MariaDB

Balanceador Galera Haproxy

Ahora procederemos a configurar un balanceador en alta disponibilidad usando HAProxy y Keepalived

En primer lugar instalaremos los paquetes necesarios

```
apt install haproxy net-tools keepalived
```

Procederemos a configurar HAProxy, para ello deberemos de crear un usuario en nuestro Cluster de Galera, para las peticiones de conectividad.

```
CREATE USER 'haproxy'@'%';
GRANT PROCESS ON *.* TO 'haproxy'@'%';
```

Ahora configuraremos HAProxy Editamos el archivo de configuración

```
global
        log /dev/log    local0
        log /dev/log    local1 notice
        chroot /var/lib/haproxy
        stats socket /run/haproxy/admin.sock mode 660 level admin expose-fd listeners
        stats timeout 30s
        user haproxy
        group haproxy
        daemon

        # Default SSL material locations
        ca-base /etc/ssl/certs
        crt-base /etc/ssl/private

        # See: https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=interm
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE
        ssl-default-bind-ciphersuites TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
        log     global
        mode    http
        option  httplog
        option  dontlognull
        timeout connect 5000
        timeout client  50000
        timeout server  50000
        errorfile 400 /etc/haproxy/errors/400.http
        errorfile 403 /etc/haproxy/errors/403.http
        errorfile 408 /etc/haproxy/errors/408.http
```

```
        errorfile 500 /etc/haproxy/errors/500.http
        errorfile 502 /etc/haproxy/errors/502.http
        errorfile 503 /etc/haproxy/errors/503.http
        errorfile 504 /etc/haproxy/errors/504.http

listen  galera_cluster
    bind 0.0.0.0:3306
    mode tcp
    #option tcplog
    option tcpka
    option mysql-check user haproxy
    balance source
    server db1 192.168.15.221:3306 check
    server db2 192.168.15.222:3306 check
    server db3 192.168.15.223:3306 check

listen stats
    bind 0.0.0.0:8080
    mode http
    option httplog
    stats enable
    stats uri /
    stats realm Strictly\ Private
    stats auth admin:MIPASSWORD
```

En las opciones de configuración pondremos en el balance source, existe la opción de roundrobin, o leastconn, pero teniendo en cuenta que es un sistema de tolerancia y no de reparto de carga, la mejor opción es la de source.

Una vez configurado, procederemos a probar.

En primer lugar arrancaremos el HAProxy

```
service haproxy start
```

Suponiendo que la IP de nuestro HAProxy sea la 192.168.15.222, desde cualquier máquina de la red, ejecutaremos:

```
mysql -uroot -pMYPASSWORD -h192.168.15.222  -e "show variables like 'wsrep_node_name' ;"
```

Nos devolverá algo así

```
+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| wsrep_node_name | Nodo1 |
+-----------------+-------+
```

Si tenemos algún fallo del tipo "Layer7 wrong status: Host '193.168.15.224' is blocked because of many connection errors; unblock with 'mariadb-admin flush-hosts'" En la interfaz gráfica

ejecutaremos el comando:

```
mariadb-admin flush-hosts
```

Interfaz gráfica (en el puerto 8080)

A continuación dejaremos el servicio configurado para que arranque con nuestra máquina

```
systemctl enable haproxy
```

Ahora que esto funciona, vamos con el keepalived.

Para ello la IP que hemos configurado (La 192.168.15.224) será nuestra IP flotante, es decir se mantendrá como IP de acceso independientemente de a que servidor HAProxy ataquemos.

Por esto, vamos a repetir la configuración en otra máquina a la cual le asignaremos la IP 192.168.15.226, y la que acabamos de configurar será la 192.168.15.225. Esto puede resultar un poco lioso, pero vamos a ver porqué. Cada equipo tiene una IP la 225 en el que acabamos de configurar, y la 226 en otro equipo nuevo.

Ambos tienen IP diferentes, pero comparten una IP global a través del Keealived.

```
nano /etc/keepalived/keepalived.conf
```

Y escribiremos esto:

```
global_defs {
   notification_email {
     sistemas@ateinco.com.com
   }
   notification_email_from sistemas@ateinco.com
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_script chk_haproxy {
    script "killall -0 haproxy"
    interval 1
    weight -2
}
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
    mcast_src_ip 192.168.15.225
```

```
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 876543
    }
    virtual_ipaddress {
        192.168.15.224/24 dev eth0 label eth0:1
    }
    track_interface {
        eth0
    }
    track_script {
        chk_haproxy
    }
}
```

Como vemos el equipo en la configuración de Keepalived tiene dos IP (la 225 y la 224) y un valor importante el priority que hemos puesto en 100

Ahora modificaremos el fichero /etc/sysctl.conf

Y agregaremos este valor

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

Además comprobaremos que la IP de nuestra tarjeta de red y el nombre de la interfaz están bien configurados en nuestro ejemplo de keepalived hemos puesto eth0, pero puede ser ens18, ens19, etc. Ahora arrancamos keepalived

```
systemctl start keepalived
```

Y comprobamos que todo funciona, volvemos a ejecutar:

```
mysql -uroot -pMYPASSWORD -h192.168.15.222  -e "show variables like 'wsrep_node_name' ;"
```

Si todo está correcto habilitamos keepalived al inicio

```
systemctl enable keepalived
```

Ahora ya tenemos un nodo configurado. Vamos al otro.

