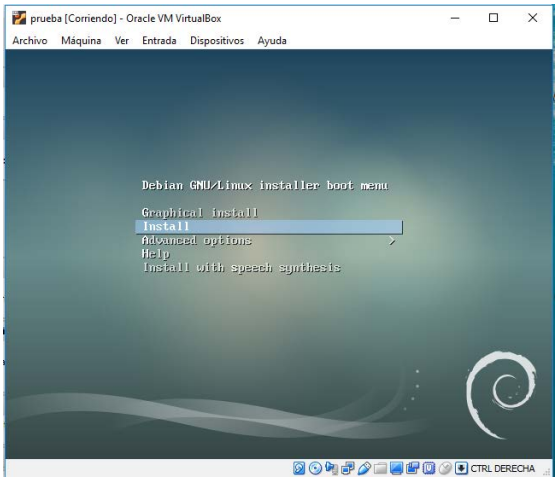
Sencillo clúster de alta disponibilidad con Pacemaker y Corosync

Se trata de configurar un clúster de alta disponibilidad activo/pasivo en el que uno de los dos equipos pueda responder siempre a la dirección IP que se pretende mantener en alta disponibilidad.

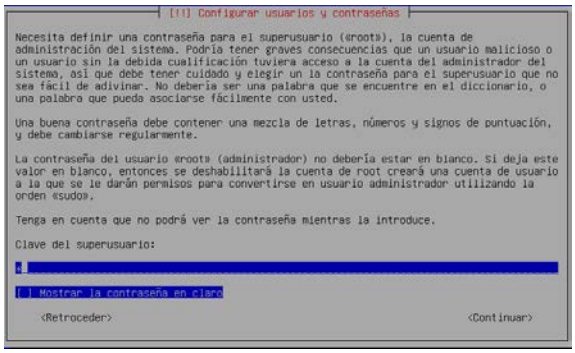
Esta situación no es totalmente real, ya que lo lógico es que esos equipos estén ofreciendo algún servicio que se quiera mantener en alta disponibilidad (ldap, http, https, etc.), sin embargo es muy útil para comprender el funcionamiento del software de HA (pacemaker y corosync) sin las configuraciones adicionales necesarias para ofrecer los servicios en HA, además esta configuración es la base de los clústeres reales.

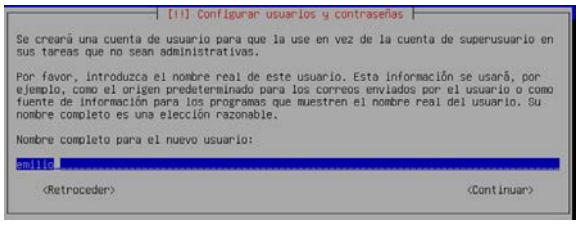
Introducción

Para llevar a cabo todo esto, necesitaremos al menos dos nodos que serán dos máquinas virtuales debían 9.6.0 (debian-9.6.0-amd64-xfce-CD-1). Os pongo alguna consideración a la hora de instalar las máquinas virtuales montadas en VirtualBox en los puntos más críticos:

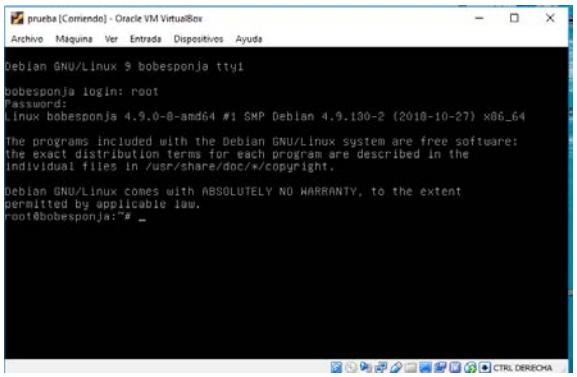


Emilio Iniesta Valero





Solo seleccionar los elementos que veis. Sin escritorios en modo consola.

