías de virtualización.

Soporte de paravirtualización en servidores guest:

Tabla III.7. Arquitectura x86 Soportadas Xen Paravirtualización

Operating system	Support level
Red Hat Enterprise Linux 4 x86 Update 5 and higher	Optimized
Red Hat Enterprise Linux 5 x86	Optimized

Utilizar paravirtualización en Xen el procesador debe tener habilitado las instrucciones Physical Address Extension (PAE).

- ✓ **ARQUITECTURA AMD64 E INTEL 64**

Xen en servidores AMD64 e Intel 64 actualmente es limitado a 126 CPU de núcleos de procesamiento.

Soporte de full virtualización en servidores guest:

Tabla III.8. Arquitectura AMD64 e Intel 64 Soportadas Xen Full Virtualización

Operating system	Support level
Red Hat Enterprise Linux 3 x86-64	Optimized
Red Hat Enterprise Linux 3 x86	Optimized
Red Hat Enterprise Linux 4 AMD64/Intel 64	Optimized
Red Hat Enterprise Linux 4 x86	Optimized
Red Hat Enterprise Linux 5 AMD64/Intel 64	Optimized
Red Hat Enterprise Linux 5 x86	Optimized
Windows Server 2000 32-Bit	supported
Windows Server 2003 32-Bit	supported
Windows XP 32-Bit	supported
Windows Vista 32-Bit	supported
Windows Vista 64-Bit	supported
Windows Server 2008 32-Bit	supported
Windows Server 2008 64-Bit	supported
Solaris 32 bit	supported

Soporte de paravirtualización en servidores guest:

Tabla III.9. Arquitectura AMD64 e Intel 64 Soportadas Xen Paravirtualización

Operating system	Support level
Red Hat Enterprise Linux 4 AMD64/Intel 64 Update 5 and higher	Optimized
Red Hat Enterprise Linux 4 x86 Update 5 and higher	Technology preview in 5.2 and 5.3
Red Hat Enterprise Linux 5 AMD64/Intel 64	Optimized
Red Hat Enterprise Linux 5 x86	Technology preview in 5.2 and 5.3

- ✓ **ARQUITECTURA INTEL ITANIUM**

La siguiente lista es para virtualizar en la arquitectura Intel Itanium:

Soporte de full virtualización en servidores guest:

Tabla III.10. Arquitectura Intel Itanium Soportadas Xen Full Virtualización

Operating system	Support level
Red Hat Enterprise Linux 3 Itanium	Supported
Red Hat Enterprise Linux 4 Itanium	Supported
Red Hat Enterprise Linux 5 Itanium	Supported
Windows Server 2003 for Itanium-based Systems	Supported

Soporte de paravirtualización en servidores guest:

Tabla III.11. Arquitectura Intel Itanium Soportadas Xen Paravirtualización

Operating system	Support level
Red Hat Enterprise Linux 5 Itanium	Optimized

Xen requiere portar los sistemas operativos para adaptarse al API de Xen. Hasta el momento hay ports para NetBSD, Linux, FreeBSD y Plan 9. En 2005, Novell muestra un port de NetWare para Xen. Un port de Windows XP fue creado durante el desarrollo inicial de Xen.

- **Xen en SO Unix**

- CentOs 5 incluye la versión 3 de Xen.
- Mandriva 2006 incluye Xen 2.6 o Novell's Suse Linux Professional 10 incluye Xen 3. Fedora Core 5 incluye la versión 3 de Xen.
- Xenophilia es una distribución Linux que se basa en Xen.
- Xen demo CD es una ISO live CD basada en Debian que permite probar Xen en nuestro sistema sin instalarla en disco duro.
- Debian también incluye los paquetes de Xen, en sus repositorios inestable y experimental.

- NetBSD 2.0 incluye soporte para Xen

3.2.5.3. Desarrollo de los Módulos en el software de virtualización de ordenadores VirtualBox

- **Módulo 1**

Para la construcción del módulo 1 se va a proceder con la instalación del sistema operativo base CentOs 5.3 para a seguidamente instalar el software de virtualización VirtualBox, y así evaluar las facilidades que éste brinda en su ejecución.

Instalar Virtual box, que aparte de ser GPL, es fácil de usar aunque esté en inglés. Para ello debemos de bajarnos el paquete correspondiente. Desde la página web de VirtualBox la última versión.

La instalación es muy sencilla, se puede hacer en modo texto a través de la consola.

Para la instalación mediante la consola pulsamos doble click sobre el archivo descargado y se nos abre el instalador de paquetes, en caso de requerir 2 dependencias o paquetes, se instalará de forma automática.

VirtualBox es una aplicación que nos permite ejecutar diferentes Sistemas Operativos simultáneamente en el mismo equipo de manera virtual. La razón para usar VirtualBox es porque es libre (GNU/GPL) y porque funciona bien.

A continuación el procedimiento de instalación del software de virtualización VirtualBox:

Previo a la instalación del VirtualBox es necesario instalar algunos paquetes, tales como:

- **INSTALAR EL PAQUETE qt:**

```
[root@vboxtest CentOS]# rpm -ivh qt4-4.2.1-1.i386.rpm
warning: qt4-4.2.1-1.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
   1:qt4                    ########################################### [100%]
```

- **INSTALAR EL PAQUETE SDL:**

```
[root@vboxtest CentOS]# rpm -ivh SDL-1.2.10-8.el5.i386.rpm
warning: SDL-1.2.10-8.el5.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
        package SDL-1.2.10-8.el5.i386 is already installed
```

- **INSTALAR EL PAQUETE kernel-headers:**

```
[root@vboxtest CentOS]# rpm -ivh kernel-headers-2.6.18-128.el5.i386.rpm
warning: kernel-headers-2.6.18-128.el5.i386.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
        package kernel-headers-2.6.18-128.el5.i386 is already installed
```

- **INSTALAR EL PAQUETE kernel-devel:**

```
[root@vboxtest CentOS]# rpm -ivh kernel-devel-2.6.18-128.el5.i686.rpm
warning: kernel-devel-2.6.18-128.el5.i686.rpm: Header V3 DSA signature: NOKEY, key ID e8562897
Preparing...                ########################################### [100%]
        package kernel-devel-2.6.18-128.el5.i686 is already installed
```

- **INSTALAR EL PAQUETE dkms:**

```
[root@vboxtest tmp]# rpm -ivh dkms-2.0.17.5-1.el5.kb.noarch.rpm
Preparing...                ########################################### [100%]
   1:dkms                   ########################################### [100%]
```

- **INSTALAR EL PAQUETE VirtuaLBox:**

```
[root@vboxtest tmp]# rpm -ivh VirtualBox-3.1-3.1.2_56127_rhel5-1.i386.rpm
warning: VirtualBox-3.1-3.1.2_56127_rhel5-1.i386.rpm: Header V4 DSA signature: NOKEY, key ID 6dfbcbae
Preparing...                ########################################### [100%]
   1:VirtualBox-3.1         ########################################### [100%]

Creating group 'vboxusers'. VM users must be member of that group!
No precompiled module for this kernel found -- trying to build one. Messages
emitted during module compilation will be logged to /var/log/vbox-install.log.
Success!
```

Una vez instalado accedemos digitando la siguiente instrucción:

```
[root@vboxtest tmp]#virtualbox
```

Seguidamente se procede con: Aceptar la licencia, Registro de uso,

- **Módulo 2**

- **Guest de 64-bits**

Desde la versión 2.0, VirtualBox soporta sistemas operativos guest de 64-bit. Partiendo por la versión 2.1, incluso pude ejecutar guest de 64-bits en un sistema operativo host de 32-bit. En particular, los guest de 64-bits son soportados por debajo de las siguientes codiciones:

- ✓ Se requiere un procesador de 64-bits con hardware que soporte virtualización.
- ✓ Habilitar la virtualización de hardware para los servidores virtuales particulares para los cuales se desee el soporte de 64-bits; el software de virtualización no es soportado para servidores virtuales de 64-bits.
- ✓ Si se desea usar un guest de 64-bits que soporte en un sistema host de 32 bits, se podría también seleccionar un sistema operativo de 64 bits para un servidor virtual particular. Desde el soporte de 64 bits en host de 32 bits supone una sobrecarga adicional, VirtualBox solamente habilita este soporte a petición explícita.

El soporte de host de 64-bits y guest de 64-bits es siempre habilitado, por lo que se podría simplemente instalar un sistema operativo en el guest.

Los guest de 64-bits sobre sistemas host de 32-bits con VT-x puede causar inestabilidad en el sistema.

- **Módulo 3**

- **Importando y exportando servidores virtuales**

A partir de la versión 2.2, VirtualBox puede importar y exportar servidores virtuales en el formato abierto de virtualización industrial-estándar (OVF).

OVF es un estándar multi-plataforma soportada por muchos productos de virtualización los cuales permiten la restaurar de servidores virtuales existentes, es decir que pueden ser importandos en un virtualizador como VirtualBox. A diferencia de otros software de virtualización, VirtualBox ahora soporta OVF con una fácil interfaz gráfica de usuario, así como el uso de la línea de comandos. Esto permite el empaquetamiento denominado ***dispositivos virtual***: las imágenes de discos junto con la ajustes que se pueden distribuir facilmente. De esta manera se puede ofrecer una lista completa de paquetes de software (sistemas operativos con aplicaciones) no se requiere configurar o instalar, salvo para importar en VirtualBox.

1.- Una de las muchas imágenes de disco, normalmente en el formato VMDK ampliamente utilizado.

2.- Una description textual del archivo en un lenguaje XML con una extensión .ovf.

Estos archivos deben residir en el mismo directorio para VirtualBox para poder importar éstos.

Algunas propiedades de los servidores virtuales soportados por el propio formato del archivo XML no es exportado. Como resultado, a la hora de exportar y luego volver a importar un servidor virtual con VirtuaBox, los ajustes no deben ser idénticos. Esto es especialmente cierto para el I/O APIC establecimiento, la aceleración 3D, la virtualización de hardware, paginación anidada y propiedades de otro servidores virtuales.

Localización OVF idiomas (múltiples lenguajes en un solo archivo OVF) todavía no es posible. Algunos sectores, como OVF StartupSection, DeploymentOptionSection y InstallSection son ignorados. Documentos del ambiente OVF, incluidas sus propiedad de secciones configuración de aplicaciones con las imágenes ISO, no están todavía soportados. Archivos OVA (contenedores TAR) no están todavía soportados. Los archivos remotos a través de HTTP o de otros mecanismos no están todavía soportados.

- **Módulo 4**

- **Herramienta Virtual Media Manager**

Mantiene un registro de todos los discos duros, CD/DVD-ROM y unidades de floppy disponibles. Este registro puede ser visto y modificado en el Virtual Media Manager, al cual podemos acceder desde el menú ***File*** de la ventana principal de VirtualBox:

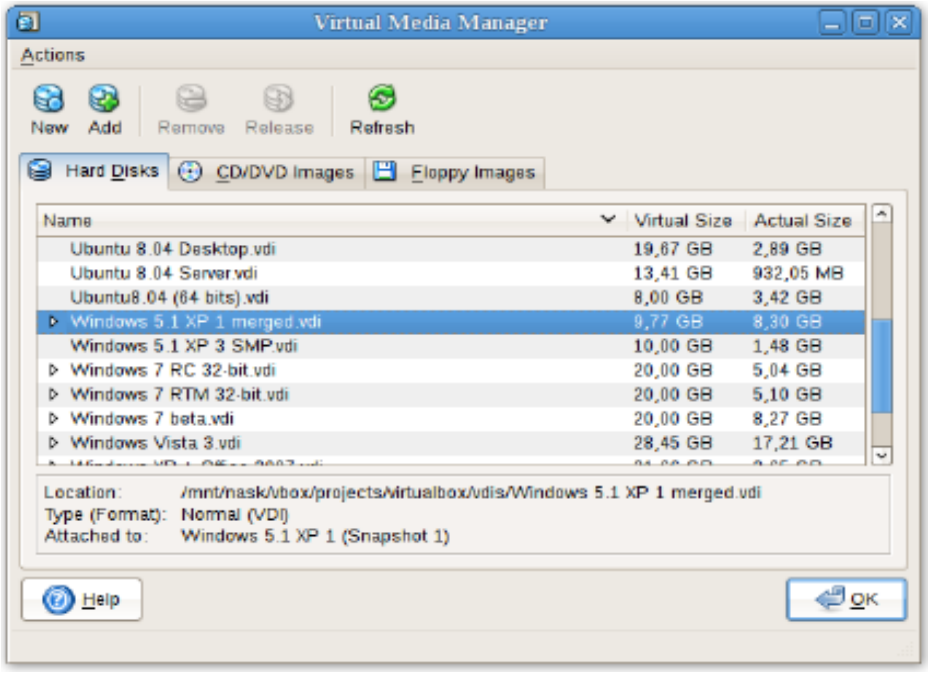

Figura III.39. VirtualBox: Registro de Uso

La ventana muestra todas las imágenes que están actualmente registradas con VirtualBox, convenientemente agrupados en las tres pestañas para los tres recursos hardware. Estos son:

- ✓ **Imágenes de Discos Duros,** están disponibles las imágenes propias del VirtualBox en el formato Virtual Disk Image (VDI) o en los formatos de terceros mencionados anteriormente.
- ✓ **Imágenes de CD/DVD,** en formato estándar ISO.
- ✓ **Imágenes floppy,** en formato RAW estándar.

Para cada imagen desplegada en la herramienta Virtual Media Manager, se puede visualizar el path completo del archivo de la imagen y otra información, como la imagen del servidor virtual que está actualmente atachada a éste.

Mediante la herramienta de Virtual podemos:

- ✓ Crear una nueva imagen del disco duro usando el botón ***New,*** éste abrirá el wizard ***Create Disk Image.***
- ✓ Importar archivos de imágenes existentes desde el disco duro dentro de VirtualBox usando el botón ***Add.***
- ✓ Remover una imagen desde el registro.
- ✓ Liberar una imagen, es decir, separarlo de un servidor virtual si es en la actualidad unida a una como un disco duro virtual.

➢ **Módulo 5**

La navegación de VirtualBox como software de virtualización de ordenadores, se ve detalla a continuación en el proceso de creación de servidores virtuales:

- **Consola de administración**

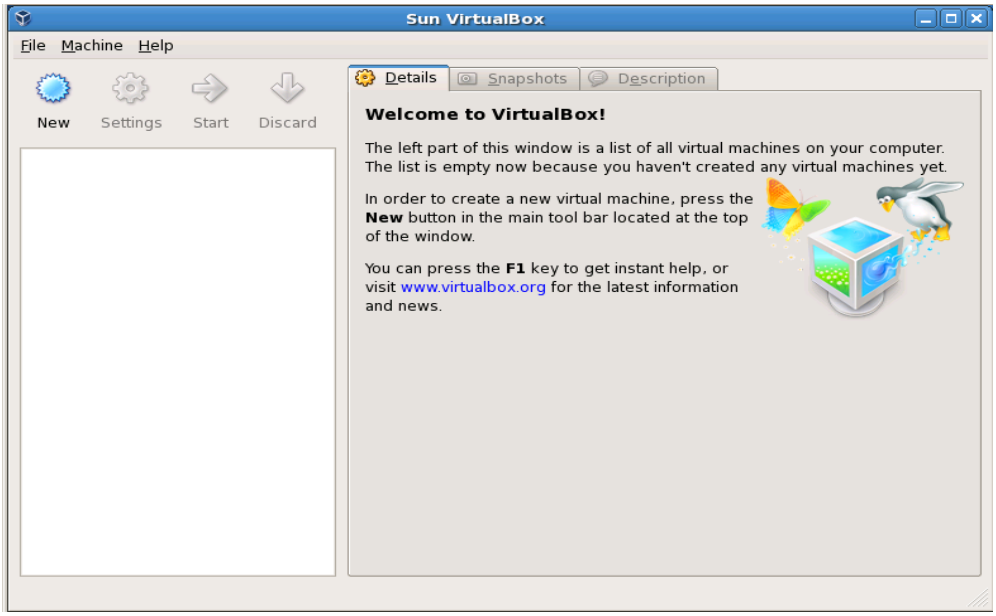

Figura III.40. VirtualBox: Consola de Trabajo

Figura III.41. VirtualBox: Nombre y Tipo de Sistema Operativo para el Servidor Virtual

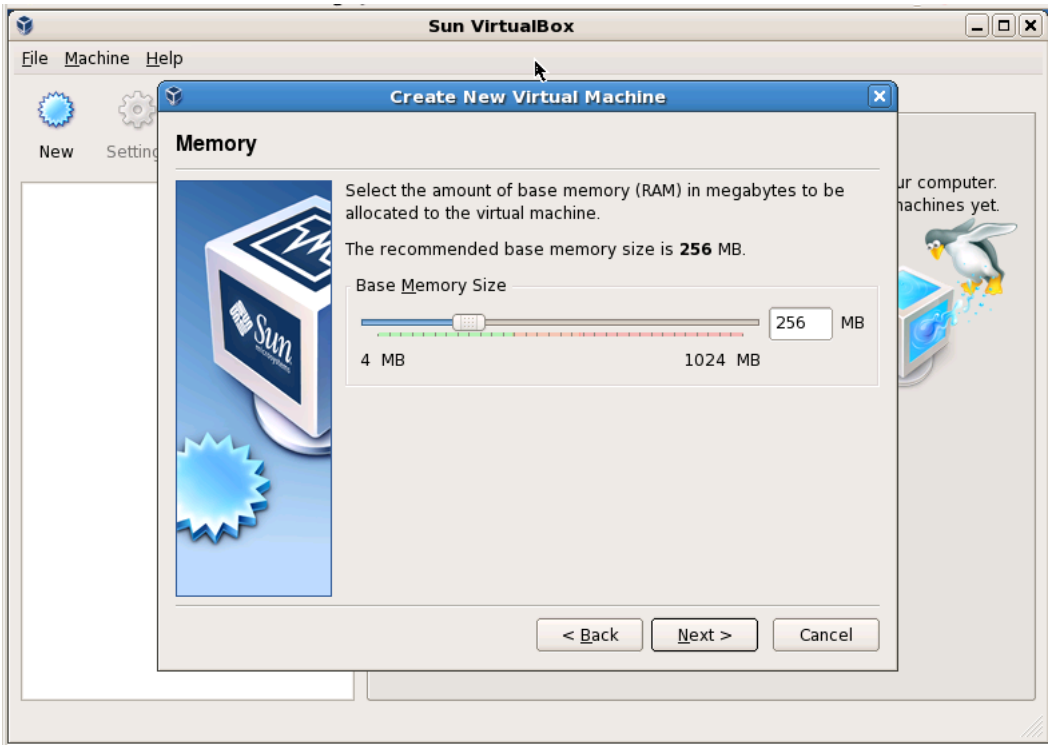

Figura III.42. VirtualBox: Asignación de Memoria para el Servidor Virtual

Figura III.43. VirtualBox: Creación de Disco Duro para el Servidor Virtual

Figura III.44. VirtualBox: Tipo de Disco Duro para Servidor Virtual

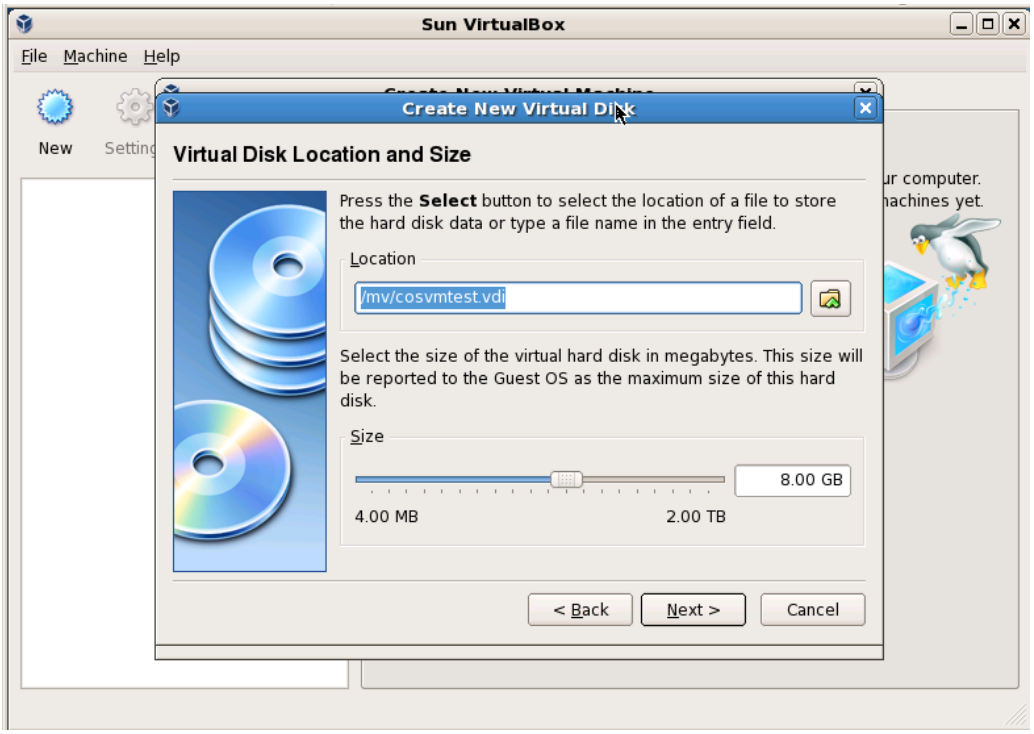

Figura III.45. VirtualBox: Ubicación y Tamaño del Disco Duro para Servidor Virtual

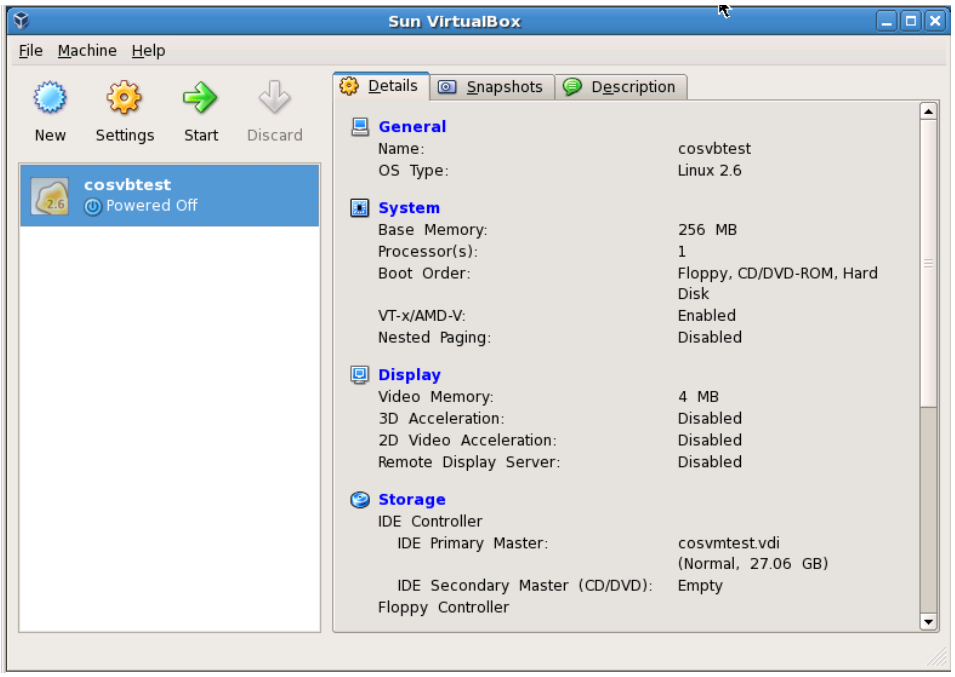

Figura III.46. VirtualBox: Detalles Servidor Virtual

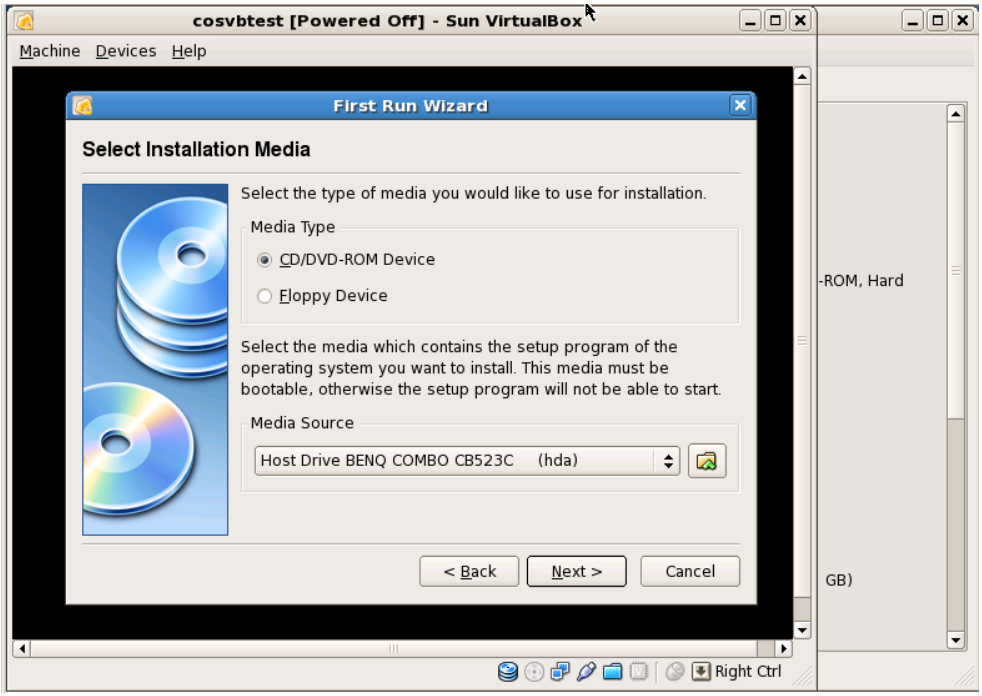

Figura III.47. VirtualBox: Selección del Medio de Instalación

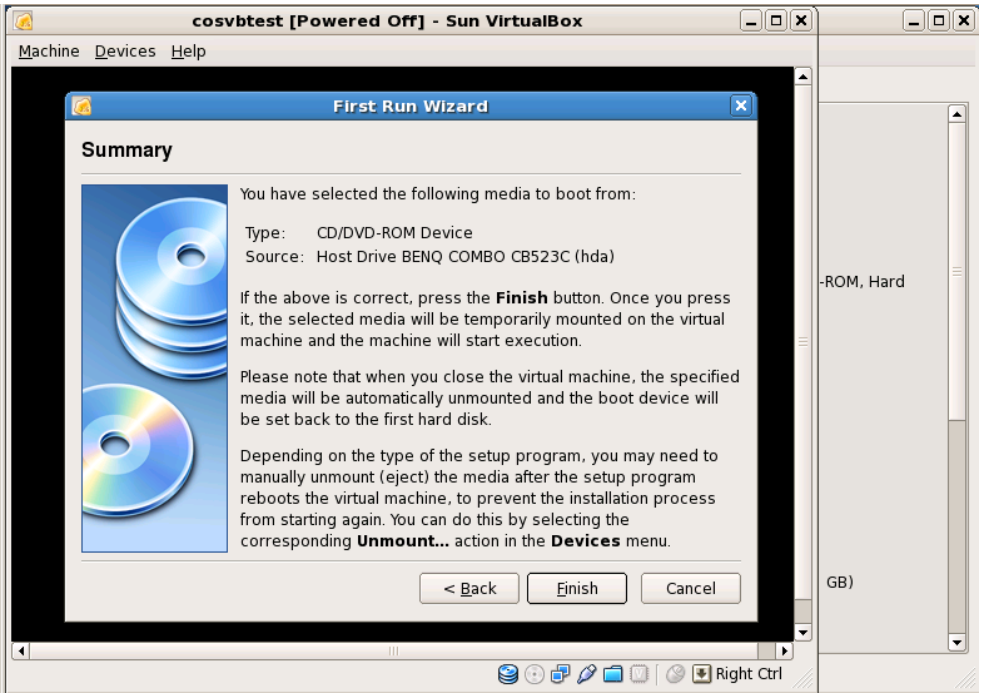

Figura III.48. VirtualBox: Confirmación del Medio de Instalación

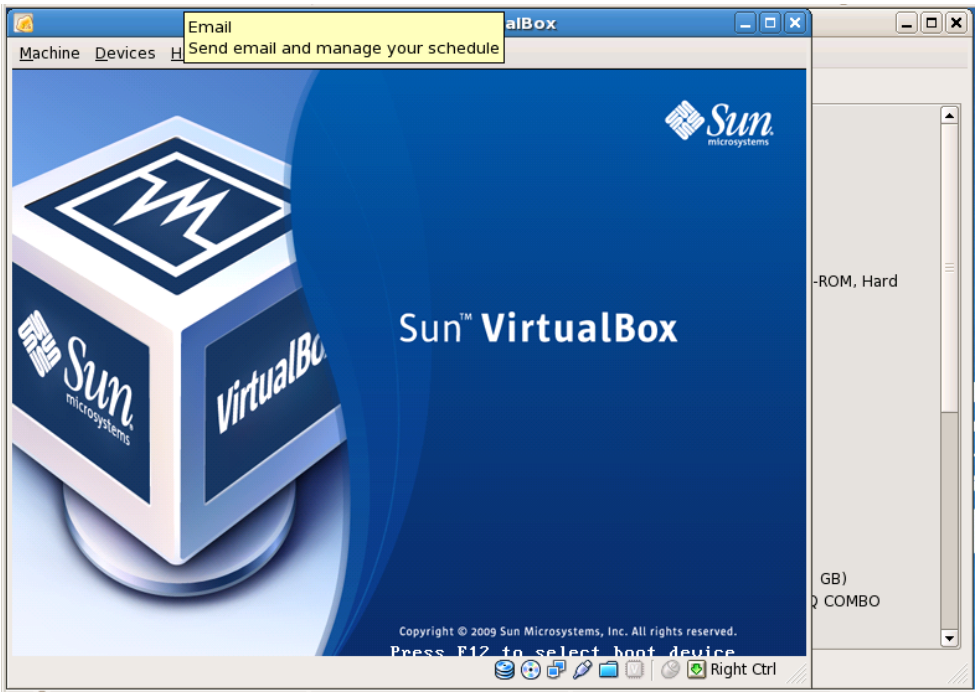

Figura III.49. VirtualBox: Arrancando Servidor Virtual

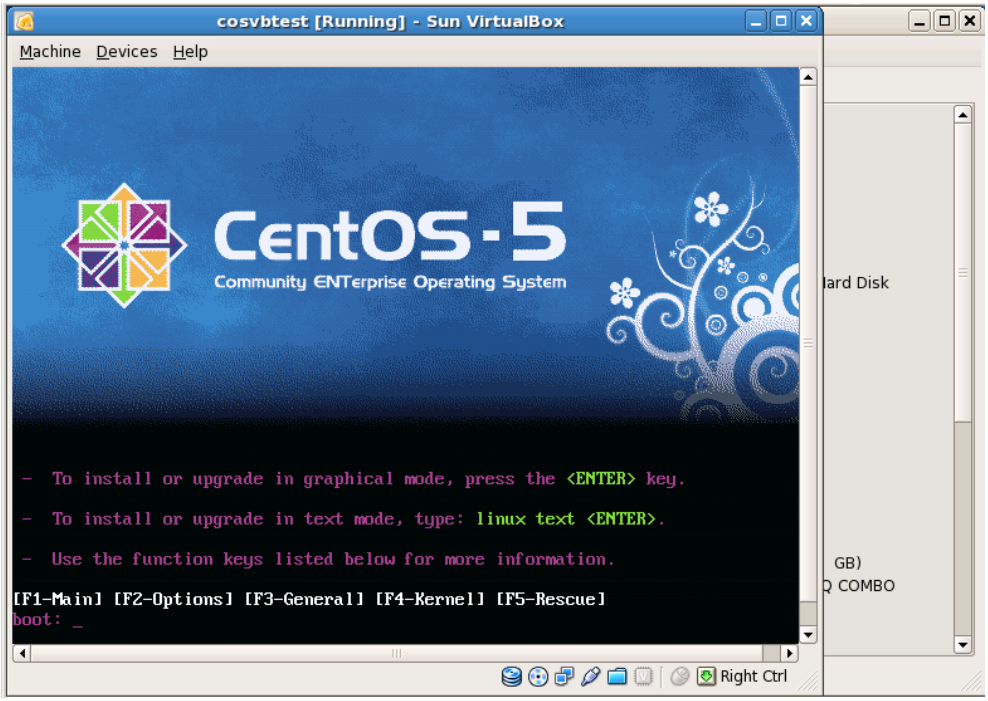

Figura III.50. VirtualBox: Iniciando Instalación Sistema Operativo en el Servidor Virtual

- **Módulo 6**

Procedemos con la instalación del utilitario stress tal como se indica a continuación:

```
[root@cosvboxtest tmp]# tar xvf stress-1.0.2.tar.tar
[root@cosvboxtest tmp]# cd stress-1.0.2
[root@cosvboxtest stress-1.0.2]#   ./configure && make && sudo make install
stress: info: [5705] successful run completed in 305s
```

Los parámetros con los que vamos a evaluar será un promedio de cuatro que se impone en el sistema especificando -c procesos limitados por CPU, -i de I/O-bound procesos, -m procesos asignados de memoria y –d procesos de disco. A continuación la ejecución del mismo:

Procedemos a ejecutar de la siguiente manera:

```
[root@cosvboxtest ~]# date;stress -c 1 -i 1 -m 1 -d 1 --verbose --timeout 5m;date
Sat Dec 26 13:04:17 ECT 2009
stress: info: [3727] dispatching hogs: 1 cpu, 1 io, 1 vm, 1 hdd
```

```
stress: dbug: [3727] using backoff sleep of 12000us
stress: dbug: [3727] setting timeout to 300s
stress: dbug: [3727] --> hogcpu worker 1 [3728] forked
stress: dbug: [3727] --> hogio worker 1 [3729] forked
stress: dbug: [3727] --> hogvm worker 1 [3730] forked
stress: dbug: [3727] --> hoghdd worker 1 [3731] forked
stress: dbug: [3730] allocating 268435456 bytes ...
stress: dbug: [3730] touching bytes in strides of 4096 bytes ...
stress: dbug: [3731] seeding 1048575 byte buffer with random data
stress: dbug: [3731] opened ./stress.S6WF5Z for writing 1073741824 bytes
stress: dbug: [3731] unlinking ./stress.S6WF5Z
stress: dbug: [3731] fast writing to ./stress.S6WF5Z
stress: dbug: [3731] slow writing to ./stress.S6WF5Z
stress: dbug: [3731] closing ./stress.S6WF5Z after 1073741824 bytes
stress: dbug: [3731] opened ./stress.86ghkb for writing 1073741824 bytes
stress: dbug: [3731] unlinking ./stress.86ghkb
stress: dbug: [3731] fast writing to ./stress.86ghkb
stress: dbug: [3727] <-- worker 3728 signalled normally
stress: dbug: [3727] <-- worker 3730 signalled normally
stress: dbug: [3727] <-- worker 3729 signalled normally
stress: dbug: [3727] <-- worker 3731 signalled normally
stress: info: [3727] successful run completed in 305s
Sat Dec 26 13:09:21 ECT 2009
```

```
[root@cosvboxtest ~]# date;stress -c 2 -i 2 -m 2 -d 2 --verbose --timeout 5m;date
Sat Dec 26 13:22:10 ECT 2009
stress: info: [3817] dispatching hogs: 2 cpu, 2 io, 2 vm, 2 hdd
stress: dbug: [3817] using backoff sleep of 24000us
stress: dbug: [3817] setting timeout to 300s
stress: dbug: [3817] --> hogcpu worker 2 [3818] forked
stress: dbug: [3817] --> hogio worker 2 [3819] forked
stress: dbug: [3817] --> hogvm worker 2 [3820] forked
stress: dbug: [3820] allocating 268435456 bytes ...
stress: dbug: [3820] touching bytes in strides of 4096 bytes ...
stress: dbug: [3817] --> hoghdd worker 2 [3821] forked
stress: dbug: [3817] using backoff sleep of 12000us
stress: dbug: [3817] setting timeout to 299s
stress: dbug: [3817] --> hogcpu worker 1 [3822] forked
stress: dbug: [3817] --> hogio worker 1 [3823] forked
stress: dbug: [3817] --> hogvm worker 1 [3824] forked
stress: dbug: [3817] --> hoghdd worker 1 [3825] forked
stress: dbug: [3821] seeding 1048575 byte buffer with random data
stress: dbug: [3821] opened ./stress.CrVQAf for writing 1073741824 bytes
stress: dbug: [3821] unlinking ./stress.CrVQAf
stress: dbug: [3821] fast writing to ./stress.CrVQAf
stress: dbug: [3824] allocating 268435456 bytes ...
stress: dbug: [3824] touching bytes in strides of 4096 bytes ...
stress: dbug: [3825] seeding 1048575 byte buffer with random data
stress: dbug: [3825] opened ./stress.KavkyU for writing 1073741824 bytes
stress: dbug: [3825] unlinking ./stress.KavkyU
stress: dbug: [3825] fast writing to ./stress.KavkyU
stress: dbug: [3817] <-- worker 3824 signalled normally
stress: dbug: [3817] <-- worker 3818 signalled normally
stress: dbug: [3817] <-- worker 3820 signalled normally
stress: dbug: [3817] <-- worker 3821 signalled normally
stress: dbug: [3817] <-- worker 3822 signalled normally
stress: dbug: [3817] <-- worker 3819 signalled normally
stress: dbug: [3817] <-- worker 3825 signalled normally
stress: dbug: [3817] <-- worker 3823 signalled normally
stress: info: [3817] successful run completed in 308s
Sat Dec 26 13:27:19 ECT 2009
```

```
[root@cosvboxtest ~]# date;stress -c 3 -i 3 -m 3 -d 3 --verbose --timeout 5m;date
Sat Dec 26 13:27:38 ECT 2009
stress: info: [3851] dispatching hogs: 3 cpu, 3 io, 3 vm, 3 hdd
stress: dbug: [3851] using backoff sleep of 36000us
stress: dbug: [3851] setting timeout to 300s
stress: dbug: [3851] --> hogcpu worker 3 [3852] forked
stress: dbug: [3851] --> hogio worker 3 [3853] forked
```

```
stress: dbug: [3851] --> hogvm worker 3 [3854] forked
stress: dbug: [3851] --> hoghdd worker 3 [3855] forked
stress: dbug: [3851] using backoff sleep of 24000us
stress: dbug: [3851] setting timeout to 300s
stress: dbug: [3851] --> hogcpu worker 2 [3856] forked
stress: dbug: [3851] --> hogio worker 2 [3857] forked
stress: dbug: [3851] --> hogvm worker 2 [3858] forked
stress: dbug: [3851] --> hoghdd worker 2 [3859] forked
stress: dbug: [3851] using backoff sleep of 12000us
stress: dbug: [3851] setting timeout to 299s
stress: dbug: [3854] allocating 268435456 bytes ...
stress: dbug: [3854] touching bytes in strides of 4096 bytes ...
stress: dbug: [3858] allocating 268435456 bytes ...
stress: dbug: [3858] touching bytes in strides of 4096 bytes ...
stress: dbug: [3855] seeding 1048575 byte buffer with random data
stress: dbug: [3851] --> hogcpu worker 1 [3860] forked
stress: dbug: [3851] --> hogio worker 1 [3861] forked
stress: dbug: [3851] --> hogvm worker 1 [3862] forked
stress: dbug: [3851] --> hoghdd worker 1 [3863] forked
stress: dbug: [3859] seeding 1048575 byte buffer with random data
stress: dbug: [3862] allocating 268435456 bytes ...
stress: dbug: [3862] touching bytes in strides of 4096 bytes ...
stress: dbug: [3855] opened ./stress.7I0MXp for writing 1073741824 bytes
stress: dbug: [3855] unlinking ./stress.7I0MXp
stress: dbug: [3855] fast writing to ./stress.7I0MXp
stress: dbug: [3859] opened ./stress.EP3dQ2 for writing 1073741824 bytes
stress: dbug: [3859] unlinking ./stress.EP3dQ2
stress: dbug: [3859] fast writing to ./stress.EP3dQ2
stress: dbug: [3863] seeding 1048575 byte buffer with random data
stress: dbug: [3863] opened ./stress.l2vNCN for writing 1073741824 bytes
stress: dbug: [3863] unlinking ./stress.l2vNCN
stress: dbug: [3863] fast writing to ./stress.l2vNCN
stress: dbug: [3851] <-- worker 3854 signalled normally
stress: dbug: [3851] <-- worker 3852 signalled normally
stress: dbug: [3851] <-- worker 3858 signalled normally
stress: dbug: [3851] <-- worker 3856 signalled normally
stress: dbug: [3851] <-- worker 3860 signalled normally
stress: dbug: [3851] <-- worker 3862 signalled normally
stress: dbug: [3851] <-- worker 3855 signalled normally
stress: dbug: [3851] <-- worker 3859 signalled normally
stress: dbug: [3851] <-- worker 3861 signalled normally
stress: dbug: [3851] <-- worker 3857 signalled normally
stress: dbug: [3851] <-- worker 3853 signalled normally
stress: dbug: [3851] <-- worker 3863 signalled normally
stress: info: [3851] successful run completed in 329s
Sat Dec 26 13:33:08 ECT 2009
```

Después de la ejecuación del stress se tiene el siguiente análisis respecto al consumo de: load average, tasks, cpu, memoria, swap:

- **LOAD AVERAGE**

Con una carga inicial de 1 todas las tareas de stress finalizan bien de igual manera para la carga 2; cuando se aumenta la carga a 3, existe un encolamiento superior a 25 procesos durante todo el tiempo de ejecución de la prueba de stress.

