



Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Virtualización con Xen en Debian Etch con kernel a medida para 32 y 64 bits

(5164 lectures)

Per **Jaume Sabater**, [Primetime](http://www.linuxsilo.net/) (<http://www.linuxsilo.net/>)

Creat el 18/12/2006 09:16 modificat el 18/12/2006 09:16

Los ordenadores actuales tienen las suficientes prestaciones como para usar la virtualización con el fin de crear la representación de muchas máquinas virtuales menos capaces, cada una ejecutando una instancia independiente de un sistema operativo. El adecuado particionamiento de una máquina para que soporte la ejecución concurrente de múltiples sistemas operativos supone bastantes retos. Primero, las máquinas virtuales deben estar aisladas las unas de las otras: no es aceptable que la ejecución de una afecte adversamente el rendimiento de otra. Esto es particularmente cierto cuando las diferentes máquinas virtuales pertenecen a diferentes usuarios. Segundo, es preciso dar soporte al máximo número de sistemas operativos existentes a fin de dar salida a la heterogeneidad de las aplicaciones más comunes. Tercero, la sobrecarga introducida por la virtualización debe ser pequeña.

Índice

1. [Introducción](#)
2. [Descripción, historia y características de Xen Hypervisor](#)
3. [Estructura de un sistema basado en Xen](#)
4. [Estado actual de Xen](#)
5. [Instalación de Xen](#)
6. [Interfaces de red virtuales en un sistema Xen](#)
 1. [Puentes de red en un sistema Xen](#)
 2. [Flujo de paquetes en el puente](#)
 3. [El script `network-bridge`](#)
 4. [El script `vif-bridge`](#)
7. [Bibliografía](#)
8. [Historial de revisiones](#)

Introducción

Los ordenadores actuales tienen las suficientes prestaciones como para usar la virtualización con el fin de crear la representación de muchas máquinas virtuales menos capaces, cada una ejecutando una instancia independiente de un sistema operativo. El adecuado particionamiento de una máquina para que soporte la ejecución concurrente de múltiples sistemas operativos supone bastantes retos. Primero, las máquinas virtuales deben estar aisladas las unas de las otras: no es aceptable que la ejecución de una afecte adversamente el rendimiento de otra. Esto es particularmente cierto cuando las diferentes máquinas virtuales pertenecen a diferentes usuarios. Segundo, es preciso dar soporte al máximo número de sistemas operativos existentes a fin de dar salida a la heterogeneidad de las aplicaciones más comunes. Tercero, la sobrecarga introducida por la virtualización debe ser pequeña.

Descripción, historia y características de Xen Hypervisor

[Xen](#)⁽¹⁾ es un hipervisor (del inglés, *más que un supervisor*) de código abierto que permite una mejor utilización de los servidores y la consolidación de los mismos al posibilitar que múltiples imágenes de sistemas operativos se ejecuten simultáneamente en una único servidor físico. Xen proporciona garantías sobre los recursos a los servidores virtuales para asegurar que los niveles de servicio de cada aplicación se respeten, incluyendo CPU, memoria y entrada/salida. Xen es la



infraestructura de virtualización por software más rápida y segura existente, y ha sido adoptada por los principales fabricantes y distribuidores, incluyendo a [Intel](#)⁽²⁾, [AMD](#)⁽³⁾, [Dell](#)⁽⁴⁾, [Hewlett-Packard](#)⁽⁵⁾, [IBM](#)⁽⁶⁾, [Novell](#)⁽⁷⁾, [Red Hat](#)⁽⁸⁾ o [Sun Microsystems](#)⁽⁹⁾. Xen se distribuye bajo la licencia [General Public License de GNU](#)⁽¹⁰⁾ y puede descargarse gratuitamente.

Un servidor virtual es simplemente una instancia de un sistema operativo y su carga de trabajo, ejecutándose bajo el paraguas del hipervisor Xen. En lugar de controlar el hardware directamente, las instancias de sistemas operativos acceden al hardware a través del hipervisor, el cuál además tiene la capacidad de compartir los recursos con otras aplicaciones e instancias de sistemas operativos virtualizadas.

Xen fue creado en el año 2003 en el laboratorio de computación de la [Universidad de Cambridge](#)⁽¹¹⁾ bajo lo que se conoce como el proyecto Xen Hypervisor liderado por Ian Pratt. Algunos de los miembros más destacados del proyecto son Keir Fraser, Steven Hand y Christian Limpach. Este mismo equipo fundó [XenSource](#)⁽¹²⁾ conjuntamente con Nick Gault y Simon Crosby, que aportaron su experiencia como empresarios en Silicon Valley.

