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

# Máquinas virtuales con user-mode-linux (31586 lectures)

Per **Carles Bayés**, ***phalangana*** (http://)

Creado el 12/09/2004 21:27 modificado el 12/09/2004 21:27

*Desde hace ya un tiempo parece que las máquinas virtuales se han puesto de moda, no sólo para entornos de desarrollo o pruebas, sino también para entornos de producción (hosting, "consolidación de servidores", ...). Os muestro con un pequeño ejemplo cómo montar máquinas virtuales con user-mode-linux.*

Cuando te planteas montar máquinas virtuales, te encuentras con muchas soluciones a tu disposición, algunas comerciales como VMWare[1] o VirtualPC[2] (recientemente adquirida per Microsoft) y otras libres, como Bochs[3], Plex86[4], coLinux[5] o user-mode-linux[6]. Cada una de estas opciones tiene sus ventajas e inconvenientes. Yo me decanté por user-mode-linux porque es libre y porque lo que según mi opinión es su inconveniente más importante (sólo permite correr Linux), no me importaba en absoluto para lo que tenía pensado. Y entre las opciones libres era la que en principio ofrecía mejor rendimiento y más simplicidad.

Como decía en el resumen inicial, los usos de estas tecnologías pueden ser muchos. Algunos de los que se habla más últimamente son los siguientes, pero no son los únicos ni mucho menos:

- **Hosting**: Cada vez son más los ISP que ofrecen servidores virtuales usando estas tecnologías.
- **Consolidación de servidores**: Se trata de agrupar todos los servidores de una empresa en una sola máquina (que tiene que tener cierta solvencia de recursos, evidentemente). La idea se basa en aprovechar mejor los recursos del servidor, ya que es habitual el desaprovechamiento de recursos de hardware en estos tiempos en los que el hardware avanza tan deprisa. En estos casos, como siempre que se usan máquinas virtuales,la realización de copias de seguridad de cada una de las máquinas resulta muy fácil, puesto que en general supondrá la copia de un solo fichero.
- ***Honeypots***: Máquinas puestas en internet para que los hackers "jueguen" con ellas. Se usan en general para aprender los comportamientos y las nuevas técnicas que usan los intrusos informáticos.
- **Máquinas de desarrollo o pruebas**: Siempre es mejor probar las cosas en una máquina que no es crítica para el negocio y que, como en el caso de las máquinas virtuales, se puede recuperar en muy poco tiempo.

Creo que no hay nada como un caso práctico, o sea que ahora que ya he hecho un poco de introducción vayamos al grano: Lo que quiero montar en primera instancia es un servidor virtual para desarrollo de aplicaciones web con Apache, Mysql y Php (y puede que Perl), pero el objetivo final es implementarlo en un servidor web en el que alojaré las aplicaciones que desarrolle, de modo que este primer paso (además de para pasar un buen rato, para que nos vamos a engañar...) me servirá como test para montarlo en producción. Lo que quiero en producción es crear dos máquinas virtuales, de modo que en una sola máquina "física" quiero tener un firewall (que situaré en el host), un IDS en una máquina virtual (snort) y el servidor web en otra (lo que no sé es si la arcaica máquina que tengo en mente usar lo soportará todo o tendré que montar el IDS en el host...).

Los pasos que seguiré son para Debian (SID, para ser exactos), pero como compilaré a partir de las fuentes, debería ser muy similar para las otras distribuciones.

La razón para compilar desde las fuentes es que en el momento de hacer estas pruebas la versión para SID tenía un bug (concretamente el 260111[7]) que hacía que la máquina virtual se colgara al arrancar. Sin este pequeño inconveniente, el proceso de instalación habría sido mucho más sencillo, como es habitual en Debian. Habría bastado con

```
# apt-get install user-mode-linux uml-utilities
```

y pasar a los siguientes pasos.

De todas formas, el *apt-get uml-utilities* sí es necesario si no queréis tener que compilar las utilidades manualmente porque el kernel UML requiere de algunas de estas utilidades para ciertas cosas, como por ejemplo, para la configuración de la red con dispositivos TAP.

