Proof-of-Property – A Lightweight and Scalable Blockchain Protocol

Christopher Ehmke*
University of Duisburg-Essen
Essen, Germany
christopher.ehmke@uni-due.de

Florian Wessling
University of Duisburg-Essen
Essen, Germany
florian.wessling@uni-due.de

Christoph M. Friedrich
University of Applied Sciences and Arts Dortmund
Dortmund, Germany
christoph.friedrich@fh-dortmund.de

ABSTRACT

The expansion of blockchain technologies from financial applications to other fields intensifies the problem of an increasing size of data stored in the blockchain. Unfortunately, new participants of the blockchain network are required to download the whole blockchain to gain an overview about the state of the system and to validate incoming transactions. Approaches like IOTA, SegWit or the Lightning Network try to solve the scalability issues of blockchain applications. Unfortunately, they focus on strategies slowing down the blockchain's growth instead of reducing the problems arising from a growing chain or introduce new concepts to oust the linear blockchain altogether. The approach proposed in this paper is based on the idea of Ethereum to keep the state of the system explicitly in the current block but further pursues this by including the relevant part of the current system state in new transactions as well. This enables other participants to validate incoming transactions without having to download the whole blockchain initially. Following this idea use cases can be supported that require scalable blockchain technology but not necessarily an indefinite and complete transaction history.

CCS CONCEPTS

• **Computer systems organization → Peer-to-peer architectures**; • **Security and privacy → Security protocols**; **Distributed systems security**;

KEYWORDS

Blockchain, scalability, Patricia Tree, Trie, Merkle Tree, Ethereum, Bitcoin

ACM Reference Format:
Christopher Ehmke, Florian Wessling, and Christoph M. Friedrich. 2018. Proof-of-Property – A Lightweight and Scalable Blockchain Protocol. In *WETSEB'18: WETSEB'18:IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB 2018), May 27, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3194113.3194122

*Also with University of Applied Sciences and Arts Dortmund.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
WETSEB'18, May 27, 2018, Gothenburg, Sweden
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5726-5/18/05…$15.00
https://doi.org/10.1145/3194113.3194122

1 INTRODUCTION

In the last years a lot of blockchain applications arose [6]. One problem of this technology is the size of the blockchain [1]. To validate incoming transactions, participants are required to check whether the input addresses own enough coins to fulfill the stated transaction (resp. enough Ether to pay the stated amount of gas). This is done by checking the state of the blockchain system the participants have agreed onto. In implementations following the original blockchain concept proposed by Nakamoto [8] this means that the participants initially have to retrieve a complete copy of the blockchain to build their local copy of the global state of the system. This requirement comes from the fact that the state of the system is not persisted in the blockchain but will be reconstructed by reapplying all transactions processed by the network [4, 11]. This process has the drawback that the participation of new members will be delayed because they have to download the whole blockchain at the beginning (e.g. the Bitcoin blockchain consists of over 150 Gigabytes of data at the moment[1]) [14]. Furthermore, possible members who do not have enough free space to download the blockchain initially cannot participate in the creation of new blocks at all because they do not know the state the system has agreed onto and therefore are not able to validate incoming transactions.

In this paper, an approach is presented solving this problem for certain use cases that require scalable blockchain technology but not necessarily an indefinite and complete transaction history. This paper is structured as follows: in section 2 the proposed approach will be described. Section 3 will point out topics that have to be considered when implementing the proposed approach and section 4 will conclude the paper.

1.1 Related work

There are various approaches on how to handle the scalability issues of blockchain networks. An interesting approach is IOTA [10] that tries to enable blockchain technology in the context of Internet-of-things and focuses on an entirely different concept than classical blockchain applications. For example, it does not use a linear blockchain anymore but a mesh of connected blocks. Furthermore, there are approaches like SegWit [5] that, in contrast to this paper, focus on another aspect of the scalability problem namely the size of the different blocks in the blockchain. Approaches like the Lightning Network [9] or Raiden[2] try to perform transactions off-chain. In doing so, they reduce the number of transactions processed by the whole network. Unfortunately, these concepts still

[1] https://blockchain.info/de/charts/blocks-size Accessed 16 January 2018
[2] https://raiden.network/ Accessed 06 February 2018

