



Applicability and Performance of Cross-Blockchain Communications Protocol in Distributed Resource Marketplaces

Alivelu Laxmi Mani Harika Badam
Isaya Ntilema

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Master of Science in Telecommunication Systems. The thesis is equivalent to 20 weeks of full-time studies.

The authors declare that they are this thesis's sole authors and have not used any sources other than those listed in the bibliography and identified as references. They further declare that they have not submitted this thesis at any other institution to obtain a degree.

Contact Information:

Author(s):

Aivelu Laxmi Mani Harika Badam

E-mail: albd20@student.bth.se

Isaya Ntilema

E-mail: isnt20@student.bth.se

University advisor:

Prof Kurt Tutschku

Department of Computer Science

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Blockchain is a distributed and decentralized database that provides transparency and immutability of the stored data. Blockchain technology is currently being used in many different areas due to its powerful features which provide integrity, authenticity, and confidentiality of the data it stores.

In this thesis, blockchains are regarded as databases for storing data from the distributed marketplace. The marketplace is made of different blockchains. The stored data will then be shared among different chains within the marketplace, which implies cross-blockchain communication also known as crosschain communication. The focus of this thesis is to study crosschain communication protocol, its implementation, and performance measurements.

A basic system model is designed using the Cosmos SDK to implement the blockchains and IBC protocol is enabled to facilitate communication between the blockchains. The model is later modified to simulate different communication scenarios. The results are captured and analyzed using statistical methods and results are presented in graphs.

Keywords: Blockchain, Performance, Inter-Blockchain Communication, Packet, Relay.

Acknowledgments

We give our sincere thanks to our supervisor Kurt Tutschku, for his guidance and support throughout our thesis work. Many thanks also go to Roman-Valentyn Tkachuk for his support with setup and access to the lab environment through the Azure virtual machines. We also give thanks to our families and friends for their motivation and encouragement to push us through to make it to the end.

Contents

| | |
|---|-----------|
| Abstract | i |
| Acknowledgments | ii |
| 1 Introduction | 1 |
| 1.1 Background | 1 |
| 1.2 Objectives and Research Questions | 2 |
| 1.3 Outline | 3 |
| 2 Related Work | 4 |
| 2.1 Review Methodology | 4 |
| 2.2 Summaries of Related Publications | 5 |
| 3 Background Technologies | 7 |
| 3.1 Blockchain Technology | 7 |
| 3.1.1 Structure of a block | 7 |
| 3.1.2 Types of blockchain | 8 |
| 3.1.3 Blockchain infrastructure model | 9 |
| 3.2 Crosschain communication | 11 |
| 3.2.1 What is crosschain communication? | 11 |
| 3.2.2 Why is crosschain communication needed? | 11 |
| 3.2.3 Crosschain communication mechanisms and protocols | 12 |
| 4 Technologies and Methods | 17 |
| 4.1 Overview on Applied Research Methodologies | 17 |
| 4.2 Distributed digital marketplaces | 18 |
| 4.3 Cosmos | 19 |
| 4.3.1 Architecture | 20 |
| 4.3.2 Cosmos Network | 20 |
| 4.4 Inter-Blockchain communication protocol | 22 |
| 4.4.1 IBC protocol structure | 22 |
| 4.4.2 Security of IBC | 24 |
| 4.4.3 Why IBC protocol? | 25 |
| 4.5 Azure | 25 |
| 4.6 Matlab | 25 |
| 4.7 Statistics | 25 |
| 4.7.1 Random Variable | 25 |
| 4.7.2 Probability distribution | 26 |

| | | |
|----------|--|-----------|
| 4.7.3 | Auto-correlation | 26 |
| 5 | System Modelling | 27 |
| 5.1 | Requirements | 27 |
| 5.2 | Models description | 27 |
| 5.2.1 | Deterministic model | 29 |
| 5.2.2 | Random model | 29 |
| 5.3 | Model selection | 29 |
| 6 | Implementation | 31 |
| 6.1 | Requirements | 31 |
| 6.2 | Setting up the required applications | 32 |
| 6.3 | Preparing the input | 32 |
| 6.4 | Implementing the setup | 34 |
| 6.5 | Collecting the output | 37 |
| 7 | Results and Discussion | 40 |
| 7.1 | Experiment 1: Impact of short interval time | 40 |
| 7.2 | Experiment 2: Impact of medium interval time | 44 |
| 7.3 | Experiment 3: Impact of long interval time | 47 |
| 7.4 | Experiment 4: Impact of random interval time | 50 |
| 7.5 | Observation | 54 |
| 7.6 | Limitations | 56 |
| 8 | Conclusions and Future Work | 57 |
| 8.1 | Conclusions | 57 |
| 8.2 | Future work | 58 |
| | References | 59 |
| | A Supplemental Information | 63 |

List of Figures

| | | |
|------|--|----|
| 3.1 | Structure of a block | 8 |
| 3.2 | Blockchain infrastructure model | 9 |
| 3.3 | Hashlock Mechanism | 14 |
| 4.1 | Research workflow | 18 |
| 4.2 | Distributed marketplace | 19 |
| 4.3 | Architechure | 21 |
| 4.4 | Cosmos SDK ecosystem | 22 |
| 4.5 | IBC protocol structure | 23 |
| 4.6 | IBC module structure | 24 |
| 5.1 | System Model | 28 |
| 6.1 | output of starting source blockchain | 34 |
| 6.2 | output of starting destination blockchain | 34 |
| 6.3 | setting relayer output | 35 |
| 6.4 | connect relayer output | 36 |
| 7.1 | EXP1: Number of packets sent in each trip | 41 |
| 7.2 | EXP1: Round trip time of each packet(seconds) | 42 |
| 7.3 | EXP1: Round trip time of packets from 1 to 500(seconds) | 42 |
| 7.4 | EXP1: Round trip time of the packets from 500 to 1000(seconds) | 43 |
| 7.5 | EXP2: Number of packets sent in each trip | 45 |
| 7.6 | EXP2: Round trip time of each packet(seconds) | 45 |
| 7.7 | EXP2: Round trip time of packets from 1 to 500(seconds) | 46 |
| 7.8 | EXP2: Round trip time of the packets from 500 to 1000(seconds) | 46 |
| 7.9 | EXP3: Number of packets sent in each trip | 48 |
| 7.10 | EXP3: Round trip time of each packet(seconds) | 48 |
| 7.11 | EXP3: Round trip time of packets from 1 to 500(seconds) | 49 |
| 7.12 | EXP3: Round trip time of the packets from 500 to 1000(seconds) | 49 |
| 7.13 | The round trip time of packets | 50 |
| 7.14 | EXP4: Round trip time of each packet | 52 |
| 7.15 | EXP4: Round trip time of each packet(seconds) | 52 |
| 7.16 | EXP4: Round trip time of each packet from 1 to 500(seconds) | 53 |
| 7.17 | EXP4: Round trip time of packets from 500 to 1000(seconds) | 53 |
| 7.18 | EXP3 re-implementation: Round trip time of packets | 56 |

List of Tables

| | | |
|-----|---|----|
| 2.1 | Search strings used in Google Scholar and IEEE Xplore databases . . . | 5 |
| 7.1 | Results obtained in EXP1 | 40 |
| 7.2 | Results obtained in EXP2 | 44 |
| 7.3 | Results obtained in EXP3 | 47 |
| 7.4 | Results obtained in random model | 51 |
| 7.5 | Comparison of all experiments | 55 |

List of Abbreviations

| | |
|--------------|---|
| ABCI | Application BlockChain Interface |
| BFT | Byzantine Fault Tolerant |
| CLI | Command Line Interface |
| Cosmos SDK | Cosmos Software Development Kit |
| IBC protocol | Inter-Blockchain Communication protocol |
| ICS | InterChain Standards |
| PoS | Proof of Stake |
| PoW | Proof of Work |
| SPV | Simplified Payment Verification |
| TAO | Transport Authentication Order |
| TCP/IP | Transmission Control Protocol/Internet Protocol |

Chapter one highlights the aim and the purpose of this thesis. We will further discuss the background of this thesis, the research questions, and the objectives of this thesis. An outline of this thesis is also mentioned in the chapter.

1.1 Background

Blockchain technology is currently being implemented in different sectors due to its features and capabilities. The centrepiece of Blockchain technology is the decentralized structure which is considered to be tamper-proof, immutable, and time-stamped record keeping. In simple words, blockchain can be described as a distributed and decentralized database that provides transparency and immutability of stored data. There are many technologies used in the blockchain which are discussed in detail in further sections. With the introduction of blockchain into different sectors and increased use of blockchains due to its different features, many issues were also faced. Scalability is one of the issues and one of the proposed solutions for scalability is inter-blockchain or crosschain communication. In simple words, crosschain communication can be referred to as transferring of information between two or more, similar or different types of blockchain. Crosschain communications are mainly motivated by two requirements common in distributed systems: accessing data and accessing functionality that is available in other systems. crosschain communication protocols ensure that crosschain consensus and different trust assumptions based on the type of blockchains are met. Crosschain consensus relates to how participants on one blockchain are convinced of the state of a remote blockchain. It describes how parties associated with a source blockchain come to an agreement and communicate with a destination blockchain such that information from the source blockchain can be trusted. Safety, Liveliness, and Atomicity are the main properties of a crosschain communication protocol. crosschain communication and the applicability of crosschain communication is an area that needs more research.

One of the applications of crosschain communication is marketplaces. Some examples are the telecommunication marketplace, the peer-to-peer energy marketplace, and many more. Here we are mainly discussing the applicability of crosschain communication in P2P energy marketplaces. Starting off the drawbacks of the traditional energy marketplace and how crosschain communication can bring a change to the traditional system are discussed. The marketplace consists of different blockchain networks and communication between these blockchains forms a federation of blockchains. This thesis mainly focuses on investigating if crosschain

communication can take place. And study the applicability and performance of Cross-Blockchain communication protocols. And how these protocols are used in a distributed resource marketplace by taking an example of peer-to-peer energy marketplaces. The outcome is a working implementation of crosschain communication protocol and its performance analysis.

1.2 Objectives and Research Questions

The main focus and objective of this thesis are to study crosschain communication and investigate the implementation and performance of crosschain communication protocol for blockchains. And also study the change that these crosschain protocols can make in the context of distributed P2P energy marketplaces. While most investigations of blockchains in peer-to-peer energy use cases focus on single blockchains, this work will investigate the use of crosschain communication between at least two blockchains. To achieve our aim, we formulated the following objectives:

- Determine and study the type of crosschain protocols that are available.
- Determine if we can implement crosschain communication protocols in the blockchain-based energy marketplaces and what protocols are suited for the communication between these blockchain networks in the marketplaces.
- Provide architecture and initial implementation of crosschain communication protocol between at least two blockchains.
- Discuss the scalability and performance of crosschain communication protocols in distributed marketplaces.

To achieve our objectives, the following Research questions were formulated;

- RQ1: What are the requirements for crosschain communication protocols in distributed digital marketplaces?
- RQ2: What are the metrics which define the performance of crosschain communication protocols?
- RQ3: What is the performance analysis and observation for implementation of crosschain communication protocol?
- RQ4: How can one measure the performance of a crosschain communication protocol?
- RQ5: What are crosschain communication protocols suited for the communication between the blockchain networks?

1.3 Outline

The remaining part of this thesis work is organized into small chapters as follows: chapter 2 is the related work where we have mentioned an introduction to all the research papers referenced in this thesis along with the review methodology used to select these research papers. Chapter 3 is background technologies which gives a brief introduction to blockchain technology and crosschain communication. Chapter 4 is Technologies and Methods which describes the research workflow and all the technologies used to implement the practical system. Chapter 5 is System Modelling focusing on system modeling, and how we designed models to implement crosschain communication. Chapter 6 is Implementation which explains how the implementation of the system is done. Chapter 7 is Results and Discussion, where we talk about the results of our implementation along with the discussion of the findings and their interpretation. Here we give an explanation of our findings and report them in a systematic manner. We also thoroughly elaborate on the meaning of our findings. Chapter 8 is Conclusion and Future work which gives a conclusion and discusses future work for the thesis.

This chapter outlines a short summary of the previous research work done in this field and how it is used in our thesis and also mentions how it is described in the later chapters. And also describes the search methodology used in this thesis.

2.1 Review Methodology

A literature review has been conducted to have a general overview of the existing research in the fields of blockchain technology and crosschain technology. Some research questions have been formulated and are mentioned in 1.2 to have a better understanding and analysis of different concepts in the thesis. The literature review helps in finding out answers to these research questions which indeed help in analyzing how to implement and evaluate crosschain communication protocol. Search criteria, inclusion-exclusion criteria, and a list of libraries used are mentioned in the following sub-sections.

Search criteria

Databases like IEEE, Google Scholar, and Science Direct were mostly used to search for the research papers used for this thesis. Search strings have been used in these databases which are discussed in the table 2.1 below. Some research papers like [37, 41] were recommended by our supervisor. Research papers like [44], [42] were referred through the references of the other research papers while reading them. As crosschain communication is an emerging research area, not many research papers were found about the latest protocols, and implementations in the crosschain communication field. So, blogs like [23] (by platforms like Medium) were referred to have a better understanding of the latest concepts in this field. Blogs and articles like [2, 10] from the Official project websites like Cosmos SDK are also referred.

Inclusion criteria

The following are the criteria based on which the research papers are selected:

- The research papers should be relevant to our thesis (searched using the search strings in table 2.1)
- The research papers must be in English.

| Strategy | Search Strings |
|--|---|
| Background research work | blockchain AND crosschain OR distributed resource marketplace blockchain interoperability AND difficulties OR crosschain blockchain AND crosschain communication AND protocols OR mechanisms |
| Implementation and evaluation of crosschain communication protocol | inter-blockchain communication protocol AND IBC OR blockchain protocol AND performance analysis OR performance evaluation AND blockchain |

Table 2.1: Search strings used in Google Scholar and IEEE Xplore databases

- The research papers should not have duplicates.

All the research papers not falling in the Inclusion criteria are excluded, thus the exclusion criteria are the opposite. After filtering out the research papers using inclusion and exclusion criteria, the abstract has been read and if relates to our thesis topic or technology, a detailed study of the research paper has been done to understand the concepts in different technologies.

2.2 Summaries of Related Publications

Initially, for a detailed understanding of these technologies and existing research in this field, research on Blockchain, crosschain communication, different mechanisms and protocols in crosschain communication, and usage of crosschain communication in distributed resource marketplaces has been done. After this, research on how to implement crosschain communication protocol and what are the different platforms to implement the protocol, and the performance evaluation of a protocol were looked into. Detailed research has been done on the technologies like IBC protocol and Cosmos SDK framework. An overview of the research papers referred to is written below. Along with these research papers, many other official websites of projects have been referred to.

Research paper [45] describes a brief about blockchain. It contains different chapters related to blockchain history, blockchain categorization, consensus models, applications of blockchain, and everything related to blockchain in a very descriptive way. Paper [47] describes the research being done, challenges faced and future directions in blockchain technology from the technical perspective at the time of this paper being published. Research paper [21] describes the application of blockchain technology in multiple industries. [28] research paper analyses the conveniences and difficulties occurring when blockchain is implemented in different sectors. These research papers were referred to for learning and gaining information about blockchain technology and its applications, especially blockchain infrastructure, and its layers.

In research papers, [32] and [42] the authors describe the application and challenges of blockchain in the energy sector and how the blockchain has revolutionized the traditional energy system. [43] contains a survey of issues and challenges in

blockchain technology applied to smart cities. Related work and background knowledge of blockchain technology has been discussed in very detail in this research paper. Apart from this, the use of blockchain technology in smart cities for example smart grid systems has been discussed in detail.

In the research paper [30], the authors present a systematic literature review on issues and challenges faced in the scalability of public blockchains. The author does a deep investigation into the problem of scalability in the case of blockchains and the fundamental reasons and solutions behind this problem. research paper [46] also issues scalability problems in the case of blockchains and proposes solutions for the issue. The author also mentioned crosschain communication as one of the solutions for blockchain scalability.

The research paper [22] contains different chapters discussing blockchain interoperability. Topics like different types of interoperability, security, modeling, and implementation are discussed in detail. Mainly the mechanisms like the hash locking mechanism were discussed with very informative and detailed steps. In a research paper [40], the author describes the concept of the Internet of Blockchains and crosschain communication between homogeneous and heterogeneous networks.

In the research paper [41], authors have conducted a survey on blockchain-based telecommunication services marketplaces. The research paper describes various telecommunication use cases where crosschain technology can be used and makes a difference. This paper has been referred to study the application of crosschain in different service marketplaces, especially the energy marketplaces.

The research paper [31] is research in crosschain technology. It contains a summary of crosschain technology and the crosschain techniques present at the time of the paper. It also analyses the challenges in this technology and presents further discussions. Regarding the crosschain technology and its mechanisms research papers like [25], [38] were referred. In [36], the author presents a survey of all the available crosschain solutions and compares them according to their architecture.

In the research paper [37], authors have done a survey of different crosschain communication protocols and presented them based on different use-case scenarios. It also analyses these protocols under different aspects like security, and consensus algorithms used.

In the research paper [44], the notary election mechanism was proposed along with an introduction to the margin pool. A Notary election mechanism is used to select one notary from a group of notaries and the margin pool limits the misconduct of elected notary. Along with this, the authors have given a clear introduction to the mechanisms in crosschain communication (especially the Notary mechanism) with reference to many different research papers on each of these different mechanisms. Regarding crosschain mechanisms and protocols, other research papers like [35], and [48] are referred to study and understand the mechanisms in more detail.

Research papers like [5] have been referred to get familiar with the cosmos and different technologies used in the cosmos. [18] contains easy and detailed documentation of the cosmos, cosmos networks, frameworks used, and everything related to the cosmos.

