A Survey on the Fundamentals, the Applications and the Security of Mainstream Cross-Chain Communication Protocols

Xiaoxi Celia Lyu
xiaoxi.lyu@duke.edu

Jason Liu
junyu.liu@duke.edu

Wangkai Jin
wangkai.jin@duke.edu

1. Introduction

The emergence of new chains (e.g. Solana, BSC) demands more frequent cross-chain communication than ever before. Our group finds that a survey of systematization of to-date cross-chain techniques is lacking since previous surveys either focus on a sub-field of cross-chain [1] or omit novel attack and defense techniques [2], [3].

In this SoK, we propose to present a systematization of knowledge on cross-chain techniques to fill in missing parts and complement previous works. Our survey can be divided into three parts, which are cross-chain communication protocols (CCCPs), their applications, and relevant attack and defense techniques.

First, we provide a review of prominent CCCPs (e.g., HTLC [4] and BTC Relay [5]). We discover that many recent developments in the area focus on standardizing cross-chain messaging/communication, such as Cross-Chain Interoperability Protocol (CCIP) by Chainlink [6] and Cross-Consensus Message Format (XCM) by Polkadot [7].

Second, we aim to discuss key technologies in cross-chain (e.g., side-chains, rollups) which breed various use cases in the real world. For example, CC-enabled asset transfers bring advanced trading experience (e.g., Central Bank Digital Currencies (CBDCs) [8]. Other potential use cases include trustless collaboration (e.g., in Supply Chain cooperation [9]), User Identification and data portability [10], etc.

Lastly, we aim to investigate novel attack and defense techniques for cross-chain designs. For instance, the vulnerabilities of CCs can be exploited by certain side-channel attacks (e.g. [11]) and there are numerous solutions that leverage various methods for building secure and privacy-preserving CCs (e.g., Formal Proof [12], Atomic Swaps [13], CP-SNARK-based verification [14]).

As cross-chain is a heated topic in the field, there are already several surveys that cover certain aspects of this technology. [3] presents a detailed summary of the existing CCCPs. [2] systematizes the interoperability of current blockchains and expands the discussion scope beyond transferring cryptocurrencies among different chains. [1] summarizes the existing challenges of current Layer 2 scaling and discusses the trade-offs of choosing different scaling techniques. The notable differences between our work with all the existing surveys are that 1) we include novel cross-chain bridge contracts and 2) we systematically discuss the attack and defense techniques for CCs, of which the knowledge is fragmented in the existing literature.

The organization of this SoK is outlined as follows. Section 2 introduces the mainstream Cross-Chain Communication Protocols. Section 3 presents a summary of current cross-chain-based applications (e.g., DEX. dApps, bridges). Section 4 categorizes existing attacks against cross-chain ecosystems and their corresponding defense mechanisms. Section 5 concludes our paper.

2. Cross-Chain Communication Protocols

This section introduces and summarizes the mainstream CCCPs. First, we list the protocols for mainstream public chains that give readers a taste of the variety and the diversity of the underlying protocols of current public chains (Section 2.1). Then, we present the general CCCP model and elaborate on the details of cross-chain communication among different chains (Section 2.2). Next, we introduce several standard CCCP schemes and present a brief review of the pioneering effort for these schemes, which are Notary Schemes (Section 2.3), Sidechains & Relays (Section 2.4 and Hash Time Hash-Locking (Section 2.5). Section 2.6 discusses our observations and takeaways for CCCPs.

2.1. Mainstream Public Chain Protocols

This section will brief the protocols for mainstream public chains to 1) familiarize readers with the current public chain ecosystems, 2) succinctly compare the differences between public chain protocols, and 3) point out the underlying values of efficient and effective CCCPs for coordinating the on-chain transactions among different public chains. By ”mainstream”, we mean public chains that currently have fair market capitalization share (e.g., top 10) as it indicates these public chains are in active daily usages and are championed by their communities. Bitcoin main chain is the first public chain that is proposed by Nakamoto in [15]. The protocol in Bitcoin is the Proof of Work (PoW) protocol that encourages participants to decentrally decide every block that is committed to the chain with their computation power. For extending the programmability of Bitcoin and supporting the development of decentralized applications, Ethereum [16] was proposed by Vitalik in 2013. The protocol for

Ethereum was primarily PoW, however, it shifts to Proof of Stake (PoS) for reducing transaction overheads and gas fees. Sharing similar goals to Ethereum, other public chains are built with a variety of core protocols, such as Cardano [17], Avalanche [18] and Solana [19], which either leverages PoS-based protocols (e.g., the Ouroboros for Cardano, the core protocol of Avalanche) or combines PoS with other novel algorithms (e.g., a combination of PoS and Proof of History in Solana). There are also public chains that focus on building multi-chain environments (e.g., Cosmos [20], Polkadot [21]), which also rely on fundamentally different protocols. We conclude by listing the protocols for mainstream public chains as it is sufficient to prove that the study of CCCPs is important for multi-chain cooperation and coordination as almost every chain is built on different protocols. Improperly handling cross-chain communication may expose severe vulnerabilities and paralyze corresponding public chain ecosystems, which we will discuss in Section 4. In the following section, we will introduce the general CCCP model that can effectively achieve cross-chain communication.

2.2. General Cross-Chain Communication Protocol

In this section, we introduce the main stages of a general Cross-Chain Communication Protocol (CCCP). For simplicity, we present a high-level illustration of cross-chain communication between two chains *Chain_B* and *Chain_B*. We will go through the general CCCP stage-by-stage in the following.

1) Setup: We first introduce the essential elements in a CCCP within the two chains. We denote the CCCP is established between Party *P_A* and Party *P_B* in *Chain_A* and *Chain_B* respectively. The CCCP is first initiated by *P_A*, who commit a transaction *TX_A* on *Chain_B* and the end of this protocol is either a transaction *TX_B* is successfully committed on *Chain_B* with the exchanged content from *Chain_B* or the protocol is aborted due to unexpected errors occurred during the process. The communication in the general protocol can be any standard cross-chain operation (e.g., digital asset transfer, token exchange).

2) Commit on *Chain_A*: As the protocol initiator, *P_A* will first commit the transaction *TX_A* to its local chain of *Chain_A* at a time *t*. The committed transaction will then be broadcast to other nodes within *Chain_A* and the block that contains *TX_A* would appear on all the honest parties of *Chain_A* after a limited delay time (i.e., synchronization delay, network delay).

3) Verify: Upon *P_A* commits the transaction, *P_B* will check the correctness of the by first checking the description of the transaction is correct and then check the transaction is indeed included in *Chain_A*. The first check is a common move introduced in Fair Exchange Protocol [22], [23], [24], which verifies the sender, receive and the type of the exchanged items are expected. The second check is always expected to be true after the limited delay time if *P_A* are honest.

4) Option 1 - Commit on *Chain_B*: If the verification is successful, *P_B* will commit a transaction *TX_B* on its local chain. Similar for *TX_A*. After a limited delay time, the block will appear on all honest parties in *Chain_B*, which indicates a successful cross-chain communication.

4) Option 2 - Abort: Aborting the protocol is the safest method for exception handling. This will occur when the verification fails or any unexpected error happens during the process. The modification of already committed transactions (e.g., *TX_A*) is reverted by submitting another transaction indicating the modification.

The general protocol presented here only describes the most naive cross-chain communication process. There are numerous optimizations such as protocols with two-phase commit designs [25], protocols that relay data across different chains (e.g., [5], [26], [16]), layer 2-enabled cross-chain communication protocols [27]. We will introduce these protocols in much detail in the following sections.

2.3. Notary Schemes

In notary schemes, a trusted entity (notary) facilitates cross-chain operations such as asset exchange and smart contract execution [28]. This mechanism of cross-chain communication is generally considered the simplest.

2.3.1. General approach of Notary schemes.

Notaries verify and supply information of one chain to parties of another chain; in the case of an exchange, this information may be that some party has tokens A and wants to exchange them for tokens B. Exchanges are considered decentralized if they provide a matching service between buyers and sellers but do not execute trades on their behalf [2]; otherwise, they are considered centralized as they manage everything in the process, including the funds. We will focus on decentralized uses of notary schemes in the following sections.

2.3.2. Limitations of Notary Schemes.

The main drawbacks of notary schemes stem from the added dependence on notaries. Notary-based protocols must therefore either trust the notaries or take care to guard against notary misbehaviors. Common defenses include combining with other cross-chain mechanisms like hash-locking3 [29] [30], and adding economic incentive for notaries to behave honestly [31].

