Leveraging Blockchain Relays for Cross-Chain Token Transfers

Philipp Frauenthaler*, Marten Sigwart*, Christof Spanring†, Stefan Schulte*

* Distributed Systems Group
TU Wien, Vienna, Austria
{m.sigwart, p.frauenthaler,
s.schulte}@dsg.tuwien.ac.at

† Pantos GmbH
Vienna, Austria
contact@pantos.io

***Abstract*—Blockchain relay schemes offer the ability to verify transactions across blockchains in a decentralized manner. While this enables blockchain interoperability applications like cross-blockchain token transfers, relays can become expensive since state-of-the-art relays require every single block header of the source blockchain to be stored by the destination blockchain.**

In this paper, we further reduce operating cost of our existing blockchain relay solution by applying the content-addressable storage pattern. Furthermore, we outline how relays can be leveraged to enable blockchain interoperability applications such as cross-blockchain token transfers. The devised protocol shows that most requirements for cross-blockchain token transfers can be fulfilled.

I. Introduction

The Token Atomic Swap Technology (TAST) research project[1] aims to create a platform for cross-blockchain interoperability. The overarching goal is to investigate possible means of interconnecting various blockchains [1]. As an intermediate step towards achieving this goal, we aim to create a cross-blockchain token, i.e., a token that can be freely exchanged between various blockchains [4].

So far, tokens that can be transferred to other blockchains either rely on centralized entities coordinating the exchange [6] or a user needs to find another party willing to swap tokens, e.g., via atomic swaps [5]. However, within TAST we aim to provide a token that can be transferred between blockchains in a decentralized manner without having to swap tokens with another party.

In prior work [4], we defined the requirements for such a token. Essentially, a transfer occurs by burning a certain amount of tokens on the source blockchain and then recreating the same amount of tokens on the destination blockchain. Of course, the tokens should only be recreated on the destination blockchain if the burning of the tokens has actually occurred on the source blockchain. Hence, the destination blockchain needs a way to verify the existence of the transaction burning tokens on the source blockchain.

One possibility to verify transaction inclusions across blockchains are so-called blockchain relays [2]. Essentially, relays replicate the state of a source blockchain within a destination blockchain and as such enable the destination blockchain to verify the existence of certain pieces of state on the source blockchain. The replication of the source blockchain happens in a completely decentralized way and consequently does not require trust in a centralized entity [2]. Using a blockchain relay, it becomes possible to verify on the destination blockchain that a transaction burning some tokens has occurred on the source blockchain [7].

In our prior work [8] we have developed a prototypical blockchain relay for Ethereum-based blockchains. The cost analysis conducted in [7] shows that the prototype already reduces the operational cost in comparison to traditional relays. In the work at hand, we show how the operational cost can be reduced even further by applying a content-addressable storage pattern. Also, we showcase how the prototype can be leveraged to implement true cross-blockchain token transfers as envisioned by TAST.

II. Recap: White Paper VII

In our prior work [8], we described the underlying concepts of our proposed blockchain relay and implemented these concepts in a first proof-of-concept prototype for Ethereum-based blockchains.[2]

To recall, a blockchain relay is operated by off-chain clients who continuously submit block headers from a source blockchain to a destination blockchain (see Fig. 1). With the source blockchain essentially being replicated within the destination blockchain, it becomes possible from within the destination blockchain to make queries such as: What is the branch of the source blockchain with the highest total difficulty (i.e., the main chain), was transaction x included in block b of the source blockchain, and so on.

Since a relay relies on continuous participation of off-chain clients for keeping the relay operating, we further introduced an incentive structure for encouraging participation of off-chain clients and analyzed the operational cost of the relay in TAST White Paper VII [7].

The devised prototype already enables cross-blockchain transaction verifications. Before we look into a first kind of application of the relay in Section IV, in the next section, we

White Paper, TU Wien; February 2020, version 1.0.

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

[2] https://github.com/pantos-io/go-testimonium

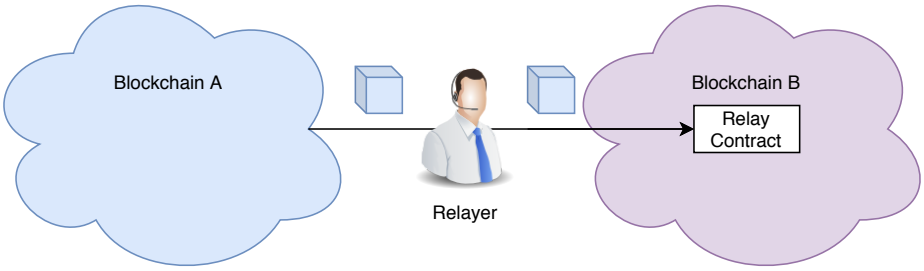

Figure 1: Basic Functionality of a Blockchain Relay

describe further improvements of the prototype that make its operation even more cost-efficient.