3.1 Blockchain Technology

The core concepts of blockchain first emerged in the late 1980s and early 1990s. Later on, in 2008 improved concepts combined and applied to electronic cash were described in the paper, Bitcoin: A Peer-to-Peer Electronic Cash System, which was published pseudonymously by Satoshi Nakamoto. Later in 2009 Bitcoin cryptocurrency blockchain network was established followed by the schemes discussed in Nakamoto's paper. Bitcoin was the first blockchain application. In a short time, bitcoin has achieved widespread use compared to other electronic cash schemes. With the increased use of bitcoin, and due to its many features, blockchain has made its way into different industries for many applications. [45]

Blockchain is an implementation of distributed ledger technology. In simple words, blockchain can be defined as a distributed and decentralized database that provides transparency and immutability of stored data. As blockchains operate in a peer-to-peer model without the involvement of any third party, it does not have a central governing authority and all the security concerns that come with normal databases. It bundles the pieces of data into blocks, where each block contains a reference to the previous one, thus, forming a chain of data blocks; hence the name blockchain. Usually, in blockchain, a ledger is maintained to track the ownership and transfer of the assets. Information stored in the ledger and the blocks are replicated on every node in the P2P network, thus preventing data loss and providing data immutability. [41] For this data to be verified, added to the blocks, and to maintain integrity between the nodes, different consensus algorithms are used in blockchain technology. In order to eliminate the need for third-party and to simplify transactions blockchains use smart contracts. Along with these, there are also many techniques underlying this terminology consisting of cryptography, mathematics, networking, and the sharing economy model. [26] A brief overview of different concepts of blockchain technology is discussed in the following sections:

3.1.1 Structure of a block

Each block contains a header and body, where the header contains a block number, the time-stamp of when the block has been created, a hash of the previous block, Merkle root which is the root hash of the Merkle tree (it is calculated using the hashes of all the transactions in the body of a block) and a nonce which is unique for each block. whereas the body contains the data or transactions in the form of a Merkle

tree, which uses a hash binary tree to store the data or transactions within a specific period of time. The Hash of the current block is calculated using a cryptographic encryption technique whose input is the previous block's hash and the data of the current block. Hence, blockchains are immutable since altering one block requires altering all subsequent blocks in order to maintain consistency. Data is stored in the form of a Merkle tree makes it easy to prove the integrity and existence of data or transactions. In this sequence, the previous block of any block is termed as parent block. Whereas the first block is called a genesis block which does not have a parent block.

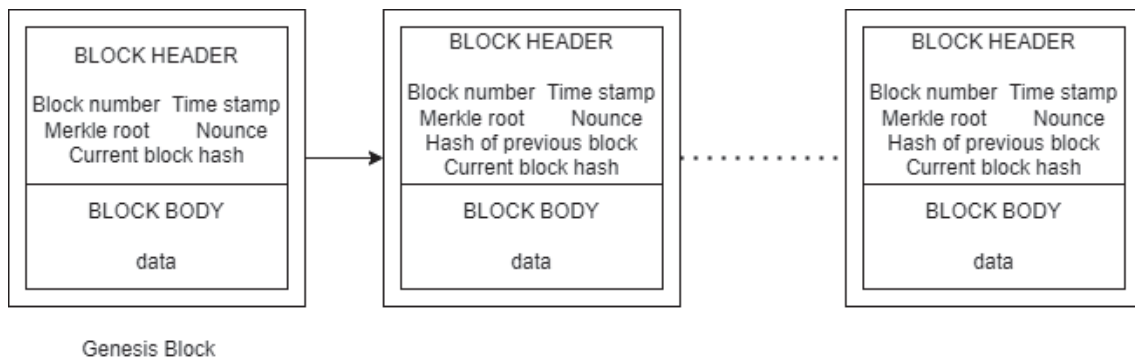


Figure 3.1: Structure of a block

3.1.2 Types of blockchain

At first, the blockchain was permissionless as the use case was dedicated to bitcoin. Later on to meet the growing requirements many other types have been introduced. There are three main blockchain architectures divided based on two things: 1) whether the access for reading the information stored on the blockchain is public or private, and 2) whether the right to write to the ledger and participate in consensus protocol execution is permissioned or permissionless. [41]

- **Public permissionless blockchain:** In this architecture, anyone can join the network and participate in the consensus process. Every node carries a copy of the ledger and the transactions are visible to all the nodes, although participants maintain a certain level of privacy. Blockchain and Ethereum are two main examples of this architecture. [41]
- **Public permissioned blockchain:** In this architecture, anyone can join the network who gets the right to propose new transactions, and read and verify the state of the ledger. Whereas only the authorized nodes can participate in the consensus process and can write new transactions and add new blocks. This type of architecture was made popular by the Sovrin Foundation in their blockchain-based identity management system implementation. [41]

- Private permissioned blockchain: In this architecture, a governing node (or a set of nodes) will have the authority to decide who can join the network and which participant can have the right to participate in the consensus process. The governing node can be also represented by a regulatory authority that issues private blockchain participation licenses and helps to sign business agreements between participating stakeholders to carry out the consensus process. Hyperledger Fabric, which is developed by the Linux Foundation, is a representative of a private permissioned blockchain system. [41]

3.1.3 Blockchain infrastructure model

In the following, an introduction to blockchain from the perspective of layers and architecture is discussed which has been briefly described in [43]. Blockchain infrastructure is mainly divided into six layers. These six layers along with some technologies and processes used in the respective layers are shown in fig.3.2. Following the figure, is the description of each of the layers.

| |
|--|
| APPLICATION LAYER Distributed resource marketplace Internet of things Smart grid |
| CONTRACT LAYER Smart contract Script code |
| INCENTIVE LAYER Issuance mechanism Allocation mechanism |
| CONSENSUS LAYER PoW PoS DPoS Ripple |
| NETWORK LAYER Verification mechanism Communication mechanism |
| DATA LAYER Block Chain structure Merkle tree Digital signature Hash algo. |

Figure 3.2: Blockchain infrastructure model

The data layer is the first layer in the blockchain infrastructure model. This layer represents a chain of blocks. A detailed introduction to the structure of the block is described in the above section. Some of the core technologies like the hash algorithm and Merkle tree are used in this layer which adds up to some of the important and fundamental features of blockchain. The Hash of the previous block is used to calculate the hash of the current block. Hence blocks are connected in a form of chains. Any change of data would lead to a new hash, which makes the block to be disconnected from the next blocks; indicating that tampering with data is done. Hence the data is tamper-proof and secure.

The network layer is the second layer in blockchain infrastructure composed of distributed networking mechanism, communication mechanism, and data verification

mechanism. In simple words, it can be said that the goal of this layer is to distribute, forward, and verify blockchain transactions. Blockchains function similarly to P2P networks i.e., the data or transactions generated are broadcasted to all neighbouring nodes. Each node will verify the received transaction according to predefined specifications. If the transaction is valid, it will be forwarded to other nodes. Otherwise, it will be discarded. In this way, only valid transactions are stored by every node in the blockchain network. To verify the data, digital signatures based on asymmetric cryptography are generally used. Every node contains a public key and a private key. Whenever data is generated, it is signed using the node's private key and then this data is shared with other neighbouring nodes which use the public key of the creator node to verify the authenticity of the data.

Next comes the consensus layer which contains different consensus mechanisms. Unlike centralized systems, decentralized systems do not have a trusted centralized node. Hence there is a need for a consensus mechanism in the blockchain that brings integrity and trust among all the nodes of the system. These mechanisms ensure consensus among all decentralized nodes before a block is included in the blockchain. Some of the main examples of consensus mechanisms are Proof of Work (PoW), and Proof of Stake (PoS). Bitcoin blockchain network uses PoW, where the miners repeatedly run hashing functions to generate a nonce value which is difficult to produce and needs a lot of computing power to validate the data and add the block. In order to reduce the computing power, an energy-saving mechanism PoS was developed, which enables the miner with the largest amount of stake (e.g., currency) to generate blocks. Many other consensus mechanisms have been developed and are used based on the requirements of the blockchain system.

The next layer is the incentive layer. Portions of cryptocurrency are rewarded to the nodes and miners as an incentive for using their computing power for the data verification process. Once a new block is generated, some economic incentives (e.g., digital currencies) will be issued as a reward and allocated to corresponding nodes according to their contributions. These incentives can be spent in the network or exchanged for fiat currencies. The miners can unite and form mining pools, where large computational "farms" are used to mine large amounts of cryptocurrency. [41]

The contract layer is the next layer which contains smart contracts. A smart contract was termed as "a computerized transaction protocol that executes the terms of a contract. The general objectives of smart contract design are to satisfy common contractual conditions (such as payment terms, liens, confidentiality, and even enforcement), minimize exceptions both malicious and accidental, and minimize the need for trusted intermediaries." by Nick Szabo. [45] When all terms within a smart contract are agreed upon by two or more participants, the contract will be signed cryptographically and broadcast to the blockchain network for verification. Once the predefined conditions are triggered, the smart contract will execute independently and automatically according to the prescribed rules.

The highest and last layer in the blockchain architecture is the application layer, which is composed of business applications, such as the Internet of Things, telecommunication systems, energy trading, digital identity, and many more. These applications can provide new services and perform business management and optimization.

3.2 Crosschain communication

3.2.1 What is crosschain communication?

crosschain communication refers to the exchange of information between two or more blockchains without any third-party entities. This technology facilitates the transfer of packets between two or more blockchains. Crosschain communications are mainly motivated by two requirements common in distributed systems: accessing data and accessing functionality that is available in other systems. Similar to how blockchains are designed to be BFT, cross-blockchain systems also need to be BFT and not rely on single parties for trust so that they can tolerate node failures, network failures, and malicious actors. In the context of crosschain communication, crosschain consensus relates to how participants on one blockchain are convinced of the state of a remote blockchain. It describes how parties associated with a source blockchain come to an agreement and communicate with a destination blockchain such that information from the source blockchain can be trusted. crosschain communication protocols ensure that crosschain consensus and different trust assumptions based on the type of blockchains are met. [37]

3.2.2 Why is crosschain communication needed?

To answer why crosschain communication is needed, we have to look into the issues being faced in blockchain systems. According to the concept of scalability trilemma described by Vitalik Buterin, the co-founder of Ethereum, blockchains usually have a trade-off between three important properties: scalability, decentralization, and security. Decentralization is the core and the nature of the blockchain. Security is an essential property, whereas scalability is the main challenge. The scalability trilemma states that trade-offs are almost inevitable among these characteristics of blockchain. [30] In other words, Scaling blockchains without decreasing the level of decentralization and security is one of the major issues. Scalability issue arises with the increase in the nodes and transactions in the blockchain. This problem arises because every node has to store and execute a computational task to validate every transaction. [30]. This major issue can be solved by blockchain interoperability or crosschain communication.

Blockchains are usually like isolated islands and are unable to communicate with or verify information on other blockchains. crosschain communication also addresses this issue and opens a new world of applications and platforms. Many experts argue that interoperable blockchains could supercharge industries like healthcare, law, or real estate, for instance by allowing important business information to be sent back and forth between private networks and public networks in a customizable and controllable manner. Interoperability across multiple blockchains presents an even more advanced embodiment of blockchain technology's promise to decentralize systems and economies. [39]

crosschain communication also enhances cross-industry collaboration. As blockchain technology has a wide range of industry-specific use cases, the collaboration between different industries and organizations can lead to many business applications. Once these blockchains are able to communicate with one another, independent markets

and business applications that were previously considered entirely separate will be able to transfer data and value. This means organizations and communities that would not typically interact with one another would be able to exchange information, leverage each other's strengths, and cultivate innovation more effortlessly and effectively. [39]

3.2.3 Crosschain communication mechanisms and protocols

The three main properties that crosschain communication protocols should satisfy are safety, liveness, and atomicity. The safety property captures the notion that bad states are not reachable. The liveness property captures the notion that good states are eventually reached. This property ensures that assets across blockchains are not locked forever. This usually happens when the protocols are based on asynchronous communications. As such, the liveness property is only achievable if the underlying communications protocol is synchronous. [37]

Atomicity is the combination of safety and liveness properties. For atomicity in the context of crosschain communications, the safety property translates to when the protocol finishes, the state updates on all blockchains involved in the crosschain transaction are either committed or rolled back. The liveness property states that the atomic crosschain transaction protocol eventually after a finite amount of time terminates. [37]

From a technical perspective, all the crosschain communication protocols and schemes can be mainly categorized into three types which are described in detail as follows:

Notary Mechanism

A notary mechanism is the simplest way in which crosschain communication can take place. In a notary mechanism, a trusted entity or set of entities that are trusted as a group is used in order to claim to chain X that a given event on chain Y took place, or that a particular claim about chain Y is true. Such entities may be active, listening, and automatically acting based on events in some chain, or reactive, issuing signed messages only when asked. [22]. These trusted entities are referred to as notaries. Notaries could be determined individually for a particular exchange; one can imagine a negotiation protocol where all participants submit their trust list and the intersection of all parties' trust lists is agreed upon as the notary set for that exchange. This is one of the ways in which notaries are set up, but the main motive here is that the notaries are decided or agreed upon by everyone [22]. The main advantages of this mechanism are that it is very easy to implement and has the flexibility to support different types of blockchain systems. However, there are some conflicts with the concept of decentralization and security when dealing with this mechanism. [44]

Interledger is one of the protocols following the notary mechanism developed by ripple. Interledger allows two different blockchain systems to transfer information and values to each other through a notary referred to as "connectors". These connectors enable value transfer between users from different blockchains. In Interledger,

firstly a user X in chain A transfers the assets to the notary (connectors). Then the notary will lock and confirm the assets and transfer the corresponding assets to user Y on chain B. The operation of the notary mechanism is simple, and there is no need for complex efforts to prove [44].

Interledger also envisions the notion of a payment chain where this notary mechanism can be composed. If parties X and Y desire to make an exchange of digital asset bundles, but these bundles exist on chains A and F, where A and F have no direct link, then one can find intermediaries on intermediate ledgers B, C, D, and E, where each pair of adjacent intermediate ledgers does have a direct link (i.e. there exist notaries and exchange opportunities between A and B, B and C, etc), then one can determine a bundle of exchanges that will satisfy A and F's preferences while at the same time earning small arbitrage revenues for the intermediaries, and then use a single consensus process for the entire exchange, ensuring that either all transfers happen or none do. This is how the protocol follows the atomic property. It uses the BFT consensus algorithm in order to achieve consensus among a set of notaries on whether or not a given event took place, and then issues a signature that can be used to finalize payments conditional on this consensus [22].

The core of this protocol is Interledger's transport protocol ILPv4. Senders communicate via Connector nodes to Receivers. Senders send request messages called Prepare packets and Receivers to respond with response messages called Fulfil packets or error messages called Reject packets. Consensus is achieved by the Prepare messages containing a hash and the Fulfil messages containing the corresponding preimage. The Prepare message provides a commitment by the Sender to pay. The Fulfil message provides acknowledgement by the Receiver that they have received the funds. None of the Connectors knows the preimage, and hence cannot forge the payment receipt [37].

Hash-locking Mechanism

Hash-locking mechanism is one of the easiest to implement and requires the blockchains to know much less about each other [22]. Hash locking first appeared in the solution of Bitcoin Lightning Network, which achieved fair transactions by locking assets and setting corresponding time and unlocking conditions. The basic procedure for cross-chain digital exchange is as follows with the description of the scenario [31].

Let us assume that user A on blockchain X wants to trade tokens with user B on blockchain Y.

- User A decides a secret value, say s , and computes a hash value V (i.e. $V = H(s)$, here H is the hash function) using the secret value.
- User A uses this calculated value V and sometimes says $t1$ to generate a transaction contract $TX1$ (i.e. $TX1 = T(V, t1)$) on chain A which locks the tokens that need to be traded for a lock time $t1$.
- User A shares the hash value along with the transaction contract $TX1$ to chain Y which proves that the tokens have already been locked for trading.

- Now user B uses this hash function shared by user A and time t_2 (t_2 should be such that, $t_2 < t_1$) to generate a transaction contract TX2 which locks the tokens that need to be traded for a lock time t_2 (ideal relation between t_1 and t_2 is $t_1 = 2t_2$).
- Now user A uses the value of s to unlock the transaction TX2 and obtains the tokens locked in the transaction TX2. At this time the secret number is exposed.
- User B uses this secret value to unlock the tokens in the transaction contract TX1.

The ideal value of t_2 is half of t_1 . Note that this mechanism is provably atomic. If A has never locked its tokens, B would not lock its tokens either. If A has done a mistake and revealed the secret value between t_1 and t_2 , then A would lose its tokens whereas B can get the tokens, but here it is A's fault. If A has never revealed the value of s or revealed it so late i.e. after t_1 seconds, then neither A nor B would receive the tokens and the trading would not happen. If B did not lock its tokens, then A would not reveal the value of s , and trading would not happen and thereafter A can recover their tokens [22].

