Free2Shard: Adversary-resistant Distributed Resource Allocation for Blockchains

Ranvir Rana
rbrana2@illinois.edu
University of Illinois at Urbana-Champaign
USA

Sreeram Kannan
ksreeram@uw.edu
University of Washington at Seattle
USA

David Tse
dntse@stanford.edu
Stanford University
USA

Pramod Viswanath
pramodv@illinois.edu
University of Illinois at Urbana-Champaign
USA

ABSTRACT

In this paper, we formulate and study a new, but basic, distributed resource allocation problem arising in scaling blockchain performance. While distributed resource allocation is a well-studied problem in networking, the blockchain setting additionally requires the solution to be resilient to adversarial behavior from a fraction of nodes. Scaling blockchain performance is a basic research topic; a plethora of solutions (under the umbrella of *sharding*) have been proposed in recent years. Although the various sharding solutions share a common thread (they cryptographically stitch together multiple parallel chains), architectural differences lead to differing resource allocation problems. In this paper we make three main contributions: (a) we categorize the different sharding proposals under a common architectural framework, allowing for the emergence of a new, uniformly improved, *uni-consensus* sharding architecture. (b) We formulate and exactly solve a core resource allocation problem in the uni-consensus sharding architecture – our solution, Free2shard, is adversary-resistant and achieves optimal throughput. The key technical contribution is a mathematical connection to the classical work of Blackwell approachability in dynamic game theory. (c) We implement the sharding architecture atop a full-stack blockchain in 3000 lines of code in Rust – we achieve a throughput of more than 250,000 transactions per second with 6 shards, a vast improvement over state-of-the-art. The code is available at [1].

ACM Reference Format:
Ranvir Rana, Sreeram Kannan, David Tse, and Pramod Viswanath. 2022. Free2Shard: Adversary-resistant Distributed Resource Allocation for Blockchains. In *Abstract Proceedings of the 2022 ACM SIGMETRICS/IFIP PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS/PERFORMANCE '22 Abstracts), June 6–10, 2022, Mumbai, India.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3489048.3522651

SIGMETRICS/PERFORMANCE '22 Abstracts, June 6–10, 2022, Mumbai, India

ACM ISBN 978-1-4503-9141-2/22/06.
https://doi.org/10.1145/3489048.3522651

1 INTRODUCTION

A classical problem in distributed systems is one of maintaining a state machine given N nodes, some fraction of which are adversarial (also termed Byzantine). Classical mechanisms for Byzantine-fault-tolerant (BFT) state-machine-replication (SMR) rely on full replication of data across multiple nodes, thus offering no scaling in efficiency as the number of replica nodes increases [2]. Since SMR is the key primitive underlying blockchains, it is no surprise that the first generation of blockchains also relied on full replication [6]. This problem has attracted wide interest in the distributed systems community, with many sharding protocols being proposed [4, 5]. These pioneering methods offer provable security as well as near-linear scaling in N of efficiency across various resources at an individual node, including computation, storage, and communication. Sharding achieves the performance gains by relying on partial replication. A potential security trade-off is made since each part of the state is not maintained by all nodes. From a security standpoint, resistance is desired against an *adaptive adversary*, one which can corrupt a (positive) fraction of nodes *after* viewing the current public state of the blockchain.

The security trade-off is managed by designing allocation algorithms that allocate nodes to shards ensuring all shards get enough honest nodes. In existing sharding solutions, the allocation happens by running a Node to Shard (N2S) allocation algorithm that mandates an on-chain node identity. The algorithm aims to distribute nodes randomly and equally to all shards. This approach has several drawbacks:

- A node identity prohibits the use of native Proof-of-work (PoW) mining, which is identity-free.
- It makes the ledger susceptible to adaptive adversaries that can target a shard by specifically targeting the small number of nodes *after they have been allocated to shard.*
- Finally, a N2S algorithm needs to know the set of nodes to allocate; this prevents *dynamic availability*, the nodes' ability to join and leave the system dynamically as per their will.

The main issue with existing schemes is the absence of a distributed identity-less resource allocation algorithm that dynamically adapts as a response to adaptive adversarial allocations; the main result of this paper is the Free2Shard-dist DSA algorithm which exactly does that. The throughput performance of Free2Shard uni-consensus architecture scales near-linearly with

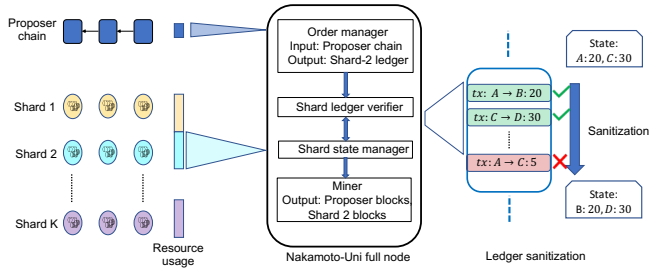

Figure 1: Nakamoto-uni full node

the number of nodes while maintaining *identity-free PoW mining*, supporting *dynamic availability* and being secure against *fully adaptive adversaries* controlling up to 50% of the nodes. The core technical result is a complete and striking solution to a dynamic Stackelberg game [7]; our approach is distinct but inspired by the classical Blackwell approachability in game theory [3].

