National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Hariharan Sathiyamoorthy
Student ID: 23201550

School of Computing
National College of Ireland

Supervisor: Ahmed Makki

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Hariharan Sathiyamoorthy
Student ID:	23201550
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Ahmed Makki
Submission Due Date:	12/12/2024
Project Title:	Configuration Manual
Word Count:	700
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Hariharan Sathiyamoorthy
Date:	11th December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Hariharan Sathiyamoorthy
23201550

1 Introduction

This Documnet offers comprehensive guidelines for configuring and executing the research experiments. It also provides the requisite software and platforms utilized in this study. This study evaluated the solution's effectiveness in improving cold start latency, by comparing it to Apache OpenWhisk *Apache OpenWhisk Documentation* (n.d.).The main tools, systems, and software libraries and packages used in this study are shown in Table 1. A quick look at the files and folders in SmartFasS can be found in Table 2. '

Component	**Details**
Virtual Machines	Google Cloud Platform (GCP) Compute Engine
Operating System	Ubuntu 22.04 LTS
Serverless Platform	OpenWhisk
Kubernetes Cluster	KinD (Kubernetes in Docker) for running OpenWhisk
Container Technology	Docker (27.3.1)
Cache Manager	Redis 6.0.16
Machine Learning	Google Colab & Keras TensorFlow
Programming Languages	Node js 20.18 & Bash
Performance/Load Testing	Java (openjdk-11) & Apache JMeter 5.6.3
Docker Image	node:20-alpine

Table 1: System Configuration & Prerequisite

File/Folder	**Description**
`colab/`	Folder containing Jupyter notebooks for analysis and modeling.
`dataset/`	Folder containing Azure dataset and ML results.
`function.js`	Lodash Node.js app for execution
`jmeter/`	Folder containing JMeter scripts for load testing.
`logs/`	Folder containing log files and results from the experiments.
`main.js`	Index file for the project.
`package.json`	Configuration file for Node.js project dependencies and scripts.
`scripts/`	Folder containing shell scripts for setup and utilities.
`utils/`	Folder containing utility JavaScript files for SmartFasS.

Table 2: SmartFasS files and folders

2 Configuration

This section will outline the whole installation procedure to set up a virtual machine in Google Cloud Platform (GCP) *Google Cloud Console - Compute Engine* (n.d.), along with the installation of SmartFasS and OpenWhisk on the created virtual machine.

2.1 Creating VM

Two GCP `e2-standard-2` instances will be used for the implementation, To use the framework without memory issues, This will ensure that both the framework and OpenWhisk will run smoothly on virtual machines.

Configure Your Project

1. **Log in to GCP Console:**.
2. **Select/Create a Project:**
   (a) Choose an existing project or create a new one by clicking `New Project` and following the prompts.

Instance Creation

1. Create an instance by navigating to Compute Engine, selecting the tire as seen in Figure 1, and selecting the `e2-standard-2` with the default configuration of 2 vCPUs and 8 GB of RAM.

