A Survey on Consensus Mechanisms and Mining Management in Blockchain Networks

Wenbo Wang, *Member, IEEE,* Dinh Thai Hoang, *Member, IEEE,* Zehui Xiong, *Student Member, IEEE,* Dusit Niyato, *Fellow, IEEE,* Ping Wang, *Senior Member, IEEE* Peizhao Hu, *Member, IEEE,* and Yonggang Wen, *Senior Member, IEEE*

***Abstract*—The past decade has witnessed the rapid evolution in blockchain technologies, which has attracted tremendous interests from both the research communities and the industry. The blockchain network was originated in the Internet finical sector as a decentralized, immutable ledger system for transactional data ordering. Nowadays, it is envisioned as a powerful backbone/framework for decentralized data processing and data-driven self-organization in flat, open-access networks. In particular, the plausible characteristics of decentralization, immutability and self-organization are primarily owing to the unique decentralized consensus mechanisms introduced by blockchain networks. This survey is motivated by the lack of a comprehensive literature review on the development of decentralized consensus mechanisms in blockchain networks. In this survey, we provide a systematic vision of the organization of blockchain networks. By emphasizing the unique characteristics of incentivized consensus in blockchain networks, our in-depth review of the state-of-the-art consensus protocols is focused on both the perspective of distributed consensus system design and the perspective of incentive mechanism design. From a game-theoretic point of view, we also provide a thorough review on the strategy adoption for self-organization by the individual nodes in the blockchain backbone networks. Consequently, we provide a comprehensive survey on the emerging applications of the blockchain networks in a wide range of areas. We highlight our special interest in how the consensus mechanisms impact these applications. Finally, we discuss several open issues in the protocol design for blockchain consensus and the related potential research directions.**

***Index Terms*—Blockchain, permissionless consensus, Byzantine fault tolerance, mining, incentive mechanisms, game theory, P2P networks.**

I. Introduction

In the past decade, blockchain networks have gained tremendous popularity for their capabilities of distributively providing immutable ledgers as well as platforms for data-driven autonomous organization. Proposed by the famous grassroot cryptocurrency project "Bitcoin" [1], the blockchain network was originally adopted as the backbone of a public, distributed ledger system to facilitate processing tokenized asset transactions between Peer-to-Peer (P2P) users. Blockchain networks (especially with open access) are distinguished by their inherent characteristics of disintermediation, public accessibility (i.e., transparency) and tamper-resilience [2]. Therefore, they have been hailed as the foundation of various spotlight FinTech applications that impose critical requirement on data security and integrity (e.g., cryptocurrencies[1] [3], [4]). Furthermore, with the distributed consensus provided by blockchain networks, blockchain technologies are also envisaged as the backbone of the emerging open-access, distributed Virtual Machines (VMs) [5] for decentralized, token-driven resource management in communication networks and distributed autonomous systems [6], [7]. For these reasons, blockchain technologies have been heralded by both the industry and academia as the fundamental "game changer" [8] in decentralization of digital infrastructures ranging from the financial industry [4] to the domain of Internet of Things (IoTs) [9].

[1]According to coinmarketcap.com, at the time of writing there are more than 1500 types of cryptocurrencies with a total market value of $320 billion.

Generally, the term "blockchain networks" can be interpreted from two levels, namely, the "blockchains" which refer to a framework of immutable data organization, and the "blockchain networks" on top of which the approaches of data deployment and maintenance are defined. These two aspects are also considered as the major innovation of blockchain technologies. From the perspective of data organization, blockchain technologies employ a number of off-the-shelf cryptographic techniques [10]–[12] and cryptographically associate the users' pseudo-identities with the transactions of their tokenized assets. Thus, it is possible with blockchains to provide the proofs of asset (i.e., token) transfer authentication and thus the proofs of asset ownerships. Furthermore, a blockchain maintains an arbitrary order of the transactional records by cryptographically linking the record subsets in the form of "blocks" to their chronic predecessors. With the help of cryptographic references, any attempt of data tampering can be immediately detected. On the other hand, the consensus protocols in blockchain networks creatively tackle the problem of replicated agreement [13], [14] on transaction recording in an open-access, weakly synchronized network with the tolerance to arbitrarily behaving nodes. Compared with the classical consensus mechanisms, blockchain consensus protocols are able to offer the agreement on the global ledger-data state among a large number of trustless nodes with no identity authentication and much smaller messaging overhead [15]. In this sense, the blockchain can be viewed as a universal memory of the network, and the network can be viewed as a distributed VM comprised by every node therein. Hence, blockchain networks provide a general-purpose platform for executing transaction-driven instructions in decentralized applications such as self-organized network orchestration [16] and P2P resource trading [17].

Although with the rapid evolution in blockchain technologies, the demand for the higher-level quality of services by blockchain-based applications presents more critical challenges in design of blockchain protocols. Particularly, the performance of blockchain networks significantly relies on the performance of the adopted consensus mechanisms, e.g., in terms of data consistency, speed of consensus finality, robustness to arbitrary behaving nodes (i.e., Byzantine nodes [14]) and network scalability. Compared with the classical protocols allowing very limited scalability in distributed systems [14], [18], most of the existing consensus protocols in open-access blockchain networks (e.g., Bitcoin) guarantee the better scalability at the cost of limited processing throughput. Also, to achieve decentralized consensus among poorly synchronized, trustless nodes, a number of these protocols also incur huge requirement on physical resource (e.g., computing power) consumption [3]. Moreover, to ensure high probability of consensus finality, the protocols may also impose higher latency for transaction confirmation. Out of such concerns, a large volume of research has been conducted with the aim of improving the performance of the open-access blockchain consensus protocols in specific aspects. However, in spite of a few short surveys [15], [19], there lacks comprehensive study to summarize the development of the consensus protocols in a uniform framework and consequently the related problems in blockchain networks.

Meanwhile, during the past decade, the capability of blockchain networks has been expanded way further from immutable, tamper-evident distributed ledgers. However, due to the recent market frenzy about cryptocurrencies, most of the existing general reviews/surveys on blockchain technologies emphasize narrowly the scenarios of using blockchain networks as the backbone technologies for cryptocurrencies, especially the market-dominant ones such as Bitcoin and Ethereum [2]–[4], [19]–[23]. For example, the issues regarding the client (user)-side application (i.e., wallet), P2P network protocols, consensus mechanisms and user privacy in the scope of Bitcoin are discussed in [3], [4]. In [20], a brief summary of the emerging blockchain-based applications ranging from finance to IoTs is provided. Further, a systematic survey is provided in [21] with respect to the security issues in Bitcoin networks including the identified attacks on the consensus mechanisms and the privacy/anonymity issues of the Bitcoin clients. In [22], [23], the special issues regarding the design, application and security of the smart contracts are reviewed in the scope of the Ethereum network. In [6], [15], two brief surveys on consensus protocols in blockchain networks are provided.

Compared with the aforementioned surveys and reiviews, our work aims to fill the gap between the rapid-growing works on blockchain networks, especially the consensus protocol-related issues, and the lack of a comprehensive survey on this topic. To distinguish our study from the existing works, we present our survey of the blockchain networks from a perspective of general-purpose distributed systems, which are organized as P2P networks. We perceive the process of blockchain consensus as a process of cooperative state transitions among distributed nodes based on the blockchain-specified data organization framework. We emphasize that such a viewpoint brings the taxonomy of blockchain networks into a paradigm that is comparable with the classical problems of global state maintenance in distributed systems. Therefore, we are able to cast the analysis of blockchain networks into the context of classical fault-tolerant analysis of the standard Agreement-Validity-Termination properties in distributed consensus systems [24]. We also emphasize that the performance of a blockchain consensus protocol heavily depends on the crypto-technique assisted data and network topology organization schemes. After revealing the interconnection between different components of the techniques/protocols for blockchain networks, we start our survey on the research of blockchain consensus protocols from a uniform framework based on Zero-Knowledge Proof (ZKP) systems and then extend to the protocols and related issues beyond it. By focusing on the blockchain protocols for data organization, network organization, and consensus maintenance, our survey contributes in the following aspects:

(1) providing a brief overview on the data organization and network protocols of the blockchain networks;
(2) providing a generic paradigm for the cryptographic consensus mechanisms in open-access blockchain networks,
(3) reviewing the studies on the behaviors of the rational (incentive-driven) nodes in the consensus processes of blockchain networks;
(4) summarizing the concerns and the roadmaps of the consensus protocol design for blockchain networks;
(5) providing an outlook of the research in the emerging decentralized applications built on top of the consensus layer, which may not be limited to the framework of the prevalent blockchain technologies.

The rest of this survey is organized as follows. Section II provides an introductory overview on the protocol organization of blockchain networks. Section III provides an in-depth survey on the state-of-the-art consensus protocol design of open-access blockchain networks. Consequently, Section IV provides a survey on the studies of the rational nodes' strategies in these consensus processes and their impact on the performance of blockchain networks. In Section V, we further extend our survey on blockchain consensus protocols to the ermerging fields including virtual block mining-mechanism and hybrid consensus. Section VI briefly reviews the emerging cross-layer design regarding the data organization and consensus protocols, namely, the "next-generation blockchains" which may have different roadmaps for scalability and performance other than the prevalent blockchain paradigm. Section VI also provides a short review of the emerging applications of blockchains as well as an outlook of the potential research directions in the context of wireless networks. Section VII concludes this survey by summarizing the contributions.

II. Protocol Overview and Preliminaries

A. Overview of Blockchain Network Protocols

From the perspective of system design, a blockchain network can be abstracted into four implementation levels from the bottom up. These levels are the protocols of data and network organization, the protocols of distributed consensus, the framework of autonomous organization based on distributed VMs (e.g., smart contracts [7]) and the implementation of human-machine interfaces. Following the approach of protocol layer definition in the Open Systems Interconnection (OSI) model, we provide in Figure 1 an overview of these layers in blockchain networks and the related ingredient technologies.

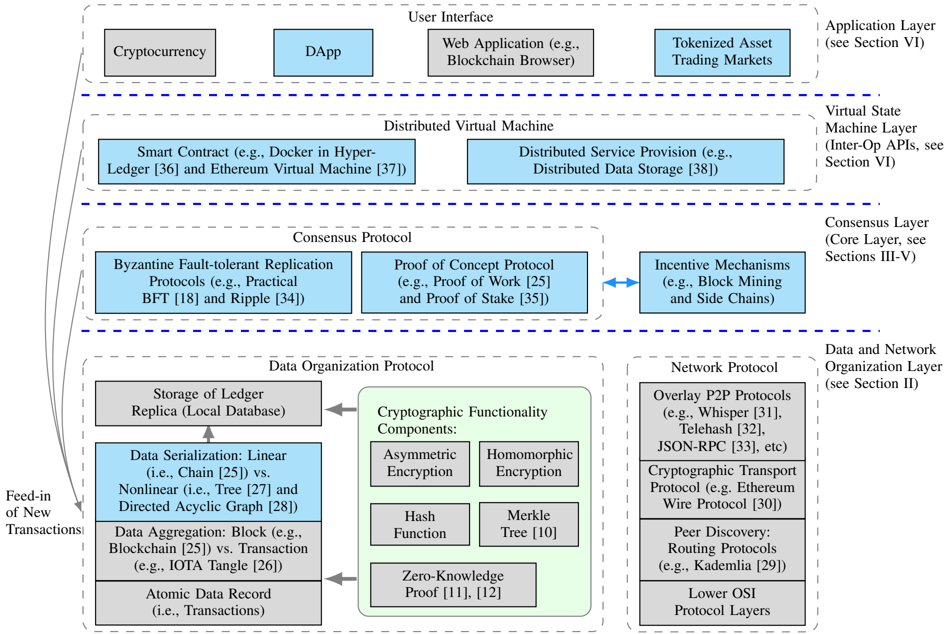


Figure 1. An overview of the blockchain network implementation stacks.

The data organization protocols provide a number of ingredient cryptographic functionalities [10]–[12] to form unique and secured node identities in a blockchain network. The protocols also define the approaches to establish the cryptographic dependence among all the records, e.g., transaction records and account balances, in a local ledger replica for ordering and tamper proof. From the perspective of data representation, the term "blockchain" is named as such mainly for historical reason. In the scope of first-generation blockchain networks, e.g., Bitcoin [1], the digitally signed transactional records are arbitrarily "packed up" into a cryptographically tamper-evident data structure known as the "block". The blocks are then organized in a chronological order as a "chain of blocks", or more precisely, a linear list of blocks linked by tamper-evident hash pointers. Nevertheless, in order to improve the processing efficiency, network scalability and security, the linear data organization framework has been expanded into the nonlinear forms such as trees and graphs of blocks [28], [39] or even block-less, nonlinear data structures [26]. Despite the different forms of block organization, cryptographic data representation provides the fundamental protection of privacy and data integrity for blockchain networks. When compared with conventional database, it also provides more efficient on-chain storage without harming the data integrity.

On the other hand, the network protocols provide the means of P2P network organization, namely, peer/route discovery and maintenance as well as encrypted data transmission/synchronization over P2P links. On top of the reliable data synchronization over the P2P connections, the consensus layer provides the core functionality to maintain the originality, consistency and order of the blockchain data across the network. From the perspective of distributed system design, the consensus protocols provide Byzantine agreement [14] in blockchain networks. More specifically, the nodes in the network expect to agree on a common update, i.e., consensus, of the blockchain state that they copy as the local replicas even in the presence of possible conflicting inputs and arbitrary faulty (Byzantine) behaviors of some nodes. When choosing the permissoned access-control schemes, blockchain networks usually adopt the well-studied Byzantine Faulty-Tolerant (BFT) consensus protocols such as Practical BFT (PBFT) [18] for reaching the consensus among a small group of authenticated nodes (e.g., HyperLedger Fabric [36]). On the contrary, in open-access/permissionless blockchain networks, probabilistic Byzantine agreement is achieved based on the combination of techniques including cryptographic Zero-Knowledge Proof (ZKP) [11], [12] and incentive mechanism design. As pointed out in [19], the permissioned consensus protocols rely on a semi-centralized consensus framework and a higher messaging overhead to provide immediate consensus finality and thus high transaction processing throughput. In contrast, the permissionless consensus protocols are more appropriate for a blockchain network with loose control on the synchronization and behaviors of the nodes. In the condition of bounded delay and honest majority, the permissionless consensus protocols provide significantly better support for scalability at the cost of a lower processing efficiency.

Provided that the robustness of the consensus protocols is guaranteed, the smart contracts are implemented on the distributed VM layer. In brief, the distributed VM layer abstracts the details of data organization, information propagation and consensus formation in the blockchain network. As the interoperation layer between the lower-layer protocols and the applications, the VM layer exposes the necessary APIs to the application layer as if the distributed computation were executed in a local virtual runtime environment on a single machine. When the functionalities of VMs are enabled, a node in the network is allowed to deploy onto a blockchain the smart contract in the form of autonomously executable procedures. Further, by controlling the exposure of APIs and the allowed state size of the VM, the smart contract protocol of a blockchain is able to adjust its level of Turing-completeness ranging from stateless circuits, e.g., Bitcoin [1] to fully Turing-complete state machines, e.g., Ethereum [37] and HyperLedger Fabric [36]. With full Turing-completeness, blockchain networks are enabled to perform general-purpose computation in a decentralized manner. For this reason, a blockchain network is not only able to provide the services of distributed, trusted data recording and timestamping, but also able to facilitate the functionalities of general-purpose autonomous organization. Therefore, blockchain networks are appropriate to work as the backbone of an autonomous organization system for managing data or transaction-driven interactions among the decentralized entities in the network. On top of the VM layer, the application layer provides the end-user-visible interfaces such as Distributed Applications (DApps) [17], [40], cryptocurrencies and blockchain browsers.

B. Cryptographic Data Organization

When viewed as a data structure, a blockchain can be abstracted as an infinitely-growing, append-only string that is canonically agreed upon by the nodes in the blockchain network [25]. For data organization, the local blockchain replica of each node is organized in a hierarchical data structure of three levels, namely, the transactions, the blocks and the chain. Each level requires a different set of cryptographic functionalities for the protection of data integrity and authenticity.

1) Transactions, Addresses and Signatures: Transactions are the atomic data structure of a blockchain. Generally, a transaction is created by a set of users or autonomous objects (i.e., smart contracts) to indicate the transfer of tokens from the senders to the specified receivers. A transaction specifies a possibly empty list of inputs associating the token values with the identities (i.e., addresses) of the sending users/objects. It also specifies a nonempty list of outputs designating the redistribution result of the input tokens among the associated identities of the receivers. A transaction can be considered as a static record showing the identities of the senders and the receivers, the token value to be redistributed and the state of token reception. To protect the authenticity of a transaction record, the functionalities of cryptographic hashing and asymmetric encryption are activated:

- *Hash Function*: A cryptographic hash function maps at random an arbitrary-length binary input to a unique, fixed-length binary output (i.e., image). With a secure hash function (e.g., SHA-256), it is computationally infeasible to recover the input from the output image. Also, the probability to generate the same output for any two different inputs is negligible.
- *Asymmetric Key*: Each node in the blockchain network generates a pair of private and public keys. The private key is associated with a digital signature function, which outputs a fixed-length signature string for any arbitrary-length input message. The public key is associated with a verification function, which takes as input the same message and the acclaimed signature for that message. The verification function only returns *true* when the signature is generated by the signature function with the corresponding private key and the input message.

