Efficient Transaction Processing in Byzantine Fault Tolerant Environments

Suyash Gupta[*] Jelle Hellings[†] Thamir Qadah[‡]
Sajjad Rahnama[§] Mohammad Sadoghi[¶]

Exploratory Systems Lab
Department of Computer Science
University of California, Davis
CA 95616-8562, USA

Abstract

The overarching goal of any state-of-the-art blockchain application [5] is ensuring the integrity of the client transactions. A blockchain in its simplest form is a *linked-list*, which helps to maintain a stable and reliable storage of client transactions. Blockchain has acted as a resolve to challenges in food production [7], energy trading [22], managing health care [4, 9, 16], and insurance fraud prevention [14]. The key reasons behind widespread interest in blockchain systems are their fundamental characteristics: transparency, integrity and decentralization.

A blockchain supports transparency and decentralization, by employing the core database principle of *replication*. Each blockchain is maintained by several replicas. These replicas need to reach an agreement on the order of client transactions, which is resolved by employing a *consensus* protocol. Thus, at the core of any blockchain system is its underlying consensus protocol. Intial blockchain applications, such as Bitcoin [20] and Ethereum [25] encourage a *permissionless* setting through the use the Proof-of-Work [10, 15] (PoW) protocol to attain consensus. PoW requires each replica to spend its resources in finding a solution for some hard cryptographic puzzle, which acts as a proof that the replica has generated the next block. Hence all the replias reach a consensus when at least one replica transmits its proof to the majority of replicas. However, PoW protocol can lead to the case where multiple replicas can claim to have generated the proof, which could create a *fork* in the blockchain.

[*]`https://gupta-suyash.github.io/`
[†]`https://jhellings.nl/`
[‡]`http://thamir.qadah.com/`
[§]`https://sajjadrahnama.com/`
[¶]`https://msadoghi.github.io/`

Further, these permissionless protocols need to provide replicas with financial incentives to maximize non-malicious participation. These requirements necessitated the design of *permissioned* systems [1, 10], which build on top of the proven consensus protocols as the identities of the replicas are known apriori. We realize this goal of a safe and efficient, permissioned blockchain system through our design of ExpoDB framework. ExpoDB provides an in-memory distributed transactional framework that allows implementing and evaluating different concurrency control and agreement protocols. Our past experience with ExpoDB includes: (i) design of an efficient concurrency control protocol, QueCC [23], (ii) design of a fast non-blocking two-phase agreement protocol, EasyCommit [11], (iii) design of toplogoy-aware geo-scale agreement protocol, GeoScale EasyCommit [12], and (iv) implementation of scalable storage layer, LStore [24]. These protocols, although exciting, do not meet the requirements of a practical blockchain application. Hence we extend the ExpoDB framework and set out goals for designing efficient blockchain systems.

Our current design of ExpoDB includes implementation of several byzantine fault-tolerant consensus algorithms, such as PBFT [6], Zyzzyva [18], RBFT [2], PoW [15], and Algorand [8]. ExpoDB also implements two popular blockchain systems, Hyperledger Fabric [1] and RapidChain [26]. Note that our focus is mainly in the design of efficient byzantine fault-tolerant (BFT) consensus protocols. These protocols differ from the Paxos-style [19, 21] consensus protocols as they allow replicas to be malicious.

The existence of malicious replicas influences the design of underlying consensus protocol, which in turn affects the performance of the system. Prior works [6, 18] have noted the expensive costs associated with the BFT consensus protocols and have advocated for efficient future designs. We tackle this cost trade-off by envisioning a manifold design. ExpoDB customizes each replica by associating with it an elaborate parallel pipeline architecture. These parallel pipelines allow us to process multiple batches of client transactions, in parallel, without comprising on linearizability [13]. Further, we employ efficient cryptographic constructs [3]: (i) to digitally sign the message (256-bit ED25519 [17]) between clients and replicas, (ii) to ensure integrity of the message (128-bit CMAC-AES) between replicas, and (iii) to authenticate the message contents using hashing (SHA256).

The use of efficient pipelines and constructs can only boost the system performance to a limited extent. The key component still affecting the system throughput is the underlying consensus protocol. Although the PBFT [6] protocol is the first to design of a practical BFT consensus, it requires three phases of which two require quadratic communication. One of the most noteworthy improvement over PBFT is Zyzzyva [18], which requires only linear communication through speculative execution. However, its safety and liveness is dependent on existence of good clients.

We believe there is ample opportunity to develop efficient BFT protocols that work in fewer phases than PBFT and do not face the limitations of Zyzzyva. Further, the existing BFT designs do not discuss the challenges of a geographically large setup, where replicas are spread across continents and latencies are the major concerns. Existing blockchain applications also assume existence of a single chain, which is maintained across all the replicas. Although this chain helps to ensure a single order, it restricts parallelism and requires massive storage. We believe through the use of *sharding*, a sharded-chain can help scaling the blockchain systems.

References

[1] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick. Hyperledger Fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, pages 30:1–30:15. ACM, 2018. doi:10.1145/3190508.3190538.

[2] Pierre-Louis Aublin, Sonia Ben Mokhtar, and Vivien Quéma. RBFT: Redundant byzantine fault tolerance. In *2013 IEEE 33rd International Conference on Distributed Computing Systems*, pages 297–306. IEEE, 2013. doi:10.1109/ICDCS.2013.53.