2.3.3. Notary-based Protocols.

0x is a DEX protocol aimed at facilitating low friction peer-to-peer exchange of ERC20 tokens [32]. Its DEX is implemented as a smart contract on the Ethereum chain. Exchange transactions involve makers and takers. Makers place orders offering some token A to exchange for some token B, and takers can agree to take such an offer with token B and receive the corresponding token A. In this protocol, a maker approves the DEX to access its balance of token A and then they can place an order offering to exchange A for B at some rate. A taker may then discover

this order and wish to trade its B for A. The taker grants permission to the DEX to access its tokens, and the DEX performs the exchange after validation checks.

However, for liquid markets to emerge, there must be public locations where parties may postand discover orders. Due to the fast changing nature of markets, posting orders on-chain is expensive and not scalable. This is where relayers (notaries) come in. Relayers facilitate signaling between market participants by maintaining and propagating order books (off-chain). In exchange, relayers receives the fee specified in the order when it is completed on-chain. Note as relayers have no power to execute orders, takers must submit a fill txn to the DEX which will then perform checks and conditionally perform the exchange.

Interledger is a set of cross-ledger payment protocols [33] [34]; in its "atomic mode", a set of notaries use a Byzantine-fault-tolerant consensus algorithm to achieve consensus and ensure the atomicity of payments. This can be particularly useful in what Interledger describes as a payment chain. If parties X and Y desire to make an exchange of digital assets, but these assets exist on chains A and C with no direct link, then they can find intermediaries on ledger B that is directly linked with A (has notaries and exchanges between them) and with C. With this, one can determine a bundle of exchanges and then use a single consensus process for the entire exchange, ensuring atomicity.

Bifröst provides a modular blockchain interoperability API to help developers interact with different blockchains without needing to know the details of blockchain implementations [35]. It employs a notary scheme where notary nodes host the Bifröst application, which users can interact with using the API. Bifröst has three components: the API, the adapters, and the databases. The API exposes functions that store, retrieve, and migrate, allowing users to read/write strings to specified blockchains. The adapters convert user calls into blockchain transactions and submit them. The databases store necessary credentials for transactions; it also stores store the transaction hash after success, which can be later used to retrieve the stored data. Notaries have extensive control here as they host the Bifröst application and interact with blockchains on users' behalf.

2.4. Sidechains & Relays

Compared with notary schemes, relay chains can directly achieve interoperability themselves, not involving a notary to monitor information or trigger transactions across chains.

2.4.1. General approach of relays.

For two existing blockchains A and B, suppose chain B wants to learn some particular information or event on chain A, and also suppose that each block in chain A has a block header, which is used to identify an individual block in a blockchain. The "light client verification" technology used in relays consists of the following three verification processes:

(a) Proof of Work (PoW) verification: Suppose on chain A, there is a block header x on a fork on chain A.

```
function sendCrossChain(destChain, to, value) {
    if (balances[msg.sender] < value) throw;
    createEvent(destChain, {name: SEND, to: to, value: value});
    balances[msg.sender] -= value;
    crossChainBalances[destChain] += value;
}
function onReceiveEvent(senderChain, params) {
    if (params.name == SEND) {
        if (crossChainBalances[senderChain] < params.value) throw;
        balances[params.to] += params.value;
        crossChainBalances[senderChain] -= params.value;
    }
...
}
```

Figure 1. Code of relays

A smart contract on chain B verifies that the block header x on chain A is generated by enough amount of proof of work. In other words, in a proof of work model, it is more likely that this fork on chain A contains this block header x than any other fork on Chain A.

(b) Signatures verification: A smart contract on chain B verifies that the number of signatures signed in the block header should meet the quorum size. For example, in Practical Byzantine Fault Tolerant (PBFT), it would verify that there are at least 2t/(3t+1) replicas' signatures that have signed the block header.

(c) Transaction/information verification: Once pass PoW verification and signatures verification on the block header, the relay can then verify any desired transactions or information.

2.4.2. Abstraction and implementation of relays.

In general, the general approach of relays can be simply abstracted and implemented with a scripting language. For example, the following code is a sketch for "Fedcoin" with relays 1. The code is executed in the following order:

(a) sendCrossChain(destChain, to, value){...}: First checks if the sender has enough money to send. If the sender does not have a sufficient account balance, then throw it. If the sender has enough money to send, using createEvent(destinationChain, params), it creates an event on the destination chain, subtracts the value from the sender's balance, and increases the same value to the destination's balance.

(b) onReceiveEvent(senderChain, params){...}: After passing (a), check if the sender has enough money to send. If the sender does not have a sufficient account balance, then throw it. If the sender has enough money to send, increase the value to the destination's balance and subtract the same value from the sender's balance.

2.4.3. Limitation of Relays.

This "light client verification" technology is very powerful. However, it still has some limitations in practice. In fact, it is impossible for a smart contract on chain B to fully validate chain A and a smart contract on chain A to fully validate chain B at the same time. Therefore, in practice, small pieces of each chain would be feasible instead of

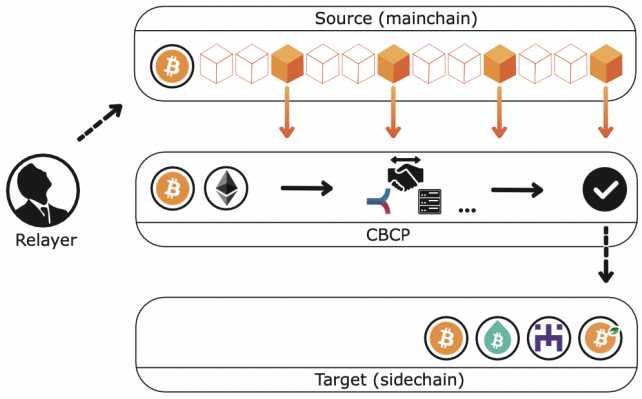

Figure 2. A general sidechain system [2]

fully validating each chain, where chain B only contains small pieces of chain A and chain A only contains small pieces of chain B at the same time.

2.4.4. Protocols of relays.

Some widely-used or newly-proposed relay-based protocols will be introduced as follows.

BTCRelay: It is the first live running relay contract implementation proposed by the Ethereum team in Sep 2016 [5], which is between Bitcoin and Ethereum. BTCRelay is a smart contract on Ethereum that can read and store block headers on Bitcoin chains. Note that BTCRelay is a one-way contract - Bitcoin cannot read or store block headers on Ethereum chains, as its scripting language is not sophisticated enough to do so [36].Figure 2 shows the workflow of a BTCRelay system:

(a) In BTCRelay, relayers are some special parties, which keep track of the block headers of the main chain (i.e. Bitcoin chains). Relays constantly submit Bitcoin block headers to the BTCRelay smart contract on Ethereum.

(b) In this way, a pool of Bitcoin block headers is hosted on Ethereum, which can be used to verify event/state information and transactions on Bitcoin chains.

(c) If a party on Ethereum chains requests a Bitcoin transaction, this Bitcoin transaction will be submitted to be verified by the smart contract that holds the Bitcoin headers' knowledge on Ethereum, and a small fee will be paid for this cross-chain transaction.

(d) The verified Bitcoin transaction is relayed to deployed Ethereum smart contracts.

Zendoo: It is a cross-chain transfer protocol proposed by the Horizen team in Jan 2020, powering a fully decentralized and verifiable network of blockchains. Zendoo supports a sidechain of any type, unlike in BTCRelay which only sidechains of Ethereum are supported. In Zendoo, parties in the sidechain can observe state of the mainchain (i.e. Horizen chain), but the main chain (i.e. Horizen chain) can only observe the sidechain via cryptographically authenticated certificates [37]. Zendoo introduces zk-SNARKS (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) to simplify the process of transferring assets from sidechains to Horizen, which is a cryptographic proof that enables verifiable cross-chain transfer protocol [38].

R-SWAP: It is a time-bounded atomic cross-chain swap protocol proposed in 2021, which is based on relays to outperform existing solutions. It can be adapted into many open blockchain systems such as Bitcoin, Ethereum or Cosmos. R-SWAP overcomes the shortcomings of other atomic cross-chain swap protocols such as hash-locking and relays, which are at risk of safety violation. R-SWAP uses a relay to verify transactions between two chains, which can achieve the contract auditing process thus reducing the risks of a client crash [39].

MAP: In 2021, the MAP team began to develop the MAP Protocol v2.0, which is the omnichain network for an interoperable Web3 built on a light-client and relay chain. This protocol has not been released yet [40].

Rangers: Rangers protocol is a relay-based new system proposed in July 2022, which creates a Multi-Chain Ethereum Virtual Machine (EVM) Network. Rangers Connector is a part of the Rangers protocol, which is responsible for the cross-chain connection between various public chains. [41]

2.5. Hash-Locking

Hash-locking enforces operations between two chains using hashlocks and timelocks, meaning each chain needs to know much less about each other than in notary schemes and relays [42].