Xen cada vez se usa más en centros de datos con el objetivo de incrementar la utilización de servidores y mejorar el coste total de propiedad (del inglés, *Total Cost of Ownership*). Xen es ampliamente utilizado en proveedores de servicios de aplicaciones y compañías de hospedaje porque ofrece un control preciso de los recursos del sistema y permite a los usuarios hospedar más servidores virtuales por máquina física. Xen también se usa en el desarrollo y verificación del funcionamiento de aplicaciones, pues la virtualización permite a los desarrolladores de aplicaciones multihilo hospedar múltiples máquinas virtuales y comprobar su correcto funcionamiento, ahorrando costes en infraestructuras. Más aún, el hardware de pruebas puede ser readaptado instantáneamente para otros usos simplemente instanciando servidores virtuales con las imágenes deseadas. Finalmente, las aplicaciones que han sido verificadas pueden ser puestas en producción directamente desde el entorno de pruebas basado en Xen simplemente migrando la máquina virtual pertinente.

En resumen, los tres principales beneficios para las empresas que Xen supone son:

- Incrementar la utilización de los servidores: los centros de datos pueden conseguir enormes mejoras en la utilización de sus servidores y posibilitar la fusión de los mismos, reduciendo los costes de capital y personal técnico.
- Reducir la cantidad de personal técnico necesario: al permitir la consolidación de servidores, Xen reduce el coste, la complejidad y la dedicación del personal requeridos para la gestión de los servidores de un centro de datos. Gracias a las herramientas de gestión, control y automatización de Xen, las empresas pueden virtualizar sus servidores por un modesto coste.
- Reducir el TCO: hoy en día, la virtualización todavía no ha llegado al corazón de los centros de datos. Las razones son dos: el bajo rendimiento de los hipervisores propietarios y su enorme coste. Xen es el hipervisor con mejor rendimiento, y es gratuito. Ésto cambia radicalmente la perspectiva económica a la hora de considerar la virtualización.

Xen se diferencia de otras técnicas de virtualización en varios puntos:

- Es código abierto, lo que supone una mejor funcionalidad, mejor rendimiento y gran extensibilidad
- Es, sin duda alguna, el hipervisor con mejor rendimiento del mercado, fruto de su tecnología de paravirtualización, pionera y líder en el mercado, que permite la colaboración de los servidores hospedados para conseguir el mejor rendimiento en aplicaciones corporativas.
- Xen usa de manera óptima las capacidades de virtualización por hardware de los procesadores VT de Intel y los Pacifica de AMD.

La paravirtualización es la clave del éxito de Xen, pues permite obtener un rendimiento drásticamente mayor que los acercamientos alternativos existentes en el mercado. La paravirtualización supone hacer que el sistema operativo del servidor virtual sea consciente de que está siendo virtualizado para permitir una colaboración entre ambas partes que facilite el rendimiento más óptimo. En [Linux](#)⁽¹³⁾, [BSD](#)⁽¹⁴⁾ y [Solaris](#)⁽¹⁵⁾, los hosts paravirtualizados ven a Xen como una capa de hardware idealizada. De hecho, Xen es simplemente una arquitectura de hardware idealizada del [kernel de Linux](#)⁽¹⁶⁾. Intel ha realizado diversas contribuciones a Xen que han permitido añadir soporte para sus extensiones de arquitectura [VT-X Vanderpool](#)⁽¹⁷⁾. Esta tecnología permite que sistemas operativos sin modificar actúen como hosts dentro de las máquinas virtuales de Xen, siempre y cuando el servidor físico soporte las extensiones VT de Intel o [Pacífica de AMD](#)⁽¹⁸⁾. Para [Microsoft Windows](#)⁽¹⁹⁾ y otros hosts que no están al tanto de la existencia de Xen, la capa de virtualización VT de Intel, combinada con la paravirtualización de los controladores de Windows, permiten a Xen conseguir el mismo nivel de rendimiento que los hosts Linux virtualizados.



Debido al pequeño tamaño del código necesario para ejecutar el hipervisor, la sobrecarga en el rendimiento típicamente se encuentra entre el 0,1% y el 3,5% (datos tomados con los benchmarks estándar del mercado). Además, la técnica de paravirtualización le permite a Xen beneficiarse de todos los controladores nativos de Linux y, por lo tanto, soportar una gran cantidad de dispositivos. Los controladores paravirtualizados de Xen se ejecutan fuera del núcleo del hipervisor, donde se implementa una política de compartición de recursos entre las diversas máquinas virtuales, proporcionando así un particionamiento muy eficiente de los recursos de E/S entre los servidores virtuales. Otro beneficio de este acercamiento es que los controladores se ejecutan en un nivel de protección más bajo que Xen, manteniendo al hipervisor a salvo de errores en los controladores.

