Clustering Quick Installation Guide

for PacketFence version 7.4.0

Clustering Quick Installation Guide

by Inverse Inc.

Version 7.4.0 - Jan 2018
Copyright © 2015 Inverse inc.

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 7+. This guide does not include advanced troubleshooting of the active/active clustering. Refer to the documentation of HAProxy and Keepalived for advanced features.

Chapter 2

Assumptions

- You have at least three (3) installed PacketFence (v7+) servers
- The servers are running one of RHEL / CentOS 7 / Debian Jessie
- The servers have identical identifiers for network interfaces (e.g. eth0)
- The servers network interfaces are on the same layer 2 network
- The servers hostnames must be resolvable via DNS resolution or be in the hostfile (`/etc/hosts`) of each server.

Note

Appended to this guide is a glossary on specialized terms used in this document.

Installation

Step 1: Install the replicated database

Note

In this example, the database stack uses the native PacketFence MariaDB Galera cluster integration. Although other MySQL based clustering stacks are supported, they aren't covered in this guide. If you use an external database or want to use another clustering stack for the database, you can ignore this section and jump to Step 2 directly.

Caution

Galera cluster is only supported in 3 nodes cluster and more (with an odd number of servers).

Installing packages

First, you will need to install Percona Xtrabackup for the synchronization to work correctly.

On RHEL / CentOS 7:

```
# yum install http://www.percona.com/downloads/percona-release/redhat/0.1-3/
percona-release-0.1-3.noarch.rpm
```

Disable Percona repositories, so that installing software from that repository requires explicit action.

```
# sed -i 's/enabled = 1/enabled = 0/g' /etc/yum.repos.d/percona-release.repo
# yum install percona-xtrabackup socat --enablerepo=percona-release-x86_64
```

On Debian:

```
# wget https://repo.percona.com/apt/percona-release_0.1-4.$(lsb_release -
sc)_all.deb
# dpkg -i percona-release_0.1-4.$(lsb_release -sc)_all.deb
# apt-get update
# apt-get -y install percona-xtrabackup-24
```

Configuring Galera

For the next steps, you want to make sure that you didn't configure anything in `/usr/local/pf/conf/cluster.conf`. If you already did, comment all the configuration in the file and do a configreload (`/usr/local/pf/bin/pfcmd configreload hard`).

Setup on the first server of your cluster

First, start packetfence-mariadb and make sure it was able to start in *standalone* mode.

```
# systemctl start packetfence-mariadb
```

Then, secure your installation

```
# mysql_secure_installation
```

Then, you will need to create a user for the database replication that PacketFence will use. You can use any username/password combination. After creating the user, keep its information close-by for usage in the configuration.

Caution

This user should **NOT** have a password that contains spaces

```
# mysql -u root -p
```

```
mysql> CREATE USER 'pfcluster'@'%' IDENTIFIED BY 'aMuchMoreSecurePassword';
mysql> GRANT PROCESS, RELOAD, LOCK TABLES, REPLICATION CLIENT, SUPER ON *.* TO
 'pfcluster'@'%';
```

```
mysql> CREATE USER 'pfcluster'@'localhost' IDENTIFIED BY
 'aMuchMoreSecurePassword';
mysql> GRANT PROCESS, RELOAD, LOCK TABLES, REPLICATION CLIENT, SUPER ON *.* TO
 'pfcluster'@'localhost';
```

```
mysql> FLUSH PRIVILEGES;
```

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

```
# mysql -u root -p
  mysql> delete from mysql.user where user = '' ;
  mysql> flush privileges;
```

Step 2 : Server configuration

First, you need to make sure the interfaces name will be the same on all servers. See *Setting the interfaces name on CentOS 7* in the Appendix section of this document if all your servers don't already have the same interface names.

