PacketFence

Clustering Quick Installation Guide

for PacketFence version 5.5.0

Clustering Quick Installation Guide

by Inverse Inc.

Version 5.5.0 - Nov 2015

Table of Contents

Chapter 1

About this Guide

This guide has been created to give a quick start to install active/active clustering in PacketFence 5+. This guide does not include advanced troubleshooting of the active/active clustering. Refer to the documentation of HAProxy and Keepalive for advanced features.

Chapter 2

Assumptions

- You have at least two servers with a fresh install of PacketFence 5+
- Both servers are identical copies for the network interfaces
- Both servers are running CentOS 6
- Both servers have access to the same layer 2 network on all their network interfaces
- Both servers have an empty, unformatted partition for the database
- Both servers hostname must be resolvable via a DNS resolution

Chapter 3

Installation

Step 1: Install the replicated database

Note

In this implementation, the database is replicated in active/passive across two servers that are part of the PacketFence cluster. Active/active replication is also possible using MariaDB Galera cluster but this subject is not covered in this guide.

Installing DRBD

Creation of the DRBD partition

During the OS installation, reduce the size of the main partition and create a new one (that will be used for the replicated MySQL database) of 30G. **Do not** create that partition during the install process, we will do it later.

Partitioning

After the install, you need to create the extra partition for drbd. Using fdisk, create your new partition and save the table. You will probably need to reboot your server after this step.

DRBD and Linux-HA Installation

CentOS 6

Add the repository ELRepo.

```
# rpm -Uvh http://www.elrepo.org/elrepo-release-6-6.el6.elrepo.noarch.rpm
```

Edit the repo file to disable ELRepo by default:

```
/etc/yum.repos.d/elrepo.repo
```

```
  [elrepo]
  name=ELRepo.org Community Enterprise Linux Repository - el6
  baseurl=http://elrepo.org/linux/elrepo/el6/$basearch/
  mirrorlist=http://elrepo.org/mirrors-elrepo.el6
  enabled=0
  gpgcheck=1
  gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-elrepo.org
  protect=0
```

Install now the package DRBD v8.4 enabling .

```
yum install kmod-drbd84 --enablerepo=elrepo
```

DRBD Configuration and setup

Caution

Initializing, configuring and troubleshooting DRBD is not straight forward! We strongly recommend that you read the online documentation available on DRBD website so you have a better idea about how it works.

Here we assume the name of the partition is mysql.

Load the DRBD kernel module:

```
modprobe drbd
```

Edit `/etc/drbd.d/global_common.conf` and put the following content:

```
global {
    usage-count yes;
}

common {
    protocol C;

    startup {
        degr-wfc-timeout 120;
    }

    syncer {
        rate 100M;
        al-extents 257;
    }

    disk {
        on-io-error     detach;
    }
}
```

Create the file /etc/drbd.d/mysql.res with the following content:

```
resource mysql {
    on <pf1_server_name> {
        device /dev/drbd0;
        disk <storage_device>;
        meta-disk internal;
        address <ha_interface_ip_address_1>:7788;
    }

    on <pf2_server_name> {
        device /dev/drbd0;
        disk <storage_device>;
        meta-disk internal;
        address <ha_interface_ip_address_2>:7788;
    }
    handlers {
        split-brain "/usr/lib/drbd/notify-split-brain.sh alert@acme.com";
    }
}
```

where:

- `mysql` is the name of the partition you created when installing the OS
- `pf1_server_name` and `pf2_server_name` by the real server names (FQDN)
- `ha_interface_ip_address_1` and `ha_interface_ip_address_2` by the IP addresses of the management interface on both servers.
- `storage_device` is the device to use for the MySQL partition (ie. `/dev/sda2`)

Then initialize the partition:

```
[root@pf1 ~]# drbdadm create-md mysql
Writing meta data...
initializing activity log
NOT initialized bitmap
New drbd meta data block successfully created.
success
```

Start DRBD on both servers:

```
# /etc/init.d/drbd start
```

You should see something similar to this when typing cat /proc/drbd:

```
...
 0: cs:Connected ro:Secondary/Secondary ds:Inconsistent/Inconsistent C r----
    ns:0 nr:0 dw:0 dr:0 al:0 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:30702640
```

Note

If you get connectivity issues make sure your iptables are disabled.

Synchronize the servers by forcing one to become the primary. So on pf1 do:

```
# drbdadm primary --force mysql
```

After issuing this command, the initial full synchronization will start. You will be able to monitor its progress via `/proc/drbd`. It may take some time depending on the size of the device. Wait until it completes.

When the sync is complete, create the filesystem on the primary node only:

```
# mkfs.ext4 /dev/drbd0
```

Make sure DRBD is started at boot time:

```
# chkconfig --level 2345 drbd on
```

Restart both servers.

When done, look in `/proc/drbd` and make sure you see:

```
...
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r---
    ns:0 nr:0 dw:0 dr:0 al:0 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:b oos:0
```

MySQL Configuration

Note

By default MySQL puts its data in `/var/lib/mysql`. In order to replicate data between the two servers, we mount the DRBD partition under `/var/lib/mysql`.