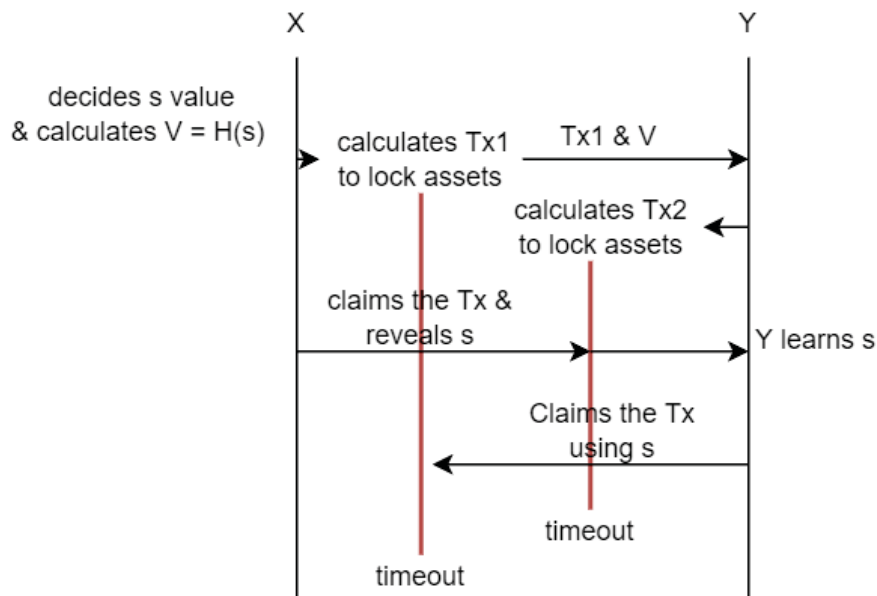


Figure 3.3: Hashlock Mechanism

As discussed in the procedure, a hash-locked atomic swap can ensure the security and atomicity of asset transactions between different chains without the participation of third parties.

However, when using this mechanism, the system can be subject to misuse. If the price of tokens deposited by user B has relatively fallen to the price of tokens deposited by user A, then user A can decide not to disclose the secret value, and the trading is not done. Whereas if the reverse happens user B can decide not to deposit the tokens into the contract and hence the trading does not happen. This

is termed as griefing attack (also known as a sore loser attack) means that user A or user A and user B do not have the utility to redeploy their tokens during the timeout period [37]. So in a world with exchange rate fluctuations, the financial safety guarantee is imperfect, as a malicious user A can wait for t_2 seconds and only publish if the exchange rate after that time moved in a favourable direction; by participating in this hash lock, B is effectively giving A a free option. [22]. Apart from griefing attacks, hash lock mechanisms are low efficiency, high payment cost, and heavy communication overhead. Due to this, some improved hash-locking mechanisms have been proposed [44].

Sidechain/Relay Mechanism

Sidechain is also known as child-chain which is an independent blockchain network relative to the main chain or the parent chain and connects to the main chain. These sidechains have their own consensus protocols. The sidechains can verify the block and ledger data of the main chain. These sidechains use a two-way pegging mechanism which allows their assets to be interchangeable and transferable between the ledgers of the main chain and the sidechain. A two-way peg feature implies that each blockchain ledger has a logical lockbox to make connections to the other (side chain or main chain) ledger possible [33].

Transferring assets to and fro between these chains follows the following steps (a pictorial representation is shown in the fig.) [33]:

- The assets which have to be sent to the side chain from the main chain are initially sent to the lockbox of the intended main chain with the address of the side chain.
- Then these assets are locked in the lockbox of the main chain and become unusable in the main chain.
- After a certain (predefined) time period waiting for confirmation, an equivalent amount of assets are released from the sidechain's lockbox to the sidechain. The assets are now usable on the sidechain.

The same procedure is followed when the assets have to be transferred from the side chain to the main chain. But in this case, the equivalent assets from the mainchain's lockbox are transferred to the main chain.

As mentioned above the sidechains use a pegging mechanism to transfer the assets between the sidechains and the mainchain. There are two types of pegging mechanisms - two-way pegging with simplified payment verification (SPV) and federated pegging.

The two-way pegging with the SPV mechanism uses SPV proof. SPV proof functions as the proof of possession of the assets in the original main chain for secure transfers to a side chain. An SPV proves that an asset is reserved on the main chain to be used on the side chain. SPV does all this by proving that the transaction made is part of a valid block, rather than checking all the previous transactions(which would

dramatically slow down the system). There are two waiting periods when SPV proof is used, the confirmation period and the contest period. The confirmation period is the period in which an asset must be locked on the parent chain before it can be transferred to a side chain. The consent period is the period in which a newly transferred asset should not be spent (or processed) on the side chain (to avoid double spending) [33].

On the other hand, Federated pegging requires a federation i.e., a bunch of nodes act as an intermediate point between the main chain and the side chain, and may also run both the ledgers. The federation resolves when a user's assets (or tokens) will be locked as well as when they will be released. A total of 51% or higher percentage of nodes should approve the transaction in order for the movement of the assets between the ledgers to be valid. The developers and admins of the side chains can assign members to the federation. The federation employs nodes on both the main chain and the side chain in order to keep track of the transfer of the assets [33].

Side chains can also be connected to other side chains with or without the intervention of the main chain, thereby increasing the functionality of the main chain. Through an efficient implementation of the side chain technology, blockchains can overcome existing limitations in the areas of security, privacy, and performance. Side chains can also be used by companies to hide parts of the data from their competitors while continuing to share the rest of the information. There are also some disadvantages as side chains are independent blockchain ledgers, they will be unsynchronized with the main chain, even though being connected to it. There are also other hardships like fraudulent transfers and shortage of miners [33].

An example following this mechanism is RootStock, a smart contract distributed platform built on the Bitcoin blockchain which aims to implement complex smart contracts as a side chain. Hence this increases functionalities for the core bitcoin network which is the main chain in this case. It mainly has four stages: submitting a locked transaction, waiting for a confirmation transaction, unlocking a transaction, and waiting for a competition period [44].

Unlike a side chain which is relative to the main chain, a Relay is a crosschain operation layer abstractly separated from each main chain. This will provide uniformity between the chains. [35] In this concept, the two chains provide the status of the data to the relayer. The data will be verified by the middleman. This way security risks can be highly reduced during the communication between the chains. The protocol following this relay mechanism is the inter-blockchain communication protocol which is used in the cosmos network. We have implemented the IBC protocol using the cosmos framework in our thesis. A detailed explanation of each of these technologies is explained in the next chapter.

This chapter gives a brief outline of the research methodology used in this thesis. And a brief introduction to different technologies used in the implementation of the crosschain communication protocol and its performance evaluation. Here we also discuss the traditional method used in the distributed digital marketplaces along with how crosschain communication can bring a change in these distributed marketplaces.

4.1 Overview on Applied Research Methodologies

In this section, we have discussed our research workflow. The compatible research method used for the thesis is literature search, experiment, and quantitative research. A literature search has been conducted to study crosschain communication and find answers to the research questions formulated. After the literature search, technology identification and study has been done. The procedure and study to select a system model for the implementation are mentioned in System modelling chapter 5. After implementation quantitative research has been done to collect output parameters and for statistical analysis of the results shown in Results and Discussion chapter 6. After the statistical analysis of the results, we conducted a research method to find out the reason for the output behaviour observed in the results. This is explained in Results and Discussion chapter 6. Originally when this thesis was proposed, the research methodology planned was following a spiral model. But as the thesis progressed we followed a waterfall method. The research workflow we have used is mentioned in figure 4.1. The large box in the figure contains five steps, where the maximum of our time has been spent on the thesis. [29]

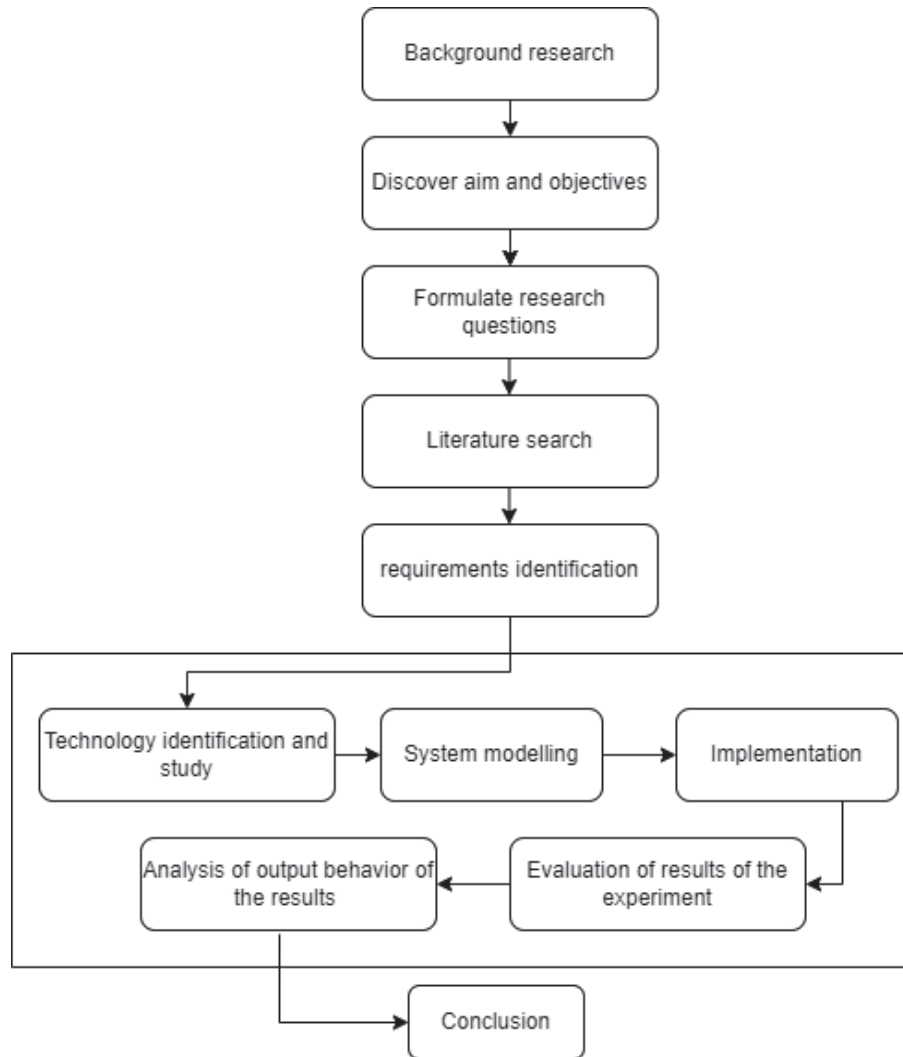


Figure 4.1: Research workflow

4.2 Distributed digital marketplaces

Digital marketplaces are a common and widely accepted concept for the formation of business opportunities. They are open platforms where IT companies or individual developers can offer their products for purchase. These marketplaces provide a place for the producers and consumers thus creating more opportunities and meeting supply and demand requirements. They allow products to be supplied to customers with increased speed and stimulate the popularity and expansion of the software. [41] To overcome the disadvantages faced by the centralized digital marketplaces and mainly to eliminate the need for central or third-party platforms between the producers and consumers, distributed digital marketplaces were started. Due to the decentralized feature of blockchain technology, many industries have proposed blockchain-based digital marketplaces. By using blockchain technology in digital marketplaces, we can eliminate the need for a third-party entity, protect from a single point of failure and bring transparency and flexibility between all the participants. The applica-

tion of blockchain-based digital marketplaces in different areas has been discussed in detail in [41].

The application of blockchain in the area of peer-to-peer (P2P) energy trading has gained traction in recent years. Blockchain technology reduces distributed energy producers' and consumers' dependence on the energy supplier during the trading process, enables P2P energy trading using smart contracts as a tool for trade settlement, and brings flexibility and transparency to all actors involved in the energy market. However, another important reason for the implementation of blockchain in marketplaces is the reduction of the need for a third party, e.g., a marketplace, in an electricity trading process. The ledger as data storage provides statistical information which helps marketplace maintainers to dynamically adjust electricity prices. This enables making trading more efficient in relation to energy consumption and production rates and also possibly allows to reduce of electricity bills for households. [41] Cross-blockchain communication or crosschain communication is the technology used for communication between individual blockchains in blockchain-based distributed marketplaces.

Here in this thesis, we are mainly concentrating on the application of crosschain communication protocol in a distributed digital marketplace. A pictorial representation of where crosschain communication is used in a distributed digital marketplace is shown in fig.4.2.

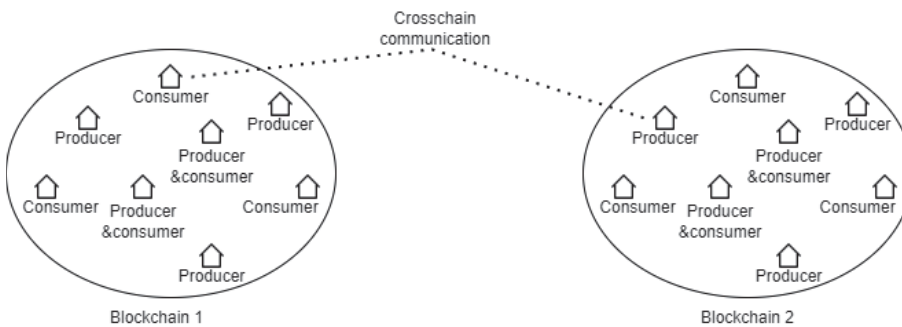


Figure 4.2: Distributed marketplace

So as seen in fig.4.2, there are many independent blockchains. Let these independent blockchains represent different areas of a city. These independent blockchains contain different nodes which are consumers as well as producers of energy. Communication between nodes on different independent blockchains is possible through the crosschain communication protocol. In the further sections, a detailed description of crosschain communication and protocols is given.

4.3 Cosmos

Cosmos is a decentralized network of independent blockchains that are powered by algorithms like Tendermint, a BFT consensus algorithm. The main aim of the cosmos is to build an ecosystem of blockchains that can scale individually and can easily communicate with other blockchains in the network in a decentralized way. And this

vision of the cosmos was achieved using open-source tools like Tendermint, Cosmos SDK, and IBC.

Tendermint

The infrastructure of the blockchain discussed in the previous section 3.1.3 can be summed up as - the application layer, consensus layer, and networking layer. Tendermint is an open-source platform that packages the networking and consensus layer into a generic engine, thus making the job of developers easy. This also helps the developers focus more on the application layer rather than the lower-level layers. Tendermint engine uses BFT (byzantine fault-tolerant) as the consensus algorithm, hence the name Tendermint BFT. More information about Tendermint can be found in [17].

Cosmos SDK

The Cosmos SDK is an open-source framework for building public Proof-of-Stake (PoS) and Proof-of-Authority (PoA) blockchains. Blockchains built with the Cosmos SDK are generally referred to as application-specific blockchains. Cosmos SDK helps developers to easily create custom blockchains from scratch on top of the tendermint. [4] Cosmos SDK contains an ecosystem of modules(modules that define the logic of cosmos SDK applications) that are built focusing on a particular functionality. Any user can easily import these modules and integrate them with other modules to use them for their application. These modules can also be developed and customized by developers for their own application-specific blockchains.

Cosmos SDK also provides different developer tools to build command line interfaces, REST servers, and other utility libraries.

IBC

IBC protocol is the crosschain communication protocol used by blockchains in the cosmos to interact with each other. More information regarding the IBC protocol will be given in the further section of this chapter.

4.3.1 Architecture

It can be seen in fig.4.3 that there are three layers - application layer, consensus layer, and network layer. As it is mentioned above, Tendermint BFT provides the consensus and network layer. And there is an interface between the consensus layer and the application layer known as the Application Blockchain Interface (ABCI). ABCI is a socket protocol. For the application layer cosmos SDK is used.

4.3.2 Cosmos Network

A modular network is provided by the cosmos which mainly contains two blockchain classes - Zones and Hubs. There is a reason why the cosmos provides this network. In the cosmos, all the blockchains can be connected with each other by IBC as mentioned above. During this process, we can end up making many connections for

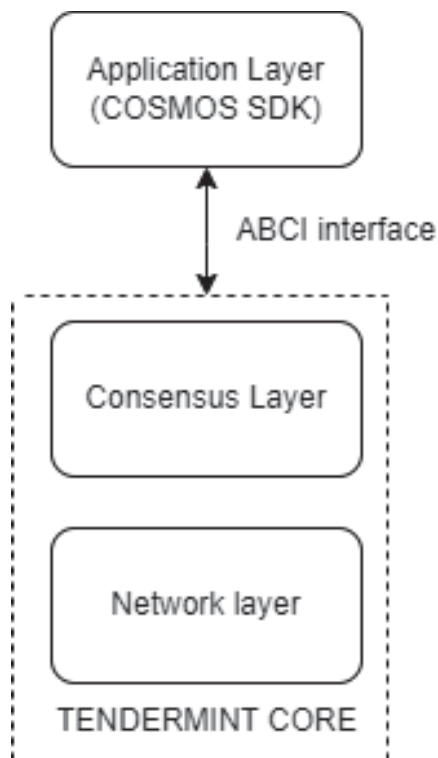


Figure 4.3: Architecture

each node, which would be difficult to be maintained. Hence cosmos came up with the solution of creating an ecosystem.

Zones are the heterogenous blockchains and hubs are the special blockchains created for the purpose of connecting these zones. Once a zone connects with a hub using IBC, it automatically gets connected to all the zones connected to that hub. And then, a zone will send and receive packets to and fro from all the zones connected to that hub. In this way, the cosmos provides a solution to limit the number of interconnections between blockchains. Hubs also help in avoiding double spending among zones.

Cosmos Hub is the first hub launched in the cosmos network. Cosmos Hub is a public blockchain following Proof-of-Stake with the native staking token known as ATOM. Following is the figure representing the cosmos network: Hubs and Zones.

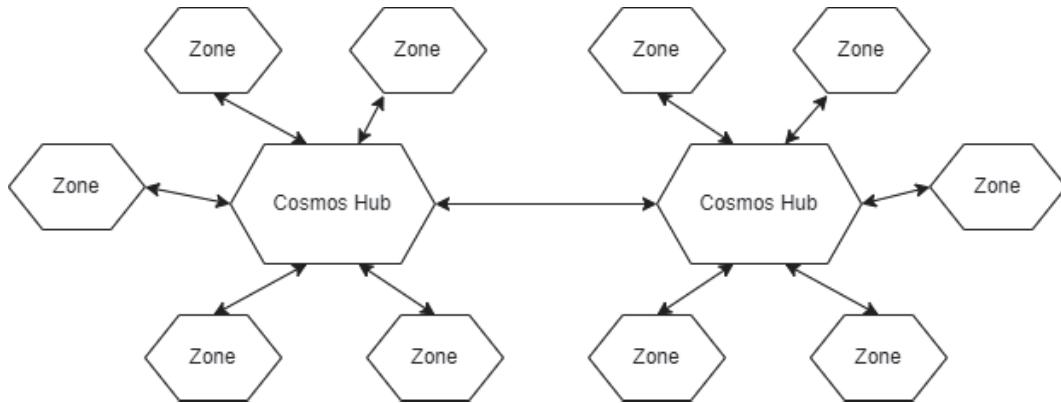


Figure 4.4: Cosmos SDK ecosystem

The Cosmos network together with all these open-source technologies forms the cosmos ecosystem. For more information and learning of the Cosmos, refer to different sections in [3] and [2].

