Alta Disponibilidad en Linux
Heartbeat y Pacemaker

Para conseguir la Alta Disponibilidad de nuestros servicios, se detallará como llevar a cabo la configuración de dos herramientas:

- Heartbeat: Encargado de revisar que cada nodo se halle funcionando. En caso un nodo falle migrará los recursos a otro nodo que también se halle ejecutando el servicio heartbeat
- Pacemaker: Verifica el estado de los recursos (o servicios) que le han sido asignados. En caso algún servicio falle, en caso se halla configurado, Pacemaker puede reiniciarlo.

Mientras heartbeat se encarga que revisar el estado de cada nodo; Pacemaker es el responsable de verificar el estado de los servicios que deseemos sean HA dentro de los nodos.

Configuración

Se tienen dos (2) servidores, cada uno de los cuales está conectado a un mismo dispositivo de almacenamiento externo. Cada uno de los servidores cuenta con un servicio mysqld instalado:

- mysql1 (192.168.190.189)
- mysql2 (192.168.190.190)

En ambos servidores se han instalado los paquetes heartbeat 3.x, pacemaker 1.x, cluster-glue (con sus respectivas dependencias)

El objetivo de la configuración es garantizar la disponibilidad de la base de datos en caso uno de los dos nodos falle.

Editar el archivo /etc/hosts

1. Ejecutamos el siguiente comando para obtener el nombre asignado a cada servidor.

```
uname -a
```

2. Se procede a editar el archivo /**etc**/**hosts** en ambos servidores agregando la siguiente línea:
    - Servidor mysql1

```
192.168.190.190 mysql2
```

    - Servidor mysql2

```
192.168.190.189 mysql1
```

Editar el archivo /etc/ha.d/ha.cf

En cada uno de los servidores se edita el archivo de configuración del heartbeat

```
# Logging
debug 1
use_logd false
logfacility daemon

# Misc Options
traditional_compression off
compression bz2
coredumps true

# Communications
udpport 691
# bcast eth0
ucast eth0 192.168.190.X # IP del servidor con el que se comunicará
# reemplazar X con 189 o 190 según corresponda
autojoin none
node mysql1 mysql2

# Thresholds (in seconds)
keepalive 500ms # how long between heartbeats
warntime 5 # late heartbeat warning
deadtime 10 # declare host dead
initdead 30 # first dead time > 2 * deadtime

# Enable Pacemaker
crm yes
```

Editar el archivo /etc/ha.d/authkeys

Este archivo contiene el algoritmo y la palabra secreta que será usada para establecer comunicación los diversos servidores interconectados por heartbeat.

- Seleccionar el algoritmo que se desea utilizar

```
auth 1
crc # Usar solo en desarrollo, no brinda seguridad alguna
# 2 sha1 SECRET-KEY
# 3 md5 SECRET-KEY
```

- Únicamente el usuario root debe poder modificar este archivo

```
chown root.root /etc/ha.d/authkeys
chmod 600 /etc/ha.d/authkeys
```

Iniciar Heartbeat

Para iniciar Heartbeat ejecutar el siguiente comando en cada uno de los nodos:

```
service heartbeat start
```

Configurar Pacemaker

Para configurar el Pacemaker, se hará uso de la herramienta crm.

Para verificar que nuestra configuración ha sido la adecuada y los nodos pueden verse entre sí, ejecutaremos la opción:

```
$ crm node show
```

La salida de este comando debería de ser similar a:

```
mysql1(3bda1ae0-a614-45f1-8d6c-3c9336103e74): normal
mysql2(0d00d69e-a6b6-4d11-9ecf-f1174932d8fe): normal
```

Procederemos a verificar la lista de clases y proveedores:

```
$ crm ra classes
```

La salida sería similar a:

```
heartbeat

lsb

ocf / heartbeat pacemaker <- Este es el valor que nos interesa, clase ocf y proveedor heartbeat

stonith
```

Para conocer el listado de recursos que **pacemaker** puede administrar ejecutamos:

```
$ crm ra list ocf
```

Dentro de la salida mostrada, los valores que para este caso nos interesarían son:

```
Filesystem: Recurso que permite montar y desmontar automáticamente unidades de disco.
IPaddr: Recurso que permite mover una misma IP virtual entre varios nodos.
mysql: Recurso encargado de iniciar / detener el servicio mysqld.
```

