Caught in Chains: Claim-First Transactions for Cross-Blockchain Asset Transfers

Michael Borkowski*, Christoph Ritzer‡, Daniel McDonald‡, Stefan Schulte*

* Distributed Systems Group
TU Wien, Vienna, Austria
{m.borkowski, s.schulte}@infosys.tuwien.ac.at

‡ Pantos GmbH
Vienna, Austria
contact@pantos.io

***Abstract*—Blockchains, the fundamental technology upon which cryptocurrencies are implemented, have gained considerable interest in finance, economics, and research. Nevertheless, the numerous blockchains in existence remain mostly unconnected, with no possibilities for interoperability. While approaches for atomic swaps (the atomic exchange of value on two blockchains) are emerging, there is still no documented implementation of a protocol for the transfer of assets from one blockchain to another.**

This white paper formalizes the cross-blockchain proof problem, showing that in practice, it is not possible to verify the existence of specific data on one blockchain from within another blockchain. Based on this, we describe the concept of a cross-blockchain asset transfer protocol using claim-first transactions. This protocol allows for decentralized transfer of assets between blockchains despite the cross-blockchain proof problem.

I. Introduction

In the past years, cryptocurrencies, as well as blockchains, the underlying technology, have gained significant interest in finance and economics, research, and public attention in general [20]. The utility and feasibility of decentralized ledgers has been demonstrated by Bitcoin [12], the first implementation of a blockchain protocol in widespread use. Through its rapid rise in interest and value, Bitcoin also sparked significant investment into research and development related to blockchains and cryptocurrencies, ranging from adding new layers to Bitcoin itself [17], proposing improvements to the Bitcoin codebase [10], or the development of entirely new blockchains [18]. At the same time, increased attention has been given to use cases for blockchains beyond cryptocurrencies, such as runtime verification for business processes [14].

The research field of blockchains is rich and varied, with an ever-increasing number of technologies and implementations. Despite general positive momentum, however, structural problems exist within the blockchain community. The vast amount of blockchains in existence simultaneously causes severe fragmentation of the research and development field. Interoperability between blockchains is mostly impossible, with blockchains instead competing for users and developers [4].

We aim to create a platform for cross-blockchain interoperability. In a first step, we seek to enable cross-blockchain asset transfers in form of PAN, a cross-blockchain token. This token is envisioned to be freely transferable between blockchains by the user in a decentralized manner. The research of such cross-blockchain interoperability is conducted within the Token Atomic Swap Technology (TAST) research project[1].

In the previous TAST white paper [4], we have outlined the current state of the art in the field of blockchains, with special focus on projects and approaches targeting cross-blockchain interoperability. We have found that numerous projects aims at creating decentralized exchanges and implementing atomic swaps. To the best of our knowledge, Metronome [11], which is currently under development, is the only project also aiming at cross-blockchain asset transfers.

In this white paper, we formalize the *cross-blockchain proof problem*, mentioned in our previous work [4]. We then use this as a motivation for the necessity of devising *claim-first transactions*, a novel concept for cross-blockchain asset transfer protocols.

To this end, Section II shows that verifying the existence of specific data on one blockchain from within another blockchain poses requirements which are not possible to meet in practice. Based on this, Section III then presents and describes the concept of claim-first transactions, which allows for the implementation of cross-blockchain asset transfer protocols while avoiding the cross-blockchain proof problem. We also outline practical considerations necessary when implementing such a protocol, and propose solutions to each of these challenges. In Section IV, we provide a brief overview of related work. Finally, in Section V, we conclude our paper, and outline future work.

II. Rooted Blockchains

The cross-blockchain proof problem, mentioned in our previous work [4], states that data recorded on one blockchain cannot be verifiably detected on another blockchain. In this section, we elaborate on this notion in a more structured way.

