



A study on universal standard for cross-ledger intercommunication

Classification, analysis and comparison of cross-chain projects

YUAN FAN

Master Thesis in Communication System
Supervisor(s): Henrik Gradin, György Dán

Examiner: Viktoria Fodor

School of Information and Communication Technology

KTH Royal Institute of Technology

Host Company: Centiglobe

August 31, 2019

Stockholm, Sweden 2019

Abstract

Write your abstract here...

Keywords: Distributed Ledger, Cross-Chain Study, Atomic swaps, Interledger, Relay, Sidechains

Sammanfattning

Write your Swedish summary (popular description) here...
Keywords: Keyword1, keyword2, ...

Acknowledgements

Write your professional acknowledgements here...

Acknowledgements are used to thank all persons who have helped in carrying out the research and to the research organizations/institutions and/or companies for funding the research.

Name Surname,
Place, Date

Contents

Contents	v
List of Tables	vii
List of Figures	viii
List of Acronyms	xii
Listings	xii
1 Introduction	1
1.1 Motivation	1
1.2 Research Questions	1
1.3 Research Methodology	2
1.4 Research Objectives	2
1.5 Delimitations	3
1.6 Thesis Organization	3
2 Literature Study	5
2.1 Background	5
2.1.1 What is cross-ledger?	5
2.1.2 Evolution of Cross-chain	5
2.2 Cross-chain manifestation	6
2.3 Difficulties	8
2.4 Ethereum	9
2.5 Related Works	9
2.6 Summary	9
3 Project Classification	11
3.1 Solutions	11
3.1.1 Ensure the atomic of transactions	11
3.1.2 Complete the transaction confirmation	15
3.1.3 Realize multiple chains interoperability	19

3.2	Project study	21
3.2.1	Lightning Network	21
3.2.2	Notary Scheme	22
3.2.3	Relay Scheme	26
3.2.4	Sidechain Scheme	33
3.2.5	PalletOne	37
3.3	Summary	41
4	Implementation and analysis	43
4.1	HTLC Atomic-swaps implementation	43
4.2	Interledger switch API study	47
4.3	Summary	48
5	Conclusion and Future Work	49
5.1	Conclusion	49
5.2	Future Work	49
	Bibliography	51
	Appendix A	53
	Appendix B	61
	I. Atomic swaps based on HTLC	61
	II. Ethereum smart contracts	62

List of Tables

3.1 Main roles in Polkadot	31
1 Summary of 20 Cross-chain Projects	54

List of Figures

1.1	Research components of cross-chain communication	3
2.1	Cross-chain development tree	6
2.2	Inter-chain exchange diagram	7
2.3	Inter-chain transfer diagram	8
3.1	Atomic swaps diagram	12
3.2	Hash Time-lock Contract diagram	14
3.3	Interledger HTLA diagram	15
3.4	Centralized Notary Scheme diagram	16
3.5	Multi-sig Notary Scheme diagram	17
3.6	Distributed signature Notary Scheme diagram	17
3.7	Direct interconnection network architecture diagram	19
3.8	Cross-chain platform network architecture diagram	20
3.9	ILP example process	23
3.10	Data transfer process from Ethereum to Wanchain ¹	24
3.11	Data transfer process from Wanchain to Ethereum ²	25
3.12	BTC-Relay cross-chain process diagram	27
3.13	Cosmos network architecture	28
3.14	IBC sequence diagram	29
3.15	IBC sequence diagram, timeout	30
3.16	Polkadot network architecture ³	32
3.17	2-way pegged sidechain diagram ⁴	34
3.18	Drivechain working diagram	36
3.19	OneLedger sidechain architecture ⁵	37
3.20	The normal process of executing the exchange contract (with Jury endorsement) ⁶	39
3.21	The normal process of executing the exchange contract (with Mediator endorsement) ⁷	40
4.1	BTS->ETH swap diagram	44
4.2	BTS->ETH swap diagram	45
4.3	XRP->ETH	47

List of Figures

ix

4.4	48
-----	----

List of Acronyms

ICT	Information Communications Technology
BTC	Bitcoin
ETH	Ethereum coin
HTLA	Hash Time-locked Agreement
HTLC	Hash Time-locked Contract
MPC	Multi-Party Computation
SPV	Simplified Payment Verification
RSMC	Recoverable Sequence Maturity Contract
addr	address
Tx	Transmit(Tx) Data
ILP	Interledger Protocol
DApps	Decentralized Applications
ELA	Elastos coin
HHCM	Hierarchical Hybrid Consensus Mechanism
PoW	Proof-of-Work
PoS	Proof-of-Stake
BFT	Byzantine Fault Tolerance
DAG	Directed Acyclic Graph
DPoS	Delegated Proof-of-Stake
PoI	Proof-of-Intelligence
API	Application Programming Interface
IBC	Inter-Blockchain Communication protocol
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
PBFT	Practical Byzantine Fault Tolerance
LFT	Loop Fault Tolerance
AUXPoW+DPoS	Auxiliary Proof of Work and Delegated Proof of Stake

Listings

1	Bitshares side	61
2	Ethereum Side	62
3	Hashed Timelock Contracts (HTLCs) on Ethereum	63
4	Ethereum micro-payment channel – Machinomy	63

Chapter 1

Introduction

1.1 Motivation

Initially, the ledger is the foundation of accounting. However, it is not difficult to find that there are many shortcomings in the long-term use of traditional ledgers, for example, low efficiency, high cost, opacity and easy to cause fraud and abuse issues.

With the development of information technology, these ledgers have gradually evolved into digital technology. The distributed ledger is a significant leap after the digitization of ledger-based technology. A distributed ledger is a database that is shared, replicated, and synchronized between network members [1]. It records transactions between network participants, such as the exchange of assets or data. From a technical point of view, the Distributed Ledger Technology (DLT) not only inherits the traditional bookkeeping philosophy but also has its unique innovations, which have some advantages that traditional ledgers cannot reach.

Based on the background above, there exist many distributed ledgers, and for them to be fully distributed they need to communicate with each other and also traditional ledgers. Otherwise, the consistency between the ledgers of different entities across the chain would not be guaranteed. There have been many different approaches, such as cross-chain protocols and platforms released to realize the cross-chain transactions, so it is worthy of finding out the differences between them.

1.2 Research Questions

There is one technical issue that affects the blockchain developers, that is the inter-communications. A single blockchain network is a relatively closed system that does not actively interact with the outside world. The assets of each chain are also an independent value system. If we can break through the interoperability among different ledgers and let the value circulate on the broader world, it will inevitably promote the rapid development of the blockchain industry. Cross-chain technology

is dedicated to building a bridge of trust between ledgers, breaking the situation of an isolated value system, and realizing asset interoperability to achieve a real win-win situation.

This project makes uses of qualitative research strategy, where the research approach always implemented as interpretivism. Therefore qualitative approach was found to be most applicable for this study.

For this research, I decided to use the literature review first to gain a deep understanding of cross-ledger history and realization of communications through published papers. Then the critical point is the case study among popular projects. By studying the different cross-chain implementations, exacting the main idea of them, we can summarize and categorize them into different groups. Hence, identify patterns and commonalities in the cross-chain field. Even help more and more developers to consummate the blockchain design.

1.3 Research Methodology

1.4 Research Objectives

My research has contributed to the universal demanding and requirements on cross-ledger communication towards blockchain areas. In particular, I have focused on the problems that the realization of cross-chain communication is facing.

For a long time since Bitcoin open the blockchain era, the blockchain world is like the single-machine time back in the 1960s. Every blockchain is highly independent and challenging to communicate with each other. Hence, the data and services in the blockchain world are confined to the individual blockchain. As a result, this phenomenon will discourage development. As if we could find a standardized cross-chain protocol/platform that could link all blockchain systems, and the services could be more specific and complete due to the co-operate of blockchains. The popularization and mature of cross-chain technology will lead a revolutionary development in the blockchain field. Moreover, different from the Internet to achieve the circulation of information, cross-ledger could realize the distribution of value.

To understand different patterns or implementations of cross-chain technology, we need to start with the history of cross-chain and grasp the main idea of chain interoperability based on literature review. Based on those findings, we could summarize the fundamental problems the blockchains now facing, according to those difficulties, I will give several examples through the case study in the following paragraph. The comparison from various aspects will next lead to a standard and universal needs of the cross-chain area. In my thesis, I have considered the following three major elements of this study as shown in Figure 1.1

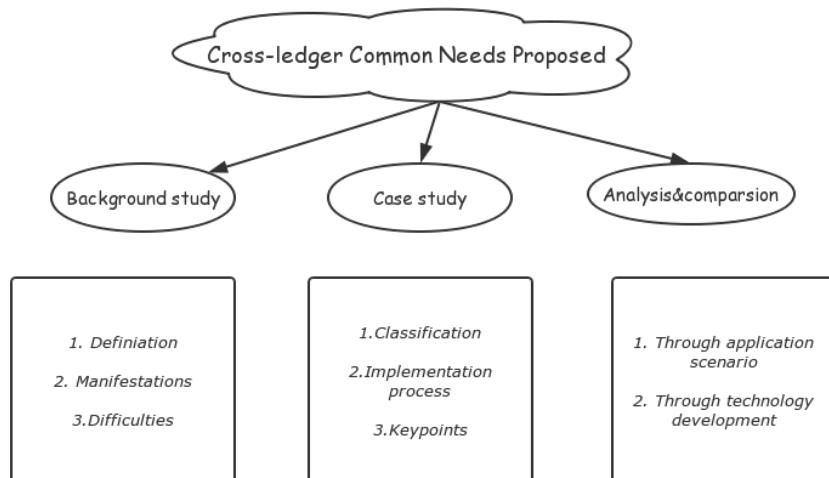


Figure 1.1: Research components of cross-chain communication

1.5 Delimitations

1.6 Thesis Organization

This thesis is organized into five chapters, as follows:

- Chapter 1 concentrates on the research value and market meaning of this thesis, then briefly introduces the concept of cross-chain communication.
- Chapter 2 studies the background of the cross-chain project by classifying different manifestations of cross-chain communication as well as pointing out the problems cross-ledger communication facing.
- Chapter 3 focuses on the theoretical solutions that will address the difficulties, discusses the communication process of various cross-chain projects based on different group rules.
- Chapter 4 put efforts in analyzing the market demand situation and discuss the applications that could be adopted, compare the technology development of some representative projects.
- Chapter 5 summarizes the findings during the research and suggests several ideas for related future work.
- Appendix A contains a comparison table that evaluating 20 cross-chain projects from several valuable aspects.

- Appendix B lists implementation code pieces with essential functions explained and the utilization of smart contracts, for further test use.

Chapter 2

Literature Study

2.1 Background

2.1.1 What is cross-ledger?

Since the development of blockchain technology, many different chains have been born, and the information isolation of many chains inevitably forms the value island effect of blockchain. There is a need for a technology that can work with different blockchains and become the bridge connecting them.

Cross-ledger (or cross-chain) in the narrow sense is the process of asset/data interoperability between two relative independent blockchains.

Traditional ICT field defines the *interoperability* as the ability to share and utilize information between different ICT systems or modules in a reliable way [2]. While VITALIK BUTERIN mentioned in CHAIN INTEROPERABILITY [3] that interoperability in the blockchain area mainly describes the ability of assets exchange, information intercommunication and atomic transactions between various blockchains. The blockchain interoperability can be achieved by introducing the third party without modifying the original chains.

2.1.2 Evolution of Cross-chain

As shown in Figure 2.1, it has not been a short period for Ripple to release the *Interledger Protocol*(ILP) [4] in 2012 since the advent of Bitcoin Network in 2009. Interledger protocol proposed a cross-ledger interoperability scheme for the first time in the blockchain area, which enables cross-ledger transfers through third-party notaries. In 2014, the BlockStream team, founded by the Bitcoin core developer group, first proposed a *Pegged Sidechains* cross-chain interaction scheme, introducing a sidechain with a two-way peg to achieve cross-chain asset transfer. Soon in 2015, the Bitcoin Lightning Network [5] adopted a *Hashed Timelock* mechanism to implement a fast transaction channel off the Bitcoin main chain. In 2016, the BTC-Relay [6] solution was released, and the one-way cross-chain communication

from Bitcoin to Ethereum was realized based on the relay cross-chain scheme. In the same year, CHAIN INTEROPERABILITY [3] by Vitalik Buterin made a comprehensive and in-depth analysis of blockchain interoperability issues. In 2017, Polkadot and Cosmos first proposed a solution for building a cross-chain network infrastructure platform. Now the Cosmos hub blockchain has launched in March 2019.

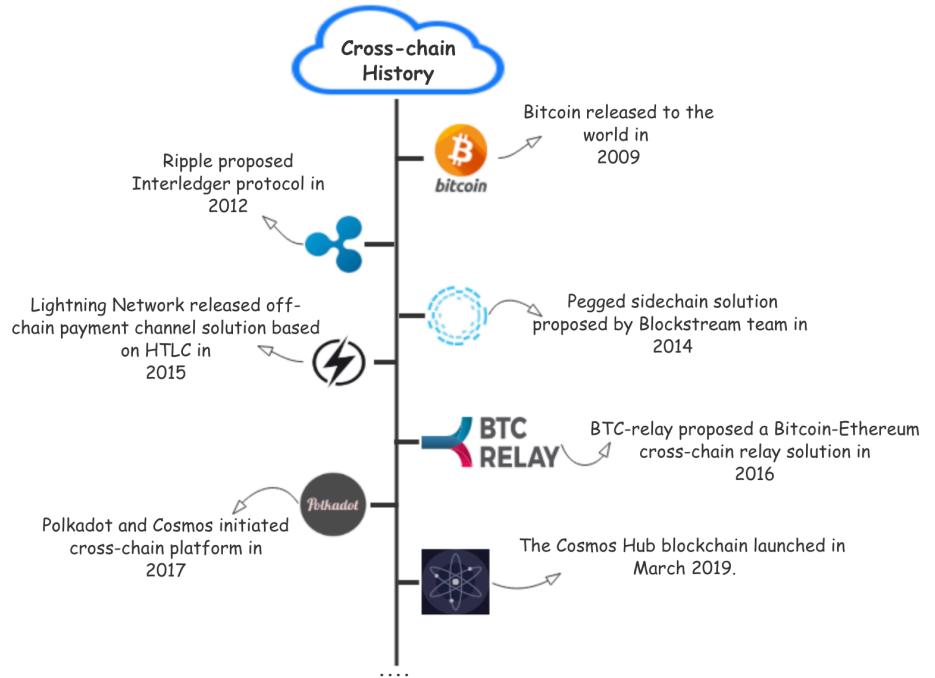


Figure 2.1: Cross-chain development tree

2.2 Cross-chain manifestation

So far we know, current public blockchain is a self-adaptive, relatively closed distributed system. Although the system allows new nodes to join and old nodes exit, it also has its fault-tolerant mechanism. It is hard to be compatible with external systems.

