BlockChain I/O: Enabling Cross-Chain Commerce

Anonymized for peer review

***Abstract*—Blockchains enable users to securely transfer digital tokens without the need for a trusted intermediary, thus fueling the rise of *Decentralized Finance* (DeFi). While initial research on blockchains focused on proving basic security properties such as safety and liveness in a trustless setting, subsequent blockchain designs have also been shown to provide additional desirable properties such as privacy, fairness, and token price stability. However, the current DeFi ecosystem consists of multiple *independent* blockchains, and the desirable properties of individual blockchains do not always generalize to a multi-chain setting. Recently, blockchain designs have been proposed that generalize the security and advanced properties mentioned above to a cross-chain setting. However, there is a lack of an overarching framework whose components provide all of the properties required for a practical platform for cross-chain commerce. Such a framework should make maximum use of synergies between different components, and no component should invalidate a desirable property provided by another.**

In this paper, we present BlockChain I/O to provide such a framework. BlockChain I/O uses entities called *cross-chain services* to relay information between different chains. Cross-chain services cannot violate transaction safety, and are disincentivized from other types of misbehavior – i.e., violating liveness by delaying transactions or violating fairness by misrepresenting information – through an audit system. BlockChain I/O uses stablecoins to mitigate price fluctuations, and a digital ID system to allow users to prove aspects of their identity without violating privacy. After presenting the core architecture of BlockChain I/O, we demonstrate how to use it to implement a *cross-chain marketplace* and discuss how the desirable properties continue to hold in the end-to-end system. Finally, we use an experimental evaluation to demonstrate BlockChain I/O's practical performance in processing an auction.

I. Introduction

Numerous blockchains, distributed ledgers,[1] and Decentralized Finance (DeFi) products have been designed and deployed, resulting in siloed ecosystems since most of these systems do not support interoperation with others by default. To ameliorate the limitations of isolated blockchains, several works have aimed at enabling different extents of intercommunication, interoperation, and integration across blockchains. These include theoretical building blocks as well as functional software artifacts, with some that are limited to specific subsets of systems, whereas others are more general-purpose [40], [15], [3], [16]. Among these, the most basic are generic payload-agnostic inter-blockchain communication mechanisms [14], [20], [39] and protocols for value transfer, payments, or exchanges across blockchains [35], [22], [36] which assume and enable upper-layer applications to orchestrate the business logic. Other examples include dedicated bridges that validate transfers between a given pair of blockchains [12], [38], mechanisms that ensure the security of cross-chain transfers through timelocks [19], [36], and relay chains that support efficient state proofs [27], [6].

[1]For the remainder of this paper, we use the term 'blockchain' loosely to also include other kinds of distributed ledgers even though they are technically a subcategory, e.g., Avalanche's Directed Acyclic Graphs (DAGs) belong to the latter category but not the former [34].

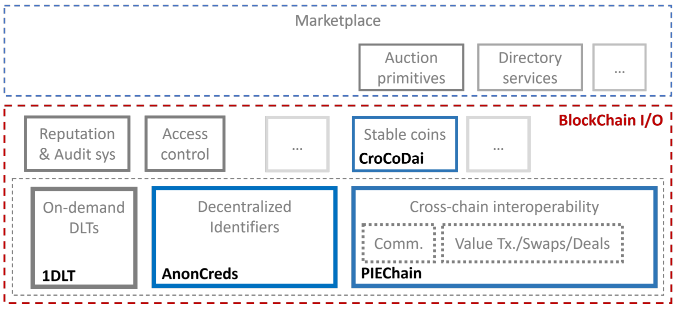

Fig. 1. The BlockChain I/O stack for a versatile cross-chain platform with an overlying open marketplace: This paper primarily focuses on the modules highlighted in **blue**.

The wealth of proposals mentioned above indicates that there is a rich set of existing solutions that ensure the basic security of cross-chain interactions, i.e., that tokens are never stolen (safety) and that transfers are never stalled forever (liveness). However, as in Maslow's hierarchy of needs, there are desirable properties *beyond* basic security in the context of Web 3.0: examples include privacy, fairness, resistance to token price fluctuations, and efficiency while maintaining support for a wide range of use cases. Although approaches exist that address some of these aspects in isolation, such as privacy [9], stablecoin support [30], and generality [31], there is a need for an umbrella framework that achieves all of the desired primitives while ensuring that no properties provided by one component are violated by another component, and that all potential synergies between the components are exploited. The Indian Government's Open Network for Digital Commerce (ONDC [29]) is a useful starting point both for the aspired nature of an open marketplace in general and also in part in determining the list of desired primitives.[2]

In this paper, we enumerate a set of desirable functionalities required as building blocks for versatile cross-chain commerce, and describe a modular technology stack, BlockChain I/O (see Figure 1) that achieves these desiderata. In addition to the aforementioned basic functions of

[2]See Fig. 7 in [29] for a visualisation of the envisioned system, including, e.g., privacy protection, inter-network interoperability, and a reputation ledger.

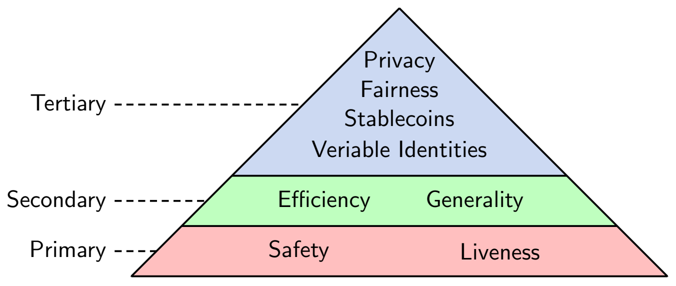

Fig. 2. A hierarchy of requirements for a decentralized marketplace.

intercommunication, value transfers, and cross-chain swaps and deals (which are realized in BLOCKCHAIN I/O through an extension of PIECHAIN [31]),[3] we identify further functionalities that are required for an infrastructure that can support versatile cross-chain commerce. Furthermore, we propose ways to achieve a key subset of these functionalities; details of which and how they interact with each other comprise the main technical contributions of this paper. We note that in the short term, BLOCKCHAIN I/O complements ONDC and other government-run open infrastructures as follows: 1) by providing integration with existing blockchains and cryptocurrency platforms, which already have considerable usage 2) by decoupling the marketplace from centralized constructs (e.g., the Aadhar identity service), instead enabling a more decentralized multi-authority system (where the government ID services might also be used, but not exclusively), and 3) to allay user concerns about government interference or censorship. Furthermore, BLOCKCHAIN I/O can be integrated with such infrastructures in the long term.