The nodes in the network or the autonomous objects identify themselves by revealing their public keys, namely, the hashcode of their public keys, as their permanent addresses on the blockchain[2]. Since each input tuple in a transaction is signed by the associated sending account, the network is able to publicly validate the authenticity of the input through verifying the signature based on the sender's public address.

2) Block Organization, Hash Pointer and Merkle Tree: A block is a container of an arbitrary subset of transaction records and can only be created by a node participating in the consensus process. To protect the integrity of the transaction records and to specify the ordering of adjacent blocks in a consensus node's local view, a data field known as the hash pointer is kept in the block's data structure. In addition, to reduce the on-chain storage, the cryptographic data structure of Merkle tree is also enabled to generate the tamper-evident digest in the transaction set of a block (see Figure 2):

- *hash pointer*: A hash pointer to a block is the hashcode of the concatenated data fields in that block. The hashcode of the current block is stored as the header of that block. The hashcodes of the reference blocks are stored as the hash pointers of a block to indicate that at the local view, the block recognizes that the transactions in the reference blocks are created earlier than those in the current block.
- *Merkle Tree* [10]: A Merkle tree represents a transaction set in the form of a binary tree. Therein, each leaf is labeled with the hashcode of a transaction and a non-leaf nodes is labeled with the hashcode of the concatenated labels of its two child nodes. The root node of the Merkle tree is known as the Merkle digest/root. A block storing only the Merkle root of the selected transactions is known to be in a lightweight form, which is sufficient for quick validation and synchronization. When using the lightweight-form storage, the node has to query its peers to retrieve the complete transaction records in the blocks.

In addition to the Merkle digest, block header and the hash pointers, a block may also contain some auxiliary data fields, whose definition varies with the adopted protocol of block generation based on different consensus schemes. At a local view of the blockchain, the blocks are organized based on the hash pointers to their references/predecessors. Every blockchain admits a unique block with no reference as the "genesis block", namely, the common ancestor block of all valid blocks in the chain. According to the number of hash pointers to the predecessors that are allowed to be kept by a block, the block organization can vary from a linear linked list (e.g., blockchains) to a Directed Acyclic Graph (DAG) (e.g., SPECTRE [28]). Without of specification, we limit most of our discussion on blockchains to the linear-list case, where the total order of the blocks is guaranteed (see Figure 2).

[2]Some cryptocurrency systems (e.g., Monero [41] and Zcash [42]) incorporate cryptographic techniques such as one-time signature and group signature to create ephemeral addresses for enhancing anonymity.

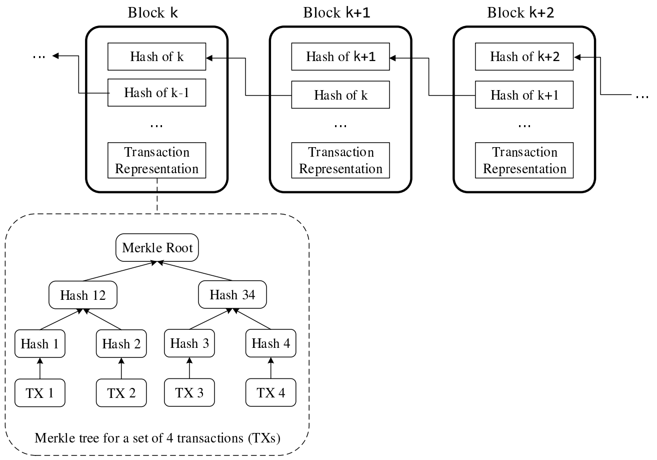


Figure 2. Illustration of a chain of blocks, where the transactions in a single block is represented by a Merkle root.

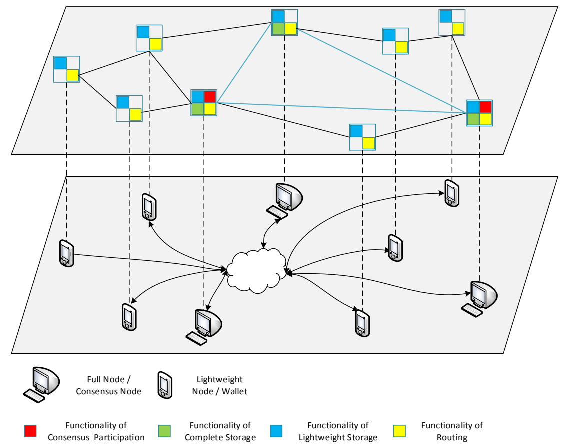


Figure 3. Illustration of different roles of the nodes in an open-access blockchain network.

C. Blockchain Networks

In a Byzantine environment, the identity management mechanism plays a key role in determining how the nodes in a blockchain network are organized. In an open-access (i.e., public/permissionless) blockchain network, a node can freely join the network and activate any available network functionalities. Without any authentication scheme, the nodes are organized as overlay P2P networks. Comparatively, in a consortium (i.e., permissioned) blockchain network, only the authorized nodes are allowed to enable the core functionalities such as consensus participation or data propagation. The authorized nodes may be organized in different topologies, e.g., fully connected networks or P2P networks, according to the consensus protocols that the networks adopt. In this paper, we will mainly focus on the network protocols in the open-access cases.

In open-access blockchain networks, the main goal of the network protocol is to induce a random topology among the nodes and propagate information efficiently for blockchain replica synchronization. Most of the existing blockchain networks employ the ready-to-use P2P protocols with slight modification for topology formation and data communication. For peer discovery and topology maintenance, the nodes in Bitcoin-like blockchain networks rely on querying a hard-coded set of volunteer DNS servers, which return a random set of bootstrapping nodes' IP addresses for the new nodes to initialize their peer lists [43], [44]. Nodes then request or advertise addresses based on these lists. In contrast, the Ethereum-like networks adopt a Kademlia-inspired protocol based on Distributed Hash Tables (DHTs) [29] for peer/route discovery through UDP connections. In blockchain networks, the connection of a node to a peer is managed based on reputation using a penalty score. A node will increase the penalty score of the peer sending malformed messages until the IP address of the faulty node is locally banned [30], [44].

To replicate the blockchain over all nodes in the network, the messages of transactions and blocks are "broadcast" through flooding the P2P links in a gossip-like manner. Typically, a P2P link in blockchain networks is built upon a persistent TCP connection after a protocol-level three-way handshake, which exchanges the replica state and the protocol/software version of each node [30], [45]. After the connections to the peer nodes are established, another three-way handshake occurs for a node to exchange new transactions/blocks with its neighbors. The node first notifies its peers with the hashcode of the new transactions/blocks that it receives or generates. Then, the peers reply with the data-transfer request specifying the hashcode of the information that they need. Upon request, the transfer of transactions/blocks is done via individual transfer messages[3]. The data transfer in blockchain networks is typically implemented based on the HTTP(s)-based Remote Procedure Call (RPC) protocol, where the messages are serialized following the JSON protocol [30].

An open-access blockchain network does not explicitly specify the role of each node. Nevertheless, according to the enabled functionalities, the nodes in the network can be categorized as the lightweight nodes, the full nodes and the consensus nodes [46]. Basically, all nodes are required to enable the routing functionality for message verification/propagation and connection maintenance. A lightweight node (e.g., wallets) only keeps the header of each block in its local storage. A full node stores locally a complete and up-to-date replica of the canonical blockchain. Compared with the lightweight nodes, a full node is able to autonomously verify the transactions without external reference. A consensus node enables the functionality of consensus participation. Therefore, it is able to publish new blocks and has a chance to influence the state of the canonical blockchain. A consensus node can adopt either complete storage or lightweight storage. In Figure 3, we present an example of different node types in a public blockchain network. It is worth noting that different roles of the nodes lead to the inconsistency in their interests. Namely, the transaction-issuing nodes (e.g., lightweight nodes) may not be the transaction-approving nodes (i.e., consensus nodes). For this reason, caution needs to be taken in protocol design to ensure that the consensus nodes act on behalf of the others in a trustless environment, especially on the consensus layer.

D. Consensus in Blockchain Networks

In the context of distributed system, maintaining the canonical blockchain state across the P2P network can be mapped

[3]For example, the details of handshake and synchronization in the Ethereum network are defined in the DEVp2p Wire Protocol [30].

as a fault-tolerant state-machine replication problem [13]. In other words, an agreement (i.e., consensus) on a unique common view of the blockchain is expected to be achieved by the consensus nodes in the condition of Byzantine failures (see [14], [18] for the formal definition). In blockchain networks, Byzantine failures cause faulty nodes to exhibit arbitrary behaviors including malicious attacks/collusions (e.g., Sybil attacks [47] and double-spending attacks [21]), node mistakes (e.g., unexpected blockchain fork due to software inconsistency [48]) and connection errors. For ease of exposition, we roughly consider that the sequence of blocks represents the blockchain state, and the confirmation of a transaction incurs a blockchain state transition. According to [13], [49], a blockchain updating protocol is said to achieve the consensus (also known as atomic broadcast[4] [13], [50], [51]) in a Byzantine environment if the following properties are satisfied [15]:

- *Validity*: If all the honest nodes activated on a common blockchain state propose to expand the blockchain by the same block, any honest node transiting to a new local replica state adopts the blockchain headed by that block.
- *Agreement*: If an honest node confirms a new block header, then any honest node that updates its local blockchain view will update with that new block header.
- *Liveness*: All transactions originated from the honest nodes will be eventually confirmed.
- *Total order*: All honest nodes accept the same order of transactions as long as they are confirmed in their local blockchain views.

The consensus protocols vary with different blockchain networks. Since the permissioned blockchain networks admit tighter control on the synchronization among consensus nodes, they may adopt the conventional Byzantine Fault-Tolerant (BFT) protocols (c.f., the primitive algorithms described in [52], [53]) to provide the required consensus properties. A typical implementation of such protocols can be found in the Ripple network [34], where a group of synchronized Ripple servers perform blockchain expansion through a voting mechanism. Further, if an external oracle is introduced to designate the primary node for block generation (e.g., with HyperLedger Fabric [36]), the Practical BFT (PBFT) protocol [18] can be adopted to implement a three-phase commit scheme for blockchain expansion. In a network of N consensus nodes, the BFT-based protocols are able to conditionally tolerate $\lfloor\frac{N-1}{5}\rfloor$ (e.g., [34]) to $\lfloor\frac{N-1}{2}\rfloor$ (e.g., [54]) faulty nodes.

On the contrary, permissionless blockchain networks admit no identity authentication or explicit synchronization schemes. Therefore, the consensus protocol therein is expected to be well scalable and tolerant to pseudo identities and poor synchronization. Since any node is able to propose the state transition with its own candidate block for the blockchain header, the primary goal of the consensus protocol in permissionless networks is to ensure that every consensus node adheres to the "longest chain rule" [3]. Namely, when the blocks are organized in a linked list, at any time instance, only the longest chain can be accepted as the canonical state of the blockchain. Due to the lack of identity authentication, the direct voting-based BFT protocols no longer ensure the consensus properties in permissionless blockchain networks. Instead, the incentive-based consensus schemes such as the Nakamoto consensus protocol [1] are widely adopted.

[4]Here, the semantic of "broadcast" is consistent with that in the context of distributed system/database. Namely, a message is atomically broadcast when it is either received by every nonfaulty nodes, or by none at all.

E. Nakamoto Consensus Protocol and Incentive Compatibility

To jointly address the problems of pseudonymity, scalability and poor synchronization, Nakamoto proposed in [1] a permissionless consensus protocol based on a framework of cryptographic block-discovery racing game. This is also known as the Proof of Work (PoW) scheme [2], [3]. From a single node's perspective, the Nakamoto consensus protocol defines three major procedures, namely, the procedure of chain validation, the procedure of chain comparison and extension and the procedure of PoW solution searching [25]. The chain validation predicate performs a validation of the structural properties of a given chain. It checks that each block in the chain provides valid PoW solution and no conflict between transactions exists. The function of chain comparison and extension compares the length of a set of chains, which may be either received from peer nodes or locally proposed. It guarantees that an honest node only adopts the longest proposal among the candidate blockchain views. The function of PoW solution searching is the main "workhorse" of the protocol and defines a cryptographic puzzle solution procedure in a computation-intensive manner.

In brief, PoW solution requires exhaustively querying a cryptographic hash function for a partial preimage generated from a candidate block, whose hashcode satisfies a pre-defined condition. For simplicity of exposition, let $\mathcal{H}(\cdot)$ denote the hash function and x denote the binary string assembled based on the candidate block data including the set of transactions (e.g., Merkle root), the reference hash pointers, etc. Then, we can formally define the PoW puzzle and solution as follows:

Definition 1. *Given an adjustable hardness condition parameter h, the process of PoW puzzle solution aims to search for a solution string, $nonce$, such that for a given string x assembled based on the candidate block data, the hashcode (i.e, the target block header bh) of the concatenation of x and $nonce$ is smaller than a target value $D(h)$:*

$$bh = \mathcal{H}(x\|nonce) \le D(h), \tag{1}$$

where for some fixed length of bits L, $D(h) = 2^{L-h}$.

The Nakamoto protocol is computation-intensive since to win the puzzle solving race, a node needs to achieve a hash querying rate as high as possible. This property financially prevents the Sybil attacks of malicious nodes by merely creating multiple pseudo identities. On the other hand, the economic cost (mainly electricity consumption) also renders it impractical for any node to voluntarily participate the consensus process at a consistent economic loss. To ensure proper functioning of a permissionless blockchain network, the Nakamoto protocol introduces incentives to probabilistically award the consensus participants based on an embedded

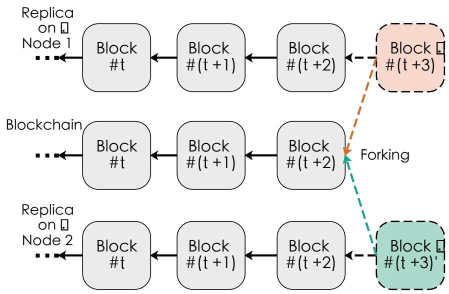


Figure 4. A (temporary) fork happens between nodes 1 and 2 when their local PoW processes lead to different proposals of the new blockchain header, i.e., $(t+3)$ and $(t+3)'$ at the same time. Both $(t+3)$ and $(t+3)'$ satisfy (1).

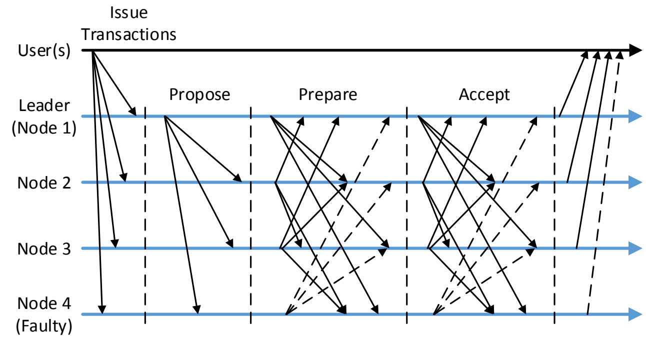


Figure 5. PBFT-based message pattern of three-way handshake in permissioned blockchains, e.g., Hyperledger Fabric [66]. The message is formed based on the granularity level of blocks, i.e., a batch of transactions.

mechanism of token supply and transaction tipping [1]. From a game theoretic point of view, an implicit assumption adopted by the Nakamoto consensus protocol is that all the participant nodes are individually rational [55]. In return, the consensus mechanism is expected to be *incentive compatible*. In other words, the consensus protocol should ensure that any consensus node will suffer from finical loss whenever it deviates from truthfully following the protocol.

However, the incentive compatibility of the Nakamoto protocol has been openly questioned [56]–[59]. Since the Nakamoto protocol allows nodes to propose arbitrary blocks from their local pending transaction set, it is inevitable for the network to experience blockchain expansion race with a (temporary) split, i.e., fork, in the local views of the blockchain state [3], [21] (see Figure 4). To guarantee the consensus properties and thus convergence to one canonical blockchain state, the Nakamoto protocol relies on the assumption that the majority of the consensus nodes follow the longest chain rule and are altruistic in information forwarding. It has been discovered in [56], [60] that rational consensus nodes may not have incentive for transaction/block propagation. As a result, the problem of blockchain forking may not be easily resolved in the current framework of the Nakamoto protocol. Special measures should be further taken in the protocol design, and a set of folklore principles has been suggested to gear the consensus mechanism towards a protocol for secured and sustainable blockchain networks [4], [61], [62]:

- The consensus mechanism should enforce that propagating information and extending the longest chain of block are the monotonic strategies of the consensus nodes [62]. In other words, all the sub-stages in the consensus process should be incentive-compatible in an open environment with the tolerance to Byzantine and unfaithful faults.
- The consensus mechanism should encourage decentralization and fairness. Namely, it should not only discourage coalition, e.g., botnets and mining pools [25], [63], but also make the consensus process an uneasy prey of the adversaries with cumulated computation power.
- The consensus mechanism should strike a proper balance between processing throughput and network scalability [51], [64].