In a way, there are two main value/data interoperability implementations between chains:

1. Inter-chain asset exchange:

It usually refers to asset swapping between different users on both chains. However, the total amount of assets in each chain does not increase or decrease. Instead, the ownership of the assets has changed. The process of this

change needs to occur synchronously on both chains.

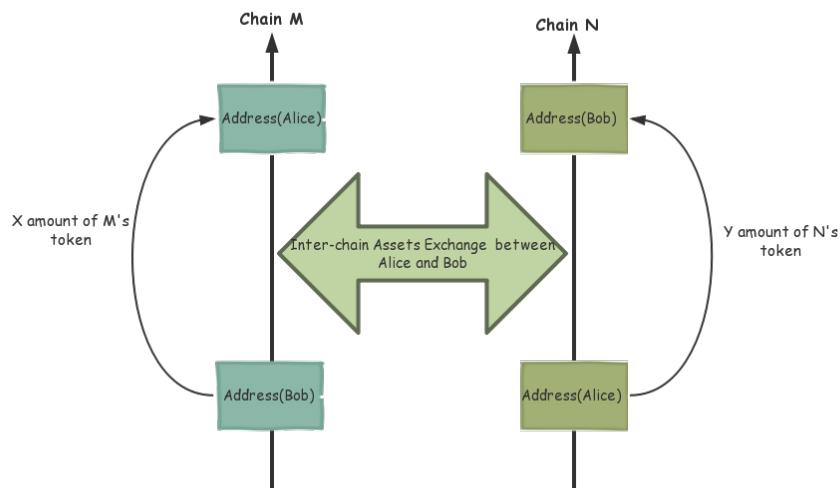


Figure 2.2: Inter-chain exchange diagram

This diagram briefly describes the situation when Alice wants to use X amount of chain M's token exchange for Bob's Y amount of chain N's token. Eventually, Bob's token in chain N swaps to Alice's address in chain N, similarly, Alice's token swaps to Bob's address located in chain M.

2. Inter-chain asset transfer (one/two-way):

Compared with the consistency in the total assets number of asset exchange, the transfer has a transfer of asset value, which is manifested in the increase or decrease of the available assets in each chain. For example in Figure 2.3, the scenario is Alice wants to transfer X number of chain M's token to chain N, as a result, chain M send X amount of token to a locked address in original chain, in turn, chain N generates an equal amount of tokens of itself in a certain address for future uses. Thus, the transfer of this asset succeed.

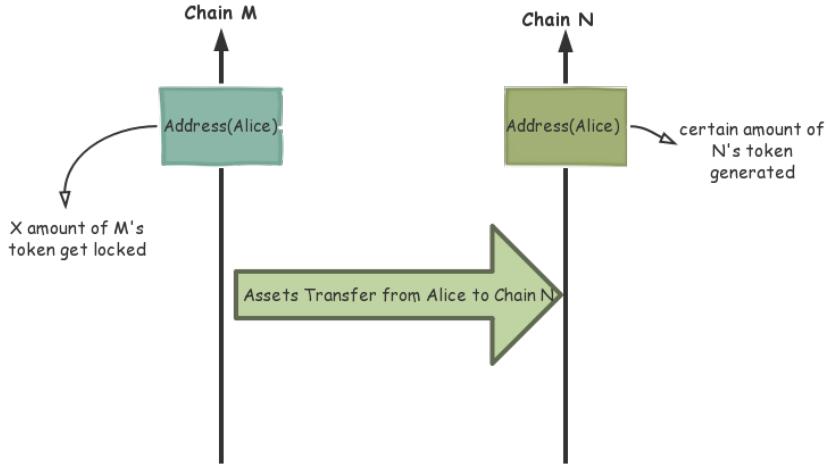


Figure 2.3: Inter-chain transfer diagram

For now, researches and applications of cross-chain are mainly focused on these two methods. Some projects have proposed the concept of cross-chain smart contracts, which is similar to the realization of cross-chain asset transfer in terms of technical implementation.

2.3 Difficulties

The number one feature in the blockchain is immutability, and every record must be accurate to protect value. Hence, in the cross-chain project, the key point is to ensure the accuracy of each transaction. There are many obstacles cross-chain need to encounter. To sum up, the following are several key points:

- **How to ensure the atomic of transactions.**
That is, cross-chain transactions either occur or do not occur. Otherwise, the inconsistency and "out-of-sync" status of the two chains will become the most significant system vulnerabilities in cross-chain transactions, and the security of both systems will be threatened. The consistency is the fundamental guarantee or realizing cross-chain transactions, and it is also a difficult point that must be solved in cross-chain transactions.
- **How to complete the confirmation of the transaction for other chains.**

The confirmation of the transaction contains two levels of problems. One is to confirm that the transaction has occurred and is wound up and written into the right block. Second is to make sure the system has approved the transaction with enough blocks. In this way, the probability of invalidation of the transaction due to system reconfiguration will be very low. The blockchain system itself is relatively closed, lacking the mechanism to obtain external information actively. Therefore, it is not an easy task to confirm the transaction status of another chain. It can be said that it is one of the core difficulties of cross-chain transactions.

- **How to ensure that the total assets of the two chains remain unchanged.**

In the scenario of asset exchange, the assets of the two chains are not substantially exchanged, so this type of situation does not change the total assets of each chain. However, in the scenario of asset transfer, the number of available assets in each chain changes, total assets can remain unchanged only when the cross-chain transactions are accurately recorded, and the accounting of the two chains is completely atomic, either at the same time or not.

- **How to ensure the independent security of the two chains.**

When two chains inter-operate, they will inevitably affect each other. How to ensure the security of their chains and the others in the process of cross-chain transactions is a problem worth considering. If the security issue cannot be isolated, then one attacked chain will affect the entire cross-chain network.

- **How to realize multiple chains interoperability.**

Take the history of the computer network as a reference. The independent blockchain network will eventually embark on the future of interconnection. How to link these existing and future blockchain networks to be unified into one whole network will be one of the most important issues of the future cross-chain network.

2.4 Ethereum

2.5 Related Works

2.6 Summary

This chapter introduces a thorough background study of the cross-chain research area. It briefly reviews the history of cross-chain development and outlines the importance of the growing technology of cross-chain. Two main manifestations of cross-chain transactions are proposed to give a classification standard for Chapter 3 project study. Chapter 2 also describes the critical difficulties that cross-chain projects are facing. Some of them will be discussed in the following chapter.

Chapter 3

Project Classification

3.1 Solutions

Based on what we discussed about the difficulties in the previous sections, some solutions came up with multiple cross-chain projects. The following paragraph will illustrate them thoroughly

3.1.1 Ensure the atomic of transactions

3.1.1.1 Atomic swaps

An atomic cross-chain swap [7] is the basic theoretical framework for multiple parties exchange assets across multiple blockchains. Atomic operations in computer science ensure every exchange either success or failure, no third intermediate state. For a more intuitive introduction to the atomic swap protocol, we assume an exchange scenario:

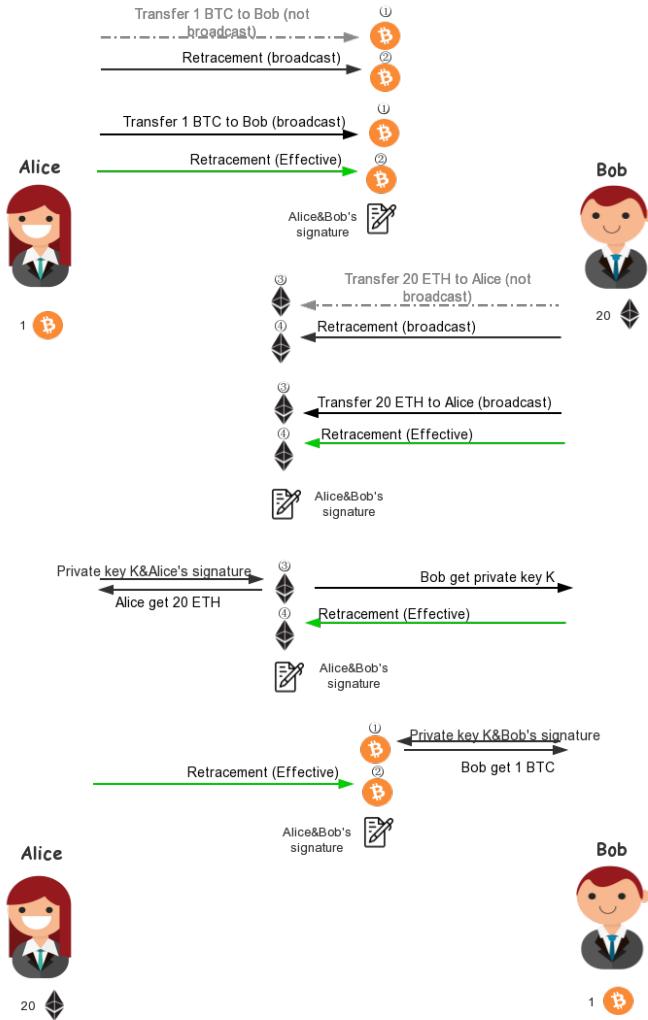


Figure 3.1: Atomic swaps diagram

1. Assume Alice has 1 BTC on chain A while Bob has 20 ETH on chain B, and Alice wants to change Bob's 20 ETH with one BTC. Both Alice and Bob have wallet addresses on both chains A and chain B.
2. To initiate this transaction, Alice needs to randomly generate a key K, which is known only to Alice and then initiates a 1 BTC on chain A transaction

(transaction ①) to Bob. The transaction can be only finished when obtains the signature of Bob and provides the key K.

3. Before broadcast transaction ①, Alice will first broadcast a retracement (transaction ②). If transaction ① does not receive the correct key and signature within 48 hours, the amount paid by that will be returned to Alice. Transaction ② must be signed by Alice and Bob after the broadcast to take effect. At the same time, Alice will only broadcast the transaction ① to the network if transaction ② is successfully validated.
4. Bob now sees the transaction ② sent by Alice. If Bob agrees, he will sign the transaction ②, of course, Alice will also complete the signature so that the retracement will take effect. Then Alice will broadcast the transaction ① to the whole network.
5. Bob can only get the value K after he pays Alice with 20 ETH. Hence Bob initiates transaction ③ on chain B to pay Alice 20 ETH. These 20 ETH are only available if Alice enters the decrypted key K and attaches Alice's signature. To prevent Alice from denying, Bob also issues a retracement transaction ④ that requires Alice and Bob to sign together before the broadcast transaction ③, when Alice does not provide the correct key or the signature within 24 hours. Then activate the retracement, 20 ETH will be returned to Bob.
6. After Alice sees the transaction ④, both Alice and Bob need to attach their signature to this transaction to take effect. At this time, Bob will broadcast transaction ③ to network.
7. In order to get 20 ETH, Alice will sign transaction ③ with the correct value K. For now, transaction ③ succeeds, Alice obtains 20 ETH, and Bob obtains key K.
8. After Bob gets the key K, he goes back to chain A, enters the key K and his signature, and finally gets 1 BTC from Alice.

From the diagram, we could obtain that the atomic swap protocol does not transfer the assets of Chain A to Chain B, but only the assets ownership of both chains. The total assets of Chain A and Chain B have not changed, so it can only achieve asset exchange between chains and cannot achieve asset transfer.

This solution not only can be applied to the decentralized ledger system but also the centralized ledgers. As long as the two systems provide the functions of retracement, time lock, and key lock.

3.1.1.2 Hash Time-Locked Contracts(HTLC)

HTLC is a very technical implementation of the atomic swap protocol. It guarantees the atomicity of the transaction through the hash lock and the time lock mechanism.

In different systems, whether it is a blockchain system or a centralized ledger system, despite the ways of implementing the lock, the principle behind it is the same, that is, only certain hash conditions or time met, the transaction is allowed to take effect.

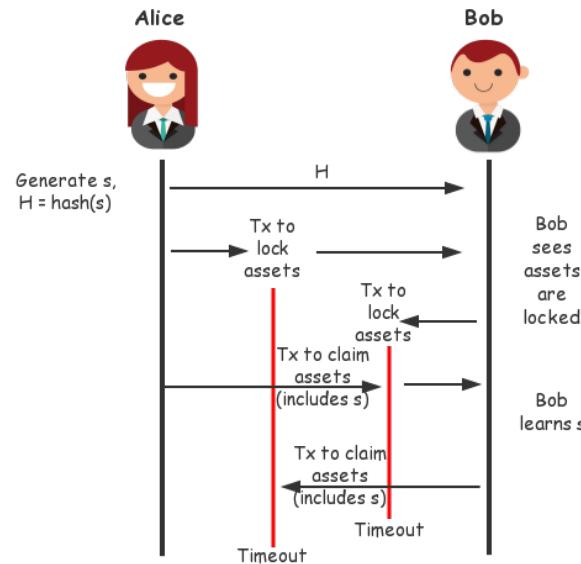


Figure 3.2: Hash Time-lock Contract diagram

Using only hash time locks is not enough when you want to achieve cross-chain asset transfer, you also need to cooperate with other cross-chain technologies to ensure the authenticity of cross-chain transactions.

3.1.1.3 Hash Time-locked Agreements(HTLA)

HTLA is one HTLC generalization protocol came up by Interledger [8], regardless whether the system support HTLC or not, whether it's a distributed or centralized ledger. HTLA can be used to implement cross-chain exchange between system or even support multi-hop cross-chain interchange between multiple systems, as shown in the Figure 3.3 below.