## Instalación

Para instalar user-mode-linux, lo primero que hay que hacer es bajarse las fuentes del kernel (en mi caso el 2.4.26, el último de la rama 2.4 con patch UML disponible el día de escribir este mini-howto) y el correspondiente patch. Lo podemos hacer desde kernel.org[8] y desde la página del proyecto UML[6], respectivamente:

~/software/uml$ wget http://www.kernel.org/pub/linux/kernel/v2.4/linux-2.4.26.tar.bz2[9]
~/software/uml$ wget http://belnet.dl.sourceforge.net/sourceforge/user-mode-linux/uml-patch-2.4.26-3.bz2[10]

Y ahora sólo es necesario descomprimir el kernel, aplicarle el patch y compilarlo. Yo lo hago con las opciones por defecto para simplificar y porque el entorno que quiero no es nada del otro mundo:

~/software/uml$ tar xvfj linux-2.4.26.tar.bz2
~/software/uml$ cd linux-2.4.26
~/software/uml/linux-2.4.26$ bzcat ../uml-patch-2.4.26-3.bz2 | patch -p1
~/software/uml/linux-2.4.26$ make xconfig ARCH=um
~/software/uml/linux-2.4.26$ make linux ARCH=um

En estos momentos ya estamos en disposición de ejecutar Linux en espacio de usuario, pero nos falta el sistema de ficheros raíz (si ejecutamos *./linux* nos lo advertirá). Podemos descargar algunos desde la página de descargas de uml[11]. En mi caso usaré el sistema de ficheros de Debian, pero hay disponibles sistemas de ficheros para muchas distribuciones (UML puede ser una buena forma de probar otras distribuciones sin tener que redimensionar particiones ni poner en riesgo el sistema que ya tenéis corriendo en la máquina):

~/software/uml$ wget http://belnet.dl.sourceforge.net/sourceforge/user-mode-linux/Debian-3.0r0.ext2.bz2[12]
~/software/uml$ cd ~/uml
~/uml$ bunzip2 ../software/uml/Debian-3.0r0.ext2.bz2
~/uml$ mv ../software/uml/Debian-3.0r0.ext2 ./
~/uml$ cp ../software/uml/linux-2.4.26/linux ./

Y ahora sí que definitivamente ya podemos (o deberíamos poder) ejecutar linux dentro de linux:

~/uml$ ./linux ubd0=Debian-3.0r0.ext2 mem=32M

Si todo ha ido como debería, se han abierto 3 consolas esperando que nos identifiquemos. Podemos autenticarnos como root (sin contraseña):

```
...

Debian GNU/Linux 3.0 (none) ttys/0

(none) login: root
Last login: Sun Aug 22 20:28:35 2004 on ttys/1
Linux (none) 2.4.26 #1 ds ago 21 20:23:10 CEST 2004 i686 unknown

Most of the programs included with the Debian GNU/Linux system are
freely redistributable; the exact distribution terms for each program
are described in the individual files in /usr/share/doc/*/copyright

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
(none):~# uname -a
Linux (none) 2.4.26 #1 ds ago 21 20:23:10 CEST 2004 i686 unknown
(none):~#

...
```

De todos modos, el sistema de ficheros original nos sirve de poco, porque no se puede decir que traiga muchas aplicaciones (para comprobarlo en Debian, *dpkg -l*) y le queda poco espacio libre:

```
(none):~# df
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/ubd/0               59386     48365      7952  86% /
```

En este punto tenemos varias opciones. Podríamos crear más particiones para incrementar el espacio disponible o podemos redimensionar la partición que tenemos. Yo opto por esta segunda opción. Podéis hacerlo mediante *resize2fs* o creando una partición más grande y copiar en ella todo el contenido de la partición original.

Con *resize2fs*:

~/uml$ cp Debian-3.0r0.ext2 root_fs

(root_fs es el nombre de fichero que uml busca por defecto. De este modo podríamos ahorrarnos el argumento ubd0=...).

~/uml$ /sbin/e2fsck -f root_fs
~/uml$ /sbin/resize2fs root_fs 250M
~/uml$ /sbin/tune2fs -j -i 30 root_fs

Creando una partición desde cero:

~/uml$ dd if=/dev/zero of=root_fs.ext3 bs=1M count=250
~/uml$ /sbin/mkfs.ext3 root_fs.ext3
~/uml$ mkdir mnt
~/uml$ mkdir mnt/debian
~/uml$ mkdir mnt/root_fs
~/uml$ su
Password:
# mount -o loop Debian-3.0r0.ext2 mnt/debian
# mount -o loop root_fs.ext3 mnt/root_fs
# cp -fa mnt/debian/* mnt/root_fs
# umount mnt/debian
# umount mnt/root_fs
# exit

También podemos crear una partición de swap (recomendable en mi caso, ya que me sobra disco y me falta ram).