En términos de seguridad, Xen soporta un aislamiento absoluto de los recursos entre dominios, lo que significa que tiene el nivel más alto posible de separación y seguridad en un hardware de tipo i386. No es posible, por ejemplo, usar [tcpdump](#)⁽²⁰⁾ en un host virtual para ver el tráfico de los demás hosts virtuales. Además, el código fuente base de Xen es muy pequeño (el núcleo del hipervisor tiene menos de 40.000 líneas), lo que permite realizar auditorías de seguridad en el código más fácilmente. Más importante aún, Xen puede usar las características de seguridad del hardware, como los [Trusted Platform Modules](#)⁽²¹⁾ (TPM), para construir una capa de monitorización del uso del hardware a través del software. En el [Intel Developer Forum](#)⁽²²⁾ de agosto del 2005, XenSource demostró una solución de hipervisor seguro al integrar Xen con el sistema de detección de intrusos Snort, aplicación de código abierto líder del mercado. De este modo los usuarios reciben una nueva y potente posibilidad para implementar las mismas políticas de seguridad en los servidores virtuales, independientemente del sistema operativo. Más aún, el hipervisor puede incluso asegurar que hosts que no han sido parcheados serán protegidos. Xen puede también impedir que un servidor virtual comprometido se use para atacar a otros servidores virtuales o físicos bloqueando su tráfico.

Las máquinas virtuales de Xen pueden migrarse en caliente entre hosts físicos sin necesidad de detenerlos. Durante este proceso, la memoria de la máquina virtual se copia iterativamente al destino sin parar su ejecución. Una pequeña pausa de entre 60 y 300 milisegundos es necesaria para llevar a cabo la sincronización final antes de que la máquina virtual empiece a ejecutarse en su nuevo destinatario, proporcionando así la apariencia de una migración sin parones. Una tecnología similar se usa para suspender a disco una máquina virtual en ejecución, cambiar a otra máquina virtual y recuperar más tarde la primera máquina virtual.

Estado actual de Xen

Actualmente, Xen corre sobre arquitecturas x86, con procesadores Pentium 4 o más modernos, x86_64 ([AMD64](#)⁽²³⁾ y [EM64T](#)⁽²⁴⁾) e IA-64. El soporte para arquitectura [POWER de IBM](#)⁽²⁵⁾ está listo y será añadido en la próxima versión. Ports para otras plataformas son técnicamente posibles y puede que estén disponibles en el futuro. La última versión de Xen, [Xen 3.0.2](#)⁽²⁶⁾, extiende el liderazgo en características y funcionalidad requerida para virtualizar los servidores que hoy en día se encuentran en los centros de datos de las empresas, incluyendo:

- Soporte para servidores virtuales con sistema operativo multiprocesador de 32 vías.
- Soporte para la virtualización por hardware de los Intel VT-X y los AMD Pacifica, que permite ejecutar sistemas operativos sin modificar dentro de las máquinas virtuales (Windows XP/2003 y UNIX antiguos)
- Soporte para Intel PAE (Physical Addressing Extensions) en servidores de 32 bits con más de 4 GB de RAM
- Soporte para las arquitecturas x86/64 (tanto AMD64 como EM64T)
- Mejores herramientas de control
- Mejor soporte de [ACPI](#)⁽²⁷⁾ (Advanced Configuration and Power Interface)
- Soporte para gráficos AGP/DRM

Estructura de un sistema basado en Xen

Un sistema Xen tiene múltiples capas, de las cuales la inferior y la que cuenta con más privilegios es el propio Xen.

Xen puede hospedar múltiples sistemas operativos huéspedes, cada uno de los cuales es ejecutado dentro de una máquina virtual segura. En terminología Xen, un *dominio*. Los dominios son gestionados por Xen de forma que se usen las CPUs físicas de la manera más efectiva. Cada sistema operativo huésped gestiona sus propias aplicaciones. Esta gestión incluye la responsabilidad de ofrecer servicio a cada aplicación dentro de la franja de tiempo dada por Xen a cada máquina virtual.

El primer dominio, o *dominio 0* (*dom0*), se crea automáticamente cuando el sistema arranca y tiene unos permisos de gestión



especiales. El dominio 0 crea los demás dominios y gestiona sus dispositivos virtuales. También realiza tareas administrativas como suspender, reanudar y migrar otras máquinas virtuales.

Dentro del dominio 0 se ejecuta un proceso llamado *xend*, que gestiona el sistema. Xend es responsable de la gestión de las máquinas virtuales (o *domU's*) y proporciona acceso a sus consolas. Xend puede recibir comandos a través de una interfaz HTTP o vía una utilidad de línea de comandos.

