Short Paper: Service-Oriented Sharding for Blockchains

Adem Efe Gencer Robbert van Renesse Emin Gün Sirer
Initiative for CryptoCurrencies and Contracts (IC3)
Computer Science Department, Cornell University

Abstract. The rise of blockchain-based cryptocurrencies has led to an explosion of services using distributed ledgers as their underlying infrastructure. However, due to inherently single-service oriented blockchain protocols, such services can bloat the existing ledgers, fail to provide sufficient security, or completely forego the property of trustless auditability. Security concerns, trust restrictions, and scalability limits regarding the resource requirements of users hamper the sustainable development of loosely-coupled services on blockchains.
This paper introduces Aspen, a sharded blockchain protocol designed to securely scale with increasing number of services. Aspen shares the same trust model as Bitcoin in a peer-to-peer network that is prone to extreme churn containing Byzantine participants. It enables introduction of new services without compromising the security, leveraging the trust assumptions, or flooding users with irrelevant messages.

1 Introduction

Blockchains offer many opportunities for facilitating innovation in traditional industries. They have received extensive attention due to the trustless auditability, tamper-resistance, and transparency they provide in networks with Byzantine participants. Not surprisingly, there is much commercial interest in developing specialized blockchain solutions [13]. There have been proposals to use blockchains as an underlying layer for services including managing digital assets [14], issuing securities [12], tracking intellectual property [8,22,28], maintaining land records and deeds [1,30], facilitating online voting [2], registering domain names [24], as well as others. Ongoing projects explore ways to make it easier to build such services using Blockchain-as-a-Service (BaaS) platforms [20, 26].

This movement, towards increased adoption of blockchains for specialized purposes, portends a dangerous trend: accommodating all of these diverse uses, either in a single blockchain or in separate blockchains, inherently requires complex tradeoffs. The simplest approach, of layering these additional blockchains on top of an existing, secure blockchain with sufficient mining power to withstand attacks, such as Bitcoin [27], leads to a stream of costly and burdensome transactions. Indeed, we have seen the controversial `OP_RETURN` opcode adopted for this purpose, and its use has been increasing rapidly [4], in line with increased usage of layered blockchains. Yet these transactions, which use Bitcoin solely as a timestamping and ordering service, increase the resource requirements for system participation and the time to bootstrap a node. In contrast, creating a dedicated, specialized, standalone blockchain avoids this problem, but suffers from a lack of independent mining power to secure the infrastructure. Duplicating the mining infrastructure used to secure Bitcoin is not only costly and

environmentally unfriendly, but it is difficult to bootstrap such a system. Faced with this dilemma, some have turned to permissioned ledgers with closed participants [11, 23], forgoing the open architecture, the flexible trust model, and the strong security guarantees of the existing Bitcoin mining ecosystem.

In this paper, we present Aspen, a protocol that securely shards the blockchain to provide high scalability to service users. This protocol employs a sharding approach that comes with the following benefits: (1) preserves the total computational power of miners to secure the whole blockchain, (2) prevents users from double-spending their funds while maintaining the same trustless setup assumptions as Bitcoin, (3) improves scalability by absolving non-miner participants – i.e. service users – from the responsibility of storing, processing, and propagating irrelevant data to confirm the validity of services they are interested in. In this protocol, a coffee shop does not have to worry about the land and deed records in the blockchain to validate the payment system.

Sharding is a well-established technique to improve scalability by distributing contents of a database across nodes in a network. But sharding blockchains is non-trivial. The main difficulty is to preserve the trustless nature while hiding parts of a blockchain from other nodes. It is an open research question whether it is possible to shard blockchains in a way that the output of a transaction in one shard can be spent at another while still satisfying the trustless validation of transaction history. In this work, the key insight behind sharding the blockchain is to distribute transactions to blocks with respect to services they are used for.

This paper outlines *service-oriented sharding*, a technique for sharding blockchains that promises higher scalability and extensibility without modifying Bitcoin's trust model. It instantiates this technique in *Aspen*, a blockchain sharding protocol that expedites user access to relevant services, makes service integration and maintenance easier, and achieves better fairness while demanding only a fraction of resources from users.