Básicamente la configuración es gemela a la de este nodo, salvo por la configuración de keepalived

Tendremos que cambiar la IP que será la 192.168.15.226 y la priority que la pondremos a 99

```
global_defs {
   notification_email {
     sistemas@ateinco.com.com
```

```
   }
   notification_email_from sistemas@ateinco.com
   smtp_server 127.0.0.1
   smtp_connect_timeout 30
   router_id LVS_DEVEL
}
vrrp_script chk_haproxy {
    script "killall -0 haproxy"
    interval 1
    weight -2
}
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
    mcast_src_ip 192.168.15.225
    priority 99
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 876543
    }
    virtual_ipaddress {
        192.168.15.224/24 dev eth0 label eth0:1
    }
    track_interface {
        eth0
    }
    track_script {
        chk_haproxy
    }
}
```

Instalación de MariaDB

Instalación de MariaDB en Debian

Instalación

Ejecutaremos un update para asegurar que todos los paquetes está en la última versión, y que hay conectividad con el repositorio de Debian

```
apt-get update
```

A continuación ejecutaremos:

Para realizar una instalación completa

```
apt-get install mariadb
```

Para instalar sólo el server

```
apt-get install mariadb-server
```

para que MariaDB se ejecute en el inicio del sistema:

```
systemctl enable mariadb
```

Y a continuación arrancamos MariaDB

```
systemctl start mariadb
```

Verificamos que el servicio está arrancado

```
root@mail:~# service mariadb status
```

Nos devolverá algo así:

```
root@mail:~# service mariadb status
● mariadb.service - MariaDB 10.1.44 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)
```

```
   Active: active (running) since Fri 2020-02-21 21:39:04 CET; 12h ago
     Docs: man:mysqld(8)
           https://mariadb.com/kb/en/library/systemd/
 Main PID: 1309 (mysqld)
   Status: "Taking your SQL requests now..."
    Tasks: 35 (limit: 4915)
   CGroup: /system.slice/mariadb.service
           └─1309 /usr/sbin/mysqld

feb 21 21:38:53 mail systemd[1]: Starting MariaDB 10.1.44 database server...
feb 21 21:38:56 mail mysqld[1309]: 2020-02-21 21:38:56 140676162280576 [Note] /usr/sbin/mysqld (
feb 21 21:39:04 mail systemd[1]: Started MariaDB 10.1.44 database server.
feb 21 21:39:05 mail /etc/mysql/debian-start[2640]: /usr/bin/mysql_upgrade: the '--basedir' opti
feb 21 21:39:05 mail /etc/mysql/debian-start[2640]: Looking for 'mysql' as: /usr/bin/mysql
feb 21 21:39:05 mail /etc/mysql/debian-start[2640]: Looking for 'mysqlcheck' as: /usr/bin/mysqlc
feb 21 21:39:05 mail /etc/mysql/debian-start[2640]: This installation of MySQL is already upgrad
feb 21 21:39:05 mail /etc/mysql/debian-start[2756]: Checking for insecure root accounts.
feb 21 21:39:05 mail /etc/mysql/debian-start[2763]: Triggering myisam-recover for all MyISAM tab
```

Asegurando MariaDB

Para mejorar la seguridad de la instalación de MariaDB ejecuta el script mysql_secure_installation:

```
mysql_secure_installation
```

El script te pedirá que establezcas una contraseña para la cuenta root, elimina el usuario anónimo, restringe el acceso del usuario root a la máquina local y elimina la base de datos de prueba.

Al final el script recargará las tablas de privilegios asegurando que todos los cambios surtan efecto inmediatamente

```
root@mail:~# mysql_secure_installation
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n] n
 ... skipping.
```

```
By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

Instalar Cluster MariaDB con Galera

Instalación de un cluster de MariaDB con Galera

Requisitos previos

Necesitaremos al menos 2 servidores, preferiblemente 3, en los que instalaremos MariaDB Una vez instalado MariaDB en los tres servidores, para que cualquier instalación en cluster funcione, debemos de instalar ntp, y asegurarnos que todos los servidores tienen la hora sincronizada. Esto lo conseguimos instalando NTP y configurándolo. Una vez que todos los servidores estén sincronizados, procederemos a comprobar que el sistema de índices de nuestra base de datos MariaDB es el correcto (debe ser InnoDB) Para ello accedemos a cada uno de los servidores mediante En una instalación limpia de Linux (en este caso Debian), deberemos proceder a ejecutar los procesos siguientes:

Actualización

Ejecutaremos un update parta asegurar que todos los paquetes está en la última versión, y que hay conectividad con el repositorio de Debian

```
apt-get update
```

Configuración del servicio NTP

Seguiremos los pasos que se explican en Configurar NTP en Debian

Instalación de MariaDB

Procederemos como se explica en Instalación MariaDB para cada uno de los nodos del cluster.

Una vez que todos los servidores estén sincronizados, procederemos a comprobar que el sistema de índices de nuestra base de datos MariaDB es el correcto (debe ser InnoDB) Para ello accedemos a cada uno de los servidores mediante

```
mysql -u root
```

Nos aparecerá el prompt

```
MariaDB [(none)]>
```

Ejecutamos el comando

```
show variables like 'default_storage_engine';
```

Nos aparecerá el resultado de la consulta