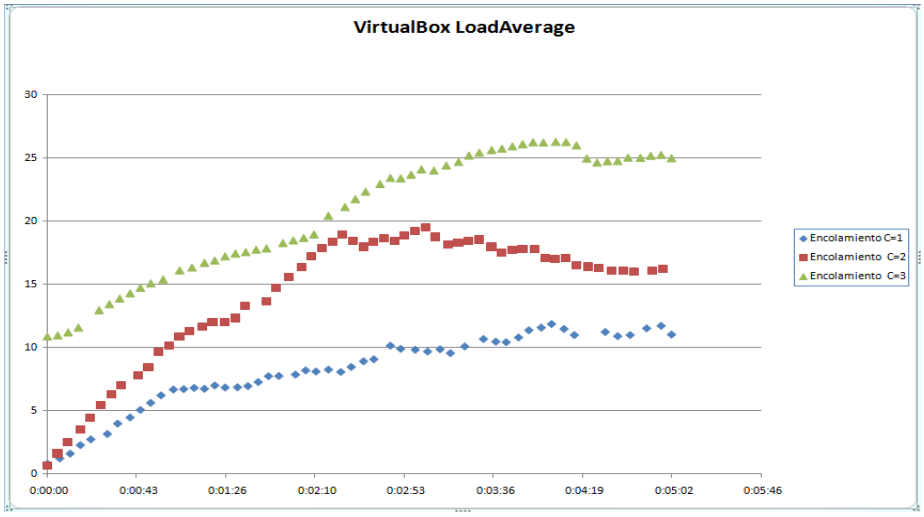

Gráfico III.13. VirtualBox: Load Average

- **TASKS – RUNNING**

Con una carga final de hasta 3 en utilización de procesos CPU, I/O, memoria y disco, el número máximo de procesos que llega a alcanzar es de 25 en el tiempo de ejecución del utilitario stress.

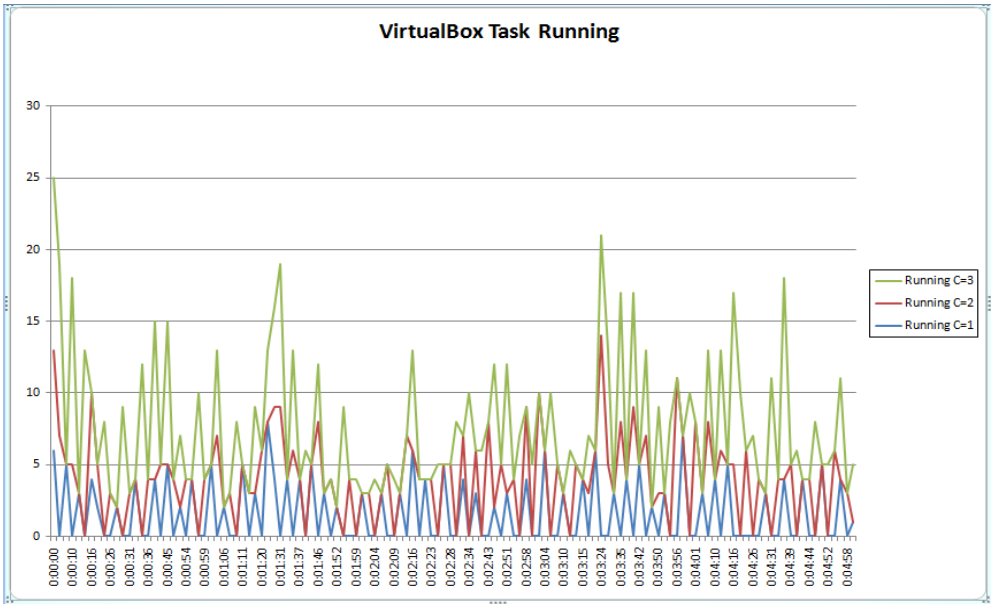

Gráfico III.14. VirtualBox: Tasks - Running

- **SLEEPING**

Los procesos que están esperando para ser ejecutados se encuentran en un rango de pasado 50 a 250 debido a que a la lentitud en el despacho de los procesos existentes, esto se evidencia con una carga final de hasta 3 en utilización de procesos CPU, I/O, memoria y disco.

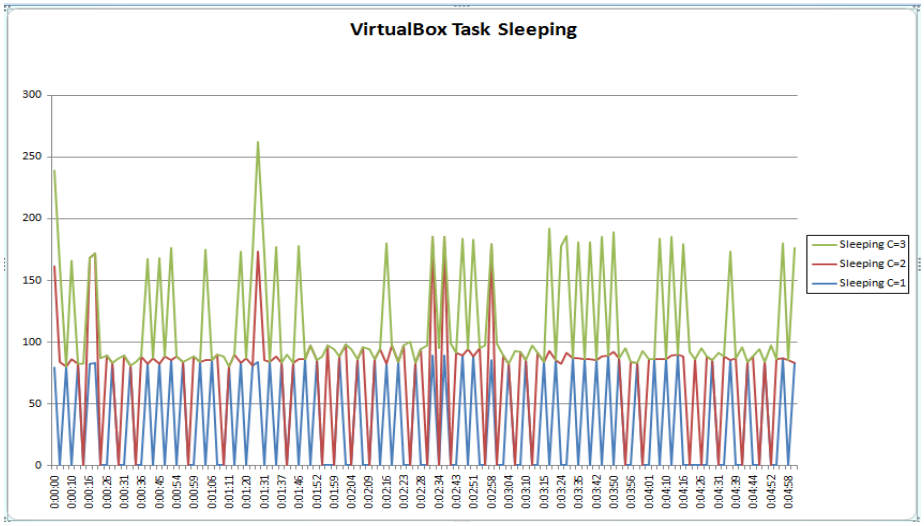

Gráfico III.15. VirtualBox: Tasks – Sleeping

- **CPU**

El consumo del recurso CPU llega a una utilización del 100% con los tres valores de parámetros CPU, I/O, memoria y disco.

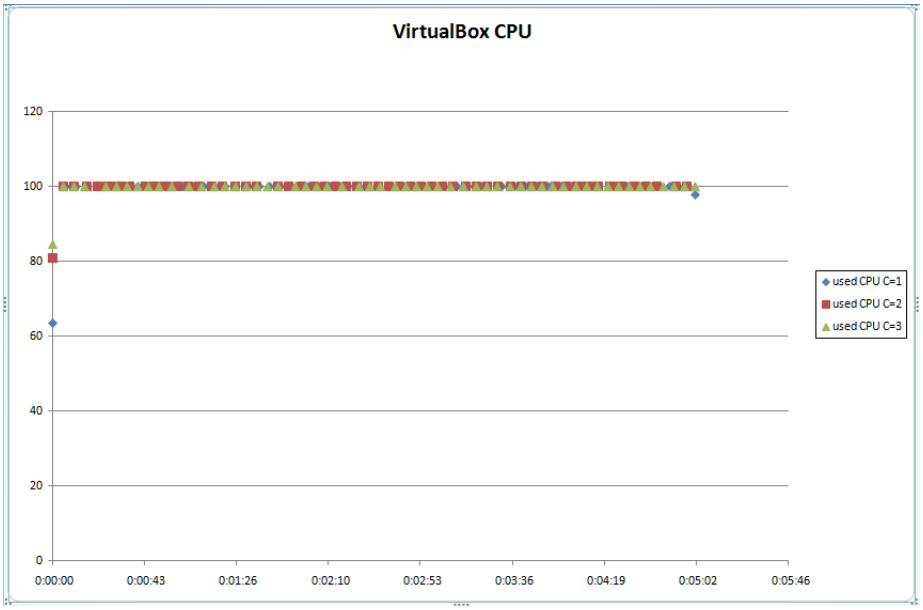

Gráfico III.16. VirtualBox: CPU

- **MEMORIA**

La utilización del recurso memoria llega a su utilización máxima del total de la memoria asignanda en kilobytes (kb) 255556k.

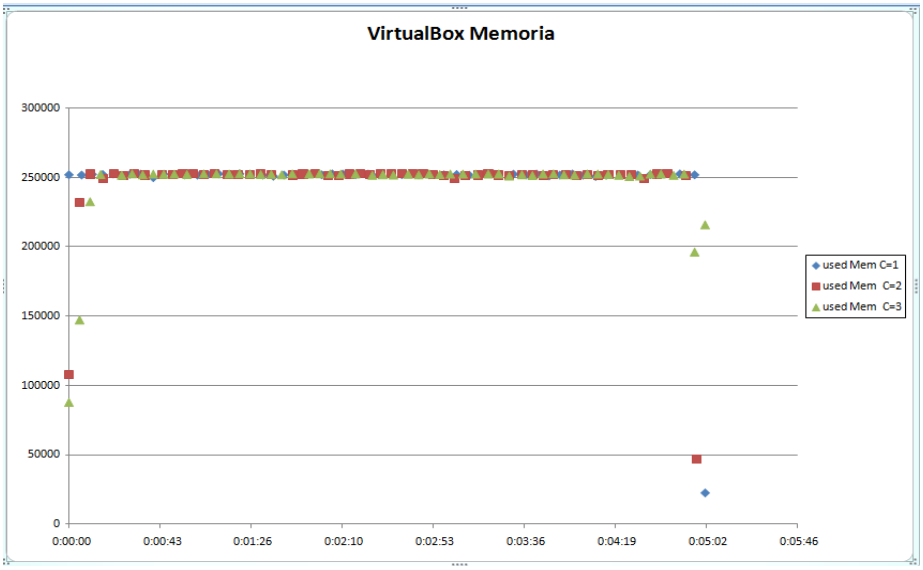

Gráfico III.17. VirtualBox: Memoria

- **SWAP**

La utilización del recurso swap se presencia a partir del parámetro 3, donde el servidor virtual requiere la utilización del swap disponible llegando a obtener 658800k, y un total de 1052248k.

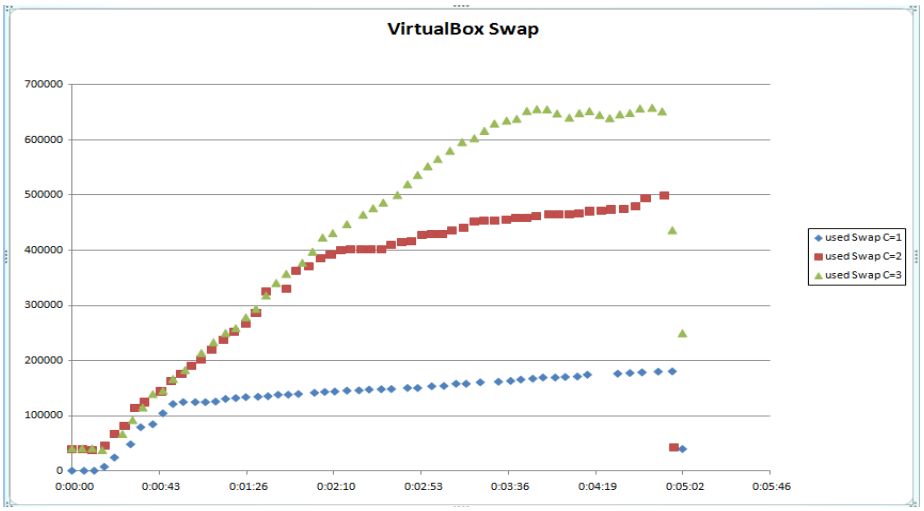

Gráfico III.18. VirtualBox: Tasks – Swap

- **Módulo 7**

Se considera la utilización del utilitario Unixbench, a continuación se detalla su desarrollo:

La instalación del utilitario Unixbench consta de siguientes pasos:

```
[root@cosvboxtest tmp]# tar zxvf unixbench-4.1.0.gz
[root@cosvboxtest tmp]# cd unixbench-4.1.0
[root@cosvboxtest unixbench-4.1.0]# make
[root@cosvboxtest unixbench-4.1.0]# ./Run
========================================================
BYTE UNIX Benchmarks (Version 4.1.0)
  System -- Linux cosvboxtest 2.6.18-128.el5 #1 SMP Wed Jan 21 10:44:23 EST 2009 i686 i686 i386 GNU/Linux
  Start Benchmark Run: Sat Dec 26 14:01:54 ECT 2009
   2 interactive users.
   14:01:54 up  1:17,  2 users,  load average: 1.84, 5.15, 13.79
  lrwxrwxrwx 1 root root 4 Dec 26 09:50 /bin/sh -> bash
  /bin/sh: symbolic link to `bash'
  /dev/hda3             26361176   3916040  21084432  16% /
Dhrystone 2 using register variables     1593968.2 lps   (10.0 secs, 10 samples)
Double-Precision Whetstone                 523.9 MWIPS (9.9 secs, 10 samples)
System Call Overhead                       101427.4 lps   (10.0 secs, 10 samples)
Pipe Throughput                            114229.1 lps   (10.0 secs, 10 samples)
```

```
Pipe-based Context Switching          1468.2 lps   (10.0 secs, 10 samples)
Process Creation                        93.7 lps   (30.0 secs, 3 samples)
Execl Throughput                        61.1 lps   (29.6 secs, 3 samples)
File Read 1024 bufsize 2000 maxblocks   138778.0 KBps  (30.0 secs, 3 samples)
File Write 1024 bufsize 2000 maxblocks  109196.0 KBps  (30.0 secs, 3 samples)
File Copy 1024 bufsize 2000 maxblocks    51053.0 KBps  (30.0 secs, 3 samples)
File Read 256 bufsize 500 maxblocks      48988.0 KBps  (30.0 secs, 3 samples)
File Write 256 bufsize 500 maxblocks     30629.0 KBps  (30.0 secs, 3 samples)
File Copy 256 bufsize 500 maxblocks      16996.0 KBps  (30.0 secs, 3 samples)
File Read 4096 bufsize 8000 maxblocks   290573.0 KBps  (30.0 secs, 3 samples)
File Write 4096 bufsize 8000 maxblocks  263307.0 KBps  (30.0 secs, 3 samples)
File Copy 4096 bufsize 8000 maxblocks   115930.0 KBps  (30.0 secs, 3 samples)
Shell Scripts (1 concurrent)          118.8 lpm   (60.1 secs, 3 samples)
Shell Scripts (8 concurrent)           16.7 lpm   (60.0 secs, 3 samples)
Shell Scripts (16 concurrent)           8.0 lpm   (60.1 secs, 3 samples)
Arithmetic Test (type = short)        268466.7 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = int)          280646.4 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = long)         285846.9 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = float)        240003.8 lps   (10.0 secs, 3 samples)
Arithmetic Test (type = double)       240834.6 lps   (10.0 secs, 3 samples)
Arithoh                               101366621.6 lps   (10.0 secs, 3 samples)
C Compiler Throughput                  88.3 lpm   (60.0 secs, 3 samples)
Dc: sqrt(2) to 99 decimal places       2026.5 lpm   (30.0 secs, 3 samples)
Recursion Test--Tower of Hanoi         38992.3 lps   (20.0 secs, 3 samples)
```

Resultados de la ejecución:

Tabla III.12 VirtualBox – UnixBench - Index Values

INDEX VALUES			
TEST	**BASELINE**	**RESULT**	**INDEX**
Dhrystone 2 using register variables	116700.0	1593968.2	136.6
Double-Precision Whetstone	55.0	523.9	95.3
Execl Throughput	43.0	61.1	14.2
File Copy 1024 bufsize 2000 maxblocks	3960.0	51053.0	128.9
File Copy 256 bufsize 500 maxblocks	1655.0	16996.0	102.7
File Copy 4096 bufsize 8000 maxblocks	5800.0	115930.0	199.9
Pipe Throughput	12440.0	114229.1	91.8
Process Creation	126.0	93.7	7.4
Shell Scripts (8 concurrent)	6.0	16.7	27.8
System Call Overhead	15000.0	101427.4	67.6
			========
FINAL SCORE			60.2

El rendimiento obtenido con el utilitario de Unixbench es de 60.2 puntos.

Utilización del benchmark Netperf en el servidor host para el análisis de redes se detalla a continuación:

```
 [root@vboxtest tmp]# tar zxvf netperf-2.4.5.tar.gz
[root@vboxtest tmp]# cd netperf-2.4.5
[root@vboxtest netperf-2.4.5]# ./configure
[root@vboxtest netperf-2.4.5]# make
[root@vboxtest netperf-2.4.5]# make install
[root@vboxtest netperf-2.4.5]# vi /etc/services
netperf 12865/tcp
[root@vboxtest netperf-2.4.5]# netserver
Starting netserver at port 12865
Starting netserver at hostname 0.0.0.0 port 12865 and family AF_UNSPEC
```

De configuración similar se instala en el servidor guest:

```
[root@cosvboxtest tmp]# tar zxvf netperf-2.4.5.tar.gz
[root@cosvboxtest tmp]# cd netperf-2.4.5
[root@cosvboxtest netperf-2.4.5]# ./configure
[root@cosvboxtest netperf-2.4.5]# make install
[root@cosvboxtest netperf-2.4.5]# vi /etc/services
[root@cosvboxtest netperf-2.4.5]# netserver
Starting netserver at port 12865
Starting netserver at hostname 0.0.0.0 port 12865 and family AF_UNSPEC
```

Ejecución del benchmark NetPerf:

```
[root@vboxtest netperf-2.4.5]# netperf -H cosvboxtest
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvboxtest (192.168.0.175) port 0 AF_INET
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    10.25      80.85
```

```
[root@vboxtest netperf-2.4.5]# netperf -H cosvboxtest -t UDP_RR
UDP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvboxtest (192.168.0.175) port 0 AF_INET
Local /Remote
Socket Size   Request  Resp.   Elapsed  Trans.
Send   Recv   Size     Size    Time     Rate
bytes  Bytes  bytes    bytes   secs.    per sec

109568 109568 1        1       10.00     693.24
109568 109568
```

```
[root@vboxtest netperf-2.4.5]# netperf -H cosvboxtest -t UDP_STREAM -- -m 1024
UDP UNIDIRECTIONAL SEND TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvboxtest (192.168.0.175) port 0 AF_INET
Socket  Message  Elapsed      Messages
Size    Size     Time         Okay Errors   Throughput
bytes   bytes    secs            #      #   10^6bits/sec

109568    1024   10.00      114723      0      93.97
109568           10.00        5461             4.47
```

```
[root@vboxtest netperf-2.4.5]# netperf -H cosvboxtest -t TCP_RR
TCP REQUEST/RESPONSE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to cosvboxtest (192.168.0.175) port 0 AF_INET
Local /Remote
Socket Size   Request  Resp.   Elapsed  Trans.
Send   Recv   Size     Size    Time     Rate
bytes  Bytes  bytes    bytes   secs.    per sec

16384  87380  1        1       10.00     662.31
16384  87380
```

- **Módulo 8**

Se procede con la instalación del utilitario Bonnie:

```
[root@cosxentest tmp]# cd bonnie/
[root@cosxentest bonnie]# cd bonnie++-1.03e/
[root@cosxentest bonnie++-1.03e]# ./configure
[root@cosxentest bonnie++-1.03e]# make install
```

```
[root@cosvboxtest bonnie++-1.03e]# bonnie++ -d /tmp/bonnie -m cosxentest -u root
Using uid:0, gid:0.
Writing with putc()...done
Writing intelligently...done
Rewriting...done
Reading with getc()...done
Reading intelligently...done
start 'em...done...done...done...
Create files in sequential order...done.
Stat files in sequential order...done.
Delete files in sequential order...done.
Create files in random order...done.
Stat files in random order...done.
Delete files in random order...done.
Version 1.03e       ------Sequential Output------ --Sequential Input- --Random-
                    -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP K/sec %CP  /sec %CP
cosxentest     496M  7587  71  8229  15  7149  26  9787  71 23552  58 168.7  12
                    ------Sequential Create------ --------Random Create--------
                    -Create-- --Read--- -Delete-- -Create-- --Read--- -Delete--
              files  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP  /sec %CP
                 16  8278  70 +++++ +++ 17449  83 10848  83 +++++ +++ 14823  85
cosxentest,496M,7587,71,8229,15,7149,26,9787,71,23552,58,168.7,12,16,8278,70,+++++,+++,17449,83,10848,83,+++++,
+++,14823,85
```

- **Módulo 9**

- **Sistemas Operativos de host soportado**

Actualmente, VirtualBox se ejecuta en los siguientes sistemas operativos de host:

- ✓ **WINDOWS HOSTS:**

- Windows XP, all service packs (32-bit)
- Windows Server 2003 (32-bit)

- Windows Vista (32-bit and 64-bit1).
- Windows Server 2008 (32-bit and 64-bit)
- Windows 7 (32-bit and 64-bit)

✓ **MAC OS X HOSTS:**

- 10.5 (Leopard, 32-bit)
- 10.6 (Snow Leopard, 32-bit and 64-bit)

✓ **LINUX HOST (32-bit and 64-bit):**

- Debian GNU/Linux 3.1 (“sarge”), 4.0 (“etch”) and 5.0 (“lenny”)
- Fedora Core 4 to 11
- Gentoo Linux
- Redhat Enterprise Linux 4 and 5
- SUSE Linux 9 and 10, openSUSE 10.3, 11.0 and 11.1
- Ubuntu 6.06 (“Dapper Drake”), 6.10 (“Edgy Eft”), 7.04 (“Feisty Fawn”),
- 7.10 (“Gutsy Gibbon”), 8.04 (“Hardy Heron”), 8.10 (“Intrepid Ibex”), 9.04
- (“Jaunty Jackalope”).
- Mandriva 2007.1, 2008.0 and 2009.1

✓ **SOLARIS HOSTS (32-bit and 64-bit)**

- OpenSolaris (2008.05 and higher, “Nevada” build 86 and higher)
- Solaris 10 (u5 and higher)

3.2.6. Análisis Comparativo

En esta sección se va mostar el estudio de los software de virtualización de ordenadores VMware Server, Xen y VirtualBox a manera de cuadros comparativos, seguidos estos de una interpretación y calificación del criterio evaluado por parte del autor, estos cuadros comparativos se encuentran clasificados de acuerdo a los parámetros de comparación definidos anteriormente.

Para obtener resultados cuantitativos y cualitativos que permitan una selección sustentada de uno de lo software de virtualización de ordenadores analizados, la calificación de cada uno de los parámetros de comparación se basa en la siguiente escala:

Tabla III.13. Escala de Puntuación para calificación de Parámetros

Regular	Bueno	Muy Bueno	Excelente
< =70 %	> 70 % y < 80 %	>= 80% y < 90 %	> = 90 %

Cada uno de los ítems de la interpetación incluye la siguiente nomenclatura:

(x,y,z)/w en donde cada letra signifcica lo siguiente:

x: Representa el puntaje que obtiene el software de virtualización VMware.

y: Representa el puntaje de obtiene la herramienta Xen

z: Representa el puntaje de obtiene la herramienta VirtualBox

w: Representa la base del puntaje sobre la cual se esta calificando el parámetro.

La calificación definitiva del software de virtualización de ordenadores en base a cada parámetro de comparación se obtiene sumando los puntajes obtenidos del análisis, utilizando las siguientes fórmulas:

$Pvmwa = \sum(x)$, $Pxen = \sum(y)$, $Pvirbox = \sum(z)$, $Pc = \sum(w)$

Calificación de Vmware $(Cc - Vmwa) = (Pvmwa/Pc) * 100\%$

Calificación de Xen $(Cc - Xen) = (Pxen/Pc) * 100\%$

Calificación de VirtualBox $(Cc - Virbox) = (Pvirbox/Pc) * 100\%$

En donde:

$Pvmwa$: Puntaje acumulado por VMware en el parámetro.

$Pxen$: Puntaje acumulado por Xen en el parámetro.

$Pvirbox$: Puntaje acumulado por VirtualBox.

Pc: Puntaje sobre el que se califica el parámetro.

$Cc - Vmwa$: Porcentaje de la calificación total que obtuvo Vmware en el parámetro.

$Cc - Xen$: Porcentaje de la calificación total que obtuvo Xen en el parámetro.

$Cc - Vvirbox$: Porcentaje de la calificación total que obtuvo VirtualBox en el parámetro.

- **Instalación**

El proceso de instalación de un software de virtualización de ordenadores se puede analizar a partir de cada una de las herramientas de virtualización, además de los resultados que se obtengan durante la actividad, y de la forma en la que se puede obtener mayor provecho de las mismas, como se muestra a continuación:

- **Determinación de Variables**

a. Facilidad para instalar
b. Opciones para instalar
c. Despliegue de valores de variable

- **Valoraciones**

a. Variable Facilidad para instalar un software de virtualización.

La facilidad que presenta un software de virtualización de ordenadores para su instalación es importante para una mejor utilización por parte de los administradores de la infraestrura virtual (2 puntos).

b. Variable Opciones para instalar.

Las opciones para la instalación que presenta un software de virtualización de ordenadores es importante para poder realizar diferentes tareas en el proceso de instalación de la aplicación (3 puntos).

c. Variable Despliegue de valores de variables.

El despliegue de los valores que toman las variables que se utiliza en una instalación es importante para poder realizar un seguimiento y detectar posibles errores (2 puntos).

Tabla III.14. Instalación

Variable	VMware	Xen	VirtualBox
Facilidad para instalar un software de virtualización	Relativamente simple, se realizó en varios pasos: Instalar prerrequisitos	Bastante Simple, se realiza desde el instalador de CentOs.	Relativamente simple, se realizó en varios pasos: Instalar prerrequisitos.
Opciones para instalar	Se requiere tener instalado sistema operativo base, para ejecutar el rpm.	Presenta dos opciones para instalar: Se requiere tener instalado sistema operativo base o se instala a partir del instalador de Cent0s	Se requiere tener instalado sistema operativo base , para ejecutar el rpm.
Despliegue de valores de variables	Muestra algunas pantallas para el registro de la instalación e inspección del software.	Muestra varias pantallas para inspeccionar los valores de variables.	Muestra algunas pantallas para el registro de la instalación e inspección del software.

- **Interpretación**

✓ La facilidad para instalar un software de virtualización es de gran importancia porque ahorra tiempo y esfuerzo al administrador de la infraestructura. (1,2,1.5)/2.

✓ Las opciones de instalación de un software de virtualización de ordenadores permite instalar el software de virtualización en conjunto con el sistema operativo o posteriormente en el sistema operativo ya instalado. (1.5,3,2)/3.

✓ El despliegue de los valores que van tomando las variables para poder detectar posibles errores que pueden ocurrir y para ir verificando si los valores son los correctos. (1,2,1)/2.

- **Calificación**

$$Pc = \sum(w) = 2 + 3 + 2 = 7$$

$Pvmwa = \sum(x) = 1 + 1.5 + 1 = 3.5$ $(Cc - Vmwa) = (Pvmwa/Pc) * 100\%$

$= \left(\frac{3.5}{7}\right) * 100\% = 50\ \%$

$Pxen = \sum(y) = 2 + 3 + 2 = 7$ $(Cc - Xen) = (Pxen/Pc) * 100\% = \left(\frac{7}{7}\right) * 100\%$

$= 100\ \%$

$Pvirbox = \sum(z) = 1.5 + 2 + 1 = 4.5$ $(Cc - Virbox) = (Pvirbox/Pc) * 100\%$

$= \left(\frac{4.5}{7}\right) * 100 = 64.29\ \%$

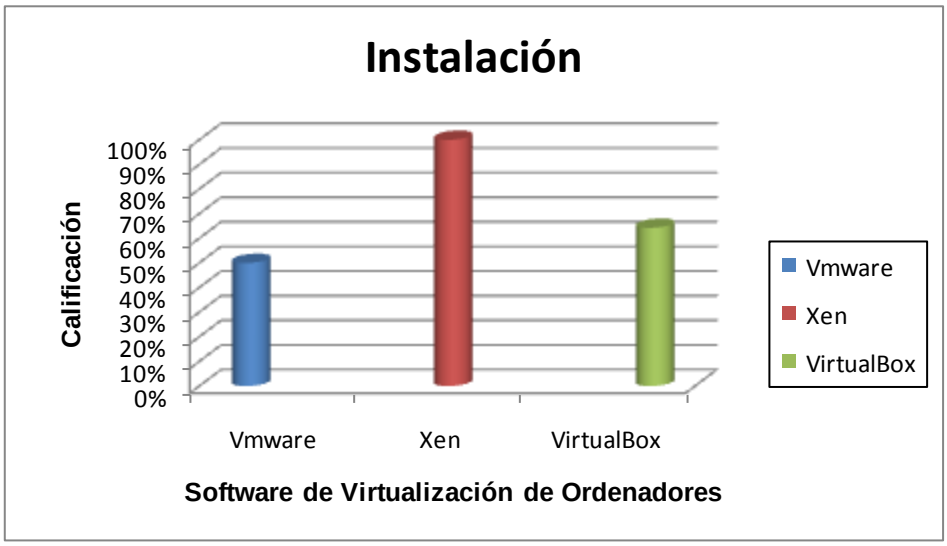

Gráfico III.19. Comparación de Porcentajes Parámetro 1

- **Escalabilidad**

La caraterística de la escalabilidad que proporciona un software de virtualización de ordenadores se puede analizar según las herramientas y utilidades que brinda cada herramienta, así como se muestra a continuación:

- **Determinación de Variables**

a. Soporte diferentes arquitecturas

b. Crecimiento de hardware

- **Valoraciones**

a. Variable Soporte diferentes arquitecturas

El soporte de diferentes arquitecturas es una gran ventaja a la hora de la implementar soluciones de infraestructura tecnológica (3 puntos).

b. Crecimiento de hardware

El crecimiento del hardware que brinda un software de virtualización de ordenadores ayuda a ampliar la infraestructura de servidores virtuales existentes, sin perder calidad en su servicio (3 puntos).

Tabla III.15. Escalabilidad

Variable	VMware	Xen	VirtualBox
Soporte diferentes arquitecturas	Soporte para varios arquitecturas 64/32 bits; Nativo sistemas operativos host 64 bits.	Soporte para muchas arquitecturas 64/32 bits	Soporte para pocas arquitecturas 64/32 bits; Inestable guest 64 bits sobre sistemas host 32 bits.
Crecimiento de hardware	Soporta crecimiento de recursos hardware.	Soporta crecimiento de recursos hardware.	Soporta crecimiento de recursos hardware.

- **Interpretación**

✓ El soporte de diferentes arquitecturas por un software de virtualización de ordenadores es indispensable para el crecimiento de la infraestructura satisfaciendo los requisitos de usuarios y administradores. (2,2.5,1.5)/3.

✓ El crecimiento de hardware permite aprovechar la capacidad de un software de virtualización asignando recursos hardware para aplicar en servidores virtuales nuevos. (3,3,3)/3.