The intuition behind the concept presented in this section is that in order to verify the presence of specific data on one blockchain, one must pull the blockchain up *by its roots*, i.e., one must verify the entire chain up to the genesis block, in order to achieve definite certainty over the presence of the data in the blockchain.

White Paper, TU Wien; August 2018, version 1.0.

[1]http://www.infosys.tuwien.ac.at/tast/

A. *Definitions*

To the best of our knowledge, in existing literature, there is no formal definition of blockchains in general. This is mostly due to the variety of existing blockchain technologies and implementations. Some blockchains, e.g., Ethereum [18], are formally defined, but many others are only defined by their source code, and no formal documentation or definition exists.

However, since we aim to reason about aspects of cross-blockchain interoperability, we require term definitions applying to as many blockchains as possible. We have therefore collected definitions, notation and wording from existing literature [1, 2, 5, 18], and in alignment with this literature to the greatest extent possible, provide our own definitions of certain technical terms in the following.

Note that due to the aforementioned variety of different existing blockchain technologies, it is not trivial to generalize definitions to include all implementations. In fact, our definitions do not fit certain specifics. For instance, we define each block to have one parent block. However, IOTA blocks may have multiple parent blocks [13], violating our definition. Other blockchain technologies might have other examples of aspects not covered by our definitions. However, our definitions cover the most commonly used blockchain technologies, including but not limited to Bitcoin [12], Ethereum [18] and its fork Ethereum Classic [3], as well as Litecoin [10], Dash [8] and Waves [16], and since we show general non-feasibility of cross-blockchain proofs, these definitions suffice.

Definition 1 (Blockchain)**.** A *blockchain* $\mathcal{C}$ is a distributed, decentralized, periodically growing, publicly writeable, append-only data structure consisting of *blocks*, with a defined genesis block, transaction consensus, and arbitration consensus.

Definition 2 (Blocks, Genesis Block)**.** Within a blockchain, a *block* B is a data structure linked to one parent block $P(B)$, containing arbitrary payload data. The graph of linked blocks must not contain cycles, and the *genesis block*, denoted as B_0, is the only block without a parent.

Definition 3 (Lineage)**.** The *lineage* of a block B, denoted as $\mathrm{lin}(B)$, is the line of descent of B from the genesis block B_0, i.e., $\mathrm{lin}(B) = \mathrm{lin}(P(B)) \cup B$, where $\mathrm{lin}(B_0) = B_0$.

Since blocks must not form cycles, the lineage of a block is always a finite set.

Definition 4 (Transaction Consensus)**.** The *transaction consensus* is a common understanding of the properties a block B needs to have in order to be deemed a *valid* next block of its lineage $\mathrm{lin}(B)$.

We define the transaction consensus as the function $\mathrm{tc} : \mathbb{B}^* \times \mathbb{B} \rightarrow \{0, 1\}$ (where $\mathbb{B}$ is the set of all possible blocks, and $\mathbb{B}^*$ is the set of all possible finite sets of blocks with artbitrary cardinality):

$$\mathrm{tc}(\mathrm{lin}(B), B) = \begin{cases} 1 & \text{if } B \text{ is a valid descendant of } \mathrm{lin}(B) \\ 0 & \text{otherwise} \end{cases}$$

The function tc decides whether a block B is a valid successor block of the lineage $\mathrm{lin}(B)$, i.e., it returns a (boolean) decision value. Note that this definition implies that a blockchain is self-contained, i.e., the validity of each block can be decided by only regarding its lineage (instead of also taking into account external, off-chain data). The transaction consensus defines the structure of a blockchain. For Bitcoin, it consists of the block and transaction structure, and the definition of the Script opcodes and their effects. For Ethereum, the transaction consensus consists of the definition of data structures required for blocks, the storage and memory definition and the opcodes of the Ethereum Virtual Machine (EVM).