```
MariaDB [(none)]> show variables like 'default_storage_engine';
+------------------------+--------+
| Variable_name          | Value  |
+------------------------+-------+|
| default_storage_engine | InnoDB |
+------------------------+--------+
>1 row in set (0.00 sec)
```

En este caso vemos que el motor es InnoDB

Configuración del Cluster

Una vez instalado en todos los nodos del cluster, procederemos a instalar Galera

```
apt install galera-3
```

Y a continuación editaremos el fichero de configuración del servicio de Mysql para cada uno de los nodos del cluster

```
nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Aparecerá el fichero e configuración de MariaDB (Mysql), y verificaremos los siguientes parametros:

```
# this is only for the mysqld standalone daemon
[mysqld]
####################################
##   A PARTIR DE AQUI
####################################
# Galera Cluster configurations
wsrep_on = ON
wsrep_provider = /usr/lib/galera/libgalera_smm.so
wsrep_cluster_address = "gcomm://192.168.250.191,192.168.250.192,192.168.250.193"
default_storage_engine = InnoDB
binlog_format = row
innodb_autoinc_lock_mode = 2
innodb_force_primary_key = 1
innodb_doublewrite = 1
wsrep_cluster_name = MariadbCluster
wsrep_node_name = Nodo2
wsrep_node_address = "192.168.250.192"
innodb_flush_log_at_trx_commit=0

####################################
## Hay que comentar esta linea ya que de lo contrario, el servidor no responderá a peticiones ex
####################################
#bind-address           = 127.0.0.1
```

```
#bind-address           = 127.0.0.1
```

Las IP proporcionadas habrá que sutituirlas por sus respectivas IP Hay que tener cuidado con el apartado

```
innodb_force_primary_key = 1
```

En muchos casos nos puede dar problemas a la hora de crear tablas. Para desactivarlo temporalmente podemos usar

```
ateinco@db01:~#mysql -u root
MariaDB [(none)]>set global innodb_force_primary_key = 0;
```

Deberemos modificar el fichero en todos los servidores cambiando los parámetros

```
wsrep_node_name y wsrep_node_address
```

Arrancar el cluster

Instalaremos el apparmor

```
apt install apparmor apparmor-profiles apparmor-utils
```

A continuación ejecutaremos lo siguiente:

```
cd /etc/apparmor.d/disable/
ln -s /etc/apparmor.d/usr.sbin.mysqld
systemctl restart apparmor
systemctl stop mariadb
galera_new_cluster
systemctl restart mariadb
```

Otra forma de ejecutarlo desde una sóla línea

Una vez que hemos modificado el fichero en todos los servidores, procederemos a modificar los servicios en los tres servidores.

```
cd /etc/apparmor.d/disable/
ln -s /etc/apparmor.d/usr.sbin.mysqld
systemctl restart apparmor
systemctl stop mariadb
```

Ahora en el primer servidor ejecutaremos

```
galera_new_cluster
```

Y en los servidores restantes arrancaremos el servicio MariaDB

```
systemctl restart mariadb
```

Comprobaciones

Para comprobar que el cluster está funcionando, ejecutaremos el siguiente comando de MariaDB

```
mysql -u root
```

```
MariaDB [(none)]> show status like 'wsrep_cluster_size';
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
1 row in set (0.00 sec)
```

Como vemos tenemos un cluster con 3 nodos de MariaDB

MariaDB Galera Status

Comandos para verificar el estatus de un Cluster de Galera con MariaDB

Desde el cliente de la base de datos, puede verificar el estado de la replicación del conjunto de escritura en todo el clúster mediante consultas estándar. Las variables de estado relacionadas con la replicación del conjunto de escritura tienen el prefijo wsrep_, lo que significa que puede mostrarlas todas utilizando las siguiente consultas. Para ejecutarlas, lo primero que deberemos realizar es hacer login en el MariaDB De uno de los nodos

Login en MariaDB

```
root@mail:~# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 1883
Server version: 10.1.44-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]>
```

Comandos de Cluster

```
SHOW GLOBAL STATUS LIKE 'wsrep_%';
+------------------------+-------+
| Variable_name          | Value |
+------------------------+-------+
| wsrep_protocol_version | 5     |
| wsrep_last_committed   | 202   |
| ...                    | ...   |
| wsrep_thread_count     | 2     |
+------------------------+-------+
```

Comprobar la integridad del Cluster

El clúster tiene integridad cuando todos los nodos en él reciben y replican conjuntos de escritura de todos los demás nodos. El grupo comienza a perder integridad cuando esto se rompe, como cuando el grupo se cae, se divide o experimenta una situación de split-brain. Puede verificar la integridad del clúster utilizando las siguientes variables de estado:

wsrep_cluster_state_uuid

wsrep_cluster_state_uuid muestra el UUID del estado del clúster, que puede usar para determinar si el nodo es parte del clúster.

```
SHOW GLOBAL STATUS LIKE 'wsrep_cluster_state_uuid';

+--------------------------+--------------------------------------+
| Variable_name            | Value                                |
+--------------------------+--------------------------------------+
| wsrep_cluster_state_uuid | d6a51a3a-b378-11e4-924b-23b6ec126a13 |
+--------------------------+--------------------------------------+
```

Cada nodo en el clúster debe proporcionar el mismo valor. Cuando un nodo lleva un valor diferente, esto indica que ya no está conectado al resto del clúster. Una vez que el nodo restablece la conectividad, se vuelve a alinear con los otros nodos.

