



# Migración en caliente en Xen usando DRBD

Jordi Prats Català

Una de las características que hacen a Xen atractivo es la migración de máquinas virtuales entre máquinas físicas sin detener su ejecución. Para realizar este proceso la memoria de la máquina *guest* a migrar es copiada entre máquinas físicas con una congelación final muy breve para su completa sincronización. Sin embargo, también requiere que los discos estén presentes en la máquina destino, por lo que hace necesario el uso de algún sistema para compartir los discos como podría ser una SAN (Storage Area Network). Esta solución requiere la compra de material adicional, por lo que para instalaciones pequeñas se tiende a buscar soluciones que usen solamente el hardware de un PC corriente emulando el resto por software.



linux@software.com.pl

**E**n este artículo se verá cómo instalar y usar DRBD para simular un disco compartido para poder hacer la migración en caliente de una máquina *guest* sin usar una SAN y así conseguir dos sistemas virtualizados de alta disponibilidad.

Se partirá de la premisa de que ya se dispone de dos sistemas independientes virtualizados con Xen con un disco libre cada uno y con LVM instalado en el sistema. Actualmente muchas distribuciones ya incluyen LVM por defecto y una instalación de Xen se puede conseguir fácilmente con distribuciones como OpenSuSE, Fedora o Debian.

Las tecnologías que se usarán para que este sistema permita, no sólo la migración en caliente de las máquinas virtuales, sino también la ampliación del disco, la repartición de las virtuales entre los dos nodos y la alta disponibilidad son:

- DRBD (Distributed Redundant Block Device), para usar dos discos locales en dos nodos como si fueran uno solo compartido entre los dos.
- CLVM (Cluster Logical Volume Manager), para la creación, ampliación y eliminación de volúmenes lógicos en un disco compartido.

- GFS2 (Global File System 2), para compartir los ficheros de configuración entre los nodos.

Con estas tecnologías se construirá un sistema compuesto por estas dos máquinas virtualizadas con Xen que se llamarán *hiro* y *ando* con un disco libre en cada una de ellas (`/dev/sdb`) que se usará para crear un disco virtual compartido con DRBD. Con este disco se creará un grupo de volúmenes con CLVM, con el que se crearán dos volúmenes lógicos. El primero (*verdurita*) será para crear una máquina virtual con Xen y el segundo (*kensei*) para almacenar los ficheros de configuración de las máquinas virtuales siendo accesibles por los dos nodos base con el sistema de ficheros GFS2.

Los ejemplos proporcionados en este artículo hacen referencia a los paquetes disponibles en Fedora, aunque tanto en OpenSuSE como en Debian existen paquetes equivalentes. Por ello, este artículo es aplicable a cualquier distribución de Linux.

## DRBD

El sistema DRBD permite que mediante el uso de la red se simule un disco compartido usando realmente dos discos

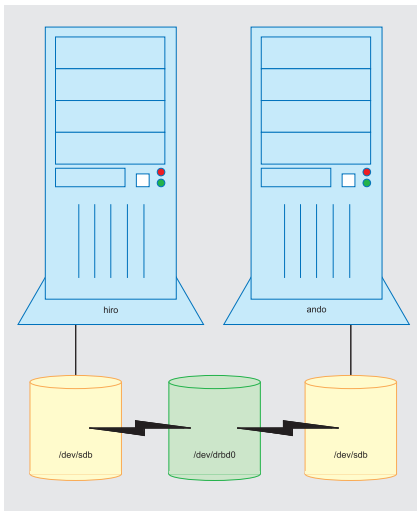


Figura 1. Disco virtual con DRBD

locales. Este software se encarga de que cada vez que se escribe un dato, se guarde tanto en el disco local como en el disco rekensei del otro nodo (ver Figura 1). Por la carga que representa la actualización constante de un disco por red, lo más recomendable es usar una tarjeta de red dedicada a esta tarea. Preferiblemente tendría que ser una GigaBit en lugar de una 10/100 Mbps ya que ésta puede acabar representando un cuello de botella si se traspasa mucha información a disco.

Para su instalación es necesario tener disponible el código fuente del kernel que se está usando. Se puede saber cuál es la versión usando el comando `uname -r`.