2 CONTRIBUTION

2.1 Nakamoto-Uni **uniconsensus architecture**

We design a sharding architecture such that even when the majority of shard nodes are adversarial, the safety is not violated. Removal of a shard's honest majority requirement allows nodes to allocate themselves to shards as they please, providing full compatibility with identity-free PoW mining and ensuring dynamic availability. The architecture (Figure 1) utilizes a single consensus engine with a global honest majority ordering proposer blocks that are maintained by everyone and K shard chains with each node maintaining a single shard. The architecture decouples ordering and shard transaction validation by running the order manager and shard ledger verifier asynchronously.

2.2 Free2Shard-dist **DSA**

We identify a liveness vulnerability in the uniconsensus architecture: An adversary can congregate in a shard, significantly reducing the fraction of honest shard blocks, creating corresponding liveness vulnerability, and restricting throughput. We design Free2Shard-dist Dynamic Self-Allocation algorithm to ensure liveness. The core idea is that the honest nodes re-allocate themselves to shards throttled by the adversary. However, the adversary can observe the honest nodes' actions and re-allocate itself to nullify the honest nodes' actions. The main technical contribution of this paper is the identification of a (computationally simple) dynamic self-allocation policy that can successfully ensure that the fraction of honest to adversarial nodes in *every shard* is essentially equal to the ratio of honest to adversarial nodes in the entire network (thus ensuring that honest transactions are not throttled by censoring adversarial blocks). The DSA policy achieves this information-theoretic limit by allocating honest power to the shards with the highest lag of time-averaged honest fraction from the target honest fraction of γ: the ratio of honest mining power to total mining power in the entire network.

Our protocol has several desirable properties that makes it attractive from a systems view: (a) asynchronous shard rotation - nodes do not all rotate at the same time; (b) requires only a small number of honest nodes per shard; (c) guarantees the aforementioned properties even when only a small minority of nodes follow the proposed rotation protocol; (d) can support heterogeneous shard throughput even when the number of shards is greater than the number of nodes.

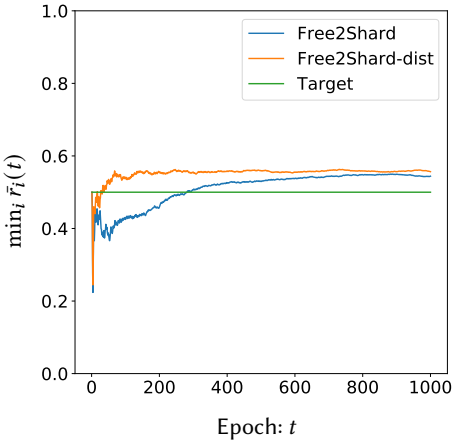

Figure 2: Worst-case average honest fraction

2.3 System implementation and simulations

We implement Free2Shard architecture in about 3000 lines of Rust code on top of a state-of-the-art PoW consensus with full UTXO integration. We see throughput growing linearly with shards in practice; with 6 shards, we achieve a throughput of more than 250,000 transactions per second. We simulated Free2Shard DSA with a fully adaptive adversary to verify its robustness in a realistic setting. The experiments showed that both Free2Shard and Free2Shard-dist perform near optimally as shown in figure 2. The code is available at [1].

3 ACKNOWLEDGEMENTS

This research was partly supported by US Army Research Office Grant W911NF-18-1-0332, National Science Foundation CCF-1705007, NeTS 1718270 and the XDC network.

REFERENCES

[1] [n.d.]. *Free2Shard uniconsensus system implementation.* https://github.com/ranvirranaiitb/free2shard-rust
[2] Ittai Abraham, TH Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. 2019. Communication complexity of byzantine agreement, revisited. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*. 317–326.
[3] David Blackwell et al. 1956. An analog of the minimax theorem for vector payoffs. *Pacific J. Math.* 6, 1 (1956), 1–8.
[4] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. 2018. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 583–598.
[5] Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, and Prateek Saxena. 2016. A secure sharding protocol for open blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 17–30.
[6] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).
[7] Heinrich von Stackelberg. 2011. *Market Structure and Equilibrium*. Springer.