2 Service-Oriented Sharding

The core idea behind service-oriented sharding is to partition a blockchain such that users can fully validate the correct functioning of their services (1) without relying on trusted entities and (2) while keeping track of only the subset of the blockchain relevant to their services. This technique shares the same network and trust model as Bitcoin and related cryptocurrencies. Service-oriented sharding is built around a multiblockchain structure, where multiple chains are rooted in the same genesis block and share common checkpoints (See Fig. 1). Building blocks comprising service-oriented sharding can be summarized as follows:

Channel. A chain in a blockchain built on a shared genesis block containing (1) all transactions of a specific service, and (2) common checkpoints involving transactions for the overall management of services. For instance, a domain name resolution service would use a dedicated channel to store custom transactions in the form of DNS resource records. Such transactions are kept separate from common checkpoints. Hence, services are loosely coupled and resilient to changes.

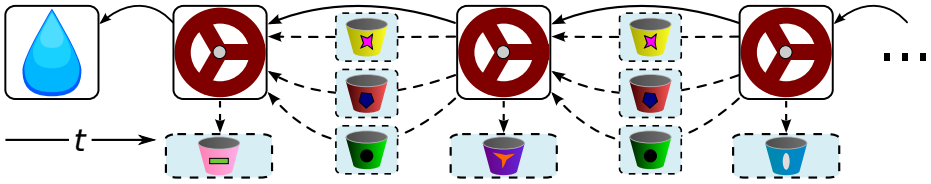


Fig. 1: Multiblockchain structure of service-oriented sharding. Each channel contains the same genesis block (drop) and checkpoints (valves), as well as the exclusive transactions of a specific service (buckets with the same symbol). Generating a checkpoint requires a proof of work. Miners distribute transactions to designated blocks (a subset of dashed rectangles) secured by checkpoints.

Service-oriented sharding handles requests associated with a certain service by annotating each channel and the corresponding transactions with the same unique identifier, called *service number*. Two special channels, *payment* and *registration*, are defined by the system and help bootstrap the network. The default service that enables users to exchange funds runs on the payment channel, and the registration channel is used to add or update services. Users store, process, and propagate transactions on channels only for the relevant services.

Protocol. A set of rules regarding services and their integration. A *service protocol* defines the validity of transactions in a given channel. It describes: (1) the syntax for each transaction type, (2) the relationship between transactions within a channel, (3) the size, frequency, and format constraints for blocks that keep transactions. The *integration protocol* specifies the security, incentive mechanism, valid service numbers, the genesis block, and the inter-channel communication process between the payment channel and the other channels.

Transaction. The smallest unit of data for adding content to a channel. Transactions are grouped into blocks and appended to each channel according to their service number. A block is valid if it (1) embodies valid transactions sharing the same service number and (2) complies with the integration protocol and the relevant service protocol.

Service Integration and Maintenance. The process of introducing services and updating the existing ones. Service-oriented sharding resolves this process completely on the blockchain in three phases. First, users propose protocols to introduce or update services by generating transactions for the registration channel. Each such transaction contains a set of service protocols with distinct service numbers. A service protocol is specified in a platform independent language such as WebAssembly [5] or Lua [3]. In the second phase, miners conduct an election to choose a registration channel transaction. This transaction specifies the protocols that miners are collectively willing to adopt. Miners indicate their choice using *ballots*. A ballot is a transaction that contains a reference to a particular transaction in the registration channel. Each ballot is part of a checkpoint that requires a proof of work to generate. This provides (1) representation proportional to mining power, and (2) protection against censorship of ballots. Finally, if a particular transaction is referred by more than a certain fraction τ of ballots, its protocols become active. An active service protocol determines the validity

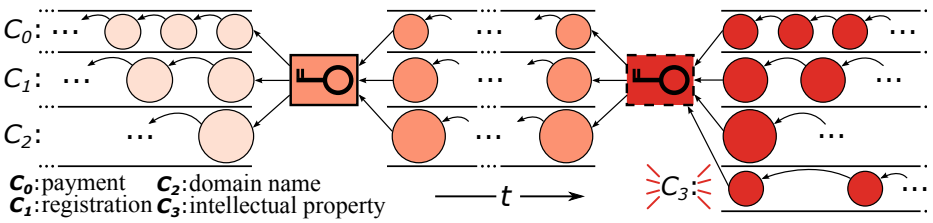


Fig. 2: Structure of the Aspen chain. Upon generating a key block shared by all channels, a miner serializes service-specific transactions only in the corresponding microblock (circles) chains. Shading indicates blocks generated by a specific miner. A bud (dashed key block) introduces the intellectual property service.

of new transactions added to the corresponding channel.