We consider three tiers of desirable properties for cross-chain commerce (see also Figure 2 for a visualization).

Primary Properties. The first and most basic desirable properties of a cross-chain framework are the traditional security properties of **safety** (i.e., bad things never happen) and **liveness** (i.e., good things eventually happen). In cross-chain commerce, in which multiple parties agree to an exchange of tokens, the essential safety requirement is that token trades are *atomic* – i.e., all transfers in the trade are committed or none are. This ensures that users who send tokens also receive the tokens agreed upon in the deal, i.e., if the winning bidder in an e-commerce auction sends a payment then she also receives the listed asset from the seller and vice versa. Primary properties are assumed to be a core requirement of the overwhelming majority of users, i.e., a framework that does not provide these properties would not be commercially viable.

Secondary Properties. Once basic security is attained, the second tier of desirable properties require that the system provides **efficiency** (i.e., transaction times or costs are not prohibitive), and **generality** (i.e., a wide range of use cases and ledger protocols are supported). Secondary properties are assumed to be a requirement of many but not all users – although a system that fails to satisfy one of these properties may still have use for a niche of users, it would struggle to achieve mass adoption.

Tertiary Properties. Finally, to achieve mass adoption, a cross-chain platform must also provide properties that are requirements for large minorities of users. A first example of such a property is for users to be assured that the entities with whom they interact have certain attributes, e.g., that they are real people or licensed companies, that they have a demonstrable track record in their field, or that they have the correct age or country of residence. This is achieved through **verifiable identities**. Challenges include the multitude of different ledgers that provide this information, and maintaining user **privacy** through pseudonymity. A second example is that the currency that is used for transactions can be used across chains by default, and that it provides stability against DeFi's notorious token price volatility or catastrophic failure of smaller blockchains and associated cryptocurrencies. This is achieved by support for a cross-chain **stablecoin**. A third and final example of a tertiary desirable property is that exchanges are **fair** – i.e., although basic safety guarantees that users cannot lose tokens without receiving an agreed compensation, the deal itself must have been established fairly, i.e., without scope for manipulation or bribery by other parties.

Blockchain I/O

In this paper, we present BLOCKCHAIN I/O to achieve all of the above desiderata in a unified framework. BLOCKCHAIN I/O uses PIECHAIN to ensure safety in a cross-chain setting: dedicated entities called *cross-chain services* (CC-SVCs) relay relevant information, e.g., bids, asset prices, and identity information, between nodes. Once the outline of a deal has become clear, e.g., after an auction has terminated, a CC-SVC sends a tentative exchange of tokens to the involved blockchains, and the users lock their tokens in escrow. Users then *vote* to commit if the exchange of tokens is agreeable, and abort otherwise. As such, BLOCKCHAIN I/O facilitates *cross-chain deals* [20], which have proven security guarantees. If a CC-SVC violates liveness by going offline while processing a deal, then this is provable to nodes who monitor the relevant blockchains. BLOCKCHAIN I/O utilizes a separate class of nodes called *auditors* to detect this misbehavior and relay it to a governance layer – the resulting reputation damage provides an incentive for CC-SVCs to behave honestly. In BLOCKCHAIN I/O, the same auditors who detect liveness violations are used to detect fairness violations, allowing for the same reputation infrastructure to be leveraged for multiple purposes. To provide decentralized identities that preserve privacy, we utilize Hyperledger AnonCreds [21]. For a cross-chain stablecoin, we utilize a recent proposal, CROCODAI [30], that can be implemented for cross-chain commerce and integrated with the core interoperability module, PIECHAIN.

To illustrate BLOCKCHAIN I/O's generality, we present two use cases from cross-chain commerce whose challenges are overcome by the framework's components. Furthermore, to validate the implementation of our ideas and viability of a

[3]Presented as a demo, the original PIECHAIN work [31] focused on the UI & example uses of the system, not on technical description and evaluation.

versatile platform for cross-chain commerce, we implement a decentralized marketplace that allows users to create and bid on token listings – this implementation provides a proof-of-concept and drives our performance benchmark experiments.

In summary, our contributions are as follows.

- We present BLOCKCHAIN I/O, a framework for cross-chain commerce (Section IV). We present a three-tier stack of desirable properties for cross-chain commerce, and discuss how the components of BLOCKCHAIN I/O provide these properties.
- We present an implementation of a decentralized marketplace that is built using BLOCKCHAIN I/O (Section V), and provide the results of benchmark experiments to show that it has practical performance (Section VI).

The rest of this paper is organized as follows. Section II presents the background and related work. Section III discusses two use cases for cross-chain commerce that require BLOCKCHAIN I/O's properties. Section IV presents the core architecture of BLOCKCHAIN I/O. Section V presents a decentralized marketplace built on top of BLOCKCHAIN I/O. Section VI presents the performance benchmark results and Section VII concludes the paper.

II. BACKGROUND & RELATED WORK

In this section, we discuss existing cross-chain mechanisms and related work on digital identities and reputation systems. We also highlight the differences between BLOCKCHAIN I/O and related frameworks, such as Polkadot and Cosmos.

A. Blockchains & Blockchain Interoperability

Blockchains are decentralized, append-only ledgers that can be used to track the ownership of digital tokens. Each individual blockchain is an ordered list of *transactions* that are grouped into blocks, and each block contains a reference to a previous block, forming a chain. A transaction may represent a change of tokens, or a call to a *smart contract* which is a software program on the blockchain. Smart contracts can be used to implement the logic of e-commerce concepts such as auctions and listings. Blockchains transactions are atomic by design, i.e., if a single transaction consists of multiple steps, then if any step is committed/aborted then all are. However, atomicity cannot be guaranteed by default in *cross-chain systems* where transactions may involve steps on different blockchains, which has led to the emergence of dedicated blockchain interoperability solutions.