4.4 Inter-Blockchain communication protocol

Inter-Blockchain communication protocol (IBC) is a crosschain communication protocol that handles authentication, transport, and ordering of data between two blockchains. IBC requires a minimal set of functions that are specified in the Interchain Standards (ICS). These specifications do not limit the network topology or consensus algorithm. [19] In other words, IBC presumes nothing about and requires nothing of the overall network topology, and of the nodes require only a minimal set of functions with specified properties that are available. [27] So IBC can be used with a wide range of blockchains. It can be used by any application which builds on top of reliable & secure inter-module communication. Example applications include crosschain asset transfer, atomic swaps, multi-chain smart contracts (with or without mutually comprehensible VMs), and data & code sharding of various kinds. [23]

To facilitate this blockchain interoperability, the IBC protocol utilizes a bottom-up approach inspired by the TCP/IP protocol. Hence IBC is also known as the TCP/IP for the blockchains. Just as TCP/IP packets contain opaque payload data with semantics interpreted by the processes on each host, IBC packets contain opaque payload data with semantics interpreted by the modules on each ledger. Just as TCP/IP provides reliable, ordered data transmission between processes, allowing a process on one host to reason about the state of a process on another, IBC provides reliable, ordered data transmission between modules, allowing a module on one ledger to reason about the state of a module on another. [27]

4.4.1 IBC protocol structure

Figure 4.5 shows different components of the IBC protocol. [19]

As shown in figure 4.5, IBC protocol contains two layers: a transport layer and an application layer, and a relay that sits in between the chains for the communication to happen. The transport layer (TAO) provides the necessary infrastructure

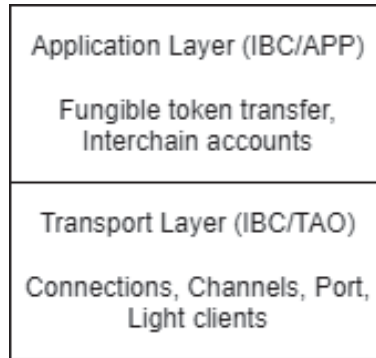


Figure 4.5: IBC protocol structure

to establish secure connections and authenticate data packets between chains. On top of the transport layer, an application layer is built which defines exactly how data packets should be packaged and interpreted by the sending and receiving chains based on different applications like fungible token transfers, NFT transfers, interchain accounts, and many more. Relay algorithms are the "physical" connection layer of IBC: off-chain processes responsible for relaying data between two chains running the IBC protocol by scanning the state of each chain, constructing appropriate data packets, and executing them on the opposite chain as is allowed by the protocol. Many relayers can serve one or more channels to send messages between the chains. [19]

IBC clients are known as light clients that are identified by a unique client ID. IBC clients track the consensus state of other blockchains and the proof specs of those blockchains that are required to properly verify proofs against the client's consensus state. The packets, acknowledgements, and timeouts that off-chain relayers send back and forth can be verified by proving that the packet commitments exist inside of these clients on each chain. A client can be associated with any number of connections to the counterparty chain. [19]

To connect two blockchains with IBC, an IBC connection should be made. IBC connections are established by on-chain ledger code and therefore do not require interaction with off-chain (trusted) third-party processes. Connections once established, are responsible for facilitating all crosschain verifications of an IBC state. Connections are established by a four-way handshake: OpenInit, OpenTry, OpenAck, and OpenConfirm. This opening handshake protocol is responsible for establishing the identity of the counterparty chain and preventing a malicious entity from forging incorrect information by pretending to be the counterparty chain. Connections encapsulate two ConnectionEnd objects on two separate blockchains. Each ConnectionEnd is associated with a light client of the other blockchain - for example, the counterparty blockchain. [19]

Connections are associated with channels and a connection can be associated with any number of channels. Through these channels, a module on one blockchain can communicate with other modules on other blockchains by sending, receiving, and acknowledging packets. Channels provide a way to have different types of information relayed between chains but do not increase the total capacity. Similar to connections, channels are established with a handshake. A channel can be ordered, where packets

from a sending module must be processed by the receiving module in the order they were sent, or unordered, where packets from a sending module are processed in the order they arrive (which might be different from the order they were sent). These channels are bound to ports and these ports have a unique port ID based on the type of the application. [19]

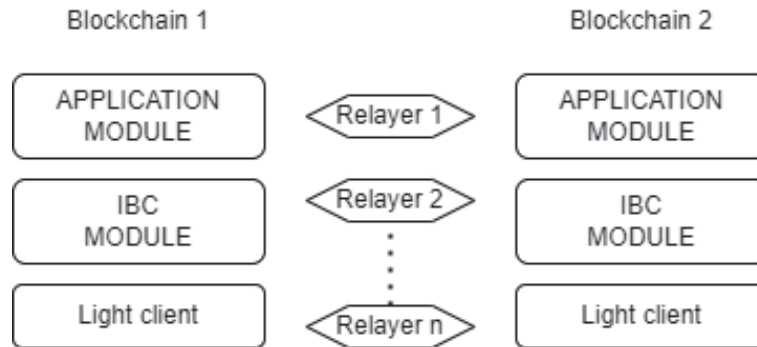


Figure 4.6: IBC module structure

For a detailed description of the protocol and the abstractions defined by the IBC protocol along with the mechanisms by which they are composed, [27] should be referred. Whereas an in-depth description of different layers in the protocol structure along with an introduction to different handshake protocols, functions, and applications of the IBC protocol [19] should be referred.

4.4.2 Security of IBC

The design of IBC security is centred around two main principles: [19]

- Trust in (the consensus of) the chains you connect with.
- The implementation of fault isolation mechanisms, in order to limit any damage done should these chains be subject to malicious behaviour.

When using the IBC protocol, the verification of packet commitment proofs is provided by the light client. The light client is able to track and efficiently verify the state of the counterparty blockchain, to check commitment proofs for the sending and receiving of packets on the source and destination chains respectively. This is important because it ensures that IBC protocol remains secure even in Byzantine environments where relayers could act in a malicious or faulty manner. Relayers need not be trusted; instead, trust the proof verification provided by the light client. In the worst-case situation where all relayers are acting in a Byzantine fashion, the packets sent would get rejected because they do not have the correct proof. [19]

IBC clients and transactions assume the trust model of the chains they are connected to. In order to represent this accurately in assets that have been passed through the Interchain, the information on the path that an asset has travelled (the security guarantee of the asset) is stored in the denomination of the asset itself. In the case that the end-user or an application does not trust a specific origin chain, they would be able to verify that their asset has not come from the untrusted chain

simply by looking at the denomination of the asset, rather than referring to the validator set of a bridge or some other trusted third party verifier. [19]

4.4.3 Why IBC protocol?

The IBC protocol is providing a reliable, permissionless, and generic base layer while allowing for composability and modularity with separation of concerns by moving application designs to a higher level layer. The protocol design enables safe, reliable interoperation of a network of heterogeneous distributed blockchains, arranged in an unknown topology, preserving data secrecy where possible, where the ledgers can diversify, develop, and rearrange independently of each other or of a particular imposed topology or ledger design. IBC protocol also ensures security and unlike many trusted bridge solutions, IBC does not depend on any intermediary to verify the validity of crosschain transactions. As mentioned earlier, the verification of packet commitment proofs is provided by the light client. So the security of the chains or the network is considered the security of the bridge. [19] [27]

4.5 Azure

Microsoft Azure often referred to as Azure is a cloud computing service operated by Microsoft for application management via Microsoft-managed data centres. It provides software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) and supports many different programming languages, tools, and frameworks, including both Microsoft-specific and third-party software and systems. [12]

In Microsoft Azure, a virtual machine (Infrastructure as a service) has been used to deploy and run the blockchains.

4.6 Matlab

MATLAB is a proprietary multi-paradigm programming language and numeric computing environment developed by MathWorks. [11] Matlab programming language has been used in this thesis to calculate the performance metrics(mentioned in the next chapter) from the output data obtained during implementation.

4.7 Statistics

The statistical concepts used in this thesis are described in this section.

4.7.1 Random Variable

A random variable X is a measurable function

$$X: \Omega \rightarrow E$$

from a set of possible outcomes Ω to a measurable space E . [16]

4.7.2 Probability distribution

A probability distribution is a mathematical function that gives the probabilities of the occurrence of different possible outcomes for an experiment. If X is a discrete random variable and $P(x)$ is the probability of x , the function given by $f(x) = P(X = x)$ for each x within the range of X is called the probability distribution of X . [15]

Exponential Distribution

The exponential distribution is the probability distribution of the time between events in a Poisson point process, i.e., a process in which events occur independently at a constant average rate. This process is used to model for a series of discrete events where the average time between events is known, but the exact timing of events is random. The best example of where this process is used is in queuing theory to model random events, such as the arrival of customers at a store, phone calls at an exchange or the occurrence of earthquakes, distributed in time. [14] [13]

The probability density function (PDF) of an exponential distribution is

$$f(x; \lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

Here $\lambda > 0$ is the parameter of the distribution, often called the rate parameter. The distribution is supported on the interval $[0, \text{Inf})$. [6]

4.7.3 Auto-correlation

Correlation defines the statistical relationship between two random variables.

Whereas auto-correlation defines the correlation of a random variable or a signal with a delayed copy of itself as a function of delay. [1] Auto-correlation helps to find repeating patterns in a signal. Hence we can find if a random variable is following a pattern or is random by calculating the auto-correlation of the probability distribution function of the random variable. The range of auto-correlation values is from -1 to +1. A value between 0 to -1 represents a negative correlation, i.e., if a variable increases, to what extent the other variable decreases, or vice-versa. Whereas when the value is between 0 to 1 it represents a positive correlation, i.e., to what extent variable increases or decreases when the other one increases or decreases.

This chapter discusses how we design and model inter-blockchain communication and gather the results for performance measurement. Here we explain the structure of the system, the different components making up the system, and how we modelled our system in accordance with how practical communication happens between two entities and also strain the system to see how it can perform under different conditions.

5.1 Requirements

In order to establish inter blockchains communications, there are essential prerequisites that need to be met. The following components are necessary to enable inter block-chains communication and measure its performance:

- Two blockchains with nodes that can communicate.
- A relayer to interlink the two block-chains and transfer messages between block-chains.
- A tool to collect different metrics to be used in performance measurement.

5.2 Models description

This section discusses different models that were designed to approach our problem and find solutions. These models were designed to test the applicability of the technology in line with the tools at our disposal. Two system models are described and evaluated on how they both fit the purpose to demonstrate the applicability of inter-blockchain communication protocol. The suggested model presents a chain of events and actual behaviour when two systems are communicating.

Both proposed models demonstrate how blockchains can communicate with each other and also we developed a way to collect performance metrics which will later be used to evaluate the performance of the system.

System Model

The design of the fundamental model is to have two blockchains that communicate with each other. The source blockchain has to send messages to the destination blockchain.

To begin the process, two separate blockchains are initialized. In each blockchain, there is a node that is going to be communicating with another node in the opposite blockchain. Both blockchains are having an IBC module. After starting the blockchains, a relay is also initialized and it is going to create a channel that connects the two blockchains. The channel interlinks the two blockchains using their IBC modules. A node in the source blockchain creates a packet which is then placed in the IBC module of its respective chain. Once the relay detects there is a packet in the module, it gathers information about the destination from the packet and then uses the respective channel connecting the source blockchain and the destination blockchain, to send/relay the packet to the destination. This process is automated by using a bash script.

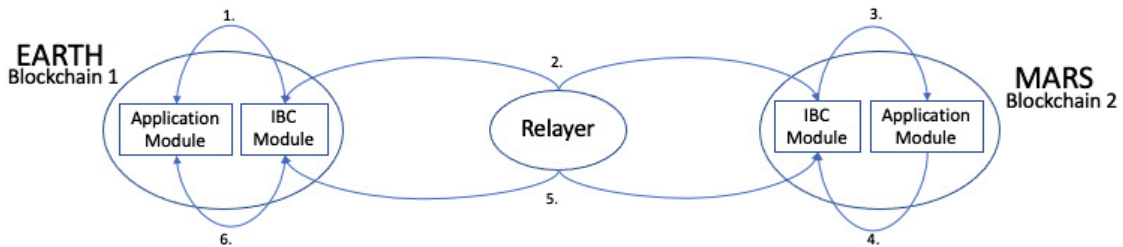


Figure 5.1: System Model

The communication flow of the model is as follows:

1. Application module sends the packet to the IBC module of the blockchain.
2. The relay transfers the packet from the IBC module of blockchain 1 to the IBC module of blockchain 2.
3. IBC module receives the packet and sends it to the application module of the destination blockchain.
4. When the packet is received by the application module of the destination blockchain, the application module sends the acknowledgement to the IBC module of the blockchain.
5. Relay transfers acknowledgement from the IBC module of blockchain 1 to the IBC module of blockchain 2.
6. IBC module receives the acknowledgement and sends it to the application module of the blockchain.

Performance metrics

From this basic model, the following performance metrics can be measured; [34]

- Rate: this is the average number of packets received per unit of time. (Number of packets received/Total simulation time)

- Packet delivery ratio. (Number of packets received/Number of packets sent)
- End-to-end delay or round trip time. This is the total time from the generation of the packet on the source blockchain, sending the packet from the source blockchain to the destination blockchain, and receiving back the acknowledgment from the destination blockchain.

Implementation model

This is the practical model which is implemented in our tests. The basic model is modified into two models to simulate different communications scenarios.

5.2.1 Deterministic model

In this model, the packet-sending events are controlled by a fixed time interval between them. The time interval is in seconds. The blockchain is an application running under a Linux operating system, hence it is automated by using a bash script.

At the beginning of the simulation, a user is asked to input a number of packets to be sent to the destination blockchain. The bash script then initiates a process to send a packet and waits for a predefined time interval before sending another packet. This process will run until all packets in accordance with the number given by the user at the beginning of the simulation have been sent.

5.2.2 Random model

This model is designed to simulate communication as a random process. In this model, the packet-sending events are following a Poisson process with an average rate already defined and the interval times between these events are following an exponential distribution. As the packet-sending activity involves a series of random events, they are defined in a calendar which is generated using Matlab. The calendar is made up of random values following an exponential distribution. The generated random values are the interval times between two packets being sent. These values are random or in other words completely independent of each other.

At the beginning of the simulation, a user is asked to input a number of packets to be sent to the destination blockchain. The bash script then initiates a process to send a packet and waits for a time interval defined in the calendar before sending another packet. This process will run until all packets in accordance with the number given by the user at the beginning of the simulation have been sent.

5.3 Model selection

All the two models discussed above will be used in our experiments to study the performance of the IBC protocol. No model is preferred over the other one. The deterministic model is used to observe the performance of the IBC protocol under a predictable environment where the arrival of packets to be sent is regulated under fixed time intervals. The random model is used to observe the performance of the

IBC protocol under an unpredictable environment where packets are sent at random time intervals.

A pre-implementation has been done before the actual implementation to determine the different parameters like how many experiments will be implemented in each model, how many packets have to be sent, and what is the rate of the Poisson process in the random model. In the pre-implementation, 100 packets have been sent from the source blockchain to the destination blockchain, and calculated the mean round trip time, the mean was 7.5 seconds.

Four experiments have been implemented for both models: the deterministic and random models. In the case of the deterministic model, we have run three experiments and in the case of the random model, one experiment was run. In each experiment, 1000 packets were sent from the source blockchain to the destination blockchain. The minimum sample size should not be less than 30 to calculate the statistic parameters. [24] Hence 1000 packets are sent. In the three experiments run the interval time between the packets was 1 second, 5 seconds, and 10 seconds. These values were decided based on the mean round trip times in the pre-implementation. In 1st experiment, the interval time is 1 second which is very less when compared to the mean value. In 2nd experiment, the interval time is 5 seconds which is near and less than the mean value. In 3rd experiment, the interval time is 10 seconds which is greater than the mean value. We wanted to run these experiments to observe the behaviour of the system when the packets were sent in a very less interval time, medium interval time, and long interval time respective to the mean of the round trip times. As mentioned before in the case of the random model, the interval times which were following an exponential distribution were generated using Matlab. The average rate to generate these values equal to the mean value generated in the pre-implementation.

In this chapter, we discuss the practical implementation of crosschain communication protocol i.e., IBC protocol using cosmos blockchain framework according to the system model discussed in the previous chapter. Technologies and software used for this implementation are also mentioned. A detailed procedure of our implementation along with the input generation and calculation of output parameters is in the following sections.

6.1 Requirements

In this section, the virtual machines, technology, and software used in this implementation are discussed below.

Virtual Machines used

The following are the specifications of the virtual machine which was used for the implementation of the crosschain communication protocol.

- Cloud platform: Azure - Microsoft cloud provider
- Operating system: Linux Ubuntu 20.04
- Memory: 8 GB
- Processor: 2 vCPUs

Framework and CLI used

The cosmos ecosystem has been used to implement the blockchains for this thesis. A detailed explanation of the cosmos ecosystem has been done in 4.3. Different modules provided by the Cosmos SDK have been used. Ignite CLI is a command line interface built on top of Cosmos SDK for chain development by scaffolding different modules that are needed. More about Ignite CLI in [8].

