

Reproducible Large Scale SkyhookDM Experiments

Project proposal for IRIS-HEP Fellowship programme

Jayjeet Chakraborty

3rd Yr, B.Tech Undergrad


National Institute of Technology, Durgapur

Overview

SkyhookDM injects programmable data management and data storage capabilities directly in the storage layer for distributed object databases such as Ceph. Skyhook allows users to dynamically grow or shrink their storage and processing needs as demands change. It can store tables either in row format using Google flatbuffers or in column format using Apache Arrow format. It is currently an incubator project at CROSS, UC Santa Cruz.

Project Idea / Problem to be Solved

Setting up a Ceph cluster and running Skyhook tests on them can be cumbersome. Compiling Ceph along with installing Skyhook plugins consists of a lot of steps and can be error prone at times. Through this project we aim to develop an end-to-end reproducible workflow to simplify and automate large-scale SkyhookDM tests with benchmarks in different cloud infrastructures like Cloudlab, GCP and Kubernetes on 10's of TB of data of different formats . We aim to use Popper as the workflow execution engine and different DevOps tools like Terraform to manage infrastructure as code and Ansible to orchestrate the provisioning and configuration of the SkyhookDM Ceph cluster. We also plan to use [Rook.io](https://rook.io)



for deploying Ceph in Kubernetes clusters and use the Kubernetes resource manager of Popper to automate the process. The Ceph kubernetes tests can be done in the SSL cluster in UChicago, the Nautilus Kubernetes Federation, or in Google Kubernetes Engine. This work will be done in collaboration with Carlos Maltzahn, Jeff LeFevre and Ivo Jimenez from UC Santa Cruz.

Expected Deliverables

- Benchmarks collected by running various SkyhookDM tests using 10s of TB's of data stored in different formats in both Kubernetes clusters and bare-metal nodes.
- Several Popper workflows that automate all the steps of the benchmark tests like spawning machine nodes/k8s clusters, compiling ceph, installing ceph on the clusters, compiling and injecting skyhook plugins, downloading datasets, running the tests using them and generating benchmarks.

Background

- **Popper 2.0:** It is a container native workflow execution engine for building reproducible workflows and executing them in different computing environments. It supports executing workflows in Local machines, in the Cloud using Kubernetes and also in HPC clusters.
- **Ceph:** Ceph is an open-source software storage platform, implements object storage on a single distributed computer cluster, and provides 3 in 1 interfaces for: object, block and file-level storage. Ceph allows decoupling data from physical hardware storage, using software abstraction layers, providing scaling and fault management capabilities. This makes Ceph ideal for cloud, Openstack, Kubernetes and other microservice and container-based workloads as it can effectively address large data volume storage needs.
- **Kubernetes:** It is an open-source system for automated deployment, scaling and management of containerized applications. It supports using different container runtimes like docker, rkt and podman and has great support from the community.
- **Docker:** It is an industry standard light-weight virtualization or container engine. It is built on modern Linux kernel features like CGroups and Namespaces. Docker makes

it easy to package and ship dependencies and applications in the form of a Linux FS and run them completely isolated from one another.

Timeline

- Week 1
 - Study and understand the project specifications and prepare a design document.
 - Gather knowledge on the related tools and technologies like Terraform, Ansible, SkyhookDM, the data formats to be used like Google Flatbuffers and Apache Arrow.
 - Chalk out the solution of deploying Ceph in Kubernetes cluster and running tests on it using Ansible.
- Week 2
 - Write steps to spawn and release a Kubernetes cluster in any cloud infrastructure using Terraform, Geni, etc.
- Week 3
 - Implement steps to compile and install Ceph along with the SkyhookDM plugins on the Kubernetes clusters using Rook.io.
 - Test the working of the SkyhookDM-Ceph setup.
- Week 4
 - Write steps to download the TB's of data and store them in the cluster or a shared File System depending upon the specifications. The SkyhookDM benchmark tests would be done using these data.
- Week 5
 - Implement the steps to run different tests in SkyhookDM for preparing and collecting benchmarks.
 - Also, implement steps for automatic collection of results of these tests and uploading them in the Cloud storages like S3.
- Week 6
 - Setup prometheus to monitor the tests in the kubernetes cluster in real time and collect different benchmarks depending on system resource usage data.
- Week 7, 8
 - Write steps for spawning and releasing bare-metal clusters in AWS, GCP, etc.
 - Implement the same above steps for running Skyhook tests and developing benchmarks in the bare-metal clusters.
- Week 9

- Prepare the dataset by running the workflow in different conditions in different infrastructures and store the produced data, also testing the reproducibility along the way.
- Make modifications and fix bugs as they come up.
- Week 10
 - Setup automated benchmarking of Skyhook tests in CI services like Travis/Circle to provide efficient and continuous testing and benchmarking.
- Week 11
 - Write documentation and tutorials on the project.
 - Write a report or blog on the work that was done.
- Week 12
 - This week is kept as a buffer period to fix any unforeseen bugs or issues that might come up and finish up the work done over the summer.

Future Work

In future, we would like to publish this work and present it in conferences and workshops.

Biographical Information and Previous Work

I am a Junior year student pursuing Bachelors of Technology in Computer Science from National Institute Of Technology, Durgapur, India. I have been working on the Popper 2.0 project since Google Summer Of Code 2019 and have added various new features since then including working on adding support for additional runtimes like Singularity and working on adding support for resource managers like SLURM. I have also worked on popperizing MLPerf workflows and built workflows to automate Kubernetes cluster creation in Cloudlab.

Contact Info

Name: Jayjeet Chakraborty

Institute Registration No. : 17U10298

Residential Address : 13/1/4 Ramanujam Road, B-Zone, Township, Durgapur - 713205, West Bengal, India

Email-ID : jayjeetchakraborty25@gmail.com

Github Profile : <https://github.com/JayjeetAtGithub>



Contact No. : +918436500886