wsrep_cluster_conf_id

wsrep_cluster_conf_id muestra el número total de cambios de clúster que se han producido, que puede usar para determinar si el nodo forma parte o no del componente primario.

```
SHOW GLOBAL STATUS LIKE 'wsrep_cluster_conf_id';

+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| wsrep_cluster_conf_id | 32    |
+----------------------+-------+
```

Cada nodo en el clúster debe proporcionar el mismo valor. Cuando un nodo lleva un diferente, esto indica que el clúster está particionado. Una vez que el nodo restablece la conectividad de la red, el valor se alinea con los demás.

wsrep_cluster_size

wsrep_cluster_size muestra el número de nodos en el clúster, que puede usar para determinar si falta alguno.

```
SHOW GLOBAL STATUS LIKE 'wsrep_cluster_size';

+------------------+-------+
| Variable_name    | Value |
+------------------+-------+
| wsrep_cluster_size | 3    |
+------------------+-------+
```

Puede ejecutar esta verificación en cualquier nodo. Cuando la verificación devuelve un valor inferior al número de nodos en su clúster, significa que algunos nodos han perdido la conectividad

de red o han fallado.

wsrep_cluster_status

wsrep_cluster_status muestra el estado primario del componente del clúster en el que se encuentra el nodo, que puede usar para determinar si su clúster está experimentando una partición.

```
SHOW GLOBAL STATUS LIKE 'wsrep_cluster_status';

+----------------------+---------+
| Variable_name        | Value   |
+----------------------+---------+
| wsrep_cluster_status | Primary |
+----------------------+---------+
```

El nodo solo debe devolver un valor de Primario. Cualquier otro valor indica que el nodo es parte de un componente no operativo. Esto ocurre en casos de múltiples cambios de membership que resultan en una pérdida de quórum o en casos de situaciones de split-brain.

Cuando estas variables de estado verifican y devuelven los resultados deseados en cada nodo, el clúster está activo y tiene integridad. Lo que esto significa es que la replicación puede ocurrir normalmente en cada nodo. El siguiente paso es verificar el estado del nodo para asegurarse de que todos estén funcionando correctamente y puedan recibir conjuntos de escritura.

Comandos de Nodo

Además de verificar la integridad del clúster, también puedes monitorizar el estado de los nodos individuales. Esto muestra si los nodos reciben y procesan actualizaciones de los conjuntos de escritura del clúster y pueden indicar problemas que pueden impedir la replicación.

wsrep_ready

wsrep_ready muestra si el nodo puede aceptar consultas.

```
SHOW GLOBAL STATUS LIKE 'wsrep_ready';

+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| wsrep_ready   | ON    |
+---------------+-------+
```

Cuando el nodo devuelve un valor de ON, puede aceptar conjuntos de escritura del clúster. Cuando devuelve el valor OFF, casi todas las consultas fallan con el error:

```
ERROR 1047 (08501) Unknown Command
```

wsrep_connected wsrep_connected muestra si el nodo tiene conectividad de red con otros nodos.

```
SHOW GLOBAL STATUS LIKE 'wsrep_connected';

+-----------------+-------+
| Variable_name   | Value |
+-----------------+-------+
| wsrep_connected | ON    |
+-----------------+-------+
```

Cuando el valor está activado, el nodo tiene una conexión de red a uno o más nodos que forman un componente de clúster. Cuando el valor es OFF, el nodo no tiene conexión con ningún componente del clúster.

Nota: La razón de una pérdida de conectividad también puede relacionarse con una configuración incorrecta. Por ejemplo, si el nodo usa valores no válidos para los parámetros wsrep_cluster_address o wsrep_cluster_name.

Verifica el registro de errores para obtener el diagnóstico.

wsrep_local_state_comment

wsrep_local_state_comment muestra el estado del nodo en un formato legible para humanos.

SHOW GLOBAL STATUS LIKE 'wsrep_local_state_comment';

```
+---------------------------+--------+
| Variable_name             | Value  |
+---------------------------+--------+
| wsrep_local_state_comment | Joined |
+---------------------------+--------+
```

Cuando el nodo es parte del Componente primario, los valores de retorno típicos son Joining, Waiting on SST, Joined, Synced o Donor. Si el nodo es parte de un componente no operativo, el valor de retorno es Initialized.

Nota: Si el nodo devuelve cualquier valor distinto al que se muestra aquí, el comentario de estado es momentáneo y transitorio. Verifique la variable de estado nuevamente para una actualización.

En el caso de que cada variable de estado devuelva los valores deseados, el nodo está en funcionamiento. Esto significa que recibe conjuntos de escritura del clúster y los replica en tablas en la base de datos local.

Comprobación del estado de la replicación

La supervisión de la integridad del clúster y el estado del nodo puede mostrarle problemas que pueden prevenir o bloquear la replicación. Estas variables de estado ayudarán a identificar problemas de rendimiento e identificar áreas problemáticas para que pueda aprovechar al máximo

su clúster.

Nota: A diferencia de otras variables de estado, estas son diferenciales y se restablecen en cada comando FLUSH STATUS. Galera Cluster activa un mecanismo de retroalimentación llamado Flow Control para administrar el proceso de replicación. Cuando la cola local recibida de los conjuntos de escritura supera un cierto umbral, el nodo activa el Control de flujo (Flow Control) para pausar la replicación mientras se pone al día.