- **Calificación**

$$Pc = \sum(w) = 3 + 3 = 6$$

$$Pvmwa = \sum(x) = 2 + 3 = 5 \qquad (Cc - Vmwa) = (Pvmwa/Pc) * 100\%$$

$$= \left(\frac{5}{6}\right) * 100\% = 83.33\ \%$$

$$Pxen = \sum(y) = 2.5 + 3 = 5.5 \qquad (Cc - Xen) = (Pxen/Pc) * 100\% = \left(\frac{5.5}{6}\right) * 100\%$$

$$= 91.67\ \%$$

$$Pvirbox = \sum(z) = 1.5 + 3 = 4.5 \qquad (Cc - Virbox) = (Pvirbox/Pc) * 100\%$$

$$= \left(\frac{4.5}{6}\right) * 100 = 75\ \%$$

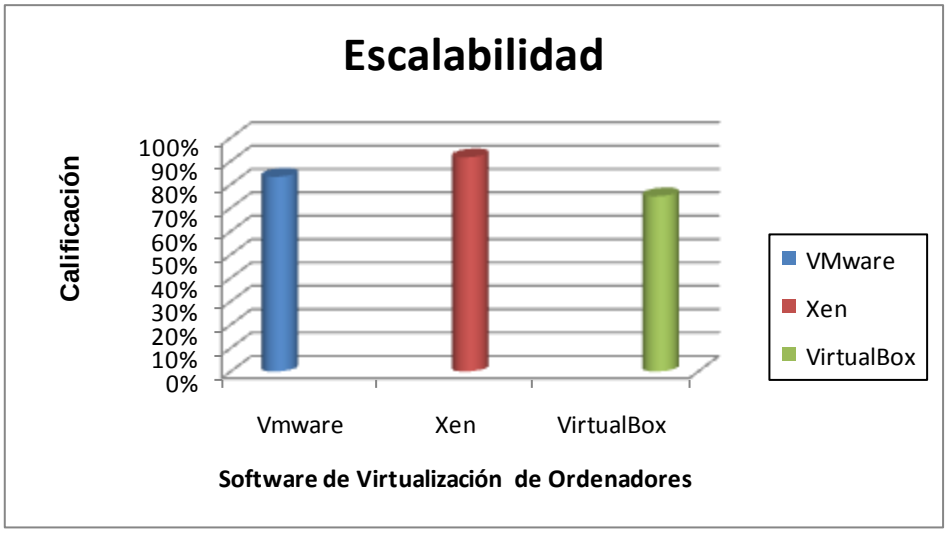

Gráfico III.20. Comparación de Porcentajes Parámetro 2

- **Alta disponibilidad**

La alta disponibilidad que debe ofrecer un software de virtualización de ordenadores se puede analizar tomando en cuenta las diferentes formas en las que se puede asegurar que el servicio funcione durante las veinticuatro horas, a continuación se indica como limitar las fallas en el servicio:

- **Determinación de Variables**

a. Prevención de errores
b. Tolerancia a errores
c. Eliminación de errores
d. Predicción de errores
e. Redundancia

- **Valoraciones**

a. Variable Prevención de errores

La prevención de errores que se puedan evitar anticipadamente en un software de virtualización de ordenadores es importante para el administrador de la infraestructura (3 puntos).

b. Variable Tolerancia a errores

La tolerancia a errores de un software de virtualización de ordenadores cuyo propósito es proporcionar un servicio de acuerdo con las especificaciones a pesar de los errores, presentando redundancias (2 puntos).

c. Variable Eliminación de errores

Al eliminar errores el software de virtualización de ordenadores será destinando a reducir la cantidad de errores por medio de acciones correctivas (2 puntos).

d. Variable Predicción de errores

La predicción de errores, anticipando errores y su posible impacto en el servicio ayuda al software de virtualización de software a estar más disponible (2 puntos).

e. Variable Redundancia

La redundancia en los recursos hardware y servidores virtuales es muy importante para la funcionalidad de la infraestrura (3 puntos).

Tabla III.16. Alta Disponibilidad

Variable	VMware	Xen	VirtualBox
Prevención de errores	Simple, se realizó en varios pasos.	Bastante simple, se realizó en pocos pasos.	Relativamente simple, se realización en varios pasos.
Tolerancia a errores	Soporta adición tolerancia a errores.	Soporta adición tolerancia a errores.	Soporta adición tolerancia a errores.
Eliminación de errores	Soporta la eliminación a errores.	Soporta la eliminación a errores.	Soporta la eliminación a errores.
Predicción de errores	Muestra una lista de todas las predicciones errores que se pueden generar.	Muestra una lista de todas las predicciones errores que se pueden generar.	Muestra una lista de todas las predicciones errores que se pueden generar.
Redundancia	Simple, se realizó en varios pasos.	Bastante simple, se realizó en pocos pasos, se comprobó la migración de servidores virtuales a físicos; se puede hacer en el modo offile o live.	Relativamente simple, se realizó en varios pasos.

- **Interpretación**

✓ La prevención de errores dentro de un software de virtualización de ordenadores consiste en evitar errores anticipándolos para garantizar el correcto funcionamiento de los sistemas y de que no pueda negar una operación realizada. (1.5,3,3)/3.

✓ La tolerancia a errores como valor de alta disponibilidad garantiza que los datos sean los que se supone que son. (2,2,2)/2.

✓ La eliminación de errores es garantizar el acceso a un servicio o a los recursos de un software de virtualización de ordenadores. (2,2,2)/2.

✓ La predicción de errores consiste en la obtención a priori de la garantía de funcionamiento del sistema de virtualización de ordenadores, para ello se realiza una evaluación del comportamiento del sistema ante la ocurrencia del fallo. (2,2,2)/2.

✓ La redundancia hace referencia al almacenamiento de backups de servidores virtuales en lugares de almacenamientos seguros como storage. (3,3,1.5)/3.

- **Calificación**

$Pc = \sum(w) = 3 + 2 + 2 + 2 + 3 = 12$

$Pvmwa = \sum(x) = 1{,}5 + 2 + 2 + 2 + 3 = 10.5$ $\quad (Cc - Vmwa) = (Pvmwa/Pc) * 100\%$

$= \left(\frac{10.5}{12}\right) * 100\%$

$= 87.5\ \%$

$Pxen = \sum(y) = 3 + 2 + 2 + 2 + 3 = 12$ $\quad (Cc - Xen) = (Pxen/Pc) * 100\%$

$= \left(\frac{12}{12}\right) * 100\% = 100\%$

$Pvirbox = \sum(z) = 3 + 2 + 2 + 2 + 1.5 = 10.5$ $\quad (Cc - Virbox) = (Pvirbox/Pc) * 100\%$

$= \left(\frac{10.5}{12}\right) * 100 = 87.5\ \%$

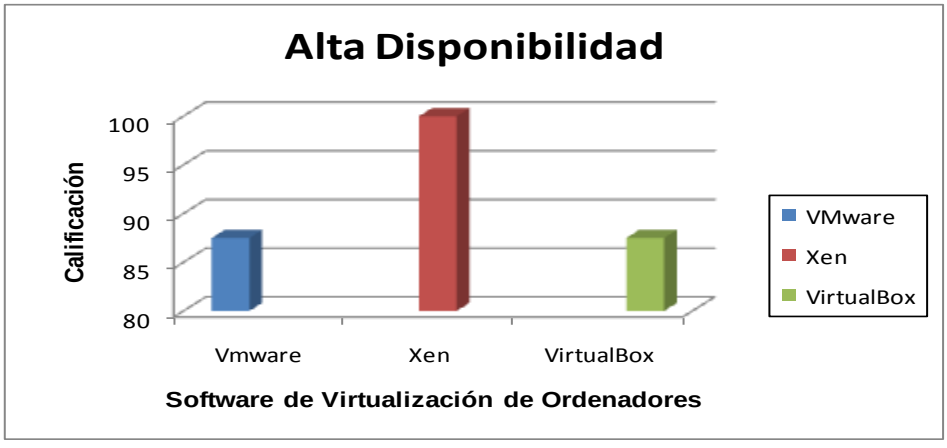

Gráfico III.21. Comparación de Porcentajes Parámetro 3

- **Flexibilidad**

La flexibilidad que debe ofrecer un software de virtualización de ordenadores se puede verificar con las herramientas utilitarias de administración de servidores virtuales, así como se indica a continuación:

- **Determinación de Variables**

a. Asignación de recursos hardware
b. Multiplicidad de vías para realizar una tarea
c. Variable Distribución

- **Valoraciones**

a. Variable Asignación de recursos hardware

La asignación de recursos hardware es importante para definir la variedad de posibilidades con las que el usuario y el sistema pueden intercambiar recursos (3 puntos).

b. Variable Multiplicidad de vías para realizar una tarea

La multiplicidad de vías para realizar una tarea, así como similitud con tareas anteriores y la optimización entre el usuario y el sistema (2 puntos).

c. Variable Distribución de trabajo

La distribución de trabajo es fundamental para planificar entre diferentes servidores virtuales (2 puntos).

Tabla III.17. Flexibilidad

Variable	VMware	Xen	VirtualBox
Asignación de recursos hardware	Soporte de asignación de recursos hardware en estado offline.	Presenta dos opciones para flexibilizar: mediante la interfaz gráfica o consola de línea de comandos en estado offline.	Soporte de asignación de recursos hardware en estado offline.
Multiplicidad de vías para realizar una tarea	Existe un número muy limitado de multiplicidad de vías para realizar una tarea.	Existe una considerable multiplicidad de vías para realizar una tarea.	Existe un número muy limitado de multiplicidad de vías para realizar una tarea.
Distribución de trabajo	La distribución de trabajo, es uniforme dentro de la infraestructua virtual.	La distribución de trabajo, es uniforme dentro de la infraestructua virtual.	La distribución de trabajo, es uniforme dentro de la infraestructua virtual.

- **Interpretación**

✓ La asignación de recursos hardware para la administración de servidores virtuales es bastante importante y rápido para gestionar y administrar eficientemente los recursos hardware, en el menor tiempo posible, permitiendo que se ejecuten. (2,3,2)/3.

✓ La multiplicidad de vías para realizar una tarea es importante para que varios procesos puedan ser ejecutados al mismo tiempo compartiendo uno o más recursos hardware. (0.5,2,0.5)/2.

✓ La distribución de trabajo que proprociona un software de virtulización de ordenados sirve para expandir su funcionalidad y propocionar más y mejor software para los administradores. (2,2,2)/.2

- **Calificación**

$Pc = \sum(w) = 3 + 2 + 2 = 7$

$Pvmwa = \sum(x) = 2 + 0.5 + 2 = 4.5$ $\quad (Cc - Vmwa) = (Pvmwa/Pc) * 100\%$

$= \left(\frac{4.5}{7}\right) * 100\% = 64.29\ \%$

$Pxen = \sum(y) = 3 + 2 + 2 = 7$ $\quad (Cc - Xen) = (Pxen/Pc) * 100\% = \left(\frac{7}{7}\right) * 100\%$

$= 100\%$

$Pvirbox = \sum(z) = 2 + 0.5 + 2 = 4.5$ $\quad (Cc - Virbox) = (Pvirbox/Pc) * 100\%$

$= \left(\frac{4.5}{7}\right) * 100 = 64.29\ \%$

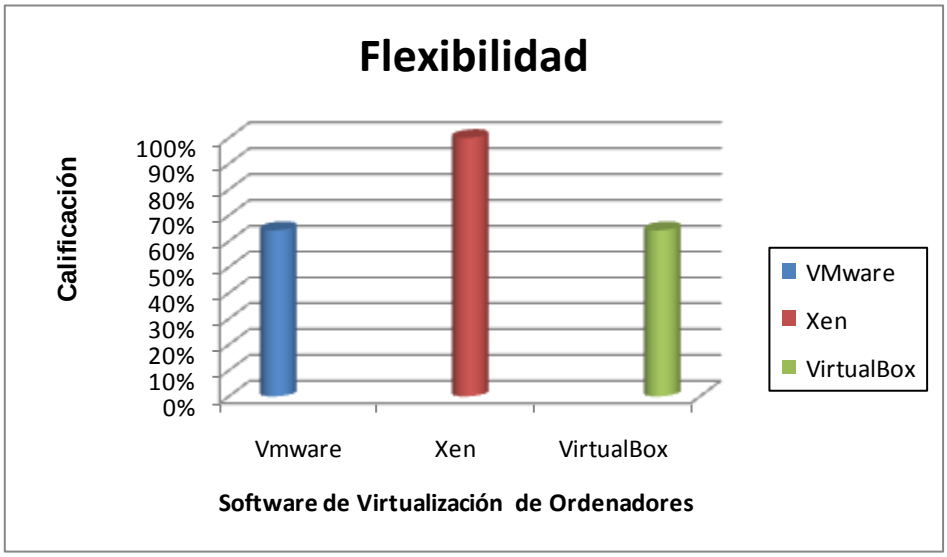

Gráfico III.22. Comparación de Porcentajes Parámetro 4

- **Usabilidad**

La facilidad que ofrece un software de virtualización de ordenadores para la gestión de los servidores virtuales, se puede analizar según las herramientas brindadas para realizar esta importante tarea, incluyendo los siguientes aspectos:

- **Determinación de Variables**

a. Crear, configurar y eliminar servidores virtuales.
b. Realizar operaciones de start, stop, reset, suspend and resume en los servidores virtuales.
c. Monitorear el funcionamiento de los servidores virtuales.
d. Compatibilidad de Navegadores Web.
e. Consola remota habilita.

- **Valoraciones**

a. Variable Crear, configurar y eliminar servidores virtuales.

El crear, configuración y eliminación de servidores virtuales es importante para la organización de la infraestuctura de servidores virtuales (3 puntos).

b. Variable Realizar operaciones de start, stop, reset, suspend y resume en los servidores virtuales.

Las operaciones de start, stop, reset, suspend y resume, permitidas realizar por una herramienta de administración es fundamental para tener un control de los diferentes procesos de ejecutados en los servidores (3 puntos).

c. Variable Monitorear el funcionamiento de los servidores virtuales.

El monitorear el funcionamiento de los servidores virtuales es importante porque ayuda a verificar a cada uno de los servidores virtuales (3 puntos).

Variable Compatibilidad de Navegadores Web.

La compatibilidad que exista de navegadores Web para una herramienta de administración es importante para darle más extensibilidad a la misma, valoración (2 puntos).

d. Variable Consola remota habilita.

Una consola remota habilita por cada uno de los servidores virtuales permite una gestión independiente de los mismos (3 puntos).

Tabla III.18. Usabilidad

Variable	VMware	Xen	VirtualBox
Crear, configurar y eliminar servidores virtuales.	Relativamente simple, se realizó en varios pasos.	Presenta dos opciones para crear servidores virtuales: fullvirtualización y paravirtualización, mediante interfaz gráfica o consola de línea de comandos. Bastante simple, se realizó en pocos pasos.	Relativamente simple, se realizó en varios pasos.
Realizar operaciones de start, stop, reset, suspend and resume en los servidores virtuales.	Bastante simple, se realizó en pocos pasos.	Bastante simple, se realizó en pocos pasos.	Bastante simple, se realizó en pocos pasos.
Monitorear el funcionamiento de los servidores virtuales.	Monitorea mediante la herramienta web Vmware Infrastructure.	Monitorea mediante dos formas: interfaz gráfica de usuario y consola de línea de comandos; virtual machine manager y xm top respectivamente. Además de la aplicación web SACV con Xymon.	Monitorea mediante la herramienta Sun VirtualBox de escritorio no es Web.
Compatibilidad de Navegadores Web.	Presenta mucha compatibilidad con nagevadores Web.	Presenta mucha compatibilidad con nagevadores Web.	No presenta herramienta ninguna herrramienta Web.
Consola remota habilita.	Soporta en la herramienta web Vmware Infrastructure.	Soporta en la herramienta virtual machine manager y mediante comados xm console NameVirtualMachine	Soporta en la herramienta Sun VirtualBox.

- **Interpretación**

✓ El software de virtualización de ordenadores debe permitir la creación, configuración y eliminación de servidores virtuales de forma sencilla para una administración eficiente. (2,3,2)/3.

✓ El realizar operaciones de start, stop, reset, suspend y resume en los servidores virtuales son tareas básicas e indispensables para la administración de una infraestructura virtual.(3,3,3)/3.

✓ El monitorear el funcionamiento de los servidores virtuales nos permite conocer el estatus y performance de la infraestructura virtual. (2.5,3,2.5)/3.

✓ La compatibilidad de navegadores web debe permitir la conexión desde diferentes tipos de broweser para la administración remota. (2,2,0)/2.

✓ La consola remota habilita permite la conexión a los servidores virtuales (guest) desde el servidor hosts para la realización de tareas del troubleshooting. (3,3,3)/3.

- **Calificación**

$$Pc = \sum(w) = 3 + 3 + 3 + 2 + 3 = 14$$

$$Pvmwa = \sum(x) = 2 + 3 + 2.5 + 2 + 3 = 12.5 \qquad (Cc - Vmwa) = (Pvmwa/Pc) * 100\%$$

$$= \left(\frac{12.5}{14}\right) * 100\%$$

$$= 89.29\ \%$$

$$Pxen = \sum(y) = 3 + 3 + \ 3 \ + \ \ 2 + 3 = 14 \qquad (Cc - Xen) = \ (Pxen/Pc) * 100\%$$

$$= \left(\frac{14}{14}\right) * 100\%$$

$$= \ 100\%$$

$$Pvirbox = \sum(z) = 2 + \ 3 + 2.5 + 0 + 3 = 10.5 \qquad (Cc - Virbox) = \ (Pvirbox/Pc) * 100\%$$

$$= \left(\frac{10.5}{14}\right) * 100 = 75 \ \%$$

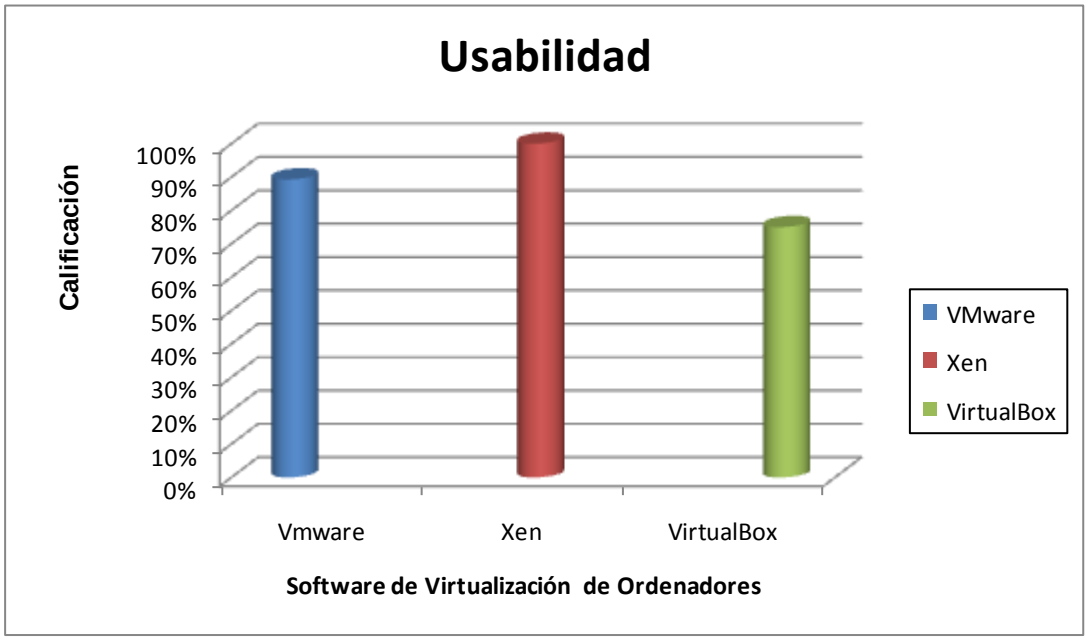

Gráfico III.23. Comparación de Porcentajes Parámetro 5

- **Estabilidad**

La estabilidad que proporciona un software de virtualización de ordenadores se puede validar a partir de las distintas fortalezas que brinda el software para la realización de este trabajo y de la forma en la que se puede obtener un mayor provecho de las mismas, como se muestra a continuación:

- **Determinación de Variables**

a. Procesos en cola de ejecución
b. Cantidad de Tareas Ejecutándose
c. Tareas en Espera
d. Consumo de CPU
e. Utilización de Memoria
f. Utilización de Swap

- **Valoraciones**

a. Variable Procesos en cola de ejecución

Los procesos en cola de ejecución son aquelloa que están en espera de un slot de tiempo por parte del procesador, un gran número de estos procesos encolados significa que no están siendo despachados (3 puntos).

b. Variable Cantidad de Tareas Ejecutándose.

La cantidad de tareas ejecutándose de acuerdo a la carga de trabajo empleado en el software de virtualización de ordenadores (3 puntos).

c. **Variable Tareas en estado sleeping.**

Las tareas en sleeping que presentan un software de virtualización de ordenadores es importante para saber cuántos procesos están esperando algún recurso para su ejecución (2 puntos).

d. Variable Consumo de CPU.

El consumo de CPU es importante evaluarlo para ver su utilización con sobre carga de trabajo (3 puntos).

e. Variable Utilización de Memoria.

La utilización de memoria en cada uno de los software de virtualización de ordenadores es importante para conocer la disponibilidad de memoria (3 puntos).

f. Variable Utilización de Swap.

La utilización de Swap es fundamental para conocer si los servidores virtuales han llegado al consumo máximo de su memoria RAM (2 puntos).

Tabla III.19. Estabilidad

Variable	VMware	Xen	VirtualBox
Proceso de encolamiento	Existe demasiado encolamiento de procesos, no hay desplacho rápido, hay saturación y colapsa.	Existe poco encolamiento, los procesos son despachados rápidamente.	Existe poco encolamiento de procesos, los procesos son despachados lentamente y a momentos no responde.
Cantidad de Tareas Ejecutándose	Alcanza un número máximo de 25 procesos en el tiempo de ejecución y colapsa inmedianamente.	El número de procesos que llega alcanzar es superior a 25 en el tiempo de ejecución.	El número de procesos que llega alcanzar es 25 en el tiempo de ejecución.
Tareas en Espera	Existen demasiados procesos en espera, por falta de capacidad del servidor virtual.	Existen varios procesos en sleeping pero el servidor siempre responde, en el top siempre generando datos.	Existen demasiados procesos en espera, por lentitud en el despacho.
Consumo de CPU	Bastante alto el consumo del recurso CPU y no logra finalizar la ejecución por falta de capacidad del servidor virtual.	Bastante alto el consumo del recurso CPU, se mantiene en ejecución.	Bastante alto el consumo del recurso CPU, se mantiene en ejecución y a momentos no responde.
Utilización de Memoria	Alcanza el máximo del total de la memoria y no logra finalizar la ejecución por falta de capacidad del servidor virtual.	Alcanza el máximo del total de la memoria, se mantiene en ejecución.	Alcanza el máximo del total de la memoria, se mantiene en ejecución y a momentos no responde.
Utilización de Swap	Requiere utilizar swap, por saturación del máximo del total de memoria asignada, no logra finalizar la ejecución por falta de capacidad del servidor virtual.	Requiere utilizar swap, se mantiene en ejecución.	Requiere utilizar swap, se mantiene en ejecución y a momentos no responde.

- **Interpretación**

✓ Los procesos en cola de ejecución nos indican la posible saturación de un servidor virtual al no ser despachados los procesos. (2,3,2.5)/3.

✓ La cantidad de tareas ejecutándose nos permite conocer la cantidad de carga de trabajo que un servidor virtual puede ejecutar manteniendo su estabilidad.(1,3,2)/3.

✓ Las tareas en estado sleeping nos indican la cantidad de trabajo que no pudo ser despachado por el servidor virtual por falta de recursos del sistema. (0.5,2,1.5)/2.

- ✓ El consumo de CPU nos indica la utilización de los procesadores lo cual ayuda a conocer la estabilización del sistema en caso de saturación del CPU. (1.5,2.5,2)/3.

- ✓ La utilización de memoria nos indica la disponibilidad de memoria existente lo cual ayuda a conocer la estabilización del sistema en caso de saturación de memoria. (1.5,2.5,2)/3.

- ✓ La utilización de swap nos indica que los servidores virtuales han utilizado el máximo de memoria Ram disponible y el uso de swap indica la degradación del sistema. (0.5,1.5,1)/2.

- **Calificación**

$Pc = \sum(w) = 3 + 3 + 2 + 3 + 3 + 2 = 16$

$Pvmwa = \sum(x) = 2 + 1 + 0.5 + 1{,}5 + 1{,}5 + 0{,}5 = 7$ $\quad (Cc - Vmwa) = (Pvmwa/Pc) * 100\% = \left(\frac{7}{16}\right) * 100\% = 43.75\ \%$

$Pxen = \sum(y) = 3 + 3 + 2 + 2{,}5 + 2{,}5 + 1{,}5 = 14.5$ $\quad (Cc - Xen) = (Pxen/Pc) * 100\% = \left(\frac{14.5}{16}\right) * 100\% = 90.63\%$

$Pvirbox = \sum(z) = 2{,}5 + 2 + 1.5 + 2 + 2 + 1 = 11$ $\quad (Cc - Virbox) = (Pvirbox/Pc) * 100\% = \left(\frac{11}{16}\right) * 100 = 68.75\ \%$

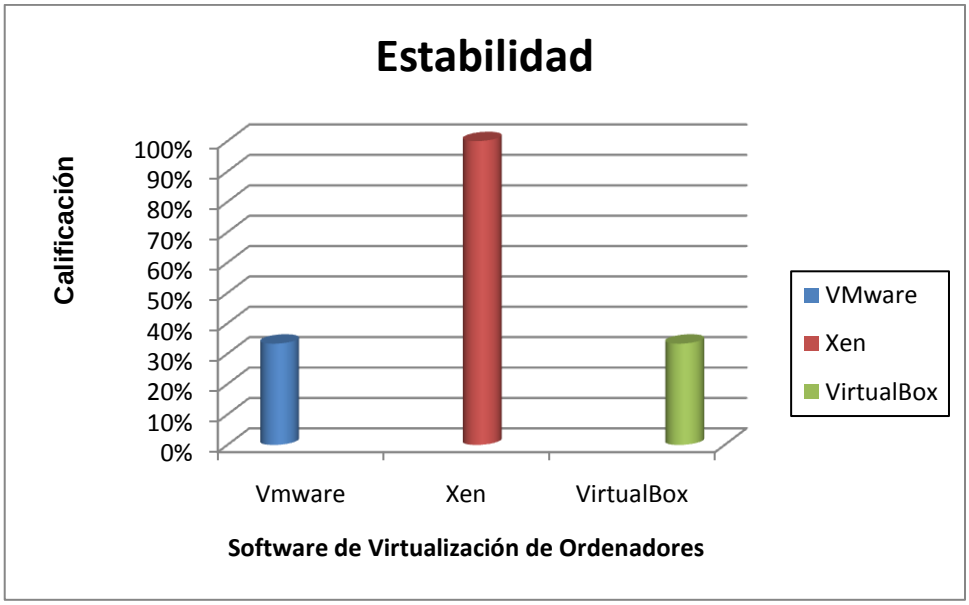

Gráfico III.24. Comparación de Porcentajes Parámetro 6

➢ 3.6.7. Rendimiento

El rendimiento que se puede analizar en un software de virtualización de ordenadores es evaluado a partir de los resultados obtenidos en las herramientas de benchmark, como se muestra a continuación:

- **Determinación de Variables**

  a. Puntaje unixbench

  b. Tráfico entre servidores host y guest

- **Valoraciones**

a. Variable Puntaje unixbench

La ejecución de Unixbench en un servidor virtual nos permite conocer su rendimiento y comparar difefrentes plataformas (3 puntos).

b. Variable Tráfico entre servidores host y guest

El mejor rendimiento de la red en las diferentes plataformas es evaluado mediante el Throughput de la transmisión de datos en las pruebas de Netperf (3 puntos).

Tabla III.20. Rendimiento

Variable	VMware	Xen	VirtualBox
Puntaje unixbench	Alcanzo un puntaje de 56.8 puntos.	Alcanzo un puntaje de 155.8 puntos.	Alcanzo un puntaje de 60.2 puntos.
Tráfico entre servidores host y guest	Test: TCP STREAM 16.45 Mb/s UDP REQUEST/RESPONSE 1186.45 trans/seg UDP UNIDIRECTIONAL SEND 15.04 Mb/s TCP REQUEST/RESPONSE 1550.19 tran/seg	Test: TCP STREAM 159.86 Mb/s UDP REQUEST/RESPONSE 2957.01trans/seg UDP UNIDIRECTIONAL SEND 94.54Mb/s TCP REQUEST/RESPONSE 1993.60 tran/seg	Test: TCP STREAM 80.85 Mb/s UDP REQUEST/RESPONSE 693.24 trans/seg UDP UNIDIRECTIONAL SEND 93.97Mb/s TCP REQUEST/RESPONSE 662.31 tran/seg

- **Interpretación**

✓ El puntaje unixbench es un benchmark que nos permite conocer que software de virtualización de ordenadores tiene el mejor rendimiento al ejecutar un conjunto de pruebas estándar.(1,3,1)/3.

✓ El tráfico entre los servidores host y guest nos permite conocer el performance del subsistema de red virtualizado.(1,3,1)/3.

- **Calificación**

$Pc = \sum(w) = 3 + 3 = 6$

$Pvmwa = \sum(x) = 1 + 1 = 2$ $\quad (Cc - Vmwa) = (Pvmwa/Pc) * 100\%$

$= \left(\frac{2}{6}\right) * 100\% = 33.33\ \%$

$Pxen = \sum(y) = 3 + 3 = 6$ $\quad (Cc - Xen) = (Pxen/Pc) * 100\%$

$= \left(\frac{6}{6}\right) * 100\% = 100\%$

$Pvirbox = \sum(z) = 1 + 1 = 2$ $\quad (Cc - Virbox) = (Pvirbox/Pc) * 100\%$

$= \left(\frac{2}{6}\right) * 100 = 33.33\ \%$

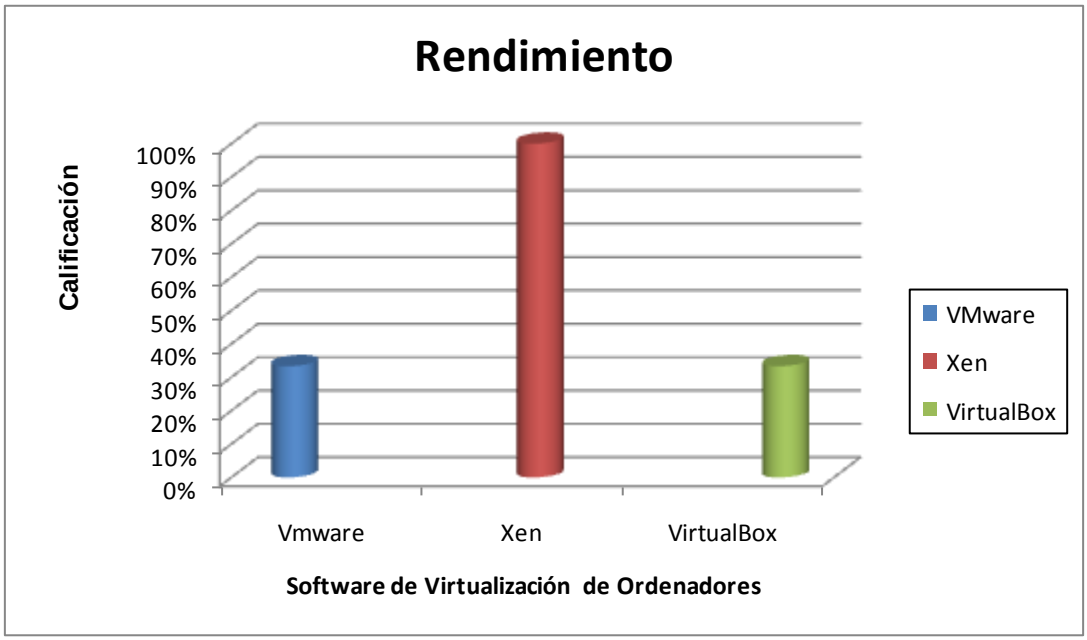

Gráfico III.25. Comparación de Porcentajes Parámetro 7

- **Velocidad**

La velocidad de respuesta de un software de virtualización de ordenadores depende de muchos factores como: CPU, memoria y velocidad de operaciones I/O, siendo éste último la variable analizada en este parámetro:

- **Determinación de Variables**

  a. Variable Velocidad del I/O

- **Valoraciones**

a. Variable Velocidad del I/O

La velocidad de respuesta de operaciones de I/O que proporciona un software de virtualización de ordenadores se puede analizar mediante el benchmark bonnie++ (3puntos).

Tabla III.21. Velocidad

Variable	VMware	Xen	VirtualBox
Velocidad del I/O	------Sequential Output------ -Per Chr- --Block-- -Rewrite- K/sec %CP K/sec %CP K/sec %CP 5506 72 10684 51 1072 4 ------Sequential Create------ -Create-- --Read--- -Delete-- /sec %CP /sec %CP /sec %CP 7831 86 +++++ +++ 12028 99 --Sequential Input- --Random- Per Chr- --Block-- --Seeks-- K/sec %CP K/sec %CP /sec %CP 711 4 913 1 75.2 5 --------Random Create-------- -Create-- --Read--- -Delete— /sec %CP /sec %CP /sec %CP 98 +++++ +++ 9495 79	------Sequential Output------ -Per Chr- --Block-- -Rewrite- K/sec %CP K/sec %CP K/sec %CP 17788 89 28400 14 10452 1 ------Sequential Create------ -Create-- --Read--- -Delete-- /sec %CP /sec %CP /sec %CP 10217 96 +++++ +++ 25846 98 --Sequential Input- --Random- -Per Chr- --Block-- --Seeks-- K/sec %CP K/sec %CP /sec %CP 15115 52 24507 1 141.0 0 --------Random Create-------- -Create-- --Read--- -Delete-- /sec %CP /sec %CP /sec %CP 10043 93 +++++ +++ 19461 77	------Sequential Output------ -Per Chr- --Block-- -Rewrite- K/sec %CP K/sec %CP K/sec %CP 7587 71 8229 15 7149 26 ------Sequential Create------ -Create-- --Read--- -Delete-- /sec %CP /sec %CP /sec %CP 70 +++++ +++ 17449 83 --Sequential Input- --Random- -Per Chr- --Block-- --Seeks-- K/sec %CP K/sec %CP /sec %CP 71 23552 58 168.7 12 --------Random Create-------- -Create-- --Read--- -Delete-- /sec %CP /sec %CP /sec %CP 10848 83 +++++ +++ 14823 85

- **Interpretación**

✓ La velocidad de I/O nos indica la respuesta del sistema a las operaciones de escritura, lectura o creación de archivos lo nos permite conocer que software de virtualización de ordenadores es más veloz. (1,3,1)/3

- **Calificación**

$Pc = \sum(w) = 3$

$Pvmwa = \sum(x) = 1$ $\quad (Cc - Vmwa) = (Pvmwa/Pc) * 100\% = \left(\frac{1}{3}\right) * 100\% = 33.33\ \%$

$Pxen = \sum(y) = 3$ $\quad (Cc - Xen) = (Pxen/Pc) * 100\% = \left(\frac{3}{3}\right) * 100\% = 100\%$

$Pvirbox = \sum(z) = 1$ $\quad (Cc - Virbox) = (Pvirbox/Pc) * 100\% = \left(\frac{1}{3}\right) * 100 = 33.33\ \%$

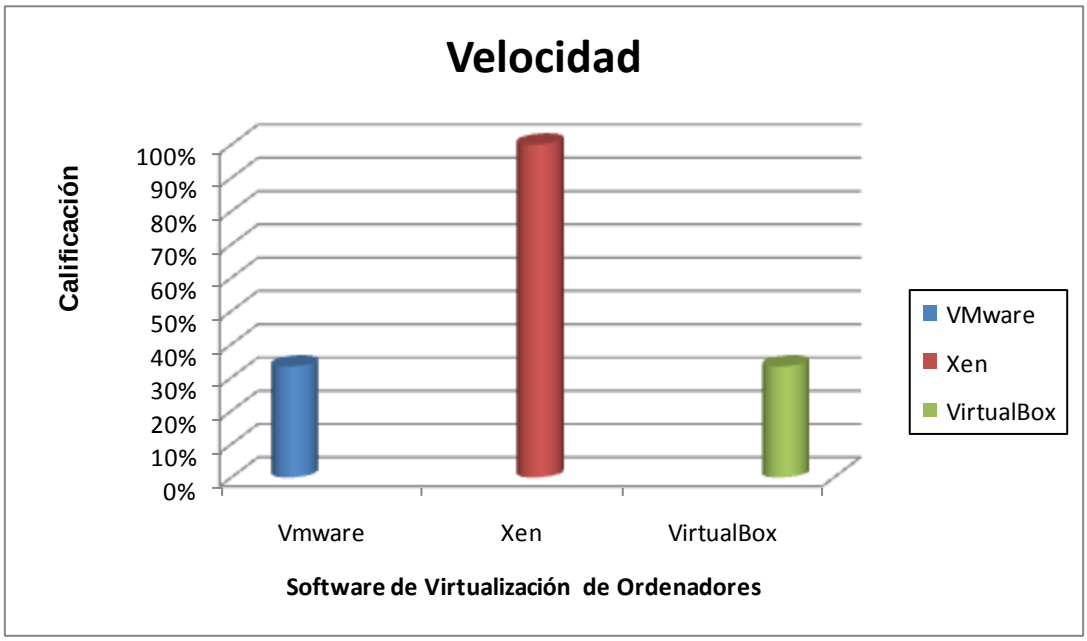

Gráfico III.26. Comparación de Porcentajes Parámetro 8

- **Extensibilidad**

La extensibilidad que ofrece un software de virtualización de ordenadores referente a la capacidad de un esquema para poder tolerar cambios en los requerimientos en una infraestructura, incluyendo los siguientes aspectos:

- **Determinación de Variables**

b. Soporte de Sistemas Operativos

c. Independencia de hardware

- **Valoraciones**

a. Variable Soporte de Sistemas Operativos

El soporte de sistemas operativos es importante para que un conjunto de programas informáticos permita la administración eficaz de los recursos de los servidores virtuales (2 puntos).

b. Variable Independencia de hardware

La independencia de hardware es elemental para establecer que la funcionalidad de cada uno de los servidores virtuales no se vea afectada entre ellos (2 puntos).