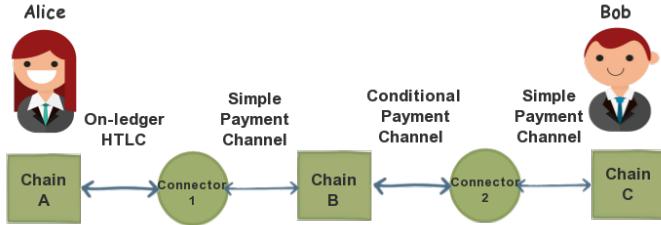


Figure 3.3: Interledger HTLA diagram

Alice and Bob can trade among blockchains A, B, and C via HTLA, and each blockchain supports different cross-chain protocols. The connector here plays a role in connection and isolation. Linking blockchains that support different cross-chain protocols together, and again isolate them, so that blockchains do not interfere with each other.

HTLA supports multiple cross-chain protocols based on HTLC; some of them are mentioned in Figure 3.3.

3.1.2 Complete the transaction confirmation

As we all know, blockchain systems are relatively independent and closed, there's no direct communication way for them to confirm every piece of records that happened. So no matter how it evolves, there will always be a "middle-man" between the two chains, taking on the role of information exchange between the two chains. Here the "middle-man" represent any entity that could interact with two chains, it may be one or a group, maybe the centralized or distributed agency, maybe a separate chain or even a functional module. The "middle-man" usually acts as a node for two blockchains at the same time, so that only one application software can be deployed on the same node to obtain the others' system data.

After the "middle-man" completes the data collection, how to confirm the transaction, which transaction to confirm, and who confirms becomes the key point of this problem. According to different schemes, this process can be summarized in three ways:

- **Notary [3]:**

In the notary scheme, a trusted one or group is used to declare to the chain X that an event has occurred on the chain Y, or that the statement is correct. These groups can both automatically or requested to listen and respond to events. There are three different sub-schemes came up in the evolution of this model:

- **Centralized Notary schemes**

The centralized notary mechanism is also called the single-signature notary mechanism, usually played by a single designated independent node or institution, which is the simplest model. Its purpose is instead of letting Alice and Bob, and two strange individuals deal, it's not as reliable as indirect transactions with third-party institutions with credit endorsements (such as Alipay). Since Alice and Bob exist in different ledger systems, the notary is technically required to be compatible with two or more systems at the same time.

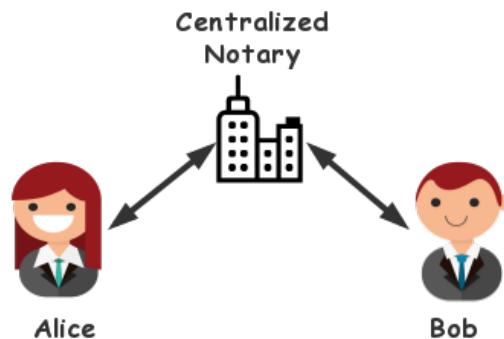


Figure 3.4: Centralized Notary Scheme diagram

To some extent, the use of centralized institutions has replaced technical credit guarantees, from professional credibility to traditional credit intermediaries. Although this kind of mode has fast transaction processing, strong compatibility, and simple technical architecture, the security of the central node has become a critical bottleneck for system stability.

- **Multi-sig Notary schemes**

The multi-signature notary mechanism is accomplished by multiple notaries that can sign a common agreement on their respective ledgers to complete the cross-chain transaction. Each node of the multi-signature notary group has its private key, and cross-chain transactions can only be confirmed when a certain number or proportions of notary signatures are reached.

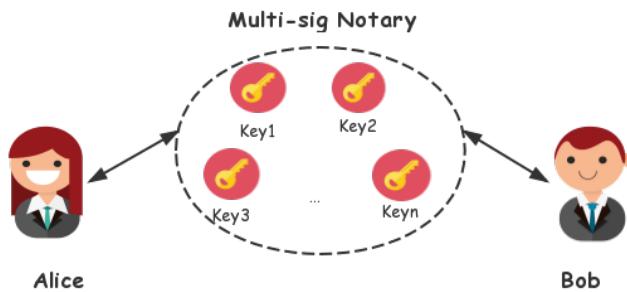


Figure 3.5: Multi-sig Notary Scheme diagram

This method is more secure than the single-signature mode, and a few notaries who are attacked or do evil will not affect the regular operation of the system. However, this approach requires both chains to have the ability to support multiple signatures.

- **Distributed signature Notary schemes**

The main difference between a distributed signature and multi-signature is the signature generation. Distributed signature using *Multi-Party Computation*(MPC), which will enhance the security as well as the implementation difficulty.

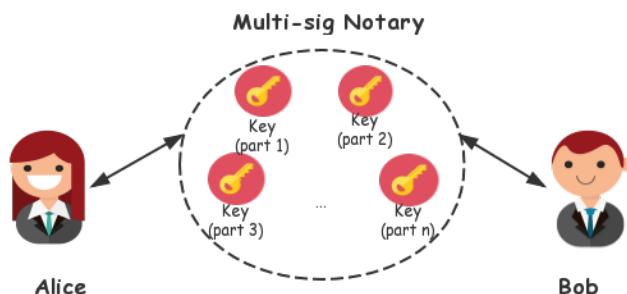


Figure 3.6: Distributed signature Notary Scheme diagram

As Figure 3.6 shows, distributed signature based on cryptography, the key point is that for cross-chain transactions, the system generates one and only one key, and no one in the notary group will have a complete key. The key is randomly sent to each notary node in the form of fragments. Meanwhile, the fragment is processed as ciphertexts, making it impossible to extract the complete key even all the participants put the pieces together. Thus, the security of the key is fully guaranteed.

- **Relay [3]:**

Relay is one flexible and easy-to-expand cross-chain technology that does not rely on trusted third parties to help with transaction verification. Instead, it is self-verified by the receiving chain after receiving the send chain data. Self-verification methods are depending on the system structure. For example, BTC-relay [6] based on *Simplified Payment Verification*(SPV), and Cosmos [9] rely on verify the number of nodes' signature.

VITALIK mentioned relay in his Chain Interoperability paper [3], pointing out that chain A and chain B can use the other party's block data for information synchronization and cross-chain calls. Currently, information synchronization can be done, but there is no mature technical solution for cross-chain calls. Two chains cannot verify the validity of each others block at the same time. Otherwise, they will fall into an infinite loop of nesting. If chain A owns the block data of chain B, then chain A needs to be confirmed in the case of chain B transaction confirmation, and chain B needs to wait for chain A's transaction confirmation because it also has block A's block data, it goes on as a loop.

- **Sidechains:**

The concept of a sidechain as defined in white paper [10] is: *sidechain is a blockchain that validates data from other blockchains*. However, this explanation was considered to be too broad and not rigorous by VITALIK BUTERIN in CHAIN INTEROPERABILITY [3]. “Sidechain” is more frequently used to refer to what Blockstream calls a “pegged sidechain”, where the functionality of a blockchain is of an anchored asset of another asset, which chain is regarded as the parent chain. In this way, this relationship is based on assets, not the blockchain itself. Pegged sidechain is a strong coupling cross-chain structure using directly embeds part of the data of the original chain into its own block or storage space. In the case of cross-chain transactions, verification can be completed directly through the original chain data stored in the system. This method is generally considered bidirectionally at the beginning of the system design.

Compared to notary and relay, the sidechain is more direct. The state of one chain will be directly reflected in the data of the other chain. When one chain is attacked, the other chain may also be affected. This model is more suitable for the design of the same system, which allows the two sides to become a whole without losing the relative independence of the ledgers.

3.1.3 Realize multiple chains interoperability

The computer network connects the original independent computers into a local area network, the local area network develops into a metropolitan area network, the metropolitan area network evolves into the Internet, and the Internet connects the people of the world like never before.

However, for the emerging technology/industry of blockchain, it is still in the “single-machine” era, and the interactions demand between chains and chains will become increasingly strong with the application of blockchain.

To realize interoperability among multiple chains, there are two potential aspects of difficulties that need to overcome:

- How to achieve interoperability among blockchains system that has already developed.
- How to prepare/setup the way for the interconnections among the new blockchains in the future.

3.1.3.1 Active compatibility

This solution is mainly aimed at the existing blockchain system. First, there are different blockchain application systems in the upper layer, and then the underlying cross-chain mechanism is developed.

Usually these systems are heterogeneous and need to be docked one by one, but there is also a different solution for a pair of connections.

1. Direct interconnection between the two chains

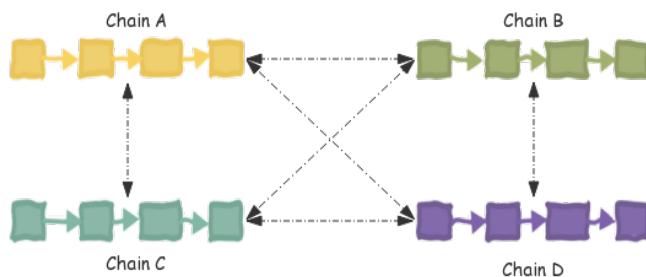


Figure 3.7: Direct interconnection network architecture diagram

This method is the most time-consuming and laborious without the support of the unified underlying protocol. It is necessary to establish six paths be-

tween the four chains to realize the interconnection between them. Also, each path needs to be customized. Although this method is not scalable, it can guarantee better security and independence. Once an attack occurs, it is difficult to affect the entire network.

2. Third-party cross-chain platform

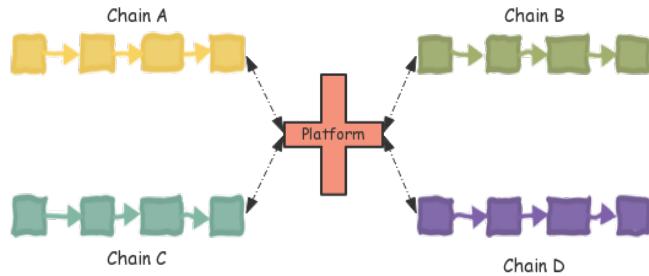


Figure 3.8: Cross-chain platform network architecture diagram

Once the cross-chain platform established, blockchains can indirectly interconnect with each other. Thus, only four paths are required to build a cross-chain network. However, in this method, the cross-chain platform will become the key point and performance bottleneck of the entire cross-chain network (not necessarily the bottleneck, but it may be in the future). Once the cross-chain platform is attacked, the entire cross-chain network will be paralyzed.

3.1.3.2 Passive compatibility

Passive compatibility is mainly aimed at the blockchain system that has not been developed. It first builds the underlying cross-chain platform, allowing other blockchain systems to be easily, conveniently and securely accessed. Cross-chain platforms will prioritize the development of systems and protocol standards that apply to interoperability between the various chains. The subsequent development of standards-compliant development on existing platforms allows for the creation of blockchains that naturally have cross-chain functionality within the system. However, the cross-chain mentioned here refers to the chain that conforms to the protocol standard can be easily connected. If it is to interoperate with other chains outside the system, it is necessary to develop a separate middleware to communicate.

Besides, different cross-chain platforms can support different types of blockchains, such as Cosmos supporting isomorphic chains and Polkadot supporting heterogeneous chains, both of which are highly scalable, and will be later discussed in Section 3.2.

3.2 Project study

3.2.1 Lightning Network

In general, we can not say lightning network realize the cross-chain function, though it provided a typical application towards atomic swaps and HTLC. The design idea of the lightning network is straightforward. It put a large number of high-frequency small-value transactions off-chain to expand the transaction processing capability of the blockchain.

Lightning network [5] is a fast and scalable Bitcoin transaction project, and it has two main technical points:

A. Recoverable Sequence Maturity Contract

RSMC is similar to a reserve mechanism in which both parties trade in an off-chain trading pool. This trading pool is a “micro-payment channel”. When a transaction occurs between two parties, the proportion share of the common assets in the trading pool will change. The new proportional data needs to be signed and confirmed by both parties, and the old proportional share version is invalid. The entire process is completed off the chain, so it does not occupy the resources of the main chain. The final proportion of assets will be confirmed, and record to the main-chain after one of the transaction party requires a withdraw.

It could happen anytime as long as both parties signed for this. To ensure the security, if someone submitted the old share of assets to make profits, others could protect themselves by proving this balance sheet is not the latest one. Then the asset of the counterfeit party will be confiscated to the challenger.

B. Hash Time-locked Contract

Lightning Network uses HTLC to guarantee the atomicity of transaction, as shown in Figure 3.2.

This diagram illustrates how HTLC works to provide limited-time transfer function. The basic process is: Bob and Alice can reach an agreement that specifies Alice to lock a certain amount of assets and provide a hashed value of H . Before the arrival of time T , if Bob can learn an appropriate s (secret) where its hash value matches with H and sent to Alice, Bob can get the corresponding amount of assets value. Conversely, the asset will unlock and return to Alice.

When there are “micro-payment channels” between multiple users, these channels are connected to form a “channel network”, which is the lightning network. The

mutual transfer of the two parties does not require a direct payment channel to connect with, through the intermediary can also achieve mutual transfer.

Disadvantages:

- Users cannot pay off-line.
- More suitable for small transactions. Large transfer amount needs to open multiple channels.
- Easy to face the dis-matched situation, if there's no response from one party, the other one may need hours to close the channel and substitute with another route.

3.2.2 Notary Scheme

3.2.2.1 Ripple Interledger protocol

Earlier we focused on cross-chain, this project is more focused on cross-ledger, which means that the agreement supports not only decentralized blockchains but also supports various centralized ledgers, which is broader support for cross-chain applications. Ripple ILP [4] uses the HTLA and Notary scheme to implement this technology.

Ripple is the first project to propose the use of blockchain technology to achieve cross-ledger exchange of assets, with a focus on resolving cross-border remittances, enabling faster and more economical international remittances via the Ripple network.

Interledger Protocol(ILP) is compatible with any online ledger systems. Specifically, the ILP will establish a two-way pegged relationship between the trader's account and a Ripple local account, enabling simultaneous changes between the two to ensure transparency in the transaction process. At the same time, for two ledger systems that do not have a direct payment channel, multi-hop indirect cross-ledger transactions can be realized through ILP.

