2023 IEEE 31st International Conference on Network Protocols (ICNP) | 979-8-3503-0322-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICNP59255.2023.10355607

A Cross-chain Payment Channel Network

Xiaoxue Zhang
University of California Santa Cruz
xzhan330@ucsc.edu

Chen Qian
University of California Santa Cruz
cqian12@ucsc.edu

***Abstract*—Blockchain interoperability and throughput scalability are two crucial problems that limit the wide adoption of blockchain applications. Payment channel networks (PCNs) provide a promising solution to the inherent scalability problem of blockchain technologies, allowing off-chain payments between senders and receivers via multi-hop payment paths. This paper presents a cross-chain PCN, called XHub, that extends PCNs to support multi-hop paths across multiple blockchains and resolves both interoperability and throughput scalability. XHub achieves service availability, transaction atomicity, and auditability. Users who correctly follow the protocols will succeed in making payments or get profits from doing the services. In addition, trustworthy information about hubs will be managed in a decentralized manner and available to all users. We conduct prototype implementation of machines that exchange Internet messages and run with two real blockchains as well as large-scale simulations based on real-world PCN topologies and transactions. The results show that XHubs has small latency for cross-chain payments and can achieve a significantly higher success rate compared to the version without hub management protocols. This work is an important step towards the big picture of a decentralized transaction system that connects a wide scope of users in different blockchains.**

I. Introduction

Blockchain is a promising solution for decentralized digital ledgers. Since Bitcoin was invented in 2008 [1], there have been many other payment systems emerging based on blockchains, such as Ripple [2], Stellar [3], and Ethereum [4]. The total number of cryptocurrencies in the world has soared to more than 20,200 in circulation currently [5]. However, despite the growing ecosystem, cryptocurrencies continue to operate in complete isolation from one another. *Interoperability*, i.e., allowing cryptocurrencies to be transferred across multiple blockchains, is currently one of the bottlenecks preventing the mass adoption of blockchain technology.

One solution of interoperability is to use sidechains [6]. The mainchain maintains a ledger and connects to the sidechain via a communication protocol that facilitates asset transfer between the mainchain and the sidechain [6]. However, it does not allow payments across sidechains. Another solution is to use a blockchain of blockchains where there is another level of blockchain recording and monitoring information and communication between different blockchains [7], [8]. However, introducing another blockchain leads to high difficulty in managing the blockchain and has high latency. Cross-chain bridges have been built in practice [9] to enable users to move funds from one blockchain to another. However, it requires users to have wallets and participate in both blockchains. In practice, users who are in different blockchains and want to make transactions might not want to participate in both blockchains. Currently, there are several commercial centralized exchange systems for executing fund transfers and exchanges across blockchains, such as Coinbase [10]. They work like banks, and users completely rely on them for token exchanges. Thus, these services break the trustless and decentralized property of blockchains.

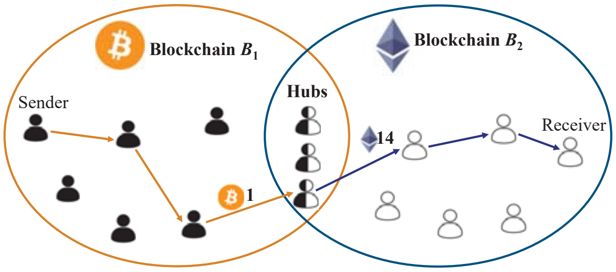


Fig. 1. An example of the cross-chain transaction.

On the other hand, *scalability* on the throughput of blockchains remains another huge problem with growing numbers of users and transactions [11], [12]. For instance, Bitcoin can only support 10 transactions per second at peak in 2022 [13]. Payment channel networks (PCNs), a type of peer-to-peer network, have been proposed to provide a high-throughput solution for blockchain [14], [15]. In a PCN, each user maintains payment channels to a few other users they trust. A transaction between two arbitrary users can be achieved by a multi-hop path of payment channels. Hence only opening and closing a payment channel need to be confirmed by the blockchain while most transactions do not, which significantly reduces the blockchain load.

One intuitive idea to achieve both interoperability and scalability is to build a cross-chain PCN that allows two arbitrary users in different blockchains to make transactions via a multi-hop path. One might immediately think of Internet routing that may cross multiple domains. Similar to the gateway routers on the Internet, there could be some users who act as hubs that have wallets in two blockchains. A user in one blockchain can make a payment to a hub and the hub forwards the payment to another user in a different blockchain, as shown in Fig. 1. In this way, most users only need to hold tokens in one blockchain and still have the freedom to make transactions with any user in other blockchains. This design significantly broadens the user space of blockchain-based applications, compared to existing solutions that rely on one blockchain

979-8-3503-0322-3/23/$31.00 ©2023 IEEE

or require every user to have tokens of multiple blockchains.

However, there are multiple challenges in designing the *decentralized network architecture* of a cross-chain PCN. First, how to manage multi-currency wallets of the hubs such that malicious hubs cannot steal funds from other users. Since the relay hubs are in both blockchains while the clients only have access to one of the blockchains, clients have no idea if the relay hub performs correctly in the other blockchain. The design of Hashed Time-Lock Contracts (HTLCs) [12], [16] and the recently proposed payment channel hubs [17]–[20] enable atomic operations, which means payment will either complete or the sender can get its funds back. However, they cannot solve another challenge: malicious hubs use low token exchange rates to attract users but fail all payment requests. In addition, as a decentralized network, how to make trustworthy hub information available and accessible to other users is another challenge. There is no solution in the literature that can address all the above challenges for a cross-chain PCN.

In this paper, we present the first network architecture and the corresponding protocols of a cross-chain PCN, called XHub. XHub addresses the above challenges by achieving service availability, transaction atomicity, and auditability. In XHub, users who correctly follow the protocols will succeed in making payments or get profits from doing the services. In addition, trustworthy information about hubs will be managed in a decentralized manner and available to all users. **To our knowledge, no prior solution can achieve all these properties.** XHub does not propose new cryptographic methods. Instead, it includes a novel design that combines existing security protocols including multi-signature (multisig) wallets [21], Byzantine agreements with blockchains [14], simplified payment verification protocol [1], and anonymous atomic locks (A^{2}L) [20].

We conduct *prototype implementation* of machines that exchange messages and run on two real blockchains, Bitcoin testnet3 [22] and Ethereum Sepolia testnet [23], as well as *large-scale simulations* based on real-world PCN topologies and transactions, Ripple [2] and Lightning [12]. The results show that the latency of cross-chain transactions is below 1 minute, and even if there are malicious hubs, dispute management takes no more than 3 minutes. The evaluation shows that XHubs could achieve a significantly higher success rate compared to the version without hub management protocols.

In summary, this work makes the following contributions:

- We propose, XHub, the first decentralized network architecture of a cross-chain PCN.
- We design a series of protocols, including the auditor communication protocol, hub registration protocol, transaction protocol, and hub management protocol to achieve the security properties.
- We use both prototype implementation and large-scale simulations to demonstrate the effectiveness of XHub.
- This work is an important step towards the big picture of a decentralized transaction system that connects a wide scope of users in different blockchains.

The rest of this paper is organized as follows. The network and security models are presented in Section II. We describe the detailed design of XHub in Section III. Section V provides the security analysis of XHub. Section V presents the evaluation results. Section VI describes the related work. Section VII concludes this work.