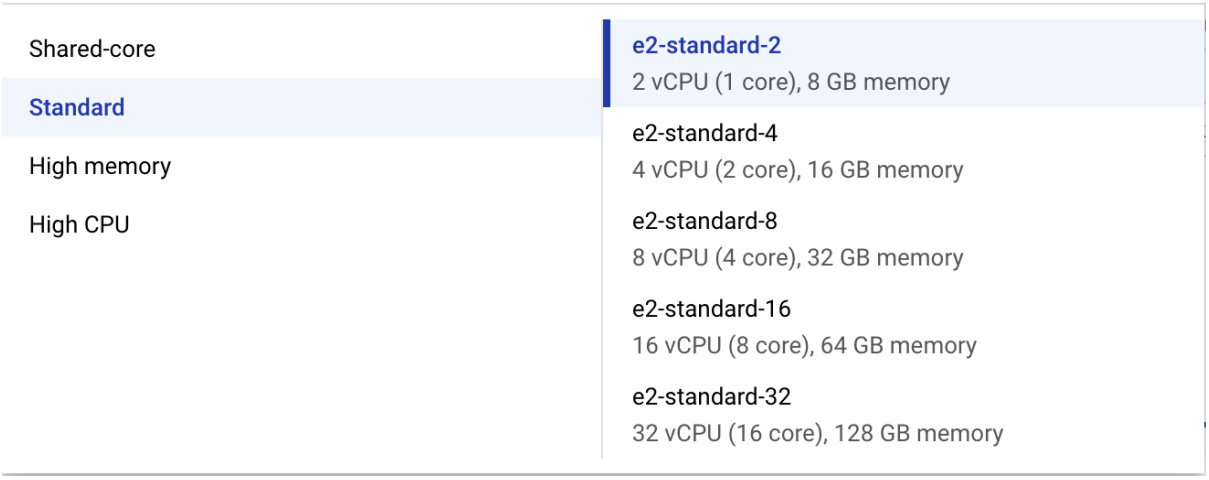


Figure 1: Tire Selection

2. Choose the Ubuntu as Operating system and choose the version 22.04 LTS with `x86/64 jammy image` and 64 GB storage Figure 2
3. Next, grant access to both HTTP and HTTPS traffic. Ensure that the access scope covers all cloud APIs, and then proceed to install the Ops Agent for monitoring and logging Figure 3.
4. Reiterate steps 1-3 to create another instance for OpenWhisk.

