Database High Availability on CentOS 7

Database High Availability on CentOS 7

We are working on the translation of the Pandora FMS documentation. Sorry for any inconvenience.

CentOS 7 will soon reach its End of Life (EOL). This documentation is preserved for historical purposes.

High Availability in the Database

The objective of this section is to offer a complete solution for HA in Pandora FMS environments. This is the only HA model with official support for Pandora FMS. This solution is provided -pre-installed- from OUM 724 to 769. This system replaces DRBD and other HA systems recommended in the past (724 and earlier).

See "Migration of HA Corosync-Pacemaker environments" to update the high availability database used in version 770.

This is the first implementation of Pandora FMS DB HA and the installation process is almost entirely manual, using the GNU/Linux console as root. In future versions we will provide configuration from the graphical interface (GUI).

Pandora FMS is based on a MySQL database to configure and store data. A database failure can momentarily paralyze the monitoring tool. The Pandora FMS high availability database cluster allows you to easily deploy a robust and fault tolerant architecture.

This is an advanced feature that requires knowledge of GNU/Linux systems.

Cluster resources are managed with Pacemaker, an advanced and scalable high availability cluster resource manager. Corosync provides a closed process group communication model for creating replicated state machines. Percona was chosen as the default RDBMS for its scalability, availability, security, and backup features.

The active/passive replication is developed from a single master node (with writable permission) to any number of children (read only). A virtual IP address always points to the current master. If the master node fails, one of the secondary nodes is promoted to master and the virtual IP address is updated accordingly.

The Pandora FMS HA database tool, `pandora_ha`, monitors the cluster and makes sure that the Pandora FMS server is in continuous operation, restarting it if necessary. `pandora_ha` is in turn monitored by systemd.

It is recommended to keep a maximum of 15 days of data and events, to save for a longer time a historical database must be configured. Also consult the topic "Management and administration of PFMS servers".

This example will set up a two-node cluster, with hosts node1 and node2. Hostnames, passwords, etc. will be changed. according to what you need to match the environment to be implemented.

Version 759 or earlier.

Commands that need to be executed on a node will be preceded by the hostname of that node:

```
node1# <command>
```

Commands that need to be executed on all nodes will be preceded by the word all:

```
all# <command>
```

There is an additional host, called pandorafms, where Pandora FMS is or will be installed.

When referring to all only refers to the Database nodes, the additional Pandora FMS node will always be referenced as pandorafms and is not part of all.

Prerequisites

CentOS 7 will soon reach its End of Life (EOL). This documentation is preserved for historical purposes.

CentOS version 7 must be installed on all hosts, and they must be able to resolve each other's hostnames.

```
node1# ping node2
PING node2 (192.168.0.2) 56(84) bytes of data.

node2# ping node1
PING node1 (192.168.0.1) 56(84) bytes of data.
```

```
pandorafms# ping node1
PING node1 (192.168.0.1) 56(84) bytes of data.

pandorafms# ping node2
PING node2 (192.168.0.2) 56(84) bytes of data.
```

An Open SSH server must be installed and running on each host. We suppressed the notice that Open SSH shows:

```
all# [ -f /etc/cron.hourly/motd_rebuild ] && rm -f /etc/cron.hourly/motd_rebuild
all# sed -i -e 's/^Banner.*g' /etc/ssh/sshd_config
all# systemctl restart sshd
```

The Pandora FMS HA database tool will not work correctly
if Open SSH has a warning configured.

We generate new SSH authentication keys for each host and copy the public key for each of the hosts:

Keys can be generated for a non-root user for a later non-
root cluster installation.

```
node1# echo -e "\n\n\n" | ssh-keygen -t rsa
node1# ssh-copy-id -p22 root@node2
node1# ssh node2

node2# echo -e "\n\n\n" | ssh-keygen -t rsa
node2# ssh-copy-id -p22 root@node1
node2# ssh node1

pandorafms# echo -e "\n\n\n" | ssh-keygen -t rsa
pandorafms# ssh-copy-id -p22 root@node1
pandorafms# ssh-copy-id -p22 root@node2
pandorafms# ssh node1
pandorafms# ssh node2
```

In the Pandora FMS node, we copy the key pair to `/usr/share/httpd/.ssh/`. The Pandora FMS console needs to retrieve the cluster status:

```
pandorafms# cp -r /root/.ssh/ /usr/share/httpd/
pandorafms# chown -R apache:apache /usr/share/httpd/.ssh/
```

The following steps are required only if the nodes are running SSH on a non-standard port. You must replace 22 with the correct port number:

```
  all# echo -e "Host node1\n Port 22">> /root/.ssh/config
  all# echo -e "Host node2\n Port 22">> /root/.ssh/config
```

Installation of Percona

Install the necessary package:

```
  all# yum install
https://repo.percona.com/yum/percona-release-latest.noarch.rpm

  all# yum install -y Percona-Server-server-57 percona-xtrabackup-24
```

For more information regarding the Percona installation process, you can consult the official product documentation:

https://www.percona.com/doc/percona-server/5.7/installation/yum_repo.html

Once the packages are installed, make sure that the Percona service is disabled, as it will be managed by the cluster:

```
all# systemctl disable mysqld
```

If the system service is not disabled, the cluster resource manager will not work correctly.