This process enables evolutionary refinement with the confidence of sustainability. Users are involved in the process through their proposals. The election mechanism ensures that the majority of the mining power intends to serialize transactions for the new or updated services.

3 Aspen

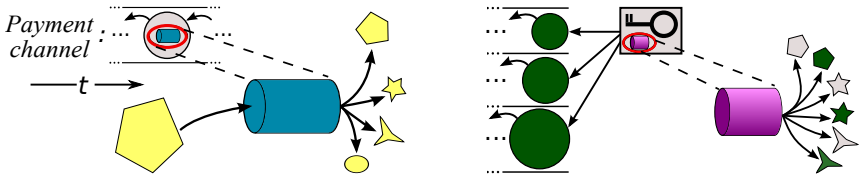
While service-oriented sharding can be built on any blockchain protocol [16, 17,21,27], we instantiate on Bitcoin-NG [17], a blockchain protocol that improves transaction throughput and consensus latency of Bitcoin under the same trust model. The protocol makes the following changes with service-oriented sharding:

Multiple Microblock Chains. Traditional blockchain protocols strive to agree on a single *main chain* in which all transactions are totally ordered. However, not all transactions are related or even need such an ordering. This leads to a seemingly irreconcilable tradeoff between the scalability of independent blockchains and the security of monolithic ones. The central idea behind Aspen is to resolve this conundrum by having a series of independent microblock chains conjoined at common key blocks. A channel represents the combination of the same genesis block, all key blocks, and the set of microblock chains containing custom transactions annotated with the same service number. Fig. 2 illustrates the structure.

Each channel maintains key blocks to enforce the integration protocol. To prevent bloating key blocks, Aspen (1) limits the number of channel references in a key block and (2) omits references to non-payment channels with no transactions on their latest microblock chain – i.e. *inert channels*. Note that users can fully validate an inert channel service using key blocks of the payment channel.

Extensibility. Aspen updates or introduces services at designated growth points, called *buds* (See Fig. 2). A bud is a key block at a protocol-defined height representing the number of key blocks from the genesis block. Aspen adopts proposals based on ballots in key blocks between the current and the preceding bud.

Flow of Funds Aspen enables users to detect double spends by making each fund spendable only in a specific channel. A special payment channel transaction, *funding pore*, enables users to lock funds to other channels. A funding pore annotates each output with the service number of an existing destination channel



(a) A funding pore. (b) A coinbase transaction.

Fig. 3: (a) A funding pore (cylinder) makes payment channel outputs (pentagons) spendable at specific channels. (b) Rewards are split between the current and the previous miner for each channel.

where it can be spent. Note that transfers across channels are allowed only in one way, from the payment channel to others. Fig. 3(a) illustrates a funding pore.

Alternatively, users can directly buy locked funds at the target channel to pay for the service running on the corresponding channel. The protocol enforces a high minimum fee for serializing funding pores to (1) discourage users from bloating the payment channel and (2) improve the fungibility of funds in non–payment channels. Contrary to Bitcoin's `OP_RETURN` transactions, this process does not yield any unspendable outputs.

Following sections detail the incentive and security mechanisms in Aspen.

3.1 Reward Structure

The process of keeping the complete blockchain, serializing transactions, and securing the system consumes miner resources. Aspen uses a Bitcoin-like cryptocurrency to encourage miners to continue facilitating this costly process. A coinbase transaction in a key block provides separate outputs to compensate the current and the previous miner for each service they provision. Each output indicates the source channel of rewards where funds can be spent (See Fig. 3(b)).

Generating Key Blocks. Miners receive a fixed subsidy for each key block they generate as an incentive for using their mining power to secure the blockchain and facilitating the voting process of service integration and maintenance.

Serializing Transactions. Each service protocol specifies the validity requirements for its transactions. The common property of all transactions is a fee that miners collect for adding them to the corresponding microblocks.

Extending the Longest Chains. As an incentive for the next miner to attach her key block to the latest microblock [17], Aspen distributes fees between the current miner and the next one for each microblock chain.

Extending Multiple Chains. Miners can spend transaction fees only in the corresponding channels that they were collected from. The high minimum fees for funding pores encourage users to purchase locked funds. Hence miners gain additional incentives to serialize non-payment channel transactions.

3.2 Security

The following properties are critical to the security of a blockchain protocol.

