ON SCALABILITY OF BLOCKCHAIN TECHNOLOGIES

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Adem Efe Gencer

August 2017

ON SCALABILITY OF BLOCKCHAIN TECHNOLOGIES

Adem Efe Gencer, Ph.D.

Cornell University 2017

In this dissertation, we explore how to improve scalability of blockchains while maintaining their fundamental premise of decentralization. Scalable blockchains are capable of delivering a target throughput and latency in the presence of increasing workload. To this end, first we present Bitcoin-NG, a new blockchain protocol designed to provide scale for services involving frequent, high-volume interactions. This Byzantine fault tolerant blockchain protocol is robust to extreme churn and shares the same trust model as Bitcoin. We experimentally demonstrate that Bitcoin-NG scales optimally, with bandwidth limited only by the capacity of the individual nodes and latency limited only by the propagation time of the network. Then, we examine the scalability challenges arising from proliferation of blockchain services. In particular, we observe that due to inherently single-service oriented blockchain protocols, services can bloat the existing blockchains, fail to provide sufficient security, or completely forego the property of trustless auditability. We introduce Aspen, a sharded blockchain protocol that securely scales with increasing number of services. Aspen enables service integration without compromising security — leveraging the trust assumptions — or flooding users with irrelevant messages. Finally, we provide the means to assess the viability of different scaling solutions. We develop and utilize custom metrics for evaluating performance and security of blockchain protocols. Moreover, we design tools and techniques for measuring decentralization in operational blockchain systems, demonstrating their use in a comparative study of decentralization in Bitcoin and Ethereum.

BIOGRAPHICAL SKETCH

Adem Efe Gencer was born in Istanbul, Turkey. He received his bachelor's degree in Computer Engineering from Boğaziçi University in 2012. During his years at Boğaziçi, he worked on a wide spectrum of topics, from parallel computing to games with a purpose, and published three papers before graduating. In 2010, he spent a semester at the Technical University of Denmark via the Erasmus Exchange Program. This visit convinced him that doing research in an internationally recognized institution is what he wants to do. In 2012, he started his PhD program in Computer Science at Cornell University to spend five beautiful years under the supervision of Emin Gün Sirer and Robbert van Renesse. In 2015 and 2016, Efe spent two summers at LinkedIn, working with the Kafka Infrastructure Team.

In memory of my mother, *Muazzez*.

ACKNOWLEDGMENTS

First and foremost, I am deeply grateful to my advisors Emin Gün Sirer and Robbert van Renesse, whose guidance and constant support have made this dissertation possible. They have taught me how to ask research questions, develop techniques to answer them, and convey my scientific findings. I would like to thank Gün and Robbert for their invaluable advice, endless patience in showing me how to build systems, and continued confidence in my abilities. Their quest for excellence in research has been a source of inspiration for my academic endeavours.

I would like to thank the other members of my committee for their support and valuable feedback. I am grateful to Robert D. Kleinberg for helping me advance my knowledge of theoretical computer science. I would like to thank Levent V. Orman for sharing his extensive expertise in electronic commerce.

I am especially thankful to Ittay Eyal for being a great mentor, collaborator, and friend. His insights on scaling principles and the demonstration of his solid knowledge in systems programming have been vital to this dissertation. I want to thank him for always being available to help me through my Ph.D. career.

I am grateful to many friends at Cornell University. Theodoros Gkountouvas, Elisavet Kozyri, Anıl and Nurten Aktürk, Stavros Nikolaou, Melik Türker, Jaeyong Sung, Deniz Altınbüken, Weijia Song, Moontae Lee, Deniz Günceler, Ayush Dubey, Han Wang, Soumya Basu, Kai Mast, Hussam Abu-Libdeh, and too many others to list here. Thanks to them, my life in snowy Ithaca has been full of warm memories.

I would not have finished this journey without the love and support of my family. I am very fortunate to have a father, who had raised me from an early age to be hardworking and self-reliant. My beloved mother taught me to reach

for the stars and pursue my goals with tenacity. My sisters, Özlem and Bilge, have always been by my side, listening to my incessant rants and giving me constant encouragement. My brother-in-law, Sinan, has always shown a keen interest in my research, our conversations have helped me to broaden my perspective and knowledge.

Finally, I want to thank Gözde for being an integral part of my life, believing in me when I doubted myself, and cheering me up when I needed it. I am grateful for her contagious positive nature and remarkable patience with me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Blockchain technology has recently emerged with a promise to streamline interactions in a wide range of settings. Exploring this potential has had a broad impact on different interest groups. Less than a decade after their inception, blockchains have attracted hundreds of companies from different circles and catalyzed the formation of many consortia [43], raised close to $1.8B in venture capital [42], and created cryptocurrency markets with total capitalization close to $100B [45]. Similarly, governments have been actively examining and encouraging the use of blockchains in the public sector [94, 57, 161, 93]. Academic interest towards the study of blockchains has steadily grown [125, 39].

The key driving force behind the interest in blockchains is decentralization of power among entities with minimal trust relationships. Traditional, non-blockchain-based systems tend to incorporate control in trusted third parties, such as international organizations, governments, and authorized middlemen – e.g. notaries and certificate authorities. Contrary to such conventional design practices, blockchain-based systems inherently provide independently verifiable guarantees, obviating the need to rely on centralized authorities. The lack of a central authority enables blockchain services to provide stronger safety and liveness properties for distributed systems. Because there is no central authority in control, corrupting a blockchain or hampering the propagation of its contents are possible only through collusion among powerful entities. Consequently, blockchain services can achieve higher resistance against tampering and censorship, even in the presence of malicious insiders.

Growing adoption of blockchains for services that are traditionally provided by nation states or large corporations, such as money and digital asset manage-

ment, has raised new challenges. In particular, a critical task in this context is to provide the level of scalability required to achieve a target throughput and latency in the presence of increasing workload, without compromising the decentralized nature of blockchains. But as of today, the most prominent blockchain instantiation to date, Bitcoin [131], achieves only 3.3 transactions per second, incurs 10-minute expected latencies for transaction serialization with longer recommended waiting times for deciding on transaction status, and consumes over 50 GB of additional disk space per year. There is a big gap between the current and the desired level of scalability to enable novel applications or to compete with centralized services.

If blockchains are indeed going to be a globally disruptive technology, they have to scale. This dissertation describes our work towards improving the scalability of blockchain technologies. To this end, we first devise a new protocol for scaling on-chain. This protocol preserves all of the standard assumptions of Bitcoin-like blockchains, and allows them to sidestep their fundamental scalability bottleneck: block propagation. Then we examine a more aggressive scenario, where the same blockchain hosts multiple different services. We propose an approach that scales with differing asset types without commingling them on a single main chain. Finally, we present a comparative empirical study of the leading blockchain-based cryptocurrencies with the largest market capitalization and user base, Bitcoin and Ethereum [66].

1.1 The Rise of Blockchains

To illustrate the need for scalability, this section presents a representative list of compelling blockchain services that have been proposed.

Digital Money. Blockchain technology emerged as the underlying infrastructure of a decentralized global currency, called Bitcoin. Following Bitcoin's lead, hundreds of alternative blockchain-based currencies have been created [167], most sharing the same consensus mechanism as Bitcoin, differing only in protocol parameters.

Blockchain-based currencies have enabled cheap international remittance and pseudonymous online payments without middlemen. These currencies are, first and foremost, payment systems that act as a *medium of exchange*. They have gained widespread adoption in underground economies [124], and received recognition as legal tenders from governments [101, 172]. The value brought on by their role as a medium of exchange has allowed them to serve as a *store of value* for others. However, most such currencies are still in their infancy as a *unit of account*. The current practice involves conversion from some other unit of account, such as USD, to a blockchain-based currency at the time of purchase. As more of the economy moves into use such currencies, goods and services could possibly be priced in them.

Smart Contracts. A smart contract is a program that enables users to implement their own autonomous execution logic on blockchains [66, 103, 58]. In particular, they enable pseudonymous parties to establish complex agreements that are enforced by a network of independent entities. Smart contract applications include prediction markets [8, 88], supply chain tracking [152], flight insurance [175], and legally binding agreements [41].

Micropayments. Small online payments, known as *micropayments*, are not feasible in traditional payment systems, mainly due to high transaction fees [138]. Off-chain transaction protocols aim to minimize this cost. Proposed protocols

establish direct payment channels between users, and use blockchains for conflict resolution and settlement [91, 55, 141]. Recent work has leveraged trusted execution environments to secure such payment channels [112].

Healthcare. Ensuring confidentiality, integrity and availability of electronic health records (EHR) have been critical for the healthcare industry [15, 79]. The centralization of medical records raises concerns about security and interoperability of patient data. Proposed services aim to integrate this information on blockchains to reduce healthcare costs, enhance data privacy, improve the quality of care, and provide tamper-resistant record keeping [173, 90, 63, 123].

Digital Asset Management. Digital assets are intangible properties typically representing tokens of value such as stocks, bonds, securities, and custom currencies. Services for managing such assets can be either layered on top of existing open blockchains [46, 50], or integrated to custom blockchains containing only the approved participants with role-based permissions [36]. Such services promise to improve the integrity of bookkeeping.

Intellectual Property Tracking. Intellectual property (IP) represents creative works based on original thought such as photos, music, or pieces of literature. IP owners possess rights to receive financial return from their creation. Blockchain services [18, 109, 134] can issue certificates of ownership for digital works and facilitate the compensation of IP owners such as artists, songwriters, and publishers.

Land Record and Deed Maintenance. People in many parts of the world do not have legal title to their assets. This makes the sale or purchase of such assets challenging. Blockchain-based services promise to store land and deed records

to create an adequate knowledge base, facilitate electronic land transactions, and reduce property rights registration costs [150, 16].

Global Name Registration. Naming systems, such as domain name resolution services and public key infrastructures, associate human-readable names with owner-specified data. Such systems are traditionally centralized, raising concerns regarding censorship and seizure of names. Decentralized blockchain naming services [115, 2, 26] can address these concerns.

File Storage. Centralized cloud storage services require implicit trust from users regarding the availability, privacy, and integrity of their data. To alleviate these concerns, storage-oriented blockchain services aim to distribute encrypted shards of user data among multiple administrative domains [169, 80].

Supply Chain Management. Supply chain management refers to tracking the movement of goods and information. But, this process often involves a lot of paperwork that is open to tampering [142]. Detailed record keeping on blockchain promises to reduce insurance and auditing costs, enable better conformity to company standards regarding the source of goods and labor, and expedite identification of missing or damaged shipments. Ongoing work explores the use of blockchains for cross-industry supply chains [56].

Online Voting. Secret ballots typically provide confidentiality, but fail to assure integrity. Blockchain-based voting services enable voters to confirm that their votes have been counted while preserving their privacy [81, 122]. Moreover, a convenient voting process potentially improves voter turnout.

All these services are incredibly demanding in terms of the frequency and

volume of interactions. Full appreciation of these services requires significant scalability. This dissertation represents a step towards achieving this goal.

1.2 Goals and Challenges in Scaling Blockchains

Blockchain scalability corresponds to the capability of achieving a target throughput and latency in the presence of increasing workload, without undermining the central tenet of decentralization that makes blockchains compelling at the first place. This section presents the goals and challenges in scaling blockchain technology.

Scaling blockchains while retaining their decentralization is critical to realize their potential. Scalable blockchains promise to (1) enable services that achieve comparable performance as mainstream technologies, (2) support novel applications, such as e-commerce based on machine-to-machine microtransactions, and (3) support a large number of users, including citizens of nation states or customers of large corporations.

1.2.1 Goals

Unfortunately, existing blockchain applications are far from satisfying the scalability requirements to compete with centralized systems. Current mainstream payment processors, such as Visa, supports orders of magnitude higher peak transaction throughput and incurs much lower latency compared to the operational Bitcoin platform [52].

The first crucial metric for scaling blockchains is the maximum throughput. This metric represents the upper bound at which a blockchain can process requests. In Bitcoin-derived blockchain protocols, this bound corresponds

to transaction confirmation rate. A transaction cannot be confirmed until it resides in a block. Hence, this rate is primarily constrained by the maximum size and frequency of blocks that bundle transactions. A high maximum throughput enables a blockchain service to perform more work in a given time.

The second metric for scaling blockchains is the consensus latency. This metric indicates the delay for transaction confirmation. A low consensus latency provides high confidence that a transition has occurred; hence, it prevents long waiting times to ensure steady blockchain state.

1.2.2 Challenges

Establishing a certain level of decentralization is the core premise of blockchains. The maintenance of this premise depends on keeping resource use in check, because different administrative entities can benefit from blockchain services only if they can satisfy the corresponding requirements. In particular, while increasing the maximum size and frequency of blocks would improve the peak throughput and latency of a service, suboptimal reparametrization leads to concentration of power under few participants with greater resources. It should be noted that, even without any reparametrization, resource consumption of such services continuously grows due to the append-only nature of blockchains.

Resource management primarily concerns the following functions:

Storing. The preservation of the blockchain state to ensure its availability. In existing blockchain protocols, state typically represents the entire history. Hence, user nodes require adequate storage space, as well as disk throughput and latency.

Processing. The execution of operations for maintenance of blockchain services. These CPU-intensive operations include the validation of transactions, invocation of smart contracts, and extension of the blockchain.

Propagation. The transmission of blockchain components, such as transactions and blocks, to other participants. Propagation performance depends on network latency and transmission capacity – i.e. provisioned bandwidth.

1.3 Contributions

This dissertation broadly explores how to improve scalability of blockchains while preserving their decentralized nature.

Bitcoin-derived blockchain protocols exhibit an inherent tradeoff between throughput and consensus latency. We outline a scalable blockchain protocol that preserves the trust assumptions of Bitcoin, yet shifts this tradeoff to achieve a significantly higher throughput with a fraction of Bitcoin's latency. To illustrate the scalability improvements of this protocol, a comparative assessment is critical. But quantitative evaluation of the security and performance of such consensus protocols is challenging. To achieve this goal, we introduce specialized metrics for systematic evaluation of blockchain protocols. These metrics enable assessing the performance and security of protocols with regard to their consensus latency, goodput, robustness against centralization, and rollback-resistance. We demonstrate the scalability and robustness of our proposed protocol on an evaluation testbed at 15% the size of the operational Bitcoin network at the time of our setup. We built and calibrated this testbed using our measurements from the actual network.

The proliferation of blockchain services, either on common or separate

blockchains, presents another scalability challenge. We observe that services on dedicated blockchains suffer from lack of mining power to secure their infrastructures; hence, this makes them open to all kinds of attacks. In contrast, combining all services in a single blockchain bloats the system and increases resource requirements. Users of such systems store, process, and propagate data of all blockchain services, regardless of the services they are interested in. We propose a novel service-oriented technique for sharding blockchains that can reduce resource requirements and bootstrap time, secure the entire blockchain with an unfragmented mining power, and provide network and trust construction similar to Bitcoin. This technique also illustrates a principled approach to facilitate the process of introducing services and updating the existing ones. We instantiate this technique in a blockchain protocol that accelerates access to relevant user services.

While blockchain-based systems derive their value from being decentralized, there have been few quantitative studies on the extent to which existing systems achieve this goal in practice. We present a global-scale empirical study of the decentralization of blockchain-based currencies with the largest market share and user community, Bitcoin and Ethereum. We introduce tools and techniques for measuring blockchain-based cryptocurrency networks through retrieval of unique data. Using these tools and techniques, we deploy a measurement system that has been collecting data over a 12 month time frame across five continents. We observe that while the Bitcoin platform offers more resource capacity, greater censorship resistance, less overhead under high churn, and more diverse mining power distribution, Ethereum provides better fairness with regard to miner revenue and less divergence in network link latency. Based on our observations, we provide concrete suggestions for improving both systems.

1.4 Organization

The rest of the dissertation is organized as follows. Chapter 2 describes the background, including how blockchain protocols work. Chapter 3 presents Bitcoin-NG, a blockchain protocol that achieves high transaction throughput with low consensus latency. Chapter 4 specifies a secure sharding technique for blockchains, and shows application of it on Aspen, a blockchain protocol that scales with increasing number of services. Chapter 5 presents a comparative and longitudinal assessment of decentralization in the two leading blockchain-based platforms, Bitcoin and Ethereum. Chapter 6 surveys the related work, and Chapter 7 concludes.

CHAPTER 2

BACKGROUND

A blockchain is a global append-only log that can serve as a shared database with an immutable record of all transactions. A consensus mechanism enables this log to be distributed securely, and cryptographic techniques ensure that transactions can be submitted solely by those in possession of the appropriate keys.

This chapter presents the relevant background on blockchain technology, including how the corresponding protocols work. In particular, it describes protocols of the two leading blockchain-based cryptocurrencies with the largest market capitalization and user base, Bitcoin [131] and Ethereum [66].

2.1 Blockchain Protocols

Blockchain protocols have been used to achieve consensus in large scale peer-to-peer net- works with open participation. Because anyone can join the network, the underlying protocols need to be robust against Byzantine behaviors by participants. To achieve this, they specify a set of communication rules that enforce the integration and validity of building blocks comprising the underlying blockchain. In particular, they define the size, frequency, and format constraints for blocks and transactions with which participants may extend the blockchain. Blockchains record transactions in units of blocks. Each block includes a unique ID, and IDs of the preceding valid blocks. The determination of such preceding blocks varies based on the particular blockchain protocol.

To account for the lack of a central authority to oversee the blockchain extension process, protocols depend on a distributed consensus mechanism that ensures global agreement on the state. The majority of existing blockchain pro-

tocols, including Bitcoin and Ethereum, currently employ a *proof of work* mechanism to achieve this goal [86].

In proof of work, generating a block requires a participant, called a *miner*, to conduct some computationally intensive task to produce a result that is easy to verify by others – i.e. a solution to a cryptopuzzle. This process is called *mining*, and, by slight abuse of terminology, we refer to the creation of blocks as *block mining*. Miners are incentivized to invest their resources to generate blocks. In particular, the mining process enables participants to extend the blockchain proportional to their computational power. This provides resistance against Sybil attacks [60]. Ongoing research explores more efficient and environmentally friendly alternatives [144, 17, 7, 35, 96].

2.1.1 Fork

Achieving Internet-scale consensus is challenging. While the first block of the blockchain, called *the genesis block*, is hard coded in the protocol specification, the others are not. This enables any peer with a valid block to extend the blockchain by simply publishing it over an overlay network to all other peers. Hence, during the execution of the protocol, the local state of peers might differ. Such disagreements in the global blockchain state are called *fork*s. A fork indicates that participants have extended some common prefix of the blockchain with different blocks, leading to multiple *branches*. Other participants may subsequently add new valid blocks to any of these branches. In protocols that adopt proof of work mechanism, when a miner tries to add a new block after an existing block, we say she *mines on* the existing block. If this block is a leaf of a branch, we say she mines on the branch.

To resolve forks, a blockchain protocol prescribes the branch that partici-

pants should pick as the extension of the main chain – e.g. the one with the most mining power to generate. Branches and blocks outside the main chain are called *pruned* – some refer to such branches or blocks as *orphans*, though this usage is technically incorrect, because they have a parent. Transactions in pruned blocks are considered as invalid until a participant serializes them in a valid block within the main chain. A transaction can be placed in the main chain at any later time, unless a contradicting one, e.g. another transaction spending the same outputs, was placed there in the meantime. The formation of forks is undesirable, as they indicate that there is no globally-agreed blockchain state.

2.1.2 Nakamoto Consensus

Nakamoto consensus [20, 22, 23, 131, 28] is a decentralized consensus protocol that is implicitly defined and implemented in Bitcoin. This protocol, named after its pseudonymous inventor, enables participants to reach an agreement on the global state of a blockchain. This section provides a specification of this core protocol.

Our definition of Nakamoto Consensus is similar to that of Garay et al. [82]. The system consists of a set of nodes $\mathcal{N}$ that are connected by a reliable peer-to-peer network. The nodes are to implement a replicated state machine (RSM) [107, 149]. The properties of Nakamoto consensus are as follows:

Termination. There exists a time difference function $\Delta(\cdot)$ such that, given a time t and a value $0 < \varepsilon < 1$, the probability is smaller than ε that at times $t', t'' > t + \Delta(\varepsilon)$ a node returns two different states for the machine at time t.

Agreement. There exists a time difference function $\Delta(\cdot)$ such that, given a $0 <$

$\varepsilon < 1$, the probability that at time t two nodes return different states for $t - \Delta(\varepsilon)$ is smaller than ε.

Validity. At a given time t, a subset of nodes $B(t) \subset \mathcal{N}$ are Byzantine and behave arbitrarily, controlled by a single adversary. If the fraction of mining power of such Byzantine nodes is bounded by f, i.e., $\forall t : \frac{\sum_{b \in B(t)} m(b)}{\sum_{n \in \mathcal{N}} m(n)} < f$, then the average fraction of state machine transitions that are not inputs of honest nodes is smaller than f.

2.2 Bitcoin and Ethereum

Bitcoin and Ethereum are decentralized cryptocurrencies based on blockchain protocols that employ Nakamoto consensus to regulate transaction serialization in their platforms. In each of these cryptocurrencies, a replicated state machine maintains the balances of the different users, and its transitions are transactions that move funds among them. These state machines are managed by miners.

While the two systems are architecturally very similar, they differ substantially in terms of their abstractions. The rest of this section presents the relevant details of the Bitcoin and Ethereum blockchain protocols.

2.2.1 The Bitcoin Protocol

Bitcoin protocol uses transactions primarily to represent exchange of funds between users. These funds are specified in terms of the Bitcoin currency. A typical Bitcoin transaction contains a set of inputs, outputs, and scripts to specify the sources, destinations, and validity conditions of fund transfers, respectively.

Each user commands *addresses*, and sends Bitcoins by forming a transaction from her address to another's address and propagating it to the nodes. More

explicitly, a transaction is from the output of a previous transaction to a specific address. Bitcoin is based on a UTXO (unspent transaction outputs) model. This model considers an output as *spent* if a transaction input in the blockchain refers to it; hence, the total unspent output values of a peer represents her funds. A user owns x Bitcoins at time t if the aggregate of unspent outputs to its address is x. Transactions are protected with cryptographic techniques that ensure only the rightful owner of a Bitcoin address can transfer funds from it. Miners accept transactions and commit them into the blockchain only if their sources have not been spent, thereby preventing users from double-spending their funds.

Transactions reside in blocks of a global main chain, which is identified as the one with the most accumulated proof of work. Miners extend this chain by building on each others' blocks; thus, they can benefit from being physically close to each other. The wire protocol is based on flooding block and transaction announcements, followed by pull requests from peers that are missing them. This avoids sending the same information redundantly to peers across multiple overlay hops. Each transaction is validated at least twice – i.e. upon receiving them and within discovered blocks.

A valid block contains (1) a solution to a cryptopuzzle involving the hash of the previous block, (2) the hash (specifically, the Merkle root) of the transactions in the current block, which have to be valid, and (3) a special transaction, called the *coinbase*, crediting the miner with the reward for solving the cryptopuzzle. The specific cryptopuzzle is a double-hash of the block header whose result has to be smaller than a set value. The *problem difficulty*, set by this value, is dynamically adjusted such that blocks are generated at an average rate hard coded into the protocol.

The fundamental protocol parameters are block size and frequency. The pro-

tocol targets a 10 minute block interval with a maximum size limit of 1 MB. At the time of our measurements, the latest 100 blocks were generated with a 0.99 MB median block size and a 9.8 minute average interval.

When a miner creates a block, she is compensated for her efforts with Bitcoins. This compensation includes a per-transaction fee paid by the users whose transactions are included, as well as an amount of new Bitcoins that did not exist before.

Block dissemination over the Bitcoin overlay network takes seconds, whereas the average mining interval takes minutes. Therefore, accidental bifurcation occurs on average about once every 60 blocks [54]. In case of such forks, the criterion for the winning chain is to be the *heaviest one*, that is, the one that required (in expectancy) the most mining power to generate. All miners add blocks to the heaviest chain of which they know, with random tie-breaking. We note that choosing a longest branch at random is suggested by Eyal and Sirer [77]. The operational client currently chooses the first branch it has heard of, making it more vulnerable in the general case. The heaviest chain a node knows is the serialization of RSM inputs it knows, and hence describes the RSM's state.

2.2.2 The Ethereum Protocol

Ethereum protocol focuses on providing a platform to facilitate the process of building decentralized applications on its blockchain. In this platform, the blockchain stores *smart contracts* written in an integrated programming language that enable users to implement their own autonomous execution logic. To achieve this goal, it adopts a design inspired by both Nakamoto consensus and the GHOST protocol [154].

