Centro de Procesamiento de Datos

Práctica 5. Creación de máquinas virtual con Vagrant y despliegue de almacenamiento redundante con GlusterFS.

Objetivo:
Crear un entorno basado en tres máquinas virtuales con Vagrant (+ VirtualBox) e instalar un sistema de ficheros GlusterFS.

Presentar un documento pdf en SWAD→Actividades→ Práctica 5 con la siguiente información:

-(obligatorio): A partir del apartado relacionado con GlusterFS realizar diversas capturas de pantalla correspondiente al proceso de cada apartado (y varias capturas del apartado de comprobación). En la captura debe aparecer algún elemento que personalice dicha captura (ej, si estamos en un escritorio y accedemos por ssh se ve la ventana de ssh y se ve parte del fondo de escritorio de forma que cada estudiante muestre su propia captura).

- (opcional): En el ejercicio 2 del apartado I puede comprobar como incluir procesos automáticos SHELL en la creación de la máquina virtual (Provisionamiento Vagrant). Modifique el fichero Vagrant de forma que se instale automáticamente el GlusterFS en cada nodo servidor. Puede utilizar otros modos de aprovisionamiento:
https://www.vagrantup.com/docs/provisioning/basic_usage.html
https://www.vagrantup.com/docs/provisioning/

Desarrollo:

En esta práctica estudiamos cómo automatizar la creación de máquinas virtuales con Vagrant y VirtualBox, creando un escenario con 3 máquinas virtuales. Sobre estas máquinas instalamos dos servidores GlusterFS trabajando en modo replicado y accediendo desde un cliente.

I) Creación de máquinas virtuales con Vagrant y VirtualBox

Instalamos Vagrant. La página principal es: https://www.vagrantup.com/intro/index.html

Vagrant tiene preconfiguradas ciertas imágenes de máquinas virtuales lo que permite desplegar un sistema rápidamente sin necesidad de instalar imágenes ISO. Podemos encontrar Vagrant Boxes en: https://app.vagrantup.com/boxes/search

Órdenes básicas para Vagrant:

-vagrant init <nombre_box> : Crea un fichero Vagrant file con la configuración básica

-vagrant up: Inicia máquinas

-vagrant ssh: accede por ssh

-vagrant halt: para máquina

-vagrant global-status: ver máquinas

-vagrant destroy : para y borra las máquinas virtuales

Cuando creamos una máquina virtual, el directorio /vagrant concide con el directorio donde tenemos el fichero Vagrantfile

Órdenes relacionadas con Boxes:

vagrant box list

vagrant box remove <box>

Ejercicio 1. Crear una máquina virtual basada en Ubuntu bionic64 (Ubuntu 18.04)

Creamos un directorio va1 y creamos el fichero de configuración con

```
vagrant init  ubuntu/bionic64
```

Iniciamos la máquina:

```
vagrant up
```

Accedemos a la máquina:

```
vagrant ssh
```

Comprobamos el acceso al directorio /vagrant

Podemos observar que el directorio /vagrant coincide con el directorio del host donde hemos creado la máquina virtual.

Ejercicio 2. Instalación de las máquinas virtuales CentOS y Ubuntu.

Creamos un directorio va2 y creamos el fichero Vagrantfile

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure(2) do |config|

  config.vm.define :centos do |centos_config|
     centos_config.vm.box = "centos/7"
     centos_config.vm.hostname = "centos.vm"
     centos_config.vm.network "private_network" , ip:"192.168.56.11"
     centos_config.vm.provider :virtualbox do |vb|
        vb.name = "centos"
        vb.customize ["modifyvm", :id, "--memory", "512"]
        vb.customize ["modifyvm", :id, "--cpus", "1"]
     end
  end

  config.vm.define :ubuntu do |ubuntu_config|
     ubuntu_config.vm.box = "ubuntu/xenial64"
     ubuntu_config.vm.hostname = "ubuntu.vm"
     ubuntu_config.vm.network "private_network" , ip:"192.168.56.12"
     ubuntu_config.vm.provider :virtualbox do |vb|
        vb.name = "ubuntu"
        vb.customize ["modifyvm", :id, "--memory", "512"]
        vb.customize ["modifyvm", :id, "--cpus", "1"]
     end
     ubuntu_config.vm.provision "shell", inline: <<-SHELL
       export DEBIAN_FRONTEND=noninteractive
       curl -s https://install.zerotier.com/ | bash
       zerotier-cli join a09acf02334c0cde
     SHELL
  end
end
```

Ejecutamos:

```
vagrant up
```

En la máquina *ubuntu* con xenial64 (ubuntu 16.04) el usuario es *ubuntu*.

Para cambiar el password

```
sudo passwd ubuntu
```

En la máquina centos, usuario *vagrant*, password *vagrant*

El acceso con password está desactivado por defecto en /etc/ssh/sshd_config

Para activarlo: *PasswordAuthentication no*

II) Destrucción de las máquinas virtuales

Antes de avanzar con el siguiente apartado nos aseguramos que hemos borrado las máquinas anteriores

Borramos las máquinas
Nos situamos en el directorio va1 y destruimos la máquinas

```
vagrant destroy
```

Igualmente en el directorio va2 para destruir las máquinas anteriores

```
vagrant destroy
```

Borramos la caché de imagen de ubuntu y dejamos sólo la de Centos7.