II. Network and Security Models

A. Network Model

The design of XHub considers the case of two blockchains, B_1 and B_2. For a system with more than two blockchains, we assume XHub is built between every pair of blockchains and leave the design of a network of networks to future work. Every user has one or more *wallets*, and each wallet includes tokens (funds) belonging to one blockchain. We use Ψ_1 and Ψ_2 to denote the names of the tokens in B_1 and B_2, respectively. There are three types of users:

Clients. Clients are users who want to make transactions, even if they are in different blockchains. Typically, clients are users with only one wallet and maintain tokens in one blockchain. They cannot directly talk to or transact with users in other blockchains.

Hubs. Users with wallets in both B_1 and B_2 can register as *hubs* that act as relays to forward payments between clients in B_1 and B_2, and get profits by charging transaction fees. For clients in different blockchains to make transactions, they both need to find a bi-directional payment channel or a path to a selected relay hub first. Each hub has an *exchange rate* for tokens. This is public knowledge on blockchains and can be periodically updated by hubs. Each hub is associated with a *reputation*, which is a score to measure its past behaviors. A hub needs to deposit collateral during registration. If the hub fails to provide the correct relay, XHub guarantees that all clients will not lose funds and clients can dispute the failed transactions to lower the reputations of misbehaving hubs.

Auditors. The system also has a committee of *auditors*. The committee provides trustworthy management of hub information, including the exchange rates, reputations, and collateral. Each auditor is a user of both blockchains and can profit by correctly performing the committee services. Any user of both blockchains can register as an auditor as long as they put enough collateral in a multisig wallet maintained by the system. If an auditor performs maliciously or remains unresponsive for some time, they will lose all their collateral and be expelled from the system.

For each blockchain B_i, all users and channels form a *payment channel network* (PCN), modeled as a graph $G = (H, V_i, E_i)$, where H is the set of hubs, V_i is the set of clients in blockchain B_i, and E_i is the set of bi-directional payment channels between users. In a PCN, two users can make transactions if they share a bi-directional channel by committing a certain fund to open the channel, or find a multi-hop path of channels between them. Existing work assumes every client needs to open channels with all hubs the client will use, which leads to significant locked-in funds from hubs [20], [24] and significantly limits scalability. Hence XHub allows a client to connect a hub via a multi-hop path within the PCN and send or receive funds through the path. There are extensive studies on how to route a payment within the same blockchain [25]–[28]. Hence we consider the research of routing within a blockchain to be out of the scope of this paper and use a decentralized solution [28] to route payments between a pair of users in the same blockchain.

B. Security Model and Assumptions

We assume users can exchange messages through a traditional secure communication channel such as TLS. Information leakages among them are beyond the scope of our discussion. We assume the attackers can gain complete control of some but not all relay hubs. For each compromised hub, including controlling the stored funds and network communication, they may drop, modify and replay messages. An attacker may also delay or prevent a hub it controls from accessing the blockchains for an unbounded amount of time. All users, including clients and hubs, are rational, selfish, and potentially malicious, i.e., they may be malicious and attempt to steal funds and deviate from the payment protocol, if it benefits them. Hubs may intentionally fail ongoing token exchanges, keep funds from senders without exchanging and forwarding them to the receiver, or overcharge transaction fees. Malicious clients may collude to keep sending cross-chain transactions through a certain hub in one direction (e.g., always from B_1 to B_2). This attack will exhaust one type of token of the hub and make it fail to serve as a hub.

Following Byzantine fault-tolerant settings, we assume the proportion of adversaries is less than 33% of the total number of consensus participants of both blockchains and the committee of auditors. The delay Δ of posting consensus information to a blockchain depends on the block generation speed. The block generation speeds vary a lot among different blockchains and might change over time.

C. Design Objectives

XHub achieves the following design objectives.

Availability and auditability: If hubs, clients, and auditors follow the XHub protocols, they will succeed in making payments or get profits from doing the services. Auditability provides resilience to *denial-of-service* (DoS) and *counterfeiting* attacks. Although malicious hubs cannot steal funds from clients due to atomicity, they still can attract clients to select them as relays and fail to forward payments. Those failures will be detected by clients and clients can dispute them to the auditors. The auditors will decrease the reputation of any misbehaving hub based on the dispute results. Clients can find the latest reputations of all hubs and avoid selecting those with low reputations. Auditability also guarantees detection and penalty of counterfeiting, i.e., a client, hub, or auditor reporting incorrect information.

Atomicity: Atomicity ensures that in a cross-chain transaction, all the payments along the path will succeed together or all fail. It guarantees that honest users will not get any loss even if there exist malicious parties.

Unlinkability: Unlinkability ensures that if there are multiple cross-chain transactions happening through one hub, the hub cannot determine the sender-receiver pairs better than a random guess.

Performance: The main performance goal of cross-chain payment hubs includes low latency for cross-chain transactions, high scalability, and high success rate.

III. Protocol Design of XHub

This section presents the design of the protocols of XHub.

A. Design Overview

XHub is a network architecture that supports clients to select preferred hubs for cross-chain transactions while preserving security properties such as atomicity. The key idea is to maintain public-available and trustworthy information about the hubs, including their reputation scores, and exchange rates, with the help of a committee of decentralized auditors. XHub includes the following protocols.

1) Auditor communication protocol supports the functions of a committee of auditors, which register and manages hub information including their reputations and exchange rates, responds to queries of this information, and handles disputes from victim clients. A client or hub needs to broadcast to the whole committee of auditors or let an arbitrary auditor broadcast to other auditors.

2) Hub registration protocol allows a user of two blockchains to register as a hub.

3) Transaction protocol allows two clients in different blockchains to make a transaction via a hub. It includes three components: hub selection, intra-blockchain routing, and cross-chain transaction.

4) Hub management protocol allows auditors, hubs, and clients to manage trustworthy information of hubs including their reputations and exchange rates. Misbehaving hubs will be penalized by lowering their reputations.

B. Auditor communication protocol

The committee of auditors manages the exchange rates, reputations, and collateral of all hubs together. All the auditors jointly maintain a multi-signature (multisig) wallet in each blockchain using the multisig protocol proposed in Bitcoin [21]. It performs a write operation to a blockchain when a threshold number of signatures are successfully collected from the auditors. In XHub, the threshold is set to be 2/3 of all auditors. Hence at least 2/3 of the auditors are required to sign each verification or update message for the information of a hub before sending the message to the blockchain. A user of both blockchains may register as an auditor to get profits when they correctly provide signatures. When a user registers as an auditor, they need to put collateral in the multisig wallet. Only the auditors who correctly sign the messages can get profit [14], [29], which is from the fees of hub registrations, exchange rate updates, reputation updates, and disputes.

Fig. 2 shows the communication processes related to the auditors. The auditor communication protocol achieves *Byzantine agreements*, because we do not assume that all auditors are honest and available all the time. In XHub, we follow the design of Byzantine agreement protocol in Algorand [14],

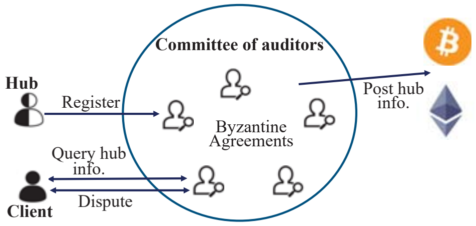


Fig. 2. Auditor communication protocol.