Con Fedora, se puede conseguir instalando el paquete `kernel-xen-devel`. Este paquete, en el caso del kernel 2.6.20-1.2952.fc6xen-i686, instalaría el código fuente en:

```
/usr/src/kernels/2.6.20-1.2952.fc6xen-i686/
```

Para la instalación del módulo DRBD y sus utilidades se tendrá que descargar el código fuente de la página web <http://www.drbd.org>. Para su instalación se deberán seguir unos sencillos pasos para compilar, instalar el módulo del kernel e instalar las herramientas:

```
make KDIR=/usr/src/kernels
  /2.6.20-1.2952.fc6xen-i686/
make install
modprobe drbd
make tools
make install-tools
```

## Configuración

Una vez instalado se tendrán que configurar los discos que se quieran usar. Se supondrá que en los dos nodos se usará el disco `/dev/sdb`

para que DRBD use estos dispositivos físicos para almacenar los datos.

En el fichero de configuración se pueden definir diversos recursos (discos compartidos) como por ejemplo, en el Listado 2, el recurso `vmdrbd`, al cual se le pueden definir varios comportamientos y opciones. A continuación se verán las más importantes.

## Protocolo de escritura

Para configurar el comportamiento de la escritura por red en los discos se permite usar uno de los siguientes protocolos. Dependiendo de la latencia de la red y la criticidad de los datos se puede escoger uno u otro:

- Protocolo A, considera escrito un dato en el momento en que se ha escrito en el disco local y ha llegado al buffer TCP local para enviarse por red para que el otro nodo lo escriba también en su disco local. Este protocolo puede ser útil en el caso que se disponga de una red con mucha latencia. Por otro lado puede provocar problemas de inconsistencia en el caso que la información no llegue nunca al otro nodo.
- Protocolo B, considera escrito un dato en el momento que se ha escrito en el disco local y además ha llegado al buffer TCP del otro nodo.
- Protocolo C, considera escrito un dato sólo cuando ha llegado a los dos discos, tanto el local como el rekensei.

## Inicialización

El tiempo que estará esperando un nodo al otro al arrancar se puede configurar en la sección `startup`. Se pueden dar dos casos:

- Si se reinician los nodos estando ambos activos, con el parámetro `wfc-timeout` se define el tiempo de espera. Por defecto se encuentra a cero. Esto se hace por si los dos están activos pero no se pueden comunicar, por ejemplo si el swith estuviera en fallo. Esto llevaría a una condición de “split-brain”, significando que los dos seguirían por su lado pensando que están solos dejando los dos discos diferentes el uno del otro.
- Con el parámetro `degr-wfc-timeout` se establece el tiempo de espera en el caso de que el nodo se haya reiniciado ya sin el otro nodo, lo que se llama un “clúster degradado” porque no contiene a todos sus miembros. Por defecto está a 2 minutos, pero se puede reducir si se considera demasiado tiempo para arrancar la máquina.

## Red

Para permitir que los dos nodos puedan usar el disco (modo primario) y no sólo estén haciendo de copia del disco 0del otro nodo (modo secundario) se tiene que añadir la opción `allow-two-primaries` en la sección `net` del recurso.

Listado 1. Configuración de ejemplo de DRBD

```
# drbd.conf
global {
  usage-count ask;
}
resource vmdrbd {
  protocol C;
  handlers {
    pri-on-incon-degr "echo o >
      /proc/sysrq-trigger ; halt -f";
    pri-lost-after-sb "echo o >
      /proc/sysrq-trigger ; halt -f";
    local-io-error "echo o >
      /proc/sysrq-trigger ; halt -f";
    outdate-peer "/usr/sbin/
      drbd-peer-outdater";
  }
  startup {
    wfc-timeout 0;
    # indefinidamente
    degr-wfc-timeout 120;
    # 2 minutos
  }
  disk {
    on-io-error detach;
  }
  net {
    allow-two-primaries;
    after-sb-0pri disconnect;
    after-sb-1pri disconnect;
    after-sb-2pri disconnect;
    rr-conflict disconnect;
  }
  syncer {
    rate 10M;
    al-extents 257;
  }
  on hiro {
    device /dev/drbd0;
    disk /dev/sdb;
    address 192.168.1.1:7788;
    flexible-meta-disk internal;
  }
  on ando {
    device /dev/drbd0;
    disk /dev/sdb;
    address 192.168.1.2:7788;
    meta-disk internal;
  }
}
```



## Nodos

Finalmente se tiene que definir qué nodos tendrán este recurso y algunos parámetros por nodo, como son:

