LOG8415E
Scaling Databases and Implementing Cloud Design Patterns

Yiwei Zhou

Polytechnique Montreal

December 28, 2023

Github: https://github.com/Presidentkidz/LOG8415E

Video: https://screenrec.com/share/cpfyqugJxs

1 Benchmarking MySQL stand-alone vs. MySQL Cluster

I could not benchmark the results because my AWS access had been revoked just like the rest of the people in my class. I have tried to follow the steps outlined in Slack, but still could not get it working. Also, I was running out of time since I am leaving on vacation on the 29th and would not have internet access. I hope you can understand my situation.

2 Implementation of The Proxy pattern

The only requests received by the Proxy are sent by the Trusted Host. All other requests are blocked. If the request received from the Trusted Host is a SQL write request, then the direct hit method is used. If the request received from the Trusted Host is a SQL read request, then depending on the "mode" attribute of the Json request, the correct mode is used (direct hit, random or customized). Depending on what mode is used, the Proxy forwards the request to the correct node (Manager or Workers) and awaits the response.

3 Implementation of The Gatekeeper pattern

The client may only send requests to the Gatekeeper. Once the Gatekeeper receives the request, it validates the inputs so as to prevent an SQL injection attack. It everything is alright, then it forwards the request to the Trusted Host and awaits a response. Then, the Trusted Host routes the request to the Proxy.

4 How the implementation works

First off, ec2.py is ran and it creates all the necessary ec2 instances for the lab. At the same time, auth.py creates all the necessary security groups and modifies all the permissions to only allow access to and from certain instances. Then, the correct security group is assigned to each instance.

After, standalone_sql.py deploys the standalone SQL instance trhough an SSH connection using fabric and installs the Sakila database.

Then, mysql_cluster.py deploys the MySQL Cluster and installs the Sakila database. It also deploys the flask apps on the manager and workers. After, tt transfers the necessary .py files into the manager and worker instances.

Next up, gatekeeper_setup.py sets up the Gatekeeper and installs the flask app onto it. It also transfers the necessary gatekeeper.py file into the gatekeeper instance. trusted_host_setup also sets up the Trusted Host and installs the flask app onto it. It also transfers the necessary trusted_host.py file into the Trusted Host instance.

Finally, proxy_setup sets up the Proxy and installs the flask app onto it. It also transfers the necessary proxy.py file into the proxy instance.

In order to test the whole setup, the requests.py file sends write requests to the Gatekeeper and await for it's response. Then it sends read requests with all 3 different modes (direct hit, random or customized) and awaits for a response.

In summary, the client sends a request to the Gatekeeper. The Gatekeeper validates and forwards the request to the Trusted Host. Then, the Trusted Host forwards the request to the Proxy. The Proxy, depending on what mode (direct hit, random or customized) is used, forwards the request to the right SQL instance. The SQL instance (manager or worker) then processes the query and returns a response to the Proxy. The Proxy then forwards the response to the Trusted Host. Then, the Trusted Host forwards the response to the Gatekeeper. Finally, the Gatekeeper forwards the response to the client.

5 Summary of results and instructions to run the code

To run the code, simply run main.sh.
NOTE: You will probably run into errors as I have only been able to test a part of my code before I lost my AWS access.