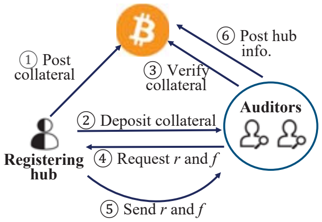


Fig. 3. Hub registration protocol.

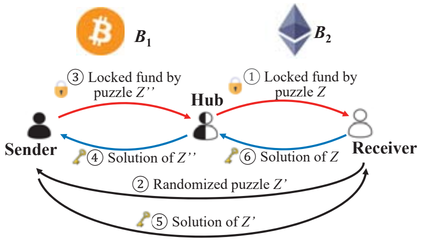


Fig. 4. Cross-chain transaction by A^{2}L [20].

to achieve Byzantine fault tolerance among the auditors. This solution can tolerate up to 1/3 faulty auditors.

In addition, auditors also communicate with hubs, clients, and blockchains in other protocols of XHub including the hub registration and hub management protocols. Those processes will be detailed in later sections.

C. Hub registration protocol

Any user with wallets in both blockchains can register as a hub and get profit by forwarding payments. To register as a hub, a user deposits collateral to the multisig wallets of auditors in both two blockchains. The value of the collateral is pre-determined by the system and sufficient to cover dispute fees until its reputation reaches a very low value.

Figure 3 shows the hub registration protocol, which includes the following steps.

1) The registering user h first sends two transactions of putting collateral $\texttt{col}_h$ to the multisig wallets of the auditors in both blockchains B_1 and B_2 respectively.

2) h keeps monitoring the two blockchains until the deposit transactions are posted. It generates two corresponding proofs of inclusion and sends them to an arbitrary auditor. The proof is constructed by signing the transaction id and the transaction data with h's private key.

3) Upon receiving the proof, each auditor searches both blockchains to verify if the deposit transactions are in the blockchains. XHub does not make each auditor download the whole blockchains to search for the transactions, because it costs a large amount of storage, computation, bandwidth, as well as long delays [30]. We use the *simplified payment verification* (SPV) protocol [1]. Instead of downloading the whole blockchain, auditors download only the header of each block, which contains the root of a Merkle tree [31] of transactions. To verify the correct inclusion of a deposit transaction, it is sufficient to provide the Merkle tree path from the root to the leaf containing the transaction of the corresponding block.

4) Once an auditor successfully verifies the two deposit transactions, it will request the registering hub for its exchange rate r and transaction fee f.

5) Upon receiving r and f, the auditor will broadcast to the committee a signed *New_Hub* message, including the hub's address, r, f, and a default reputation R_h.

6) When the auditor obtains signatures of the message from at least 2/3 of the auditors, it sends the *New_Hub* message with these signatures to the two blockchains respectively. The inclusion of the *New_Hub* message in the blockchains indicates the successful registration of the hub.

D. Transaction protocol

The transaction protocol supports that a sender in B_1 spends funds in an amount of ψ_1 and a receiver in B_2 receives funds in an amount of ψ_2. Note $\psi_2 = r_{12}(\psi_1 - f)$, where r_{12} is the exchange rate from B_1 to B_2 and f is the transaction fee charged by the hub.

Hub selection. When two clients want to make a cross-chain transaction, they need to first select a hub as the relay. There are three factors to consider: the exchange rate, transaction fee, and reputation score for each hub, and select the one that is considered ideal. The sender prefers a high exchange rate to pay fewer funds, a low transaction fee, and a high reputation – note an unsuccessful payment by a malicious hub does not make the sender's funds be stolen, but causes extra time to dispute and select another hub. Each client can self-define a 3-tuple (α, β, γ) to calculate a score $(\alpha R + \beta r_h - \gamma f_h)$ for each hub h and select the one with the highest score. After selecting the relay hub, the sender needs to confirm with the receiver to guarantee this hub is correctly registered in the other blockchain with the same reputation, exchange rate, and transaction fee information. If not, the sender needs to select another hub.

Intra-blockchain routing. After selecting the hub, the sender needs to make the payment of the corresponding amount of funds to the hub first, via a direct link or multi-hop path. We use a decentralized routing protocol [28] to find a payment path between two users in the same blockchain. Similarly, the hub uses the same routing protocol to find a path to the receiver in the other blockchain.

Cross-chain transaction. One important security requirement when a hub forwards payments between two blockchains is atomicity. Hubs are not necessarily honest and, in particular, they might attempt to steal money from clients, such as withholding funds from the sender without relaying them to the receiver, or overcharging in conversion fees than what they are allowed to. Atomicity guarantees that either a transaction of the correct amount is successful or payment funds go back to the original sender. In addition, a further requirement is unlinkability, which ensures that if there are multiple cross-chain transactions happening through one hub, the hub cannot determine the sender-receiver pairs better than a random guess [32], [33]. The cross-chain transaction step of XHub is developed by a recently proposed solution called anonymous atomic locks (A^{2}L) [20] that achieves both atomicity and unlinkability at a single hub. A^{2}L cannot achieve availability or auditability: a malicious hub can keep failing payments without penalty.

We briefly present the protocol of a cross-chain transaction as shown in Fig. 4.

1) Suppose a sender A wants to send a cross-chain payment to a receiver B. The selected hub h first creates a fresh cryptographic puzzle Z and its corresponding solution. The hub then sends a locked fund $\texttt{Lock}(h, B, \psi_2, Z, T_2)$ with this puzzle to B, indicating that B can receive the payment in the amount of ψ_2 from the hub only if he provides the correct solution to Z within time T_2.

2) B randomizes Z into a new puzzle Z' using a randomness $rand$ and sends Z' to A. A^{2}L applies a *homomorphic cryptographic scheme* to guarantee that the solution of Z can be obtained using the solution of Z' and $rand$, but one cannot link Z with Z'.

3) A re-randomizes Z' to a new puzzle Z'' with a randomness $rand'$ and sends a locked fund $\texttt{Lock}(h, A, \psi_1, Z'', T_1)$ to h, indicating that h can get the funds if it provides the solution of Z''.

4) h can solve Z'' using a universal trapdoor tp but cannot link Z'' with Z. h then provides the solution of Z'' and get the payment from A.

5) A computes the solution of Z' from the solution of Z'' and $rand'$ and sends the solution to B.

6) B computes the solution of Z from the solution of Z' and $rand$ and sends the solution to h. Then B gets the payment from h.

In the end, B_2 receives funds in an amount of ψ_2. If any of the three parties fails to perform the correct operations, the whole transaction will fail but no one loses funds. In addition, the party that causes the failure will be detected by others and reported to auditors.

In the above protocol, the communication between A and h and B and h can be direct Internet packet exchanges, but the payments from h to B in 1) and from A to h in 3) could take one or more hops in the PCN, based on intra-blockchain routing. We further implement payment forwarding at every hop using A^{2}L for unlinkability within a blockchain.

In practice, observations show that the exchange rate between the two cryptocurrencies may be susceptible to strong fluctuations. Hence XHub locks the exchange rate once a transaction is set up until the transaction completes.

E. Hub management protocol

To achieve the availability of hub services and auditability of hub behaviors, the reputations of hubs should be correctly managed in XHub and accessible to clients.