The main idea of ILP is to secure cross-ledger transactions by setting up *escrow account* on Ripple. So the process will need the preparation of escrow account of several parties in the transaction. As an example, in Figure 3.9 Bob, and one selected market maker should have their Ripple escrow accounts set up on two bank systems before the transaction.

- Alice first selects a market maker with the most suitable exchange rate, and fill in the remittance information, receipt address and timeout period on the Ripple application.
- The Interledger Module will pack this information and sent to the Ripple Account 1, Ripple Account 1 records the changed amount of currency in the escrow account 1 and sends the transfer certificate to the **Validator**

- For Bob, Company B fills in the Ripple application with information such as the remittance address and timeout period and broadcasts it on the Ripple network. At this time, the liquidity provider selected by A will transfer a certain amount of assets in B's currency from Ripple Acc.3 to Ripple Acc.2 in advance, then send the transfer certificate to the **Validator**
- Validator checks the two transfer certificates; after the verification is passed, the ILP ledger will be liquidated simultaneously according to the Hashed Time Lock Agreement.
- The final step is when liquidation completed, Ripple will synchronize all account changes through an interledger module, thus realizing the cross-ledger transactions.

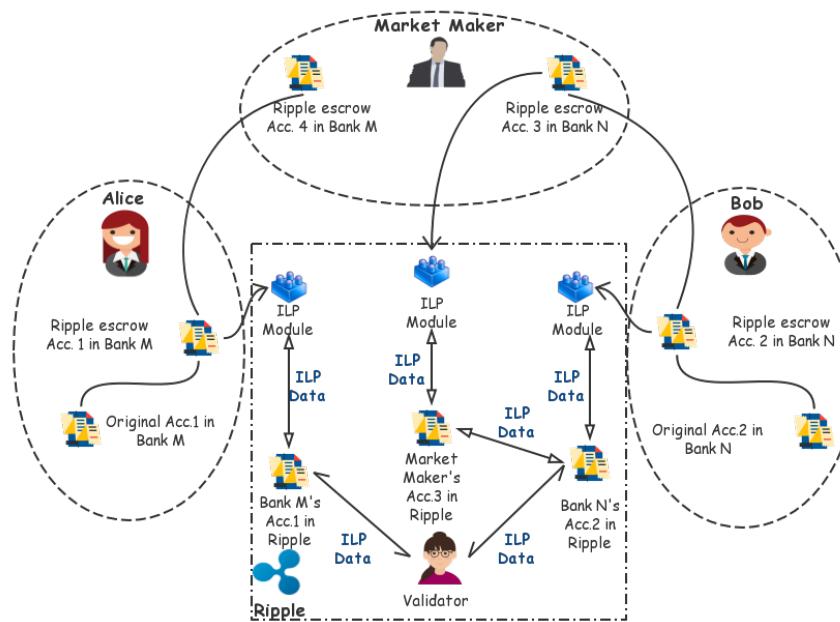


Figure 3.9: ILP example process

3.2.2.2 Wanchain

Wanchain [11] is a cross-chain platform project initiated in 2016. It is a heterogeneous cross-chain framework that implements cross-chaining based primarily on

distributed notary scheme. This model mainly uses cryptography "Secure Multi-Party Computation" and "Threshold Key Sharing Scheme" to achieve the Authenticator distributed signature.

Wanchain provides the infrastructure for asset cross-chain transfer channels for different blockchain networks, realizing the transfer of assets between Wanchain and original chain. Wanchain3.0 now launches bridges from Bitcoin to the Ethereum network. The transaction reliability verification is completed by multiple Storeman of Wanchain. The following figures 3.10 and 3.11 are shown the transfer process between Wanchain and Ethereum.

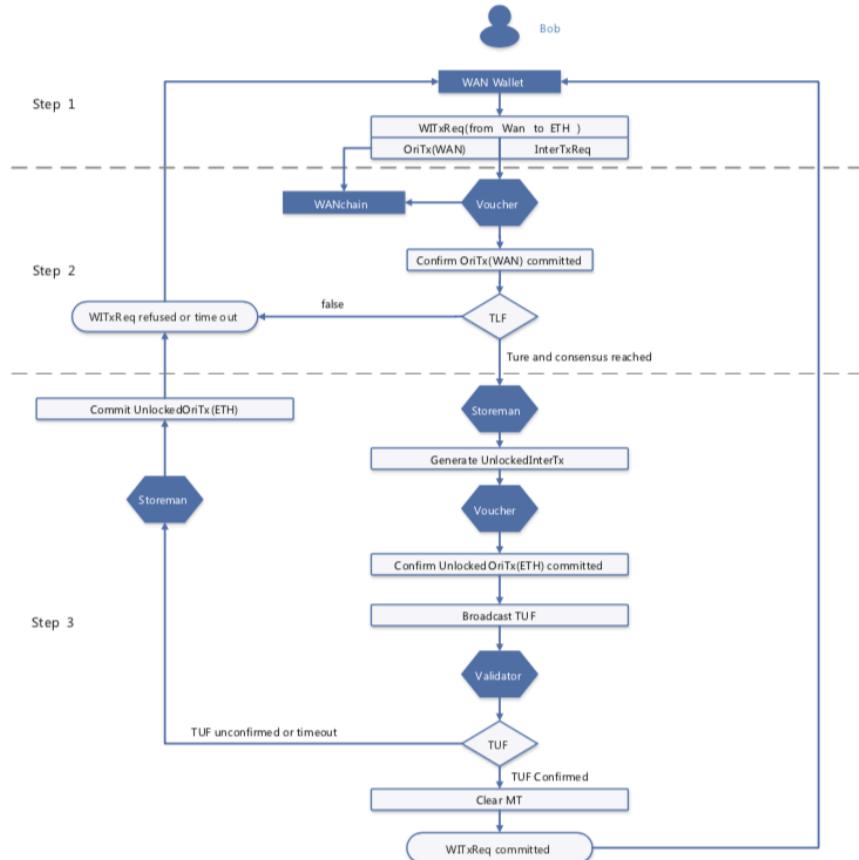


Figure 3.10: Data transfer process from Ethereum to Wanchain¹

¹Image courtesy of Wanchain white paper [11]

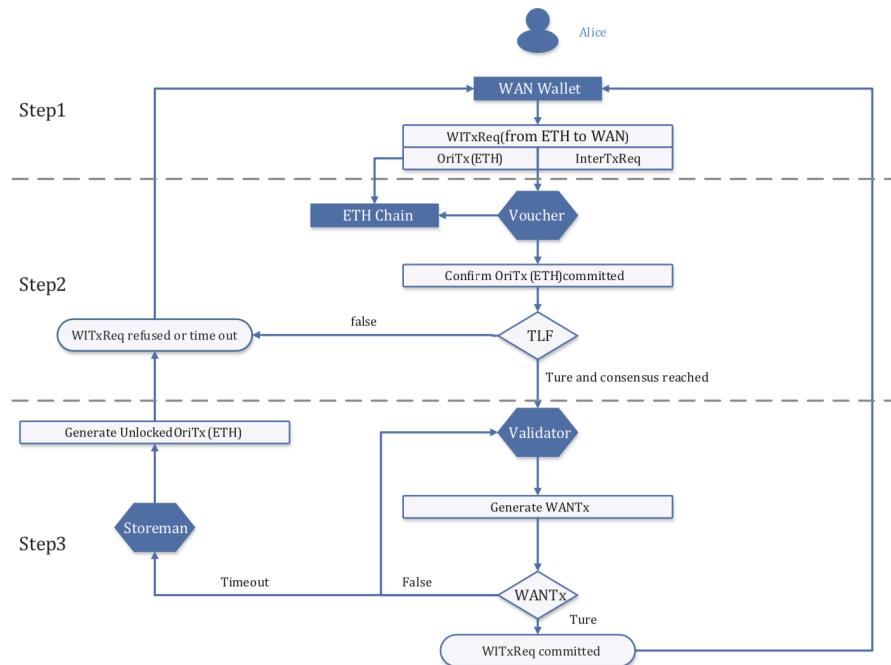


Figure 3.11: Data transfer process from Wanchain to Ethereum²

1. The token of the user in the original chain will be sent to Wanchain in the locked account, and Hash Time Lock locks the transaction;
2. After the **Voucher** verified the transaction on the original chain, the **Storeman** will initiate a cross-chain contract transaction on the Wanchain, and transfer the mapping token in Wanchain to the user's cross-chain account on Wanchain, and locked;
3. After the user's wallet detects the transaction locked by the cross-chain contract, release the Secret to the cross-chain contract;
4. Storeman obtains the control of the original chain token through the secret number, thus achieving confirmation of the original chain transaction.
5. If the user does not release the secret number within the scope of the hash time lock, the hash time lock expires

²Image courtesy of Wanchain white paper [11]

6. The transaction of the post-cross-contract is automatically invalidated, and the user regains control of the original token.

In Wanchain, when Storeman locked an account, the private key of the locked account is scattered into multiple pieces and sent to Storeman, and it requires more than a certain percentage of Storeman to complete the signature before the final confirmation. To avoid conspiracy, Storeman has to pay a certain amount of token to participate in the verification in case of doing evil. To ensure atomicity, Wanchain uses a hash time lock to lock cross-chain transactions, ensuring that no user or Storeman will complete a one-sided transaction.

Since the Wanchain mechanism does not change the original chain. It is necessary to adapt the development according to the characteristics of the original chain, which is also the difficulty of heterogeneous cross-chain solution. The transaction speed is affected by the confirmation speed of the original chain.

3.2.3 Relay Scheme

3.2.3.1 BTC-Relay

In 2016, the BTC-Relay released by the Consensys team is the most classic relay cross-chain solution, enabling cross-chain transactions between Ethereum and Bitcoin, as well as realizing Ethereum's DApp applications to support BTC payments. Since Bitcoin scripts are non-Turing complete and difficult to support complex applications, BTC-Relay only implements one-way cross-chain from Bitcoin to Ethereum.

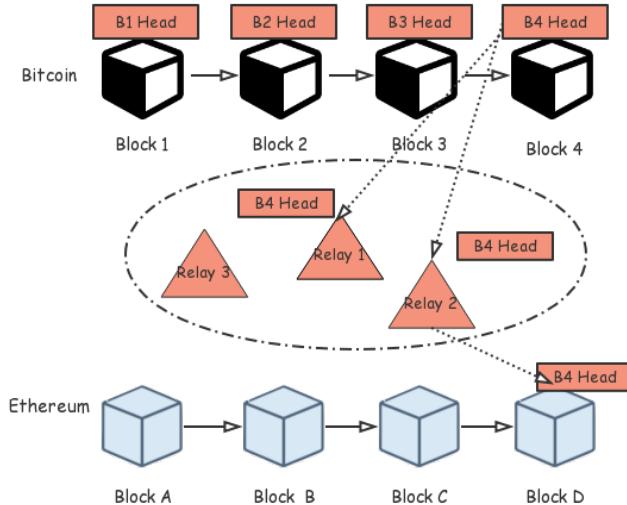


Figure 3.12: BTC-Relay cross-chain process diagram

Like Figure 3.12 shows, BTC-relay itself is a smart contract for Ethereum. The function of the contract is to verify certain transactions on Bitcoin and provide verification information to other DApp users on the Ethereum. Relay is a group of users who obtain block header data from Bitcoin and has the account address of the Ethereum network. The Relay that submits the block header data to the BTC-Relay contract as soon as possible can get the ETF transaction fee reward. After obtaining the block header data, the BTC-Relay smart contract can verify a transaction according to the principle of SPV proof. When a transaction in the Bitcoin network does occur, it can trigger the specific transaction or smart contract execution of the Ethereum network.

3.2.3.2 Cosmos

Cosmos [9] is a cross-chain platform project initiated by the Tendermint team in 2017. It supports the modular establishment of Cosmos isomorphism chain and also supports the external heterogeneous chain through Bridge. Its most important feature is that all the chains in the Cosmos system are isomorphic and can more easily support the flow of assets across the chain. All the zones share a set of network protocols, consensus mechanisms, and data storage methods. Assemble the new Zone blockchain through the API interface.

The overall architecture of Cosmos network are shown in Figure3.13:

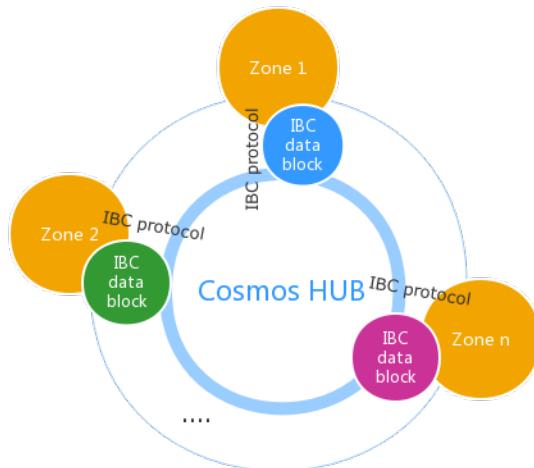


Figure 3.13: Cosmos network architecture

There are many Zones connected to the Hub (Hub is a chain, and each Zone is also a chain). Cosmos Hub maintains a multi-asset distributed ledger and masters the asset status of all the Zones that connected to it. Each Zone will synchronize the state of the Hub, but the communication among Zones can only be done indirectly through the Hub. Each cross-chain asset transfer requires a standard successful confirmation from Zones and Hub. The information is transmitted between zones through packets based on the IBC (Inter-Blockchain Communication) protocol. The blocks in a space pack the data to be delivered into standard IBC data packets and finally complete the transmission through the network layer UDP or TCP protocol.