Puede monitorizar la cola local recibida y el Control de flujo utilizando las siguientes variables de estado:

wsrep_local_recv_queue_avg

wsrep_local_recv_queue_avg muestra el tamaño promedio de la cola local recibida desde la última consulta de estado.

```
SHOW STATUS LIKE 'wsrep_local_recv_queue_avg';

+--------------------------+----------+
| Variable_name            | Value    |
+--------------------------+----------+
| wsrep_local_recv_que_avg | 3.348452 |
+--------------------------+----------+
```

Cuando el nodo devuelve un valor superior a 0.0, significa que no puede aplicar conjuntos de escritura tan rápido como los recibe, lo que puede conducir a la aceleración de la replicación.

Nota Además de esta variable de estado, también puede usar wsrep_local_recv_queue_max y wsrep_local_recv_queue_min para ver los tamaños máximo y mínimo que el nodo registró para la cola local recibida.

wsrep_flow_control_paused

wsrep_flow_control_paused muestra la fracción de tiempo, desde la última vez que se llamó FLUSH STATUS, que el nodo se detuvo debido al control de flujo.

```
SHOW STATUS LIKE 'wsrep_flow_control_paused';

+--------------------------+----------+
| Variable_name            | Value    |
+--------------------------+----------+
| wsrep_flow_control_paused | 0.184353 |
+--------------------------+----------+
```

Cuando el nodo devuelve un valor de 0.0, indica que el nodo no se detuvo debido al Control de flujo durante este período. Cuando el nodo devuelve un valor de 1.0, indica que el nodo pasó todo el período en pausa. Si el tiempo entre FLUSH STATUS y SHOW STATUS fue de un minuto y el nodo devolvió 0.25, indica que el nodo se detuvo por un total de 15 segundos durante ese período de

tiempo.

Idealmente, el valor de retorno debería permanecer lo más cerca posible de 0.0, ya que esto significa que el nodo no se está quedando atrás del clúster. En caso de que encuentre que el nodo se detiene con frecuencia, puede ajustar el parámetro wsrep_slave_threads o puede excluir el nodo del clúster.

wsrep_cert_deps_distance

wsrep_cert_deps_distance muestra la distancia promedio entre el número de secuencia más bajo y el más alto, o seqno, valores que el nodo posiblemente puede aplicar en paralelo.

```
SHOW STATUS LIKE 'wsrep_cert_deps_distance';

+--------------------------+---------+
| Variable_name            | Value   |
+--------------------------+---------+
| wsrep_cert_deps_distance | 23.8889 |
+--------------------------+---------+
```

Esto representa el grado potencial del nodo para la paralelización. En otras palabras, el valor óptimo que puede usar con el parámetro wsrep_slave_threads, dado que no hay razón para asignar más hilos esclavos que las transacciones que puede aplicar en paralelo.

Detectando problemas de lentitud de red

Si bien verificar el estado de Flow Control y la cola recibida puede decirle cómo el servidor de la base de datos hace frente a los conjuntos de escritura entrantes, puede verificar la cola de envío para monitorear los problemas de conectividad saliente.

Nota A diferencia de otras variables de estado, estas son diferenciales y se restablecen en cada comando FLUSH STATUS.

wsrep_local_send_queue_avg

wsrep_local_send_queue_avg muestra un promedio para la longitud de la cola de envío desde la última consulta FLUSH STATUS.

```
SHOW STATUS LIKE 'wsrep_local_send_queue_avg';

+----------------------------+----------+
| Variable_name              | Value    |
```

```
+----------------------------+----------+
| wsrep_local_send_queue_avg | 0.145000 |
+----------------------------+----------+
```

Los valores mucho mayores que 0.0 indican problemas de velocidad de replicación o rendimiento de la red, como un cuello de botella en el enlace de la red. El problema puede ocurrir en cualquier capa desde los componentes físicos de su servidor hasta la configuración del sistema operativo.

Nota Además de esta variable de estado, también puede usar wsrep_local_send_queue_max y wsrep_local_send_queue_min para ver los tamaños máximos y mínimos que el nodo registró para la cola de envío local.

Mariadb.service: Failed to set up mount namespacing: Permission denied

Después de actualizar el sistema de un contenedor Proxmox a Debian **Buster o algún cambio en Proxmox**, el servicio **mariadb** no arranca y muestra el siguiente error:

```
mariadb.service: Failed to set up mount namespacing: Permission denied
```

Una posible solución es crear un fichero de configuración personalizado para el servicio **mariadb**:

```
systemctl edit mariadb
```

Al ejecutar el comando anterior, se creará un fichero /etc/systemd/system/mariadb.service.d/override.conf en el que introduciremos las siguientes líneas:

```
# /lib/systemd/system/mariadb.service
[Service]
ProtectHome=false
ProtectSystem=false
PrivateDevices=false
```

Una vez añadida la configuración ejecutamos el comando:

```
systemctl daemon-reload
```

Y, por último, arrancamos el servicio

```
systemctl start mariadb
```

Solucionar Problemas Galera MariaDB

Comprobaciones del quórum

Aunque es poco probable, es posible que sus nodos ya no se consideren parte del componente primario. Podría haber habido un fallo en la red, quizás más de la mitad del clúster falló, Ha habido un apagado de todos los servidores del cluster, o hay una situación de split-brain. En estos casos, el nodo llega a sospechar que hay otro Componente Primario, al que ya no están conectados.