III. Distributed Consensus Mechanisms Based on Proof of Concepts

A. Permissionless Consensus via Zero-Knowledge Proofs

For traditional BFT consensus protocols, e.g., Byzantine Paxos [65] and PBFT [18], it is generally necessary to assume a fully connected topology among the consensus nodes as well as a leader-peer hierarchy for block proposal. The BFT consensus process is organized explicitly in rounds of three-way handshakes, thus synchronization between nodes with bounded execution time and message latency is also required. As illustrated in Figure 5, only the leader is responsible for proposing new blocks to a consortium of peer nodes at the proposal (pre-prepare) phase. This is followed by two all-to-all messaging phases, where a peer node only accepts the proposal (i.e., commit) when it receives more than a certain number of proposal approvals from the other peers ($\lfloor \frac{n+f+1}{3} \rfloor$ with PBFT for a network of n honest nodes and f Byzantine nodes). These classical state-machine replication approaches guarantee the properties of deterministic agreement and liveness in Byzantine environment, and are well-known for their low processing latency [19]. However, the characteristics of leader-peer hierarchy and high communication complexity in $\Theta(n^2)$ [65] naturally require the BFT-based blockchain consensus protocols to be implemented in a small-scale permissioned network with centralized admission control. In order to achieve full decentralization and high consensus scalability, alternative approaches such as Nakamoto protocols become critical in the design of blockchain's consensus layer.

According to our discussion in Section II-E, the primary functionality of PoW in the Nakamoto protocol is to simulate the leader election in the traditional BFT protocols. The PoW process abstracted by Definition 1 is essentially a verifiable process of weighted random coin-tossing, where the probability of winning is no longer uniformly associated with the nodes' identities but in proportion to the resources, e.g., hashrate casted by the nodes. Then, we can consider that each new block is generated by a time-independent "lottery", where the probability of being elected as the leader for block proposal depends on the ratio between the casted resource of a node (or a node coalition) and the total resources presented in the entire network. Let w_i denote the resource held by node i in a network of node set $\mathcal{N}$, then, the probability of node i winning the leader-election in a PoW-like process should follow:

$$\Pr_i^{\text{win}} = \frac{w_i}{\sum_{j\in\mathcal{N}} w_j}, \tag{2}$$

where w_i generalizes the share of any verifiable resource such as computational power [1], memory [42], storage [67], etc. In contrast to the BFT protocols, the peer nodes accept the received block proposal following the longest-chain-rule after they verify the validity of the block and the transactions therein. Therefore, no all-to-all messaging phase is needed and the Nakamoto protocol may have a much smaller message complexity $\Omega(n)$ when the majority of the peers are honest [52].

As the core component of the Nakamoto protocol, the PoW scheme originates from the idea of indirectly validating nodes' identities in pseudonymous P2P networks through an identity pricing mechanism [68], [69]. More specifically, the PoW scheme described by Definition 1 is originally designed to measure the voting power or the trustworthiness of a node according to the constrained resources presented by the node in the P2P network. Thus, the tolerable fraction of Byzantine nodes in BFT protocols is replaced by a limited fraction of the total computational power of the network [69]. Compared with the original design, the PoW scheme in blockchain networks is no longer used for direct identity verification between peers. Instead, the PoW processes of all the nodes in a blockchain network are expected to collectively simulate a publicly verifiable random function to elect the leader of block proposal following the distribution given by (2). Based on such a design paradigm, PoW can be generalized into the framework of Proof-of-Concepts (PoX) (cf. [3]). With PoX, the nodes in the network are required to non-interactively prove the possession or commitment of certain measurable resources beyond hashrates in PoW. Furthermore, their collective behavior should also yield a stochastic process for leader assignment following the distribution given in (2).

From a network-level perspective, PoX generally relies on a pseudorandom oracle to provide the property of verifiable unpredictability. It also needs to implement a one-way cryptographic puzzle for resource proof in the framework of non-interactive ZKP systems[5]. Especially, a complexity gap is expected such that the puzzle is easy to verify (in polynomial-time) but (moderately) hard for adversaries to invert/solve [70]. From a single node's perspective, the adopted puzzle also has to satisfy the basic soundness and completeness properties of ZKP [11], [12]. Namely, an invalid proof should always be rejected by nonfaulty verifying nodes while a valid proof should always be accepted by nonfaulty verifiers. Furthermore, since permissionless blockchain networks do not prevent non-leader nodes from publishing invalid block-header proposals, an interactive ZKP scheme with verifier-designated challenges and multiple rounds of execution-verification stages will lead to excessive communication overhead. This is the major reason for requiring a non-interactive puzzle to be designed for permissionless blockchain networks. Following the generation-computation-verification paradigm of non-interactive ZKP (cf. the verifiable random function defined in [71]), we can abstract a PoX process into the three stages described in Table I.

[5]A node is non-interactive when it can only choose between publishing messages to the network and remaining passive. Otherwise it is interactive.

Table I
THREE-STAGE ABSTRACTION OF A POX PROCESS

Initialization (generator of random seed or keys)	The *initialization stage* provides the prover and the verifier the necessary information to run in subsequent stages according to the PoX specifications. Typical non-interactive ZKP systems, e.g., zk-SNARK [72] have to query a trusted third-party key/random seed generation protocol to produce a common reference string for both the prover and the verifier.
Execution (challenge and proof generator)	For non-interactive ZKP, the *execution stage* requires the prover to generate according to the common reference string a random challenge that constitutes a self-contained, uncompromisable computational problem, namely, the puzzle. Meanwhile, a corresponding proof (a.k.a. witness or puzzle solution) is also generated.
Verification	In the *verification stage*, a verifier checks about the proof's correctness, which is determined solely based on the information issued by the prover.

With the paradigm of PoX described above, we are now ready to investigate the puzzle design problem for different PoX schemes, which can be seen as modification or extension to the existing PoW-based Nakamoto protocol (see [38], [73]–[76] for examples). Since a trusted third party does not exist in a permissionless blockchain network, special caution should be taken in the puzzle design such that the freshness of the puzzle is guaranteed at the execution stage. Namely, the puzzle solution is unpredictable and the proof is non-reusable. Theoretical analyses of blockchain networks, e.g., [74] may assume such a property on the condition that the network has access to a universal random sampler (a.k.a., oracle) or an idea randomness beacon. Nevertheless, due to full decentralization of the permissionless blockchain networks, a case-by-case study for different PoX schemes is usually needed for practical implementation of the random oracle in order to prevent puzzle grinding and leader election manipulation. Apart from the aforementioned properties of non-invertibility, completeness, soundness and freshness, the other requirements for puzzle design in PoX may include but are not limited to the following:

- The puzzle should be resistant to either the aggregation [77] or the outsourcing [78] of the computational resources.
- The puzzle-solving process should be eco-friendly [35], [73], [75], [76], [79].
- In addition to providing incentive based on resource pricing mechanism, the puzzle-solving process should provide useful services in the meanwhile [38], [80].

B. Nakamoto Protocol Based on Primitive Proof of Work

As we have pointed out in the previous discussion, the primitive PoW scheme proposed in [1] works to financially disincentivize the Sybils attack on block proposal and maintains a biased random leader election process in proportion to the hashrate casted by each node. Recall that the input string x to the PoW puzzle is a concatenation of the previous block's hash pointer and the payload data of the proposed block. For the puzzle design of PoW, the reason of choosing the hash function $\mathcal{H}(\cdot)$ in (1), e.g., SHA-256 in practice lies in the fact that a hash function is computationally indistinguishable from a pseudorandom function, if it preserves the properties of collision resistance[6] and pre-image resistance [81]. Since the random output of $\mathcal{H}(\cdot)$ is time-independent and only determined by the input string, it plays the role of an uncompromisable random oracle and outputs a unique, unpredictable result every time when it is queried with a different x [82]. This means that a node in the blockchain network is able to construct a fresh random challenge solely based on its block proposal without referring to any designated verifier or third-party initializer. Meanwhile, it is well-known that with a proper cryptographic hash function, the search for a preimage $(x, nonce)$ satisfying the condition $\mathcal{H}(x\|nonce) \leq 2^{L-h}$ in (1) cannot be more efficient than exhaustively querying the random oracle for all $nonce \in [0, 2^L]$. This leads to a puzzle time complexity of $\mathcal{O}(2^h)$ [61]. On the other hand, verifying the puzzle only requires a single hash query. Therefore, the properties of non-invertibility, completeness, soundness and freshness are all satisfied by the PoW puzzle given by Definition 1.

For a given difficulty level $D(h)$ in (1), each single query to $\mathcal{H}(\cdot)$ is an i.i.d. Bernoulli trial with a success probability

$$\Pr\left(y : \mathcal{H}(x\|y) \leq D(h)\right) = 2^{-h}. \tag{3}$$

We adopt the typical assumption of loosely network synchronization for analyzing PoW-based blockchains [25], [82]. Namely, all messages are delivered with bounded delay in one round. Then, (3) indicates that the frequency for a node to obtain the puzzle solutions during a certain number of loosely synchronized rounds is a Bernoulli process. Since the probability given in (3) is negligible for a sufficiently large h with cryptographic hash functions $\mathcal{H}(\cdot)$, the Bernoulli process of node i converges to a Poisson process as the time interval between queries/trails shrinks [52].

To analyze the PoW scheme, let w_i in (2) refer to the number of queries that node i can make to $\mathcal{H}(\cdot)$ in a single round. Then, we can approximate the rate of the Poisson process for node i's puzzle solution by $\lambda_i = w_i/2^h$ [83]. Note that every node in the network is running an independent puzzle-solving process. Since a combination of N independent Poisson processes is still a Poisson process, then, the collective PoW process of a network with N nodes has a rate

$$\lambda = \sum_{i=1}^{N} \lambda_i = \frac{\sum_{i=1}^{N} w_i}{2^h}. \tag{4}$$

The property of the combined Poisson processes in (4) leads to the probability distribution for leader election in (2). From a single node's perspective, the repeated PoW puzzle-solving processes take the form of a block-proposal competition across the network. From the perspective of the network, for a given difficulty level $D(h)$, this puzzle-solving race simulates a verifiable random function for leader election and guarantees to follow the distribution in (2). Most importantly, it tolerates any fraction of the Byzantine nodes in the network.

Nevertheless, the PoW by itself cannot guarantee any of the principle Byzantine consensus properties as described in Section II-D. On top of the designed PoW puzzle and the P2P information diffusion functionality, three external functions are abstracted in [25] to describe the Nakamoto cnesensus protocol from a single node's perspective. These functions are

1) the *chain reading function* that receives as input a blockchain and outputs an interpretation for later use;
2) the *content validation function* that validates a blockchain replica and checks the data consistency with the applications (e.g., Bitcoin) on top of the blockchain;
3) the *input contribution function* that compares the local and the received views of the blockchain and adopts the "best" one following the rule of longest chain.

The input contribution function realizes the puzzle execution stage and the content validation function realizes the puzzle verification stage in Table I. Due to the independent Poisson processes in the block-proposal competition, more than one node may propose to extend the blockchain using different blocks with corresponding valid PoW solutions at the same time. As a result, the nodes may read from the network multiple valid views of the blockchain and choose different forks as their "best" local views (see also Figure 4). Theoretically, it has been shown in [84] that deterministic consensus in permissionless blockchain networks cannot be guaranteed unless all non-faulty nodes are reachable from one to another and the number of consensus nodes is known. For this reason, in [25], [82], [85], Garay et al. propose to capture the properties of validity, agreement and liveness of the Nakamoto consensus protocol by the three chain-based properties in Table II. Then, the PoW-based Nakamoto protocol can be modeled as a probabilistic Byzantine agreement protocol.

Table II
THREE PROPERTIES OF NAKAMOTO PROTOCOLS FOR BLOCKCHAINS

Nakamoto Protocol-Specified Properties	Corresponding Properties of Byzantine Agreement	Explanation in Details
Common-prefix property	Agreement (and permanent order)	In the condition of multiple local blockchain views due to forking, the *common-prefix property* indicates that after cutting off (pruning) a certain number of block from the end (header) of the local chain, an honest node will always obtain a sub-chain that is a prefix of another honest node's local view of the blockchain.
Chain-quality property	Validity	Among a given length of consequent blocks in the local blockchain view of an honest node, the number of blocks that is proposed by Byzantine nodes (adversaries) is upper-bounded.
Chain-growth property	Liveness	For any given rounds of block proposals, the number of blocks appended to the local view of any honest node is lower-bounded.

In order to quantify the Byzantine agreement properties for blockchains, three conditions, i.e., the upper-bounded information diffusion delay, a "flat network" with equal and limited hashrates and the upper-bounded number of Byzantine nodes are assumed in [25], [82], [85]. It is shown in [25] that the three properties in Table II are quantified by three parameters, namely, the collective hashrates of the honest nodes, the hashrate controlled by the adversaries and the expected block arrival rate of the network-level Poisson process given in (4). It has been further proved in [25] that under the condition of honest majority, the basic properties of validity and agreement are satisfied by the Nakamoto protocol with overwhelming

[6]The collision probability of $\mathcal{H}(\cdot)$ is $e^{-\Omega(L)}$ and thus negligible [25].

probability. Furthermore, the common-prefix property and the chain-growth property formalize the presumption in [1] that a transaction is secured when a sufficient length of subsequent blocks is appended to the chain. In other words, when a block is a certain number of blocks deep from the end of the chain, or equivalently, the repeated block-proposal competition has passed sufficiently many rounds, the transaction data in that block is non-reversible/persistent and thus guaranteed to be double-spending proof. It is worth noting that the studies in [25], [85] provide a generalizable approach for evaluating the security and the efficiency of the PoX-based Nakamoto protocols in permissionless blockchains. Based on the quantitative analysis of the properties in Table II, the same framework of security evaluation has been adopted by the studies in consensus protocols using other types of puzzle design such as Proof of Stakes (PoS) [74], [86].

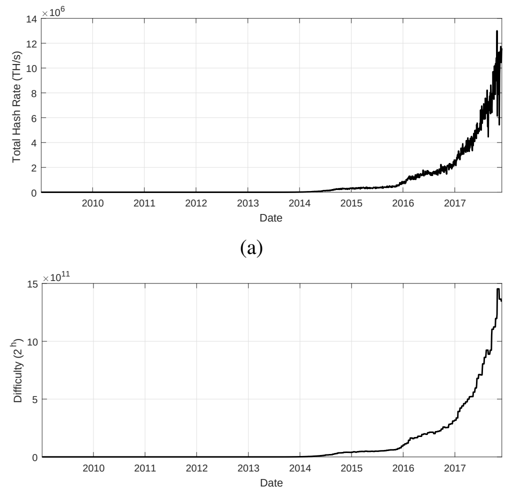
Due to the open access nature of permissionless blockchains, the hashrate presented in a practical blockchain network is generally unstable. As indicated by Figure 6, since the introduction of the Application Specific Integrated Circuit (ASIC) for hash acceleration in 2013, the practical PoW-based blockchain networks, e.g., Bitcoin, have experienced an explosive increase of the total hashrate with huge fluctuation [87]. Practically, blockchain networks adopt a heuristic, periodic difficulty-adjustment policy to maintain a roughly fixed time interval, i.e., λ^{-1} in (4), between two neighbor blocks. However, the expected value of λ^{-1} is usually chosen in an arbitrary manner and is frequently reduced in favor of a higher transaction throughput (see Litecoin [88] and Zcash [42] for example). Following the assumption of partial synchronization [25], the roughly fixed time interval indeed implies an upper bound for the information dissemination latency in the P2P network [89].

With such a consideration in mind, a theoretical study is provided in [90] between the upper bound of the information latency and the persistence of the block data in a node's local view of the blockchain. Consider a flat network of N nodes with a maximum block propagation delay of T. It is found in [90] that for a given fraction of adversary node ρ $(0 \leq \rho < 0.5)$, the block generation probability for each node should satisfy the following condition in order to ensure the property of data persistence (Theorem 1.1 in [90]):

$$\mathrm{Pr}_i^g \leq \frac{1}{T\rho\sum_{i=1}^{N} w_i}, \tag{5}$$

where Pr_i^g can be calculated based on (3) and a given hashrate.

Furthermore, a carefully chosen block interval is critical to achieving a proper trade-off between satisfying the Byzantine agreement properties and providing a decent transaction throughput. In [43], [89], examination on the block propagation delay T in (5) shows that a safe upper bound on T is jointly determined by the block size, the network scale, e.g., measured in hop counts, and the average round-trip time of the links. The empirical study in [43] reveals that for small-size blocks, e.g., less than 20kB for Bitcoin, the round-trip delay is the dominant factor of the block propagation delay. Otherwise, transaction validation time becomes the major factor of the block propagation delay, and the block propagation delay grows linearly with respect to the size of a block, e.g., 80ms/kB for Bitcoin. In [91], an implicit metric to capture the impact of network scale on the block propagation delay is adopted. Therein, the ratio between the block size and the propagation time required to reach a certain percentage of the nodes in the network is measured for the Bitcoin network. The experiments show that in the Bitcoin network with 55kb/s propagation rate for 90% of the nodes, the block interval should not be smaller than 12s, which leads to a peak transaction throughput of 26TX/s for 250Byte transactions.