When first starting MySQL, the partition must be mounted.

In order to do so:

On the master server (the server you are working on), tell DRBD to become the primary node with:

```
# drbdadm primary mysql
```

`mysql` being the name of the DRBD partition.

The command `cat /proc/drbd` should display something like:

```
...
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r----
    ns:145068 nr:4448 dw:149516 dr:10490 al:31 bm:14 lo:0 pe:0 ua:0 ap:0 ep:1
 wo:d oos:0
```

Stop and backup previous MySQL data

```
# service mysqld stop
# mkdir /var/lib/mysql.bak
# mv /var/lib/mysql/* /var/lib/mysql.bak/
```

Mount the partition with:

```
# mount /dev/drbd0 /var/lib/mysql
```

Start MySQL

```
# service mysqld start
```

Execute the secure installation script in order to set the root password, remove the test databases and anonymous user created by default:

```
# /usr/bin/mysql_secure_installation
```

Make sure MySQL does **not** start at boot time:

```
# chkconfig --level 2345 mysqld off
```

Heartbeat configuration

Install the EPEL repository

```
# rpm -Uvh http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-
release-6-8.noarch.rpm
```

In `/etc/yum.repos.d/epel.repo`, disable the repository

```
[epel]
name=Extra Packages for Enterprise Linux 6 - $basearch
#baseurl=http://download.fedoraproject.org/pub/epel/6/$basearch
mirrorlist=https://mirrors.fedoraproject.org/metalink?repo=epel-6&arch=$basearch
failovermethod=priority
enabled=0
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
```

Now install Heartbeat

```
# yum install heartbeat --enablerepo=epel
```

Create `/etc/ha.d/ha.cf` with the following content:

```
bcast eth0
keepalive 2
warntime 30
deadtime 60
auto_failback off
initdead 120
node pf1.example.com
node pf2.example.com
use_logd yes
logfile /var/log/ha-log
```

Here we assume that the connection for the Heartbeat between the 2 servers is on `eth0`. Make sure you change pf1.example.com and pf2.example.com to the hostnames of your servers.

Create `/etc/ha.d/haresources` with the following content:

```
pf1.example.com drbddisk::mysql Filesystem::/dev/drbd0::/var/lib/mysql::ext4 \
mysqld
```

Create `/etc/ha.d/authkeys` with the following content:

```
auth 1
1 sha1 10abf064f24a3e807dde7b945d0303392f10777d
```

and change its rights like this:

```
# chmod 600 /etc/ha.d/authkeys
```

Note

Make sure port 694 is opened (through iptables) on both servers

Start Heartbeat:

```
# service heartbeat start
```

Look at Heartbeat log file `/var/log/ha-log` to make sure that everything is fine.

Enable HB automatic start

```
# chkconfig --level 345 heartbeat on
```

MySQL network configuration

In order for your PacketFence cluster to communicate properly with the MySQL database, you need to have MySQL bind on the management IP address.

Adjust your MySQL configuration in `/etc/my.cnf` on both servers and make sure the bind_address line is :

```
bind_address=<management_ip_address>
```

Step 2 : Server configuration

You will need to configure the server so the services can bind on IP addresses they don't currently have configured. This allows faster failover of the services.

On CentOS, add the following line in /etc/sysctl.conf and then reload with *sysctl -p*

```
net.ipv4.ip_nonlocal_bind = 1
```

Create the pem that combines the key and certificate for the http services. Adapt to your own paths if you are using different certificates.

```
# cd /usr/local/pf/conf/ssl
# cat server.key server.crt > server.pem
```

MySQL configuration

In order for PacketFence to communicate properly with your MySQL cluster, you need to change the following.

In `conf/pf.conf` :

```
[database]
host=127.0.0.1

[monitoring]
db_host=127.0.0.1
....
```

In `conf/pfconfig.conf` :

```
[mysql]
host=127.0.0.1
....
```

Make sure you restart MySQL, packetfence-config and packetfence

```
# service mysqld restart
# service packetfence-config restart
# service packetfence restart
```

Next, you need to remove the empty users from your MySQL database

```
# mysql
mysql> delete from mysql.user where user = '' ;
mysql> flush privileges;
```

Step 3 : Create a new cluster

In order to create a new cluster, the only thing needed is to configure /usr/local/pf/conf/cluster.conf

You will need to configure it with your server hostname. To get it use : **hostname** in a command line.

In the case of this example it will be *pf1.example.com*.

The *CLUSTER* section represents the virtual IP addresses of your cluster that will be shared by your servers.

In this example, eth0 is the management interface, eth1.2 is the registration interface and eth1.3 is the isolation interface.