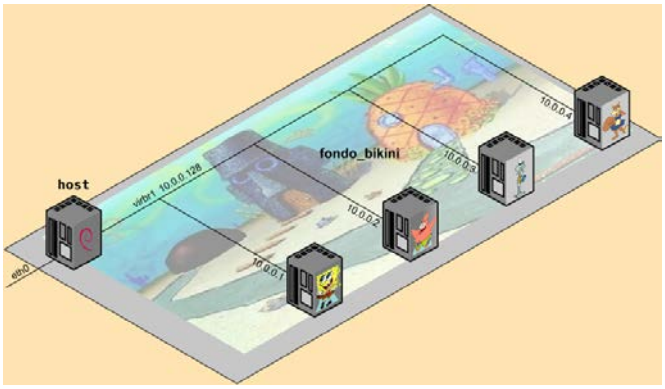


Entrar en la máquina en modo root

login: root

password: contraseña administrador

Esquema de red



Utilizaremos como nodos del clúster los equipos bobesponja que tendrá la IP 10.0.0.1 y patricio que tendrá la IP 10.0.0.2, ambos con debían 9 en modo texto y como dirección IP virtual (la que usaremos como un recurso en alta disponibilidad) utilizaremos la 10.0.0.11.

Para cambiar el nombre de una máquina (si fuese el caso de una máquina clonada) tendremos que cambiar los ficheros /*etc*/hostname y /*etc*/hosts o ejecutando

hostnamectl set-hostname nombre_nuevo

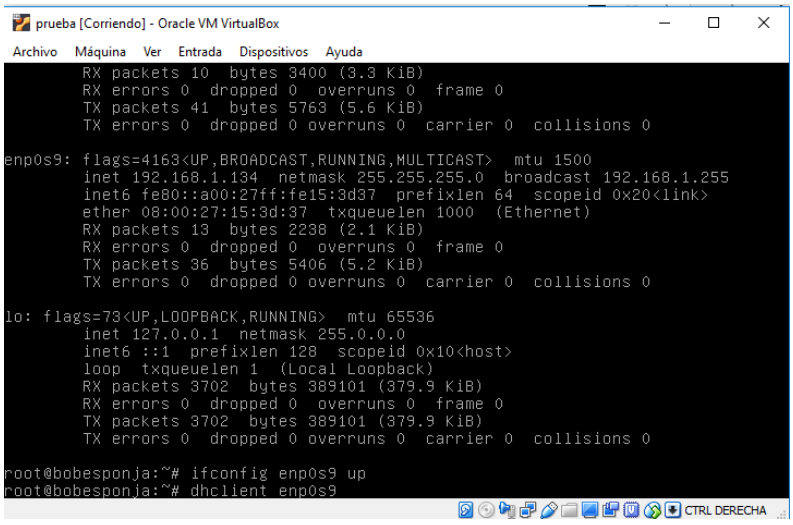
Nota: En un clúster de HA siempre es recomendable que los nodos estén interconectados por interfaces dedicadas en otro segmento de red diferente del que se utiliza para ofrecer los recursos, aunque nosotros por simplicidad utilizaremos el mismo.

Instalación de pacemaker, corosync, pcs y crmsh

En ambos nodos instalamos pacemaker, corosync, pcs y crmsh:

```
root@bobesponja:~#apt-get install pacemaker corosync pcs crmsh
root@patricio:~#apt-get install pacemaker corosync pcs crmsh
```

Si habéis puesto varias tarjetas de red, solamente aparecerá una de ellas activa con su IP desde ifconfig (para que aparezca este comando será necesario que esté instalado ya el paquete pacemaker). Las otras no están levantadas todavía, para hacerlo posible tendremos que levantarlas manualmente primero viendo cuáles son esas interfaces mediante el comando ifconfig –a y luego mediante el comando ifconfig <interfaz> up la levantamos y luego forzamos a que la interfaz levantada pida una IP al servidor DHCP con el comando dhcpclient <interfaz> como vemos en la siguiente imagen:



Creamos en bobesponja la clave de autenticación de corosync y la copiamos a patricio, dando permisos de lectura al fichero de clave en patricio:

```
root@bobesponja:~#corosync-keygen.
```