- ignite CLI version used: v0.19.5
- ignite documentation: <https://docs.ignite.com/>
- Github source: <https://github.com/ignite/cli>

- Cosmos SDK version used: v0.46.5
- Cosmos SDK documentation: <https://docs.cosmos.network/main>
- Github source: <https://github.com/cosmos/cosmos-sdk>

6.2 Setting up the required applications

Building blockchains

Two different blockchains have been scaffolded in the virtual machine using the Ignite CLI and Cosmos SDK modules. Documentation of how a simple blockchain application can be built by importing SDK modules using the ignite CLI is mentioned below. The document contains how a blockchain named Hello is built. Similarly, following all the steps in the documentation, two blockchain applications named Earth and Mars are built which are used in this thesis in the process of implementation of crosschain communication. The source code for these two blockchains is the same but they have different blockchain IDs. The path of the directory in the virtual machine where the Earth and Mars blockchain applications are built is also mentioned.

- Documentation of blockchain app: `/home/user15/planet`
- path in virtual machine: <https://docs.ignite.com/guide/hello>

Building relay

After the two independent blockchains Earth and mars are built, an IBC module should be implemented. Cosmos SDK contains all the necessary logic for IBC in a module. When a module is scaffolded with an IBC flag, all the logic for IBC will be implemented in that module of the blockchain application. There are many actions like creating and sending a packet, creating and sending an acknowledgement upon receiving the packet, storing and listing the packets sent, and storing the packets that have not been sent because of timeout. These action points can be scaffolded in the already scaffolded IBC module. Cosmos SDK provides a base for these action points. Using this base with some modifications and added code, we can customize it accordingly for our application. For this thesis, the same approach is followed and some code modifications have been done to implement the action points mentioned above. Also defined are the input and output arguments of a packet and an acknowledgement. [9] can be followed for a detailed guide and more information about the scaffolding IBC module along with different action points that can be done in that module.

6.3 Preparing the input

In this thesis we are implementing four experiments for two models: the deterministic model and the random model.

In the case of the deterministic model, we have run three experiments and in the case of the random model, one experiment was run. In each experiment, 1000 packets were sent from one blockchain to other blockchains. These packets were sent with a pre-determined or random time in between them based on the model and experiment. In this thesis, this time calculated in between sending the packets is known as interval times.

In the deterministic model, for each experiment, fixed interval times of packets have been set. For the first experiment, a short interval time of packets has been set which is one second. For the second experiment, a medium interval time of packets has been set which is five seconds. For the third experiment, a long interval time of packets has been set which is ten seconds.

In the random model, only one experiment was run where the interval times between the packets were random and followed an exponential distribution. These random interval times of packets were generated using Matlab. The randomness of the interval times of packets is calculated using auto-correlation. The code in Matlab to generate the times is as follows:

```

1 clc;
2 clear all;
3 i = 0;
4 a = [];
5 lambda = 7.5;
6 while i<=1000
7     rng('shuffle');
8     a = [a, exprnd(lambda)];
9     i = i + 1;
10 end
11
12 b = a(:); %interpacket times which we have to copy.
13
14 [corr, lag] = autocorr(a); %autocorrelation to find out if the data
    is random
15 autocorr = corr(2);
16 figure;
17 stem(lag, corr);
18
19 r = exprnd(lambda,1000,1);
20
21 [bins,p]=acNhist2(a, 1); %plotting the pdf of the data generated
22
23 pd = fitdist(b,'Exponential');
24
25
26 h = kstest2(b,r)

```

The values which are generated from Matlab were exported to a text file and were given as input to the bash script which will be explained in the next section. The above code generates 1000 random values following an exponential distribution. Along with the generation, the autocorrelation value is also calculated to check the randomness of the values. And also checks if the values are really following an exponential distribution by Kolmogorov–Smirnov test. Matlab provides a Statistics toolbox that contains a pre-defined function `kstest2` which is used here.

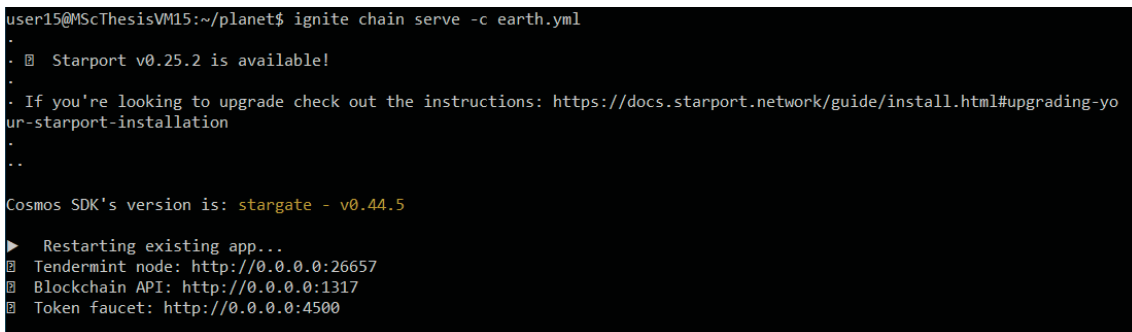
6.4 Implementing the setup

In this section, we have mentioned all the steps that we have done for the implementation of the IBC protocol in a detailed manner from start to end.

1. SSH into the virtual machine and change the directory to the path where the blockchain application has been built.
2. Start the Earth blockchain that has been set up which is discussed in the previous sections. The command to start the Earth blockchain is as follows:

```
1 ignite chain serve -c earth.yml
2
```

The flag `c` specifies the config file. When this command is run the output is as shown in the figure.



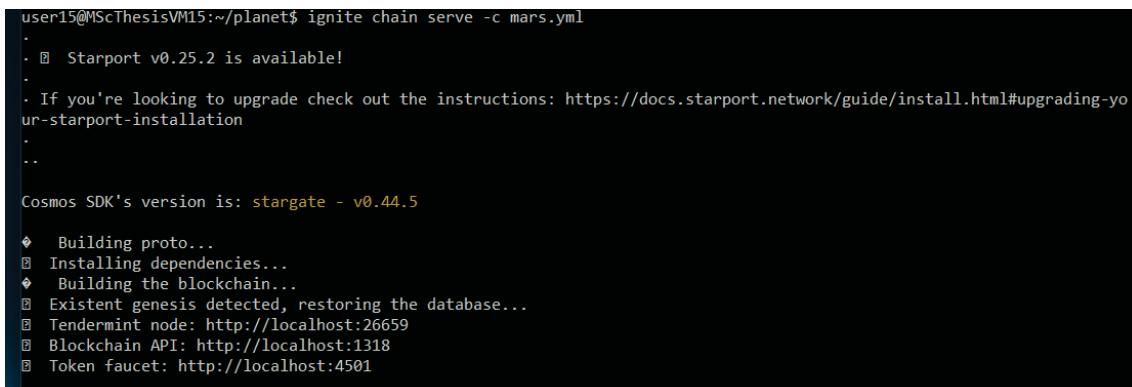
```
user15@MScThesisVM15:~/planet$ ignite chain serve -c earth.yml
.
. [ ] Starport v0.25.2 is available!
.
. If you're looking to upgrade check out the instructions: https://docs.starport.network/guide/install.html#upgrading-your-starport-installation
.
.
.
Cosmos SDK's version is: stargate - v0.44.5
▶ Restarting existing app...
[ ] Tendermint node: http://0.0.0.0:26657
[ ] Blockchain API: http://0.0.0.0:1317
[ ] Token faucet: http://0.0.0.0:4500
```

Figure 6.1: output of starting source blockchain

3. Now in a new terminal, start the Mars blockchain. The command to start the blockchain is as follows:

```
1 ignite chain serve -c mars.yml
2
```

Output when starting the Mars blockchain is shown in the following figure.



```
user15@MScThesisVM15:~/planet$ ignite chain serve -c mars.yml
.
. [ ] Starport v0.25.2 is available!
.
. If you're looking to upgrade check out the instructions: https://docs.starport.network/guide/install.html#upgrading-your-starport-installation
.
.
.
Cosmos SDK's version is: stargate - v0.44.5
◆ Building proto...
[ ] Installing dependencies...
◆ Building the blockchain...
[ ] Existent genesis detected, restoring the database...
[ ] Tendermint node: http://localhost:26659
[ ] Blockchain API: http://localhost:1318
[ ] Token faucet: http://localhost:4501
```

Figure 6.2: output of starting destination blockchain

- Then in a new terminal, configure the relayer between the blockchains by giving the appropriate port numbers of the blockchain applications running. The command to start the relayer is as follows:

```

1  ignite relayer configure -a \
2  --source-rpc "http://0.0.0.0:26657" \
3  --source-faucet "http://0.0.0.0:4500" \
4  --source-port "blog" \
5  --source-version "blog-1" \
6  --source-gasprice "0.0000025stake" \
7  --source-prefix "cosmos" \
8  --source-gaslimit 300000 \
9  --target-rpc "http://0.0.0.0:26659" \
10 --target-faucet "http://0.0.0.0:4501" \
11 --target-port "blog" \
12 --target-version "blog-1" \
13 --target-gasprice "0.0000025stake" \
14 --target-prefix "cosmos" \
15 --target-gaslimit 300000
16

```

The output of the above command is shown in the following figure.

```

❏ Starport v0.25.2 is available!
.
If you're looking to upgrade check out the instructions: https://docs.starport.network/guide/install.html#upgrading-your-starport-installation
.
-----
Setting up chains
-----
? Source Account default
? Target Account default
❏ Account on "source" is default(cosmos19hyu80eahltpmcj7gw8al298zhxy6rvrngazwr)
|· received coins from a faucet
|· (balance: 1894698stake,95token)
❏ Account on "target" is default(cosmos19hyu80eahltpmcj7gw8al298zhxy6rvrngazwr)
|· received coins from a faucet
|· (balance: 2093409stake,105token)
❏ Configured chains: earth-mars

```

Figure 6.3: setting relayer output

- Now we have to start the relayer in the same terminal. The command to start the relayer is as follows:

```

1  ignite relayer connect
2

```

The output on the terminal is in the following figure.

From the figure, we observe that a channel has been set up between the chains and the relayer is up and listening.

- A bash script file to record the contents of this terminal is written and run which is used later for the performance evaluation. This bash script is written such that it starts the configured relayer, prints the date and time before each line in the output terminal, and saves this output into a file with a .log extension. The bash script file to do this is as follows:

```

user15@MScThesisVM15:~$ ignite relay connect
. Starport v0.25.2 is available!
. If you're looking to upgrade check out the instructions: https://docs.starport.network/guide/install.html#upgrading-your-starport-installation
.
-----
Paths
-----
earth-mars:
earth > (port: blog) (channel: channel-19)
mars > (port: blog) (channel: channel-20)
-----
Listening and relaying packets between chains...
-----

```

Figure 6.4: connect relayer output

```

1 #!/bin/bash
2
3 #Test4-3.log file contains the output
4 exec &> >(while read line; do echo "$(date +%s%3N) $line" >>
5     Test4-3.log; done;)
6
7 cd planet
8
9 ignite relay connect

```

- Now we have to send a message from one blockchain to other. The command to send a message is as follows.

```

1 planetd tx blog send-ibc-post blog channel-0 "Hello" "Hello
2     Mars, I m Alice from Earth" --from alice --chain-id earth
3     --home ~/planet/.earth -y

```

As can be seen in the command above, we have to specify the message that we have to send, from which node are we sending, the chain id, and the path where the blockchain has been started. The flag y here indicates signing the packet before sending it.

- To send messages in an automatic way, a bash script has been written. The bash script file to send packets in a deterministic way is as follows:

```

1 #!/bin/bash
2
3 echo "How many messages do you want to send"
4 read count
5 i=1
6
7 echo "Enter the time interval"
8 read interval
9
10 #Iteration to send messages as determined by input from user
11 while [ $i -le $count ]
12 do
13     planetd tx blog send-ibc-post blog channel-0 "Hello" "Hello
14         Mars, I'm Alice from Earth" \
15         --from alice --chain-id earth --home ~/planet/.earth -y
16     sleep $interval
17     i=$((i+1))

```

```
17 done
```

The bash script to send packets in an exponentially random way is as follows. This bash file takes input from the text file generated from the Matlab code mentioned in the previous section.

```
1 #!/bin/bash
2
3 echo "How many messages do you want to send"
4 read count
5
6 #Iteration to read the data file
7 while IFS= read -r line
8 do
9     s=$line
10    sleep $s
11    i=1
12
13    #Iteration to send messages as determined by input from user
14    while [ $i -le $count ]
15    do
16        planetd tx blog send-ibc-post blog channel-0 "Hello" "Hello
17            Mars, I'm Alice from Earth" \
18            --from alice --chain-id earth --home ~/planet/.earth -y
19        i=$((i+1))
20    done
21 done < file.txt #txt file contains interval times
```

9. We have to run two bash scripts in two different terminals at the same time, one to send the packets and the other to note down the times at which these packets are sent, and how many packets are being sent and received as mentioned in the previous steps. The output is a log file that is exported and used to calculate the performance metrics.

6.5 Collecting the output

The Matlab script mentioned below is used to calculate and measure all the performance metrics.

```
1 A = importdata("Test3.xlsx");
2 string_cell = A.textdata;
3 size_string_cell = size(string_cell);
4 size_row = size_string_cell(1);
5 value_packet = string([]); %packets
6 value_ack = string([]); %acks
7 individual_packet_time = [];
8 individual_ack_time = [];
9 noof_packets = [];
10 noof_acks = [];
11 packet_time = [];
12 ack_time = [];
13 start_index = 25;
14
```

```

15 for i = start_index:size_row
16     value = string(A.textdata{i,2});
17     %value1 = string(A.textdata{i,2});
18     time = A.data(i,1);
19     number = str2num(regexpi(value, '\d*', 'Match'));
20     if contains(value, "packets")
21         noof_packets(end+1) = number;
22         value_packet(end+1) = value;
23         packet_time(end+1) = time;
24         for j = 1:number
25             individual_packet_time(end+1) = time;
26         end
27     end
28     if contains(value, "acks")
29         value_ack(end+1) = value;
30         ack_time(end+1) = time;
31         noof_acks(end+1) = number;
32         for j = 1:number
33             individual_ack_time(end+1) = time;
34         end
35     end
36 end
37
38 value_packet = value_packet';
39 value_ack = value_ack';
40 packet_time = packet_time';
41 ack_time = ack_time';
42
43 %round_trip_time for each trip in seconds
44 diff_time = (ack_time - packet_time)/1000;
45 % individual round trip times for each packet in seconds
46 individual_diff_time = (individual_ack_time - individual_packet_time
47     )/1000;
48
49 %total simulation time from start to end of all packets in seconds
50 total_simulation_time = (ack_time(end) - packet_time(1))/1000;
51
52 %rate - average number of packets sent per time unit
53 rate = sum(noof_acks)/(total_simulation_time);
54 fprintf("rate = %.6f\n", rate);
55
56 %packet_delivery_ratio
57 packet_delivery_ratio = sum(noof_acks)/sum(noof_packets);
58 fprintf("Packet delivery ratio = %d\n", packet_delivery_ratio);
59
60 %mean and variance of round trip time or end to end delay
61 mean_time = mean(diff_time);
62 var_time = var(diff_time);
63 std_time = std(diff_time);
64
65 fprintf("mean round trip time = %.4f\n", mean_time);
66 fprintf("variance of round trip time = %.4f\n", var_time);
67 fprintf("standard deviation of round trip time = %.4f\n", std_time);
68 fprintf("Total simulation time = %.6f\n", total_simulation_time);
69
70 %graph of round trip times

```

```
70 figure;
71 bar(1:1000, individual_diff_time);
72 hold on;
73 plot(1:1000, ones(length(1:1000),1)*mean(individual_diff_time), 'r',
74      'LineWidth', 1)
75 ylim([5 9]);
76 yticks(5:1:9);
77 xlabel("packet number");
78 ylabel("individual round trip time");
79 %graph of number of packet sent at one trip time
80 figure;
81 bar(1:length(noof_packets), noof_packets);
82 ylim([0 4]);
83 yticks(0:1:4);
84 xlabel("Trip number");
85 ylabel("number of packets delivered in the trip");
```

In this chapter, we are discussing the results that we have observed from the implementation, and also we present the calculations and analysis of these results. These results were obtained from the experiments done on the two models: the deterministic model and the random model. Under the deterministic model, three experiments have been done for three different fixed interval times. In the random model, one experiment has been run with exponentially random interval times.

7.1 Experiment 1: Impact of short interval time

This experiment was done under the deterministic model. Here we are sending 1000 packets on a fixed interval time of 1 second. The aim is to test the performance of the IBC relay as it receives packets after a very short duration.

The following are the results obtained from this experiment:

| Parameters | Values |
|-----------------------------|------------------|
| Fixed interval time | 1 second |
| Number of packet sent | 1000 |
| Number of packets received | 1000 |
| Total simulation time | 2089.097 seconds |
| Rate | 0.478676 seconds |
| Packet delivery ratio | 1 |
| Mean of round trip time | 7.5376 seconds |
| Variance of round trip time | 0.2774 seconds |

Table 7.1: Results obtained in EXP1

In the analysis of this experiment, graphs have been plotted and presented as shown in figures, 7.1, 7.3, 7.4, 7.2. The following are the observations noted down from the values and graphs obtained from this experiment.

- There is a 100 per cent delivery ratio as we get acknowledgements of all the packets sent.
- Fig.7.1 represents the number of packets being sent at a time in each trip. As we can observe from the figure, the packets are being grouped before they are

sent from the relay. The minimum and the maximum number of packets being grouped in this experiment are 6 and 8.