Next, start the Percona server:

```
all# systemctl start mysqld
```

A new temporary password connected to `/var/log/mysqld.log` will be generated. Connect to the Percona server and change the root password:

```
all# mysql -uroot -p$(grep "temporary password" /var/log/mysqld.log | \
rev | cut -d' ' -f1 | rev)
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('Pandor4!');
mysql> UNINSTALL PLUGIN validate_password;
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('pandora');
mysql> quit
```

Reinstall the server package with the ha option (flag):

```
pandorafms# ./pandora_server_installer --install --ha
```

Once the server is installed with the HA tools enabled, you will find the configuration generator for database replication in the path: `/usr/share/pandora_server/util/myconfig_ha_gen.sh`

```
Example: ./myconfig_ha_gen.sh -i serverid [-l file_location] [-d database] [-b
binlog] [-u dbuser] [-p dbpass] [-s poolsize] [-h help]
Mandatory parameters:
        -i --serverid Set the server id for the database (Mandatory)
Optional parameters:
        -l --location Set my.cnf custom location including filename. [ default
value: /etc/my.cnf ] (optional)
        -d --database Set the database to be replicated. [ default value:
pandora ] (optional)
        -b --binlog Set binlog file. [ default value: mysql-bin ] (optional)
        -u --dbuser Set dbuser for mysql connection and backups. [ default
value: root ] (optional)
        -p --dbpass Set dbpassword for mysql connection and backups. [ default
value: pandora ] (optional)
        -s --poolsize Set innodb_buffer_pool_size static size in M (Megabytes)
or G (Gigabytes). [ default value: autocalculated ] (optional)
        -h --help Print help.
```

In the current case where the databases are not on the same server as the application, it will be necessary to copy the script to the nodes to be executed locally.

```
pandorafms# scp /usr/share/pandora_server/util/myconfig_ha_gen.sh
root@node1:/root/
pandorafms# scp /usr/share/pandora_server/util/myconfig_ha_gen.sh
root@node2:/root/
```

It will only be necessary to pass the serverid parameter (mandatory) in standard environments and some optional parameters for custom environments.

If the default or defined user does not connect to the database, the script will end with a connection error.

There is also the possibility of passing the database, the user and the password as arguments. If you don'tThe default ones will be used.

In this case, the script will be executed on both nodes only by passing the server id if we have the default credentials, otherwise define the necessary parameters.

```
  node1# /root/myconfig_ha_gen.sh -i 1
  node2# /root/myconfig_ha_gen.sh -i 2
```

Important: Each node must have a unique identifier

The Percona configuration file will be written to `/etc/my.cnf` where the server identifier and the recommended configuration for database replication will be defined.

You must restart the mysqld service to verify that the configuration has been applied correctly.

```
all# systemctl restart mysqld
```

Installation of Pandora FMS

New installation of Pandora FMS

Install Pandora FMS in the newly created database. Stop the Pandora FMS server:

```
pandorafms# /etc/init.d/pandora_server stop
```

As of version NG 754, it has additional options in manual startup and shutdown of High Availability Environments (HA).

Existing Pandora FMS installation

Stop the Pandora FMS server:

```
pandorafms# /etc/init.d/pandora_server stop
```

Make a backup of the Pandora FMS database:

```
pandorafms# mysqldump -uroot -ppandora --databases pandora> /tmp/pandoradb.sql
pandorafms# scp /tmp/pandoradb.sql node1:/tmp/
```

Now load the information to the new database:

```
node1# mysql -uroot -ppandora pandora -e source "/tmp/pandoradb.sql"
```

Replication Settings

Grant the necessary privileges for replication to work on all databases:

```
all# mysql -uroot -ppandora
mysql> GRANT ALL ON pandora.* TO 'root'@'%' IDENTIFIED BY 'pandora';
```

```
mysql> GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.*
TO 'root'@'%' IDENTIFIED BY 'pandora';
mysql> GRANT REPLICATION CLIENT, REPLICATION SLAVE, SUPER, PROCESS, RELOAD ON
*.*
TO 'root'@'localhost' IDENTIFIED BY 'pandora';
mysql> GRANT select ON mysql.user TO 'root'@'%' IDENTIFIED BY 'pandora';
mysql> FLUSH PRIVILEGES;
mysql> quit
```