(a)

(b)

Figure 6. Evolution of (a) the total hash rate and (b) the PoW puzzle difficulty in the Bitcoin network over time. Data source: https://www.blockchain.com.

Furthermore, the studies in [92], [93] also consider the impact of the propagation delay on the incidence of abandoning a proposed block with valid PoW solution. More specifically, finding a valid puzzle solution does not necessarily mean that the proposed block will be finally accepted by the network. Due to the propagation delay, a blockchain fork (see Figure 4) can only be adopted as the canonical blockchain state when it is first disseminated across the network. By considering both the round-trip delay and the block verification delay, the average block propagation delay across a P2P network is modeled as a function of the block size s in [93]:

$$T(s) = T_p(s) + T_v(s) = \frac{s}{aC} + bs, \tag{6}$$

where a is a network scale-related parameter, C is the average effective channel capacity of each link [94] and b is a coefficient determined by both the network scale and the average verification speed of each node (cf. [43]). Based on (6), the probability for the network to abandon/orphan a valid block proposal of size s due to the delay of block diffusion is modeled as follows [92], [93]:

$$\mathrm{Pr}^{\text{Orphan}}(s) = 1 - e^{-\lambda T(s)}, \tag{7}$$

where λ is the expected block arrival rate.

From a user's perspective, it is insufficient to know only the network-level probability of block orphaning due to the latency. Alternatively, it is of more interest to determine the safe time interval between locally observing on the chain a transaction and confirming it. With this in mind, the study in [90] considers a scenario where the adversary gets additional computation time by delaying the block propagation with a certain number of rounds Δ. Based on the analysis of the common-prefix property [25], A new metric, i.e., K-consistency is proposed in [90] to examine whether any two honest nodes are able to agree on the blockchain state that is at least K blocks deep from the end of the chain. Let α and β denote the probabilities that an honest node and the attackers can propose a valid block within a round, respectively. The analytical study in [90] (cf. Lemma 8 in [89]) shows that the required waiting time T is jointly determined by α, β, Δ and the parameter determining the searching space of the hash function, i.e., L in Definition 1. More specifically, as long as the following condition is satisfied with an arbitrary small constant $\delta > 0$ (see Theorem 1.2 in [90])

$$\alpha(1-(2\Delta+2)\alpha) \geq (1+\delta)\beta, \tag{8}$$

and $K > K_0(L) = c\log(L)$ for some constant c, the Nakamoto protocol satisfies the property of K-consistency (except with negligible probability in K). However, the closed-form threshold $K_0(L)$ for K-consistency is not provided in [90].

C. Proof of Concepts Attached to Useful Resources

Under the framework of Nakamoto protocol, a number of alternative PoX schemes have been proposed to replace the original PoW scheme in permissionless blockchain networks. Generally, these PoX schemes aim at two major designing goals, i.e., to incentivize useful resource provision, e.g., [38], [67], [80], [95], [96] and to improve the performance, e.g., in terms of security, fairness and eco-friendliness [79], [97], [98] of the blockchain networks. Starting from this subsection, we will focus on the principles of puzzle design discussed in Section III-A and provide a close examination on different PoX schemes in the literature.

With the purpose of useful resource provision, the idea of "Proof of Useful Resources" (PoUS) has been proposed to tackle the resource wasting problem of PoW. Instead of enforcing the consumption of computational cycles for merely hash queries, a number of studies are devoted to the design of puzzles that are attached to useful work. An early attempt, i.e., Primecoin [99], proposed to replace the PoW puzzle in (1) by the puzzle of searching three types of prime number chains, i.e., the Cunningham chain of the first/second kind or the bi-twin chain [100]. However, the verification stage of Primecoin puzzle is based on classical Fermat test of base two (pseudoprime) [99], hence violates the principle of soundness in non-interactive ZKP. Meanwhile, since the induced solution arrival does not follow the i.i.d. Bernoulli model in (3), the Primecoin puzzle does not simulate the random distribution for leader selection as required by (2).

In [101], a similar scheme, i.e., the proof of exercise is proposed to replace the preimage searching problem in PoW with the useful "exercise" of matrix product problems. The scheme uses a pool of task proposals to associate the puzzle solving processes with the tasks of non-authenticated clients. Each consensus node needs to bid for a specific task to determine its puzzle. For this reason, the puzzle solution-generating scheme behaves more like a Computation as a Service (CaaS) platform. Since the matrix problems in the task pool may present different complexity levels, the puzzle competition does not fully simulate on the network level the random distribution in (2). Also, the solution verification can only be done probabilistically due to the lack of $O(n)$ verification schemes. Therefore, the proposed scheme in [101] suffers from the same problems as in the Primecoin [99].

In [80], a new puzzle framework, i.e., useful Proof of Work (uPoW) is designed to replace the primitive PoW puzzle in (1) with a specific set of problems satisfying not only the properties of completeness, soundness and non-invertibility (hardness), but also the additional requirement of usefulness. Here, the usefulness is implied in the execution stage of the puzzle (cf. Table I). Formally, by assuming completeness and soundness, the properties of usefulness can be defined as follows (cf. Definition 1 in [80]):

Definition 2 (Usefulness)**.** *Suppose that a challenge c_x and an accompanying puzzle solution (proof) s are generated from an input string x. If there exists an algorithm Recon(c_x, s) such that for a target function $F(\cdot)$ its output satisfies Recon$(c_x, s) = F(x)$, the challenge is known to be useful for delegating the computation of $F(x)$.*

The study in [80] proposes to replace preimage searching in (1) with a family of one-way functions satisfying the property of fine-grained hardness [102] for uPoW puzzle design. A special case of uPoW puzzles based on the problem of k-Orthogonal Vectors (k-OV) is discussed. In brief, the solution to k-OV performs an exhaustive search over k sets of identical-dimension vectors and determines whether for each set there exists a vector such that these k vectors are k-orthogonal. In order to construct non-interactive proofs, uPoW in [80] employs the hash function $\mathcal{H}(\cdot)$ as a random oracle. Simply put, given the number of vectors in each set, non-interactive uPoW treats the elements of each vector as the random coefficients of polynomials with the identical order. uPoW initializes the first element of each vector, i.e., the lowest order coefficient with a publicly known input string x and then uses it as the input to $\mathcal{H}(\cdot)$ for generating the next-order coefficient. The output of $\mathcal{H}(x)$ will then be iteratively used as the input for generating the next-order coefficient. This can be considered as a typical example of applying the Fiat-Shamir heuristic [103] to construct non-interactive PoW out of interactive ZKP schemes. With such an approach, uPoW does not need to explicitly define the vector sets. It also guarantees that the solutions of k-OV found by each prover follow a Bernoulli distribution. Therefore, the uPoW scheme fits well in the existing Nakamoto protocols by simulating a provable random function. As stated in [80], besides k-OV, uPoW is compatible with computation delegation for other problems such as 3SUM [102], all-pairs shortest path [102], and any problem that reduces to them[7].

Schemes that are similar to uPoW can also be found in [96]. In [96], the problem of untrusted computational work assignment is addressed in a Trusted Execution Environment

[7]These problems should be worst-case hard for some time bound and can be represented by low-degree polynomials.

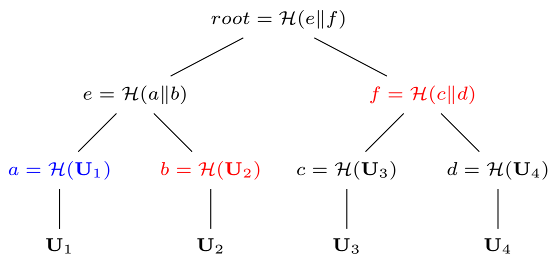


Figure 7. Illustration of Merkle proof: for segment $\mathbf{U}_1$, the Merkle proof is obtained by climbing up the tree until the root (as the nodes in red).

(TEE). The TEE can be constructed using Intel Software Guard Extensions (SGX), which is a set of new instructions available on certain Intel CPUs to protect user-level codes from attacks by hardware and other processes on the same host machine. In the permissionless network, the clients supply their workloads in the form of tasks that can be run in an SGX-protected enclave (i.e., protected address space). The study in [96] exploits the feature of the Intel attestation service [104] in the SGX-protected platform to verify and measure the software running in an enclave. With the designed puzzle, the work of each consensus node is metered on a per-instruction basis, and the SGX enclave randomly determines whether the work results in a valid block proof by treating each instruction as a Bernoulli trial. Based on the TEE, each executed useful-work instruction is analogous to one hash query in the primitive PoW, and the enclave module works as a trusted random oracle.

Apart from delegation of useful computation, PoX can also be designed to incentivize distributed storage provision. For example, Permacoin [105] proposes a scheme of Proof of Retrievability (PoR) in order to distributively store an extremely large size of data provided by an authoritative file dealer. The file dealer divides the data into a number of sequential segments and publishes the corresponding Merkle root using the segments as the leaves. A consensus node uses its public key and the hash function to select a random group of segment indices for local storage. For each locally stored segment, the node also stores the corresponding Merkle proof derived from querying the Merkle tree. The challenge-proof pair is generated based on a subset of the locally stored segments and the corresponding Merkle proof. To ensure the non-interactiveness and freshness of the puzzle (cf. interactive PoR in [106]), the node needs a publicly known, unpredictable and non-precomputable puzzle ID to seed the "scratch-off" process of segment selection. To help the readers understand the puzzle generation process, we present a simplified execution stage of PoR as follows (see also Figure 1 in [105]):

The execution stage of PoR: Suppose a node is given the key pair (sk, pk), the puzzle ID id_{puz}, the vector of locally stored segment indices $\mathbf{v}$, the required number of Merkle proofs k, the vectors of all the file segments $\mathbf{U}$ and the corresponding Merkle proof vector π. The random IDs of the local segments for challenge generation can be determined by:

$$\forall 1\leq j\leq k: r_j = \mathbf{v}\left(\mathcal{H}(id_{puz}\|pk\|j\|nonce) \mod |\mathbf{v}|\right), \quad (9)$$

where $nonce$ is a random value chosen by the node. For each segment $\mathbf{U}(\mathbf{v}(r_j))$ in the challenge, the proof is in the form of $(pk_i, nonce, \mathbf{U}(\mathbf{v}(r_j)), \pi(\mathbf{v}(r_j)))$.

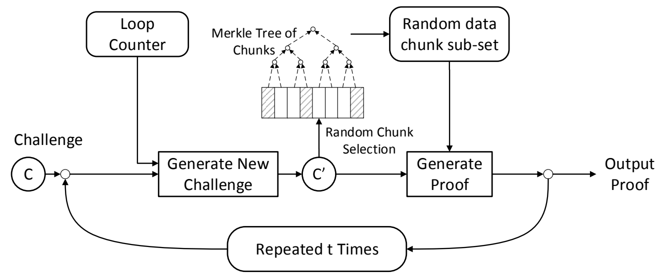


Figure 8. Illustration of the PoST scheme based on iterative PoR over time.

The execution stage of PoR in [105] is based on a fixed number of queries to the random oracle $\mathcal{H}$. As a result, although PoR satisfies the principle properties of non-interactive ZKP, it does not simulate the random leader election process. In this sense, the proposed PoR scheme may not be able to achieve the claimed goal of "repurposing PoW" in [105]. Instead, it is more similar to the existing systems such as Stoj [107], Sia [108] and TorCoin [95], where PoX is only used to audit the execution of the smart contracts or script-based transactions instead of facilitating the consensus mechanism.

Further improvement to PoR can be found in the proposals of KopperCoin [67] and Filecoin [38]. In [67], KopperCoin adopts the same framework of distributed storage for a single file as in Permacoin [105]. Compared with Permacoin, the main improvement of the puzzle design in KopperCoin is to simulate the random leader election process for block proposal. KopperCoin introduces a bitwise XOR-based distance metric between the index of a locally stored data segment and a random, publicly known challenge c. A node needs to provide the valid Merkle proof (PoR) of a segment, of which the index (denoted by j) should satisfy the following condition:

$$\mathcal{H}(x)\cdot 2^{|j\oplus c|} \leq D(h), \quad (10)$$

where the block payload x and the difficulty threshold $D(h)$ are defined in the same way as in Definition 1. Compared with (1), the solution searching for (10) is now performed within the range of the locally-stored segment indices. The more segments a node offers to store, the better chance the node has to find a solution to (10). Again, the generation of the public, unpredictable random challenge c can be derived based on hashing the header of the most recent block. This approach presents another example of applying the Fiat-Shamir transformation to realize non-interactiveness [103].

In the Filecoin network [38], the concept of "spacetime" is introduced to allow metering the data stored in the network with an expiry time. Filecoin aims to provide the functionality of recycling and re-allocating the storage on the provider (miner) side as well as easing the files retrieval process on the client side. Like in the proof-of-exercise scheme, Filecoin designs the market for storage and retrieval of multiple files based on smart contracts. A new puzzle, i.e., Proof of SpaceTime (PoST) [79], is adopted based on the intuition of generating a PoR sequence during a certain period to prove the holding time of useful storage. As illustrated by Figure 8, the major difference of PoST from PoR lies in the repeated execution phases for challenge updating without rerunning the initialization stage. Namely, a consensus node is required by the Filecoin network to submit PoR (e.g., in a similar way t Permacoin [105]) every time when the blockchain is extende by a certain number of blocks. Instead of simulating randoı leader election based on adjustable difficulty [67], the Filecoi network uses the following mechanism to determine whethe a node i is elected for block proposal:

$$\frac{1}{2^L}\mathcal{H}(t|\mathrm{rand}(t)) \leq \frac{w_i}{\sum_{j\in\mathcal{N}} w_j}, \qquad (11$$

where t is the index of consensus round (i.e., block index L is the output string length of the hash function (see (1) $\mathrm{rand}(\cdot)$ is an assumed random oracle, and w_i represents th storage power of node i (see also (2)). It is worth of noting tha the evaluation of w_i in (11) can only be done through PoST Thus, the Filecoin network admits a double-challenge scheme where the leader election is performed based on a second challenge, i.e., (11). The nodes with the better quality of PoST proofs (storage power) are more likely to win the second challenge. Under such a framework of double challenges, a similar approach of puzzle design can also be found in the proof of space-based cryptocurrency proposal known as SpaceMint [79], [97].

D. *Proof of Concepts for Performance Improvement*

Alternative PoX schemes have also been designed with the emphasis on improving the performance of PoW in the aspects such as security, fairness and sustainability. To alleviate the problem of computation power centralization due to the massive adoption of ASICs, memory-hard PoW, also known as the Proof of Memory (PoM), is adopted by Zcash [42] and Ethereum [37] networks. In the Zcash network, the Equihash scheme [77] is adopted based on the generalized birthday problem [109]. The study in [77] has pointed out that any identified NP-complete problem can be the natural candidate for the PoX puzzle due to their proved hardness, as long as the solution verification can be completed in polynomial time. However, a puzzle design only satisfying the hardness requirement may not be able to combat the botnet or ASIC-based manipulation of hashrate. Thus, a suitable PoX is expected to be optimization-free and parallelism-constraint.

An ideal approach of imposing parallelism constraint is to ensure that the PoW scheme is inherently sequential. However, an inherently sequential NP problem that is known to be verified in short time is yet to be found [77]. Therefore, the study in [77] adopts an alternative approach by imposing enormous memory bandwidth to the parallel solution of the puzzle. According to [109], the generalized k-dimensional birthday problem is to find k strings of n bits from k sets of strings, such that their XOR operation leads to zero. Equihash employs the hash function $\mathcal{H}(\cdot)$ to randomly generate the k strings using the block payload data x and a nonce (as in (9)), such that both the XOR-based birthday problem solution and a PoW preimage of a given difficulty are found. It is shown in [109] that the best solution algorithm to this problem presents $O(2^{n/k})$ complexity in both time and space and thus is memory-intensive. More importantly, for a k-dimensional problem, a discounting factor $1/q$ in memory usage leads to

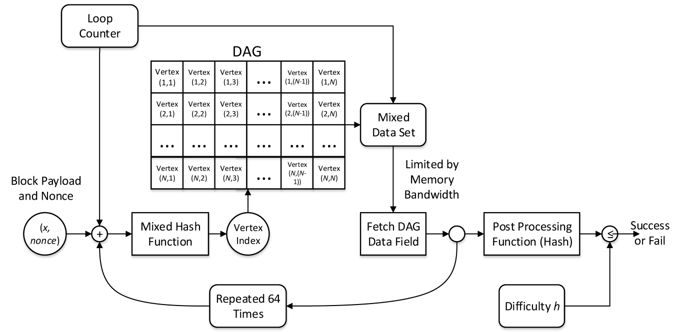


Figure 9. One query to the random oracle in Ethash for a given $nonce$ based on the iterative mixed hash opeartion for vertex searching.

