novática
CEPIS UPGRADE Revista de la Asociación de Técnicos de Informática

Novática, revista fundada en 1975 y decana de la prensa informática española, es el órgano oficial de expresión y formación continua de **ATI** (Asociación de Técnicos de Informática), organización que edita también la revista **REICIS** (Revista Española de Innovación, Calidad e Ingeniería del Software). ***Novática*** co-edita asimismo **UP**GRADE, revista digital de **CEPIS** (*Council of European Professional Informatics Societies*), en lengua inglesa, y es miembro fundador de **UP**ENET (***UP****GRADE* **E***uropean* **NET***work*).

<http://www.ati.es/novatica/>
<http://www.ati.es/reicis/>
<http://www.cepis.org/upgrade>

ATI es miembro fundador de **CEPIS** (*Council of European Professional Informatics Societies*) y es representante de España en **IFIP** (*International Federation for Information Processing*); tiene un acuerdo de colaboración con **ACM** (*Association for Computing Machinery*), así como acuerdos de vinculación o colaboración con **AdaSpain**, **AI2**, **ASTIC**, **RITSI** e **Hispalinux**, junto a la que participa en **ProInnova**.

Coordinación Editorial, Redacción Central y Redacción ATI Madrid
Padilla 66, 3º, dcha., 28006 Madrid
Tlfn.914029391; fax.913093685 <novatica@ati.es>
Composición, Edición y Redacción ATI Valencia
Av. del Reino de Valencia 23, 46005 Valencia
Tlfn./fax 963330392 <secreval@ati.es>
Administración y Redacción ATI Cataluña
Via Laietana 46, ppal. 1ª. 08003 Barcelona
Tlfn.934125235; fax 934127713 <secregen@ati.es>
Redacción ATI Aragón
Lagasca 9, 3-B, 50006 Zaragoza.
Tlfn./fax 976235181 <secreara@ati.es>
Redacción ATI Andalucía <secreand@ati.es>
Redacción ATI Galicia <secregal@ati.es>
Suscripción y Ventas <http://www.ati.es/novatica/interes.html>, ATI Cataluña, ATI Madrid
Publicidad
Padilla 66, 3º, dcha., 28006 Madrid
Tlnf 914029391; fax.913093685 <novatica@ati.es>
Imprenta: Derra S.A., Juan de Austria 66, 08005 Barcelona.
Depósito legal: B 15.154-1975 -- ISSN: 0211-2124; CODEN NOVAEC
Portada: Luces feéricas - Concha Arias Pérez / © ATI
Diseño: Fernando Agresta / © ATI 2003

Nº 210, marzo-abril 2011, año XXXVII

sumario

Monografía del próximo número: "*Business Intelligence*"

Creación de un Clúster de Alta Disponibilidad con software libre

Miguel Vidal López, José Castro Luis
Grupo GSyC/Libresoft ,Universidad Rey Juan Carlos

<{mvidal,jfcastro}@libresoft.es>

Este artículo corresponde a la solución al desafío relacionado con el Software Libre que planteamos a nuestros lectores en el número 209 de ***Novática*** (enero-febrero 2011, p. 75).

Resumen: *En la primera parte de este artículo expusimos los conceptos básicos para entender y crear un clúster de Alta Disponibilidad (HA) con dos nodos en modo activo/activo y se proponía un "desafío" consistente en evitar un "split-brain". Este segundo artículo resuelve el desafío propuesto en el número anterior. Además, se detalla el despliegue del clúster HA y su gestión básica. Para ello se utiliza Linux-HA, un producto de software libre multiplataforma basado en el concepto RAS (Reliability, Availability, Serviceability).*

Palabras clave: *Alta disponibilidad, Corosync, clusters, HA, Heartbeat, High Availability, Linux.*

1. Solución al desafío

En el número anterior se proponía a los lectores un desafío consistente en exponer el esquema de red necesario para garantizar que nunca va a producirse un *split-brain* entre ambos nodos, y que por tanto no pudieran producirse escrituras simultáneas en la unidad de almacenamiento compartida.