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

On all your servers, add the following line in `/etc/sysctl.conf` and then reload with *sysctl -p*

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

Then, disable the default MariaDB systemd service unit as PacketFence already provides one (packetfence-mariadb)

```
# systemctl disable mariadb
```

Now, on the first server, create the PEM that combines the key and certificate for the http services. Adapt to your own paths if you are using different certificates. What matters is that the path to the PEM file should be /usr/local/pf/conf/ssl/server.pem since HAProxy will expect it to be there.

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

Going through the configurator

Now, on the first server of your cluster, you should go through the configurator, configuring the server as if it was in standalone (no HA interfaces). You should leave the services stopped at the end of the configurator.

Then restart PacketFence's mariadb on the first server:

```
# systemctl restart packetfence-mariadb
```

On the other servers of your cluster, configure only the network interfaces without going past that section in the configurator. If the other servers already have the right IP addresses configured on their interfaces, you can ignore this step.

Step 3 : Create a new cluster

PacketFence Configuration Modification

In order for PacketFence to communicate properly with your MySQL cluster, you need to change the following. This change only needs to be done on the first server of the cluster. It will be synchronized later.

In `conf/pf.conf` :

```
[database]
host=127.0.0.1

[graphite]
db_host=127.0.0.1

[active_active]
# Change these 2 values by the credentials you've set when configuring MariaDB
 above
galera_replication_username=pfcluster
galera_replication_password=aMuchMoreSecurePassword
```

Then, in `conf/pfconfig.conf` :

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

Then, in `/usr/local/fingerbank/conf/fingerbank.conf`

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

Now, reload the configuration and restart packetfence-config. You will see errors related to a cache write issue but you can safely ignore it for now. These appear because packetfence-config cannot connect to the database yet.

```
# systemctl restart packetfence-config
# /usr/local/pf/bin/pfcmd configreload hard
# systemctl restart packetfence-haproxy
```

Configure cluster.conf

In order to create a new cluster, you need to configure `/usr/local/pf/conf/cluster.conf` on the first server of your cluster.

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.

On the first server, 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
```

```
[pf2.example.com]
management_ip=192.168.1.6

[pf2.example.com interface eth0]
ip=192.168.1.6
type=management,high-availability
mask=255.255.255.0

[pf2.example.com interface eth1.2]
enforcement=vlan
ip=192.168.2.6
type=internal
mask=255.255.255.0

[pf2.example.com interface eth1.3]
enforcement=vlan
ip=192.168.3.6
type=internal
mask=255.255.255.0
```

```
[pf3.example.com]
management_ip=192.168.1.7

[pf3.example.com interface eth0]
ip=192.168.1.7
type=management,high-availability
mask=255.255.255.0

[pf3.example.com interface eth1.2]
enforcement=vlan
ip=192.168.2.7
type=internal
mask=255.255.255.0

[pf3.example.com interface eth1.3]
enforcement=vlan
ip=192.168.3.7
type=internal
mask=255.255.255.0
```

Once this configuration is done, reload the configuration and perform a checkup.

```
# /usr/local/pf/bin/pfcmd configreload hard
# /usr/local/pf/bin/pfcmd checkup
```

The reload and the checkup will complain about the unavailability of the database, which you can safely ignore for now. Most important is that you don't see any cluster configuration related errors during the checkup.

Then make sure the PacketFence clustering services will be started at boot by running the following command on all of your servers

```
# systemctl set-default packetfence-cluster
```

Still on the first server, start MariaDB forcing it to create a new cluster.

Note

If the cluster is unable to get created properly, it may be because the other nodes have their MariaDB service running. Make sure you stop it on all nodes except the first one for now (`systemctl stop packetfence-mariadb` and `systemctl stop mariadb`).

