DepChain - Stage 1

Simão de Melo Rocha Frias Sanguinho[1[102082]], José Augusto Alves Pereira[1[103252]], and Guilherme Silvério de Carvalho Romeiro Leitão[1[99951]]

Instituto Superior Técnico, Lisboa, Portugal

Abstract. This project aims to develop a simplified permissioned blockchain system, named Dependable Chain (DepChain), with high dependability guarantees. The system is designed to be built iteratively, with the first stage focusing on the communication and consensus layer of a simple blockchain implementation, and the development of a client and a library that can interact with the blockchain system. The project leverages the Byzantine Read/Write Epoch Consensus algorithm, with simplifying assumptions such as static system membership, a predefined leader process, and a Public Key Infrastructure (PKI). For message communication, the implementation will use authenticated perfect links, with the assumption that the network is unreliable: it can drop, delay, duplicate, or corrupt messages, and communication channels are not secured. The implementation is structured in Java, utilizing the Java Crypto API for cryptographic functions, and is designed to handle malicious behavior from a subset of blockchain members while ensuring safety and liveness under the assumption of a correct leader. The final submission for stage 1 includes a self-contained zip archive with the source code, demo tests, and a concise report detailing the design, threats, and dependability guarantees of the system.

Keywords: Blockchain · Byzantine Fault Tolerance · Consensus Algorithm · Dependability · Java Implementation

1 Introduction

This report presents the design and implementation of a Byzantine Fault Tolerant (BFT) blockchain service, designed to withstand malicious behavior from a subset of blockchain members and operate reliably in an unstable network environment. The system is resilient to arbitrary (Byzantine) behavior from faulty nodes and can handle unreliable network conditions, including message drops, delays, duplication, and corruption, without relying on secure communication channels. To achieve consensus in such adversarial conditions, the project uses the Byzantine Read / Write Epoch Consensus algorithm, as described in the course book [1] (Algorithms 5.17 and 5.18). The report outlines the system architecture, which includes the network, client, library, blockchain, and consensus layers. It also describes how the system addresses various Byzantine attack scenarios, ensuring safety and liveness under the assumption of a correct leader. Finally, the report concludes with key findings, lessons learned, and potential areas for future improvement.

2 Architecture

2.1 Network Layer

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce non laoreet mauris. Aliquam vel sem neque. Sed sit amet risus in enim volutpat porttitor nec nec risus. Donec ultricies tortor at massa iaculis venenatis. Suspendisse aliquam justo dictum, faucibus magna consectetur, scelerisque erat. Integer quis convallis dui. Integer ante nulla, tincidunt quis dignissim et, condimentum non lorem.

2.2 Client Layer

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc ac erat nibh. Morbi quis accumsan turpis. Etiam est augue, vehicula ut erat sit amet, molestie vehicula lacus. Quisque eu nunc nulla. In nisi ex, auctor quis nunc laoreet, rutrum dapibus nulla. Vestibulum eget arcu sed mi hendrerit porta. Donec risus est, imperdiet a sem vel, elementum malesuada nisl. Duis posuere mollis erat eu egestas. Donec facilisis ut neque nec bibendum. Integer et leo id enim egestas consequat in id eros. Vivamus eget quam sapien. Ut efficitur cursus risus, tincidunt dapibus magna tempus id. Integer a tincidunt ipsum, vel volutpat leo..

2.3 Library Layer

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec facilisis et urna lobortis finibus. Nulla molestie lorem lacus, nec suscipit arcu mattis ac. Proin sed vehicula magna. Cras a iaculis velit, eu fermentum nisl. Pellentesque at magna in massa scelerisque suscipit sed ac tortor. Vivamus a bibendum leo, eget blandit felis. Donec vulputate ultrices dignissim. Donec vel elit a massa pellentesque euismod eget at velit. Aliquam eget est et urna auctor sodales. Aenean vulputate finibus libero ac pretium. Etiam lacinia ultrices odio, vitae bibendum metus accumsan sed. Etiam sit amet condimentum eros, pulvinar pharetra nisi. Phasellus ac purus a libero euismod ultrices. Duis feugiat, mi id facilisis blandit, magna magna commodo ante, ut porta lacus justo ut neque. Donec eget nisl feugiat, pretium libero eu, mollis dolor.

2.4 Blockchain Layer

liquam facilisis ante lacus, at scelerisque libero iaculis non. Etiam ex velit, iaculis blandit tristique ac, imperdiet a lectus. Etiam condimentum pharetra lectus non elementum. Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Integer imperdiet urna ipsum, nec tempor libero tincidunt ut. Proin in sapien arcu.

2.5 Consensus Layer

Proin sed vehicula magna. Cras a iaculis velit, eu fermentum nisl. Pellentesque at magna in massa scelerisque suscipit sed ac tortor. Vivamus a bibendum leo, eget blandit felis. Donec vulputate ultrices dignissim. Donec vel elit a massa pellentesque euismod eget at velit.

3 Implementation details

3.1 Detail XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

3.2 Detail XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

3.3 Detail XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

3.4 Detail XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4 Possible threats and corresponding protection mechanisms

To demonstrate the system's resilience against various Byzantine scenarios, the implementation was tested under multiple configurations, each representing a different type of attack. The following sections provide a detailed explanation of how each attack is executed and how the system effectively mitigates it.

4.1 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.2 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.3 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.4 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.5 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.6 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.7 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.8 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.9 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.10 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

4.11 Threat XYZ

Cras quis bibendum erat. Aliquam massa tortor, euismod a venenatis eget, convallis ut odio. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

5 Conclusion

Donec vulputate ultrices dignissim. Donec vel elit a massa pellentesque euismod eget at velit. Aliquam eget est et urna auctor sodales. Aenean vulputate finibus libero ac pretium. Etiam lacinia ultrices odio, vitae bibendum metus accumsan sed. Etiam sit amet condimentum eros, pulvinar pharetra nisi. Phasellus ac purus a libero euismod ultrices. Duis feugiat, mi id facilisis blandit, magna magna commodo ante, ut porta lacus justo ut neque. Donec eget nisl feugiat, pretium libero eu, mollis dolor.

References

1. Christian Cachin, Rachid Guerraoui, Luís Rodrigues: Introduction to Reliable and Secure Distributed Programming. 2nd edn. Springer, 2011
2. Java Crypto API, https://docs.oracle.com/javase/8/docs/api/javax/crypto/package-summary.html