$O(q^{k/2})$ times more queries to the hash function. Due to the physical memory bandwidth limit, the computation advantage of parallelization is limited. These properties guarantee the ASIC-resistance of Equihash.

With the same purpose of preventing the "super-linear" profit through hashrate accumulation, Ethreum currently adopts a different puzzle design known as Ethash for ASIC resistance [110]. Ethash requires the consensus nodes to search for the PoW puzzle solution based on a big pseudorandom dataset, which increases linearly over time. The dataset is organized as the adjacency matrix of a DAG, where each vertex represents a randomly generated data field of 128 bits. In the execution stage of Ethash, the node starts a one-time search of the solution with a hash query, and uses the concatenation of the block payload and a nonce to seed the hash function for locating a random vertex in the DAG. Then, the search is completed in a fixed-iteration loop of queries to the hash function, for which the output of the last iteration, i.e., the data field of the last vertex in the path is used as the input to determine the position of the next vertex in the DAG. The final output of the loop is used to check against the preimage condition as in (1). As illustrated in Figure 9, the designed puzzle of Ethash makes the searching algorithm inherently sequential. With Ethash, the rate of data field fetching from the DAG is limited by the memory bandwidth. Then, paralleling the hash queries with ASICs cannot lead to much performance improvement in a single search of the puzzle solution.

Ethash [110] only makes the puzzle solution partially sequential within a single attempt of preimage search. Therefore, Ethash still faces the problem of PoW outsourcing since a consensus node can divide the puzzle solution search into multiple sub-problems and outsource them to different "mining workers" (i.e., puzzle solvers). Such a problem is also known as the formation of mining coalition (pool) [59] and may result in a serious problem of consensus manipulation by a handful of full nodes [4]. In [78], a nonoutsourceable "scratch-off puzzle" is proposed to disincentivize the tendency of mining task outsourcing. Intuitively, a puzzle is nonoutsourceable such that if a node effectively outsources its puzzle-solving work to some mining machines, these miners can steal the block proposal reward of that node without producing any evidence to implicate themselves. The study in [78] employs Merkle proofs for puzzle design, which can be considered as a generalization of the PoR [105]. In [78], a Merkle tree is created based on a number of random strings. To generate a

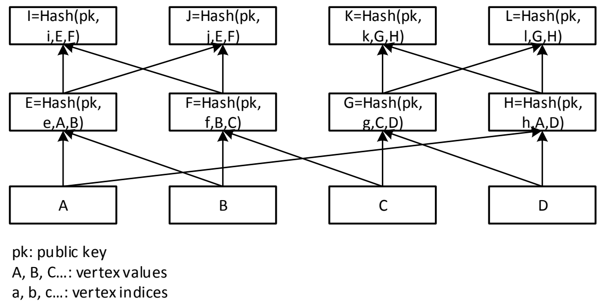


Figure 10. An example of DAG formation based on the hash of the parent vertices: for miner i adopting a public key pk_i, the value v_j of the j's vertex in its DAG with m parent vertices $\{p1, \ldots, pm\}$ is obtained as $v_j = H(pk_i, j, v_{p1}, \ldots, v_{pm})$.

fresh puzzle, a node queries the hash function for the first time with a random nonce and the constructed Merkle root. The output of this query is used to select a random subset of distinct leaves on the Merkle tree. Then, the concatenation of the Merkle proofs for each leaf in subset and the same nonce is used as the input to the second query of the hash function. The output is used to compare with the preimage condition as given in (1). If a solution (nonce) is found, the payload of the proposed block is used as the input of the third query to the hash function, and the output is used to select another subset of random leaves on the Merkle tree. The corresponding Merkle proofs are treated as the "signature" of the payload of the proposed block. With such puzzle design, mining workers only need to know a sufficiently large fraction of the Merkle tree leaves to "steal" the reward by replacing the Merkle proof-based signature with their own proofs.

It is worth noting that the nonoutsourceable puzzle in [78] is generated in a way of making the preimage search for (1) independent of the payload of the proposed block, i.e., using the randomly generated Merkle tree. Then, a mining worker is able to replace the original payload including the public keys from the outsourcer by its own payload without being detected. A similar proposal of nonoutsourceable puzzle can be found in [111], where a nonoutsourceable puzzle is designed based on two-fold puzzle. Namely, an inner puzzle is solved as a typical PoW puzzle, whose solution is used as the input of an additional PoW puzzle known as the outer puzzle. To prevent outsourcing the work load, a mining worker's signature is required for the inner puzzle solution to be used by the outer puzzle. However, it is pointed out in [111] that such design can only be considered heuristic and is not guaranteed to have the formal properties of weak outsourceability [78].

Apart from the manipulation-resistant puzzles, other puzzles are proposed in [97], [98] with the emphasis on eco-friendliness. Therein, the major goal is to reduce/remove the repeated hash queries to curb energy consumption due to hash queries. In [97], the SpaceMint network is proposed based on Proof of SPace (PoSP) [112]. Similar to PoR [105], PoSP requires the consensus nodes to provide non-interactive proofs of storage dedication during puzzle solution searching. The major difference from PoR lies in that PoSP does not need the prover to store useful data (from the verifiers), and the proof is based on a large volume of randomly data stored on the provers' hard drive. As in Ethash [110], the committed space is also organized as a DAG, where the value of each vertex is determined based on the hash of its parent vertices (see Figure 10). A consensus node is required to use the hash of an earlier block as the seed to sample a random set of vertex values. The set of the vertex values forms the challenge of the node's local PoSP puzzle. If the node is able to provide the Merkle proofs for all the vertices in the challenge set, namely, the sibling vertices that lie on the path between each challenge vertex and the end vertex in the DAG with no outgoing edge, the proposed block is considered a valid block candidate. SpaceMint also proposes to measure the quality of a set of Merkle proofs based on the hash value of the concatenated vertex in a Merkle tree. Then, the blockchain network is able to select the block with the best quality of proof from the candidate blocks when a fork occurs.

The study in [98] proposes to introduce a human-in-the-loop puzzle, i.e., the Proof of Human-work (PoH) into the Nakamoto protocol. The designing goal of PoH is to guarantee the properties of eco-friendliness, usefulness and centralization-resistance at the same time. It is proposed in [98] that PoH should be able to provide non-interactive, computer-generated puzzles which are moderately hard for a human but hard for a computer to solve, even for the computer that generates the puzzles. PoH is inspired by the widely-adopted systems of Completely Automated Public Turing-Test to tell Computers and Humans Apart (CAPTCHA) [113]. Traditional CAPTCHA systems usually take human-efficient input (e.g., images) with a known solution and generates the puzzle based on distortion to the solution. For PoH, a universal sampler [114] is assumed to be available to generate a random CAPTCHA instance for the consensus node such that the puzzle-generating machine is not able to directly obtain the puzzle solution. Then, the node (i.e., miner) needs human work to obtain the corresponding solution of the CAPTCHA puzzle. A two-challenge puzzle design is adopted and the solution of the CAPTCHA puzzle is used as the input of a small PoW puzzle as defined in (1). A complete PoH solution includes a CAPTCHA solution and a nonce such that they together satisfy the preimage condition in (1). PoH implicitly assumes that some Artificial Intelligence (AI) problems (e.g., recognition of distorted audios or images) are human-efficient but difficult for machines. Then, by selecting a proper underlying CAPTCHA scheme, it is possible to extend the PoH with a variety of meaningful human activities ranging from that educational purposes to a number of socially beneficial programs [114].

For a progressive summary, we summarize in Table III the major properties of the PoX schemes that have been discussed in Section III.

IV. Incentive and Node Strategies in the Framework of Nakamoto Consensus Protocols

In this section, we review the studies on the incentive compatibility of the Nakamoto consensus protocols. By adopting the basic assumption on rationality of the consensus nodes (i.e., block miners), we provide a comprehensive survey on the node strategies in the consensus process for block mining. It is worth noting that most of the analysis in the literature

Table III
COMPARISON OF DIFFERENT POX SCHEMES FOR PERMISSIONLESS BLOCKCHAINS

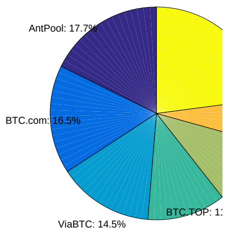
Puzzle Name	Origin of Hardness (One-way Function)	Designing Goal	Implementation Description	ZKP Properties	Simulation of Random Function	Features of Puzzle Design	Network Realization
Primitive proof of work [25], [82]	Partial preimage search via exhaustive queries to the random oracle	Sybil-proof	Repeated queries to cryptographic hash function	Yes	Yes	Single challenge	Bitcoin [1], Litecoin [88]
Proof of exercise [101]	Matrix product	Computation delegation	Probabilistic verification	N/A	No	Single challenge	N/A
Useful proof of work [80]	K-orthogonal vector, 3SUM, all-pairs shortest path, etc.	Computation delegation	Non-interactiveness via Fiat-Shamir transformation	Yes	Yes	Single challenge with sequential hash queries	N/A
Resource-efficient mining [96]	N/A	Computation delegation	Guaranteed by TEE	Yes	Yes	Trusted random oracle implemented by dedicated hardware	N/A
Proof of retrievability [106]	Merkle proofs of file fragments in the Merkle tree	Distributed storage	Non-interactiveness via Fiat-Shamir transformation and random Merkle proofs	Yes	Conditional	Two-stage challenge	Permacoin [105], KopperCoin [67]
Proof of space -time [38]	The repeated proof of retrievability over time	Decentralized storage market	Repeated PoR	Yes	Conditional	Two-stage challenge and repeated PoR over time	Filecoin [38]
Equihash [77]	The generalized birthday problem	ASIC resistance	Time-space complexity tradeoff in proof generation [77]	Yes	Yes	Memory-hard	Zcash [42]
Ethash [110]	Random path searching a random DAG	ASIC resistance	Repeated queries to cryptographic hash function	Yes	Yes	Sequential, memory-hard puzzle	Ethereum [37]
Nnonoutsourceable scratch-off puzzle [78]	Generalization of proof of retrievability	Centralization resistance	Random Merkle proof	Yes	Yes	Two-stage challenge	N/A
Proof of space [112]	Merkle proofs of a vertex subset in a random DAG	Energy efficiency	Random Merkle proof	Yes	Yes	Two-stage challenge and measurement of proof quality	SpaceMint [112]
Proof of human work [98]	Radom CAPTCHA puzzle requiring human effort	Useful work and energy efficiency	CAPTCHA and PoW	Yes	Yes	Human in the loop	N/A

about the consensus nodes' mining strategies are presented in the context of the PoW-based Bitcoin network. Nevertheless, they can be readily extended to other PoX schemes under the framework of Nakamoto protocols. In particular, we focus on the game theoretic formulation of resource allocation during the mining process, and then explore how miners can exploit the vulnerability of the incentive mechanism of the Nakamoto protocols in permisionless blockchain networks.

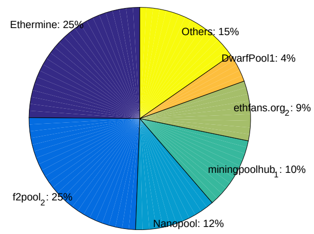
A. Incentive Compatibility of Nakamoto Protocols

For Nakamoto protocols, monetary incentive plays the key role to ensure that most of the consensus nodes/miners follow the rules of blockchain state transition during the puzzle solution competition. In permissionless blockchain networks, the incentive mechanism is built upon the embedded digital token issuing and transferring schemes. In a typical PoW-based blockchain network, the leader/winner in the block proposal competition not only collects transaction fees from the approved transactions in the new block, but also gets token issuing reward, e.g., the "coinbase reward" in Bitcoin, for expanding the blockchain with the new block. For this reason, the puzzle competition process is compared to the process of "gold mining", since by casting resources into the competition, the nodes expect to receive monetary rewards carried by the tokens. As a result, the consensus participant nodes are better known as block "miners" to the public.

In [62] the consensus in blockchain networks is divided into three folds, namely, the consensus about the rules, e.g., about transaction dissemination and validation, the universality of the blockchain state and financial value that the digital token carries. Then, the studies on the Nakamoto protocol's incentive compatibility can also be categorized according to these three aspects. Since the introduction of ASIC devices and pool mining for PoW-based blockchain networks, concerns have been raised about the nodes' incentive to fully abide by the protocol [58], [59], [62], [115]. Due to the explosion of network-level hashrates (see Figure 6(a)), most of the practical blockchain networks, i.e., cryptocurrency networks, are nowadays dominated by the proxies of mining pools [63] (see Figure 11). An individual node in a mining pool is known as a mining worker, since it no longer performs the tasks of transaction validation or propagation and does not even keep any blockchain data. On the contrary, only the proxy of the pool, i.e., the pool server/task operator maintains the replica of the blockchain. The pool server divides the exhaustive preimage search for PoW solution into a number of sub-tasks



(a)



(b)

Figure 11. Hash rate distribution among mining pools in (a) Bitcoin (data source: https://blockchain.info) and (b) Ethereum (data source: https://etherscan.io) networks.

and outsource them to the mining workers[8]. In this sense, only the pool server can be considered as a node in the blockchain network. Studies have shown that joining a mining pool has become the more plausible strategy than working as an individual consensus node, since such a strategy reduces the income variance and secures stable profits [4], [59]. However, this leads to the formation of mining-pool Cartel [59] and is against the design goal of Nakamoto consensus in [1], that "the network is robust in its unstructured simplicity".

A further study in [56] reveals that under the current framework of Nakamoto protocols, no incentive is provided for nodes to propagate the transactions that they are aware of. The study considers the situation when transaction fees dominate the block rewards [117], and the number of nodes competing for block proposal is small. The analysis in [56] models the paths of transaction dissemination as a forest of d-ary directed trees, where each transaction issuer considers its peer nodes as the tree roots and the nodes on the far end of the network as the leafs. It is shown that a consensus node tends to not broadcast the transaction to the others in order to reduce the competition and increase its expected reward. An improved protocol is introduced in [56] by introducing a broadcasting incentive mechanism. The analysis of the new protocol is based on the formulation of an extensive-form game [118], and thus the equilibrium strategy of each node can be obtained through iterative removal of dominated strategies. The designed incentive mechanism is shown to guarantee that only the non-Sybil and information propagating strategies survive in the iterated removal of weakly dominated strategies, as long as the miners are connected to sufficient many peers.

Similar studies on incentive mechanism design to enforce block and transaction propagation can be found in [60], [119]. The study in [60] casts the problem of incentivizing block propagation into the framework of routing in P2P networks. A protocol of transaction fee-sharing is designed therein to guarantee that the rational strategy of honest nodes in the network is to propagate the received transactions. However, it is worth noting that instead of using the Nakamoto protocol, a hybrid consensus protocol is adopted in [60] such that the leader must be validated before it proposes the block (cf. hybrid protocols in Section V such as [120]). In [119], the incentive mechanism is designed based on a similar hybrid protocol. In the proposed protocol, a chain of signatures is kept by the leader, and the leader determines the amount of processing fee passed to the next node (i.e., receiver).

When block creation reward dominates the mining reward, incentive incompatibility may appear in different forms. Intuitively, it is plausible for a rational miner to pack up a proper number of transactions with decent fees in the new block for profit maximization. However, empty blocks with only coinbase transaction or blocks with a tiny number of transactions can be frequently observed in the practical blockchain networks[9]. An informal game theoretic analysis in [121] indicates that the consensus nodes tend to ignore the received blocks of large size in a flat network and relay the smaller competing blocks instead. The reason is that large blocks result in longer delay due to transaction validation, hence increasing the probability of orphaning any blocks that are mined based on them. Although mining empty block does not violate the current Nakamoto protocol, it results in the same situation as a Distributed Denial of Service (DDoS) attack [122] by blocking the confirmation of normal transactions.

Furthermore, the statistical studies in [123], [124] have shown that the consensus nodes display rational behaviors and are prone to prioritize the transactions with higher transaction fees during block packing. However, when the coinbase reward dominates the block mining reward, the miners are yet not incentivized to enforce strictly positive fees [124]. At the performance cost, extra delays for the small-value transactions are identified ranging from 20 minutes [124] to as long as 30 days [123] in the case study of Bitcoin network. Also, it is observed in [124] that most of the lightweight nodes still set an arbitrary transaction fee in the real-world scenarios. It is unclear whether the miners or the transaction issuers adopt best-response strategies systematically. The study in [125] simplifies the consensus process as a supply game subject to the trade of a specific type of physical goods. In the considered scenario, the miners essentially become the Stackelberg followers [118] in a hierarchical game. Then, they are expected to have an incentive for including all transactions if there exists no block-size limit. On the other hand, it is pointed out in [94] that, since the block orphaning probability exponentially grows with the block size, a healthy transaction fee market does not exist for unlimited block size due to the physical constraint of link capacity in the network.

