CLUSTER ALTA DISPONIBILIDAD MARIADB GALERA HAPROXY

CentOS MariaDB

Alejandro Agroba Castro 2º ASIR

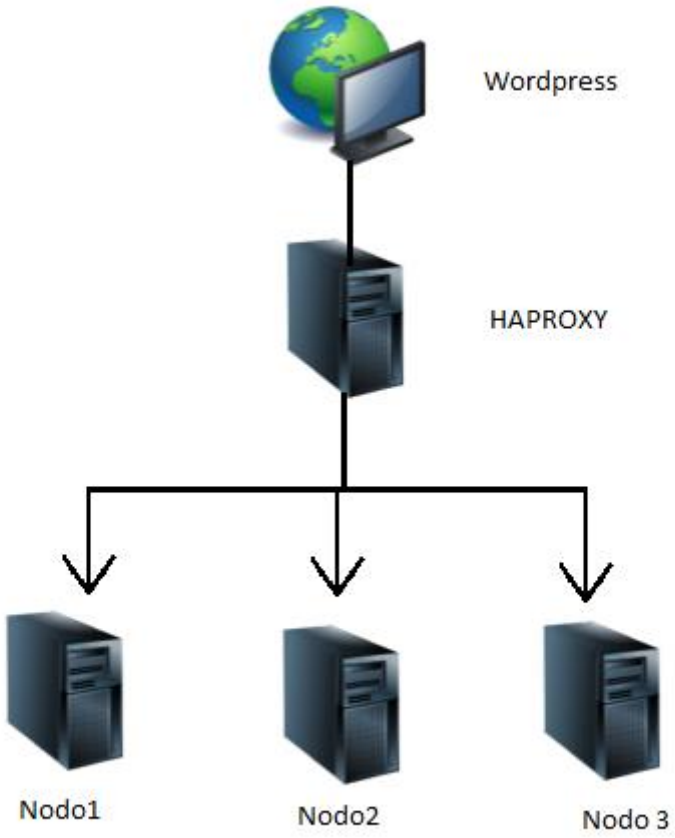
Índice de contenido

1.1 Alta disponibilidad ¿Qué es un cluster en alta disponibilidad?

Un cluster HA (High Availability) es un sistema orientado a ofrecer y garantizar servicios en Alta Disponibilidad, fiabilidad y de continuidad operativa. Se basa en máquinas redundantes (o nodos) que asumen el servicio cuando algún componente del sistema falla el otro sigue ofreciendo el servicio.

1.2Objetivos del proyecto

El objetivo de este proyecto es la instalación de un clúster MariaDB Galera en CentOS 7, con un balanceador de carga con HAProxy. Voy a utilizar 5 máquinas CentOS7, 3 de ellos serán los nodos servidores de bases de datos, 2 nodos estarán activos y 1 actúa como nodo de copia de seguridad. El cuarto servidor será utilizado para el balanceador de carga HAProxy. Para el algoritmo de balanceo de carga, voy a utilizar leastconn (Este algoritmo, entrega una nueva conexión al servidor que tenga la menor cantidad de conexiones en el momento. Least Connections trabaja mejor en ambientes donde los servidores y otros equipamientos tienen capacidades similares. La última máquina será un servidor web donde se instalará un Wordpress para comprobar el funcionamiento del balanceo de carga.



2.1 Servidores MariaDB Galera

MariaDB Galera Cluster es un cluster multi-maestro síncrono que se basa en MariaDB. Las ventajas:

- Replicación síncrona.
- Topología multi-maestro.
- Lectura y escritura en cada nodo del cluster.
- Control activo de los miembros del cluster.

Modificamos el fichero */etc/hosts* y añadimos los nombres y la direcciones de los nodos.

```
[root@localhost mariadb1]# nano /etc/hosts
192.168.1.134   mariadb1
192.168.1.135   mariadb2
192.168.1.136   mariadb3
```

Desactivamos Selinux ya que causa conflicto con MariaDB

```
[root@localhost mariadb1]# nano /etc/sysconfig/selinux
SELINUX=disabled
```

A continuación creamos un fichero con el repositorio en el directorio .repo

```
[root@localhost mariadb1]# nano /etc/yum.repos.d/mariadb.repo
# MariaDB 10.0 CentOS repository list - created 2015-07-09 14:56 UTC
# http://mariadb.org/mariadb/repositories/
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.0/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

Instalamos los paquetes que nos harán falta para seguir con el proyecto

