By Falko Timme

Published: 2007-11-02 19:53

Setting Up A High-Availability Load Balancer (With Failover and Session Support) With Pound/Keepalived On Debian Etch

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 10/24/2007

This article explains how to set up a two-node load balancer in an active/passive configuration with **Pound** and keepalived on Debian Etch. The load balancer sits between the user and two (or more) backend Apache web servers that hold the same content. Not only does the load balancer distribute the requests to the two backend Apache servers, it also checks the health of the backend servers. If one of them is down, all requests will automatically be redirected to the remaining backend server. In addition to that, the two load balancer nodes monitor each other using keepalived, and if the master fails, the slave becomes the master, which means the users will not notice any disruption of the service. Pound is session-aware, which means you can use it with any web application that makes use of sessions (such as forums, shopping carts, etc.).

From the Pound web site: " ***The Pound program is a reverse proxy, load balancer and HTTPS front-end for Web server(s). Pound was developed to enable distributing the load among several Web-servers and to allow for a convenient SSL wrapper for those Web servers that do not offer it natively. Pound is distributed under the GPL - no warranty, it's free to use, copy and give away.*** "

I do not issue any guarantee that this will work for you!

1 Preliminary Note

In this tutorial I will use the following hosts:

- Load Balancer 1: *lb1.example.com*, IP address: *192.168.0.100*
- Load Balancer 2: *lb2.example.com*, IP address: *192.168.0.101*
- Web Server 1: *http1.example.com*, IP address: *192.168.0.102*
- Web Server 2: *http2.example.com*, IP address: *192.168.0.103*

- We also need a virtual IP address that floats between *lb1* and *lb2*: *192.168.0.99*

Here's a little diagram that shows our setup:

```
    shared IP=192.168.0.99
 192.168.0.100  192.168.0.101 192.168.0.102 192.168.0.103
 -------+------------+-------------+-----------+----------
        |            |             |           |
     +--+--+      +--+--+      +----+----+ +----+----+
     | lb1 |      | lb2 |      |  http1  | |  http2  |
     +-----+      +-----+      +---------+ +---------+
     pound        pound        2 web servers (Apache)
     keepalived   keepalived
```

The shared (virtual) IP address is no problem as long as you're in your own LAN where you can assign IP addresses as you like. However, if you want to use this setup with public IP addresses, you need to find a hoster where you can rent two servers (the load balancer nodes) in the same subnet; you can then use a free IP address in this subnet for the virtual IP address. Here in Germany, ***Hetzner*** is a hoster that allows you to do this - just talk to them.

http1 and *http2* are standard Debian Etch Apache setups with the document root */var/www* (the configuration of this default vhost is stored in */etc/apache2/sites-available/default*). If your document root differs, you might have to adjust this guide a bit.

To demonstrate the session-awareness of Pound, I'm assuming that the web application that is installed on *http1* and *http2* uses the session id *JSESSIONID* .

2 Preparing The Backend Web Servers

Pound works as a transparent proxy, i.e., it will pass on the original user's IP address in a field called *X-Forwarded-For* to the backend web servers. Of course, the backend web servers should log the original user's IP address in their access logs instead of the IP addresses of our load balancers. Therefore we must modify the *LogFormat* line in */etc/apache2/apache2.conf* and replace *%h* with *%{X-Forwarded-For}i*:

http1/http2:

```
vi /etc/apache2/apache2.conf
```

```
[...]
#LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%{X-Forwarded-For}i %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
[...]
```

Afterwards we restart Apache:

```
/etc/init.d/apache2 restart
```

We are finished already with the backend servers; the rest of the configuration happens on the two load balancer nodes.

3 Installing Pound

We can install Pound on our two load balancers like this:

lb1/lb2:

```
apt-get install pound
```

4 Configuring The Load Balancers

The Pound configuration is stored in `/etc/pound/pound.cfg` and is pretty straight-forward.

We back up the original `/etc/pound/pound.cfg` and create a new one like this:

lb1/lb2:

```
cp /etc/pound/pound.cfg /etc/pound/pound.cfg_orig

cat /dev/null > /etc/pound/pound.cfg

vi /etc/pound/pound.cfg
```

```
## Minimal sample pound.cfg
##
## see pound(8) for details


######################################################################
## global options:

User            "www-data"
Group           "www-data"
#RootJail       "/chroot/pound"

## Logging: (goes to syslog by default)
##      0       no logging
##      1       normal
##      2       extended
##      3       Apache-style (common log format)
LogLevel        1

## check backend every X secs:
Alive           2

## use hardware-accelleration card supported by openssl(1):
#SSLEngine      "<hw>"
```

```
####################################################################################
## listen, redirect and ... to:

## redirect all requests on port 80 ("ListenHTTP") to the virtual IP address:
ListenHTTP
        Address 192.168.0.99
        Port    80
End

Service
        BackEnd
                Address http1.example.com
                Port    80
        End
        BackEnd
                Address http2.exaample.com
                Port 80
        End
        Session
                Type Cookie
                ID   "JSESSIONID"
                TTL  300
        END
End
```

