Deploying a simple Percona Cluster

October 30, 2018

This document describes the steps necessary to deploy a Percona cluster using two Centos/RHEL servers.

Executive Summary

All of these steps have been automated. You can execute them by editing `SERVER_LIST` variable at the top of the the `copy-files.sh` file found at the end of this document. Put the server that will act as the bootstrap, or master, server first and list the others after that. You should be able to access all these servers from the machine that you run the script on.

The `copy-files.sh` will create a `setup-primary.sh` script and use scp to copy it and the other files to the master server. It will also create similar setup scripts for the other servers in the `SERVER_LIST` and copy them along with the other files to their respective servers.

You should run the `setup-primary.sh` script first on the master server and then run the setup scripts on the other servers.

When this the process is complete, you can run `check-db-status.sh` to see if the server has properly joined the cluster.

Setting up database servers

This section describes the steps necessary to configure the database cluster

Pre-requisites

IP Addresses

For each server, get the IP address. You can get this from the following script or by hand using the `/sbin/ip` command.

```
for server in $SERVER_LIST; do
    echo $server $(ssh $server /sbin/ip -o -4 a | awk '!/ lo / {print $4}' | sed s,/.*
done
```

This will result in output similar to the following:

```
utility 10.0.0.101
web 10.0.0.102
```

Make sure the server that is to serve as the master DB is listed first and write this output into a file named `server-ips.txt`. If you do this by hand, make sure to follow the same format when creating the file.

Cluster config snippet

Each server will need the following snippet replaced in `/etc/percona-xtradb-cluster.conf.d/wsrep.c`. When the configuration snippet is added, `NODE_LIST` will be replaced with the IP addresses comma from the previous step. `THIS_NODE` will be replaced with the IP address of the machine being set up.

```
[mysqld]
# Path to Galera library
wsrep_provider=/usr/lib64/libgalera_smm.so

# In order for Galera to work correctly binlog format should be ROW
binlog_format=ROW

# MyISAM storage engine has only experimental support
default_storage_engine=InnoDB

# This InnoDB autoincrement locking mode is a requirement for Galera
innodb_autoinc_lock_mode=2

# SST method
wsrep_sst_method=xtrabackup-v2

# Cluster name
wsrep_cluster_name=pxc-cluster

# Authentication for SST method
wsrep_sst_auth="SSTUSER:SSTSECRET"
```

```
# Slave thread to use
wsrep_slave_threads=8

wsrep_log_conflicts

#pxc_strict_mode allowed values: DISABLED,PERMISSIVE,ENFORCING,MASTER
pxc_strict_mode=ENFORCING

# Cluster connection URL contains the IPs of node#1 & node#2
wsrep_cluster_address=gcomm://NODE_LIST

# This node's address
wsrep_node_address=THIS_NODE

# Log bin
log_bin=NODE_NAME-bin
```

Steps

Setting up the DB master (utility server)

1. Install MySQL on the RHEL7 server (utility) (ref):

```
sudo yum install -y http://www.percona.com/downloads/percona-release/redhat/0.1-6/perc
sudo yum remove -y MariaDB-common
sudo yum install -y Percona-XtraDB-Cluster-57
```

2. Set up the data dir on the large drive and fix what look like some bugs in the other scripts:

```
if [ ! -d /u01 ]; then
    sudo mkdir -p /u01
fi
sudo usermod -m -d /u01/mysql mysql
if [ ! -d /u01/mysql ]; then
    sudo mv /var/lib/mysql /u01/mysql
fi
grep -R -l /var/lib/mysql /etc/init.d/ /etc/percona-xtradb-cluster.c* /etc/my.cnf* \
     /etc/logrotate.d | xargs sudo sed -i s,/var/lib/mysql,/u01/mysql,g
grep -R -l /var/lock/subsys/mysqld /etc/init.d/mysql /etc/percona-xtradb-cluster.c* \
     /etc/my.cnf* | xargs sudo sed -i 's,/var/lock/subsys/mysqld,/var/run/mysqld,'
```

```
(echo '[mysql]'; echo 'socket=/u01/mysql/mysql.sock') | sudo tee -a /etc/my.cnf
sudo cp wsrep.cnf /etc/percona-xtradb-cluster.conf.d/wsrep.cnf
```