```
# systemctl stop packetfence-mariadb
# /usr/local/pf/bin/pfcmd generatemariadbconfig
# /usr/local/pf/sbin/pf-mariadb --force-new-cluster
```

This last command will not return until you break it, so leave it running in the background and continue on.

Then, restart PacketFence to apply all your changes

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

If no error is found in the previous configuration, the previous restart of packetfence should have started keepalived and radiusd-loadbalancer along with the other services.

You should now have service using the first server on the IP addresses defined in the *CLUSTER* sections.

Step 4: Integrating the two other nodes

Note

If you reboot any of the nodes you're joining, you will need to stop all the PacketFence services (/usr/local/pf/bin/pfcmd service pf stop) and restart the steps from here.

Note

If you reboot the management node, you will need to stop packetfence-mariadb (systemctl stop packetfence-mariadb) and start it with the new cluster option so the servers can join (/usr/local/pf/sbin/pf-mariadb --force-new-cluster)

Now, you will need to integrate your 2 other nodes in your cluster.

Webservices configuration

On the first server, configure your webservices username and password by adding the following in `pf.conf`:

```
[webservices]
user=packet
pass=fence
```

While you can set the username and password to any value, make sure to keep it safe as you will need it while initializing the cluster below.

And reload the config, then restart httpd.webservices on the first server:

```
# /usr/local/pf/bin/pfcmd configreload hard
# /usr/local/pf/bin/pfcmd service httpd.webservices restart
```

Note

The reload will complain about the unavailability of the database, which you can safely ignore.

Sync the nodes

The following instructions have to be done on each server that will be joined in the cluster.

Do (and make sure it completes 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 you have configured on the master node
- *fence* is the webservices password you have configured on the master node

Reload the configuration and start the webservices on the new server

```
# systemctl restart packetfence-config
# /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
```

MariaDB sync

First, ensure your MariaDB instance running with `--force-new-cluster` is still running on the first node, if its not, start it again.

Then, ensure packetfence-mariadb is stopped on the two servers that will be joined

```
# systemctl stop packetfence-mariadb
```

Now, flush any MySQL data you have on the new node and restart packetfence-mariadb so that the servers join the cluster.

Warning

If you have any data in MySQL on this node, this will destroy it.

```
# rm -fr /var/lib/mysql/*
```

```
# systemctl restart packetfence-mariadb
```

Checking the MariaDB sync

In order to check the MariaDB sync, you can look at the status of the wsrep status values inside MySQL.

```
mysql> show status like 'wsrep%';
```

Important variables:

- *wsrep_cluster_status*: Display whether or not the node is part of a primary view or not. A healthy cluster should always show as primary
- *wsrep_incoming_addresses*: The current members of the cluster. All the nodes of your cluster should be listed there.
- *wsrep_local_state_comment*: Current sync state of the cluster. A healthy state is *Synced*. Refer to the Galera cluster documentation for the meaning of the other values this can have.

In order for the cluster to be considered healthy, all nodes must be listed under `wsrep_incoming_addresses` and `wsrep_local_state_comment` must be `Synced`. Otherwise look in the MariaDB log (`/usr/local/pf/logs/mariadb_error.log`)

Starting the first server normally

Once all servers are synced, go on the first server that should still be running with the `--force-new-cluster` option, break the command and then start packetfence-mariadb normally:

```
# systemctl start packetfence-mariadb
```

Wrapping up

Now start packetfence

```
# /usr/local/pf/bin/pfcmd service pf start
```

Next, make sure to join domains through *Configuration/Policies And Access Control/Domains/Active Directory Domains* on each node.

Disable the access to the configurator by setting the currently-at on each server:

```
# /usr/local/pf/bin/pfcmd version > /usr/local/pf/conf/currently-at
```

Step 5: Securing the cluster

Keepalived secret

It is highly recommended to modify the keepalived shared secret in your cluster to prevent attacks. From the PacketFence web administration interface, go in *Configuration/System Configuration/Cluster* and change the *Shared KEY*. Make sure you restart keepalived on all your servers using `/usr/local/pf/bin/pfcmd service keepalived restart`

Understanding the Galera cluster synchronization

The Galera cluster stack used by PacketFence resembles a lot to how a normal MariaDB Galera cluster behaves but it contains hooks to auto-correct some issues that can occur.

Note

A lot of useful information is logged in the MariaDB log which can be found in `/usr/local/pf/logs/mariadb_error.log`

Quorum behavior

A loss of quorum is when a server is not able to be part of a group that represents more than 50% of the configured servers in the cluster. This can occur if a node is isolated from a network perspective or if more than 50% of its peers aren't alive (like in the case of a power outage).

The Galera cluster stack will continuously check that it has a quorum. Should one of the server be part of a group that doesn't have the quorum in the cluster, it will put itself in read-only mode and stop the synchronization. During that time, your PacketFence installation will continue working but with some features disabled.

- RADIUS MAC Authentication: Will continue working and will return RADIUS attributes associated with the role that is registered in the database. If VLAN or RADIUS filters can apply to this device, they will but any role change will not be persisted.
- RADIUS 802.1X: Will continue working and if *Dot1x recompute role from portal* is enabled, it will compute the role using the available authentication sources but will not save it in the database at the end of the request. If this parameter is disabled, it will behave like MAC Authentication. VLAN and RADIUS filters will still apply for the connections. If any of your sources are external (LDAP, AD, RADIUS, …), they must be available for the request to complete successfully.
- Captive portal: The captive portal will be disabled and display a message stating the system is currently experiencing an issue.
- DHCP listeners: The DHCP listeners will be disabled and packets will not be saved in the database. This also means Firewall SSO will not work during that time.
- Web administration interface: It will still be available in read-only mode for all sections and in read-write mode for the configuration section.

Once the server that is in read-only mode joins a quorum, it will go back in read-write mode and the system will go back to its normal behavior automatically.

Graceful shutdown behavior

When you are gracefully shutting down servers for a planned maintenance, you should always aim to leave a quorum alive so that once the server joins its peers again, it will always re-join the cluster gracefully. You can also leave only one of the nodes alive but keep in mind it will fall in read-only mode until all the nodes that were part of the last healthy quorum rejoin the cluster.

Should all your nodes shutdown gracefully, the last node to be shutdown will be the one that will be able to self-elect as master when you bring the machines back online. Bringing this node back online first is a best practice but not mandatory. In order to know which server would be able to self-elect as master, look for the node that has `safe_to_bootstrap: 1` when executing the following command `cat /var/lib/mysql/grastate.dat | grep 'safe_to_bootstrap:'`.

Ungraceful shutdown behavior

Note

You can know a node was hard-shutdown if `/var/lib/mysql/gvwstate.dat` exists on the node.

If at least one node is still alive, other nodes will be able to connect to it and re-integrate the cluster.

When all nodes are ungracefuly shutdown, they will be able to recover when the last 2 running nodes will be able to talk to each other. If all nodes were shutdown at the same time, all of them must rejoin the cluster for it to form again. Until the cluster reaches a healthy state, you will not have any database service.

Recovering when a node is missing

If one of the nodes cannot recover, you can manually reset the cluster state. Note that this procedure is only valid if all the nodes were hard-shutdown at the same time.

First, stop packetfence-mariadb on all servers:

```
# systemctl stop packetfence-mariadb
```

Then remove `/var/lib/mysql/gvwstate.dat` on all the servers that are still alive.

Next, see the section *Electing a new master* to start up a new cluster.

Electing a new master

Caution

Before doing this, we suggest you are confortable working with the MariaDB Galera cluster stack in order to fix potential conflicts that arise from this procedure.

At any time, you can start up a new cluster without the management node which is the source of truth by default. In order to do so, execute: `/usr/local/pf/sbin/pf-mariadb --force-new-cluster`. Wait for another server to connect to this new cluster, break the command and start packetfence-mariadb like you normally would.

Note that if your management node is alive, but not communicating with the new master node, then it will create a new cluster on its side automatically unless it is in maintenance mode. When electing a new cluster, you should ensure your management has (or will have when its alive) valid communication with the newly promoted cluster. Otherwise, you will end up in a split brain situation. See *Recovering from a split brain* to fix this.

Recovering from a split brain

If you leave the cluster in auto-recovery mode without electing a master manually, the structure of the cluster is made so that a split-brain can never occur since a server will fallback in read-only if it can't join a primary view (quorum).

Should you decide to elect a new master and cause a split-brain for crash recovery purpose, you will have to wipe the data on all the servers that are part of one side of the split brain. This should be done once all the servers have re-gained communication.

Full recovery

If you want to perform a full recovery which can be necessary after experiencing an issue with Galera cluster, you must stop the node you wish to keep the data from and start it with the `--force-new-cluster` option. If this is not the case, you can continue onto the next step