[3] Elaine Barker. Recommendation for key management part 1: General. Technical report, NIST, 2016. NIST Special Publication 800-57 Part 1, Revision 4. doi:10.6028/NIST.SP.800-57pt1r4.

[4] Burkhard Blechschmidt. Blockchain in Europe: Closing the strategy gap. Technical report, Cognizant Consulting, 2018. URL: https://www.cognizant.com/whitepapers/blockchain-in-europe-closing-the-strategy-gap-codex3320.pdf.

[5] Christian Cachin and Marko Vukolic. Blockchain consensus protocols in the wild (keynote talk). In *31st International Symposium on Distributed Computing*, volume 91 of *Leibniz International Proceedings in Informatics*, pages 1:1–1:16. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.DISC.2017.1.

[6] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, pages 173–186. USENIX Association, 1999.

[7] Lan Ge, Christopher Brewster, Jacco Spek, Anton Smeenk, and Jan Top. Blockchain for agriculture and food: Findings from the pilot study. Technical report, Wageningen University, 2017. URL: https://www.wur.nl/nl/Publicatie-details.htm?publicationId=publication-way-353330323634.

[8] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 51–68. ACM, 2017. doi:10.1145/3132747.3132757.

[9] William J. Gordon and Christian Catalini. Blockchain technology for

healthcare: Facilitating the transition to patient-driven interoperability. *Computational and Structural Biotechnology Journal*, 16:224–230, 2018. doi:10.1016/j.csbj.2018.06.003.

[10] Suyash Gupta and Mohammad Sadoghi. *Blockchain Transaction Processing*, pages 1–11. Springer International Publishing, 2018. doi:10.1007/978-3-319-63962-8_333-1.

[11] Suyash Gupta and Mohammad Sadoghi. EasyCommit: A non-blocking two-phase commit protocol. In *Proceedings of the 21st International Conference on Extending Database Technology*. Open Proceedings, 2018. doi:10.5441/002/edbt.2018.15.

[12] Suyash Gupta and Mohammad Sadoghi. Efficient and non-blocking agreement protocols. *Distributed and Parallel Databases*, 2019. doi:10.1007/s10619-019-07267-w.

[13] Maurice P. Herlihy and Jeannette M. Wing. Linearizability: A correctness condition for concurrent objects. *ACM Transactions on Programming Languages and Systems*, 12(3):463–492, 1990. doi:10.1145/78969.78972.

[14] Matt Higginson, Johannes-Tobias Lorenz, Björn Münstermann, and Peter Braad Olesen. The promise of blockchain. Technical report, McKinsey&Company, 2017. URL: https://www.mckinsey.com/industries/financial-services/our-insights/the-promise-of-blockchain.

[15] Markus Jakobsson and Ari Juels. *Proofs of Work and Bread Pudding Protocols*, pages 258–272. Springer US, 1999. doi:10.1007/978-0-387-35568-9_18.

[16] Maged N. Kamel Boulos, James T. Wilson, and Kevin A. Clauson. Geospatial blockchain: promises, challenges, and scenarios in health and healthcare. *International Journal of Health Geographics*, 17(1):1211–1220, 2018. doi:10.1186/s12942-018-0144-x.

[17] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2nd edition, 2014.

[18] Ramakrishna Kotla, Lorenzo Alvisi, Mike Dahlin, Allen Clement, and Edmund Wong. Zyzzyva: Speculative byzantine fault tolerance. In *Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles*, pages 45–58. ACM, 2007. doi:10.1145/1294261.1294267.

[19] Leslie Lamport. The implementation of reliable distributed multiprocess systems. *Computer Networks (1976)*, 2(2):95–114, 1978. doi:10.1016/0376-5075(78)90045-4.

[20] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. URL: https://bitcoin.org/en/bitcoin-paper.

[21] Brian M. Oki and Barbara H. Liskov. Viewstamped replication: A new primary copy method to support highly-available distributed systems. In *Proceedings of the Seventh Annual ACM Symposium on Principles of Distributed Computing*, pages 8–17. ACM, 1988. doi:10.1145/62546.62549.

[22] PwC. Blockchain – an opportunity for energy producers and consumers?, 2016. URL: https://www.pwc.com/gx/en/industries/energy-utilities-resources/publications/opportunity-for-energy-producers.html.

[23] Thamir M. Qadah and Mohammad Sadoghi. QueCC: A queue-oriented, control-free concurrency architecture. In *Proceedings of the 19th International Middleware Conference*, pages 13–25. ACM, 2018. `doi:10.1145/3274808.3274810`.

[24] Mohammad Sadoghi, Souvik Bhattacherjee, Bishwaranjan Bhattacharjee, and Mustafa Canim. L-Store: A real-time OLTP and OLAP system. In *Proceedings of the 21th International Conference on Extending Database Technology*, pages 540–551, 2018. `doi:OpenProceedings.org`.

[25] Gavin Wood. Ethereum: a secure decentralised generalised transaction ledger. EIP-150 revision. URL: `https://gavwood.com/paper.pdf`.

[26] Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. Rapidchain: Scaling blockchain via full sharding. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 931–948. ACM, 2018. `doi:10.1145/3243734.3243853`.