The key insight in the GHOST protocol is to adopt a chain selection rule to harness the residual mining power over pruned blocks for improved security. Ethereum protocol relies on the chain selection rule of Nakamoto consensus [86], but it benefits from the pruned blocks in different ways [171]. The protocol includes such blocks, called *uncle*s in Ethereum nomenclature, in its blockchain and rewards the corresponding miners with the intent of making the system more decentralized. While uncles do not directly contribute to the security of the chain, rewarding their miners conceivably prevents well-connected mining entities from uniting under a single administrative domain by incentivizing independent block generation. In particular, uncle rewards enable Ethereum to afford more ambitious design choices – e.g. high block frequency. Finally, inclusion of uncles in the blockchain provides better transparency for tracking miner activities; hence, it facilitates the detection of fluctuations in mining power utilization.

Ethereum has a block interval between 10 to 20 seconds [86]. The block size is indirectly determined by an execution and storage fee, called *gas*. Each user operation, such as adding two numbers or storing a byte, has an associated cost represented by a fixed amount of gas. With every transaction, the sender specifies a gas price she is willing to pay in terms of the native token of Ethereum, called *Ether*. Hence, the fee for a transaction is calculating by multiplying its total gas consumption by this user-specified exchange rate. Consequently, Ethereum miners are compensated for their efforts with the sum of all such transaction fees in their blocks. Each such block specifies a limit for the maximum amount of gas that its transactions can cumulatively consume. This limit helps keeping the block size in check, providing protection against forks due to increased block propagation and processing times. Contrary to Bitcoin's

maximum block size, miners dynamically adjust this limit for each block. At the time of our measurements, the latest 100 blocks were generated with a 2.9 KB median block size and a 16.3 second average interval.

CHAPTER 3

BITCOIN-NG: A SCALABLE BLOCKCHAIN PROTOCOL

Despite their potential, blockchain protocols face a significant scalability barrier [153, 111, 55, 12]. The maximum rate at which these systems can process transactions is capped by the choice of two parameters: block size and block interval. Increasing block size improves throughput, but the resulting bigger blocks take longer to propagate in the network. Reducing the block interval reduces latency, but leads to instability where the system is in disagreement and the blockchain is subject to reorganization. Bitcoin currently targets a conservative 10 minutes between blocks, yielding 10-minute expected latencies for transactions to be encoded in the blockchain. The block size is currently set at 1MB, yielding only 1 to 3.5 transactions per second for Bitcoin for typical transaction sizes. Proposals for increasing the block size are the topic of heated debate within the Bitcoin community [140].

In this chapter, we present Bitcoin-NG, a scalable blockchain protocol, based on the same trust model as Bitcoin. Bitcoin-NG's latency is limited only by the propagation delay of the network, and its bandwidth is limited only by the processing capacity of the individual nodes. Bitcoin-NG achieves this performance improvement by decoupling Bitcoin's blockchain operation into two planes: *leader election* and *transaction serialization*. It divides time into epochs, where each epoch has a single leader. As in Bitcoin, leader election is performed randomly and infrequently. Once a leader is chosen, it is entitled to serialize transactions unilaterally until a new leader is chosen, marking the end of the former's epoch.

While this approach is a significant departure from Bitcoin's operation, Bitcoin-NG maintains Bitcoin's security properties. Implicitly, leader election is already taking place in Bitcoin. But in Bitcoin, the leader is in charge of serial-

izing history, making the entire duration of time between leader elections a long system freeze. In contrast, leader election in Bitcoin-NG is forward-looking, and ensures that the system is able to continually process transactions.

Evaluating the performance and functionality of new consensus protocols is a challenging task. To help perform this quantitatively and provide a foundation for the comparison of alternative consensus protocols, we introduce several metrics to evaluate implementations of Nakamoto consensus. These metrics capture performance metrics such as protocol goodput and latency, as well as various aspects of its security, including its ability to maintain consensus and resist centralization.

We evaluate the performance of Bitcoin-NG on a large emulation testbed consisting of 1000 nodes, amounting to over 15% of the current operational Bitcoin network [129]. This testbed enables us to run unchanged clients, using realistic Internet latencies. We compare Bitcoin-NG with the original Bitcoin client, and demonstrate the critical tradeoffs inherent in the original Bitcoin protocol. Controlling for network bandwidth, reducing Bitcoin's latency by decreasing the block interval and improving its throughput by increasing the block size both yield adverse effects. In particular, fairness suffers, giving large miners an advantage over small miners. This anomaly leads to centralization, where the mining power tends to be concentrated under a single controller, breaking the basic premise of the decentralized cryptocurrency vision. Additionally, mining power is lost, making the system more vulnerable to attacks. In contrast, Bitcoin-NG improves latency and throughput to the maximum allowed by network conditions and node processing limits, while avoiding the fairness and mining power utilization problems.

In summary, this chapter makes three contributions. First, it outlines the

Bitcoin-NG scalable blockchain protocol, which achieves significantly higher throughput and lower latency than Bitcoin while maintaining the Bitcoin trust assumptions. Second, it introduces quantitative metrics for evaluating Nakamoto consensus protocols. These metrics are designed to ground the ongoing discussion over parameter selection in Bitcoin-derived currency. Finally, it quantifies, through large-scale experiments, Bitcoin-NG's robustness and scalability.

3.1 Model and Goal

As in Bitcoin [131] and enhancements thereof [66, 153, 111], the goal of Bitcoin-NG is to implement a replicated state machine (RSM) in an open system. The exact assumptions and guarantees are explored in different works [28, 128, 82]. Our model is similar to those of Aspnes et al. [5] and Garay et al. [82]. These are different from the model and goal of classical Byzantine fault tolerant RSMs. The latter, by and large, (1) assume static or slow-to-change membership, allowing for quorum systems and reconfigurations thereof, and (2) do not guarantee fairness of representation of honest parties in the state machine transitions.

The system is comprised of a set of nodes $\mathcal{N}$ connected by a reliable peer-to-peer network. Each node can poll a random oracle [14] as a random bit source. Nodes can generate key-pairs, but there is no trusted public key infrastructure.

The system employs a cryptopuzzle system, defined by a cryptographic hash function H. The solution to a puzzle defined by the string y is a string x such that $H(y|x)$ — the hash of the concatenation of the two — is smaller than some target. Each node i has a limited amount of compute power, called *mining power*, measured by the number of potential puzzle solutions it can try per second. A solution to a puzzle constitutes a *proof of work*, as it statistically indicates the

amount of work a node had to perform in order to find it.

At any time t, a subset of nodes $B(t) \subset \mathcal{N}$ are Byzantine and behave arbitrarily, controlled by a single adversary. The other nodes are *honest* — they abide by the protocol. The mining power of each node i is $m(i)$. The mining power of the Byzantine nodes is less than $1/4$ of the total compute power at any given time:

$$\forall t : \sum_{b \in B(t)} m(b) < \frac{1}{4} \sum_{n \in \mathcal{N}} m(n)$$

because proof-of-work blockchains, Bitcoin-NG included, are vulnerable to selfish mining by attackers larger than $1/4$ of the network [77].

3.2 Bitcoin-NG

Bitcoin-NG is a blockchain protocol that serializes transactions, much like Bitcoin, but allows for better latency and bandwidth without sacrificing other properties.

The protocol divides time into epochs. In each epoch, a single leader is in charge of serializing state machine transitions. To facilitate state propagation, leaders generate blocks. The protocol introduces two types of blocks: *key blocks* for leader election and *microblocks* that contain the ledger entries. Each block has a header that contains, among other fields, the unique reference of its predecessor; namely, a cryptographic hash of the predecessor header.

We detail the operation of the protocol in this section and explain its incentive system in Section 3.3.

3.2.1 Leader Election and Key Blocks

The problem of leader election was apparently first formulated and solved in 1977 by Gerard LeLann [108]. In 1982, Hector Garcia-Molina addressed the problem in a distributed system that admits failures [83]. Since then leader election has been extensively used to improve the performance of distributed systems (e.g., [61, 130]). In these classical consensus protocols, the leader's role is to propose decisions that have to be confirmed by a quorum. This can be compared to blockchain protocols where the block of a leader (as defined here) is confirmed in retrospect by subsequent blocks of subsequent leaders.

In Bitcoin-NG, key blocks are used to choose a leader. Like a Bitcoin block, a key block contains the reference to the previous block (either a key block or a microblock, usually the latter), the current Unix time, a coinbase transaction to pay out the reward, a target value, and a nonce field containing arbitrary bits. As in Bitcoin, for a key block to be valid, the cryptographic hash of its header must be smaller than the target value. Unlike Bitcoin, a key block contains a public key, whose corresponding signature identifies the valid subsequent microblocks. In Bitcoin-NG, a leader is a miner that holds the private key associated with the public key in the latest key block. This private key is independent of a miner's communication endpoint; hence, a leader is not tied to a particular IP address.

As in Bitcoin, for a miner to generate a key block, it must iterate through nonce values until the cryptopuzzle condition is met. Consequently, the interval between consecutive key blocks is exponentially distributed. To maintain a set average rate, the difficulty is adjusted by deterministically changing the target value based on the Unix time in the key block headers.

In case of a fork, just as in Bitcoin, the nodes pick the branch with the most work, aggregated over all key blocks, with random tie breaking.

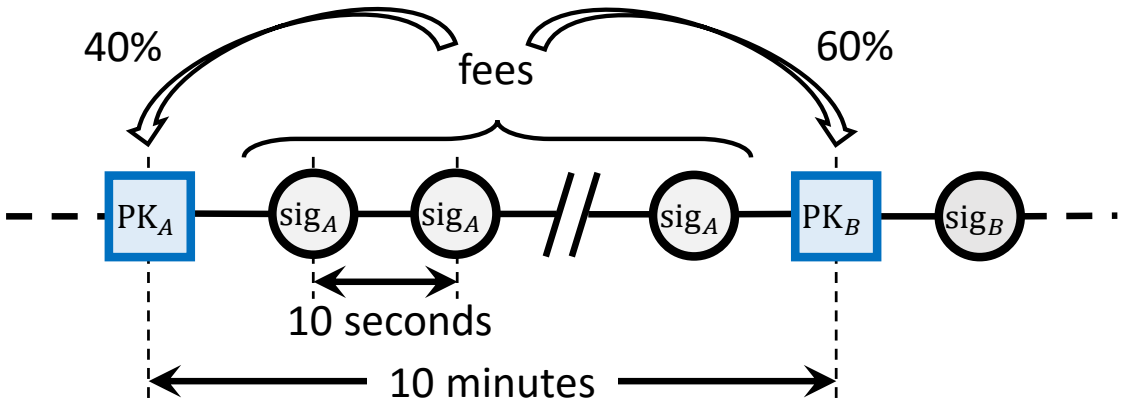


Figure 3.1: Structure of the Bitcoin-NG chain. Microblocks (circles) are signed with the private key matching the public key in the last key block (squares). Fee is distributed 40% to the leader and 60% to the next one.

3.2.2 Microblocks

Once a node generates a key block it becomes the leader. As a leader, the node is allowed to generate microblocks at a set rate smaller than a predefined maximum. The maximum rate is deterministic, and can be much higher than the average interval between key blocks. The size of microblocks is bounded by a predefined maximum. Specifically, if the timestamp of a microblock is in the future, or if its difference with its predecessor's timestamp is smaller than the minimum, then the microblock is invalid. This bound prohibits a leader (malicious, greedy, or broken) from swamping the system with microblocks.

A microblock contains ledger entries and a header. The header contains the reference to the previous block, the current Unix time, a cryptographic hash of its ledger entries, and a cryptographic signature of the header. The signature uses the private key that matches the public key in the latest key block in the chain. For a microblock to be valid, all its entries must be valid according to the specification of the state machine, and the signature has to be valid. Figure 3.1 illustrates the structure.

Note that microblocks do not affect the weight of the chain, as they do not

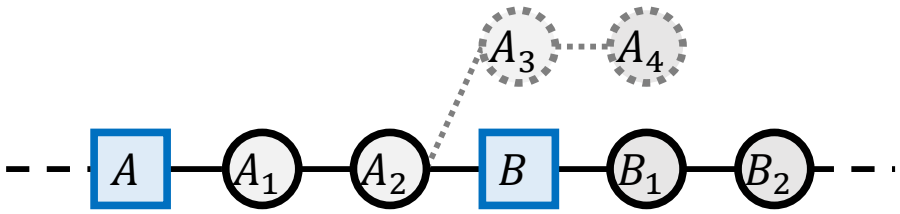


Figure 3.2: When microblocks are frequent, short forks occur on almost every leader switch.

contain proof of work. This is critical for keeping the incentives aligned, as explained in Section 3.3.

3.2.3 Confirmation Time

When a miner generates a key block, he may not have heard of all microblocks generated by the previous leader. If microblock generation is frequent, this can be the common case on leader switching. The result is a short microblock fork, as illustrated in Figure 3.2. Such a fork is observed by any node that receives the to-be-pruned microblock (blocks A_3 and A_4 in the figure) before the new key block (block B in the figure). It is resolved once the key block propagates to that node. Therefore, a user that sees a microblock should wait for the propagation time of the network before considering it in the chain, to make sure it is not pruned by a new key block.

3.2.4 Remuneration

To motivate mining, a leader is compensated for her efforts by the protocol. Remuneration is comprised of two parts. First, each key block entitles its generator a set amount. Second, each ledger entry carries a fee. This fee is split by the leader that places this entry in a microblock, and the subsequent leader that generates the next key block. Specifically, the current leader earns 40% of the

fee, and the subsequent leader earns 60% of the fee, as illustrated in Figure 3.1. The choice of this distribution is explained in Section 3.3.

In practice, the remuneration is implemented by having each key block contain a single coinbase transaction that mints new coins and deposits the funds to the current and previous leaders. As in Bitcoin, this transaction can only be spent after a maturity period of 100 key blocks, to avoid non-mergeable transactions following a fork.

3.2.5 Microblock Fork Prevention

Since microblocks do not require mining, they can be generated cheaply and quickly by the leader, allowing it to split the brain of the system, publishing different replicated-state-machine states to different machines. This allows for double spending attacks, where different nodes believe the same coins were spent with different transactions.

To demotivate such behavior, we use a dedicated ledger entry that invalidates the revenue of fraudulent leaders. Past work has used such entries in different contexts [74, 11, 32]. In Bitcoin-NG, the entry is called a *poison transaction*, and it contains the header of the first block in the pruned branch as a *proof of fraud*. The poison transaction has to be placed on the blockchain within the maturity window of the misbehaving leader's key block, and before the revenue is spent by the malicious leader. Besides invalidating the compensation sent to the leader that generated the fork, a poison transaction grants the current leader a fraction of that compensation, e.g., 5%. The choice of this value is explained in Section 3.3.

Only one poison transaction can be placed per cheater, even if the cheater creates many forks. The cheater's revenue that is not relayed to the poisoner is

lost.

3.3 Security Analysis

In this section, we first explain the incentive system of Bitcoin-NG. Then we explore the other concerns regarding the security of Bitcoin-NG, including its censorship resistance, resilience to variations in mining power, key block and microblock forks, double spending, and wallet security.

3.3.1 Incentives

This section describes how miners with capacity smaller than $1/4$ of the total network are incentivized to follow the protocol. Specifically, miners are motivated to (1) extend the heaviest chain, (2) include transactions in their microblocks, and (3) extend the longest chain. Unlike in Bitcoin, the heaviest and the longest chains are not identical.

Heaviest Chain Extension. The motivation for extending the heaviest chain is the same as in Bitcoin. Since the honest majority will extend the heaviest chain, it will remain the main chain with high probability. A dishonest majority may arbitrarily switch to any branch and win [105]. A minority choosing to mine on another branch will not catch up with an honest majority, therefore it will mine on the main chain to ensure its revenues. We therefore argue that the guarantees of Bitcoin-NG are similar to those of Bitcoin [128] with respect to the Termination and Agreement properties of Nakamoto consensus.

Microblocks carry no weight, not even as a secondary index. If they did, it would increase the system's vulnerability to selfish mining [76, 132, 148]. In

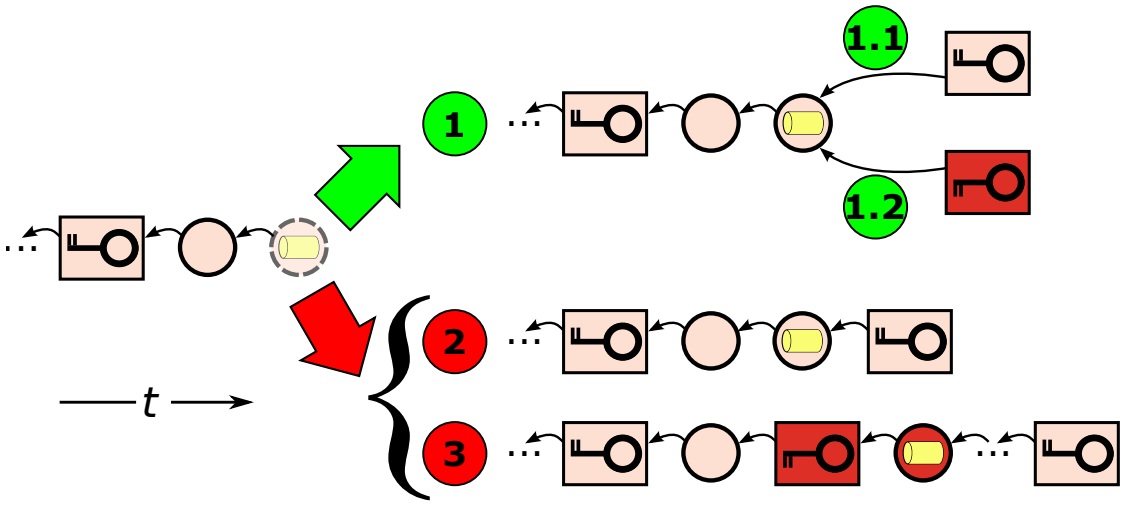


Figure 3.3: Attacks based on hiding microblocks. Shading indicates blocks generated by a specific miner. An honest miner publishes her microblock (case 1), but she may (case 1.1) or may not (case 1.2) generate the subsequent key block. The fraction of transaction fees that goes to the preceding miner should be large enough to prevent her from performing attacks by hiding microblocks (case 2 and 3).

selfish mining, an attacker withholds blocks it has mined and publishes them judiciously to obtain a superior presence in the main chain. If microblocks carried weight, an attacker could keep secret microblocks and gain advantage by mining on microblocks unpublished to anyone else.

We conclude that Bitcoin-NG does not introduce a new vulnerability to selfish mining strategies, and so Bitcoin-NG is resilient to selfish mining against attackers with less than $1/4$ of the mining power. We therefore argue that the guarantees of Bitcoin-NG are similar to those of Bitcoin with respect to the validity property of Nakamoto consensus.

Transaction Inclusion. A leader collects 40% of a transaction's revenue by placing it in a microblock, and may earn the remaining revenue if she succeeds in mining the subsequent key block. However, she could potentially improve her revenue by secretly trying to earn 100% of the fee. To do so, first, the leader

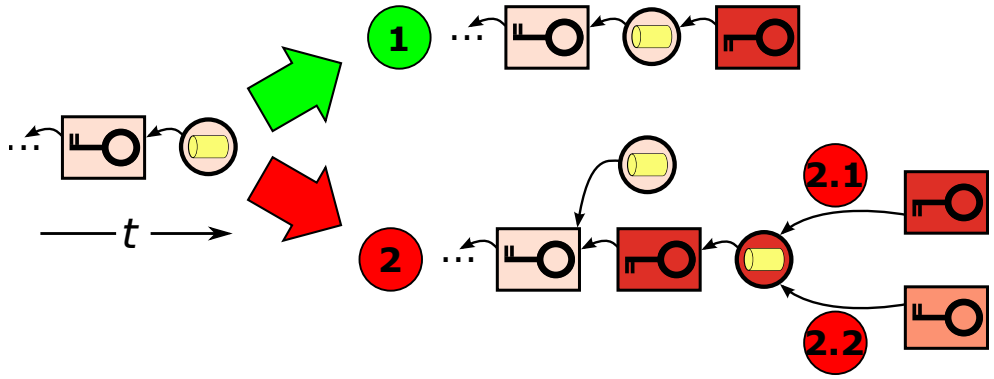


Figure 3.4: Attacks based on ignoring microblocks. Shading indicates blocks generated by a specific miner. An honest miner generates a key block on the latest microblock that she knows of (case 1). Whereas an attacker (case 2) that ignores microblocks may (case 2.1) or may not (case 2.2) obtain 100% of the transaction fees. The fraction of transaction fees that goes to the subsequent miner should be large enough to prevent such attacks.

creates a microblock with the transaction, but does not publish it. Then, she tries to mine on top of this secret microblock, while other miners mine on older microblocks. If the leader succeeds in mining the subsequent key block, she obtains 100% of the transaction fees. Otherwise, she waits until the transaction is placed in a microblock by another miner and tries to mine on top of it. Figure 3.3 illustrates these cases.

Consider a miner whose mining power ratio out of all mining power in the system is α. Denote by r_{leader} the revenue of the leader from a transaction, leaving $(1 - r_{\text{leader}})$ for the next miner. In Bitcoin-NG, we have $r_{\text{leader}} = 40\%$. The value of r_{leader} has to be such that the average revenue of a miner trying the above attack is smaller than her revenue placing the transaction in a public microblock as it should:

$$\overbrace{\alpha \times 100\%}^{\text{Win } 100\%} + \overbrace{(1-\alpha) \times \alpha \times (100\% - r_{\text{leader}})}^{\text{Lose } 100\%\text{, but mine after txn}} < r_{\text{leader}} + (1-\alpha) \times 0 + \alpha \times (1 - r_{\text{leader}}) \ ,$$

therefore $r_{\text{leader}} > \frac{\alpha}{1+\alpha}$. Assuming the power of an attacker is bounded by $1/4$ of the mining power, we obtain $r_{\text{leader}} > 20\%$, hence $r_{\text{leader}} = 40\%$ is within range.

Longest Chain Extension. A miner collects 60% of a serialized transaction's revenue if she mines the subsequent key block following the microblock chain containing this transaction. But to increase her revenue from a transaction, a miner could avoid the transaction's microblock and mine on a previous block. Then he would place the transaction in its own microblock and try mining the subsequent key block. Figure 3.4 illustrates these cases. A miner's revenue in the malicious case must be smaller than her revenue by mining on the transaction's microblock as prescribed:

$$\overbrace{r_{\text{leader}}}^{\substack{\text{Place in}\\ \text{microblock}}} + \overbrace{\alpha(100\% - r_{\text{leader}})}^{\substack{\text{Mine next}\\ \text{key block}}} < \overbrace{100\% - r_{\text{leader}}}^{\substack{\text{Mine on existing}\\ \text{microblock}}},$$

therefore $r_{\text{leader}} < \frac{1-\alpha}{2-\alpha}$. Assuming the power of an attacker is bounded by $1/4$ of the mining power, we obtain $r_{\text{leader}} < 43\%$, hence $r_{\text{leader}} = 40\%$ is within range.

Optimal Network Assumption. Incentive compatibility cannot be maintained in Bitcoin-NG for an attacker larger than about 29%. For larger attackers, the intersection of the two conditions is empty. But this limit does not come into play in the general case, where Bitcoin-NG, like Nakamoto's blockchain with random tie breaking [77], are secure only against attackers smaller than 23.2% [148] due to selfish mining attacks.

However, under optimal network assumptions, Bitcoin's blockchain is more resilient than Bitcoin-NG: Assuming a zero latency network where an attacker cannot rush messages — i.e., receive a message and send its own such that other nodes receive the attacker's message before the original one — Bitcoin is believed to be secure against selfish mining attackers of size up to almost $1/3$.

Bypassing Fee Distribution. We note that a user can circumvent the $40 - 60\%$ transaction fee distribution by paying no transaction fee, and instead paying

the current leader directly, using the coinbase address of the leader's key block. However, a user does not gain a significant advantage by doing so. As we have seen above, paying only the current leader increases the direct motivation of the current leader to place the transaction in a microblock, but reduces the motivation of future miners to mine on this microblock. Moreover, if the leader does not include the transaction before the end of its epoch, subsequent leaders will have no motivation to place the transaction.

Other motives for fee manipulation, such as paying a large fee to encourage miners to choose a certain branch after a fork, apply to Bitcoin as well as Bitcoin-NG, and are outside the scope of this work.

3.3.2 Censorship Resistance

A central goal of Bitcoin is to prevent a malicious discriminating miner from dropping a user's transactions. Censorship resistance is not impacted by the frequent microblocks of Bitcoin-NG.

First, we note that a leader's absolute power is limited to her epoch of leadership. A malicious leader can perform a DoS attack by placing no transactions in microblocks. Similarly, a benign leader that crashes during her epoch of leadership will publish no microblocks. Their influence ends once the next leader publishes her key block. The impact of such behaviors is therefore similar to that in Bitcoin, where nodes may mine empty blocks, but rarely do.

Assuming an honest majority and no backlog, a user will have her transaction placed in the first block generated by an honest miner. Since at least $3/4$ of the blocks are generated by honest miners, the user will have to wait for $4/3$ blocks on average, or 13.33 minutes. Key block intervals can be set to a rate that would reduce censorship to the minimum allowed by the network without

incurring prohibitive deterioration of other metrics.

