

Sistema HA bajo Linux

Jose M. Suarez

Noviembre - 2002

jmsuarez@goa.es

¿Porque usar HA?

Siguiendo la ley de Murphy, "si algo puede fallar entonces fallará!", en una infraestructura donde los fallos son importantes porque se deja de dar servicio mantener ciertos servicios corriendo en varias maquinas de forma redundante ayuda a evitar estas situaciones.

HA son las siglas de High Availability o Alta Disponibilidad. Se trata de hacer un sistema redundante a un fallo de hardware (una fuente de alimentación rota, un disco duro, una tarjeta de red...). Uno de los servidores posee un **Logical Host** (a partir de ahora LH) formado por una dirección IP, un volumen de discos y un servicio asociado (HTTP, NFS, DNS...). Si este servidor tiene un problema y deja de dar servicio el otro sería capaz detectarlo y hacerse con el Logical Host.

Configuración inicial

Vamos a partir de dos maquinas de idéntica configuración (en mi caso dos Pentium-II 500Mhz / 500MB RAM) con dos interfaces de red en ambas.

Las conexiones entre las dos maquinas son: serie-serie, ethernet-ethernet con un cable cruzado

Aparte de unimos las dos maquinas a la red donde van a dar servicio mediante el otro ethernet.

Si queremos un almacenamiento compartido (mas tarde veremos su importancia) necesitaremos una controladora SCSI en cada maquina y un dispositivo de almacenamiento externo SCSI con dos salidas.

Conectamos las dos controladoras al dispositivo cuidando siempre de que **el SCSI ID de ambas controladoras sea diferente**. Para ello habrá que cambiarlo en la controladora.

Ejemplos de configuraciones

Los **servicios** más comunes que se pueden instalar en HA son los siguientes:

- HTTP
- DNS
- NFS
- Samba
- LDAP
- SMTP
- BBDD

En nuestras pruebas vamos a montar un cluster HA en **activo/pasivo** y otro en **activo/activo**:

- El **modo activo/pasivo** significa que la maquina que tiene el LH (llamada **master**) esta dando servicio mientras que la otra no da servicio y solo si se produce un **takeover** (balanceo del LH) comenzara a dar servicio. Es la configuración mas segura pero también la "mas cara" ya que en condiciones normales nuestros servicios corren en una sola maquina teniendo la otra 'parada'.
- El **modo activo/activo** se caracteriza porque ambas maquinas están dando servicio, es decir, tenemos definidos al menos dos LH uno corriendo en cada maquina. En caso de producirse un takeover una de las maquinas asumiría los dos servicios. Es la configuración mas practica ya que aprovechamos la potencia de ambos servidores, pero debemos tener siempre en cuenta que uno de los servidores tiene que poder asumir la carga de dos LH, si no tenemos esto en cuenta cuando se produzca un fallo nuestro sistema dejara de dar servicio.

El almacenamiento

Según nuestras necesidades podemos pensar en:

- **Almacenamiento individual:** Cada maquina almacena los datos del servicio en su espacio local y los tiene replicados (normalmente con rsync) en el otro nodo del cluster. Este sistema es el **mas barato** pero solo es practico cuando el volumen de datos a mantener sincronizado es muy pequeño y puede ser sincronizado sin problemas. Por ejemplo: DNS o SMTP donde solamente hay que mantener sincronizados los archivos de configuración.

- **Almacenamiento compartido:** Cada LH dispone de un volumen de disco (o discos en RAID) donde mantiene los datos de la aplicación. Cuando se produce un takeover este volumen es montado en el otro nodo del cluster y la aplicación dispone de los mismos datos. Normalmente se hace con dos tarjetas SCSI y un dispositivo de almacenamiento externo SCSI con dos entradas o también se puede hacer con dos tarjetas de fibra óptica y una red SAN. Estos volúmenes nunca pueden estar montados por los dos nodos del cluster porque se puede producir una corrupción de los datos en el almacenados si ambas maquinas tratan de escribir simultáneamente.

Existe una variante a este ultimo sistema que permite el acceso simultaneo a los volúmenes llamada [Global File System \(GFS\)](#). Pero todavía no la he probado.

Como ejemplo podemos ver diferentes configuraciones:

- Un servidor **DNS en activo/activo**, dando servicio en un nodo de DNS primario y en el otro de DNS secundario
- Un servidor **NFS (o Samba) activo/activo**, con dos LH y en cada uno de ellos un volumen de discos distinto
- Un servidor **SMTP activo/activo**, con dos LH, uno de ellos para correo saliente y el otro para correo entrante.
- Un servidor **HTTP activo/activo**

Para este ejemplo vamos a usar la configuración mas sencilla que es un cluster HA de **HTTP activo/activo**.

Ejemplo: HA HTTP activo/activo.

¿Como “hablan” los dos nodos?