Existing interoperability solutions can be typically categorized as sidechains, relays, notary schemes, or ledgers of ledgers [3]. A *sidechain* is a blockchain that interacts with another (typically primary) blockchain as an extension, aiming to improve its scalability or interoperability. Major examples of sidechains in the context of Bitcoin include RSK [25] and the Liquid Network [28]. A *notary scheme* is a system where an entity initiates a transaction on one blockchain in response to a specific event occurring on another blockchain – one example is the PIECHAIN framework that we use in the core architecture of Section IV. Similarly, a *relay* refers to a mechanism where a designated entity keeps track of events or transactions on one blockchain and then relays this information to another blockchain. One of the most popular relay solutions, BTC Relay [7], was released in 2016 by the Ethereum Foundation. Finally, a '*ledger of ledgers*' system is one where a central blockchain is connected to multiple other blockchains (known as sidechains or parachains). These blockchains collectively form an interconnected ecosystem. Polkadot [8] exemplifies such a system. It relies on an underlying relay chain for security, which can be used by other parachains (parallel chains), which could in principle run distinct protocols, inducing in effect a logical star topology with the relay chain in the center. Thus Polkadot achieves not only sharding to improve scalability, but also provides support for heterogeneity, and given that all the parachains rely on the same relay chain, the parachains can natively interoperate. However, this does not immediately help in solving the larger problem of facilitating arbitrary existing blockchain pairs which do not share Polkadot's relay chain, which is addressed in BLOCKCHAIN I/O. Similar to Polkadot, Cosmos [23] also uses a central 'hub' which ensures governance at a global level, while supporting parallel chains called 'zones'.

B. Digital Identities

A *digital identity* connects an individual's *attributes*, e.g., her name, age, location, or reputation, to a digital presence such as an online account or public key. Different methods exist for storing and sharing digital identities – for example, they can be provided by a corporation (e.g., Google), by the individual herself, or by a public ledger. Digital identity systems that are decentralized – i.e., not maintained by a single entity – are also called *Self-Sovereign Identity* (SSI) systems [11]. Individual SSI data entries that contain attribute information are called *Decentralized Identifiers* (DIDs), and a database that contains DIDs is commonly called a *Verifiable Data Registry* (VDR). One prominent example of an SSI is the Sovrin Network [37] which uses a public blockchain as a VDR. However, a major challenge in SSI systems is establishing trust between identity issuers and validators [11]: the validator must decide whether a DID and its attribute information come from a trusted source. In Hyperledger AnonCreds [21], this challenge is addressed by assigning the creation and verification of digital identities a variety of user types, particularly *holders* who have digital attribute information, *issuers* who issue DIDs, and *verifiers* who verify DIDs. AnonCreds specifies a set of protocols for zero-knowledge proofs and schema definitions that allow any consortium of users to run an SSI system on a permissioned blockchain, e.g., Hyperledger Indy, where the consortium members have write access and arbitrary users can have read access. AnonCreds is inherently decentralized – subject to acceptance of participants within an ecosystem, arbitrary entities may participate as issuers, in the creation of VDRs, or the creation of DIDs given a VDR. SSI schemes provide privacy through the use of zero-knowledge cryptography to prove attributes from a DID, and accountability because issuers sign the DIDs so that issuers of incorrect DIDs suffer reputation damage.

C. E-Commerce & Reputation Systems

E-commerce refers to the electronic sale of goods or services – a prominent example is an online marketplace in which a website is maintained by a dedicated entity (e.g., eBay or Amazon), and a multitude of independent vendors create listings that allows customers to browse and bid on items. Depending on the marketplace, vendors can set a fixed price for each item (e.g., Amazon), or buyers can bid for the items through an *e-auction* mechanism (e.g., e-Bay). Recent advances in multi-party computation and zero-knowledge cryptography have enabled e-auction approaches that are both privacy-preserving and verifiable [1], [5], thus enabling e-auctions on public blockchains [13].

Trust [2] is a critical factor that determines the success of e-commerce platforms. Vendors can establish trust through repeated interactions with buyers, generating (if successful) positive feedback. In an online marketplace, vendor reputation metrics can be computed automatically from feedback and displayed alongside listings. For example, on eBay, the percentage of positive feedback is displayed on each vendor's account page. Privacy [17] is an important aspect of reputation systems: if user identities are known, then users may avoid giving negative feedback out of fear of retaliation – however, full anonymity may allow vendors to inflate their reputation (or damage their competitors') through dummy accounts – a so-called *Sybil attack*. Recent advances in reputation systems include a blockchain-based e-commerce platform in which buy orders are pooled and sellers compete to fill the order [26]: this raises the cost of Sybil attacks as fake buyer(s) who collude with sellers to boost the seller's rating risk being obligated to purchase a real item if an honest seller wins the auction. Finally, Beaver [32] is a decentralized anonymous marketplace in which the cost of a Sybil attack can be made explicit.

III. Decentralized E-Commerce: Use Cases

In this section, we discuss two use cases for a decentralized e-commerce platform and discuss why the components discussed in the introduction provide non-trivial solutions to the challenges.

A. Scalping-Resistant Ticket Sales

Ticket scalping refers to the practice where tickets are bought by third parties for the sole purpose of re-selling them at a higher price. Although ticket scalping may increase the efficiency of the sales process [33], [4], it is generally regarded as unfair by customers who observe tickets that were previously affordable being sold at prices that are (far) beyond their budget, and by vendors whose potential profits are usurped by a different entity [4]. Ticket scalping is non-trivial to avoid in a decentralized marketplace because the entities who make purchases are pseudonymous. For example, a scalper can trivially create a multitude of different accounts to circumvent restrictions on the number of tickets bought per

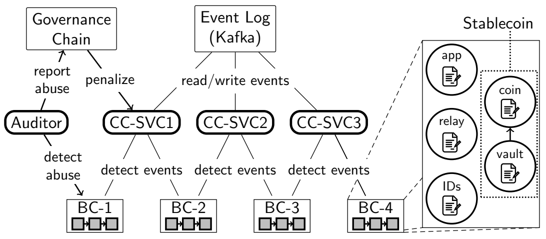

Fig. 3. Interaction between the main components of BlockChain I/O's core architecture.

user, and it is impossible by design to determine whether the customer who uses the ticket paid the original price or a higher price at an external marketplace.

To address ticket scalping, we use the existence of dedicated blockchains that contain identity information to link ticket purchases to digital identities. These blockchains are typically different from the ones on which the ticket is sold and/or the payment is made, so this is necessarily a cross-chain challenge. In particular, as part of the function call that initiates the ticket purchase, the customer must also submit a DID. When the ticket is shown at the event, the customer reveals their name and/or any other associated information using an ID card to prove that it matches the DID, thus thwarting large-scale systematic ticket scalping.