The entirety of valid blocks forms a tree with the genesis block at the root. There can be multiple blocks which are currently not referenced by any other block as parents, i.e., *leaf blocks*. For practical purposes, we require a method of determining which leaf block to use (e.g., to reference as parent for a newly created block). We call this process arbitration and define a corresponding consensus:

Definition 5 (Arbitration Consensus)**.** The *arbitration consensus*, given a set of leaf blocks, deterministically returns one preferred leaf blocks.

The arbitration consensus is hard-coded into each node participating in the blockchain. For instance, in the original implementation of Bitcoin, the arbitration seeks the longest chain, i.e., the block with the longest lineage wins[2]. For Ethereum, the leaf of the lineage with the highest total difficulty (an attribute of Ethereum blocks) is selected. The arbitration consenus only considers valid blocks.

We now define the *main chain*:

Definition 6 (Main Chain, Orphans)**.** The *main chain* consists of the leaf node currently selected by the arbitration consensus, together with its lineage. Valid blocks not part of the main chain are called *orphans*.

Definition 7 (Data Containment)**.** Data D is *included in* blockchain $\mathcal{C}$, denoted as $D \in \mathcal{C}$, iff[3] D is part of a valid block within the main chain of $\mathcal{C}$.

Data within blocks which are not on the main chain is not regarded as the canonical state of the blockchain as a whole. For the purpose of this work, when examining whether certain data is *included in* a certain blockchain, we are technically interested whether the data is included in a block on the main chain. However, no node can be certain that the chain it currently regards as the main chain is not superseded by another chain, unknown to the node. We can therefore only realistically answer questions regarding whether data is or is not part of any (valid) block, either on the main chain, or on the lineage of an orphan.

[2]The original paper [12] defines “Nodes always consider the longest chain to be the correct one”, but implementations use block difficulty. Elaborating on this differentiation is outside of the scope of our work.

[3]“iff” is equivalent to “if and only if”

B. Cross-Blockchain Proofs

We now return to the cross-blockchain proof problem and, for the sake of reasoning, assume that the creation of a cross-blockchain proof is indeed possible. For our purposes, this means that the presence of this proof implies strictly the presence of the data to be proven:

Assumption 1. For any $D \in \mathcal{C}_B$, data $D_{\text{proof}} \in \mathcal{C}_A$ can serve as reliable proof that $D \in \mathcal{C}_B$, such that $D_{\text{proof}} \in \mathcal{C}_A \implies D \in \mathcal{C}_B$.

Without loss of generality, we assume that D_{proof} is included in B_a on $\mathcal{C}_A$, and D is included in B_b on $\mathcal{C}_B$. We denote tc_A as the transaction consensus of $\mathcal{C}_A$, and tc_B as the transaction consensus of $\mathcal{C}_B$.

$D \in \mathcal{C}_B$ holds iff $\text{tc}_B(\text{lin}(B_b), B_b) = 1$, i.e., if B_b is valid according to the transaction consensus. Since tc accepts the lineage $\text{lin}(B_b)$, in general, the outcome can depend on any data within $\text{lin}(B_b)$.

Accordingly, $D_{\text{proof}} \in \mathcal{C}_A$ holds iff $\text{tc}_A(\text{lin}(B_a), B_a) = 1$. However, due to Assumption 1, $D_{\text{proof}} \in \mathcal{C}_A \implies D \in \mathcal{C}_B$ holds, so we arrive at:

$$D_{\text{proof}} \in \mathcal{C}_A \iff \text{tc}_A(\text{lin}(B_a), B_a) = 1 \implies \text{tc}_B(\text{lin}(B_b), B_b) = 1 \iff D \in \mathcal{C}_B$$

Thus, $\text{tc}_A(\text{lin}(B_a), B_a) = 1 \implies \text{tc}_B(\text{lin}(B_b), B_b) = 1$ follows, i.e., the transaction consensus of $\mathcal{C}_A$ verifying the validity of B_a must verify the validity of B_b, which depends on its lineage of $\text{lin}(B_b)$ according to Definition 4. Note that this does not mean that tc_B necessarily requires all of the lineage data $\text{lin}(B_b)$. Nevertheless, in the general case, the entire lineage data of B_b might be required for verifying the validity of B_a.