Instalación en Debian Etch

Las siguientes instrucciones toman como punto de partida un sistema [Debian Etch](#)⁽²⁸⁾ funcional con un kernel compilado a medida mediante las herramientas del paquete *kernel-package* sin módulos con una única interfaz de red pero, por supuesto, puede servir como base para que el lector las adapte a sus necesidades. Todas las sentencias deben ejecutarse como usuario *root* o con permisos de *root*. El primer paso será instalar los paquetes Debian necesarios:

```
apt-get install xen-docs-3.0 xen-tools xen-utils-3.0 bridge-utils iproute sysfsutils debootstrap udev
```

Para 32 bits habrá que añadir *libc6-xen* y *xen-hypervisor-3.0-i386* a la lista (o *xen-hypervisor-3.0-i386-pae* si se dispone de 4 o más GB de RAM). Para 64 bits deberemos añadir *xen-hypervisor-3.0-amd64* a la lista. El siguiente paso será descargar y descomprimir la versión 2.6.16 del kernel:

```
cd /usr/src/
wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.16.tar.bz2
tar -xjf linux-2.6.16.tar.bz2
```

Acto seguido necesitaremos descargar y descomprimir el parche 2.6.16.19 del kernel:

```
wget http://www.kernel.org/pub/linux/kernel/v2.6/patch-2.6.16.19.bz2
bunzip2 patch-2.6.16.19.bz2
```

Finalmente, descargaremos y descomprimiremos los parches de Xen para el kernel del servidor [Alioth de Debian](#)⁽²⁹⁾. La [última versión de los parches](#)⁽³⁰⁾ presente en el servidor en el momento de escribir este artículo es la 3.0.2 para el kernel 2.6.16.19:

```
wget http://alioth.debian.org/download.php/1605/linux-2.6.16.19-xen2.0.2-hg9697.patch.gz
gunzip linux-2.6.16.19-xen2.0.2-hg9697.patch.gz
```

Renombramos el directorio del núcleo y lo parcheamos:

```
mv linux-2.6.16 linux-2.6.16.19-xen
mv patch-2.6.16.19 linux-2.6.16.19-xen/
mv linux-2.6.16.19-xen2.0.2-hg9697.patch linux-2.6.16.19-xen/
cd linux-2.6.16.19-xen/
patch -s -p1 < patch-2.6.16.19
patch -s -p1 < linux-2.6.16.19-xen2.0.2-hg9697.patch
```

En este artículo se asume que el lector desea crearse su propio kernel a medida en lugar de usar el que viene precompilado con la distribución de Debian, y que quiere hacerlo sin usar módulos (con todos los controladores compilados dentro del kernel). Para ello usaremos la configuración existente del kernel en ejecución (el cuál se considera que cumple las condiciones mencionadas) y lo adaptaremos a los cambios que introduce Xen. Luego deberemos copiar la configuración actual dentro del nuevo árbol del kernel (se supone la existencia de un enlace débil */usr/src/linux* que apunta al árbol actual del kernel, */usr/src/linux-2.6.16.27* en mi sistema en el momento de escribir este artículo):

```
cp /usr/src/linux/.config /usr/src/linux-2.6.16.19-xen/
```

Configuramos el nuevo kernel:

```
make menuconfig
```

Al menos deberán seleccionarse las siguientes opciones (dependiendo de la configuración previa que se tuviera, éstas opciones pueden variar):

**Arquitectura de 32 bits:**

```
Processor type and features ---> Subarchitecture Type (Xen-compatible)
```

Arquitectura de 64 bits:

```
Processor type and features ---> Enable Xen compatible kernel
```

Ambas arquitecturas (32 and 64 bits):

```
Xen --->
```

```
Privileged Guest (domain 0)
Backend driver support
PCI device backend driver
    PCI Backend Mode (Virtual PCI)
    Block-device backend driver
    Network-device backend driver
        Network-device loopback driver
Block-device frontend driver
Network-device frontend driver
Scrub memory before freeing it to Xen
Disable serial port drivers
Export Xen attributes in sysfs
```

```
Networking ---> Networking Options --->
```

```
802.1d Ethernet Bridging
Network packet filtering (replaces ipchains) --->
    Bridged IP/ARP packets filtering
    Bridge: Netfilter Configuration --->
        Ethernet Bridge tables (eatables) support
        ebt: broute table support
        ebt: filter table support
        ebt: nat table support
        ebt: 802.3 filter support
        ebt: among filter support
        ebt: ARP filter support
        ebt: IP filter support
        ebt: limit match support
        ebt: mark filter support
        ebt: packet type filter support
        ebt: STP filter support
        ebt: 802.1Q VLAN filter support
        ebt: arp reply target support
        ebt: dnat target support
        ebt: mark target support
        ebt: redirect target support
        ebt: snat target support
        ebt: log support
```

```
Device Drivers ---> Network device support ---> Dummy net driver support
```

