Open Source Research Experience Program Summer 2021

Facilitate continuous benchmarking/regression testing for the critical components of SkyhookDM

Mentors: **Jeff LeFevre, Jayjeet Chakraborty**

Student: Rahul Agrawal

About Me

I am Rahul Agrawal, a final-year Undergraduate student from IIT Kharagpur, India. I have good experience in Software Development. Since my freshman year, I have been an active open-source contributor and have successfully completed my Google Summer of Code 2020 project.

My fields of interest are DevOps, Machine Learning & AI. I have worked on various DevOps technologies like CI/CD (Github-Actions/Jenkins), Docker, Kubernetes, Prometheus, Grafana, etc. This year I got selected for the Linux Foundation Mentorship program under the CNCF organization, where I am working on adding Monitoring support to Kubernetes storage engine **'OpenEBS'** using Grafana, Prometheus.

I have been an active member of the PEcAn project, where I mainly worked on DevOps-related tasks. I have made a code linting workflow using Github Actions and some linting packages.

Apart from open source projects, I have equal exposure to the corporate world. I have done internships at OYO, Skuad.io, and Accenture. During these internships, I was primarily working on testing (unit and integration) and DevOps domains. At Skuad.io, I was responsible for migrating the whole existing monolithic architecture to six different microservices and handling the deployments in three different environments (dev, staging, and prod).

To improve my problem-solving skills, I used to tackle lots of Leetcode problems, where I primarily used C++ language to solve all the problems. I have also uploaded all the solutions, which I usually maintain on my local system for reference.

Contact Information

Name :	Rahul Agrawal
Github Profile:	rahul799
University:	Indian Institute of Technology (IIT), Kharagpur
Email:	rahulagrawal799110@gmail.com
Phone :	(+91) 7991106736
Timezone :	Indian Standard Time (UTC +05:30)

Platform Details

OS :	**Ubuntu 20.04**
Editor :	**Visual Studio Code**
Version Control :	**Git**

Introduction

Overview

The Skyhook Data Management project adds data management capabilities to object storage for tabular data. SkyhookDM is a Ceph distributed object storage module that allows you to store and query database tables.

