Práctica 7.2: Instalación de un clúster básico para proporcionar un servidor web Apache en alta disponibilidad

El objetivo de esta actividad es diseñar y desplegar un clúster básico que ofrezca como servicio un servidor web Apache en alta disponibilidad.
Para acometer esta práctica es necesario que previamente haya realizado completamente la práctica 7.1 sobre el diseño e instalación de la infraestructura básica del clúster.

1 Introducción

Como hemos indicado el objetivo fundamental de esta actividad es diseñar y desplegar un clúster básico que ofrezca como servicio un servidor web *Apache* en alta disponibilidad.

Para ello, en la primera parte de la actividad el estudiante deberá diseñar la infraestructura básica del clúster para dar soporte a dicho servicio: número de nodos, rol de cada nodo, infraestructura de red necesaria, … (actividad práctica 7.1). Una vez diseñada y desplegada la infraestructura básica del clúster, se debe proceder a la instalación de los módulos software para el soporte de computación en clúster, poner en marcha dichos módulos software y finalmente crear un clúster.

Información más detallada se encuentra en las siguientes fuentes bibliográficas:

- *"High Availability Add-On Overview" [1]*. En este manual de Red Hat se introduce el conjunto de herramientas y componentes de Red Hat 7 que dan soporte a la computación en Clúster (denominado *High Avalilability Add-On*). Su lectura nos proporciona una visión general de los diferentes componentes del conjunto de herramientas y sus funciones.
- *"High Availability Add-On Administration" [2]*. En el capítulo 1 de este manual de administración de Red Hat se explica el proceso de despligue e instalación del conjunto de herramientas y componentes de Red Hat 7 que dan soporte a la computación en Clúster (denominado *High Avalilability Add-On)*. Además, en el capítulo 2 se explica, como ejemplo, cómo realizar el despliegue de un servidor *apache* en alta disponibilidad con *Red Hat High Avalilability Add-On*.

2 Requisitos previos

Para abordar esta práctica se debe haber completado la práctica 7.1.

3 Plan de actividades y orientaciones

1) Instalación y configuración del software *High Availability Add-On* de *Red Hat*

Paso 1: Instalar el software de soporte para la computación clúster en *Nodo1* y *Nodo2* (*High Availability Add-On*).

Paso 1.1. Instalar el software de soporte del clúster (*High Availability Add-On*) en ambos nodos.

```
# dnf install pcs pacemaker fence-agents-all
```

Paso 1.2. Configurar el cortafuegos para abrir los puertos de comunicación asociados al servicio de computación en clúster en ambos nodos.

```
# firewall-cmd --permanent --add-service=high-availability
# firewall-cmd --reload
```

Paso 2: Para poder emplear la utilidad **`pcs`** para configurar el clúster y permitir la comunicación entre los nodos, es necesario establecer una clave en los nodos 1 y 2 para el usuario **`hacluster`**, que es la cuenta de usuario de administración de la utilidad **`pcs`**. Se recomienda que la clave de dicho usuario sea la misma en cada nodo. Esta orden deberá ejecutarse en *Nodo1* y *Nodo2*.

```
# passwd hacluster
Changing password for user hacluster.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

Paso 3: Antes de poder configurar el clúster, es necesario arrancar el servicio **`pcsd`** y habilitarlo para que se ejecute en el arranque de cada nodo. Este proceso es el encargado de ejecutar las acciones establecidas por el usuario a través de la orden de administración **`pcs`** para gestionar la configuración del clúster en los nodos que lo conforman. Estas órdenes deberán ejecutarse en *Nodo1* y *Nodo2*.

```
# systemctl start pcsd.service
# systemctl enable pcsd.service
```

Paso 4: Para poder realizar este paso es necesario haber completado los pasos anteriores en ambos nodos. Antes de poder emplear la utilidad **`pcs`** para la creación del clúster, es necesario autenticar al usuario **`hacluster`** en ambos nodos mediante una opción de la orden **`pcs`**. Esta orden de autenticación se puede ejecutar en cualquiera de los nodos del clúster. En este caso se ejecutará en *Nodo1* que será el nodo desde donde se realizará la configuración del clúster. Es importante entender que estas operaciones no hay que realizarlas en todos los nodos, se realizan en un nodo y sus efectos se trasladan a los nodos indicados en la propia orden (la ejecución previa del servicio **`pcsd`** en los nodos permite

precisamente que esto sea posible). Pueden ver todas las opciones y parámetros de la orden **pcs** accediendo al manual de ayuda en línea (**man pcs**).

