HDS Serenity Ledger 1st Stage

MEIC-A 2023/2024
Group Campus: Alameda
Group Number: 22

Group Members
Daniel Pereira, 99194
Ricardo Toscanelli, 99315
Simão Gato, 99328

Contents

1 Introduction

This report details the initial stage of development for the HDS Serenity Ledger project. Our analysis of the existing codebase revealed a strong foundation with implemented stubborn links, nodes, node services, and a partially implemented IBFT protocol lacking round change functionality.

Stage 1 focused on building upon this base by implementing a client, a client service/library for interaction, the missing round change functionality within IBFT, improvements to the existing stubborn links, and the establishment of a testing framework. This report outlines the progress made in these areas, paving the way for a complete and operational distributed system.

2 Design

2.1 Client

Leveraging the similarities between nodes and clients, we adopted a solution inspired by the existing node implementation. Client configuration utilizes the same ProcessConfigBuilder to define settings. However, to facilitate communication with clients, we introduced a new field named clientPort within the node configuration JSON file. This allows clients to connect to nodes using the designated port.

The core functionality of the client resides within a dedicated class responsible for receiving user input and forwarding it to the client service for proper handling.

2.2 Client Service

Focusing on seamless user interaction, we developed a client-side library to interact with the enhanced node service. This library acts as a bridge, allowing clients to effortlessly communicate with the network. The node service itself underwent minor modifications to initiate a consensus round upon receiving a client request. However, to ensure order and prevent overwhelming the system, the client library is designed to wait for a response from the server before accepting a new request. This ensures serialized communication and avoids potential conflicts.

2.3 Links

Building upon the existing foundation of stubborn links, which ensure reliable message delivery, we aimed to enhance the security of our communication layer. Drawing from the theoretical concepts introduced in our lectures, we implemented authenticated perfect links. This approach adds a layer of authentication to messages exchanged between clients and nodes.

In essence, we implemented digital signatures for all messages. This involves signing messages at the send method within the Link class and verifying the signatures at the receive method of the same class. This ensures that messages originate from authorized sources (either clients or nodes) and have not been tampered with during transmission.

2.4 Round Change

3 Tests

4 Conclusion