# A Wake-up Call: Managing Data in an Untrusted World

Divyakant Agrawal, Amr El Abbadi
University of California, Santa Barbara

Once upon a time databases were structured, one size fitted all and they resided on machines that were trustworthy and even when they failed, they simply crashed. This era has come and gone as eloquently stated by Stonebraker and Cetintemel [16]. We now have key-value stores, graph databases, text databases, and a myriad of unstructured data repositories. The database community has wholeheartedly accepted the fact that the same information might come in different formats, modes and representations. We also accept that data might not be "clean" and that data might need to be "cleaned" due to the diverse sources of information. However, we, as a database community still cling to our $20^{th}$ century belief that databases always reside on trustworthy, honest servers. Although the database community has always considered fault-tolerance as an integral building block of data management (remember "D" in ACID is for Durability), we still have trouble accepting the fact that not all failures are simply crash failures and might in fact involve malicious and non-trustworthy infrastructure. This notion has been challenged and abandoned by many other Computer Science communities, most notably the security and the distributed systems communities. The rise of the cloud computing paradigm as well as the rapid popularity of blockchains demand a rethinking of our naïve, comfortable beliefs in an ideal benign infrastructure. In the cloud, clients store their sensitive data in remote servers owned and operated by cloud providers. The Security and Crypto Communities have made significant inroads to protect both data and access privacy from malicious untrusted storage providers using encryption and oblivious data stores. The Distributed Systems and the Systems Communities have developed consensus protocols to ensure the fault-tolerant maintenance of data residing on untrusted, malicious infrastructure. However, these solutions face significant scalability and performance challenges when incorporated in large scale data repositories. Novel database designs need to directly address the natural tension between performance, fault-tolerance and trustworthiness. This is a perfect setting for the database community to lead and guide.

In this opinion article, we will illustrate an interesting atomicity problem in the context of exchanging cryptocurrencies using permissionless blockchains. We also illustrate that transaction management can learn from the blockchain approach when attempting to restrict untrusted behaviour from the underlying infrastructure. We hope this will illustrate some of the challanges that need to be addressed in malicious, untrusted settings, as well as connections with standard database problems like commitment, replication and transaction management in general.

Bitcoin [14] took the world by surprise over a decade ago by introducing a novel cryptocurrency, which supports the execution of transactions, albeit relatively simple transfers of assets. As with any data management system, bitcoin needed to address the challenge of durability. Its approach, rather than using a trusted persistence storage device to store the current state, e.g., a disk, relies on global replication of a distributed data structure called a *blockchain*. A blockchain is basically a linked list of blocks, each of them containing a set of transactions. The blockchain is replicated on a potentially unknown number of *untrusted* servers. Two of the main challenges that bitcoin needed to address are (1) the unknown number of participants and (2) the untrusted nature of the participants. The former challenge basically manifests itself when consensus is needed to add a new block to the blockchain. This is resolved by avoiding communication to achieve consensus, as is typical in Byzantine Agreement protocols, and replacing it with computation, namely the well-known *Proof of Work* mining puzzle


**Bulletin of the IEEE Computer Society Technical Committee on Data Engineering**

operation. The latter challenge is addressed using many different cryptographic techniques, including digital signatures, hash pointers, etc. Currently, it seems that database vendors have adopted blockchains, with their untrusted infrastructure, but only in contexts where the number of untrusted participants is known *a priori*. This problem is suitable for supply-chain, financial and healthcare applications. In blockchain terminology, this is referred to as *permissioned blockchains*. Over the past few years different permissioned blockchain systems have been developed in both industry and academia. Some known industrial permissioned blockchain systems include Hyperledger Fabric [5] which was introduced by IBM and is widely used in supply chain management, Quorum [6] by JPMorgan which supports financial applications, and Libra [13] is a digital currency by Facebook. Similarly, in academia, Fast Fabric [8], ParBlockchain [4], ResilientDB [9], Caper [3], AHL [7], etc. have been proposed. This is a good step, and demonstrates our willingness to address the challenges of untrusted as well as unreliable infrastructures. However, we believe database researchers have a lot more to offer. These efforts go in both directions, i.e., to extend support for untrusted infrastructures in diverse data management contexts, as well as exploiting data management techniques in diverse novel and non-traditional contexts, as for example in permissionless blockchains.