3.3.3 Resilience to Mining Power Variation

Following Bitcoin's success, hundreds of alternative currencies were created [167], most with Bitcoin's exact blockchain structure, and many with the same proof-of-work mechanism. To maintain a stable rate of blocks, different instances of the blockchain tune their proof of work difficulty at different rates: Bitcoin once every 2016 blocks – about 2 weeks, Litecoin [114] every 2016 blocks (produced at a higher rate) – about 3.5 days, and Ethereum [66] on every block – about 12 seconds. However, whichever adjustment rate is chosen, these protocols are all sensitive to sudden mining power drops. Such drops happen when miners are incentivized to stop mining due to a drop in the currency's exchange rate, or to mine for a different currency that becomes more profitable due to a change in mining difficulty or exchange rate of either currency. Such changes are especially problematic for small alt-coins. When their value rises, they observe a rapid rise in mining power, and subsequently a drop in mining power once the difficulty rises. Then, since the difficulty is high, the remaining miners need a longer time to generate the next block, potentially orders of magnitude longer.

In Bitcoin-NG, difficulty adjustments can create a similar problem; however, it only affects key blocks. Microblocks are generated at the same constant rate. As a consequence, in case of a sudden mining power drop, Bitcoin-NG's censorship resistance is reduced, as key blocks are generated infrequently. If a malicious miner becomes a leader, it will generate microblocks until an honest leader finds a key block. Nevertheless, transaction processing continues at the same rate, in microblocks. Additionally, even until the difficulty is tuned

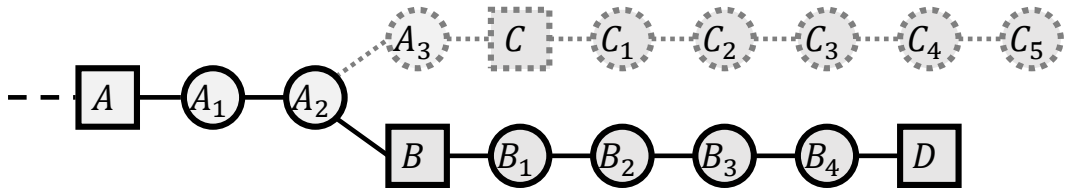


Figure 3.5: Key block fork. Blocks B and C have the same chain weight, and the fork is not resolved until key block D is published.

to a correct value, the ratio of time during which malicious miners are leaders remains proportional to their mining power.

3.3.4 Forks

When issuing microblocks at a high frequency, Bitcoin-NG observes a fork almost on every key block generation, as the previous leader keeps generating microblocks until it receives the key block (Figure 3.2). These forks are resolved quickly — once the new key block arrives at a node, it switches to the new leader. In comparison, when running Bitcoin at such high frequency, forks are only resolved by the heaviest chain extension rule, and since different miners may mine on different branches, branches remain extant for a longer time compared to Bitcoin-NG.

Bitcoin-NG may also experience key block forks, where multiple key blocks are generated after the same prefix of key blocks, as shown in Figure 3.5. This rarely happens, due to the low frequency and quick propagation of the small key blocks. The duration of such a fork may be long, lasting until the next key block. The result is therefore infrequent, but long, key block forks.

Although such long forks are undesirable, they are not dangerous. The knowledge of the fork is propagated through the network, and once it reaches the nodes, they are aware of the undetermined state. All transactions that appear only on one branch are therefore uncertain until one branch gains a lead.

Competition on Key Block Forks. Key block forks raise additional considerations regarding the security of the system. One such consideration is the case, where competing leaders publish transactions that pay a large fee to the subsequent miner in order to entice the other miners to choose their branch. While this competition may introduce interesting dynamics beyond the scope of this work, we note that each branch may copy the transactions placed in the microblocks of the competing branch. Hence, even if an attacker is motivated to place significant fees due to external incentives, her competitor would place the same transactions in her microblocks and remove the attacker's advantage.

3.3.5 Double Spending

Double-spending attacks remain a vulnerability in Bitcoin-NG, though to a lesser extent than in Bitcoin.

Consider a Nakamoto blockchain and a Bitcoin-NG blockchain with the same bandwidth, where the Nakamoto block interval is the same as the key-block interval. A double-spending attacker publishes a transaction t_A, receives a service from a merchant, and publishes an alternative conflicting transaction t_B. A merchant that requires very high confidence should wait for several Nakamoto blocks, or an equivalent number of Bitcoin-NG key blocks. With lower confidence requirements, the guarantees of the protocols differ.

In Nakamoto's blockchain, blocks are infrequent, and transactions are collected by miners until they find a block. Until that time, a transaction t_A can be replaced by another transaction t_B without cost. Publication of conflicting transactions with different destinations is prohibited by the standard Bitcoin software, which also warns the user of conflicting transactions propagating in the network [100].

In contrast, in Bitcoin-NG, microblocks are frequent, and so a leader commits to a transaction by placing it in a microblock. It cannot place t_B without forming a fork and subsequently losing all of its prize from its leadership epoch via a poison transaction.

Other attacks are still possible, where a miner mines before the microblock of transaction t_A and later places a conflicting t_B. Here, the attacker loses the fees of all transactions in pruned microblocks, but this may be worthwhile since the loot from the double-spend can be arbitrarily high. An attacker can mine to prune the chain in advance, and then place a conflicting transaction, or try to prune after the fact.

Reasoning about such attacks calls for a formalization of the attacker's incentives and total mining power. We defer formal analysis that quantifies the security guarantees of Bitcoin-NG and Nakamoto's blockchain to future work. In practice, merchants perform risk analysis to choose a strategy appropriate for their business.

3.3.6 Wallet Security

Wallets are secured by one or more private keys. The holder of such keys can generate transactions to spend funds in a wallet. The possibility of placing a poison transaction enables an attacker that obtains private keys of a leader to revoke her revenue retroactively and earn a small amount. However, such an attacker is better off trying to steal the full revenue of the leader when it becomes available, therefore the introduction of the poison transaction does not add a significant vulnerability.

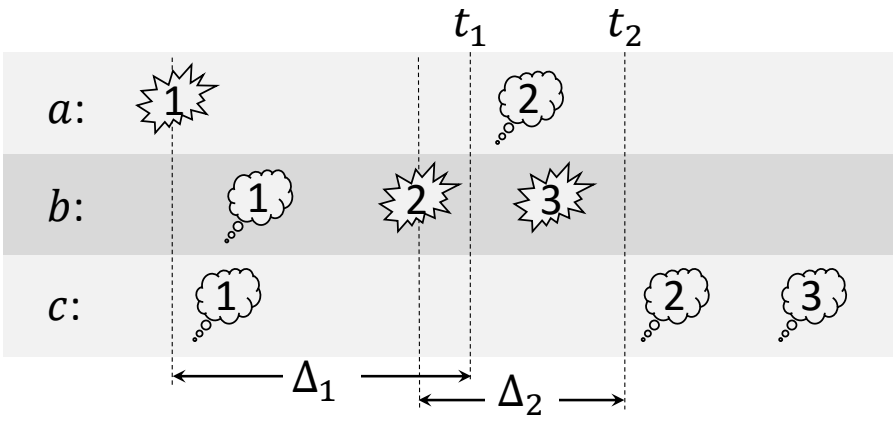


Figure 3.6: Point-consensus delay example with three Bitcoin nodes a, b, and c that generate blocks at heights 1, 2, and 3 (explosions) and learn that these blocks are in the main chain (clouds). Intervals Δ_1 and Δ_2 are the 50%-point consensus delays at times t_1 and t_2, respectively: At least a majority of the nodes at t_i agree on the history until $t_i - \Delta_i$.

3.4 Metrics

We now detail novel metrics by which blockchains can be evaluated. These metrics are designed to evaluate the unique properties of Nakamoto consensus.

Consensus Delay. Intuitively, *consensus delay* is the time it takes for a system to reach agreement. We start by defining, for a specific execution and time, how long back nodes have to look to find a point where they agree on the state.

In a specific execution of an algorithm, given a time t and a ratio $0 < \varepsilon \leq 1$, the ε *point consensus delay* is the smallest time difference Δ such that at least $\varepsilon \cdot |\mathcal{N}|$ of the nodes at time t report the same state machine transition prefix up to time $t - \Delta$. An example for the Bitcoin protocol is illustrated in Figure 3.6.

The consensus delay is the best point-consensus-delay the system achieves for a certain fraction of the time, on average. More formally, the (ε, δ) *consensus delay* of a system is the δ-percentile ε-point-consensus-delay. For example, if 90% of the time, 50% of the nodes agree on the state of the state machine 10 seconds ago (but not less than that), then the $(50\%, 90\%)$-consensus delay is 10 seconds.

Fairness. We calculate two ratios: (1) the ratio of transitions not coming from the largest miner with respect to all transitions, and (2) the ratio of mining power not owned by the largest miner with respect to all mining power. We call the ratio of these ratios the *fairness*.

Optimally the fairness is 1.0: The largest miner and the non-largest miners' representation in the transitions set should be the same as their respective mining powers.

Mining Power Utilization. The security of a proof-of-work system derives from the mining power used to secure it; that is, the mining power an attacker has to outrun to obtain disproportionate control. The *mining power utilization* is the ratio between the mining power that secures the system and the total mining power. Mining power wasted on work that does not appear on the blockchain accounts for the difference.

Subjective Time to Prune. Due to the probabilistic nature of Nakamoto consensus, a node may learn of a state machine transition and subsequently learn that this transition has not occurred – that it was pruned from history. This is the case with pruned branches in Bitcoin.

The δ time to prune is the δ-percentile of the difference between the time a node learns about a transition that will eventually be pruned, and the time it learns that this transition has not occurred. This implies what time a user has to wait to be confident a transition has occurred. Note that this metric only considers transitions that are eventually pruned. Figure 3.7 illustrates an example with the Nakamoto Blockchain.

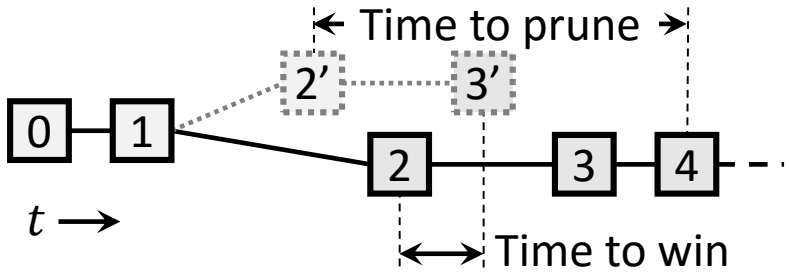


Figure 3.7: A fork in the blockchain with blocks drawn at their generation times, on a time X axis. *Subjective time to prune* is measured from when a node learns of a block in a branch until it realizes what the main chain is. *Time to win* is measured from the creation time of a block until the last time a node generated a conflicting block.

Time to Win. The δ time to win is the δ percentile of the difference between the first time a node believes a never-to-be-pruned-transition has occurred and the last time a (different) node disagrees, believing an alternative transition has occurred. It is zero if there are no disagreements, or if the latter time is earlier. Figure 3.7 illustrates an example for the Bitcoin protocol.

3.5 Experimental Setup

We evaluate Bitcoin and Bitcoin-NG with 1000-node experiments running in real time on an emulated network. Our experiments runs the original operational clients directly on the operating system, emulating only the network properties and mining events.

Implementation. For Bitcoin, we run the standard client (release 0.10.0), hereinafter *Bitcoin*, with minimal instrumentation to log sufficient information.

We implemented all Bitcoin-NG elements that are significant for a performance analysis in the absence of an adversary, by modifying the standard Bitcoin client (release 0.10.0). We did not implement the fee distribution and the microblock signature check. Both elements have negligible impact on perfor-

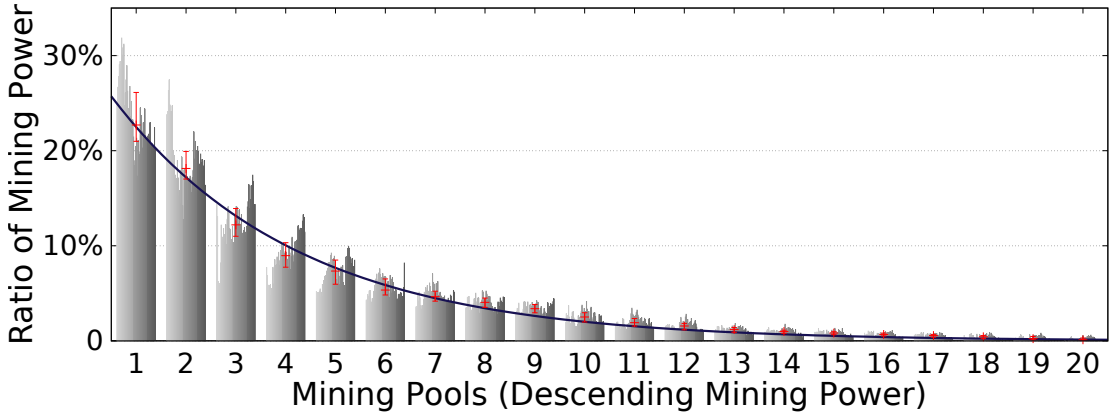


Figure 3.8: Bars represent the 75th, 50th and 25th percentiles of the corresponding batch.

mance — fee distribution requires about one fixed point operation per transaction and signature checking adds several milliseconds per microblock.

Simulated Mining. The time it takes a miner to find a solution follows a geometric probability distribution, which can be approximated as an exponential distribution due to the improbability of a success in each guess and the rate of guessing.

In our experiments we replace the proof of work mechanism with a scheduler that triggers block generation at different miners with exponentially distributed intervals.

Mining Power. The probability of mining a block is proportional on average to the mining power used for solving the cryptopuzzle. Since blocks are generated at average set intervals and the total amount of mining power is large, the interval between block generation events of a small miner is extremely large. A single home miner using dedicated hardware is unlikely to mine a block for years [158].

Consequently, mining power tends to centralize in the form of industrial

mining and open mining pools. Industrial miners are companies that operate large-scale mining facilities. Smaller miners that run private mining rigs typically join forces and form mining *pools*. All members of a pool work together to mine each block, and share their revenues when one of them successfully mines a block.

To reflect in our setup the varying power of miners, we examined the hash power distribution among Bitcoin mining entities. The information we require for the analysis, the identity of the entities generating each block, is voluntarily provided by miners. We used a public API [27] to gather this information for the year ending on August 31, 2015. We note that about 9% of the blocks are unidentified. We considered each such block as generated by a different individual miner.

For each week of the year, we calculate the *weekly mining power* of each entity, and assign rank 1 to the largest weekly mining power, rank 2 to the second largest, and so on. Figure 3.8 shows the weekly mining power of each entity by rank up to 20. Bars of the same shade at different ranks show the distribution of a specific week. Each batch of bars represents the collection of ratios for the n^{th} highest block generating pool. We note that the ranks of different entities is not preserved throughout the weeks. The y-axis represents the weekly ratio of blocks generated by a pool.

To model the size distribution of mining entities, we approximate it with an exponential distribution with an exponent of -0.27. It yields a 0.99 coefficient of determination compared with the medians of each rank.

Network. The structure of Bitcoin's overlay network is complicated, and much of it is intentionally hidden to preserve Bitcoin's security against denial of service (DoS) and to maintain participants' privacy. (Other work [92, 129]

discusses details on the peer-to-peer network.) Nodes do not reveal their neighbors, but provide superset of nodes they have discovered. Many of the nodes are hidden behind firewalls making it difficult to even estimate the full size of the network. The latency among nodes is unknown. Moreover, for many of the metrics that we measure, a critical measure is the time it takes between the generation of a block by some miner and the time at which another miner starts mining on it. The block not only has to be propagated and verified by the second miner, but that second miner must also propagate the details to its mining hardware. In the case of mining pools with many distant worker miners, this may incur a non-negligible delay.

Lacking an existing model of the system, we construct a random network by connecting each node to at least 5 other nodes, chosen uniformly at random. We measured the latency to all visible Bitcoin nodes from a single vantage point on April 7th, 2015, and created a latency histogram. We then set the latency among each pair of nodes in the experiments based on this histogram. The bandwidth is set to about 100kbit/sec among each pair of nodes.

To verify the validity of our setup and topology, we compare Bitcoin's propagation properties in our setup and in the operational system. We perform experiments with different block sizes while changing the block frequency so that the transaction-per-second load is constant. Figure 3.9 shows a linear relation between the block size and the propagation time, similar to the linear relation measured in the Bitcoin operational network by Decker and Wattenhofer [54].

No Transaction Propagation. The goal of this work is to optimize the consensus mechanism of the Blockchain. However, when generating blocks at high frequencies, the overhead of filling in the blocks by generating and propagating transactions becomes a dominant factor with Bitcoin's current implementation.

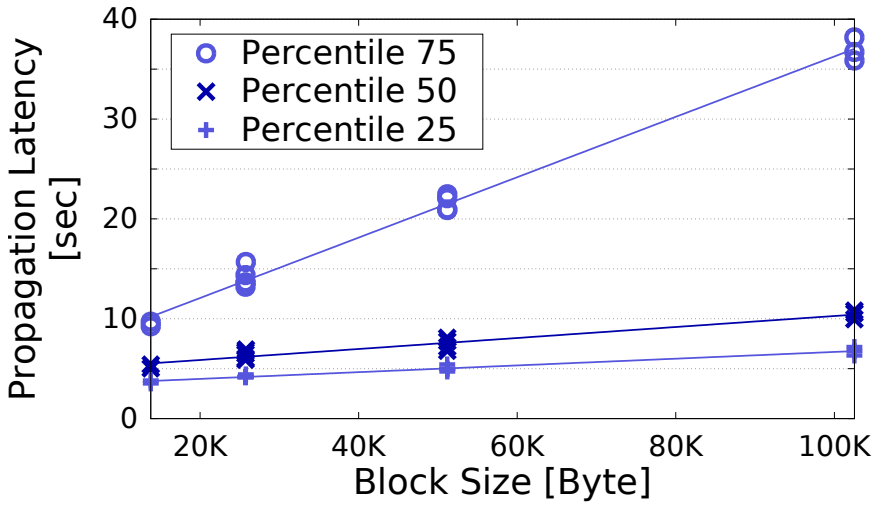


Figure 3.9: In our system, block propagation time grows linearly with block size. This qualitatively matches the linear relation observed in measurements of the operational Bitcoin network [54].

This is not an inherent property of Bitcoin's protocol, or of a Blockchain protocol in general. To reduce the noise caused by the transaction generation and propagation mechanism, we reduce transaction handling to the minimum. Before starting an experiment, we initialize the blockchain with artificial transactions and top up the mempools (the data structure storing yet-to-be-serialized transactions) of all nodes with the same set of transactions. The transactions are of identical size; the operational Bitcoin system as of today, at 1MB blocks every 10 minutes, has a bandwidth of 3.5 such transactions per second.

3.6 Evaluation

We evaluate Bitcoin-NG and compare it with Bitcoin in two sets of experiments, varying block frequency and block size.

Overall, the experiments show that it is possible to improve Bitcoin's consensus delay and bandwidth by tuning its parameters, but its performance deteriorates dangerously on all security-related metrics. Bitcoin-NG qualitatively

outperforms Bitcoin, as it suffers no such deterioration, while enjoying superior performance in almost all metrics across the entire measured range. The bandwidth of Bitcoin-NG is only limited by the processing speed of the individual nodes, as higher throughput does not introduce key-block forks. The consensus delay is determined directly by the network propagation time, because in the common case all nodes agree on the main chain once they receive the latest key block.

In the experiments that follow, we choose the 90th percentile. Lower percentiles maintain the same trends, and very low percentiles show excellent performance – there is always a small subset of nodes that has the correct chain. However, with higher percentiles, the results are lost in the noise. With 1000 nodes and at high percentiles, e.g., 99%, we are measuring the 10th slowest node. Since there are always a few nodes that lag behind, either consistently or temporarily, the results then are dominated by this random behavior, and the trends are not visible.

We measure the metrics we introduced by instantiating them to Nakamoto's blockchain and to Bitcoin-NG as follows.

Consensus delay. We take the $(90\%, 90\%)$-consensus delay based on block generation times. Point-consensus-delay for Bitcoin is illustrated in Figure 3.6. As mentioned in Section 3.3, a user who requires high confidence (e.g., 99%) will not gain better latency with Bitcoin-NG, and must wait for several key blocks to accept a transaction as completed. The guarantees in such cases are similar to those of Bitcoin with the same block interval as Bitcoin-NG's key-block interval.

Fairness. We calculate the proportion of (1) the ratio of blocks in the main chain not generated by the largest miner with respect to all blocks in the main

chain, and (2) the ratio of blocks not generated by the largest miner with respect to all generated blocks.

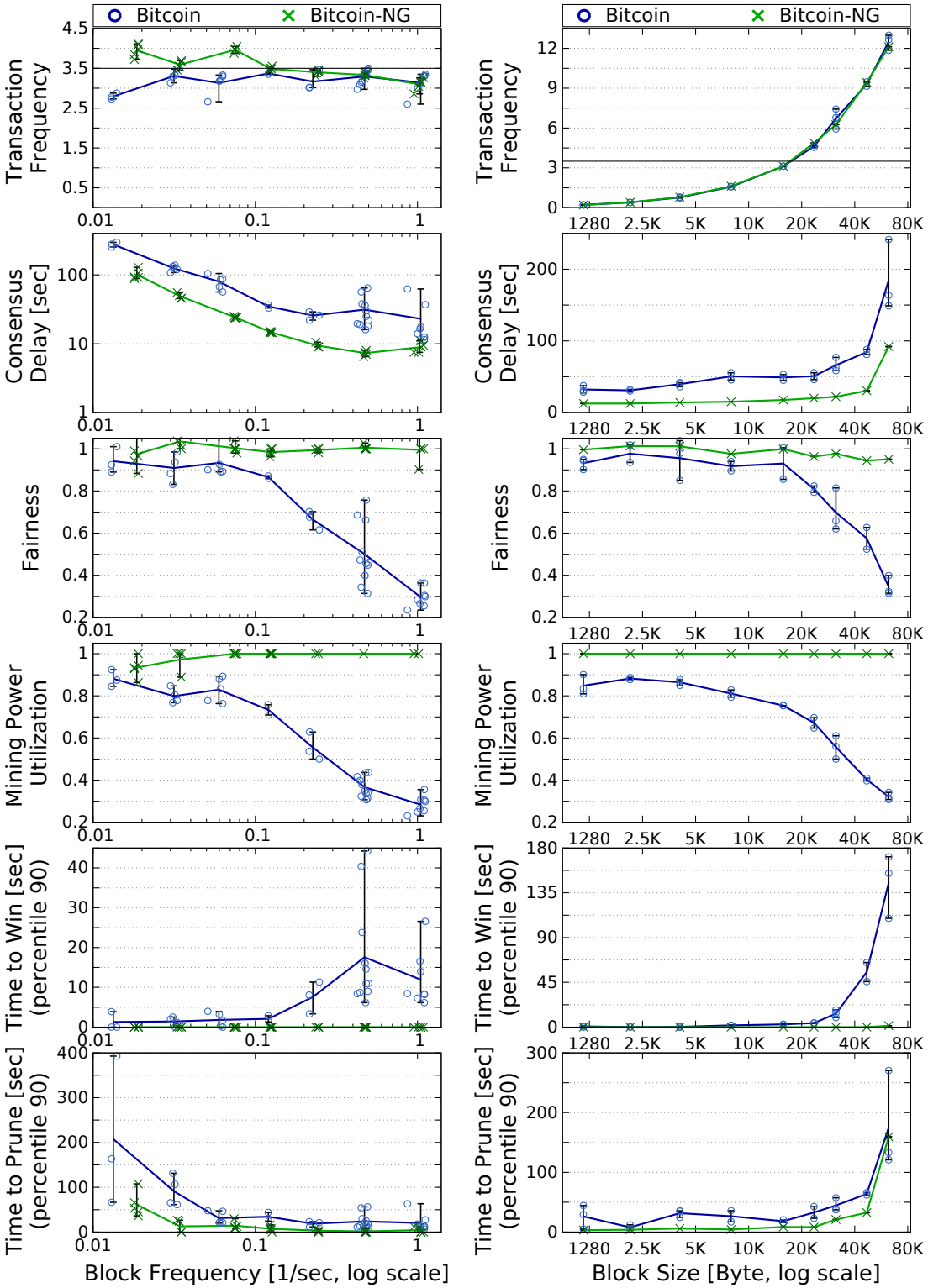
Mining power utilization. We calculate the proportion between the aggregate work of the main chain blocks and all blocks. In Bitcoin-NG, difficulty is only accrued in key blocks, so microblock forks do not reduce mining power utilization.