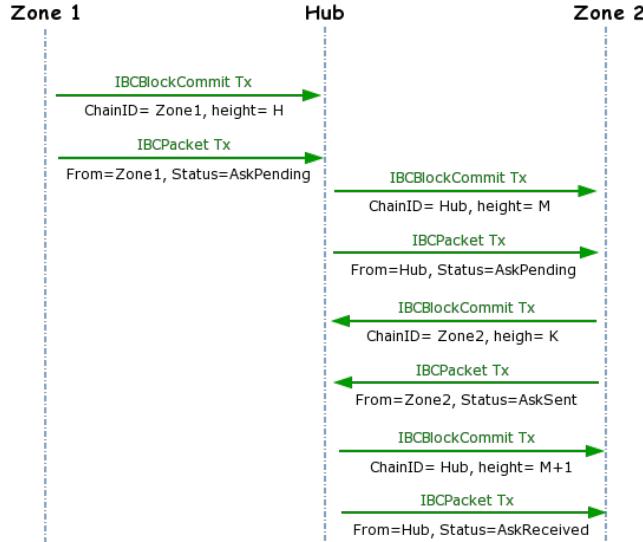


Figure 3.14: IBC sequence diagram

1. Zone 1 initiates an IBCBlockCommitTx transaction, passing the new block header information (including all Validator's public key) to the Hub;
2. Zone 1 initiates an assets transaction;
3. Zone 1 verifies the transaction;
4. Send the valid transaction into the message queue for Hub;
5. Zone 1 listens to a new message in the queue, which generates Merkle proof, and sent to Hub as IBCPacketTx's Payload. (In each space there is an independent third-party relay program that produces Merkle Proof from the original chain and assembles it into a Packet and initiates the transaction, passing it to the receiving chain);
6. Hub verifies that Merkle Proof is valid, if it is valid, send a message to Zone2 (The process of sending a message to Zone2 by Hub is the same as steps 1 6);
7. Zone2 verifies that Zone1 is a valid transaction after receiving the Hub message. Send a message to Hub confirming that it can acquire assets from Zone1;

8. The Hub sends a message to Zone2, sends the asset to Zone 2, and the asset transaction is completed. Due to an attack or network error during the transfer process, it is possible for the message sent by Zone 2 to the Hub to be lost, as shown in the figure 3.15 below, after waiting for a period of time, the Hub sends a message telling Zone1 that the current transaction is timeout and the transaction fails.

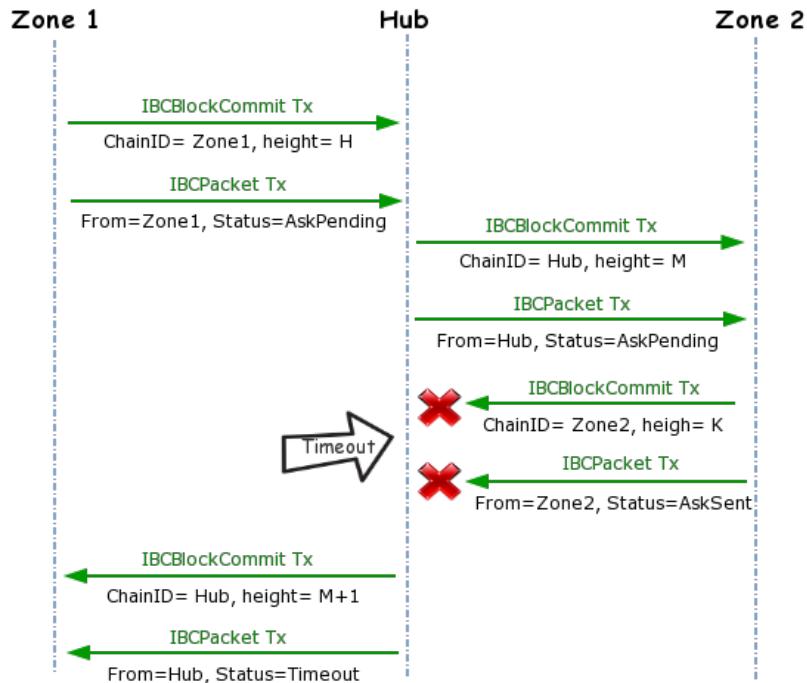


Figure 3.15: IBC sequence diagram, timeout

Cosmos based on **Tendermint** consensus, which is the combination of PBFT and PoS, it improves the processing power of Cosmos Network. The cross-chain transaction between Cosmos and other heterogeneous blockchains need to be carried out through Cosmos Bridge, Bridge-Zone will be responsible for the docking with the original chain, including the confirmation of the original chain transaction, create or destroy the corresponding tokens.

Overall, Cosmos is a blockchain development framework, allowing developers to focus on their own business without having to consider the underlying technology of the blockchain so the plug-in design can use Cosmos as needed.

3.2.3.3 Polkadot

Polkadot [12] is a cross-chain project supported by the Web3 Foundation. Creating a scalable, heterogeneous, multi-chain architecture designed to address blockchain interoperability, scalability, and security.

The overall architecture of Polkadot is shown in the figure below, consisting of a Relay Chain, Parachains and Bridges.

Role	Description
Relay chain	The central system of the Polkadot network, which coordinates the consensus and transactions between the chains, records account information and transaction status;
Parachains	Built by developers to collect and process transactions, and transfer to Relay chain;
Bridges	Connects other heterogeneous blockchains (such as Ethereum) with Polkadot network.
Collators	Collects user transactions and submitting to Validators
Validators	Validates and broadcasts transaction data committed by Collators, verifys blocks and paying deposits
Nominators	Investment deposits selected for trust Validators
Fishermen	Supervises the evil behavior in the network

Table 3.1: Main roles in Polkadot

Collators and Validators are the performers of the main transaction when doing cross-chain operations, while Nominators and Fishermen are the participants in maintaining system trust.

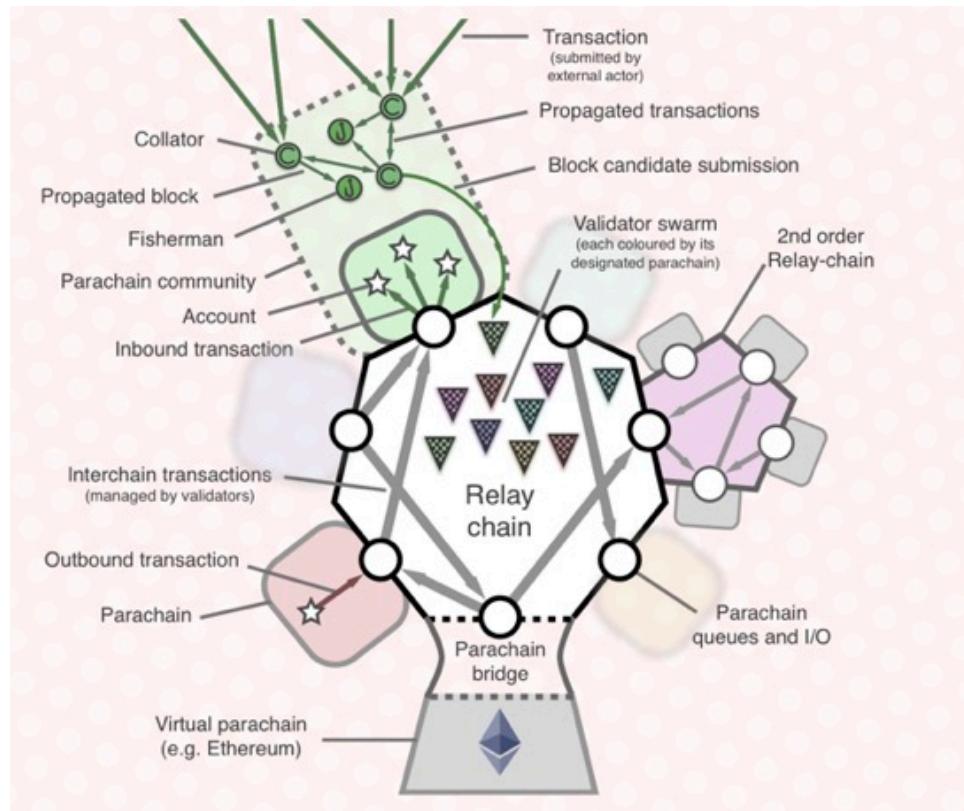


Figure 3.16: Polkadot network architecture ³

Parachain can share trust of the entire Polkadot network, but it also gives a certain confirmation right to the Relay chain; when the user's transaction information on Parachain A is transmitted to Parachain B, the process is as follows:

1. The initiated transaction is sent to the Collator on Parachain A;
2. Collator validates the transaction and packs it into the block;
3. Collator submits the block and state transition proof to the Validator on Parachain A;
4. The Validator verifies the block it receives that contains only valid transactions and pays a certain amount of deposit;
5. After enough Nominators have paid a deposit for the Validator, and they broadcast the block to the Relay Chain;

³Image courtesy of Polkadot white paper [12]

6. The transaction of data has been executed.

Unlike Cosmos, Polkadot supports cross-chain interoperability between heterogeneous chains. It does support not only asset transactions but also data transfer. Its system complexity and implementation are more difficult than Cosmos.

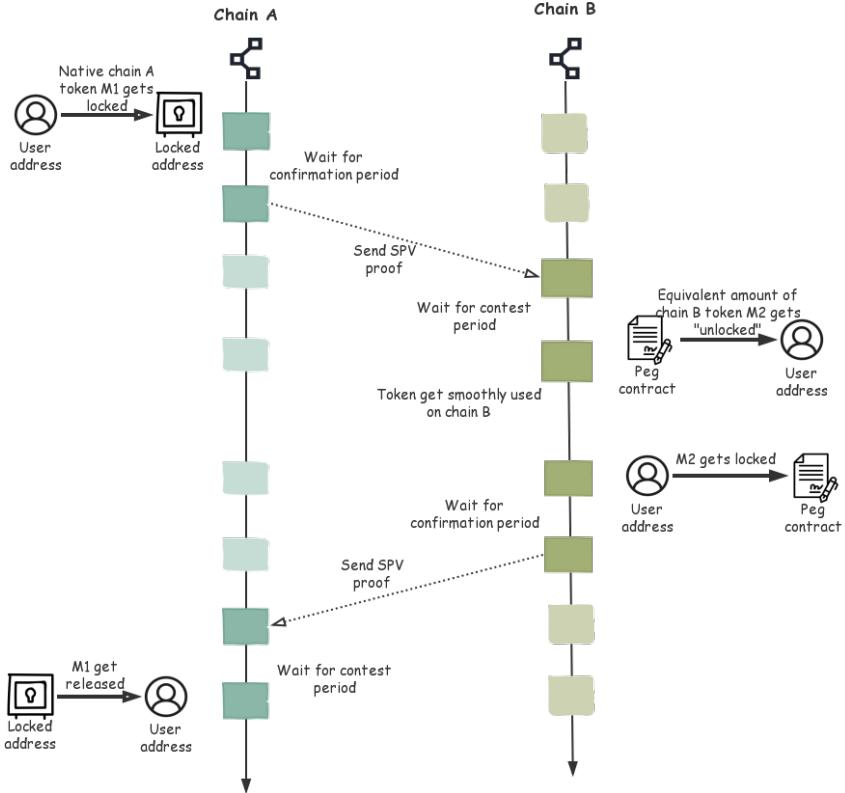
3.2.4 Sidechain Scheme

3.2.4.1 Plasma

As the second-layer expansion framework of Ethereum, Plasma has been proposed by JOSEPH POON (founder of Lightning Network) and VITALIK BUTERIN (founder of Ethereum) in 2017 [13]. The first thing to be clear is that Plasma is essentially a set of frameworks instead of a separate project. It provides an off-chain solution for a variety of different project real projects.

Layer 2 expansion of blockchains often known as off-chain expansion, similar to lightning network, this kind of expansion scheme does not need to modify the underlying protocol of the blockchain, but by transferring a large amount of frequent calculation work “off-chain”, and submitting the calculation results to the “main chain” guarantee its finality as we discussed previously. Plasma works like blockchains in blockchains, where anyone can create different Plasma on top of the underlying blockchain to support different business needs.

As an example of sidechain, Plasma is derived from the general concept of symmetric 2-way pegged scheme to realize the transfer solution. The overall process of asset exchange is shown in figure 3.17.

Figure 3.17: 2-way pegged sidechain diagram⁴

1. When chain A wants to transfer the asset to chain B, it first needs to initiate a transfer transaction Tx1 (chain A's locking addr1 + chain B's receiving address addr2), and the asset M1 is locked on the addr1.
2. After Tx1 transaction is submitted, it is necessary to wait for a *confirmation period* so that there are enough blocks and calculations to ensure that the cross-chain transaction Tx1 is confirmed, reducing the impact of refactoring on cross-chain transactions.
3. After the confirmation period, the **SPV** certificate containing Tx1 will be sent to chain B. B knows that chain A has indeed initiated and locked asset

⁴SPV to verify that the transaction exists (recognized by other nodes on the blockchain)

M1, so it generates a corresponding amount of M2 on chain B according to a certain ratio. The value of M1 is transferred to M2 means the assets on chain A are transferred to chain B.

4. After M2 generated on chain B. It is necessary to wait for the *competition period* before unlocking M2 to avoid double-spend attack in chain A reconstruction.
5. After unlocking, M2 can be used freely on chain B.
6. The process of a transaction from chain B to chain A is similar to previous steps.

Plasma supports multi-level sidechains and uses MapReduce mode to perform parallel computing, which greatly improves sidechain performance. The block header and hash data of the side chain will be sent to the main chain, and *Proof of Fraud* can be used to ensure the correctness of the sidechain transaction.

3.2.4.2 OneLedger

As one cross-chain consensus protocol, OneLedger [14] uses sharding and improved practical Byzantine fault-tolerant consensus. By creating sidechain, it can easily realize cross-chain interactions between individuals or business in OneLedger. OneLedger defines a three-layer consensus protocol to integrate different blockchain applications more efficiently.

Different from what we have discussed before, OneLedger as a sidechain, it realizes the synchronization of assets and values between the main chain and sidechain by applying multi-sig federation and drive-chain.