```
[root@localhost mariadb1]# yum -y install MariaDB-Galera-server
MariaDB-client galera rsync xinetd

[root@mariadb2 mariadb2]# yum install
http://www.percona.com/downloads/percona-release/redhat/0.1-3/percona-
release-0.1-3.noarch.rpm

[root@mariadb2 mariadb2]# yum install percona-xtrabackup percona-
toolkit
```

2.2 Firewalld

Ahora abrimos los puertos de CentOS7 para los servicios que hemos instalado. Esto se hace con el siguiente comando, firewall-cmd.

- 3306 = Se utiliza para las conexiones de cliente de MySQL / MariaDB.
- 4567= Tráfico de replicación Galera Cluster.
- 873= Puertos rsync.
- 4444 = Para todos los demás State Transfer instantánea (SST).
- 9200 = Xinetd - Clustercheck.

```
 [root@localhost mariadb1]#systemctl start firewalld
[root@localhost mariadb1]#firewall-cmd --permanent --add-port=3306/tcp
[root@localhost mariadb1]#firewall-cmd --permanent --add-port=4567/tcp
[root@localhost mariadb1]#firewall-cmd --permanent --add-port=873/tcp
[root@localhost mariadb1]#firewall-cmd --permanent --add-port=4444/tcp
[root@localhost mariadb1]#firewall-cmd --permanent --add-port=9200/tcp
```

Recargamos el servicio para que los cambios tengan efecto

```
[root@localhost mariadb1]#firewall-cmd --reload
```

2.3 MariaDB Galera Cluster

Nos dirigimos al archivo de configuración para añadir los siguientes atributos que formarán nuestro cluster. Este fichero tenemos que modificarlos en los 3 nodos donde el atributo **wsrep_node_address** cambiará según la dirección IP del nodo.

```
[root@localhost mariadb1]#nano /etc/my.cnf.d/ server.cnf
[galera]
# Mandatory settings
wsrep_provider=/usr/lib64/galera/libgalera_smm.so   # Ruta libreria
wsrep_cluster_address="gcomm://192.168.1.134,192.168.1.135,192.168.1.1
36"   #  Ip de los nodos
binlog_format=row
default_storage_engine=InnoDB #Motor almacenamiento
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0
# Nombre del cluster
wsrep_cluster_name="mariadb_cluster"
# Añadir las direcciones IP de los nodos
wsrep_node_address="192.168.1.134"
# Metodo de replicación
wsrep_sst_method=rsync
# Optional setting
#wsrep_slave_threads=1
#innodb_flush_log_at_trx_commit=0
```

Iniciar el servicio para crear el cluster con el siguiente comando.

```
[root@localhost mariadb1]#/etc/rc.d/init.d/mysql bootstrap
```

```
[root@mariadb1 mariadb1]# /etc/rc.d/init.d/mysql bootstrap
Bootstrapping the cluster.. Starting MySQL.161216 18:24:16 mysqld_safe Logging t
o '/var/lib/mysql/mariadb1.err'.
. SUCCESS!
```

Ahora vamos a por los otros dos nodos donde solo tenemos que levantar el servicio de MySQL

```
root@localhost mariadb2]# /etc/init.d/mysql start
root@localhost mariadb3]# /etc/init.d/mysql start
```

```
[root@mariadb2 mariadb2]# service mysql start
Starting MySQL.161216 18:31:13 mysqld_safe Logging to '/var/lib/mysql/mariadb2.e
rr'.
..SST in progress, setting sleep higher. SUCCESS!
```

```
[root@mariadb3 mariadb3]# /etc/init.d/mysql start
Starting MySQL.161216 18:36:03 mysqld_safe Logging to '/var/lib/mysql/mariadb3.e
rr'.
.. SUCCESS!
```

Podemos comprobar el estado de los log entrando en el siguiente fichero:

```
[root@mariadb2 mariadb2]# cat /var/lib/mysql/mariadb2.err
```

Comprobamos el tamaño del cluster para que ver que los 3 nodos están activos