Esta pérdida de integridad puede ser un problema. Cuando ocurre, los nodos comenzarán a devolver un error de comando Desconocido a todas las consultas que se les dé que ejecutar: simplemente dejan de realizar sus tareas por temor a empeorar la situación al estar demasiado sincronizados con su verdadero clúster .

Comprobación del estado del Cluster

Puede ver si esto sucede ejecutando la instrucción SHOW STATUS y verificando la variable de estado wsrep_cluster_status. Específicamente, esto se realiza ejecutando la siguiente instrucción SQL en cada nodo:

```
SHOW GLOBAL STATUS LIKE 'wsrep_cluster_status';

+----------------------+---------+
| Variable_name        | Value   |
+----------------------+---------+
| wsrep_cluster_status | Primary |
+----------------------+---------+
```

El valor de retorno Primario indica que el nodo en el que se ejecutó es parte del Componente primario. Cuando la consulta devuelve cualquier otro valor, indica que el nodo es parte de un componente no operativo. Si ninguno de los nodos devuelve el valor Primario, debe restablecer el quórum.

Las situaciones en las que ninguno de los nodos muestra que son parte del componente primario son muy raras. Si descubre uno o más nodos con un valor Primario, puede indicar un problema con la conectividad de red, en lugar de la necesidad de restablecer el quórum. Investigue la posibilidad

de conexión. Una vez que los nodos recuperan la conectividad de la red, se vuelven a sincronizar automáticamente con el componente primario.

Encontrar el nodo más avanzado

Antes de poder restablecer el quórum, debe identificar el nodo más avanzado del clúster. Es decir, debe encontrar el nodo cuya base de datos local confirmó la última transacción. Independientemente del método que utilice para restablecer el quórum, este nodo debe servir como punto de partida para el nuevo Componente primario.

Identificar el nodo más avanzado requiere que encuentre el nodo con el número de secuencia más alto (es decir, seqno). Puede determinar esto comprobando la variable de estado wsrep_last_committed. Desde el cliente de la base de datos en cada nodo, ejecute la siguiente consulta:

```
SHOW STATUS LIKE 'wsrep_last_committed';

+----------------------+--------+
| Variable_name        | Value  |
+----------------------+--------+
| wsrep_last_committed | 409745 |
+----------------------+--------+
```

El valor de retorno es el número de secuencia para la última transacción que el nodo confirmó. Si el demonio mysqld está inactivo, puede reiniciar mysqld sin iniciar Galera. Si no desea reiniciar las bases de datos, puede determinar el número de secuencia del archivo grastate.dat, ubicado en el directorio de datos.

Una vez que haya encontrado los números de secuencia de cada nodo, el que tenga el valor más alto es el más avanzado del clúster. Utilice ese nodo como punto de partida al iniciar el nuevo componente primario. Esto se explica en la siguiente sección aquí.

Recuperar Cluster Versiones Nuevas

La gente de MariaDB ha añadido un excelente binario que facilita la tarea y fácil de recordar: galera_new_cluster

```
root# galera_new_cluster

# Verificamos que el nodo está activo (Primary)
```

```
root# mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_status';"
Enter password:
+----------------------+---------+
| Variable_name        | Value   |
+----------------------+---------+
| wsrep_cluster_status | Primary |
+----------------------+---------+

# Verificamos el tamaño del clúster, solo 1 nodo
root# mysql -u root -p -e "SHOW GLOBAL STATUS LIKE 'wsrep_cluster_size';"
Enter password:
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 1     |
+--------------------+-------+
```

Restablecer el quórum

Cuando restablece el quórum, lo que está haciendo es arrancar el componente primario en el nodo más avanzado que tiene disponible. Este nodo funciona como el nuevo componente primario, alineando el resto del clúster con su estado.

Hay dos métodos disponibles para usted en este proceso: automático y manual. El recomendado para restablecer el quórum es el método automático. A diferencia del método manual, el arranque automático preserva la memoria caché del conjunto de escritura, o GCache, en cada nodo. Lo que esto significa es que cuando se inicia el nuevo componente primario, algunos o todos los nodos de unión se pueden aprovisionar rápidamente utilizando el método de transferencia de estado incremental (IST), en lugar del método más lento de transferencia de instantáneas de estado (SST).

Bootstrap automático

Al restablecer el quórum se iniciará el componente primario en el nodo más avanzado. Con el método automático, esto se hace habilitando dinámicamente pc.bootstrap a través de wsrep_provider_options a través del cliente de la base de datos; no se hace a través del archivo de configuración. Una vez que establezca esta opción, hará que el nodo sea un nuevo componente primario.

Para realizar un arranque automático, ejecute el siguiente comando utilizando el cliente mysql del nodo más avanzado:

```
SET GLOBAL wsrep_provider_options='pc.bootstrap=YES';
```

El nodo ahora funciona como el nodo inicial en un nuevo componente primario. Los nodos en componentes no operativos que tienen conectividad de red intentan iniciar transferencias de

estado incrementales si es posible, transferencias de instantáneas de estado si no, con este nodo, actualizando sus propias bases de datos.

En el caso de que no arranque deberemos revisar el fichero

```
/var/lib/mysql/grastate.dat
```

Revisar el valor safe_to_bootstrap que estará a 0 y ponerlo a 1

```
safe_to_bootstrap: 1
```

Matar los procesos de MariaDB o Mysql y reiniciar el cluster con el comando

```
galera_new_cluster
```

A lo mejor debemos de ejecutarlo un par de veces hasta que arranque correctamente.