Finally, it is worth noting that most of the existing studies are based on the presumption that the tokens carried by a blockchain have monetary value and their exchange rate volatility is small. An optimistic prediction is provided in [57] based on an assumption excluding any state variables on the user sider except the belief in "proper functioning of a cryptocurrency". In the absence of investors and when the blockchain is used only for the purpose of remittance, it is shown in [57] that the tokens of a blockchain network admits a unique equilibrium exchange rate in each period of the belief evolution. Conditioned on the survival of a cryptocurrency, the equilibrium state depends on the excess in users' valuation of the blockchain over the other payment options as well as

[8]According to the Stratum mining protocol [116], the pool server only needs to send a miner the Merkle root of the transactions in the block (see Figure 2) and a difficulty level to complete the puzzle solving sub-task.

[9]See Block #492972 in Bitcoin and Block #3908809 in Ethereum for examples.

the supply of the tokens in the market. Together with the Stackelberg game-based interpretation in [125], it is reasonable to consider that the equilibrium price of a blockchain token is determined by the demand-supply relation in the market. It is worth noting that the data security is only guaranteed by sufficient PoW computation power in the blockchain network. Currently, except a few studies such as [126], it is generally unclear how the impact of security issues is reflected in the users' valuation of the blockchain. As a result, whether the security requirement of the Nakamoto protocol is compatible with the market clearing price remains an open question.

B. Resource Investment and Transaction Selection for Mining under Nakamoto Protocols

According to (2), an honest consensus node has to invest in the mining resources, e.g., hashrates, disk space, etc, to win the puzzle solution competition under Nakamoto consensus protocols. Intuitively, the more resources a miner casts into the network, the higher chance the miner has to win the puzzle competition and obtain the mining reward. However, the success is not guaranteed because this also depends on the mining resources of other miners. Since mining resources are usually expensive, how to properly invest in the mining resources to maximize the profit is a big concern of the miners.

The study in [127] abstracts the mining investment in the Bitcoin network as the energy consumption cost. It is assumed that N active miners in the network are competing in the "all-pay contest" for block-mining rewards. The cost of presenting a unit mining resource by each miner may be different, e.g., with different electricity prices in different areas. The miners determine how much to invest in mining resources (hashrates) such that the expected profit is maximized. This forms a non-cooperative game among the miners. Analysis of the game's unique Nash equilibrium in [127] shows that the decision of a miner to participate in the mining process or not solely depends on its individual mining cost, as long as the block reward is positive. Meanwhile, the structure of the formulated mining game prevents the emergence of a monopolistic mining activity. Namely, it is guaranteed that at least two miners will remain active in the game with positive expected profits.

By (6) and (7), even if a miner succeeds in the puzzle solution competition, it is still possible for the proposed block to get orphaned due to the propagation delay. For ease of exposition, we can assume that all transactions in a block set the same amount of transactions fee F. Let R denote the fixed reward for block generation and m denote the number of transactions in the block. Then, the revenue to mine this block is $R + mF$. Apparently, a rational miner expects to include as many as possible transactions in a block to maximize the received reward. However, due to the risk of block orphaning, a miner also has to carefully balance the tradeoff between the mining reward and the risk of block orphaning. In [94], the author proposes a mining profit model by assuming the propagation delay of a block to follow a Poisson distribution. Thus, the orphaning probability can be approximated by (7). Let η denote the monetary cost per hash query and ψ denote the probability for the miner being the leader (see also (3)). Then, for an average block arrival duration T and block propagation time τ, a miner's profit can be modeled as follows:

$$U = (R + F)\psi e^{-\frac{\tau}{T}} - \eta h T. \tag{12}$$

The profit model in (12) is capable of reflecting the impact of miners' strategies in both resource investment and transaction selection. Therefore, this model is especially appropriate for game-theoretic formulation of mining resource management problems. Recently, (12) and its variation have been adopted to construct the payoff function of miners by a series of studies, which propose to use different game-based models, e.g., evolutionary game [93], hierarchical game [128] and auctions [129], to capture the rational behaviors of individual miners in different network setups.

In [130], an alternative model of winning probability is proposed to explicitly capture the influence of the adversary miners' strategy of block-size selection. We denote s_i as block size of miner i in a blockchain network and w_i as its computational power. Then, the block winning probability of miner i can be expressed by [130]:

$$\Pr_i^{\text{win}} = \frac{w_i}{T}\left[\prod_{j\neq i}\left(e^{-\frac{w_j(t+\tau(s_i)-\tau(s_j))}{T}}\right)\right], \tag{13}$$

where t is the time when all miners start mining a new block and $\tau(s_i)$ is the time needed for a block with size s_i to reach consensus. In (13), the first and second terms represent the probability for miner i to first solve the puzzle based on its block, for this block to be the first one reaching the consensus across the network, respectively. (13) implies that the strategy of mining a large block may have positive externalities to other miners in the network. By analyzing the Nash equilibrium of the non-cooperative mining game with two miners, the author of [130] shows an interesting result, namely, the miner with higher computational power will prefer blocks of larger sizes. Meanwhile, the author also discusses the scenarios in which the Nash equilibrium is a breaking point, i.e., miners adopt the strategy of including no transaction in their proposed blocks.

The studies in [94] and [130] essentially assume that the mining process is synchronized and all miners honestly follow the rules of block/transaction propagation in Nakamoto protocols. However, such assumptions may not be met in practical scenarios. Thus, related strategies may not be the miners' best response and further investigation is needed on this topic.

C. Rational Mining and Exploitation of Nakamoto Protocols

1) Selfish Mining Strategy: The study in [59] shows that selfish miners may get higher payoffs by violating the information propagation protocols and postponing their mined blocks. Specifically, a selfish miner may hold its newly discovered block and continue mining on this block secretly. Thereby, the selfish miner exploits the inherent block forking phenomenon of Nakamoto protocols. In this case, honest miners in the network continue their mining based on the publicly known view of the blockchain, while the selfish miners mine on their private branches. If a selfish miner discovers more blocks in the same time interval, it will develop a private longer branch of the blockchain. When the length of the public

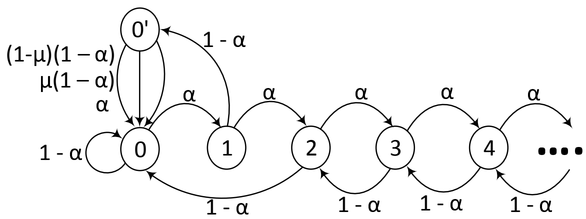


Figure 12. Blockchain state transition in the presence of a selfish pool (adapted from [59]).

chain known by honest miners approaches that of the selfish miner's private chain, the selfish miner will reveal its private chain to the network. According to the longest-chain rule, the honest nodes will discard the public chain immediately when they learn the longer view of the chain from the selfish miner. Such a strategy of intentionally forking result in the situation of wasted computation by the honest miners, while the revenue of the selfish miner can be significantly higher than strictly following the block revealing protocol. More seriously, if selfish miners collude and form a selfish mining pool with sufficiently large amount of computational power, other rational miners will be forced to join the selfish mining pool, which can devastate the blockchain network [59].

In [59], the authors introduce an approach based on the Markov chain model to analyze the behavior as well as performance of a selfish mining pool. Figure 12 illustrates the progress of the blockchain as a state machine. The states of the system, i.e., the numbers in the circles represent the lead of the selfish pool in terms of difference in block number between the private branch and the public branch. In Figure 12, state 0 is the original state when the selfish pool has the same view as the public chain. State $0'$ indicates that two branches of the same length are published in the network by the selfish pool and the honest miners, respectively. The transitions in Figure 12 correspond to the mining event, i.e., a new block is mined either by the selfish pool or the honest miners. α in Figure 12 represents the computational power of the selfish mining pool. Note that the transition from state 0 to state $0'$ depends on not only the computational power of the selfish pool, but also the fraction, i.e., μ of honest miners that mine on the selfish pool's branch. In [59], the analysis on the steady state probability of the Markov chain leads to the following two important observations:

- For a given μ, a selfish pool of size α obtains a revenue larger than its relative size in the range of $\frac{1-\mu}{3-2\mu} < \alpha < \frac{1}{2}$.
- A threshold on the selfish-pool size exists such that each pool member's revenue increases with the pool size.

Extended from [59], the study in [131] introduces a new mining strategy known as the stubborn mining strategy, which is supposed to outperform the typical selfish mining strategy. The key idea behind the stubborn mining strategy is that the selfish miner is stubborn and may only publish part of the private blocks even when it loses the lead to the honest nodes. As shown in Figure 13, the major difference between the two selfish strategies lies in how the selfish miner publishes the private blocks. For example, at state 2, the typical selfish miner will immediately publish all the private blocks once the lead to the honest miners decreases by one block (see Figure 12).

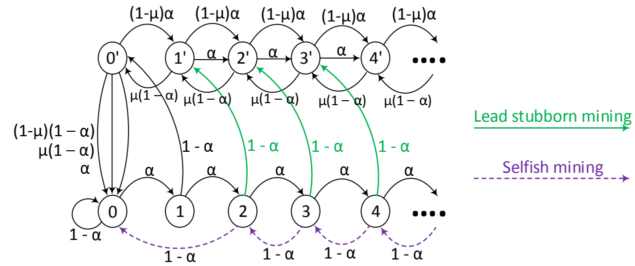


Figure 13. Lead-stubborn mining. The black and purple transitions together define the selfish mining state machine. The black and green transitions define the stage machine of lead-stubborn mining (adapted from [131]).

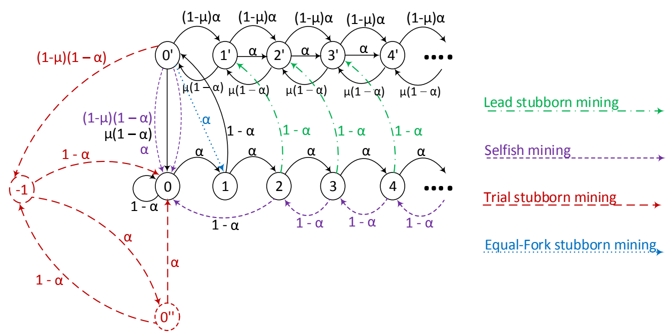


Figure 14. Lead, Equal-Fork, and Trail Stubborn mining. Black and purple transitions denote selfish mining. Black and green transitions denote lead-stubborn mining. Black and blue transitions denote Equal-Fork stubborn mining. Black and brown transitions denote Trail-stubborn mining (adapted from [131]).

Then, the system transits to state 0. In contrast, every time when the honest miners mine a new block, the stubborn miner will stubbornly reveal one block of the private chain, even by doing so it will lose the lead. Simulations in [59] show that stubborn mining achieves up to 13.94% higher gains than selfish mining strategy.

Furthermore, the study in [131] also introduces another two extensions of the stubborn mining strategy, namely, the Equal-Fork Stubborn (EFS) and the Trail Stubborn (TS) mining strategies (see Figure 14). In Figure 14, state -1 indicates that the public chain is one block longer than the private chain. As indicated by the transitions from other states to state -1, the TS miner is more stubborn and keeps mining on the secret branch even when it is one block behind the public chain. From state -1, when the TS miner finds one new block ahead of the honest miners, the system will transit to state $0''$. Namely, the private chain catches up with the public chain and the block numbers on both chains are equal. In contrast, if the honest miners find a new block ahead of the ST miner, the system transits to state 0. Namely, the ST miner starts to mine new blocks based on the public chain. Here, the difference between state $0''$ and state $0'$ lies in that only the ST miner knows the existence of the private chain in state $0''$, while in state $0'$ the honest miners can freely choose to mine on one of the two chains. The comparisons between the three stubborn mining strategies are given in Figure 14. Simulations in [131] show that stubborn mining strategies can improve the profit by up to 25% than the original selfish mining strategy proposed in [59].

The author in [132] studies the impact of transaction fees on selfish mining strategies in the Bitcoin network. Due to the

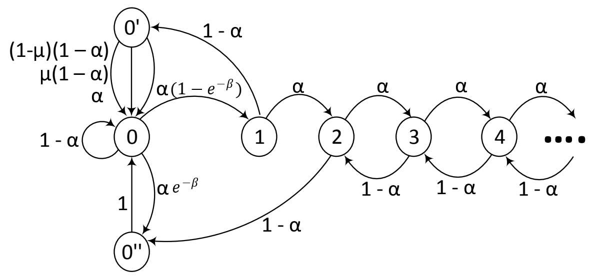


Figure 15. Improved Markov model for selfish mining with transaction fees (adpated from [132]).

fact that the fixed block mining reward reduces over time due to the inherent design of the token-issuing protocol for Bitcoin, it is natural to increase the transaction fee to compensate for the mining cost of the consensus nodes. The arbitrary levels of transaction fees lead to a situation where some hidden blocks may have very high values. As a result, selfish miners want to publish it immediately due to the risk of orphaning. Hence, in the revised Markov chain model for selfish mining in Figure 15, the author introduces a new state $0^{''}$. State $0^{''}$ is almost identical to state 0, except that, if the selfish miner mines on the next block in state $0^{''}$, it will immediately publish that block instead of holding it. Compared with the original selfish mining model in Figure 12, state 0 transits to state 1 with probability $\alpha(1-e^{-\beta})$ and to state $0^{''}$ with probability $\alpha e^{-\beta}$, where β is the size of the mining block. The new factor β is introduced to model the impact of transaction fees on the miner's decisions. With the revised transition probability, if the selfish miner finds a block of high value in state 0, it may publish the block (i.e., transiting to state $0''$) instead of holding it (i.e., transiting to state 1). The analysis in [132] shows that this improved selfish mining strategy leads to positive profit for all miners regardless of their hashrates.

From the aforementioned Markov models, we note that the selfish miner may adopt various policies by choosing to release an arbitrary number of block in each state. In [133]–[135], a Markov Decision Process (MDP) model is proposed to generalize such a process of policy derivation. As an example, the study in [133] considers the honest miners as non-adaptive players following the Nakamoto protocol. Then, the problem of searching optimal selfish-mining strategy can be modeled as a single-player MDP. Four actions are considered to control the state transitions in the MDP:

- *Adopt*: the selfish miner accepts the honest network's chain and all private blocks are discarded.
- *Override*: when taking the lead, the selfish miner publishes its private blocks such that the honest network discards its current view.
- *Match*: the selfish miner publishes a conflicting branch of the same height. A fraction of the honest network will fork on this branch.
- *Wait*: the selfish miner does not publish new blocks and keeps working on its private branch.

The state the MDP is defined by the difference in block lengths between the selfish miner and the honest network as well as the situation of computation forking among the honest miners. By controlling the maximum difference in block lengths, it is possible to obtain a finite-state MDP. Using standard MDP solution techniques, an ϵ-optimal policy for selfish mining can be obtained based on such a truncated-state MDP.

In [136], the authors consider a similar mining competition between a selfish mining pool and the honest nodes. The study in [136] extends the model of selfish mining by considering the propagation delay between the selfish mining pool and the honest community. The delay is assumed to be exponentially distributed with rate μ. The block-mining Markov model in [136] adopts a 2-dimensional state of (k,l), which denotes the length of blocks built by the pool and the community upon the common prefix blocks, respectively. Let λ_1 and λ_2 denote the block-arrival rate for the pool and the community. The authors then derive the following transition rates of the block mining system:

$$
\begin{aligned}
q\big((k,l),(k+1,l)\big) =& \lambda_1, \quad k\geq 0, l\geq 0,\\
q\big((k,l),(k,l+1)\big) =& \lambda_2, \quad k\geq 0, l\geq 0,\\
q\big((k,l),(0,0)\big) =& \mu, \qquad\qquad k<l,\\
q\big((k,k-1),(0,0)\big) =& \mu, \qquad\qquad k\geq 2,\\
q\big((k,l),(k',l')\big) =& 0, \qquad\quad \text{otherwise}.
\end{aligned}
\tag{14}
$$

Based on this transition map, the authors in [136] propose to detect selfish mining behaviors by monitoring the proportion of orphaned blocks. Specifically, if there is a significant increase in the fraction of orphaned blocks, it is highly possible that selfish mining exists in the network.

In [137], the authors adopt a more general assumption of multiple selfish miners in a Bayesian game-based formulation. In the considered game, miners decide on whether to report a new block (R), i.e., to mine honestly, or not (NR), i.e., to mine selfishly. when a miner makes a decision, its does not know whether it is the real leader of the mining competition, or whether some other miners have secretly started mining on their private blocks. To ease the analysis of this mining game with incomplete information, the authors assume that a miner always reports when it finds two successive blocks. With this extra assumption, a decision tree can be constructed (see Figure 16), and the backward induction approach is adopted to find the miners' equilibrium strategies. Figure 16 presents the decision tree in a case of three miners. In the presented subgame, miner 1 believes that it is the real leader of the mining competition. Here, let h_i denote the normalized computational power of miner i, and $\mu_i(h_i)$ denote miner i's belief of being the leader of the puzzle solution competition. From the decision tree and following the Bayesian rule, we can obtain the information about the states, transition probabilities, and expected payoffs after miner 1 takes the action of NR. The authors provide the condition on the fraction of computational power for action NR to become the optimal mining strategy.