Time to prune. For each node and for each branch, we measure the time it took for the node to prune this branch. This is the time between the receipt of the first branch block and the receipt of the main chain block that is longer than this branch (Figure 3.7). We take the 90th percentile of all samples.

Time to win. We take the 90th percentile of the time from the generation of each main-chain block to the last time another miner generates a block that is not its descendant (Figure 3.7).

Experiments. We run multiple experiments with different parameters. The figures show the average value for each group of measurements with error bars marking the extreme values. The sampled values are shown as markers.

For each execution we run for 50-100 Bitcoin blocks or Bitcoin-NG microblocks. We perform multiple short runs since all transactions are preloaded for each execution. The mean key-block interval in our experiments is 10 seconds, so each experiment includes leader changes. We do not consider cases where key-block forks occur, since in reality one would choose a much larger key-block interval, e.g., 10 minutes, making key-block forks extremely rare (more rare than with the operational Bitcoin system).



(a) Reducing latency.

(b) Increasing throughput.

Figure 3.10: Experiment results.

3.6.1 Block Frequency

First, we run experiments targeted at improving the consensus delay. For Bitcoin, we vary the frequency of block generation by reducing the proof-of-work difficulty. For Bitcoin-NG, keeping the key block generation at one every 100 seconds, we vary the frequency of microblock generation. For each frequency, we choose the block size (microblock size for Bitcoin-NG) such that the payload throughput is identical to that of Bitcoin's operational system, that is, one 1MB block every 10 minutes. Figure 3.10(a) shows the results.

We confirm that the bandwidth, measured as transaction frequency, is close to 3.5, the operational Bitcoin rate of for such transactions. In our experiments, Bitcoin's bandwidth is smaller than that of Bitcoin-NG, giving Bitcoin an advantage with respect to the other metrics.

As expected, a higher block frequency reduces Bitcoin's consensus delay as transactions are placed in the ledger at a higher frequency. Time to prune improves significantly as block frequency increases. Nevertheless, Bitcoin's frequent forks leave it with higher consensus delay and time to prune than Bitcoin-NG. We note that although they can be made arbitrarily rare, key block forks do occur. Such key-block forks are only resolved once one branch has more key blocks than the others, resulting in a long time to prune if key block intervals are long.

Bitcoin's mining power utilization drops quickly as frequency increases, tending towards $1/4$, the size of the largest miner. At the extreme, block generation is so fast that by the time a miner learns of a block generated by another miner, that other miner has generated more blocks. Then, only the largest miner generates main chain blocks, and the other miners catch up. This also implies the deterioration of fairness, as forks are likely to be resolved by the largest

miner extending its preferred branch. As miners struggle to catch up with the leading pack, slow miners mine on old blocks and the time to win metric increases.

Since contention in Bitcoin-NG is limited to key block generation, forks remain rare despite high frequencies of microblocks. Increasing the microblock frequency achieves reduction of both consensus delay and time to prune. All other metrics are unaffected and remain at the optimal level.

In the low-frequency experiments of Bitcoin-NG, we observe a slight mining power utilization decrease and time to prune increase. This is an artifact of the experimental setup. We run the experiments over a set number of blocks, therefore these low contention experiments run for an extended period, enough to observe key block forks. Note, however, that a realistic Bitcoin-NG implementation can space the key blocks much further apart without affecting performance. Then, due to their small size, key-block forks are highly unlikely, even more so than with standard blocks of Nakamoto's blockchain at the same rate, due to the small size of the key blocks.

3.6.2 Block Size

To study bandwidth scalability, we run experiments with different block sizes. We use high frequencies, similar to those of Ethereum [66], setting Bitcoin's block frequency to 1/10sec and Bitcoin-NG's microblock frequency to 1/10sec and key block frequency to 1/100sec. Figure 3.10(b) shows the result.

As expected, the transaction frequency increases with block size; the horizontal line shows the operational Bitcoin rate.

Large blocks take longer to verify and propagate. Therefore, although block frequency is constant, the time it takes for a miner to learn of a new block is

longer, and so the chance for forks increases.

These experiments demonstrate the expected tradeoff between bandwidth and latency. Consensus delay increases due to forks, as it takes longer to choose the main chain. The time to win also increases, as blocks take longer to catch up with the larger blocks, as does time to prune due to the many forks.

While this tradeoff may be acceptable, allowing for some hunt for a sweet spot on the tradeoff curve, the real problem pertains to security. The forks cause significant mining power loss, reaching about 80% at Bitcoin's bandwidth (though at a higher block frequency), making the system vulnerable to attackers that are much smaller.

Even more detrimental is the reduction in fairness. Even a minor degradation in fairness is dangerous, since it provides incentives to miners to avoid losses by joining forces to enjoy the advantage of mining in a larger pool. This leads to centralization of the mining power, obviating Bitcoin's security properties.

Bitcoin-NG demonstrates qualitative improvement, suffering no significant degradation in the security-related metrics of fairness and mining power. Under heavy load, however, the clients are approaching their processing capacity, making it hard for them to keep up, and we observe degradation in consensus delay and time to prune.

3.7 Conclusion

As Bitcoin and related cryptocurrencies have become surprisingly popular, they have hit scalability limits. The technical debate to improve scalability has been hampered by a perceived inherent tradeoff between performance metrics and security goals of the system. Consequently, the discussions have become acri-

monious, long-term solutions have seemed elusive, and the current sentiment has centered around short-term, incremental, compromise solutions.

Bitcoin-NG shows that it is possible to improve the scalability of blockchain protocols to the point where the consensus latency is limited solely by the network diameter and the throughput bottleneck lies only in node processing power. Such scaling is key in allowing for blockchain technology to fulfill its promise of implementing trustless consensus for a variety of demanding applications including payments, digital asset transactions, and smart contracts — at global scale.

CHAPTER 4

SERVICE-ORIENTED SHARDING FOR BLOCKCHAINS

Solutions for scaling blockchains typically overlook the fact that blockchains host multiple different services and involve participants with diverse expectations. But taking these aspects into consideration promises new on-chain scalability opportunities, providing approaches that scale with increasing number of services. Combining existing scaling solutions with these approaches enables further improvements.

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 [44]. There have been proposals to use blockchains as an underlying layer for services including managing digital assets [46], issuing securities [36], tracking intellectual property [134, 18, 109], maintaining land records and deeds [150, 16], facilitating online voting [81], registering domain names [115], as well as others. Ongoing projects explore ways to make it easier to build such services using Blockchain-as-a-Service (BaaS) platforms [126, 95].

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 [131], 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 [34], 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 [113, 29], forgoing the open architecture, the flexible trust model, and the strong security guarantees of the existing Bitcoin mining ecosystem.

In this chapter, 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 chapter 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.

4.1 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. 4.1). 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

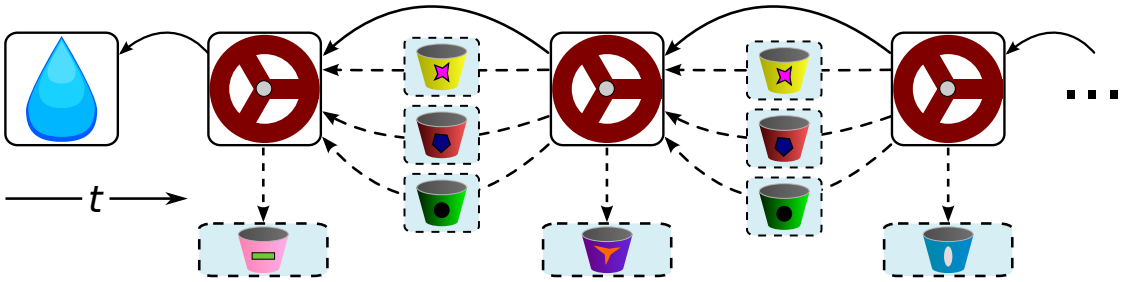


Figure 4.1: Multiblockchain structure of service-oriented sharding. Each channel contains the same genesis block (drop) and checkpoints (valves), as well as the exclusive transactions of a specific service (buckets with the same symbol). Generating a checkpoint requires a proof of work. Miners distribute transactions to designated blocks (a subset of dashed rectangles) secured by checkpoints.

from common checkpoints. Hence, services are loosely coupled and resilient to changes.

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 [166] or Lua [117]. 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 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.

4.2 Aspen

While service-oriented sharding can be built on any blockchain protocol [131, 75, 102, 66], we instantiate on Bitcoin-NG [75] (see Chapter 3), 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. 4.2 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. 4.2). A bud is a key block at a protocol-defined height in terms of the number of key blocks from the genesis block. Aspen adopts proposals based on ballots in key blocks between the current and the

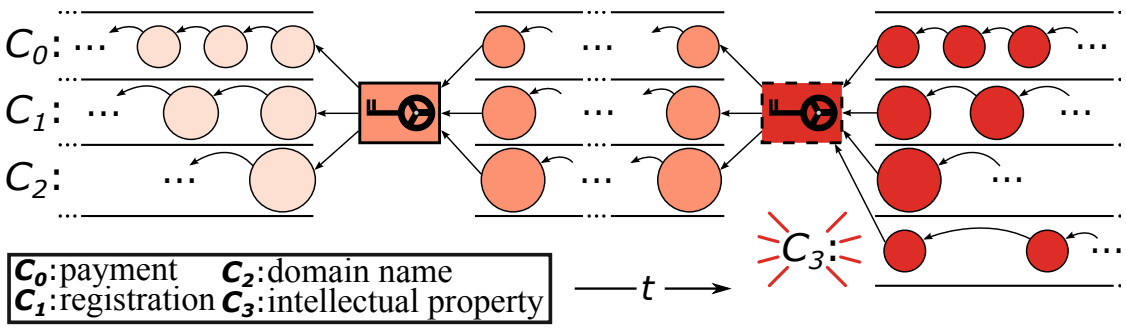
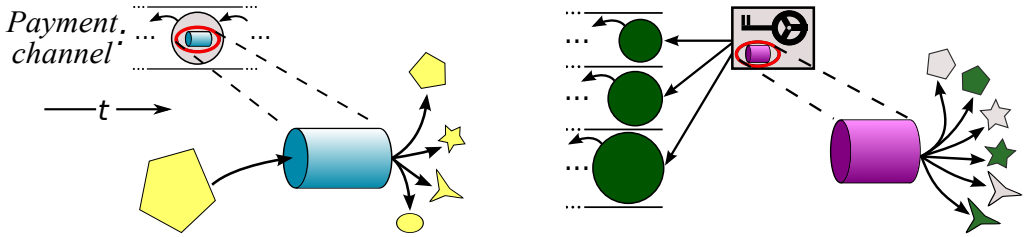


Figure 4.2: Structure of the Aspen chain. Upon generating a key block shared by all channels, a miner serializes service-specific transactions only in the corresponding microblock (circles) chains. Shading indicates blocks generated by a specific miner. A bud (dashed key block) introduces the intellectual property service.



(a) A funding pore.

(b) A coinbase transaction.

Figure 4.3: (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.

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 where it can be spent. Note that transfers across channels are allowed only in one way, from the payment channel to others. Fig. 4.3(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 en-

forces 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.

4.2.1 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. Miners collect their rewards through coinbase transactions in key blocks. A coinbase transaction 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. 4.3(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 [75], 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.

4.2.2 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 [75].

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 [92] or delaying the delivery of valid transactions to her [87]. Countermeasures to mitigate such attacks apply to this work, as well.

4.3 Discussion

In Aspen, funds do not flow back to the payment channel, or across non-payment channels – i.e. each fund is spendable only in a specific channel. The absence bi-directional fund flow makes it possible for service users to detect double spends without keeping track of all channels.

Extending Aspen to enable the atomic exchange of funds between any two channels, x and y, may prove problematic: The simplest approach of adding exchange transactions to key blocks would bloat the blockchain for users of services other than x and y. Whereas keeping the exchange information in channels of x and y would make these channels co-dependent; hence, it enforces users of these services to store, process, and propagate all transactions on both channels for trustless validation. An ideal solution should avoid such co-dependence and bloating.

4.4 Conclusion

Service-oriented sharding provides a means for improving the scalability and extensibility of blockchains with multiple services. To achieve these goals, this technique separates service users from miners, securely addressing the diverse expectations of different participants.

Aspen, the instantiation of service-oriented sharding on Bitcoin-NG (see Chapter 3), 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.

CHAPTER 5

DECENTRALIZATION IN BITCOIN AND ETHEREUM NETWORKS

There is a fundamental tension between scalability and decentralization of blockchains. While scaling solutions would improve the peak throughput and latency of given services, they may also weaken the decentralized nature of the corresponding blockchains. Hence, a scientific quantification of decentralization is critical in assessing the level of decentralization in operational systems and demonstrating the viability of different scaling proposals.

The key feature that empowers innovative services and makes blockchain-based platforms interesting is *decentralization*. Decentralization is a property regarding the fragmentation of control over the protocol. In the Bitcoin and Ethereum protocols, participants take steps by generating blocks at a rate proportional to their computational power. Better decentralization means higher resistance against censorship and tampering. However, if the underlying network properties are inadequate, this fairness property does not hold, as participants are lagging behind and taking deprecated steps.

While ongoing research explores ways to make the Bitcoin and Ethereum networks more decentralized, their current state of decentralization is not well-understood. Hence, debates and decisions about the underlying networks are usually based on assumptions rather than quantitative studies.

In this chapter, through a comprehensive and longitudinal analysis of the decentralization in these operational systems, we shed light on whether existing assumptions are really satisfied in practice. We build on prior Internet measurement techniques, as well as novel approaches for blockchain-based platforms. Our main data sources are (1) direct measurements on these networks, (2) a Bitcoin relay network called *Falcon* that we deployed and operated for a year,

and (3) blockchain histories of Bitcoin and Ethereum. Our study presents findings regarding the network properties, impact of protocol requirements, security, and client interactions.

This chapter makes three contributions. First, it provides new tools and techniques for measuring blockchain-based cryptocurrency networks. A critical tool that provided invaluable data for this study is the Falcon relay network that we built to serve as a backbone for ferrying blocks. We deployed this network for Bitcoin across five continents, capturing unique data. Second, we perform a comparative, quantitative study of the level of decentralization in Bitcoin and Ethereum. Our key findings are: (1) the Bitcoin network is substantially better provisioned, (2) the geolocation and the neighbor selection algorithms of peers are critical to their performance, (3) higher protocol overhead for establishing links between peers makes Ethereum network prone to centralization under high churn, (4) the hash power is more widely distributed in Bitcoin, (5) due to its lower block generation rate and better network properties, Bitcoin's mining utilization is higher. Nevertheless, Ethereum demonstrates better fairness in terms of miner revenue. (6) Bitcoin nodes are significantly more up to date. Finally, we identify potential protocol and infrastructure improvements. For example, Bitcoin IPv4 network undergoes a surprising improvement in its bandwidth capacity; hence, offers opportunities for enhancing the scalability – e.g. higher block size/frequency. Moreover, lower network resources in Ethereum infrastructure indicates that peers would benefit from a relay network.

5.1 Bitcoin and Ethereum Wire Protocols

Despite their architectural similarity, Bitcoin and Ethereum exhibit substantial differences in their API, abstractions, and wire protocol. Chapter 2 provided

the background on these two systems. In this section, we present the details of the corresponding wire protocols that are relevant to the measurements in this chapter.

5.1.1 The Bitcoin Wire Protocol

The first step to communicate with a Bitcoin client is to establish a TCP connection. Once the connection is made, clients exchange unencrypted, protocol-level messages. Each such message consists of a header and a payload, where the header contains: (1) a so-called magic number as a network identity – e.g. `0xD9B4BEF9` for the main network, (2) the command name – e.g. `version`, `getdata`, (3) the payload length, and (4) the first 4 bytes of the double-SHA256 hash of the payload

Handshake. Protocol-level communication starts with a handshake. To initiate this process, the local client sends a `version` message to the remote client. This message contains: (1) the protocol version of the local client, (2) a flag indicating services that the local client provides, (3) the local time, (4) the network addresses of the local and the remote clients, (5) a random node id to detect connections to self, (6) a human-readable name of the local client, (7) the last block height known to the local client, and (8) an optional relay flag indicating whether the remote client should announce relayed transactions or not.

The remote client acknowledges the communication request with a message header for *verack*, and sends a `version` message to connect back.

Requesting Data. Clients use `getdata` messages to ask for the desired blockchain components. Each such message specifies the number of requested

objects, and a list of inventories each containing a SHA256 hash and a flag indicating the identity and type of the object – e.g. a block or a transaction.

Upon receiving a `getdata` message, a client responds with the requested objects. While the latest version of the protocol supports compact [48] and Merkle blocks, the measurement system presented in this chapter focuses on retrieval of full blocks, having transactions and a solution to a cryptopuzzle.

5.1.2 The Ethereum Wire Protocol

Contrary to Bitcoin, not all communication in Ethereum is over TCP. Ethereum employs a UDP-based node discovery mechanism inspired by Kademlia [121], but the rest of the P2P communication is over TCP. Messages between nodes are encrypted and authenticated. Our primary sources in this section are client implementations [89, 143, 51, 137] and Ethereum wiki pages [69, 64, 67, 68, 65]

Upon establishing a TCP connection, the *initiator* and the *responder* perform an encryption, a P2P protocol, a subprotocol, and an optional DAO handshake (for subprotocol versions ≥ 62) before exchanging any other messages. In addition to the address and the port number, nodes rely on unique public keys, dubbed *the node ID*s, for communication.

Encryption Handshake. The initial handshake sets up an encrypted session between the initiator and the responder. First, the initiator sends an `auth` message encrypted with responder's node ID. This message, based on RLPx v3 [68], contains (1) a signature to verify if clients agree on the shared secret, (2) a SHA3 hash of initiator's ephemeral public key to authenticate the signature, (3) the node ID of the initiator, (4) a nonce value, and (5) a flag indicating if the responder knows the initiator. If clients recognize each other, the shared secret is the

token of the last session between these parties, otherwise the secret is generated based on their node IDs.

The responder decrypts the `auth` message, verifies the secret, authenticates the signature, and recovers the initiator's ephemeral public key. The responder's acknowledgement [68] is encrypted with initiator's node ID, and contains (1) the ephemeral public key of the responder, (2) a nonce value, and (3) a flag indicating if the responder knows the initiator.

Recent implementations of popular clients [89, 143, 51, 137] come with the support for EIP-8 [64], which introduces RLPx v4 involving backwards-compatible changes to the encryption handshake process. This version replaces the `auth` flags and the corresponding acknowledgement messages with version numbers and removes the ephemeral public key hash from `auth` messages.

P2P Protocol Handshake. Upon establishing an encrypted session, parties exchange `hello` messages, which contains (1) the local P2P protocol version, (2) a human-readable name of the local client, (3) a list of capabilities each indicating supported versions of a communication subprotocol – e.g. `eth/62`, (4) the port number on which the client is listening, and (5) the node ID of the local client. Starting with this handshake, a client may send a disconnect message for reasons such as being connected to too many peers or having an incompatible P2P protocol version.

Ethereum Subprotocol Handshake. Finally, clients exchange parameters regarding the `eth` subprotocol. To achieve this, each party sends a `status` message, which contains (1) a chain difficulty representing the total difficulty over the latest main chain block, (2) a hash of the local client's chain head, (3) the genesis block hash, (4) a network ID – e.g. `1` for the main network, and (5) the

subprotocol version. Clients choose the subprotocol among the alternatives exchanged during the P2P protocol handshake based on the highest commonly supported version. Handshake in subprotocol versions ≥ 62 ends with the optional DAO handshake.

The Optional DAO Handshake. Hard forks change protocol rules to force participants to bypass the chain selection rule. After the DAO hack [30], as part of a remedial hard fork, the subprotocol has introduced an additional handshake phase to enable clients in the main network to validate remote peers regarding this fork. This challenge succeeds if both clients support or oppose the fork. The process involves validating the `extra-data` field in block headers, which clients normally use for keeping the optional information. A pro-fork client requires the value of this field to be `"dao-hard-fork"` for blocks in the height range $[1920000, 1920010)$. In practice, clients send a `getblockheaders` message to request the header of the block that the DAO hard-fork has commenced on the main network. Then they check the `extra-data` field only for this block. While sending this request is optional, if a client receives such a request, she must respond to prevent the remote peer from disconnecting due to timeout.

Requesting Data. Clients request data using `getblocks` and `getblockbodies` messages in subprotocol versions < 62 and ≥ 62, respectively. Each such message contains a set of hashes identifying the requested block bodies or full blocks.

Response to a `getblockbodies` message contains a list of transactions and uncles for known blocks. Response to a `getblocks` message contains an additional block header for each such request. Starting with version 63, the subpro-

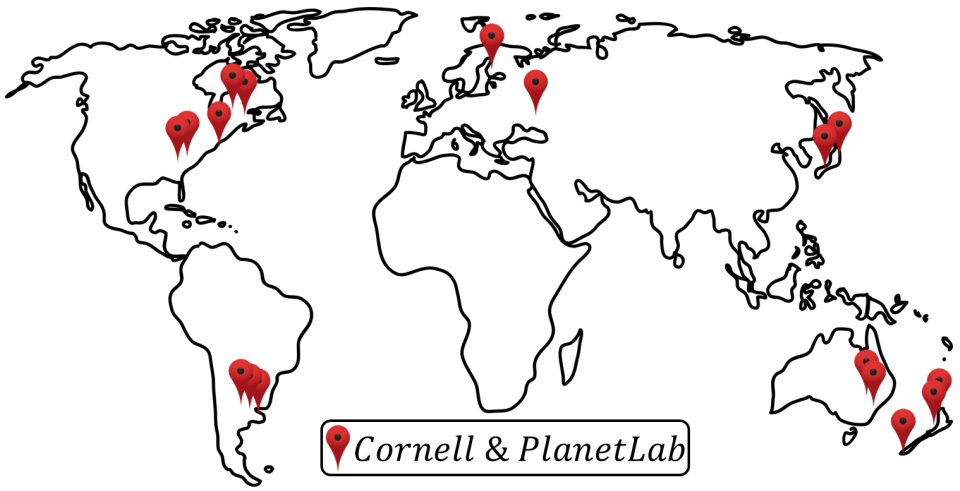


Figure 5.1: The measurement infrastructure is built on 18 globally distributed nodes.

tocol supports transaction receipt requests for fast synchronization. The measurement system presented in this chapter focuses on retrieval of full blocks and block bodies.

5.2 Measurement Infrastructure

Blockchain-based cryptocurrencies operate on global peer-to-peer networks that span multiple administrative domains. Measurement of such networks concerns the exploration of the relationship between peers, the capabilities of individual peers, and the properties of the system as a whole – e.g. its security and fairness. To characterize Bitcoin and Ethereum networks, we deployed *Blockchain Measurement System* (*BMS*), a measurement system that has been collecting data for 12 months.

To enable an in-depth analysis, BMS relies on (1) execution on an infrastructure that spans 5 continents, (2) long-running processes for collecting data, (3) a number of resourceful nodes, and (4) the Falcon Relay Network.

Measurement	Network		Measured Nodes	Dates
(Single Beacon) • Bandwidth (All) • Latency (Bitcoin IPv4)	Bitcoin	IPv4	3441	Jan 11 to 16, 2017 & Jan 30 to Mar 16, 2017
		IPv6	515	Jan 13 to 14, 2017 & Apr 20 to 25, 2017
		Tor	127	Jan 13, 2017 & Apr 23 to 25, 2017
	Ethereum	IPv4	285	Mar 27 to Apr 25, 2017
(Multiple Vantage Points) • P2P Latency	Bitcoin	IPv4	3390	Jan 10 to 15, 2017 & Jan 30 to Mar 01, 2017
	Ethereum	IPv4	4302	Mar 01, 2017 to Apr 11, 2017
(Single Beacon) • Establishing Links (All) • Latency (Bitcoin IPv6)	Bitcoin	IPv4	Connected: 4210 Handshake: 4131	Jan 11 to 16, 2017 & Jan 30 to Mar 16, 2017
		IPv6	Connected: 845 Handshake: 837	Jan 13 to 14, 2017 & Feb 03 to Apr 25, 2017
		Tor	Connected: 183 Handshake: 180	Jan 13, 2017 & Apr 23 to 25, 2017
	Ethereum	IPv4	Connected: 1976 PH1: 995 PH2: 742 PH3: 442	Mar 27, 2017 to Apr 25, 2017
• Peer Freshness	Bitcoin	IPv4	3450	Jan 16, 2017
		IPv6	737	
		Tor	151	
	Ethereum	IPv4	514	Mar 27 to Apr 25, 2017
• Pruned Blocks (Falcon)	Bitcoin	IPv4	5977	May 5, 2016 to Apr 29, 2017

Table 5.1: Timeline of measurements. "Connected" keyword in the "Measured Nodes" column shows the number of nodes with which the BMS has established a TCP connection. PH1–3 indicate measurements for encryption, P2P protocol, and Ethereum subprotocol handshakes, respectively. Falcon is deployed on 10 distinct Amazon EC2 regions.