- El dispositivo que simulará el disco compartido se define con la palabra clave `device`. En el ejemplo `/dev/drbd0`.
- El disco que se usará para la copia local, como se ha dicho anteriormente, sería `/dev/sdb` pero podrían ser discos diferentes en cada nodo o, incluso, una partición de un disco. Eso sí, tienen que tener el mismo tamaño. Para definirlo se usará la palabra clave `disk`.
- El lugar donde se guardan los metadatos se define con el parámetro `meta-disk`. Se puede usar un disco o partición en concreto dedicado o `internal` que los guarda en una zona de los discos locales.
- La IP de cada nodo y el puerto que usará DRBD (por defecto es el 7788).

El resto de parámetros que no se detallan, establecen el comportamiento del sistema en varios casos, como por ejemplo, si ocurriese un fallo de escritura en el disco local. En el caso del ejemplo se han definido dos nodos (`hiro` y `ando`) con las IPs 192.168.1.1 y 192.168.1.2 que usan el disco `/dev/sdb` con metadatos internos siendo el disco virtual `/dev/drbd0`.

## Inicialización del disco

Para poder empezar a usar el disco se tendrá que inicializar en uno de los dos nodos, por ejemplo desde `hiro`, usando el comando `drbdadm create-md vmdrbd`, levantar los servicios con `drbdadm up all` y empezar la sincronización de los dos discos con `drbdadm --overwrite-data-of-peer primary all`. Se debe tener cuidado con este último comando pues sobrescribirá los datos del segundo nodo, dejándolo igual que el disco del primero por lo que cualquier dato que esté allí se perderá. Se puede ver el progreso de la sincronización haciendo un `cat` al fichero especial `/proc/drbd`. La salida de este comando sería similar a la que se puede ver en el Listado 2. Usando el comando `watch -n1 cat`

### Listado 2. Sincronización de los discos

```
versión: 8.0.3 (api:86/proto:86)
SVN Revision: 2881 build by jprats@hiro, 2007-06-25 18:01:52
0: cs:SyncSource st:Primary/Secondary ds:UpToDate/Inconsistent C r--
ns:385952 nr:0 dw:0 dr:385952 al:0 bm:23 lo:0 pe:230 ua:0 ap:0
[=====>.....] sync'ed: 37.5% (655284/1040316)K
finish: 0:00:57 speed: 11,260 (10,404) K/sec
resync: used:0/31 hits:192722 misses:24 starving:0 dirty:0 changed:24
act_log: used:0/257 hits:0 misses:0 starving:0 dirty:0 changed:0
```

`/proc/drbd` se podrá ver cómo va avanzando segundo a segundo.

Una vez haya finalizado el proceso, desapareciendo la barra de progreso, tendremos que pasar al otro nodo (`ando`) para cambiar de modo secundario (en espera) a modo primario y, así, poder usar el disco. Esto se tiene que hacer con el comando `drbdadm primary vmdrbd`. Para la próxima vez que se inicien las máquinas se tendrá que añadir el script que se instala en `/etc/init.d/drbd` a su correspondiente `rc`. Con Fedora se puede hacer con la utilidad `chkconfig`.

## LVM y CLVM

Teniendo ya el disco compartido como `/dev/drbd0` sólo queda poderlo dividir para crear más de una máquina virtual. Para ello usaremos LVM, que normalmente ya viene en la instalación por defecto de muchas distribuciones.

Se usarán los siguientes comandos para la creación del disco: primero se tendrá que marcar como disco físico de LVM con el comando `pvcreate /dev/drbd0`, luego se tendrá que crear un grupo de volúmenes llamado, por ejemplo, `vm`. En este caso sólo estará compuesto por un disco. Esto se puede hacer con el comando `vgcreate vm /dev/drbd0` (ver Figura 2). Finalmente para crear un volumen lógico para poder usarlo como disco principal de la máquina virtual al que se llamará `verdurita`, se usará el comando `lvcreate -n verduras -L 10G vm`, que creará un disco de 10 GB llamado también `verdurita`.

Una vez hecho esto hace falta ir al otro nodo para leer la configuración del disco, esto se hace con el comando `pvscan`.