2) Block Withholding in Pool-Based Mining: Block withholding (BWH) is a mining strategy used by selfish miners to increase their revenues through diminishing the winning probability of honest miners in mining pools [138], [139]. In [139], the authors study the impact of BWH on the Bitcoin network. It is assumed that a selfish miner is able to split the computational power into different mining pools. It may spend most of its computational power to honestly mine on one pool,

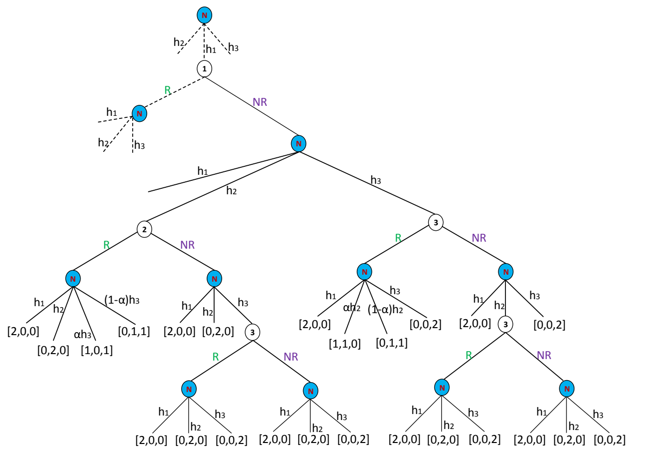


Figure 16. An illustration of the Bayesian mining game (adapted from [137]). Miner 1 believes that its is the real leader of the puzzle solving competition and decides to take action NR. Here α is the probability for miners to mine on the first block when they receive two blocks in a short time.

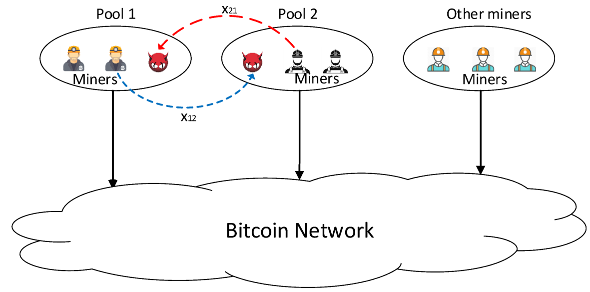


Figure 17. BWH attacks between two miners.

and use the rest computational power to perform BWH on the other pools. The mining pools are supposed to adopt the pay-per-share protocol [63]. In the victim mining pools, the selfish miner submits all shares[10] to the pool operators except the valid puzzle solutions. Although this mining strategy reduces the attacker's revenue in the attacked pools, it will increase the attacker's revenue in the pool that it chooses to mine honestly. A computational power splitting game with multiple players is formulated in [139]. In the game, one selfish miner adopts BWH and all the other miners mine honestly. The selfish miner chooses which pools to attack and how much computational power to allocate in the targeted pools. It is shown that the attacker always gains positive reward by mining dishonestly regardless of its mining power. This findings implies a risk for big mining pools to dominate the network through BWH attacks on smaller mining pools.

The study in [140] considers a more complicated case where mining pools attack each other with BWH. The author of [140] considers a scenario of two mining pools which attempt to send their miners to each other to diminish their opponents. As illustrated in Figure 17, pool P_1 uses x_{12} out of the m_1 computational power to attack pool P_2. Meanwhile, pool P_2 uses x_{21} out of the m_2 computational power to attack pool P_1. Then, the revenue of each pool can be derived as follows:

$$
\begin{aligned}
R_1 &= \frac{m_1 - x_{12}}{m - x_{12} - x_{21}},\\
R_2 &= \frac{m_2 - x_{21}}{m - x_{12} - x_{21}},
\end{aligned}
\tag{15}
$$

where m is the total mining power in the blockchain network. By [140], the revenues of the pools can be expressed as the functions of x_{12} and x_{21}:

$$
\begin{aligned}
r_1(x_{12}, x_{21}) &= \frac{m_2 R_1 + x_{12}(R_1 + R_2)}{m_1 m_2 + m_1 x_{12} + m_2 x_{21}},\\
r_2(x_{21}, x_{12}) &= \frac{m_1 R_2 + x_{21}(R_1 + R_2)}{m_1 m_2 + m_1 x_{12} + m_2 x_{21}}.
\end{aligned}
\tag{16}
$$

Thus, by observing the attack rate of its opponent, a mining pool can adjust its attack rate in the next round to maximize its long-term revenue through repeated plays. The analysis of this repeated game reveals that the game admits a unique equilibrium, and the pool size will be the main factor that determines the attacking rates of each pool. A similar conclusion about the impact of the pool size on BWH attacks between two pools can also be found in [117].

Extended from the studies in [139], [140], it is found out in [141] that when a mining pool performs a BWH attack to a victim mining pool, the other mining pools will benefit from this attack even if they do not adopt BWH. Thus, the other pools are interested in sponsoring the attacker to launch the BWH attack to the victim pool. Consequently, the expected gain of the attacker will be greater than the case in [139]. This implies that miners have more incentives to perform BWH attacks with the Nakamoto consensus protocols.

To alleviate the impact of BWH attacks, modifications to the Nakamoto protocol and the pool-mining protocols are suggested in the literature. The author in [63] proposes that the pool operator should insert mining tasks for which the solutions are known in advance, and tag the miners that do not submit the results. Since it is difficult to find puzzles with expected solutions, the author suggests that new data fields should be added to the block data structure. These fields enable the pool operator to allocate mining tasks to its miners, but the miners are unable to know the exact puzzle solutions. Alternatively, in [142], the authors propose to give an extra reward to the miners that find the valid blocks, hence reducing the revenue of selfish miners and discouraging BWH attacks.

3) Lie-in-Wait Mining in Pools: Lie-in-wait (LIW) is a strategic attack where a selfish miner postpones submitting the block that it finds to a mining pool, and uses all of its computational power resources to mine on that pool [63]. In this case, an attacker is assumed to first split its computational power to mine in different pools. Then, if it finds a block in a pool, instead of submitting the block to get the reward from the pool, the attacker holds the block, and concentrates all of its computational power in other pools to mine on the pool where it finds the block. However, the attacker may take a risk by not releasing the block immediately and concentrating all the computational resources on the target pool. The reason is that if one of other pools finds a new block before this block is published, the selfish miner will lose its reward as well as suffer from the cost of mining in the target pool. It is shown in [63] that the success of attacks follows an exponential distribution, and the maximum expected gain of the LIW attacker is solely determined by the pool numbers

[10] A share is a preimage solution for a block that meets the relaxed (approximated) difficulty requirement set by the pool.

Table IV
SUMMARY OF SELFISH MINING STRATEGIES AND THEIR INCURRED RISKS IN BLOCKCHAIN NETWORKS

Attacks	Selfish mining	Block withholding	Lie-in-wait	Pool hopping
References	[59], [131]–[133], [136], [137]	[117], [139]–[142]	[63]	[63]
Concept	After finding a new block, the attacker hides the block and continues mining on the mined block secretly.	After finding a new block in the victim pool, the attacker discards that block and continues mining on its block in another pool.	After finding a new block in a mining pool, the attacker holds the block and uses all the computational power to mine on that pool.	The attacker moves to another pool or start mining by himself when the mining time at its current pool reaches a threshold.
Risks of attackers	A new attacker's found block can be discarded if one of other miners finds a new block before it finds a next new block.	The attacker loses its reward at the victim pool if it finds a new block in this pool.	The attacker can lose its reward for its mined block and all computational power at the pool it found the block.	There is no risk and loss for the attacker if its mining pools use pay-per-share protocol.
Risks of honest miners	Lose their rewards for their mined blocks.	Lose their rewards for blocks found by attackers.	Can lose their rewards if the block found by the attacker in their mining pool is discarded from the network.	Their profits will be reduced if they are in mining pools using pay-per-share protocol.
Suggested solutions	Modification to the mining protocol, e.g., blockchain propagation method and blockchain update rule.	Modification to the task assignment protocol in pools such that miners does not know real results of their mining tasks.	Modification to the task assignment protocol in pools such that miners does not know real results of their mining tasks.	Change the payment method for mining pools.

and block interval in the network.

4) Pool Hopping Strategy: With the strategy of pool hopping, the miners exploit the vulnerability of the payment mechanism of mining pools to increase their own profits. With the pay-per-share protocol, the number of submitted shares in one block competition round follows a geometric distribution with the success parameter δ and mean D [63]. For I shares submitted to a pool, the pool still needs D more shares on average to mine the block. When ignoring the transaction fees, the more shares submitted to a pool in a round, the less each share is worth. Since a miner immediately receives the payment for the submitted share, this implies that a share submitted early may have a higher reward. Therefore, a selfish miner can benefit by mining only at the early stage of a round, and then hop to other pools to increase his revenue. The study in [63] shows that there exists a critical point measured in the number of submitted shares. The best strategy of a selfish miner is to mine on a pool until this point is reached, then hop to another pool or mine by himself.

One straightforward way to address the block hopping problem in pay-per-share mining pools is to increase the value of shares at the end of each round. The pool operator may score the shares according to the elapsed time since the beginning of each round. A share can be scored by an exponential score function $s(t) = e^{t/\delta}$, where t is the time stamp of the submitted share and δ is a parameter controlling the scoring rate of shares. With the help of share scoring, we can handle pool hopping attacks in mining pools by decreasing the score of shares at the beginning and increasing the score of shares later. Such score-based method is also known as Slush's method and has been implemented in the mining pools such as Slushpool [143]. In [63], other incentive mechanisms such as pay-per-last-N-shares and payment-contract-based methods are also sketched. However, Analytical studies on these mechanisms are missing and their effectiveness in preventing pool hopping attacks still remains an open issue.

V. VIRTUAL BLOCK MINING AND HYBRID CONSENSUS MECHANISMS BEYOND PROOF OF CONCEPTS

A. *Proof of Stake and Virtual Mining*

The concept of PoS was first proposed by Peercoin [73] as a modified PoW scheme to reduce the energy depletion due to exhaustive hash queries. Peercoin proposes a metric of "coin age" to measure the miner's stake as the product between the held tokens and the holding time for them. Miner i solves a PoW puzzle as in (1) with an individual difficulty $D(h_i)$. The Peercoin kernel protocol allows a miner to consume its "coin ages" to reduce the difficulty for puzzle solution. The public verification of the "coin ages" is done through empirically estimating the holding time of the miner's Unspent Transaction Output (UTXO) based on the latest block on the public chain.

By completely removing the structure of PoW-based leader election, the protocols of pure PoS are proposed in [35], [74], [75], [144]. To simulate a verifiable random function following the stake distribution (see also (2)), an algorithm, follow-the-coin (a.k.a., follow-the-satoshi), has been proposed by [75] and widely adopted by these works[11]. Here, the terms "coin" or "satoshi" are used to indicate the minimum unit of the digital tokens carried by the blockchain. Briefly, all the tokens in circulation are indexed, for example, between 0 and the total number of available coins in the blockchain network. A simplified PoS protocol can use the header of block $t-1$ to seed the follow-the-coin algorithm and determine the random mining leader for block t. Specifically, the hash function $\mathcal{H}(\cdot)$ is queried with the header of block $t-1$, and the output is used as the random token index to initialize the searching algorithm. The algorithm traces back to the minting block (i.e., the first coinbase transaction [35]) for that token or the UTXO account that currently stores it [75]. Then, the creator or the holder of the token is designated as the leader for generating block t. To enable public verification of the block, the valid leader is required to insert in the new block its signature, which replaces the data field "nonce" for PoW-based blockchains.

It is worth emphasizing that the pure PoS protocols do not rely on a Poisson process-based puzzle solution competition to

[11] A reference implementation in Python (see also [75]) can be found at http://www.cs.technion.ac.il/~idddo/test-fts.py.

simulate the random generator of the block leader. Therefore, the ZKP process can be simply replaced by asymmetric key-based signature verification, and the proof of resource is no longer needed. For this reason, PoS is also known as a process of "virtual mining" [4] since the block miners do not consume any resources. In the literature, a number of protocol proposals are claimed to be able to (partially) achieve the same purpose. However, these protocols either need special hardware support, e.g., Intel SGX-enabled TEEs for proof of luck/elapsed-time/ownership [76], [145], or are still under the framework of PoW, e.g., Proof of Burn (PoB) [146], Proof of Stake-Velocity (PoSV) [147] and "PoS" using coin base [73]. As a result, they cannot be considered as the real "virtual mining" schemes in permissionless blockchain networks.

Compared with the PoX-based protocols, PoS keeps the longest-chain rule but adopts an alternative approach for simulating the verifiable random function of block-leader generation. For this reason, the same framework for analyzing the properties of Byzantine agreements in PoW-based blockchain networks [25] can be readily used for the quantitative analysis of PoS protocols. For example, the investigations in [74], [148] mathematically evaluate the properties of common prefix, chain quality and chain growth based on the same definition in Table II. The authors propose in [74] the "Ouroboros" protocol, and consider that the stakes are distributed at the genesis block by an ideal distribution functionality. By assuming an uncorrupted ideal sampling functionality, "Ouroboros" guarantees that a unique leader is elected in each block generation round following the stake distribution among the stakeholders (see also (2)). With "Ouroboros", forking no longer occurs when all the nodes are honest. However, when adversary exists, forking may be caused by the adversarial leader through broadcasting multiple blocks in a single round. The study in [74] shows that the probability for honest nodes to fork the blockchain with a divergence of k blocks in m rounds is no more than $\exp(-\Omega(k)+\ln(m))$ under the condition of honest majority. It is further shown that the properties of chain growth and chain quality are also guaranteed with negligible probability of being violated.

The studies in [75], [148] introduce the mechanism of epoch-based committee selection, which dynamically selects a committee of consensus nodes for block generation/validation during an epoch (i.e., a number of rounds). Compared with the single-leader PoS protocol, i.e., "Ouroboros" [74] and its asynchronous variation [149], the committee-based PoS gears the protocol design toward the leader-verifier framework of traditional BFT protocols (see also Figure 5). In [75], the scheme of Proof of Activity (PoA) is proposed with the emphasis that only the active stake-holding nodes get rewarded. The PoA is featured by the design that the leader is still elected through a standard PoW-based puzzle competition, and is only responsible for publishing an empty block. Using the header of this block to seed the follow-the-coin algorithm, a committee of N ordered stakeholders is elected and guaranteed to be publicly verifiable. The first $N-1$ stakeholders work as the endorsers of the new empty block by signing it with their private keys. The N-th stakeholder is responsible for including the transactions into that block. The transaction fees are shared among the committee members and the block miner. In this sense, PoA can be categorized as a hybrid protocol that integrates both PoW and PoS schemes.

In [148], the authors propose a protocol called "Snow White", which uses a similar scheme to select a committee of nodes as in [75]. However, only the selected committee members are eligible for running for the election of the block generation leader. Under the "Snow White" protocol, the leader of an epoch is elected through a competition based on repeated preimage search with the hash function. At this stage, the difference of "Snow White" from the standard PoW puzzle in (1) is that the hash function is seeded with the time stamp instead of an arbitrary nonce. Like PoA, "Snow White" also pertains the characteristics of a hybrid protocol. The analysis in [148] shows that the proposed protocol supports frequent committee reconfigurations and is able to tolerate nodes that are corrupted or offline in the committee.

The recent proposal by Ethereum, Casper [150] provides an alternative design of PoS that is more similar to traditional BFT protocols. The current proposal of Casper does not aim to be an independent blockchain consensus protocol, since it provides no approach of leader election for block proposal. Instead, the stakeholders join the set of validators and work as the peer nodes in a BFT protocol. The validators can broadcast a vote message specifying which block in the blockchain is to be finalized. The validator's vote is not associated with its identity, but with the stake that it holds. According to [150], Casper provides plausible liveness (instead of probabilistic liveness with PoW) and accountable safety, which tolerate up to $1/3$ of the overall voting power (weighted by stake) that is controlled by the Byzantine nodes.

Regarding the incentive compatibility of PoS, an informal analysis in [74] shows that being honest is a δ-Nash equilibrium[12] strategy when the stakes of the malicious nodes are less than a certain threshold and the endorsers are insensitive to transaction validation cost. However, a number of vulnerabilities are also identified in PoS. In [151], the nothing-at-stake attack is considered. In order to maximize the profits, a block leader could generate conflicting blocks on all possible forks with "nothing at stake", since generating a PoS block consumes no more resource than generating a signature. A dedicated digital signature scheme is proposed to enable any node to reveal the identity of the block leader if conflicting blocks at the same height are found. Alternatively, a rule of "three strikes" is proposed in [35] to blacklist the stakeholder who is eligible for block creation but fails to properly do so for three consecutive times. In addition, an elected mining leader is also required to sign an auxiliary output to prove that it provides some extra amount tokens as the "deposit". In case that this node is malicious and broadcasts more than one block, any miner among the consecutive block creation leaders can include this output as an evidence in their block to confiscate the attacker's deposit. Such a scheme is specifically designed to disincentivize block forking by the round leader.

Grinding attack is another type of attacks targeting PoS [74].

[12] At a δ-NE, the payoff of each player is within a distance of $\delta > 0$ from the equilibrium payoff.