```
[root@mariadb1 mariadb1]# mysql -u root -p -e "SHOW STATUS LIKE
'wsrep_cluster_size'"
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
```

2.4 Instalación y configuración Clustercheck

Clustercheck es un script capaz de hacer que un proxy pueda monitorizar el servidor de MariaDB. Vamos con la configuración:

Lo primero que debemos hacer es ir al directorio /tmp y descargarnos el script

```
[root@mariadb1mariadb1]# cd /tmp
[root@mariadb1mariadb1]# wget https://raw.githubusercontent.com/olafz/
percona-clustercheck/master/clustercheck
```

A continuación debemos darle permiso de ejecución y moverlo al directorio /usr/bin

```
[root@mariadb1mariadb1]# chmod + x clustercheck
[root@mariadb1mariadb1]# mv clustercheck / usr / bin /
[root@mariadb1mariadb1]# nano /etc/xinetd.d/mysqlchk
# default: on
# description: mysqlchk
service mysqlchk
{
        disable = no
        flags = REUSE
        socket_type = stream
        port = 9200             # Puerto utilizado por xinetd para
clustercheck
        wait = no
        user = nobody
        server = /usr/bin/clustercheck
        log_on_failure += USERID
        only_from = 0.0.0.0/0
        per_source = UNLIMITED
}
```

Añadimos Clustercheck a la lista de servicios. Debemos buscar el servicio que está escuchando en el puerto 9200 y comentarlo y añadir una nueva línea para mysqlchk.

```
[root@mariadb1mariadb1]# nano /etc/services
#wap-wsp        9200/tcp                # WAP connectionless session service
#wap-wsp        9200/udp                # WAP connectionless session service
mysqlchk        9200/tcp                # mysqlchk
```

Iniciamos el servicio de xinetd que nos muestra el estado del cluster para saber si le están llegando las peticiones

```
[root@mariadb1mariadb1]# systemctl start xinetd
```

Para poder comprobar que Clustercheck funciona correctamente, debemos crear un nuevo usuario en MySQL. Accedemos a MariaDB y crear un nuevo usuario con el nombre "clustercheckuser" y la contraseña "clustercheckpassword!". Es el usuario y la contraseña que viene definida en el script que vamos a utilizar

```
MariaDB [(none)]> GRANT PROCESS ON *.*TO'clustercheckuser'@'localhost'
IDENTIFIED BY 'clustercheckpassword!';
```

Comprobamos que todos los nodos están sincronizados

```
[root@mariadb1 mariadb1]# /usr/bin/clustercheck
HTTP/1.1 200 OK
Content-Type: text/plain
Connection: close
Content-Length: 40

Percona XtraDB Cluster Node is synced.
```

2.5 Instalación y configuración HAProxy

Modificamos el fichero */etc/hosts* y añadimos los nombres y la direcciones de los nodos.

```
[root@localhost clustercheck]# nano /etc/hosts
192.168.1.134   mariadb1
192.168.1.135   mariadb2
192.168.1.136   mariadb3
```

Instalamos el HAProxy

Modificamos el fichero */etc/hosts* y añadimos los nombres y la direcciones de los nodos.

```
[root@localhost clustercheck]# yum install haproxy
```

Quitamos los comentarios de los puertos UDP.

```
[root@localhost clustercheck]# nano /etc/rsyslog.conf
# Provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 51
```

Añadir la configuración en el directorio HAProxy rsyslog.d donde indicamos los log de acceso e información guardamos y reiniciamos el servicio

```
[root@localhost clustercheck]# nano /etc/rsyslog.d/haproxy.conf
local2.=info     /var/log/haproxy-access.log
local2.notice    /var/log/haproxy-info.log
[root@localhost clustercheck]# systemctl restart rsyslog
```

Creamos el nuevo fichero de configuración de HAProxy. Primero hacemos una copia de seguridad para que en caso de equivocación poder restaurar el archivo original.