```
# systemctl stop packetfence-mariadb
# /usr/local/pf/sbin/pf-mariadb --force-new-cluster
```

On each of the discarded servers

First, stop packetfence-mariadb on **all** the servers you want to discard data from.

```
systemctl stop packetfence-mariadb
```

On each of the servers you want to discard the data from, you must destroy all the data in `/var/lib/mysql` and start packetfence-mariadb so it resyncs its data from scratch.

```
rm -fr /var/lib/mysql/*
systemctl start packetfence-mariadb
```

You should then see `/var/lib/mysql` be populated again with the data and once MySQL becomes available again on the server, it means the sync has completed. In case of issues, look in the MySQL log file (`/usr/local/pf/logs/mariadb_error.log`)

Note

If you were performing a full recovery, you should now break the `--force-new-cluster` command and start packetfence-mariadb normally. (`systemctl start packetfence-mariadb`)

Putting a node in maintenance

When doing maintenance on a node (especially the management node), it is always better to put it in maintenance mode so it doesn't try to join an existing cluster.

In order to activate the maintenance mode on a node:

```
# /usr/local/pf/bin/cluster/maintenance --activate
```

In order to deactivate the maintenance mode on a node:

```
# /usr/local/pf/bin/cluster/maintenance --deactivate
```

In order to see the current maintenance state on a node:

```
# /usr/local/pf/bin/cluster/maintenance
```

Chapter 5

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.

```
# systemctl stop packetfence
# 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.

Note that if you remove a node and end up having an even number of servers, you will get unexpected behaviors in MariaDB. You should always aim to have an odd number of servers at all time in your cluster.

Resynchronizing the configuration manually

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

In order to be sure the configuration is properly synced 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 additional 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

Configuration conflict handling

Note

It is not recommended to perform configuration while one or more node of the cluster is experiencing issues. Still, should that be the case, this section will explain the conflict resolution that will occur when the nodes will reattach together.

When modifying the configuration through the administration interface, the configuration will be automatically synchronized to all the nodes that are online. In the event that one or more nodes cannot be updated, an error message will be displayed with affected nodes.

A scheduled check runs on the management server (controlled through maintenance.cluster_check_interval) in order to validate if all servers are running the same configuration version. When the failed node(s) will come back online, that scheduled check will ensure that the new configuration is pushed on the new node(s). You can disable this check by setting maintenance.cluster_check_interval to 0 and restarting pfmon. In that case, you will need to manually resolve the conflict when the node(s) come back online by running `/usr/local/pf/bin/cluster/sync --as-master` on the node you want to keep the configuration of.

General facts about conflict resolution:

- If the configuration is not pushed to at least half of the servers of your cluster, when the failed nodes will come back online, they will have quorum on the previous configuration and the one they are running will be pushed to all the servers.
- In a two node cluster, the most recent configuration is always selected when resolving a conflict.
- In a two node cluster, no decision is taken unless the peer server has its webservices available.

Going deeper in the conflict handling

The section below will explain with more details, the steps that are taken in order to take the decision of which server should be declared as the master when one or more servers have conflicting configuration version.

The first step is to get the configuration version from each server through a webservice call.

The results are then organized by version identifier. Should all alive servers run the same version, the state is considered as healthy and nothing happens.

Then, should there be more than one version identifier across the alive servers, the algorithm validates that there are at least 2 servers **configured** in the cluster. If there aren't, then the most recent version is pushed on the peer node.