Multiple Vantage Points. Each vantage point in the measurement infrastructure provides a partial view of the networks from a given source. Combining data from vantage points generates a more complete view of the system. In particular, this enables us to examine peer-to-peer latency. Figure 5.1 shows the geographic distribution of the measurement infrastructure. While 15 out of 18 nodes reside in PlanetLab's global research network [40], the remaining nodes

are part of Cornell's academic network, located in Ithaca, NY.

Long-term Measurements. Collecting data for an extended period of time helps in minimizing measurement errors, capturing the ongoing evolution of networks, and collecting more distinct measurements on various network aspects. To achieve these goals, BMS has been continuously collecting data regarding the provisioned bandwidth of peers, peer-to-peer latency, and link establishment process. These measurements target (1) Bitcoin nodes connected over IPv4, IPv6, and Tor and (2) Ethereum nodes connected over IPv4. As of May 2017, Ethereum does not have any Tor nodes mainly because Tor is exclusively TCP, whereas Ethereum node discovery is UDP-based. Moreover, this study excludes Ethereum's IPv6 network because BMS was not able to discover sufficient number of nodes in this network. Table 5.1 shows the timeline of the data collection for each network containing different number of nodes that respond to the corresponding measurements. BMS obtains the node information from Bitcoin and Ethereum node crawling sites [1, 70], and a locally deployed Ethereum supernode configured with a high peer limit.

Interpretations in this chapter assume that results that BMS has captured from the reachable public nodes are representative of their entire networks. In reality, these networks contain nodes that hide behind firewalls and are not visible to the public. In particular, a fraction of these nodes are part of *mining pools*, involving participants that are willing to invest their resources in Bitcoin [129] or Ethereum networks to generate blocks for the corresponding rewards. Once a member of such a pool discovers a block, she shares the corresponding rewards with the other members. The evaluation of miners within such pools is an open research question.

Resourceful Nodes. Measuring a network at scale requires measurement nodes with extensive resources. In particular, measuring the maximum bandwidth that Bitcoin and Ethereum nodes have access to require nodes with (1) high download capacities to ensure that the bottlenecks are not in the measurement apparatus, and (2) sufficient disk capacities to store detailed results. Since these machines need access to orders of magnitude higher bandwidth capacity than what is achievable on shared infrastructure, such as PlanetLab nodes, BMS data was collected using dedicated, well-provisioned beacon nodes located at Cornell University. We experimentally verified the bandwidth capacity of these nodes by observing the transfer rates of large data chunks transferred from verified servers in Europe and the US.

Falcon Relay Network. The Falcon Relay Network provided critical data that is difficult or impossible to collect otherwise. Specifically, Bitcoin's peer-to-peer network may mask pruned blocks before some nodes receive them. However, Bitcoin has also had public relay networks for providing faster block propagation. Miners are motivated to send blocks to these networks, due to a conceivable decrease in their pruned block rate. This incentive enables our relay network to observe a much larger set of pruned blocks.

5.3 Measurements

In this section, we present comparative measurements of decentralization in Bitcoin and Ethereum. For each measurement, we describe the methodology, explore the corresponding challenges, and examine the results.

5.3.1 Provisioned Bandwidth

Provisioned bandwidth is a lower bound estimate on a node's transmission capacity. It typically corresponds to the limits imposed by the last-mile connection to the Internet. Provisioned bandwidth forms the bottleneck for a peer to receive/transmit components of the blockchain and the corresponding metadata. A well-connected miner benefits from greater bandwidth, because it enables her to propagate/collect blocks to/from the network faster. Malicious miners with such properties also gain additional advantages [77], because once they discover competing blocks, they can rapidly propagate their secret blocks to the network to avoid repercussions.

Methodology

BMS measures the bandwidth of each peer independently in three phases. This process relies on transfer of data from the remote peer's blockchain. First, BMS requests a large amount of data from a peer. Each such request refers to the same set of (1) blocks in Bitcoin and (2) blocks or the corresponding bodies in Ethereum. To minimize the possibility of the remote peer not knowing about the data, requests are limited to blocks that are older than a year. In the second phase, BMS divides the time into discrete epochs and records the number of bytes that it has collected during each one. This process continues until either BMS receives all data or a predefined timeout of 30 seconds is reached. This large timeout conceivably helps BMS (1) extend the data collection period beyond the TCP slow start phase, which typically involves transfer rates substantially lower than the usual and (2) minimize the impact of spurious spikes in throughput over the measurement accuracy. Finally, BMS processes the collected data to determine the provisioned bandwidth. To do so, first, it identifies

the independent data streams by combining successive epochs containing active data transfers. Then, to observe a stabilized transfer rate, it eliminates streams that are shorter than a certain threshold, which is experimentally determined to be 0.5 seconds. Finally, it presents the highest throughput measurement among the remaining distinct continuous streams as the provisioned bandwidth of the remote peer.

Challenges. As with every measurement technique in the real world, the presented approach is subject to experimental limitations and even expected errors. The accuracy of the measurements may drop under certain circumstances, including the cases where: (1) the network bottleneck lies on the side of the measurement beacon rather than the remote peer – i.e. the transmission capacity limit of a remote peer is not determined by its last-mile connection or by a common link that connects it to other network participants, (2) network traffic on the side of BMS interferes with the collected results, and (3) the remote peer intentionally shapes the traffic to selectively limit the bandwidth available to BMS – i.e. bandwidth throttling. The meticulous setup of the bandwidth infrastructure helps us minimize the potential inaccuracies due to the first two cases. Moreover, analysis of popular Bitcoin [20] and Ethereum client implementations [89, 143, 51, 137] shows that the last case is not supported by this software and would require additional, potentially non-trivial work to set up.

Results

Table 5.2 summarizes per-node bandwidth results that BMS has collected. The results indicate that Bitcoin nodes in both IPv4 and IPv6 networks have consistently higher bandwidth compared to Ethereum IPv4 nodes. In particular, typical Bitcoin IPv4 and IPv6 nodes have about $1.9\times$ and $2.7\times$ the bandwidth

	Bitcoin (IPv4) [Mbit/s]	Bitcoin (IPv6) [Mbit/s]	Bitcoin (Tor) [Mbit/s]	Ethereum (IPv4) [Mbit/s]
10%	5.7	11.0	2.1	3.4
33%	23.3	45.2	3.1	11.2
50%	56.1	78.2	4.1	29.4
67%	91.1	94.3	5.6	68.3
90%	177.0	207.9	8.1	144.4
Avg.	73.1	86.5	4.7	55.0
Std. Dev.	68.4	66.9	2.4	58.8

Table 5.2: Observed per-node provisioned bandwidth.

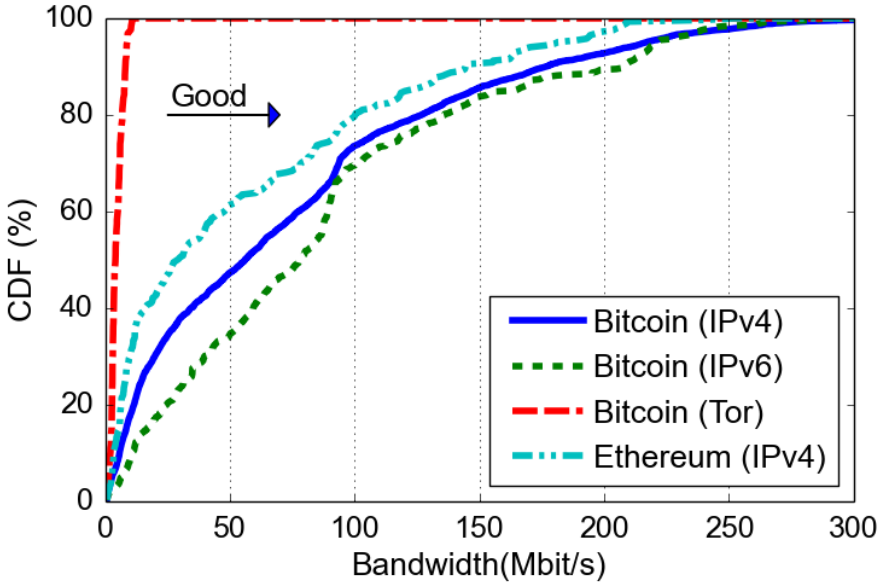


Figure 5.2: Provisioned bandwidth (CDF).

of a typical Ethereum IPv4 node, respectively. This high capacity difference between the networks is also preserved for the average bandwidth. In contrast, Bitcoin Tor nodes have an order of magnitude lower bandwidth compared to directly connected nodes, though they are not unusably slow – e.g. 90% has more than 2 Mbit/s.

Figure 5.2 shows the cumulative distribution of the bandwidth measurements. Steep increases in Bitcoin IPv4/IPv6 curves at around 10 Mbit/s and 100 Mbit/s regions represent typical bandwidth capacities of a home user, and

a typical Amazon EC2 Bitcoin instance. For Ethereum, we observe a similar accumulation around 10 Mbit/s region, but the bandwidth is more evenly distributed over the remaining nodes. As the long tailed distribution and higher standard deviation indicates, the bandwidth of Bitcoin IPv4/IPv6 nodes are spread out over a wider range of values compared to Ethereum nodes. While Bitcoin nodes reach capacities around 300 Mbit/s, the highest Ethereum node capacity that BMS has observed is limited to 250 Mbit/s.

Overall, the results indicate that a significant fraction of Ethereum and Bitcoin IPv4/IPv6 nodes have bandwidth capacities beyond typical home users. 90% of the nodes in these networks have provisioned bandwidth over 3.4 to 11.0 Mbit/s. In contrast, even the 90^{th} percentile bandwidth of Tor nodes is around 8.1 Mbit/s. These results show the tradeoff between bandwidth and network anonymity in cryptocurrency networks. Ongoing research explores alternatives to Tor network that also provide efficient communication [106].

Shared Node Analysis. Analysis of nodes that BMS has established a connection with let us discover 33 common IPv4 addresses that constitute 0.78% of Bitcoin and 1.67% of Ethereum nodes. These nodes use the same addresses with different port numbers. The same IP address indicates the same operator, if not the same physical machine.

Out of commonly connected nodes, BMS determined the bandwidth of 30 Bitcoin and 9 Ethereum nodes. First, we validated the measurement accuracy through one-to-one comparison of Bitcoin to Ethereum bandwidth results for the same addresses. Then, we analyzed why BMS has failed to determine the bandwidth of 3 Bitcoin and 24 Ethereum nodes. We found that, in Bitcoin, 1 such node did not respond to BMS's data requests and the remaining 2 did not send a continuous data stream for a sufficient period of time to let BMS determine the

bandwidth. In Ethereum, 20 out of 24 failed nodes instantly disconnected due to being connected to too many peers. For the remaining nodes, BMS retrieved either insufficient amount of data or no response to complete the handshake.

The results indicate that the majority of the Ethereum nodes are either (1) already connected to the maximum number of peers that they are configured with, or (2) accept connections only from the static and trusted peers. In both cases, remote clients would instantly disconnect from BMS. Compared to Bitcoin, Ethereum has a lower default limit on the maximum number of peer connections. In Ethereum, the hard coded default protocol limit is 25, whereas in the mainline Bitcoin Core client, this limit is 125. This $5\times$ difference would be one possible explanation of these observations.

Evolution of Provisioned Bandwidth in Bitcoin. One of the most interesting discoveries of this study is that the Bitcoin network has improved tremendously in terms of its provisioned bandwidth. The results show that Bitcoin IPv4 nodes, which used to be connected to the network with a median bandwidth of 33 Mbit/s in 2016 [52], now have a median bandwidth of 56 Mbit/s, as of February 2017. In other words, the provisioned bandwidth of a typical full node is now $1.7\times$ of what it was in 2016. Considering the fact that over the past year, Bitcoin IPv4 nodes have consistently accounted for about 80% of known public full nodes [1], these results shed light on the overall evolution of Bitcoin peers and the network structure.

The bandwidth of a node is critical in determining system parameters, such as the maximum block size and block frequency. The increase in provisioned bandwidth suggests that, for people who were happy with the level of decentralization that Bitcoin exhibited last year, block size or frequency can be increased by the observed improvement factor without impacting their central-

ization concerns.

5.3.2 Network Latency

The network latency between peers impacts the time to propagate blocks, transactions, and the associated metadata.

Methodology

Single Beacon Latency. Latency between a measurement node and peers in Bitcoin or Ethereum networks provide a partial network view from the perspective of a specific source. The distribution of collected latencies enables a node to pinpoint its relative position in the global network. BMS measures latencies using minimum network ping times. For repeated measurements, it picks the minimum observed value.

Geographical Distance. BMS uses geolocation data to estimate the distribution of Bitcoin and Ethereum nodes on Earth, as well as the impact of distance on measured latencies. These estimations represent the shortest path between (1) peers within cryptocurrency networks and (2) the source measurement node and each peer. To calculate distances, BMS applies the Haversine formula [151] using the coordinate values gathered from an IP-based geolocation service [97].

Peer-to-Peer Latency. Measuring the exact peer-to-peer latency requires access to the endpoints. But, BMS bypasses this requirement by providing estimates based on observed latencies from multiple beacons. First, from a single vantage point, it measures the latency to each peer. Then it uses the triangle inequality to estimate upper and lower bounds for the latency between peers. Then it repeats

the same process from other vantage points. Overall, this process yields a set of bounds for each pair of peers. Finally, BMS determines a range for latency estimates between each peer by picking the maximum lower bound and the minimum upper bound. The chapter presents the estimated mid-range latency between peers.

In both Bitcoin and Ethereum, peers do not reveal their neighbors. Hiding the network structure boosts privacy and security [92, 129], but also makes it harder to infer properties of the network. BMS provides latency estimates for a superset of the actual links between known peers.

Challenges. Previous studies have shown that triangle inequality violations are more common than expected in the Internet [118, 176, 165, 174, 33]. Reasons of such inaccuracies include variations in the routing policies, traffic load, and measurement errors. Through retrieval of long-term data using multiple vantage points, we try to minimize the occurrence of the last two errors. However, due to inherent routing properties of the Internet, BMS may over- or underestimate some latency values. Moreover, BMS cannot collect measurements from IPv4/IPv6 nodes that block ICMP traffic and from Tor nodes due to their dependence on TCP. Finally, the external geolocation service might produce inaccurate estimates. We detect a subset of such cases by crosschecking counterintuitive findings – e.g. low ping latency to a large estimated distance.

Results

Single Beacon Latency. The first two columns of Table 5.3 present single beacon latency in Bitcoin IPv4/IPv6 networks. The results indicate that both the median and the average latency to IPv4 nodes are less than IPv6 node latencies. However, comparison of percentiles beyond 67% shows that IPv6 latencies

	Single Beacon Bitcoin		Peer-to-Peer Bitcoin	Peer-to-Peer Ethereum
	(IPv4) [ms]	(IPv6) [ms]	(IPv4) [ms]	(IPv4) [ms]
10%	29	40	48	92
33%	78	80	79	125
50%	89	95	109	152
67%	98	95	152	200
90%	201	165	286	276
Avg.	97	103	135	171
Std. Dev.	59	62	88	76

Table 5.3: The minimum observed single beacon latencies and peer-to-peer latency estimates.

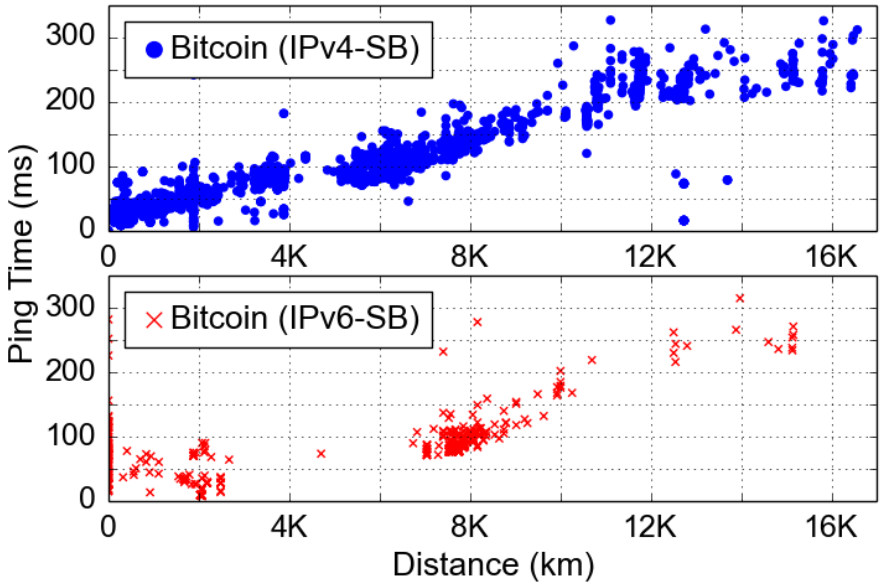


Figure 5.3: Distribution of latency with increasing distance to nodes measured from Ithaca, NY.

are smaller. To understand this phenomenon, we examined the relationship between the latency and distance from the measurement point to remote nodes.

Figure 5.3 illustrates that the majority of IPv6 nodes are located at a midrange distance, whereas IPv4 nodes show higher accumulation at a closer proximity. As expected, latencies increase with the distance; hence, these closer IPv4 nodes make the corresponding latencies smaller for percentiles below 67%.

Equidistant node latencies in 7K to 8K km range shows that IPv6 nodes are faster than IPv4 nodes. While such variations might be due to differences in the underlying routing topologies, no clear trend is observable in other regions. Note that due to the location of nodes, certain distances from our vantage point contain fewer data points – e.g. no Bitcoin client in the ocean. Finally, we see a few (1) IPv6 nodes around 0 km distance, and (2) IPv4 nodes between 12K to 14K km distance with surprisingly low ping times. Crosschecking these nodes using estimates provided by another geolocation service [120] shows that the corresponding estimates are indeed due to inaccuracies of the geolocation service.

Peer-to-Peer Latency Estimates. Contrary to single beacon latencies, peer-to-peer latency estimates represent a global overview of the network. Estimates summarized in the last two columns of Table 5.3 are based on 5.75 million and 9.25 million virtual links in Bitcoin and Ethereum networks, respectively. Peers use a subset of these links to connect to their neighbors. The results show that compared to global peer-to-peer latency estimates, measurements from our Bitcoin IPv4 beacon exhibits lower latency properties. In particular, the median latency from the measurement node is 23% faster than the global peer-to-peer estimates. This demonstrates that the geolocation and neighbor selection of peers have a substantial impact in their performance.

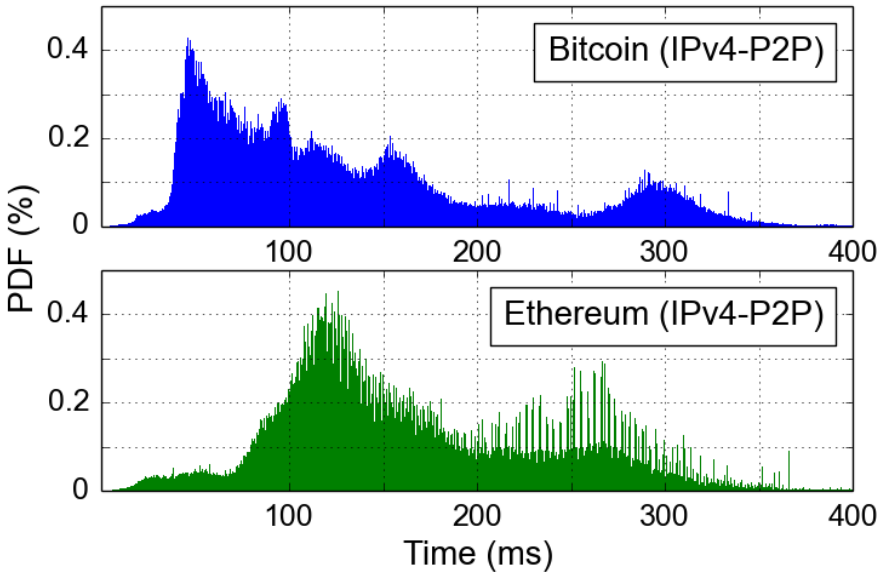
Table 5.3 indicates considerable differences between peer-to-peer latencies of the Bitcoin and Ethereum IPv4 networks. This difference is particularly noticeable in the lower half of the data set. To examine this region in detail, we plotted the PDF of the network latencies as shown in Figure 5.4(a). As also indicated by the lower standard deviation, latency values in the Ethereum network fall within a narrower range. Interestingly, density within this narrow region is

centered around a 120 ms peak, which comes after the second such peak of Bitcoin. Maybe the most compelling observation from these latency distributions is the substantial difference between the two networks regarding the estimated latencies under 100 ms. The figure shows that while 13% of Ethereum latency estimations are under this value, in Bitcoin, this ratio is surprisingly high at 46%. Results indicate a significant divergence within the Bitcoin network, providing much better latency properties to the half of its platform. Given our previous observation in Figure 5.3 that the latency tends to increase with distance, an appropriate question is whether the geographical distribution of peers has any impact on the observed P2P latencies.

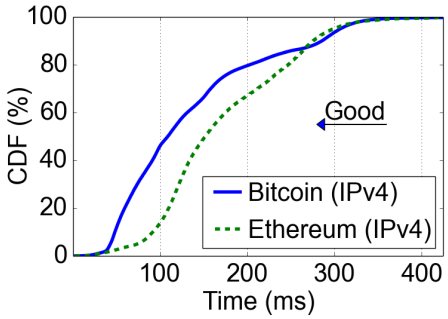
To assess this, we compare Figure 5.4(c), which shows the CDF of peer-to-peer distances in Bitcoin and Ethereum networks, with the CDF of latencies in Figure 5.4(b). We found that typically Ethereum nodes are not accumulated in a single geographical region, but are more evenly distributed around the world. Compared to Bitcoin, the distance between pingable peers in Ethereum is consistently higher – i.e. 19.4% on average. Hence, we conclude that there is a strong correlation between the high peer-to-peer latency and more decentralized structure of Ethereum network.

5.3.3 Link Establishment

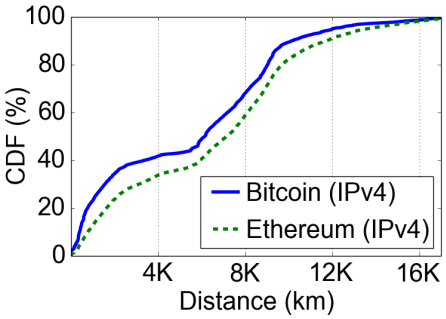
Churn is an inherent property of peer-to-peer networks [157]. Neudecker et al. [133] show that in the Bitcoin network, the majority of observed session lengths are less than 6 hours. In a measurement study, Heilman et al. [92] observe that 43% of connections persist less than two days. On churn, peers look for new neighbors to form links. We study the overhead associated with this process.



(a) Peer-to-peer latency (Histogram).



(b) Peer-to-peer latency (CDF).



(c) Peer-to-peer distance (CDF).

Figure 5.4: Estimates are based on (a)-(b) triangular inequality, and (c) IP-based geolocation.

Methodology

Establishing the TCP Connection. The standard TCP requires a three-way handshake to establish a connection. This process enables endpoints to estimate the round trip time (RTT) before starting data exchange. BMS disables Nagle's algorithm and measures the time to establish these connections from a single vantage point. To ensure that the collected data presumably represent RTT but not the retransmission timeout, it picks the minimum observed value among repeated measurements between the same source and destination.

	Bitcoin			Ethereum
	(IPv4) [ms]	(IPv6) [ms]	(Tor) [ms]	(IPv4) [ms]
10%	28	44	377	27
33%	78	82	627	79
50%	88	95	830	93
67%	99	96	1140	119
90%	204	167	2601	237
Avg.	98	110	1177	131
Std. Dev.	68	112	1040	212

Table 5.4: Observed time to establish TCP connections.

Establishing Protocol-Level Connection. Peers perform a protocol handshake to negotiate the communication rules, set dynamic parameters, or secure the channel. Depending on the protocol design, a handshake may consist of a single message exchange as in Bitcoin, or may involve multiple phases as in Ethereum. While additional phases provide functionalities, such as encryption, extensibility, and resistance against protocol failures, they also come with performance penalties. To examine this tradeoff, BMS records the minimum times to complete each handshake phase.