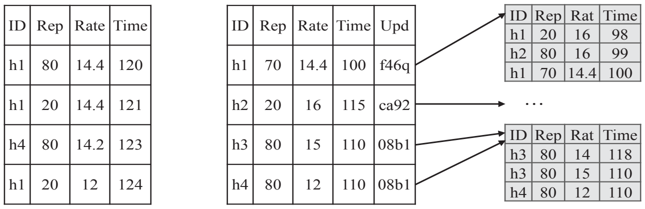
For all new hubs that join XHub, they are assigned the same initial reputation $\hat{r}$. The reputation of a hub is updated for each time epoch. If the misbehavior of a hub is disputed by a client and verified by auditors in an epoch, the reputation of this hub will decrease by 50%. If a hub keeps behaving correctly for 10 epochs, they can request the auditors to raise its reputation by 5% by paying a transaction fee. Note that there is always a latency to post the new reputation to the blockchains. Hence, clients may contact an arbitrary auditor for the latest reputation in the current epoch. Later clients can verify this in blockchains. If a hub does not respond to a transaction request from the beginning, it is not considered misbehavior because the hub can be offline. The clients can easily choose another hub. However, a hub cannot fail on Step 4) of the cross-chain transaction protocol, i.e., providing a correct solution of Z''. Besides, since auditors can be arbitrary users with enough collateral in both blockchains, an auditor could be offline or malicious. In order to mitigate the potential influence on client requests, clients can always make multiple queries to several auditors. In this way, they can detect the malicious hub that provided the wrong hub information. Once such misbehavior is identified, clients can submit disputes to other auditors for compensation and expel the malicious hub from the committee. Furthermore, if clients notice an auditor who remains unresponsive, they can also dispute to remove it from the committee.

Dispute handling. When a hub fails to provide a correct solution of Z'', intentionally or unintentionally, this event is guaranteed to be detected by the sender and receiver, based on the transaction protocol. Although the sender does not lose funds due to the atomicity of the protocol, the clients can dispute this event to decrease the reputation of the hub. In order to incentivize the clients and auditors to do so, if auditors successfully verify a misbehavior and update the reputation to the blockchains, this hub's collateral will be used to compensate the users and auditors.

We first discuss the case of the hub h providing a wrong solution of Z''. When a sender A detects that a hub h fails to perform the transaction protocol, e.g., h sends a wrong solution of Z'', A can send a dispute message $D_A = \{msg_{k_h}||Z''||col_A\}$ to an arbitrary auditor U, where msg_{k_h} is the signed message from h including the wrong solution of Z'' and col_A is A's collateral. If the dispute launched by A is incorrect, A will get punished by losing her collateral. This prevents clients from abusing the dispute process. Upon receiving the dispute message D_A, the auditor will broadcast D_A to other auditors and then verify if the solution from h is correct. If the solution is incorrect, the auditor signs a $dispute\_success$ message and broadcasts it to all auditors. When 2/3 of the auditors sign $dispute\_success$ messages, a reputation update can be posted to the blockchains.

The second case is that the hub h does not send the solution of Z'' to A. A will first try extra x attempts of requesting the solution of Z'', where x is a random integer in $[1, 5]$. If there is still no response, A sends a dispute message to an auditor. After receiving A's dispute, an auditor will also send Z'' to h and ask for the solution. Note the transaction protocol achieves unlinkability, hence h cannot tell whether a request is from A or an auditor. If h intentionally declines to send the solution of Z'', even with a very small probability, this misbehavior will eventually be detected by the auditor. The only way h can avoid being detected is by always responding with the correct solution of Z''. When 2/3 of the auditors detect that h declines to provide the solution, a reputation update can be posted to the blockchains.

Exchange rate management. Each hub can set its own exchange rate r. It can simply set r to the market rate or use the



ID	Rep	Rate	Time
h1	80	14.4	120
h1	20	14.4	121
h4	80	14.2	123
h1	20	12	124

(a) Update Table

ID	Rep	Rate	Time	Upd
h1	70	14.4	100	f46q
h2	20	16	115	ca92
h3	80	15	110	08b1
h4	80	12	110	08b1

ID	Rep	Rat	Time
h1	20	16	98
h2	80	16	99
h1	70	14.4	100

ID	Rep	Rat	Time
h3	80	14	118
h3	80	15	110
h4	80	12	110

(b) Hub Table

Fig. 5. Example of the update table and hub information table on an auditor.

exchange rate to balance its two tokens in both blockchains. Token balancing is necessary because if one token is depleted, the hub cannot relay any transactions between two blockchains. Assume that r_{12} is the rate such that the sender pays the hub 1 Ψ_1 token and the hub will pay the receiver r_{12} Ψ_2 tokens. We have $r_{21} = \frac{1}{r_{12}}$. For example, if a hub has more Ψ_2 tokens than Ψ_1 tokens, the exchange rate r_{12} can be higher than the market price to encourage clients to make payments in Ψ_1 tokens, while the exchange rate r_{21} can be lower than the market price.

Hub information management. In every epoch, all active hubs can update their exchange rate by sending the new one to auditors. Auditors maintain a local table that stores the exchange rate and reputation information of all hubs that will also be posted to the blockchains. Auditors might update the hub information every epoch with the current timestamp and their signatures. The updated hub table needs to be signed by at least 2/3 of the auditors to be put in the blockchains.

Using blockchain to maintain hub information raises two performance problems. 1) An update of hub information takes a considerable amount of time to be available on a blockchain, due to its long processing time. Hence the hub information on the blockchains might not be the latest. 2) The hub information including hub reputation and exchange rates might change frequently, resulting in both a large amount of update requests and a large size of blockchain data to be posted.

To resolve the above problems, we propose a hybrid solution that combines the strengths of trustworthy information on blockchains and large storage capacity on auditors. Auditors locally maintain a hub table storing all hub information and an update table recording all the hub updates in the current epoch. At the end of each epoch, auditors send the digest of these two tables to the blockchains. Users query hub information directly from auditors and verify the correctness by checking the blockchains after the digest is processed by the blockchain and posted. In this way, users can access the latest hub information while still benefiting from the security and trust provided by blockchains. Furthermore, the combination of blockchains and auditors ensures that hub information can be updated and maintained in a timely and efficient manner, reducing the risk of failed transactions due to outdated or incorrect hub information. We provide an example of the hub table and update table in Fig. 5. The Hub table maintains the latest hub information at the end of the previous epoch. For each hub, the *Upd* entry indicates the time of the last update and includes a link to the corresponding update table in the corresponding epoch. The update table records every hub update in the current epoch. The update message needs to be signed by at least 2/3 of the auditors in order to be confirmed in blockchains.

User query. A user can query an arbitrary auditor for the hub information. The auditor sends the current update table and hub table with the timestamp and its signature. The user searches the blockchain to check if this version of the tables has been confirmed. If this version exists in the blockchains, the user computes the table digest to verify its correctness. If this version does not exist, the user can ask the auditor for the most recent version of the tables that is available on the blockchains. If that version passes the digest checking, the user can temporarily trust the information from the auditor and wait for some time to check its correctness later. In the future, if the information is proved to be incorrect by the blockchains, the user can dispute the misbehavior of the auditor and the auditor will be removed by the committee.

F. Multiple Blockchains

XHub can be extended to the scenario of more than two blockchains, as long as there exist hubs and auditors between any two of them. For hubs that have wallets for multiple blockchains and are willing to work between all of them, they must register in each blockchain and set up exchange rates and transaction fees for every pair of blockchains. Hub reputation in each blockchain pair is independently managed by auditor committees responsible for that pair. Auditors who can serve between multiple blockchains, similarly, will participate in one auditor committee for each blockchain pair. And they have to maintain one hub information table for each pair.