Create a configuration similar to this :

```
[CLUSTER]
management_ip=192.168.1.10

[CLUSTER interface eth0]
ip=192.168.1.10
type=management,high-availability

[CLUSTER interface eth1.2]
ip=192.168.2.10
type=internal

[CLUSTER interface eth1.3]
ip=192.168.3.10
type=internal
```

```
[pf1.example.com]
management_ip=192.168.1.5

[pf1.example.com interface eth0]
ip=192.168.1.5
type=management,high-availability
mask=255.255.255.0

[pf1.example.com interface eth1.2]
enforcement=vlan
ip=192.168.2.5
type=internal
mask=255.255.255.0

[pf1.example.com interface eth1.3]
enforcement=vlan
ip=192.168.3.5
type=internal
mask=255.255.255.0
```

Once this configuration is done, restart PacketFence to have it applied. If done properly this should start two additionnal services : haproxy and keepalived

You should now have service on the IP addresses defined in the *CLUSTER* sections

Additionnal steps

It is highly recommended to modify the keepalive shared secret in your cluster to prevent attacks. In the administration interface, go in *Configuration/Clustering* and change the *Shared KEY*. Make sure you restart keepalive on your server using `/usr/local/pf/bin/pfcmd service keepalived restart`

Step 4 : Connect a slave packetfence server

First, connect the server to the database cluster using the instructions above

On CentOS, add the following line in /etc/sysctl.conf and then reload with *sysctl -p*

```
net.ipv4.ip_nonlocal_bind = 1
```

Go through the configurator to setup the interfaces and then stop

Get the webservices user and password on the master node in *Configuration/Web Services* If there's none, set the user, password and then restart httpd.webservices

Do (and make sure it does it without any errors) :

```
# /usr/local/pf/bin/cluster/sync --from=192.168.1.5 --api-user=packet --api-
password=fence
```

Where :

- *192.168.1.5* is the management IP of the other PacketFence node
- *packet* is the webservices username on the master node
- *fence* is the webservices password on the master node

Edit /usr/local/pf/conf/cluster.conf and create the server's configuration using the other node as an example.

Reload the configuation and start the webservices on the new server

```
# service packetfence-config restart
# /usr/local/pf/bin/pfcmd configreload
# /usr/local/pf/bin/pfcmd service haproxy restart
# /usr/local/pf/bin/pfcmd service httpd.webservices restart
```

Make sure that this server is binding to it's own management address. If it's not, verify the /usr/local/pf/conf/cluster.conf management interface configuration.

```
# netstat -nlp | grep 9090
```

Now replicate this server configuration to the other nodes in the cluster

```
# /usr/local/pf/bin/cluster/sync --as-master
```

Make sure at least /usr/local/pf/conf/cluster.conf was replicated to the other servers

Now restart packetfence on each cluster server keeping the new node as the last one to be restarted.

Advanced configuration

Removing a server from the cluster

Note

Removing a server from the cluster requires a restart of the PacketFence service on all nodes.

First, you will need to stop PacketFence on your server and put it offline.

```
# service packetfence stop
# shutdown -h 0
```

Then you need to remove all the configuration associated to the server from /usr/local/pf/conf/cluster.conf on one of the remaining nodes. Configuration for a server is always prefixed by the server's hostname.

Once you have removed the configuration, you need to reload it and synchronize it with the remaining nodes in the cluster.

```
# /usr/local/pf/bin/pfcmd configreload hard
# /usr/local/pf/bin/cluster/sync --as-master
```

Now restart PacketFence on all the servers so that the removed node is not part of the clustering configuration.

Resynchronizing the configuration manually

If you did a manual change in a configuration file, an additionnal step is now needed.

In order to be sure the configuration is properly synched on all nodes, you will need to enter this command on the previously selected master node.

```
# /usr/local/pf/bin/cluster/sync --as-master
```

Adding files to the synchronization

In the event that you do modifications to non-synchronized files like switch modules, files in raddb/, etc, you can add those files to be synchronized when using /usr/local/pf/bin/cluster/sync.

On one of the nodes, create /usr/local/pf/conf/cluster-files.txt

Add the additionnal files one per line in this file. We advise you add this file to the synchronization too.

Example :

```
/usr/local/pf/conf/cluster-files.txt
/usr/local/pf/raddb/modules/mschap
```

haproxy dashboard

You have the possibility to configure the haproxy dashboard which will give you statistics about the current state of your cluster.

In order to active it uncomment the following lines from /usr/local/pf/conf/haproxy.conf

```
listen stats %%active_active_ip%%:1025
  mode http
  timeout connect 10s
  timeout client 1m
  timeout server 1m
  stats enable
  stats uri /stats
  stats realm HAProxy\ Statistics
  stats auth admin:packetfence
```

Note

We strongly advise you change the username and password to something else than admin/packetfence although access to this dashboard doesn’t compromise the server.

Next, uncomment the following line from /usr/local/pf/conf/iptables.conf

Caution

If you’re upgrading from a version prior to 5.0, the line may not be there. Add it close to the other management rules

```
-A input-management-if --protocol tcp --match tcp --dport 1025 --jump ACCEPT
```

Now restart haproxy and iptables in order to complete the configuration

```
# /usr/local/pf/bin/pfcmd service haproxy restart
# /usr/local/pf/bin/pfcmd service iptables restart
```

You should now be able to connect to the dashboard on the following URL : http://pf.local:1025/stats

Unsupported features in active/active

The following features are not supported when using active/active clustering.

- Switches using SNMP based enforcement (port-security, link up/down, ...)
- SNMP roaming with the Aerohive controller. (4.7+ includes support for accounting roaming)