- Fig. 7.2 represents the round trip time of all the packets in seconds. The red line represents the mean of all the round-trip times. we have observed from the figure that the mean is varying slightly throughout the whole simulation time. Hence, we have calculated the means of round trip times by splitting the data into two halves, i.e., from 1 to 500 packets represented in fig. 7.3 and from 500 to 1000 packets represented in fig. 7.4. All these mean values are recorded and presented in the table 7.5.

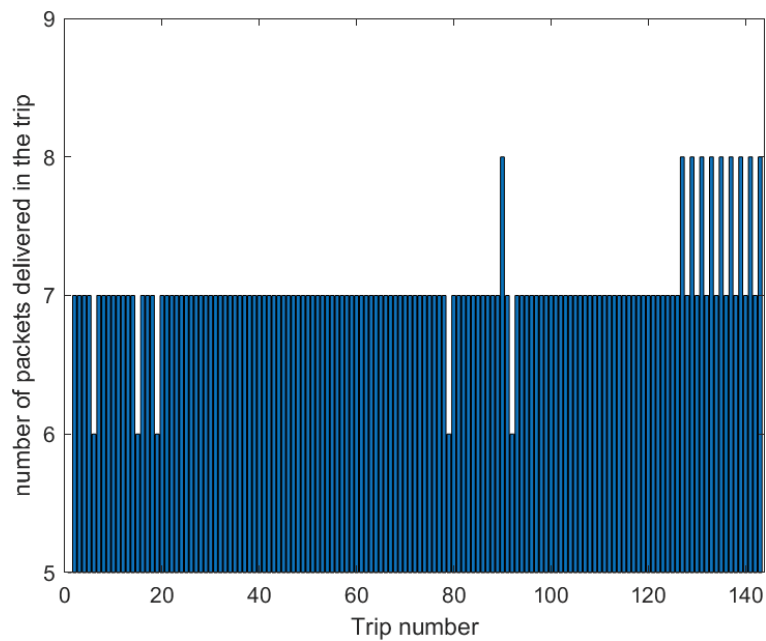


Figure 7.1: EXP1: Number of packets sent in each trip

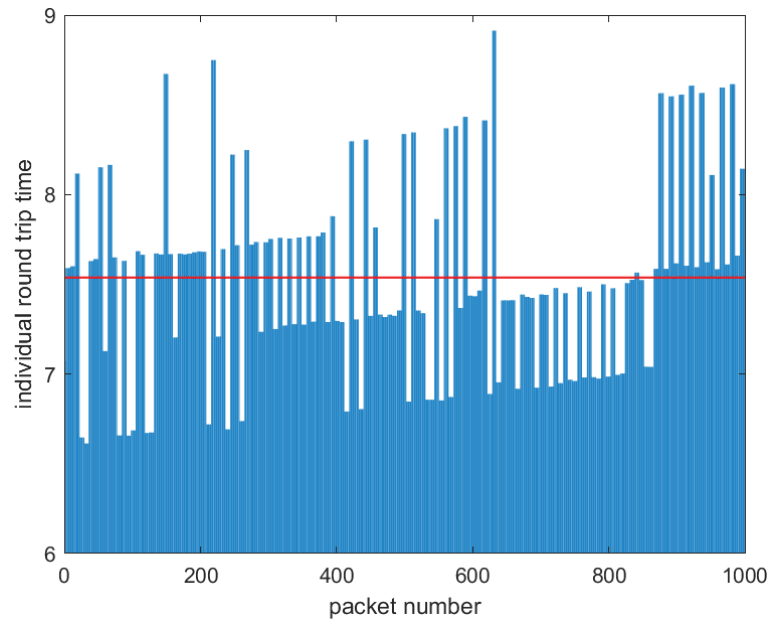


Figure 7.2: EXP1: Round trip time of each packet(seconds)

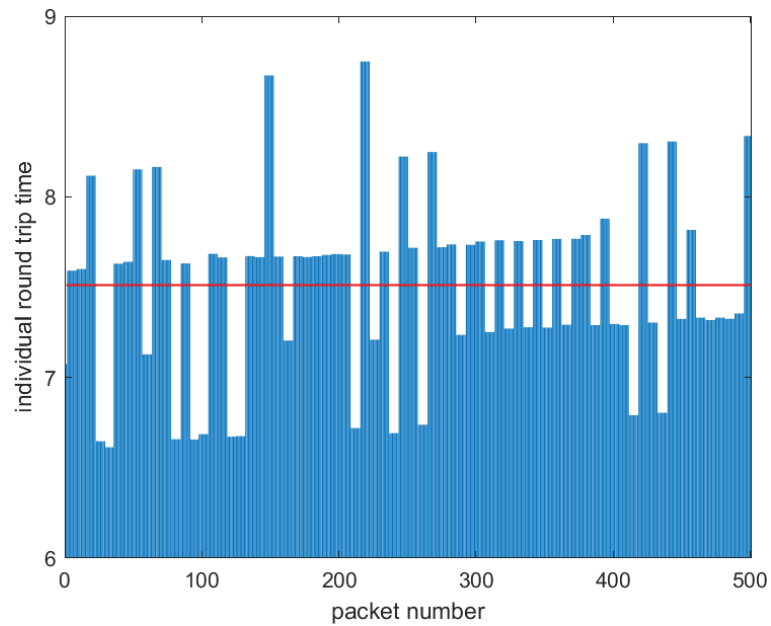


Figure 7.3: EXP1: Round trip time of packets from 1 to 500(seconds)

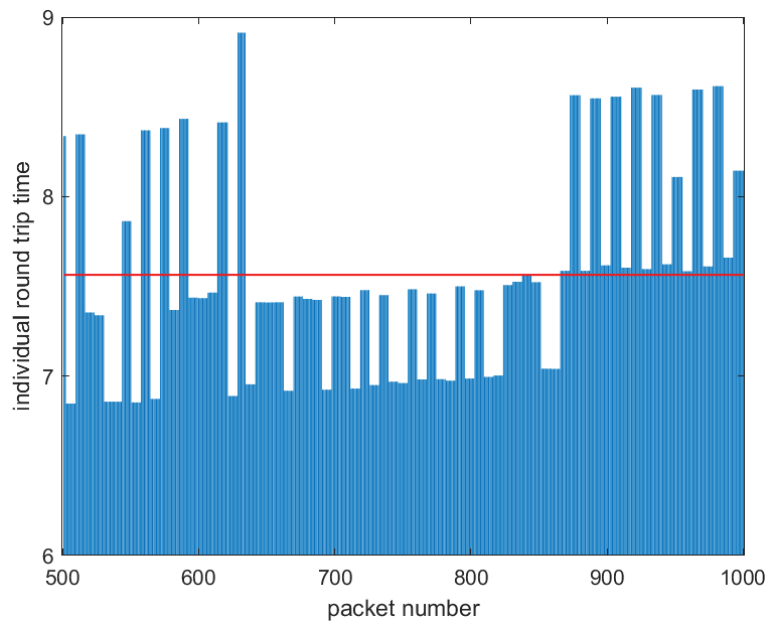


Figure 7.4: EXP1: Round trip time of the packets from 500 to 1000(seconds)

7.2 Experiment 2: Impact of medium interval time

This experiment was done under the deterministic model. Here we are sending 1000 packets on the same interval time of 5 seconds. The aim is to test the performance of the IBC relay as it receives packets after a medium time duration when compared with the average round-trip time of the previous experiment.

The following are the results obtained from this experiment:

| Parameters | Values |
|-----------------------------|------------------|
| Fixed interval time | 5 seconds |
| Number of packet sent | 1000 |
| Number of packets received | 1000 |
| Total simulation time | 5242.687 seconds |
| Rate | 0.190742 seconds |
| Packet delivery ratio | 1 |
| Mean of round trip time | 7.4211 seconds |
| Variance of round trip time | 0.2846 seconds |

Table 7.2: Results obtained in EXP2

Graphs plotted in this experiment are shown as figures, 7.5, 7.7, 7.8, 7.6. The following are the observations noted down from the values and graphs obtained from this experiment.

- There is a 100 per cent delivery ratio as we get acknowledgements of all the packets sent.
- Fig.7.5 represents the number of packets being sent at a time in each trip. As we can observe from the figure, the packets are being grouped before they are sent from the relay. The minimum and the maximum number of packets being grouped in this experiment are 2 and 3.
- Fig. 7.6 represents the round trip time of all the packets in seconds. The red line represents the mean of all these round-trip times. we have observed from the figure that the mean is varying along the whole simulation time. Hence, we have calculated the means of round trip times by making the whole data into two halves, i.e., from 1 to 500 represented in fig. 7.7 and from 500 to 1000 represented in fig. 7.8. All these mean values are represented in the table 7.5.

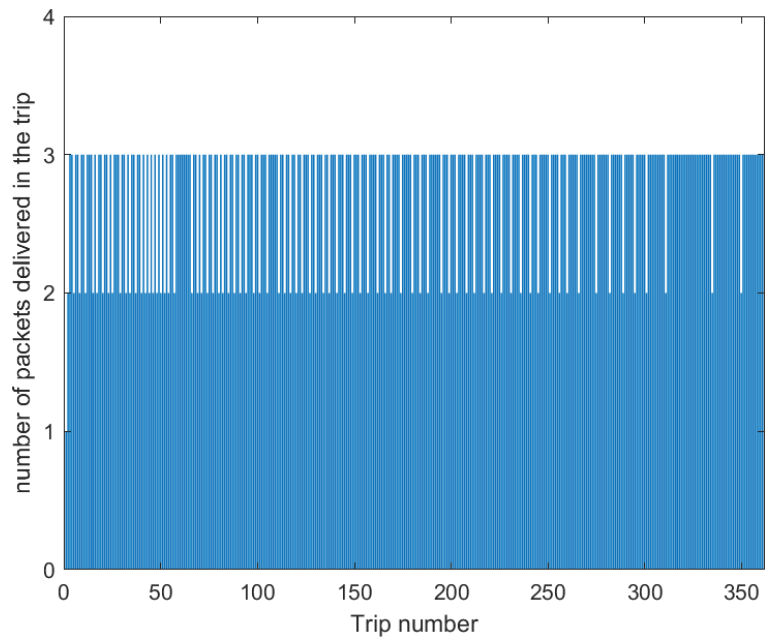


Figure 7.5: EXP2: Number of packets sent in each trip

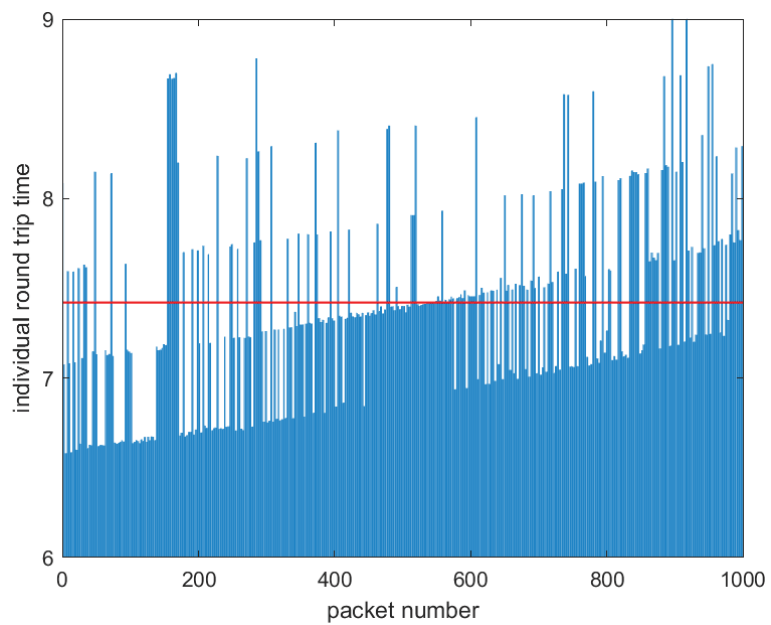


Figure 7.6: EXP2: Round trip time of each packet(seconds)

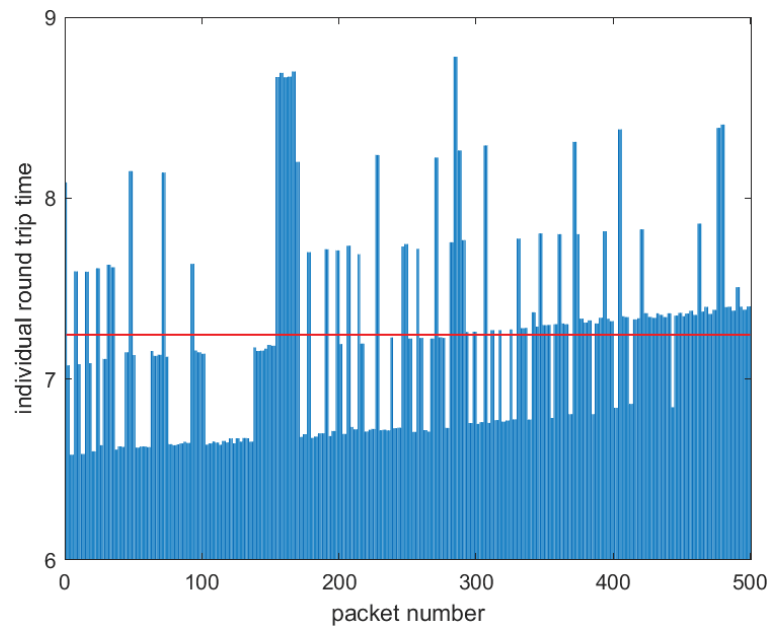


Figure 7.7: EXP2: Round trip time of packets from 1 to 500(seconds)

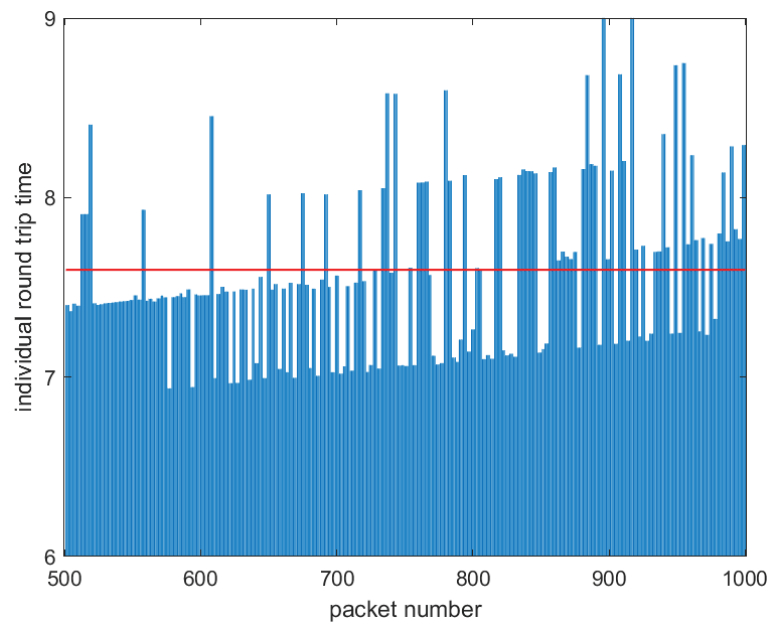


Figure 7.8: EXP2: Round trip time of the packets from 500 to 1000(seconds)

7.3 Experiment 3: Impact of long interval time

This experiment was done under the deterministic model. Here we are sending 1000 packets on the same interval time of 10 seconds. The aim is to test the performance of the IBC relay as it receives packets after a longer time duration compared with the average or mean round trip time of the first experiment.

The following are the results obtained from this experiment:

| Parameters | Values |
|-----------------------------|-------------------|
| Fixed interval time | 10 seconds |
| Number of packet sent | 1000 |
| Number of packets received | 1000 |
| Total simulation time | 10383.525 seconds |
| Rate | 0.096306 seconds |
| Packet delivery ratio | 1 |
| Mean of round trip time | 7.4801 seconds |
| Variance of round trip time | 0.2857 seconds |

Table 7.3: Results obtained in EXP3

A few graphs have been plotted in this experiment and are shown as figures, 7.9, 7.11, 7.12, 7.10. The following observations were noted from the values and graphs obtained from this experiment.

- There is a 100 per cent delivery ratio as we get acknowledgements of all the packets sent.
- Fig.7.9 represents the number of packets being sent at a time in each trip. As we can observe from the figure, the packets are being grouped before they are sent from the relay. The maximum number of packets being grouped in this experiment is 2.
- Fig. 7.10 represents the round trip time of all the packets in seconds. The red line represents the mean of all these round-trip times. we have observed from the figure that the mean is varying throughout the whole simulation time. Hence, we have calculated the means of round trip times by splitting the whole data into two halves, i.e., from 1 to 500 represented in fig. 7.11 and from 500 to 1000 represented in fig. 7.12. All these mean values are represented in the table 7.5.