Con esta configuración LVM considera el disco como local, por lo que no contempla que otro nodo pueda modificar el disco pudiendo llegar a corromper los discos que usan LVM al hacer cambios en un grupo de volúmenes. Así para permitir hacer cambios como ampliar, crear o eliminar un volumen lógico y para que estos cambios se propaguen al resto de los nodos hace falta instalar CLVM (LVM en clúster) y la infraestructura de clúster de Red Hat llamada Cluster Manager (CMAN).

## Instalación de CMAN y CLVM2

Para la instalación de estos dos paquetes de software se puede descargar el código fuente desde <http://sourceware.org/lvm2> para CLVM y <http://source.redhat.com/cluster> para CMAN. En Fedora usando `yum install cman` para uno y `yum install lvm2-cluster` para el otro es suficiente para instalar los dos paquetes.

## Configuración CMAN y CLVM

Para la configuración del Cluster Manager se tendrá que crear el fichero `/etc/cluster/cluster.conf`. En este fichero se tienen que añadir todos los nodos del clúster y su `fence device`. Un `fence device` es un dispositivo hardware que permite la conexión y desconexión física de un nodo.

### Listado 3. Configuración del Cluster Manager

```
<?xml version="1.0"?>
<cluster alias="xencluster" config_
  version="22" name="xencluster">
<fence_daemon post_fail_delay="0"
  post_join_delay="3"/>
<clusternodes>
<clusternode name="hiro" nodeid="1"
  votes="1">
<fence>
<method name="1">
<device name="fdhiro"
  nodename="hiro"/>
</method>
</fence>
</clusternode>
<clusternode name="ando"
  nodeid="2" votes="1">
<fence>
<method name="1">
<device name="fdando"
  nodename="ando"/>
</method>
</fence>
</clusternode>
</clusternodes>
<fencedevices>
<fencedevice agent="fence_ilo"
  host="hiro.ilo"
  login="Administrator"
  passwd="iLOpassword"
  name="fdhiro"/>
<fencedevice agent="fence_ilo"
  host="ando.ilo" login="
  Administrator"
  passwd="iLOpassword"
  name="fdando"/>
</fencedevices>
<cman expected_votes=
  "1" two_node="1"/>
</cluster>
```

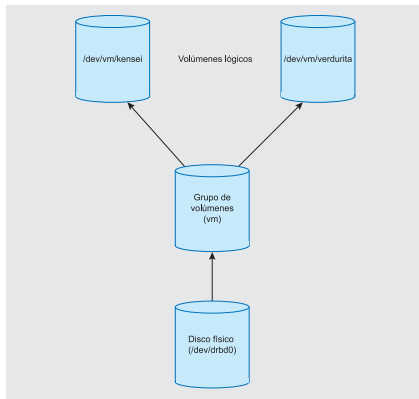


Figura 2. Volúmenes de LVM

Por ejemplo, HP integra en la mayoría de su gama de servidores ProLiant un puerto ethernet que permite acceder por red a una interfaz de administración remota, permitiendo así apagar físicamente el nodo sin estar delante de éste. En el ejemplo del Listado 3 se aprecia como están ambos configurados para usar una iLO.

También se puede usar como fence device el dispositivo ficticio fence\_manual. Éste, en lugar de realizar las acciones él mismo, escribe en el fichero /var/log/messages la acción a realizar esperando que lo haga el administrador manualmente. Para su configuración, a diferencia de la iLO, no es necesario especificar los atributos login y password. Ahora tan sólo queda arrancar el servicio Cluster Manager usando el script /etc/init.d/cman start y el servicio cluster-lvm con el script /etc/inid./clvmd start en ambos nodos y añadirlos a los scripts de inicio para poder modificar los grupos de volúmenes LVM libremente en ambos nodos.

## GFS2

Aprovechando la instalación del Cluster Manager se puede instalar GFS2 para montar, en modo lectura y escritura, un mismo sistema de ficheros en dos máquinas diferentes. Si se hiciera esto en un sistema de ficheros normal provocaría la corrupción de los datos. Así, este sistema de ficheros puede usarse para almacenar los ficheros de configuración de las máquinas virtuales para que puedan ser accesibles y modificables desde los dos nodos.

Para su instalación tan sólo hace falta descargar las utilidades de GFS2, ya que el módulo del kernel ya viene instalado en Fedora. Simplemente ejecutando yum install gf2-utils ya quedan instaladas las herramientas de GFS2.