~/uml$ dd if=/dev/zero of=swap bs=1M count=64
~/uml$ /sbin/mkswap swap
~/uml$ ./linux ubd0=root_fs ubd1=swap mem=32M

Después de arrancar la máquina virtual tenemos que informar al nuevo sistema de que ubd1 es la partición de swap. Lo podemos hacer con *swapon* o introduciendo la entrada correspondiente en */etc/fstab* para que en sucesivos inicios del sistema ya se vea la partición desde el principio:

```
(none):~# echo /dev/ubd/1 none swap sw 0 0 >> /etc/fstab
```

## Configuración de la red

Ya tenemos una máquina virtual corriendo Linux 2.4.26 en nuestro sistema, pero sin red, es decir, estamos aislados del mundo. Y un Linux aislado... No me lo puedo imaginar. Me duele sólo con pensarlo!

Hay varias maneras de proporcionar conectividad a la instancia UML (podéis consultarlas en la sección de red[13] de la web del proyecto). Una de las más sencillas es a través de la utilidad *uml_net* (UML se encarga de crearte un dispositivo TAP y asignarle una IP simplemente con pasar un argumento a la línea de comandos), pero yo me inclino por crear un dispositivo TAP en el host y crear un bridge donde añadiré este dispositivo (que se corresponderá con eth0

de la instancia UML) y la interfaz eth0 del host. De este modo la máquina virtual tiene acceso directo a la red local. No hace falta decir que necesitáis soporte en el host tanto para *bridging* como para dispositivos *TUN/TAP*.

Pero antes de todo esto hay que dar permisos de escritura sobre el dispositivo /dev/net/tun al usuario que ejecutará UML (de otro modo no tendría acceso a la red). Para hacerlo, creo que lo mejor es asignar al dispositivo un nuevo grupo, darle permisos de escritura e ir añadiéndole todos los ususarios que tengan que ejecutar instancias UML:

```
# addgroup umlnetwork
S'està afegint el grup umlnetwork (1001)...
Fet.
# chgrp umlnetwork /dev/net/tun
# adduser umluser umlnetwork
S'està afegint l'usuari umluser al grup umlnetwork...
Fet.
# chmod g+w /dev/net/tun
```

Ahora tenemos que ejecutar los pasos necesarios para hacer todo el montaje. Lo pongo en forma de script por simplicidad (el script hay que ejecutarlo como *root* y recordad que tenéis que tener instaladas las *uml-utilities*, así como *bridge-utils*):

```
#!/bin/bash
# Inici de l'uml en mode bridge. Basat en els exemples de
#http://user-mode-linux.sourceforge.net/networking.html.
# S'ha d'executar com a root.

#carreguem el mòdul tun
modprobe tun

#creem un dispositiu per a l'usuari
tunctl -d umltap0 2>/dev/null
tunctl -u umluser -t umltap0

#afegim el bridge
ifconfig br0 down 2>/dev/null
brctl delbr br0 2>/dev/null
brctl addbr br0
ifconfig eth0 0.0.0.0 promisc up
ifconfig umltap0 0.0.0.0 promisc up
ifconfig br0 192.168.0.5 netmask 255.255.255.0 up
brctl stp br0 off
brctl setfd br0 1
brctl sethello br0 1
brctl addif br0 eth0
brctl addif br0 umltap0
route add default gw 192.168.0.1

#arranquem la instància uml
su umluser -c './linux ubd0=root_fs ubd1=swap mem=32M eth0=tuntap,umltap0'
```

Una vez dentro de la máquina virtual hay que configurar la red:

```
(none):~# ifconfig eth0 192.168.0.100 up
(none):~# ifconfig
eth0      Link encap:Ethernet  HWaddr FE:FD:C0:A8:02:64
          inet addr:192.168.0.100  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:5
```

Y con esto ya tenemos la máquina conectada con el exterior:

```
(none):~# ping -c 1 192.168.0.5
```

```
PING 192.168.0.5 (192.168.0.5): 56 data bytes
64 bytes from 192.168.0.5: icmp_seq=0 ttl=64 time=1.0 ms

--- 192.168.0.5 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.0/1.0/1.0 ms
```

De todas maneras, para ir bien deberíamos introducir las entradas pertinentes en /etc/network/interfaces de modo que cuando arranquemos la máquina virtual ya tengamos red:

```
auto lo
iface lo inet loopback

iface eth0 inet static
        address 192.168.0.100
        netmask 255.255.255.0
        gateway 192.168.0.1
```

(Para ir aún mejor, habría que poner las entradas adecuadas en /etc/resolv.conf, para resolución de nombres).