```
# pcs host auth nodo1.vpd.com nodo2.vpd.com
Username: hacluster
Password:
nodo1.vpd.com: Authorized
nodo2.vpd.com: Authorized
```

2) Creación del clúster

En esta fase se creará un clúster formado por los nodos **nodo1.vpd.com** y **nodo2.vpd.com**.

Paso 1: Tal y como se ha indicado en el apartado anterior, las órdenes de administración del clúster se pueden ejecutar desde cualquiera de los nodos, eligiéndose *Nodo1*. La orden para la creación del clúster (**pcs cluster setup**) deberá indicar el nombre que se le desea dar al clúster y los nodos que lo van a conformar. La ejecución de la orden implica la propagación de un fichero que almacena la configuración del clúster a todos los nodos que lo integran. Además, se empleará la opción **--start** para que arranque los servicios que dan soporte al clúster en todos los nodos.

```
# pcs cluster setup Apache --start nodo1.vpd.com \
nodo2.vpd.com
nodo1.vpd.com: Succeeded
nodo1.vpd.com: Starting Cluster...
nodo2.vpd.com: Succeeded
nodo2.vpd.com: Starting Cluster...
```

Paso 2: Habilitar los servicios que dan soporte al clúster para que arranquen cada vez que se inician los nodos sin necesidad de tener que iniciarlos manualmente (**pcs cluster start**).

```
# pcs cluster enable --all
```

Paso 3: Comprobar el estado del clúster.

```
# pcs cluster status
```

3) Establecimiento de un mecanismo de aislamiento de nodos, conocido en inglés por el término "*fencing configuration*".

Durante el funcionamiento de un clúster, los nodos pueden sufrir algún fallo de hardware o software que provoque un mal funcionamiento del nodo. En un clúster esto puede conllevar que un nodo deje de tener información actualizada del estado del clúster o pueda realizar accesos concurrentes no autorizados al almacenamiento compartido pudiendo corromper los datos, y por ello, la forma de

actuar generalmente consiste en aislar el nodo del clúster que falla para que no intervenga en los servicios ofrecidos por el clúster. Un mecanismo de aislamiento o de *fencing* persigue este objetivo: aislar el nodo en cuestión (normalmente restableciendo o apagando el nodo). A este mecanismo también se le denomina *STONITH* ("*Shoot the other node in the head*"; disparar al otro nodo en la cabeza). Existen diversos mecanismos que permiten llevar a cabo este aislamiento, por ejemplo, a través de un dispositivo físico (conmutador de alimentación) que permita apagar el nodo, emplear un *switch* inteligente que permita desconectarlo de la red o mediante un dispositivo de almacenamiento compartido al que todos los nodos tengan acceso en combinación con un vigilante.

En este caso, dado que todos los nodos son máquinas virtuales controladas por un mismo host anfitrión, se va a configurar un mecanismo de aislamiento a través de la utilidad "`fence_virtd`" que se ejecutará en el host anfitrión. De esta forma, la comunicación del host anfitrión con los nodos se realizará a través de una dirección IP *multicast*, permitiendo así la comunicación de un dispositivo origen (host anfitrión) con un grupo de dispositivos (*Nodo1* y *Nodo2*).

Paso 1: Configuración del **host anfitrión**.

Paso 1.1. Instalación de los paquetes de aislamiento o *fencing*.

```
# dnf install fence-virt fence-virtd fence-virtd-multicast \
fence-virtd-libvirt
```

Paso 1.2. Configurar el cortafuegos para permitir el tráfico multicast IP por la interface conectada a la red de control, que en este caso es `virbr2` .

```
# firewall-cmd --permanent –-zone=libvirt \
--add-rich-rule='rule family="ipv4" \
source address="10.22.132.11" accept'
# firewall-cmd --reload
# firewall-cmd --permanent –-zone=libvirt \
--add-rich-rule='rule family="ipv4" \
source address="10.22.132.12" accept'
# firewall-cmd --reload
```

Paso 1.3. Para el funcionamiento del mecanismo de *fencing* es necesario generar un fichero de configuración `/etc/fence_virt.conf` que contiene diversos parámetros: dirección multicast, puerto de comunicación, fichero que contiene una clave compartida para que sólo los nodos que dispongan de dicha clave puedan realizar peticiones de *fencing*, etc... Existe una utilidad de ayuda que permite la generación de dicho fichero a partir de las opciones que indique el usuario para cada parámetro. Para cada pregunta se puede aceptar las opciones por defecto. En la pregunta donde haya que especificar la interfaz a través de la cual llegarán las peticiones de *fencing*, indicaremos la interfaz correspondiente a la red de control (en nuestro caso supondremos que es la interfaz `virbr2`, pero podría ser otra). Al terminar se puede comprobar el contenido del fichero de configuración generado `/etc/fence_virt.conf` (ver ejemplo en la Figura 1).