## Creación del sistema de ficheros

Para la creación del sistema de ficheros hace falta crear un disco compartido usando el grupo de volúmenes de LVM con el comando: lvcreate -n kensei -L 5G vm, que creará el volumen lógico /dev/vm/kensei de 5 GB de espacio. En éste

se tiene que definir el sistema de ficheros con la utilidad mkfs.gfs2, donde los parámetros de esta utilidad son:

- -t: Identificador de tabla. Tiene que ser una cadena del estilo nombrecluster:nombre-sistema-ficheros. En el caso del ejemplo el nombre del clúster es “xencluster” (ver Listado 4).
- -p: Protocolo de bloqueo. Si se dispone de más de un nodo se tiene que usar lock\_dlm. Se podría usar lock\_nolock si sólo se dispone de un nodo pero se quiere usar GFS2.
- -j: Número de journals. Los journals son los registros de operaciones del sistema de ficheros. Tiene que haber al menos uno por cada nodo.

Así que finalmente se podría crear el sistema de ficheros para dos nodos usando el comando:

```
mkfs.gfs2 -t xencluster:etcxen
-p lock_dlm -j 2 /dev/vm/kensei.
```

Ahora, para poder usarlo para almacenar la configuración de las máquinas virtuales hace falta mover el contenido actual de /etc/xen al nuevo disco, por lo se tiene que montar el disco en un punto temporal: mount /dev/vm/kensei /mnt/tmp y mover los ficheros con un mv /etc/xen/\* /mnt/tmp. Al finalizar, ya se puede desmontar el punto temporal con un umount /mnt/tmp.

Finalmente, se monta el disco en su lugar definitivo con: mount /dev/vm/kensei /etc/xen. Haciendo esto en los dos nodos ya se puede empezar a crear los ficheros de configuración desde cualquiera de los dos nodos siendo los cambios visibles directamente desde ambos.

## Configuración en el arranque

Para poder montar el sistema en el momento de arranque se tiene que editar el fichero /etc/fstab y añadir la siguiente línea: /dev/vm/kensei /etc/xen gfs2 defaults 0 0.

## Xen

Finalmente, después de todos estos pasos, se configurará Xen para poder hacer la migración en caliente.

## Configuración

Para permitir la migración de máquinas virtuales se tiene que activar el parámetro (xend-relocation-server yes) en el fichero xend-config.sxp y definir a qué nodos se permite hacer la migración. Para ello se usa el parámetro xend-relocation-hosts-allow con un listado separado por espacios de expresiones regulares que



## Sobre el autor

**Jordi Prats Català** es Ingeniero en Informática por la Universitat Pompeu Fabra. Actualmente trabaja como administrador de sistemas en el Centre de Supercomputació de Catalunya encargándose de diversos clústers, entornos virtualizados y el sistema de copias de seguridad. Sus intereses son los sistemas de alta disponibilidad y balanceo de carga tanto en sistemas físicos como virtualizados y la preservación de datos digitales.

identifiquen los nodos ya sea por su IP o por su nombre. En el caso del ejemplo de este artículo sería: (xend-relocation-hosts-allow '^hiro\$ ^ando\$').

## Migración en caliente

Finalmente, una vez configurado todo y con los servicios arrancados de nuevo para que lean la nueva configuración ya se podrá realizar la primera migración con el comando: xm migrate --live verdurita ando. Éste, ejecutado desde hiro, migrará la máquina verdurita hacia ando sin apagarla. Esto, dependiendo de la tarjeta de red que estemos usando y del tamaño de la RAM, tardará más o menos tiempo. El tiempo final de sincronización, es decir, donde la máquina quedará congelada, será de 60 a 300 ms aproximadamente.

## Conclusión

Se ha visto como configurar dos máquinas para permitir la migración en caliente usando solamente hardware corriente. Esta configuración es indicada para poder mantener en alta disponibilidad una máquina virtual, ya que en caso de fallo de uno de los nodos físicos el otro puede recuperar las máquinas virtuales que han caído disminuyendo el tiempo para la recuperación del servicio, la única condición es que haya suficiente memoria RAM para mantener todas las máquinas virtuales en un solo nodo. ⚠



## En la Red

- DRBD <http://www.drbd.org>
- CLVM <http://sourceware.org/vm2>
- Cluster Manager y GFS2 <http://source.redhat.com/cluster>
- Xen <http://www.xen-source.com/xen>