Si todo ha funcionado, ya deberíamos tener un Linux plenamente funcional y conectado a la red. Ahora es cuando (después de asignar contraseña a *root* y crear una cuenta de usuario "normal") deberíamos actualizar el sistema y empezar a instalar aplicaciones o, como un servidor, pasarnos primero de todo a SARGE o SID.

## Notas finales

Modo skas

Por razones de seguridad y rendimiento, es recomendable aplicar el patch skas al kernel del host. Yo no le he hecho porque todavía no ha salido el patch para la versión del kernel que uso actualmente y porque de momento no lo pongo en producción, donde sí que lo considero imprescindible. Para más información consultad la página sobre skas[14].

Acceso al host

Si queréis poder acceder al disco del host desde la máquina virtual, no necesitáis NFS ni FTP. Podéis hacerlo de manera muy cómoda por medio de hostfs[15], simplemente montando el sistema de ficheros como cualquier otro:

```
(none):~# mkdir /mnt/host
(none):~# mount -t hostfs none /mnt/host
(none):~# ls /mnt/host
bin   cdrom  etc     home     initrd  lost+found  mnt  proc  root-n  sys  usr  vmlinuz
boot  dev    floppy  include  lib     man         opt  root  sbin    tmp  var
(none):~#
```

Evidentmente, el acceso al sistema de ficheros del host se hará con los privilegios que tenga el usuario que ejecuta UML. De todas formas, hay entornos en los que seguramente no interese que los usuarios de la máquina virtual accedan al host (por ejemplo, en entornos de hosting). En este caso, podemos compilar el kernel UML sin soporte para *hostfs*.

Cómo compartir sistemas de ficheros

Como medida para ahorrar espacio en disco, podéis compartir sistemas de ficheros entre diferentes instancias UML a través de COW[16] (*copy-on-write*). El sistema de ficheros original será de sólo lectura (si no, las instancias UML entrarían en conflicto) y es la capa COW la que da acceso de lectura-escritura a cada una de las instancias. Esto lo hace guardando y manteniendo los cambios en un dispositivo privado de cada una. Para entendernos: cuando desde una instancia UML se escribe en el sistema de ficheros compartido, se guardan los datos en un dispositivo de la propia instancia, mientras que cuando el acceso es de lectura la información se va a buscar a los dos dispositivos (al privado y, si no está allí, al compartido).

Información de depuración

Si seguís el artículo al pie de la letra os encontraréis con un ejecutable (*./linux*) extremadamente voluminoso puesto que contendrá información de depuración. Para eliminar esta información podéis ejecutar el comando:

~/uml$ strip linux

(Gracias *ivaniclixx*).

## Enlaces

- http://user-mode-linux.sourceforge.net[17]. La página del proyecto. Es realmente completa (está TODO).
- User Mode Linux Howto[18]. También está en la página del proyecto, pero lo pongo porque realmente vale la pena.
- usermodelinux.org[19]. Grupo de usuarios de user mode linux
- Artículo interesante[20]. Del creador de la criatura.

---

**Lista de enlaces de este artículo:**

1. http://www.vmware.com
2. http://www.microsoft.com/windows/virtualpc
3. http://bochs.sourceforge.net
4. http://plex86.sourceforge.net
5. http://www.colinux.org
6. http://user-mode-linux.sourceforge.net
7. http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=260111
8. http://www.kernel.org
9. http://www.kernel.org/pub/linux/kernel/v2.4/linux-2.4.26.tar.bz2
10. http://belnet.dl.sourceforge.net/sourceforge/user-mode-linux/uml-patch-2.4.26-3.
11. http://user-mode-linux.sourceforge.net/dl-fs-sf.html
12. http://belnet.dl.sourceforge.net/sourceforge/user-mode-linux/Debian-3.0r0.ext2.b
13. http://user-mode-linux.sourceforge.net/networking.html
14. http://user-mode-linux.sourceforge.net/skas.html
15. http://user-mode-linux.sourceforge.net/hostfs.html
16. http://user-mode-linux.sourceforge.net/shared_fs.html
17. http://user-mode-linux.sourceforge.net/
18. http://user-mode-linux.sourceforge.net/UserModeLinux-HOWTO.html
19. http://usermodelinux.org
20. http://www.linux-mag.com/cgi-bin/printer.pl?issue=2001-04&am

---

E-mail del autor: phalangana _ARROBA_ yahoo.com
**Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=2094**