Virtualización con Xen

- Revisión 3 -

Saúl Ibarra Corretgé

Licencia

Introducción

En esta guía se montará un servidor sobre el sistema de virtualización Xen, para posteriormente arrancar varias máquinas virtuales sobre él.

¿Qué es Xen?

[Wikipedia] Xen utiliza una técnica llamada paravirtualización para alcanzar alto rendimiento (es decir, bajas penalizaciones del rendimiento, típicamente alrededor del 2%, con los peores casos de rendimiento rondando el 8%; esto contrasta con las soluciones de emulación que habitualmente sufren penalizaciones de un 20%). Con la paravirtualización, se puede alcanzar alto rendimiento incluso en arquitecturas (x86) que no suelen conseguirse con técnicas tradicionales de virtualización. A diferencia de las máquinas virtuales tradicionales, que proporcionan entornos basados en software para simular hardware, 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, pero las licencias de Microsoft prohíben su lanzamiento público.

¿Qué vamos a montar?

El sistema que vamos a montar consiste en un dominio dom0, que será una máquina Debian, y después montaremos varios domU, que también serán Debian (mediante debootstrap).

Instalar Xen ya no es algo extremadamente complicado, ya que existen paquetes precompilados en Debian, pero eso nos impide tocar la configuración del Kernel, cosa que necesitaremos.

Virtualización vs Paravirtualización

Xen tiene 2 formas de operar: HVM o 'Full Virtualization', es decir, virtualización completa, que consiste en la instalación de un domU como si fuera un host independiente; o la paravirtualización, que consiste en utilizar un kernel modificado para que pueda comunicarse con el hypervisor de Xen.

El usar HVM tiene la ventaja de que se puede virtualizar Windows, y que no es necesario tener un kernel especial para virtualizar sistemas GNU/Linux, pero su rendimiento es inferior, ya que la ausencia de un kernel adaptado, ciertos componentes son emulados. Con HVM no es posible virtualizar sistemas *BSD.

La paravirtualización tiene el inconveniente de necesitar un kernel adaptado, pero el rendimiento es superior, y se pueden virtualizar distintos sabores BSD.

Para poder utilizar HVM, necesitamos soporte para ello en el procesador. Podemos comprobarlo mirando si tiene la instrucción 'vmx':

```
cat /proc/cpuinfo | grep flags
flags: fpu de 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
```

Manos a la obra: Instalando el sistema dom0

Lo primero que haremos será realizar una instalación básica de Debian, utilizando el CD de netinst, de manera que obtendremos una instalación de Debian mínima.

Una vez el sistema esta instalado, conviene configurar la red e instalar el servidor SSH, para proseguir con la instalación desde otro PC.

```
# apt-get install ssh openssh-server
```

Instalaremos los paquetes necesarios antes de comenzar a instalar Xen:

```
# apt-get install iproute bridge-utils python-twisted binutils
zlib1g-dev python-dev transfig bzip2 screen ssh debootstrap
libcurl3-dev libncurses5-dev x-dev build-essential gettext gawk
mercurial yaird
```

Si vamos a querer utilizar HVM, también instalaremos estos paquetes:

```
bcc libsdl1.2debian-all libsdl1.2-dev libx86-dev libvncserver-dev
```

Compilar un Kernel para Xen (dom0)

Primero descargamos las fuentes de Xen de: http://bits.xensource.com/oss-xen/release/3.2.0/xen-3.2.0.tar.gz

y las descomprimimos:

```
# tar -xvzf xen-3.2.0.tar.gz
```

Descargamos los sources del Kernel con Mercurial:

```
# hg clone http://xenbits.xensource.com/linux-2.6.18-xen.hg
```

Ahora compilamos e instalamos el Xen Hypervisor y las herramientas necesarias:

```
# cd xen-3.2.0
```

```
# make xen
# make install-xen
# make tools
# make install-tools
```

Una vez el Hypervisor y las herramientas están compiladas, pasamos a compilar el kernel Xen:

```
# cd linux-2.6.18-xe.hg
# make menuconfig
```

Aquí se nos presenta el típico menú ncurses donde tenemos que seleccionar los componentes del kernel. Se recomienda seleccionar lo menos posible, e integrarlo todo en el kernel, generando el menor número de módulos posibles.