After that, the algorithm looks at which version is on the most servers. In the event that the dead servers are in higher number than the alive ones, the most recent version is taken. Otherwise, the version that is present on the most servers will be selected.

When pushing a version to the other servers, if the current server has the most recent version or is part of the quorum (depending on which push strategy was defined above), then it will be the one pushing the new configuration to the other servers. Otherwise, a webservice call is made to one of the servers running the selected version so that it pushes its configuration to its peers.

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)
- Inline enforcement

Appendix

Glossary

- *Alive quorum*: An alive quorum is when more than 50% of the servers of the cluster are online and reachable on the network (pingable). This doesn't imply they offer service, but only that they are online on the network.
- *Hard-shutdown*: A hard shutdown is when a node or a service is stopped without being able to go through a proper exit cleanup. This can occur in the case of a power outage, hard reset of a server or `kill -9` of a service.
- *Management node/server*: The first server of a PacketFence cluster as defined in `/usr/local/pf/conf/cluster.conf`.
- *Node*: In the context of this document, a node is a member of the cluster while in other PacketFence documents it may represent an endpoint.

Setting the interfaces name on CentOS 7

On CentOS 7 you need to make sure that all the servers in the cluster use the same interfaces name. This section covers how to set the interfaces to the ethX format. Note that you can set it to the format you desire as long as the names are the same on all servers.

First, go in `/etc/default/grub` and add `net.ifnames=0` to the variable: `GRUB_CMDLINE_LINUX`.

```
GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap
 rhgb quiet net.ifnames=0"
```

Then regenerate the GRUB configuration by executing the following command:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

Then, rename the network script of your management interface (`eno16780032` in this example) to be in the ethX form (`eth0` in this example)

```
# mv /etc/sysconfig/network-scripts/ifcfg-eno16780032 /etc/sysconfig/network-
scripts/ifcfg-eth0
```

And rename the name of the interface in the following files (making sure you replace `eno16780032` and `eth0` by the appropriate values):

```
# sed -i.bak "s/eno16780032/eth0/g" /etc/sysconfig/network-scripts/ifcfg-eth0
# sed -i.bak "s/eno16780032/eth0/g" /usr/local/pf/conf/pf.conf
```

Apply the last two steps for any other interface you have on your server. Keep in mind, you can use the PacketFence configurator to reconfigure them later as long as your management interface is correctly configured and is accessible on your network.

Now, reboot your server and when it finishes starting, your interfaces name should now be using the format `ethX`

Performing an upgrade on a cluster

Caution

Performing a live upgrade on a PacketFence cluster is not a straightforward operation and should be done meticulously.

In this procedure, the 3 nodes will be named A, B and C and they are in this order in `cluster.conf`.

Backups

First, ensure you have taken backups of your data. We highly encourage you to perform snapshots of all the virtual machines prior to the upgrade. You should also take a backup of the database and the `/usr/local/pf` directory using:

```
# mysqldump --opt -u root -p pf | gzip > /root/packetfence_db.sql.gz
# tar -C /usr/local -czf /root/packetfence.tar.gz pf --exclude='pf/logs' --
exclude='pf/var'
```

Disabling the auto-correction of configuration

The PacketFence clustering stack has a mechanism that allows configuration conflicts to be handled accross the servers. This will come in conflict with your upgrade, so you should disable it.

In order to do, so go in *Configuration→Maintenance* and set *Cluster Check Interval* to `0`.

Once this is done, restart pfmon on all nodes using:

```
# /usr/local/pf/bin/pfcmd service pfmon restart
```

Upgrading node C

In order to be able to work on node C, we first need to stop all the PacketFence application services on it.

```
# /usr/local/pf/bin/pfcmd service pf stop
```

Next, upgrade your operating system and PacketFence on node C:

```
# yum upgrade --enablerepo=packetfence
```

Note

If you reboot your server during this procedure, make sure you set the systemd target to multi-user (`systemctl set-default multi-user.target`) before rebooting and back to packetfence-cluster (`systemctl set-default packetfence-cluster.target`) after it boots up. This will make sure your services don't start up after the reboot.