```
# fence_virtd -c
```

```
backends {
        libvirt {
                uri = "qemu:///system";
        }
 }

 listeners {
        multicast {
                key_file = "/etc/cluster/fence_xvm.key";
                interface = "virbr2";
                port = "1229";
                address = "225.0.0.12";
                family = "ipv4";
        }
 }

 fence_virtd {
        backend = "libvirt";
        listener = "multicast";
        module_path = "/usr/lib64/fence-virt";
 }
```

Figura 1: Contenido del fichero `/etc/fence_virt.conf`

Paso 1.4. Crear el fichero que contiene la clave compartida en el directorio adecuado.

```
# mkdir -p /etc/cluster
# touch /etc/cluster/fence_xvm.key
# chmod 0600 /etc/cluster/fence_xvm.key
# dd if=/dev/urandom bs=512 count=1 \
of=/etc/cluster/fence_xvm.key
```

Paso 1.5. Finalmente lanzar el servicio y habilitarlo para que se lance cada vez que se inicia el host anfitrión.

```
# systemctl start fence_virtd
# systemctl enable fence_virtd
```

Paso 1.6. Comprobar la conectividad *multicast* a través de la orden `fence_xvm -o list` (ver Figura 2). En caso de que la orden no responda, reiniciar el host anfitrión.

```
[root@pc1223 ~]# fence_xvm -o list
Almacenamiento                   053daf68-d083-4005-b01b-569ba097db4e on
Nodo1                            e5aada4e-af9c-48d1-b332-f1d771ecf2e8 on
Nodo2                            fe5561d1-1366-4187-841c-49dadd89afa9 on
[root@pc1223 ~]#
```

Figura 2: Salida de la orden `fence_xvm -o list` una vez instalado el mecanismo de *fencing* `fence_xvm` en el host anfitrión.

Paso 2: Configuración de los **nodos 1 y 2**.

Paso 2.1. Instalación de los paquetes de aislamiento o *fencing* **en ambos nodos**.

```
# dnf install fence-virt fence-virtd
```

Paso 2.2. Configurar el cortafuegos **en ambos nodos** para permitir el tráfico de comunicación con el host anfitrión a través de la red de control (`10.22.132.1`) y del puerto `1229` (que ha sido el puerto establecido en el fichero de configuración).

```
# firewall-cmd --permanent --add-rich-rule='rule family="ipv4" \
source address="10.22.132.1" port port="1229" protocol="tcp" \
accept'
# firewall-cmd --reload
```

Paso 2.3. Crear el directorio administrativo `/etc/cluster` y copiar el fichero que contiene la clave compartida que emplean los nodos para realizar las peticiones de *fencing* **en ambos nodos**.

```
# mkdir -p /etc/cluster
# scp root@10.22.132.1:/etc/cluster/fence_xvm.key /etc/cluster
```

Paso 2.4. Comprobar la conectividad *multicast* a través de la orden `fence_xvm -o list` (ver Figura 2). En caso de que la orden no responda, reiniciar los nodos.

Paso 2.5. Finalmente, crear el mecanismo de *fencing* de tipo **`fence_xvm`** que hemos configurado en el clúster a través de la orden **`pcs stonith create` en uno de los nodos** (el primer parámetro es el identificador o nombre que le damos, en nuestro caso *xvmfence*).

```
# pcs stonith create xvmfence fence_xvm \
key_file=/etc/cluster/fence_xvm.key
```

Reiniciar ambos nodos y comprobar el estado del mecanismo de *fencing* mediante la orden **`pcs stonith`** y el estado global del clúster empleando la orden **`pcs status.`**

4) Configuración del servicio `httpd` para su ejecución en un clúster.

En esta sección se describen las acciones a realizar para que el servicio `httpd` se ejecute en un contexto de un clúster de alta disponibilidad. Estas acciones se realizan en dos pasos, que como se verá implican modificar archivos de configuración de este servicio.

Paso 1. Este paso se realiza en los dos nodos del clúster y se lleva a cabo para que el agente del clúster responsable de controlar el servicio `httpd` pueda obtener la información de estado (registros de "log") de este servicio. Consiste en añadir al archivo `/etc/httpd/conf.d/status.conf` las siguientes líneas:

```
<Location /server-status>
```

```
    SetHandler server-status
    Require local
</Location>
```

Paso 2. Este paso se realiza en todos los nodos del clúster que pueden ejecutar el servicio `httpd`, que en nuestro caso son *Nodo1 y Nodo2*. El propósito de este paso en configurar cada uno de estos nodos para que el control de este servicio lo realice el agente del clúster responsable de le ejecución de este servicio y no lo realice el servicio `systemd`. Para ello, se debe proceder la siguiente manera en los dos nodos.

En el archivo de configuración /etc/logrotate.d/httpd eliminar la siguiente línea:

```
/bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
```

A continuación, reemplazar la línea eliminada por las siguientes líneas:

```
/usr/bin/test -f /var/run/httpd-Website.pid >/dev/null 2>/dev/null &&
/usr/bin/ps -q $(/usr/bin/cat /var/run/httpd-Website.pid) >/dev/null 2>/dev/null &&
/usr/sbin/httpd -f /etc/httpd/conf/httpd.conf -c "PidFile /var/run/httpd-Website.pid"
-k graceful > /dev/null 2>/dev/null || true
```

5) Configurar los nodos de clúster para que el espacio de almacenamiento compartido sea no sea controlado localmente en cada nodo del clúster.

La finalidad de este paso es configurar el sistema para que el grupo de volúmenes de almacenamiento que alberga el volumen lógico en el que se despliega el espacio compartido sea controlado por solo por el clúster y no localmente por el servicio lvm de cada nodo. Esto se hace excluyendo en la variable de configuración `volume_list`, existente en el fichero de configuración local de cada nodo `/etc/lvm.conf`, el grupo de volúmenes en el que está ubicado el volumen lógico que alberga el espacio compartido.

La secuencia de pasos que se especifican a continuación se debe realizar en ambos nodos del clúster (Nodo1 y Nodo2).

Paso 1. Obtener los grupos de volúmenes que están configurados para que sean controlados localmente por el servicio lvm.

```
# vgs --noheadings -o vg_name
```

Paso 2. A partir del resultado de la orden anterior, especificar en la variable de configuración `volume_list` del archivo `/etc/lvm/lvm.conf` la relación de grupos de volúmenes exceptuando el grupo de volumen en el que se ha creado el volumen lógico que alberga el espacio compartido para el servicio Apache. Si no existieran grupo de volúmenes a especificar, entonces esta variable debería definirse conteniendo una lista vacía.

Paso 3. Una vez ejecutado el paso anterior en ambos nodos, reiniciar ambos nodos.

Paso 6. Una vez reiniciados los nodos, verificar que los servicios de control del clúster se están ejecutando correctamente.

```
# pcs cluster status
```

Si la orden anterior produjera un error consistente en que el clúster no se está ejecutando, entonces ejecutar la siguiente orden para iniciar la ejecución de clúster:

```
# pcs cluster start
```

Alternativamente, se podrían reiniciar todos los nodos del clúster y, una vez todos los nodos estuvieran arrancados, entonces ejecutar en todos los nodos la orden:

```
# pcs cluster start --all
```

6) Creación de los recursos y grupos de recursos del clúster.

Para poder ofrecer el servicio Apache en alta disponibilidad, el clúster deberá disponer de cuatro recursos.

- El primero de ellos es un recurso de tipo *LVM* que se deberá llamar *Apache_LVM*.
- El segundo es un recurso de tipo Filesystem que se deberá llamar *Apavhe_FS.*
- El tercero es un recurso de tipo `Ipaddr2`, que deberá llamarse *Apache_IP* y se corresponde con una dirección IP flotante. Mediante esta dirección los clientes harán uso del servicio y estará vinculada al nodo que esté ofreciendo el servicio en cada momento.
- El cuarto recurso es un recurso de tipo *apache* y se deberá llamarse *Apache_Script*. Este recurso será el responsable de controlar el servicio `httpd` desde el propio clúster

Estos cuatro recursos conformarán lo que se denomina un grupo de recursos (el grupo que contiene todos los recursos necesarios para ofrecer el servicio) y se identificará con el nombre `Apachegroup`. Los recursos del grupo de recursos se crean en el orden en que es necesario arrancarlos para poder ofrecer el servicio. Además, el grupo de recursos se crea con la creación del primer recurso que se añada al grupo. Seguidamente se muestran las cuatro órdenes que debe ejecutar para crear cada uno de estos recursos.

