Res. No. 16740, 2017-2021. Vigilada MinEducación.

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE

Proyecto final de Balanceo de Carga de bases de datos con MySQL y ProxySQL

Karim Andrés Perea Villany-COD:2167415

Diego Fernando Pacheco Esquivel-COD:2215739

Jhon Brandon Idarraga Cardenas-COD:2215737

Oscar David Quiñones Franco-COD:2215738

Ingeniería Electrónica y Telecomunicaciones, Facultad de Ingeniería, Universidad Autónoma de Occidente, Cali Colombia.

***Abstract*:** *The objective of this report is to analyze and understand database load balancing with MySQL and SQL Proxy which helps to solve the problems like overloaded servers, slow response times, server failures and limited scalability. It provides a more efficient, reliable, and scalable database environment, which improves application performance and the user experience.*

***Resumen*:** *El objetivo de este informe es analizar y comprender el balanceo de carga de bases de datos con MySQL y Proxy SQL que permite solucionar los problemas como la sobrecarga de servidores, los tiempos de respuesta lentos, los fallos de servidor y la escalabilidad limitada. Proporciona un entorno de base de datos más eficiente, confiable y escalable, lo que mejora el rendimiento de las aplicaciones y la experiencia del usuario.*

Palabras claves: Proxy Sql ; Balanceador de cargas; Mysql; base de datos; telecomunicación.

I. Introducción

El balanceo de cargas es una técnica esencial en la administración de bases de datos para distribuir de manera equitativa las consultas y la carga de trabajo entre varios servidores de bases de datos. Proxy SQL, un proxy de base de datos de código abierto, ofrece diferentes modos de configuración que permiten implementar estrategias de balanceo de cargas eficientes. En este informe, exploraremos en detalle los diferentes modos de configuración de Proxy SQL relacionados con el balanceo de cargas.

II. Marco Teórico

- ¿Qué es el balanceo de carga?

[1]El balanceo de carga es un mecanismo que permite repartir el consumo de recursos de cpu, red, almacenamiento para poder atender un servicio determinado sin las limitaciones que ofrecería un único dispositivo. Un ejemplo típico, es el tráfico web que recibe una determinada URL, y que puede llegar a saturar los recursos disponibles en una sola máquina. Si en lugar de ofrecer ese servicio web desde una sola máquina, lo hacemos repartiendo el tráfico y la carga entre varios servidores, seguro que la capacidad de ese servicio puede crecer hasta límites mucho mayores. En eso consiste el balancear la carga, repartir el tráfico o el procesamiento entre varios servidores para poder atender más peticiones y reducir los tiempos de respuesta.

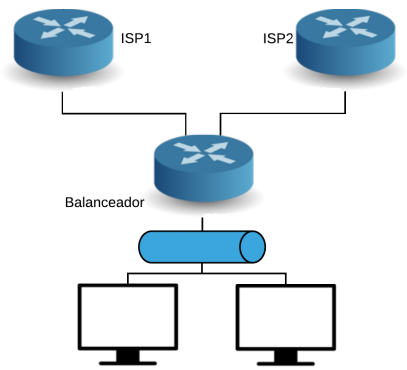

Imagen 1.Balanceo de la carga..

Fuente: https://soporte.syscom.mx/es/articles/1439490-mikrotik-balanceo-de-cargas

- Tipos de balanceo de carga:

[1]Se clasifican en distintos tipos de balanceos de carga en base a su función determinada:

- Balanceo de tráfico web: Es uno de los casos más habituales, donde se recibe una determinada cantidad de tráfico (por ejemplo tráfico web), y que se reparta entre varios servidores web (basado en Apache o Nginx por ejemplo) para poder atender muchas más visitas de las que podría soportar un único servidor, aunque pudiéramos aumentar los recursos al máximo.
- Balanceo de bases de datos: a nivel de bases de datos, también hay soluciones que permiten balancear las peticiones entre varios servidores diferentes. En MySQL, hay varias soluciones dependiendo de si se necesita un entorno multi-master o con varios nodos de solo lectura, por ejemplo Galera Clúster o MySQL Clúster son varias de las opciones más recomendables. En SQL Server se puede hacer uso de AlwaysOn para repartir peticiones de lectura desde las versiones 2014/2016. En Oracle database también se puede conseguir un balanceo de la carga utilizando Oracle RAC.
- Balancear comunicaciones: hay situaciones donde tenemos por ejemplo, varias líneas de comunicaciones que nos permiten tener conectividad hacia un determinado destino, que puede ser Internet o puede ser conectividad por ejemplo hacia otra sede o hacia una plataforma de Cloud (GCP, Azure, …).