IV. Protocol Security Analysis

A. Availability

The availability of XHub refers to the property that any party who fails to follow the protocols will be detected by others and (eventually) be excluded from the system. Hence users can receive the services of XHub. First, we show that:

Proposition 1. *A registering hub that follows the registration protocol to lock collateral is guaranteed to be posted to the public as a valid hub even if there exist malicious auditors.*

This proposition holds based on the property of the multisig wallets. Suppose t_{lock} denotes the transaction of a registering hub locking funds to a multisig wallet of the auditors. t_{lock} is guaranteed to be confirmed if more than 2/3 of auditors are honest.The multisig protocol settles a transaction to a blockchain when signatures are successfully collected from more than 2/3 of the auditors [21]. Hence if more than 2/3 of auditors are honest, t_{lock} is guaranteed to be confirmed on the blockchain. When the transactions of both blockchains are confirmed, the hub is successfully registered.

Proposition 2. *All the clients have access to the information of a hub that is successfully registered.*

Clients always query hub information from several auditors. If they notice the hub information from some auditors is not consistent, they will follow the one with a correct digest in the blockchains. Since we assume 2/3 of the auditors are honest and trusted, the digest of the correct hub information will receive signatures from at least 2/3 of the auditors and be successfully posted to the blockchain, based on Byzantine agreement protocol [14]. And clients can always get the latest hub information if they query a sufficient number of auditors.

Denial-of-Service (DoS) attacks. There are two main types of DoS attacks. 1) A hub accepts a payment request but fails to complete the transaction protocol by either providing a wrong solution of Z'' or declining to provide the solution. For a wrong solution, the sender may submit a dispute message to an auditor. If the solution is incorrect, the auditor signs a $dispute\_success$ message and broadcasts it to all auditors. When 2/3 of the auditors sign $dispute\_success$ messages, a reputation update can be posted to the blockchains. For a hub that declines to provide the solution, the dispute protocol allows an auditor to anonymously request the solution. Hence hub either always provides the solution or it will be detected by an auditor. 2) A malicious auditor intentionally refuses to provide the hub information. In this case, clients can simply query another auditor. Moreover, at the end of each time slot, auditors send the digest of both the update and hub tables to the blockchains. Even if an adversary refuses to verify and sign, the system can still rely on the remaining honest auditors to sign the digest and make sure it can be confirmed in the blockchain. If an adversary tries to perform a Sybil attack to submit false tables to the blockchain, it would need to register a large number of auditors to approve this message, which requires it to lock up a large amount of collateral to be effective, making this attack expensive and irrational.

Counterfeiting. When a client request hub information from auditors, a malicious auditor might send a false hub table, creating counterfeit T'_h. However, at the end of the epoch, auditors will work together to put the hub table digest to the blockchains which require verification and signatures from more than 2/3 of auditors. If the malicious auditor tends to put a counterfeit T'_h in the blockchains, it has to compromise more than 2/3 of auditors in the system, which is impractical in the real world. The correct table digest $D(T_h)$ will ultimately be verified and confirmed in the blockchains. Auditability ensures that any client with read access to the blockchains can detect the misbehavior of the malicious auditor by comparing the table digest $D(T_h)$ retrieved from the blockchains with the $D(T'_h)$ computed using T'_h got from the auditor. If two digests do not match, clients can submit proof showing the auditor manipulates false tables.

Stale table. During the middle of an epoch, the update table remains incomplete. When clients query the update table, the current version of the update table is T'_u. But the adversary might send a stale version T^s_u to its own benefit. For example, as shown in Fig.5, at time 122, the current update table T'_{upd} shows that hub h_1 has a reputation of 20. But the malicious auditor could collude with the hub h_1 and send a stale update table T^s_u at time 120 when h_1 had a reputation of 80. After a while, the clients retrieve the digest of the update table $D(T_u)$ from the blockchains and verify the correctness of T^s_u by checking its inclusion in the T_u. Since T^s_u is not fabricated, the adversary can still pass this inclusion test, and clients remain unaware of the stale table T^s_u. To prevent this attack, the system requires that all parties include a timestamp when querying or transmitting hub information. When a client queries the hub information at time t_1, they construct a query with timestamp $Q(t_1)$ and send it to an auditor. The auditor constructs the response by including the latest update table T'_u at t_1, the received request $Q(t_1)$, and its signature. Consequently, the client can confirm that the response corresponds to their request at time t_1 and cannot deny the timestamp. At the end of the epoch, when the auditor sends the digest of the update and hub table to the blockchains or helps to verify the table digest, it will also send a current version of the update table T_u to the client. At the client side, after verifying the correctness of T_u by checking the $D(T_u)$ from the blockchains, the client compares its locally stored T'_u with T_u. If the subset of T_u that all the entries satisfy $t <= t_1$, denoted as $T_{u_{t_1}}$, is larger than the table T'_u, T'_u is detected to be a stale table.

B. Atomicity and unlinkability

Atomicity guarantees that honest parties will make transactions successfully or get all their money back, which ensures balance security for the involved parties. The atomicity and unlinkability properties of XHub relies on the security of the randomized puzzle scheme, as proved in A2L [20]. The hub can only provide the solution of Z'' in order to receive the funds and Z'' will be used by A to generate a solution of Z', which will be used by B to unlock the funds from h. The clients can only steal the funds if they can generate a correct solution to the randomized puzzle, without paying h. However, this breaks the discrete logarithm (DLOG) problem, which is believed to be a hard problem [20]. In addition, the unlinkability is achieved by the fact that the adversary cannot break the indistinguishability of the adaptor signature scheme [20]. And we skip the detailed proof due to space limit.

We now show that the Cross-chain payment hub achieves funds security using the Universal Composability (UC) framework [34] similar to prior work [35], [36] which is based on a system of interactive Turing machines (ITMs). The UC framework includes parties executing the protocol in the real world, ideal functionalities performed by idealized third parties, and a set of adversaries $\mathcal{A}$. A protocol is said to be UC-secure if the real-world execution of the protocol cannot be distinguished from the idealized protocol execution by the environment. The UC framework includes an environment ε, which represents the external world. The environment chooses the inputs given to each ITMs in the system and observes the outputs. The framework also includes honest parties who follow the protocol, and a set of adversary $\mathcal{A}$ who try to break the security of the system. Besides real-world functionalities, the framework also includes ideal functionalities, which act as idealized third parties, and implement some target specifications. They exhibit the desired properties of the protocol. We define the ideal world functionality F_{atomic} for the transaction protocol. The clients and relay hubs interact with F_{atomic} implemented by a trusted third party to perform the cross-chain transactions. F_{atomic} manages a list P to keep track of the cryptographic puzzles and timelocks, and another list K to keep track of the valid key to the puzzles. Atomicity for a cross-chain transaction means that a puzzle can only be solved if there is a corresponding execution of the solution for that puzzle. This is enforced by F_{atomic} because it keeps track of the puzzles in the list P, and checks whether the puzzle matches one of the existing entries in the list P that has already been solved. Since the puzzles can only be solved by a PuzzleSolver function inside F_{atomic} which is trusted, this ensures that PuzzleSolver must be called before checking the validity of the puzzle solution in order for it to succeed.

V. Experimental Evaluation

We evaluate the performance and security of XHub using both simulations and prototype implementation.