Bootstrap manual

Al restablecer el quórum, se inicia el componente primario en el nodo más avanzado. Con el método manual, esto se realiza cerrando el clúster, apagando mysqld en todos los nodos, y luego iniciando mysqld con Galera en cada nodo, comenzando con el más avanzado.

Para iniciar manualmente un clúster, primero determine el nodo más avanzado ejecutando lo siguiente desde la línea de comandos en cada nodo:

```
mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_last_committed'"
```

Una vez que haya determinado qué nodo tiene el número de secuencia más alto, puede comenzar a cerrar el clúster. Simplemente apague mysqld en todos los nodos del clúster, dejando el nodo más avanzado hasta el final. Para los servidores que usan init, teclea lo siguiente desde la línea de comandos:

```
service mysql stop
```

Para los servidores que usan systemd, ejecuta esto desde la línea de comandos:

```
systemctl stop mysql
```

Ahora está listo para iniciar el clúster nuevamente. Inicie el nodo más avanzado con la opción --wsrep-new-cluster, no los otros nodos. Para los servidores que usan init, ejecuta el siguiente comando:

```
service mysql start --wsrep-new-cluster
```

Para los servidores que usan systemd y Galera Cluster 5.5 o 5.6, teclea en su lugar lo siguiente desde la línea de comandos:

```
systemctl start mysqld --wsrep-new-cluster
```

Para los servidores MySQL que usan systemd y al menos la versión 5.7 de Galera Cluster, puede ejecutar el siguiente script desde la línea de comandos solo en el primer nodo:

```
mysqld_bootstrap
```

Para los servidores MariaDB que usan systemd, puedes intentar ejecutar el siguiente script desde la línea de comandos, nuevamente, solo en el primer nodo:

```
galera_new_cluster
```

Con ese primer nodo ejecutándose y actuando como Componente primario, no está listo para iniciar todos los demás nodos en el clúster. Para los servidores que usan init, ejecuta el siguiente comando:

```
service mysql start
```

Para los servidores que usan systemd, teclea en su lugar lo siguiente desde la línea de comandos:

```
systemctl start mysqld
```

En todos estos scripts está escrita la opción --wsrep-new-cluster, pero se hace con cierta delicadeza. Independientemente del método o secuencia de comandos que utilice, cuando el primer nodo comienza con la opción --wsrep-new-cluster, inicializa un nuevo clúster utilizando los datos del estado más avanzado disponible del clúster anterior. A medida que los otros nodos comienzan, se conectan a este nodo y solicitan transferencias de instantáneas de estado para actualizar sus propias bases de datos. En poco tiempo, todos deberían sincronizarse y funcionar sin problemas.

Backups de MySQL/MariaDB con el comando mysqldump

Aunque existen diferentes métodos para realizar **copias de seguridad de bases de datos MySQL o MariaDB**, el más común y eficiente se basa en el uso de una herramienta nativa que tanto MySQL como MariaDB ponen a nuestra disposición para este cometido: **el comando mysqldump.** Este comando se incluye dentro de las utilidades de MySQL, por lo que, la tendremos disponible sin necesidad de instalarla.

Realizar backup con mysqldump

Para realizar un backup de nuestra base de datos, vamos a tener que indicar los detalles de la base de datos de laque queremos realizar el backup. Los detalles de la base de datos, son tres. Nombre de la base de datos, usuario de la base de datos, y contraseña de la base de datos. Si conocemos dichos datos, podremos obtener el backup de la base de datos de la siguiente forma:

```
mysqldump --user=USUARIO_BASE_DATOS --password=PASSWORD_BASE_DATOS NOMBRE_BASE_DATOS > nombredelacopiadeseguridad.sql
```

Hay que tener en cuenta, que introduciendo dicho comando, podemos dejar en el historial datos sensibles como la contraseña, por lo que es posible dejar el password vacío. Una vez presionemos enter, la herramienta nos solicitará el backup que podremos introducir, y una vez realizado, volvemos a pulsar enter, para que el backup se realice.

```
mysqldump -u USUARIO_BASE_DATOS -p NOMBRE_BASE_DATOS > nombredelacopiadeseguridad.sql
```

También lo podremos realizar desde root si lo necesitamos

```
mysqldump -u root -p NOMBRE_BASE_DATOS > nombredelacopiadeseguridad.sql
```

Backup de todas las bases de datos

En ocasiones, en lugar de querer realizar backup de una base de datos, es posible que queramos tener una copia de seguridad de todas las que tengamos en el servidor. Para ello, en lugar de utilizar el nombre de la base de datos, utilizaremos la opción «–all-databases». Ten en cuenta, que en este caso, por permisos, necesitarás utilizar el de un usuario que tenga permisos para volcar todas las bases de datos. Generalmente, esto ocurre con el usuario root (en paneles cPanel, por

ejemplo), o el usuario admin (en paneles Plesk). El comando, quedaría tal que así:

```
mysqldump --uadmin -p --all-databases > nombredelacopiadeseguridad.sql
```

