University of Science and Technology of Hanoi

Cloud and Big Data

Project Report

Student ID: 2440057

Student name: Nguyen Nhat Anh

I. Architecture Overview

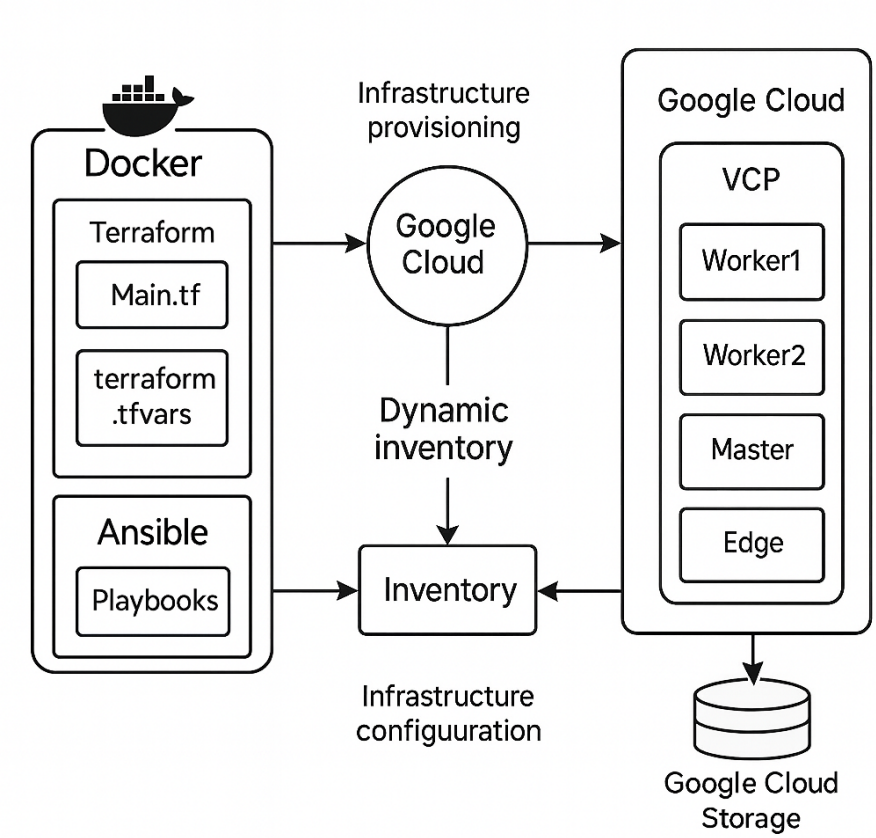

Figure 1: Overall architecture

1. Infrastructure Components

This project automates the deployment of an Apache Spark cluster on Google Cloud Platform (GCP) using Docker, Terraform, and Ansible. Terraform creates the main cloud resources such as the VPC network, firewall rules, and a virtual machine that runs Docker. Ansible installs Docker on this machine and starts all Spark containers.

The Spark cluster runs completely inside Docker:

- **Spark Master**: manages scheduling and coordinates the workers.
- **Spark Workers**: run tasks in parallel.
- **Edge Container**: sends jobs to the cluster and handles data.
- **Google Cloud Storage (GCS)**:
  - Stores datasets and the WordCount JAR file.
  - Provides safe and scalable storage for the cluster.
  - Acts as the main place where the Edge container reads and writes data.

2. Automation Workflow

The workflow works as follows:

1. **Terraform** creates the VPC, firewall rules, and the VM that will host Docker.
2. Terraform outputs are used to build a **dynamic inventory** for Ansible.
3. **Ansible** installs Docker and starts the Spark Master, Worker, and Edge containers.
4. The input file and the WordCount JAR are uploaded to **Google Cloud Storage**.
5. The **Edge container** downloads the data from GCS and sends the WordCount job to the Spark Master.

This automated setup makes the Spark deployment simple, repeatable, and easy to scale. The system is tested by running the WordCount program with different numbers of executors to measure performance.

II. Methodology

1. Terraform Configuration

The Terraform design prioritizes modularity and minimal infrastructure:

- One compute instance used as a **Docker host** for all Spark containers
- GCS bucket automatically created for:
  - storing datasets,
  - storing compiled JAR applications,
  - logs or result output
- Firewall rules restricted using `source_ranges = [var.admin_ip]`
- Service accounts created to provide GCS access for containers through key-less metadata authentication

2. Ansible Roles

Three roles automate the deployment:

- **common**: Installs Docker, configures networking, prepares directories
- **master**: Starts the Spark Master container with published ports
- **worker**: Starts Spark Worker containers and links them to the Master

The dynamic inventory reads Terraform outputs to determine the IP of the Docker host.

3. Security

Security is enhanced across the pipeline through:

- Firewall allowing only necessary ingress traffic
- IAM roles restricting access to GCS buckets
- SSH key authentication to the Docker host
- No exposed Docker APIs; container control is local only

III. Benchmark

At the moment, the full benchmark for the WordCount job cannot be completed because the Spark cluster is unable to read data from Google Cloud Storage (GCS). The issue is caused by restricted access to the original GCS bucket, which prevents the Edge container from downloading the input file.

During job submission, Spark reports a storage access error, confirming that the problem comes from GCS permissions rather than from Docker, Terraform, or Ansible configuration.

A new GCS bucket has already been created under an active billing project, and the dataset is being uploaded again. Once storage access is restored, the WordCount job will be executed to collect accurate performance results.

For reference, the expected benchmark table is shown below, illustrating how execution time typically scales when additional executor cores are used:

Although the final numbers are not available yet, the automated workflow has been validated up to the point of data access:

- Terraform successfully provisions the Docker host and network resources.
- Ansible consistently deploys and configures all Spark containers.
- The cluster starts correctly and is ready to receive jobs.
- The remaining task is restoring GCS access so that the WordCount job can run normally.