A. Methodology

We implement the system prototype of XHub on two blockchains, Bitcoin testnet3 [22] and Ethereum Sepolia testnet [23]. The testnets are real blockchains but the tokens do not have any value. They are used for software testing and research purposes. Both Bitcoin and Ethereum use ECDSA with the secp256k1 Koblitz curve [37], [38], proving native support for the corresponding cryptographic operations. The prototype is mainly used for evaluating the real latency to set up a hub, make transactions, and dispute management. The transaction protocol of XHub is built based on the RELIC library [39] for the cryptographic operations and on the PARI library [40] for the arithmetic operations in class groups.

Our large-scale simulations use two real PCN topology and transaction datasets: Ripple [2] and Lightning [12]. We treat the Ripple data as transactions sent by clients in Ripple and sent to clients in the Lightning network. And Lightning data as transactions from clients in Lightning to those in Ripple. For Ripple, we use the data from January 2021 to December 2021 and get the network with 1,783 users in our simulation. For Lightning, we get the network with 3,519 nodes on one day in January 2022. We assume 500 users are both in Ripple and Lightning networks, and they can volunteer to be relay hubs. We generate payments from Ripple by randomly sampling the Ripple transactions for the sender-receiver pair in Ripple and Lightning respectively. Due to the lack of user information in the Lightning network, we randomly sample the transaction volumes and sender-receiver pairs for transactions from Lightning. We generate cross-chain transactions by randomly selecting transactions from the above two groups of payments.

Metrics. We use average processing latency to evaluate the performance of the prototype system. The processing latency of payment is calculated as the total delay from hub selection to fulfilling a transaction. Similar to prior work [26], [41], we also use the transaction success rate as an evaluation metric for resource utilization, defined as the percentage of successful payments whose demands are met overall generated payments. Note a transaction may fail due to limited funds on payment channels. We report the average results over 10 runs, each of which includes hundreds of communication pairs.

B. Results of cross-chain transactions in real systems

We conduct a testbed evaluation with the XHub prototype of 7 machines including 2 clients, 1 hub, and 4 auditors, running on two real blockchains. In the experiments, we construct a payment path from a client in the Bitcoin testnet to a user in the Ethereum testnet via a hub. We execute several transactions between them. We also have a group of 4 auditors that are both registered in Bitcoin and Ethereum testnets. Table I and II show the performance and cost of different operations of XHub respectively. Hub registrations take approximately 75 minutes with 1, 3, or 4 auditors, and hub information updates take 60 minutes with 1, 3, or 4 auditors. The reason why both operations take a relatively long time to complete is that they require writing operations to real blockchains. For example, each hub registration incurs two transactions in the blockchains: one is to lock collateral to the blockchain and the other is to post the new hub information to the blockchain by the auditors. **The time waiting for confirmation on the blockchains contributes to more than 99.9% of the latency while the XHub protocol execution time is negligible compared to it. Also, both operations can be executed in parallel to save latency.** For example, multiple hubs can register at the same time and the whole process still takes around 75 minutes. The time of a new hub proving the locked collateral to auditors and auditors verifying this information only takes around 105 ms and 729 ms respectively. Hence the number of auditors does not play an important part in the processing latency. Hub information update is to update the reputation and exchange rate information on the blockchains. This information is kept in one table, and this table will be updated every epoch by the auditors. As long as one relay hub has information changes in an epoch, the auditors will post the digest of the new table to the blockchains, which takes around 60 mins. The latency again is dominated by the transaction processing time in the blockchains. The number of auditors does not affect it.

Off-chain operations of XHub that do not involve blockchain transactions have much less latency. The time to make a cross-chain transaction is the time to perform the atomic swap protocol. Before initializing the transaction, two clients need to negotiate and determine the hub with the help of auditors. So it is the network latency that leads to the processing time difference with the varying number of auditors. For dispute management, users need to first send a dispute message to all the auditors, then the auditors check deposits and verify the malicious behaviors, and make the penalty. It requires the participation of all the auditors in every step, and the penalty can be executed only if more than 2/3 of the auditors approve. Even if the system has 4 auditors, the dispute management time is still less than 3 minutes.

C. Results of exchange rate management

We use simulations to demonstrate that the dynamic exchange rate in XHub helps to improve the transaction success rate. A hub can use dynamic rates to encourage clients to make payments in one of the tokens to achieve token balancing, while fixed rates may cause one type of token to be exhausted.

TABLE I
CROSS-CHAIN PAYMENT HUB PERFORMANCE OF XHUB

Operation and Latency	1 Auditor	3 Auditors	4 Auditors
On chain:			
Hub registrations	75 mins	75 mins	75 mins
Hub info updates	60 mins	60 mins	60 mins
Off-chain:			
Cross-chain transaction	592 ms	661 ms	748 ms
Dispute management	1,903 ms	2,439 ms	2,987 ms

TABLE II
COST OF XHUB OPERATIONS ON THE TWO BLOCKCHAINS

	ETH	BTCTEST
Hub registration	0.000032	0.000057
Reputation update	0.000129	0.000158
Exchange fee update	0.000138	0.000174
Dispute	0.000031	0.000092

We compared it with the version of fixed exchange rates, in which all hubs have the same standard exchange rates, constant over time. We assume there is no malicious hub in the system. Thus, each hub has an equal likelihood to be chosen to conduct cross-chain transactions by sender-receiver pairs. Fig. 6 shows that XHub with dynamic exchange rates always achieves higher transaction success rates compared to that using the fixed rate, by varying the numbers of transactions and relay hubs. In Fig. 6(a), we set the number of hubs to 200 and vary the number of transactions, and in Fig. 6(b) we set the number of transactions to 1,000 and vary the number of hubs. With a fixed exchange rate, a hub might become imbalanced in two blockchain tokens when the transactions across it are higher in one direction than the other. Eventually, the hub runs out of one token and cannot support further payments in this direction. On the contrary, in XHub, with adaptive exchange rates, hubs set a good rate to attract the transactions which can make their tokens balance. Hubs are less likely to run out of their funds, and thus, can serve more transactions.

D. Results of reputation management

We evaluate the XHub with different proportions and different behaviors of malicious hubs, and compare the results with and without our reputation mechanism. In this experiment, we use 200 relay hubs and 1,000 transactions in total. We consider the following types of malicious behavior of hubs: 1) hubs provide rational exchange rates, but fail all transactions going through them deliberately; 2) hubs provide rational exchange rates, but fail transactions in one direction; 3) hubs provide rational exchange rates, but fail small transactions below a threshold which provide less profit; 4) hubs provide extremely low exchange rates to attract more transactions, but fail all those transactions; 5) hubs provide extremely low exchange rates to attract more transactions going through them, but fail some of them according to their own interests. Fig. 7(a) shows the performance of XHub varying with the percentage of malicious hubs, ranging from 0% to 50%, with and without reputation management, under different malicious attacks. The figure shows the results of Attacks 1 to 5 without reputation management and XHub under Attack 4 (the one that causes the lowest success rate with no reputation management). Without the reputation mechanism, the success rate of XHub decreases a lot with the growing percentage of malicious hubs. When the percentage of malicious hubs is below 15%, the performance of XHub with reputation management is similar to that with no malicious hub. When the percentage of malicious hubs achieves 50% (unlikely to happen in practice), the success rate of XHub is still above 60%, while the success rate without reputation management is only 30% for Attack 4. In Fig. 7(b) and Fig. 7(c), we provide the transaction success rates by varying the numbers of transactions and hubs. Specifically, we focus on cases where 5% and 10% of the hubs are malicious and perform Attack 4. We believe this focus is reflective of more realistic systems where attackers tend to fail as many transactions as possible, and it is uncommon to encounter a high percentage of malicious hubs. With reputation management, the success rate of 5% and 10% malicious hubs are both close to that of no malicious hub. However, without reputation management, the success rate is significantly lower.