In addition, we observe that the outcome of tc_A with regards to B_a must be equivalent to the outcome of tc_B with regards to B_b. More formally, there must exist a mapping $m : \mathbb{B}^* \times \mathbb{B} \to \mathbb{B}$, where for each $m(\text{lin}(\beta), \beta) = \alpha$, it holds that $\text{tc}_A(\text{lin}(\alpha), \alpha) = 1 \implies \text{tc}_B(\text{lin}(\beta), \beta) = 1$. In other words, for every block β (and its lineage) on $\mathcal{C}_B$ where tc_B evaluates to 1, a block α must exist (must be createable) on $\mathcal{C}_A$ where tc_A returning 1 for α is a sufficient condition of tc_B returning 1 for β. In simpler terms, tc_A must be able to *mimic* tc_B.

Summarizing the above, there are two main challenges to cross-blockchain proofs: (i) proving $D \in \mathcal{C}_B$ on $\mathcal{C}_A$ requires the inclusion of a subset of the data of $\text{lin}(B_b)$ on $\mathcal{C}_A$, and (ii) additionally, tc_A must be powerful enough to mimic tc_B.

From the above reasoning, we return to the original intuition that the presence of certain data (e.g., a specific transaction) on a given blockchain is *rooted* in this blockchain, and it cannot be verified without verifying the entire blockchain:

Lemma 1 (Lemma of Rooted Blockchains)**.** For any $D \in \mathcal{C}_B$, the existence of $D_{\text{proof}} \in \mathcal{C}_A \iff D \in \mathcal{C}_B$ implies (i) access to the lineage of the block containing D on $\mathcal{C}_A$, and (ii) a sufficiently powerful transaction consensus tc_A to mimic tc_B.

C. Implications

Lemma 1 presents two requirements for verification of the presence of data on $\mathcal{C}_B$ within $\mathcal{C}_A$: (i) the presence of a subset of the block lineage of $\mathcal{C}_B$ on $\mathcal{C}_A$, and (ii) the verifiability of the transaction consensus of $\mathcal{C}_B$ by the transaction consensus of $\mathcal{C}_A$.

Considering the first requirement, the practical implication is that blocks stored on $\mathcal{C}_B$ must, in one way or another, be stored on $\mathcal{C}_A$. While for a given instance, the subset of required lineage blocks might be small, in general, this can affect many or all blocks of $\mathcal{C}_B$. Compression algorithms can be used to minimize the amount of data required for this storage. However, in practice, data stored on blockchains is already stored in a relatively minimized form, since storage space is relatively expensive in blockchains [6, 9]. This means that both $\mathcal{C}_A$ and $\mathcal{C}_B$ can be assumed to be coded in a relatively storage-saving form (i.e., with high entropy). Furthermore, even if compression is feasible, information has a lower limit on required storage space [15].

Since storage is expensive, storing a (potentially large) subset of a blockchain's block history on another blockchain is infeasible. We therefore argue that this aspect alone makes cross-blockchain proofs impossible under practical considerations.

Furthermore, we consider the second requirement. Even if this block history is provided, according to the second requirement, the transaction consensus tc_B must be able to validate blocks on $\mathcal{C}_A$. In practice, blockchains use a transaction consensus consisting of simple operations stored in transactions (e.g., Script, the scripting system used by Bitcoin), or, in more complex cases, smart contracts (e.g., EVM) [2].

In practical terms, this implies that the instruction set used by $\mathcal{C}_B$ must be able to simulate the instruction set used by $\mathcal{C}_A$. In cases both chains use Turing-complete virtual machines, such as the EVM, this is the case. However, in other cases, such as Script, verifying more complex blockchains (such as Ethereum) imposes a limitation with regards to computability.