Como solución para evitar el temido *split-brain* utilizamos una doble vía de comunicación de red para el Heartbeat, con un cable cruzado entre ambas máquinas y utilizando la red de servicio, de modo que, si se rompiese un enlace de red, el otro mantiene la comunicación (el "latido") entre los nodos del *clúster*. En el improbable caso de que se llegasen a romper ambos enlaces de forma simultánea, no habría escrituras pues los nodos no serían accesibles desde la red de servicios y el cliente de base de datos no tendría acceso al *clúster*.

El sistema de almacenamiento se importa a través del protocolo *Fibre Channel* (FCP) y cada uno de los nodos lo ve como un volumen lógico (LVM). Este sistema no permite el montaje simultáneo de un mismo volumen en dos máquinas diferentes, lo que asegura que no exista corrupción de datos por un eventual *split-brain*.

En la **figura 1** se muestra la solución descrita en forma esquematizada.

2. Montaje del *clúster*

2.1. Instalación del sistema operativo

El *clúster* gestionará dos recursos de bases de datos PostgreSQL (una en producción y otra en explotación) y el almacenamiento compartido entre ambos nodos a través de volúmenes lógicos. Se ha elegido Ubuntu 10.04 LTS en su versión Server debido a que es la distribución con los paquetes más actualizados tanto de `corosync` como de `pacemaker`.

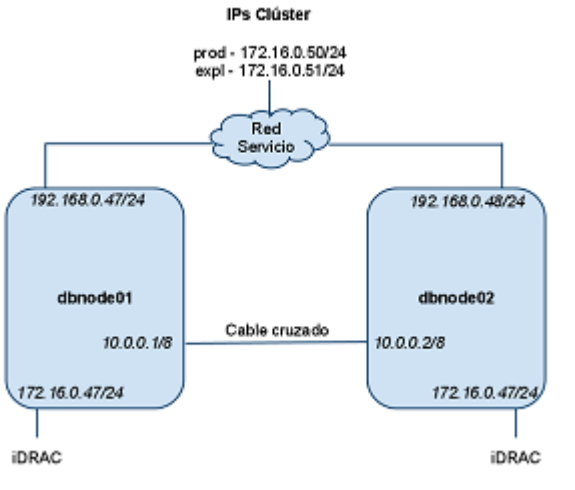

Figura 1. Esquema de la arquitectura de un *clúster* de alta disponibilidad diseñada para evitar un *split-brain*.

2.2. Instalación de paquetes

Antes de cualquier instalación se debe actualizar el sistema en ambos nodos:

```
# aptitude update
# aptitude safe-upgrade
```

Se instalan en los dos nodos los paquetes necesarios para la creación del *clúster*:

```
# apt-get install corosync pacemaker
```

2.3. Configuración de Corosync

Se crea en uno de los nodos la clave que se utilizará para la comunicación entre ellos:

```
# corosync-keygen
```

Para aumentar la entropía del sistema y generar la clave más rápidamente, se puede descargar un *kernel* y descomprimirlo (o incluso compilarlo) varias veces. Una vez creada la clave, queda almacenada en `/etc/corosync/authkey`. Se ha de copiar la clave al otro nodo y asignar los propietarios y permisos adecuados:

```
dbnode01# scp /etc/corosync/authkey dbnode02
dbnode02# mv authkey /etc/corosync/
dbnode02# chown root:root /etc/corosync/authkey
dbnode02# chmod 400 /etc/corosync/authkey
```

Para activar corosync al inicio se ha de configurar la siguiente línea en el fichero `/etc/default/corosync` de ambos nodos: `START=yes`

Se configuran las redes de latido en el fichero `/etc/corosync/corosync.conf` para cada uno de los nodos. En el caso de `dbnode01`:

```
rrp_mode: passive
interface {
# The following values need to be set based on your
environment
ringnumber: 1
bindnetaddr: 10.0.0.1 & 255.0.0.0
mcastaddr: 226.94.1.1
mcastport: 5405
}
interface {
```

```
# The following values need to be set based on your
environment
ringnumber: 0
bindnetaddr: 192.168.0.47 & 255.255.255.0
mcastaddr: 226.94.1.1
mcastport: 5407
}
```

Se inicia el proceso `corosync` en ambos nodos:

```
# /etc/init.d/corosync start
```