Tabla III.22. Extensibilidad

Variable	VMware	Xen	VirtualBox
Soporte de Sistemas Operativos	Soporte para pocos sistemas operativos guest.	Soporte para muchos sistemas operativos host y guest.	Soporte para varios sistemas operativos host.
Independencia de hardware	Facilidad en la independencia del hardware de cada servidor virtual.	Facilidad en la independencia del hardware de cada servidor virtual.	Facilidad en la independencia del hardware de cada servidor virtual.

- **Interpretación**

✓ El soporte de diferentes sistemas operativos permite virtualizar varios tipos de servidores llegando abarcar la totalidad de una infraestuctura virtual. (1.5,2,2)/2.

✓ La independencia de hardware, permite que los servidores virtuales se maneje como sistemas independientes, brindando seguridad y eficiencia en el uso de hardware.(2,2,2)/2.

- **Calificación**

$$Pc = \sum(w) = 2 + 2 = 4$$

$$Pvmwa = \sum(x) = 1.5 + 2 = 3.5 \qquad (Cc - Vmwa) = (Pvmwa/Pc) * 100\%$$

$$= \left(\frac{3.5}{4}\right) * 100\% = 87.5\ \%$$

$$Pxen = \sum(y) = 2 + 2 = 4 \qquad (Cc - Xen) = (Pxen/Pc) * 100\% = \left(\frac{4}{4}\right) * 100\%$$

$$= 100\%$$

$$Pvirbox = \sum(z) = 2 + 2 = 4 \qquad (Cc - Virbox) = (Pvirbox/Pc) * 100\%$$

$$= \left(\frac{4}{4}\right) * 100 = 100\ \%$$

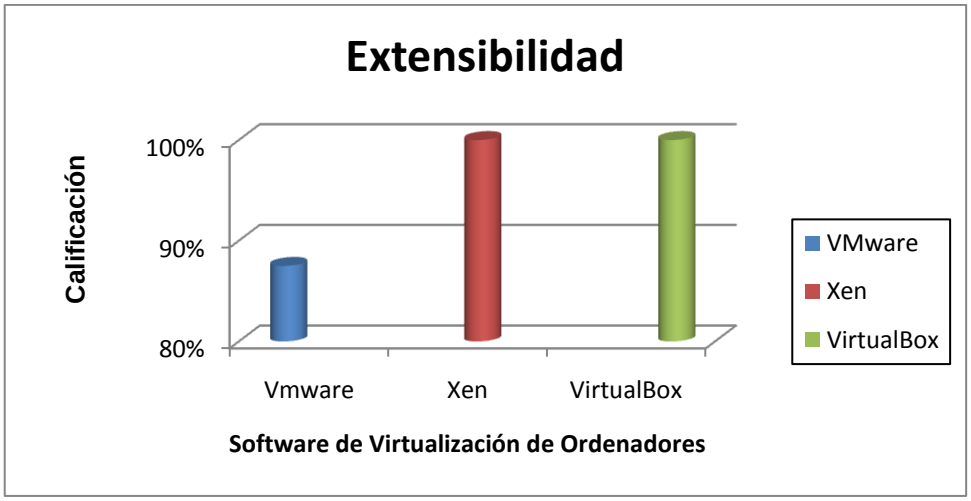

Gráfico III.27. Comparación de Porcentajes Parámetro 9

3.2.7. Puntajes Alcanzados

El puntaje final y el porcentaje que ha obtenido cada software de virtualización de ordenadores se obtiene de la siguiente manera:

Puntaje Total del Análisis: $(PT) = \sum(Pc)$

Puntaje Total de VMware: $(PTVmwa) = \sum(Pvmwa)$

Puntaje Total de Xen: $(PTXen) = \sum(Pxen)$

Puntaje Total de VirtualBox: $(PTVirbox) = \sum(Pvirbox)$

Porcentaje Total de Vmware: $(\%Vmwa) = ({}^{PTVmwa}/_{PT}) * 100\%$

Porcentaje Total de Xen: $(\%Xen) = ({}^{PTXen}/_{PT}) * 100\%$

Porcentaje Total de VirtualBox: $(\%Virbox) = ({}^{PTVibox}/_{PT}) * 100\%$

Tabla XXIII. Tabla General de Resultados

Parámetro	Pc	Variable	Vmware	Xen	VirtualBox
1	2	1.1	1	2	1,5
	3	1.2	1,5	3	2
	2	1.3	1	2	1
Total 1	**7**	**-**	**3,5**	**7**	**4,5**
2	3	2.1	2	2,5	1,5
	3	2.2	3	3	3
Total 2	**6**	**-**	**5**	**5,5**	**4,5**
3	3	3.1	1,5	3	3
	2	3.2	2	2	2
	2	3.3	2	2	2
	2	3.4	2	2	2
	3	3.5	3	3	1,5
Total 3	**12**	**-**	**10,5**	**12**	**10,5**
4	3	4.1	2	3	2
	2	4.2	0,5	2	0,5
	2	4.3.	2	2	2
Total 4	**7**	**-**	**4,5**	**7**	**4,5**
5	3	5.1	2	3	2
	3	5.2	3	3	3
	3	5.3	2,5	3	2,5
	2	5.4	2	2	0
	3	5.5	3	3	3
Total 5	**14**	**-**	**12,5**	**14**	**10,5**
6	3	6.1	2	3	2,5
	3	6.2	1	3	2
	2	6.3	0,5	2	1,5
	3	6.4	1,5	2,5	2
	3	6.5	1,5	2,5	2
	2	6.6	0,5	1,5	1
Total 6	**16**	**-**	**7**	**14,5**	**11**
7	3	7.1	1	3	1
	3	7.2	1	3	1
Total 7	**6**	**-**	**2**	**6**	**2**
8	3	8.1	1	3	1
Total 8	**3**	**-**	**1**	**3**	**1**
9	2	9.1	1,5	2	2
	2	9.2	2	2	2
Total 9	**4**	**-**	**3,5**	**4**	**4**

$(PT) = 7 + 6 + 12 + 32 + 14 + 16 + 6 + 3 + 4 = 75$

$(PTVmwa) = 3.5 + 5 + 10.5 + 4.5 + 12.5 + 7 + 2 + 1 + 3.5 = 49.5$

$(PTXen) = 7 + 5.5 + 12 + 7 + 14 + 14.5 + 6 + 3 + 4 = 73$

$(PTVirbox) = 4.5 + 4.5 + 10.5 + 4.5 + 10.5 + 11 + 2 + 1 + 4 = 52.5$

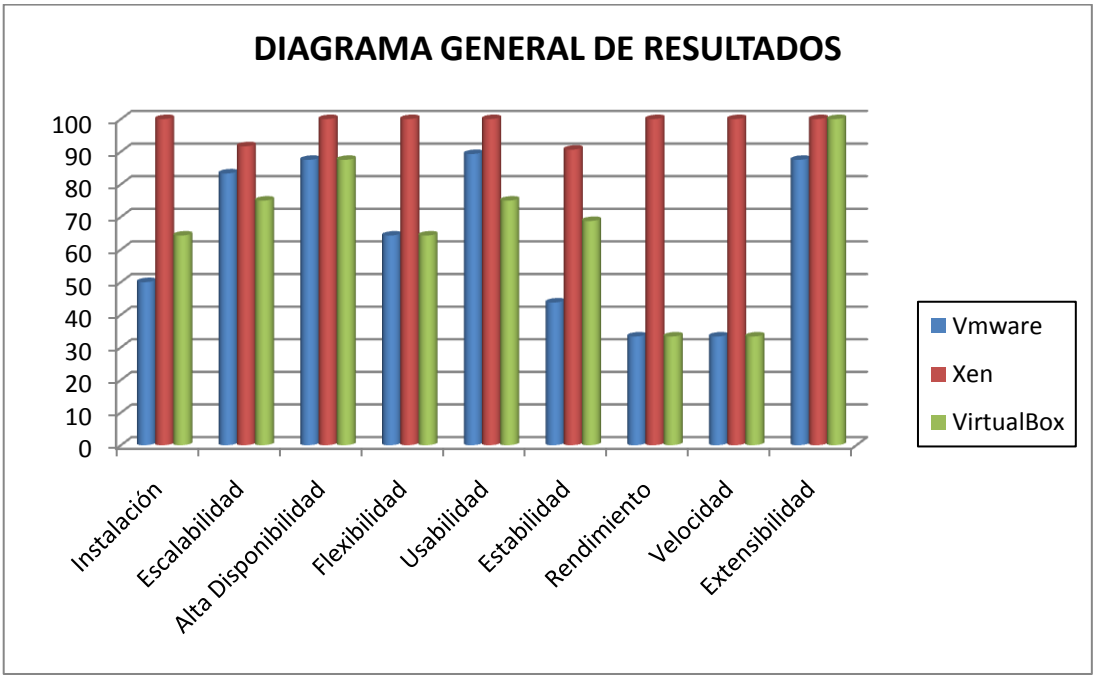

Gráfico III.28. Diagrama General de Resultados

$(\%\, Vmwa) = (49.5|75) * 100 = 66\,\%\ \ Equivalente\ a\ Regular$

$(\%\, Virbox) = (52.5|75) * 100 = 70\,\%\ \ Equivalente\ a\ Bueno$

$(\%\, Xen) = (73|75) * 100 = 97.33\,\%\ \ Equivalente\ a\ Excelente$

3.2.8. Interpretación

Como resultado del análisis y de acuerdo al puntaje obtenido para cada una de las variables se ha obtenido que el software de virtualización de ordenadores Xen ha

obtenido el puntaje más alto con un porcentaje del 97,33 % que es equivalente a Excelente.

3.2.9. Resultado del Análisis

- ✓ Al ser el Xen un software incluído en la distribución CentOs 5.0 o supeior facilita la instalación ya que las dependencias y prerrequisitos son validadas por el proceso de instalación.

- ✓ El Xen al ser instalado en conjunto con el sistema operativo ahorra tiempo y esfuerzo al administrador de la infraestructura.

- ✓ El Xen reconoce nuevo hardware instalado y mediante sus herramientas de administración gestiona un crecimiento escalable en los servidores.

- ✓ La infraestructura virtual de alta disponibilidad desarrollado en el Xen mediante opciones de brigde para tarjetas de red, uso de volúmenes lógicos con protección RAID, asignación de CPU independientes, clonación de servidores virtuales permite estar protegidos contra posibles de errores.

- ✓ Es importante destacar que el software de virtualización de ordenadores Xen aisla errores de hardware como fallas de discos, tarjetas, CPU e incluso de software al administrar independiente los servidores virtuales.

- ✓ La virtualización con Xen se basa en asignar recursos hardware tales como CPU, cores de CPU, memorias, particiciones de discos mediante su interfaz de administración.

- ✓ La creación de servidores virtuales permite asignar tareas específicas a cada uno de ellos, utilizando un sólo servidor físico.

- ✓ La interfaz gráfica de usuario, virtual machine manager, y la consola de línea de comandos permite una creación, configuración y eliminación de servidores virtuales de forma rápida, eficiente y segura.

- ✓ Mediante la aplicación SACV, podemos realizar tareas operativas en los servidores virtuales del Xen, mediante la utilización de un navegador Web.

- ✓ El componente Xymon de la aplicación SACV monitorea toda la infraestrura de servidores virtuales, utilizando comandos y facilidades que brinda el Xen.

- ✓ En la prueba de stress más exigente, el Xen mantiene la cola de procesos en ejecución más baja, de los tres software de virtualización de ordenadores, permitiéndole mantener la estabilidad de los servidores virtuales al poder ejecutar procesos en el sistema operativo bajo estas condiciones de stress.

- ✓ En promedio el Xen mantiene más procesos en estado de ejecución que el resto de software de virtualización de ordenadores, lo que le da mayor estabilidad al sistema en un estado de stress.

- ✓ En la pruebas de stress el Xen tuvo el menor número de tareas en estado sleeping y los servidores virtuales respondieron durante todo el lapso de tiempo de la ejecución de las mismas, por lo que es el sistema de virtualización de ordenadores que presenta la mayor estabilidad.

- ✓ El Xen obtuvo el puntaje más alto en las pruebas de unixbench con una diferencia notable a VMware y Virtualbox, ya que se utilizó el mismo hadware y características

idénticas en los servidores virtuales demostrándonos que el Xen tiene un rendimiento mucho mejor.

- ✓ El análisis de tráfico realizado por Netperf nos demuestra que el Xen tiene una velocidad de transferencia mucho mayor que VMware y Virtualbox, lo que lleva a concluir que el rendimiento general de este software de virtualización de ordenadores es el mejor.

- ✓ El Xen obtiene escrituras y lecturas secuenciales y randómicas mucho mayores ya que tiene un acceso directo a disco mucho más rápido a diferencia de VMware y VirtualBox que acceden al disco a través un directorio o filesystem del host.

- ✓ De acuerdo a la información disponible por el fabricante, bajo el esquema de full virtualización el Xen soporta una variable amplia de sistemas operativos.

- ✓ El Xen al igual que el VMware y VirtualBox dispone de indepencia de hardware entre sus servidores virtuales existentes para garantizar la extesibilidad de la infraestructura virtual.

3.2.10. Conclusión

Por todo lo expuesto anteriormente y de acuerdo a los puntajes alcanzados para cada uno de los parámetros de evaluación se puede concluir que el software de virtualización de ordenadores Xen es el que brinda las mejores prestaciones para virtualizar cualquier sistema operativo.

CAPÍTULO IV

DESARROLLO DE LA INFRAESTRUCTURA DE SERVIDORES VIRTUALES

4.1. INTRODUCCIÓN

Para el desarrollo de la infraestructura virtual en la EIS se aplicará la teoría de software de virtualización de ordenadores virtuales del Xen.

4.2. GENERALIDADES

4.2.1. Instalación del software de virtualización de ordenadores

4.2.1.1. Instalación Xen

CentOS desde la versión 5 soporta Xen, ofrece Xen como uno de sus paquetes. Al tener 7 años de soporte, tendremos la certeza de tener soportado a Xen durante 7 años.

CentOS ofrece alojar máquinas de forma paravirtualizada (más rápidas pero más complejas) y de forma de full virtualization (requiere de procesadores que soporten vmx (intel) o svm (amd).

El proceso de instalación es bien simple y considerando que tenemos el DVD de CentOS 5 o los 6 CDs.

Esta instalación seguro que nos formateará el disco duro. Es responsabilidad nuestra respaldar la información.

Insertamos el primer CD o el DVD y arrancamos la máquina. Seguimos el proceso de instalación normalmente, como si estuvieramos instalando cualquier CentOS o Fedora.

Al momento de particionar, lo haremos de la siguiente forma:

1- **/boot** - 100 megas, forzada a ser primaria.

2- El resto del disco de tipo **LVM** (en vez de ext3, escoger LVM y aplicale al resto del disco).

3- Pulsar en el botón de LVM y en el VolGroup (arriba arriba) poner: **dsk** (dice: VolGroup00, como nombre al volumen lógico, pondremos dsk que lo usaremos así.

4- Abajo de VolGroup, dará opción para crear nuevos volúmenes, es en un recuadrito que a la derecha tiene varios botones (new, delete, etc).

5- Crear un Logical Volume (LV, VolName) llamado **swap** de 1024M que sea de tipo swap. Y

6- en el mismo lugar: otro llamado **root** que sea de tipo ext3, montarlo en / y que tenga digamos 5000MB

7- Pulsar ok, salir de la configuración de LVM y regresar al fdisk.

8- Seguir instalando.

Nos sobrará bastante disco (el resto). La idea es que en ese resto de espacio de LVM crearemos las máquinas virtuales (llamadas domU).

A la hora de escoger los paquetes, miraremos bien abajo, dirá: Virtualization. Es este el único que nos interesa que se instale, los demás podríamos quitarles, al menos los de ambiente gráfico, open office, etc.

Se sugiere no instalar muchos paquetes. En el sistema éste, que se llamará hospedero (dom0) no hacen falta muchos paquetes, sólo los de virtualización, es más, no instalo siquiera el ambiente gráfico en éste hospedero (dom0).

Escogemos esa opción, y seguimos instalando normalito, como si nada pasara. Al finalizar reiniciamos la máquina y arrancará CentOS 5.

Es OBLIGATORIO actualizar el sistema, el Xen que vino en el CentOS 5.0 tenía algunos problemas y es mejor actualizarle. Utilizaremos el siguiente comando:

```
yum update
```

La actualización nos llevará un instante en dependencia de la cantidad de paquetes que se haya instalado y la velocidad de nuestra conexión.

Al finalizar, reiniciamos para que se levante el nuevo kernel y ahi procedemos con el resto.

4.2.1.2. Pasos iniciales

Una vez instalado el Xen parece un linux normal. Se empieza a trabajar en algo llamado el hipervisor, el dom0, el hospedero. A la final es un kernel de linux modificado para poder ejecutar maquinas virtuales.

Se sugiere que en el dom0, que es ese linux donde ahora estamos, no instalemos nada raro, ningún paquete adicional, sólo SSH (se sugiere quitar el SSH). El xen tiene básicamente una herramienta desde la que se puede hacer todo, se llama xm. A continuación se listan algunos comandos útiles para trabajar con Xen:

- xm list (lista las máquinas virtuales)
- xm shutdown ID (apaga la máquina virtual que tenga ese ID)
- xm destroy ID (Es como quitarle la corriente a ese ID)
- xm create ID (arranca, como ponerle la corriente a ese ID, tiene que haber sido previamente instalado)
- xm console ID (permite conectarnos via consola a la máquina virtual ID. Es como estar delante de la pantalla, en tty1)

4.2.1.3 Uso de LVM

Hay que tener conocimientos previos sobre el uso de LVM (Logical Volume Manager), para crear las máquinas virtuales.

4.2.1.3.1 Uso de LVM2 en Linux CentOS

Los LVM nos permiten unir varios discos o particiones (conocidos como Physical Volumes, PV) formando un gran pool de almacenamiento de datos conocido como Volume Group (VG) que puede ser posteriormente subdividido en "particiones".

En caso de necesidad de más espacio podemos redimensionar estas particiones, conocidas como Logical Volumes (LV).

En resumen:

VG: Volume Group: Es la unión de todos los discos o particiones a utilizar. Del VG se toma segmentos de espacio de disco para crear las particiones.
PV: Pysical Volume: Es en sí todos y cada uno de los dispositivos de almacenamiento que se vaya a utilizar. Discos, particiones, etc., (conocido como discos duros).
LV: Son las particiones. Segmentos del VG que se asignará.

Es posible cualquier combinación de volúmenes lógicos igual o menor que la capacidad total. Las posibilidades están limitadas solamente a nuestras necesidades.

La característica que la mayoría de los administradores de sistemas aprecian más sobre LVM es su habilidad para dirigir fácilmente el almacenamiento donde se necesite. En una

configuración de sistemas no LVM, el quedarse sin espacio significa - en el mejor de los casos - mover archivos desde un dispositivo lleno a uno con espacio disponible. A menudo esto significa la reconfiguración de los dispositivos de almacenamiento masivo; una tarea que toma tiempo adicional.
Sin embargo, LVM hace posible incrementar fácilmente el tamaño de un volumen lógico.

Estos 3 PV los unimos y formamos un group de volúmenes (VG) que llamaremos: disk
Ahora, tenemos a nuestra disposición 120GB en el VG y podemos hacer uso de este espacio como nos convenga. Particionemos nuestro VG con varios Logical Volumes (LV) que vienen siendo las particiones:

```
/dev/disk/home con 45GB
/dev/disk/var con 40GB
/dev/disk/root con 10GB
/dev/disk/usr con 15GB
/dev/disk/swap con 1GB
```

En total estaremos consumiendo 111GB, nos quedarían unos 9GB disponibles.

Si te fijas, no importa el tamaño de los PV que es de 40GB cada uno, sino el conjunto.

Una vez creados los LV nos referimos como /dev/nombredelVG/nombredelLV, en este caso sería /dev/disk/home por ejemplo.

Ahora podemos usarlos normalmente en nuestro fstab, editamos el fstab y en vez de poner /dev/hda1 por ejemplo, ponemos /dev/disk/var o /dev/disk/root (como no puedo usar el / en el nombre ponenmos root). Y empezamos a utilizarlos normalmente.

- **Comandos LVM2**

- **Creando PV:**

Supongamos tenemos 3 discos disponibles, hda, hdb y hdc, para agregarlos a nuestro LVM lo que hacemos es particionarlos y a las particiones las ponemos del tipo: 8e Linux LVM

Esto lo haremos con el comando fdisk, escogiendo con "t" la partición que queremos crear como LVM normalmente hago sólo una partición por disco, y le cambio el tipo a 8e.

Una vez cambiados los tipos de los discos, procedemos a agregarles la información de LVM a cada uno, con el comando **pvcreate**:

```
pvcreate /dev/hda1
pvcreate /dev/hdb1
pvcreate /dev/hdc1
```

Con estos simples pasos les hemos agregado dentro del sector de configuración los datos necesarios para que ellos actúen como un Physical Volume.

Ahora tenemos que agregar estos 3 PV a un Grupo de Volumenes (VG):

```
vgcreate disk /dev/hda1 /dev/hdb1 /dev/hdc1
vgchance -a y disk
```

Con esto hemos agrupado dentro de un grupo de volumenes (la información realmente se guarda en cada uno de los PV en su sector de configuración) a los tres PV y después hemos activado el VG.

El VG que tenemos contiene espacio para 120GB, pero si nos hace falta más sencillamente agrego los dispositivos que necesite o tenga.

- **Creando LV (particiones)**

Creemos los LV indicados anteriormente:

```
lvcreate -L45G -nhome disk
lvcreate -L40G -nvar disk
lvcreate -L10G -nroot disk
lvcreate -L15G -nusr disk
lvcreate -L1G -nswap disk
```

la **-L** sirve para indicar el tamaño del LV, se puede medir en megas (M), gigas (G) pues sólo he usado G y M para los discos.

la **-n** sirve para indicar cómo se llamará el LV

La última entrada es el nombre del VG del que tomaremos la información.

Asi que ahora tendremos el VG llamado disk conteniendo 5 LV: home, var, root, usr y swap.

Una vez hecho esto podemos formatear todos ellos:

```
mkfs.ext3 /dev/disk/root
mkswap /dev/disk/swap
mkfs.ext3 /dev/disk/home
mkfs.ext3 /dev/disk/var
mkfs.ext3 /dev/disk/usr
```

4.2.1.3.2 Usando Comandos adicionales de LVM2

- **Extender un VG:**

Para eso le damos de alta a un nuevo PV y lo agregamos al VG, en éste ejemplo agregaremos /dev/hdd1 al VG del howto inicial:

```
pvcreate /dev/hdd1

vgextend disk /dev/hdd1
```

- **Visualizar la información:**

Estos tres comandos nos podrán dar información sobre los PV, VG y LV del sistema, como por ejemplo: espacio utilizado, disponible, etc.

pvdisplay

vgdisplay

lvdisplay

- **Eliminar un LV**

Supongamos que no queremos más el LV llamado home, por alguna razón. Y que deseamos eliminarlo para disponer de ese espacio en otras cosas. Lo hacemos con:

```
umount /dev/disk/home

lvremove /dev/disk/home
```

- **Extender un LV**

Supongamos que por el contrario, sí queremos home, pero queremos agregarle 5GB extras, para eso:

```
umount /dev/disk/home
lvextend -L+5G /dev/disk/home
e2fsck -f /dev/disk/home
resize2fs /dev/disk/home
```

Con esto:

1- Desmontamos home, pues no puede estar activo para el cambio.

2- Extendemos con lvextend, el **-L** ahora va con +5G, esto es, sumándole 5GB al valor que tenía originalmente.

3- Chequeamos el **FS** (file system) Linux second extended file system (formato ext3)

4- hacemos un resize del FS, con el comando **resize2fs**

- **Reducir un LV**

Veamos el caso contrario, pensemos que queremos reducir home, le quitaremos 5GB, el proceso es parecido al anterior:

```
umount /dev/disk/home
e2fschk -f /dev/disk/home
resize2fs /dev/disk/home 35G
lvreduce -L-5G /dev/disk/home
```

Con esto:

1- Desmontamos home, no debe estar montado para la operación

2- Revisamos el **FS** ext3

3- Reducimos el tamaño del FS ext3 (aquí sí hay que saber a cuánto vamos a reducir)

4- Reducimos el tamaño del LV home, usamos **-L-5G** aunque también podíamos haber usado -L35G para indicarle el valor hacia donde lo reduciríamos.

- **Remover un PV**

Para remover un PV, puede ser porque hemos agregado un nuevo PV que es muy grande y ya este PV anterior nos queda pequeño en capacidad el almacenamiento, por lo que queremos eliminar el disco.

Supongamos que queremos remover hdc1 del ejemplo inicial.

```
pvmove /dev/hdc1
```

En este proceso tomará un tiempo, pvmove saca la información que tenga guardada y la almacena en los otros PV, esto sin alterar el esquema de los LV.

La suma de capacidad disponible en los otros PV debe ser mayor o igual a la cantidad de información que esté moviendo.

Al finalizar de mover el PV, lo podemos eliminar del VG con:

```
vgreduce disk /dev/hdc1
```

Listo, con este paso deja de constar dentro del VG y lo podemos sacar físicamente de la PC donde lo tenemos.

- **Otras sugerencias:**

- No se sugiere usar LVM para la partición /, esta partición prácticamente no crece durante el tiempo de vida del sistema (claro, si tenemos /var y /usr y /home en particiones diferentes).
- No se sugiere usar LVM para /boot, los gestores de arranque normalmente no entienden de lvm y nos fallará el arranque. Después de todo /boot tampoco crece tanto durante el tiempo de vida de la máquina.
- Este proceso se ve grandemente agilitado cuando estamos recién instalando un nuevo servidor. Pues en el instalador de CentOS/RHEL aparece una opción para LVM que nos permite fácilmente crear los LVM que necesitemos.
- Se utiliza en servidores de producción y ha sido de utilidad para poder quitar y/o poner espacio a particiones, ayuda y nos evita la reinstalación o el pasar trabajo insertando nuevos discos.

- Se Sugiere usarlo encima de un RAID, pues si un PV se daña, puede dañarse todo el sistema.

4.2.1.4. Tipos de virtualización

Existen dos tipos de virtualización en Xen: Paravirtualización y full virtualización.

- **Paravirtualización:** Es mucho más rápida pues se integran llamadas desde el kernel al hipervisor sin requerir de la ayuda del procesador. Sin embargo se necesita tener el mismo kernel en el dom0 y en los domU, esto es: sólo podríamos instalar linux con el mismo kernel.

No podríamos bootear un servidor normal, sino que tendríamos que hacer una instalación en base a yum o debootstrap desde un chroot y después bootear.

En todo caso, la paravirtualización no ofrece velocidades cercanas a la velocidad nativa del procesador (con sólo un 8% de degradación en el peor de los escenarios).

- **Full Virtualización:** El procesador se encargará de emular las llamadas, de ésta forma se puede arrancar desde un CD, el sistema trabajará full nativo es decir, el sistema se pensará que está él solito en el procesador. No se requieren modificaciones al kernel, por lo tanto se pueden instalar varios sistemas operativos como domU sinque ellos se enteren en principio.

Mediciones previas afirman que existe una gran degradación del desempeño de la full virtualización en lo que respecta a la red (casi un 50% menos de aprovechamiento de la red). Estudios analizados indican que es menos eficiente que la paravirtualización, no es

algo que pueda o no mejorar el xen, sino que la full virtualización depende del procesador (probado en intel con vmx).

Para determinar si podemos correr en *full virtualización*, realizamos lo siguiente:

```
cat /proc/cpuinfo | grep vmx
```

```
[root@xen ~]# cat /proc/cpuinfo |grep vmx
flags		: fpu tsc msr pae mce cx8 apic mtrr mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe nx lm constant_tsc pni monitor ds_cpl vmx est tm2 cx16 xtpr lahf_lm
flags		: fpu tsc msr pae mce cx8 apic mtrr mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe nx lm constant_tsc up pni monitor ds_cpl vmx est tm2 cx16 xtpr lahf_lm
```

Como podemos visualizar nos aparece vmx significa que permite full virtualización éste procesador.

Si no nos apareciera **vmx**, podemos intentar con un procesador AMD que sí lo permita, para lo cual realizamos lo siguiente:

```
cat /proc/cpuinfo|grep svm
check for vmx flag (intel) and svm flag (amd) in /proc/cpuinfo!)
```

svm es como se llama la virtualización en amd.

Caso contrario a esta respuesta, nuestro procesador AMD no permitirá full virtualización, sino solamente paravirtualización.

4.2.1.5. Creación de una partición LVM

Debemos considerar siempre que antes de instalar cualquier máquina virtual, necesitamos definir cuánto espacio en disco vamos a asignarle a la máquina virtual.
Lo primero que hay que hacer antes de crear la máquina virtual es crear el volumen lógico que la contendrá. En este caso, voy a trabajar con un disco de 160 GB. Usaré 20GB para crear 3 volumenes lógicos, para cada una de las máquinas virtuales. Para

identificarles se recomienda usar 3 letras y dos numeros. Las letras indicarán el sistema y los numeros serán un consecutivo.

Comenzaremos instalando una máquina de CentOS, será la primera por lo que la llamaremos "cen01".

```
lvcreate –L20G -ncen01 dsk
```

Aquí lo que hicimos fue crear un LV llamado cen01 que está basado en el VG llamado dsk y que tiene 20GB de tamaño.

A continuación procedemos a siguir creando el resto de volúmenes lógico, instalaremos otra máquina con CentOs, llamada “cen02”.

```
lvcreate –L20G -ncen01 dsk
```

Y trabajaremos con una máquina virtual con Windows 2003 server, la misma que la llamaremos “win01”

```
lvcreate –L20G -nwin01 dsk
```

4.2.1.6. Preparación del repositorio de instalación de CentOS

En principio instalaremos CentOS como domU, para lo cual lo realizaremos desde una máquina aparte, montaremos el DVD de CentOS5 en /var/www/html y arrancamos el servidor apache:

```
mount /dev/dvd /var/www/html
service httpd start
```

En caso de no tener apache instalado, digitamos: **yum install httpd**

Por motivos de seguridad es recomendable realizarlo desde una máquina aparte, pero si no tenemos más máquinas, lo podemos monta el dvd en el dom0 y arranca el httpd en el dom0.

4.2.1.7 Creación de una máquina virtual

4.2.1.7.1. Instalación de sistema operativo Linux como Guest – Método Paravirtualización

- **Usando el virt-install**

CentOS 5 ofrece una herramienta llamada virt-install, que a través de una serie de preguntas nos permite instalar una máquina paravirtual o full virtual.

En el caso de las paravirtuales solamente podríamos instalar RHEL (Red Hat Enterprise Linux), CentOS o Fedora. Pues son las distribuciones que vienen con un kernel preparado desde el mismo CD que les permite bootear en ambiente paravirtual.

Al ejecutar "virt-install" se mostrará algunas preguntas, como:

```
[root@xen ~]# virt-install
Would you like a fully virtualized guest (yes or no)?  This will allow you to run unmodified operating systems. No
What is the name of your virtual machine? cen01
How much RAM should be allocated (in megabytes)? 256
What would you like to use as the disk (path)? /dev/dsk/cen01
Would you like to enable graphics support? (yes or no) no
What is the install location? http://172.30.60.104/CENTOS/centos
```

En resumen: Le indicamos que no quieremos full virtual. Le dijimos que mi máquina se llamaría cen01, tendría 256MB de RAM, el disco sería el LV que acabamos de crear antes (/dev/dsk/cen01), que no quiero soporte gráfico y que la URL donde estaba el CD de instalación es: http://172.30.60.104/CENTOS/centos

Termiando esta etapa comenzará el proceso de instalación de CentOs en el domU (llamado cen01).

Esta última es la máquina donde montamos el DVD de centos5. El DVD lo montamos en /var/www/html y arrancamos apache, por lo que podemos acceder.

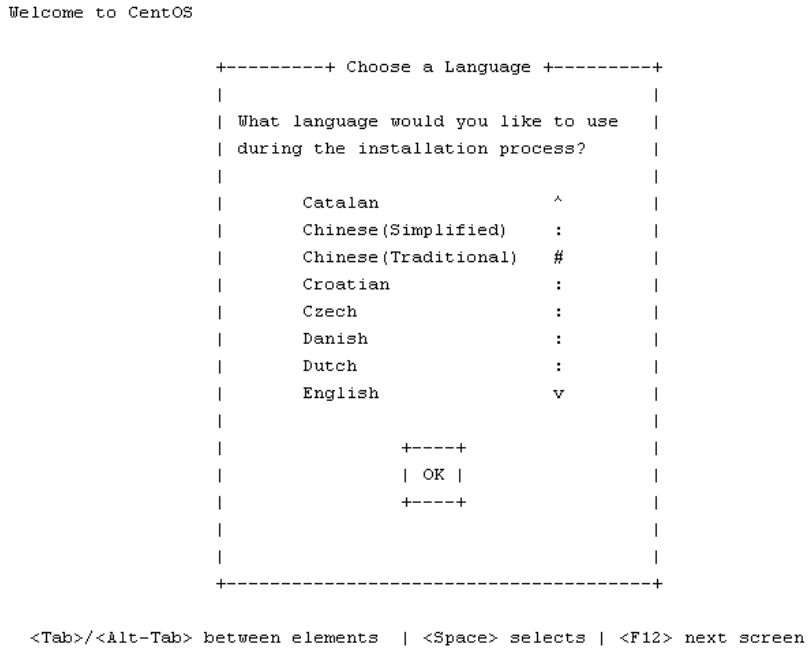

Figura IV.51. Choose a Language

La instalación inicia en ambiente gráfico, con la ayuda de las teclas TAB y enter y las teclas de cursor vamos instalando.