III. Claim-First Asset Transfers

In Section II, we show that in practice, it is not possible to verify on blockchain $\mathcal{C}_A$ whether specific data has been recorded on blockchain $\mathcal{C}_B$. This fact, which we call the cross-blockchain proof problem [4], is a major challenge to any technologies aiming at cross-blockchain collaboration. One example of such a challenge arises when considering asset transfers across blockchains, as we show in the following sections.

A. Spend-First Transactions

Regular asset transfers within one blockchain are performed by first posting proof to the blockchain that the owner of the assets is willing to transfer (spend) the assets. This is usually done using cryptographic signatures. In case of Bitcoin, a so-called unspent transaction output (UTXO) represents such a proof. Subsequently, the intended receiver of the transferred assets presents proof to the blockchain that indeed, they are the receiver. A Bitcoin UTXO, in most transactions, requires the hash value of a data vector only obtainable when the private key of the wallet to which the UTXO transfers assets is held. This *spends* the assets from the UTXO in the subsequent transaction.

Other blockchains might have slightly varying transaction models. For instance, Ethereum uses the so-called *account balance* model. Instead of UTXOs used for transactions, the account balance (in Ether) is recorded for a given address, and a transaction can transfer Ether to another address. In this case, both the intent and the reception occur in one transaction.

Nevertheless, the proof that assets are available and ready to be spent (without running the risk of double spending) is presented to the blockchain not later than the target address to which the assets are transferred.

When attempting to create a cross-blockchain asset transfer system, the intuitive approach is to first mark assets as spent on the source blockchain (in order to avoid double spending), and then claim them on the target blockchain. We denote this approach as SPEND $\rightarrow$ CLAIM, or *spend-first*. However, as mentioned in our previous work [4], the issue arising is that due to the cross-blockchain proof problem, the target blockchain is not (practically) able to verify within the CLAIM transaction that the assets have indeed been marked as spent in the source blockchain in a SPEND transaction. This verification is required to prevent double spending.

B. Claim-First Transactions

In order to mitigate this, we propose the reversal of these transactions. Instead of relying on the verification of the existence of a SPEND transaction on the source blockchain, we post the CLAIM transaction on the destination blockchain first, but define that a valid CLAIM transaction must include data that allows anyone to create a corresponding SPEND transaction. We incentivize this behavior by defining a reward for posting a SPEND transaction, which is paid to the party creating the SPEND transaction (the *witness*), similar to the transaction and mining fees in present blockchains. The witness reward is ultimately paid by either the receiver or the sender of the assets, depending on the concrete implementation of the presented concept.

We denote our approach as CLAIM $\rightarrow$ SPEND, or *claim-first*, because the CLAIM transaction is created before the SPEND transaction. While this initially creates a situation where the transferred assets exist on both blockchains (because they have been already claimed at the target blockchain, but not yet marked as spent on the source blockchain), the fact that anyone can claim a reward by marking them as spent on the source blockchain eventually leads to consistency. The only scenario where such eventual consistency would not occur is if the CLAIM transaction is not noticed by any party aware of the transfer protocol. Since a basic assumption of blockchains is willingness of participation by numerous nodes, we assume that such a scenario is not realistic. Furthermore, we assume that any party, given the possibility of a reward for a certain action, will perform this action, provided that its effect do not cause harm to the party itself (*assumption of selfishness*).

By using such claim-first transactions, we avoid the problem of having to prove the marking of assets as spent when claiming them. Instead, we rely on an eventual spending on the source blockchain by rewarding parties that do so.

C. Practical Challenges

In practice, such claim-first transactions pose their own implementation challenges. In this section, we outline these challenges and provide possible approach strategies.