Ahora activamos la cláusula PermitRootLogin en /etc/ssh/sshd_config de la forma: PermitRootLogin yes en patricio.

```
root@bobesponja:~#scp /etc/corosync/authkey patricio:/etc/corosync/authkey
root@patricio:~#chmod 400 /etc/corosync/authkey
```

Editamos el fichero de configuración de corosync de ambos nodos /etc/corosync/corosync.conf) y añadimos la red que se va a utilizar para controlar el latido entre los nodos (en nuestro caso va a ser la misma por la que se ofrecen los recursos, pero lo habitual sería que fuese una interfaz dedicada):

bindnetaddr: 10.0.0.11

Además para que corosync se inicie de forma automática al arrancar la máquina, marcamos a "yes" el parámetro START del fichero de configuración del demonio (/etc/default/corosync) en ***ambos nodos***. Es decir, que añadamos en ese fichero START=yes y activemos las demás cláusulas que hay quitando el símbolo #.

```
  GNU nano 2.7.4            Fichero: /etc/default/corosync

# Corosync runtime directory
COROSYNC_RUN_DIR=/var/lib/corosync

# Path to corosync.conf
COROSYNC_MAIN_CONFIG_FILE=/etc/corosync/corosync.conf

# Path to authfile
COROSYNC_TOTEM_AUTHKEY_FILE=/etc/corosync/authkey

# Command line options
#OPTIONS=""
START=yes
```

Reiniciamos corosync en los dos nodos:

```
root@bobesponja:~#service corosync restart
root@patricio:~#service corosync restart
```

Si comprobamos el estado del clúster, podremos comprobar que inicialmente tenemos algo como:
`root@bobesponja:~#crm_mon`

```
root@bobesponja:~# crm_mon
============
Last updated: Sun Mar  4 08:46:15 2012
Current DC: NONE
0 Nodes configured, unknown expected votes
0 Resources configured.
============
```

Pero al cabo de unos instantes, pasamos al siguiente estado:

```
============
Last updated: Sun Mar  4 08:49:40 2012
Stack: openais
Current DC: bobesponja - partition with quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, 2 expected votes
0 Resources configured.
============

Online: [ bobesponja patricio ]
```

Donde podemos comprobar que los dos nodos se han detectado, aunque todavía no hay ningún recurso configurado.

Configuración de la IP virtual como recurso

El recurso que vamos a configurar en este ejemplo va a ser una dirección IP 10.0.0.11, para ello en primer lugar desactivamos el mecanismo de STONITH (*Shoot The Other Node In The Head*), que se utiliza para parar un nodo que esté dando problemas y así evitar un comportamiento inadecuado del clúster:

```
root@bobesponja:~# crm configure property stonith-enabled=false
```

Ahora configuramos el recurso de la IP virtual (10.0.0.11):

```
root@bobesponja:~#crm configure primitive FAILOVER-ADDR ocf:heartbeat:IPaddr2 params
ip=”10.0.0.11” nic=”enp0s3” op monitor interval=”10s” meta is-managed=”true”
```

Al monitorizar ahora el clúster, veremos que aparece el recurso FAILOVER-ADDR asociado en este momento a bobesponja:

```
root@bobesponja:~#crm_mon
```

```
============
Last updated: Sun Mar  4 09:15:33 2012
Stack: openais
Current DC: bobesponja - partition with quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, 2 expected votes
1 Resources configured.
============

Online: [ bobesponja patricio ]

FAILOVER-ADDR   (ocf::heartbeat:IPaddr2):       Started bobesponja
```