Authenticity. The property of having an indisputable origin. Transactions require a set of cryptographic signatures to prove the ownership of funds that are

used as fees. Hence, provided that it is infeasible to forge signatures, pseudonymous identities cannot deny committing transactions.

Irreversibility. The protection against overwriting or deleting transactions. Double spending is an instance of violating this property. Malicious miners may modify or remove a set of transactions from a blockchain by updating some common prefix with different blocks – i.e. forks. Aspen secures the blockchain against (1) key block forks by picking the chain containing the most proof of work with random tie-breaking and (2) microblock forks using poison transactions [17].

Censorship. The ability of miners to block submission or retrieval of transactions. A key block miner becomes eligible to update the blockchain for a discrete epoch. However, she may ignore certain transactions in particular channels due to benign failures or malicious behavior. The extend of such censorship is limited to the miner's epoch, which can be adjusted by changing the key block frequency.

An adversary can leave a victim unable to retrieve transactions by controlling all of its connections [19] or delaying the delivery of valid transactions to her [18]. Countermeasures to mitigate such attacks apply to this work, as well.

4 Related Work

Federated Chains. Sidechains [7] allow users to transfer funds across blockchains. However, this leads to fragmentation of the hash power. A compromised sidechain makes the main chain and the other sidechains vulnerable. Transfers across sidechains bloat the main chain. To guarantee that funds will not be pruned from the corresponding chains, such transfers incur high latencies. Drivechain [31] attempts to minimize the impact of sidechains on the main chain regarding the required knowledge and effort to prove validity of transfers. However, this approach does not address inherent limitations regarding the security of sidechains.

Multiple Services in Bitcoin's Blockchain. Bitcoin permits storage of arbitrary data on its blockchain using `OP_RETURN` transactions [10]. While there is no format requirement for the data, the size limit (currently 80 bytes) usually enforces users to store only a hash of their original content on the blockchain, which they externally validate [14,29]. This limitation imposes a critical tradeoff between data growth management and the diversity of services.

Users download and process the full history to validate the state of the existing blockchain protocols [16, 17, 27]. Using commodity hardware, this bootstrapping process takes many hours in Bitcoin [15]. Such protocols force users to handle the complexity of irrelevant services. Therefore, a monolithic history is not a viable option for scaling blockchains with multiple services.

Outsourcing the Security. Services with distinct blockchains attempt to improve their security with merged mining [9] and anchoring.

In merged mining, a blockchain with insufficient mining power accepts proof of work submissions from a designated parent chain. This approach raises three issues. First, if a miner is already part of the parent blockchain, she can use her mining power to attack the merged-mined blockchain at no cost. Second, a merged-mined blockchain becomes dependent on its parent chain, making it

fragile with respect to changes in the parent's security. Finally, it is non-trivial to maintain the miner coordination across blockchains. Ali et al. [6] show that even the largest merged-mined cryptocurrency, NameCoin [24], suffers from a single merged mining pool whose mining power exceeds the 51% threshold.

Anchoring relies on periodically submitting the cumulative hash of all data, such as the root of a Merkle tree, to a trusted publishing medium, such as the blockchain of Bitcoin. Anchoring bloats the external blockchain and becomes dependent on its security.

Sharding the Same Service. Elastico is a service-agnostic protocol for sharding blockchains [25]. This approach assigns miners to committees for serializing transactions using a classical Byzantine consensus protocol. As in anchoring, a final committee creates a cumulative digest based on all shards and broadcasts it to the network. However, to prevent double spends, Elastico requires splitting up the payment functionality into as many sub-services as the number of shards, which effectively means as many cryptocurrencies.

Treechains [32] is a sharding idea based on restructuring a blockchain into a tree of blocks, where each output has a dedicated branch to spend. However, this proposal is at an early stage with no prototype or a detailed technical analysis.

5 Conclusion

Service-oriented sharding provides a means for improving the scalability and extensibility of blockchains with multiple services. Aspen, the instantiation of this technique, reduces the resource requirements and the bootstrapping time to participate in the system. It provides trustless validation while preserving the same network and trust model as Bitcoin. Finally, it avoids fragmentation of the mining power that secures the blockchain.

Acknowledgements

The authors thank Ittay Eyal, and the anonymous reviewers for their comments and suggestions. This work was supported in part by NSF grants CNS-1422544, CNS-1561209, CNS-1518779, NIST, a Google Faculty Research Award, IC3 sponsorship from Chain, IBM, and Intel, as well as gifts from Infosys and Facebook.

