Project1

Bina Ramamurthy

9/12/2021

Distribured Pub/Sub System using Docker Containers

1. Problem statement

Publish/Subscribe (or pub/sub for short) is a popular indirect communication system. Pub/sub systems disseminates events to multiple recipients (called subscribers) through an intermediary. Examples of successful pub/sub include Twitter and "Bloomberg terminal"-like financial systems. In this project, we will emulate a pub/sub system using the light weight virtualization or container technology in Docker [1]. The main resource for learning about pub/sub is your text book [2] Chapter 6.

2. Learning outcomes

1. Understand the architecture, components, algorithms and operation of a pub/sub model.
2. Implement a centralized pub/sub distributed application.
3. Implement a distributed pub/sub distributed application using all nodes managing subscribers and events.
4. Implement a distributed rendezvous based efficient pub/sub system.
5. Learn the concept of container technology and Docker conatiners [1].
6. Deploy a set of interacting Docker images to emulate a pub/sub application (publishers, subscribers, and pub/sub mediators(s))

3. Project Description

Phase 1: Learn container technology Docker – 15% – Deadline: 9/27/2021 by 11.59PM

Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels. – From Wikipedia

Demonstrate with your **own laptop** and simple docker container-based application development, deployment and interaction. Use a sample application in your **subject of interest**. This subject of interest should be unique to each team. Meet your TA to sign up your team members, your emails, person numbers and the suject area of choice for the project in the Google sheet provided. https://docs.google.com/spreadsheets/d/1jLmP8bwvrmpajuuzgjIZPgxxJDpC2gWQ3kCZNdVhZOA/edit?usp=sharing

What to do?

Download Docker, install it, and learn how it works. Create a simple container Docker application that bundles a user interface client and server, code logic, and database (even file system will do.) Demonstrate it by a user interacting with the container with a message. This message is processed by the logic server and stored in the database. The data stored in the database is accessed and returned to the user via the web interface.

Expected Outcomes

On completion of this phase a student will have:

1. Command line experience and skill
2. Docker purpose and usage
3. Programming language chosen
4. Own a laptop or rent your own
5. Docker setup and working on own laptop
6. Groups members (1 or2): the group is functional, i.e., both partners are contrubuting (if not F for the project)
7. Chosen a subject area and possible topics in the subject area
8. Completed docker app with documentation/readme

What to submit?

Proof of ownership of laptop: both team members have to have a laptop, both need to submit on ublearns
NO LATE submission: submit early if you anticipate any issues around due date.

Command line interface example

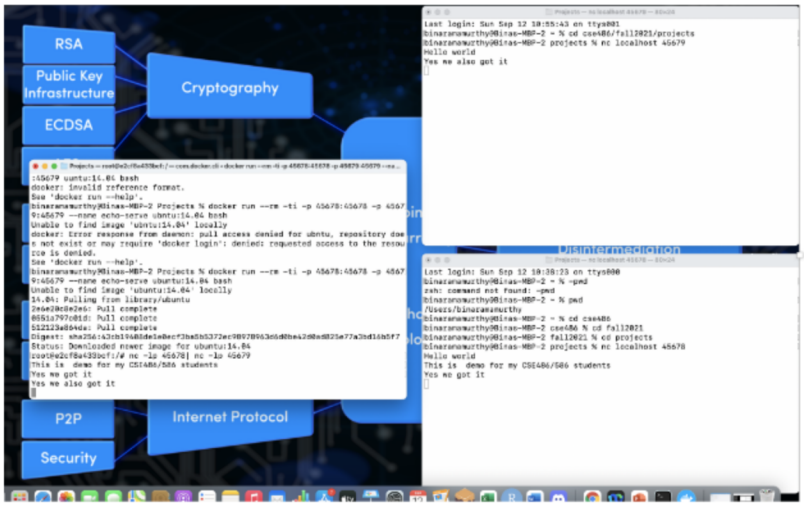

Figure 1: Docker command line example

Grading

Demo required (in-person - zoom acceptable: 15, 10, 5, 0 (Completion criteria: 100%, 75%, 50%, 0%))

Phase 2: Centralized pub/sub – 45% – Deadline: 10/18/2021 by 11.59PM

Decide the subject area (e.g. football). Determing publishers of football related messages, different topics of interest around football, and the possible consumers of these messages. Implement a centralized system, that receives (pull or push?) messages, analyzes the messages, partitions them into topic pools, and provisions them for the subcribers. Then the system pushes it to the subscribers. Decide on the methods and mechanisms you will use for the various functions involved.

What to do?

Now that you are familiar with Docker, implement a centralized version of the pub/sub application. You will need to implement the functions described in the Figure 1, the event generator and subscription generator for fully testing the application.