Now, make sure you follow the directives in the upgrade guide (UPGRADE.asciidoc) as you would on a standalone server with the exception of the database schema updates.

Migrating service on node C

Node C should currently be running the latest version of PacketFence while A and B should still be running the older version and still be offering service. The database is also still being synced to node C via the packetfence-mariadb service.

Note

The steps below will cause a temporary loss of service.

Detach node C from the cluster

First, we need to tell A and B to ignore C in their cluster configuration. In order to do so, execute the following command **on A and B** while changing `node-C-hostname` with the actual hostname of node C:

```
# /usr/local/pf/bin/cluster/node node-C-hostname disable
```

Once this is done proceed to restart the following services on nodes A and B **one at a time**. This will cause service failure during the restart on node A

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

Then, we should tell C to ignore A and B in their cluster configuration. In order to do so, execute the following commands on node C while changing `node-A-hostname` and `node-B-hostname` by the hostname of nodes A and B respectively.

```
# /usr/local/pf/bin/cluster/node node-A-hostname disable
# /usr/local/pf/bin/cluster/node node-B-hostname disable
```

Now on packetfence-mariadb on all nodes

```
# systemctl stop packetfence-mariadb
```

Then start them in the following order: A then B then C

```
# systemctl start packetfence-mariadb
```

Note

From this moment on, you will lose the configuration changes and data changes that occur on nodes A and B.

The commands above will make sure that nodes A and B will not be forwarding requests to C even if it is alive. Same goes for C which won't be sending traffic to A and B. This means A and B will continue to have the same database informations while C will start to diverge from it when it goes live. We'll make sure to reconcile this data afterwards.

Complete upgrade of node C

From that moment node C is in standalone for its database. We can proceed to update the database schema so it matches the one of the latest version. In order to do so, upgrade the database schema using the instructions provided in UPGRADE.asciidoc.

Stop service on nodes A and B

Next, stop all application service on node A and B along with the database:

```
# /usr/local/pf/bin/pfcmd service pf stop
# systemctl stop packetfence-mariadb
```

Start service on node C

Now, start the application service on node C

```
# systemctl isolate packetfence-cluster
```

Validate migration

You should now have full service on node C and should validate that all functionnalities are working as expected. Once you continue past this point, there will be no way to migrate back to nodes A and B in case of issues other than to use the snapshots taken prior to the upgrade.

If all goes wrong

If your migration to node C goes wrong, you can fail back to nodes A and B by stopping all services on node C and starting them on nodes A and B

On node C

```
# systemctl stop packetfence-mariadb
# /usr/local/pf/bin/pfcmd service pf stop
```

On nodes A and B

```
# systemctl start packetfence-mariadb
# systemctl isolate packetfence-cluster
```

Once you are feeling confident to try your failover to node C again, you can do the exact opposite of the commands above to try your upgrade again.

If all goes well

If you are happy about the state of your upgrade, you can continue on the steps below in order to complete the upgrade of the two remaining nodes.

Upgrading nodes A and B

Next, upgrade your operating system and PacketFence on each server:

```
# yum upgrade --enablerepo=packetfence
```

Note

If you reboot your server during this procedure, make sure you set the systemd target to multi-user (`systemctl set-default multi-user.target`) before rebooting and back to packetfence-cluster (`systemctl set-default packetfence-cluster.target`) after it boots up. This will make sure your services don't start up after the reboot.