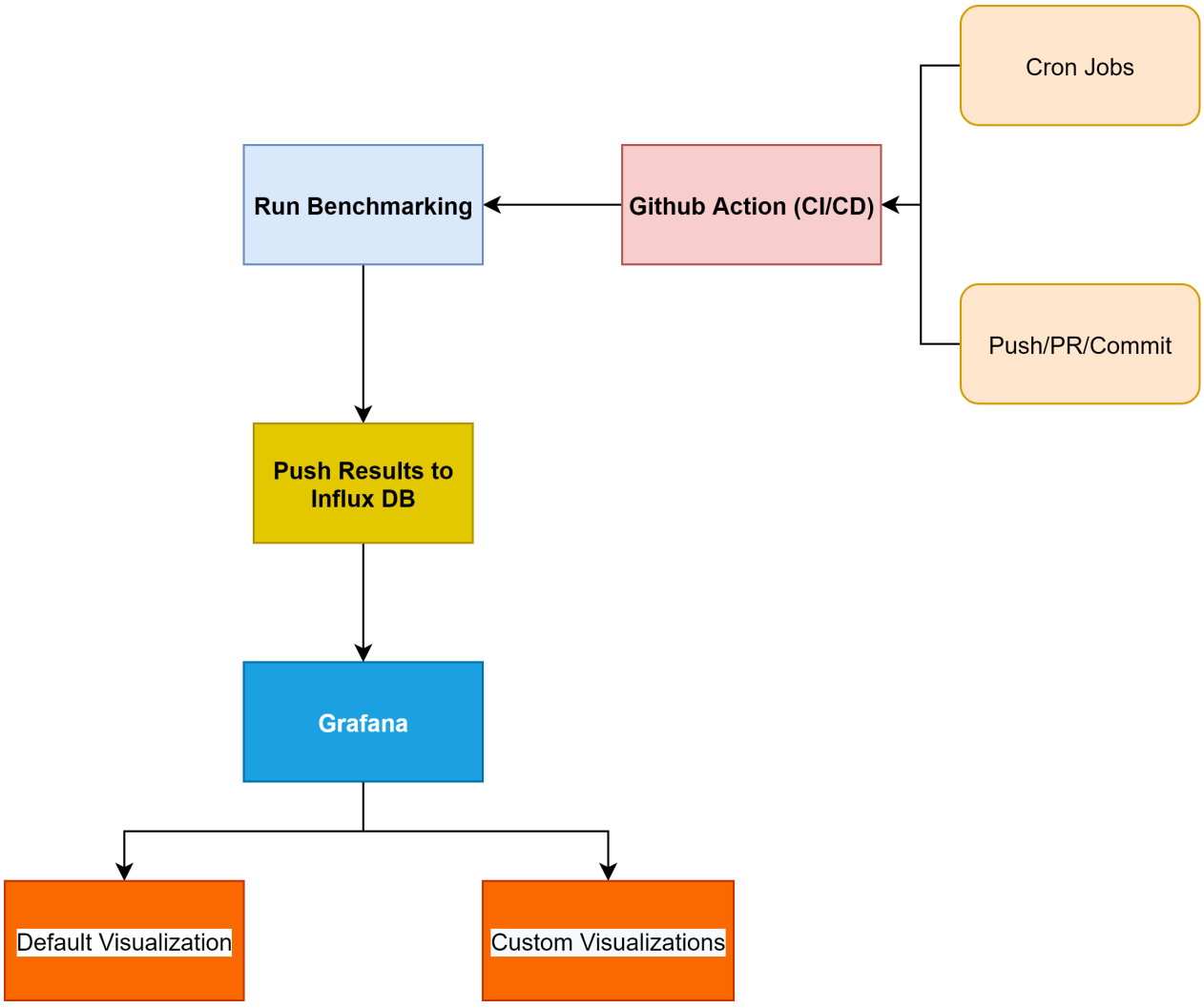
SkyhookDM, a performance-critical distributed storage system developed by embedding Apache Arrow, is a computational storage system. Small changes in the source code's performance-critical parts will often result in significant performance changes. It's essential to keep track of these performance changes so that the project can become more performant over time and avoid silent performance deterioration.

To overcome these challenges, the Google benchmark framework can be used to create benchmarks (very similar to unit tests) for all the performance-critical parts of the source code. These benchmarks can be added as a separate job in the CI/CD pipeline, which will get triggered when any particular events like commit/push happen. A web dashboard can also be integrated to monitor the performance results of the CI tests.

The Project

Approach & Implementation

The rough architecture of the system will look like the below diagram.



- **Adding Google Benchmarks in SkyhookDM**

Google benchmark is a C++ library. This library is lightweight and supports different types of benchmarks: both value- and type parameterized. Google benchmark also provides various options for running the benchmarks including multithreading, and custom report generation.

It is pretty straightforward to use, for example

```
#include <benchmark/benchmark.h>

static void BM_SomeFunction(benchmark::State& state) {
  // Perform setup here
  for (auto _ : state) {
    // This code gets timed
    SomeFunction();
  }
}
// Register the function as a benchmark
BENCHMARK(BM_SomeFunction);
// Run the benchmark
BENCHMARK_MAIN();
```

- **Integrating With CI/CD**

  All the benchmark tests will be integrated with the Continuous Integration tool so that the benchmarking will be completely automated. I will be using Github Actions to make the pipeline. The data/artifacts generated after the benchmarking job is finished will be pushed to influxDB.

  I have added some sample code below.

```
name: CI-Monitoring

# Controls when the action will run. Triggers the workflow on push or pull
request
# events but only for the master branch
on:
```

```
  push:
    branches: [ master ]
Jobs:
  Run_benchmark:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2
    - name: Run benchmarking
      run: sh script.sh  # Runs a command or script to execute the tests
  push_to_db:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2
      with:
        ref: ${{ github.ref }}
    # Runs a set of commands using the runners shell
    - name: Deploys repo to cloud
      env:
        INFLUX_TOKEN: ${{ secrets.INFLUX_TOKEN }}
        INFLUX_ORG: ${{ secrets.INFLUX_ORG }}
        INFLUX_URL: ${{ secrets.INFLUX_URL }}
        GITHUB_REPO: ${{ github.repository }}
      run: Some CLI commands
```

- **Integrating Grafana Dashboard for monitoring**

  The time-series data present in the influxDB will be utilized by the Grafana Dashboard to get excellent graphical visualization of the performance results.