A drivechain [15] gives custody of the locked coins to the miners, allowing them to (algorithmically) vote on when to unlock coins and where to send them. As Figure 3.18 shows:

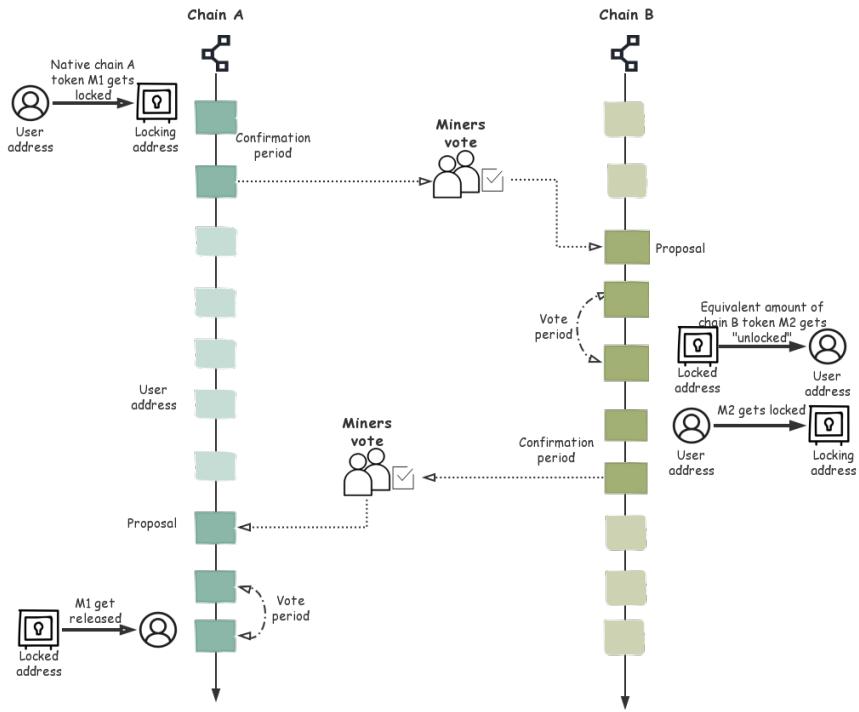


Figure 3.18: Drivechain working diagram

According to OneLedger's white paper, the core of OneLedger is a set of consensus protocols that enable OneLedger to integrate different blockchain products effectively. As far as I understand that the protocol mentioned here is not a specific consensus protocol algorithm in the traditional sense, but a series of concepts and application scenarios. Among all three layers, I specifically studied the *Public Chain Consensus* which apply on the atomic transfer between blockchains through OneLedger Network on the base layer.

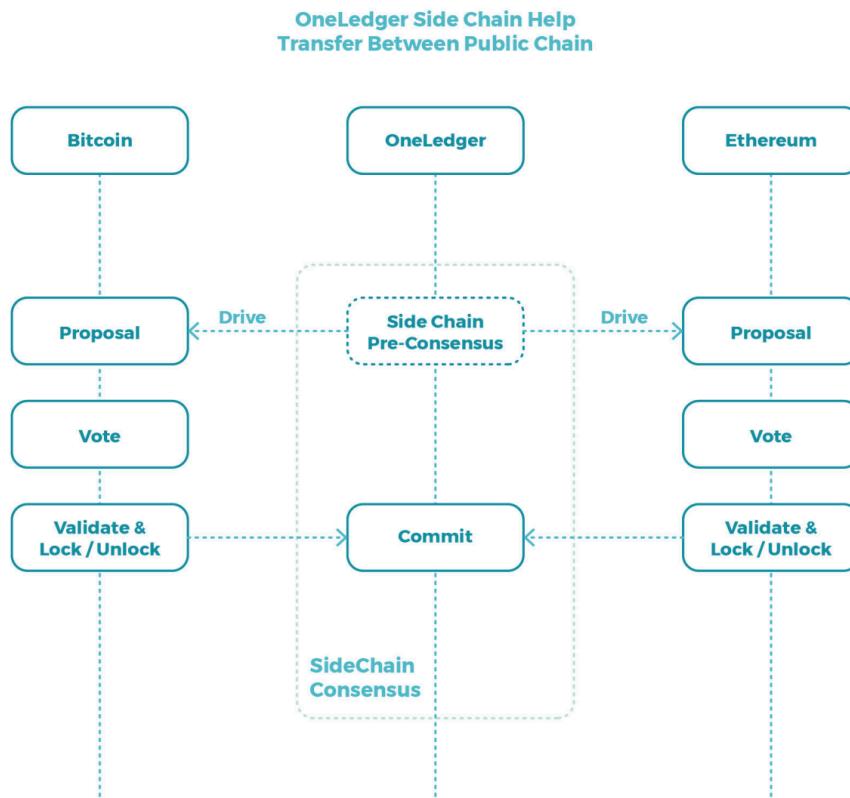


Figure 3.19: OneLedger sidechain architecture⁵

There are two steps in the sidechain consensus algorithm:

- **Round base pre-consensus:** Use to obtain a consensus proposal with more than 2/3 participants' votes.
- **Commit:** When the proposed block has reached a pre-consensus stage, it needs to drive to the public chain when necessary, and accept the verification process. Once both public chains accept the proposal, the new block will be officially 'committed' to the OneLedger network, and once more than 2/3 of the participants complete the commits, the block is finalized.

3.2.5 PalletOne

To face with the lack of interoperability in the blockchain world, PalletOne aims to become the “IP protocol” of the blockchain network, which provides an operating

⁵Image courtesy of OneLedger white paper [14]

environment for decentralized applications via abstract interfaces. [16]

On the one hand, PalletOne encapsulates all underlying blockchains into adapters in the form of interfaces so that it could exchange values and data with different blockchains. On the other hand, the PalletOne virtual machine provides a secure and stable smart contract running environment for common programming languages. Developers can write cross-chain blockchain applications using their common development language without paying attention to the details of the blockchain. At the same time, PalletOne’s original jury mechanism and the Mediator mechanism of DAG Data Storage and DPoS enable both contract execution and data storage to be processed in parallel, enabling a high-performance “super-chain.”

According to their technical yellow paper, they try to address scalability issues, enhance user-friendliness, and transaction issues in different chains. However, the core problem that PalletOne solves is the problem of cross-chain interoperability. The most creative feature in PalletOne is the consensus mechanism, it is complicated and divide the network nodes into two roles:

- **Jury**

Instead of the traditional consensus model, PalletOne delegates the operation of smart contracts and the management of multi-signature accounts to the Jury. The Jury randomly selected by the candidate jurors will use the consensus to reduce the occurrence of network congestion, and at the same time use the deposit penalty mechanism to ensure that the jurors do not do evil.

- **Mediator**

Mediator in PalletOne is responsible for the overall security of the whole PalletOne network, so Mediator needs to ensure that all decisions are correct. It is similar to the function of miners in traditional blockchains to create blocks. The nodes elect Mediator by using DPoS consensus. Most of the work in PalletOne only needs to be done by the Jury without calling Mediator, in case that Mediator is causing bottleneck performance.

The following Figures 3.20 and 3.21 provided by the official yellow paper are the vivid examples of the cross-chain process between BTC and ETH with different endorsement.

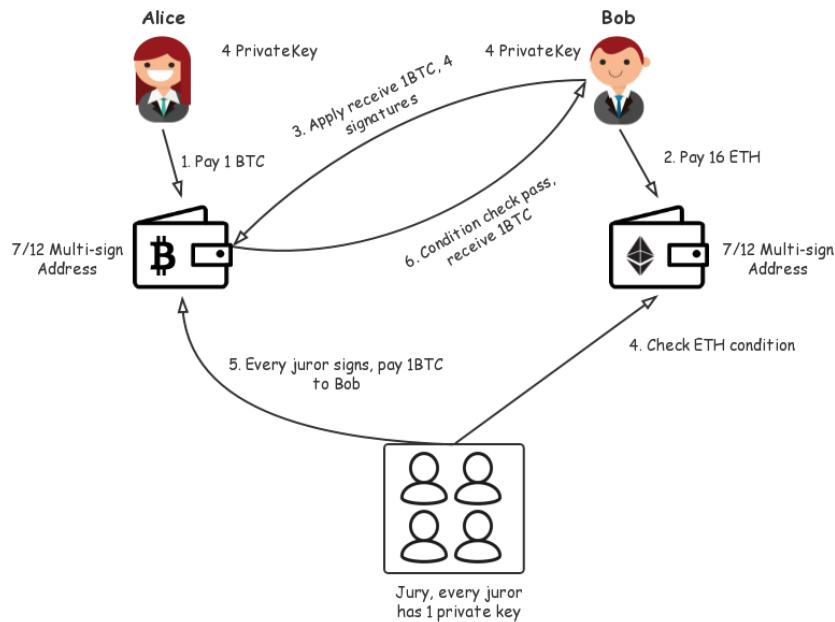


Figure 3.20: The normal process of executing the exchange contract (with Jury endorsement)⁶

1. Jury members and two user accounts first executed the proper contract that generated 7/12 multi-signature address for both crypto-currency.
2. Then both parties transferred a certain amount of exchange currency to corresponding multi-signature address.
3. Each party will initiate the signature request with their own four private keys to collect the tokens.
4. Jury verified the validity of both contracts and checked the status of two multi-signature address. If failed, go back to step 2.
5. After verifying the satisfaction, each juror will sign with the private key and invoke the transactions.

⁶Image courtesy of PalletOne yellow paper [16]

6. If one party of this transaction failed or timed-out, the Jury will terminate the contract, and there will be a compensation to the other by the one caused the fault.

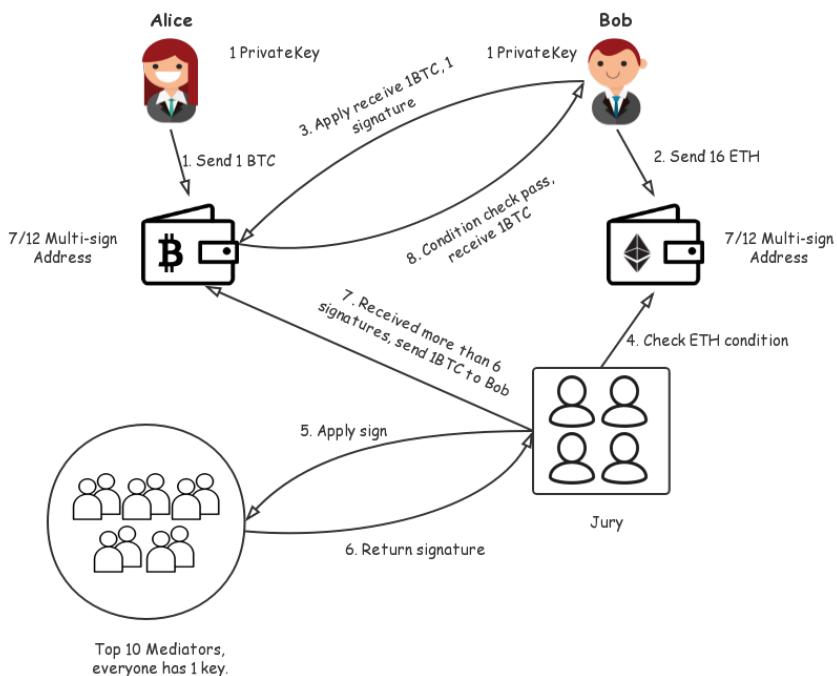


Figure 3.21: The normal process of executing the exchange contract (with Mediator endorsement) ⁷

The process of Mediator endorsement during the preparation stage differs from the previous one is, the signed contract will inquire the current votes of the Mediator and select the first ten nodes to form the 7/12 multi-signature address. Also, since the limited number of juror in the Jury endorsement, Mediator mode guarantees the persistence of a multi-signature wallet, so there is additional step for the Jury that has already reached the consensus to send the execution results to Mediators to verify, then the jury leader can broadcast the result once received the majority of Mediators' signature.

⁷Image courtesy of PalletOne yellow paper [16]

Among the projects I have studied so far, PalletOne has the most ambitious and powerful team with a purely technical project that does not have much community operation.

3.3 Summary

This Chapter has presented a thorough case study of 14 representative cross-chain projects and classified them into several working schemes. Through the official white paper and other documents, the preceding sections explained the working process and theory of them using diagrams to help the understanding. Including some comments and thoughts towards them. The total work of comparison will be shown as a table in Appendix A.

Chapter 4

Implementation and analysis

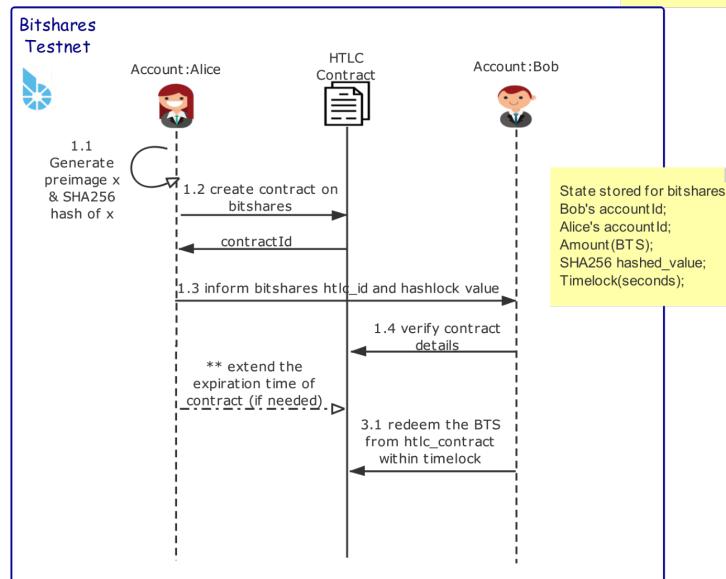
In the previous chapter, we mainly identified several subsets of cross-ledger communication schemes and explained the related projects in details. Nevertheless, clear logic levels do not necessarily ensure a good user experience or functional applications. Thus, to evaluate the performance of different protocols or platforms, it is necessary to design and implement appropriate approaches.

