CSC8426 - Emerging Technologies

Coursework 1

Assessment Overview

This assessment contributes 50% towards the total mark for this module. It is an individual exercise: no group work is permitted for the assessment. You are advised to watch all the instructional videos, provided above and in-line with the individual task specifications, as well as read all tutorial resources before you start answering these questions. Marks shown are indicative only.

The coursework report must be submitted on NESS by 4:30 pm on May 26, 2023. More details on the content of the report are given as part of Task 5. Additionally, you will be required to demonstrate successful executions of Tasks 1 - 4. Before the report submission deadline, you will be provided a 15 mins slot to conduct live demonstrations.

Objectives

This coursework helps you to learn and understand the fundamentals of programming and deploying Docker-based (an emerging cloud virtualisation technology) application hosting environment. By successfully completing the coursework, you will be able to gain hands on experience on the following interrelated aspects including:

- Configuring a Docker-based application hosting environment;
- Pushing and pulling images from the Docker Hub (global repository of software components' images maintained by the developers);
- Creating and deploying a complex web application stack consisting of multiple software components (e.g., database, web server, etc.);
- Monitoring ongoing performance of your application stack using the inherent features provided by the Docker environment.

Pre-Requisites

Before attempting this course work, you are advised to go through the training content and videos given at [https://github.com/ncl-iot-team/csc8426] which provides details on:

- how to download and install VMWare in your laptop/PC and
- how run a virtual machine image, which is packaged with all necessary programming environments required for this coursework, in your laptop/PC.

Specification of Tasks

The coursework consists of 5 tasks.

Task 1: Deploy a web application component in Docker Environment (7.5 Marks)

Task Objective : Understand and learn how to *pull and run* Docker image from Docker Hub using the command line interface.

Hints : Please first watch the video tutorial (Task1 EmergingTechnologies Coursework) given at [https://github.com/ncl-iot-team/csc8426].

A sample *Java web application component* image, named "nclcloudcomputing/javabenchmarkapp", has been uploaded to the Docker Hub. The image contains a ready-to-use implementation of a web application deployed in a Tomcat server (an open source web server). In terms of computational logic, the web application implements a prime number check on a large number. By doing so, the application can generate high CPU and memory load.

1. Pull and run the Docker image "nclcloudcomputing/javabenchmarkapp" from Docker Hub in the virtual machine running on your laptop/PC. Perform these tasks using the command line interface (CLI).
2. Verify whether the web application is working by pointing your web browser to [http://localhost:8080/primecheck]. The page should display the time taken to perform a prime number checking calculation.

Task 2: Deploy a complex web application stack in Docker Environment (12.5 Marks)

Task Objective : Understand how to create a complex web application stack consisting of multiple Docker images delivering distinct services including web server, load generation, database server, monitoring engine, and visualisation. This task will also help you in understanding how native *Docker Compose and Swarm techniques* can be leveraged to manage and deploy such complex web application stacks.

Hints : You are advised to carefully view the relevant video tutorial (Introduction to Docker swarm), given at [https://github.com/ncl-iot-team/csc8426], before attempting the below sub-tasks.

1. Pull the following images from the Docker Hub:
   (a) load generator Docker image (**locustio/locust**);
   (b) MongoDB Docker image (**mongo**);
   (c) cAdvisor monitoring Docker image (**gcr.io/cadvisor/cadvisor:v0.45.0**);
   (d) Docker Swarm Visualizer (**dockersamples/visualizer**);

2. Define a Docker compose file which contains all necessary configurations and instructions for deploying and instantiating the above (step 1) Docker images (web server, locust, MongoDB, cAdvisor, Docker Swarm Visualizer);

3. Create a web application topology, as shown in Figure 1, on a single Docker swarm node by using the above (step 2) Docker compose configurations.

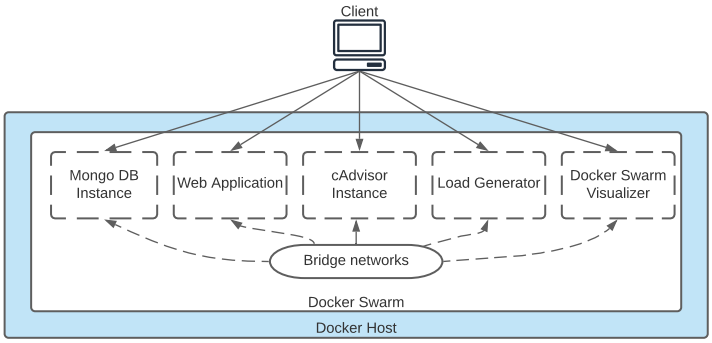

Figure 1: Deployment Architecture

Task 3: Build your own Docker image and push it to the Docker Hub (10 Marks)

Task Objective : Understand how to build a Docker image from scratch and how to push it to Docker Hub so that other developers can view, use and extend the same.

Hints : The video tutorial (Docker Build Tutorial) is provided at [https://github.com/ncl-iot-team/csc8426].

1. You need to build a Dooker image of a Java program. To download the source code of the provided program, use the following link [https://github.com/ncl-iot-team/cadvisor-scraper]. In terms of the application logic, the program implements features including how to extract the monitoring information from the Docker cAdvisor container (an running instance of **gcr.io/cadvisor/cadvisor** image) and how to insert these monitoring information into a MongoDB Docker containter (an running instance of **mongo** image). The cAdvisor is able to monitor run-time performance (CPU load, network I/O and memory usage) of all instances of Docker images running within a *Docker Swarm*.

2. Edit the provided Java code using the Eclipse editor (already installed in the provided virtual image). The instructions in Java code that need editing include *containerId, mongodbAddress, and endpoint address.*

3. After code modification, pack the program as a standalone Docker image and push it to the Docker Hub. Name the image as *cadvisor-scraper*.

Task 4: Fully deploy and run the complex web application stack and undertake performance benchmarking activities **(5 Marks)**

Task Objective : To learn and understand how to deploy complex web application stack, how to benchmark their run-time performance, and how to store the benchmarking results into a database server.

Hints : The video tutorial (Introduction to cAdvisor) is provided at [https://github.com/ncl-iot-team/csc8426].

1. Pull *cAdvisor-scraper* image, created in Task 3, back to the virtual machine on your laptop/PC.
2. Send dynamic workload (e.g., http requests) to the *Web Application* Image Instance (in other words Web Application Container) by using the *Load Generator* Instance.
3. Run *cAdvisor-scraper* as a standalone container away from the swarm.
4. Verify that *cAdvisor-scraper* instance is logging the run-time performance statistics to the *MongoDB* instance.

Task 5: Report **(15 Marks)**

Prepare the Final Report in plain English. The report should consist of:

1. Detailed response to each task and related sub-tasks;
2. Screenshots of running services in the Docker Environment;
3. Screenshots of Code Snippets and/or Docker console;
4. Plots of Benchmarking results using any of the tools (e.g. R, MATLAB, Excel).
5. Discussion of the results and related conclusions.