2.5.1. General approach of Hash-Locking.

For two existing parties Alice on blockchains A and Bob on blockchain B, suppose Alice wants to exchange her 0.1 BTC for Bob's 50 ETH. This atomic swap (i.e., transfer) can be realized as follows 3:

Step 1: Alice generates a random secret s, and hashes s, $hash(s) = h$, yielding h.

Step 2: Alice sends h to Bob.

Step 3: Alice creates a smart contract with a timelock t_b to lock her 0.1 BTC. t_b is an upper bound in which the smart contract can be unlocked, i.e., Bob can unlock the smart contract up to t_b. Moreover, this smart contract will be unlocked only if Bob can provide the secret s. In other words, if Bob can provide the secret s within t_b, Bob will obtain 0.1 BTC from Alice. Note that the smart contract can be protected with the hash h, since the hash h here cannot be inverted - this is called hashlock.

Step 4: Bob creates another smart contract with a timelock t_a to lock his 50 ETH. $t_a < t_b$, and ta is an upper bound in which this smart contract can be unlocked, i.e., Alice can unlock the smart contract up to t_a. Moreover, this smart contract will be unlocked only if Alice can provide the secret s. In other words, if Alice can provide the secret s within t_a, Alice will obtain 50 ETH from Bob.

Step 5: Alice sends s to Bob within t_a to unlock Bob's smart contract and obtain 50 ETH from Bob.

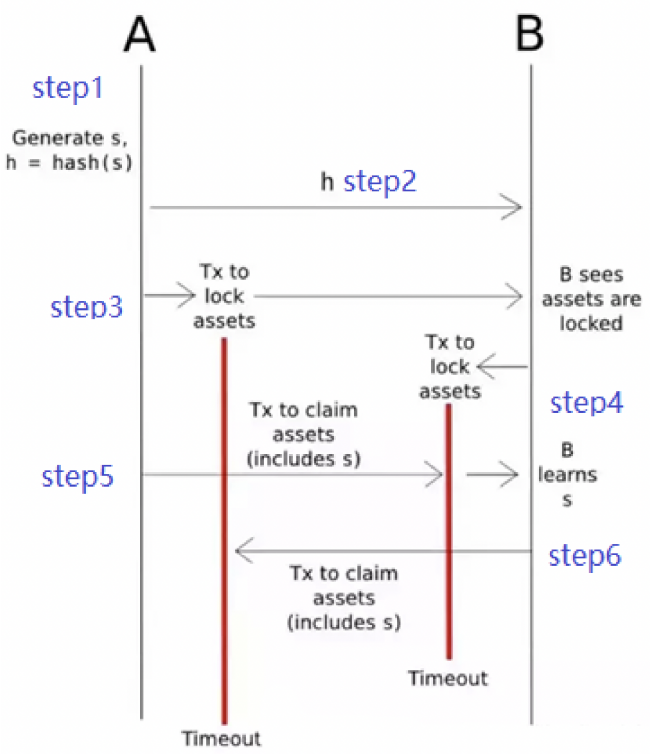

Figure 3. Hash locking mechanism [28]

Step 6: Bob records the s sent by Alice in step 5, and sends it to Alice with t_b to unlock Alice's smart contract to obtain 0.1 BTC from Alice. In this way, the asset exchange is completed. If any party times out, the locked asset will be recovered.

2.5.2. Limitation of Hash-locking.

(a) Financial safety violation: In the real world, exchange rate fluctuations exist all the time thus the financial safety guarantee is possible to be violated. For instance, if Alice is malicious, she can wait for $\frac{t_a}{2}$ time, and then decide not to send s to Bob if the exchange rate after $\frac{t_a}{2}$ moves in an unfavorable direction for her. In such a case, by participating in this hash lock, Bob has to give A a free option. It means that to exchange assets, honest parties have to try several times or suffer a financial loss, which weakens participants' confidence and ability to hedge risk in this market [28].

(b) Not useful for asset portability or cross-chain oracle: Hash-locking can only be used for atomic swap, but not asset portability or cross-chain oracle. It cannot be used for cross-chain oracle because the chain being read in cross-chain oracle is inherently passive, but the two chains in hash-locking are both inherently active. It cannot be used for asset portability because hash-lock guarantees that the total assets on each chain remain the same, no matter how many atomic swaps they have gone through, i.e., hash-lock cannot take assets from one chain to another chain [43].

2.5.3. Protocols of Hash-locking.

Some widely-used or newly-proposed protocols built on hash-locking will be introduced as follows.

InterLedger protocol (ILP): ILP is a protocol built on hash-locking proposed in 2016, where two different ledger systems can make transactions. Only parties involved in this transaction can track the transaction details, i.e. the transaction can be hidden from outsiders. ILP is now used for existing banking systems, as this protocol alleviates banks' main safety concerns. In general, banks do not use services from third parties to verify their transactions, as it may lead to the disclosure of internal transaction information [33][34].

Ligthning Network (LN): Bitcoin LN is a famous off-chain instant payment scheme proposed in 2016, which enables high-volume, fast-speed transactions on the Bitcoin network. Cross-chain transactions can occur without trusted third parties so long as the chains can support the same hash function [44].

XClaim: XClaim was the first generic framework for achieving financially trustless blockchain interoperability, which was proposed in 2019. Researchers have implemented XCLAIM between Bitcoin and Ethereum, and found it is faster and much cheaper to make a transaction than atomic cross-chain swaps. Also, XCLAIM can enable novel applications such as efficient multi-party swaps, without modification of most existing blockchains [45].

Multy-Party Hash Time Locked Contract (MPHTLC) Protocol: MPHTLC is an augmented form of Hashed Timelock Contract (HTLC) protocol proposed in Feb 2022, which overcomes the main shortcoming of the existing Hashed Timelock Contract (HTLC) protocol. Specifically, one limitation of HTLC is that it only applies to a network consisting of two parties in two different chains, each of the parties has sole ownership of a single asset. However, in the real world, a single asset can be jointly owned by different parties. To solve this problem, an augmented HTLC protocol for atomic multi-owner-and-asset exchanges named MPHTLC was proposed [46].

2.6. Discussions

Most existing relays are designed for Proof-of-Work (PoW) environments, which entails unsustainably high energy consumption, limited scalability, and high transaction fees. To mitigate this, a Transition to Proof-of-Stake (PoS) is needed. This is a developing research field focusing on protocols in PoS environments. For instance, Verilay, the first verifiable PoS chain relay scheme which enables validating PoS protocols that produce finalized blocks was proposed in 2022 [47].

Hash locking offers benefits for users to achieve cross-chain interaction without intermediaries. However, hash locking still has some limitations. One is adaptability. Hash locking can only be used between cross-chain exchanges which have the same hash algorithm or at least can understand the other's hash algorithm. Another one is popularity.

The number of wallets supporting Hash locking is still small nowadays.

3. Application

3.1. Cross-chain Bridges

Cross-chain Bridges are a set of designs that enable fast and cost-efficient asset transferring among different blockchains. As modern blockchian has distinctive features that lead to specialized usages, users prefer to use different chains which breeds the need for efficient user resource coordination among different chains. For example, people may prefer to buy NFTs on a Solana-based trading platform while playing some games that are deployed on Ethereum. Before the advent of Cross-chain Bridges, when users want to move all their assets to one account, they need to rely on a non-custodial wallet such as MetaMask and connect it to a centralized Exchange to move all their assets to the same account, which is expensive and time-consuming since the exchange relies on fiat currency to control the price.

Cross-chain Bridges can reduce the extra fees generated by converting to fiat currency in two general methods. First, without converting tokens into fiat currency, the tokens are wrapped by the cross-chain bridge platform. By using the wrapped token, the value of the token from the original blockchain can now be encapsulated into another token. The wrapped token technique is widely-used for ERC-20 protocols on the Ethereum chain. One of the most common wrapped tokens is the Wrapped Bitcoin (WBTC), which can be viewed as ERC-20-token Bitcoins that can be used by other blockchains for more diversified trading scenarios (e.g., decentralized finance, decentralized apps, and NFTs).

The second method uses liquidity pools for cross-chain bridge transfers. These pool-based bridges require setting up liquidity pools for the to-be-transferred tokens both on the origin chain and the receiver chain. For example, if a number of Ethereum is going to be bridged from Chain A and Chain B, an ETH liquidity pool should be set up on both chains respectively. During the bridge process, x number of ETH is locked up in the ETH liquidity pool on Chain A and the same amount of ETH is sent to Chain B from the liquidity pool on Chain B, thus reducing the transferring fees and time. Celer's cBridge [48], Umbria Narni Bridge [49], and Multichain [50] all use liquidity-pool-based methods for asset bridging and the price of locked assets in their liquidity pools are controlled by automated market maker mechanism[1]. Based on the liquidity pool methods, Wormhole [52] proposes a verification-based protocol to further enable the security of cross-chain bridges, where they use a set of *Guardians* as validators to observe and verify bridge behaviors. Another line of Bridges that require additional trust for the cross-chain bridge platforms are Optimistic Rollup Bridge and zkSync Bridge, which has more narrow applications scenarios (i.e., between Ethereum Layer 1 and Layer 2) and the design purpose is more like specialized optimizations for Ethereum which is different from the aforementioned cross-chain bridges. Recently, LayerZero [53] provides a trustless omnichain communication protocol which is expected to facilitate trust-free cross-chain communication and serves as the base layer for the entire blockchain system. The most significant feature of LayerZero is that it does not require *Middle Chain* to provide consensus or coordination for a bridge process, which is the standard design for bridges or blockchains that are designed for formulating multi-chain environments (e.g., Polkdot). Instead, LayerZero deploys light nodes on original and receiver chains to store the ledger transaction history and validate swapping transactions.