As we discussed in section 3.1, to realize cross-chain communication

4.1 HTLC Atomic-swaps implementation

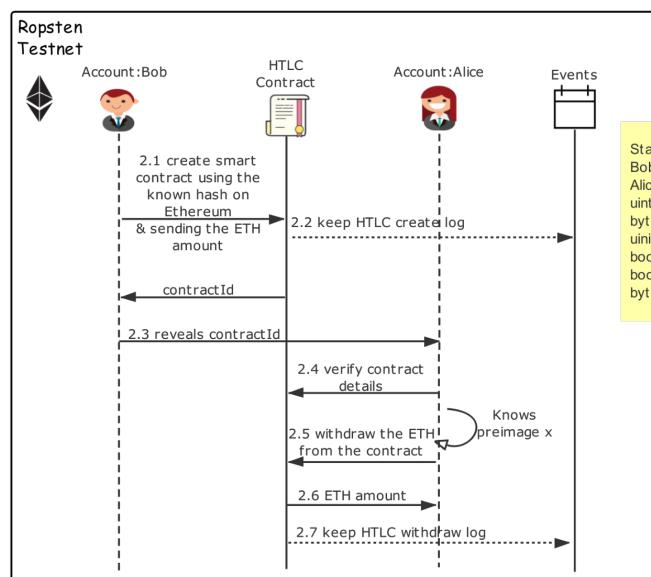
Scenario1, Trade initiated by BTS holder from bitshares

For both side to define:
 Bitshares addr.
 Ethereum addr.
 amount
 exchange rate(e.g. 1ETH=5060BTS)
 time-lock difference
 hash algorithm



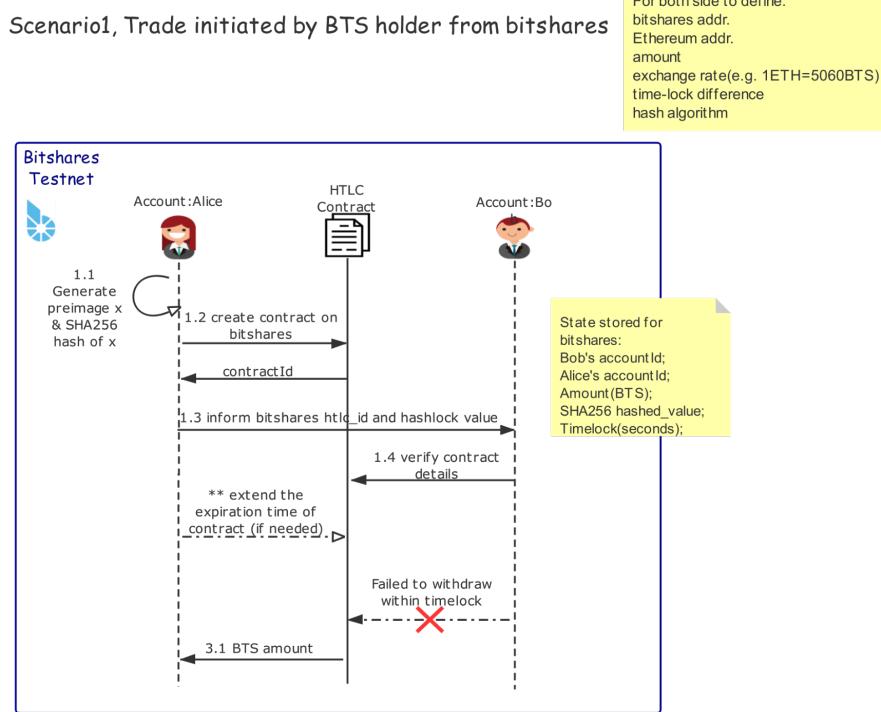
Methods:

- 1.2 htlc_create(Alice, Bob, hash, amount, timelock, preimage)
- 1.4 verifyHTLC(htlc_id) check hashvalue, amount and timelock
- 3.1 htlc_redeem(htlc_id, Bob, preimage)
- ** htlc_extend(htlc_id, Alice, seconds_to_add) can only be extended by contract creator

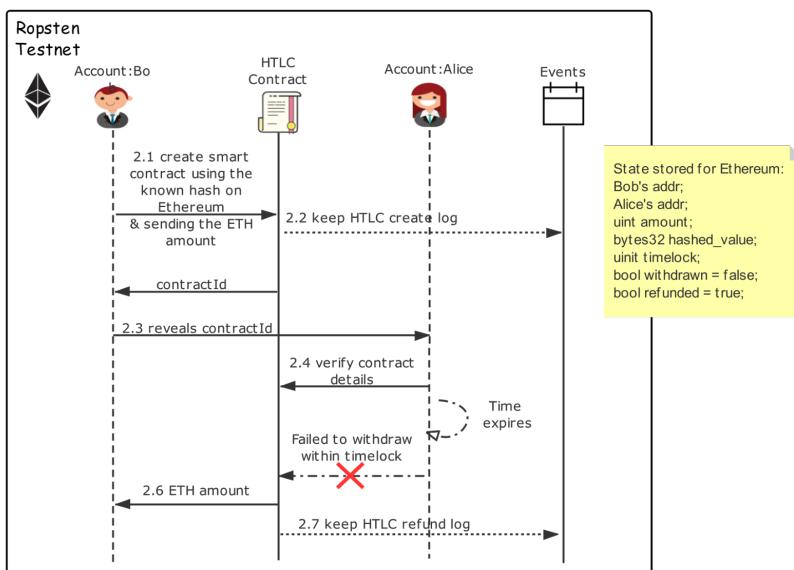


- 2.1 newContract(Alice, hash_value, timelock) the expiration time should be less than the contract Alice generated
- 2.2 LogHTLCNew(contractId, Bob, Alice, amount, hash_value, timelock)
- 2.4 verifyHTLC(htlc_id) check hashvalue, amount and timelock
- 2.5 withdraw(contractId, preimage)
- 2.7 LogHTLCWithdraw(contractId)

Figure 4.1: BTS->ETH swap diagram

**Methods:**

1.2 `htlc_create(Alice, Bob, hash, amount, timelock, preimage)`
 1.4 `verifyHTLC(htlc_id)` check hashvalue, amount and timelock
 ** `htlc_extend(htlc_id, Alice, seconds_to_add)` can only be extended by contract creator
 3.1 Contract will automatically return the refund amount of BTS to sender account



2.1 `newContract(Alice, hash_value, timelock)` the expiration time should be less than the contract Alice generated
 2.2 `LogHTLCNew(contractId, Bob, Alice, amount, hash_value, timelock)`
 2.4 `verifyHTLC(htlc_id)` check hashvalue, amount and timelock
 2.5 `refund(Bob, contractId)`
 2.7 `LogHTLCRefund(contractId)`

Figure 4.2: BTS->ETH swap diagram

4.2 Interledger switch API study

Cross-chain Swap diagram using Interledger

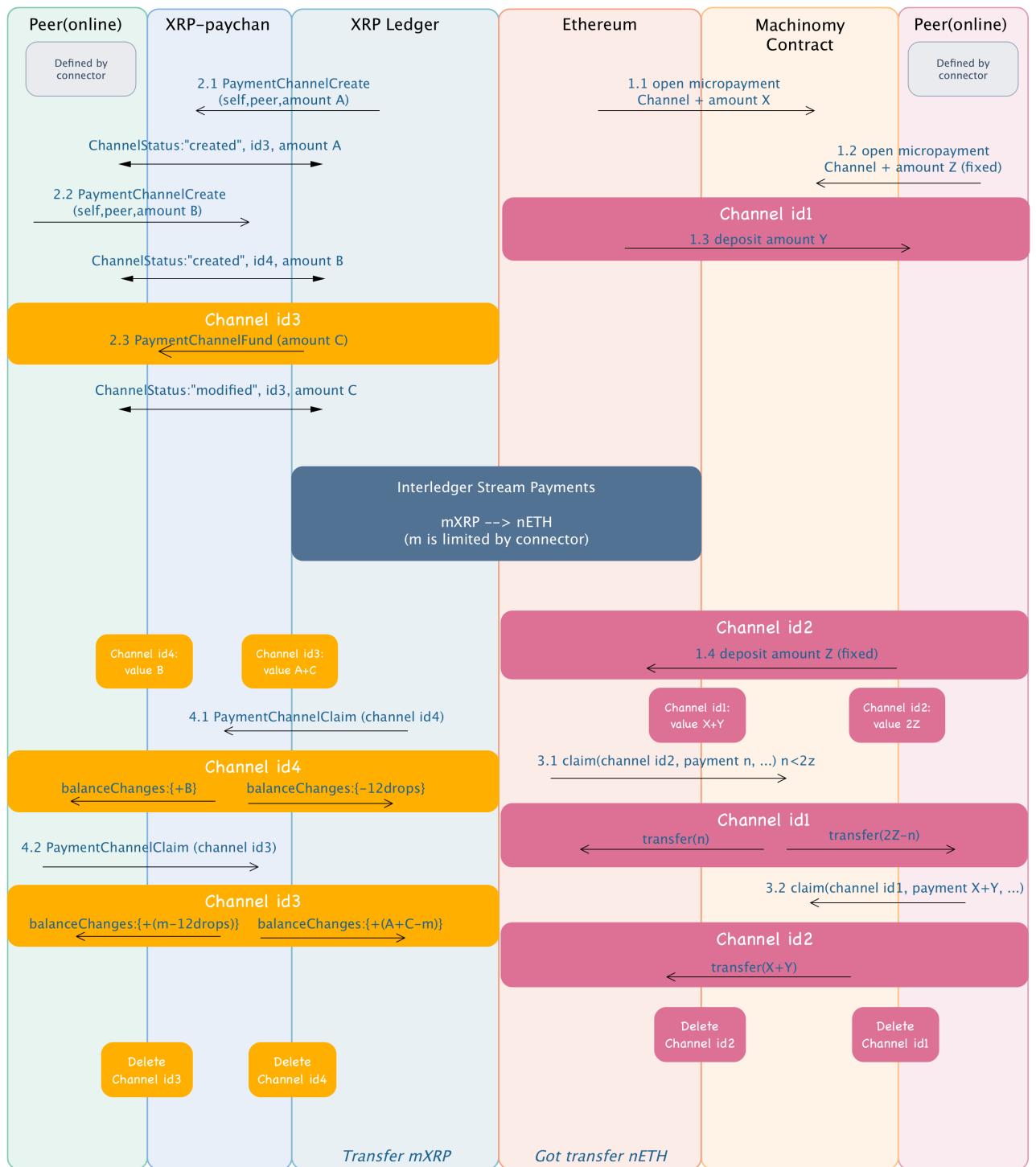


Figure 4.3: XRP->ETH

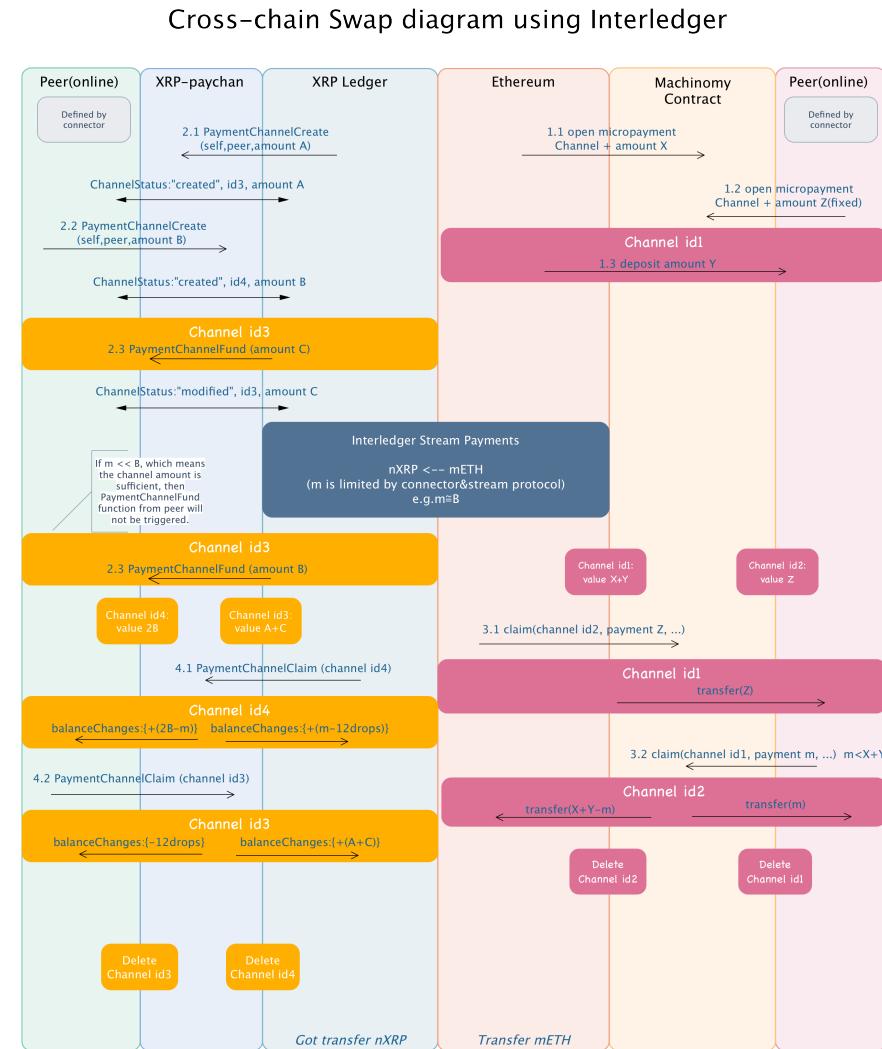


Figure 4.4

4.3 Summary

Write here...

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Write your conclusion here...