Desde un equipo de la red 10.0.0.0/24 hacemos ping a los dos nodos y a la dirección virtual 10.0.0.11 y comprobamos que la IP virtual está asociada a la dirección física del nodo activo que es (bobesponja):

```
root@host:~#ping –c 1 10.0.0.1

root@host:~#ping –c 1 10.0.0.2

root@host:~#ping –c 1 10.0.0.11

root@host:~#cat /proc/net/arp
```

```
host:~$ ping -c 1 10.0.0.1
ping -c 1 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_req=1 ttl=64 time=0.735 ms

--- 10.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.735/0.735/0.735/0.000 ms
host:~$ ping -c 1 10.0.0.11
PING 10.0.0.11 (10.0.0.11) 56(84) bytes of data.
64 bytes from 10.0.0.11: icmp_req=1 ttl=64 time=1.35 ms

--- 10.0.0.11 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.352/1.352/1.352/0.000 ms
host:~$ ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=0.667 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.667/0.667/0.667/0.000 ms
host:~$ cat /proc/net/arp
IP address       HW type     Flags       HW address            Mask
  Device
10.0.0.1         0x1         0x2         52:54:00:af:f8:9d     *
  virbr1
10.0.0.11        0x1         0x2         52:54:00:af:f8:9d     *
  virbr1
10.0.0.2         0x1         0x2         00:16:36:2d:70:4e     *
  virbr1
```

La MAC de la dirección virtual 10.0.0.11 es la misma que la MAC del nodo bobesponja 10.0.0.1

Ahora probamos el funcionamiento del clúster apagando bobesponja y monitorizamos el clúster desde patricio:

```
root@bobesponja:~#service pacemaker stop
```

```
root@patricio:~#crm_mon o crm status
```

```
============
Last updated: Sun Mar  4 09:20:54 2012
Stack: openais
Current DC: patricio - partition WITHOUT quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, 2 expected votes
1 Resources configured.
============

Online: [ patricio ]
OFFLINE: [ bobesponja ]
```

Sin embargo patricio no pasa a ofrecer el recurso directamente porque no hay cuórum en el clúster. El cuórum (*quorum*) es una propiedad que utiliza pacemaker para tomar las decisiones apropiadas mediante consultas consensuadas a todos los nodos, pero no tiene razón de ser en un clúster de solo dos nodos, ya que sólo habrá quorum cuando los dos nodos estén operativos, así que ignoramos las decisiones basadas en cuórum:

```
root@patricio:~#crm configure property no-quorum-policy=ignore
```

Podemos comprobar ahora como patricio pasa a ofrecer el recurso, aunque nos advierte que no ha habido cuórum a la hora de tomar esa decisión:

```
root@patricio:~#crm status
```

```
============
Last updated: Sun Mar  4 09:29:58 2012
Stack: openais
Current DC: patricio - partition WITHOUT quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, 2 expected votes
1 Resources configured.
============

Online: [ patricio ]
OFFLINE: [ bobesponja ]

FAILOVER-ADDR   (ocf::heartbeat:IPaddr2):       Started patricio
```

Desde el equipo externo host podemos comprobar que la dirección IP sigue respondiendo al ping, pero ahora está asociada a la dirección MAC de patricio:

```
host:~$ ping -c 1 10.0.0.11
PING 10.0.0.11 (10.0.0.11) 56(84) bytes of data.
64 bytes from 10.0.0.11: icmp_req=1 ttl=64 time=1.83 ms

--- 10.0.0.11 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.835/1.835/1.835/0.000 ms
host:~$ cat /proc/net/arp
IP address       HW type     Flags       HW address            Mask
  Device
10.0.0.11        0x1         0x2         00:16:36:2d:70:4e     *
  virbr1
10.0.0.2         0x1         0x2         00:16:36:2d:70:4e     *
  virbr1
```

Si por último, volvemos a arrancar bobesponja, éste volverá a ofrecer el recurso:

```
root@bobesponja:~#service pacemaker start
```

```
============
Last updated: Sun Mar  4 09:35:18 2012
Stack: openais
Current DC: patricio - partition with quorum
Version: 1.0.9-74392a28b7f31d7ddc86689598bd23114f58978b
2 Nodes configured, 2 expected votes
1 Resources configured.
============

Online: [ bobesponja patricio ]

FAILOVER-ADDR   (ocf::heartbeat:IPaddr2):       Started bobesponja
```

Comandos:

crm status (ver el estado del clúster)

crm resource stop FAILOVER-ADDR (parar el recurso en alta disponibilidad)

cmr configure delete FAILOVER-ADDR (quitar el servicio en alta disponibilidad)

service pacemaker stop (poner en marcha o apagar el servicio pacemaker)