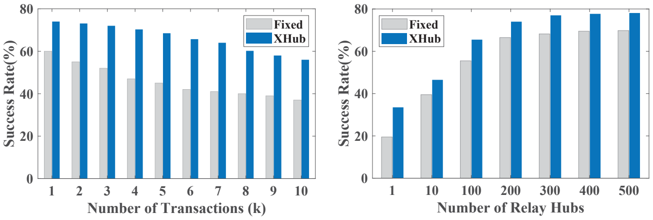
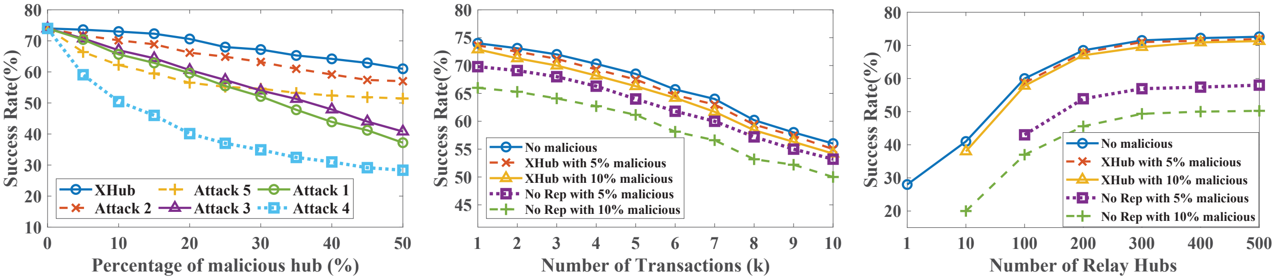


Fig. 6. The success rates with fixed and adaptive exchange rates.

We also monitor the change of reputation scores for both honest and malicious hubs. The initial reputation of each hub is set to be 0.5. The reputation value will be dynamically updated according to the hub behaviors. The percentage of malicious hubs in this set of experiments is set to 5%. Fig. 8(a) shows the reputation changes for 5 randomly chosen honest hubs and Fig. 8(b) shows that of 5 randomly chosen malicious hubs. Hub 1 to 5 in Fig. 8(b) denotes the corresponding malicious behavior from the aforementioned 5 types. We find that honest hubs always gradually achieve the maximum reputation value after 500 epochs, even if they might not be able to fulfill some cross-chain transactions due to their fund limits. On the other hand, the reputation value of malicious hubs will shortly decrease to 0, and they will be excluded from the system. Hubs 4 and 5 experience a rapid decrease in reputation as they attract a larger volume of transactions, and thus will be detected immediately and excluded from the system. While transactions do not frequently route through Hub 1, once it is selected by any transaction, Hub 1 will be detected and get a low reputation. For Hubs 2 and 3, even though they do not misbehave all the time, they can still be detected and penalized with low reputations. Fig. 8(c) shows the number of total transactions served by these 5 malicious hubs and one randomly selected honest hub. After a short duration, those malicious hubs receive fewer cross-chain transaction requests and eventually cannot serve any transaction, while the honest hub can keep serving cross-chain transactions.

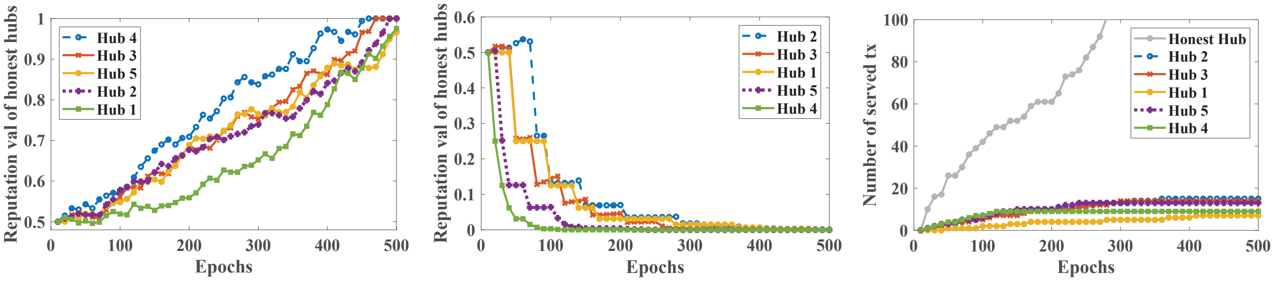
VI. RELATED WORK

Blockchain interoperability, i.e., how to enable multiple parties to exchange tokens across multiple blockchains has



(a) Varying the percentage of malicious hubs (b) Varying the number of transactions (c) Varying the number of hubs

Fig. 7. The success rate with the varying number of transactions and relay hubs with and without reputation mechanism.



(a) Reputation changes of 5 honest hubs (b) Reputation changes of 5 malicious hubs (c) Number of served transactions

Fig. 8. Reputation changes and the number of served transactions of honest and malicious hubs.

been an important problem that attracts increasing attention. Centralized exchange systems are widely used such as Coinbase [10]. However, these services require trust and therefore undermine the decentralized nature of the blockchains. Atomic cross-chain swaps (ACCS) is a mechanism to perform a trustless cross-chain transfer based on hashed timelocks [12], [16]. Although ACCS enables trustless exchanges, they rely on all parties monitoring the blockchain throughout the exchange to ensure security. Moreover, ACCS is vulnerable to packet and transaction memory-pool sniffing, allowing an adversary to exploit blockchain race conditions to steal funds. Many decentralized exchanges remove the need to trust centralized intermediaries for blockchain transfers through the use of ACCS [42]–[44]. However, they only enable the exchange of cryptocurrency assets within a single blockchain [6]. Interledger [45] is a protocol that supports multi-hop payments where each link represents a payment channel defined in a different cryptocurrency. It also relies on the HTLC contract, aiming to ensure payment atomicity across different hops. However, the HTLC contract breaks the unlinkability property and has privacy issues. XCLAIM [46] defines the notion of cryptocurrency-backed assets for blockchains and builds a secure system to construct cryptocurrency-backed assets without trusted intermediaries. It enables one cryptocurrency one-to-one backed by other cryptocurrencies. It suffers from the scalability problem that it cannot support a large number of users back up and construct different cryptocurrency-backed assets. It also leads to large lock-in funds if a user wants to participate in many different blockchains, requiring one backed asset for each individual blockchain. zkBridge [47] designs a trustless cross-chain bridge to move users' funds from one blockchain to another. The similar problem as XCLAIM also exists, large lock-in funds for multiple blockchains. Moreover, cross-chain bridges always require users to have wallets in both blockchains and monitor them. But the overall complexity of managing funds across multiple blockchains can be overwhelming for some users. Different from previous works, XHub is the first to develop the network architecture of flexible cross-chain payments, which considers the problem of hub selection and management and service availability.

VII. Conclusion

Extending the concept of PCNs to support multi-hop paths across multiple blockchains and resolve both interoperability and throughput scalability is an attractive idea. XHub is the first cross-chain PCN architecture to achieve service availability, transaction atomicity, and auditability. We design a series of protocols, including the auditor communication protocol, hub registration protocol, transaction protocol, and hub management protocol. Both prototype implementation and large-scale simulations show that XHub has small latency for cross-chain payments and can achieve a significantly higher success rate compared to the version without hub management protocols. We expect XHub would be an important step for a decentralized transaction system that connects a wide scope of users in different blockchains.