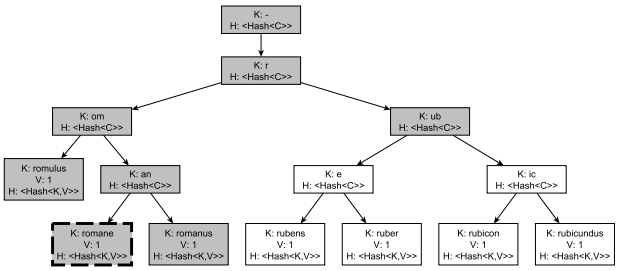


Figure 1: Example Merkle Patricia Tree. For the sake of readability, normal words (like "romane") were used as keys instead of the hashes of the public keys of the addresses. Highlighted (gray background) is the validation path of the key "romane" (bold dashed). All nodes with a white background are not needed for the validation path and could be erased.

need at least two transactions (one to initialize the off-chain channel and one to finally terminate it). Therefore, they slow down the growing of the blockchain by reducing the number of transactions needed to be included. The backbones of this off-chain transactions are still classical blockchains which rely on full nodes (which need to download the whole blockchain initially) to validate and maintain the global state of the network. In contrast to the approach presented in this paper these projects might slow down the growing of the blockchain but do not present a general concept to handle the problems coming up with the ever-growing blockchain.

2 PROOF-OF-PROPERTY

The base idea of the proposed procedure is to include a proof-of-property (respectively the proof that the input addresses of a certain transaction own enough coins to fulfill the stated transaction) in each transaction. To be usable in an untrusted environment like blockchain applications, this proof-of-property has to represent the state the blockchain network has agreed onto without introducing any possibilities of fraud. The proposed procedure is based on the concept of Ethereum to include the current state of the blockchain system in new blocks ([3, 13]). Besides other changes, the architecture of the Ethereum network differs from the Bitcoin architecture in the way the global state of the system is stored. As already stated, following the Bitcoin architecture the state of the network (respectively the credit of the different addresses) is only available indirectly through the processed transactions. In contrast to this, the Ethereum architecture stipulates that the state of the system is explicitly given in each block. To do so efficiently a Merkle Patricia Tree is used where unchanged nodes simply point to nodes available in earlier blocks ([2, 3]). One advantage of Patricia trees over simple Merkle trees is that the insertion order of new nodes does not affect the resulting structure of the tree [7], thus resulting in the same root hash. This enables users to refer to nodes in a previous tree instead of explicitly including them without affecting the calculated root hash value. Therefore, this property of the data structure allows the optimizations on which Ethereum relies on and which would not be possible with normal Merkle trees.

Depending on the blockchain under examination, the state of the system represents different types of information. In the simplest case, e.g. blockchains like Bitcoin, the state directly represents the properties owned by the various addresses of the network. In other blockchains (like Ethereum) the state of the network consists of more information but the balances are part of it most of the times as well.