B. Sybil-Resistant Reputations

As discussed in Section II-C, reputation systems that allow users to rate their interactions with vendors may enhance their trust in the marketplace. One challenge in a reputation system is that a vendor may create Sybil accounts to boost its reputation or hurt its competitors' through dishonest feedback. Centralized systems can link each customer or vendor account to a credit card, making large-scale Sybil attacks impractical. However, in a fully decentralized anonymous marketplace, such an approach is impossible by design.

To address the challenge of Sybil attacks, we use DIDs to provide an analogous defense mechanism as a centralized marketplace. In particular, vendors who list an item can include a reputation metric signed by a cross-chain service, such that feedback is included in the metric only if it was issued by a customer who meets certain attributes, such as inclusion on a ledger maintained by trusted (e.g., government) organizations. Although this does not protect against Sybil attacks completely (e.g., a vendor could still ask family members or friends to give favorable feedback) it emulates the level of protection of centralized systems.

IV. The BlockChain I/O Framework

In this section, we describe BlockChain I/O's core architecture and its main components. Figure 3 visualizes the different entities and their interactions, and the different types of smart contracts on each chain.

A. System Components

Blockchains (BCs): BLOCKCHAIN I/O supports multiple independent blockchains that each support their own set of tokens, including the native token (e.g., Ethereum's ETH token) and user-created tokens (e.g., Ethereum's ERC-20 tokens and NFTs). We assume that all blockchains support smart contracts and use the account model for native token balances.[4]

Cross-Chain Services (CC-SVCs): In BLOCKCHAIN I/O, we use the PIECHAIN framework for communication between the underlying blockchains. In PIECHAIN, information is relayed between blockchains by CC-SVCs, which are entities that use full nodes or light clients to detect *events* – i.e., interactions with BLOCKCHAIN I/O smart contracts. Detected events are written to an event log – in PIECHAIN, Apache Kafka is used for this purpose. Users and CC-SVCs can subscribe to events and hence track the interactions with BLOCKCHAIN I/O contracts across the supported blockchains.

In BLOCKCHAIN I/O, all nodes who participate in a deal must submit their tokens for escrow – they are released after all nodes sign a commit vote or after the expiration of a timer. As such, even if CC-SVCs or the event log are compromised, then users are not at risk of having tokens stolen or frozen permanently. Although CC-SVCs can delay the conclusion of cross-chain deals, or misrepresent digital identities, they are disincentivized from doing so by *auditors* (as discussed below). As the same is true for the messaging service, we choose a performance-oriented solution – i.e., Kafka – instead of a security-oriented one such as a private blockchain. The rarity of misbehavior in Ethereum's block proposal market [18], which also relies on relay nodes to transmit information, empirically supports the assumption that relay services are sufficiently disincentivized by reputation damage in practice.

Stablecoins: Stablecoins are tokens whose value is pegged to a real-world asset, e.g., the US dollar. The use of stablecoins for cross-chain deals minimizes the influence of token price fluctuations on users' valuation of the involved assets. For example, if a user were to bid on an auction item using bitcoins, then a sudden change of the bitcoin price could cause the user to reconsider whether winning would lead to an acceptable outcome and abort. Although such a change of heart cannot be ruled out entirely as other offline circumstances may change (e.g., the user's valuation of the item), the use of stablecoins mitigates one prominent source of uncertainty.

To enable stablecoins in BLOCKCHAIN I/O, we use the design of CROCODAI [30] which relies on an optimized (to reduce volatility) portfolio of cryptoassets from multiple chains: customers who need stablecoins can buy them locally or deposit collateral tokens on supported chains. Collateral tokens are stored in dedicated smart contracts called *vaults*, and can be reclaimed later at the cost of paying some interest. If price changes or interest cause the ratio of the collateral's value to the amount of created stablecoins to become too low, then the collateral can be *liquidated* through an auction. Price information about collateral tokens is provided to the vaults by price oracles (e.g., Chainlink or Uniswap contracts). Stablecoins can be transferred between chains if approved by the governance chain (see below). Governance nodes also decide on changes to the system-level parameters, e.g., the interest rate or liquidation ratio. To receive input from the governance layer, supported blockchains have *relay* contracts that validate governance chain messages, e.g., by validating (group) signatures or a zero-knowledge proof-of-state.

Digital Identities: The pseudonyms of each user can be linked to a set of *attributes* that represent important information about the user's identity, e.g., her name, location, or age. We assume that this information itself is stored (typically in encrypted form) on blockchains, e.g., in an *ID contract* or using a dedicated data type. In BLOCKCHAIN I/O, vendors can indicate during the specification of a cross-chain deal which attributes of potential customers must be submitted and what conditions must be met – e.g., to ensure that customers submit (a hash of) their full name to prevent ticket scalping – or they can provide aggregated feedback from users who meet certain attributes to prevent Sybil attacks. As part of any cross-chain deal, the creator must specify which ID ledgers are trusted, and which CC-SVCs are trusted to act as verifiers. The end-to-end flow of obtaining and verifying a DID in BLOCKCHAIN I/O is depicted in Figure 4 and discussed in more detail in Section IV-B.

Governance Chain: This is a special blockchain (which can be an existing blockchain) on which nodes who monitor the underlying blockchains are present. The main role of governance chain nodes is to vote on changes to system-level parameters, approve cross-chain stablecoin transfers, and to verify claims of misbehavior by CC-SVCs.

Auditors: Misbehavior/abuse by CC-SVCs is detected by BLOCKCHAIN I/O users called *auditors*. Since the CC-SVCs' actions are entirely restricted to blockchain actions, misbehavior is provable to entities who have a view of the different blockchains. We identify three main types of misbehavior by CC-SVCs: i) not concluding a cross-chain deal fairly (e.g., not awarding an auction's item to the highest bidder), ii) causing a cross-chain deal to abort by failing to forward messages, and iii) misrepresenting the reputation or attributes of customers or vendors. Upon detecting misbehavior by a CC-SVC, an auditor can submit a claim of misbehavior to a smart contract on the governance chain. If the claim is invalid, then the auditor loses a small deposit, but if it is valid, the CC-SVC suffers a reputation penalty, which hampers the prospects of the CC-SVC being used in the future.