También será necesario tener soporte para los sistemas de ficheros que se quieran usar en las imágenes de Xen (Second Extended con journaling, también conocido como *ext3*, es el valor por defecto). Compilamos el nuevo kernel:

```
make-kpkg clean
make-kpkg --append-to-version -xen kernel-image
```

Instalamos el nuevo kernel:

Arquitectura de 32 bits:

```
dpkg --install /usr/src/linux-xenu-2.6.16.19-xen_2.6.16.19-xen-10.00.Custom_i386.deb
```

Arquitectura de 64 bits:

```
dpkg --install /usr/src/linux-xen0-2.6.16.19-xen_2.6.16.19-xen-10.00.Custom_amd64.deb
```



Es muy recomendable arrancar un kernel con Xen mediante [GRUB](#)⁽³¹⁾ (del inglés, *GRand Unified Bootloader*), por lo que deberemos instalar el paquete Debian correspondiente (se aconseja, aunque no es imprescindible, desinstalar LILO (del inglés, *Linux LOader*) si se tiene instalado una vez se haya conseguido arrancar el sistema con GRUB):

```
apt-get install grub
```

El comando anterior nos dejará GRUB instalado en el sistema, pero no configurado. En el documento */usr/share/doc/grub/README.Debian.gz* pueden hallarse instrucciones sobre cómo realizar dicha configuración. En mi caso, teniendo el sistema de ficheros XFS en mi partición raíz (existen ciertas incompatibilidades entre GRUB y XFS), fue necesario usar la consola de GRUB y teclear en ella los siguientes comandos:

```
grub
grub> root (hd0,0)
grub> setup (hd0)
grub> quit
```

Una vez instalado GRUB en el MBR (del inglés, *Master Boot Record*), deberemos configurar las entradas que queremos que aparezcan. En Debian existe un script llamado *update-grub*, que se encuentra en */sbin/update-grub*. Aunque no es capaz de detectar el kernel con soporte para Xen que acabamos de compilar e instalar, es necesario ejecutarlo una primera vez para que nos cree el fichero */boot/grub/menu.lst* con las entradas de los kernels que tengamos y las directivas y parámetros necesarios para el correcto funcionamiento de GRUB:

```
mkdir /boot/grub
update-grub
```

En el caso de que ocurran problemas al tratar de escribir GRUB en el MBR, arrancar la máquina en modo monousuario o pasar a modo *init 1* puede ayudar, pues es muy recomendable no tener ningún proceso escribiendo en el disco en el mismo momento en el que GRUB está intentando modificar el Master Boot Record.

Finalmente, editaremos */boot/grub/menu.lst* y añadiremos una entrada similar a la que sigue:

Arquitectura de 32 bits:

```
title Xen 3.0.2, Debian GNU/Linux, kernel 2.6.16.19-xen
root (hd0,0)
kernel /boot/xen-3.0-i386.gz noreboot
module /boot/xen0-linux-2.6.16.19-xen root=/dev/sda1 ro console=tty0
savedefault
boot
```

Arquitectura de 64 bits:

```
title Xen 3.0.2, Debian GNU/Linux, kernel 2.6.16.19-xen
root (hd0,0)
kernel /boot/xen-3.0-amd64.gz noreboot
module /boot/xen0-linux-2.6.16.19-xen root=/dev/sda1 ro console=tty0
savedefault
boot
```

Cuando se necesiten más de 4 máquinas virtuales, será necesario añadir el parámetro *max_loop=X* tras *console=tty0*, donde *X* será el número de máquinas virtuales que se desean ejecutar multiplicado por 2. Más allá de la octava máquina virtual, será necesario crear los dispositivos */dev/loop8*, */dev/loop9*, etc. (hasta el número que nos haga falta). Para ello, el siguiente script podrá sernos útil (sustituir el valor del segundo parámetro en la llamada al comando *seq -15* en el ejemplo— por el valor máximo menos uno de dispositivos *loopback* que se vayan a necesitar):

```
for i in $(seq 8 15)
do
    mknod /dev/loop$i b 7 $i
done
chmod 660 /dev/loop*
chown 0.disk /dev/loop*
```



Una vez terminada la configuración del kernel y de GRUB, el siguiente paso será configurar el fichero `/etc/xen/xend-config.sxp` según nuestras necesidades. Nos interesa crear un puente ethernet que estará formado por una interfaz ethernet falsa para cada máquina virtual y todas ellas unidas a la verdadera conexión ethernet. Si contamos con una única interfaz la configuración es muy sencilla. En el caso de tener más de una, el propio script viene repleto de comentarios que nos ayudarán a ajustar la configuración a nuestra medida.

Para ello, comentaremos la directiva (`network-script network-dummy`) y descomentaremos la directiva (`network-script network-bridge`). También nos hará falta la directiva (`vif-script vif-bridge`), pero ésta ya viene descomentada por defecto.