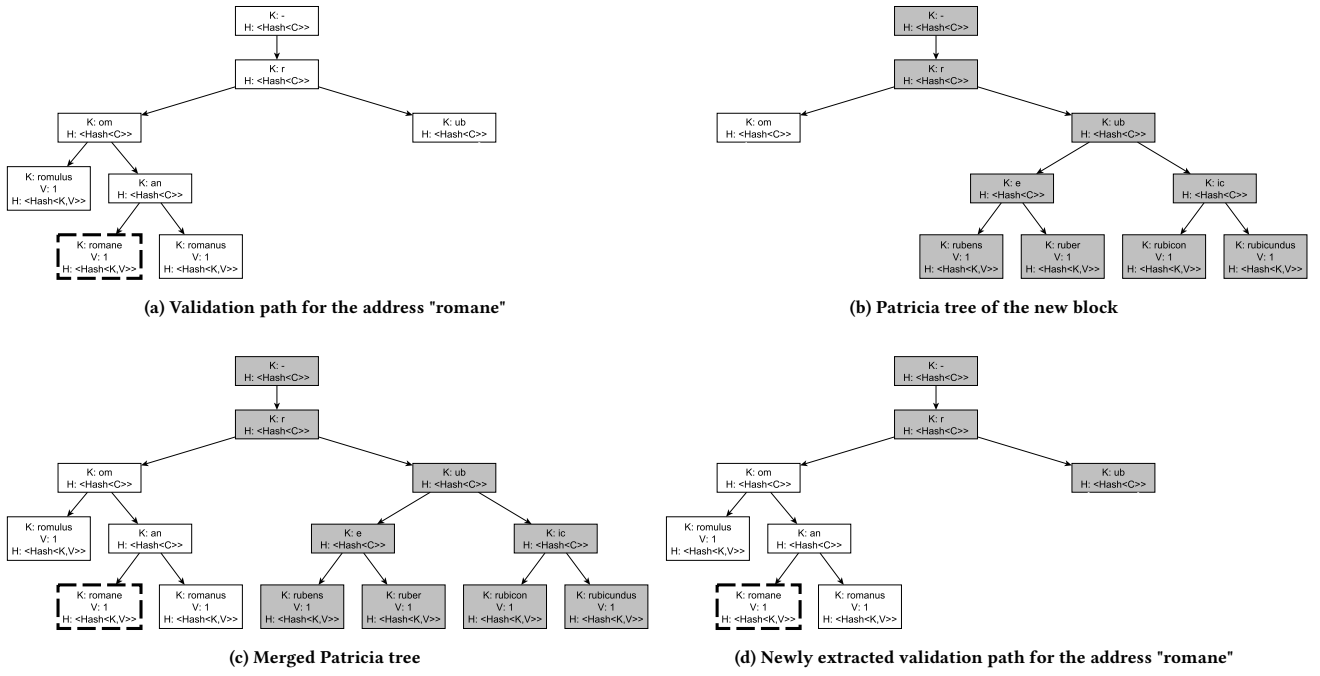
Using a Merkle Tree to model the state of the system leads to one important possibility: Users are able to extract a proof that the content of a certain node in this tree was not manipulated without having to present all the data in the tree. Instead of that only $O(Log_2(N))$ (where N is the number of leafs in the Tree) nodes have to be presented [12]. To be more precise, it is sufficient to have the own value and the hashes of all of the siblings and all parent nodes and their siblings up to the root node to prove the correctness of a given value. Figure 1 visualizes which nodes are required to prove the correctness of a certain value. The required nodes will be called "validation path" in the following.

The approach of this paper proposes to enrich each transaction with the validation paths of each of its input addresses. If this is the case, it could be checked whether the addresses are able to fulfill the transaction (resp. if they own enough coins) without having to have access to the whole blockchain. If relying only on the information given alongside the transaction, it has to make sure that this data really represents the properties the network has agreed onto and that the data was not manipulated. This can be easily done be recalculating the root hash of the received Merkle Patricia Tree and comparing it with the value given in the newest block. If both hashes are the same the verifier can be sure that received data really represent the state the network has agreed onto and can use this information for the check whether enough coins are available. Sending validation paths instead of the whole network state alongside the transaction assures that only data that is needed for the validation is added. The impact of this additional data is therefore as small as possible.

2.1 Creating and maintaining the validation paths

One problem of the idea to include the state information in the form of validation paths in the transactions comes from the question how this data is created. This paper follows the idea of the Ethereum block structure where the current network state is present in the newest block. Therefore, it is possible to extract the validation path needed for a new transaction from this data in some cases. To be more precise when the address serving as an input address is present in the Merkle Patricia Tree of the newest block, the validation path can be extracted directly. The Ethereum concept optimizes the size of the block data not by including the whole Tree in each block but by referring to nodes in previous blocks if these nodes were not modified. This convention leads to the problem that the data needed for a new transaction might not be available in the newest block but needs to match its root hash.

For this case, an update mechanism is proposed. When a new address gets some coins assigned, it will be present in the state Tree of the newest block. At this point in time the owner of that address is able and required to extract a correct validation path for



(a) Validation path for the address "romane"

(b) Patricia tree of the new block

(c) Merged Patricia tree

(d) Newly extracted validation path for the address "romane"

Figure 2: Update process of an example validation path for the node "romane" (bold dashed). Nodes that have been changed through transactions in the new block are highlighted with a gray background.

his address. With each new block he is able to update his validation path so that it stays synchronized with the state data of the new block. If the owner updates his validation path each time a new block is broadcasted he is able to present a correct validation path for his address at every point in time.