1. We would refer interested readers to [51] to understand the working principles for the automated market maker.

3.2. Cross Chain dApps

Multi-chain decentralized applications (dApps) are applications that deploy the same contracts on different blockchains, which have no chain interoperability. Specifically, even for the same multi-chain dApp, the internal states for managing data in different blockchain networks are isolated. The isolated deployment among different blockchains can cause negative impacts on user experience, as user experience in different blockchain networks may not be the same. As Web 3.0 will eventually become a multi-chain ecosystem, there is a growing need for communication among different blockchains.

Cross-chain dApps allow communication and coordination of smart contracts in different blockchain networks. Ideally, in cross-chain dApps, applications are splitted and then synchronized together to work. Specifically, different parts have different functionalities according to the capabilities of different blockchain networks, and they are managed to synchronously collaborate as a whole. To achieve data exchange between different networks, cross-chain bridges mentioned in section 3.1 can be used. In the following, two kinds of new emerging dApps will be introduced.

GameFi is a fusion of "game" and "finance", which runs on blockchain and offers in-game financialization to incentive players. It runs on distributed ledgers to provide verifiable asset ownership and reliable in-game trading markets for players. In GameFi, players can earn and spend cryptocurrencies and non-fungible tokens (NFTs), and transfer their assets outside of the game to trade on NFT marketplaces and cryptocurrency exchanges. This revolutionary gaming mode is called play-to-earn (P2E), which attracts players by offering them opportunities to make money while playing games. Figure 4 below shows different services adding value to the GameFi ecosystem, ordered from gamers to developers [54].

SocialFi is a fusion of "social" and "DeFi (decentralized finance)", which runs decentralized social media platforms on blockchains and offers rewards to incentive users. On web 2.0 social media platforms, content and engagement generated by users are monetized by platforms, and platforms also have the power to ban users, delete or edit content posted by users. Such centralized authority of social media

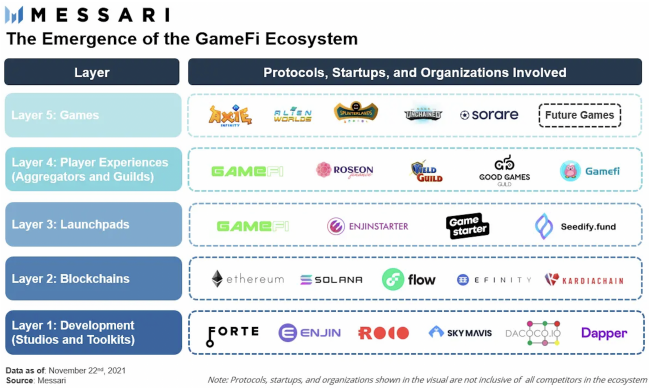

Figure 4. The emergence of the GameFi ecosystem [54]

leave a lot of room to manipulate freedom of speech. In addition, it is very difficult to accurately define or track digital ownership of the contents created by users, opening up opportunities for the digital piracy of content. In SocialFi, users can have complete control over their own data, and have the power to monetize their content and engagement with cryptos. There are some popular SocialFi platforms nowadays, such as PeakD, ApolloFi, Chingari, and Ecency.

3.3. Blockchain scaling

Blockchain scaling solutions (both L1 and L2) often require communication with outside parties and/or blockchains. Thus, many scaling solutions can be viewed as application of cross-chain communication. In the following sections, we will give an overview of Plasma, Rollups, and Channels, with an emphasis on how they utilize cross-chain communication.

3.3.1. Plasma.

Plasma is a Layer 2 scalability solution for Ethereum proposed by Joseph Poon and Vitalik Buterin in 2017 [55] [1]. In order to gain greater capacity, a Plasma sidechain is run by centralized operators with faster consensus mechanisms like proof-of-authority. To curb the drawbacks of a centralized operator, Plasma proposes to publish each sidechain's block header to the parent chain. This anchors the security and availability of Plasma blocks to the parent chain, greatly reducing the trust needed on Plasma operators.

In depositing funds, a Plasma sidechain is quite similar to a cross-chain bridge in that users deposit to a smart contract belonging to the sidechain on the parent chain. Upon withdrawal, however, users cannot prove that their UTXO (Unspent Transaction Output) has not already been spent within the Plasma chain. Therefore, a challenge period is opened where users of the Plasma chain can submit fraud proofs to the Plasma smart contract on the parent chain. In a fraud proof, a user presents a block where the UTXO is spent and the block is matched to a block header recorded on the parent chain, then the withdrawal is rejected and the user is also punished.

3.3.2. Rollups.

Rollups are another type of scaling solutions that aim to reduce the amount of data needed to be stored on the parent chain per transaction [1]. However, rollups also dictate that some information, albeit compressed, be stored on the parent chain. This design can address the data availability issue which is a concern in Plasma, as all the transaction information is recoverable from the parent chain.

3.3.3. Payment Channels.

A payment channel is a private peer-to-peer ledger that is maintained off-chain [**?**] [1]. This type of solution can manage transactions between directly or indirectly connected nodes, and the main chain only has to track channel openings and closings (creating/destroying direct connections), therefore reducing the main chain's workload.

Payment routing is an important part of many payment channel solutions. Routing finds intermediaries who can facilitate transactions between parties that have no on-chain connection. This is usually accomplished with the help of hash-locking3. Hash verification guarantees the safety of funds (only authorized users can get funds), and time verification guarantees liveness of a channel in case a user becomes unresponsive. Additional security can be achieved via a monitoring service that which watches for cheating behavior and prevents honest users from being exploited while unresponsive/away [56].

3.4. Discussions

The design of cross-chain protocols forms a solid foundation for various which provides low transaction fees and high inter-chain exchange throughput. However, at the time of paper writing, the security guarantee of cross-chain transactions are not enough for user trusts. Relying on cross-chain applications (e.g., bridges) for exchange requires placing additional trust for bridge providers, whose security prevention are not transparent to users. Therefore, cross-chain bridges become the perfect target for attackers for high reward. For instance, we have seen the Ronin Sidechain attack from which attackers exploited some vulnerability in the bridge that connects Ronin to Ethereum and extracted over $622M from it [57]. In the later section, we also include a series of bridge attacks and discuss the security aspects of cross-chain bridges.

Even though more and more cross chain dApps have emerged in recent years, the ideal multi-chain ecosystem allowing free exchange among most blockchain networks still has not been implemented yet. For instance, Cosmos IBC and Polkadot are projects allowing information exchange inside their ecosystem, while they are currently unable to support a more interconnected ecosystem. It is conceivable to see a cross chain ecosystem which will not be limited by walls between any set of networks in the future.

4. Security

4.1. Bridge Attacks

At a high level, bridges have a few key components to enable asset transfers between two blockchains [58]. First is the *custodian* on the source blockchain, implemented as a smart contract that locks up assets deposited into it until release conditions are met. Second is the *debt issuer* on the destination blockchain, which can *mint* (issue) digital representations of tokens held by the custodian. Such a representational token on the destination chain is called a *debt token*. To sync the chains, a *communicator* reads deposit events on the source chain and sends signals for debt issuance on the destination chain. Finally, a user can also send tokens back by *burning* their debt tokens; then depending on the bridge implementation, either the communicator or anyone with a burn proof can signal to the custodian to release tokens on the source chain.

When implemented properly, this custodian-debt issuer architecture should ensure that a token is available for use on exactly one chain at a time, accomplishing asset transfer and preventing double-spending. Unfortunately, all key components have had implementational vulnerabilities in practice, allowing cross-chain bridges to become frequent victims of attacks. In 2022 alone, tokens worth more than $1.7 billion have been lost from bridges to attacks [59], involving even some of the largest players in the space including Binance.

4.1.1. Custodian attacks.

Custodian attacks involve tricking the custodian smart contract into performing unauthorized/unintended transactions.

