Do your readers need a blockchain?

Ivan Visconti[1], Andrea Vitaletti[2], and Marco Zecchini[1]

[1] University of Salerno, Fisciano, Italy
{visconti, mzecchini}@unisa.it
[2] Sapienza University of Roma, Rome, Italy
vitaletti@diag.uniroma1.it

In 2008, Bitcoin [Nak09] and its blockchain technology allowed mutually mistrusting entities to perform financial payments without relying on a central trusted third party while offering a transparent The birth of Ethereum [Vit14] enabled the adoption of the blockchain for many other use cases beyond financial transactions such as real estate, supply-chain management, royalty distribution and many others. The innovative potential of blockchain technology has created a huge hype among stakeholders deceiving them to adopt it also for applications where the blockchain is not the more appropriate solution.

"Do you need a blockchain?" [WG18]. In light of these considerations, many attempts have been made to provide a simple decision flow to allow users to verify whether blockchain technologies are indeed suitable for their application scenario. Among these attempts, Wüst and Gervais [WG18] proposed a very popular decisional flow[1] to critically evaluate whether the blockchain is the appropriate technical solution for an application. A blockchain is a distributed ledger with a growing list of records (i.e., blocks) securely linked together via cryptographic hashes allowing a set of *distrusting* peers (i.e., writers or validators) to agree on the state of the system through algorithmic consensus without relying on a third-trusted party (e.g., a server in client/server architecture). If a set of more writers do not trust each other and cannot use an always online third-trusted party, the diagram in [WG18] concludes that the use of the blockchain (permissioned or permissionless) is the appropriate technical solution. Hence, [WG18] justifies the use of the blockchain focusing only on the conditions of the writers.

In accordance with [WG18] conclusion, in supply-chain management application scenarios, a set of distrusting companies should adopt the blockchain to track information on their products. In this way, they increase transparency between themselves and with product customers. This process is called *notarization* and it is explored both in the scientific literature [KMGD+18, PCS+22] and industry [sku24, Hac21]. Such pieces of information are written in transactions.

The reading problem. [WG18] and applications focusing on notarization consider trivial the operation of reading transactions.

To read a transaction, a user has to access and verify whether transactions are actually in the ledger. Transactions are grouped into blocks and each block is linked to the previous one via a cryptographic hash. To verify that a transaction is in the ledger, users have to check that a) a transaction is included in a block and that b) this block is included in the blockchain, namely, the block is linked to the genesis block (i.e., the first block of the ledger) by a sequence of blocks.

Furthermore, blockchain readers may desire more complex readings. For example, readers might be interested in counting transactions with specific criteria (e.g., if a transaction stores expiration date, counting transactions of expired products). Hence, we are interested in the following challenging case: we consider a reader that wants to compute any function $f(BC, param)$, where BC denotes the blockchain (containing the transactions in the form described above) and $param$ are some relevant parameters used to compute f. For example, f can be used to count all the transactions matching a specific criterion passed as $param$. The reading of a specific transaction is a special use case of f where $param$ specifies the transaction we are looking for.

Peers that entirely store the ledger can perform this computation autonomously and trivially. However, not all the peers have sufficient resources to store the ledger[2]. This issue introduces the following problem: *How can peers with limited memory or bandwidth compute* $f(BC, param)$*?* We refer to this problem as the *reading problem*. The key observation here is that if we do not carefully design the reading process, the natural ambition of implementing a decentralized notarization process on-chain is at risk, since for several practical reasons, the reading process might require the introduction of some level of trust which is in contradiction with [WG18].

An updated diagram. In this oral communication, we focus on the reading problem and how it affects the choice of application designers adopting the blockchain. In particular, we present the flow diagram in Figure 1 which extends the diagram of [WG18]. The diagram summarizes the counter-effects of not considering properly the reading problem. In particular, as we said above,

[1] At the time of writing the paper is cited by 1469 documents according to Google Scholar database

[2] The Ethereum documentation recommends that to set up a full node client users need more than 2TB of fast SSD memory, 25 MBit/s bandwidth, 16GB of RAM.

