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 system. This report outlines the progress made in these areas, paving the way for a complete and operational system.

2 Design

2.1 Client and Client Service

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.

To ensure seamless user interaction, we developed a client-side library that 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.

2.2 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.

2.3 Round Change

To ensure smooth transitions between consensus rounds within the IBFT protocol, we implemented the round change mechanism based on the algorithm outlined in the paper *"The Istanbul BFT Consensus Algorithm"* by Henrique Moniz.

This algorithm dictates a dynamic timeout value for each round. This timeout is calculated using an exponential function, where the function considers the current round number and outputs the corresponding waiting time in seconds. As the round number increases, the timeout value increases exponentially. This approach prevents nodes from waiting indefinitely for a response in the case of failures and facilitates timely progression to the next round.

3 Tests

4 Conclusion

Stage 1 development has yielded a solid foundation for our project. The implemented client, client service, and authenticated links demonstrate a well-designed and secure communication layer. The round change mechanism based on the Istanbul BFT protocol provides a functional solution for transitioning between consensus rounds. However, further work is necessary to refine the round change algorithm and address potential edge cases.

Throughout this stage, we established a testing framework to evaluate system functionality under various scenarios. The results from these tests provide confidence in the project's direction.