Study and Implementation of the Sieve Protocol in BFT

Samuele De Tuglie
samuele.detuglie@studio.unibo.it
ID: 0001137795

Pablo Sebastian Vargas Grateron
pablo.vargasgrateron@studio.unibo.it
ID: 0001137787

Emanuele Artegiani
emanuele.artegiani@studio.unibo.it
ID: 0001140446

December 6, 2023

Abstract

This project centers around comprehending and implementing the Sieve Protocol within the context of Non-determinism in a Byzantine Fault-Tolerant setting. The primary objective is to develop a functional implementation of the Sieve Protocol using Python, considering the challenges posed by Byzantine faults in distributed systems. The Sieve Protocol, designed for replicated state machines, incorporates non-deterministic operations to ensure fault tolerance. The implementation aims to facilitate a deeper understanding of distributed systems, Byzantine fault tolerance, and the intricate dynamics of handling non-deterministic operations within the context of the Sieve Protocol. This report provides an overview of the algorithmic details, the rationale behind its design, and the Python implementation specifics, offering a foundation for further exploration and refinement in the realm of fault-tolerant distributed systems.

Goals/Requirements

The main goals of this project include:

- **Understanding of Sieve Protocol**
  Conduct a preliminary study of the Sieve protocol to understand its design principles and functionality.
- **Algorithm Implementation**
  Develop a Python script to implement the Sieve Protocol, following the specifications outlined in the given paper [1].
- **Automated Testing**
  Implement a comprehensive suite of automated tests to ensure the reliability and stability of the code under various conditions.

Expected Work Plan

To achieve the outlined goals and objectives of this project, the following work plan is proposed:

- **Understanding**
  Conduct an in-depth study of the Sieve Protocol with a specific emphasis on its application in achieving Byzantine Fault Tolerance in distributed systems. Explore the theoretical foundations of Byzantine Fault Tolerance, understanding the challenges posed by malicious nodes and the strategies employed by the Sieve Protocol to address these challenges.
- **Implementation**
  Develop a Python program to simulate the Sieve Protocol.
  Create a suite of automated tests that specifically target Byzantine fault scenarios, simulating various malicious behaviors and verifying that the Sieve Protocol effectively detects and mitigates these faults. Ensure that the implementation adheres to relevant BFT specifications and standards.
- **Documentation**
  Provide a detailed and exhaustive report documenting the implementation of the Sieve Protocol for Byzantine Fault Tolerance. Explain the theoretical underpinnings of Byzantine Fault Tolerance and how the Sieve Protocol addresses these challenges.
  Document the steps to run the Python program, simulate Byzantine faults, and observe the system's behavior. Include comprehensive examples of Byzantine fault scenarios and the expected outcomes. Discuss any trade-offs made during the implementation and the rationale behind design decisions.
  In the documentation, consider adding a section that discusses the scalability and performance characteristics of our BFT implementation.

Bibliography

[1] M. V. Christian Cachin, Simon Schubert. Non-determinism in byzantine fault-tolerant replication. *IBM Research - Zurich*, 03 2016.