Para conocer el listado de parámetros que recibe cada uno de los recursos, es necesario ejecutar lo siguiente:

```
$ crm ra meta <nombre_del_recurso>
```

Procedemos a configurar cada uno de los servicios:

```
$ crm configure primitive <nombre del recurso> <clase>:<proveedor>:<tipo> [params
<parámetro>=<valor> [<parámetro>=<valor>...]]
```

Ejemplo:

```
$crm configure primitive Mysql-rsc ocf:heartbeat:mysql \
params datadir=/mnt/mysql log=/mnt/log/mysqld.log socket=/tmp/mysql.sock
binary="/usr/bin/mysqld_safe"
```

Configuramos el tiempo de monitoreo para cada recurso

```
$ crm configure monitor
```

Ejemplo:

```
$ crm configure monitor Mysql-rsc 10s:30s
```

Es posible modificar los valores ingresados hasta el momento (o agregar nuevos valores haciendo uso del comando:

```
$ crm configure edit
```

En este punto verificaremos que los valores relacionados a la configuración de los recursos sean similares a:

```
primitive Filesystem-rsc ocf:heartbeat:Filesystem \
params device="/dev/sdb1" directory="/mnt" fstype="ext3" \
op monitor interval="20s" timeout="40s" \
op start interval="0s" timeout="100s" \
op stop interval="0s" timeout="100s"
primitive IPaddr-rsc ocf:heartbeat:IPaddr \
params ip="192.168.190.179" \
op monitor interval="5s" timeout="20s" \
op start interval="0s" timeout="90s" \
op stop interval="0s" timeout="100s"
primitive Mysql-rsc ocf:heartbeat:mysql \
params datadir="/mnt/mysql" pid="/var/run/mysqld/mysqld.pid" log="/mnt/log/mysqld.log"
socket="/tmp/mysql.sock" binary="/usr/bin/mysqld_safe" \
op monitor interval="10s" timeout="30s" \
op start interval="0s" timeout="120s" \
op stop interval="0s" timeout="120s" \
meta target-role="Started"
```

Procedemos a definir un orden en el que se iniciarán los recursos:

```
order
```

Ejemplo:

```
$ crm configure order order-1 INFINITY: Filesystem-rsc Mysql-rsc
$ crm configure order order-2 INFINITY: IPaddr-rsc Mysql-rsc
```

Nota: Un score de tipo INFINITY significa el orden definido es de cumplimiento obligatorio.

El siguiente paso consiste en indicar en qué nodos han de iniciarse los recursos.

```
colocation <id> <score>: <rsc>[:<role>] <rsc>[:<role>]
```

Ejemplo:

```
$ crm configure colocation colocation-1 INFINITY: Mysql-rsc Filesystem-rsc
$ crm configure colocation colocation-2 INFINITY: Mysql-rsc IPaddr-rsc
```

Nota: En este caso un score de tipo INFINITY significa que ambos recursos especificados por cada colocation deben de ser iniciados en un mismo nodo. Lo que se busca es que tanto la dirección IP virtual, el montado de la unidad externa que contendrá los valores de la base da datos y el servicio mysqld sean iniciados en un mismo nodo, en caso el nodo fallara, el conjunto de recursos serían migrados al otro nodo.

Configuraremos el valor de la propiedad start-failure-is-fatal en false. Este paso es efectuado para que, en

caso de error el servicio mysqld se detuviera, este sea reiniciado automáticamente por el **Pacemaker** en lugar de ser migrado al otro nodo. Si la cantidad de fallos alcanzado por este recurso fuera igual al valor definido en migration-threshold, el conjunto de recursos serán migrados a otro nodo.

```
$ crm configure property start-failure-is-fatal=false
$ crm configure property stonith-enabled=false
```

Finalmente para guardar la configuración es necesario ejecutar

```
$ crm configure commit
```

Nota: La configuración serán replicados entre los diferentes nodos de forma automática.

Para guardar una copia de la configuración efectuada puede utilizarse el comando:

```
$ cibadmin --query > configuracion.xml
```

Para restaurarlo, puede hacerse uso de:

```
$ cibadmin --replace --xml-file configuracion.xml
```

Para mayor información pueden consultar el manual de Pacemaker: Manual de Pacemaker

---

Documento elaborado por: Dennis Cohn Muroy (Lima, Perú)
Blog: "Solo Confía..." (http://denniscm20.blogspot.com/)
16 de abril de 2010