Smart Contracts: As depicted in Figure 3, each blockchain contains five main types of smart contracts: the coin and vault contracts for the stablecoin, the relay contract to receive input from the governance layer, and possibly an IDs contract to store DID information. Finally, a blockchain would also have one or more *app* contracts to implement applications on top of BLOCKCHAIN I/O– in Section V, we give an example of such an app, namely a decentralized marketplace.

[4]For blockchains that do not support smart contract and/or are UTXO-based, e.g., Bitcoin, we assume that a *wrapped* version of its native token exists on a chain that meets our criteria [24].

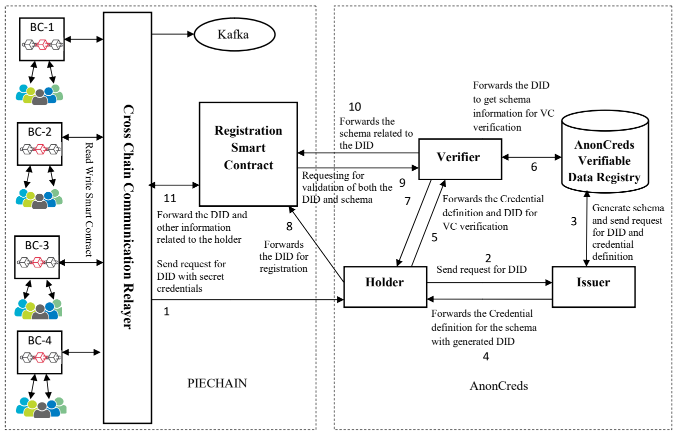
BC-1
BC-2
BC-3
BC-4
Read Write Smart Contract
Cross Chain Communication Relayer
Kafka
Registration Smart Contract
10
Forwards the schema related to the DID
Forwards the DID to get schema information for VC verification
Verifier
6
AnonCreds Verifiable Data Registry
Requesting for validation of both the DID and schema
9
7
11
Forward the DID and other information related to the holder
8
Forwards the DID for registration
Forwards the Credential definition and DID for VC verification
5
3
Generate schema and send request for DID and credential definition
Send request for DID with secret credentials
2
Send request for DID
Holder
Issuer
1
Forwards the Credential definition for the schema with generated DID
4
PIECHAIN
AnonCreds

Fig. 4. PIECHAIN and AnonCreds integration

B. Digital Identifiers

The process of creating and verifying a DID is depicted in Figure 4. We first discuss the entities in this figure, and then the steps of creation (1–4) and verification (5–11).

Entities: In BLOCKCHAIN I/O, a *holder* can be any entity who tries to establish its credentials – e.g., in the cross-chain marketplace, this could be a vendor or bidder. An *issuer* can be any trusted organization that is specified by a participant in a trade. The *VDR*, for which we use Hyperledger AnonCreds [21], stores the DIDs, schema information, credential definition identity, and revocation lists for future reference and the validation of VCs. *Verifiers*, which in BLOCKCHAIN I/O are CC-SVCs, communicate with the Hyperledger AnonCreds VDR [21] and with the *Registration Smart Contract* (RSC), which is a module of PIECHAIN that notifies verifiers through an event listing action.

DID Creation: Prior to the generation of DIDs, the issuer must first determine the schema and credential definition for the holder (which can be reused for many users/holders when suitable). Upon completion, both the issuer and verifier forward their requests to the system pool to obtain their identifiers, verification keys, and roles as Trust Anchor or Trustee respectively (step 1 of Figure 4). After obtaining a suitable role (Trust Anchor), the holder asks the issuer (step 2) to request the AnonCreds VDR to generate the schema of the holder based on some secret credentials provided by the holder using a predefined reusable template (step 3). As a response, a schema ID is returned by AnonCreds VDR. Using this schema ID, the issuer requests the credential definition alongside other information such as tag values and type and revocation information for the schema. Next, a DID is generated internally by the issuer. After getting credential definition information from AnonCreds, the issuer forwards it to the holder along with the DID (step 4).

DID Verification: To verify the DID, the holder first sends a verification request to the verifier (step 5). Next, the verifier obtains confirmation about the schema definition from AnonCreds (step 6), and either sends confirmation to the holder if the schema is valid, and rejects the request otherwise (step 7). If successful, the holder sends the DID to the RSC (step 8), which emits a PIECHAIN event that prompts the verifier to validate the DID (step 9). Next, the verifier sends a confirmation to the RSC (step 10). Finally, the successful verification of the DID is emitted as an event by the RSC (step 11), after which this information can be used by other BLOCKCHAIN I/O smart contracts.

C. Properties of Blockchain I/O

We now discuss how BLOCKCHAIN I/O satisfies the desirable properties mentioned in the introduction.

Primary Properties: BLOCKCHAIN I/O provides *safety* because all involved parties sign a commit vote before any exchange of assets, as such realizing a cross-chain deal as per Herlihy et al. [20]. In particular, CC-SVCs cannot misappropriate tokens as the aggrieved party would not sign the final commit vote. As in [20], tokens are eventually released even if one party drops out or if the CC-SVC fails to transmit any of the messages. A single honest and online CC-SVC can provide liveness for any cross-chain trade, and CC-SVCs are disincentivized from going offline during a trade through the reputation system. If a CC-SVC still does go offline during a trade (e.g., an auction), then the trade can eventually be restarted, while the CC-SVC suffers a reputation penalty.

Secondary Properties: BLOCKCHAIN I/O can support any trade in which digital assets, including cryptocurrencies, are exchanged on multiple chains, as such realizing *generality*. Two examples of use cases supported by BLOCKCHAIN I/O have been presented in Section III. Furthermore, BLOCKCHAIN I/O is efficient as the time costs of cross-chain interactions are in the order of seconds and gas costs are limited. We illustrate this through a case study involving a cross-chain marketplace in Sections V and VI.

Tertiary Properties: BLOCKCHAIN I/O achieves verifiability of user attributes through its AnonCreds component, which also achieves user *privacy* through pseudonymity. BLOCKCHAIN I/O satisfies *stablecoin* support through its CROCODAI integration, and *fairness* through the reputation system. If CC-SVCs violate fairness, e.g., by misrepresenting the winner of a cross-chain auction, or if they sign an incorrect ID proof, then this can be detected and penalized through the same reputation system as for liveness, which exploits a synergy between BLOCKCHAIN I/O and the PIECHAIN and AnonCreds components.