Results

TCP Connection. Table 5.4 summarizes the times to establish TCP connections. Connection times for Bitcoin IPv4/IPv6 nodes exhibit a similar distribution to corresponding single beacon latencies in Table 5.3. However, connection times for Tor nodes are an order of magnitude higher compared to Ethereum and Bitcoin IPv4/IPv6 due to (1) extra network hops, (2) differential treatment for Tor traffic [33, 170], and (3) forwarding delays caused by encrypted communication, swaps between kernel and user-space, and packet queuing. Cumulative distribution of TCP connection times in Figure 5.5 illustrates this significant difference more clearly. Interestingly, diverging connection times in Ethereum

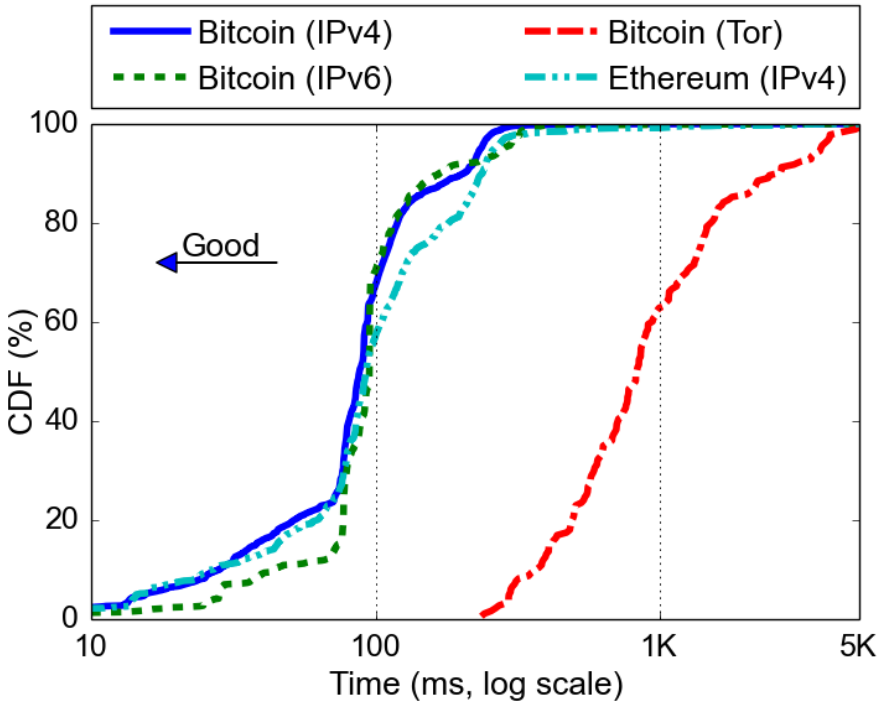


Figure 5.5: TCP connection time (CDF).

	Bitcoin			**Ethereum**
	(IPv4) [ms]	(IPv6) [ms]	(Tor) [ms]	(IPv4) [ms]
10%	32	52	182	34/29/27
33%	80	86	265	93/80/82
50%	93	96	326	129/94/94
67%	114	104	399	175/123/123
90%	228	193	728	322/248/251
Avg.	138	127	462	170/133/151
Std. Dev.	363	280	660	217/220/272

Table 5.5: Handshake times. Ethereum times show Encryption/P2P protocol/subprotocol phases.

nodes beyond 60% indicate that Bitcoin IPv4/IPv6 nodes would be less affected from frequent neighbor changes. This might justify Ethereum's smaller default maximum connection limit.

Protocol Handshake. Table 5.5 provides a list of measured handshake times. The results show that IPv6 handshakes are the fastest and most predictable –

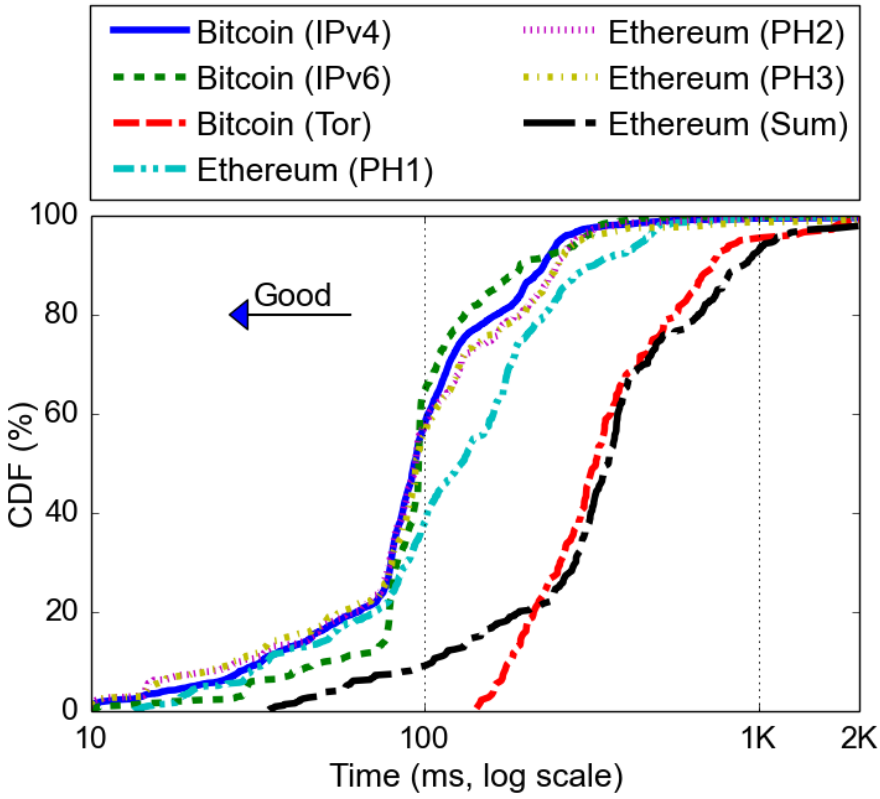


Figure 5.6: Handshake Time (CDF). PH1–3 indicate encryption, P2P protocol, and subprotocol handshakes, respectively. Sum indicates the overall handshake time for Ethereum.

typically in the range of 50 ms to 200 ms. Tor handshakes, by contrast, are around $3\times$ higher than the others. While this difference can be attributed to Tor requirements to route the traffic over ≥ 3 relays, surprisingly, the total handshake time of Ethereum nodes have roughly the same magnitude.

Figure 5.6 shows the cumulative distribution of handshake times and phases of Ethereum handshake. We observe that curves of these handshake phases are located close to Bitcoin curves. This collocation implies that compared to the encrypted communication penalty, multiple round trips have a higher impact on the total Ethereum handshake time.

Overall, we observe that the link establishment in Ethereum incurs higher delays compared to Bitcoin. This makes the process of connecting to Ethereum

peers more costly.

5.3.4 Distribution of Mining Power

Mining on cryptocurrency networks is a complex process that typically requires large computation power. With the current mining difficulty of Bitcoin and Ethereum, using commodity hardware to generate blocks is not feasible [53]. This leads to commercialization of the mining process by means of centralized pools and industrial miners, which typically rely on specialized hardware such as GPU farms or ASICs. Mining pool members represent the same entity.

Methodology

To identify the power of miners in the Bitcoin and Ethereum networks, we examined their weekly distribution over the last 10 months starting on July 15, 2016. The selection of this start date is based on observed stabilization in the mining power distribution in both platforms after this time. Factors that contributed to this phenomenon are outside the scope of this work. Our mining power estimations are based on the ratio of main chain blocks generated by distinct entities. Hence, pruned blocks in Bitcoin and uncles in Ethereum do not affect these estimations. In both networks, miners voluntarily provide the identity information as part of each block they mine. We gathered this data from a public API for Bitcoin [27] and a blockchain explorer for Ethereum [71]. In Bitcoin, 1.8% of the blocks were unidentified, which we treated as if they were generated by distinct individual miners. Finally, we manually processed identities to detect and merge information of the same miners. This includes pools operated by the same admin [98] and multiple identities representing the same pool.

Challenges. The core assumption that our analysis rests on is that the miners accurately self-identify themselves. To see why this is the case, let us look at how miners are involved in the process of adding a feature to the cryptocurrency.

Miners must comply with protocol changes; thus, they often indicate whether they support a proposed change or not. However, this process is not well-established – it might be conducted offline as in Ethereum's DAO hard fork proposal [145], or on-blockchain as in Bitcoin's SegWit proposal [116]. For example, SegWit requires 95% of the latest 2000 blocks to signal readiness before activating.

While strong miners gain political clout and attract more members, getting too large raises alarms among the community about centralization. Thus, such miners may conceal or obfuscate this information to appear less powerful – e.g. by generating multiple identities. For instance, two major mining pools, Ethpool and Ethermine, publicly reveal that they share the same admin [98]. Thus, any analysis based on the voluntary miner data tends to skew towards a more decentralized network than the reality.

Results

For each week of the analysis period, we calculated the corresponding mining power of entities and assigned indices to each miner corresponding to their rank. Figure 5.7 shows the top 20 weekly mining power distribution in Ethereum and Bitcoin networks. Each batch of bars represents a chronologically ordered collection of weekly mining power ratios.

Figure 5.7 illustrates that, in Bitcoin, the weekly mining power of a single entity has never exceeded 21% of the overall power. Moreover, the top 4 Bitcoin miners have represented more than 53% of the average power. In contrast, the

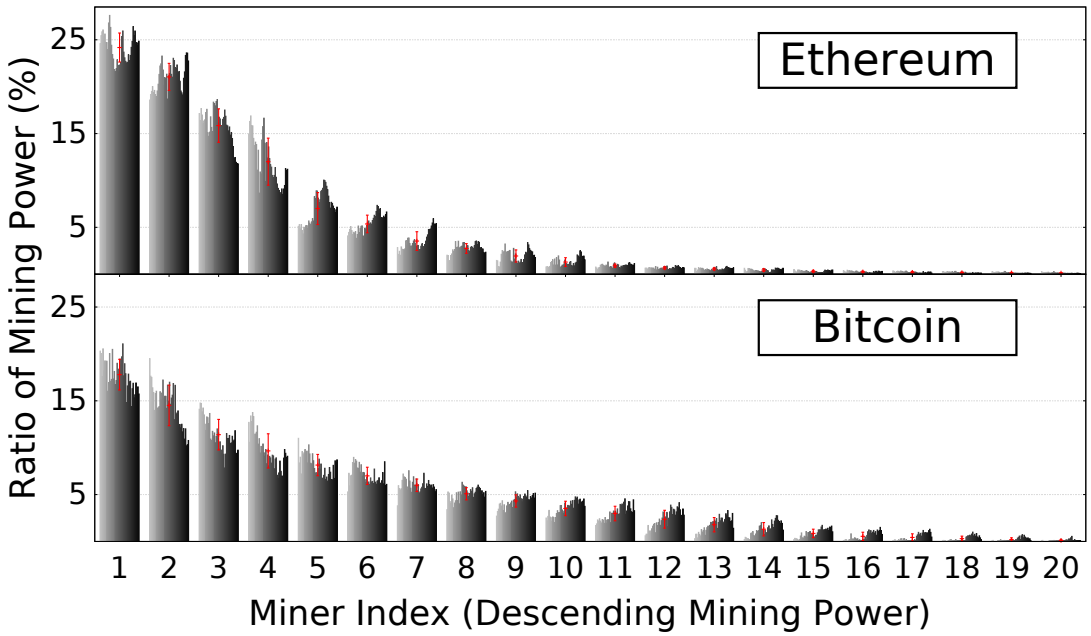


Figure 5.7: Distribution of mining power in Bitcoin and Ethereum networks. Bars indicate observed standard deviation from the average.

top Ethereum miner has never had less than 21% of the power. This entity has consistently possessed a higher power ratio compared to Bitcoin. On average, 61% of the weekly power was shared by only 3 Ethereum miners. These observations signal a more centralized network for Ethereum, making its blockchain vulnerable to censorship.

Although miners do not necessarily keep their rank over the observation period, each rank has limited diversity. In particular, only 2 Bitcoin and 3 Ethereum miners ever held the top rank. Interestingly, a mining pool with the same identity has been the top miner for 29% and 14% of the time in Bitcoin and Ethereum networks, respectively. Over 50% of the mining power has exclusively been shared by 8 miners in Bitcoin and 5 miners in Ethereum throughout the observed period. Even the 90th percentile of the mining power seems to be controlled by only 16 miners in Bitcoin and only 11 miners in Ethereum. Hence, both platforms provide limited diversity in terms of distinct entities they rely

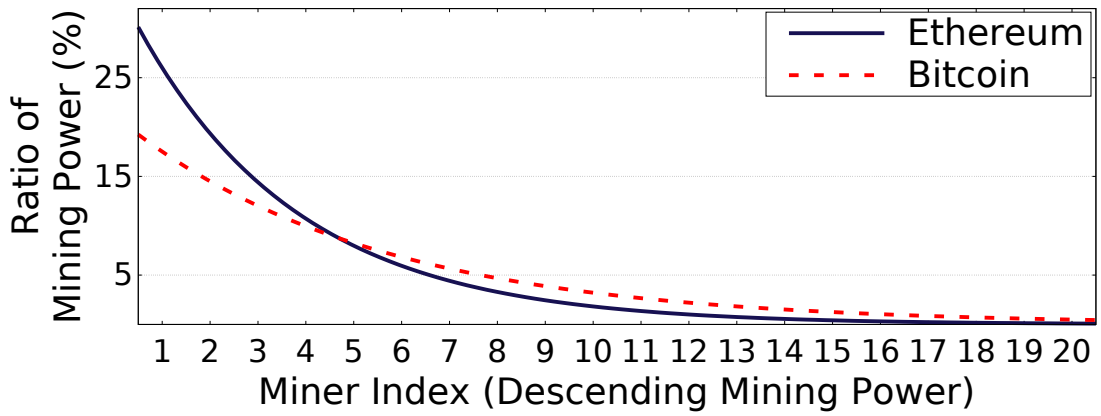


Figure 5.8: Exponential trendlines fitted to the average distribution of mining power.

on for maintenance of their main data structure.

Figure 5.8 shows trend lines of mining power as exponential distributions with curves $0.21e^{-0.19x}$ and $0.35e^{-0.3x}$ in Bitcoin and Ethereum, respectively. These curves fit well to the corresponding average of each rank in Bitcoin and Ethereum, each yielding a coefficient of determination value of 0.99.

5.3.5 Mining Power Utilization

Mining power utilization [75], which measures the fraction of mined blocks that remain in the main chain, is a metric for evaluating the efficiency of a protocol, as well as a second order metric for robustness against rollbacks. As mining power utilization increases, the protocol is able to convert more of the energy spent to useful work, and therefore the cost to launch an attack is higher.

Methodology

To study the mining power utilization, we analyzed weekly and daily distribution of pruned blocks in Bitcoin and uncles in Ethereum, compared to the main chain blocks. We retrieved this data from (1) the Falcon network, (2) a

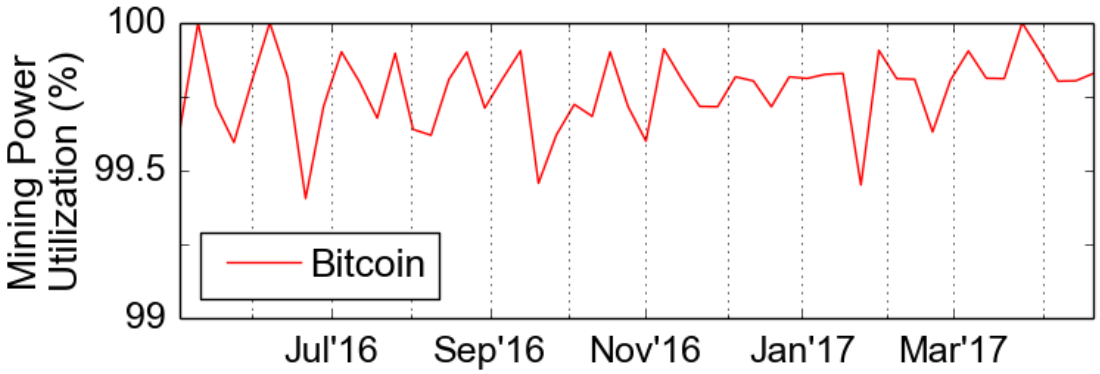


Figure 5.9: Weekly mining power utilization (Bitcoin).

local Bitcoin client, and (3) public blockchain explorers for Bitcoin [27] and Ethereum [71]. In particular, the Bitcoin blockchain explorer and Falcon exclusively provided 12% and 20% of the total 124 pruned blocks, respectively. Both sources commonly discovered the remaining 68%.

Challenges. The design of Ethereum protocol requires peers to store and propagate a fraction of blocks that are out of the main chain – i.e. uncles. In contrast, Bitcoin's blockchain only stores the main chain. Moreover, peers don't propagate blocks once they identify them as pruned. Hence, capturing such blocks in Bitcoin requires actively watching the network. While increasing the number of globally distributed vantage points may boost the detection rate, this discovery process is still best effort. Consequently, our analysis may overestimate the utilization.

Results

Figure 5.9 and Figure 5.10 show weekly and daily distributions of mining power utilization in Bitcoin and Ethereum networks, respectively. The results show that Bitcoin utilization is always above 99%, which represents a small wasted mining effort in this network. In contrast, daily utilization in Ethereum is typi-

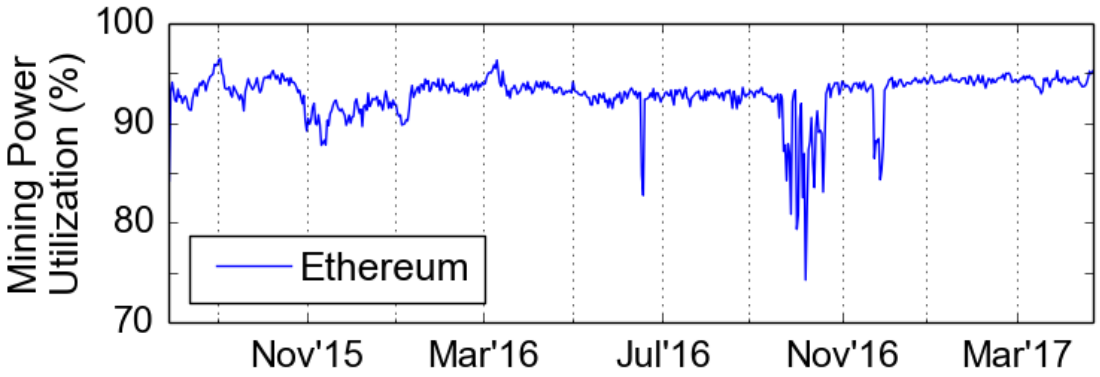


Figure 5.10: Daily mining power utilization (Ethereum).

cally between 90% to 94% range and never goes above the 97% threshold. During 2016, Ethereum faces occasional rapid drops in its utilization down to 74% to 88% range, including (1) the days following the exploitation of the DAO vulnerability from June 17 to June 18, (2) attacks on Ethereum network [168, 31, 159] between September 22 to October 19, and (3) the days following the Spurious Dragon hard fork [99] between November 23 to 29. The results indicate a strong correlation between mining power utilization and real life events in Ethereum. This correlation conceivably indicates preventive measures that spam the network to slow down the DAO attacker, bad actors generating blocks with excessive resource demands leading to increased validation times, and miners with outdated clients.

5.3.6 Peer Freshness

Participants in blockchain-based platforms may be censored by delaying/blocking their information retrieval/propagation [87, 92]. Higher decentralization requires that a larger part of the network knows more about the latest global state. In blockchain- based networks, this state is called *the best blockchain*. As updating the local view is not an instantaneous process, a peer might have

an outdated blockchain. We define a peer as *stale* if its blockchain height is behind the tip of the best blockchain. Such peers are likely to experience a heavier network traffic due to the potential bootstrapping process. We study the level of staleness to examine information diffusion

Methodology

Determination of staleness relies on the comparison of each peer's local view with the best blockchain state. For each peer, BMS identifies the level of staleness in three steps. First, it retrieves the height of each peer's local blockchain during the protocol handshake. While Bitcoin nodes directly send their height as part of `version` messages, Ethereum nodes share the hash of their chain head. BMS translates Ethereum block hashes to their blockchain heights. Note that BMS keeps each such measurement with the corresponding retrieval time. In the second step, for each such retrieval time, BMS (1) identifies the latest block in the best blockchain, and (2) gathers the corresponding height. Then it marks a peer as stale if its local height is less than the height of the best blockchain at the measurement retrieval time. Finally, BMS calculates the level of staleness. This represents the exact time difference between the measurement retrieval and the generation of the first unknown block.

Challenges. Preliminary analysis revealed that a fraction of Bitcoin IPv4/IPv6 peers ($\leq 0.1\%$) report a common height that is even higher than the tip of the best blockchain. We suspect that these nodes are part of a network experiment. BMS removes such Bitcoin nodes from its analysis.

While Ethereum nodes do not report their height directly, we observed that 24% of the measured nodes announced a head hash that belongs to a block out of the DAO hard fork chain. This result is not surprising because Ethereum

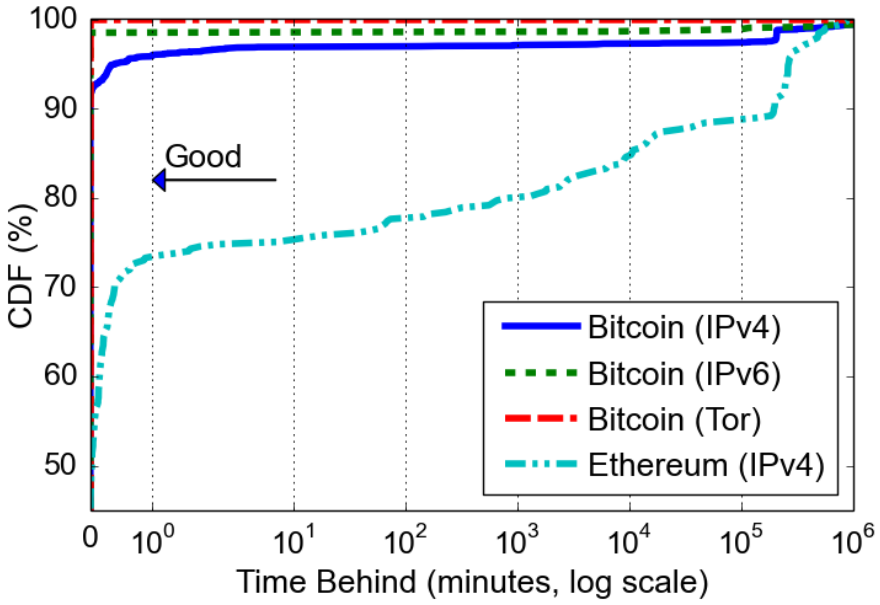


Figure 5.11: Distribution of the level of staleness.

nodes accept a conflicting main chain depending on their stance on the DAO hard fork. However, the divergence in the perceived world views makes it challenging to analyze the global staleness distribution in the network. As the majority of the measurements indicated pro hard fork chain heads, BMS focuses its examination on these peers.

Results

Figure 5.11 illustrates the cumulative distribution of the staleness level in Bitcoin and Ethereum. The results show significant variations in the ratio of stale nodes between the two networks. While in Bitcoin 91.6% of IPv4, 98.5% IPv6, and 100% of Tor nodes have completely up-to-date block information, in Ethereum this ratio remains at 45.8%. The steep increase in the Ethereum curve from 0 to 1 minute indicates a substantial accumulation in this range, which accounts for 27.5% of all nodes. The same range in Bitcoin IPv4 network contains 4.2% of all nodes. The long tail of Ethereum curve beyond 1 minute shows that the

significant staleness gap between the two cryptocurrencies persists.

Ethereum embodies an average block frequency that is an order of magnitude higher than Bitcoin. Figure 5.11 indicates that the local blockchain heights in Ethereum nodes are more smoothly spread out over a wide range. To understand the impact of relative height, we dissected the staleness based on height. We found out that (1) 2.6% of IPv4 and 1% of IPv6 Bitcoin nodes, and (2) 15.7% of Ethereum nodes are more than 1% behind the height of the corresponding best blockchains. Freshness of Tor nodes might indicate that at the time of the measurements, no Tor node has recently joined the network. Note that, at the time of our measurements, 1% height corresponds to 29 and to 6 days of staleness in Bitcoin and Ethereum, respectively.

Dissecting Bitcoin's Stale Nodes. Despite Bitcoin's comparably small staleness ratio, the high ratio of IPv4 nodes that are more than 1% behind the height of the best blockchain was unexpected. On February 15, 2017, we performed a followup study focused on these nodes. Of the 92 stale IPv4 nodes with these properties, 74 responded to this followup. Of these 74, 62 have the same height as before – i.e. they do not try to catch up. We suspect that a software error is keeping these nodes from making forward progress. 74% of stale peers that don't try to catch up use the same agent with sub-version string `"/BTCC:0.13.1/"` – i.e. a client from a Chinese Bitcoin exchange/mining pool.