  Grafana Dashboards have got really good integrations with the influxDB and configuration is very simple.

Background

- **Github Actions :-** It is a Continuous Integration as well as continuous delivery service which helps to build, test, and deploy applications. It can also be used to automate other

tasks common to the developer workflows like taging and managing issues, automating releases, collaborating with your user base, and more.

- **InfluxDB :-** It's an open-source time series database developed by InfluxData. It is written in Go and optimized for fast, high-availability storage and retrieval of time series data in fields such as operations monitoring, application metrics, Internet of Things sensor data, and real-time analytics
- **Grafana:-** it is a multi-platform open source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources

Communication

I will be in regular contact with mentors using email. In case I have a problem regarding any functions or external library, I'll try to reach them out using the Mailing list, etc.

If selected, I also plan to make a weekly blog describing the work I did during the entire week. I believe that the problems faced by me and their solutions would undoubtedly help other fellow developers later just like I have received great help from various blogs on the internet.

I'm comfortable with any form of communication that suits my mentor. Below are the various options available:

- Email: rahulagrawal799110@gmail.com , rahulagrawal9926@iitkgp.ac.in
- Phone (Call, Whatsapp & telegram): (+91) 7991106736
- Hangouts: rahulagrawal799110@gmail.com

Other Commitments

I'll be mostly working full-time on the code on weekdays. On weekends, I'll be focusing on clearing any delay in the schedule, otherwise utilizing it to communicate progress with my mentor. My awake hours would usually be in between 10 AM IST (4:30 AM UTC) to 2 AM IST the next day (8:30 PM UTC) and I'm comfortable working anytime during this period. Also, the **Linux foundation mentorship** which I mentioned above in the *About me* section will be **completed** before the project starts.

In case of me being unavailable during the mentorship period due to reasons such as traveling, or any other unavoidable circumstances, I will inform my mentors beforehand about it and make sure I make up for the lost time.

Timeline

This Timeline is tentative and lists the objectives I aim to complete within the timeframe specified. There could be situations in which the project could get delayed or reach early completion. To handle such cases, I have made sure to have buffer weeks to complete the incomplete tasks. I am very enthusiastic about working with SkyhookDM Community and would be contributing by taking up issues outside of my OSRE project as well.

- **Community Bonding Period (May 17 to June 14):**
    - During this period, I will discuss implementation ideas with my mentors, and form a solid roadmap for the project.
    - I will familiarize myself with the codebase.
    - This would be the time I'd utilize properly and learn the concepts which I ain't so familiar with so that I don't lag once the coding period begins. This would involve:
        - Researching more about the Google Benchmark framework
        - Researching more about the influxDB.
    - Discuss the workflow with the mentor and make mockups/ wireframes of the design going to be implemented.
    - Decide the goals and non-goals of the project and get a clear idea of the features to implement.

      This period is basically a learning period where I plan to break the ice and get a broader and yet clear picture of the project.

- **Week 1-6 (June 1 to July 25)**
    - I will start writting benchmarks for different functions present in the source code.
    - This will take a bit longer, and that's why I have allotted six weeks of time, and I am targeting to finish this by the end of the 6th week.

- **Week 7 (July 25 to Aug 1) [Buffer Week]**
    - This week I will complete all the backlogs, if any and will try to wrap up the benchmarking-related tasks.

- **Week 8-9 (Aug 2 to Aug 15)**
    - In this period, I will make the CI pipeline.

- **Week 10 (Aug 16 to 22)**
    - This week I will integrate the influxDB in the CI workflow and wrap up all the CI-related tasks.

- **Week 11 (Aug 23 to 29)**
    - Adding Grafana Dashboard will be quiet straightforward, and I will try to wrap up the integration this week

- **Code Submission and Final Evaluations - August 30 - Sept 10, 2021**
    - Complete all the backlogs if any
    - I will prepare for the final evaluations by writing a project report and making a presentation also.