Proof of Intent. The CLAIM transaction is posted on the target blockchain by the receiver, but requires proof that the sender indeed authorized the asset transfer. We propose to provide this proof by introducing the so-called *proof of intent* (PoI), a transfer authorization signed using the sender's private key. This signature is publicly verifiable. An incorrect or missing signature invalidates the corresponding CLAIM transaction.

Balances. Since in our approach, we deliberately create the CLAIM transaction first, and trust witnesses to create the corresponding SPEND transaction, one challenge consists in determining at the time of the CLAIM transaction whether a SPEND transaction will be possible at all, i.e., whether the sender of the asset transfer is in possession of enough assets on the source blockchain to initiate the transfer. Otherwise, the impossibility of a SPEND transaction due to insufficient assets would lead to inflation by unlawful creation of new assets on the target blockchain.

We propose to mitigate this by recording the asset balance of all participating parties (e.g., wallets) on all participating blockchains. This way, a CLAIM transaction can be defined to be valid only if the balance of the source blockchain (also recorded on the target blockchain) shows a sufficient asset amount.

Double Spending. Similarly, it is not possible to detect double spending attempts during the CLAIM transaction, since two CLAIM transactions can be posted to two different target blockchains. A party attempting to perform double spending might hold an amount of x assets on one blockchain, and sign two PoIs with the same amount x to be transferred, one to blockchain $\mathcal{C}_A$, and one to $\mathcal{C}_B$. The resulting CLAIM transactions will both look valid to the respective destination blockchain, and the first SPEND transaction will be created successfully (spending x assets on the source blockchain). However, when trying to create the second SPEND transaction, the balance will already be zero on the source blockchain.

We propose to avoid double spending by adding a validity time frame to the PoI. Two conflicting PoIs, i.e., two PoIs authorizing a transfer to two different blockchains from one source wallet with overlapping time frames, would then prove malicious intent by the sender (since signing two conflicting PoIs is defined as invalid). We propose to add an additional transaction type (VETO) to record the PoI conflict on a blockchain, and invalidate its previous transfers. The validity time frame additionally serves as a time-lock for the transferred assets: the CLAIM transaction releases the claimed assets only after the time period has ended, in order to enable VETO transactions.

Double Destruction. Similar to the issue of double spending, a concrete protocol must make sure that a single signed PoI is not used to mark assets as spent multiple times (destroying them). For instance, two parties witnessing the CLAIM transaction on the destination blockchain might both create SPEND transactions on the source blockchain, marking as spent twice as many assets as required. This is easily mitigated by introducing a method of distinguishing different PoI instances. The time period discussed above serves as such an identifier.

Reward Assets. The witness reward, as described above, is used to incentivize the creation of SPEND transactions by witnesses on the source blockchain. The asset type of this reward is a major design decision for cross-blockchain asset transfer protocols. We identify three possible solutions: (i) native assets, (ii) the transferred assets themselves, and (iii) dedicated reward assets.

Using native assets (e.g., Ether for Ethereum, or Bitcoins for Bitcoin) poses the challenge of how to create the reward on the source blockchain. The native assets must be paid to the witness (the creator of the SPEND transaction) within the SPEND transaction itself, and a source for these assets is required. Since in general, a smart contract cannot deduct an amount from another wallet's balance (e.g., deduct the reward from the sender's wallet), the contract has to hold a certain amount of native currency reserve by itself. This adds another level of complexity to the protocol, since there arises the necessity for a party to maintain the balance of native asset reserves across blockchains.

Using a fraction of the assets transferred themselves (similar to the transaction fee in Bitcoin) as a reward decouples the transferred assets from the native currency, enabling the independence of the cross-blockchain assets from the native assets. However, since it is necessary to maintain the complete asset balances of each wallet in each blockchain, assigning the witness a reward for creating the SPEND transaction poses an asset transfer on its own, and has to be recorded on all blockchains, again calling for a separate reward, and so on. For this infinite recursion to be avoided, it is necessary to use an algorithm for deterministically deriving the recipient of the reward without having to re-propagate this information across the blockchains with additionally required rewards.