V. DECENTRALIZED MARKETPLACE

In this section, we present a decentralized marketplace built on top of the BLOCKCHAIN I/O infrastructure described in Sections IV and IV-B. We discuss both the core design of the marketplace and the smart contract implementation. We then explain how auditors detect abuse by CC-SVCs, and how to implement the two use cases from Section III.

A. Overview

Users: The marketplace has the following types of users.

- *Bidders* who hold digital (crypto) tokens and who are interested in purchasing listed tokens.
- *Vendors* who want to sell tokens, and who seek to exchange them for (other) digital tokens.

In addition, the core user types of BLOCKCHAIN I/O, i.e., CC-SVCs, auditors, and governance token holders, also participate in the marketplace.

Listings: A listing represents an intent to sell n equivalent tokens – these can be same-sized batches of cryptocurrencies, or NFTs that represent identical goods. Each listing belongs to one of the following common auction [10] or listing types: 1) fixed-price listings, 2) open-bid increasing-price auctions, 3) closed-bid first-price auctions, 4) open-bid decreasing-price auctions, and 5) closed-bid second-price auctions. Listings of type 1, 2, and 4 are resolved through three phases: bidding, conclusion, and feedback, whereas listings of type 3 and 5 are resolved through four phases: bidding, revealing, conclusion, and feedback. The actions in the four phases are as follows:

1) *Bidding.* Bidders submit their intent to purchase (fixed-price), their bid (open-bid auction), or their bid's hash (closed-bid auction), and transfer either the full bid or a minimum amount (the abort penalty) for escrow.
2) *Revealing.* Users reveal their bids, and transfer the remaining value of their bid (i.e., their full bid minus the abort penalty) for escrow.
3) *Conclusion.* A final transfer of assets is proposed, after which the parties who transfer tokens vote to commit if agreeable. Upon commitment or abortion, the tokens are transferred or returned to the intended users. If an exchange is aborted because a user neglects to commit or reveal, then this user loses the abort penalty.
4) *Feedback.* If the token represents physical items whose quality cannot be determined unambiguously, customers may provide feedback on the vendor – e.g., to indicate their opinion of the speed of delivery, whether the item matched the description, etc.

Events: The following types of PIECHAIN events are recorded by the CC-SVCs: 1) *AuctionCreationEvent*, emitted after a new listing has been created, 2) *BiddingAuctionEvent*, emitted after a new bid has been created, and 3) *AuctionEndingEvent*, emitted after a listing has been concluded.

B. Smart Contract Functions

Listings are processed through a single *market* smart contract, which is a type of *app* contract as depicted in Figure 3. The *market* contract has the following functions.

createListing: Takes as input a start time, a reveal time, a conclusion time, a feedback time, CC-SVC addresses, a listing type, and an initial/fixed price. If successful, creates an entry for the listing using a hashmap in the *market* contract.

bidFixed: Takes as input a listing ID. If the listing has the 'fixed' type, the current time is between the start and conclusion times, and the sender has enough stablecoins in her wallet, then a bid is recorded, and an amount of stablecoins equal to the listing's fixed price is transferred to the *market* contract for escrow.

bidOpen: Takes as input a listing ID and bid value. If the listing has the 'open' type, the current time is between the start and conclusion times, the bid value either exceeds the previous highest bid and the starting price (increasing-price auction) or is the first bid (decreasing-price auction), and the sender has enough stablecoins in her wallet, then a bid is recorded, and an amount of stablecoins equal to the bid value is transferred to the *market* contract for escrow.

bidSealed: Takes as input a listing ID and bid hash. If the listing has the 'sealed' type, the current time is between the start and reveal times, and the sender has enough stablecoins in her wallet to pay the abort fee, then a tentative bid is recorded, and an amount of stablecoins equal to the abort fee is transferred to the *market* contract for escrow.

revealBid: Takes as input a listing ID and bid value. If the listing has the 'sealed' type, the current time is between the reveal and conclusion times, the hash of the value equals the hash from *bidSealed*, the sender has enough stablecoins in its wallet to pay the bid value minus the already escrowed abort fee, then a bid is recorded, and an amount of stablecoins equal to the bid value is transferred to the *market* contract for escrow.

finAuction: Takes as input a listing ID, a number of winners, and a price (for second-price auctions). If called by the vendor, and if the current time is between the conclusion and feedback times, then the auction is concluded. Auctions that are not concluded before the feedback time are aborted and all tokens in escrow are returned.

feedback: Takes as input a listing ID and a bid ID. If the auction has been concluded and the current time is between the conclusion and feedback times, then the user's feedback is recorded in a hashmap in the contract.

C. Audits

In the marketplace, CC-SVCs are relied on for faithfully concluding an auction through the *concludeListing* function, and for signing DIDs for personal identifying information and for the feedback aggregates. If they misbehave in any of these roles, then this is detectable by auditors and governance token holders. For example, if they misrepresent the winning bidder, then a higher bid must exist on a chain than the one that was declared the winner. An auditor can send a proof of the existence of this bid to the governance chain, upon which a reputation penalty can be administrated to the CC-SVC. Similarly, if a CC-SVC has misrepresented a DID or reputation aggregate, then this can be demonstrated to the governance chain, as the zero-knowledge proof that was signed as valid by the CC-SVC must be invalid. This creates necessary checks and balances to ensure that end-users can identity and isolate misbehaving CC-SVCs, thus creating a mechanism to satisfy the implicit trust assumptions in PIECHAIN's design.

D. Use Cases

The two use cases of Section III can be implemented by combining DIDs with the smart contract discussed in Section V-B in the following way. For the first use case (i.e., mitigating ticket scalping), each bidder who calls the *bidFixed*, *bidOpen*, or *bidSealed* function also includes a DID that contains a hash of personal identifying information, e.g., the bidder's full name. When redeeming the ticket, the ticket owner can then reveal that it was indeed her who issued the winning bid. For the second use case (i.e., a reputation system that is resilient against Sybil attacks), a vendor includes an aggregate feedback score (signed by a CC-SVC who acts as a verifier) when creating a new auction.

VI. Experiments

In this section, we present our experimental results to demonstrate that the marketplace built using BlockChain I/O (as discussed in Section V) has practical performance. In particular, we evaluate the time and gas costs of the various steps of processing an NFT auction that accepts bids from multiple chains. We display results for an open-bid increasing auction, and focus on the steps that are common between the use cases of Section III. The performance of the other listing types from Section V is similar (e.g., the time and gas costs of the reveal phase in a closed-bid auction resemble those in the bidding phase of an open-bid auction).