III. Cost Improvements

As described in Section II, the developed blockchain relay relies on off-chain clients continuously submitting block headers of the source blockchain to the destination blockchain. When submitting new block headers, clients incur cost. As shown in [7], the developed relay already achieves a cost reduction of up to 82% over state-of-the-art relay designs. In this section, we describe an optimization that reduces the cost of submitting block headers even further.

As storing data is one of the most expensive tasks in Ethereum-based blockchains [9], reducing the amount of data being stored within each submitted block header is a promising approach to lower the cost. Of course, with a reduced amount of data that is stored for each block header, information may be missing that is needed later on, e.g., in case the header is disputed or transaction inclusion verifications are carried out. Hence, we need a way to access this information when verifying transaction inclusions and disputing headers.

For that, we use a slightly modified version of the content-addressable storage pattern [3]. Essentially, when receiving a new header, we store only as much information as required for keeping track of the different blockchain branches and the main chain. The remaining block header information is no longer stored within the relay contract. However, since we need access to the complete block header data during transaction inclusion verifications and header disputes, the remaining information needs to be kept somewhere else.

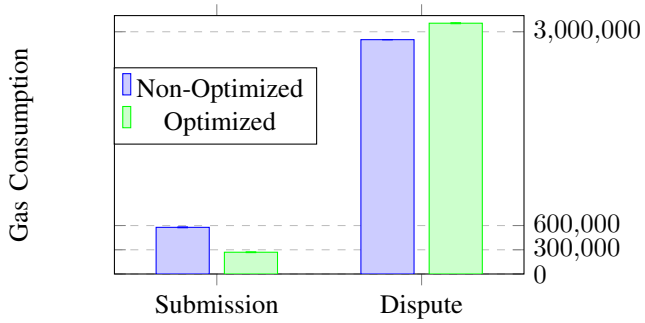

Figure 2: Avg. Gas Consumption for Submitting and Disputing Block Headers

As the relay is implemented as smart contract, each invocation (e.g., submitting a block header) is implicitly logged in the blockchain's transaction history. This log contains also the parameters (i.e., the submitted block headers) used for the invocation of the smart contract. Hence, each submitted block header gets implicitly logged in the transaction history of the blockchain. However, the smart contract itself has no access to this log. Thus, when clients trigger a transaction inclusion verification or a dispute on a certain block header, they have to provide the full header from the blockchain's transaction history to the smart contract.

Since all information stored in a public blockchain is publicly available, clients can extract the logged block headers from the transaction history. Using the provided information, the relay contract can execute transaction inclusion verifications and header disputes exactly in the same way as if the information was directly stored in the relay contract. The contract only needs to make sure that the provided information belongs indeed to the header on which transaction inclusion verifications or disputes should be carried out.

This can be achieved by storing the cryptographic hash of each submitted block header within the relay contract at submission time. Later on, when verifying transaction inclusion verifications or performing disputes on a certain block header, the contract uses the stored hash to verify that the provided information belongs to the block header previously submitted. The relay contract simply calculates the hash of the provided information and compares it to the hash stored at submission time. If they match, the information belongs indeed to the submitted header.

With this approach, the amount of data being stored within the relay contract is significantly reduced. To quantify the achieved cost reduction, we repeated the evaluation presented in [7] using the optimized implementation of the relay. Fig. 2 shows the average gas consumption of both the optimized and the non-optimized version of the developed relay. Notably, Ethereum-based blockchains use gas as measuring unit for quantifying the cost of executing instructions such as adding two numbers or storing a value. The overall gas consumption of a header submission or a header dispute is determined by all instructions performed when invoking the corresponding smart

Table 1: Operational Cost of the Non-Optimized Relay

Gas Price (GWei)	Cost/Submission (EUR)	Cost/Hour (EUR)	Cost/Day (EUR)	Cost/Year (EUR)
1	0.12	28.40	681.50	248,746.77
3	0.35	85.19	2,044.49	746,240.31
10	1.18	283.96	6,814.98	2,487,467.69