You do not need to follow the upgrade procedure when upgrading these nodes. You should instead do a sync from node C on nodes A and B

```
# /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 node C
- *packet* is the webservices username (Configuration→Webservices)
- *fence* is the webservices password (Configuration→Webservices)

Reintegrating nodes A and B

Optional step: Cleaning up data on node C

When you will re-establish a cluster using node C in the steps below, your environment will be set in read-only mode for the duration of the database sync (which needs to be done from scratch).

This can take from a few minutes to an hour depending on your database size.

We highly suggest you delete data from the following tables if you don't need it:

- radius_audit_log: contains the data in Auditing→RADIUS Audit Log
- ip4log_history: Archiving data for the IPv4 history
- ip4log_archive: Archiving data for the IPv4 history
- locationlog_archive: Archiving data for the node location history

You can safely delete the data from all of these tables without affecting the functionnalities as they are used for reporting and archiving purposes. Deleting the data from these tables can make the sync process considerably faster.

In order to truncate a table:

```
# mysql -u root -p pf
mysql> truncate TABLE_NAME;
```

Elect node C as database master

In order for node C to be able to elect itself as database master, we must tell it there are other members in its cluster by re-enabling nodes A and B

```
# /usr/local/pf/bin/cluster/node node-A-hostname enable
# /usr/local/pf/bin/cluster/node node-B-hostname enable
```

Next, enable node C on nodes A and B by executing the following command on the two servers:

```
# /usr/local/pf/bin/cluster/node node-C-hostname enable
```

Now, stop packetfence-mariadb on node C, regenerate the MariaDB configuration and start it as a new master:

```
# systemctl stop packetfence-mariadb
# /usr/local/pf/bin/pfcmd generatemariadbconfig
# /usr/local/pf/sbin/pf-mariadb --force-new-cluster
```

You should validate that you are able to connect to the MySQL database even though it is in read-only mode using the MySQL command line. If its not, make sure you check the MySQL log (`/usr/local/pf/logs/mariadb_error.log`)

Sync nodes A and B

On each of the servers you want to discard the data from, stop packetfence-mariadb, you must destroy all the data in `/var/lib/mysql` and start packetfence-mariadb so it resyncs its data from scratch.

```
# systemctl stop packetfence-mariadb
# rm -fr /var/lib/mysql/*
# systemctl start packetfence-mariadb
```

Should there be any issues during the sync, make sure you look into the MySQL log (`/usr/local/pf/logs/mariadb_error.log`)

Once both nodes have completely synced (try connecting to it using the MySQL command line), then you can break the cluster election command you have running on node C and start node C normally (using `systemctl start packetfence-mariadb`).

Start nodes A and B

You can now safely start PacketFence on nodes A and B using:

```
# systemctl isolate packetfence-cluster
```

Restart node C

Now, you should restart PacketFence on node C so it becomes aware of its peers again.

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

You should now have full service on all 3 nodes using the latest version of PacketFence.

Reactivate the configuration conflict handling

Now that your cluster is back to a healthy state, you should reactivate the configuration conflict resolution.

In order to do, so go in *Configuration→Maintenance* and set *Cluster Check Interval* to `1 minute` (or the value you prefer).

Once this is done, restart pfmon on all nodes using:

```
# /usr/local/pf/bin/pfcmd service pfmon restart
```

MariaDB Galera cluster troubleshooting

Maximum connections reached

In the event that one of the 3 servers reaches the maximum amount of connections (defaults to 1000), this will dead-lock the Galera cluster synchronization. In order to resolve this, you should first increase database_advanced.max_connections, then stop packetfence-mariadb on all 3 servers, and follow the steps in the section *Recovering from a split brain* of this document. Note that you can use any of the database servers as your source of truth.

Investigating further

The limit of 1000 connections is fairly high already so if you reached the maximum number of connections, this might indicate an issue with your database cluster. If this issue happens often, you should monitor the active connections and their associated queries to find out what is using up your connections.

You can monitor the active TCP connections to MySQL using this command and then investigate the processes that are connected to it (last column):

```
# netstat -anlp | grep 3306
```

You can have an overview of all the current connections using the following MySQL query:

```
mysql> select * from information_schema.processlist;
```

And if you would like to see only the connections with an active query:

```
mysql> select * from information_schema.processlist where Command!='Sleep';
```