How to achieve load balancer high availability in CCS

Contents

[Objective 1](#_Toc400372391)

[Overview 1](#_Toc400372392)

[Assumption 2](#_Toc400372393)

[Introduction to Keepalived 2](#_Toc400372394)

[Implementation 2](#_Toc400372395)

[Install HAProxy and Keepalived 2](#_Toc400372396)

[Bind the virtual IP 3](#_Toc400372397)

[Create a neutron port in CCS for the virtual IP 3](#_Toc400372398)

[Assign a floating IP to the VIP port 3](#_Toc400372399)

[Allow VRRP and VIP traffic 3](#_Toc400372400)

[Configure the Keepalived 3](#_Toc400372401)

[Master 4](#_Toc400372402)

[Slave 4](#_Toc400372403)

[Test and validation 5](#_Toc400372404)

# Objective

This document describes a way to achieve high availability for software load balancer.

# Overview

The idea is to have 2 load balancers share one virtual IP. Depending on the host and process status, the virtual IP can be switched (failover) to the other host. We will use HAProxy and Keepalived to achieve that.

Keepalived

HAProxy

Keepalived

HAProxy

HTTP Server

HTTP Server

HTTP Server

HTTP Server

The above graph shows a typical structure.

# Assumption

This document focus on setting up the HA feature. We assume that the reader knows how HAProxy works.

We will present the sample Keepalived configurations. You may need to customize them before using in production.

# Introduction to Keepalived

Keepalived is an open source implementation of Virtual Router Redundancy Protocol (VRRP). In short, it can dynamically assign a predefined IP address (virtual IP) to a server in a predefined pool.

# Implementation

## Install HAProxy and Keepalived

Boot 2 instances and install the packages.

## Bind the virtual IP

In /etc/sysctl.conf of both servers, define the following line:

net.ipv4.ip\_nonlocal\_bind=1

Then, run this command:

sysctl –p

## Create a neutron port in CCS for the virtual IP

neutron port-create --fixed-ip ip\_address=${vip\_ip} --security-group $secgroup-vrrp $net\_id

* $vip\_ip: the IP address used as virtual IP
* $secgroup-vrrp: the security group used on this port
* $net\_id: the network ID of the load balancers on

## Assign a floating IP to the VIP port

neutron floatingip-create --port-id=$(vip\_port\_id) public-floating-601

* $vip\_port\_id: the port ID of the port we just created

This floating IP will be used to access the active load balancer. When a failover occurred, the floating IP will be switched to the other load balancer instance. The external user will continue to access the same floating IP.

## Allow VRRP and VIP traffic

By default, neutron ports only accept traffic to its only IP. However, VRRP depends on multicast. We need to update the ports on the 2 load balancer instances to accept VRRP traffic and traffic targeting the VIP.

Run this command on both instances.

neutron port-update $server\_port --allowed-address-pairs type=dict list=true ip\_address=${vip\_ip} ip\_address=224.0.0.18

* $server\_port: the port ID of the load balancer instance
* $vip\_ip: the IP address used as virtual IP

# Configure the Keepalived

In this section, we present the Keepalived sample configures on the 2 load balancer instance. Pay attention to the highlighted parts, you may need to change them.

The configuration is very easy to understand. It runs a simple script to see if the haproxy process is still there. If the haproxy process is gone, it will trigger a failover. If the Master node is down, it will also trigger a failover.

## Master

**global\_defs {**

**router\_id ub3 # can be host name**

**}**

**vrrp\_script haproxy {**

**script "killall -0 haproxy"**

**interval 2**

**weight 2**

**}**

**vrrp\_instance 50 {**

**virtual\_router\_id 50**

**advert\_int 1**

**priority 101 #priority higher than SLAVE**

**state MASTER**

**interface eth0**

**virtual\_ipaddress {**

**10.100.0.200 dev eth0 # virutal IP**

**}**

**track\_script {**

**haproxy**

**}**

**}**

## Slave

**global\_defs {**

**router\_id ub4**

**}**

**vrrp\_script haproxy {**

**script "killall -0 haproxy"**

**interval 2**

**weight 2**

**}**

**vrrp\_instance 50 {**

**virtual\_router\_id 50**

**advert\_int 1**

**priority 100**

**state SLAVE**

**interface eth0**

**virtual\_ipaddress {**

**10.100.0.200 dev eth0**

**}**

**track\_script {**

**haproxy**

**}**

**}**

# Test and validation

* Which server got the VIP?

The Master node should get the VIP IP, in our example, 10.100.0.200. If you found the IP on both of your instances, you may have a problem with multicast. Make sure the “neutron port-update” command run successfully.

* Listen to the multicast

Run this command,

tcpdump -v -i eth0 host 224.0.0.18

You should get the VRRP advertisement like this:

22:13:57.181678 IP (tos 0xc0, ttl 255, id 3543, offset 0, flags [none], proto VRRP (112), length 40)

host-10-100-0-11.cisco.com > vrrp.mcast.net: vrrp host-10-100-0-11.cisco.com > vrrp.mcast.net: VRRPv2, Advertisement, vrid 50, prio 103, authtype none, intvl 1s, length 20, addrs: host-10-100-0-200.cisco.com

If you don’t receive them, double check your neutron port, make sure the keepalived is running.

* Test failover

You can stop the haproxy on the Master to trigger the failover. The VIP IP should show up on the Slave node. Then, restarting the haproxy on Master, the VIP IP should switch back.

You can also shut down the Master node or the interface of the Master node and check if the load balancer is still available. It should failover to the Slave node, and load balancer should always be accessible.