Table 2: Operational Cost of the Optimized Relay

Gas Price (GWei)	Cost/Submission (EUR)	Cost/Hour (EUR)	Cost/Day (EUR)	Cost/Year (EUR)
1	0.06	13.25	317.94	116,049.55
3	0.17	39.74	953.83	348,148.64
10	0.55	132.48	3,179.44	1,160,495.47

contract. The higher the gas consumption, the more expensive the invocation becomes. As shown in Fig. 2, the optimized implementation achieves a reduction of submission cost of up to 53% compared to the non-optimized implementation. However, disputing a block header is slightly more expensive with the optimized version since the full block header information has to be provided for every dispute. For the non-optimized case, this required informational input is reduced to the header's hash. We expect the significant reduction of the gas consumption for header submission to outweigh the slight increase of the gas consumption for header disputes. Successful disputes only arise in the case where a submitted header is incorrect. Thus, dispute cost can not be directly attributed to operational cost as they are merely reflected by the amount of required stake for block header submissions as outlined in [7].

For converting the gas consumptions into EUR, we consider an average block submission rate of 4 blocks per minute and assume both relay versions to run on the Ethereum main chain with an exchange rate of 204.79 EUR/ETH at the time of writing[3]. Notably, users can specify a gas price when submitting transactions (typically in GWei). The higher the gas price of a transaction, the faster it will be included in a block by some miner. On the downside, a higher gas price leads to higher transaction cost. For the calculation, we consider three gas prices, namely 1 Gwei, 3 Gwei, and 10 Gwei. Table 1 and Table 2 display the approximate operational cost of header submission for both relay versions.

In the next section, we examine how blockchain relays can be leveraged to implement blockchain interoperability solutions such as cross-blockchain token transfers.

IV. Cross-Blockchain Token Transfers

Within TAST, we aim to create a cross-blockchain token. As mentioned in Section I, such a token can be transferred between different blockchains freely and in a decentralized manner. Ideally, users can hold different denominations of the token on different blockchains at the same time.

In [4], we defined the following requirements that must be fulfilled for such a token:

[3]February 10, 2020

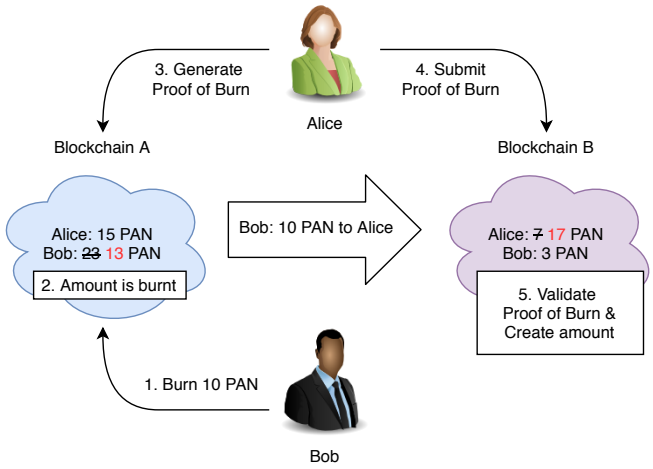

Figure 3: Executing a Cross-blockchain Token Transfer

(1) When transferring some amount of tokens from the source blockchain to the destination blockchain, the amount should only be created on the destination blockchain, if it can be proven that the same amount has already been burned on the source blockchain.
(2) It should not be possible to fake the burning of tokens.
(3) Every token that was burned on one blockchain can only be (re-)created once on another blockchain.
(4) It should not be possible to burn tokens on one chain without recreating them on another.

With these requirements in mind, we can define the execution steps of a typical cross-blockchain token transfer. As example, we assume that Bob wants to transfer 10 PAN tokens to Alice from blockchain A to blockchain B (see Fig. 3). Bob first submits a "burn" transaction tx_{burn} to blockchain A. Blockchain A executes the transaction, effectively reducing Bob's PAN account on blockchain A by 10 units. Alice notices that the tokens have been burned successfully. Using tx_{burn}, she generates a proof of burn. She then submits this proof of burn to blockchain B. Blockchain B verifies the proof and—if successful—creates 10 PAN assigning them to Alice's account.