Finally, using dedicated reward assets, which can only be used on one blockchain and not transferred to other blockchains, avoid the aforementioned challenges. However, this solution adds its own dimension of uncertainty by decoupling the value of the transferred asset from the reward asset. The economic implications of this decoupling, and how the value of the transferred asset behaves in relation to the value of the reward asset, requires additional research and observation.

When implementing the protocol, we suggest to use either the transferred asset itself, or a dedicated reward asset, since the aforementioned balancing of native assets adds complexity exceeding the complexity added by the other two possible reward asset types.

D. Summary

Summarizing Section III, we propose to implement cross-blockchain asset transfers by defining an asset claim on the target blockchain to be valid only if information is revealed which can then be used by anyone to mark the assets as spent on the source blockchain (*claim-first*).

In order to provide incentive for witnesses of such an asset claim to propagate this information to the source blockchain by marking the assets as spent, we introduce a witness reward. We identify three possible asset types which can be used as a witness reward, and identify challenges for all of these asset types, proposing to use either a fraction of the transferred assets themselves, or a dedicated reward asset.

IV. Related Work

In this section, we briefly discuss other work formalizing concepts in the field of blockchains, and the state of the art in cross-blockchain asset transfers.

Formal Blockchain Definitions. As mentioned in Section II, to the best of our knowledge, there is no work formally describing blockchains in a cross-blockchain manner, i.e., using concepts not specific to a given blockchain.

For instance, [18] provides a formal definition of the Ethereum blockchain, including all aspects of blocks, transactions (state transitions), the data storage and related aspects. From this work, we have derived most of the notation used in this paper. [2], in contrast, formally defines Bitcoin transactions. [7] presents a formal definition of Cardano wallets and UTXOs. UTXOs are also discussed in [19], along with account-based cryptocurrencies and conversions between these concepts. However, apart from the notation we have adopted, all of these works do not contain suitable definitions for our purposes, i.e., reasoning about cross-blockchain proofs.

Work providing informal definitions of blockchains includes [1] and [5], from which we have derived our Definitions 1 and 2.

Cross-Blockchain Asset Transfers. As mentioned in Section I, to the best of our knowledge, Metronome [11] is the only project aiming at cross-blockchain asset transfers. It is currently under development, and the technical details of these cross-blockchain asset transfers remain unclear. According to the project documentation [11], users can *export* assets on one blockchain, gaining a *receipt*, which can be redeemed for assets which are then *imported* on another blockchain. These receipts are validated by parties called *validators*. The validation relies on a number of validators confirming that a given receipt is legitimate, but the precise mechanism of these validations (e.g., how validators are authenticated) is currently not specified.

V. Conclusion

In this paper, we have presented two core contributions. First, we have formalized the cross-blockchain proof problem, by deriving the lemma of rooted blockchains, showing that using practical means, it is not possible to verify the existence of specific data on one blockchain from another blockchain.

Second, we have proposed a conceptual protocol for transferring a given asset type across blockchains. We have described the concept, showing how reversing the traditional order of asset transactions can potentially overcome the cross-blockchain proof problem. Our protocol requires only the participation of selfish witnesses, which is incentivized using a reward. In addition, we have identified the main challenges for the implementation of the protocol, and outlined our proposed mitigation strategies for each one of these challenges.

In future work, we will propose concrete protocols in technical detail. We plan to propose both a protocol using dedicated reward assets, as well as a protocol using the transferred asset type itself as a reward.

Disclaimer

Information provided in this paper is the result of research, based on publicly available resources of varying quality. Popular use of cryptocurrencies includes investment and speculation on price developments of currencies and assets. However, the goal of this paper is to describe technical aspects relevant for TAST. Economic considerations or future price developments are therefore not discussed. Technologies are described from a purely technical point of view. Therefore, the contents of this paper do not constitute advice, information, predictions, or recommendations for investment.