3. Configure selinux to allow mysql on the new drive (ref):

```
sudo yum install -y policycoreutils-python
sudo semanage fcontext -a -t mysqld_db_t "/u01/mysql(/.*)?"
sudo restorecon -Rv /u01/mysql
```

4. Configure selinux to allow pecona configuration files (ref):

```
sudo semanage fcontext -a -t mysqld_etc_t "/etc/percona-xtradb-cluster\.cnf"
sudo semanage fcontext -a -t mysqld_etc_t "/etc/percona-xtradb-cluster\.conf\.d(/.*)?"
sudo restorecon -v /etc/percona-xtradb-cluster.cnf
sudo restorecon -R -v /etc/percona-xtradb-cluster.conf.d/
```

5. Configure iptables and selinux to allow network traffic

```
sudo semanage port -a -t mysqld_port_t -p tcp 4568
sudo semanage port -m -t mysqld_port_t -p tcp 4444
sudo semanage port -m -t mysqld_port_t -p tcp 4567 # may have to use -a instead of -m
```

6. Create a text file named `PXC.te` for the custom SELinux configuration

```
module PXC 1.0;
require {
        type anon_inodefs_t;
        type kernel_t;
        type mysqld_safe_t;
        type mysqld_t;
        type port_t;
        type proc_net_t;
        type sysctl_net_t;
        type tmp_t;
# type unconfined_service_t;
        type unconfined_t;
        class dir search;
        class file { getattr open read write ioctl };
        class process { getattr setpgid };
        class system module_request;
        class tcp_socket { name_bind name_connect };
```

```
        class unix_stream_socket connectto;
}
#============= mysqld_t ==============
allow mysqld_t anon_inodefs_t:file write;
allow mysqld_t kernel_t:system module_request;
allow mysqld_t port_t:tcp_socket name_bind;
allow mysqld_t port_t:tcp_socket name_connect;
allow mysqld_t proc_net_t:file read;
allow mysqld_t proc_net_t:file { getattr open ioctl };
allow mysqld_t self:process { getattr setpgid };
allow mysqld_t self:unix_stream_socket connectto;
allow mysqld_t sysctl_net_t:dir search;
allow mysqld_t sysctl_net_t:file { getattr open read };
allow mysqld_t tmp_t:file write;
```

7. Compile and load the file in the previous step

```
checkmodule -M -m -o PXC.mod PXC.te
semodule_package -o PXC.pp -m PXC.mod
sudo semodule -i PXC.pp
```

8. Open the ports for network traffic

```
ports="3306 4444 4567 4568"

allowPort() {
    port=$1
    if [ -f /usr/bin/firewall-cmd ]; then
        sudo firewall-cmd --zone=public --add-port=$port/tcp --permanent
    else
        sudo sed -i "/-A INPUT -j REJECT/ i -A INPUT -p tcp --dport $port -j ACCEPT" \
             /etc/sysconfig/iptables
    fi
}

restartFirewall() {
    if [ -f /usr/bin/firewall-cmd ]; then
        sudo firewall-cmd --reload
    else
        sudo /etc/init.d/iptables restart
    fi
```

```
}

for port in $ports; do
    allowPort $port
done

restartFirewall
```

9. Start the server

```
sudo systemctl start mysql@bootstrap.service
```

10. Change the root password for mysql:

```
PW=$(sudo perl -ne '/temporary password is generated for root\@localhost: (.*)/ && {pr
echo "set password for root@localhost = '$NEWPW';"|mysql -u root -p"$PW" --connect-exp
```

Note that if you just cut-n-paste all the above you will end up with a password of "echo". You can paste up until `read -s NEWPW` and then type the password you want to use.