A. Experimental Set-Up

We use an iMac with an i9-10900k CPU to simulate an experimental environment with local test networks for Ethereum, Quorum, and Fabric. The NFT is implemented using an **Asset** on the Fabric chain. For the stablecoin, we use a version of the Dai stablecoin for coin transfers based on the coin and relay contracts from [30]. In our experiments, the different agent types all run on the same machine, so our results exclude time costs due to network latency.

B. Experimental Steps

Create Listing. 1) The auctioneer calls the *addAsset* function of an **Asset** contract that is deployed on the Fabric network, which is detected by the CC-SVC. 2) The CC-SVC deploys **Auction** contracts on the Ethereum and Quorum platforms, and publishes an *AuctionCreationEvent* to Kafka. We create a new contract for each asset auction to obtain a worst-case estimate for the cost of creating a new listing.

Issue Bid. 3) A bidder calls *bidOpen* on one of the deployed **Auction** contracts to issue a bid and submit stablecoins for escrow. 4) The CC-SVC detects a bid and successful coin transfer, and publishes a *BiddingAuctionEvent* on Kafka.

Conclude Listing. 5) After the end of the conclusion timer, the auctioneer may conclude the listing by invoking a *closeAuction* call to the **Asset** contract on Fabric. Alternatively, this action can be automatically triggered when the auction reaches its predetermined conclusion time. 6) Upon detection by the CC-SVC, it sends the listing's outcome to the **Auction** contract on each chain (i.e., whether the highest bid on that chain has won or not). This changes the state of these contracts to *ending* – which means that they await further action from the user – and logs this activity as an *AuctionClosingEvent*. (The gas cost has a minor dependence on whether the winner is on the chain or not – the table depicts the costs for the chain with the winner.)

Commit/Abort. 8) The winning bidder either commits or aborts the auction result, which is detected and published on Kafka by the CC-SVC. 9) The CC-SVC forwards the winner's response to the **Asset** contract on Fabric, then either returns or collects the coins transferred by the user in the previous stage. 10) When the related event **AuctionResponse** has been posted by one CC-SVC and received by Kafka, the CC-SVC transfers the asset from the auctioneer to the winner.

Feedback. 11) If the auction result has been committed, then the winner can eventually submit her feedback about the purchased asset to the **Auction** contract on her chain.

C. Experimental Results

An overview of the costs of steps 1-11 are displayed in Table I. For each step, the first three data columns indicate which entities perform an action and what this entails (e.g., a function call), and the last three data columns indicate the time costs (*italic*) and the gas costs for the coin blockchains (**bold**). Quorum's time costs are typically lower than Ethereum's because it has a higher block frequency by default. We observe reasonable time costs: each step takes fewer than 5 seconds, whereas an auction would typically run for more than a day. Furthermore, a bid costs ≈130000 gas, which at current (December 2023) gas prices (≈32 GWei) would cost \$7.50 USD on Ethereum's main chain, but typically (much) less on other EVM-compliant chains (enabling bids on less-congested chains is a core motivation for BlockChain I/O).

VII. Conclusions

We have presented BlockChain I/O, a framework for cross-chain commerce. It satisfies properties from three tiers of importance: primary (safety and liveness), secondary (efficiency and generality) and tertiary (pseudonymous digital identities, stablecoin support, and fairness). We have demonstrated the framework's versatility by creating a decentralized marketplace built on top of BlockChain I/O, hosting a variety of application use cases. We validated our proof-of-concept implementation for functional correctness and by benchmarking the overheads to demonstrate the practicality of the BlockChain I/O framework.

TABLE I
Time costs in seconds (*italic*) and gas costs (**bold**) of the various stages of processing a listing.

		Active Entity			Platforms		
		Auctioneer	Bidder	CC-SVC	Fabric	Ethereum	Quorum
Create Listing	1.	addAsset		detect	*2.03*		
	2.			deploy		*2.55* (**1505408**)	*1.59* (**1505408**)
Issue Bid	3.		bidOpen			*2.02* (**132008**)	*1.01* (**132008**)
	4.			detect		*3.90*	*2.30*
Conclude Listing	5.	closeAuction		detect	*2.02*		
	6.			change state		*2.00* (**29938**)	*1.01* (**29938**)
	7.			detect	*3.00*		
Commit/ Abort	8.		respond	detect		*2.62* (**55974**)	*1.16* (**55974**)
	9.			finAuction	*2.43*		
	10.			transfer	2.34		
Feedback	11.		feedback			2.56 (**73774**)	1.56 (**73518**)

References

[1] Samiran Bag, Feng Hao, Siamak F Shahandashti, and Indranil Ghosh Ray. Seal: Sealed-bid auction without auctioneers. *IEEE Transactions on Information Forensics and Security*, 15:2042–2052, 2019.

[2] Patricia Beatty, Ian Reay, Scott Dick, and James Miller. Consumer trust in e-commerce web sites: A meta-study. *ACM Computing Surveys (CSUR)*, 43(3):1–46, 2011.

[3] Rafael Belchior, André Vasconcelos, Sérgio Guerreiro, and Miguel Correia. A survey on blockchain interoperability: Past, present, and future trends. *ACM Computing Surveys (CSUR)*, 2021.

[4] Jonathan Bell. Ticket scalping: Same old problem with a brand new twist. *Loy. Consumer L. Rev.*, 18, 2005.

[5] Erik-Oliver Blass and Florian Kerschbaum. Strain: A secure auction for blockchains. In *ESORICS*, pages 87–110. Springer, 2018.

[6] Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Mina: Decentralized cryptocurrency at scale. https://docs.minaprotocol.com/assets/technicalWhitepaper.pdf, 2020.

[7] BTC Relay, 2023. http://btcrelay.org/.

[8] Jeff Burdges, Alfonso Cevallos, Peter Czaban, Rob Habermeier, Syed Hosseini, Fabio Lama, Handan Kilinc Alper, Ximin Luo, Fatemeh Shirazi, Alistair Stewart, et al. Overview of Polkadot and its design considerations. *arXiv:2005.13456*, 2020.

[9] Apoorvaa Deshpande and Maurice Herlihy. Privacy-preserving cross-chain atomic swaps. In *Workshop on Trusted Smart Contracts (In Association with Financial Crypto)*, pages 540–549. Springer, 2020.

[10] Jannik Dreier, Hugo Jonker, and Pascal Lafourcade. Defining verifiability in e-auction protocols. In *AsiaCCS*, pages 547–552, 2013.