The update process consists of the following steps (Figure 2 visualizes this procedure):

(1) Take both the validation path to be updated (Figure 2a) and the network state Patricia Tree of the newest block (Figure 2b).
(2) Merge the information from the validation data into the Patricia Tree (Figure 2c). It is important not to override any new data hereby but only to insert nodes and leafs that are needed to create a new validation path.
(3) Create and extract a new validation path from the newly created merged Patricia tree (Figure 2d).

This update procedure has to be performed for each new block. If the process was performed successfully, the root hash of the resulting validation path matches the root hash of the network state Patricia Tree present in the newest block again. It can therefore be used again to prove that the address owns a certain amount of money on which the network has agreed onto when creating a new transaction.

2.2 Benefits of the proposed procedure

When following the proposed process, the creators of new transactions are able to prove that their address owns enough coins to fulfill a certain transaction without forcing the network participants to have a local copy of the state of the system. Thus new participants can validate incoming transactions without downloading the complete blockchain initially. Instead of that they only have to have access to the root hash of the system state Patricia Tree of the newest block. This root hash could be used to verify that the data that have been send alongside the transaction were not manipulated.

Even though only the root hash of the newest block is needed to verify transactions, the blockchain is still needed. Only the chain of blockheader, which contains the network state Patricia Tree root hash, guarantees a consistent view on the network state for all members. In contrast to traditional blockchain applications, the block body of older blocks are not needed anymore, because they are not required to recreate the local system state. Because of this, members of the network could delete the block bodies of all old blocks. Since over 90% of the size of a block is made up of the block body a lot of local space could be saved when this data could be erased.

3 FURTHER RESEARCH

There are a few points that should be taken into consideration when thinking about further research in the context of the proposed approach. In the following, four important points will be described.

- **Tree Type**: There should be some considerations regarding the type of Patricia Trees used. It is often proposed to use alphanumerical keys for the Patricia Tree [7]. Morrison

expected his tree to be used for exactly that use case [7]. Nevertheless, that would mean that each node has a potentially high number of siblings (in the worst case each element out of the mass of alphanumerical signs is a sibling). Thus, a very high number of nodes is needed to extract a validation path for one value. Binary keys could be a potential solution for this problem. A conversion of the alphanumerical keys to binary ones before each tree operation would not mean a huge performance issue. A binary key Tree would mean that each node could only have one sibling. At the other hand it would mean a deeper Tree structure.

- **Forks**: One of the problems of decentralized systems like Bitcoin is that different clients could be in a different state. This case is called a "fork" in the blockchain. One disadvantage of the possibility of forks is that transactions already being processed might become invalid if the block they're included in is in the part of the forked blockchain that will be dismissed. There should be some considerations if the possibility of forks might lead to security threats in the context of the proposed changes.
- **Use cases**: Following the presented approach, the bodies of all blocks but the newest are not needed anymore because incoming transactions could be validated without the data contained in them. This might lead to the idea to delete these old blocks. In order to keep their validation paths synchronized with the root hash of the newest block the users have to update their validations in the same order new blocks were created. Unfortunately it might be possible that they can not get the data of an older block after some time anymore because their communication partners decided to delete this data since they don't need it anymore. To avoid this, users have to stay online all the time to be able to update their validation paths correctly. This scenario might be impossible for normal users of a currency. It has to be discussed whether this constraint hinders the usage of the presented approach in a blockchain network primary focusing on a financial usage. Nevertheless, there might be other uses cases where this constraint is insignificant. There should be some considerations if the requirement to stay online and update validation paths requires the developer to focus on a different application architecture than used in traditional blockchain applications. Furthermore, the fact that the block bodies of older blocks could be deleted leads to the fact that the traceability of data is not guaranteed to be given any more. Users are not required to hold the whole blockchain anymore to be able to validate incoming data. It has to be examined whether this changed property of the blockchain might affect possible use cases of the blockchain technology.
- **Adaption for other blockchains**: The proposed approach focuses on Bitcoin-like blockchains where only one state (namely the balance of the different addresses) is present. There should be some research how the approach has to be adapted when facing an application supporting multiple states (e.g. the Ethereum network knows amongst other things the states for the current balances and the state of the executed smart contracts).