Backup completo de una base de datos

Imaginemos que disponemos de un servidor de base de datos MySQL o MariaDB con varias bases de datos y que queremos hacer una copia de seguridad de todo el contenido de una de ellas. En este sentido mysqldump nos ofrece la posibilidad de exportar por separado la estructura, los datos, los triggers y los procedimientos o rutinas, o todo ello en su conjunto. Veamos en primer lugar lo que sería el ejemplo estrella, es decir, cómo realizar una **exportación completa de toda la información de una base de datos** que incluya estructura, datos, eventos, procedimientos, triggers y vistas:

```
mysqldump -v --opt --events --routines --triggers --default-character-set=utf8 -u
USUARIO_BASE_DATOS -p NOMBRE_BASE_DATOS > nombredelacopiadeseguridad.sql
```

Comprimir la copia

Si además queremos que se genere directamente un **fichero comprimido con gzip o bzip2** sin necesidad de que se escriba en disco primero el fichero SQL resultante de la ejecución del comando mysqldump, el cual tendrá un tamaño mucho mayor, para luego realizar la compresión del mismo en un segundo paso, ejecutaremos el comando añadiendo | *gzip -c* después del nombre de la base de datos a exportar.

```
mysqldump -v --opt --events --routines --triggers --default-character-set=utf8 -u
USUARIO_BASE_DATOS -p NOMBRE_BASE_DATOS  | gzip -c  > nombredelacopiadeseguridad.sql.gz
```

En el ejemplo simple

```
mysqldump -u USUARIO_BASE_DATOS -p NOMBRE_BASE_DATOS | gzip -c >
nombredelacopiadeseguridad.sql.gz
```

Exportar solo la estructura

```
mysqldump -v --opt --no-data --default-character-set=utf8 -u USUARIO_BASE_DATOS -p
NOMBRE_BASE_DATOS > copiadeseguridadestructura.sql
```

Sincronizar base de datos entre dos servidores

Podemos hacer backup y restore en un solo comando usando los ejemplos anteriores. Para sincronizar el contenido completo de una base de datos en otra sin crear ningún fichero intermedio

```
ssh ssh_username@server "mysqldump -v --opt --events --routines --triggers --default-character-set=utf8 -u USUARIO_BASE_DATOS --password=PASSWORD_BASE_DATOS NOMBRE_BASE_DATOS | gzip -c" | gunzip | mysql --password=PASSWORD_BASE_DATOS -u USUARIO_BASE_DATOS NOMBRE_BASE_DATOS
```

Crear un nuevo usuario en MariaDB

Para crear una nueva cuenta de usuario en MariaDB o MYSQL, en primer lugar accede al servidor mysql

```
#mysql -u root
```

Una vez en la consola de Mysql ejecutaremos

```
CREATE USER 'nuevo_usuario'@'localhost' IDENTIFIED BY 'contraseña';
```

Esto crea un usuario local, si el usuario debe de poder acceder desde otro equipo

```
CREATE USER 'nuevo_usuario'@'IP_DEL_EQUIPO' IDENTIFIED BY 'contraseña';
```

O si por el contrario puede acceder desde cualquier IP

```
CREATE USER 'nuevo_usuario'@'%' IDENTIFIED BY 'contraseña';
```

> **NOTA**: Aunque esta opción es la que deberemos de usar si se accede directamente desde otros equipos, es muy peligrosa, ya que carecemos de control sobre las conexiones a nuestra BBDD, hay que usarla con extrema precaución.

Para otorgar todos los privilegios de la base de datos para un usuario recién creado, ejecuta el siguiente comando:

```
GRANT ALL PRIVILEGES ON BASE_DE_DATOS. * TO 'nuevo_usuario'@'localhost';
```

Esto asigna todos los privilegios en la BBDD BASE_DE_DATOS

```
GRANT ALL PRIVILEGES ON *.* TO 'nuevo_usuario'@'localhost';
```

Esto proporciona permisos a todas las BBDD del sistema

Una vez que has finalizado los permisos que deseas configurar para los nuevos usuarios, asegúrate siempre de volver a cargar todos los privilegios.

```
FLUSH PRIVILEGES;
```

Permisos de Usuario en MySQL/MariaDB

Cómo otorgar diferentes permisos de usuario

Aquí se incluye una breve lista de otros posibles permisos comunes que los usuarios pueden utilizar.

- ALL PRIVILEGES: Esto le otorgaría a un usuario de MySQL acceso completo a una base de datos designada (o si no se selecciona ninguna base de datos, acceso global a todo el sistema).
- CREATE: Permite crear nuevas tablas o bases de datos.
- DROP: Permite eliminar tablas o bases de datos.
- DELETE: Permite eliminar filas de las tablas.
- INSERT: Permite insertar filas en las tablas.
- SELECT: Les permite usar el comando `SELECT` para leer las bases de datos.
- UPDATE: Permite actualizar las filas de las tablas.
- GRANT OPTION: Permite otorgar o eliminar privilegios de otros usuarios.

Asignar permisos

Para proporcionar un permiso a un usuario específico

```
GRANT type_of_permission ON database_name.table_name TO 'username'@'localhost';
```

Eliminar permisos

```
REVOKE type_of_permission ON database_name.table_name FROM 'username'@'localhost';
```

Visualizar los permisos de un usuario

```
SHOW GRANTS FOR 'username'@'localhost';
```

Restaurar Base de datos Mysql/MariaDB

En otro artículo vimos como hacer backups de Mysql/MariaDB

Ahora vamos a ver como recuperar esos backups.

Supongamos que tenemos una BBDD llamada basededatos. El comando para restaurarla sería:

```
mysql -u usuario -p basededatos < backupbasededatos.sql
```