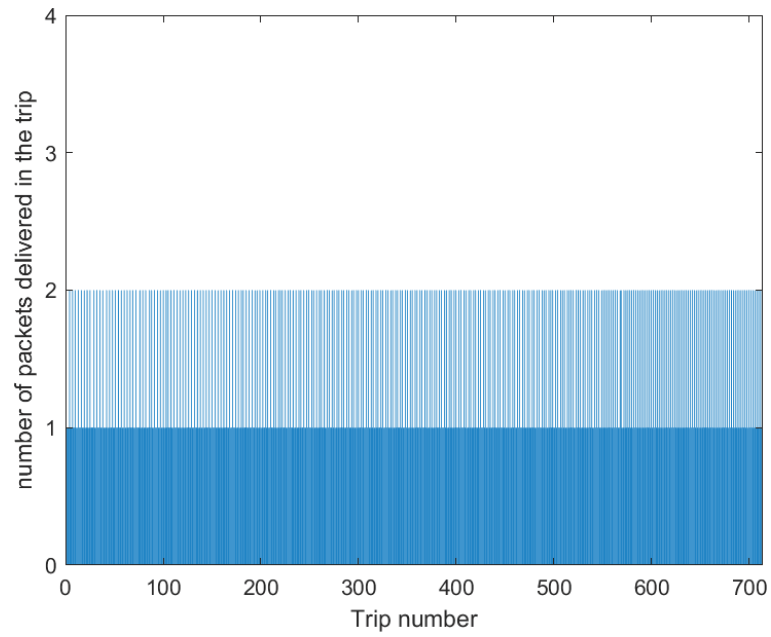


Figure 7.9: EXP3: Number of packets sent in each trip

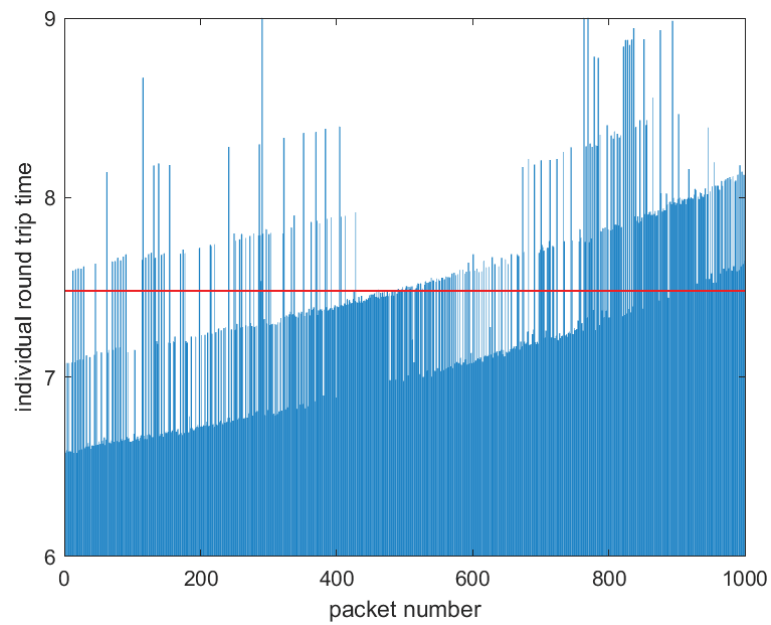


Figure 7.10: EXP3: Round trip time of each packet(seconds)

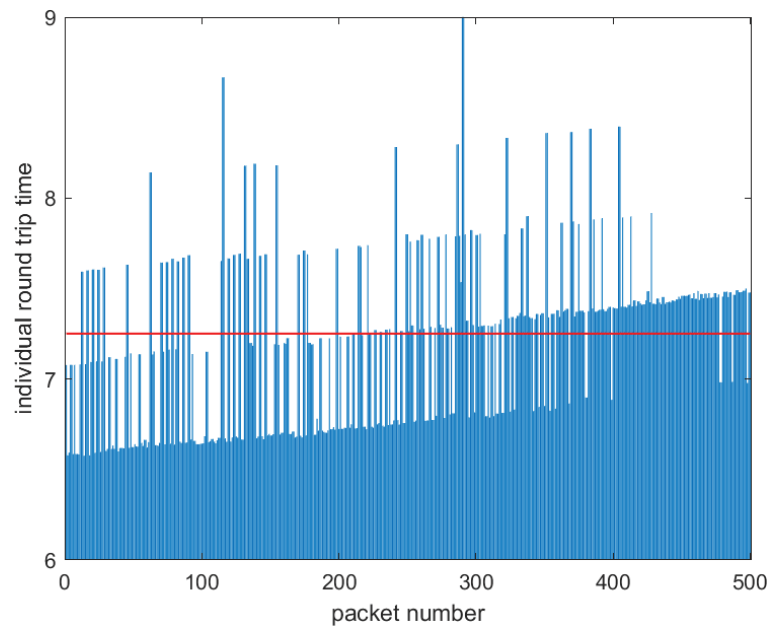


Figure 7.11: EXP3: Round trip time of packets from 1 to 500(seconds)

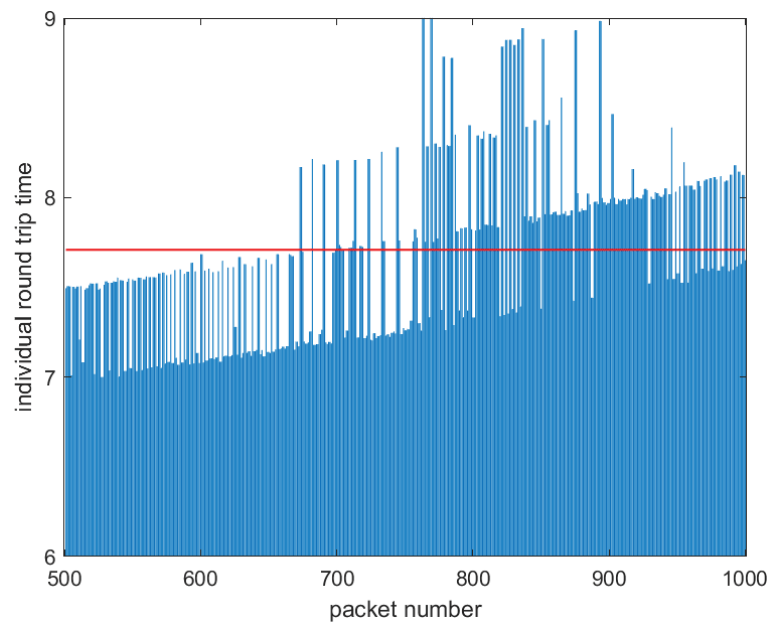


Figure 7.12: EXP3: Round trip time of the packets from 500 to 1000(seconds)

7.4 Experiment 4: Impact of random interval time

This experiment was done under the random model. Here 1000 packets were sent with interval times following an exponential distribution. These values are independent of each other which is random. The Matlab script used to generate these random interval times following an exponential distribution is given in the implementation chapter. Auto-correlation value for the generated values is 0.0055. As the value is very much less than 1, we can say that the generated interval times of packets are random. The probability distribution of these generated values is as shown in fig.7.13. And we can observe from the figure that the values are following an exponential distribution. To confirm that these values are following an exponential distribution, Kolmogorov–Smirnov test has been conducted using the `kstest2` built-in function in Matlab. The result of the `kstest2` is 0 indicating that the null hypothesis is accepted, that is the data is following an exponential distribution. So after these tests, we have verified that these interval times are random and follow an exponential distribution.

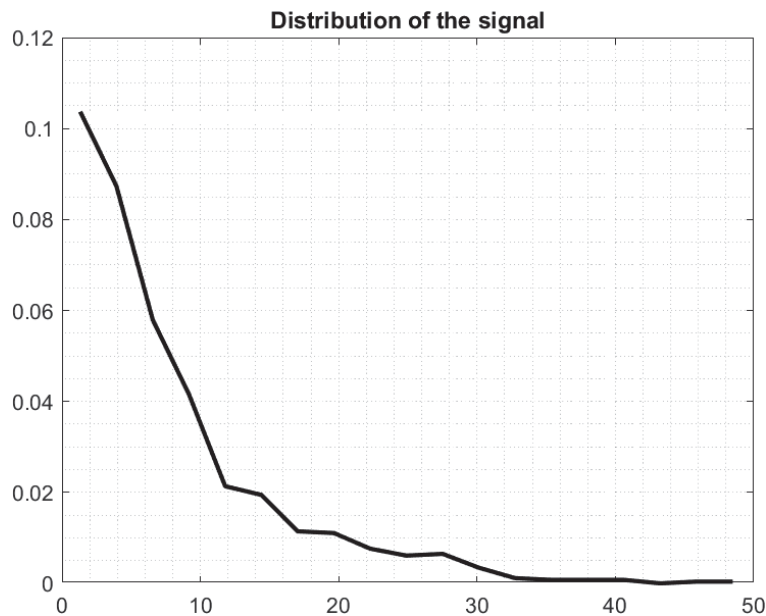


Figure 7.13: The round trip time of packets

The aim of this experiment is to observe the performance of the IBC relay as it receives packets in a random manner, which is usually the case in real life. The graphs plotted for this experiment are represented as figures below 7.15, 7.16, 7.17. Table 7.4 represents the results observed in this experiment.

| Parameters | Values |
|-----------------------------|------------------|
| Number of packet sent | 1000 |
| Number of packets received | 1000 |
| Total simulation time | 8324.795 seconds |
| Rate | 0.120123 seconds |
| Packet delivery ratio | 1 |
| Mean of round trip time | 7.4392 seconds |
| Variance of round trip time | 0.2123 seconds |

Table 7.4: Results obtained in random model

The following are observations made from the graphs shown in the figures below.

- There is a 100 per cent delivery ratio as we get acknowledgements of all the packets sent.
- Fig.7.14 represents the number of packets being sent at a time in each trip. As we can observe from the figure, the packets are grouped before they are sent from the relay. The maximum and the minimum number of packets being grouped in this experiment are 2 and 6 respectively.
- Fig. 7.15 represents the round trip time of all the packets in seconds. The red line represents the mean of all these round-trip times. we have observed from the figure that the mean is varying throughout the whole simulation time. Hence, we have calculated the means of round trip times by splitting the whole data into two halves, i.e., from 1 to 500 packets represented in fig. 7.11 and from 500 to 1000 packets represented in fig. 7.12. All these mean values are represented in the table 7.5.

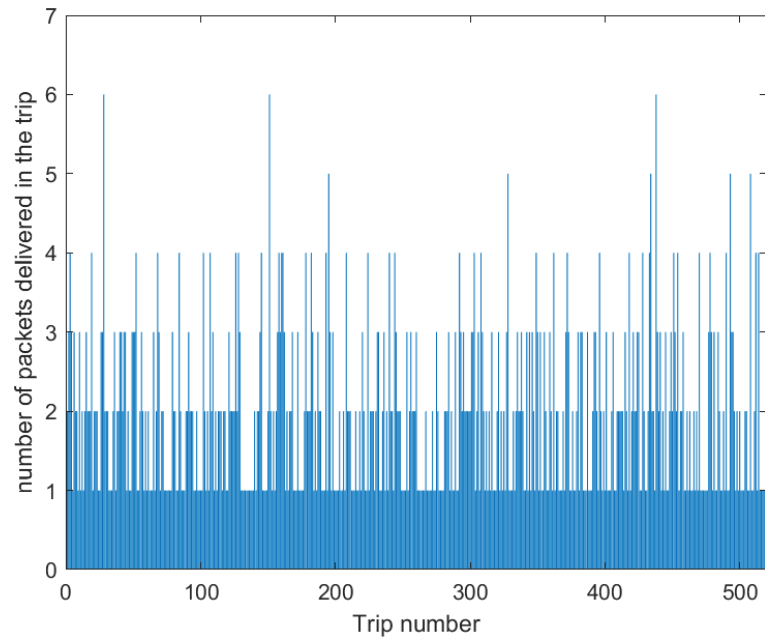


Figure 7.14: EXP4: Round trip time of each packet

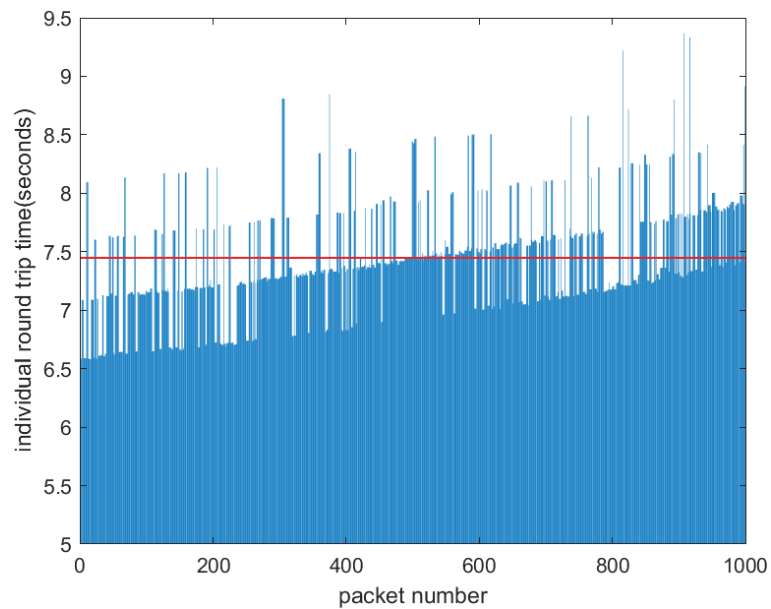


Figure 7.15: EXP4: Round trip time of each packet(seconds)

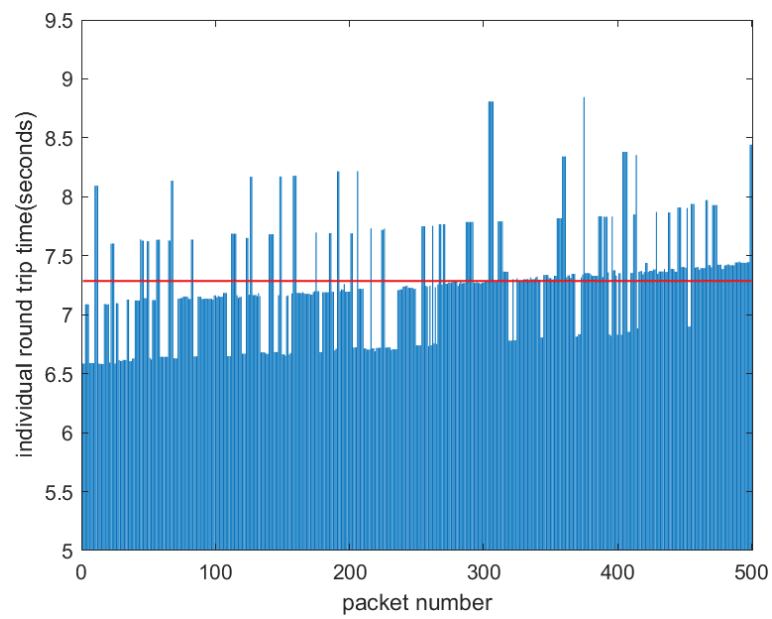


Figure 7.16: EXP4: Round trip time of each packet from 1 to 500(seconds)

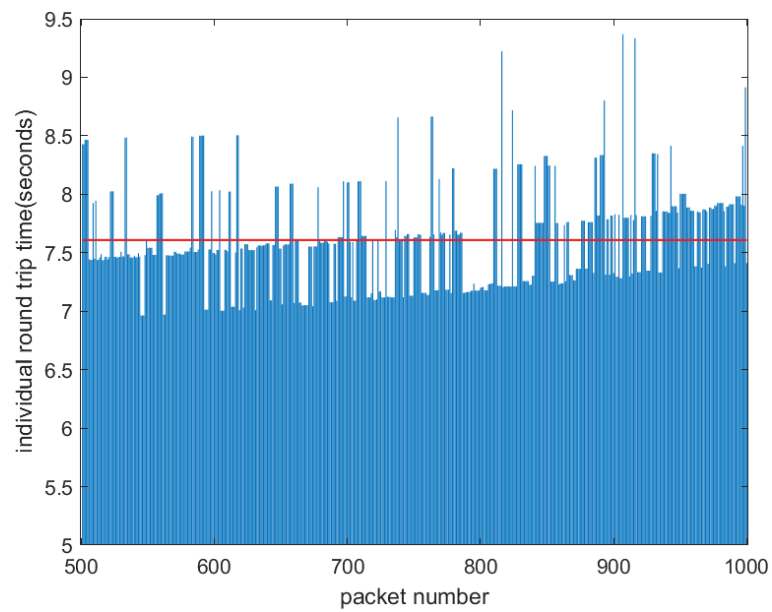


Figure 7.17: EXP4: Round trip time of packets from 500 to 1000(seconds)

7.5 Observation

As we have already seen in all the experiments the packet delivery ratio is 1 or in other words, the packet delivery ratio is 100 per cent. So we can confirm that there is no packet loss.

The figures, 7.1, 7.5, 7.9 and 7.14 represent the number of packets being sent from one blockchain to another blockchain in one trip along the whole simulation time for all the experiments. In all these experiments we are sending a total of 1000 packets from the source blockchain to the destination blockchain with a pre-defined time interval between the packets being sent. From the graphs, we are observing that packets are being transferred or relayed in groups as they are being sent from the source blockchain to the destination blockchain. This grouping behaviour has been observed in all the experiments that we have conducted. In Experiment 1, 7.1, the maximum number of packets that have been grouped is 8 and the value keeps varying between 6 and 8. The most repeating number when grouping the packets in this experiment is 7. In Experiment 2, 7.5, the maximum number of packets being grouped is 3 and this value keeps varying between 2 and 3. And the most repeating number while grouping the packets in this experiment is 2. In Experiment 3, 7.9, the maximum number of packets that are being grouped before they are sent is 2. In this experiment, we have observed that the number of packets being grouped is the least when compared with the other experiments that have been conducted in this thesis. In experiment 3, the most repeating number while sending the packets in this experiment is 1. In Experiment 4, 7.14, the maximum number of packets being grouped is 6 and this value keeps varying between 1 and 6. And the most repeating number while grouping the packets in this experiment is 1. In this random case, we have also observed many fluctuations in the number of packets being grouped when compared to other experiments.

From these values, we can say that in experiment 1, the maximum number of grouping has been done, while in experiment 3, the least number of grouping has been done, or basically, no grouping has been done. This is because, in experiment 1, the interval time is 1 second whereas in experiment 4 the interval time is 10 seconds. As in experiment 1, the interval time is very small, and more packets are being grouped and sent at a time by the relay. The opposite is the case for experiment 3, as the interval time is higher, and for the maximum of trips, there is no grouping of packets.