```
[root@localhost clustercheck]# cd /etc/haproxy/
[root@localhost clustercheck]# mv haproxy.cfg haproxy.cfg.orig
```

```
[root@localhost clustercheck]#nano /etc/haproxy/haproxy.cfg
global
        log 127.0.0.1   local2
        maxconn 1024
        user haproxy
        group haproxy
        daemon
        stats socket /var/run/haproxy.sock mode 600 level admin
# Creamos fichero del socket para HAProxy


defaults
        log     global
        mode    http
        option  tcplog
        option  dontlognull
        retries 3
        option  redispatch
        maxconn 1024
        timeout connect 5000ms
        timeout client 50000ms
        timeout server 50000ms


listen mariadb_cluster 0.0.0.0:3306
## MariaDB balanceo leastconn – El cluster escucha en el puerto 3030.
        mode tcp
        balance leastconn
        option httpchk
        server mariadb1 192.168.1.134:3306 check port 9200
        server mariadb2 192.168.1.135:3306 check port 9200
```

```
        server mariadb3 192.168.1.136:3306 check port 9200 backup
# Utilizamos el nodo 3 como copia de seguridad
```

```
listen stats 0.0.0.0:9000

## HAProxy stats web gui running on port 9000 - username and password: howtoforge.

        mode http

        stats enable

        stats uri /stats

        stats realm HAProxy\ Statistics

        stats auth howtoforge:howtoforge

        stats admin if TRUE
```

Añadimos los Puerto *3030* es el puerto MariaDB que balancea la carga y el puerto *9000* es el puerto para la web-GUI, que permite el seguimiento de HAProxy desde el navegador.

```
[root@localhost clustercheck]# systemctl start firewalld
[root@localhost clustercheck]# firewall-cmd --permanent --add-port=9000/tcp
[root@localhost clustercheck]# firewall-cmd --permanent --add-port=3030/tcp
```

Reiniciamos el servicio.

```
[root@localhost clustercheck firewall-cmd –reload
```

2.6 Pila LAMP

LAMP es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Linux, el sistema operativo.
- Apache, el servidor web;
- MySQL/MariaDB, el gestor de bases de datos;
- Perl, PHP, o Python, los lenguajes de programación.

La combinación de estas tecnologías es usada principalmente para definir la infraestructura de un servidor web. A pesar de que el origen de estos programas de código abierto no han sido específicamente diseñado para trabajar entre sí, la combinación se popularizó debido a su bajo coste de adquisición y ubicuidad de sus componentes (ya que vienen pre-instalados en la mayoría de las distribuciones Linux). Cuando son combinados, representan un conjunto de soluciones que soportan servidores de aplicaciones.

2.6.1 Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux), Microsoft Windows, Macintosh, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. El servidor Apache es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server (httpd). Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido.

Para la instalación de Apache en Centos7 solo debemos escribir el siguiente comando. Una vez instalado iniciamos el servicio con la siguiente orden. La última instrucción es para que se inicie automáticamente el servicio cuando arrancamos el sistema.

```
[root@localhost servidorweb]# yum install httpd
[root@localhost servidorweb]# systemctl start httpd.service
[root@localhost servidorweb]# systemctl start httpd.service
```

2.6.2 Mariadb

Como hemos visto anteriormente Mariadb se instala con una sencilla orden. Una vez instalado iniciamos el servicio con el siguiente comando. Ahora que nuestra base de datos MySQL se está ejecutando, ejecutamos un script de seguridad que eliminará algunos valores predeterminados y bloqueará el acceso a nuestro sistema de base de datos. Por último añadimos la siguiente orden para que MariaDB se inicia automáticamente cuando arranquemos el sistema

```
[root@localhost servidorweb]# yum install  mariadb-server mariadb
[root@localhost servidorweb]# systemctl start mariadb
[root@localhost servidorweb]# mysql_secure_installation
[root@localhost servidorweb]# systemctl enable mariadb.service
```

2.6.3 PHP

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

Fue creado originalmente por Rasmus Lerdorf en el año 1995. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP. Este lenguaje forma parte del software libre publicado bajo la licencia PHP, que es incompatible con la Licencia Pública General de GNU debido a las restricciones del uso del término PHP.

Para instalar PHP introducimos el siguiente comando. Una vez instalado debemos reiniciar el servicio de apache para que la configuración tenga efecto.

```
[root@localhost servidorweb]# yum install php php-mysql
[root@localhost servidorweb]# systemctl restart httpd.service
```

2.6.4 Wordpress

WordPress es un sistema de gestión de contenidos o CMS (por sus siglas en inglés, Content Management System) enfocado a la creación de cualquier tipo de sitio web. Originalmente alcanzó una gran relevancia usado para la creación de blogs, para convertirse con el tiempo en una de las principales herramientas para la creación de páginas web comerciales. Ha sido desarrollado en el lenguaje PHP para entornos que ejecuten MySQL y Apache, bajo licencia GPL y es software libre. Las causas de su enorme crecimiento son, entre otras, su licencia, su facilidad de uso y sus características como gestor de contenidos.

El primer paso que vamos a dar está en preparación. WordPress utiliza una base de datos relacional para administrar la información del sitio y de sus usuarios, pero necesitamos crear una base de datos y un usuario para WordPress pueda trabajar.