(a) Changing privileged address: An attack that PolyNetwork bridge experienced changed the custodian's keeper address to the attacker's [60] [61]. The attacker mainly exploited two facts: the Ethereum function selector only checks 4 bytes of function hash; the custodian contract on the Ethereum chain has a function to execute data stored on the Poly Chain, without performing proper checks. First, the attacker used brute force to find a function name that has a partial hash collision with the function to change the keeper address. Then the attacker invokes the Ontology relayer to send a transaction with a malicious payload containing this function name to the Poly chain. This transaction is then included on the Poly chain, even though it doesn't execute properly. However, the attacker can now use this signed Poly chain transaction to pass the custodian's verification and invoke the function to change the keeper address. Note that even though the Ethereum relayer would not allow invoking the custodian contract from the Poly chain, the attacker invoked it manually, bypassing the communicator completely. Thus, the attacker was able to become the custodian keeper and steal the funds.

(b) Forging proof-of-burn: In cases where the custodian release tokens based on proof-of-burn submitted by users instead of relying on a centralized communicator, an attacker can try to forge proof-of-burn to fool the custodian into releasing tokens. Such an exploit was found and patched on the Polygon/Matic EthereumPlasma bridge [62]. Due to faulty logic, the custodian may discard the first byte of a proof-of-burn mid-verification. The attacker can find a valid proof-of-burn that triggers this behavior, and then manipulate the value of its first byte to produce more seemingly valid proof-of-burn. As such, the attacker can withdraw much more tokens from the custodian than intended.

(c) False deposits: The custodian is supposed to emit a deposit event if and only if a user deposits tokens correctly and successfully. However, in the case of the Meter bridge hack [63], the attacker used incorrect deposits and still got the custodian to emit deposit events. As a fork of ChainBridge, the bridge's custodian is supposed to only accept specific wrapped tokens, but they also added logic to support the automatic wrap and unwrap of native tokens. However, in doing so, the original deposit function that expects and assumes wrapped tokens are exposed and the attacker called it with native tokens. Although the deposits failed in that the custodian received nothing, they tricked the custodian into emitting deposit events and allowed the attacker to obtain debt tokens on the destination chain.

Defenses: These attacks show that proper verification before performing any sensitive action is critical. Privileged components should be thoroughly analyzed, and their access should be carefully guarded, especially when modifying/porting existing architectures.

4.1.2. Debt Issuer Attacks.

The goal of such attacks is to make the debt issuer execute unauthorized transactions, resulting in wrongly-minted tokens.

(a) Bypassing signature verification: Normally, the debt issuer mints tokens on the destination chain only after verifying a signed message from a communicator. One would expect the verifier to be a highly analyzed and protected piece of architecture. Yet, in Wormhole's case [64], in order to have flexible and modular support for different communicators and their verification, communicators can supply their own signature verifier. The attacker created a signature verifier that simply "verifies" any transaction, and they sent a debt issuance signal to the debt issuer with this verifier. The debt issuer thus minted a large sum of tokens to the attacker on the Solana chain, which they then "sent back" to the Ethereum chain and obtained 93750 ETH from the bridge's custodian.

Defenses: Again, we see that a highly privileged component, the signature verifier, was misused. Such components need to be carefully examined and trusted, so they cannot be supplied arbitrarily by users. New communicator-verifier pairs need to be analyzed and approved before being accepted by the debt issuer.

4.1.3. Communicator Attacks.

A communicator watches certain events of a blockchain and forwards messages to corresponding parties/components when appropriate. Communicator attacks aim to cause

communicators to forward messages inconsistent with the blockchain state.

(a) Spoofing deposit events: In the network hack [65], the communicator watches all events involving the custodian contract, not just those emitted by it. This is partially due to the technical fact that the Ethereum protocol does not provide cryptographic authentication for event logs, making it possible for attackers to spoof events unless extra authentication is implemented. The attacker set up a smart contract on the source chain with a method to 1) interact with the custodian normally and 2) emit a fake deposit event that looks legit to the communicator. This made the communicator think that a real deposit event took place and signaled debt tokens to be minted to the attacker.

Defenses: Communicators need to carefully authenticate events. As some chains (e.g., Ethereum) don't provide secure built-in event authentication, the custodian or another bridge component needs to emit events that can be securely authenticated by communicators.

4.1.4. Discussion.

We notice a common theme across the aforementioned bridge vulnerabilities: bridge developers often exhibit a poor understanding of components that need protection and of ways to protect them. For example, Wormhole allowed users to supply blatantly wrong signature verification logic [64], which should be a tightly guarded component of the bridge's stack. This is somewhat expected as blockchain developers are often rushing to implement new concepts or features, without established standards to use as guides. Therefore, we think more effort should be put into developing safety standards for components (e.g., custodian) and domain-specific software engineering practices/frameworks [66], especially in such high-stake settings as cross-chain bridges.

4.2. Replay Attacks

Replay attacks are a family of attacks that are traditionally seen in network communication. Replay attacks can happen when adversaries are able to observe the network communication among different parties and intercept or delay their transmitted messages. The thread model is similar to the Man-in-the-Middle attack except for no decryption on the intercepted messages conducted by adversaries. Instead, they can resend the intercepted messages to the expected destination party or any other parties within the network to obtain beneficial information, which are important hints for exploitation. Normally, modern blockchains all implement replay protection features that are effective to prevent such risks.

However, when forking happens within the chain, it is possible for adversaries to exploit replay attacks before the newly-forked chain enables replay protection, as reported after the recent Ethereum Merge [67]. Also, in a more strict cross-chain context, researchers find replay attacks are critical to sharded distributed ledgers protocols (e.g., Omniledger [68], Chainspace [69]), where the shard-to-shard communication can be viewed as a weakened version of cross-chain communication. By the time of report writing, replay attacks in the cross-chain context have not been extensively studied. To the best of our knowledge, we argue that they are expected to occur with fewer difficulties in the cross-chain context as the heterogeneous security levels among different chains leave more room for attackers to exploit vulnerabilities and the cross-chain nature enables them to jump between different chains easily [70], [71].

Defenses: [72] proposes a novel cross-shard atomic commit protocol to provide defenses to replay attacks for cross-shard consensus in sharded distributed ledgers, which can also be extended to protect replay attacks on different chains. The two main goals of the protocol is to address replay attacks are 1) mapping a specific session identifier to each transaction so that receivers can verify the sender and 2) instantiating dummy objects for both ends so that dummy objects can act as internal validators for their corresponding transactions. [73] proposes an atomic trade protocol for Bitcoin and Ethereum for swapping coins during merges from hardforks. They also conduct case studies on how cross-chain trade is handled on Bitcoin and Ethereum respectively when the hardfork happens.

4.3. Data Availability Attacks

Numerous blockchain scaling strategies encounter the Data Availability problem. The gist of the Data Availability problem is that how can active nodes guarantee the newly-received blocks contains all the data in them. In other words, how active nodes ensure the data coherence is maintained among all the new blocks. The exploit in this scenario is that, if the coherence is not guaranteed, malicious nodes can insert invalid transactions into some blocks and lead to disagreement among honest nodes (e.g., double spending and counterfeiting). In the blockchain networks, there are generally two types of nodes: 1) Full nodes who store all the block data and actively act like validators for confirming new blocks and 2) light clients who have limited hardware resources therefore relying on the validation results (i.e. Fraud Proof [74]) of full nodes for accepting new blocks. If only partial data are available for full nodes' validation, light clients are obligated to track the block header (i.e., tracking the root of the Merkel tree) committed on the chain for validation, which defeats the point of having light clients.

The problem is critical in scaling technologies as optimization methods for scaling can potentially compromise protection mechanisms for data availability attacks. For instance, if the block size increases, more nodes will prefer to host light clients rather than full nodes due to the increase in hardware burdens, which leaves fewer validation nodes and weakened protections. Also, when blockchains are split into different shards to improve throughput, a full node will be responsible for multiple shards while it also runs light clients for every other shard. Similarly, the number of full nodes will be significantly less than the number of light clients, which leads to the same results as the previous example.

Defenses: To address the Data Availability problem, a line of works was proposed by leveraging the erasure

code feature for validation. [74] first proposed to encode a block with erasure code, which allows users to recover full data using only a portion of the available data. The validation process in a sampling manner and the recovered results for a sampled portion of data are compared. The light clients will accept the new block based on the proof results. [75] extends this idea by implementing erasure code in a novel data structure, Coded Merkle Tree (CMT), which can effectively reduce the proof size and speed up the decoding process due to the Low-Density Parity-Check (LDPC) code. [76] further improves this line of work by co-designing the LDPC codes with the sampling strategies (e.g., entropy-constrained, greedy sampling) to meet the requirements for adversaries with different threat levels while reducing the computation overhead of the previous methods. [77] proposes to decouple the consensus layer in blockchain systems with the execution systems so that proof to prevent data availability attacks can proceed parallelly with transaction processing while the ordering of the transactions is protected.

