Automated SW Testing for MongoDB in Chameleon Cloud

Youngjae Moon, Lisa Liu, Robert Sheng

Contents

- Project Overview

- Features

- Tools and Technologies

- VM Distribution

- VM Roles

- Team Member Roles

- Future Work

- Conclusion

Project Overview

- This project aims to simplify the deployment and evaluation of MongoDB by:
  - Automating the setup of MongoDB clusters using Docker and Kubernetes.
  - Providing a suite of unit tests to validate basic functionality of MongoDB interactions.
  - Integrating fuzz testing, a type of automated software testing, through American Fuzzy Lop (AFL) to identify potential vulnerabilities.

Features

- Automated MongoDB Setup
  - Easily deploy a MongoDB cluster on Chameleon Cloud using Docker and Kubernetes.
- Unit Testing
  - Validate core MongoDB operations such as CRUD operations and replica set configurations.
- Fuzz Testing
  - Stress-test the MongoDB client or server with random or invalid inputs to identify crashes and vulnerabilities.
- Kubernetes StatefulSet
  - Deploy and manage MongoDB clusters on Kubernetes with persistent storage and replica sets.

Tools and Technologies

- Docker
  - For containerization of MongoDB.
- Kubernetes
  - For orchestrating MongoDB clusters.
- Python 3.12.6
  - For running benchmark and test scripts.
- MongoDB CLI Tools
  - To interact with MongoDB from the command line.
- American Fuzzy Lop (AFL)
  - For fuzz testing.

VM Distribution

- This project employes Docker and Kubernetes across a four-VM setup to automate the deployment, configuration, & testing of MongoDB clusters.

- Each VM is designated for specific roles to maximize efficiency, scalability, and reliability.

-

VM Roles

- VM1: Control and Orchestration Node
  - Master node in Kubernetes.
  - Deploys all pods.
- VM3: Primary MongoDB Replica Set
  - Runs Kubernetes pod that hosts the primary MongoDB replica set.
- VM4: Secondary MongoDB Replica Set + MongoDB Client
  - Runs Kubernetes pod that hosts the secondary MongoDB replica set.
  - Runs Kubernetes pod that runs a Python MongoDB client to interact with the MongoDB pods.

VM Roles

- This VM distribution maximizes the project's flexibility by leveraging Docker for containerized MongoDB deployments, and Kubernetes for high-availability and clustered management.

- This setup facilitates comprehensive testing of MongoDB across different environments, ensuring reliability, scalability, and performance.

Team Member Roles

- Robert Sheng
  - Writing codes in VM1, VM3 and VM4 for MongoDB setup through Docker and Kubernetes.
  - Generating various test metrics for evaluation.

- Lisa Liu
  - Writing codes in VM1, VM3 and VM4 for MongoDB setup through Docker and Kubernetes.
  - Generating various test metrics for evaluation.

- Youngjae Moon
  - Writing test codes in VM1
  - Making up slides and final report

Future Work

- CI/CD Integration
  - Automate deployment and testing pipeline using GitLab or GitHub Actions.

- Develop more sophisticated test metrics.
  - And achieve high coverage.

- Note: Demo will be done through recording a video.

Conclusion

- This project automates MongoDB deployment and testing, enhancing reliability and reducing setup time.

- By integrating unit and fuzz testing, it ensures MongoDB is robust under various conditions.

For more information…

https://github.com/Pingumaniac/Automated_Software_Testing_for_Cloud

Thank you!

Any questions?