Hay que comprobar que la configuración del *clúster* es correcta:

```
# crm_verify -L -V
```

Los errores que arroja son debidos a que por defecto *corosync* está configurado para manejar dispositivos STONITH que aún no se han configurado. De momento, se deshabilita esa propiedad del *clúster*:

```
# crm configure property stonith-enabled=false
```

Para que funcione correctamente el *clúster* de dos nodos, se debe desactivar el *quorum*:

```
# crm configure property no-quorum-policy=ignore
```

2.4. Configuración de almacenamiento

En un único nodo, se crean los volúmenes lógicos para los datos de las bases de datos y los logs:

```
# lvcreate -L132,5G -n lvdbnodedata-prod0 vgdbnodedata
# lvcreate -L66,5G -n lvdbnodelog-prod0 vgdbnodelog
# lvcreate -L132,5G -n lvdbnodedata-expl0 vgdbnodedata
# lvcreate -L66,5G -n lvdbnodelog-expl0 vgdbnodelog
```

Se formatean los volúmenes lógicos:

```
# mkfs.ext3 /dev/vgdbnodedata/lvdbnodedata-prod0
# mkfs.ext3 /dev/vgdbnodedata/lvdbnodedata-expl0
# mkfs.ext3 /dev/vgdbnodelog/lvdbnodelog-prod0
# mkfs.ext3 /dev/vgdbnodelog/lvdbnodelog-expl0
```

Y se deshabilita el chequeo del sistema de ficheros:

```
# tune2fs -i 0 /dev/vgdbnodedata/lvdbnodedata-prod0
# tune2fs -i 0 /dev/vgdbnodedata/lvdbnodedata-expl0
# tune2fs -i 0 /dev/vgdbnodelog/lvdbnodelog-expl0
# tune2fs -i 0 /dev/vgdbnodelog/lvdbnodelog-prod0
```

En ambos nodos se crean los puntos de montaje para el almacenamiento:

```
# mkdir /var/lib/postgresql/8.4/prod
# mkdir /var/lib/postgresql/8.4/expl
# chown postgres:postgres prod/
# chown postgres:postgres expl/
# chmod 700 expl/
# chmod 700 prod/
```

Y para los logs:

```
# mkdir /var/log/postgresql/prod
# mkdir /var/log/postgresql/expl
# chown root:postgres prod/
# chown root:postgres expl/
# chmod 1777 prod/
# chmod o-w prod/
# chmod 1777 expl/
# chmod o-w expl/
```

Importante: Es necesario reiniciar el nodo donde no se han creado los LVMs para que queden visibles y accesibles.

2.5. Instalación de PostgreSQL

Hay que instalar en ambos nodos el gestor de base de datos PostgreSQL:

```
# apt-get install postgresql-8.4
```

Se para el servicio y se deshabilita del arranque para dejar que sea el *clúster* quien lo inicie o pare:

```
# /etc/init.d/postgresql-8.4 stop
# update-rc.d -f postgresql-8.4 remove
```

En el nodo dbnode01 se monta la partición de datos de la base de datos *prod*:

```
dbnode01# mount /dev/vgdbnodedata/lvdbnodedata-
prod0 \
/var/lib/postgresql/8.4/prod
```

Se crea el *clúster* PostgreSQL *prod*:

```
dbnode01# pg_createcluster 8.4 prod
```

Y se desmonta la partición de datos de la base de datos *prod*:

```
dbnode01# umount /var/lib/postgresql/8.4/prod
```

En el nodo dbnode02 se monta la partición de datos de la base de datos *expl*:

```
dbnode02# mount /dev/vgdbnodedata/lvdbnodedata-
expl0 \
/var/lib/postgresql/8.4/expl
```

Se crea el *clúster* PostgreSQL *expl*:

```
dbnode02# pg_createcluster 8.4 expl
```

Y se desmonta la partición de la base de datos *expl*:

```
dbnode02# umount /var/lib/postgresql/8.4/expl
```

Como no se utilizará el *clúster* PostgreSQL main que se instala por defecto, se elimina en ambos nodos:

```
# pg_dropcluster 8.4 main
```

2.6. Configuración de los recursos

Cada uno de los recursos que forman parte del *clúster* tiene un agente de recurso asociado que se encarga de toda la lógica del recurso concreto: inicialización, monitorización y parada. El nombre de los agentes de recurso suele seguir el esquema `ocf:heartbeat:<nombre recurso>`. La configuración de los recursos ha de hacerse tan solo en uno de los nodos del *clúster* ya que la configuración se replica automáticamente.