4.4. Stake Bleeding Attacks

One particularly serious threat in Proof of Stake (PoS) is the "long-range attacks", which was first proposed by Buterin [78]. This is a theoretical threat to undermine the PoS blockchain protocol starting from the genesis block or any old state, and take over the actual blockchain with a false chain. Once such an attack occurs, newly joining nodes will not be able to reliably distinguish the false chain from the actual chain.

Stake-bleeding is an effective form of long-range attacks against the PoS environment. The only requirement for the attack is that transaction fees are allowed to incentive validators to run the protocol [79].

4.4.1. Stake-bleeding Attack Working Principle.

The main idea of the attack is as follows:

1. On the one hand, the malicious coalition plays the role of validator on the public chain; and on the other hand, the malicious coalition conspires to build a private chain starting from the genesis block. Since the initial information of all participants has already been recorded in the genesis block, no matter on the public chain or the private chain, the probability of being selected as a validator remains the same. The malicious coalition creates a false chain secretly, and will not broadcast this private chain until its length exceeds the length of the public chain.

2. In order to increase the success rate of the attack, the malicious coalition tries to delay the growth of the public chain. Whenever the malicious coalition is elected as a block validator on the public chain, it will always relinquish this verification opportunity. This does not mean that another validator will replace its role and verify this block -actually, no other participants will be chosen to do verification for this block. In this way, the malicious coalition can successfully delay the growth speed of the public chain, so as to make its private chain grow faster and become more competitive.

3. A basic property of blockchain is that transaction fees are used to reward participants to produce new blocks. Since the malicious coalition does not produce blocks when being selected as the validator on the public chain, it will not be able to collect transaction fees from the public chain. In order to get some rewards, every time the malicious coalition has a chance to be selected as a block validator, it will generate a block. In fact, the malicious coalition is the only validator on its private chain. In this way, a lot of transaction fees in the false blockchain will be collected by the malicious coalition. In addition, the malicious coalition copies transactions from the public chain and then broadcasts them on its private chain, thereby increasing the transaction fees it earns.

4. By adopting such a strategy, after a substantial period of time, it is conceivable that the malicious coalition can successfully delay the growth of the public chain and can finally generate blocks on its private chain at a speed faster than the honestly maintained public chain. Once the length of its private chain exceeds the public chain, it will broadcast its private chain, so the coalition can rewrite the transaction history.

4.4.2. Defenses of Stake-bleeding attacks.

There are some strategies to mitigate stake-bleeding attacks:

1. Generally, stake-bleeding attacks have the following characteristics: the block density of the private blockchain is sparse initially, and the block density gradually increases as time goes by. This can be used as a rule to select suspicious chains.

2. Another defense strategy is to introduce context into each transaction, which includes the hash of the blockchain at some time point. In this way, such transactions with hash context cannot be transferred to an attacker private chain maintained by the malicious coalition [80].

4.5. Untraceability across Cryptocurrencies

One of the most common misconceptions about cryptocurrencies is that holding and trading them is completely anonymous and untraceable. A series of recent studies have shown that even in cryptocurrencies specifically designed for better anonymity (e.g. Dash, Monero, and Zcash), it is possible to track the flow of funds and diminish the level of anonymity. This is mainly because the cryptocurrency ledger contains all transactions that have ever occurred and are globally visible with complete transparency. In traditional cryptocurrency crime cases, criminals often use cryptocurrency exchanges to cash out illicit funds. However, in recent years, cryptocurrency exchanges have begun to implement more strict policies to identify addresses associated with these illegal transactions. This means that criminals have to run higher risks of exposing their real identities when engaging in illegal criminal finances [81]. However, given the proliferation of cross-chain transactions, it is important to trace transactions within the ledger of a single cryptocurrency, but also relevant to trace transactions across ledgers

of different cryptocurrencies as well. This is especially important as there have been a few crimes of using cross-chain transactions to obscure illicit financial flows. For example, in January 2018, an investment firm called Starscape Capital sent ETH worth $517,000 from their wallet to ShapeShift for Monero [82]. Meanwhile, all of their social media accounts disappeared. As another example, a ransomware operator called WannaCry used ShapeShift to convert ransomed Bitcoin into Monero [83].

Starting in 2014, frictionless trading platforms such as ShapeShift and Changelly have emerged, which enable cross-cryptocurrency transactions without storing their coins with the platform provider [84]. Next, I will take ShapeShift as an example to introduce how those frictionless platforms trace transactions across cryptocurrency ledgers.

4.5.1. Digital Asset Trading Platforms Working Principle.

The following notations are defined by ShapeShift.

curIn: the currency that the user wants to send.

curOut: the currency that the user wants to obtain.

addrs: the address in the **curIn** blockchain that the currency is sent from.

addru: the destination address in the **curOut** blockchain that the output currency will be sent to.

rate: the exchange rate between **curIn** and **curOut**.

amt: the amount of money in **curIn** that the user is going to convert.

fee: the miner fee for the transaction.

The user sends to **addrs** the amt in **curIn**, and after some delay, an amount of **curOut** will be sent to **addru**, which roughly equals to (**rate** · **amt** - **fee**).

4.5.2. Blockchain Transactions Identification.

ShapeShift offers an API that enables users to execute their transactions. The API mainly provides three kinds of information:

1. The current **rate** between **curIn** and **curOut**.
2. Up to 50 recent transactions across all users.
3. Full information of a ShapeShift transaction, if the corresponding **addrs** in the **curIn** blockchain is given.

Also, ShapeShift defines the whole process of transaction into the following 2 phases:

Phase 1: **amt** is sent from the user to **addrs** on the **curIn** blockchain.

Phase 2: (**rate** · **amt** - **fee**) will be withdrawn on the **curOut** blockchain.

To start with Phase 1, ShapeShift seeks to identify the correct deposit transaction on the **curIn** blockchain, where two main requirements must be guaranteed:

1. The timing of it is reasonably close to the time when it was advertised via the API.
2. The amount being sent was identical to the advertised amount.

Then, ShapeShift seeks to identify the correct Phase 2 transaction on the **curOut** blockchain. Only if valid results in both Phase 1 and Phase 2 are gained, the API returns a valid result. Of course there are some other alternative ways to identify Phase 2, but we will not get into details in this paper.

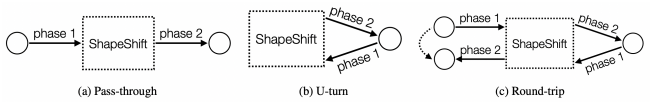

Figure 5. Different transactional patterns in ShapeShift

4.5.3. Cross-currency Activity Tracing.

In ShapeShift, both on-chain transactions and cross-chain transactions are possible to be identified. Figure 5 shows three main transactional patterns in ShapeShift. These patterns are categorized primarily based on the role they play in tracking across-chain transactions of different cryptocurrencies.

Pass- through transaction: It is the full flow of the deposit sent from the input address **addrs** on the curIn blockchain, and converted to withdrawal to the output address **addru** on the curOut blockchain. ShapeShift identifies the flow of transactions and is able to create a link between a pair of **addrs** and **addru**.

U-turn: We consider it as a pattern in which a user has shifted into one currency from **curIn** to **curOut**, only to turn around and immediately shift back to **curIn**. It is likely that shifting back and forth between two different cryptocurrencies is motivated by benefits such as fluctuations in their price or money laundering.

Round-trip transaction: It is the combination of the results of pass-through and U-turns transactions, which performs two ShapeShift transactions: (1) coins move from one currency to another in a pass-through transaction; (2) then the output of the pass-through transaction get involved in a U-turn transaction to convert the coins back to the original currency. Such behavior can be motivated by money laundering as well. Similar to pass-through transactions, identifying round-trip transactions allows ShapeShift to create a link between a pair of **addrs** and **addru**. If it is detected, ShapeShift will sever the link between their two addresses.

4.6. Discussion

The Replay attacks and the Data Availability attacks exploit the vulnerabilities of heterogeneous security guarantees for different chains as well as the imbalance between decentralization and security in some blockchain systems. To tackle such a challenge, future designers should incorporate heterogeneity into cross-chain communication protocols. For example, the lower and upper bounds of resources (e.g., computation powers, stakes) of in-network nodes should be set with a certain threshold which can prevent attacks from sneaking into the network easily. Also, the coordination and management of heterogeneous cryptographic signatures for different chains are also essential for a secure multi-chain environment as different chains' different implementations of cryptography verification (e.g., different elliptic curves for

verification) can expose vulnerabilities to attackers. Another potential solution to better enhance the security level of cross-chain communication protocols is to leverage formal verification of transactions in the multi-chain ecosystem which can provide a mathematically-guaranteed behavior for each node and their submitted transactions. A line of works in this direction [85], [86] has already pioneered this.