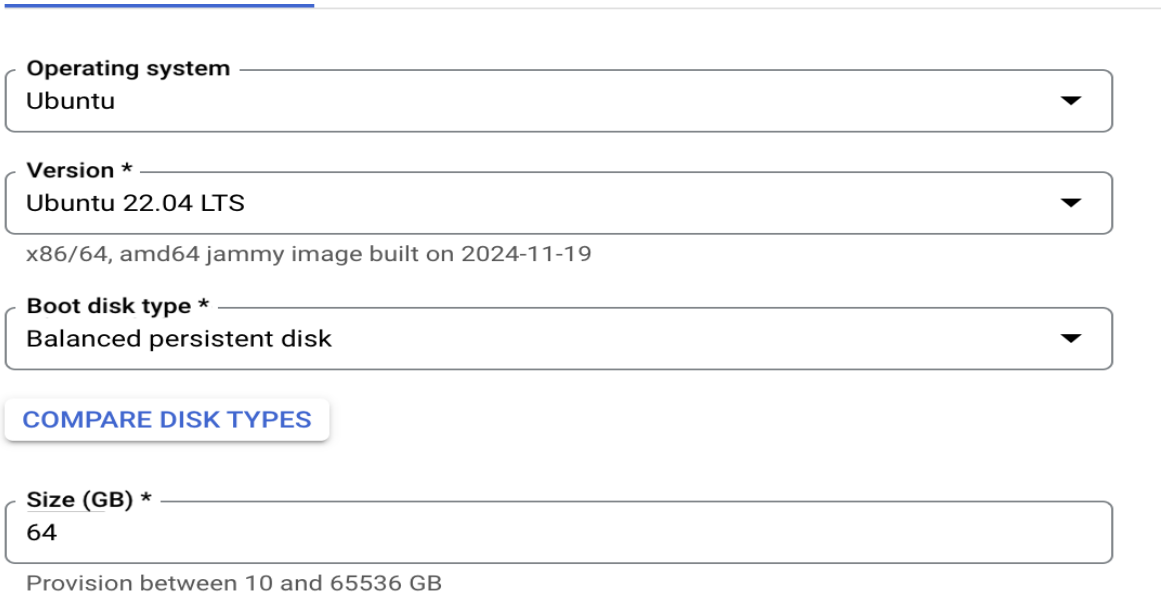


Figure 2: OS Configuration

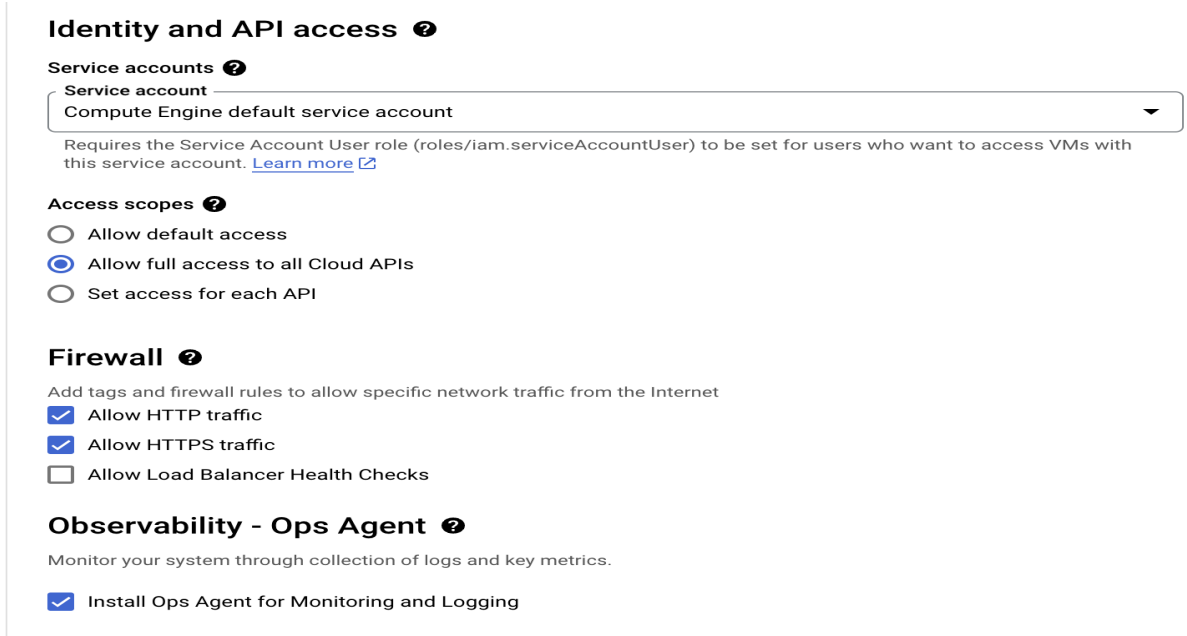


Figure 3: Identity and Firewall Configuration

2.2 SmartFasS and OpenWhisk installation

SSH into the server by clicking the SSH button in the Compute engine page as shown in Figure 4

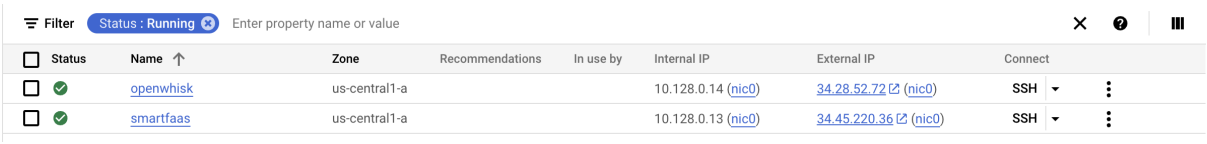


Figure 4: SSH into server

1. Run the following command to clone the repository:

   ```
   git clone https://github.com/Hariharan-Sathiyamoorthy/SmartFasS.git
   ```

2. `cd` into the `SmartFasS/scripts` folder

3. Execute the command `./env_setup.sh` it will prompt you to select an option based on whether the instance is OpenWhisk or SmartFaaS server Figure 5. Choose the appropriate option to continue with the installation.The installation will take approximately 15mins

```
hari73118@openwhisk:~/SmartFasS/scripts$ ./env_setup.sh
Select an option:
1. Install SmartFasS setup
2. Install OpenWhisk setup
Enter your choice (1 or 2):
```

Figure 5: Choose Installation

4. After installation, simply run this command to set the path variables.

   ```
   source ~/.bashrc
   ```

5. The full installation log will be in `root/env_setupOut.log`. If there were any errors, they will be shown there.

6. On an OpenWhisk server, run this command to see if the KinD cluster is working correctly. It will show the pods running in the `openwhisk` namespace Figure 6.