```
[root@localhost servidorweb]# mysql –u root –p

MariaDB [(none)] > CREATE DATABASE wordpress;

MariaDB [(none)] > CREATE USER wordpress@% IDENTIFIED BY 'usuario';

MariaDB [(none)] > GRANT ALL PRIVILEGES ON wordpress.* TO usurio@%
IDENTIFIED BY 'usurio';

MariaDB [(none)] > FLUSH PRIVILEGES;
```

Antes de descargar WordPress, hay un módulo PHP que debemos instalar para asegurarnos de que funciona correctamente. Sin este módulo, WordPress no podrá cambiar el tamaño de las imágenes para crear miniaturas. Podemos obtener ese paquete directamente desde los repositorios predeterminados de CentOS. Una vez instalado el paquete vamos a descargarnos el paquete de wordpress desde su página principal.

```
[root@localhost servidorweb]# yum install php-gd

[root@localhost servidorweb]# wget http://wordpress.org/latest.tar.gz
```

Una vez descomprimido solo nos queda modificar el archivo de configuración de wordpress e indicar los parámetros de (Nombre de la base de datos, usuario, contraseña, y el nombre del equipo donde estará la base de datos).

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
```

```
define('DB_USER', 'wordpress');

/** MySQL database password */
define('DB_PASSWORD', 'usuario');

/** MySQL hostname */
define('DB_HOST', 'balanceador');
```

Muy importante desactivar la política de selinux de permissive a disabled y reiniciar el equipo porque hasta que no se reinicia no tiene efecto ya que crea un conflicto con el puerto del haproxy.

```
[root@servidor servidorweb]# nano /etc/sysconfig/selinux

SELINUX = disabled
```

3.1 Replicación MariaDB

Para comprobar que funciona correctamente la replicación probamos a crear una base de datos en cada uno de los nodos.

```
MariaDB [(none)]> create database Mariadb1 #Lo hacemos en el nodo1

MariaDB [(none)]> create database Mariadb2 #Lo hacemos en el nodo2

MariaDB [(none)]> create database Mariadb3 #Lo hacemos en el nodo2
```

Comprobamos que se han creado todas las bases de datos en los 3 nodos

```
MariaDB [(none)]> create database Mariadb3;
Query OK, 1 row affected (0.05 sec)

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| Mariadb1           |
| Mariadb2           |
| Mariadb3           |
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
7 rows in set (0.00 sec)
```

3.2 Acceso a la base de datos desde HAProxy

Comprobamos que desde el balanceador de carga HAProxy podemos acceder a todos los nodos de Mariadb.

```
[root@balanceador balanceador]# mysql -u root -p -h mariadb1
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 1611
Server version: 10.0.28-MariaDB-wsrep MariaDB Server, wsrep_25.16.rc3fc46e

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| Mariadb1           |
| Mariadb2           |
| Mariadb3           |
| iesgn              |
| iesgn1             |
| information_schema |
| mysql              |
```

