Distributed Systems

P2: Kubernetes

by
Asger Song Høøck Poulsen
Firas Harbo Saleh

A Distributed Systems
Project

December 3, 2023

Contents

1 Introduction

In the realm of distributed systems, the efficient management and coordination of multiple service instances are paramount for ensuring high availability and resilience. Kubernetes, an advanced container orchestration system, has emerged as a pivotal technology in this domain, enabling scalable and efficient management of containerized applications. Similarly, the Bully algorithm plays a crucial role in distributed systems, providing a robust mechanism for leader election, which is vital for maintaining operational consistency and fault tolerance.

This project delves into the practical application of these concepts through the deployment of a Kubernetes-powered web application - a digital fortune cookie service. By integrating the Bully algorithm within a Kubernetes environment, this project not only demonstrates a novel use case in distributed computing but also explores the synergy between container orchestration and leader election algorithms.

This report will cover the design and implementation of the application, the challenges encountered during the process, and will conclude with a discussion of the results and a brief overview of the lessons learned from the project.

2 Methods and materials

This section of the report elaborates on the comprehensive approach and the array of tools and technologies employed in the development of the Kubernetes-based fortune cookie service, underscored by the Bully algorithm for leader election. The methodology adopted for this project is a testament to the harmonious blend of software engineering practices, containerization techniques, and orchestration strategies. Here, we dissect the various components and processes that form the backbone

of this project, ranging from the Kubernetes infrastructure and the Bully algorithm's integration to the development environment and tools that facilitated the seamless realization of the service.

2.1 Kubernetes

Kubernetes, a cornerstone of modern cloud-native applications, is an open-source container orchestration system for automating software deployment, scaling, and operations of containerized applications. In this project, Kubernetes is not just the infrastructure platform but also an integral part of the application's architecture. The following aspects of Kubernetes play a crucial role in the implementation of the fortune cookie service:

- **Pods**: As the smallest deployable units created and managed by Kubernetes, pods are used to host instances of the application. Each pod runs a containerized version of the fortune cookie service.
- **Replication and Scaling**: Kubernetes manages the desired number of pod replicas, ensuring high availability and load distribution. This is vital for maintaining service continuity, especially during leader election phases.
- **Services and Networking**: A Headless Service in Kubernetes is used for enabling network identity to the pods. This allows pods to discover each other and communicate, which is essential for the Bully algorithm to function.
- **Deployment and Management**: Kubernetes streamlines the deployment process, allowing for consistent and repeatable deployment of the application, and provides tools for monitoring and managing the application's state.

2.2 Bully Algorithm

At the heart of this project lies the implementation of the Bully algorithm. The Bully algorithm, described in [1], is a classic method used in distributed systems for leader election. It is characterized by its simplicity and straightforward approach to determining the leader node in a cluster of computers. In the context of Kubernetes, this algorithm assumes a new dimension of relevance:

- **Election Process**: The algorithm operates by electing the pod with the highest identifier as the leader. When a pod believes it should be the leader (e.g., when the current leader fails), it starts an election process to assert its role.
- **Implementation in Kubernetes**: Implementing the Bully algorithm in Kubernetes involves pods communicating over the cluster network to perform the election process. Each pod must be aware of others and capable of sending and receiving election messages.
- **Handling Failures**: A key feature of this project is the ability of the system to handle leader pod failures gracefully. The Bully algorithm ensures that a new leader is elected swiftly, minimizing downtime and maintaining the availability of the fortune cookie service.
- **Integration with the Service**: The elected leader pod hosts the web interface of the fortune cookie service. This integration demonstrates the practical application of the algorithm in a real-world scenario.

The project's focus on the Bully algorithm showcases its potential in a Kubernetes environment, particularly in handling scenarios involving leader failure and the subsequent re-election process, by ensuring the continuity of the fortune cookie service.

2.3 Web Application

The Web Application serves as the user-facing component of our Kubernetes-based fortune cookie service. It represents the culmination of the project's backend functionalities, providing a simple, interactive and intuitive interface for users to engage with the service. The following aspects of the web application are worth highlighting:

- **User Interface Design**: The design of the web application is focused on simplicity and ease of use. It features a clean layout with a prominent button for requesting a fortune and a display area where the fortune appears.
- **Client-Side Scripting and Interactivity**: JavaScript is utilized to enhance the interactivity of the web application. It handles user events, such as button clicks, and manages the communication with the backend to fetch and display fortune cookies. This scripting ensures that the user interface is dynamic and responsive to user actions.
- **Backend Integration**: The frontend seamlessly integrates with the backend service hosted on Kubernetes. It communicates via HTTP requests, retrieving fortune cookie data from the service managed by the elected leader pod.
- **Frontend Technologies**: The application is built using standard web technologies - HTML for structure, CSS for styling, and JavaScript for functionality. These technologies were chosen for their wide support across browsers and ease of integration.