Acknowledgment

The authors were partially supported by NSF Grants 1750704, 1932447, and 2114113. C. Qian was partially supported by the Army Research Office and was accomplished under Grant Number W911NF-20-1-0253. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. We thank our shepherd Dejun Yang and anonymous reviewers for their comments.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Manubot, Tech. Rep., 2009.

[2] F. Armknecht, G. O. Karame, A. Mandal, F. Youssef, and E. Zenner, "Ripple: Overview and outlook," in *Proceedings of Springer TRUST*, 2015.

[3] S. D. Foundation, "Stellar website," http://www.stellar.org/, 2020.

[4] E. Foundation, "Ethereum project," http://www.ethereum.org/, 2020.

[5] "Coinmarketcap," https://coinmarketcap.com/, 2021.

[6] 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," *URL: http://www.opensciencereview. com/papers/123/enablingblockchain-innovations-with-pegged-sidechains*, 2014.

[7] G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White paper*, 2016.

[8] J. Kwon and E. Buchman, "Cosmos whitepaper," *A Netw. Distrib. Ledgers*, 2019.

[9] "Poly network," https://poly.network/, 2020.

[10] Coinbase, "cbeth whitepaper," https://www.coinbase.com/cbeth/whitepaper, 2022.

[11] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer *et al.*, "On scaling decentralized blockchains," in *Proceedings of Springer FC*, 2016.

[12] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016.

[13] "Transaction rate of bitcoin," http://www.blockchain.com/en/charts/transactions-per-second, 2022.

[14] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proceedings of ACM SOSP*, 2017.

[15] X. Zhang and C. Qian, "Towards aggregated payment channel networks," in *IEEE International Conference on Network Protocols (ICNP)*. IEEE, 2022.

[16] M. Herlihy, "Atomic cross-chain swaps," in *Proceedings of the 2018 ACM symposium on principles of distributed computing*, 2018, pp. 245–254.

[17] E. Heilman, L. Alshenibr, F. Baldimtsi, A. Scafuro, and S. Goldberg, "Tumblebit: An untrusted bitcoin-compatible anonymous payment hub," in *Network and distributed system security symposium*, 2017.

[18] S. Dziembowski, L. Eckey, S. Faust, and D. Malinowski, "Perun: Virtual payment hubs over cryptocurrencies," in *Proceedings of IEEE SP*, 2019.

[19] X. Qin, S. Pan, A. Mirzaei, Z. Sui, O. Ersoy, A. Sakzad, M. Esgin, J. K. Liu, J. Yu, and T. H. Yuen, "Blindhub: Bitcoin-compatible privacy-preserving payment channel hubs supporting variable amounts," in *2023 IEEE Symposium on Security and Privacy (SP)*.

[20] E. Tairi, P. Moreno-Sanchez, and M. Maffei, "A2l: Anonymous atomic locks for scalability in payment channel hubs," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 1834–1851.

[21] T. B. community., "M-of-n multisig, multisig output." https://developer.bitcoin.org/devguide/transactions.html, 2020.

[22] "Bitcoin testnet," https://tbtc.bitaps.com/.

[23] "Sepolia testnet," https://github.com/eth-clients/sepolia, 2021.

[24] V. Mavroudis, K. Wüst, A. Dhar, K. Kostiainen, and S. Capkun, "Snappy: Fast on-chain payments with practical collaterals," in *Proceedings of USENIX NDSS*, 2020.

[25] G. Malavolta, P. Moreno-Sanchez, A. Kate, and M. Maffei, "Silentwhispers: Enforcing security and privacy in decentralized credit networks." in *Proceedings of USENIX NDSS*, 2017.

[26] S. Roos, P. Moreno-Sanchez, A. Kate, and I. Goldberg, "Settling payments fast and private: Efficient decentralized routing for path-based transactions," in *Proceedings of USENIX NDSS*, 2017.

[27] V. Sivaraman, S. B. Venkatakrishnan, K. Ruan, P. Negi, L. Yang, R. Mittal, G. Fanti, and M. Alizadeh, "High throughput cryptocurrency routing in payment channel networks," in *Proceedings of USENIX NSDI*, 2020.

[28] X. Zhang, S. Shi, and C. Qian, "Low-overhead routing for offchain networks with high resource utilization," *Proceedings of International Symposium on Reliable Distributed Systems (SRDS)*, 2023.

[29] X. Zhang, Y. Hua, and C. Qian, "Secure decentralized learning with blockchain," in *IEEE International Conference on Mobile Ad-Hoc and Smart Systems (MASS)*. IEEE, 2023.

[30] B. Bünz, L. Kiffer, L. Luu, and M. Zamani, "Flyclient: Super-light clients for cryptocurrencies," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 928–946.

[31] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Advances in Cryptology—CRYPTO'87: Proceedings 7*. Springer, 1988, pp. 369–378.

[32] M. Green and I. Miers, "Bolt: Anonymous payment channels for decentralized currencies," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 473–489.

[33] G. Malavolta, P. Moreno-Sanchez, A. Kate, M. Maffei, and S. Ravi, "Concurrency and privacy with payment-channel networks," in *Proceedings of ACM CCS*, 2017.

[34] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in *Proceedings IEEE Symposium on Foundations of Computer Science*, 2001.

[35] J. Lind, O. Naor, I. Eyal, F. Kelbert, E. G. Sirer, and P. Pietzuch, "Teechain: a secure payment network with asynchronous blockchain access," in *Proceedings of ACM SOSP*, 2019.

[36] S. Dziembowski, S. Faust, and K. Hostáková, "General state channel networks," in *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, 2018.

[37] M. Qu, "Sec 2: Recommended elliptic curve domain parameters," *Certicom Res., Mississauga, ON, Canada, Tech. Rep. SEC2-Ver-0.6*, 1999.

[38] N. Koblitz, "Cm-curves with good cryptographic properties." in *Crypto*, vol. 91. Springer, 1991, pp. 279–287.

[39] D. F. Aranha, "Relic is an efficient library for cryptography," *http://code.google. com/p/relic-toolkit/*, 2020.

[40] U. B. The PARI Group, "Pari/gp version 2.12.0," 2019.

[41] P. Wang, H. Xu, X. Jin, and T. Wang, "Flash: efficient dynamic routing for offchain networks," in *Proceedings of ACM CoNEXT*, 2019.

[42] W. Warren and A. Bandeali, "0xproject whitepaper," https://www.0x.org/pdfs/0x_white_paper.pdf.

[43] M. Oved and D. Mosites, "Airswap whitepaper," https://www.airswap.io/whitepaper.htm.

[44] A. Labs, "Idex: A real-time and high-throughput ethereum smart contract exchange," https://www.allcryptowhitepapers.com/wp-content/uploads/2018/05/IDEX.pdf, 2019.

[45] S. Thomas and E. Schwartz, "A protocol for interledger payments," *https://interledger.org/interledger.pdf*, 2015.

[46] 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)*. IEEE, 2019, pp. 193–210.

[47] T. Xie, J. Zhang, Z. Cheng, F. Zhang, Y. Zhang, Y. Jia, D. Boneh, and D. Song, "zkbridge: Trustless cross-chain bridges made practical," *arXiv preprint arXiv:2210.00264*, 2022.