**High Availability using Corosync + Pacemaker on Ubuntu 16.04**

Corosync is an open source program that provides cluster membership and messaging capabilities, often referred to as the messaging layer, to client servers.

Pacemaker is an open source cluster resource manager (CRM), a system that coordinates resources and services that are managed and made highly available by a cluster. In essence, Corosync enables servers to communicate as a cluster, while Pacemaker provides the ability to control how the cluster behaves.

All commands should be run with root privileges.

**Prerequisites**:

On two servers, run those commands, select the same timezone on both servers and see the “NTP synchronized: yes” line with timedatectl command.

# apt-get update

# dpkg-reconfigure tzdata

# apt-get -y install ntp

# timedatectl

Corosync uses UDP transport between ports 5404, 5405 and 5406 . If you are running a firewall, ensure that communication on those ports are allowed between the servers.

# ufw allow 5404, 5405, 5406

Or

# iptables -A INPUT -i eth1 -p udp -m multiport --dports 5404,5405,5406 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

# iptables -A OUTPUT -o eth1 -p udp -m multiport --sports 5404,5405,5406 -m conntrack --ctstate ESTABLISHED -j ACCEPT

**Installation:**

On both servers install pacemaker. Corosync is dependency of corosync.

# apt-get install pacemaker

On Server A:

# apt-get install haveged

# corosync-keygen

# scp /etc/corosync/authkey root@server\_B\_ip:/etc/corosync/

On Server B:

# chown root: /etc/corosync/authkey

# chmod 400 /etc/corosync/authkey

On both servers:

# vi /etc/corosync/corosync.conf

totem {

version: 2

cluster\_name: my\_new\_cluster

transport: udpu

interface {

ringnumber: 0

bindnetaddr: **private\_binding\_IP\_address**

broadcast: yes

mcastport: 5405

}

}

quorum {

provider: corosync\_votequorum

two\_node: 1

}

nodelist {

node {

ring0\_addr: **server\_A\_private\_IP\_address**

name: primary

nodeid: 1

}

node {

ring0\_addr: **server\_B\_private\_IP\_address**

name: secondary

nodeid: 2

}

}

logging {

to\_logfile: yes

logfile: /var/log/corosync/corosync.log

to\_syslog: yes

timestamp: on

}

If private IP configuration is like this:

Server A : 192.168.1.101

Server B : 192.168.1.102

Cluster IP : 192.168.1.100

Then;

private\_binding\_IP\_address : 192.168.1.255

server\_A\_private\_IP\_address : 192.168.1.101

server\_B\_private\_IP\_address : 192.168.1.102

On both servers:

# mkdir -p /etc/corosync/service.d

# vi /etc/corosync/service.d/pcmk

service {

name: pacemaker

ver: 1

}

# vi /etc/default/corosync

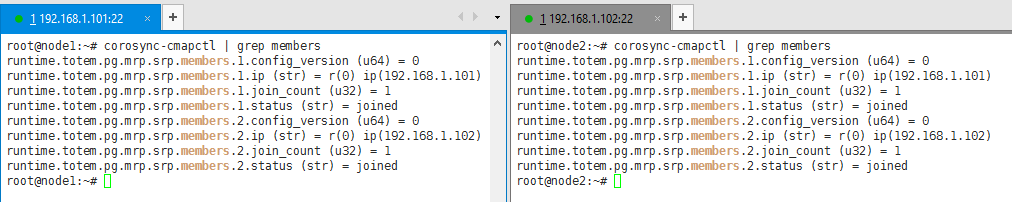
START=yes

# service corosync start

# service corosync restart

# corosync-cmapctl | grep members

You should see both servers’ private IP addresses near the strings r(0) ip(….) like this:



On both servers:

# update-rc.d pacemaker defaults 20 01

# service pacemaker start

# crm status

On Server A:

# crm configure property stonith-enabled=false

# crm configure property no-quorum-policy=ignore

# crm configure primitive virtual\_public\_ip ocf:heartbeat:IPaddr2 params ip="q.w.e.r" cidr\_netmask="32" op monitor interval="10s" meta migration-threshold="2" failure-timeout="60s" resource-stickiness="100"

Replace “q.w.e.r” with cluster IP address for example: 192.168.1.100

**Verify and Management:**

# crm status

# ip addr list

You should see cluster IP address on one server.