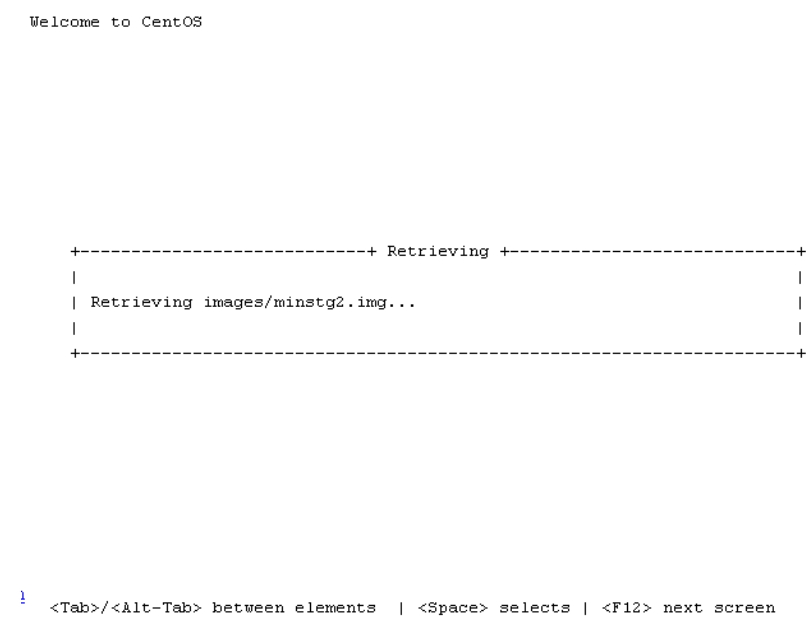

Figura IV.52. Retrieving

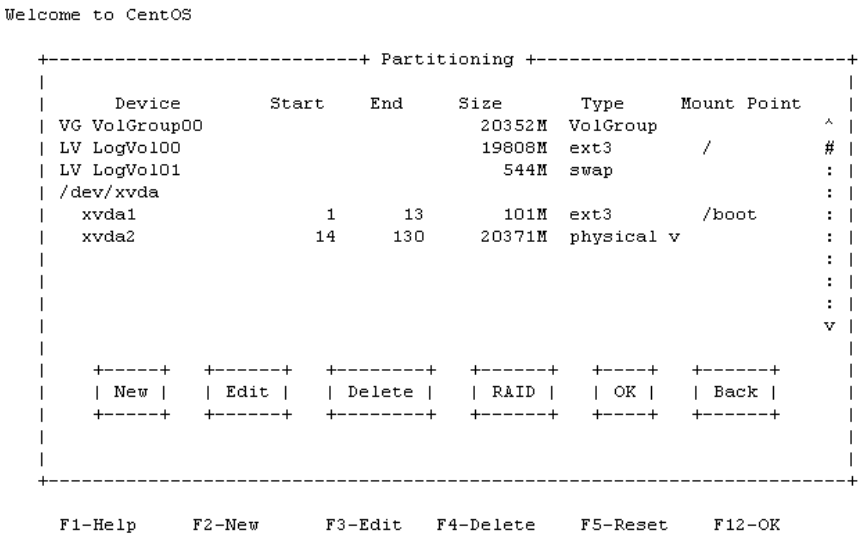

Figura IV.53. Partitioning

Aquí nos muestra la pantalla del particionamiento del disco del domU. Se sugiere conservarla.

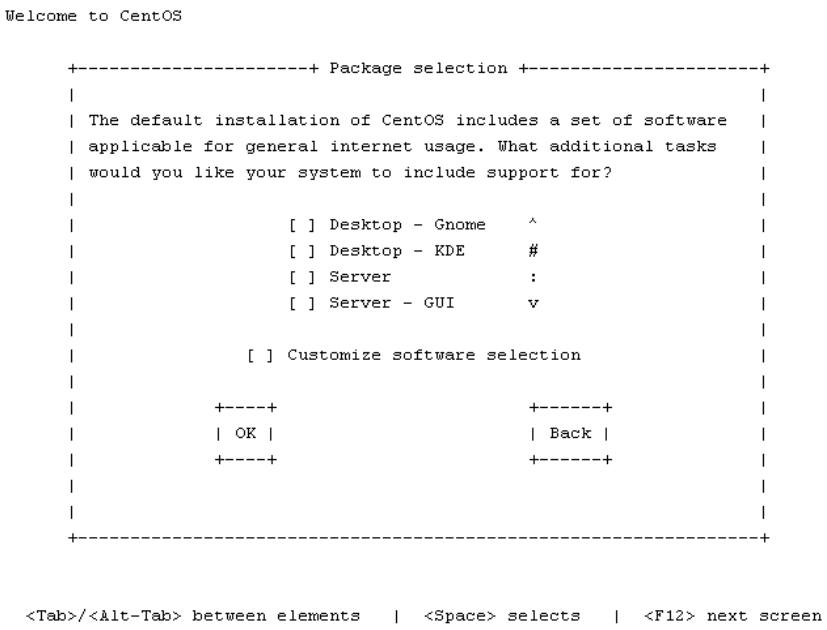

Figura IV.54. Package selecction

Aquí nos lista los paquetes, se sugiere lo minimo en principio, después vamos incrementando de acuerdo a la necesidad.

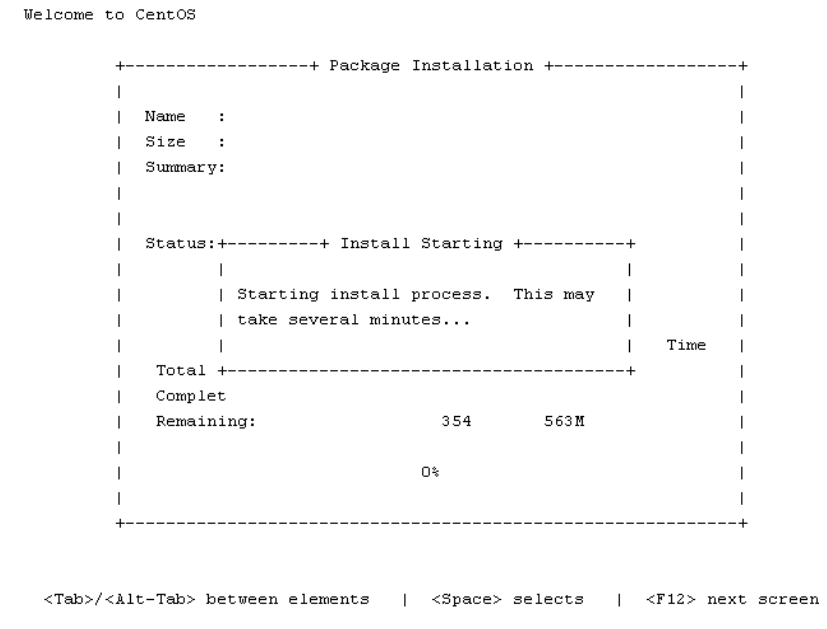

Figura IV.55. Install Starting

Comienza la instalación.

Si no nos da error, es que todo estuvo muy bien. Si nos algún error, seguro que no actualizamos el dom0, o seguro que no hemo montado bien el dvd, o no hemos creado bien el LV.

- **Usando la herramienta Virtual Machine Manager**

Figura IV.56. Conexión del Administrador de máquinas virtuales

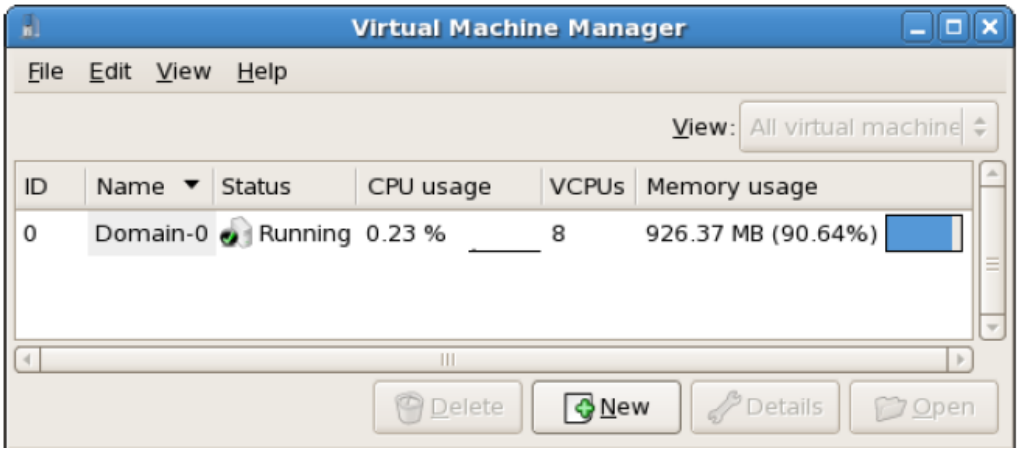

Figura IV.57. Ventana Principal de Administrador de Máquinas Virtuales

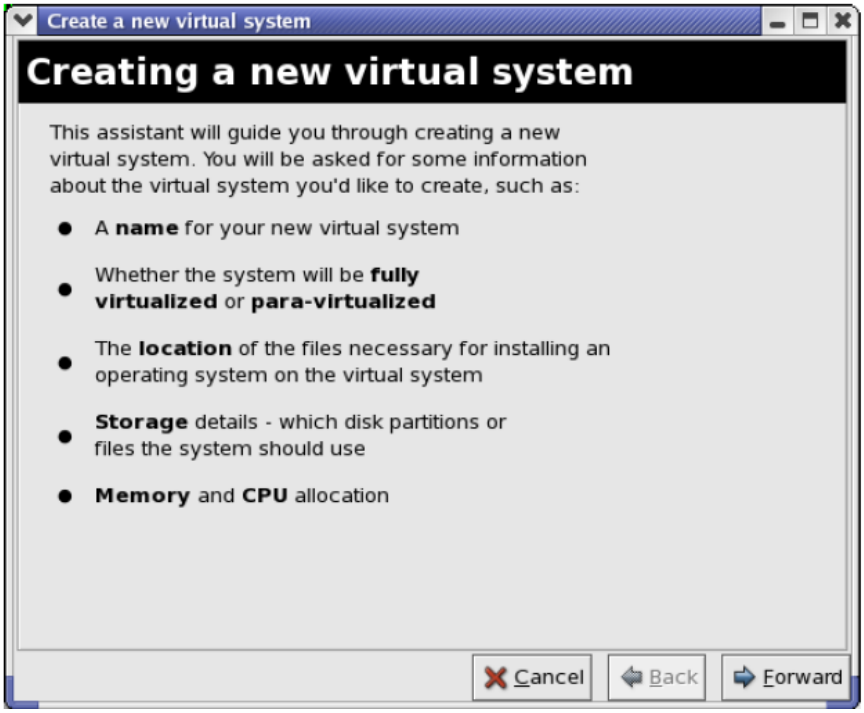

Figura IV.58. Asistente Creación de un Nuevo Sistema Virtual

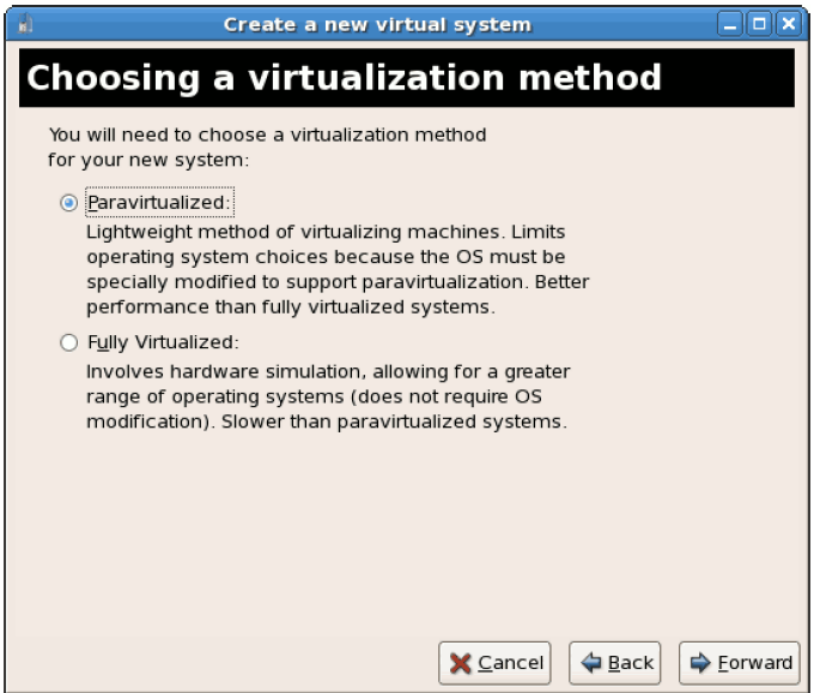

Figura IV.59. Escoger un método

Figura IV.60. Ingreso de un Nombre al Sistema Virtual

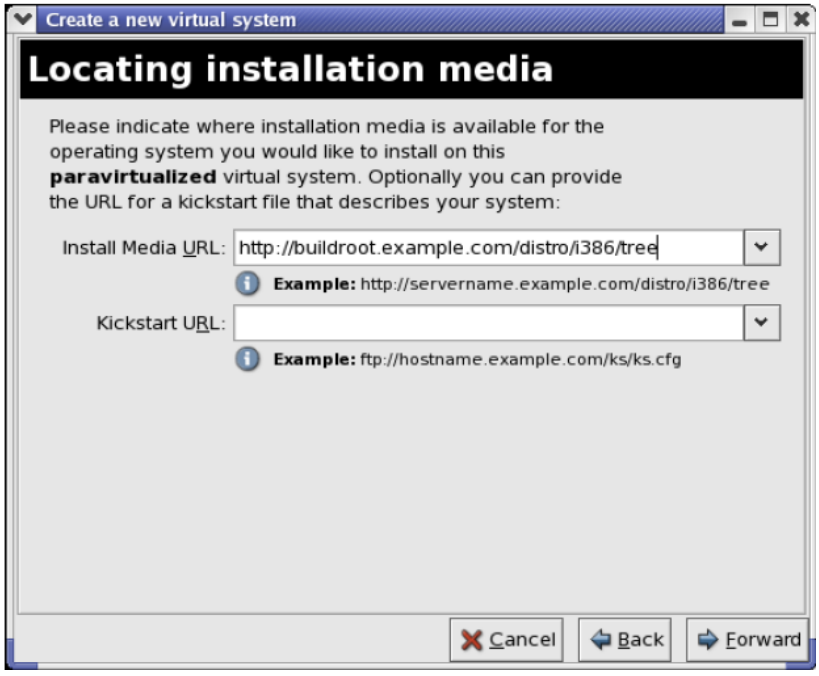

Figura IV.61. Ubicar el medio de instalación

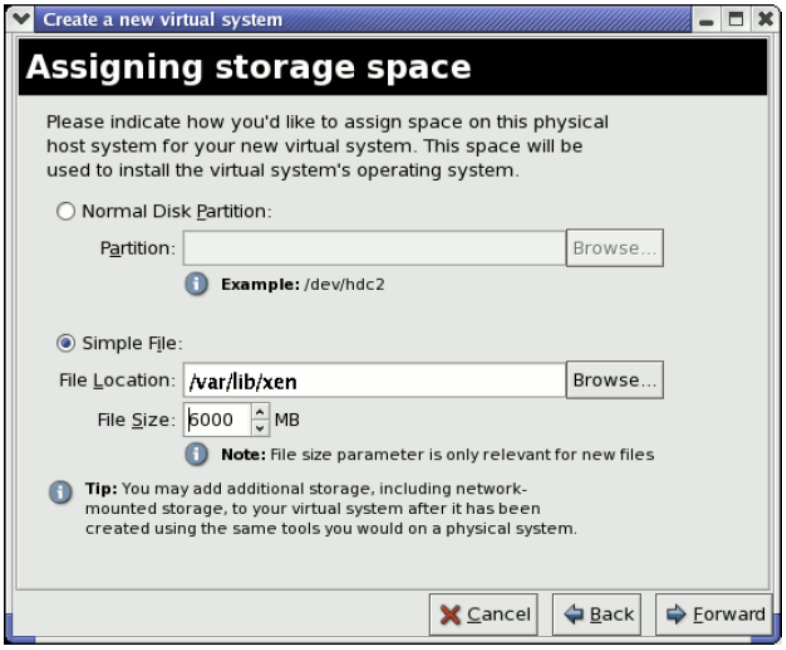

Figura IV.62. Asignar espacio de almacenamiento

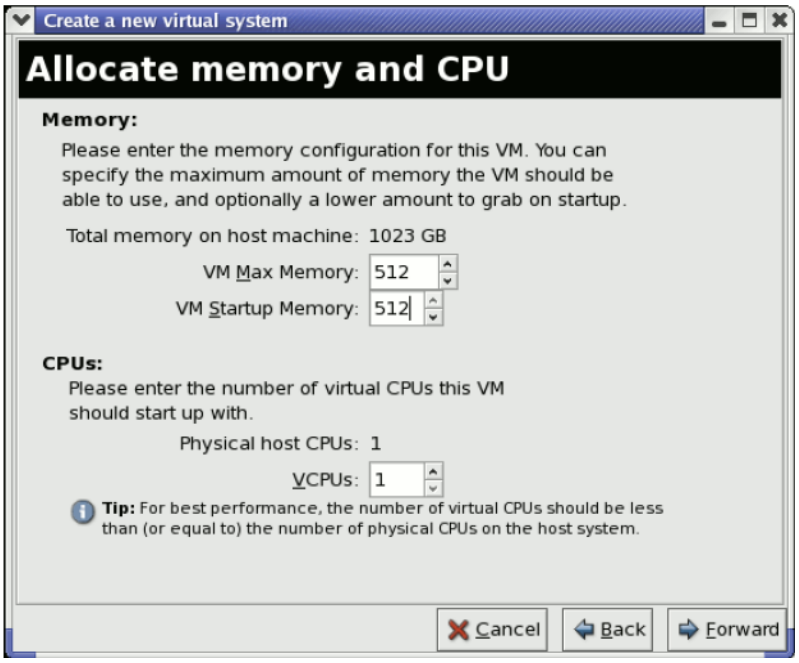

Figura IV.63. Asignar memoria y CPU

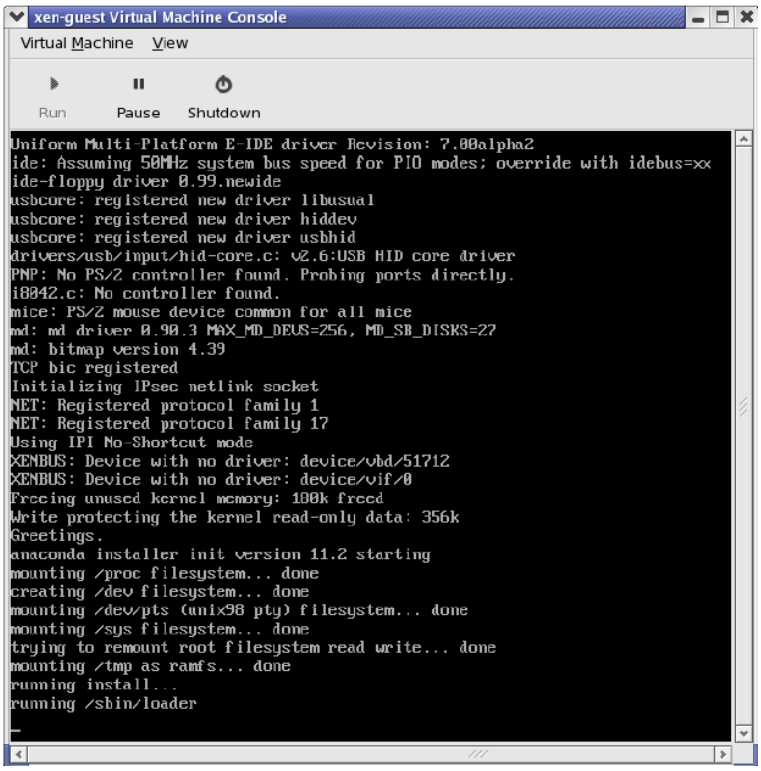

Figura IV.64. Inicio de la instalación

4.2.1.7.2. Instalación de sistema operativo Guest Windows – Método Full Virtualización

- **Usando el virt-install**

Para la instalación del sistema operativo Windows 2003 Server se realizará con el método Full Virtualización, el procedimiento se describe a continuación:

- **Crear un logical volumen**

```
[root@xen ~]# lvcreate -L20G -nwin01 dsk
  Logical volume "win01" created
```

- **Utilizar virt-install**

```
[root@xen ]# virt-install
Would you like a fully virtualized guest (yes or no)?  This will allow you to run unmodified operating systems. yes
What is the name of your virtual machine? win01
How much RAM should be allocated (in megabytes)? 512
What would you like to use as the disk (file path)? /dev/dsk/win01
Would you like to enable graphics support? (yes or no) yes
What is the virtual CD image, CD device or install location? /dev/scd0
```

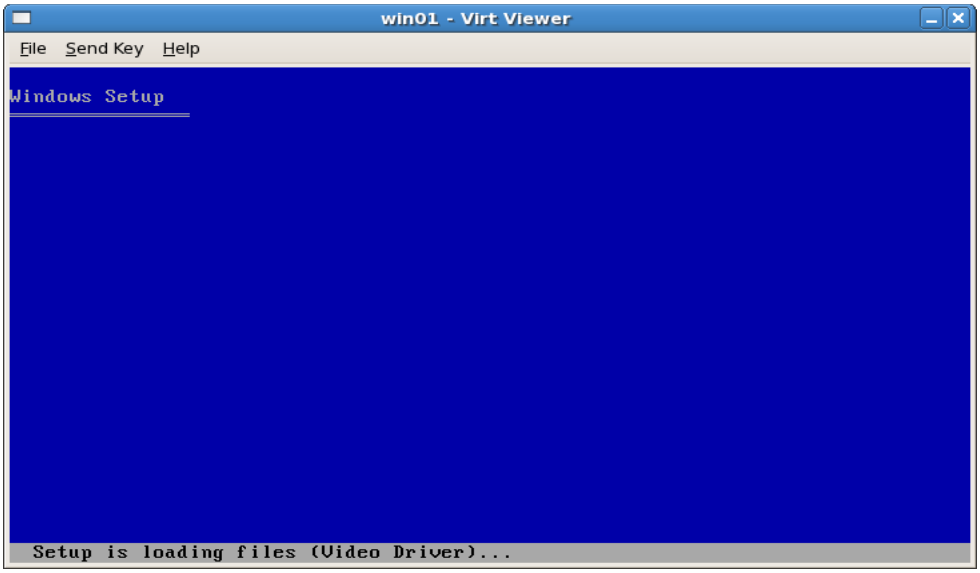

Figura IV.65. Windows Setup

- **Usando Virtual Machine Manager**

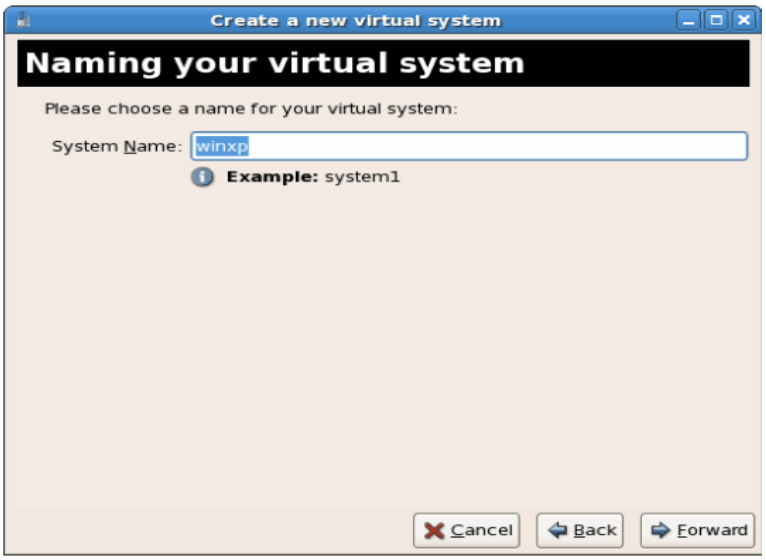

Figura IV.66. Ingreso de un Nombre al Sistema Virtual

Figura IV.67. Ubicar el medio de instalación

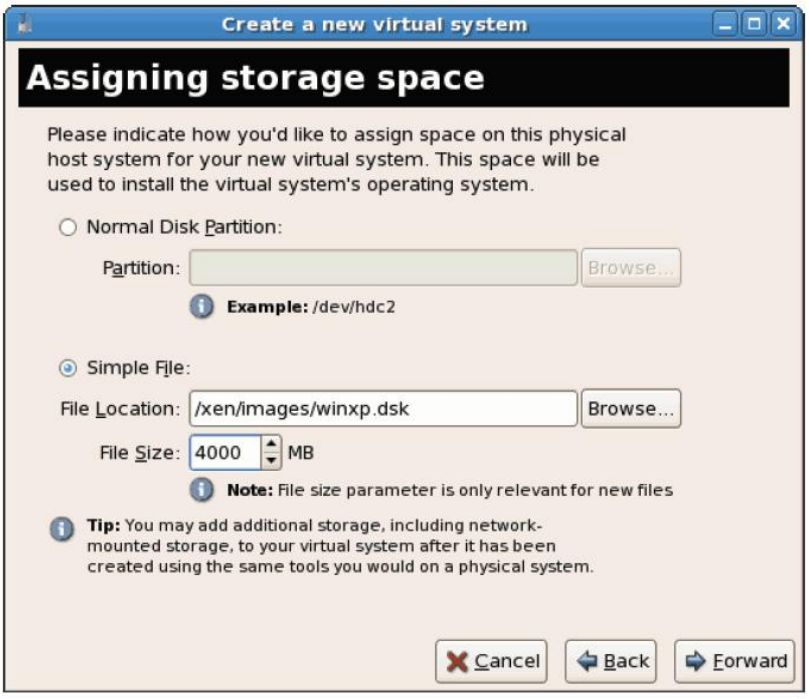

Figura IV.68. Asignar espacio de almacenamiento

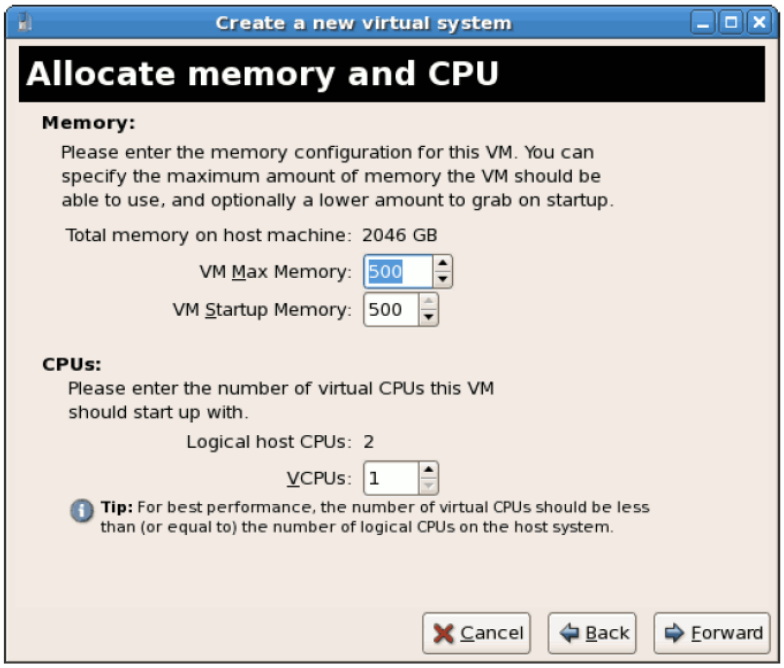

Figura IV.69. Asignar memoria y CPU

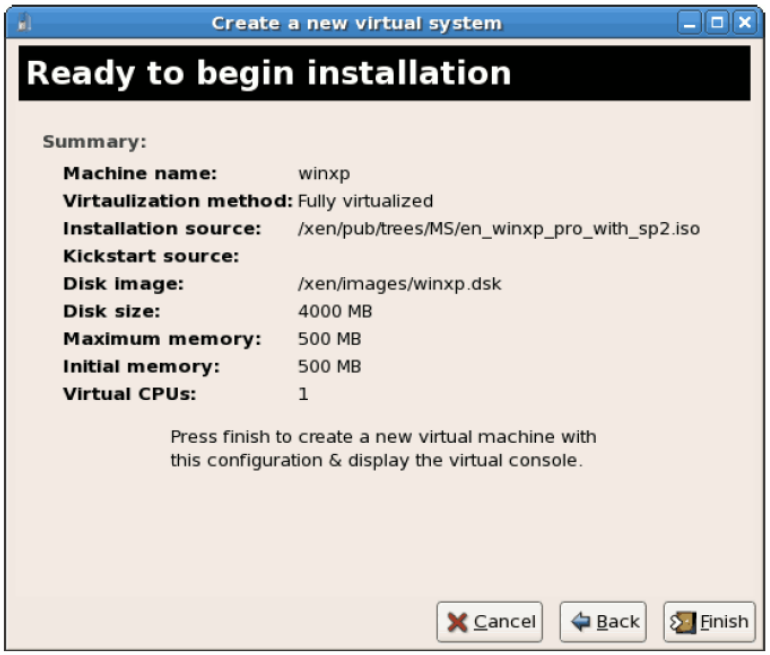

Figura IV.70. Empezar la instalación

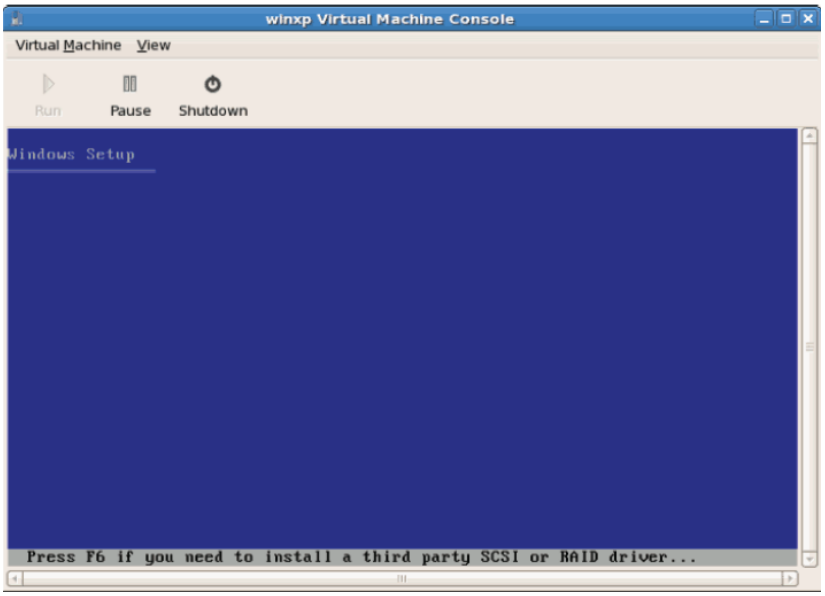

Figura IV.71. Instalación de Windows

4.2.2. Gestión de los servidores virtuales

4.2.2.1. Línea de Comandos

- **Listar servidores virtuales**

- **xm list**

Visualiza todas las máquinas virtuales existentes:

```
[root@xen ~]# xm list
Name                        ID Mem(MiB) VCPUs State   Time(s)
Domain-0                     0     458     2          r-----   2293.9
cos01                        7     511     1 -b----     20.5
win01                        5     519     1 -b----    118.0
```

Podemos apagar y encender las máquinas por su nombre (Name)

- **Encender servidores virtuales**

```
xm create -c cen01
```

Con esto haremos que se arranque la máquina llamada cen01 y para ver cómo arranca (**-c**: Es la consola)

Para salir de la consola, apretamos: **CTRL -**]

- **Apagar servidores virtuales**

Podemos dentro de la máquina poner: **poweroff**

O desde el dom0 podemos poner: **xm shutdown** cen01

Todas las configuraciones de los domU (máquinas virtuales) se guardan en: */etc/xen*

- **Arrancar automáticamente servidores virtuales**

Si queremos arrancar una máquina virtual cada vez que arranque el dom0, ponemos:

```
ln -s /etc/xen/cen01 /etc/xen/auto/cen01
```

xen al arrancar mira siempre los contenidos de /etc/xen/auto y las máquinas que ahi ve, las arranca en orden alfabético.

4.2.2.2. Administración Remota VNC

4.2.2.2.1. Configuración de VNC

Previo a esta configuración es necesario especificar en /etc/host el full name

```
[root@xen ~]# vi /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1               localhost.localdomain localhost
172.30.60.39         xen.eis.espoch xen
::1                        localhost6.localdomain6 localhost6
```

- **Validar el nombre del host**

```
[root@xen ~]# hostname
xen.eis.espoch
```

- **Modificar el archivo: vncservers**

```
[root@xen sysconfig]# pwd
/etc/sysconfig
[root@xen sysconfig]# vi vncservers
# The VNCSERVERS variable is a list of display:user pairs.
#
# Uncomment the lines below to start a VNC server on display :2
# as my 'myusername' (adjust this to your own).  You will also
# need to set a VNC password; run 'man vncpasswd' to see how
# to do that.
#
# DO NOT RUN THIS SERVICE if your local area network is
# untrusted!  For a secure way of using VNC, see
# <URL:http://www.uk.research.att.com/archive/vnc/sshvnc.html>.

# Use "-nolisten tcp" to prevent X connections to your VNC server via TCP.

# Use "-nohttpd" to prevent web-based VNC clients connecting.

# Use "-localhost" to prevent remote VNC clients connecting except when
# doing so through a secure tunnel.  See the "-via" option in the
# `man vncviewer' manual page.

# VNCSERVERS="2:myusername"
# VNCSERVERARGS[2]="-geometry 800x600 -nolisten tcp -nohttpd -localhost"

VNCSERVERS="1:root"
VNCSERVERARGS[1]="-geometry 1280x800"
```

- **Poner clave al vnc**

```
[root@xen sysconfig]# vncpasswd
Password:
Verify:
```

- **Reiniciar el servicio de VNC**

```
[root@xen sysconfig]# service vncserver status
Xvnc is stopped
[root@xen sysconfig]# service vncserver start
Starting VNC server: 1:root xauth:  creating new authority file /root/.Xauthority

New 'xen.eis.espoch:1 (root)' desktop is xen.eis.espoch:1

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/xen.eis.espoch:1.log

[  OK  ]
```

- **Habilitamos el servicio vncserver permanente**

```
[root@xen sysconfig]# chkconfig vncserver on
```

- **6.- Descomentamos la dos primeras filas para mantener el escritorio normal del servidor, en el archivo xstartup ubicado en el home directorio del usuario root.**

```
[root@xen ~]# pwd
/root
[root@xen ~]# cd .vnc
[root@xen .vnc]# ls
passwd  xen.eis.espoch:1.log  xen.eis.espoch:1.pid  xstartup
[root@xen .vnc]# cat xstartup
#!/bin/sh

# Uncomment the following two lines for normal desktop:
unset SESSION_MANAGER
exec /etc/X11/xinit/xinitrc

[ -x /etc/vnc/xstartup ] && exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] && xrdb $HOME/.Xresources
xsetroot -solid grey
vncconfig -iconic &
xterm -geometry 80x24+10+10 -ls -title "$VNCDESKTOP Desktop" &
twm &
```

- **En un computador cliente Windows instalamos el cliente VNC, llamado tightvnc-1.3.10-setup, para acceder al servidor Xen, a continuación las pantallas del procedimiento:**