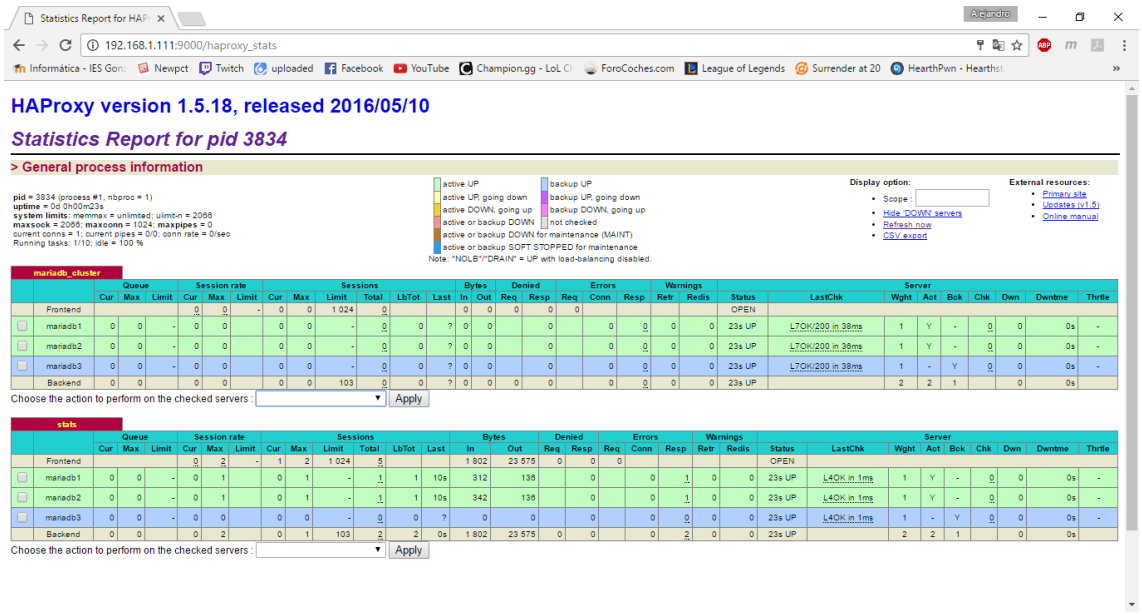
3.3 Balanceo de carga desde Servidorweb

Vamos a comprobar que nuestro servidor web funciona correctamente para ello vamos a acceder a los nodos desde nuestro servidor con unas sentencias para ver cómo funciona el algoritmo que hemos configurado. En la imagen vemos como el nodo alterna en cada consulta.

```
[root@servidor servidor]# mysql -u wordpress -p -h balanceador -P 3306 -e "selec
t @@hostname"
Enter password:
+------------+
| @@hostname |
+------------+
| mariadb2   |
+------------+
[root@servidor servidor]# mysql -u wordpress -p -h balanceador -P 3306 -e "selec
t @@hostname"
Enter password:
+------------+
| @@hostname |
+------------+
| mariadb1   |
+------------+
```

3.3 Monitorización de HAProxy

Comprobamos que desde cualquier nodo o desde el mismo balanceador podemos monitorizar el estado de los nodos. Para ello en la barra de direcciones del indicamos la IP del balanceador y el puerto que en este caso HAProxy utiliza el 9000 y el directorio que /haproxy_stats, una vez introducidos introducimos el usuario y contraseña que hemos configurado en el archivo de configuración.



Vemos como Mariadb1 y Mariadb2 actúan como nodos activos mientras que Mariadb3 está configurado como nodo de respaldo o (backup).

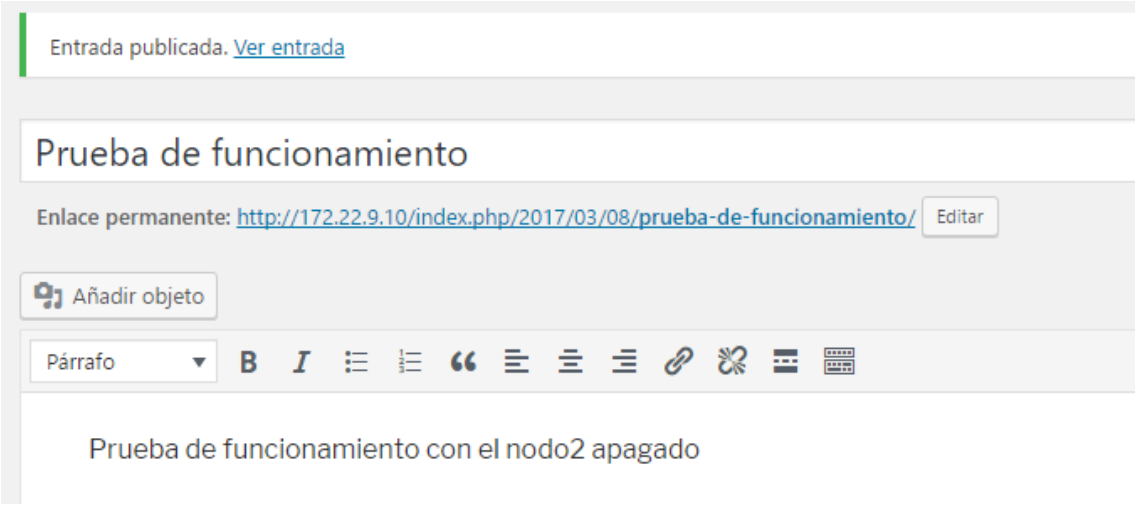
3.4 Pruebas de funcionamiento Wordpress

Otra comprobación que vamos a hacer es que vamos a para uno de los dos nodos de mariadb y crearemos una entrada en el WordPress para comprobar que se crea sin problemas.