Si nos interesa agrupar todas nuestras máquinas virtuales en una red diferente, por ejemplo 192.168.0.x para `dom0` y las demás máquinas físicas de la red y 10.0.0.x para las máquinas virtuales, y luego usar DNAT con `iptables` para enrutar el tráfico, necesitaremos usar las directivas (`network-script network-nat`) y (`vif-script vif-nat`). Pero en el caso medio la solución ofrecida por un `ethernet bridge` cubrirá nuestras necesidades y es el caso que se ha usado en este artículo.

Ya estamos listos para reiniciar la máquina:

```
reboot
```

Una vez el nuevo sistema haya terminado de cargar, deberíamos ver el puente creado mediante el comando `ifconfig`. Acto seguido, editaremos el fichero `/etc/xen-tools/xen-tools.conf` según nos convenga, por ejemplo tal que así:

```
dir           = /home/xen
debootstrap  = 1
size         = 2Gb
memory       = 256Mb
swap         = 512Mb
fs           = ext3
dist         = sarge
image        = sparse
gateway      = 192.168.0.1
netmask      = 255.255.255.0
passwd       = 1
kernel       = /boot/xen0-linux-2.6.16.19-xen
initrd       =
mirror       = http://ftp.es.debian.org/debian/
```

Notas:

- Es conveniente usar el mirror más cercano a nuestra situación geográfica.
- El valor de la directiva `gateway` deberá cambiarse a la puerta de enlace que usemos para salir a Internet.
- La máscara de red deberá de modificarse en concordancia a la red que tengamos configurada también.

Ahora crearemos el directorio que contendrá las imágenes de los sistemas huéspedes:

```
mkdir /home/xen
```

Con lo que ya estaremos listos para crear nuestra primera imagen de sistema virtual mediante la utilidad `xen-create-image` del paquete `xen-tools`:

```
xen-create-image --hostname=xen01 --ip=192.168.0.5 --passwd
```

Este script creará la imagen de la memoria de intercambio o `swap` y la imagen del disco, a la cuál dará luego formato en el sistema de ficheros que hayamos elegido y, por último, instalará el nuevo sistema mediante `debootstrap`⁽³²⁾ usando el mirror seleccionado. Finalmente, creará un fichero de configuración de la máquina virtual en `/etc/xen/xen01.cfg` y nos pedirá que introduzcamos una clave de `root`. Una vez se hayan completado todos estos pasos, podremos arrancar la nueva imagen:

```
xm create xen01.cfg -c
```

Una vez haya arrancado la máquina virtual, podremos validarnos en el sistema como `root` y cambiar la clave si nos apetece. Luego deberemos comprobar que la red está correctamente configurada mediante algunos pings a hosts cuya dirección IP conozcamos (no tenemos nameservers configurados en estos momentos, así es que no funcionará la resolución de nombres). Una vez hayamos terminado, pulsaremos `Ctrl + J` (Ctrl + AltGr + <signo de suma> en los teclados españoles) para salir de la



consola virtual y trataremos de conectarnos a la máquina virtual mediante [OpenSSH](#)⁽³³⁾:

```
ssh -l root 192.168.0.5
```

Una vez validados en el sistema, añadiremos nuestros servidores de nombres al fichero `/etc/resolv.conf` y nuestra nueva máquina virtual ya estará disponible para su uso.

Algunas herramientas y comandos de interés que pueden sernos muy útiles durante la configuración son las siguientes:

- `xm console xen01` nos permitirá conectar una consola a la máquina virtual `xen01`
- `xm list` nos mostrará una lista de las máquinas virtuales en ejecución y algunos datos de ellas. Un ejemplo de lo que veríamos sería lo siguiente:

Name	ID	Mem(MiB)	VCPUs	State	Time(s)
Domain-0	0	98	1	r-----	5068.6
xen01	164	128	1	r-----	7.6

- `xen-list-images` nos mostrará una lista de las imágenes existentes y su dirección IP actual.

Interfaces de red virtuales en un sistema Xen

Xen crea, por defecto, siete pares de *interfaces ethernet virtuales interconectadas* para que `dom0` las utilice. Podríamos concebirlas como dos interfaces ethernet conectadas por un cable ethernet cruzado interno. `veth0` está conectada a `vif0.0`, `veth1` está conectada a `vif0.1`, y así sucesivamente hasta `veth7`, que está conectada a `vif0.7`. Pueden accederse o usarse configurando una dirección IP y una MAC en el costado de la `veth#` y luego enlazando el extremo `vif0.#` al puente.

Cada vez que se crea una instancia `domU`, ésta recibe un identificador numérico (asignado automáticamente y sin la posibilidad de que el usuario lo elija). El primer `domU` será el número 1, el segundo el número 2, incluso aunque el número 1 ya no se esté ejecutando, etc.