A modo de recomendación, conviene seleccionar las siguientes opciones:

```
File systems --> [*] Quota support
<M> Old quota format support
<M> Quota format v2 support
```

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

```
Networking ---> Networking options ---> [*] Network packet
filtering (replaces ipchains) ---> Core Netfilter Configuration
---> <M> Netfilter Xtables support (required for ip_tables)
```

```
Networking ---> Networking options ---> [*] Network packet
filtering (replaces ipchains) ---> IP: Netfilter Configuration -->
<M> IP tables support (required for filtering/masq/NAT)
```

```
Processor Type and Features ---> Timer Frequency (1000 Hz)
```

IMPORTANTE: Hay que seleccionar todos los componentes del hardware del servidor (tarjeta de red, controladora, etc) sino obtendremos un kernel que no arranca.

A continuación compilamos e instalamos el Kernel:

```
# make
# make modules
# make modules_install
# make install
```

NOTA: Actualmente no funciona el make-kpkg para kernels Xen, así que hay que hacerlo así.

Últimos ajustes y lo incluimos en el GRUB:

```
# depmod 2.6.18.8
# mkinitrd.yaird -o /boot/initrd.img-2.6.18.8 2.6.18.8
# update-grub
```

Una vez ejecutado esto se nos habrá creado una nueva entrada en el GRUB, como esta:

```
title           Xen 3.2.0 / Debian GNU/Linux, kernel 2.6.18.8
root            (hd0,0)
kernel          /boot/xen-3.2.0.gz
module          /boot/vmlinuz-2.6.18.8 root=/dev/sda1 ro
module          /boot/initrd.img-2.6.18.8
savedefault
```

Antes de reiniciar el PC, tenemos que incluir las utilidades xend y xendomains en el arranque así:
roo

```
# update-rc.d xend defaults 20 21
# update-rc.d xendomains defaults 21 20
```

Ya podemos reiniciar el PC y arrancar con el nuevo Kernel con Xen para el Dom0. Cuando arranquemos por primera vez, comprobamos que todo el hardware es detectado y que funciona correctamente.

Crear un nuevo domU con HVM

Para crear un domU con HVM tenemos que crear su fichero de configuración en /etx/xen. Podemos partir del ejemplo en /etc/xen/xmecample.hvm y crearnos el nuestro. Por ejemplo:

```
(omitida la cabecera del fichero)

kernel = "/usr/lib/xen/boot/hvmloader"
builder='hvm'
memory = 128
name = "base"
vif = [ 'type=ioemu, mac=0a:0b:0c:1a:1b:1c,bridge=xenbr0' ]
disk = [ 'file:/var/xen-guests/base.disk,hda,w', ',hdc:cdrom,r' ]

on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

```
device_model = '/usr/' + arch_libdir + '/xen/bin/qemu-dm'

boot="dc"
sdl=0
vnc=1
nographici=0
stdvga=0
serial='pty'
localtime=1
keymap='es'
```

Parámetros importantes:

- memory: indica cuanta memoria va a poder utilizar este domU.
- name: indica el nombre del domU.
- vif: especifica la forma de manejar la tarjeta de red. En el ejemplo le hemos asignado una mac y la hemos vinculado al puente xenbr0. Si queremos que el domU salga a red bridgeado a través del dom0 hay que indicar el xenbr0.
- Disk: indica los discos duros y CD-ROM que tendrá el domU. La sintáxis es: dispositivo en el dom0, nombre en el domU, modo de acceso. Así, un disco duro en forma de fichero en el dom0 es:
  `file:/var/xen-guests/base.disk,hda,w`
  y será visto en el domU como /dev/hda.

  NOTA: Para crear un fichero que podamos usar de disco duro usaremos la utilidad dd:
  `dd if=/dev/zero of=imagen.disk bs=1024 count=(tamaño en KB)`

  Un CD-ROM sería así:
  `file:/var/iso/debian-netinst.iso,hdc:cdrom,r`
  que sería visto en el domU como /dev/hdc.
- boot: indica la prioridad en el arranque: d indica CD-ROM y c disco duro.
- vnc: habilita el VNC para conectarse a esta máquina y ver su 'pantalla'.

Hay bastantes más parámetros que vienen comentados en el fichero de ejemplo, pero con los comentados arriba es suficiente.

Antes de comenzar a instalar el nuevo domU es necesario editar el fichero /etc/xen/xend-config.sxp indicando lo siguiente:

```
# Para que la red sea en modo bridge.
(network-script network-bridge)
(vif-script vif-bridge)


# Mínimo de memoria para el dom0
(dom0-min-mem 196)
```

```
# Permitimos que el dom0 use todas las CPUs.
(dom0-cpus 0)

# Habilitamos la consola VNC desde el exterior y sin password.
(vnc-listen '0.0.0.0')
(vncpasswd '')
```

Llegado este punto ya tenemos todo lo necesario para comenzar la instalación del domU, así que ejecutamos:

```
# xm create /etc/xen/nombre-del-domu.hvm
```

Así el domU ya esta funcionando, y para conectarnos a su pantalla, desde otro PC ejecutamos:

```
# vncviewer IP_del_servidor_Xen
```

Y veremos en una ventana de VNC lo que veríamos si ese domU fuera un host real. ¿Mola, no?

Crear un nuevo domU (ParaVirtualización)

Para crear un domU para virtualizado lo primero que necesitamos es un kernel 'especial', es decir, un kernel capaz de comunicarse con el Hypervisor de Xen. Desde el kernel 2.6.23, esta característica esta disponible en el 'mainline' del kernel, es decir, no hay que parchear nada, con descargarlo de kernel.org y seleccionar las opciones adecuadas es suficiente.

IMPORTANTE: El soporte para Xen que tienen los kernels vanilla (los de kernel.org) es SOLO para el domU, para el dom0 es necesario utilizar el kernel descargado de Mercurial.

Para compilar un Kernel con soporte de paravirtualización haremos lo siguiente:

Descargamos y descomprimimos las fuentes del kernel en /usr/src:

```
# wget http://kernel.org/pub/linux/kernel/v2.6/linux-2.6.24.3.tar.bz2
# tar -jxvf linux-2.6.24.3.tar.bz2
# ln -s /usr/src/linux-2.6.24.3 /usr/src/linux
```

La configuración de este tipo de kernel puede ser algo problemática, así que yo opté por bajarme el fichero de configuración del paquete precompilado de Debian Sid y luego hacer modificaciones.

Para ello descargaremos el paquete precompilado y extraeremos el fichero de configuración:

```
# wget http://ftp.es.debian.org/debian/pool/main/l/linux-2.6/linux-
modules-2.6.24-1-xen-686_2.6.24-4_i386.deb
# dpkg -x linux-modules-2.6.24-1-xen-686_2.6.24-4_i386.deb linux-tmp
# cp linux-tmp/boo/config.* /usr/src/linux/.config
```

Con esto ya tenemos lo que necesitamos para compilar el Kernel. Ahora ejecutamos:

```
# make oldconfig
# make menuconfig
```

Aquí lo más importante es habilitar 'Paravirtualization Support' -> 'Xen guest support' en la sección 'General options'.

```
# make
# make modules
# make modules_install
```

Aquí no podemos hacer un 'make install', porque Xen no soporta cargar bzImages, así que haremos lo siguiente:

```
# strip vmlinux -o vmlinux-stripped
# gzip vmlinux-stripped -c > /boot/vmlinuz-2.6.24.3
```

Y creamos el initrd:

```
# depmod 2.6.24.3
# mkinitramfs -o /boot/initrd.img-2.6.24.3 2.6.24.3
```

Ya tenemos todo lo necesario para comenzar a crear domUs!

Para crear las máquinas virtuales, utilizaremos el paquete xen-tools, que nos permiten automatizar el proceso, y hacerlo de una manera sencilla. Para ello instalamos el paquete:

```
#apt-get install xen-tools
```

La configuración sobre como se crearán las nuevas máquinas virtuales se almacena en el fichero /etc/xen-tools/xen-tools.cfg. Lo editamos, dejándolo así:

```
dir = /var/xen-gests     #donde se almacenan las máquinas virtuales
debootstrap = 1          #indicamos que haremos deboot
size   = 2Gb             #tamaño del disco
memory = 128Mb           #cuanta memoria queremos utilizar
swap   = 128Mb           #tamaño de la swap
fs     = ext3            #tipo de sistema de ficheros
dist   = etch            #distribución de GNU/Linux
arch = i386              #arquitectura
passwd = 1               #indicar si se pedirá contraseña
mirror = http://ftp.es.debian.org/debian/   #mirror
gateway   = 192.168.1.100     #puerta de enlace
netmask   = 255.255.255.0     #mascara de subred
kernel = /boot/vmlinuz-2.6.24.3        #el kernel que creamos
```

Ahora ejecutamos lo siguiente para crear una nueva máquina virtual, haciendo debootstrap de Debian Etch, con la IP 192.168.1.125:

```
#xen-create-image --hostname=base-domU --ip=192.168.1.25
```

Con esto se habrá creado un fichero de configuración /etc/xen/base-domU.cfg parecido a este:

```
kernel  = '/boot/vmlinuz-2.6.24.3'
ramdisk = '/boot/initrd.img-2.6.24.3'
extra = 'console=hvc0 nomce'
memory  = '128'


root    = '/dev/xvda1 ro'
disk    = [ 'phy:/dev/xen-guests/lvmbase,xvda1,w', 'phy:/dev/xen-
guests/lvmbase-swap,xvda2,w' ]


name    = 'lvmbase'
vif  = [ '' ]

on_poweroff = 'destroy'
on_reboot   = 'restart'
on_crash    = 'restart'
```

En este caso estoy utilizando volúmenes físicos para los discos (phy:) pero se puede utilizar un fichero, aunque el rendimiento es menor.

RECOMENDACIÓN: El rendimiento es mejor utilizando discos físicos, por lo que LVM es una buena opción, además nos permite levantar nuevos domUs clonando la base en cuestión de minutos.

NOTA: Es conveniente montar el sistema de ficheros antes de arrancar por primera vez el domU y hacer algunos ajustes:

- Configuración del hostname.
- Configuración de red.
- Desactivar las reglas de red persistentes de udev**

** Al haber hecho debootstrap no viene udev. Pero tras instalarlo, la red puede dejar de funcionar, ya que en cada arranque la MAC del domU cambia aleatoriamente. Para evitar esto editamos el fichero `/etc/udev/rules.d/z45_persistent-net-generator.rules` y comentamos estas líneas:

```
# ignore interfaces without a driver link like bridges and VLANs
#KERNEL=="eth*|ath*|wlan*|ra*|sta*", DRIVERS=="?*",\
#        IMPORT{program}="write_net_rules $attr{address}"
```

Si ya se habían creado asignaciones persistentes, podemos eliminarlas borrando el fichero: `/etc/udev/rules.d/z25_persistent-net.rules`

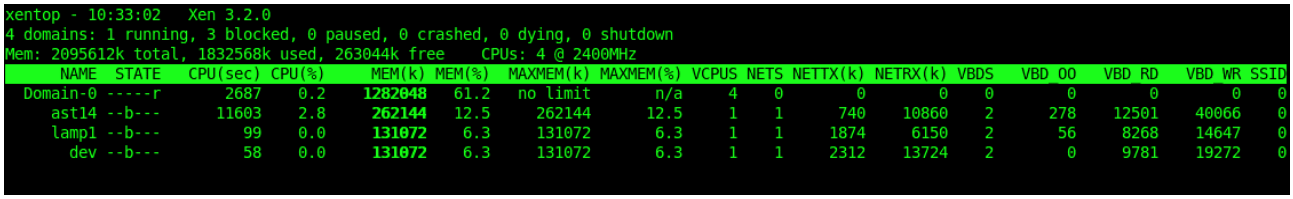
Una vez hemos hecho esto, desmontamos el sistema de ficheros y ya podemos arrancar la máquina virtual ejecutando:

```
#xm create /etc/xen/base-domU.cfg -c
```

La opción -c indica que lo arrancaremos en 'primer plano', de manera que veremos todo el arranque hasta llegar al login.

Ahora es necesario ajustar el sistema, ya que al haber sido instalado mediante debootstrap no tiene ni udev, ni las locales, etc instaladas.

Ya tenemos nuestro Xen server en marcha. Enjoy it!



Más información:

Xen User Guide:
http://bits.xensource.com/Xen/docs/user.pdf

Xen Wiki:
http://wiki.xensource.com/xenwiki/

Links y ejemplos:
http://del.icio.us/saghul/xen