To create and verify the proof of burn, we can leverage the functionality that is offered by blockchain relays. As explained in Section II, blockchain relays enable a blockchain to verify

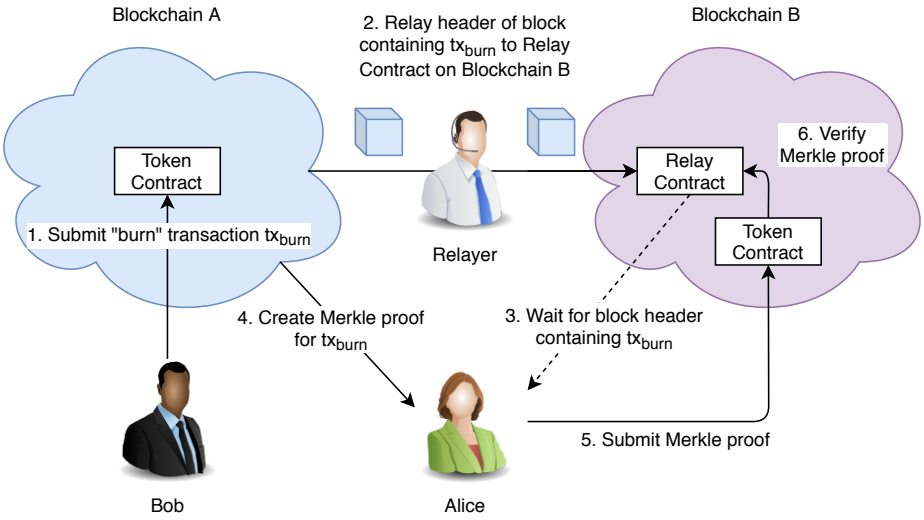

Figure 4: Cross-blockchain Token Transfer with Blockchain Relay

whether some transaction is included in a different blockchain. Hence, we can use a relay to verify the existence of tx_{burn} on the source blockchain from the destination blockchain. The interaction that takes place between the different participants is depicted in Fig. 4.

At first, Bob submits tx_{burn} to the source blockchain A. Eventually, tx_{burn} is included in a new block and appended to blockchain A. When this happens, off-chain clients participating in the blockchain relay are notified of the new block and relay the corresponding block header to the relay contract deployed on blockchain B.

Since Alice is the intended recipient of the cross-blockchain transfer, she waits until the relay contract has received the header of the block containing tx_{burn}. She then creates a Merkle proof of membership for tx_{burn}. She submits the Merkle proof to the token contract deployed on blockchain B. Since the relay contract on blockchain B has already received the corresponding block header, the token contract can request the verification of the Merkle proof from the relay contract. If the verification of the Merkle proof is successful, the token contract can be sure that tx_{burn} has been successfully included in blockchain A. The contract subsequently creates the amount of tokens that were burned and assigns them to Alice.

Next, we examine how this scheme holds up against the defined requirements. Requirement 1 essentially says that no claim should take place without a corresponding burn. When Alice submits a claim to blockchain B she has to submit a Merkle proof contesting that some transaction x is included in blockchain A. If Alice is honest, she submits a Merkle proof for transaction tx_{burn}. Of course, Alice could also try to create a Merkle proof for any transaction which is not tx_{burn} and try to submit the proof to blockchain B. However, a Merkle proof for some transaction x also contains x itself, thus it becomes trivial to check on blockchain B that x is actually transaction tx_{burn}. Since the Merkle proof contains tx_{burn} itself, it is also trivial to check how many tokens were actually burnt by tx_{burn}. Of course, the token contract on blockchain B will only recreate exactly the same amount of tokens. Hence, **??** can be considered fulfilled as a claim cannot be successful without a Merkle proof for some tx_{burn} burning exactly the same amount of tokens that is being claimed.

Of course, Alice could try to fake tx_{burn}. For instance, she could set up her own token contract on blockchain A that mimics the behaviour of the actual token contract and submit a fake burn transaction tx'_{burn} there. She then submits a Merkle proof containing tx'_{burn} to the token contract on blockchain B. Just from the transaction parameters, blockchain B has no way of knowing that tx'_{burn} is invalid. However, the transaction data also contains information to which smart contract the transaction has been submitted. Hence, the token contract on blockchain B needs to check that the transaction was submitted to the right token contract on blockchain A.

Alice could also try to create a fake Merkle proof with a seemingly valid transaction tx_{burn} that has not been included in blockchain A yet. In this case, verifying the Merkle proof on blockchain B fails, and subsequently also the claim. Therefore, it is not possible to fake the burning of tokens (requirement 2).