✓ Instalación de VNC en cliente Windows

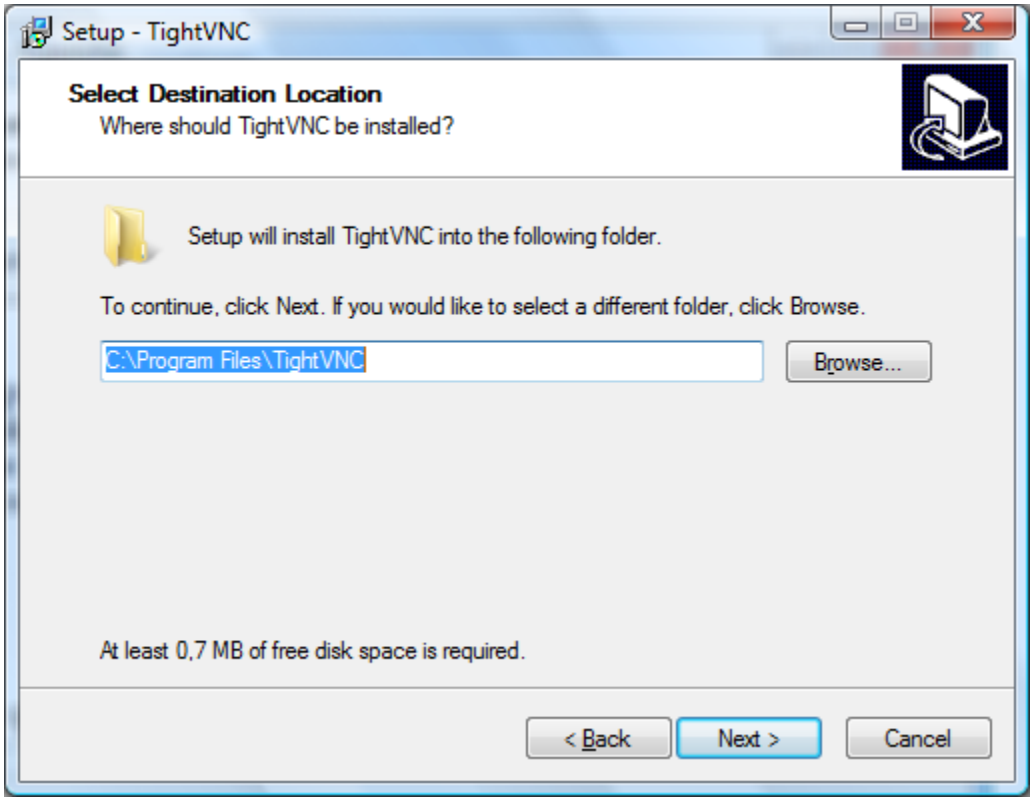

Figura IV.72. TightVNC – Selección Path de Instalación

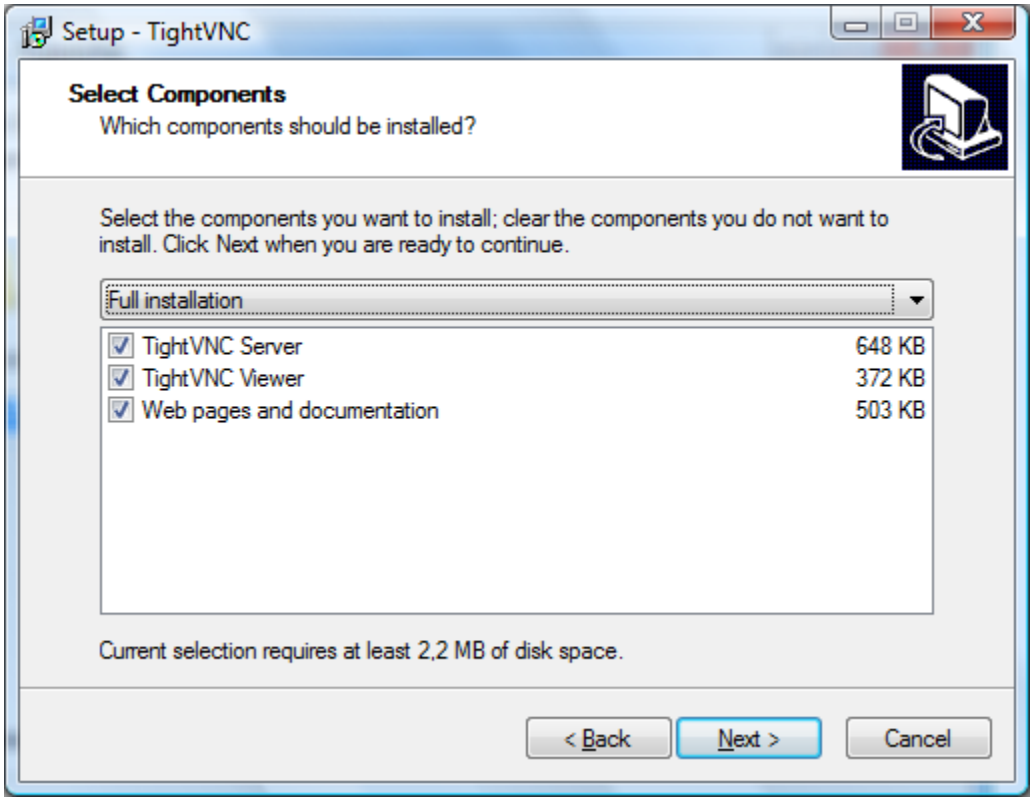

Figura IV.73. TightVNC – Seleccionar Componentes

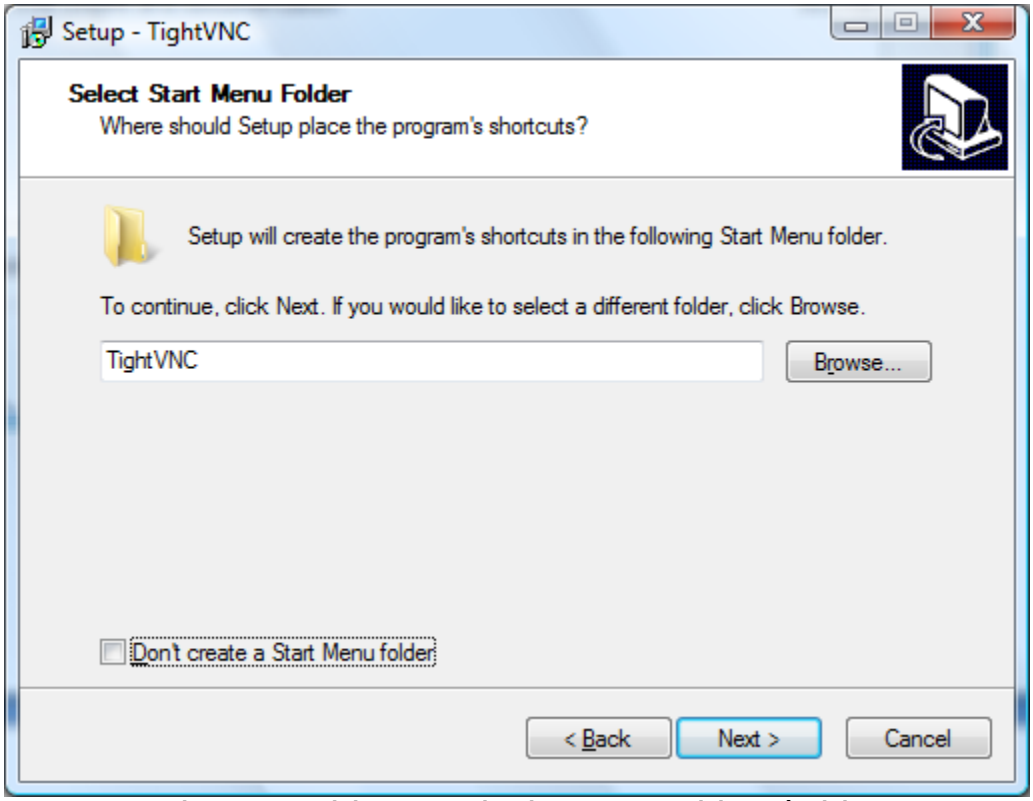

Figura IV.74. TightVNC – Seleccionar Carpeta del Menú Inicio

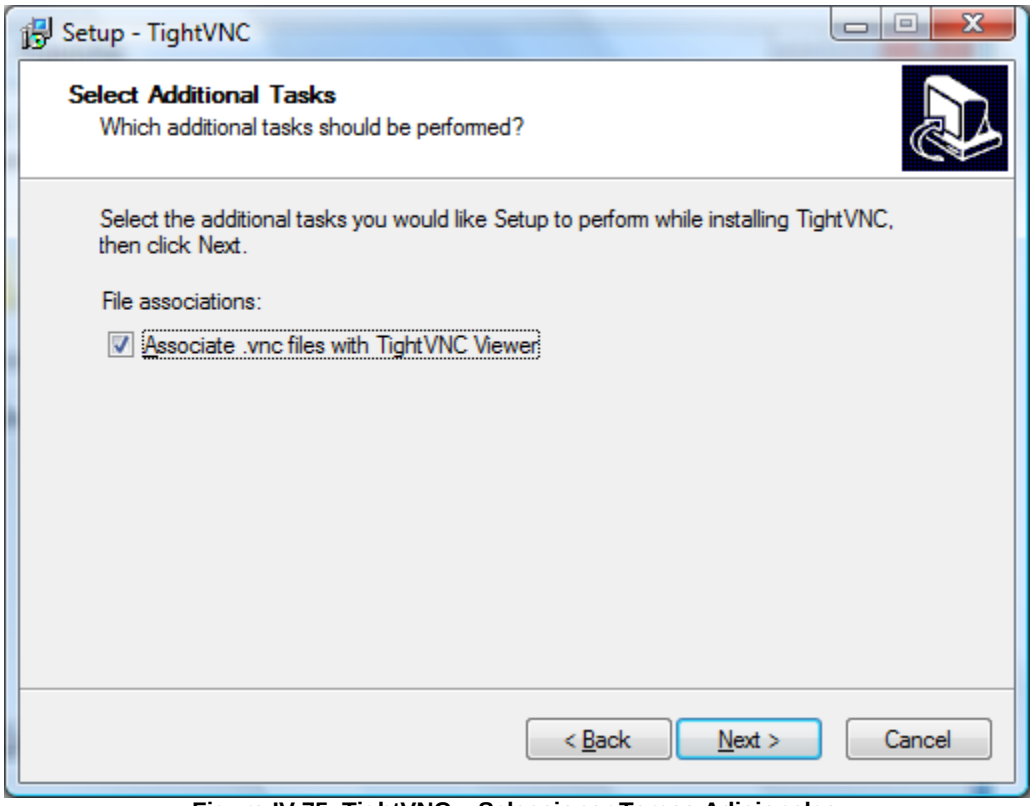

Figura IV.75. TightVNC – Seleccionar Tareas Adicionales

✓ Acceso

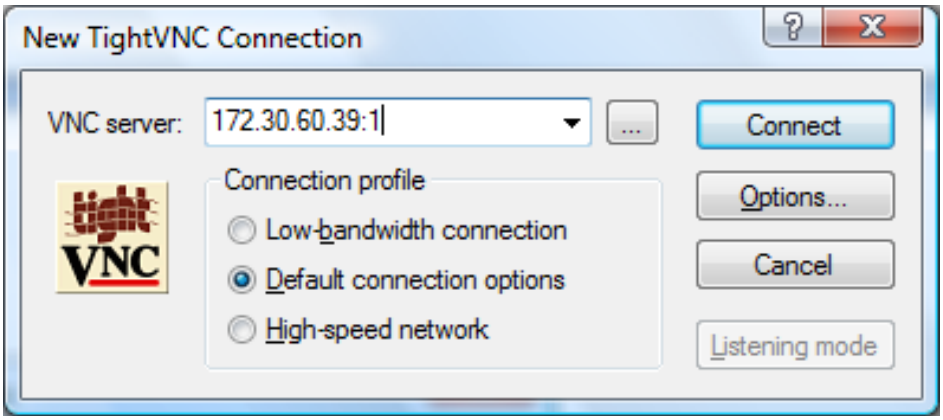

Figura IV.76. TightVNC – Nueva Conexión TightVNC

4.2.2.3. Administración Web SACV

4.2.2.3.1. Aplicación Web SACV

La aplicación Web SACV, un front-end con licencia GPL bastante útil para gestionar vía Web máquinas virtuales que usen los hypervisores XEN.

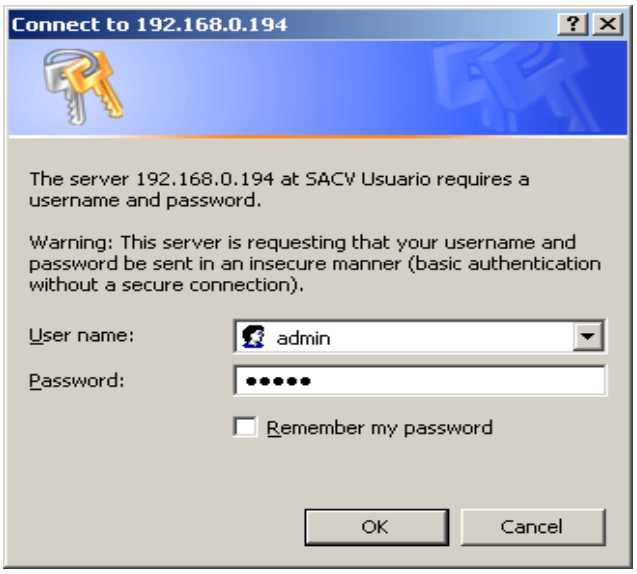

Figura IV.77. SACV – Autentificación

Figura IV.78. SACV: Administración de Servidore Virtuales

Figura IV.79. SACV: Creación de Usuarios

4.2.2.4. Monitoreo Web Xymon

4.2.2.4.1. Instalación del Xymon en el Host xen.eis.espoch

- **Prerequisitos**

- Una navegador de internet como Internet explore 5 o superior /Mozilla Firefox
- Sistema Operativo Unix
- Instalación y configuración de un servidor Web por ejemplo Apache

```
[root@xen CentOS]# rpm -ivh httpd-2.2.3-22.el5.centos.i386.rpm
```

- Instalar un compilar C. El hobbit esta escribo en lenguaje C y requiere ser compilado para esto es necesario instalar el compilador gcc, luego se utilizaría el utilitario make.

```
[root@xen CentOS]# rpm -ivh gcc-4.1.2-44.el5.i386.rpm
```

- Instalar las librerías para el Hobbit: PCRE, RRDtool, libpng, OpenSSL, OpenLDAP:

✓ Librería PCRE Perl Compatible Regular Expression, enlace cadenas de texto. Paquete *pcre-6.6-2.el5_1.7.i386.rpm* y dependencias:

```
[root@xen 32bits]# rpm -ivh pcre-6.6-2.el5_1.7.i386.rpm
[root@xen 32bits]# rpm -ivh pcre-devel-6.6-2.el5_1.7.i386.rpm
```

✓ Librería RRDtool Round-Robin Databases, es una base de datos para almacenar la data histórica del Hobbit. Paquete rrdtool-1.3.9-1.el5.wrl.i386.rpm y dependencias:

```
[root@xen 32bits]# rpm -ivh ruby-libs-1.8.5-5.el5_2.6.i386.rpm
[root@xen 32bits]# rpm –ivh ruby-devel-1.8.5-5.el5_2.6.i386.rpm
[root@xen 32bits]# rpm -ivh ruby-1.8.5-5.el5_2.6.i386.rpm
[root@xen 32bits]# rpm -ivh rrdtool-devel-1.3.9-1.el5.wrl.i386.rpm
[root@xen 32bits]# rpm -ivh rrdtool-perl-1.3.9-1.el5.wrl.i386.rpm
[root@xen 32bits]# rpm -ivh rrdtool-1.3.9-1.el5.wrl.i386.rpm
```

✓ Librería libpng, para generar imágenes en formato PNG. Utiliza la RRDtool.

```
[root@xen 32bits]# rpm -ivh libpng-1.2.10-7.1.el5_0.1.i386.rpm
```

✓ Librería OpenSSL, permite la comunicación con servicios de red, que utiliza encriptación SSL, por ejemplo para sitio web seguros.

```
[root@xen CentOS]# rpm -ivh openssl-0.9.8e-7.el5.i386.rpm
```

✓ Librería OpenLDAP, es usado para consultar en servidores directorios LDAP.

```
[root@xen CentOS]# rpm -ivh openldap-2.3.43-3.el5.i386.rpm
```

✓ Instalar la libería fping.

```
[root@xen 32bits]# rpm -ivh fping-2.4b2-7.el5.kb.i386.rpm
```

- Creación del grupo y usuario hobbit en el host

```
[root@xen ~]# groupadd hobbit
[root@xen ~]# useradd -g hobbit -d /home/hobbit hobbit
[root@localhost tmp]# passwd hobbit
Changing password for user hobbit.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

➢ **Instalación y Configuración del script para Xymon**

- Se procede a descomprimir y desempaquear el Xymon

```
[root@xen hobbit]# tar zxfv xymon-4.2.3.tar.gz
[root@xen hobbit]# cd xymon-4.2.3
[root@xen xymon-4.2.3]# ./configure
```

- Al ejecutar el script solicitará algunas preguntas y creará el Makefile para compilar el Xymon.

```
Configuration script for Xymon
This script asks a few questions and builds a Makefile to compile Xymon

Checking your make-utility
Checking pre-requisites for building Xymon

Checking for fping ...
Hobbit has a built-in ping utility (hobbitping)
However, it is not yet fully stable and therefore it
may be best to use the external fping utility instead.
I found fping in /usr/sbin/fping
Do you want to use it [Y/n] ?
Y
Checking to see if '/usr/sbin/fping 127.0.0.1' works ...
127.0.0.1 is alive
OK, will use '/usr/sbin/fping' for ping tests
```

```
NOTE: If you are using an suid-root wrapper, make sure the 'hobbit'
      user is also allowed to run fping without having to enter passwords.
      For 'sudo', add something like this to your 'sudoers' file:
      hobbit: ALL=(ALL) NOPASSWD: /usr/local/sbin/fping
Checking for RRDtool ...
test-rrd.c: In function ‘main’:
test-rrd.c:30: error: too few arguments to function ‘rrd_graph’
make: *** [test-compile] Error 1
Not RRDtool 1.0.x, checking for 1.2.x
Found RRDtool include files in /usr/include
Found RRDtool libraries in /usr/lib

Checking for PCRE ...
Found PCRE include files in /usr/include
Found PCRE

Checking for OpenSSL ...
Found OpenSSL include files in /usr/include
Found OpenSSL libraries in /usr/lib

Xymon can use the OpenSSL library to test SSL-enabled services
like POP3S, IMAPS, NNTPS and TELNETS. If you have the OpenSSL
library installed, I recommend that you enable this.

Do you want to be able to test SSL-enabled services (y) ?

Checking for LDAP ...
Found LDAP include files in /usr/include
Found LDAP libraries in /usr/lib

Xymon can use your OpenLDAP LDAP client library to test LDAP servers.

Do you want to be able to test LDAP servers (y) ?

Enable experimental support for LDAP/SSL (OpenLDAP 2.x only) (y) ?

Checking for clock_gettime() requiring librt ...
clock_gettime() requires librt

Checking for Large File Support ...
Large File Support OK

Setting up for a Xymon server

What userid will be running Xymon [xymon] ?
hobbit
Found passwd entry for user hobbit:x:500:500::/home/hobbit:/bin/bash


Where do you want the Xymon installation [/home/hobbit] ?

OK, will configure to use /home/hobbit as the Xymon toplevel directory

What URL will you use for the Xymon webpages [/xymon] ?
/hobbit

Where to put the Xymon CGI scripts [/home/hobbit/cgi-bin] ?
(Note: This is the filesystem directory - we will get to the URL shortly)

What is the URL for the Xymon CGI directory [/xymon-cgi] ?
(Note: This is the URL - NOT the filesystem directory)
/hobbit-cgi

********** SECURITY NOTICE **********
If your Xymon server is accessible by outsiders, then you should
restrict access to the CGI scripts that handle enable/disable of
hosts, and acknowledging of alerts. The easiest way to do this is
to put these in a separate CGI directory and require a password to
access them.
Even if your Xymon server is on a secured, internal network, you
may want to have some operations (like disabling a host) be password-
protected - that lets you see who disabled or acknowledged an alert.
```

```
Where to put the Xymon Administration CGI scripts [/home/hobbit/cgi-secure] ?
(Note: This is the filesystem directory - we will get to the URL shortly)

What is the URL for the Xymon Administration CGI directory [/xymon-seccgi] ?
(Note: This is the URL - NOT the filesystem directory)
/hobbit-seccgi

** Note that you may need to modify your webserver configuration.
** After installing, see /home/hobbit/server/etc/hobbit-apache.conf for an example configuration.

To generate Xymon availability reports, your webserver
must have write-access to a directory below the Xymon
top-level directory. I can set this up if you tell me
what group-ID your webserver runs with. This is typically
'nobody' or 'apache' or 'www-data'

What group-ID does your webserver use [nobody] ?

Where to put the Xymon logfiles [/var/log/xymon] ?
/var/log/hobbit
What is the name of this host [xen] ?
What is the IP-address of this host [127.0.0.1] ?
172.30.60.39

Where should I install the Xymon man-pages (/usr/local/man) ?

Using Linux Makefile settings

Created Makefile with the necessary information to build Xymon
Some defaults are used, so do look at the Makefile before continuing.

Configuration complete - now run make (GNU make) to build the tools
```

```
[root@xen xymon-4.2.3]# make
```

- Después muestra el siguiente mensaje

```
Build complete. Now run 'make install' as root
```

- Y continuamos digitamos

```
[root@xen xymon-4.2.3]# make install
```

- Y obtendremos el mensaje

```
Installation complete.
You must configure your webserver for the Hobbit webpages and CGI-scripts.
A sample Apache configuration is in /home/hobbit/server/etc/hobbit-apache.conf
If you have your Administration CGI scripts in a separate directory,
then you must also setup the password-file with the htpasswd command.

To start Hobbit, as the hobbit user run '/home/hobbit/server/bin/hobbit.sh start'
To view the Hobbit webpages, go to http:// xen.eis.espoch/hobbit
```

```
[root@xen xymon-4.2.3]# cp /home/hobbit/server/etc/hobbit-apache.conf /etc/httpd/conf.d/
```

- Habilitamos los permisos 755 al directorio hobbit:

```
[root@xen home]# pwd
```

```
/home
[root@xen home]# ll
total 4
drwx------ 9 hobbit hobbit 4096 Feb  7 23:55 hobbit
[root@xen home]# chmod 755 hobbit/
[root@xen home]# ll
total 4
drwxr-xr-x 9 hobbit hobbit 4096 Feb  7 23:55 hobbit
```

- Reiniciamos el servicio de http

```
[root@xen home]# service httpd restart
Stopping httpd: [  OK  ]
Starting httpd: [  OK  ]
```

- Con el usuario hobbit, reiniciamos el cliente y server

```
[root@xen home]# su – hobbit
```

- Reinicio cliente

```
[hobbit@xen ~]$ cd client
[hobbit@xen client]$ ./runclient.sh start
Hobbit client for linux started on xen.eis.espoch
```

- Reinicio server

```
[hobbit@xen ~]$ cd server/
[hobbit@xen server]$ ./hobbit.sh start
Hobbit started
```

- Salimos del usuario hobbit

```
[hobbit@xen server]$ exit
logout
```

4.2.2.4.2. Instalación del Xymon en los Guest

- **Instalación en un Guest Unix**

Para la instalación del Xymon en un Guest Unix, se seguirá el siguiente procedimiento:

- Especificar en /etc/host el full name

```
[root@cos01 ~]# vi /etc/hosts
```

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1               localhost.localdomain localhost
172.30.60.40        cos01.eis.espoch cos01
::1             localhost6.localdomain6 localhost6
```

- Validar el nombre del host

```
[root@cos01 ~]# hostname
cos01.eis.espoch
```

- Instalar las prerrequisitos y librerías para el Xymon: PCRE, RRDtool, libpng, OpenSSL, OpenLDAP:

```
[root@cos01 32bits]# rpm -ivh ruby-libs-1.8.5-5.el5_2.6.i386.rpm
[root@cos0132bits]# rpm –ivh ruby-devel-1.8.5-5.el5_2.6.i386.rpm
[root@cos0132bits]# rpm -ivh ruby-1.8.5-5.el5_2.6.i386.rpm
[root@cos0132bits]# rpm -ivh rrdtool-devel-1.3.9-1.el5.wrl.i386.rpm
[root@cos0132bits]# rpm -ivh rrdtool-perl-1.3.9-1.el5.wrl.i386.rpm
[root@cos0132bits]# rpm -ivh rrdtool-1.3.9-1.el5.wrl.i386.rpm
[root@cos0132bits]# rpm –ivh fping-2.4b2-7.el5.kb.i386.rpm
[root@cos0132bits]# rpm –ivh libpng-1.2.10-7.1.el5_0.1.i386.rpm
[root@cos0132bits]# rpm –ivh openssl-0.9.8e-7.el5.i386.rpm
[root@cos0132bits]# rpm –ivh openldap-2.3.43-3.el5.i386.rpm
```

- Creación del usuario hobbit en el sistema

```
[root@cos01 ~]# groupadd hobbit
[root@cos01 ~]# useradd -g hobbit -d /home/hobbit hobbit
[root@cos01 ~]# passwd hobbit
Changing password for user hobbit.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

- Exportar la variable LIBRTDEF

Esta bandera es utilizada para el compilador

```
[root@cos01 ~]# export LIBRTDEF=-lrt
```

- Descomprimir el Xymon

```
[root@cos01 hobbit]# tar zxfv xymon-4.2.3.tar.gz
[root@cos01 hobbit]# cd xymon-4.2.3

[root@cos0 xymon-4.2.3]# ./configure.client

Configuration script for Xymon client
This script asks a few questions and builds a Makefile to compile Xymon
Checking your make-utility
Xymon normally keeps all of the client configuration files
on the Xymon server. If you prefer, it is possible to use
a local client configuration file instead - if so, answer
'client' to the next question.
```

```
NB: Local configuration requires the PCRE libs on each host.

Server side client configuration, or client side [server] ?

Checking for Large File Support ...
ERROR: Compiler doesnt recognize the off_t C type.

What userid will be running Xymon [xymon] ?
hobbit
Found passwd entry for user hobbit:x:500:500::/home/hobbit:/bin/bash

Where do you want the Xymon installation [/home/hobbit] ?
OK, will configure to use /home/hobbit as the Xymon toplevel directory

What is the IP-address of your Xymon server [127.0.0.1] ?
172.30.60.39
Using Linux Makefile settings

Created Makefile with the necessary information to build Xymon
Some defaults are used, so do look at the Makefile before continuing.

Configuration complete - now run make (GNU make) to build the tools
```

- Creación del Makefile con la información necesaria para construir el Xymon

```
[root@cos01 xymon-4.2.3]# make
```

- Y muestra el siguiente mensaje:

```
Build complete. Now run 'make install' as root
```

- Y continuamos con:

```
[root@cos01 xymon-4.2.3]# make install
```

- Y obtendremos el mensaje:

```
Installation complete.
To start the Hobbit client, as the hobbit user run '/home/hobbit/client/runclient.sh start'
```

- Iniciar el servicio del hobbit en el cliente

```
[root@cos01 xymon-4.2.3]# su - hobbit
[hobbit@cos01 ~]$ cd client
[hobbit@cos01 client]$ ls
bin  etc  ext  logs  runclient.sh  tmp
[hobbit@cos01 client]$ ./runclient.sh start
Hobbit client for linux started on cos01
```

- **Instalación en un Guest Windows**

Para que los Guest Windows sean monitoreados por la aplicación Xymon, se instalará el BBWin, a continuación se define los pasos:

- Ejecutar el instalador llamador BBWin_0.12.msi, seguidamente se mostrará la pantalla de bienvenida.
- Aceptar la licencia.
- Se realizará una instalación por defecto.

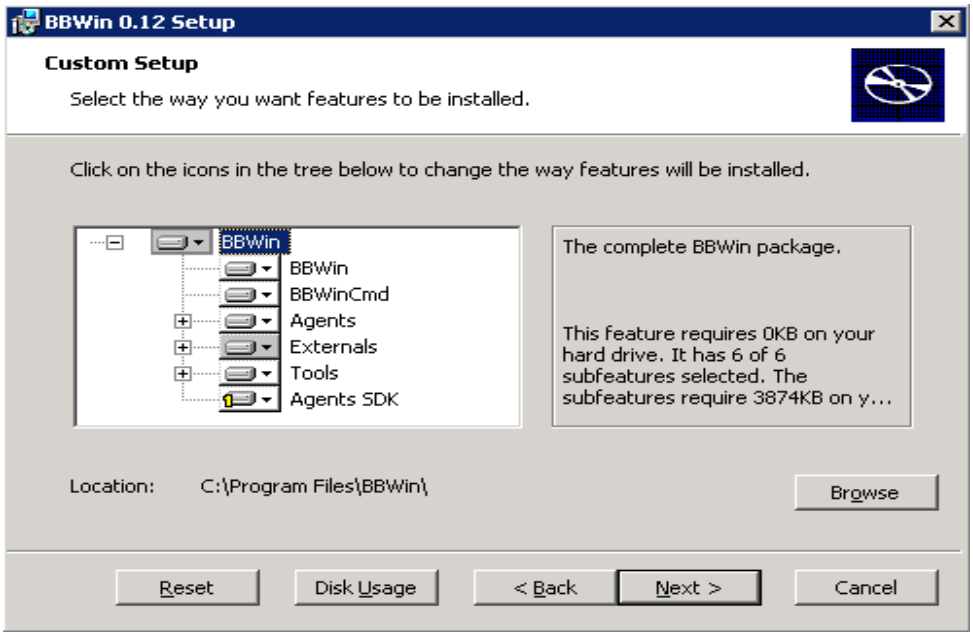

Figura IV.69. BBWin - Custom Setup

- Iniciar la instalación.
- Instalación completada.
- Editar el archivo de configuración del BBWin que se encuentra en el siguiente path: C:\Program Files\BBWin\etc\BBWin.cfg

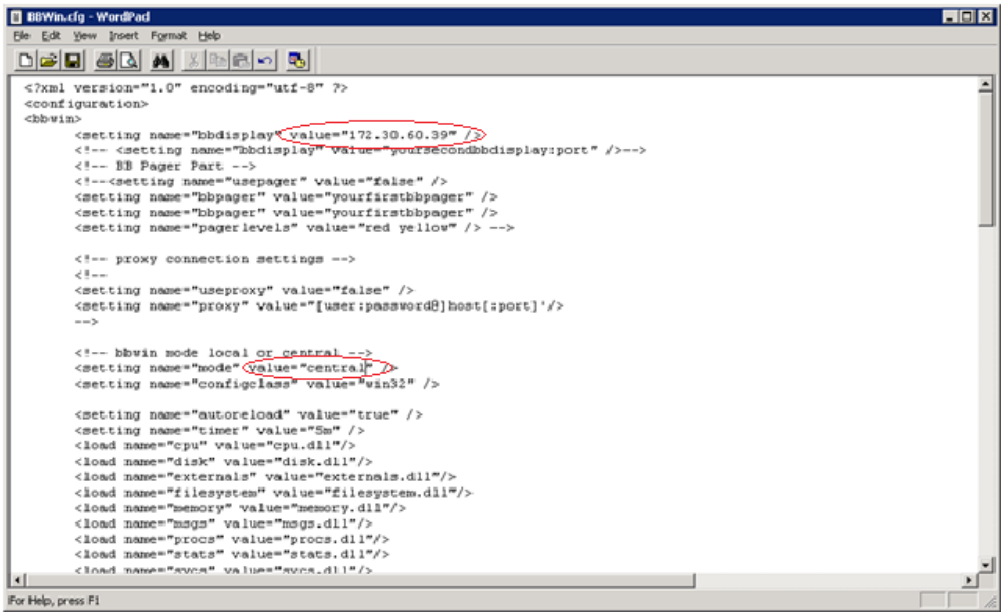

Figura IV.70. BBWin - Configurar archivo BBWin.cfg

- Reiniciar el servicio Big Brother Hobbit Client en el guest win01.

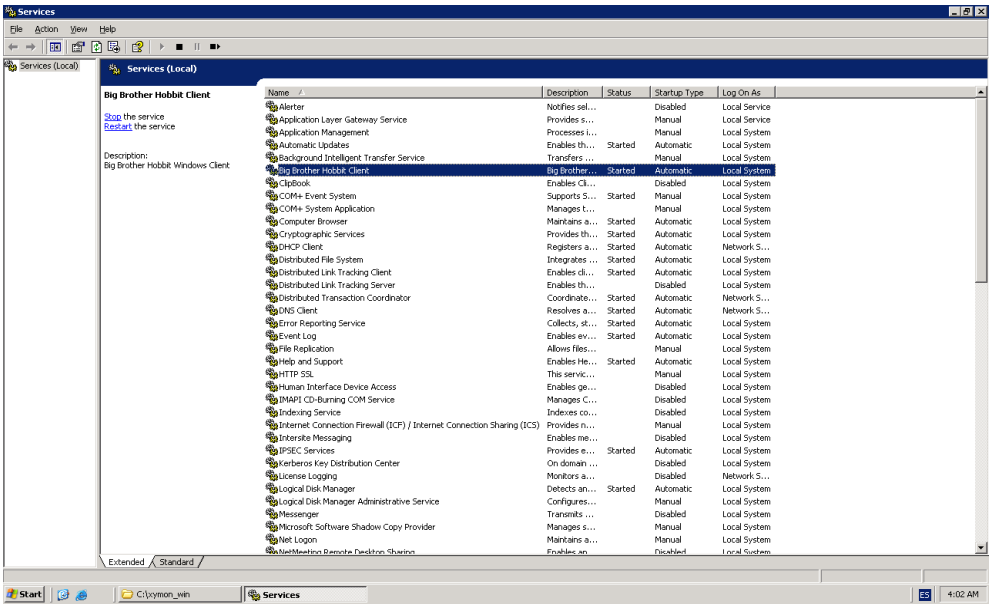

Figura IV.71. BBWin - Reinicio Servicio Big Brother Hobbit Client

4.2.2.4.3. Arranque automático de los servicios de Xymon

- **Servidor Host**

Para que los servicios de Xymon se levanten automáticamente después de un reinicio del sistema operativo en el servidor host, debemos configurarlos en los archivos de arranque.

A continuación se detalla el procedimiento:

- En el directorio init.d vamos a crear los scripts para subir y detener el servicio de hobbit, el path es el siguiente:

```
[root@xen ~]# cd /etc/init.d
```

- Crear los siguientes archivos:

```
[root@xen init.d]# touch hobbit-client
[root@xen init.d]# vi hobbit-client
[root@xen init.d]# touch hobbit-server
[root@xen init.d]# vi hobbit-server
```

El contenido de los archivos hobbit-server y hobbit-client se puede visualizar con más detalle en el anexo[6]

Estos archivos deben tener permisos de ejecución.

```
[root@xen init.d]# chmod 755 hobbit-*
```

En los directorios rc.d, especificamos la ejecución de los scripts, para ello creamos links de acceso.

[6] Configuración archivos hobbit-server y hobbit-client

✓ **Para el server:**

```
[root@xen etc]# cd rc.d
[root@xen rc.d]# pwd
/etc/rc.d
[root@xen rc.d]# ls
init.d  rc  rc0.d  rc1.d  rc2.d  rc3.d  rc4.d  rc5.d  rc6.d  rc.local  rc.sysinit
[root@xen rc.d]# cd rc3.d
[root@xen rc3.d]# ln -s ../init.d/hobbit-server S96hobbit-server
[root@xen rc3.d]# cd ../rc4.d
[root@xen rc4.d]# ln -s ../init.d/hobbit-server S96hobbit-server
[root@xen rc4.d]# cd ../rc5.d
[root@xen rc5.d]# ln -s ../init.d/hobbit-server S96hobbit-server
[root@xen rc5.d]# cd ../rc0.d
[root@xen rc0.d]# ln -s ../init.d/hobbit-server K01hobbit-server
[root@xen rc0.d]# cd ..
[root@xen rc.d]# cd rc1.d
[root@xen rc1.d]# ln -s ../init.d/hobbit-server K01hobbit-server
[root@xen rc1.d]# cd ..
[root@xen rc.d]# cd rc2.d
[root@xen rc2.d]# ln -s ../init.d/hobbit-server K01hobbit-server
[root@xen rc2.d]# cd ../rc6.d
[root@xen rc6.d]# ln -s ../init.d/hobbit-server K01hobbit-server
```

✓ **Para el cliente:**

```
[root@xen etc]# cd rc.d
[root@xen rc.d]# pwd
/etc/rc.d
[root@xen rc.d]# ls
init.d  rc  rc0.d  rc1.d  rc2.d  rc3.d  rc4.d  rc5.d  rc6.d  rc.local  rc.sysinit
[root@xen rc.d]# cd rc0.d
[root@xen rc0.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@xen rc1.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@xen rc2.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@xen rc3.d]# ln -s ../init.d/hobbit-client S96hobbit-client
[root@xen rc4.d]# ln -s ../init.d/hobbit-client S96hobbit-client
[root@xen rc5.d]# ln -s ../init.d/hobbit-client S96hobbit-client
[root@xen rc6.d]# ln -s ../init.d/hobbit-client K01hobbit-client
```

Tabla IV.23. Arranque Automático hobbit-server y hobbit-client en Servidor Host

NIVEL	S96hobbit-server	K01hobbit-server	S96hobbit-client	K01hobbit-client
rc0.d	-	✓	-	✓
rc1.d	-	✓	-	✓
rc2.d	-	✓	-	✓
rc3.d	✓	-	✓	-
rc4.d	✓	-	✓	-
rc5.d	✓	-	✓	-
rc6.d	-	✓	-	✓

- **Servidor Guest**

En el servidor guest la configuración del servicio de arranque automático es similar a la configuración en el servidor host, seguidamente se detalla el procedimiento:

El archivo hobbit-client ya creado para el cliente del host, copiamos en el path del servidor guest:

```
[root@xen init.d]# scp hobbit-client cos01:/etc/init
```

Procedemos a crear los links de acceso desde el directorio init.d hacia el nombre que tendrá en cada uno de los niveles de ejecución.