[11] Kai Jun Eer, Jesus Diaz, and Markulf Kohlweiss. Bottom-up trust registry in self sovereign identity. In *International Congress on Blockchain and Applications*, pages 423–433. Springer, 2022.

[12] Ethereum.org. Blockchain bridges. https://ethereum.org/en/bridges/, last checked, July 2023.

[13] Hisham S Galal and Amr M Youssef. Verifiable sealed-bid auction on the ethereum blockchain. In *Financial Crypto*, pages 265–278. Springer, 2019.

[14] Christopher Goes. The interblockchain communication protocol: An overview. *arXiv:2006.15918*, 2020.

[15] Lewis Gudgeon, Pedro Moreno-Sanchez, Stefanie Roos, Patrick McCorry, and Arthur Gervais. SoK: Off the chain transactions. *IACR Cryptol. ePrint Arch.*, 2019:360, 2019.

[16] Panpan Han, Zheng Yan, Wenxiu Ding, Shufan Fei, and Zhiguo Wan. A survey on cross-chain technologies. *Distrib. Ledger Technol.*, 2(2), jun 2023.

[17] Omar Hasan, Lionel Brunie, and Elisa Bertino. Privacy-preserving reputation systems based on blockchain and other cryptographic building blocks: A survey. *ACM Computing Surveys (CSUR)*, 55(2):1–37, 2022.

[18] Lioba Heimbach, Lucianna Kiffer, Christof Ferreira Torres, and Roger Wattenhofer. Ethereum's proposer-builder separation: Promises and realities. *arXiv preprint arXiv:2305.19037*, 2023.

[19] Maurice Herlihy. Atomic cross-chain swaps. In *ACM PODC*, pages 245–254, 2018.

[20] Maurice Herlihy, Barbara Liskov, and Liuba Shrira. Cross-chain deals and adversarial commerce. *The VLDB journal*, 2022.

[21] Hyperledger AnonCreds, 2022. https://www.hyperledger.org/use/anoncreds.

[22] Xiaofeng Jia, Zhe Yu, Jun Shao, Rongxing Lu, Guiyi Wei, and Zhenguang Liu. Cross-chain virtual payment channels. *IEEE Transactions on Information Forensics and Security*, 18:3401–3413, 2023.

[23] Jae Kwon and Ethan Buchman. Cosmos whitepaper: A network of distributed ledgers. https://wikibitimg.fx994.com/attach/2020/12/16623142020/WBE16623142020_55300.pdf, 2020.

[24] Sung-Shine Lee, Alexandr Murashkin, Martin Derka, and Jan Gorzny. SoK: Not quite water under the bridge: Review of cross-chain bridge hacks. In *IEEE ICBC*, pages 1–14, 2023.

[25] Sergio Demian Lerner, Javier Álvarez Cid-Fuentes, Julian Len, Ramsès Fernàndez-València, Patricio Gallardo, Nicolás Vescovo, Raúl Laprida, Shreemoy Mishra, Federico Jinich, and Diego Masini. RSK: A Bitcoin sidechain with stateful smart-contracts. *Cryptology ePrint Archive*, 2022.

[26] João Martins, Manuel Parente, Mário Amorim-Lopes, Luís Amaral, Gonçalo Figueira, Pedro Rocha, and Pedro Amorim. Fostering customer bargaining and e-procurement through a decentralised marketplace on the blockchain. *IEEE Transactions on Engineering Management*, 69(3):810–824, 2020.

[27] Silvio Micali, Leonid Reyzin, Georgios Vlachos, Riad S Wahby, and Nickolai Zeldovich. Compact certificates of collective knowledge. In *IEEE S&P*, pages 626–641. IEEE, 2021.

[28] Jonas Nick, Andrew Poelstra, and Gregory Sanders. Liquid: A bitcoin sidechain. *Liquid white paper. URL https://blockstream.com/assets/downloads/pdf/liquid-whitepaper. pdf*, 2020.

[29] ONDC.ORG. Open network for digital commerce (strategy paper, chapter 3). https://resources.ondc.org/strategypaper, 2022.

[30] Daniël Reijsbergen, Bretislav Hajek, Tien Tuan Anh Dinh, Jussi Keppo, Hank Korth, and Anwitaman Datta. Crocodai: A stablecoin for cross-chain commerce, 2023.

[31] Daniël Reijsbergen, Aung Maw, Jingchi Zhang, Tien Tuan Anh Dinh, and Anwitaman Datta. Piechain – a practical blockchain interoperability framework, 2023.

[32] Kyle Soska, Albert Kwon, Nicolas Christin, and Srinivas Devadas. Beaver: A decentralized anonymous marketplace with secure reputation. *Cryptology ePrint Archive*, 2016.

[33] Xuanming Su. Optimal pricing with speculators and strategic consumers. *Management Science*, 56(1):25–40, 2010.

[34] Team Rocket, Maofan Yin, Kevin Sekniqi, Robbert van Renesse, and Emin Gün Sirer. Scalable and probabilistic leaderless BFT consensus through metastability. *arXiv preprint arXiv:1906.08936*, 2019.

[35] Stefan Thomas and Evan Schwartz. A protocol for interledger payments. *URL https://interledger.org/interledger.pdf*, 2015.

[36] Sri AravindaKrishnan Thyagarajan, Giulio Malavolta, and Pedro Moreno-Sanchez. Universal atomic swaps: Secure exchange of coins across all blockchains. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1299–1316, 2022.

[37] Phillip J Windley. Sovrin: An identity metasystem for self-sovereign identity. *Frontiers in Blockchain*, page 30, 2021.

[38] Tiancheng Xie, Jiaheng Zhang, Zerui Cheng, Fan Zhang, Yupeng Zhang, Yongzheng Jia, Dan Boneh, and Dawn Song. zkBridge: Trustless cross-chain bridges made practical. *ACM CCS*, 2022.

[39] Yingjie Xue, Di Jin, and Maurice Herlihy. Fault-tolerant and expressive cross-chain swaps. In *ICDCN*, 2023.

[40] Alexei Zamyatin, Mustafa Al-Bassam, Dionysis Zindros, Eleftherios Kokoris-Kogias, Pedro Moreno-Sanchez, Aggelos Kiayias, and William J Knottenbelt. SoK: Communication across distributed ledgers. In *Financial Crypto*, 2021.