5.3.7 Fairness

Section 5.3.4 presented the mining power distribution that is correlated with the main chain presence of miners. However, the impact of this distribution

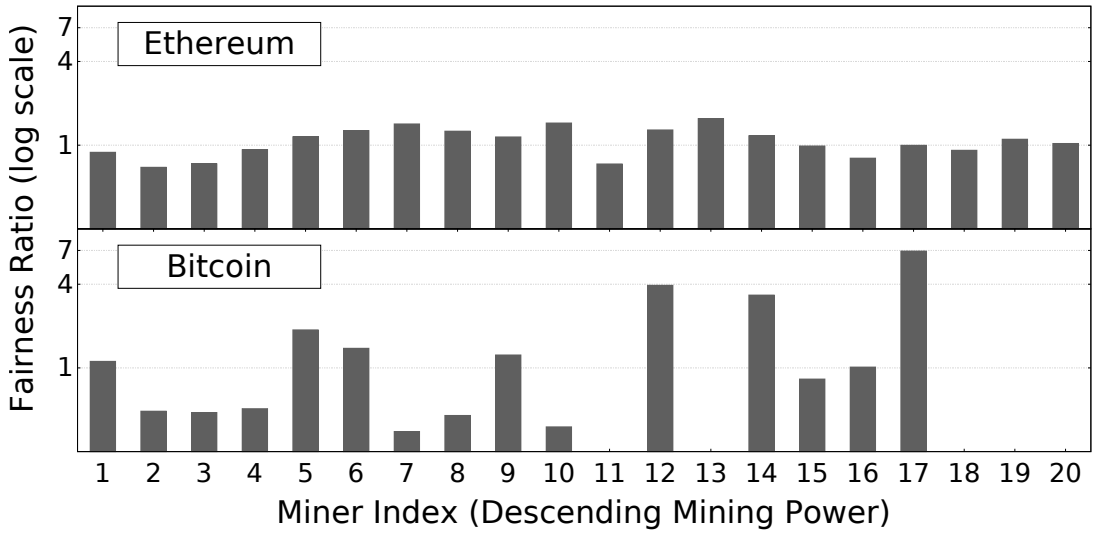


Figure 5.12: Distribution of fairness. Missing bars indicate absence of observed pruned blocks.

on a miner's pruned block generation is not clear. To study this relationship, we define and examine *fairness* as the ratio of (1) a miner's share of pruned blocks to (2) her mining power. In a fair protocol, miners generate pruned blocks proportional to their mining power; hence, the fairness is close to 1.

Methodology

We used the Falcon network, and a Bitcoin blockchain explorer [25] to retrieve pruned Bitcoin blocks. These sources have, respectively, provided 109 and 99 blocks, yielding a total of 124 distinct pruned blocks. We collected uncles from an Ethereum blockchain explorer [71].

Challenges. Our analysis assumes that miners voluntarily identify themselves in uncles/pruned blocks. Identification process for such blocks is unlikely to be biased differently, because miners do not know if their blocks will make it to the main chain or not. Moreover, we also assume that the collected dataset is representative of all pruned Bitcoin blocks. We suspect that explicit storage of

uncles in the blockchain of Ethereum lets us obtain a more accurate analysis on it.

Results

Figure 5.12 shows the distribution of fairness of 20 miners with the highest mining power. The results indicate that, in both networks, the top 4 miners incorporate a higher ratio of blocks into the main chain compared to their uncle/pruned block ratio. While Bitcoin fairness shows a high variance between 0 and 6.93, Ethereum has a more stable distribution in the range of 0.69 to 1.55. In particular, high fairness over smaller Bitcoin miners means more revenue loss for them. In contrast, Ethereum network demonstrates a substantial overall fairness with an average of 1.08.

5.4 Conclusion

The extent of decentralization in blockchain-based platforms is critical in attracting people to these systems. As the size of the user base or the frequency of using blockchain services increase, the corresponding workloads tend to grow, requiring better peak throughput and latency. However, a scaling solution that fulfills these requirements may induce centralization, which undermines the core feature of blockchains. Thus, having a quantitative knowledge of a system's decentralization enables a principled way of assessing the applicability of different scaling solutions.

This chapter presents a comparative assessment of decentralization in two most popular cryptocurrencies, Bitcoin and Ethereum. To do so, it relies on measurements from actual networks, monitoring through Falcon relay network, and static blockchain analysis. In particular, our observations show that Bitcoin pro-

vides a network with higher resource capacity, less overhead under high churn, more diverse mining power distribution, and greater resistance to censorship. Whereas, Ethereum achieves better fairness in miner revenue, and provides less network-link-latency divergence within its infrastructure.

Despite their different level of decentralization, our analysis reveals that both Bitcoin and Ethereum have room for improvement in their protocol and infrastructure.

CHAPTER 6

RELATED WORK

Blockchain technology emerged in 2008 as the infrastructure of a decentralized cryptocurrency, called Bitcoin, but research on cryptocurrencies dates back to early 1980s. Chaum introduced blind signatures [37] and described their use in providing untraceable payments with double-spend prevention in a centralized setting. Following work [38] removed the requirement for the bank to be online at the purchase time, but still needed a central authority. A major mechanism towards decentralized cryptocurrencies, *proof of work*, was introduced by Dwork and Naor [62] for deterring spam emails. MicroMint [147] described the use of this mechanism in minting coins for a micropayment system. Hashcash [10] reinvented the core idea previously discovered by Dwork and Naor, adopting it to use a more efficient hash-based scheme instead of RSA. Aspnes et al. [6] examined the use of continuous proof of work puzzles to deter Sybil attacks [60]. Karma [164] was the first implemented currency to employ proof of work to mint coins. Bitcoin, however, integrated proof of work into the core consensus protocol, yielding the first open-participation microcurrency system that can tolerate Byzantine participants.

In this chapter, we review past work on scaling blockchains while retaining their decentralization.

6.1 Scaling Blockchains

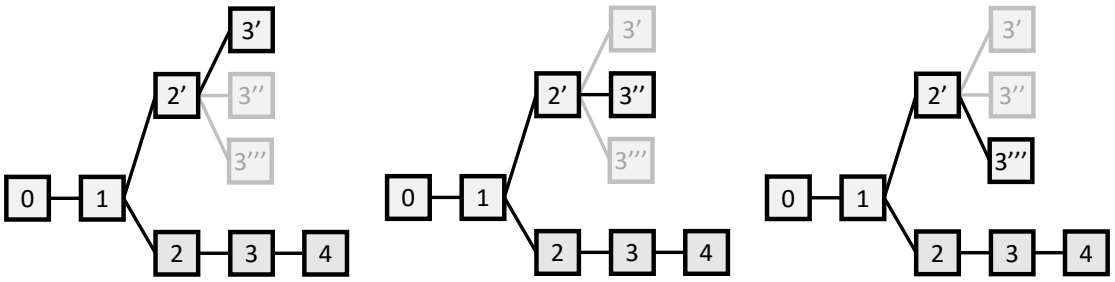
This section presents previous work on scaling blockchains, including the ones that explore scalability with increasing number of services.

6.1.1 Customizing the Chain Selection Rule

Nakamoto consensus suffers from pruned blocks under high contention, and wastes effort. The GHOST protocol of Sompolinsky et al. [153] is an attempt to resolve this issue by changing Bitcoin's chain selection rule. This rule specifies the branch that participants should pick as the extension of the main chain. While, in Bitcoin, the chain with the most work (accumulated over all chain blocks, based on their proofs of work) is the main chain, with GHOST, at a fork, a node chooses the side whose sub-tree contains more work (accumulated over all sub-tree blocks). The benefit is that the heaviest sub-tree choice takes into account proof of work that does not end up in the main chain. Thus, GHOST improves both fairness and the mining power utilization (see Section 3.4) under high contention.

To use GHOST in an operational system, a challenge remains. In protocols that use Bitcoin's chain selection rule, at any given time, at least one node knows what the main chain is since it knows all of its blocks. In GHOST, this is not the case, and it is possible that no single node has enough information to determine which is the main chain. Figure 6.1 illustrates an example of this case with three nodes, 1, 2, and 3, each of which is aware of only a subset of the blocks. Each node knows a chain with a length of height 4, and each knows of a branch of height 3 starting at a block $2'$ and ending at either block $3'$, $3''$, or $3'''$, as shown in Figures 6.1(a), 6.1(b), and 6.1(c), respectively.

One solution to finding the true main chain in GHOST is to propagate all blocks, or all block headers [153]. However, this exposes the system to denial-of-service attacks, as a malicious node can overwhelm the network with low-difficulty blocks. There may be heuristics to avoid the security danger; we do not address this question, but have evaluated the system by implement-



(a) View of node 1 (b) View of node 2 (c) View of node 3

Figure 6.1: A partial view of the GHOST block tree by node 1 (a), node 2 (b), and node 3 (c) does not allow either of them to surmise which is the main chain.

ing it, propagating all blocks. Under these conditions, GHOST performs worse than Bitcoin as the overhead of propagating all blocks outweighs the benefits of the chain selection rule. Nevertheless, a practical implementation of GHOST, overcoming remaining challenges, can be used to complement Bitcoin-NG (see Chapter 3) and allow for a higher frequency of key blocks.

6.1.2 Off-chain Protocols

An alternative to improving the peak bandwidth and latency of the blockchain is to perform transactions off the chain. This basic premise apparently originated in Hearn and Spilman's two-point channel protocol [91]. The Lightning network [141] and duplex micropayment channels [55] allow for payment networks layered on top of a blockchain. More recent work, Teechan [112], leverages trusted execution environments to secure payment channels, without requiring changes in the underlying on-chain protocol.

The security and privacy guarantees of off-chain payment networks differ from those of Bitcoin; as an extreme example, if the nodes performing transactions over a channel crash, all their transactions are lost, as they were never stored in the blockchain. Moreover, the efficacy of such solutions depends on

properties of the emergent payment network, its topology, the amount of value locked in payment channels, as well as the protocol's ability to discover and use payment paths. Overall, these solutions may be suitable for targeted use cases where the additional layer may reduce the number of transactions seen at the lower layers, but, unlike on-chain protocols, they do not address the fundamental problem of scaling a Nakamoto-consensus RSM.

6.1.3 Relay Networks

Relay networks increase network efficiency through faster block propagation. Corallo has built the first such system [47], which provides a centralized fast relay for Bitcoin, parallel to the standard peer-to-peer network. This system significantly improves network throughput and latency by avoiding full block verification and retransmission of transactions already known to peers. Contrary to store-compress-and-forward architecture in this system, Falcon [13], another relay network for Bitcoin, relies on cut-through routing for faster block propagation. Finally, FIBRE [49] incorporates cut-through routing with compact blocks [48] and forward error correction over UDP. Despite their performance benefits, relay networks increase centralized control and reduces fairness — miners outside the fast relay are at a disadvantage.

6.1.4 Multiple Services in Bitcoin's Blockchain

Bitcoin permits storage of arbitrary data on its blockchain using `OP_RETURN` transactions [24]. 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 [46, 135]. 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 [131, 75, 66]. Using commodity hardware, this bootstrapping process takes many hours in Bitcoin [52]. 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.

6.1.5 Federated Chains

Another proposition for improving scalability with increasing blockchain services is that of federated chains. Sidechains [11] enable users to create transactions that can move coins from one blockchain to another; hence, they provide extensibility, as different chains can offer different services. However, this leads to fragmentation of the hash power into distinct blockchains. A compromised sidechain makes the main chain and the other sidechains vulnerable. Moreover, when the payor has funds on one sidechain and the payee would like to spend them on another, the funds have to cross the main chain in order to get the value to their intended destination. Such transfers across sidechains bloat the main chain. Finally, their contribution for efficiency is limited, because they incur high latencies for crossing chains to guarantee that funds will not be pruned from the corresponding chains. Drivechain [160] 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.

6.1.6 Outsourcing the Security

Services with distinct blockchains attempt to improve their security with merged mining [21] 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. [2] show that even the largest merged-mined cryptocurrency, NameCoin [115], 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.

6.1.7 Service-Agnostic Sharding

A possible technique for improving the scalability of blockchains is to shard them – i.e. distribute their contents across nodes in their corresponding networks. Sharding promises to improve the throughput and reduce per-node processing, storage, and bandwidth requirements.

Elastico is a service-agnostic protocol for sharding blockchains [119]. This approach assigns miners to committees for serializing transactions using a classical Byzantine consensus protocol. As in anchoring (see Section 6.1.6), 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 [163] 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.

6.1.8 Customizing the Blockchain Structure

Lewenberg et al. [111] replace the blockchain structure with a directed acyclic graph. There still is a main chain, but its blocks may refer to pruned branches to include their transactions. Analysis demonstrates considerable improvement of fairness and mining power utilization.

ByzCoin [102] is a blockchain protocol that combines proof of work mechanism with PBFT [35]. The corresponding blockchain consists of two interdependent subchains. Using these subchains, the protocol decouples transaction serialization from block mining – an approach inspired by Bitcoin-NG [75].

Bitcoin-NG (see Chapter 3) achieves optimal fairness and mining power utilization. Using Bitcoin-NG with an inclusive blockchain to increase key block frequency may prove problematic: Decommissioned leaders could retroactively introduce transactions and have them included by the current leader. This could allow for DoS and double-spending attacks.

6.1.9 Analysis

Given a cryptopuzzle difficulty and a topology, Sompolinsky et al. [154] calculate upper and lower bounds for the growth rate of the Bitcoin main chain. This analysis can be translated to the expected forking frequency at different difficulty levels when there are exactly two miners. Our experiments target a larger number of miners, modeled according to Bitcoin's operational system, that tune difficulty arbitrarily to reach a target main chain extension rate.

Miller and Jansen [127] describe a methodology for evaluating a large-scale Bitcoin blockchain system on a single machine using an event-driven simulator. To facilitate manageable experiment times, they replace time-consuming cryptographic operations with a delay of an appropriate length.

6.1.10 Faster Bitcoin

Significant effort by Bitcoin's core developers is put into improving the performance of the Bitcoin client and technical aspects of its protocol. While this work can provide significant improvement and enable better scaling, it does not eliminate the inherent limitation that stems from forks forming at high rates.

Stathakopoulou et al. suggest reducing propagation delay in the Bitcoin network [155]. However, their suggestions imply significant compromises on security. First, they have nodes propagate transaction inventories before they know the actual transactions in each inventory; this allows an attacker to swamp the network at no cost by publishing transaction IDs for non-existent transactions. Second, they form a network by having nodes prefer connections with close neighbors — exactly the opposite of the current security-oriented algorithm.

Improving the efficiency of the client [3, 139, 162] can improve propagation time and reduce the collision window (time before A hears B found a block).

However, the improvement is limited — a processing speed increase of $x\%$ (e.g., $x = 200\%$ with [162]) allows for block size increase of $x\%$ at the same fork rate.

6.2 Retaining Decentralization

This section presents existing work on decentralization of blockchain-based platforms, approaches to improve decentralization, and existing sources that provide useful data to interpret decentralization of operational systems.

6.2.1 Centralization in Operational Systems

Network measurements in blockchain-based systems have mainly focused on Bitcoin. One such study [54] demonstrated that the latency is the dominating factor in propagation of blocks smaller than 20 KB. Following work [52] has shown that (1) this limit has increased to 80 KB and (2) nodes are provisioned with substantially higher bandwidth capacity than what the protocol demands. Feld et al. [78] pointed out a strong AS-level centralization that may impact Bitcoin network's connectivity – i.e. 10 ASes contain over 30% of peers. Recent work [4] presented the level of vulnerability, showing that 13 ASes cover the same fraction of peers, but only 39 IP prefixes host half of the overall mining power.

Other work studied various aspects of the Bitcoin overlay network. Miller et al. [129] found that a small fraction of the network, containing around 100 nodes, represents more than 75% of the mining power. The study conjectured that these nodes are well-connected to major mining pools; hence, provide higher efficiency in broadcasting blocks. Biryukov et al. [19] examined peer neighbors to find out IP addresses that correspond to pseudonymous identities.

Another study [104] deanonymized peers by observing anomalous relaying behavior in network. Pappalardo et al. [136] observed that low value transactions may experience waits over a month before being incorporated into blockchain. Other work measured churn and geolocated peers [59]. Gervais et al. [85] discussed centralization concerns regarding the client development process, distribution of mining power, and spendable coins.

6.2.2 Incentives

Incentive compatibility has been a key issue in the investigation of cryptocurrencies and in maintaining their decentralized nature. Babaioff et al. [9] suggest a mechanism to motivate transaction propagation. Lewenberg et al. [110] propose an alternative to the chain structure to motivate the participation of badly-connected miners. Eyal [73] shows that a natural incentive system deters the formation of large open mining pools.

6.2.3 Resource Requirements for System Participation

Recent work presented ways to reduce resource requirements to participate in blockchain systems. These solutions enhance decentralization by increasing the diversity of participants. One such approach [146] relies on authenticated data structures to reduce the load on nodes. Aspen [84] (see Chapter 4) achieves this through sharding the blockchain. In this system, users store, process, and propagate only the data that is relevant to them; hence, need much less resources to join the network.

6.2.4 Blockchain Explorers

Blockchain explorers constitute a valuable data source in assessment of the extent of decentralization in existing systems. Such services [156, 25, 71, 72] provide a variety of information on cryptocurrency networks, including (1) online blockchain history, (2) statistics on blockchain components, transaction fees, and market value, and (3) node information.

CHAPTER 7

CONCLUSION

Scaling blockchains is a multifaceted challenge. First and foremost, increasing adoption of blockchains for demanding services spurs the need for solutions that can achieve a target throughput and latency as the frequency and volume of interactions grow. Moreover, the proliferation of blockchain services entails the exploration of specialized techniques for scaling with growing number of services. Finally, assessing the viability of scaling solutions requires understanding the level of decentralization they can achieve in operational systems and custom metrics for evaluating them. In this dissertation, we broadly investigate how to improve the scalability of blockchains while preserving their decentralized nature.

To improve on-chain scalability, we introduce two blockchain protocols. The first protocol, Bitcoin-NG, shows that high scale can be achieved by separation of duties. In particular, it employs this concept to reassign the roles that are implicitly combined in a single block to new blocks, decoupling the process of block mining from transaction serialization. Bitcoin-NG achieves significant higher throughput and lower latency compared to Bitcoin, without leveraging Bitcoin's trust assumptions. The second protocol, Aspen, complements the first one, providing additional scalability in the presence of growing number of services commingled in a blockchain. This protocol exploits the fact that different participants have different expectations; hence, to achieve high scale, it employs a novel technique that distributes the overall complexity and resource cost among users based on their use cases. Critically, this technique, called *service-oriented sharding*, provides greater resilience against attacks by keeping the total mining power that secures all such use cases combined – i.e. unfragmented.

Furthermore, service-oriented sharding is applicable to the existing blockchain protocols.

To enable a principled way of evaluating blockchain protocols, we develop five specialized metrics for consensus protocols. These are the first and only dedicated metrics for blockchain protocols, and they are crucial for moving the debate over potential protocol modifications to a scientific, quantitative foundation. Using these metrics, we demonstrate quantitative scalability improvements that Bitcoin-NG achieves in a large scale emulation testbed. To make this testbed realistic, we calibrate it using our measurements from the Bitcoin network, and validate it against known network properties from the literature. Finally, we perform evaluations based on different scenarios and constraints from the real world. This includes reflecting the varying power of miners in our testbed.

To enrich the understanding provided by the custom metrics to evaluate blockchain protocols, we design, implement, and deploy tools and techniques for the assessment of decentralization in operational blockchain systems. In particular, we perform a comparative assessment of the extent of decentralization in Bitcoin and Ethereum. This assessment constitutes the first global-scale empirical study of decentralization in blockchain-based networks. Based on our observations, we offer concrete suggestions for potential protocol and infrastructure improvements in both systems.

Scalability of blockchain technologies is closely tied to the ability of the corresponding systems to evolve. This evolution usually involves changes in the protocol, infrastructure, or targeted use cases of the existing platforms. Contrary to fresh systems built to instantiate scaling solutions, existing platforms that adopt new proposals typically benefit from a large user base and strong se-

curity guarantees due to their available mining ecosystem. However, conflicts of interests in existing platforms tend to make the dispute resolution process more challenging.

While this dissertation provides scaling solutions and how to scientifically quantify their viability, it does not explore the governing structure of existing blockchain platforms that manage the adoption of proposals. We believe that an ideal process should be in line with the decentralized nature of blockchains: involving distinct entities in decision making, enabling open discussions with honest feedback, and providing clear roadmaps.

Scaling blockchains while keeping their decentralized essence is necessary for a full appreciation of their potential. Providing services with performance comparable to mainstream technologies, support for novel applications, and sustaining the workload generated by growing user base all depend on successful instantiation of scalability solutions. Our contributions present a step towards addressing these challenges.

BIBLIOGRAPHY

[1] 21.CO. Bitnodes. https://bitnodes.21.co/, retrieved Jun. 2017.

[2] Muneeb Ali, Jude Nelson, Ryan Shea, and Michael J Freedman. Blockstack: A global naming and storage system secured by blockchains. In *Proceedings of the USENIX Annual Technical Conference (ATC)*, Denver, CO, USA, 2016.

[3] Gavin Andresen. O(1) block propagation. https://gist.github.com/gavinandresen/#file-blockpropagation-md, retrieved July. 2015.

[4] Maria Apostolaki, Aviv Zohar, and Laurent Vanbever. Hijacking Bitcoin: Routing attacks on cryptocurrencies. *arXiv preprint arXiv:1605.07524*, 2016.

[5] James Aspnes. Randomized protocols for asynchronous consensus. *Distributed Computing*, 16(2-3):165–175, 2003.

[6] James Aspnes, Collin Jackson, and Arvind Krishnamurthy. Exposing computationally-challenged Byzantine impostors. Technical report, Yale University, 2005.

[7] Giuseppe Ateniese, Ilario Bonacina, Antonio Faonio, and Nicola Galesi. Proofs of space: When space is of the essence. In *Proceedings of the Conference on Security and Cryptography for Networks (SCN)*, pages 538–557. Springer, 2014.

[8] Augur Team. Augur. https://augur.net/, retrieved Jun. 2017.

[9] Moshe Babaioff, Shahar Dobzinski, Sigal Oren, and Aviv Zohar. On Bitcoin and red balloons. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pages 56–73, Valencia, Spain, 2012.

[10] Adam Back. Hashcash-a denial of service counter-measure, 2002.

[11] Adam Back, Matt Corallo, Luke Dashjr, Mark Friedenbach, Gregory Maxwell, Andrew Miller, Andrew Poelstra, Jorge Timón, and Pieter Wuille. Enabling blockchain innovations with pegged sidechains. https://blockstream.com/sidechains.pdf, 2014.

[12] Tobias Bamert, Christian Decker, Lennart Elsen, Roger Wattenhofer, and Samuel Welten. Have a snack, pay with Bitcoins. In *Proceedings of the IEEE International Conference on Peer-to-Peer Computing*, pages 1–5, 2013.

[13] Soumya Basu, Ittay Eyal, and Emin Gün Sirer. The Falcon relay network. `http://www.falcon-net.org/`, retrieved June. 2017.

[14] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 62–73, 1993.

[15] Josh Benaloh, Melissa Chase, Eric Horvitz, and Kristin Lauter. Patient controlled encryption: ensuring privacy of electronic medical records. In *Proceedings of the Cloud Computing Security Workshop (CCSW)*, pages 103–114. ACM, 2009.

[16] Benben Team. Benben. `http://benben.com.gh/`, retrieved Oct. 2016.

[17] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. Proof of activity: Extending Bitcoin's proof of work via proof of stake. Cryptology ePrint Archive, Report 2014/452, 2014. `http://eprint.iacr.org/2014/452`.

[18] BigchainDB GmbH. Ascribe. `https://www.ascribe.io/`, retrieved Oct. 2016.

[19] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. Deanonymisation of clients in Bitcoin P2P network. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 15–29, 2014.

[20] Bitcoin Community. Bitcoin source. `https://github.com/bitcoin/bitcoin`, retrieved Jun. 2017.

[21] Bitcoin Community. Merged mining specification. `https://en.bitcoin.it/wiki/Merged_mining_specification`, retrieved Jun. 2017.

[22] Bitcoin Community. Protocol rules. `https://en.bitcoin.it/wiki/Protocol_rules`, retrieved Jun. 2017.

[23] Bitcoin Community. Protocol specification. `https://en.bitcoin.it/wiki/Protocol_specification`, retrieved Jun. 2017.

[24] Bitcoin Community. OP_RETURN. https://en.bitcoin.it/wiki/OP_RETURN, retrieved Jun. 2017.

[25] Blockchain Info Team. Blockchain Info. https://blockchain.info/, retrieved May. 2017.

[26] Blockstack Inc. Onename. https://onename.com/, retrieved Jun. 2017.

[27] BlockTrail Team. Blocktrail API. https://www.blocktrail.com/api/docs\#api_data, retrieved Apr. 2017.

[28] Joseph Bonneau, Andrew Miller, Jeremy Clark, Arvind Narayanan, Joshua A Kroll, and Edward W Felten. Sok: Research perspectives and challenges for Bitcoin and cryptocurrencies. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 104–121, 2015.