This web application acts as the gateway for users to interact with the underlying Kubernetes-managed services, illustrating the effectiveness of combining modern web technologies with cloud-native backend systems.

2.4 Development Environment and Tools

The development of the fortune cookie service and the Bully algorithm implementation was done using the following tools and technologies:

- **Docker**: Docker is a containerization platform that allows for the creation and deployment of containerized applications. It was used to containerize the fortune cookie service and the Bully algorithm implementation.
- **Kubernetes**: Kubernetes is an open-source container orchestration system for automating software deployment, scaling, and operations of containerized applications. It was used to deploy and manage the fortune cookie service and the Bully algorithm implementation.
- **Minikube**: Minikube is a tool that enables the running of Kubernetes locally. It was used to create a local Kubernetes cluster for testing and development purposes.
- **Kubectl**: Kubectl is a command-line tool for interacting with Kubernetes clusters. It was used to deploy and manage the fortune cookie service and the Bully algorithm implementation.
- **Git**: Git is a distributed version control system for tracking changes in source code during software development. It was used to manage the source code of the fortune cookie service and the Bully algorithm implementation.
- **GitHub**: GitHub is a web-based hosting service for version control using Git. It was used to host the source code of the fortune cookie service and the Bully algorithm implementation.

- **Visual Studio Code**: Visual Studio Code is a source-code editor. It was used to write the source code of the fortune cookie service and the Bully algorithm implementation.

3 Experiments, results, and discussion

3.1 Kubernetes Configuration

1. **Deployment Configuration**: Defined in the `deployment.yaml` file, the deployment configuration specifies the desired state of the fortune cookie service. It defines the number of replicas, the container image to use, environmental variables, and the container port.

2. **Headless Service Configuration**: Specified in `headless-service.yaml`, this service configuration defines the network identity of the pods. It enables the pods to discover each other and communicate, which is essential for the Bully algorithm to function.

3.2 Application Architecture

Overview

The architecture of the fortune cookie service is designed to be scalable, resilient, and distributed. It comprises several components, each with a specific role in the overall functionality of the service. The key components include:

- **Frontend**: The user interface of the web application, responsible for presenting the information and interacting with the user.
- **Backend**: The backend service running in Kubernetes pods, which processes requests, implements the Bully algorithm, and manages the delivery of fortune cookie texts.
- **Data Storage**: While this application primarily operates in-memory, it can be extended to include a database or file system for persistent data storage.
- **Networking**: Internal communication between pods, facilitated by Kubernetes networking, especially vital for the Bully algorithm's operation.

Sequence Diagram

A sequence diagram is included to illustrate the flow of operations from the moment a user requests a fortune cookie to the display of the fortune. [Include diagram here]

Class Diagram

The depicted class diagram provides an overview of the structure of `app.py`. [Include diagram here]

3.3 Deployment and Testing Process

This subsection details the critical steps involved in deploying and testing the fortune cookie service within a Kubernetes environment. It highlights the processes of building and deploying the application, followed by load testing and failure simulation, ensuring the system's robustness and reliability.

1. **Building and Pushing Docker Images**: The first step in the deployment process is to build the Docker images for the fortune cookie service and the Bully algorithm implementation. The images are then pushed to Docker Hub, a cloud-based repository for storing and sharing container images. This step is streamlined using a `Makefile`.

2. **Apply Kubernetes Configuration**: The next step is to apply the Kubernetes configuration files to the cluster. Kubernetes configurations were applied using `kubectl apply` commands for the deployment and the headless service. This step is also streamlined using a `Makefile`.

3. **Load Testing**: Load tests were performed by sending rapid HTTP requests to the service endpoint to evaluate the systems' response times and robustness under heavy load.

4. **Failure Simulation**: Leader pod failures were simulated by manually deleting the leader pod responsible for serving the web interface and observing the behavior of the system.

3.4 Monitoring and Observations

Monitoring tools within Kubernetes, such as the Dashboard and log output, were used to observe the system's behavior during experiments. Key observations include response times during load testing and the time taken to elect a new leader during failure simulation.

3.5 Discussion

4 Conclusion and perspectives

4.1 Conclusion

4.2 Lessons Learned

4.3 Future Work

References

[1] H. Garcia-Molina, "Elections in a distributed computing system," *Transactions on Computing Systems C-31*, 1982.

[2] M. van Steen and A. S. Tanenbaum, *Distributed Systems*, 4th ed. Maarten van Steen, 2023, ch. 5.

[3] Kubernetes, "Kubernetes documentation / service," 2023, last accessed 3 December 2023. [Online]. Available: https://kubernetes.io/docs/concepts/services-networking/service/#headless-services.