```
[root@cos01 rc1.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@cos01 rc1.d]# cd ../rc0.d
[root@cos01 rc0.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@cos01 rc0.d]# cd ../rc2.d
[root@cos01 rc2.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@cos01 rc2.d]# cd ../rc6.d
[root@cos01 rc6.d]# ln -s ../init.d/hobbit-client K01hobbit-client
[root@cos01 rc6.d]# cd ../rc3.d
[root@cos01 rc3.d]# ln -s ../init.d/hobbit-client S96hobbit-client
[root@cos01 rc3.d]# cd ../rc4.d
[root@cos01 rc4.d]# ln -s ../init.d/hobbit-client S96hobbit-client
[root@cos01 rc4.d]# cd ../rc5.d
[root@cos01 rc5.d]# ln -s ../init.d/hobbit-client S96hobbit-client
```

Tabla IV.24. Arranque Automático hobbit-client en Servidor Guest

NIVEL	S96hobbit-client	K01hobbit-client
rc0.d	-	✓
rc1.d	-	✓
rc2.d	-	✓
rc3.d	✓	-
rc4.d	✓	-
rc5.d	✓	-
rc6.d	-	✓

4.2.2.4.4. Arquitectura del Xymon

Para el monitoreo de los servidores virtuales (guest) y hosts, vamos a configurar algunos archivos de monitoreo del Xymon. A continuación de ilustra la arquitectura de los archivos de monitoreo Simón:

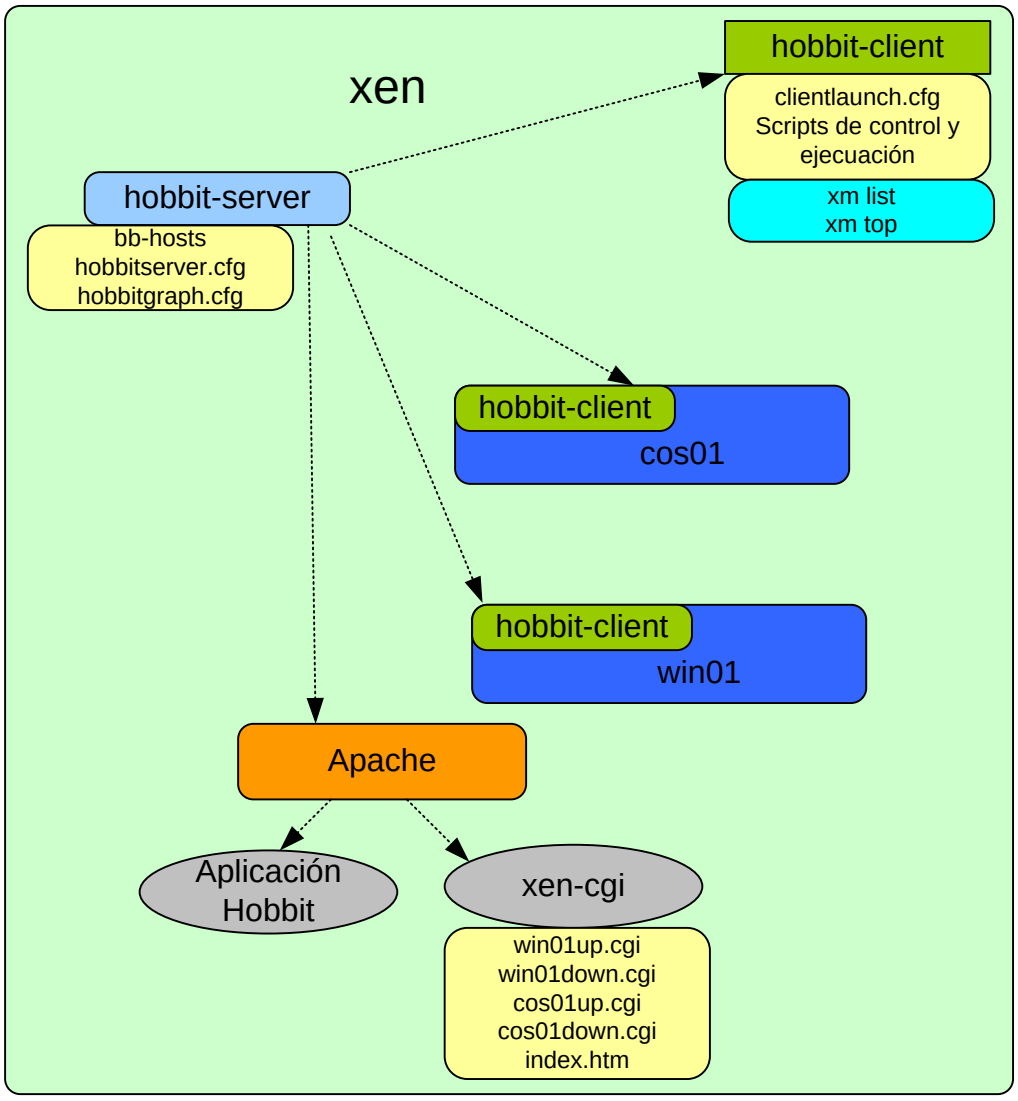

Figura IV.72. Arquitectura de los Archivos de Monitoreo Xymon

Vamos a crear los scripts para el M*onitoreo Detallo* de los servidores virtuales, estos se crearán en el cliente del servidor hosts, los pasos del procedimiento se describen a continuación:

En el path /home/hobbit/client/ext, creamos los scrips de control para el monitoreo detallado de los servidores guest:

```
[hobbit@xen ext]$ touch ctl_xm_list.sh
[hobbit@xen ext]$ touch ctl_infMV.sh
[hobbit@xen ext]$ touch ctl_xmtop.sh
[hobbit@xen ext]$ touch ctl_dom0disp.sh
[hobbit@xen ext]$ touch ctl_cos01disp.sh
[hobbit@xen ext]$ touch ctl_win01disp.sh
```

Estos scripts se ejecutan automaticamente según la definicion en el archivo "clientlaunch.cfg" ubicado en el directorio "/.../hobbit/client/etc".

El contenido detallo de estos scripts se encuentran en el anexo[7]

Creamos también los scripts que serán llamados en cada uno de los scripts de control.

```
[hobbit@xen ext]$ touch xmlist.sh
[hobbit@xen ext]$ touch infMV.sh
[hobbit@xen ext]$ touch xmtop.sh
[hobbit@xen ext]$ touch dom0.sh
[hobbit@xen ext]$ touch cos01.sh
[hobbit@xen ext]$ touch win01.sh
```

Para el servidor host y guest se implemento el script de control, *uptime*.

- ✓ **Para el host**

```
[hobbit@xen ext]$ touch ctl_UpTime.sh
```

- ✓ **Para el guest**

Para los siguientes scripts, su contenido se podrá visualizar en el anexo[8]

[7] Script de Control

Estos scripts deben tener permisos 755, y tiene se pertener al grupo hobbit.

```
[root@xen ext]# chown hobbit:hobbit ctl_xm_list.sh
[root@xen ext]# chown hobbit:hobbit xm_list.sh
[root@xen ext]# chmod 755 ctl_xm_list.sh
[root@xen ext]# chmod 755 xm_list.sh
```

Su correspondiente script que es llamado en el script de control, uptime:

```
[hobbit@xen ext]$ touch  UpTime
```

El archivo de configuración *clientlaunch.cfg* es cargado por el hobbitlaunch, controla los módulos que van a ejecutarse, la frecuencia con la que se ejecutan, parámetros, opciones y variables de entorno.

En el anexo[9] se visualiza con más detalle este archivo de configuración.

A continuación una descripción de lo scripts

Tabla IV.25. Scripts para el Monitoreo Detallado

Script de Control	Script de Ejecución	DESCRIPCIÓN
ctl_xmlist.sh	xmlist.sh	Script utilizado para listar las maquinas virtuales creadas en xen.
ctl_infMV.sh	sudo /bin/cat /etc/xen/cos01	Script que visualiza el contenido de los archivos de confirguración de los servidores virtuales.
ctl_xmtop.sh	xmtop.sh	Script que visualiza el consumo de recursos hardware del dominio 0 y guest.
ctl_dom0disp.sh	dom0.sh	Script para graficar el estado de funcionalidad del dominio 0.
ctl_cos01disp.sh	cos01.sh	Programa para graficar el estado de funcionalidad del servidor virtual cos01.
ctl_win01disp.sh	win01.sh	Programa para graficar el estado de funcionalidad del servidor virtual win01.
ctl_UpTime.sh	UpTime.sh	Programa utilizado para visualizar el tiempo de Operacion del Servidor Host.

[8] Script ejecutado en el Script de Control
[9] Archivo de control de módulos clientlaunch.cfg

4.2.2.4.5. Configuración de los archivos Xymon en el Host

- **Archivo de Configuración Master**

En el archivo de configuración Master para Xymon se puede definir muchas cosas:

- Añadir hosts al archivo para que sea monitoreado por Xymon.

- Definición de páginas, subpaginas, grupo para las páginas web del Xymon y distribuirlos en cada una de las páginas.

Por defecto, es necesario definir en este archivo de configuración el hostname del mismo servidor donde está configurado el Xymon para ser monitoreado.

```
[root@xen etc]# pwd
/home/hobbit/server/etc
[root@xen etc]# vi bb-hosts
```

El contenido de este archivo se encuentra en el anexo[10]

- Después de efectuado cualquier cambio en este archivo de configuración tenemos que reiniciar el servidor Apache y el servicio de hobbit server:

```
[root@xen server]# service httpd restart
Stopping httpd: [FAILED]
Starting httpd: [  OK  ]
```

- Y con el usuario hobbit, reiniciar el servicio de hobbit:

```
[root@xen server]# su - hobbit
[hobbit@xen ~]$ cd server
[hobbit@xen server]$ ./hobbit.sh restart
Hobbit stopped
Hobbit started
```

[10] Contenido archivo de configuración bb-hosts

- **Archivo de configuración hobbitserver.cfg**

- En el archivo de configuración hobbitserver.cfg, el parámetro *TEST2RRD* define el estado y mensajes de datos que se desea colectar en la data RRD.

```
[hobbit@xen etc]$ pwd
/home/hobbit/server/etc
[hobbit@xen etc]$ vi hobbitserver.cfg
```

- Después de modificar este archivo es importante reiniciar el hobbir server ejecutando el script *hobbit.sh*.

- Visualizar la generación del archivo *name.rrd* que se encuentra en el siguiente path /home/hobbit/data/rrd/xen.eis.espoch.

En el anexo[11] se visualiza un segmento del archivo de configuración utilizado en nuestro escenario.

- **Archivo de configuración hobbitgraph.cfg**

- Este archivo define como los gráficos RRD son definidos por el CGI hobbitgraph.

```
[hobbit@xen etc]$ pwd
/home/hobbit/server/etc
[hobbit@xen etc]$ vi hobbitgraph.cfg
```

En el anexo[12] se visuliza una segmento del achivo de configuración utilizado en nuestro escenario.

- Después de modificar este archivo es importante reiniciar el hobbir server ejecutando el script *hobbit.sh*.

[11] Archivo de configuración hobbitserver.cfg
[12] Archivo de configuración hobbitgraph.cfg

4.2.2.4.5. Funcionalidad Xymon

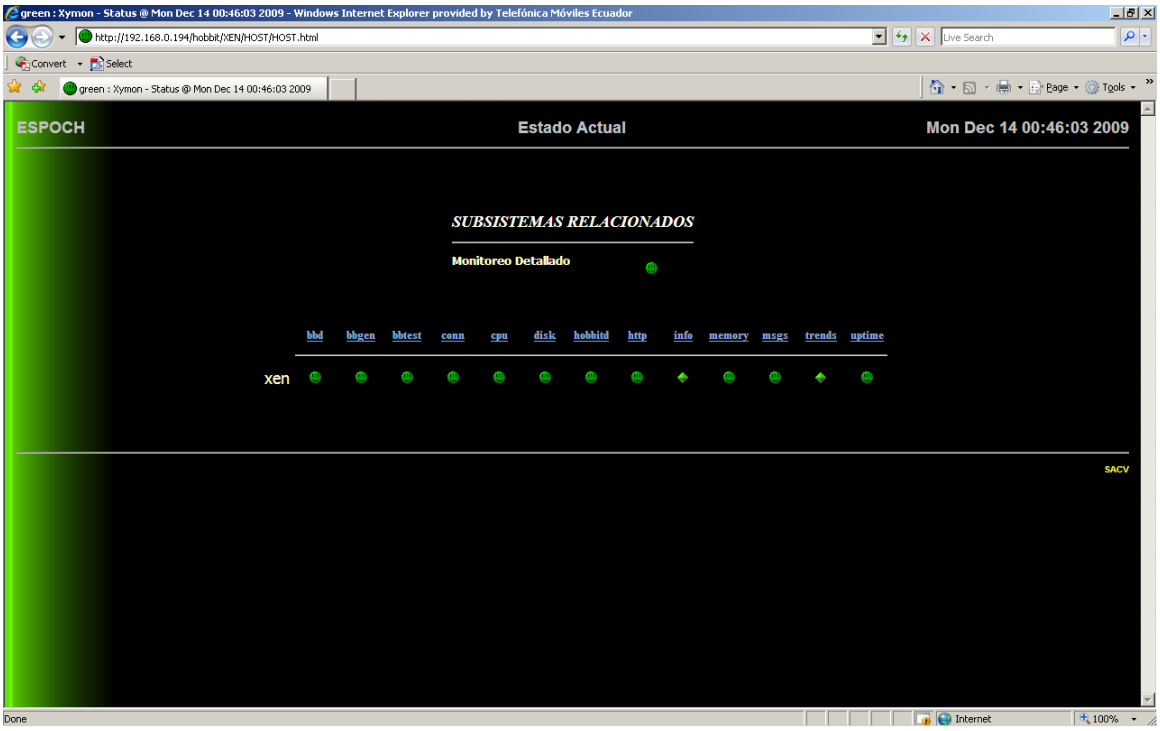

Figura IV.73. Xymon – Monitoreo Detallado

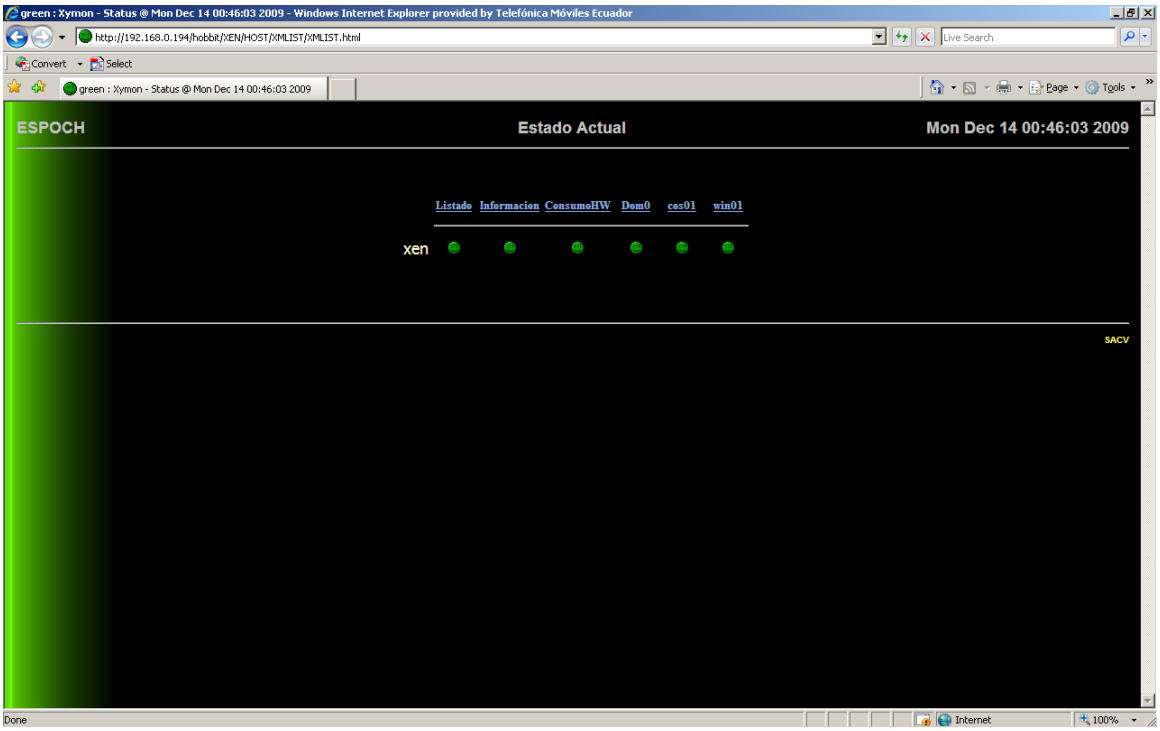

Figura IV.74. Xymon – Estado Actual Xen

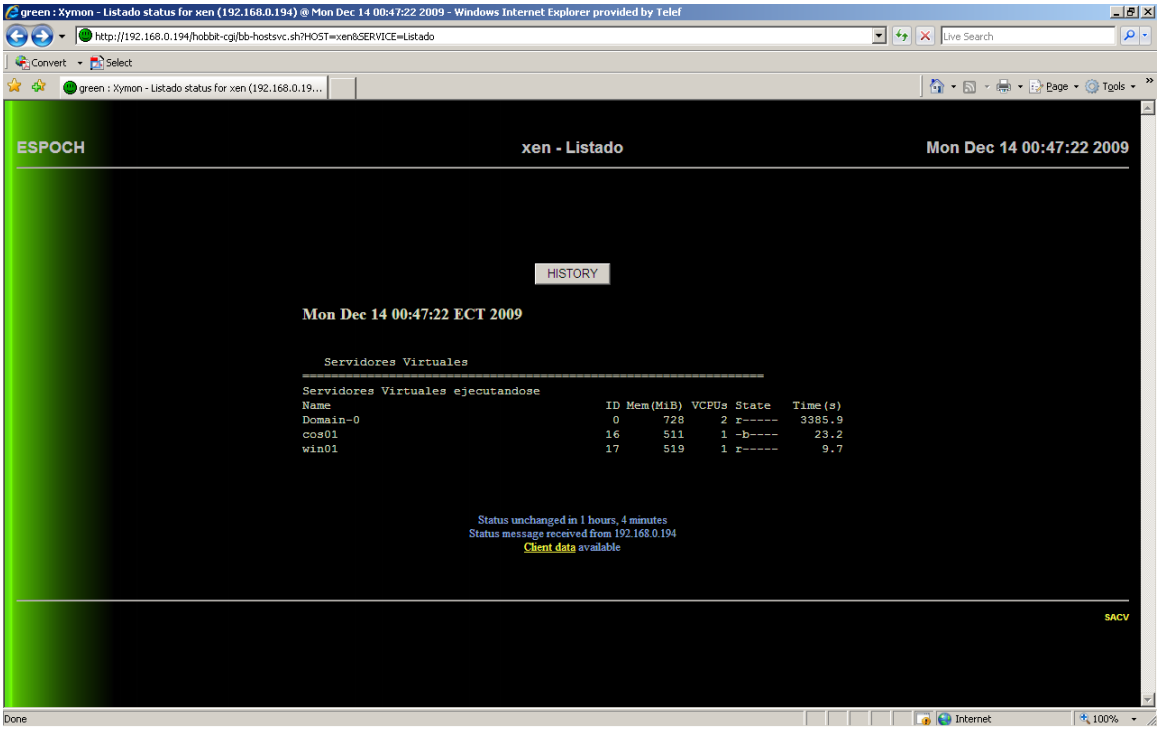

Figura IV.75. Xymon – Listado Servidores Virtuales Xen

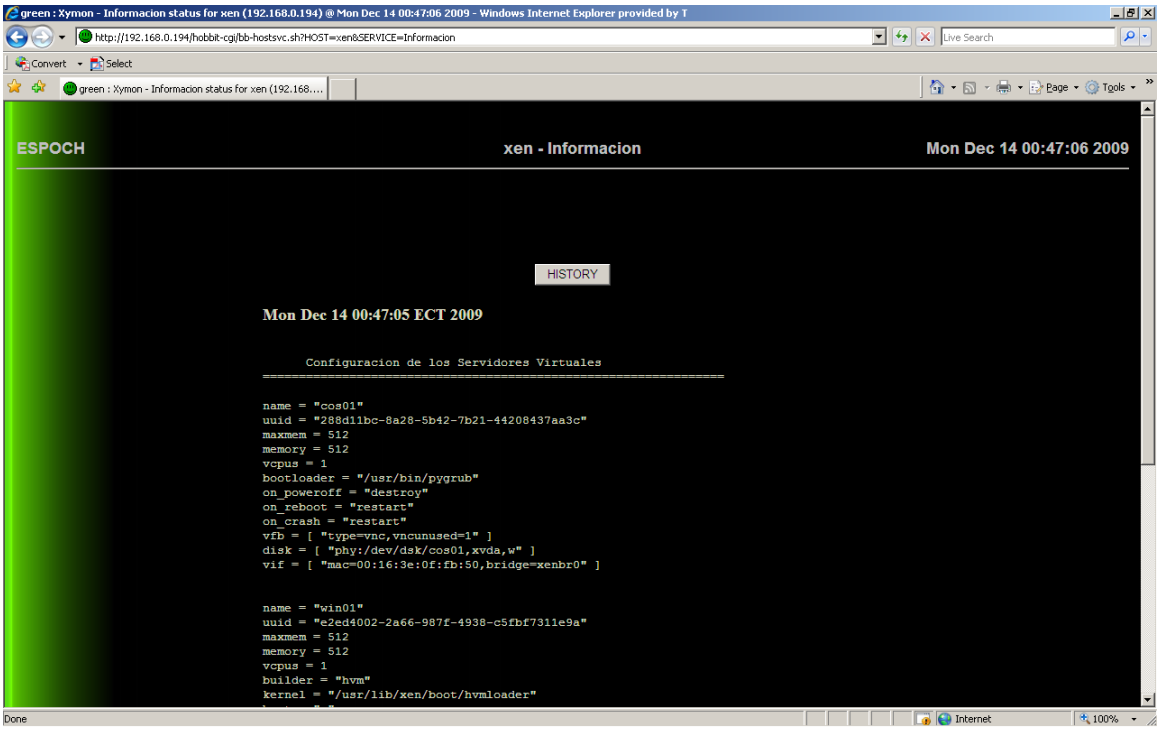

Figura IV.76. Xymon – Configuración Servidores Virtuales Xen

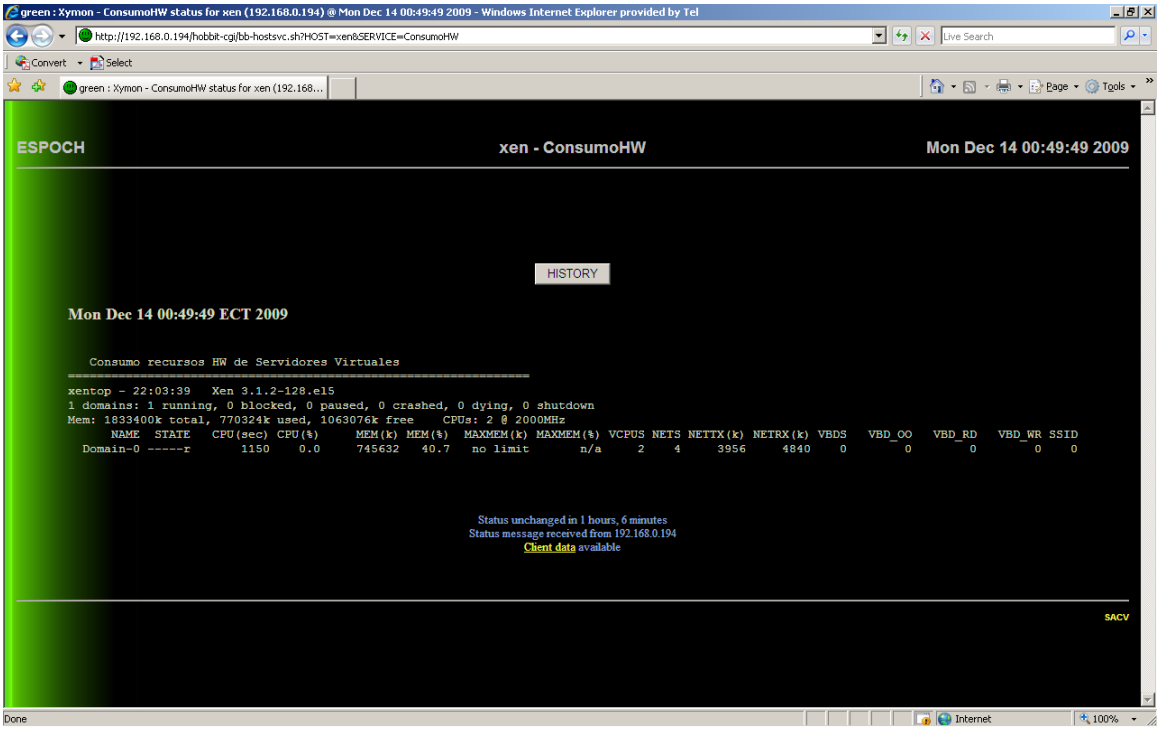

Figura IV.77. Xymon – Consumo Recursos Hardware Servidores Virtuales Xen

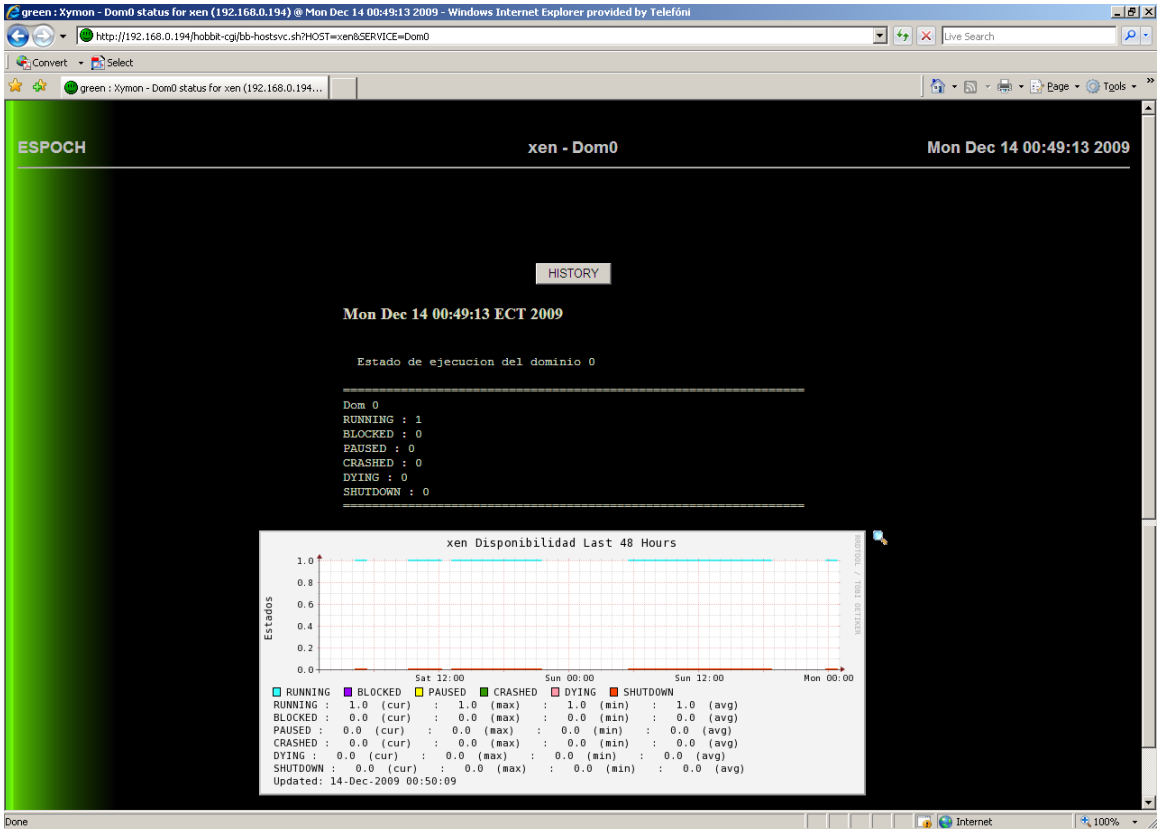

Figura IV.78. Xymon – Estado de Ejecución Xen

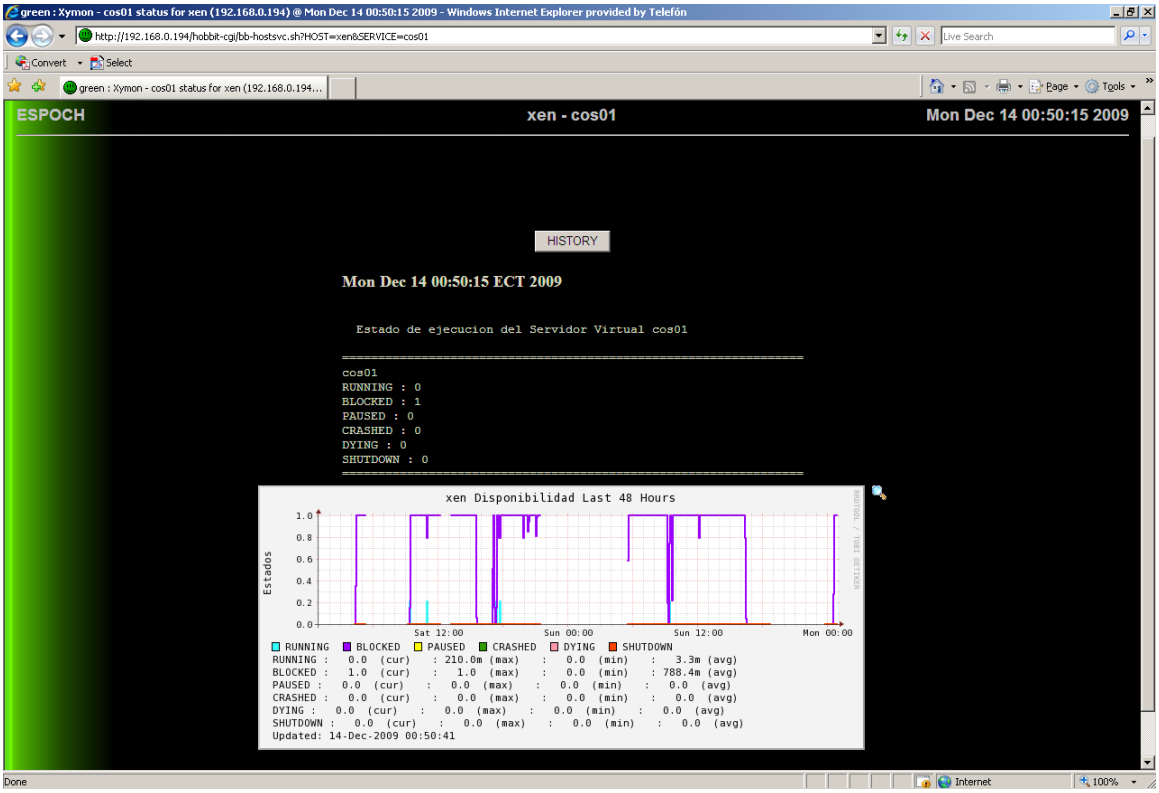

Figura IV.79. Xymon – Estado de Ejecución Servidor Virtual cos01

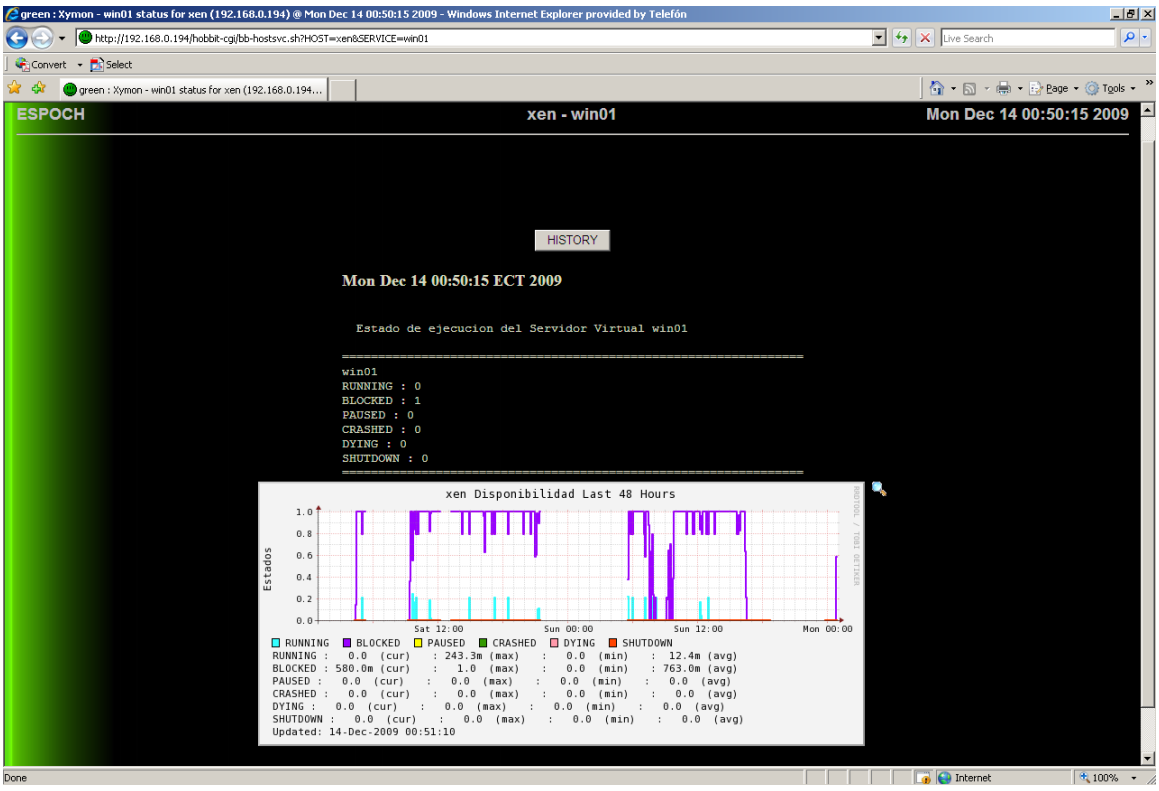

Figura IV.80. Xymon – Estado de Ejecución Servidor Virtual win01

CONCLUSIONES

- ✓ Cuando se va a elegir un software de virtualización de ordenadores, es indispensable que inicialmente se haga un estudio de las necesidades que tiene su organización.

- ✓ Se puede sustentar y concluir que Xen será el software de virtualización de ordenadores con el que se desarrollará una infraestructura de servidores virtuales en la EIS. Esta conclusión es respaldada en los resultados de las tablas donde se pondera los valores alcanzados del análisis comparativo de los parámetros así como de las conclusiones en cada una de ellas.

- ✓ Mediante software GPL podemos implementar software de virtualización de ordenadores potentes que utilicen distintas capas y plataformas; es decir que no presente limitantes considerables en su desarrollo.

- ✓ Mediante este tema de investigación, la EIS obtiene un software potente de virtualización de ordenadores, y se beneficia porque no hay costo de licencia por el software. Demostrando que las herramientas GPL compiten de igual a igual con software cuyas licencia son muy costosas.

- ✓ La idea de la virtualización es sencilla, nos permite utilizar más de un sistema operativo en un mismo ordenador, pero de forma simultánea y persistente. Los arranques múltiples permiten más de un sistema operativo pero no simultáneamente.

- ✓ Xen se destaca sobre otros software de virtualización de ordenadores por su performance, su simplicidad y su flexibilidad. Su principio de funcionamiento y su

puesta en marcha es simple, se adapta a ambientes corporativos y se combina perfectamente con otras herramientas como LVM (Logical Volume Manager) o ISCSI para ofrecer altas prestaciones. Hoy en día las computadoras modernas tienen una capacidad que nos es aprovechada y la virtualización de servidores nos permite mejorar el uso de recursos, ahorrando costos y mejorando las tareas de administración.

- ✓ La tecnología de virtualización aplicada en la EIS permitió optimizar el uso de los recursos hardware con los que cuenta logrando crear servidores virtuales utilizado para ambientes de producción, test y desarrollo.

- ✓ Con la ingeniería del software SACV, se pudo implementar un sistema donde se involucran algunos niveles de usuarios que interactuarán con el sistema sin importar su ubicación geográfica, brindándole seguridad en sus datos. Cada uno de ellos maneja interfaces de navegación, administración, servicios y monitoreo.

RECOMENDACIONES

- ✓ Cuando se vaya a elegir un software de virtualización de ordenadores, es importante realizar un estudio de la infraestructura hardware a nivel de servidores con que cuenta su organización; con el fin de definir parámetros determinantes en la elección del software de virtualización de ordenadores.

- ✓ Para la administración del software de virtualización de ordenadores es importante que la persona designada para su gestión adquiera conocimientos sobre virtualización, con eso se evitará el mal uso del software que afecte el buen funcionamiento de los servidores virtuales.

- ✓ La difusión de la virtualización, tiene que llegar a todos los entes que conforman la EIS, con el fin de que aprovechen este software para la creación de servidores virtuales y sea un soporte en los aspectos académicos, científicos, culturales, y economicos de la EIS.

- ✓ Para realizar pruebas, experimentos e implementaciones muy interesantes se recomienda trabajar con software de virtualización de ordenadores, en plataformas de distribución libre, ya que se basan en paquetes bien estables y ajustados por distros de linux.

- ✓ Es mejor implementar soluciones de virtualización en XEN que en VMWARE SERVER, puesto que con XEN puede implementarse PARAVIRTUALIZACIÓN o incluso VIRTUALIZACIÓN COMPLETA, mientras que con VMWARE SERVER y

similares como VIRTUALBOX solo puede disfrutarse de una Virtualización por Software, la cual tiene un rendimiento inadecuado para entornos de producción.

- ✓ La investigación adicional que se invierta en futuras versiones de software de virtualización de ordenadores estarán enmarcadas en el reto que se impuso al inicio; utilizar herramientas GPL.

- ✓ Utilizar software libre en la EIS servirá de un gran aporte académico para que los usuarios puedan hacer uso del mismo y compartir.

- ✓ Se recomienda tomar como base investigativa la información obtenida durante el desarrollo de la investigación para implementar otras aplicaciones con software GPL.

RESUMEN

Estudio comparativo de sistemas de virtualización de ordenadores para optimizar la infraestructura tecnológica de la Escuela de Ingeniería en Sistemas de la Escuela Superior Politécnica de Chimborazo (EIS-ESPOCH).

De acuerdo al método científico, se desarrolló la virtualización en un servidor de la EIS Intel core 2 duo E6400 con arquitectura de 32 bits, con sistemas VMware, Xen y VirtualBox, creando servidores virtuales en cada uno y, a través de módulos de prueba, observándose que el Xen alcanzó el porcentaje más alto (97.33%) en: instalación, escalabilidad, alta disponibilidad, flexibilidad, usabilidad, estabilidad, rendimiento, velocidad y extensibilidad; por ello fue seleccionado.

La infraestructura de servidores virtuales con Xen es gestionada y monitoreada mediante una aplicación Web, denominada “Sistema Administrador Central de Virtualización (SAVC)”.

La Virtualización de los servidores nos permitió a los usuarios realizar actividades de producción, investigación, obtención de respaldos, sin interferir una actividad con la otra. Al haber creado servidores virtuales y que funcionan como un servidor físico se ahorra un 50 % de: espacio físico, consumo de energía, personal de mantenimiento, y otros dispositivos que se requieren para los servidores físicos, y por ende se minimiza los costos económicos.

Con adquisición de software y hardware adecuado para la virtualizacion se puede extender su aplicación a: la facultad y la Espoch en general, y a otras organizaciones que trabajen con ordenadores físicos. Se recomienda realizar estudios sobre la implementación de alta disponibilidad y procesamiento distribuido mediante Xen en ambientes clusterizados.

SUMARY

This is a comparative study of virtualization systems of computers to optimize the technological infrastructure of the System Engineering School of the Chimborazo Higher Education Polytechnic School (EIS-ESPOCH).

According to the scientific method, the virtualization in a server of the EIS Intel core 2 duo E6400 was developed with an architecture of 32 bits, with VMware, Xen and VirtualBox systems, creating virtual servers in each and through test modules observing that Xen reached the highest percentage (97.33%) in: installation, scalability, high availability, flexibility, usability, stability, performance, velocity and extensibility; this is why it was selected.

The virtual server infrastructure with Xen is managed and monitered through a Web application called "Virtualization Central Managing System" (SACV).

The server virtualization permitted the users to carry out production, investigation activities, back up obtainment without interfering an activity with another. Creating virtual servers functioning as physical server saves 50% of physical space, energy consumption, personal maintenance and other devices required for the physical servers, thus minimizing the economic cost.

With the software and hardware acquisition adequate for virtualization it is possible to extend its application to the faculty and Espoch in general and other organizations working with physical computers. It is recommended to carry out studies on the implementation of high availability and processing distributed through Xen in clusterized environments.

GLOSARIO DE TÉRMINOS

Este glosario está destinado a definir los términos utilizados en este proyecto de tesis:

Bare-metal	El término bare-metal se refiere a la arquitectura física subyacente de un ordenador. Ejecución de un sistema operativo en el bare-metal es otra forma de referirse a la ejecución de una versión modificada del sistema operativo en el hardware físico.
dom0	También conocido como host o sistema operativo anfitrión. dom0 se refiere a la instancia host de Xen corriendo sobre el hipervisor el cual facilita la virtualización de sistemas operativos invitados.
Domains	domU y domains son ambos dominios. Dominios se ejecutan en el Hypervisor. Los términos dominios tienen un significado similar a máquinas Virtuales y los dos son técnicamente intercambiables.
domU	domU se refiere al sistema operativo invitado que se ejecutan en el sistema (Domains).
Full virtualization	Puede implementar virtualización de CentOs en una de dos opciones: virtualización completa o para-virtualización. La virtualización completa proporciona total abstracción del sistema físico subyacente (bare-metal) y crea un nuevo sistema virtual en que los sistemas operativos el invitados puede ejecutarse. No son necesarias modificaciones en el sistema operativo invitado. Los sistemas operativos invitados y las aplicaciones en los invitados no tienen conocimiento del ambiente de virtualización y se ejecutan normalmente. Paravirtualización requiere una versión modificada del operativo Linux sistema.
Hypervisor	El hipervisor es la capa de software que abstrae el hardware desde el sistema operativo permitiendo múltiples sistemas operativos funcionar en el mismo hardware.
Host	El sistema operativo host, también conocido como dom0.
Kernel-based Virtual Machine	KVM es un módulo del kernel de virtualización completa que se incorporará en futuras versiones de CentOs. KVM es en la actualidad disponible en la distribución de Linux Fedora y otras distribuciones de Linux.

Migration	La migración se refiere al proceso de traslado de un para-virtualizados clientes imágenes de un servidor de virtualización de Red Hat a otro. Este otro el servidor podría estar en el mismo servidor o un servidor diferente, incluyendo servidores en otras ubicaciones.
Para-virtualization	Para-virtualización utiliza un núcleo especial, a veces se denomina el núcleo xen kernel o kernel-xen xen para virtualizar otro entorno, mientras se utiliza las librerías de ordenadores y dispositivos. Una instalación de para-virtualizada tendrá acceso completo a todos los dispositivos en el sistema. Paravirtualización es significativamente más rápido que la virtualización completa puede ser efectivamente usaada para balanceo de carga, aprovisionamiento, seguridad y ventajas de consolidación.
Para-virtualized drivers	Drives paravirtualizados son los dispositivos de drives que operan en la virtualización completa de los Linux invitados. Estos drivers aumentan en gran medida el rendimiento de la red y de dispositivos de bloque de I/O para inviatdos completamente virtualizados.
Universally Unique Identifier	Un Identificador Único Universal (UUID) es un sistema estandarizado de numeración para dispositivos, sistemas y objetos de cierto tipo de software en entornos de computación distribuida. Tipos de UUID en la virtualización incluyen: ext2 y ext3 identificadores de sistema de archivos, RAID identificadores de dispositivo, iSCSI y LUN identificadores de dispositivo, las direcciones MAC y virtuales identificadores de la máquina.

BIBLIOGRAFÍA

BIBLIOGRAFIA GENERAL

1. AVE, H. Vmware Server User´s Guide. 2a. ed. Palo Alto, USA: Vmware, 2008. pp. 71-322

2. CURRAN, C. and HOLZER, J.M. Virtualization Guide. 3a. ed. Raleigh: Red Hat, 2009. pp. 3-253

3. RULE, D. and DITTNER, R. The Best Damn Server Virtualization Book Period. 2a. ed. Burlington: Syngress Publishing, 2001. pp. 421-462

4. SUN MICROSYSTEMS. Sun VirtualBox. 3a. ed. San Francisco, USA: Sun Microsystems, 2009. pp. 44-106

5. TAKEMURA, C. and CRAWFORD, L. THE BOOK OF XEN. 3a. ed. San Francisco, USA: Starch Press, 2010. pp. 2-198

6. WILLIAMS, D. and GARCIA, J. Virtualization with Xen. 3a ed. Burlington: Syngress Publishing, 2007. pp. 43-329

REFERENCIAS WEB

7. HIPERVISOR MONITOR DE MAQUINA VIRTUAL

http://es.wikipedia.org/wiki/Hipervisor#Enlaces_externos

2009/11/16

8. HOBBIT MONITOR: HOBBIT MONITOR PROYECT

http://hobbitmon.sourceforge.net/

2008/10/

9. INSTALLING VMWARE SERVER ON CENTOS 5 OR RED HAT ENTERPRISE LINUX 64 BIT VERSION

http://www.cyberciti.biz/tips/vmware-on-centos5-rhel5-64-bit-version.html

2009/08/10

10. LINUX VIRTUALIZATION WITH XEN

http://linuxdevcenter.com/pub/a/linux/2006/01/26/xen.html

2009/03/10

11. RENDIMIENTO EN LA VIRTUALIZACION CON XEN

http://www.saghul.net/blog/2009/01/11/rendimiento-en-la-virtualizacion-con-xen/

2009/01/11

12. VIRTUALBOX: VIRTUAL MACHINES

http://www.virtualbox.org/wiki/Virtualization

2009/12/17

13. VIRTUALIZACION: ¿QUE ES LA VIRTUALIZACION?

http://www.virtualizate.es/virtualizacion.html

2010/02/03

14. VIRTUALIZACION: VIRTUALIZACIÓN CON XEN

http://www.virtualizate.es/virtualizacion_xen.html

2009/03/16

15. VIRTUALIZACION: MAQUINAS VIRTUALES.

http://www.devjoker.com/asp/ver_contenidos.aspx?co_contenido=73

2008/02/16

16. VMWARE: TECHNICAL VIRTUALIZATION RESOURSES

http://www.vmware.com/

2010/01/26

17. XEN: ¿WHAT IS XEN?

http://www.xen.org/

2009/10/12

ANEXOS

Contenido de los archivos hobbit-server y hobbit-client

hobbit-server