[29] Richard Gendal Brown, James Carlyle, Ian Grigg, and Mike Hearn. Corda: An introduction. http://r3cev.com/s/corda-introductory-whitepaper-final.pdf, Aug. 2016.

[30] Vitalik Buterin. Critical update re: DAO vulnerability. https://blog.ethereum.org/2016/06/17/critical-update-re-dao-vulnerability/, retrieved Apr. 2017.

[31] Vitalik Buterin. Transaction spam attack: Next steps. https://blog.ethereum.org/2016/09/22/transaction-spam-attack-next-steps/, retrieved Apr. 2017.

[32] Vitalik Buterin. Slasher: A punitive proof-of-stake algorithm. https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/, January 2015.

[33] Frank Cangialosi, Dave Levin, and Neil Spring. Ting: Measuring and exploiting latencies between all Tor nodes. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 289–302, 2015.

[34] Riccardo Casatta. OP_RETURN stats. http://opreturn.org/, retrieved Nov. 2016.

[35] Miguel Castro and Barbara Liskov. Practical Byzantine fault tolerance.

In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 173–186, Berkeley, CA, USA, 1999.

[36] Chain Inc. Chain open standard: A secure blockchain protocol for high-scale financial networks. http://chain.com/os/, retrieved Sep. 2016.

[37] David Chaum. Blind signatures for untraceable payments. In *Proceedings of the International Cryptology Conference (CRYPTO)*, volume 82, pages 199–203, 1982.

[38] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *Proceedings of the International Cryptology Conference (CRYPTO)*, pages 319–327, 1990.

[39] Susan Chu. Universities are vying for blockchain dominance. https://www.tun.com/blog/universities-and-governments-are-vying-for-blockchain-dominance/, retrieved Jun. 2017.

[40] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. PlanetLab: an overlay testbed for broad-coverage services. *ACM SIGCOMM CCR*, 33(3):3–12, 2003.

[41] Christopher D. Clack, Vikram A. Bakshi, and Lee Braine. Smart contract templates: foundations, design landscape and research directions. *arXiv preprint arXiv:1608.00771*, 2017.

[42] CoinDesk. Blockchain venture capital. http://www.coindesk.com/bitcoin-venture-capital/, retrieved Jun. 2017.

[43] CoinDesk. CoinDesk research launches state of blockchain q1 report. http://web.archive.org/web/20170606144234/http://www.coindesk.com/coindesk-releases-state-of-blockchain-q1-2017-research-report/, retrieved Jun. 2017.

[44] CoinDesk. State of blockchain q1 2016: Blockchain funding overtakes Bitcoin. http://coindesk.com/state-of-blockchain-q1-2016/, retrieved Oct. 2016.

[45] CoinMarketCap. Cryptocurrency market capitalizations. https://coinmarketcap.com/, retrieved May. 2017.

[46] Colu. Colored Coins. http://coloredcoins.org/, retrieved Sep. 2016.

[47] Matt Corallo. The Bitcoin relay network. BIP 152, http://bitcoinrelaynetwork.org/, retrieved May. 2017.

[48] Matt Corallo. Compact block relay. BIP 152, https://github.com/bitcoin/bips/blob/master/bip-0152.mediawiki, retrieved Jun. 2017.

[49] Matt Corallo. FIBRE: Fast internet Bitcoin relay engine. https://github.com/bitcoinfibre/bitcoinfibre, retrieved Apr. 2017.

[50] Counterparty Team. Counterparty. https://counterparty.io/, retrieved Jun. 2017.

[51] Cpp-ethereum Authors. Ethereum C++ client. https://github.com/ethereum/cpp-ethereum, retrieved Apr. 2017.

[52] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, Dawn Song, and Roger Wattenhofer. On scaling decentralized blockchains (a position paper). In *Proceedings of the Workshop on Bitcoin and Blockchain Research (BITCOIN)*, Barbados, 2016.

[53] Cryptocompare Team. Cryptocurrency mining calculator. https://www.cryptocompare.com/mining/calculator, retrieved Jun. 2017.

[54] Christian Decker and Roger Wattenhofer. Information propagation in the Bitcoin network. In *Proceedings of the IEEE International Conference on Peer-to-Peer Computing*, pages 1–10, Trento, Italy, 2013.

[55] Christian Decker and Roger Wattenhofer. A fast and scalable payment network with Bitcoin Duplex Micropayment Channels. In *Proceedings of the International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 3–18. Springer, 2015.

[56] Michael del Castillo. Microsoft's blockchain supply chain project grows to 13 partners. https://web.archive.org/web/20170602124336/http://www.coindesk.com/microsofts-blockchain-supply-chain-project-grows-to-13-partners/, retrieved Jun. 2017.

[57] Michael del Castillo. Vladimir Putin and Vitalik Buterin discuss Ethereum 'opportunities'. http://web.archive.org/web/20170605195402/http://www.coindesk.com/vladimir-putin-vitalik-buterin-discuss-ethereum-opportunities-recent-forum/, retrieved Jun. 2017.

[58] Kevin Delmolino, Mitchell Arnett, Ahmed Kosba, Andrew Miller, and Elaine Shi. Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab. In *Proceedings of the International Financial Cryptography and Data Security Conference*, pages 79–94, Barbados, 2016.

[59] Joan Antoni Donet Donet, Cristina Pérez-Sola, and Jordi Herrera-Joancomartí. The Bitcoin P2P network. In *Proceedings of the International Financial Cryptography and Data Security Conference*, pages 87–102, Barbados, 2014.

[60] John R Douceur. The Sybil attack. In *Proceedings of the International Workshop on Peer-to-Peer Systems*, pages 251–260. Springer, 2002.

[61] Cynthia Dwork, Nancy A. Lynch, and Larry J. Stockmeyer. Consensus in the presence of partial synchrony. *Journal of the ACM*, 35(2):288–323, 1988.

[62] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Proceedings of the International Cryptology Conference (CRYPTO)*, pages 139–147. Springer, 1992.

[63] Ariel Ekblaw, Asaph Azaria, John D Halamka, and Andrew Lippman. A case study for blockchain in healthcare: "MedRec" prototype for electronic health records and medical research data. http://dci.mit.edu/assets/papers/eckblaw.pdf, retrieved Jun. 2017, 2016.

[64] Ethereum Community. devp2p forward compatibility requirements for homestead. https://github.com/ethereum/EIPs/blob/master/EIPS/eip-8.md, retrieved Apr. 2017.

[65] Ethereum Community. Ethereum wire protocol. https://github.com/ethereum/wiki/wiki/Ethereum-Wire-Protocol, retrieved Apr. 2017.

[66] Ethereum Community. A next generation smart contract and decentralized application platform. https://github.com/ethereum/wiki/wiki/White-Paper, retrieved Apr. 2017.

[67] Ethereum Community. RLPx: Cryptographic network & transport protocol. https://github.com/ethereum/devp2p/blob/master/rlpx.md, retrieved Apr. 2017.

[68] Ethereum Community. RLPx encryption. https://github.com/ethereum/go-ethereum/wiki/RLPx-Encryption, retrieved Apr. 2017.

[69] Ethereum Community. ÐΞVp2p wire protocol. https://github.com/ethereum/wiki/wiki/%C3%90%CE%9EVp2p-Wire-Protocol, retrieved Apr. 2017.

[70] EthereumJ. The Ethereum nodes explorer. https://www.ethernodes.org/, retrieved Jun. 2017.

[71] Etherscan Team. Etherscan: The Ethereum block explorer. https://etherscan.io/, retrieved Jun. 2017.

[72] Ethstats Team. Ethstats. https://ethstats.net/, retrieved Jun. 2017.

[73] Ittay Eyal. The miner's dilemma. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 89–103, 2015.

[74] Ittay Eyal, Ken Birman, and Robbert van Renesse. Cache serializability: Reducing inconsistency in edge transactions. In *Proceedings of the International Conference on Distributed Computing Systems (ICDCS)*, pages 686–695. IEEE, 2015.

[75] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert van Renesse. Bitcoin-NG: A scalable blockchain protocol. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 45–59, Santa Clara, CA, USA, 2016.

[76] Ittay Eyal and Emin Gün Sirer. Bitcoin is broken. http://hackingdistributed.com/2013/11/04/bitcoin-is-broken/, 2013.

[77] Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In *Proceedings of the International Financial Cryptography and Data Security Conference*, Barbados, 2014.

[78] Sebastian Feld, Mirco Schönfeld, and Martin Werner. Analyzing the deployment of Bitcoin's P2P network under an AS-level perspective. *Proceedings of the International Workshop on Secure Peer-to-Peer Intelligent Networks and Systems*, 32:1121–1126, 2014.

[79] José Luis Fernández-Alemán, Inmaculada Carrión Señor, Pedro Ángel Oliver Lozoya, and Ambrosio Toval. Security and privacy in electronic health records: A systematic literature review. *Journal of Biomedical Informatics*, 46(3):541–562, 2013.

[80] Filecoin Team. Filecoin: A cryptocurrency operated file storage network. `http://filecoin.io/filecoin.pdf`, retrieved Jun., 2017, July 2014.

[81] FollowMyVote Team. Follow my vote. `https://followmyvote.com/`, retrieved Oct. 2016.

[82] Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos. The Bitcoin backbone protocol: Analysis and applications. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 281–310. Springer, 2015.

[83] Hector Garcia-Molina. Elections in a distributed computing system. *IEEE Transactions on Computers (TC)*, 100(1):48–59, 1982.

[84] Adem Efe Gencer, Robbert van Renesse, and Emin Gün Sirer. Short paper: Service-oriented sharding for blockchains. In *Proceedings of the International Financial Cryptography and Data Security Conference*, Sliema, Malta, 2017.

[85] Arthur Gervais, Ghassan O Karame, Vedran Capkun, and Srdjan Capkun. Is Bitcoin a decentralized currency? *Proceedings of the IEEE Symposium on Security and Privacy*, 3(12):54–60, 2014.

[86] Arthur Gervais, Ghassan O. Karame, Karl Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 3–16, Vienna, Austria, 2016.

[87] Arthur Gervais, Hubert Ritzdorf, Ghassan O Karame, and Srdjan Capkun. Tampering with the delivery of blocks and transactions in Bitcoin. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 692–705, Denver, CO, USA, 2015.

[88] Gnosis Team. Gnosis. https://gnosis.pm/, retrieved Jun. 2017.

[89] Go-ethereum Authors. Official Go implementation of the Ethereum protocol. https://github.com/ethereum/go-ethereum, retrieved Apr. 2017.

[90] Hashed Health Team. Hashed health. https://hashedhealth.com/, retrieved Jun. 2017.

[91] Mike Hearn and Jeremy Spilman. Rapidly-adjusted (micro)payments to a pre-determined party. https://en.bitcoin.it/wiki/Contract, retrieved Sep. 2015.

[92] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. Eclipse attacks on Bitcoin's peer-to-peer network. In *Proceedings of the USENIX Security Symposium*, pages 129–144, Washington, D.C., USA, 2015.

[93] Stan Higgins. Australia's government publishes blockchain research studies. http://www.coindesk.com/australias-government-publishes-blockchain-research-studies/, retrieved May. 2017.

[94] Stan Higgins. Russian PM orders research on public sector blockchain use. http://web.archive.org/web/20170602111419/http://www.coindesk.com/russian-pm-orders-government-research-public-sector-blockchain-use/, retrieved Jun. 2017.

[95] IBM Corporation. IBM Blockchain on Bluemix. http://www.ibm.com/blockchain/bluemix.html, retrieved Oct. 2016.

[96] Intel Corporation. Sawtooth lake. https://intelledger.github.io/, retrieved Apr. 2017.

[97] IP Info Team. IP Info. http://ipinfo.io/, retrieved Apr. 2017.

[98] jackwinters. Ethpool & Ethermine voting on the soft fork. https://forum.daohub.org/t/ethpool-ethermine-voting-on-the-soft-fork/5364, retrieved Apr. 2017.

[99] Hudson Jameson. Hard fork no. 4: Spurious dragon. https://blog.ethereum.org/2016/11/18/hard-fork-no-4-spurious-dragon/, retrieved Apr. 2017.

[100] Ghassan O. Karame, Elli Androulaki, and Srdjan Capkun. Double-spending fast payments in Bitcoin. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 906–917, 2012.

[101] Garrett Keirns. Japan's Bitcoin law goes into effect tomorrow. `https://web.archive.org/web/20170611014408/http://www.coindesk.com/japan-bitcoin-law-effect-tomorrow/`, retrieved Jun. 2017.

[102] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. Enhancing Bitcoin security and performance with strong consistency via collective signing. *arXiv preprint arXiv:1602.06997*, 2016.

[103] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 839–858, 2016.

[104] Philip Koshy, Diana Koshy, and Patrick McDaniel. An analysis of anonymity in Bitcoin using P2P network traffic. In *Proceedings of the International Financial Cryptography and Data Security Conference*, pages 469–485, Barbados, 2014.

[105] Joshua A. Kroll, Ian C. Davey, and Edward W. Felten. The economics of Bitcoin mining or, Bitcoin in the presence of adversaries. In *Proceedings of the Workshop on the Economics of Information Security (WEIS)*, 2013.

[106] Albert Kwon, David Lazar, Srinivas Devadas, and Bryan Ford. Riffle: An efficient communication system with strong anonymity. *Proceedings of the Privacy Enhancing Technologies Symposium (PETS)*, 2016.

[107] Leslie Lamport. Using time instead of timeout for fault-tolerant distributed systems. *ACM Transactions on Programming Languages and Systems*, 6(2):254–280, April 1984.

[108] Gérard Le Lann. Distributed systems—towards a formal approach. In *IFIP Congress*, volume 7, pages 155–160. Toronto, 1977.

[109] Ledger Assets Pty Ltd. Uproov. `https://uproov.com/`, retrieved Sep. 2016.

[110] Yoad Lewenberg, Yoram Bachrach, Yonatan Sompolinsky, Aviv Zohar, and Jeffrey S Rosenschein. Bitcoin mining pools: A cooperative game theoretic analysis. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 919–927, 2015.

[111] Yoad Lewenberg, Yonatan Sompolinsky, and Aviv Zohar. Inclusive block chain protocols. In *Proceedings of the International Financial Cryptography and Data Security Conference*, Puerto Rico, 2015.

[112] Joshua Lind, Ittay Eyal, Peter Pietzuch, and Emin Gün Sirer. Teechan: Payment channels using trusted execution environments. In *Proceedings of the Workshop on Bitcoin and Blockchain Research (BITCOIN)*, Sliema, Malta, 2017.

[113] Linux Foundation. Hyperledger. `https://hyperledger.org/`, retrieved Sep. 2016.

[114] Litecoin Project. Litecoin, open source P2P digital currency. `https://litecoin.org`, retrieved Nov. 2014.

[115] Andreas Loibl. Namecoin. `namecoin.info`, 2014.

[116] Eric Lombrozo, Johnson Lau, and Pieter Wuille. Segregated witness. BIP 141, `https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki`, retrieved Jun. 2017.

[117] Lua Team. Lua. `http://www.lua.org/`, retrieved Nov. 2016.

[118] Cristian Lumezanu, Randy Baden, Neil Spring, and Bobby Bhattacharjee. Triangle inequality variations in the internet. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 177–183, 2009.

[119] Loi Luu, Viswesh Narayanan, Kunal Baweja, Chaodong Zheng, Seth Gilbert, and Prateek Saxena. A secure sharding protocol for open blockchains. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, Vienna, Austria, 2016.

[120] MaxMind Inc. Maxmind. `https://www.maxmind.com`, retrieved May. 2017.

[121] Petar Maymounkov and David Mazieres. Kademlia: A peer-to-peer infor-

mation system based on the XOR metric. In *Proceedings of the International Workshop on Peer-to-Peer Systems*, pages 53–65. Springer, 2002.

[122] Patrick McCorry, Siamak Shahandashti, and Feng Hao. A smart contract for boardroom voting with maximum voter privacy. In *Proceedings of the International Financial Cryptography and Data Security Conference*, Sliema, Malta, 2017.

[123] Chrissa McFarlane, Michael Beer, Jesse Brown, and Nelson Prendergast. Patientory: A healthcare peer-to-peer EMR storage network v1.1. `https://patientory.com/patientory_whitepaper.pdf`, retrieved Jun. 2017.

[124] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. A fistful of Bitcoins: characterizing payments among men with no names. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 127–140, Barcelona, Spain, 2013. ACM.

[125] Members at Cornell University, Cornell Tech, UC Berkeley, UIUC and the Technion. Initiative for cryptocurrencies and contracts. `http://www.initc3.org/`, retrieved Jun. 2017.

[126] Microsoft. Blockchain-as-a-Service. `https://azure.microsoft.com/en-us/solutions/blockchain/`, retrieved Oct. 2016.

[127] Andrew Miller and Rob Jansen. Shadow-Bitcoin: Scalable simulation via direct execution of multi-threaded applications. *IACR Cryptology ePrint Archive*, 2015:469, 2015.

[128] Andrew Miller and LaViola Joseph J. Jr. Anonymous Byzantine consensus from moderately-hard puzzles: A model for Bitcoin. `https://socrates1024.s3.amazonaws.com/consensus.pdf`, 2009.

[129] Andrew Miller, James Litton, Andrew Pachulski, Neal Gupta, Dave Levin, Neil Spring, and Bobby Bhattacharjee. Discovering Bitcoin's public topology and influential nodes, 2015.

[130] Iulian Moraru, David G. Andersen, and Michael Kaminsky. Egalitarian Paxos. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, 2012.

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

[132] Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. *IACR Cryptology ePrint Archive*, 2015:796, 2015.

[133] Till Neudecker, Philipp Andelfinger, and Hannes Hartenstein. A simulation model for analysis of attacks on the Bitcoin peer-to-peer network. In *International Symposium on Integrated Network Management (IM)*, pages 1327–1332. IEEE, 2015.

[134] OMI. Open music initiative. `http://open-music.org/`, retrieved Oct. 2016.

[135] Omni Team. Omni layer. `http://www.omnilayer.org/`, retrieved Oct. 2016.

[136] Giuseppe Pappalardo, Tiziana Di Matteo, Guido Caldarelli, and Tomaso Aste. Blockchain inefficiency in the Bitcoin peers network. *arXiv preprint arXiv:1704.01414*, 2017.

[137] Parity Authors. Ethereum Rust client. `https://github.com/paritytech/parity`, retrieved Apr. 2017.

[138] Rafael Pass and Abhi Shelat. Micropayments for decentralized currencies. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 207–218, Denver, Colorado, USA, 2015.

[139] Juan Eduardo Pazmiño and Carlo Kleber da Silva Rodrigues. Simply dividing a Bitcoin network node may reduce transaction verification time. *The SIJ Transactions on Computer Networks and Communication Engineering (CNCE)*, 3(2):17–21, February 2015.

[140] Morgen E. Peck. Adam Back says the Bitcoin fork is a coup. `http://spectrum.ieee.org/tech-talk/computing/networks/the-bitcoin-for-is-a-coup`, Aug 2015.

[141] Joseph Poon and Thaddeus Dryja. The Bitcoin Lightning Network. `http://lightning.network/lightning-network.pdf`, February 2015. Draft 0.5.

[142] Nathaniel Popper and Steve Lohr. Blockchain: A better way to track pork chops, bonds, bad peanut butter? *The New York Times*, 04 March 2017.

[143] Pyethapp Authors. Python based client implementing the Ethereum protocol. `https://github.com/ethereum/pyethapp/`, retrieved Apr. 2017.

[144] QuantumMechanic. Proof of stake instead of proof of work. `https://bitcointalk.org`, retrieved Oct. 2016.

[145] Andrew Quentson. Miners vote overwhelmingly in support of Ethereum's hardfork. *Cryptocoins News*, 17 July 2016.

[146] Leonid Reyzin, Dmitry Meshkov, Alexander Chepurnoy, and Sasha Ivanov. Improving authenticated dynamic dictionaries, with applications to cryptocurrencies. In *Proceedings of the International Financial Cryptography and Data Security Conference*, Sliema, Malta, 2017.

[147] Ronald Rivest and Adi Shamir. PayWord and MicroMint: Two simple micropayment schemes. In *Proceedings of the International Workshop on Security Protocols*, pages 69–87. Springer, 1996.

[148] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in Bitcoin. *arXiv preprint arXiv:1507.06183*, 2015.

[149] Fred B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys (CSUR)*, 22(4):299–319, December 1990.

[150] Laura Shin. Republic of Georgia to pilot land titling on blockchain. *Forbes*, 21 April 2016.

[151] Roger W. Sinnott. Virtues of the haversine. *Sky and Telescope*, 1984.

[152] Skuchain Team. Skuchain. `https://skuchain.com/`, retrieved Jun. 2017.

[153] Yonatan Sompolinsky and Aviv Zohar. Accelerating Bitcoin's transaction processing. fast money grows on trees, not chains. In *Proceedings of the International Financial Cryptography and Data Security Conference*, Puerto Rico, 2015.

[154] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in Bitcoin. In *Proceedings of the International Financial Cryptography and Data Security Conference*, pages 507–527, Puerto Rico, 2015.

[155] Chrysoula Stathakopoulou. A faster Bitcoin network. Technical report, ETH, Zürich, January 2015. Semester Thesis, supervised by Christian Decker and Roger Wattenhofer.

[156] Statoshi Team. Statoshi info. `http://statoshi.info/`, retrieved May. 2017.

[157] Daniel Stutzbach and Reza Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 189–202. ACM, 2006.

[158] Eric Swanson. Bitcoin mining calculator. `http://www.alloscomp.com/bitcoin/calculator`, retrieved Sep. 2013.

[159] Martin Swende. Announcement of imminent hard fork for EIP150 gas cost changes. `https://blog.ethereum.org/2016/10/13/announcement-imminent-hard-fork-eip150-gas-cost-changes/`, retrieved Apr. 2017.

[160] Paul Sztorc. Drivechain. `http://www.truthcoin.info/blog/drivechain/`, 2015.

[161] Engen Tham. China turns to blockchain to make markets clearer and cleaner. `http://web.archive.org/web/20170515215252/http://www.reuters.com/article/us-china-fintech-blockchain-idUSKBN15A368`, retrieved May. 2017.

[162] The Bitcoin Community. Release notes, bitcoin 0.12.0. `https://github.com/bitcoin/bitcoin/blob/0.12/doc/release-notes.md`, Jul 2016.

[163] Peter Todd. [bitcoin-development] Tree-chains preliminary summary. `https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2014-March/004797.html`, 2014.

[164] Vivek Vishnumurthy, Sangeeth Chandrakumar, and Emin Gün Sirer. Karma: A secure economic framework for peer-to-peer resource sharing.

In *Proceedings of the ACM SIGCOMM Workshop on the Economics of Peer-to-Peer Systems (P2PECON)*, volume 35, 2003.

[165] Guohui Wang, Bo Zhang, and TS Ng. Towards network triangle inequality violation aware distributed systems. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 175–188. ACM, 2007.

[166] WebAssembly Community Group. WebAssembly. `http://webassembly.org/`, retrieved Nov. 2016.

[167] Wikipedia. List of cryptocurrencies. `https://en.wikipedia.org/wiki/List_of_cryptocurrencies`, retrieved Jun. 2017.

[168] Jeffrey Wilcke. The Ethereum network is currently undergoing a DoS attack. `https://blog.ethereum.org/2016/09/22/ethereum-network-currently-undergoing-dos-attack/`, retrieved Apr. 2017.

[169] Shawn Wilkinson, Tome Boshevski, Josh Brandoff, James Prestwich, Gordon Hall, Patrick Gerbes, Philip Hutchins, Chris Pollard, and Vitalik Buterin. Storj: A peer-to-peer cloud storage network. `https://storj.io/storj.pdf`, retrieved Jun. 2017.

[170] Philipp Winter and Stefan Lindskog. How the great firewall of China is blocking Tor. 2012.

[171] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper*, 2014.

[172] Joseph Young. The Philippines officially legitimize Bitcoin as payment method. `https://web.archive.org/web/20170614065411/https://cointelegraph.com/news/the-philippines-officially-legitimize-bitcoin-as-payment-method`, retrieved Jun. 2017.

[173] Xiao Yue, Huiju Wang, Dawei Jin, Mingqiang Li, and Wei Jiang. Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control. *Journal of medical systems*, 40(10):218, 2016.

[174] Bo Zhang, TS Ng, Animesh Nandi, Rudolf Riedi, Peter Druschel, and Guohui Wang. Measurement based analysis, modeling, and synthesis of

the internet delay space. In *Proceedings of the ACM SIGCOMM Conference on Internet Measurement (IMC)*, pages 85–98. ACM, 2006.

[175] Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town Crier: An authenticated data feed for smart contracts. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 270–282, Vienna, Austria, 2016.

[176] Han Zheng, Eng Keong Lua, Marcelo Pias, and Timothy G Griffin. Internet routing policies and round-trip-times. In *Proceedings of the International Workshop on Passive and Active Network Measurement*, pages 236–250. Springer, 2005.