4 CONCLUSION

In this paper, we presented an approach based on the concept of Ethereum to include the state of the system (resp. the properties of the different addresses) in the blocks. It was proposed that this information could be used to create a proof that a certain address owns a certain property. This proof-of-property can be included into new transactions allowing other users to validate the transaction without having complete knowledge about the balances of every address of the system. The possibility to create these proofs derives from the fact that not all nodes in a Merkle Tree are needed to prove the absence of manipulations. Using this fact a proof for the owned property of a certain address can be created that does not contain any unneeded information. This enables sending transactions that include said proof without too much data overhead. The fact that transactions could be validated without having access to the whole system state enables users to participate in the network without having to download the blockchain beforehand and reduces the hard drive space needed to participate as a full node in the network.

ACKNOWLEDGMENTS

The European Union supported this work through the project CPS.HUB NRW, EFRE No. 0-4000-17.

REFERENCES

[1] Andreas M. Antonopoulos. 2015. *Mastering Bitcoin - Unlocking Digital Cryptocurrencies* (2 ed.). O'Reilly Media, Inc., Sebastopol.
[2] Ethan Buchman. 2014. Understanding the ethereum trie. (2014). Retrieved January 14, 2018 from https://easythereentropy.wordpress.com/2014/06/04/understanding-the-ethereum-trie/
[3] Vitalik Buterin. 2018. Ethereum white paper. (2018). Retrieved March 12, 2018 from https://github.com/ethereum/wiki/wiki/White-Paper
[4] Pedro Franco. 2014. *Understanding Bitcoin - Cryptography, Engineering and Economics* (1 ed.). John Wiley & Sons, New York.
[5] Antonio Madeira. 2018. What is SegWit? (2018). Retrieved January 14, 2018 from https://www.cryptocompare.com/coins/guides/what-is-segwit/
[6] Deborah Maxwell, Chris Speed, and Dug Campbell. 2015. 'Effing' the Ineffable: Opening Up Understandings of the Blockchain. In *Proceedings of the 2015 British HCI Conference (British HCI '15)*. ACM, New York, NY, USA, 208–209. https://doi.org/10.1145/2783446.2783593
[7] Donald R. Morrison. 1968. PATRICIA - Practical Algorithm To Retrieve Information Coded in Alphanumeric. *Journal of the ACM (JACM)* 15, 4 (Oct 1968), 514–534. https://doi.org/10.1145/321479.321481
[8] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008). Retrieved June 24, 2016 from https://bitcoin.org/bitcoin.pdf
[9] Joseph Poon and Thaddeus Dryja. 2016. The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments. (2016). Retrieved January 14, 2018 from https://lightning.network/lightning-network-paper.pdf
[10] Serguei Popov. 2017. The Tangle. (2017). Retrieved January 14, 2018 from https://iota.org/IOTA_Whitepaper.pdf
[11] Simone Porru, Andrea Pinna, Michele Marchesi, and Roberto Tonelli. 2017. Blockchain-oriented Software Engineering: Challenges and New Directions. In *Proceedings of the 39th International Conference on Software Engineering Companion (ICSE-C '17)*. IEEE Press, Piscataway, NJ, USA, 169–171. https://doi.org/10.1109/ICSE-C.2017.142
[12] Michael Szydlo. 2004. Merkle tree traversal in log space and time. In *Advances in Cryptology - EUROCRYPT 2004*. Springer, Berlin, Heidelberg, 541–554.
[13] Gavin Wood. 2017. Ethereum: A secure decentralised generalised transaction ledger. (2017). Retrieved March 12, 2018 from http://www.cryptopapers.net/papers/ethereum-yellowpaper.pdf
[14] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang. 2017. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. *2017 IEEE International Congress on Big Data (BigData Congress)* (June 2017), 557–564. https://doi.org/10.1109/BigDataCongress.2017.85