Bitcoin and other cryptocurrencies are *permissionless* blockchains. In a permissionless blockchain, the network is public, and anyone can participate without a specific identity. The recent adoption of blockchain technologies and open permissionless networks has demonstrated user needs to exchange assets and especially without depending on centralized intermediaries such as banks or exchanges. This problem requires infrastructure enablers and protocols that allow users to *atomically* exchange assests without giving up trust-free decentralization, the main reasons behind using permissionless blockchain. We motivate the problem of atomic cross-chain transactions and discuss the current available solutions and their limitations through the following example.

Suppose Alice owns X bitcoins and she wants to exchange them for Y ethers. Luckily, Bob owns ether and he is willing to exchange his Y ethers for X bitcoins. In this example, Alice and Bob want to atomically exchange assets that reside in different blockchains. In addition, both Alice and Bob *do not trust* each other and in many scenarios, they might not be co-located to do this atomic exchange in person. Current infrastructures do not support these direct peer-to-peer transactions. Instead, both Alice and Bob need to *independently* exchange their assets through a trusted centralized exchange, Trent (e.g., Coinbase [1] and Robinhood [2]) either through fiat currency or directly. Using fiat, both Alice and Bob first exchange their assets with Trent for a fiat currency (e.g., USD) and then use the earned fiat currency to buy the other assets also from Trent or from another trusted exchange. Alternatively, some exchanges (e.g., Coinbase) allow their customers to directly exchange assets (ether for bitcoin or bitcoin for ether) without going through fiat currencies.

A two-party cross-chain commitment protocol was originally proposed by Nolan [15] and generalized by Herlihy [10] to process multi-party cross-chain transactions, or swaps. Both Nolan's protocol and its generalization by Herlihy use smart contracts, hashlocks, and timelocks to execute cross-chain transactions. A *smart contract* is a self executing contract (or a program) that is executed in a blockchain once all the terms of the contract are satisfied. A *hashlock* is a cryptographic one-way hash function $h = H(s)$ that locks assets in a smart contract until a hash secret $s$ is provided. A *timelock* is a time bounded lock that triggers the execution of a smart contract function after a pre-specified time period. These proposals solve the cross-chain commitment problem and ensure that untrusted participants comply and do not try to misuse the system. However, an expired timelock could lead to a violation of the all-or-nothing atomicity property. An honest participant who fails to react in time due to a crash failure, network delays or even a denial of service attack might end up losing their assets. Although a crashed participant is the only participant who ends up worse off, current proposals are unsuitable for atomic cross-chain transactions in asynchronous environments where crash failures and network delays are the norm. This is a problem familiar to the database community since its inception, namely the *atomic commitment problem*. In [17], we present a decentralized all-or-nothing *atomic* cross-chain commitment protocol. Events for redeeming and refunding smart contracts to exchange assets are modeled as conflicting events. An open permissionless network of witnesses is used to guarantee that conflicting events could never simultaneously occur and either all smart contracts in an atomic cross-chain transaction are redeemed or all of them are refunded. A similar protocol

was also concurrently proposed by Herlihy et al. [11], which addresses the same atomicity challenge, but in the context of *cross chain deals*, where a *deal* is a weaker notion of an atomic transaction.

Now, we briefly come back to traditional databases and large scale data fault-tolerant transaction management but on untrusted infrastructures. As increasing amounts of data are currently being stored and managed on third-party servers, there is emerging demand for a suite of protocols to manage data on untrusted infrastructures. In Fides [12], we introduce a novel atomic commitment protocol, TFCommit, that executes transactions on data stored across multiple untrusted servers. This novel atomic commitment protocol executes transactions in an untrusted environment without the need for expensive Byzantine replication. One main obstacle for the adoption of Byzantine Agreement protocols are the various impossibility results, that require *a priori* knowledge of the maximum possible number of failures in the system (e.g., one third in case of malicious failures). From a practical point of view, this might be difficult to ascertain. We propose using TFCommit in an *auditable* data management system, Fides, residing completely on untrustworthy infrastructures. As an auditable system, Fides guarantees the detection of potentially malicious failures occurring on untrusted servers using blockchain inspired tamper-resistant logs with the support of cryptographic techniques. As a result, Fides is scalable and does not require any *a priori* known bounds on the number of malicious faults when executing transactions on untrusted infrastructure. If malicious behaviour occurs, it is recorded and can be detected by an external auditor. Just the threat of investigation should usually deter improper behaviour.