2.6.1. IP

Se crea el recurso *prodIP* para la IP de la base de datos *prod*:

```
# crm configure primitive prodIP
ocf:heartbeat:IPaddr2 \
params ip="172.16.0.50" cidr_netmask="24"
nic="eth0" \
op monitor interval="30" on-fail="restart"
requires="nothing"
```

Se crea el recurso *explIP* para la IP de la base de datos *expl* con un comando análogo (ip=172.16.0.51).

2.6.2. Filesystem para logs

Se crea el recurso *prodFilesystemLog* para la partición de logs de la base de datos *prod*:

```
# crm configure primitive prodFilesystemLog
ocf:heartbeat:Filesystem \
params device="/dev/vgdbnodelog/lvdbnodelog-prod0"
\
directory="/var/log/postgresql/prod/"
fstype="ext3" \
op monitor interval="10" timeout="60" on-
fail="restart" requires="nothing" \
op start interval="0" timeout="60" on-fail="restart"
\
```

```
op stop interval="0" timeout="60" on-fail="block"
```

Se crea el recurso *explFilesystemLog* para la partición de logs de la base de datos *expl* con un comando análogo.

2.6.3. Filesystem para datos

Se crea el recurso *prodFilesystemDB* para la partición de datos de la base de datos *prod*:

```
# crm configure primitive prodFilesystemDB
ocf:heartbeat:Filesystem \
params device="/dev/vgdbnodedata/lvdbnodedata-
prod0" \
directory="/var/lib/postgresql/8.4/prod"
fstype="ext3" \
op monitor interval="10" timeout="60" on-
fail="restart" requires="nothing" \
op start interval="0" timeout="60" on-fail="restart"
\
op stop interval="0" timeout="60" on-fail="block"
```

Se crea el recurso *explFilesystemDB* para la partición de datos de la base de datos *expl* con un comando análogo.

2.6.4. Base de datos PostgreSQL

Se crea el recurso *prodDatabasePG* para el proceso de PostgreSQL que ejecuta la base de datos *prod*:

```
# crm configure primitive prodDatabasePG
ocf:heartbeat:pgsql \
params pgctl="/usr/lib/postgresql/8.4/bin/pg_ctl"
psql="/usr/bin/psql" \
pgdata="/var/lib/postgresql/8.4/prod" \
pgport="5433" config="/etc/postgresql/8.4/prod/
postgresql.conf" \
logfile="/var/log/postgresql/prod/postgresql-8.4-
prod.log" \
op monitor interval="15" timeout="120" on-
fail="restart" requires="nothing" op start
interval="0" timeout="120" on-fail="restart" \
op stop interval="0" timeout="120" on-fail="block"
```

Se crea el recurso *explDatabasePG* para el proceso PostgreSQL que ejecuta la base de datos *expl* con un comando análogo.

2.7. Grupos

Los grupos se crean para simplificar la administración y para agrupar los recursos. Además, el orden de los recursos del grupo definen una relación de dependencia, de manera que el primer recurso del grupo es el primero en levantar y sucesivamente el resto hasta el último recurso del grupo. En nuestro caso, levantamos primero los sistemas de ficheros, luego la IP y por último la base de datos.

Se define el grupo *prod* para todos los recursos relacionados con la base de datos *prod*:

```
# crm configure group prod \
prodFilesystemDB prodFilesystemLog prodIP
prodDatabasePG \
meta target-role="Started"
```

Se define análogamente el grupo *expl* para todos los recursos relacionados con la base de datos *expl*.