5. Conclusion

In this SoK, we present a systematic literature review of the cross-chain communication protocols, which are designed to cater to the growing diversified usages on different blockchains. Specifically, we summarize mainstream cross-chain communication protocol schemes, report applications of modern cross-chain communication protocols, and present common attacks and defenses in the cross-chain context.

References

[1] C. Sguanci, R. Spatafora, and A. M. Vergani, "Layer 2 blockchain scaling: a survey," 2021. [Online]. Available: https://arxiv.org/abs/2107.10881

[2] R. Belchior, A. Vasconcelos, S. Guerreiro, and M. Correia, "A survey on blockchain interoperability: Past, present, and future trends," 2020. [Online]. Available: https://arxiv.org/abs/2005.14282

[3] P. Robinson, "Survey of crosschain communications protocols," 2020. [Online]. Available: https://arxiv.org/abs/2004.09494

[4] T. Nolan, "Alt chains and atomic transfers," May 2013. [Online]. Available: https://bitcointalk.org/index.php?+topic=193281.0

[5] E. Team, "Btc relay documentation release 1.0," 9 2016. [Online]. Available: https://buildmedia.readthedocs.org/media/pdf/btc-relay/latest/btc-relay.pdf

[6] Chainlink, "Introducing the cross-chain interoperability protocol (CCIP)," Aug 2021. [Online]. Available: https://blog.chain.link/introducing-the-cross-chain-interoperability-protocol-ccip/

[7] R. Dasari, E. Surmeli, K. Yeung, and K. Alfaro, "Cross-Consensus Message Format (XCM)," Sep 2022. [Online]. Available: https://wiki.polkadot.network/docs/learn-xcm#resources

[8] H. Sun, H. Mao, X. Bai, Z. Chen, K. Hu, and W. Yu, "Multi-blockchain model for central bank digital currency," in *2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, 2017, pp. 360–367.

[9] A. Sardon, T. Hardjono, and M. McBride, "Blockchain Gateways: Use-Cases," Internet Engineering Task Force, Internet-Draft draft-sardon-blockchain-gateways-usecases-03, Apr. 2022, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-sardon-blockchain-gateways-usecases/03/

[10] R. Belchior, B. Putz, G. Pernul, M. Correia, A. Vasconcelos, and S. Guerreiro, "Ssibac: Self-sovereign identity based access control," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020, pp. 1935–1943.

[11] F. Tramèr, D. Boneh, and K. Paterson, "Remote Side-Channel attacks on anonymous transactions," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020, pp. 2739–2756. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/tramer

[12] Z. Nehaï, F. Bobot, S. Tucci-Piergiovanni, C. Delporte-Gallet, and H. Fauconnier, "A tla+ formal proof of a cross-chain swap," in *23rd International Conference on Distributed Computing and Networking*, ser. ICDCN 2022. New York, NY, USA: Association for Computing Machinery, 2022, p. 148–159. [Online]. Available: https://doi.org/10.1145/3491003.3491006

[13] Y. Manevich and A. Akavia, "Cross chain atomic swaps in the absence of time via attribute verifiable timed commitments," in *2022 IEEE 7th European Symposium on Security and Privacy (EuroSP)*, 2022, pp. 606–625.

[14] Y. Li, J. Weng, M. Li, W. Wu, J. Weng, J.-N. Liu, and S. Hu, "Zero-cross: A sidechain-based privacy-preserving cross-chain solution for monero," *Journal of Parallel and Distributed Computing*, vol. 169, pp. 301–316, 2022.

[15] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Cryptography Mailing list at https://metzdowd.com*, 03 2009.

[16] V. Buterin *et al.*, "Ethereum white paper."

[17] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Advances in Cryptology – CRYPTO 2017*, J. Katz and H. Shacham, Eds. Cham: Springer International Publishing, 2017, pp. 357–388.

[18] T. Rocket, M. Yin, K. Sekniqi, R. van Renesse, and E. G. Sirer, "Scalable and probabilistic leaderless bft consensus through metastability," 2019. [Online]. Available: https://arxiv.org/abs/1906.08936

[19] A. Yakovenko, "Solana: A new architecture for a high performance blockchain."

[20] J. Kwon and E. Buchman, "A network of distributed ledgers."

[21] G. Wood, "Pokadot: Vision for a heterogeneous multi-chain framework."

[22] N. Asokan, "Fairness in electronic commerce," 07 1998.

[23] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures," *Selected Areas in Communications, IEEE Journal on*, vol. 18, pp. 593 – 610, 05 2000.

[24] ——, "Asynchronous protocols for optimistic fair exchange," in *Proceedings. 1998 IEEE Symposium on Security and Privacy (Cat. No.98CB36186)*, 1998, pp. 86–99.

[25] R. Community, "Router whitepaper," Accessed: 2022-11-5.

[26] B. Bünz, S. Goldfeder, and J. Bonneau, "Proofs-of-delay and randomness beacons in ethereum," 2017.

[27] Consensys, "Nomad: A generalized cross-chain communication protocol."

[28] V. Buterin, "Chain interoperability," Sep 2016. [Online]. Available: https://theblockchaintest.com/uploads/resources/R3%20-%20Chain%20Interoperability%20-%202016%20-%20Sep.pdf

[29] G. Wang, "Sok: Exploring blockchains interoperability," Cryptology ePrint Archive, Paper 2021/537, 2021, https://eprint.iacr.org/2021/537. [Online]. Available: https://eprint.iacr.org/2021/537

[30] Y. Sun, L. Yi, L. Duan, and W. Wang, "A decentralized cross-chain service protocol based on notary schemes and hash-locking," in *2022 IEEE International Conference on Services Computing (SCC)*, 2022, pp. 152–157.

[31] 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

[32] "0x: An open protocol for decentralized exchange on the ethereum blockchain," 2017. [Online]. Available: https://www.0x.org/pdfs/0x_white_paper.pdf

[33] "The interledger protocol white paper," 2016. [Online]. Available: https://github.com/interledger/paper

[34] T. Stefan and S. Evan, "A protocol for interledger payments," 2016. [Online]. Available: https://interledger.org/interledger.pdf

[35] E. J. Scheid, T. Hegnauer, B. Rodrigues, and B. Stiller, "Bifröst: a modular blockchain interoperability api," in *2019 IEEE 44th Conference on Local Computer Networks (LCN)*, 2019, pp. 332–339.

[36] "Btc-relay: Ethereum contract for bitcoin spv," 2015. [Online]. Available: https://github.com/ethereum/btcrelay

[37] A. Garoffolo, D. Kaidalov, and R. Oliynykov, "Zendoo: A zk-snark verifiable cross-chain transfer protocol enabling decoupled and decentralized sidechains," *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, 2020. [Online]. Available: https://eprint.iacr.org/2020/123.pdf

[38] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct Non-Interactive zero knowledge for a von neumann architecture," in *23rd USENIX Security Symposium (USENIX Security 14)*. San Diego, CA: USENIX Association, Aug. 2014, pp. 781–796. [Online]. Available: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/ben-sasson

[39] L. Lys, A. Micoulet, and M. Potop-Butucaru, "R-swap: Relay based atomic cross-chain swap protocol," in *Algorithmic Aspects of Cloud Computing*, G. D'Angelo and O. Michail, Eds. Cham: Springer International Publishing, 2021, pp. 18–37.

[40] maplabs.io, "Omnichain layer of web3 with provably secure cross-chain communication built upon light-client and zero-knowledge technology litebook v4," 2021. [Online]. Available: https://files.maplabs.io/pdf/mapprotocol_Litebook_en.pdf

[41] R. Team, "Rangers protocol white paper v.2.0," 7 2022. [Online]. Available: https://www.rangersprotocol.com/pdf/RangersProtocolWhitepaper.pdf

[42] P. Zappalà, M. Belotti, M. Potop-Butucaru, and S. Secci, "Game theoretical framework for analyzing blockchains robustness," Cryptology ePrint Archive, Paper 2020/626, 2020, https://eprint.iacr.org/2020/626. [Online]. Available: https://eprint.iacr.org/2020/626

[43] B. Adam, C. Matt, D. Luke, F. Mark, M. Gregory, M. Andrew, P. Andrew, T. Jorge, and W. Pieter, "Enabling blockchain innovations with pegged sidechains," Oct 2014. [Online]. Available: https://blockstream.com/sidechains.pdf

[44] P. Joseph and D. Thaddeus, "The bitcoin lightning network: Scalable off-chain instant payments," Jan 2016. [Online]. Available: https://lightning.network/lightning-network-paper.pdf

[45] A. Zamyatin, D. Harz, J. Lind, P. Panayiotou, A. Gervais, and W. Knottenbelt, "Xclaim: Trustless, interoperable, cryptocurrency-backed assets," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 193–210.