Change of primary node to secondary and secondary to primary:

You can use these commands manually on both servers.

# crm node standby secondary

# crm node online primary

# crm node standby primary

# crm node online secondary

Or run the script below in order to float the cluster IP on the servers without any packet loss.

#!/bin/bash

# Private IP address of the server is assumed to have static IP address configuration.

# Public IP address of the server is assumed to have dAnsweramic IP address configuration with dhcp.

Static\_Interface\_Name=`cat /etc/network/interfaces | grep iface | grep -v lo | grep static | awk '{print $2}'`

Public\_Interface\_Name=`cat /etc/network/interfaces | grep -v lo | grep auto | grep -v $Static\_Interface\_Name | awk '{print $2}'`

Node\_Static\_IP=`ifconfig | grep -A1 $Static\_Interface\_Name | tail -1 | awk '{print $2}' | cut -d':' -f 2`

Cluster\_IP\_Exists=`ip addr list | grep inet | grep -v inet6 | grep -v "127.0.0.1" | grep -v "$Node\_Static\_IP" | grep -v $Public\_Interface\_Name | awk '{print $2}' | cut -d'/' -f 1 | wc -l`

Cluster\_IP=`ip addr list | grep inet | grep -v inet6 | grep -v "127.0.0.1" | grep -v "$Node\_Static\_IP" | grep -v $Public\_Interface\_Name | awk '{print $2}' | cut -d'/' -f 1`

CorosAnswerc\_Application\_Full\_Path=`which corosAnswerc-cmapctl`

Crm\_Application\_Full\_Path=`which crm`

Cluster\_Member\_Count=`$CorosAnswerc\_Application\_Full\_Path | grep members | cut -d'.' -f 7 | uniq | wc -l`

Cluster\_Members=`$CorosAnswerc\_Application\_Full\_Path | grep members | grep "r(0)" | cut -d' ' -f 5 | cut -d'(' -f 2 | tr -d ')'`

while true

do

if [ $Cluster\_IP\_Exists -eq 1 ]

then

echo "THIS NODE IS PRIMARY. This node has cluster IP Address: "$Cluster\_IP

echo "Cluster has "$Cluster\_Member\_Count" members. Cluster Members: "$Cluster\_Members

echo

read -p "Do you want to change primary node? [yY/nN] " Answer

Flag=1

else

echo "THIS NODE IS NOT PRIMARY."

echo "Cluster has "$Cluster\_Member\_Count" members. Cluster Members: "$Cluster\_Members

echo

read -p "Do you want to change primary node? [yY/nN] " Answer

Flag=2

fi

case $Answer in

[Yy]\*)

if [ $Flag -eq 1 ]

then

$Crm\_Application\_Full\_Path node standby primary

$Crm\_Application\_Full\_Path node online secondary

Cluster\_IP\_Exists=`ip addr list | grep inet | grep -v inet6 | grep -v "127.0.0.1" | grep -v "$Node\_Static\_IP" | grep -v $Public\_Interface\_Name | awk '{print $2}' | cut -d'/' -f 1 | wc -l`

if [ $Cluster\_IP\_Exists -eq 0 ]

then

echo "Success. Check for cluster IP on the other node with 'ip addr list' command."

else

echo "Fail!..."

fi

else

$Crm\_Application\_Full\_Path node standby secondary

$Crm\_Application\_Full\_Path node online primary

Cluster\_IP\_Exists=`ip addr list | grep inet | grep -v inet6 | grep -v "127.0.0.1" | grep -v "$Node\_Static\_IP" | grep -v $Public\_Interface\_Name | awk '{print $2}' | cut -d'/' -f 1 | wc -l`

if [ $Cluster\_IP\_Exists -eq 1 ]

then

echo "Fail!..."

else

echo "Success. Check for cluster IP on this node with 'ip addr list' command."

fi

fi

break

;;

[Nn]\*)

echo "No changes have been done."

break

;;

\*)

echo "Please answer y/Y or n/N."

;;

esac

done