Back up the database on the first node and write down the name and position of the master log file (in the example, `mysql-bin.000001` and `785`):

```
node1# [ -e /root/pandoradb.bak ] && rm -rf /root/pandoradb.bak
node1# innobackupex --no-timestamp /root/pandoradb.bak/
node1# innobackupex --apply-log /root/pandoradb.bak/
node1# cat /root/pandoradb.bak/xtrabackup_binlog_info
mysql-bin.000001 785
```

Load the database on the second node and configure to replicate from the first node (you must set `MASTER_LOG_FILE` and `MASTER_LOG_POS` to the values from the previous step):

```
node2# systemctl stop mysqld

node1# rsync -avpP -e ssh /root/pandoradb.bak/ node2:/var/lib/mysql/

node2# chown -R mysql:mysql /var/lib/mysql
node2# chcon -R system_u:object_r:mysqld_db_t:s0 /var/lib/mysql
node2# systemctl start mysqld
node2# mysql -uroot -ppandora
mysql> CHANGE MASTER TO MASTER_HOST='node1',
MASTER_USER='root', MASTER_PASSWORD='pandora',
MASTER_LOG_FILE='mysql-bin.000001', MASTER_LOG_POS=785;
mysql> START SLAVE;
mysql> SHOW SLAVE STATUS \G
    *************************** 1.row ******************* *******
                Slave_IO_State: Waiting for master to send event
                   Master_Host: node1
                   Master_User: root
                   Master_Port: 3306
                  Connect_Retry: 60
                Master_Log_File: mysql-bin.000002
            Read_Master_Log_Pos: 785
                 Relay_Log_File: node2-relay-bin.000003
                  Relay_Log_Pos: 998
          Relay_Master_Log_File: mysql-bin.000002
               Slave_IO_Running: Yes
              Slave_SQL_Running: Yes
                Replicate_Do_DB: pandora
            Replicate_Ignore_DB:
             Replicate_Do_Table:
         Replicate_Ignore_Table:
```

```
          Replicate_Wild_Do_Table:
      Replicate_Wild_Ignore_Table:
                       Last_Errno: 0
                       Last_Error:
                     Skip_Counter: 0
              Exec_Master_Log_Pos: 785
                  Relay_Log_Space: 1252
                  Until_Condition: None
                   Until_Log_File:
                    Until_Log_Pos: 0
               Master_SSL_Allowed: No
               Master_SSL_CA_File:
               Master_SSL_CA_Path:
                  Master_SSL_Cert:
                Master_SSL_Cipher:
                   Master_SSL_Key:
            Seconds_Behind_Master: 0
    Master_SSL_Verify_Server_Cert: No
                    Last_IO_Errno: 0
                    Last_IO_Error:Last_SQL_Errno: 0
                   Last_SQL_Error:
      Replicate_Ignore_Server_Ids:
                 Master_Server_Id: 1
                      Master_UUID: 580d8bb0-6991-11e8-9a22-16efadb2f150
                 Master_Info_File: mysql.slave_master_info
                        SQL_Delay: 0
              SQL_Remaining_Delay: NULL
          Slave_SQL_Running_State: Slave has read all relay log; waiting for
more updates
                Master_Retry_Count: 86400
                       Master_Bind:
          Last_IO_Error_Timestamp:
         Last_SQL_Error_Timestamp:
                   Master_SSL_Crl:
               Master_SSL_Crlpath:
               Retrieved_Gtid_Set:
                Executed_Gtid_Set:
                    Auto_Position: 0
             Replicate_Rewrite_DB:
                     Channel_Name:
               Master_TLS_Version:
    1 row in set (0.00 sec)
mysql> QUIT
```

You must ensure that `Slave_IO_Running` and `Slave_SQL_Running` return `Yes`. Other values may be different from the example.

It is recommended not to use the root user to carry out this process. It is

advisable to give permissions to another user in charge of managing the database to avoid possible conflicts.