```
# pcs resource create Apache_LVM LVM volgrpname=ApacheVG \
exclusive=true --group apachegroup
# pcs resource create Apache_FS Filesystem \
device="/dev/ApacheVG/ApacheLV" directory="/var/www" \
fstype="xfs" --group apachegroup
# pcs resource create Apache_IP IPaddr2 ip=192.168.140.253 \
cidr_netmask=24  --group apachegroup
# pcs resource create Apache_Script apache \
```

```
configfile="/etc/httpd/conf/httpd.conf" \
statusurl="http://127.0.0.1/server-status" --group apachegroup
```

En relación con el recurso de tipo `VirtualIP IPaddr2` se está asumiendo que la dirección IP elegida no está en uso, Tenga en cuenta que esta IP pertenece a una red que tiene activo un servicio DHCP y, por tanto, estrictamente hablando habría que garantiza que esta dirección no es asignable por dicho servicio. Este aspecto no hay que afrontarlo en la práctica.
Para obtener final el resultado de la ejecución de estas cuatro órdenes ejecutar la orden:

```
# pcs resource show
```

Seguidamente se muestra la orden que se ha de ejecutar para obtener el estado de clúster. También se muestra una salida muy similar a la que debería proporcionar en el caso de todo se halla ejecutado correctamente:

```
# pcs status
Cluster name: Apache
Cluster Summary:
  * Stack: corosync (Pacemaker is running)
  * Current DC: nodo1.vpd.com (version 2.1.7-4.fc39-0d5cb5b) - partition
with quorum
  * Last updated: Wed Mar 20 12:39:32 2024 on nodo1.vpd.com
  * Last change:  Wed Mar 20 10:36:35 2024 by root via root on nodo1.vpd.com
  * 2 nodes configured
  * 5 resource instances configured

Node List:
  * Online: [ nodo1.vpd.com nodo2.vpd.com ]

Full List of Resources:
  * Resource Group: Apachegroup:
    * Apache_LVM (ocf::heartbeat:LVM):             Started nodo1.vpd.com
    * Apache_FS (ocf::heartbeat:Filesystem):       Started nodo1.vpd.com
    * Apache_IP (ocf::heartbeat:IPaddr2):          Started nodo1.vpd.com
    * Apache_Script (ocf::heartbeat:apache):      Started nodo1.vpd.com
  * xvmfence (stonith:fence_xvm):                  Started nodo2.vpd.com

Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

7) Validación del funcionamiento del clúster.

Una vez que el clúster está configurado y funcionando, debe ser posible acceder al servidor web desde el navegador del host anfitrión accediendo a la dirección pública del servicio, en nuestro caso, **`192.168.122.253`**. Además, puede comprobar qué nodo está dando el servicio en cada momento mediante la orden **`pcs status`**. También puede mostrar la configuración de red en cada nodo (**`ip addr show`**) y analizar cuál de ellos tiene la IP flotante asignada.

Finalmente, puede realizar alguna comprobación que conlleve afectar al nodo que está dando el servicio para que quede inoperativo y ver que efectivamente el servicio sigue estando disponible a través del segundo nodo del clúster. Esto lo podemos hacer aprovechando el mecanismo de *fencing*, aislando el nodo mediante la orden **`pcs node standby nodo1.vpd.com`** (para volver a dejar el nodo en un estado operativo es necesario ejecutar la orden contraria, **`pcs node unstandby nodo1.vpd.com)`** o bien directamente parando el nodo (**`poweroff`**).

Checklist

Cuando finalice las tareas, los profesores de la asignatura realizarán las siguientes comprobaciones:

- ☐ Verificación de las características básicas de configuración de las distintas máquinas que intervienen: interfaces de red, configuraciones de estas y comunicación entre las distintas máquinas utilizando estas interfaces.
- ☐ Verificación del servicio NFS. Configuración del servidor y de los clientes, así como que la exportación del servidor como el montaje de los clientes se realiza correctamente.
- ☐ Verificación que los servicios que proporcionan alta disponibilidad están correctamente instalados y configurados.
- ☐ Para el caso del servicio web Apache propuesto, que se ejecuta el ciclo de servicio correctamente: se inicia automáticamente con el arranque de los nodos y que se sigue proporcionando cuando el nodo que lo suministra cae.

Bibliografía

[1] Steven L. *Red Hat Enterprise Linux 7. High Avalilability Add-On Overview. Overview of the components of the High Availability Add-On*, Red Hat; 2018. Disponible en: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/ [accedido el 01/09/2020]

[2] Steven L. *Red Hat Enterprise Linux 7. High Avalilability Add-On Administration. Configuring Red Hat High Availability deployments*, Red Hat; 2018. Disponible en: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/ [accedido el 01/09/2020]