```
vagrant box remove <box>
```

GlusterFS

GlusterFS es un sistema de almacenamiento distribuido que permite utilizar los recursos de almacenamiento de diversos nodos para crear un sistema de ficheros redundante y escalable.

Esto permite tanto añadir nuevos nodos de almacenamiento como poder reemplazar algún nodo si falla.

En esta práctica contamos con 3 nodos virtuales: 2 nodos actúan de servidores de almacenamiento (centos1 y centos2) y el 3er nodo (centos3) actúa de cliente.

Crearemos una unidad "virtual" compuesta por los discos de los dos servidores en un modelo redundante.

III) Instalación del plugin vagrant

Instalamos el plugin vagrant-hostmanager. Este plugin modifica automáticamente los ficheros /*etc/hosts* de las máquinas virtuales.

```
vagrant plugin install vagrant-hostmanager
```

IV) Creación de las máquinas

Creamos una carpeta llamada cpd5 e insertamos el fichero Vagrantfile que se encuentra en el fichero vagranfile5.zip en SWAD.

Para iniciar y crear las máquinas virtuales ejecutamos

```
vagrant up
```

Para acceder a la máquina centos1

```
vagrant ssh centos1
```

V)Instalación de glusterFS

Comprobamos la versión disponible (Long Term Stable)

```
yum search centos-release-gluster
```

Instalamos la última versión en los nodos: centos1 y centos2

```
yum -y install centos-release-gluster312
yum -y update
yum -y install glusterfs glusterfs-cli glusterfs-libs glusterfs-server
```

VI) Iniciamos el servicio

```
systemctl enable glusterd.service
systemctl start glusterd.service
```

Probamos la conexión (desde centos1)

```
gluster peer probe centos2
gluster peer status
```

VII) Creación de los bricks

Creamos las particiones basadas en XFS y utilizando volúmenes lógicos.

Creamos una partición en /dev/sdb del tipo Linux LVM

```
fdisk /dev/sdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
```

```
Be careful before using the write command.

Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0xe785a287.

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-20971519, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-20971519, default 20971519):
Using default value 20971519
Partition 1 of type Linux and of size 10 GiB is set

Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): 8e
Changed type of partition 'Linux' to 'Linux LVM'

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Actualizamos las particiones del SO con:

```
partprobe
```

Creamos los volúmenes físicos, lógicos y la partición XFS

```
pvcreate /dev/sdb1
vgcreate vg01 /dev/sdb1
lvcreate -l 100%FREE -n lv01 vg01
mkfs.xfs /dev/mapper/vg01-lv01
```

Creamos el punto de montaje

```
mkdir -p /gluster/bricks/brick1
```

Editamos /etc/fstab y añadimos

```
/dev/mapper/vg01-lv01    /gluster/bricks/brick1  xfs     defaults  0 0
```

Montamos todas las particiones

```
mount -a
```

VIII) Creamos el FS

Probamos la conexión

```
gluster peer probe centos2
gluster peer status
gluster pool list
```

Creamos el directorio vol1 en centos1 y centos2

```
mkdir /gluster/bricks/brick1/vol1
```

Creamos los sistemas de ficheros en las unidades */dev/sdb*

```
gluster volume create glustervol1 replica 2 transport tcp centos1:/gluster/bricks/brick1/vol1 centos2:/gluster/bricks/brick1/vol1

gluster volume start glustervol1
```

Mostramos la información del volumen creado

```
gluster volume info glustervol1
```

IX) Instalación del cliente en centos3

Instalamos la última versión en el nodo centos3

```
yum -y install centos-release-gluster312
yum -y update
yum -y install glusterfs-cli glusterfs-fuse
```

Creamos un directorio como punto de montaje

```
mkdir /gdatos1
```

```
mount -t glusterfs centos1:/glustervol1 /gdatos1
```

X) Comprobación

-Creamos ficheros y directorios en /gdatos1

-Comprobamos en cada servidor que los ficheros generados en centos3 están compartidos

```
ls -la /gluster/bricks/brick1/vol1/
```

Paramos centos1

```
shutdown -h now
```

-Creamos nuevos ficheros en /gdatos1

Tras unos segundos, detecta el fallo del nodo1 y continua funcionando sólo con el nodo2.

-Volvemos a levantar el nodo1

Podemos comprobar que el nodo centos1 se resincroniza viendo el directorio /gluster/bricks/brick1/vol1