2.8. Localizaciones

Para la localización de los recursos se opta por una política *opt-in* que previene la ejecución de los recursos en cualquier nodo en el momento del arranque por defecto:

```
# crm_attribute –attr-name symmetric-cluster –
attr-value false
```

A continuación, se le asignan puntuaciones a cada uno de los nodos para la ejecución de los recursos de manera que el *nodo01* ejecutará el grupo de recursos *prod* y el *nodo02* ejecutará el grupo de recursos *expl* siempre que estén disponibles:

```
# crm configure location cli-prefer-prod-dbnode01
prod 200: dbnode01
# crm configure location cli-prefer-prod-dbnode02
prod 0: dbnode02
# crm configure location cli-prefer-expl-dbnode01
expl 0: dbnode01
# crm configure location cli-prefer-expl-dbnode02
expl 200: dbnode02
```

2.9. Recursos stonith

Para poder configurar recursos STONITH en el *clúster*, se debe activar la propiedad *stonith-enabled* e indicar el método a utilizar por el recurso STONITH:

```
# crm configure property stonith-enabled=false
# crm configure property stonith-action=reboot
```

La configuración del recurso STONITH que apunta al nodo *dbnode01* es la siguiente:

```
# crm configure primitive dbnode01-stonith
stonith:external/ipmi \
params hostname="dbnode01" ipaddr="172.16.0.47"
userid="root" passwd="pass" \
op monitor interval="60" timeout="20"
requires="nothing" \
op start interval="0" timeout="300"
```

La configuración del recurso STONITH que apunta al nodo *dbnode02* es análoga a la anterior (ipaddr="172.16.0.48").

Como se ha explicado anteri*orme*nte, el recurso STONITH no debe correr en el nodo al que apunta, de manera que se configuran las localizaciones de la siguiente manera:

```
# crm configure location dbnode01-fencing-placement
dbnode01-stonith \
rule $id="dbnode01-fencing-placement-rule-01" inf:
#uname eq dbnode02 \
rule $id="dbnode01-fencing-placement-rule-02" -
inf: #uname eq dbnode01
Y análogamente con el dbnode02.
```

3. Gestión del *clúster*

3.1. Configuración del *clúster*

El fichero de configuración de Corosync se encuentra en `/etc/corosync/corosync.conf` de cada uno de los nodos. Es importante remarcar que este fichero ha de ser exactamente igual en todos los nodos y debe editarse con el *clúster* totalmente parado.

Por su parte, *pacemaker* no tiene fichero de configuración. La configuración es dinámica y va cambiando dependiendo de la disponibilidad de los nodos y de los servicios para adaptarse a las reglas de comportamiento establecidas.

El modo de editar la configuración se realiza a través de una interfaz de línea de comandos llamada `crm: # crm`

3.2. Modificación de recursos

Para modificar algún recurso utilizamos *crm*, modificamos la configuración con el editor por defecto del sistema y la hacemos efectiva con un *commit*:

```
# crm configure
crm(live)configure# edit
[editamos la configuraci_on]
crm(live)configure# commit
```

En caso de algún error sintáctico o utilización de valores para los *timeouts* por debajo de los recomendables, se nos informa de ello y es necesario confirmar la nueva configuración.

3.3. Migración manual de recursos

Si se desea migrar un recurso o grupos de recursos a un nodo concreto de manera manual se ha de utilizar *crm*:

```
crm(live)resource# migrate
usage: migrate (<<rsc>>) [<<node>>] [<<lifetime>>]
```

3.4. Parada de recursos

Si se desea parar la ejecución de un recurso o grupo de recursos se ha de utilizar *crm*:

```
crm(live)resource# stop
usage: stop <<rsc>>
```

3.5. Configuración de las bases de datos

La configuración de las bases de datos PostgreSQL se establece en los ficheros habituales de postgres: `/etc/postgresql/8.4/prod/postgresql.conf` en el caso de la base de datos *prod* y de manera análoga en el caso de la base de datos *expl*.

Importante: Como la configuración de las bases de datos no está replicada en los nodos, es importante que los ficheros de configuración de las bases de datos mencionados antes (y cualquier otro que se modifique) han de ser exactamente iguales en ambos nodos.