Mysql:

[2]MySQL es un sistema de administración de bases de datos relacionales. Es un software de código abierto desarrollado por Oracle. Se considera como la base de datos de código abierto más utilizada en el mundo.

Soporta una amplia gama de tipos de datos, lo que permite tener una gran versatilidad en cuanto a las situaciones, industrias o casos de uso donde puede implementarse una base de datos MySQL. Puede emplearse para la industria financiera, al manejar datos con mucha precisión; por otro lado, también puede utilizarse en ámbitos de geolocalización por sus datos de tipo espacial. De igual forma puede competir, en ciertas situaciones, con las bases de datos no relacionales con su tipo de dato JSON.

Las características y ventajas de MySQL son muchas, pero sin duda todas ellas son mejor explotadas cuando están integradas dentro de un sistema de información. Para ello existe un amplio abanico de API nativas, librerías, paquetes, etc. que permiten integrar una base de datos MySQL en un sistema desarrollado en cualquier lenguaje de programación.

Imagen 2. Logo de Mysql.

Fuente: https://openwebinars.net/blog/que-es-mysql/

Imagen 3 . Logo de ProxySql.

Fuente: https://proxysql.com

- ProxySql:

[3]mysql-proxy es una ligera aplicación binaria entre uno o más clientes de MySQL y un servidor. Los clientes se conectan al proxy en lugar de conectar con el servidor. El proxy funciona como un intermediario entre el cliente y el servidor.

En su forma básica, el proxy es sólo una redirección. Se pone un cubo vacío desde el cliente (una consulta), lo lleva al servidor, llena el cubo con los datos, y se lo pasa al cliente.

Si eso fuese todo, el proxy como tal sería una herramienta inútil. Pero, dicha herramienta provee consigo una utilidad importante, el intérprete LUA. Usando el lenguaje Lua, puede definir qué hacer con una consulta o un conjunto de resultados antes de retornar la respuesta consultada.

- Optimización de consultas:

[4]Proxy SQL es un servidor proxy con reconocimiento de SQL que puede posicionarse entre su aplicación y su base de datos. Ofrece muchas funciones, como el equilibrio de carga entre varios servidores MySQL y además sirve como capa de almacenamiento en caché para las consultas. Este tutorial se centrará en la función de almacenamiento en caché de Proxy SQL, y en la forma en que puede optimizar las consultas de su base de datos MySQL.

[4]El almacenamiento en caché de MySQL se produce cuando el resultado de una consulta se almacena de modo que, cuando se repite la consulta, el resultado pueda mostrarse sin necesidad de realizar búsquedas en la base de datos. Esto puede aumentar considerablemente la velocidad de las consultas comunes. Sin embargo, en muchos métodos de almacenamiento en caché, los desarrolladores deben modificar el código de su aplicación, lo que podría introducir un error en la base de código. Para evitar esta práctica propensa a errores, Proxy SQL le permite

configurar un método de almacenamiento en caché transparente.

III. Análisis

Requerimiento:

En los requerimientos se permite implementar un balanceador de bases de datos MySQL con el objetivo de alcanzar una mayor eficiencia y velocidad en lo referente a escritura y lectura de datos, incrementar la tolerancia a fallos y obtener una alta disponibilidad. Para esto se debe configurar un pool de servidores de bases de datos.

Imagen 4. Balanceo de carga de base de datos.
Fuente:
https://blogofsysadmins.com/replicacion-y-ditribucion-de-carga-de-mysql

Paso 1 Descargar y descomprimir el archivo BalanceoMySQL-master.zip del repositorio

Imagen 5. Vagrantfile y los scripts para el aprovisionamiento.
Fuente: Autoría propia

El archivo vagrantfile es usado para la configuración de las máquinas como los 2 scripts para hacer el aprovisionamiento para que pueda instalar las aplicaciones que se requieren de forma automática.

```ruby
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do |config|

  if Vagrant.has_plugin? "vagrant-vbguest"
    config.vbguest.no_install  = true
    config.vbguest.auto_update = false
    config.vbguest.no_remote   = true
  end

  config.vm.define :servidorRest do |servidorRest|
    servidorRest.vm.box = "generic/centos8"
    servidorRest.vm.network :private_network, ip: "192.168.60.3"
    servidorRest.vm.provision "shell", path: "script.sh"
    servidorRest.vm.hostname = "servidorRest"
  end

  config.vm.define :cliente1 do |cliente1|
    cliente1.vm.box = "generic/centos8"
    cliente1.vm.network :private_network, ip: "192.168.60.4"
    cliente1.vm.provision "shell", path: "scriptcl.sh"
    cliente1.vm.hostname = "cliente1"
  end

  config.vm.define :cliente2 do |cliente2|
    cliente2.vm.box = "generic/centos8"
    cliente2.vm.network :private_network, ip: "192.168.60.5"
    cliente2.vm.provision "shell", path: "scriptcl.sh"
    cliente2.vm.hostname = "cliente2"
  end

end
```