Another possibility is for Alice to submit multiple claims to blockchain B every time using the same transaction tx_{burn}. If tx_{burn} has been submitted to the right token contract on blockchain A, it would be marked as valid "burn" transaction by the token contract on blockchain B. To avoid double spends, i.e., multiple claims using the same burn, the token contract on blockchain B needs to keep track of all "burn" transactions that have already been used to claim tokens. Double spends in scenarios involving more than two blockchains can be easily avoided as well by encoding the recipient blockchain within tx_{burn} so that only the recipient blockchain accepts the corresponding claim. This way, tx_{burn} cannot be used to claim tokens multiple times by submitting the claim to different blockchains. We therefore consider requirement 3 as fulfilled as well.

Requirement 4 states that tokens should not be burned if no corresponding (successful) claim takes place. This way, the total amount of tokens within the system would remain constant at all times since cross-blockchain transfers would have to be executed in an atomic fashion. In fact, this is not covered by the cross-blockchain token transfers presented above yet, and therefore remains part of our future work.

V. Conclusion

Blockchain relays like the one introduced in the last TAST White Papers offer the ability to verify transactions across blockchains. In this White Paper, we improved the relay by applying a content-addressable storage pattern to further reduce operating cost. We then demonstrated how blockchain interoperability applications like the cross-blockchain token as envisioned by TAST can be built on top of it. While the interaction outlined in this paper fulfills most of the requirements for decentralized token transfers across blockchains, in future work, we will extend the protocol to ensure that tokens can only be burned on the source blockchain if they are also recreated on the destination blockchain, keeping the total amount of tokens in the system constant.

The proof of concept implementation of the relay is currently restricted to Ethereum-based blockchains. In future work, the approach will be extended to other blockchain platforms as well. Finally, we will explore blockchain interoperability solutions beyond cross-blockchain token transfers.

Disclaimer

Information provided in this paper is the result of research, partly based on publicly available resources of varying quality. Popular use of cryptocurrencies includes investment and speculation on price developments of currencies and assets. The goal of this paper is to describe technical aspects relevant for the TAST research project. Economic considerations or future price developments are therefore not discussed. Technologies are described from a purely technical point of view. Therefore, the information in this paper is provided for general information purposes only and is not intended to provide advice, information, predictions, or recommendations for any investment. We do not accept any responsibility and expressly disclaim liability with respect to reliance on information or opinions published in this paper and from actions taken or not taken on the basis of its contents.

Acknowledgment

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

References

[1] M. Borkowski et al. *Cross Blockchain Technologies: Review, State of the Art, and Outlook*. 2019. URL: http://dsg.tuwien.ac.at/projects/tast/pub/tast-white-paper-4.pdf. White Paper, Technische Universität Wien. Version 1.0. Accessed 2020-02-06.

[2] V. Buterin. *Chain Interoperability*. https://allquantor.at/blockchainbib/pdf/vitalik2016chain.pdf. Accessed 2020-02-10.

[3] J. Eberhardt et al. "On or off the blockchain? Insights on off-chaining computation and data". In: *European Conference on Service-Oriented and Cloud Computing*. Springer. 2017, pp. 3–15.

[4] P. Frauenthaler et al. *Towards Efficient Cross-Blockchain Token Transfers*. 2019. URL: http://dsg.tuwien.ac.at/projects/tast/pub/tast-white-paper-5.pdf. White Paper, Technische Universität Wien. Version 1.0. Accessed 2020-02-06.

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

[6] Loom Network. *Transfer Gateway*. https://loomx.io/developers/docs/en/transfer-gateway.html. Accessed 2020-02-10.

[7] M. Sigwart et al. *Preparing Simplified Payment Verifications for Cross-Blockchain Token Transfers*. 2019. URL: http://dsg.tuwien.ac.at/projects/tast/pub/tast-white-paper-7.pdf. White Paper, Technische Universität Wien. Version 1.0. Accessed 2020-02-06.

[8] M. Sigwart et al. *Towards Cross-Blockchain Transaction Verifications*. 2019. URL: http://dsg.tuwien.ac.at/projects/tast/pub/tast-white-paper-6.pdf. White Paper, Technische Universität Wien. Version 1.0. Accessed 2020-02-06.

[9] G. Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. 2018. URL: https://ethereum.github.io/yellowpaper/paper.pdf. White Paper. Version 69351d5, 2018-12-10. Accessed 2019-02-14.