Los dos nodos van a 'hablarse' a través del cable serie y de la conexión ethernet con cable cruzado que los une (doble chequeo). Esta conexión es muy importante, ya que si falla sin que a los nodos les haya ocurrido algo grave ambos se pelearan por el volumen de discos y la direccion IP. Esto se hace mediante un programa llamado **Heartbeat** (latido de corazón) que manda un pulso de un nodo del cluster al otro para indicarle que no tiene ningún problema. Heartbeat se puede descargar desde <http://www.linux-ha.org/download/> y el programa usado para

realizar el takeover de la dirección IP se llama **Fake** y esta integrado en el paquete **Heartbeat**

Preparando la instalación

Instalamos los sistemas operativos (en mi caso RedHat 8.0) y asignamos direcciones IP en ambos nodos:

Vamos a asignar una IP a cada nodo al interfaz por el que van a dar servicio (eth0):

```
nodo1 como 192.168.0.1  
nodo2 como 192.168.0.2
```

Vamos a asignar una IP a cada nodo al interfaz por el que va a correr el heartbeat (eth1):

```
nodo1 como 10.0.0.1  
nodo2 como 10.0.0.2
```

Además damos de alta dos direcciones en nuestro servidor DNS por las que vamos a dar servicio:

```
http://service1.foo.bar como 192.168.0.11  
http://service2.foo.bar como 192.168.0.12
```

Testeamos la conectividad por las tarjetas ethernet y de los puertos serie mediante:

En el nodo1:

```
nodo1# cat < /dev/ttyS0
```

En el nodo2:

```
nodo2# echo "TTY test" > /dev/ttyS0
```

Y deberíamos ser capaces de ver la salida por el nodo1.

Descarga e instalacion de HeartBeat

Después de descargar Heartbeat de <http://www.linux-ha.org/download>, lo instalamos en ambos nodos con el comando:

```
nodo1# rpm -ivh /heartbeat-0.4.9-1.i386.rpm
```

Configurando el cluster

El procedimiento de instalación está bien documentado y podemos encontrar muchos ejemplos de configuración en **/usr/share/doc/packages/heartbeat**

Solamente hay tres archivos que tenemos que configurar para poner en marcha nuestro cluster: **authkeys**, **ha.conf** y **haresources** (podemos copiarlos de

/usr/share/doc/packages/heartbeat).

Configuración de /etc/ha.d/authkeys

En este archivo se configura las claves de autenticación para que el cluster, que deben ser las mismas para los dos nodos. Se pueden usar tres tipos distintos: **crc**, **md5**, or **sha1**, dependiendo de la seguridad que necesites. Yo he elegido md5. El archivo de ejemplo podría ser:

```
# use md5 with key "mycluster"  
auth 3 3 md5 mycluster
```

Por razones de seguridad es conveniente asignar los permisos **600** a este archivo.

Configurando /etc/ha.d/ha.cf

En este archivo se definen los nodos del cluster y los interfaces que el heartbeat usa para verificar si un sistema está 'levantado' o 'caído'. Mi archivo es el siguiente:

```
# definicion de nodos del cluster  
node nodo1  
node nodo2  
  
# tiempo en segundos en el que se considera un sistema  
como 'muerto' si no responde  
deadtime 5
```

```
# Configuración del puerto serie para el heartbeat  
serial /dev/ttyS0  
baud 19200
```

```
# interfaz de red para el heartbeat  
udp eth1
```

```
# interfaz de red para el heartbeat  
udp eth1
```

Configurando /etc/ha.d/haresources

En este archivo vamos a definir los LH, para nuestro ejemplo queremos dos servidores en activo/activo y cada uno de estos servidores es master de un servicio. Por tanto el archivo /etc/ha.d/haresources será:

```
nodo1 192.168.0.11 httpd
```

```
nodo2 192.168.0.12 httpd
```

Donde se indica en el primer campo el nodo que es master, la IP de cada servicio y el servicio asociado a esta IP.

Nota: Si solo definimos un LH estaremos creando un nodo activo/pasivo

Configurando /etc/hosts

Aquí está el archivo del nodo1, pero en el nodo2 habrá que añadir líneas equivalentes:

```
127.0.0.1 nodo1 localhost.localdomain localhost  
192.168.0.2 nodo2  
192.168.0.11 service1  
192.168.0.12 service2
```

Levantando el cluster

Nos conectamos a ambos nodos y ejecutamos:

```
nodo1# /etc/rc.d/init.d/heartbeat start
```

Si se ha levantado correctamente podemos ir al nodo2 y levantarlo:

```
nodo2# /etc/rc.d/init.d/heartbeat start
```

En el momento de levantarlo veremos que se nos levanta un alias en el interfaz de red **eth0:1** con la dirección IP del servicio (192.168.1.11 y 192.168.1.12)

Probando la configuración:

Nuestro cluster HA debería ser capaz de levantarse sin problemas. Podemos conectarnos a las IPs de servicio podemos teclear en un navegador:

```
http://service1.foo.bar  
http://service2.foo.bar
```

Si paramos el servicio en alguna de las dos maquinas podremos ver como se balancea el LH a la otra. Si hacemos un 'halt' en uno de los nodos se puede comprobar que el LH pasa al otro nodo en unos segundos y no hay apenas perdida de servicio.

Enlaces

Linux HA

<http://www.linux-ha.org>

High-Availability File Server with heartbeat

<http://www.samag.com/documents/s=1146/sam0109c/0109c.htm>

High Availability systems under Linux by [Atif Ghaffar](#)

<http://www.linuxfocus.org/English/November2000/article179.shtml>

Global File System

http://www.sistina.com/products_gfs.htm