Imagen 6. configuración de las máquinas del vagrantfile.
Fuente: Autoría propia

El archivo script.sh es usado para aprovisionar la maquina llamada servidor Rest la cual para este proyecto es el maestro MySQL y ProxySQL.

```bash
#!/bin/bash

#Update
sudo yum update

#Install MySQL
sudo yum install wget -y
wget http://repo.mysql.com/mysql-community-release-el7-5.noarch.rpm
wget https://github.com/sysown/proxysql/releases/download/v2.5.2/proxysql-2.5.2-1-centos8-clang.x86_64.rpm
sudo rpm -ivh mysql-community-release-el7-5.noarch.rpm
sudo yum install mysql-server -y
sudo yum install mysql-devel -y
sudo yum install gcc -y
sudo yum install proxysql -y
sudo systemctl start mysqld
sudo systemctl start proxysql
```

Imagen 7. El primer script del aprovisionamiento.
Fuente: Autoría propia

El archivo scriptcl.sh es usado para aprovisionar las maquinas llamadas cliente1 y cliente2 las cuales para este proyecto son los esclavos MySQL.

```bash
#!/bin/bash

#Update
sudo yum update

#Install MySQL
sudo yum install wget -y
wget http://repo.mysql.com/mysql-community-release-el7-5.noarch.rpm
sudo rpm -ivh mysql-community-release-el7-5.noarch.rpm
sudo yum install mysql-server -y
sudo yum install mysql-devel -y
sudo yum install gcc -y
sudo systemctl start mysqld
```

Imagen 10. El segundo scriptcl del aprovisionamiento.
Fuente: Autoría propia

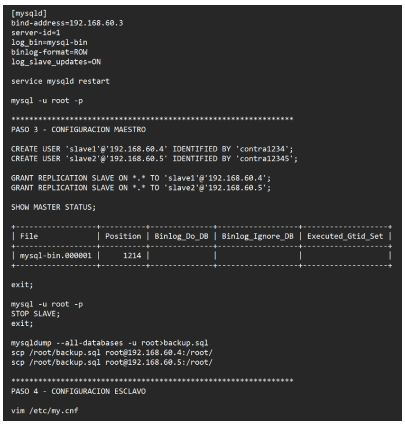

Imagen 11. Configuración del maestro.
Fuente: Autoría propia

CREATE USER 'slave1'@'192.168.60.4' IDENTIFIED BY 'contra1234'; CREATE USER 'slave2'@'192.168.60.5' IDENTIFIED BY 'contra12345';

Estos comandos crean dos usuarios en el servidor maestro: 'slave1' con dirección IP '192.168.60.4' y contraseña 'contra1234', y 'slave2' con dirección IP '192.168.60.5' y contraseña 'contra12345'. Estos usuarios serán utilizados por los servidores esclavos para establecer la conexión de replicación.

GRANT REPLICATION SLAVE ON . TO 'slave1'@'192.168.60.4'; GRANT REPLICATION SLAVE ON . TO 'slave2'@'192.168.60.5';

Estos comandos otorgan permisos de replicación a los usuarios 'slave1' y 'slave2' en el servidor maestro. Esto permite que los servidores esclavos se conecten al servidor maestro y reciban los datos replicados.

SHOW MASTER STATUS;

Este comando muestra información sobre el estado del registro binario en el servidor maestro. Proporciona detalles como el nombre del archivo binario actual y la posición del registro binario.

exit;

Este comando indica que se debe salir de la interfaz de línea de comandos de MySQL.

mysql -u root -p STOP SLAVE; exit;

Estos comandos inician una nueva sesión de MySQL y detienen la replicación en los servidores esclavos. Esto es necesario para evitar conflictos durante la configuración inicial de la replicación

- **Problemas del proyecto:**

Al configurar las máquinas, pueden surgir algunos problemas comunes que se deben tener en cuenta. A continuación, se mencionan algunos de los problemas más frecuentes y posibles soluciones:

1. Incompatibilidad de versiones: Puede haber problemas de compatibilidad si se utiliza una versión antigua de Vagrant con CentOS 8. Asegúrate de utilizar la versión más reciente de Vagrant compatible con CentOS 8 y verifica la documentación oficial de Vagrant para obtener información específica sobre la compatibilidad.
2. Problemas de red: Pueden surgir problemas de conectividad de red en la configuración de Vagrant con CentOS 8. Asegúrate de que la configuración de red en el Vagrantfile sea correcta y que los puertos necesarios están redirigidos correctamente. Además, verifica la configuración de red en la máquina anfitriona y asegúrate de que no haya conflictos de direcciones IP.
3. Problemas de aprovisionamiento: Si experimentas problemas durante el aprovisionamiento de la máquina virtual CentOS 8, verifica que los scripts de aprovisionamiento estén correctamente escritos y no contengan errores. Asegúrate de que los comandos de instalación de software y configuración de servicios sean compatibles con CentOS 8 y que no haya dependencias faltantes.

4. Problemas de configuración de la máquina virtual: Algunos problemas pueden estar relacionados con la configuración de la máquina virtual en sí. Asegúrate de que la cantidad de memoria y CPU asignada en el Vagrantfile sea adecuada para tus necesidades. Además, verifica la configuración de almacenamiento y asegúrate de que haya suficiente espacio en disco disponible.
5. Problemas de sincronización de carpetas: Si encuentras problemas al sincronizar carpetas entre la máquina anfitriona y la máquina virtual CentOS 8, verifica que la configuración de sincronización de carpetas en el Vagrantfile sea correcta. Puedes probar diferentes métodos de sincronización, como el método predeterminado o NFS, y ver cuál funciona mejor para tu caso específico.
6. Problemas de rendimiento: Si experimentas problemas de rendimiento en la configuración de Vagrant con CentOS 8, verifica que la máquina anfitriona cumpla con los requisitos mínimos de hardware y que no haya procesos o servicios en segundo plano que consumen recursos excesivos. Además, considera ajustar la configuración de recursos en el Vagrantfile para asignar más memoria o CPU según sea necesario.

Alternativas al balanceo de carga

MariaDB: está desarrollando activamente MaxScale, un servidor proxy de código abierto para la familia de bases de datos MySQL.

Está basada en un núcleo ligero que acelera su rendimiento. Tiene una arquitectura modular que acentúa su fexibilidad y facilidad de configuración.

MaxScale se configura entre una aplicación cliente y un cluster de bases de datos, y la versión actualmente disponible ofrece balanceo de carga basado en conexión y en sentencias SQL. El plan de desarrollo tiene proyectadas otras numerosas funcionalidades.

Percona XtraDB Clúster: es la solución de alta disponibilidad y balanceo de carga multi-máster del fork de MySQL «Percona». La solución recomendada para un cluster necesita un mínimo de 3 nodos, pero siempre un número impar de nodos.

- Conclusiones:

En conclusión, al configurar Vagrant con CentOS 8, es importante abordar los desafíos comunes relacionados con la compatibilidad, la configuración de red, el aprovisionamiento, la configuración de la máquina virtual, la sincronización de carpetas y el rendimiento. Siguiendo las soluciones propuestas, como utilizar la versión correcta de Vagrant, configurar adecuadamente la red y los puertos, verificar y corregir los scripts de aprovisionamiento, ajustar la configuración de la máquina virtual, configurar correctamente la sincronización de carpetas y optimizar el rendimiento mediante la asignación de recursos adecuada, se puede lograr una configuración exitosa de Vagrant con CentOS 8. Estas medidas permitirán superar los obstáculos y asegurar un entorno de desarrollo y pruebas eficiente y estable.

Referencias

- [1]Byte. (2019, April 17). *Balanceo de carga*. Redesteleco.com; RedesTeleco. https://redesteleco.com/balanceo_de_carga/
- [2]*MySQL Proxy - wiki de elhacker.net*. (n.d.). Elhacker.net. Retrieved May 13, 2023, from https://wiki.elhacker.net/bases-de-datos/mysql/introduccion/mysql-proxy
- [3]*MySQL Proxy - wiki de elhacker.net*. (s/f). Elhacker.net. Recuperado el 13 de mayo de 2023, de

https://wiki.elhacker.net/bases-de-datos/mysql/introduccion/mysql-proxy

- [4]Ibrahim, M. (2020, febrero 20). *Cómo optimizar las consultas de MySql con almacenamiento en caché de Proxy SQL en Ubuntu 16.04*. Digitalocean.com; DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-optimize-mysql-queries-with-proxysql-caching-on-ubuntu-16-04-es