References

1. Benben. `http://benben.com.gh/`, retrieved Oct. 2016.
2. Follow my vote. `https://followmyvote.com/`, retrieved Oct. 2016.
3. Lua. `http://www.lua.org/`, retrieved Nov. 2016.
4. `OP_RETURN` stats. `http://opreturn.org/`, retrieved Nov. 2016.
5. WebAssembly. `http://webassembly.org/`, retrieved Nov. 2016.
6. M. Ali, J. Nelson, R. Shea, and M. J. Freedman. Blockstack: A global naming and storage system secured by blockchains. In *USENIX Annual Technical Conference*, Denver, CO, USA, 2016.
7. A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille. Enabling blockchain innovations with pegged sidechains. `https://blockstream.com/sidechains.pdf`, 2014.

8. BigchainDB GmbH. Ascribe. https://www.ascribe.io/, retrieved Oct. 2016.
9. Bitcoin Community. Merged mining specification. https://en.bitcoin.it/wiki/Merged_mining_specification, retrieved Oct. 2016.
10. Bitcoin community. OP_RETURN. https://en.bitcoin.it/wiki/OP_RETURN, retrieved Oct. 2016.
11. R. G. Brown, J. Carlyle, I. Grigg, and M. Hearn. Corda: An introduction. http://r3cev.com/s/corda-introductory-whitepaper-final.pdf, Aug. 2016.
12. Chain Inc. Chain open standard: A secure blockchain protocol for high-scale financial networks. http://chain.com/os/, retrieved Sep. 2016.
13. CoinDesk. State of blockchain q1 2016: Blockchain funding overtakes Bitcoin. http://coindesk.com/state-of-blockchain-q1-2016/, retrieved Oct. 2016.
14. Colu. Colored Coins. http://coloredcoins.org/, retrieved Sep. 2016.
15. K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer. On scaling decentralized blockchains (a position paper). In *3rd Workshop on Bitcoin and Blockchain Research*, Barbados, 2016.
16. Ethereum Foundation. A next generation smart contract and decentralized application platform. https://github.com/ethereum/wiki/wiki/White-Paper, retrieved Oct. 2016.
17. I. Eyal, A. E. Gencer, E. G. Sirer, and R. van Renesse. Bitcoin-NG: A scalable blockchain protocol. In *13th USENIX Symposium on Networked Systems Design and Implementation*, pages 45–59, Santa Clara, CA, USA, 2016.
18. A. Gervais, H. Ritzdorf, G. O. Karame, and S. Capkun. Tampering with the delivery of blocks and transactions in Bitcoin. In *Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 692–705, Denver, CO, USA, 2015.
19. E. Heilman, A. Kendler, A. Zohar, and S. Goldberg. Eclipse attacks on Bitcoin's peer-to-peer network. In *24th USENIX Security Symposium*, pages 129–144, Washington, D.C., USA, 2015.
20. IBM Corporation. IBM Blockchain on Bluemix. http://www.ibm.com/blockchain/bluemix.html, retrieved Oct. 2016.
21. E. Kokoris-Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford. Enhancing Bitcoin security and performance with strong consistency via collective signing. *arXiv preprint arXiv:1602.06997*, 2016.
22. Ledger Assets Pty Ltd. Uproov. https://uproov.com/, retrieved Sep. 2016.
23. Linux Foundation. Hyperledger. https://hyperledger.org/, retrieved Sep. 2016.
24. A. Loibl. Namecoin. namecoin.info, 2014.
25. L. Luu, V. Narayanan, K. Baweja, C. Zheng, S. Gilbert, and P. Saxena. A secure sharding protocol for open blockchains. In *Conference on Computer and Communications Security*, Vienna, Austria, 2016. ACM.
26. Microsoft. Blockchain-as-a-Service. https://azure.microsoft.com/en-us/solutions/blockchain/, retrieved Oct. 2016.
27. S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.
28. OMI. Open music initiative. http://open-music.org/, retrieved Oct. 2016.
29. Omni Team. Omni layer. http://www.omnilayer.org/, retrieved Oct. 2016.
30. L. Shin. Republic of Georgia to pilot land titling on blockchain. *Forbes*, 21 April 2016.
31. P. Sztorc. Drivechain. http://www.truthcoin.info/blog/drivechain/, 2015.
32. P. Todd. [bitcoin-development] Tree-chains preliminary summary. https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2014-March/004797.html, 2014.