This behaviour of grouping packets before relaying them is observed because the IBC works that way. Routing IBC packets include options like timeout requirements and ordering guarantees. So a series of packets come together within a certain time and in a certain order and then they are sent. The acknowledgement packets also follow the same process.

The figures, 7.2, 7.6, 7.10 and 7.15 represent the round trip times of each packet being sent from one blockchain to another blockchain for all the respective experiments. From all these figures we observe that these round trip times are following a certain output behaviour. The red line in the graphs represents the mean of the round trip times. We have noticed that the mean is varying throughout the whole simulation time. So we have split the whole data of round trip times into two halves,

that is batches. Batch 1 represents the round trip times of packets from 1 to 500 and batch 2 represents the round trip times of packets from 500 to 1000. Mean 1 corresponds to the mean of batch 1 and mean 2 corresponds to the mean of batch 2. The overall mean represents the mean of round trip times of all the packets that are from 1 to 1000.

Table 7.5 shows mean 1, mean 2, overall mean, and rate of all the experiments.

| Experiment | interval time | rate | mean 1 | overall mean | mean 2 |
|--------------|---------------|--------|--------|--------------|--------|
| Experiment 1 | 1 second | 0.4786 | 7.5111 | 7.5376 | 7.5641 |
| Experiment 2 | 5 second | 0.1907 | 7.2450 | 7.4211 | 7.5973 |
| Experiment 3 | 10 second | 0.0963 | 7.2509 | 7.4801 | 7.7092 |
| Experiment 4 | random | 0.1201 | 7.2603 | 7.4392 | 7.6184 |

Table 7.5: Comparison of all experiments

From the statistical values above we can say that as the rate decreases, the difference in mean 1 and mean 2 from the overall mean increases. In experiment 1, the difference from the overall mean is +0.0265 and -0.0265. In experiment 2, the difference from the overall mean is +0.1762 and -0.1762. In experiment 3, the difference from the overall mean is +0.2292 and -0.2292. In experiment 4, the difference from the overall mean is +0.1792 and -0.1789.

In experiment 1, the rate is highest at 0.4786, and the deviation of mean 1 and mean 2 from the overall mean is the least at 0.0265, whereas in the case of experiment 3, the rate is least with 0.1201 and the deviation of mean1 and mean2 from overall mean is highest with 0.2292. From these observations, we can conclude that the simulation time increases, hence the rate decreases, and the deviation from the overall mean increases. In other words, the longer the time the relay is up and running, the more the deviation in the mean of round trip times.

Further analysis

To understand why this behaviour is occurring, we have done a re-implementation and conducted an investigation to find out which part of the implementation would be causing this behaviour in the round-trip times. In the initial implementation mentioned in chapter 7, input to the sleep function in the bash script was given in units of seconds and the output, that is packet sending time and acknowledgement time were recorded in the unit of milliseconds. Hence, we decided to re-implement by modifying the units of the output to also be in seconds. Here we have re-implemented experiment 3. In the re-implementation, the results are similar to the initial implementations which are, the variations of the mean round-trip times still follow similar behaviour as previous implementations. Results, in this case, are given in the figures below.

From the figure, we observe a similar behaviour as the initial implementation. There is a drastic variation in the mean during the whole simulation time. From the results of the re-implementation, we can validate that the observed output behaviour is not because of the bash scripts. To investigate further the cause of this output behaviour, we need to dive deep into the IBC relay code. The code to implement

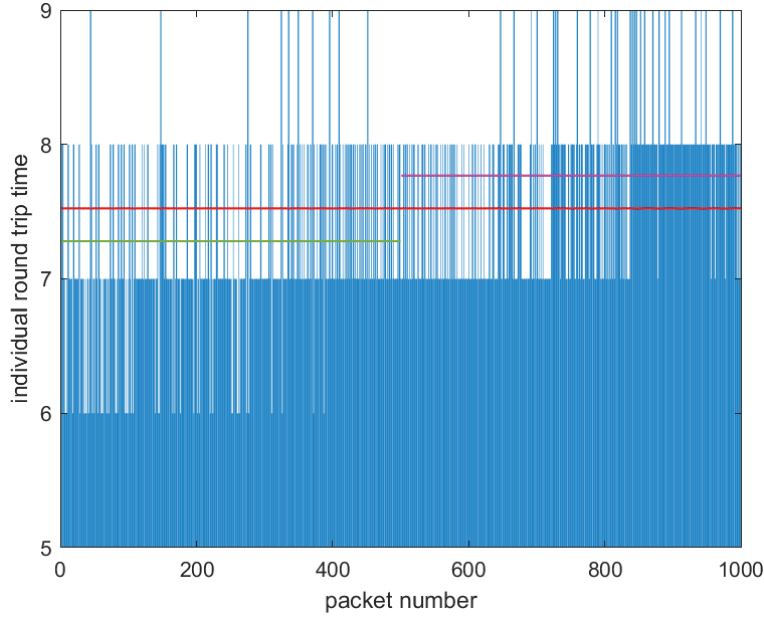


Figure 7.18: EXP3 re-implementation: Round trip time of packets

the IBC relayer is not written by us, we have used the modules provided by Cosmos SDK. The IBC relayer code review to investigate possible causes of this output behaviour is out of the scope of the thesis as it requires more time and understanding of the go programming language which is used in implementing the IBC relayer. As crosschain communication is an emerging topic, not many research papers on IBC implementation are found. The Inter-blockchain communication protocol implemented in this thesis has been launched in the cosmos ecosystem in April 2021. [20] After the launch many changes have been done and different versions have been released. Hence, we did not find more research papers describing the implementation of the IBC protocol in the Cosmos ecosystem.

7.6 Limitations

In this thesis we have some limitations which are stated below:

- This thesis is limited to sending the packets containing only messages from one blockchain to another blockchain. No tokens were sent from one blockchain to another blockchain. The aim of this thesis is to implement a crosschain communication protocol and do a performance evaluation of the protocol, which has been achieved by sending packets across the blockchains, and those packets contain messages only. Hence this thesis is only limited to packets sending messages and not token transfer.
- We could not figure out the reason for the increasing round trip time as the simulation time increases. This behaviour is observed in the variation of the mean of round trip times in all our experiments. This can be further studied by doing code reviews in modules responsible for implementing the IBC relayer. As this procedure requires more time, hence our thesis is limited.

This chapter discusses the conclusion of our thesis. We are also suggesting future work as the continuation of this thesis.

8.1 Conclusions

This thesis work aims to determine if crosschain communication protocol can be implemented in blockchain-based energy marketplaces and also focuses on designing a system model for the initial implementation of crosschain communication protocol. The system model has been designed and implemented on the Virtual machine in the Microsoft Azure cloud. The model was further enhanced to be able to collect various metrics which were later used to do performance analysis. The developed system model is limited to the current implementation of the IBC protocol in Cosmos which is the platform used to develop the model. Both blockchains which are communicating are residing in the same virtual machine, and the communication between them is done by sending packets containing messages. After the implementation, a detailed analysis of the performance of the IBC protocol has been done. We, therefore, present the solutions for the research questions which were set out in section 1.2.

In response to RQ1, the requirements for crosschain communication are safety, liveness, and atomicity. A detailed explanation of the three properties and references is given in section 3.2.3. In response to RQ2, the metrics that define the performance and scalability of crosschain communication protocol are rate, packet delivery ratio and round trip time. [34] From the implementation of our system model, rate tells how many packets are successfully received by the destination blockchain per unit of time, and packet delivery ratio articulates the loss ratio in the packet transmission by dividing the number of packets received by the number of packets sent. The round trip time shows the total delay in the transmission of a packet from the source blockchain to the destination blockchain. More details about these parameters are already given in section 5.2. Regarding RQ3, A detailed description of performance analysis and our observation of the implemented system model is given in section 7.5. In response to RQ4, we have been able to identify how to measure the performance of the crosschain communication protocol. This is articulated in chapter 7 where we discuss results and analysis. We developed different Matlab functions which were used to plot the presented graphs from the raw data collected from the model during crosschain communication experimentation. Regarding RQ5, some crosschain communication protocols following the crosschain mechanisms are the Interledger

protocol, hash-lock atomic swap, and IBC protocol. A detailed description of the available crosschain communication mechanisms and protocols is given under section 3.2.3. For the protocols to be suited for crosschain communication, they have to meet the requirements of the crosschain communication protocol which are already identified in RQ1.

8.2 Future work

As already pointed out in chapter 7 we have had limitations that hinder us from thoroughly analyzing the performance of the IBC protocol. Our results have shown that the longer the time relayer is up and running, there is more deviation in the mean of round trip times from the overall mean. To explicitly understand this, we need to review the code that implements the IBC relayer.

The IBC relayer in Cosmos SDK is implemented by using the Golang programming language. Since we do not have knowledge of the Golang programming language we cannot review that code. Moreover, we do not have enough time to learn the Golang programming language; hence, this becomes out of the scope of this thesis work.

Since we have already developed a model for measuring the performance of cross-chain communication protocols, further studies can be done by reviewing the code which implements the IBC relayer module to figure out why there is an increase in deviation of round-trip times from the overall mean as the relayer runs for a longer time.

References

- [1] “Autocorrelation,” <https://en.wikipedia.org/wiki/Autocorrelation>.
- [2] “The cosmos ecosystem,” <https://tutorials.cosmos.network/academy/1-what-is-cosmos/2-cosmos-ecosystem.html>.
- [3] “Cosmos history,” <https://v1.cosmos.network/about>.
- [4] “Cosmos sdk documentation,” <https://docs.cosmos.network/main/intro/overview#why-the-cosmos-sdk>.
- [5] “Cosmos whitepaper,” <https://v1.cosmos.network/resources/whitepaper>.
- [6] “Exponential distribution,” https://en.wikipedia.org/wiki/Exponential_distribution.
- [7] “Github link,” https://github.com/Isaya-Ntilema/IBC_relayer/tree/main.
- [8] “Ignite cli,” <https://ignite.com/>.
- [9] “Inter-blockchain communication: Basics,” <https://docs.ignite.com/guide/ibc>.
- [10] “Introduction to cosmos sdk modules,” <https://docs.cosmos.network/main/building-modules/intro>.
- [11] “Matlab,” <https://en.wikipedia.org/wiki/MATLAB>.
- [12] “Microsoft azure,” https://en.wikipedia.org/wiki/Microsoft_Azure.
- [13] “The poisson distribution and poisson process explained,” <https://towardsdatascience.com/the-poisson-distribution-and-poisson-process-explained-4e2cb17d459>.
- [14] “poisson point process,” https://en.wikipedia.org/wiki/Poisson_point_process.
- [15] “Probability distribution,” https://en.wikipedia.org/wiki/Probability_distribution.
- [16] “Random variable,” https://en.wikipedia.org/wiki/Random_variable.
- [17] “Tendermint core overview,” <https://docs.tendermint.com/v0.34/tendermint-core/#>.
- [18] “What is cosmos?” <https://v1.cosmos.network/intro>.
- [19] “What is ibc?” [Online]. Available: <https://tutorials.cosmos.network/academy/3-ibc/1-what-is-ibc.html>
- [20] “Why is ibc essential for blockchain developers?” <https://ignite.com/blog/why-is-ibc-essential-for-blockchain-developers>.

- [21] T. Ahram, A. Sargolzaei, S. Sargolzaei, J. Daniels, and B. Amaba, "Blockchain technology innovations," in *2017 IEEE technology & engineering management conference (TEMSCON)*. IEEE, 2017, pp. 137–141.
- [22] V. Buterin, "Chain interoperability," *R3 Research Paper*, vol. 9, 2016. [Online]. Available: <https://theblockchaintest.com/uploads/resources/R3%20%20Chain%20Interoperability%20-%202016%20-%20Sep.pdf>
- [23] J. Cosmos. (2020) Inter-blockchain communication protocol (ibc): The tcp/ip protocol of blockchains, bridges trends. [Online]. Available: [https://johnniecosmos.medium.com/inter-blockchain-communication-protocol-ibc-the-tcp-ip-protocol-of-blockchains-bridges-trends-~:text=The%20Inter%2DBlockchain%20Communication%20protocol%20\(IBC\)%20is%20a%20reliable,data%20across%20different%20sovereign%20blockchains](https://johnniecosmos.medium.com/inter-blockchain-communication-protocol-ibc-the-tcp-ip-protocol-of-blockchains-bridges-trends-~:text=The%20Inter%2DBlockchain%20Communication%20protocol%20(IBC)%20is%20a%20reliable,data%20across%20different%20sovereign%20blockchains)
- [24] A. Delice, "The sampling issues in quantitative research," *Educational Sciences: Theory and Practice*, vol. 10, pp. 2001–2018, 03 2010.
- [25] L. Deng, H. Chen, J. Zeng, and L.-J. Zhang, "Research on cross-chain technology based on sidechain and hash-locking," in *Edge Computing – EDGE 2018*, S. Liu, B. Tekinerdogan, M. Aoyama, and L.-J. Zhang, Eds. Springer International Publishing, 2018, pp. 144–151.
- [26] P. T. Duy, D. T. T. Hien, and V.-H. Pham, "A survey on blockchain-based applications for reforming data protection, privacy and security," 2020. [Online]. Available: <https://arxiv.org/abs/2009.00530>
- [27] C. Goes, "The interblockchain communication protocol: An overview," *arXiv preprint arXiv:2006.15918*, 2020.
- [28] J. Golosova and A. Romanovs, "The advantages and disadvantages of the blockchain technology," in *2018 IEEE 6th workshop on advances in information, electronic and electrical engineering (AIEEE)*. IEEE, 2018, pp. 1–6.
- [29] R. Jidrot and G. Perumal, "Workload characterization and performance evaluation of a blockchain implementation for managing federated cloud resources—assuming a peer-to-peer energy management use case," 2021.
- [30] D. Khan, L. T. Jung, and M. A. Hashmani, "Systematic literature review of challenges in blockchain scalability," *Applied Sciences*, vol. 11, no. 20, 2021. [Online]. Available: <https://www.mdpi.com/2076-3417/11/20/9372>
- [31] S. Lin, Y. Kong, S. Nie, W. Xie, and J. Du, "Research on cross-chain technology of blockchain," in *2021 6th International Conference on Smart Grid and Electrical Automation (ICSGEA)*. IEEE, 2021, pp. 405–408.
- [32] D. S. D. D. P. A. Merlinda, Valentin, "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," *Renewable and Sustainable Energy Reviews*, vol. 100, pp. 143–174, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1364032118307184>

- [33] B. N. Musungate, B. Candan, U. C. Çabuk, and G. Dalkılıç, “Sidechains: Highlights and challenges,” in *2019 Innovations in Intelligent Systems and Applications Conference (ASYU)*. IEEE, 2019, pp. 1–5.
- [34] J. Mwanza and B. Nyirenda, “Performance evaluation of routing protocols in mobile ad hoc networks (manets),” 2008.
- [35] W. Ou, S. Huang, J. Zheng, Q. Zhang, G. Zeng, and W. Han, “An overview on cross-chain: Mechanism, platforms, challenges and advances,” *Computer Networks*, vol. 218, p. 109378, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128622004121>
- [36] I. A. Qasse, M. Abu Talib, and Q. Nasir, “Inter blockchain communication: A survey,” in *Proceedings of the ArabWIC 6th Annual International Conference Research Track*, ser. ArabWIC 2019. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: <https://doi-org.miman.bib.bth.se/10.1145/3333165.3333167>
- [37] P. Robinson, “Survey of crosschain communications protocols,” *Computer Networks*, vol. 200, p. 108488, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1389128621004321>
- [38] E. Scheid, B. Rodrigues, and B. Stiller, “Toward a policy-based blockchain agnostic framework,” in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019, pp. 609–613.
- [39] C. Staff. (2021) Cross-chain interoperability: What it means for blockchain. [Online]. Available: <https://www.gemini.com/cryptopedia/why-is-interoperability-important-for-blockchainsection-benefits-of-blockchain-interoperability>
- [40] H. Tam Vo, Z. Wang, D. Karunamoorthy, J. Wagner, E. Abebe, and M. Mohania, “Internet of blockchains: Techniques and challenges ahead,” pp. 1574–1581, 2018.
- [41] R.-V. Tkachuk, D. Ilie, K. Tutschku, and R. Robert, “A survey on blockchain-based telecommunication services marketplaces,” *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 228–255, 2022.
- [42] N. Wu, Jiani Tran, “Application of blockchain technology in sustainable energy systems: An overview,” *Sustainability*, vol. 10, no. 9, 2018. [Online]. Available: <https://www.mdpi.com/2071-1050/10/9/3067>
- [43] J. Xie, H. Tang, T. Huang, F. R. Yu, R. Xie, J. Liu, and Y. Liu, “A survey of blockchain technology applied to smart cities: Research issues and challenges,” *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2794–2830, 2019.
- [44] A. Xiong, G. Liu, Q. Zhu, A. Jing, and S. W. Loke, “A notary group-based cross-chain mechanism,” *Digital Communications and Networks*, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S235286482200061X>
- [45] D. Yaga, P. Mell, N. Roby, and K. Scarfone, “Blockchain technology overview,” *arXiv preprint arXiv:1906.11078*, 2019.

- [46] D. Yang, C. Long, H. Xu, and S. Peng, “A review on scalability of blockchain,” ser. ICBCCT’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–6. [Online]. Available: <https://doi-org.miman.bib.bth.se/10.1145/3390566.3391665>
- [47] C. P. S. Yli-Huumo, Ko, “Where is current research on blockchain technology?—a systematic review,” 2016. [Online]. Available: <https://doi.org/10.1371/journal.pone.0163477>
- [48] G. Yu, X. Wang, and R. P. Liu, “Cross-chain between a parent chain and multiple side chains,” *arXiv preprint arXiv:2208.05125*, 2022.

Appendix A

Supplemental Information

Github link to refer code: [7]