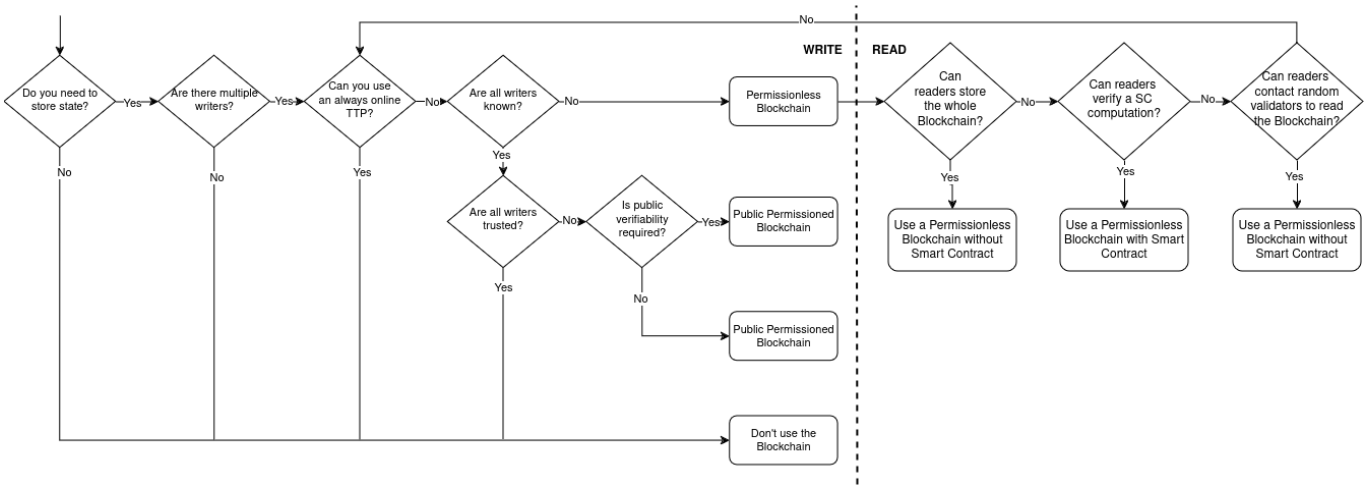

Figure 1: An extension of [WG18] flow chart to determine if the reading problem is not introducing some level of trust.

if readers can store the entire ledger, they can compute $f(BC, params)$ autonomously without relying on external nodes. In this case, we can also use a blockchain that does not have smart contracts (e.g., Bitcoin). However, since many readers can realistically run on constrained devices (e.g., smartphones), storing the whole BC is unfeasible. In some cases, $f(BC, params)$ can be computed on smart contracts. However, some functions cannot be implemented on SC (e.g., the storage space of SC can be limited) or running the SC might be too expensive. Furthermore, even when the $f(BC, params)$ can be implemented on a SC, clients should be able to verify the correctness of the output. Concepts such as the Ethereum light nodes [lig24] support a verifiable and lighter computation, however to the best of our knowledge they are not yet available in production. Finally, if the reader device is highly limited in computation and/or storage or $f(BC, params)$ cannot be computed by a smart contract, a final solution is to contact at random multiple validators querying $f(BC, params)$ and compare their outputs choosing the one proposed by the majority of the network. Note that this solution is weaker than the one adopting smart contracts when a limited number of validators implement $f(BC, params)$. This might happen when the reading function is highly application-specific (e.g., see this use case as an example [VZ23]). If also this case is not realizable because $f(BC, params)$ is implemented by only one node we introduce again a TTP in the picture. Recent advancements in zkSNARKS might suggest novel approaches where the verifiable computation is performed offline while the reader should only be able to verify the proofs. However, implementing such zkSNARKS for generic functions might be challenging and SC should be able to verify the proof to properly incentivize its computation by third parties (notice that in this case, we do not need TTP, since the proof can be verified and consequently does not require trust).

References

[Hac21] Robert Hackett. Walmart and IBM Are Partnering to Put Chinese Pork on a Blockchain. *Fortune*, June 2021. https://fortune.com/2016/10/19/walmart-ibm-blockchain-china-pork.

[KMGD+18] Athina-Styliani Kleinaki, Petros Mytis-Gkometh, George Drosatos, Pavlos S. Efraimidis, and Eleni Kaldoudi. A blockchain-based notarization service for biomedical knowledge retrieval. *Computational and Structural Biotechnology Journal*, 16:288–297, 2018.

[lig24] Light clients | ethereum.org, April 2024. https://ethereum.org/en/developers/docs/nodes-and-clients/light-clients [Online; accessed 8. Apr. 2024].

[Nak09] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. http://www.bitcoin.org/bitcoin.pdf.

[PCS+22] Tonino Palmisano, Vito Nicola Convertini, Lucia Sarcinella, Luigia Gabriele, and Mariangela Bonifazi. Notarization and anti-plagiarism: A new blockchain approach. *Applied Sciences*, 12(1), 2022.

[sku24] Skuchain, April 2024. https://www.skuchain.com [Online; accessed 8. Apr. 2024].

[Vit14] Buterin Vitalik. Ethereum white paper: A next generation smart contract & decentralized application platform, 2014. https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum_Whitepaper_-_Buterin_2014.pdf.

[VZ23] Andrea Vitaletti and Marco Zecchini. A tale on decentralizing an app: the case of copyright management. In *5th Distributed Ledger Technology Workshop (DLT 2023)*, 2023.

[WG18] Karl Wüst and Arthur Gervais. Do you need a blockchain? In *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pages 45–54, 2018.