Lo primero que hacemos es para el nodo de Mariadb2

```
[root@mariadb2 mariadb2]# service mysql stop
Shutting down MySQL.... SUCCESS!
```

Ahora vamos a crear una nueva entrada en Wordpress

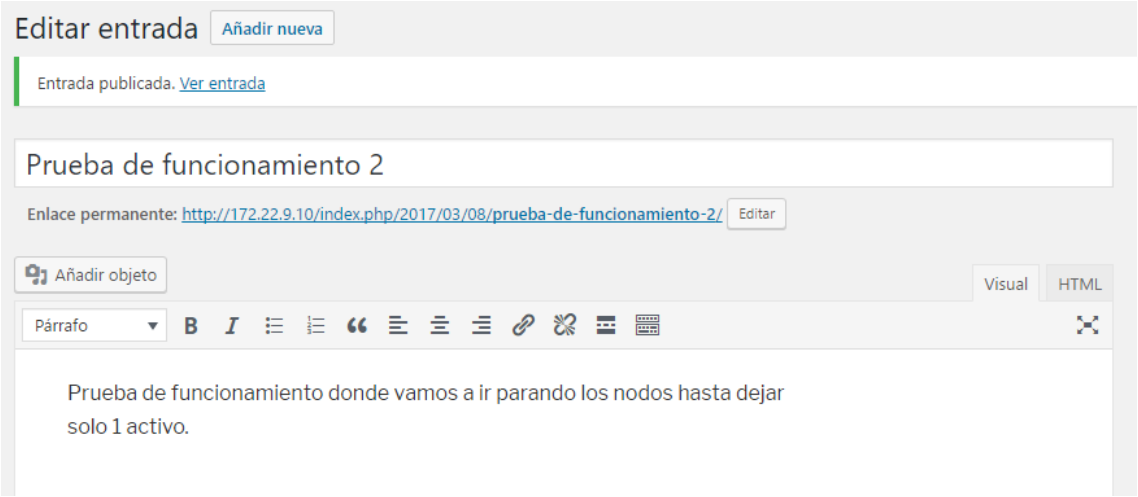


Aquí vemos en la imagen como la entrada se ha creado sin ningún problema.

Volvemos a levantar el Nodo2 y entramos a la base de datos de Wordpress y hacemos una consulta para ver que la entrada se ha creado en la tabla wp_posts.

```
MariaDB [wordpress]>
MariaDB [wordpress]> use wordpress
Database changed
MariaDB [wordpress]> select * from wp_posts;
```

Otra prueba de funcionamiento que vamos a realizar es crear una entrada en nuestro Wordpress y vamos a ir parando los nodos y que la entrada sigue siendo visible.



Ahora vamos a ir parando los nodos donde veremos que se sigue viendo la noticia. Primero paramos el Nodo1 y luego el Nodo2.

```
[root@mariadb1 mariadb1]# service mysql stop
Shutting down MySQL..... SUCCESS!
```

```
[root@mariadb2 mariadb2]# service mysql stop
Shutting down MySQL..... SUCCESS!
```

Ahora comprobamos que la noticia sigue siendo visible.

ENTRADAS

8 MARZO, 2017 EDITAR

Prueba de funcionamiento 2

Prueba de funcionamiento donde vamos a ir parando los nodos hasta dejar solo 1 activo.

4. Conclusión

El cluster formado por MariaDB Galera Cluster es una buena solución para montar un sistema de base de datos. MariaDB Galera Cluster es compatible con los motores de almacenamiento InnoDB y XtraDB, proporciona replicación automática y permite automática incorporación de nuevos nodos.

5. Bibliografía

https://www.howtoforge.com/tutorial/how-to-setup-haproxy-as-load-balancer-for-mariadb-on-centos-7/

https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-on-centos-7`

https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-centos-7

https://www.centos.org/docs/5/html/5.1/Deployment_Guide/sec-sel-enable-disable.html