3.6. Monitorización básica por correo

Es posible configurar el comando *crm_mon* para enviar *e-mails* de aviso cada vez que se produzca un evento en el *clúster*:

```
# crm_mon –daemonize –mail-to user@example.com –
mail-host mail.example.com
```

3.7. Recuperación de red

Se puede comprobar el estado de cada uno de los "anillos" de red con el comando *corosync-cfgtool*:

```
# corosync-cfgtool -s
Printing ring status.
Local node ID 16781484
RING ID 0
id = 192.168.0.47
status = ring 0 active with no faults
RING ID 1
id = 10.0.0.1
status = ring 1 active with no faults
```

En caso de que alguna de las redes fallase, este mismo comando informaría de ello:

```
# corosync-cfgtool -s
Printing ring status.
Local node ID 16781484
RING ID 0
id = 192.168.0.47
status = ring 0 active with no faults
RING ID 1
id = 10.0.0.1
status = Marking seqid 5561 ringid 0 interface
10.0.0.1 FAULTY
- adminisrtative intervention required.
```

Importante: Una vez recuperada la red es tarea del administrador del *clúster* recuperar el interfaz en el *clúster* con el comando `corosync-cfgtool`:

```
# corosync-cfgtool -r
Re-enabling all failed rings
```

Socios institucionales de ati

Según los Estatutos de ATI, pueden ser socios institucionales de nuestra asociación "*las personas jurídicas, públicas y privadas, que lo soliciten a la Junta Directiva General y sean aceptados como tales por la misma*".

Mediante esta figura asociativa, todos los profesionales y directivos informáticos de los socios institucionales pueden gozar de los beneficios de participar en las actividades de ATI, en especial congresos, jornadas, cursos, conferencias, charlas, etc. Asimismo los socios institucionales pueden acceder en condiciones especiales a servicios ofrecidos por la asociación tales como Bolsa de Trabajo, cursos a medida, *mailings*, publicidad en Novática, servicio ATInet, etc.

Para más información dirigirse a <*info@ati.es*> o a cualquiera de las sedes de ATI. En la actualidad son socios institucionales de ATI las siguientes empresas y entidades:

AGENCIA DE INFOR. Y COMUN. COMUNIDAD DE MADRID
AGROSEGURO, S.A.
AIGÜES TER LLOBREGAT
ALC ORGANIZACIÓN Y SISTEMAS,S.L.
ALMIRALL, S.A.
AVANTTIC, CONSULTORÍA TECNOLOGICA, S.L.
CENTRO DE ESTUDIOS VELAZQUEZ S.A. (C.E. Adams)
CETICSA, CONSULTORIA Y FORMACIÓN
CONSULTORES SAYMA, S.A.
COSTAISA, S.A
DEPARTAMENT D´ENSENYAMENT DE LA GENERALITAT
ELOGOS, S. L.
EPISER, S.L.
ESPECIALIDADES ELÉCTRICAS, S.A. (ESPELSA)
ESTEVE QUÍMICA, S.A.
FUNDACIÓ BARCELONA MEDIA - UNIVERSITAT POMPEU FABRA
FUNDACIÓ CATALANA DE L´ESPLAI
FUNDACIÓ PRIVADA ESCOLES UNIVERSATÀRIES GIMBERNAT
IN2
INFORMÁTICA Y COMUNICACIONES AVANZADAS, S.L.
INSTITUT D'ESTUDIS CATALANS
INSTITUT MUNICIPAL D´INFORMÀTICA
INVERAMA
KRITER SOFTWARE, S.L.
MUSEU D´ART COMTEMPORANI DE BARCELONA - MACBA
ONDATA INTERNATIONAL, S.L.
PRACTIA CONSULTING, S.L.
QRP MANAGEMENT METHODS INTERNATIONAL
SOFTWARE, S.L.
SADIEL, S.A.
SCATI LABS, S.A.
SISTEMAS DATASIX
SISTEMAS TÉCNICOS LOTERIAS ESTADO (STL)
SOCIEDAD DE REDES ELECTRÓNICAS Y SERVICIOS, S.A.
SOGETI ESPAÑA, S.L.
SOPORTES, SISTEMAS, SOFTWARE, S.L.
SQS, S.A
TISA ORDENADORES, S.A.
TRAINING & ENTERPRISE RESOURCES
T-SYSTEMS ITC Services España S.A.
UNIVERSIDAD ANTONIO DE NEBRIJA
UNIVERSITAT DE GIRONA
UNIVERSITAT OBERTA DE CATALUNYA