11. Set the password for the State Snapshot Transfer (SST) user with the right permissions:

```
(
cat<<EOF
  CREATE USER '$SSTUSER'@'localhost' IDENTIFIED BY '$SSTSECRET';
  GRANT PROCESS, RELOAD, LOCK TABLES, REPLICATION CLIENT
     ON *.* TO '$SSTUSER'@'localhost';
  FLUSH PRIVILEGES;
EOF
) | mysql -u root -p"$NEWPW"
```

12. Verify that cluster software is running:

```
getWsrep() {
    arg=$1

    WSREP=$(echo "show status like 'wsrep_$arg';" |
                mysql -u root -p"$NEWPW" 2>&1 |
                awk "/^wsrep_$arg/ {print \$2}")
```

```
    echo $WSREP
}
ready=$(getWsrep ready)
if [ "$ready" != "ON" ]; then
    echo "Not ready."
    exit 1
fi

connected=$(getWsrep connected)
if [ "$connected" != "ON" ]; then
    echo "Not connected."
    exit 1
fi

local_state_uuid=$(getWsrep local_state_uuid)
if [ -z "$local_state_uuid" ]; then
    echo "Could not find the local_state_uuid."
    exit 1
fi

local_state_comment=$(getWsrep local_state_comment)
if [ "$local_state_comment" != "Synced" ]; then
    echo "Local state is not synced."
    exit 1
fi

echo "Looks like everything is in place"
```

Until the cluster is set up, the above should end on "Not connected."

Setting up the DB replica (web server, RHEL6)

Repeat steps 1-10 from the DB master setup.

1. Start the database

```
sudo service mysql start
```

2. Update the root password by using step 10 above and add the sst user as in step 11.

3. Verify that the cluster is running by using the script in step 12 above.

Complete script

The following script will execute all scripts here in order. WEB and UTIL variables are expected to be replaced with the names of the relevant servers.

```
# Put the DB server to bootstrap on first
export SERVER_LIST="centos7-blank.default centos6.default"
export SSTUSER=sstuser
export SSTSECRET=s3cret

getIP() {
    name=$1
    ip=‘awk "/^$name / {print \\$2}" server-ips.txt‘

    echo $ip
}

./fill-ip.sh > server-ips.txt

echo -n Please enter a new root password for the database:
read -s NEWPW
echo
export NEWPW

NODE_LIST=$( (
    first=1
    for host in $SERVER_LIST; do
        if [ $first -ne 1 ]; then
            echo -n " "
        fi
        first=0
        echo -n ‘getIP $host‘
    done ) | sed "s/ /,/g;" )
export NODE_LIST

rm -f setup-primary.sh
for host in $SERVER_LIST; do
    FILES="server-ips.txt install-percona.sh PXC.te start-mysql.sh bootstrap-server.sh
                update-db-users.sh check-db-status.sh wsrep.cnf"
```

```
    node=‘getIP $host‘
    cat cluster-config-snippet.txt |
        sed "s,THIS_NODE,$node,g; s/NODE_LIST/$NODE_LIST/g; s/NODE_NAME/$host/g;
             s,SSTSECRET,$SSTSECRET,g; s,SSTUSER,$SSTUSER,g;" > wsrep.cnf

    SETUP=bogus
    if [ ! -f setup-primary.sh ]; then
        SETUP=setup-primary.sh
        (
            cat<<EOF
#!/bin/sh -e
export NEWPW=$NEWPW
export THIS_NODE=$node
export NODE_LIST=$NODE_LIST
export SSTUSER=$SSTUSER
export SSTSECRET=$SSTSECRET
./install-percona.sh
./bootstrap-server.sh
./update-db-users.sh;
./check-db-status.sh
EOF
        ) > $SETUP
    else
        SETUP=setup-$host.sh
        (
            cat<<EOF
#!/bin/sh -e
export NEWPW=$NEWPW
export THIS_NODE=$node
export NODE_LIST=$NODE_LIST
./install-percona.sh;
./start-mysql.sh;
./update-db-users.sh;
./check-db-status.sh
EOF
        ) > $SETUP
    fi
    chmod +x $SETUP
    scp -p $FILES $SETUP $host:
    echo '*************************************************************************
```

```
    echo '*************************************************************************
    echo '**
    echo "           Run $SETUP on $host"
    echo '**
    echo '*************************************************************************
    echo '*************************************************************************
done
```