5.2 Future Work

Bibliography

- [1] S. Brakeville and B. Perepa, “Blockchain basics: Introduction to distributed ledgers,” *IBM, May*, 2016.
- [2] A. Osello, A. Acquaviva, D. Dalmasso, D. Erba, M. Del Giudice, E. Macii, and E. Patti, “Bim and interoperability for cultural heritage through ict,” in *Handbook of Research on Emerging Digital Tools for Architectural Surveying, Modeling, and Representation*. IGI Global, 2015, pp. 274–291.
- [3] V. Buterin, “Chain interoperability,” *R3 Research Paper*, 2016.
- [4] S. Thomas and E. Schwartz, “A protocol for interledger payments,” *URL https://interledger.org/interledger.pdf*, 2015.
- [5] J. Poon and T. Dryja, “The bitcoin lightning network: Scalable off-chain instant payments,” 2016.
- [6] Consensys, “Btc-relay documentation,” <https://media.readthedocs.org/pdf/btc-relay/latest/btc-relay.pdf>, september 10, 2016.
- [7] M. Herlihy, “Atomic cross-chain swaps,” in *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*. ACM, 2018, pp. 245–254.
- [8] Interledger.org, “Hash time-lock agreements,” <https://interledger.org/rfcs/0022-hashed-timelock-agreements/#simple-ayment-channels>.
- [9] irisnet, “Cosmos white paper,” <https://github.com/irisnet/irisnet/blob/master/WHITEPAPER.md>, 21st Feb, 2019.
- [10] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille, “Enabling blockchain innovations with pegged sidechains (2014),” *URL: tinyurl.com/mj656p7*, 2014.
- [11] J. Lu, “Building super financial markets for the new digital economy,” <https://wanchain.org/files/Wanchain-Whitepaper-EN-version.pdf>, 20th Jan, 2019.
- [12] D. Wood, “Polkadot light paper,” <https://polkadot.network/Polkadot-lightpaper.pdf>, 20-09-2017.

- [13] J. Poon and V. Buterin, “Plasma: Scalable autonomous smart contracts,” *White paper*, pp. 1–47, 2017.
- [14] Oneledger.io, “Oneledger white paper,” <https://oneledger.io/whitepaper/oneledger-whitepaper.en.pdf>, v2.0.
- [15] S. D. Lerner, “Drivechains, sidechains and hybrid 2-way peg designs,” 2016.
- [16] P. team, “Palletone yellow paper v1.0 beta,” https://pallet.one/doc/PalletOne_yellowpaper_en.pdf, May, 2018.
- [17] G. Verdian, P. Tasca, C. Paterson, and G. Mondelli, “Quant overledger whitepaper,” 2018.

Appendix A

Crosswise Comparison Table

The comparison table below summarizes 20 state-of-art cross-chain solutions, varying from protocols to platforms. The level of blockchain interoperability¹ is defined by Overledger develop team, using the following notation:

- 1-c-1: Two connected blockchains per time with a connector;
- N-c-N: Many connected blockchains per time with connectors;
- 1-1: Two connected blockchain connected per time without a connector;
- N-N: Many connected blockchains per time without connectors.

¹This concept is adopted from Quant Overledger White paper [17]

Table 1: Summary of 20 Cross-chain Projects

Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
Ripple	Ripple aims to build a global payment network based on blockchain, which proposes the Interledger protocol for establishing links with traditional financial institutions.	Notary	ILP	Platform	1-c-1	Most situation	Yes
BTC-Relay	In 2016, the ConsenSys team introduced BTC-Relay, the most classic relay cross-chain solution that enables cross-chain transactions between Ethereum and Bitcoin, as well as realizes Ethereum DApps to support BTC payments.	Relay	PoS	Blockchain	1-c-1	Asset exchange	No
Cosmos	Cosmos is a cross-chain platform project initiated by the Tendermint team in 2017. It supports the modular establishment of Cosmos isomorphism chains and also supports the external heterogeneous chain through Bridge.	Relay	Tendermint	Blockchain	N-c-N	Decentralized exchange	No

Continued on Next Page

Continued from Previous Page							
Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
Polkadot	Polkadot is a cross-chain platform project initiated by Parity Technologies that links the blockchain networks through relaychains and parachains, and links other chains outside the Polkadot network via BridgeChain.	Relay	PoA	Blockchain	N-c-N	Scalability of decentralized computation	No
	Lightning Network is a project running on Bitcoin. There are two main technical points, one is the RSMC (Recoverable Sequence Maturity Contract) and the other is the Hash Time Lock Contract. The former solves the problem of confirmation of the chain transaction, and the latter solves the problem of the payment channel.	Hash Timelock	PoW	Layer 2	1-1	Real-time payment	No
Aion	Aion aims to build a multi-tiered cross-chain platform that supports cross-chain interoperability of heterogeneous chains and connects the blockchain systems by connecting networks and bridges.	Relay	DPoS+ PoI	Platform	N-c-N	Information exchange	No

Continued from Previous Page							
Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
ICON	ICON is committed to building a cross-chain network that connects all types of blockchain systems, enabling DAPPs to be interconnected across all types of blockchains. ICON handles cross-chain transactions primarily through a notary mechanism.	Notary	LFT	Blockchain	N-c-N	Most situation	Yes
Wanchain	It is a cross-chain platform project that provides a cross-chain platform for interoperability of existing heterogeneous chains. It adopts a distributed signature notary mechanism to protect the accuracy of cross-chain transactions.	Notary	WANPos	Platform	N-c-N	Asset exchange	With chainlink
Fusion	Fusion supports multi-platform cross-chain asset transfer, using a distributed signature notary model for cross-chain transaction processing.	Notary	HHCM	Platform	N-c-N	Transactional	No
Chainlink	The ChainLink project was launched in 2014 to address an important issue of blockchain interaction with external data by creating a secure link to link real-world data to blockchain systems.	Notary	N/A	Protocol	N-N	Most situation	Yes

Continued on Next Page

Continued from Previous Page							
Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
ArcBlock	ArcBlock is a platform dedicated to the development and deployment of decentralized blockchain applications using the Open Chain Access Protocol to realize the cross-chain ability of most applications. It is a high-performance distributed cross-chain protocol, and has become the IP protocol of the blockchain world. By unifying the interface of each blockchain, it provides a multi-language smart contract running environment, uses randomly elected jurors to manage multi-signed public keys.	Relay	PoS	Platform	N-c-N	DApps	No
PalletOne	Quant Network creates the Overledger system that operates on top of blockchains, providing the interface for enabling the interoperability between blockchain and traditional network. It provides unlimited possibilities for blockchain data and applications.	Notary	DPoS+Jury	Protocol	N-N	Most situation	No
Quant Overledger		Relay	Protocol based	Platform	N-N	DApps	Yes

Continued on Next Page

Continued from Previous Page

Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
Plasma	Plasma is a side chain design model of Ethereum. Its main idea is to provide a model that can perform cross-chain transactions, and rely on the Ethereum blockchain to ensure its safety. The MapReduce mode is used to perform parallel computing, which greatly improves the performance of the sidechain.	Sidechain	PoS	Layer 2	1-1	Smart contract framework	No
Elastos	It is a public chain designed with a sidechain scheme. The main chain relies on the Bitcoin POW mechanism to ensure credibility without increasing energy consumption. Sidechain increases the computing power in the form of cluster services to avoid overloading the main chain.	Sidechain	AUXPoW +DPoS	Platform	1-N	DApps	No
Liquid	Liquid is a sidechain of the BTC and is a typical representative of the multi-signature notary mechanism. It is specifically designed to meet the BTC fast transfer needs of exchanges, market makers, and brokers.	Notary	PoW	Layer 2	1-1	Bitcoin-related	No

Continued on Next Page

Continued from Previous Page							
Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
OneLedger	OneLedger is a cross-chain consensus protocol that enables individuals or businesses using OneLedger to easily implement cross-chain interactions by creating sidechains. All transactions are performed on the sidechain, which greatly improves the efficiency.	Sidechain	3-layer consensus	Protocol	1-1	Decentralized exchange	No
Bytom	A multi-asset swap platform that operates on other chain assets. Developers can create a smaller version of relay of other chain, and API calls can be made from smart contracts to the relay of other chain to validate network activity for cross-chain communication.	Relay	PoW	Platform	1-c-N	Asset exchange	No
Aelf	Aelf adopts the architecture of sidechain, which mainly solves the problem of resource isolation. Different applications have different requirements on resources and performance. Therefore, their operation in separate spaces is the optimal asset allocation of system resources.	Sidechain	DPoS	Layer 2	1-1	Sidechain expansion	No

Continued on Next Page

Continued from Previous Page							
Project	Description	Scheme	Consensus	Type	Interoperability	Application	Compatibility with traditional ledger
Zipper	Zipper is a decentralized transit network that implements peer-to-peer messaging and transaction settlement across multiple blockchain networks. The Zipper Cross-chain Gateway (CCG) will be responsible for connecting all external blockchains.	Notary	BFT	Platform	N-c-N	Transactional	Yes

Appendix B

Code Pieces

I. Atomic swaps based on HTLC

Listing 1: Bitshares side

```
1  /*
2   * Here we need to specify the rpc_endpoint_url since I am using the
3   * local private testnet
4   * you can change to your own network ws address
5   * Keep the both side private key to perform the signature generate
6   * could be update to input parameters.
7   */
8 const rpc_endpoint_url = "ws://127.0.0.1:8090";
9 var sprivKey = "5Kecd9SoyHEYbSrUnadGzSokptuTWNMKi4M4CgXh7dSNSzLkNLq";
10 let spKey = PrivateKey.fromWif(sprivKey);
11 var rprivKey = "5Hwv9FXXrMd4o3FaHFJRLwuMmsihLz29bAGQYon4arK6ZzXCQhB";
12 let rpKey = PrivateKey.fromWif(rprivKey);
13 Apis.instance(rpc_endpoint_url, true).init_promise.then(
14   res => {
15     console.log("Successfully connected to BTS local test network.")
16     return ChainStore.init(false);
17   });
18 /*
19  * Instance the connection using the api
20  * since I use the local private testnet the network_name is not
21  * defined
22  * script adopted from bitsharesjs/examples/createHtlc.js
23  * check your environment before use
24  */
25
26
27 async function deployHTLC(sender, recipient, Hash, amount, timelock,
28   secret) {
29   let fromAccount = sender;
30   let toAccount = recipient;
```

```

31 let time_lock = parseInt(timelock);
32 let hash = Hash;
33
34
35 return Promise.all([
36   ChainStore.FetchChain("getAccount", fromAccount),
37   ChainStore.FetchChain("getAccount", toAccount)
38 ]) .then(res => {
39
40   let [fromAccount, toAccount] = res;

```

see line 29

Listing 2: Ethereum Side

```

1  * the first account id is 0
2  * the commented line are meant for other users
3  * till now is test only
4  */
5
6 // async function connectAcc(mnemonic, api_key, id) {
7 //   MY_SECRET_MNEMONIC = mnemonic;
8 //   env_api = api_key;
9 //   provider = new HDWalletProvider(MY_SECRET_MNEMONIC, env_api);
10 //   const web3 = new Eth(provider);
11 //   web3.eth.getAccounts().then( function(e) => {
12 //     getAcc(e,id);
13 //   });
14 // }
15
16
17 async function connectAcc(id) {
18
19   let account = await web3.eth.getAccounts();
20   address = account[id];
21   return address;
22 }
23
24 const sha256 = x =>
25   crypto
26     .createHash('sha256')
27     .update(x)
28     .digest()
29
30 const txLoggedArgs = txReceipt => txReceipt.events.LogHTLCNew.
31   returnValues
32 const txContractId = txReceipt => txLoggedArgs(txReceipt).contractId

```

II. Ethereum smart contracts

This contract provides a way to create and keep HTLCs for ETH. Detail protocol:

1. `newContract(receiver, hashlock, timelock)`- sender calls this to create a new HTLC and gets back a 32 byte contract id
2. `withdraw(contractId, preimage)` - once the receiver knows the preimage of the hashlock hash they can claim the ETH with this function
3. `refund()` - after timelock has expired and if the receiver did not withdraw funds the sender / creator of the HTLC can get their ETH back with this function.

Listing 3: Hashed Timelock Contracts (HTLCs) on Ethereum

```

1  /*
2   * 1) newContract(receiver, hashlock, timelock) – a sender calls this
3   *      to create
4   *      a new HTLC and gets back a 32 byte contract id
5   * 2) withdraw(contractId, preimage) – once the receiver knows the
6   *      preimage of
7   *          the hashlock hash they can claim the ETH with this function
8   * 3) refund() – after timelock has expired and if the receiver did
9   *      not
10  *          withdraw funds the sender / creator of the HTLC can get their
11  *          ETH
12  *      back with this function.
13  */
14  contract HashedTimelock {
15
16      event LogHTLCNew(
17          bytes32 indexed contractId ,
18          address indexed sender ,
19          address indexed receiver ,
20          uint amount ,
21          bytes32 hashlock ,
22          uint timelock
23      );
24      event LogHTLCWithdraw(bytes32 indexed contractId);
25      event LogHTLCRefund(bytes32 indexed contractId);
26
27      struct LockContract {
28          address payable sender;
29          address payable receiver;
30          uint amount;
31          bytes32 hashlock; // sha-2 sha256 hash
32          uint timelock; // UNIX timestamp seconds – locked UNTIL this
33          bool withdrawn;
34      }
35  }
```

Listing 4: Ethereum micro-payment channel – Machinomy

```

1  mapping (bytes32 => PaymentChannel) public channels;
2
3  event DidOpen(bytes32 indexed channelId, address indexed sender,
4              address indexed receiver, uint256 value);
```

```

4   event DidDeposit(bytes32 indexed channelId, uint256 deposit);
5   event DidClaim(bytes32 indexed channelId);
6   event DidStartSettling(bytes32 indexed channelId);
7   event DidSettle(bytes32 indexed channelId);
8
9   /*** ACTIONS AND CONSTRAINTS ***/
10
11  ///@notice Open a new channel between 'msg.sender' and 'receiver',
12  ///and do an initial deposit to the channel.
13  ///@param channelId Unique identifier of the channel to be
14  ///created.
15  ///@param receiver Receiver of the funds, counter-party of 'msg.
16  ///sender'.
17  ///@param settlingPeriod Number of blocks to wait for receiver to
18  ///'claim' her funds after the sender starts settling period (see
19  ///'startSettling').
20  ///After that period is over anyone could call 'settle', and move
21  ///all the channel funds to the sender.
22  function open(bytes32 channelId, address receiver, uint256
23  settlingPeriod) public payable {
24    require(isAbsent(channelId));
25
26    channels[channelId] = PaymentChannel({
27      sender: msg.sender,
28      receiver: receiver,
29      value: msg.value,
30      settlingPeriod: settlingPeriod,
31      settlingUntil: 0
32    });
33
34    DidOpen(channelId, msg.sender, receiver, msg.value);
35  }
36
37  ///@notice Ensure 'origin' address can deposit money into the
38  ///channel identified by 'channelId'.
39  ///@dev Constraint 'deposit' call.

```