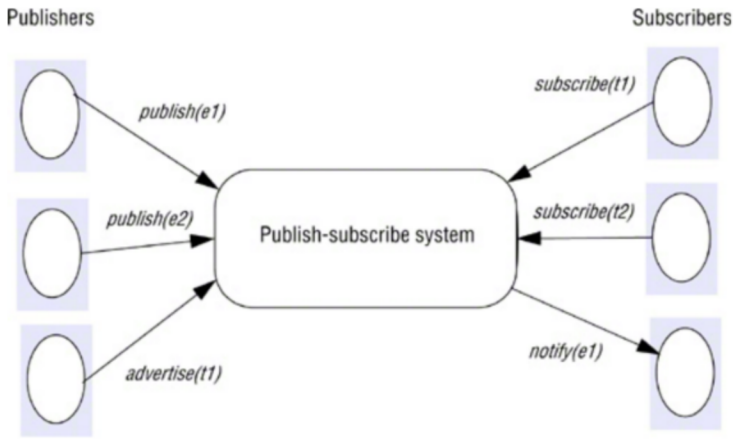

Figure 2: Centralized pub/sub system

Expected Outcomes

1. A working pub/sub system in the subject area you signed up.
2. The docker-based distributed system is implemented where publishers (docker application) obtain real events in your subject area of interest and send them to a mediator docker application. Mediator docker application filters the received messages and pushes to subscribers (docker applications) the messages on the topic of their interest. These messages are eventually displayed on the subscriber's interface.
3. Able to filter the published messages under at least **three** topics.
4. Manage a database of publishers, subcribers, raw messages and filtered messages.
5. A user interface subscripbers for specifying the topics and receiving messages.
6. Use real sources (APIs) for publisher messages.

What to submit?

1. An architectural diagram for pub/sub system delivering messages to users on selected topic on a chosen subject.
2. The implemented system with instructions on how to run it and interact with it.
3. The Docker images for the system's components.
4. A demo with you TA on your laptop.
5. Submit on ublearns before the due date.

Grading

Demo required (in-person - zoom acceptable: 45, 40, 30, 25, 15, 10, 0 (Completion criteria: 100%, 90%, 80%, 60%, 40%, <40%, 0%))

Phase 3: Distributed pub/sub – 30% – Deadline: 11/1/2021 by 11.59PM

Armed with the knowledge and skills from earlier phases, you are ready to design, and develop the distributed version of pub-sub. Design, develop, deploy and test a distributed pub-sub to improve scalability and accessibility (and other characteristics possible with distrbuted architectures.)

Next step is to implement the distributed version of the pub/sub as described by rendezvous-based routing described in Figure 6.9, 6.10 of Coulouris text, one of which shown here.

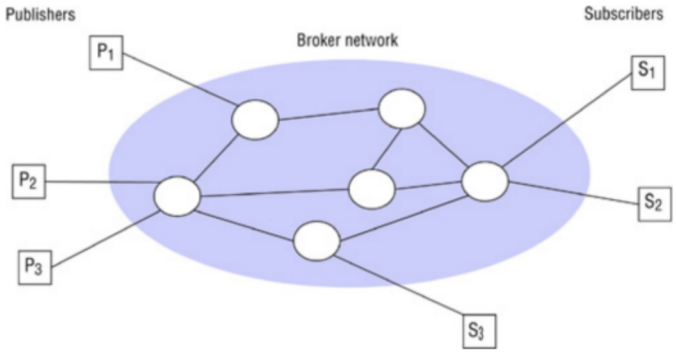

Figure 6.9 A network of brokers

Figure 3: Decentralized pub/sub system

Expected Outcomes

1. A distributed verson of the pub/sub system implemneted using containers.
2. Reuse the components developed in Phase 2.
3. Details on the differences explained with a diagram.

Grading

Demo required (in-person - zoom acceptable: 30, 25, 20, 15, 10, 0 (Completion criteria: 100%, 90%, 80%, 60%, 40%, 0%))

Documentation – 10%

1. Comment your code appropriately.
2. Provide diagrams explaining yoru design.
3. Provide README files to instruct how to run the submitted code and how to run the code, interact and the expected outputs.

Help and Academic Integrity

1. Please visit TA office hours on zoom and register your subject area and your team.
2. Enroll in Piazza and discuss any questions you have. **Do NOT** post code on Pizza. (There is no Discord server for this course.)
3. Start working on the project today.
4. If you found cheating, copying from each other (teams) and from the web and other outside sources **YOU WILL GET AN F** for the course.

References

1. Use containers to build, run and share your applications, https://www.docker.com/resources/what-container, last viewed September 2021.
2. Distributed Systems, By: George Coulouris; Jean Dollimore; Tim Kindberg; Gordon Blair Publisher: Pearson Print ISBN: 9780132143011, 0132143011 eText ISBN: 9780133464917, 0133464911 Edition: 5th Pages: 1008 Copyright year: 2012 https://www.vitalsource.com/referral?term=9780133464917