Para cada nuevo `domU`, Xen crea un nuevo par de interfaces ethernet virtuales conectados, con un extremo de cada par dentro del `domU` y el otro en el `dom0`. Si el `domU` usa Linux, el nombre de dispositivo se mostrará como `eth0`. El otro extremo de ese par de interfaces ethernet virtuales aparecerá dentro del `dom0` como interfaz `vif#.0`. Por ejemplo, la interfaz `eth0` del `domU` número 5 está conectada a `vif5.0`. Si se crean múltiples interfaces de red dentro de un `domU`, sus extremos se verán como `eth0`, `eth1`, etc, mientras que dentro de `dom0` aparecerán como `vif#.0`, `vif#.1`, etc. La siguiente figura muestra las tarjetas de red lógicas conectadas entre el `dom0` y `dom1`:

Cuando un `domU` se detiene (usando el comando `xm shutdown domId`, por ejemplo), los interfaces ethernet virtuales que se crearon son eliminados.

Puentes de red en un sistema Xen

La configuración por defecto de Xen crea puentes de red (del inglés, *bridges* o *bridging*) dentro de `dom0` para permitir que todos los dominios aparezcan en la red como hosts independientes. Si se va a hacer un uso extensivo de [iptables](#)⁽³⁴⁾, por ejemplo para un cortafuegos, podría verse afectado el puente pues los paquetes transferidos por el puente pasan a través de las cadenas `PREROUTING`, `FORWARD` y `POSTROUTING`. Esto supone que los paquetes que viajan entre los dominios huésped y la red exterior deberán de tener permiso para poder pasar por esas cadenas. El problema más probable es que la cadena `FORWARD` esté configurada para descartar (`DROP`) o rechazar (`REJECT`) paquetes.

Para evitar que `dom0` actúe como un enrutador IP podemos usar el siguiente comando:

```
echo 0 > /proc/sys/net/ipv4/ip_forward
```

Un método ligeramente más seguro consistiría en permitir el reenvío de paquetes (`FORWARD`) a nivel de `iptables` entre la interfaz física externa y las interfaces virtuales `vif's` de los huéspedes. En una máquina con una única interfaz deberíamos usar las siguientes reglas:

```
iptables --append FORWARD --match physdev --physdev-in eth0 --physdev-out '!' eth0 --jump ACCEPT
iptables --append FORWARD --match physdev --physdev-out eth0 --physdev-in '!' eth0 --jump ACCEPT
```



(será preciso que el kernel sea compilado con soporte para *ipt_physdev*, que es posible que aparezca como *xt_physdev*)

Flujo de paquetes en el puente

Cuando un paquete llega al hardware, el controlador ethernet del dominio 0 lo gestiona y aparece en la interfaz *peth0*. *peth0* está ligado al puente, por lo que es transferido al puente desde ahí. Este paso se ejecuta a nivel ethernet (no hay ninguna dirección IP establecida en la *peth0* o en el puente).

Acto seguido el puente distribuye el paquete del mismo modo que lo haría un concentrador (del inglés, *switch*). Filtrar a este nivel sería posible mediante *eatables*⁽³⁵⁾. Luego, de entre las interfaces *vifX.Y* conectadas al puente se decide a dónde mandar el paquete basándose en la dirección MAC del receptor.

La interfaz *vif* pasa el paquete a Xen, el cuál a continuación lo envía de vuelta al dominio al cuál la *vif* apunta (también se hace así para el *dom0*, pues *vif0.0* está conectada a *veth0*). Finalmente, el dispositivo destinatario del *dom0/domU* tiene una dirección IP, por lo que se puede aplicar *iptables* aquí.

El script *network-bridge*

Cuando Xen arranca, ejecuta el script */etc/xen/scripts/network-bridge*, el cuál lleva a cabo las siguientes tareas:

- Crea un nuevo puente llamado *xenbr0*.
- Desactiva la interfaz ethernet real *eth0*.
- Copia las direcciones MAC e IP de la *eth0* a la interfaz virtual de red *veth0*.
- Renombra la interfaz real *eth0* a *peth0*.
- Renombra la interfaz virtual *veth0* a *eth0*.
- Conecta *peth0* y *vif0.0* al puente *xenbr0*.
- Activa el puente, *peth0*, *eth0* y *vif0.0*.

Es conveniente tener la interfaz física y la interfaz del *dom0* separadas, pues así es posible crear un cortafuegos en el *dom0* que no afecte al tráfico de los *domU*'s (que proteja únicamente el *dom0*).