Afterwards, we set *startup* to *1* in */etc/default/pound*:

```
vi /etc/default/pound
```

```
# Defaults for pound initscript
# sourced by /etc/init.d/pound
# installed at /etc/default/pound by the maintainer scripts

# prevent startup with default configuration
# set the below varible to 1 in order to allow pound to start
startup=1
```

5 Setting Up keepalived

We've just configured Pound to listen on the virtual IP address *192.168.0.99*, but someone has to tell *lb1* and *lb2* that they should listen on that IP address. This is done by keepalived which we install like this:

lb1/lb2:

```
apt-get install keepalived
```

To allow Pound to bind to the shared IP address, we add the following line to */etc/sysctl.conf*:

```
vi /etc/sysctl.conf
```

```
[...]
net.ipv4.ip_nonlocal_bind=1
```

... and run:

```
sysctl -p
```

Next we must configure keepalived (this is done through the configuration file `/etc/keepalived/keepalived.conf`). I want `lb1` to be the active (or master) load balancer, so we use this configuration on `lb1`:

lb1:

```
vi /etc/keepalived/keepalived.conf
```

```
vrrp_script chk_pound {           # Requires keepalived-1.1.13
        script "killall -0 pound"     # cheaper than pidof
        interval 2                    # check every 2 seconds
        weight 2                      # add 2 points of prio if OK
}

vrrp_instance VI_1 {
        interface eth0
        state MASTER
        virtual_router_id 51
        priority 101                  # 101 on master, 100 on backup
        virtual_ipaddress {
            192.168.0.99
        }
        track_script {
            chk_pound
        }
}
```

(It is important that you use priority 101 in the above file - this makes lb1 the master!)

Then we start keepalived on `lb1`:

lb1:

```
/etc/init.d/keepalived start
```

Then run:

lb1:

```
ip addr sh eth0
```

... and you should find that `lb1` is now listening on the shared IP address, too:

```
lb1:/etc/keepalived# ip addr sh eth0
2: eth0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0c:29:a5:5b:93 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.100/24 brd 192.168.0.255 scope global eth0
    inet 192.168.0.99/32 scope global eth0
    inet6 fe80::20c:29ff:fea5:5b93/64 scope link
       valid_lft forever preferred_lft forever
lb1:/etc/keepalived#
```

Now we do almost the same on `lb2`. There's one small, but important difference - we use `priority 100` instead of `priority 101` in `/etc/keepalived/keepalived.conf` which makes `lb2` the passive (slave or hot-standby) load balancer:

lb2:

```
vi /etc/keepalived/keepalived.conf
```

```
vrrp_script chk_pound {           # Requires keepalived-1.1.13
```

```
    script "killall -0 pound"     # cheaper than pidof
    interval 2                    # check every 2 seconds
    weight 2                      # add 2 points of prio if OK
}

vrrp_instance VI_1 {
    interface eth0
    state MASTER
    virtual_router_id 51
    priority 100                  # 101 on master, 100 on backup
    virtual_ipaddress {
        192.168.0.99
    }
    track_script {
        chk_pound
    }
}
```

Then we start keepalived:

lb2:

```
/etc/init.d/keepalived start
```

As `lb2` is the passive load balancer, it should not be listening on the virtual IP address as long as `lb1` is up. We can check that with:

lb2:

```
ip addr sh eth0
```

The output should look like this:

```
lb2:~# ip addr sh eth0
2: eth0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0c:29:e0:78:92 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.101/24 brd 192.168.0.255 scope global eth0
    inet6 fe80::20c:29ff:fee0:7892/64 scope link
       valid_lft forever preferred_lft forever
lb2:~#
```

6 Starting Pound

Now we can start Pound:

lb1/lb2:

```
/etc/init.d/pound start
```

7 Testing

Our high-availability load balancer is now up and running.

You can now make HTTP requests to the virtual IP address `192.168.0.99` (or to any domain/hostname that is pointing to the virtual IP address), and you should get content from the backend web servers.

You can test its high-availability/failover capabilities by switching off one backend web server - the load balancer should then redirect all requests to the remaining backend web server. Afterwards, switch off the active load balancer (`lb1`) - `lb2` should take over immediately. You can check that by running:

lb2:

```
ip addr sh eth0
```

You should now see the virtual IP address in the output on *lb2*:

```
lb2:~# ip addr sh eth0
2: eth0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0c:29:e0:78:92 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.101/24 brd 192.168.0.255 scope global eth0
    inet 192.168.0.99/32 scope global eth0
    inet6 fe80::20c:29ff:fee0:7892/64 scope link
       valid_lft forever preferred_lft forever
lb2:~#
```

When *lb1* comes up again, it will take over the master role again.

8 Links

- Pound: **http://www.apsis.ch/pound**
- Keepalived: **http://www.keepalived.org**
- Debian: **http://www.debian.org**