In conclusion, we encourage researchers to explore a relatively new and uncharted domain for the database community. Many techniques have indeed been developed by the security, cryptography and distributed systems communities. However, these solutions are often too expensive to use in the practical scalable settings data management systems encounter on a daily basis. In fact, we view this as a two way exchange of ideas and innovations. Over four decades of fault-tolerant data management innovations can be beneficial and can have significant impact in solving many of the practical large scale problems that need to be addressed in untrusted infrastructures. On the other hand, many of the techniques developed to restrict malicious behaviour using cryptographic and distributed computing approaches can facilitate the development of fault-tolerant database management systems in untrused infrastructures, which are becoming more prevalent and commonplace for storing data.

## Acknowledgement

This work is partially funded by NSF grants CNS-1703560 and CNS-1815733.

## References

[1] Coinbase. `https://coinbase.com`, 2018.

[2] Robinhood. `https://robinhood.com/`, 2018.

[3] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. Caper: a cross-application permissioned blockchain. *Proceedings of the VLDB Endowment*, 12(11):1385–1398, 2019.

[4] Mohammad Javad Amiri, Divyakant Agrawal, and Amr El Abbadi. Parblockchain: Leveraging transaction parallelism in permissioned blockchain systems. In *39th International Conference on Distributed Computing Systems (ICDCS)*, pages 1337–1347. IEEE, 2019.

[5] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *EuroSys Conference*, page 30. ACM, 2018.

[6] JP Morgan Chase. Quorum white paper, 2016.

[7] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. Towards scaling blockchain systems via sharding. In *Proceedings of the 2019 ACM SIGMOD International Conference on Management of Data*. ACM, 2019.

[8] Christian Gorenflo, Stephen Lee, Lukasz Golab, and S. Keshav. Fastfabric: Scaling hyperledger fabric to 20,000 transactions per second. *arXiv preprint arXiv:1901.00910*, 2019.

[9] Suyash Gupta, Sajjad Rahnama, Jelle Hellings, and Mohammad Sadoghi. Resilientdb: Global scale resilient blockchain fabric. *arXiv preprint arXiv:2002.00160*, 2020.

[10] Maurice Herlihy. Atomic cross-chain swaps. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 245–254. ACM, 2018.

[11] Maurice Herlihy, Liuba Shrira, and Barbara Liskov. Cross-chain deals and adversarial commerce. *Proc. VLDB Endow.*, 13(2):100–113, 2019.

[12] Sujaya Maiyya, Danny Hyun Bum Cho, Divyakant Agrawal, and Amr El Abbadi. Fides: Managing data on untrusted infrastructure. *arXiv preprint arXiv:2001.06933*, 2020.

[13] Libra Association Members. An introduction to libra. *https://libra.org/en-US/white-paper/*, 2020.

[14] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[15] Tier Nolan. Alt chains and atomic transfers. `https://bitcointalk.org/index.php?topic=193281.msg2224949#msg2224949`, 2013.

[16] Michael Stonebraker and Ugur Çetintemel. "one size fits all": An idea whose time has come and gone (abstract). In Karl Aberer, Michael J. Franklin, and Shojiro Nishio, editors, *Proceedings of the 21st International Conference on Data Engineering, ICDE 2005, 5-8 April 2005, Tokyo, Japan*, pages 2–11. IEEE Computer Society, 2005.

[17] Victor Zakhary, Divyakant Agrawal, and Amr El Abbadi. Atomic commitment across blockchains. *Proceedings of the VLDB Endowment*, 13:1, 2020.