Two Node Cluster Configuration

Install the necessary packages:

```
  all# yum install -y epel-release corosync ntp pacemaker pcs

  all# systemctl enable ntpd
  all# systemctl enable corosync
  all# systemctl enable pcsd
  all# cp /etc/corosync/corosync.conf.example /etc/corosync/corosync.conf
```

```
  all# systemctl start ntpd
  all# systemctl start corosync
  all# systemctl start pcsd
```

Stop the Percona server:

```
node1# systemctl stop mysqld
```

```
node2# systemctl stop mysqld
```

Authentication of all nodes in the cluster

Create and start the cluster:

```
all# echo hapass | passwd hacluster --stdin
```

```
node1# pcs cluster auth -u hacluster -p hapass --force node1 node2
node1# pcs cluster setup --force --name pandoraha node1 node2
node1# pcs cluster start --all
node1# pcs cluster enable --all
node1# pcs property set stonith-enabled=false
node1# pcs property set no-quorum-policy=ignore
```

Check the status of the cluster:

```
node#1 pcs status
    Cluster name: pandoraha
    Stack: corosync
    Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with
quorum
    Last updated: Fri Jun 8 12:53:49 2018
```

```
    Last change: Fri Jun 8 12:53:47 2018 by root via cibadmin on node1

    2 nodes configured
    0 resources configured

    Online: [ node1 node2 ]

    No resources

    Daemon Status:
      corosync: active/disabled
      pacemaker: active/disabled
      pcsd: active/enabled
```

Both nodes should be online (`Online: [ node1 node2 ]`). Other values may be different from the example.

Installing the Percona Pacemaker Replication Agent

It can be downloaded manually from PFMS library.

```
all# cd /usr/lib/ocf/resource.d/
all# mkdir percon
all# cd percona
all# curl -L -o pacemaker_mysql_replication.zip
https://pandorafms.com/library/wp-content/uploads/2019/12/pacemaker_mysql_replic
ation.zip
all# unzip pacemaker_mysql_replication.zip
all# rm -f pacemaker_mysql_replication.zip
all# chmod u+x mysql
```

Configure the cluster resources. Replace < VIRT_IP > with your preferred virtual IP address:

If the default password used in this guide for the database user root has been changed, you should update `replication_passwd` and `test_passwd` respectively.

The names of the cluster resources must be exactly those indicated in this guide (`pandoraip` and `pandoradb`)

```
node1# export VIP='<VIRT_IP>'

node1# pcs resource create pandoradb ocf:percona:mysql config="/etc/my.cnf" \
pid="/var/run/mysqld/mysqld.pid" socket="/var/lib/mysql/mysql.sock" \
```

```
replication_user="root" replication_passwd="pandora" max_slave_lag="60" \
evict_outdated_slaves="false" binary="/usr/sbin/mysqld" datadir="/var/lib/mysql"
\
test_user="root" test_passwd="pandora" op start interval="0" timeout="60s" \
op stop interval="0" timeout="60s" op promote timeout="120" op demote
timeout="120" \
op monitor role="Master" timeout="30" interval="5" on-fail="restart" op monitor
role="Slave" \
timeout="30" interval="10"
node1# pcs resource create pandoraip ocf:heartbeat:IPaddr2 ip=$VIP
cidr_netmask=24 \
op monitor interval=20s
node1# pcs resource master master_pandoradb pandoradb meta master-max="1" \
master-node-max="1" clone-max="2" clone-node-max="1" notify="true" \
globally-unique="false" target-role="Master" is-managed="true"
noode1# pcs constraint colocation add master master_pandoradb with pandoraip
node1# pcs constraint order promote master_pandoradb then start pandoraip
```

Check the status of the cluster:

```
node1# pcs status
    Cluster name: pandoraha
    Stack: corosync
    Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with
quorum
    Last updated: Fri Jun 8 13:02:21 2018
    Last change: Fri Jun 8 13:02:11 2018 by root via cibadmin on node1

    2 nodes configured
    3 resources configured

    Online: [ node1 node2 ]

    Full list of resources:

     Master/Slave Set: master_pandoradb [pandoradb]
         Masters: [ node1 ]
         Slave: [ node2 ]
     pandoraip (ocf::heartbeat:IPaddr2): Started node1

    Daemon Status:
      corosync: active/disabled
      pacemaker: active/disabled
      pcsd: active/enabled
```

Both nodes should be online (`Online: [ node1 node2 ]`). Other values may be different from the example.

Configuring the two-node cluster with "non-root" user

It will be done similar to the previous way. The user credentials, previously explained, must have been copied, and the following steps must be carried out:

```
# All nodes:

  useradd <user>
  passwd <user>
  usermod -a -G haclient <user>

# Enable PCS ACL system
pcs property set enable-acl=true --force

# create role
pcs acl role create <role> description="RW role" write xpath /cib

# Create PCS user - Local user
pcs acl user create <user> <role>

# Login into PCS from ALL nodes
su - <user>
pcs status
Username: <username>
Password: *****

# Wait for 'Authorized' message, ignore output. Wait a second and retry 'pcs
status' command
```

Back to Pandora FMS Documentation Index