[46] K. Narayanam, V. Ramakrishna, D. Vinayagamurthy, and S. Nishad, "Atomic cross-chain exchanges of shared assets," *arXiv e-prints*, p. arXiv:2202.12855, Feb. 2022.

[47] M. Westerkamp and M. Diez, "Verilay: A verifiable proof of stake chain relay," in *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2022, pp. 1–9.

[48] "Celer network," 2022. [Online]. Available: https://www.celer.network/

[49] "Umbria network," 2020. [Online]. Available: https://umbria.network/

[50] "Multichain," 2020. [Online]. Available: https://multichain.org/

[51] "Stableswap - efficient mechanism for stablecoin liqudity," 2019. [Online]. Available: https://curve.fi/files/stableswap-paper.pdf

[52] "Wormhole," 2021. [Online]. Available: https://docs.wormhole.com/wormhole/

[53] R. Zarick, B. Pellegrino, and C. Banister, "Layerzero: Trustless omnichain interoperability protocol," 2021. [Online]. Available: https://arxiv.org/abs/2110.13871

[54] J. Sun, "Explain it like i'm 5: Gamefi," Nov 2021. [Online]. Available: https://messari.io/report/explain-it-like-i-m-5-gamefi

[55] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart contracts," 2017. [Online]. Available: https://www.plasma.io/plasma.pdf

[56] [Online]. Available: https://raiden.network/101.html

[57] "Hacker drains $622m from axie infinity's ronin ethereum sidechain." [Online]. Available: https://decrypt.co/96322/hacker-622-million-axie-infinity-ronin-ethereum

[58] S.-S. Lee, A. Murashkin, M. Derka, and J. Gorzny, "Sok: Not quite water under the bridge: Review of cross-chain bridge hacks," 2022. [Online]. Available: https://arxiv.org/abs/2210.16209

[59] C. Moran, "Blockchain bridges keep getting attacked. here's how to prevent it," *CoinDesk*, Oct 2022. [Online]. Available: https://www.coindesk.com/layer2/2022/10/14/blockchain-bridges-keep-getting-attacked-heres-how-to-prevent-it/

[60] Y. Hu, S. Wu, L. Wu, and Y. Zhou, "The initial analysis of the polynetwork hack - blocksec," *Medium*, Aug 2021. [Online]. Available: https://blocksecteam.medium.com/the-initial-analysis-of-the-polynetwork-hack-270ac6072e2a

[61] ——, "The further analysis of the poly network attack - blocksec," *Medium*, Aug 2021. [Online]. Available: https://blocksecteam.medium.com/the-further-analysis-of-the-poly-network-attack-6c459199c057

[62] Immunefi and G. Wagner, "Polygon double-spend bugfix review — $2m bounty - immunefi - medium," *Immunefi*, Jun 2022. [Online]. Available: https://medium.com/immunefi/polygon-double-spend-bug-fix-postmortem-2m-bounty-5a1db09db7f1

[63] C. Adams, "Breaking down the meter hack," *ChainSafe*, Sep 2022. [Online]. Available: https://blog.chainsafe.io/breaking-down-the-meter-io-hack-a46a389e7ae4

[64] D. Goodin, "How $323m in crypto was stolen from a blockchain bridge called wormhole," Feb 2022. [Online]. Available: https://arstechnica.com/information-technology/2022/02/how-323-million-in-crypto-was-stolen-from-a-blockchain-bridge-called-wormhole/

[65] p. Team, "pnetwork post mortem: pbtc-on-bsc exploit - pnetwork - medium," *pNetwork*, Sep 2021. [Online]. Available: https://medium.com/pnetwork/pnetwork-post-mortem-pbtc-on-bsc-exploit-170890c58d5f

[66] G. Destefanis, M. Marchesi, M. Ortu, R. Tonelli, A. Bracciali, and R. Hierons, "Smart contracts vulnerabilities: a call for blockchain software engineering?" in *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, 2018, pp. 19–25.

[67] "Ethw confirms contract vulnerability exploit, dismisses replay attack claims," 2020. [Online]. Available: https://cointelegraph.com/news/ethw-confirms-contract-vulnerability-exploit-dismisses-replay-attack-claims

[68] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "Omniledger: A secure, scale-out, decentralized ledger via sharding," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 583–598.

[69] M. Al-Bassam, A. Sonnino, S. Bano, D. Hrycyszyn, and G. Danezis, "Chainspace: A sharded smart contracts platform," in *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018. [Online]. Available: http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_09-2_Al-Bassam_paper.pdf

[70] D. Meshkov, A. Chepurnoy, and M. Jansen, "Short paper: Revisiting difficulty control for blockchain systems," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology - ESORICS 2017 International Workshops, DPM 2017 and CBT 2017, Oslo, Norway, September 14-15, 2017, Proceedings*, ser. Lecture Notes in Computer Science, J. García-Alfaro, G. Navarro-Arribas, H. Hartenstein, and J. Herrera-Joancomartí, Eds., vol. 10436. Springer, 2017, pp. 429–436. [Online]. Available: https://doi.org/10.1007/978-3-319-67816-0_25

[71] Y. Kwon, H. Kim, J. Shin, and Y. Kim, "Bitcoin vs. bitcoin cash: Coexistence or downfall of bitcoin cash?" in *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 2019, pp. 935–951. [Online]. Available: https://doi.org/10.1109/SP.2019.00075

[72] A. Sonnino, S. Bano, M. Al-Bassam, and G. Danezis, "Replay attacks and defenses against cross-shard consensus in sharded distributed ledgers," in *2020 IEEE European Symposium on Security and Privacy (EuroSP)*, 2020, pp. 294–308.

[73] P. McCorry, E. Heilman, and A. Miller, "Atomically trading with roger: Gambling on the success of a hardfork," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology - ESORICS 2017 International Workshops, DPM 2017 and CBT 2017, Oslo, Norway, September 14-15, 2017, Proceedings*, ser. Lecture Notes in Computer Science, J. García-Alfaro, G. Navarro-Arribas, H. Hartenstein, and J. Herrera-Joancomartí, Eds., vol. 10436. Springer, 2017, pp. 334–353. [Online]. Available: https://doi.org/10.1007/978-3-319-67816-0_19

[74] M. Al-Bassam, A. Sonnino, and V. Buterin, "Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities," 2018. [Online]. Available: https://arxiv.org/abs/1809.09044

[75] M. Yu, S. Sahraei, S. Li, S. Avestimehr, S. Kannan, and P. Viswanath, "Coded merkle tree: Solving data availability attacks in blockchains," 2019. [Online]. Available: https://arxiv.org/abs/1910.01247

[76] D. Mitra, L. Tauz, and L. Dolecek, "Overcoming data availability attacks in blockchain systems: Short code-length ldpc code design for coded merkle tree," *IEEE Transactions on Communications*, vol. 70, no. 9, pp. 5742–5759, 2022.

[77] M. Al-Bassam, "Lazyledger: A distributed data availability ledger with client-side smart contracts," 2019. [Online]. Available: https://arxiv.org/abs/1905.09274

[78] V. Buterin, "Long-range attacks: The serious problem with adaptive proof of work," May 2014. [Online]. Available: https://blog.ethereum.org/2014/05/15/long-range-attacks-the-serious-problem-with-adaptive-proof-of-work

[79] P. Gazi, A. Kiayias, and A. Russell, "Stake-bleeding attacks on proof-of-stake blockchains," 06 2018, pp. 85–92.

[80] D. Larimer, "Transactions as proof-of-stake - crypto chain university," Nov 2013. [Online]. Available: https://cryptochainuni.com/wp-content/uploads/Invictus-Innovations-Transactions-As-Proof-Of-Stake.pdf

[81] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. Voelker, and S. Savage, "A fistful of bitcoins: Characterizing payments among men with no names," *Communications of the ACM*, vol. 59, no. 4, pp. 86–93, Apr. 2016.

[82] A. F. News, "How dirty money disappears into the black hole of cryptocurrency," Oct 2018. [Online]. Available: https://www.wsj.com/articles/how-dirty-money-disappears-into-the-black-hole-of-cryptocurrency-1538149743

[83] J. Dunietz, "The imperfect crime: How the wannacry hackers could get nabbed," Aug 2017. [Online]. Available: https://www.scientificamerican.com/article/the-imperfect-crime-how-the-wannacry-hackers-could-get-nabbed/

[84] H. Yousaf, G. Kappos, and S. Meiklejohn, "Tracing transactions across cryptocurrency ledgers," 08 2019.

[85] H. Su, "Cross-chain interaction model in a fully verified way," 2021. [Online]. Available: https://arxiv.org/abs/2106.05463

[86] B. Pillai, Z. Hou, K. Biswas, and V. Muthukkumarasamy, "Formal verification of the burn-to-claim protocol for blockchain interoperability," 06 2022.