Acknowledgment

The work presented in this paper has received funding from Pantos GmbH within the TAST research project.

References

[1] M. Ali, J. C. Nelson, R. Shea, and M. J. Freedman. “Blockstack: A Global Naming and Storage System Secured by Blockchains.” In: *USENIX Annual Technical Conference*. 2016, pp. 181–194.

[2] N. Atzei, M. Bartoletti, S. Lande, and R. Zunino. “A formal model of Bitcoin transactions”. In: *Financial Cryptography and Data Security. Springer* (2018).

[3] M. Beck. *Into the Ether with Ethereum Classic*. 2017. URL: https://ethereumclassic.github.io/assets/etc-thesis.pdf. White Paper. Accessed 2018-04-13.

[4] M. Borkowski, D. McDonald, C. Ritzer, and S. Schulte. *Towards Atomic Cross-Chain Token Transfers: State of the Art and Open Questions within TAST*. 2018. URL: http://dsg.tuwien.ac.at/staff/mborkowski/pub/tast/tast-white-paper-1.pdf. White Paper, Technische Universität Wien. Version 1.2. Accessed 2018-08-09.

[5] K. Christidis and M. Devetsikiotis. “Blockchains and Smart Contracts for the Internet of Things”. In: *IEEE Access* 4 (2016), pp. 2292–2303. ISSN: 2169-3536.

[6] M. Conoscenti, A. Vetro, and J. C. De Martin. “Blockchain for the Internet of Things: A systematic literature review”. In: *IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*. IEEE. 2016, pp. 1–6.

[7] D. Coutts and E. de Vries. *Formal specification for a Cardano wallet*. URL: https://cardanodocs.com/files/formal-specification-of-the-cardano-wallet.pdf. Technical Documentation. Version 2018-07-16. Accessed 2018-08-22.

[8] E. Duffield and D. Diaz. *Dash: A Privacy-Centric Crypto-Currency*. 2015. URL: https://github.com/dashpay/dash/wiki/Whitepaper. Accessed 2018-04-13.

[9] T. Hukkinen. *Reducing blockchain transaction costs in a distributed energy market application*. 2018. Master thesis, Aalto University, Helsinki, Finland.

[10] *Litecoin*. URL: https://litecoin.org/. Website. Accessed 2018-04-13.

[11] *Metronome: Owner's Manual*. URL: https://www.metronome.io/pdf/owners_manual.pdf. White Paper. Version 0.967, 2018-04-17. Accessed 2018-04-19.

[12] S. Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. 2008. White Paper.

[13] S. Popov. *The Tangle*. 2017. URL: https://iota.org/IOTA_Whitepaper.pdf. White Paper. Version 1.3. Accessed 2018-04-13.

[14] C. Prybila, S. Schulte, C. Hochreiner, and I. Weber. “Runtime verification for business processes utilizing the Bitcoin blockchain”. In: *Future Generation Computer Systems* (2017).

[15] C. E. Shannon. “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[16] WAVES Platform. *WAVES Whitepaper*. 2016. URL: https://wesdewayne.files.wordpress.com/2017/05/waves-whitepaper.pdf. Accessed 2018-04-13.

[17] J. Willett, M. Hidskes, D. Johnston, R. Gross, and M. Schneider. *Omni Protocol Specification*. 2017. URL: https://github.com/OmniLayer/spec. Version 0.5. Accessed 2018-04-13.

[18] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. 2018. URL: https://ethereum.github.io/yellowpaper/paper.pdf. White Paper. Accessed 2018-04-13.

[19] J. Zahnentferner. *Chimeric Ledgers: Translating and Unifying UTXO-based and Account-based Cryptocurrencies*. Cryptology ePrint Archive 2018/262. 2018.

[20] A. Zohar. “Bitcoin: under the hood”. In: *Communications of the ACM* 58.9 (2015), pp. 104–113.