El script *vif-bridge*

Cuando arranca un *domU*, *xend*, que se está ejecutando en *dom0*, lanza el script *vif-bridge*, el cuál lleva a cabo las siguientes tareas:

- Enlaza la interfaz *vif#.0* al puente *xenbr0*.
- Levanta la interfaz *vif#.0*.

Bibliografía

- [Computer Laboratory – Xen virtual machine monitor](#)⁽¹⁾
- [Computer Laboratory – Architecture](#)⁽³⁶⁾
- [Installing Xen 3.0 upon Debian Sarge](#)⁽³⁷⁾
- [Installing Xen 3.0 upon Debian Unstable, with a custom Kernel](#)⁽³⁸⁾
- [Xen 3 for Debian](#)⁽³⁹⁾
- [Xen Virtualization and Linux Clustering, Part 1](#)⁽⁴⁰⁾
- [Building A Virtual Cluster with Xen \(Part One\)](#)⁽⁴¹⁾
- [The Perfect Xen 3.0 Setup For Debian](#)⁽⁴²⁾
- [How To Set Up Xen 3.0 From Binaries In Ubuntu 6.06 LTS \(Dapper Drake\)](#)⁽⁴³⁾
- [X86 virtualization](#)⁽⁴⁴⁾
- [Xen](#)⁽⁴⁵⁾
- [Hypervisor](#)⁽⁴⁶⁾
- [XenSource Products – Xen 3.0](#)⁽⁴⁷⁾



- [Xen Wiki – XenDownloads](#)⁽⁴⁸⁾
- [Xen Wiki – XenNetworking](#)⁽⁴⁹⁾
- [Xen Wiki – IntelVT](#)⁽⁵⁰⁾
- [Linux Virtualization with Xen](#)⁽⁵¹⁾

Historial de revisiones

Fecha	Versión	Cambios
29/08/2006	1.0	Documento inicial

Lista de enlaces de este artículo:

1. <http://www.cl.cam.ac.uk/Research/SRG/netos/xen/>
2. <http://www.intel.com/>
3. <http://www.amd.com/>
4. <http://www.dell.com/>
5. <http://www.hp.com/>
6. <http://www.ibm.com/>
7. <http://www.novell.com/>
8. <http://www.redhat.com/>
9. <http://www.sun.com/>
10. <http://www.gnu.org/licenses/gpl.html>
11. <http://www.cam.ac.uk/>
12. <http://www.xensource.com/>
13. <http://www.linux.org/>
14. <http://www.freebsd.org/>
15. <http://www.sun.com/software/solaris/>
16. <http://www.kernel.org/>
17. <http://www.intel.com/technology/computing/vpotech/>
18. <http://enterprise.amd.com/us-en/Solutions/Consolidation/virtualization.aspx>
19. <http://www.microsoft.com/windows/>
20. <http://www.tcpcdump.org/>
21. <http://www.trustedcomputinggroup.org/>
22. <http://www.intel.com/idf/>
23. <http://www.x86-64.org/>
24. <http://www.intel.com/technology/64bitextensions/>
25. <http://www.ibm.com/chips/products/powerpc/>
26. <http://www.cl.cam.ac.uk/Research/SRG/netos/xen/downloads.html>
27. <http://www.acpi.info/>
28. <http://www.debian.org/>
29. <http://alioth.debian.org/>
30. http://alioth.debian.org/project/showfiles.php?group_id=30894
31. <http://www.gnu.org/software/grub/>
32. <http://packages.debian.org/stable/admin/debootstrap>
33. <http://www.openssh.com/>
34. <http://www.netfilter.org/>
35. <http://ebtables.sourceforge.net/>
36. <http://www.cl.cam.ac.uk/research/srg/netos/xen/architecture.html>
37. <http://www.debian-administration.org/articles/304>
38. <http://www.debian-administration.org/articles/320>
39. <http://julien.danjou.info/xen.html>
40. <http://www.linuxjournal.com/article/8812>
41. <http://www.clustermonkey.net/content/view/139/33/>
42. http://www.howtoforge.com/perfect_setup_xen3_debian
43. http://www.howtoforge.com/xen_3.0_ubuntu_dapper_drake
44. http://en.wikipedia.org/wiki/X86_virtualization



45. <http://es.wikipedia.org/wiki/Xen>
46. <http://en.wikipedia.org/wiki/Hypervisor>
47. <http://www.xensource.com/products/xen/>
48. <http://wiki.xensource.com/xenwiki/XenDownloads>
49. <http://wiki.xensource.com/xenwiki/XenNetworking>
50. <http://wiki.xensource.com/xenwiki/IntelVT>
51. <http://www.linuxdevcenter.com/pub/a/linux/2006/01/26/xen.html>

E-mail del autor: jsabater_ARROBA_linuxsilo.net

Podrás encontrar este artículo e información adicional en: <http://bulma.net/body.phtml?nIdNoticia=2362>