   ```
   sudo kubectl get pods -n openwhisk
   ```

```
hari73118@openwhisk:~$ sudo kubectl get pods -n openwhisk
NAME                                      READY   STATUS      RESTARTS   AGE
owdev-alarmprovider-5d57d4b879-82lrq      1/1     Running     0          30h
owdev-apigateway-d6d756db8-mwx8v          1/1     Running     0          30h
owdev-controller-0                        1/1     Running     0          30h
owdev-couchdb-89b8469bc-zf94g             1/1     Running     0          30h
owdev-gen-certs-6hpcr                     0/1     Completed   0          30h
owdev-init-couchdb-zm2dw                  0/1     Completed   0          30h
owdev-install-packages-f6hnn              0/1     Completed   0          30h
owdev-invoker-0                           1/1     Running     2          30h
owdev-kafka-0                             1/1     Running     0          30h
owdev-kafkaprovider-7867778f74-5bvvz      1/1     Running     0          30h
owdev-nginx-67b8974c77-rxkxb              1/1     Running     0          30h
owdev-redis-74d7479464-g2zhn              1/1     Running     0          30h
owdev-wskadmin                            1/1     Running     0          30h
owdev-zookeeper-0                         1/1     Running     0          30h
wskowdev-invoker-00-198-prewarm-nodejs14  1/1     Running     0          3m38s
```

Figure 6: OpenWhisk Installation

2.3 Running the Experiments

2.3.1 SmartFasS Experiments

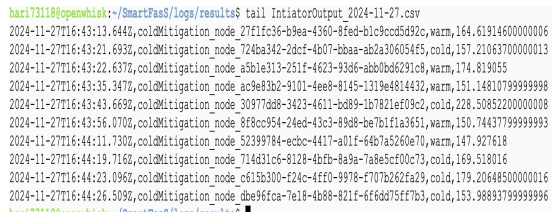
1. To run the experiments, navigate to the `SmartFasS` folder on SmartFasS server.
2. Run the following command to install all the dependencies:

```
npm install
```

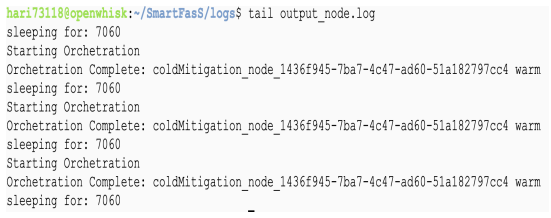
3. Run this command to execute the orchestrate function and start the JMeter script parallelly

```
npm run coldMitigation
```

4. there will be two logs files will be generated from this experiment `output_node.log` and `inIntiatorOutput.csv` both will be in the `SmartFasS/logs` folder, this will contain comprehensive information about the experiment Figure 7.



(a) Initiator Logs



(b) Orchestration Logs

Figure 7: SmartFasS Logs

2.3.2 OpenWhisk Experiments

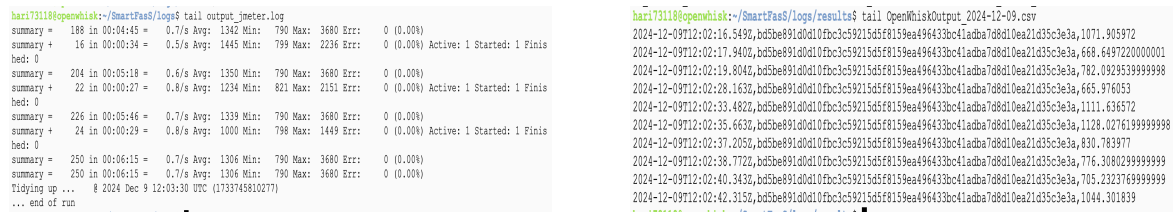
1. To run the experiments, navigate to the `SmartFasS` folder on OpenWhisk server.
2. Run the following command to install all the dependencies:

```
npm install
```

3. Run this command to execute the OpenWhisk function through JMeter script.

```
npm run OpenWhisk
```

4. This experiment is being run entirely with JMeter. The JMeter logs and results can be found in the `SmartFasS/logs` folder, where they are named `output_jmeter.log` and `inInitiatorOutput.csv`. This will have all the details about the experiment shown in Figure 8.



(a) JMeter Logs (b) OpenWhisk results

Figure 8: OpenWhisk Logs

2.4 Evaluating the Experiments

1. To evaluate the experiments, navigate to the `SmartFasS/colab` folder.
2. Inside the folder, the `analysis.ipynb` file can be found. This file contains the code to analyze the results of the experiments.

References

Apache OpenWhisk Documentation (n.d.). `https://openwhisk.apache.org/documentation.html`.

Google Cloud Console - Compute Engine (n.d.). `https://console.cloud.google.com/compute`.