Proyecto: clúster de alta disponibilidad con Pacemaker y Corosync

Resumen

Este proyecto consiste en la creación de un clúster de alta disponibilidad con dos nodos Linux utilizando Pacemaker y Corosync. El objetivo es compartir un recurso (un servicio web) entre ambos nodos, con gestión automática de failover en caso de que uno de los nodos falle. La solución está diseñada para garantizar la continuidad del servicio y la alta disponibilidad, permitiendo que un nodo asuma la carga del otro de manera automática.

Objetivos

Crear un clúster de alta disponibilidad con 2 nodos Linux.

Configurar Pacemaker y Corosync para la gestión de los recursos y failover.

Implementar un servicio web (Apache) como recurso compartido entre los nodos.

Garantizar la sincronización horaria y la red común entre los nodos del clúster.

Realizar una simulación de failover para verificar la alta disponibilidad del clúster.

Tecnologías Utilizadas

Sistemas Operativos: Ubuntu Zorin OS en los dos nodos

Herramientas: Pacemaker, Corosync, Apache, Chrony

Redes: Configuración de IP flotante (VIP)

Sincronización Horaria: Chrony

---

1. Configuración inicial en ambos nodos

1. Asignar un nombre único a cada nodo:

   hostnamectl set-hostname nodo1

   hostnamectl set-hostname nodo2 (en el otro nodo)

2. Editar el archivo /etc/hosts en ambos nodos:

   192.168.1.76 nodo1
   192.168.1.200 nodo2

3. Habilitar el firewall y abrir los puertos necesarios:

   sudo ufw allow 5404,5405,2224,3121,21064/tcp
   sudo ufw allow 5404,5405,2224,3121,21064/udp

4. Instalar y habilitar la sincronización horaria:

   sudo apt install chrony -y
   sudo systemctl enable chrony --now

2. Instalar Pacemaker, Corosync y demás herramientas

1. Instalar los paquetes necesarios:

   sudo apt update
   sudo apt install pacemaker corosync pcs crmsh resource-agents -y

3. Configurar Corosync

1. Editar el archivo de configuración de Corosync (/etc/corosync/corosync.conf) en ambos nodos con la siguiente configuración:

```
totem {
  version: 2
  secauth: on
  cluster_name: cluster1
  transport: udpu
  interface {
    ringnumber: 0
    bindnetaddr: 192.168.1.0
    mcastport: 5405
  }
}
nodelist {
  node {
    ring0_addr: nodo1
    nodeid: 1
  }
  node {
    ring0_addr: nodo2
    nodeid: 2
  }
}
quorum {
  provider: corosync_votequorum
  two_node: 1
}
logging {
  to_logfile: yes
  logfile: /var/log/corosync/corosync.log
  to_syslog: yes
}
```

2. Copiar este archivo de configuración al segundo nodo con scp:

   scp /etc/corosync/corosync.conf nodo2:/etc/corosync/

4. Habilitar y configurar el clúster

1. Habilitar los servicios necesarios en ambos nodos:

   sudo systemctl enable pcsd --now
   sudo systemctl enable corosync --now
   sudo systemctl enable pacemaker --now

2. Establecer una contraseña para el usuario hacluster:

   sudo passwd hacluster

3. Autenticación entre nodos:

   pcs host auth nodo1 nodo2 -u hacluster -p TU_PASSWORD

4. Crear y arrancar el clúster:

   pcs cluster setup cluster1 nodo1 nodo2
   pcs cluster start --all
   pcs cluster enable --all

5. Verificar y Ajustar el Estado del Clúster

1. Verificar el estado del clúster:

   pcs status

2. Ajusta las propiedades del clúster:

   pcs property set stonith-enabled=false
   pcs property set no-quorum-policy=ignore

6. Crear la IP Flotante y el Recurso Web

1. Instalar Apache en ambos nodos:

   sudo apt install apache2 -y

2. Crear el recurso de IP flotante:

   pcs resource create vip IPaddr2 ip=192.168.1.50 cidr_netmask=24 op monitor interval=30s

3. Crear el recurso web:

   pcs resource create apache ocf:heartbeat:apache configfile=/etc/apache2/apache2.conf statusurl="http://localhost" op monitor interval=30s

4. Crear un grupo para agrupar los recursos:

   pcs resource group add webgroup vip apache

5. Verificar el estado de los recursos:

   pcs status resources

7. Simular Failover

1. Por ejemplo deteniendo corosync en cualquiera de los nodos

   sudo systemctl stop corosync

2. Verificamos que el recurso se mueva automáticamente al otro nodo.

Lecciones Aprendidas

Alta Disponibilidad: El proyecto me permitió entender cómo implementar un sistema que asegura la disponibilidad continua de un servicio en caso de fallo de un nodo. Aprendí a utilizar herramientas clave como Pacemaker y Corosync para gestionar el failover.

Gestión de Clústeres: Adquirí experiencia práctica en la creación y administración de clústeres de alta disponibilidad, incluyendo la configuración de nodos, la sincronización horaria, y la gestión de recursos.

Resolución de Problemas: Durante la implementación, enfrenté varios desafíos, como la configuración de red y la integración de los servicios. Estos problemas me enseñaron a trabajar con herramientas de diagnóstico y a resolver conflictos en un entorno distribuido.