```
[root@xen init.d]# cat hobbit-server
#!/bin/sh
# description: Servicio de Sevidor Hobbit
# creado por:  Maria Dolores Santos

case "$1" in
        start)
                echo -n "Starting Hobbit Server"
                su - hobbit -c "/home/hobbit/server/hobbit.sh start"
                ;;

        stop)
                echo -n "Halting Hobbit Server"
                su - hobbit -c "/home/hobbit/server/hobbit.sh stop"
                ;;

        reload)
                echo -n "Hobbit Server reload"
                su - hobbit -c "/home/hobbit/server/hobbit.sh reload"
                ;;
        restart)
                $0 stop
                /bin/sleep 1
                $0 start
                ;;
        status)
                echo -n "Checking..."
                su - hobbit -c "/home/hobbit/server/hobbit.sh status"
                ;;
        *)
                echo "Usage: $0 {start|stop|reload|restart|status}"
                exit 1
        ;;
esac
```

hobbit-client

```
[root@xen init.d]# cat hobbit-client
#!/bin/sh
# chkconfig: 345 40 60
# description: Servicio de Hobbit
# creado por Maria Dolores Santos

case "$1" in
        start)
                echo -n "Starting Hobbit Client"
                su – hobbit –c “/home/hobbit/client/runclient.sh start”
                ;;

        stop)
                echo -n "Halting Hobbit Client"
                su –hobbit –c “/home/hobbit/client/runclient.sh stop”
                ;;

        reload)
                echo -n "Hobbit Client reload"
                su – hobbit –c “/home/hobbit/client/runclient.sh reload”
                ;;
        restart)
                $0 stop
                /bin/sleep 1
                $0 start
                ;;
       status)
                echo -n "Checking..."
                su – hobbit –c “/home/hobbit/client/runclient.sh status”
                ;;
        *)
                echo "Usage: $0 {start|stop|reload|restart|status}"
                exit 1
```

```
        ;;
esac
```

Contenido de los scripts de control

Script ctl_xm_list.sh

```
[hobbit@xen ext]$ cat ctl_xmlist.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               10-Nov-2009
# Nombre del script:    ctl_xm_list.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#############################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
#############################################################
# DESCRIPCION:
# ===========
# Programa utilizado para listar los servidores
# virtuales creadas en xen
#############################################################
WRK=/home/hobbit/client/ext
COLUMN=Listado            # Name of the column
COLOR=green               # By default, everything is OK
MSG="Listado de Guests"
echo "
   Servidores Virtuales
=================================================================
Servidores Virtuales ejecutandose
" > $WRK/ctl_xmlist.head
$WRK/xmlist.sh > $WRK/xmlist.out
$BB $BBDISP "status $MACHINE.$COLUMN green `date`
`cat $WRK/ctl_xmlist.head`
`cat $WRK/xmlist.out`"
exit 0
```

Script ctl_infMV.sh

```
[hobbit@xen ext]$ cat ctl_infMV.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               11-Nov-2009
# Nombre del script:    ctl_infMV.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#############################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
#############################################################
# DESCRIPCION:
# ===========
# Programa que muestra la configuracion de los servidores
# virtuales
#############################################################
WRK=/home/hobbit/client/ext
COLUMN=Informacion            # Name of the column
COLOR=green               # By default, everything is OK
MSG="Informacion"
echo "
     Configuracion de los Servidores Virtuales
=================================================================
```

```
" > $WRK/ctl_infMV.head
$BB $BBDISP "status $MACHINE.$COLUMN green `date`
`cat $WRK/ctl_infMV.head`

`sudo /bin/cat /etc/xen/cos01`


`sudo /bin/cat /etc/xen/win01`"
exit 0
```

Script ctl_xmtop.sh

```
[hobbit@xen ext]$ cat ctl_xmtop.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               11-Nov-2009
# Nombre del script:    ctl_xmtop.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#########################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
#########################################################
# DESCRIPCION:
# ===========
# Programa utilizado para visualizar el consumo de
# recursos HW de los servidores virtuales
#########################################################
WRK=/home/hobbit/client/ext
COLUMN=ConsumoHW          # Name of the column
COLOR=green               # By default, everything is OK
MSG="Consumo de Recursos HW"
echo "
   Consumo recursos HW de Servidores Virtuales
================================================================

" > $WRK/ctl_xmtop.head
$WRK/xmtop.sh
$BB $BBDISP "status $MACHINE.$COLUMN green `date`
`cat $WRK/ctl_xmtop.head`
`cat $WRK/xmtop.out`"
exit 0
```

Script ctl_dom0disp.sh

```
[hobbit@xen ext]$ cat ctl_dom0disp.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               12-Nov-2009
# Nombre del script:    ctl_disp.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#########################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
#########################################################
# DESCRIPCION:
# ===========
# Programa para graficar el estado de funcionalidad
# del dominio 0
#########################################################
WRK=/home/hobbit/client/ext
```

```
COLUMN=Dom0             # Name of the column
COLOR=green             # By default, everything is OK
MSG="Funcionalidad dominios"
echo "
  Estado de ejecucion del dominio 0

=================================================================
Dom 0
" > $WRK/dom0.head
$WRK/dom0.sh > $WRK/dom0.out

$BB $BBDISP "status $MACHINE.$COLUMN $COLOR `date`
`cat $WRK/dom0.head`
`cat $WRK/dom0.out`"
exit 0
```

Script ctl_cos01disp.sh

```
[hobbit@xen ext]$ cat ctl_cos01disp.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               29-Nov-2009
# Nombre del script:    ctl_cos01disp.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#############################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
#############################################################
# DESCRIPCION:
# ============
# Programa para graficar el estado de funcionalidad
# del servidor virtual cos01
#############################################################
WRK=/home/hobbit/client/ext
COLUMN=cos01            # Name of the column
COLOR=green             # By default, everything is OK
MSG="Funcionalidad dominios"
echo "
  Estado de ejecucion del Servidor Virtual cos01

=================================================================
cos01
" > $WRK/cos01.head
$WRK/cos01.sh > $WRK/cos01.out

$BB $BBDISP "status $MACHINE.$COLUMN $COLOR `date`
`cat $WRK/cos01.head`
`cat $WRK/cos01.out`"
exit 0
```

Script ctl_win01disp.sh

```
[hobbit@xen ext]$ cat ctl_win01disp.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               29-Nov-2009
# Nombre del script:    ctl_win01disp.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#############################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
```

```
#########################################################
# DESCRIPCION:
# ===========
# Programa para graficar el estado de funcionalidad
# del servidores virtual win01
#########################################################
WRK=/home/hobbit/client/ext
COLUMN=win01            # Name of the column
COLOR=green             # By default, everything is OK
MSG="Funcionalidad dominios"
echo "
  Estado de ejecucion del Servidor Virtual win01

================================================================
win01
" > $WRK/win01.head
$WRK/win01.sh > $WRK/win01.out

$BB $BBDISP "status $MACHINE.$COLUMN $COLOR `date`
`cat $WRK/win01.head`
`cat $WRK/win01.out`"
rm $WRK/win01.out
exit 0
```

Script ctl_UpTime.sh

```
[hobbit@xen ext]$ cat ctl_UpTime.sh
#!/bin/sh
############ DESCRIPCION GENERAL DE FUNCIONAMIENTO ######
# Autor:               Maria Dolores Santos
# Fecha:               11-Nov-2009
# Nombre del script:    ctl_UpTime.sh
# Path:                /export/hobbit/client/ext
# Scripts relacionados:
# Version:             1
# Plataforma:          XEN
# Plataforma gestion:   Hobbit (xen)
#########################################################
# USO:
# ====
# Este programa se ejecuta automaticamente segun la de-
# finicion en el archivo "clientlaunch.cfg" ubicado en
# el directorio "/.../hobbit/client/etc".
#########################################################
# DESCRIPCION:
# ===========
# Programa utilizado para visualizar el tiempo de Operacion
# del Servidor Host
#########################################################
WRK=/home/hobbit/client/ext
COLUMN=UpTime            # Name of the column
COLOR=green             # By default, everything is OK
MSG="Tiempo de Operacion"
echo "
   Tiempo de Operacion del Host
================================================================

" > $WRK/ctl_UpTime.head
$WRK/UpTime.sh > $WRK/UpTime.out
$BB $BBDISP "status $MACHINE.$COLUMN green `date`
`cat $WRK/ctl_UpTime.head`
`cat $WRK/UpTime.out`"
exit 0
```

Contenido de los scripts a ejecutarse en el script de control
Script xm_list.sh

```
[hobbit@xen ext]$ cat  xmlist.sh
sudo /usr/sbin/xm list
```

Script infMV.sh

```
[hobbit@xen ext]$ cat infMV.sh
```

```
sudo /bin/cat /etc/xen/cos01
echo
sudo /bin/cat /etc/xen/win01
```

Script xmtop.sh

```
[hobbit@xen ext]$ cat  xmtop.sh
sudo /usr/sbin/xentop -i 1 -b > xmtop.out
```

Script dom0

```
[hobbit@xen ext]$ cat dom0.sh
sudo /usr/sbin/xm list | grep Domain | awk '{ print $5}'  > dom0graph.out

dom=`wc -l dom0graph.out | awk '{ print $1}'`
if [ $dom = 1 ]; then
        i=1
        x=`cut -b $i dom0graph.out`
        while [ $x = "-" ]
        do
        i=$((i+1))
        x=`cut -b $i dom0graph.out`
        done
else
        i=0
fi


case $i in
        1 ) echo "RUNNING : 1"
                        echo "BLOCKED : 0"
                        echo "PAUSED : 0"
                        echo "CRASHED : 0"
                        echo "DYING : 0"
                        echo "SHUTDOWN : 0"
                        echo  "================================================================="
                        ;;

        2 ) echo "RUNNING : 0"
                        echo "BLOCKED : 1"
                        echo "PAUSED : 0"
                        echo "CRASHED : 0"
                        echo "DYING : 0"
                        echo "SHUTDOWN : 0"
                        echo  "================================================================="
                        ;;

        3 ) echo "RUNNING : 0"
                        echo "BLOCKED : 0"
                        echo "PAUSED : 1"
                        echo "CRASHED : 0"
                        echo "DYING : 0"
                        echo "SHUTDOWN : 0"
                        echo "================================================================="
                        ;;

        4 ) echo "RUNNING : 0"
                        echo "BLOCKED : 0"
                        echo "PAUSED : 0"
                        echo "CRASHED : 1"
                        echo "DYING : 0"
                        echo "SHUTDOWN : 0"
                        echo "================================================================="
                        ;;

        5 ) echo "RUNNING : 0"
                        echo "BLOCKED : 0"
                        echo "PAUSED : 0"
                        echo "CRASHED 0: "
                        echo "DYING : 1"
                        echo "SHUTDOWN : 0"
                        echo  "================================================================="
                        ;;

        6 ) echo "RUNNING : 0"
                        echo "BLOCKED : 0"
                        echo "PAUSED : 0"
                        echo "CRASHED 0: "
```

```
                echo "DYING : 0"
                echo "SHUTDOWN : 1"
                echo  "================================================================"
                ;;

        * ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
                ;;


esac
```

Script cos01.sh

```
[hobbit@xen ext]$ cat cos01.sh
sudo /usr/sbin/xm list | grep cos01 | awk '{ print $5}'  > cos01graph.out

dom=`wc -l  cos01graph.out | awk '{ print $1}'`
if [ $dom = 1 ]; then
        i=1
        x=`cut -b $i cos01graph.out`
        while [ $x = "-" ]
        do
        i=$((i+1))
        x=`cut -b $i cos01graph.out`
        done
else
        i=0
fi


case $i in
        1 ) echo "RUNNING : 1"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED : 0"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
                ;;

        2 ) echo "RUNNING : 0"
                echo "BLOCKED : 1"
                echo "PAUSED : 0"
                echo "CRASHED : 0"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
                ;;

        3 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 1"
                echo "CRASHED : 0"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo "================================================================"
                ;;

        4 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED : 1"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo "================================================================"
                ;;

        5 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 1"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
```

```
                ;;

        6 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 0"
                echo "SHUTDOWN : 1"
                echo  "================================================================"
                ;;

        * ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
                ;;


esac
```

Script win01.sh

```
[hobbit@xen ext]$ cat win01.sh
sudo /usr/sbin/xm list | grep win01 | awk '{ print $5}'  > win01graph.out

dom=`wc -l  win01graph.out | awk '{ print $1}'`
if [ $dom = 1 ]; then
        i=1
        x=`cut -b $i win01graph.out`
        while [ $x = "-" ]
        do
        i=$((i+1))
        x=`cut -b $i win01graph.out`
        done
else
        i=0
fi


case $i in
        1 ) echo "RUNNING : 1"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED : 0"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
                ;;

        2 ) echo "RUNNING : 0"
                echo "BLOCKED : 1"
                echo "PAUSED : 0"
                echo "CRASHED : 0"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "================================================================"
                ;;

        3 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 1"
                echo "CRASHED : 0"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo "================================================================"
                ;;

        4 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED : 1"
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo "================================================================"
                ;;

        5 ) echo "RUNNING : 0"
```

```
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 1"
                echo "SHUTDOWN : 0"
                echo  "============================================================="
                ;;
        6 ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 0"
                echo "SHUTDOWN : 1"
                echo  "============================================================="
                ;;

        * ) echo "RUNNING : 0"
                echo "BLOCKED : 0"
                echo "PAUSED : 0"
                echo "CRASHED 0: "
                echo "DYING : 0"
                echo "SHUTDOWN : 0"
                echo  "============================================================="
                ;;


esac
```

Script UpTime.sh

```
[hobbit@xen ext]$ cat UpTime.sh
uptime | awk '{ print $3}'| sed 's/,//g'
```

Archivo de control de módulos clientlaunch.cfg

```
[hobbit@xen etc]$ cat clientlaunch.cfg
#
# The clientlaunch.cfg file is loaded by "hobbitlaunch".
# It controls which of the Hobbit client-side modules to run,
# (both the main client "hobbitclient.sh" and any client-side
# extensions); how often, and with which parameters, options
# and environment variables.
#
# Note: On the Hobbit *server* itself, this file is normally
#       NOT used. Instead, both the client- and server-tasks
#       are controlled by the hobbitlaunch.cfg file.
#

# msgcache is used for passive clients, that cannot connect
# directly to the Hobbit server. This is not the default
# setup, so this task is normally disabled.
[msgcache]
        DISABLED
        ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
        CMD $HOBBITCLIENTHOME/bin/msgcache --no-daemon --pidfile=$HOBBITCLIENTHOME/logs/msgcache.pid
        LOGFILE $HOBBITCLIENTHOME/logs/msgcache.log

# The main client task
[client]
        ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
        CMD $HOBBITCLIENTHOME/bin/hobbitclient.sh
        LOGFILE $HOBBITCLIENTHOME/logs/hobbitclient.log
        INTERVAL 5m

# ORCA data collector. This is an experimental add-on module,
# the data sent by this module are not processed by Hobbit 4.2.
[orcadata]
        DISABLED
        ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
        CMD $HOBBITCLIENTHOME/bin/orcahobbit --orca=/usr/local/orca/orcallator
        LOGFILE $HOBBITCLIENTHOME/logs/hobbitclient.log
        INTERVAL 5m

[xmlist]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_xmlist.sh
INTERVAL 10
```

```
[xmtop]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_xmtop.sh
INTERVAL 10

[informacionMV]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_infMV.sh
INTERVAL 1m

[UpTime]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_UpTime.sh
INTERVAL 1m

[Dom0disp]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_dom0disp.sh
INTERVAL 1m

[cos01disp]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_cos01disp.sh
INTERVAL 1m

[win01disp]
ENVFILE $HOBBITCLIENTHOME/etc/hobbitclient.cfg
CMD $HOBBITCLIENTHOME/ext/ctl_win01disp.sh
INTERVAL 1m
```

Archivo de Configuración Master bb-hosts

```
[hobbit@xen etc]$ cat bb-hosts
#
# Master configuration file for Xymon
#
# This file defines several things:
#
# 1) By adding hosts to this file, you define hosts that are monitored by Xymon
# 2) By adding "page", "subpage", "group" definitions, you define the layout
#    of the Xymon webpages, and how hosts are divided among the various webpages
#    that Xymon generates.
# 3) Several other definitions can be done for each host, see the bb-hosts(5)
#    man-page.
#
# You need to define at least the Xymon server itself here.

page XEN <H5>Infraestructura XEN</H5>

      subpage HOST <H5>Servidor HOST</H5>

            group-except files|ports|procs|ConsumoHW|Listado|Informacion|xmlist|Disponibilidad|Estado|dispon|xmtop|disp|Dom0|cos01|win01
            172.30.60.39   xen.eis.espoch      # bbd http://xen/

            subparent HOST XMLIST <H5>Monitoreo Detallado</H5>
                  group-compress Ctrl_XMLIST
                  group-only Listado|Informacion|ConsumoHW|Dom0|cos01|win01
                  172.30.60.39 xen.eis.espoch      # bbd http://xen.eis.espoch/

      subpage GUESTS <H5>Servidores GUEST</H5>

            subparent GUESTS SERVLNXS  <H5>Servidores Linux</H5>
                  group-except files|ports|procs
                  172.30.60.40 cos01.eis.espoch

            subparent GUESTS SERVWINDOWS <H5>Servidores Windows</H5>
                  group-except files|ports|procs|svcs
                  172.30.60.45 WIN01_EIS_ESPOC
```

Archivo de configuración hobbitserver.cfg

```
TEST2RRD="cpu=la,disk,inode,qtree,memory,$PINGCOLUMN=tcp,http=tcp,dns=tcp,dig=tcp,time=ntpstat,vmstat,iostat,netstat,temperature,a
pache,bind,sendmail,mailq,nmailq=mailq,socks,bea,iishealth,citrix,bbgen,bbtest,bbproxy,hobbitd,files,procs=processes,ports,clock,lines,ops,s
```

```
tats,cifs,JVM,JMS,HitCache,Session,JDBCConn,ExecQueue,JTA,TblSpace,RollBack,MemReq,InvObj,snapmirr,snaplist,snapshot,if_load=de
vmon,temp=devmon,Dom0=ncv,cos01=ncv,win01=ncv"

NCV_Dom0="RUNNING:GAUGE,BLOCKED:GAUGE,PAUSED:GAUGE,CRASHED:GAUGE,DYING:GAUGE,SHUTDOWN:GAUGE"
NCV_cos01="RUNNING:GAUGE,BLOCKED:GAUGE,PAUSED:GAUGE,CRASHED:GAUGE,DYING:GAUGE,SHUTDOWN:GAUGE"
NCV_win01="RUNNING:GAUGE,BLOCKED:GAUGE,PAUSED:GAUGE,CRASHED:GAUGE,DYING:GAUGE,SHUTDOWN:GAUGE"
```

Archivo de configuración hobbitgraph.cfg

```
[Dom0]
	TITLE Disponibilidad
	YAXIS Estados
	DEF:RUNNING=Dom0.rrd:RUNNING:AVERAGE
	DEF:BLOCKED=Dom0.rrd:BLOCKED:AVERAGE
	DEF:PAUSED=Dom0.rrd:PAUSED:AVERAGE
	DEF:CRASHED=Dom0.rrd:CRASHED:AVERAGE
	DEF:DYING=Dom0.rrd:DYING:AVERAGE
	DEF:SHUTDOWN=Dom0.rrd:SHUTDOWN:AVERAGE
	LINE2:RUNNING#33FFFF:RUNNING
	LINE2:BLOCKED#9900FF:BLOCKED
	LINE2:PAUSED#FFFF00:PAUSED
	LINE2:CRASHED#339900:CRASHED
	LINE2:DYING#FF99AA:DYING
	LINE2:SHUTDOWN#FF4400:SHUTDOWN
	COMMENT:\n
	GPRINT:RUNNING:LAST:RUNNING \: %5.1lf%s (cur)
	GPRINT:RUNNING:MAX: \: %5.1lf%s (max)
	GPRINT:RUNNING:MIN: \: %5.1lf%s (min)
	GPRINT:RUNNING:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:BLOCKED:LAST:BLOCKED \: %5.1lf%s (cur)
	GPRINT:BLOCKED:MAX: \: %5.1lf%s (max)
	GPRINT:BLOCKED:MIN: \: %5.1lf%s (min)
	GPRINT:BLOCKED:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:PAUSED:LAST:PAUSED \: %5.1lf%s (cur)
	GPRINT:PAUSED:MAX: \: %5.1lf%s (max)
	GPRINT:PAUSED:MIN: \: %5.1lf%s (min)
	GPRINT:PAUSED:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:CRASHED:LAST:CRASHED \: %5.1lf%s (cur)
	GPRINT:CRASHED:MAX: \: %5.1lf%s (max)
	GPRINT:CRASHED:MIN: \: %5.1lf%s (min)
	GPRINT:CRASHED:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:DYING:LAST:DYING \: %5.1lf%s (cur)
	GPRINT:DYING:MAX: \: %5.1lf%s (max)
	GPRINT:DYING:MIN: \: %5.1lf%s (min)
	GPRINT:DYING:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:SHUTDOWN:LAST:SHUTDOWN \: %5.1lf%s (cur)
	GPRINT:SHUTDOWN:MAX: \: %5.1lf%s (max)
	GPRINT:SHUTDOWN:MIN: \: %5.1lf%s (min)
	GPRINT:SHUTDOWN:AVERAGE: \: %5.1lf%s (avg)\n

[cos01]
	TITLE Disponibilidad
	YAXIS Estados
	DEF:RUNNING=cos01.rrd:RUNNING:AVERAGE
	DEF:BLOCKED=cos01.rrd:BLOCKED:AVERAGE
	DEF:PAUSED=cos01.rrd:PAUSED:AVERAGE
	DEF:CRASHED=cos01.rrd:CRASHED:AVERAGE
	DEF:DYING=cos01.rrd:DYING:AVERAGE
	DEF:SHUTDOWN=cos01.rrd:SHUTDOWN:AVERAGE
	LINE2:RUNNING#33FFFF:RUNNING
	LINE2:BLOCKED#9900FF:BLOCKED
	LINE2:PAUSED#FFFF00:PAUSED
	LINE2:CRASHED#339900:CRASHED
	LINE2:DYING#FF99AA:DYING
	LINE2:SHUTDOWN#FF4400:SHUTDOWN
	COMMENT:\n
	GPRINT:RUNNING:LAST:RUNNING \: %5.1lf%s (cur)
	GPRINT:RUNNING:MAX: \: %5.1lf%s (max)
	GPRINT:RUNNING:MIN: \: %5.1lf%s (min)
	GPRINT:RUNNING:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:BLOCKED:LAST:BLOCKED \: %5.1lf%s (cur)
	GPRINT:BLOCKED:MAX: \: %5.1lf%s (max)
	GPRINT:BLOCKED:MIN: \: %5.1lf%s (min)
	GPRINT:BLOCKED:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:PAUSED:LAST:PAUSED \: %5.1lf%s (cur)
	GPRINT:PAUSED:MAX: \: %5.1lf%s (max)
	GPRINT:PAUSED:MIN: \: %5.1lf%s (min)
	GPRINT:PAUSED:AVERAGE: \: %5.1lf%s (avg)\n
	GPRINT:CRASHED:LAST:CRASHED \: %5.1lf%s (cur)
```

```
      GPRINT:CRASHED:MAX: \: %5.1lf%s (max)
      GPRINT:CRASHED:MIN: \: %5.1lf%s (min)
      GPRINT:CRASHED:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:DYING:LAST:DYING \: %5.1lf%s (cur)
      GPRINT:DYING:MAX: \: %5.1lf%s (max)
      GPRINT:DYING:MIN: \: %5.1lf%s (min)
      GPRINT:DYING:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:SHUTDOWN:LAST:SHUTDOWN \: %5.1lf%s (cur)
      GPRINT:SHUTDOWN:MAX: \: %5.1lf%s (max)
      GPRINT:SHUTDOWN:MIN: \: %5.1lf%s (min)
      GPRINT:SHUTDOWN:AVERAGE: \: %5.1lf%s (avg)\n

[win01]
      TITLE Disponibilidad
      YAXIS Estados
      DEF:RUNNING=win01.rrd:RUNNING:AVERAGE
      DEF:BLOCKED=win01.rrd:BLOCKED:AVERAGE
      DEF:PAUSED=win01.rrd:PAUSED:AVERAGE
      DEF:CRASHED=win01.rrd:CRASHED:AVERAGE
      DEF:DYING=win01.rrd:DYING:AVERAGE
      DEF:SHUTDOWN=win01.rrd:SHUTDOWN:AVERAGE
      LINE2:RUNNING#33FFFF:RUNNING
      LINE2:BLOCKED#9900FF:BLOCKED
      LINE2:PAUSED#FFFF00:PAUSED
      LINE2:CRASHED#339900:CRASHED
      LINE2:DYING#FF99AA:DYING
      LINE2:SHUTDOWN#FF4400:SHUTDOWN
      COMMENT:\n
      GPRINT:RUNNING:LAST:RUNNING \: %5.1lf%s (cur)
      GPRINT:RUNNING:MAX: \: %5.1lf%s (max)
      GPRINT:RUNNING:MIN: \: %5.1lf%s (min)
      GPRINT:RUNNING:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:BLOCKED:LAST:BLOCKED \: %5.1lf%s (cur)
      GPRINT:BLOCKED:MAX: \: %5.1lf%s (max)
      GPRINT:BLOCKED:MIN: \: %5.1lf%s (min)
      GPRINT:BLOCKED:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:PAUSED:LAST:PAUSED \: %5.1lf%s (cur)
      GPRINT:PAUSED:MAX: \: %5.1lf%s (max)
      GPRINT:PAUSED:MIN: \: %5.1lf%s (min)
      GPRINT:PAUSED:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:CRASHED:LAST:CRASHED \: %5.1lf%s (cur)
      GPRINT:CRASHED:MAX: \: %5.1lf%s (max)
      GPRINT:CRASHED:MIN: \: %5.1lf%s (min)
      GPRINT:CRASHED:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:DYING:LAST:DYING \: %5.1lf%s (cur)
      GPRINT:DYING:MAX: \: %5.1lf%s (max)
      GPRINT:DYING:MIN: \: %5.1lf%s (min)
      GPRINT:DYING:AVERAGE: \: %5.1lf%s (avg)\n
      GPRINT:SHUTDOWN:LAST:SHUTDOWN \: %5.1lf%s (cur)
      GPRINT:SHUTDOWN:MAX: \: %5.1lf%s (max)
      GPRINT:SHUTDOWN:MIN: \: %5.1lf%s (min)
      GPRINT:SHUTDOWN:AVERAGE: \: %5.1lf%s (avg)\n
```