With PoS, the committee or the leader is usually determined before a round of mining starts. Then, the attacker has incentive to influence the leader/committee election process in an epoch to improve its chances of being selected in the future. When the verifiable random generator takes as input the header of the most recent block for leader/committee election, the attacker may test several possible block headers with different content to improve the chance of being selected in the future (e.g., [74], [75]). It is expected to use an unbiased, unpredictable random generator to neutralize such a risk [74]. In practice, the protocol usually selects an existing block that is a certain blocks deep to seed the random function instead of using the current one [75], [148].

With all the aforementioned studies, a significant limit of the existing analyses about PoS-based protocols lies in the simplified assumption that ignores the stake trade outside the blockchain network (e.g., at an exchange market) [152]. A study in [153] provides an counterexample for the persistence of PoS in such a situation. The study in [153] assumes no liquidity constraint in a blockchain network, where nodes own the same stake at the beginning stage. The author of [153] considers a situation where a determined, powerful attacker attempts to destroy the value of the blockchain by repeatedly buying the stake from each of the other nodes at a fixed price. After taking into account the belief of the nodes that the attacker will buy more tokens, the interaction between the attackers and the stakeholders is modeled as a Bayesian repeated game. The study concludes that the success of the attack depends on two factors, namely, the attacker's valuation of the event "destroying the blockchain" and the profit (e.g., monetary interest) that the nodes can obtain from holding the stake. When the first factor is large and the latter is small, the nodes in the network will end up in a competition to sell their stake to the attackers. As a result, the blockchain can be destroyed at no cost.

B. Hybrid Consensus Protocols

Despite the unique characteristics of permissionless consensus protocols, public blockchain networks are known to be limited in performance (e.g., transaction throughput) due to the scalability-performance tradeoff [19]. To boost permissionless consensus without undermining the inherent features such as scalability, a plausible approach is to combine a permissionless consensus mechanism (e.g., Nakamoto protocol) with a fast permissioned consensus protocol (e.g., BFT). Following our previous discussion (cf. PoA [75] and Casper [150]), we study in this subsection how a standard permissionless consensus protocol can be improved by incorporating (part of) another consensus protocol in the blockchain networks.

In [154], Bitcoin-NG is proposed to extend the PoW-based Nakamoto protocol. The prominent feature of Bitcoin-NG is to decouple the consensus process in a blockchain network (e.g., Bitcoin network) into two planes: leader election and transaction serialization. To bootstrap the transaction throughput, the protocol introduces two types of blocks, namely, the key blocks that require a PoW puzzle solution for leader election and the microblocks that require no puzzle solution and are used for transaction serialization. The time interval between two key blocks is known as an epoch. In an epoch, the same leader is allowed to publish microblocks with the limited rate and block size. Although operation decoupling in Bitcoin-NG does not ensure strong consistency, it paves the way for incorporating additional mechanisms on the basis of standard Nakamoto protocols.

Following the methodology of [154], hybrid consensus mechanisms atop Nakamoto protocols are proposed in [120], [155] with the goal of providing strong consistency and immediate finality. In [120], the "PeerCensus" protocol is proposed by decoupling block creation and transaction committing/confirmation. "PeerCensus" consists of two core components, namely, a PoW scheme named as BlockChain (BC) and a BFT-based scheme named as Chain Agreement (CA). With the proposed BC protocol, nodes acquire the voting right of the CA protocol when they propose new blocks through PoW and are approved by the committee of CA. The CA protocol is adapted from BFT protocols such as PBFT [18] and the Secure Group Membership Protocol (SGMP) [156]. Through the four stages of propose, pre-prepare, prepare, and commit of BFT protocols (cf. Figure 5), CA designates the miner of the newest block in the chain as the leader for the next block proposal. The leader proposes one from the multiple candidate blocks obtained in BC. The peer nodes in the committee extend the pre-prepare stage with an operation of block validation. The design of "PeerCensus" ensures that committing transactions (i.e., CA) is independent of block generation (i.e., BC). Therefore, no forking occurs in the condition of honest majority and strong consistency is guaranteed.

In [155], a hybrid consensus protocol is proposed by combining the data framework of two-type blocks in Bitcoin-NG and the hybird PoW-BFT design in "PeerCensus". As in "PeerCensus", the Nakamoto protocol is used to construct a "snailchain", which is allowed to commit transactions from a specific mempool of outstanding transactions known as the "snailpool". Following the quantitative analysis of the common prefix blocks in a chain in [25], only a fixed number of miners whose recently minted blocks are a certain blocks deep in the chain can be used to form the committee for the BFT protocol. In contrast to "PeerCensus", the BFT committee of miners in the proposed protocol has no influence on how the next block on the snailchain is determined. Instead, it is responsible for committing transactions from an independent mempool known as the "txpool". For this reason, the transactions approved by the BFT protocol are committed off the snailchain without relying on any mining mechanism. In this sense, these transactions can be considered similar to those in the microblocks of Bitcoin-NG. The hybrid consensus protocol in [155] explicitly addresses the problem of BFT-committee scalability in "PeerCensus" and provides a secured (with theoretical proof) consensus property of immediate finality. Namely, the transaction confirmation time from the txpool only depends on the network's actual propagation delay. The method of using Nakamoto protocols to select nodes into a BFT committee is also known as the proof of membership mechanism [157]. A sliding-window mechanism is proposed in [157] to generalize the mechanisms of dynamic BFT-committee selection in [120],

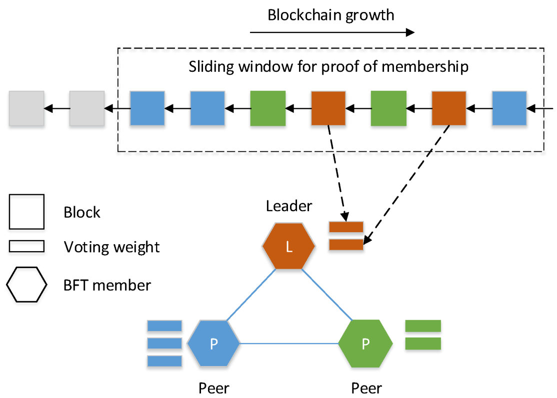


Figure 18. Illurstration of BFT-committee formation with weighted voting power. Valid weights are only credited to the miners of the blocks in the sliding window (adapted from [157]).

[155]. As illustrated in Figure 18, the BFT committee is maintained by a fixed-size sliding window over the PoW-based blockchain. The sliding window moves forward along the blockchain as new blocks are appended/confirmed. Consensus nodes minting multiple blocks in the window are allowed to create the same number of pseudo-identities in the BFT consensus process to gain the proportional voting power.

For hybrid consensus using BFT protocols to guarantee strong consistency, a natural thinking is to replace the Nakamoto protocols with virtual mining (e.g., PoS) for selecting the leader or committee in BFT-consensus processes. A typical example for such an approach can be found in the "Tendermint" protocol [158], where a node joins the BFT committee of block validators by posting a bond-deposit transaction. The validator no longer needs to prove its membership by competing for the PoW-puzzle solution. Alternatively, its voting power is equal to the amount of stake measured in bonded tokens. Meanwhile, instead of randomly electing the leader of block proposal in the committee (cf. [154]), "Tendermint" adopts a round-robin scheme to designate the leader in the committee. The similar design can be found in a number of recent proposals such as Proof of Authority (PoAu) [159] and Delegated Proof of Stake (DPoS) [160]. To generalize the mechanisms of BFT-committee selection based on virtual mining, the authors in [161] further propose a consensus protocol called "Algorand". Like the other hybrid protocols, "Algorand" relies on BFT algorithms for committing transactions. It assumes a verifiable random function to generate a publicly verifiable BFT-committee of random nodes, just as in [75]. The probability for a node to be selected in the committee is in proportion to the ratio between its own stake and the overall tokens in the network. For leader election, "Algorand" allows multiple nodes to propose new blocks. Subsequently, an order of the block proposals is obtained through hashing the random function output with the nodes' identities specified by their stake. Only the proposal with the highest priority will be propagated across the network.

In Table V, we provide a summary of the virtual-mining mechanisms and the hybrid consensus protocols discussed in this section.

VI. Outlook of Blockchain-Based Applications in Wireless Networks

A. Emerging Applications of Blockchain in Wireless Networks

Generally, the studies on the emerging, blockchain-related applications can be divided into two main categories: the provision of network consensus under the existing blockchain protocols and the provision of services on top of the blockchain consensus layer. The former studies usually focus on the P2P structure of the networks, where individual nodes are incentivized to devote their controlled resources in the PoX process for the monetary reward. The latter studies usually exploit the special characteristics of blockchain networks, namely, decentralization, data security/privacy and trustless framework of market organization, and employ blockchain networks to guarantee specific features in their respective services. In this section, we provide an extensive review on how the blockchain protocols and the numerous applications exert mutual influence on each other.

1) Computation Provision and Offloading under Nakamoto Protocols: Recently, there is a series of work exploiting the potential of consensus provision for PoW-based blockchain applications in mobile environments [128], [129], [162]. Recall that Nakamoto consensus protocols require competition in PoW puzzle solution. The resource intensity prevent the complete migration of blockchain networks to the mobile environment [163]. With mobile edge computing, the local edge servers are deployed at the "edge" of mobile networks. Through wireless access, the mobile devices can offload the computing tasks of PoW puzzle solution to the local edge computing services, which allows for the operation of mobile blockchains [164]. In this sense, the edge computing is considered as the network enabler for mobile blockchain.

In [128], the authors study optimal pricing-based edge computing resource management to support mobile blockchain applications, where the search for PoW solution is offloaded to an edge computing service povider. Therein, a two-stage Stackelberg game is formulated to capture the interactions between the edge provider and mobile miners. Through backward induction, the computation offloading market is proved to admit a unique Stackelberg equilibrium. Based on the similar formulation of computation offloading, the authors in [129], [162] investigate the auction mechanisms for edge computing resource management. In particular, the proposed scheme in [129] maximizes the social welfare while guaranteeing incentive compatibility, i.e., trustfulness of the mobile miners. To further improve the problem of sub-optimal revenue for the edge providers in [129], the authors in [162] further explore the potential of applying deep learning techniques to achieve optimal auction for revenue maximization.

2) Content Distribution and Delivery: In [165], the authors propose a decentralized mechanism that aims for securing content distribution in information centric networks [166]. The proposed mechanism utilizes a public key encryption scheme, i.e., Hierarchical Identity Based Encryption (HIBE) [167] to provide secure content distribution. The system uses content name as HIBE public keys and employ the Bitcoin-

Table V
SUMMARY OF VIRTUAL MINING AND HYBRID CONSENSUS PROTOCOLS FOR PERMISSIONLESS BLOCKCHAINS

Protocol Name	Virtual Mining	Hybrid Consensus	Simulating Leader Election with	Rule of Longest Chain	Decoupling Block Proposal from Transaction Commitment	Featured Consensus Properties
Proof of stake [35], [74], [144]	Yes	No	Verifiable random function, e.g., follow-the-coin	Yes	N/A	No resource consumption
Proof of luck, elapsed-time and ownership [76], [145]	Yes	No	Verifiable random function implemented by Intel-SGX-protected enclave	Yes	N/A	No resource consumption. Special hardware support is needed
Proof of burn [146]	Partially	No	PoW puzzle competition	Yes	N/A	Reduced resource consumption
Proof of stake-velocity [147]	Partially	No	PoW puzzle competition	Yes	N/A	Reduced resource consumption
Snow White [148]	Partially	PoS-PoW	Modified preimage search with the hash function	Yes	N/A	Robust consensus through reconfigurable PoS committee
Proof of activity [75]	Partially	PoW-PoS	PoW puzzle competition for empty block proposal	Yes	Transactions are committed by a random group of stakeholders	Higher cost for attackers to compromise the network consensus than both PoW and PoS
Casper [150]	No	PoW-PoS	PoW puzzle competition	Yes	N/A	Validators use BFT protocols to anchor checkpoint blocks in the block tree
Bitcoin-NG [154]	No	Partially	PoW puzzle competition	Yes	Proposals of microblocks do not need PoW solutions	Leader election is only performed at key blocks
PeerCensus [120]	No	PoW-BFT	PoW puzzle competition	N/A	Yes, Blocks are committed by BFT committees	Strong consistency without blockchain forking
Hybrid consensus protocol [155]	No	PoW-BFT	PoW-puzzle competition in the snailchain	Yes	Partially, only the transactions in txpools are committed following BFT protocols	Immediate finality
Tendermint [158], Proof of authority [159] and delegated proof of stake [160]	Yes	PoS-BFT	Verifiable random function or deterministic mechanism	N/A	Yes, following typical BFT protocols	Deterministic consensus properties
Algorand [161]	Yes	PoS-BFT	Verifiable random function	N/A	Yes, following typical BFT protocols	Safety and liveness are guaranteed under strong synchrony

like blockchain network, Namecoin[13] [168] for information registration and transfer. A further study in [169] directly maps the account addresses of blockchain networks to the addresses in the Named Data Networking (NDN). The studies in [165], [169] show that blockchain system over NDN circumvents the problems in IP network such as eavesdropping and traffic analysis. Here, blockchain networks serve as the distributed auditor of content usage and provenance. Other blockchain applications with the same goal can be found in cloud storage [170] and general-purpose data usage auditing infrastructures [171].

In [172], the authors study how distributed consensus-powered smart contracts can be employed to support video and content delivery networks. The authors propose a content distribution model with a scalable blockchain-based brokering mechanism, which consists of a series of smart contracts for several content providers to collaboratively provide the requested service. The proposed model is built upon three blockchains, namely, the content brokering blockchain, the delivery monitoring blockchain and provisioning blockchain. The content brokering blockchain deals with the content delivery sessions and helps create contracts. The delivery monitoring blockchain processes the delivery contract, and the delivery provisioning blockchain handles the diffusion of content on the storage devices. The proposed content delivery framework adopts the blockchain networks as the backbone for transaction execution and auditing among decentralized network entities. As pointed out in a similar work [173], such design offloads the transaction execution and auditing functionalities from the centralized accounting server to the distributed network entities and ensures that the market is self-organized.

3) Cognitive Wireless Networks: The cloud-centric cognitive wireless networks utilizes the dynamic spectrum access and opportunistic access to prevent spectrum crunch [174]. However, offering the carrier with personally identifiable information to users poses the risk of privacy leakage in cognitive cellular networks. To address this issue, the authors in [175] propose user identity management system using blockchain to enhance user security in the cellular domain. The pseudonymous identities in blockchain network makes it possible for separating the identity provider from the network operator, with cryptographic contracts for managing data access. The proposed blockchain based system is shown to possess better network access provision against the existing network access protocols. Nevertheless, the setup in [175] did not provide provisions to address user handoff for ensuring service continuity, which is also a major concern in cognitive cellular networks. To address this inadequacy, the authors

[13]https://namecoin.org.

in [176] propose elastic handoff as a composite framework of conventional cellular and voluntary spectrum handoffs. Therein, the Network Access Exchange (NAE) approach is used for a Cognitive Cellular User (CCU) to make a rational decision on carrier selection [175]. The NAE is considered as a consortium blockchain, where CCUs and NAE have varying privileges, e.g., read and write. The interaction between CCUs and NAE in blockchain is governed by smart contracts. The consensus mechanism in blockchain is adopted to achieve the actual spectrum utilization level in a given location. It is shown that the proposed approach is able to reduce spectrum sensing cost and minimize access signaling traffic, since the consensus method hosts a spectrum options exchange.

Similarly, in cognitive radio networks, the spectrum sharing has the potential to relieve recent increasing spectrum bandwidth needed, where the underlying security concerns are present. In [177], the authors propose a blockchain verification protocol method allowing for secure spectrum sharing in cognitive radio networks. This sharing mechanism is considered as a medium access protocol for accessing bandwidth resource among competitive cognitive radios. By utilizing blockchain, the cognitive radios lease and access available wireless channels using this secure distributed protocol. Each spectrum leasing transaction is verified, cleared, and stored in block. Therein, a virtual currency named Specoins is proposed as the payment to access spectrum as well as the reward for mining to update the blockchain. The blockchain-based protocol manages the transactions between users and is adopted for validating each user's virtual wallet. The transactions are generated during accessing or leasing available spectrum with currency exchange. It is demonstrated in [177] that the proposed protocol outperforms the current conventional system under both moderate and severe fading conditions to share available unexploited spectrum. However, in addition to the fading conditions, the impact of other parameters in wireless channel on the performance needs to be investigated in future works.

4) Crowdsourcing: Crowdsourcing systems utilize an undetermined set of mobile devices to incrementally solve or complete complex tasks, e.g., sensing. Traditionally, the requester posts the task to the crowdsourcing system, and the system broadcasts the problem and finds the workers to solve. Then, the worker receives the task and works for it. However, as the crowdsourcing relies on central servers, the high service fees from the platform impedes the development of crowdsourcing. In response to this challenge, the authors in [178] propose a blockchain-based decentralized crowdsourcing system, in which a task can be solved by a set of workers without depending on a trusted entity. Smart contract is adopted to depict complex logics and perform the whole crowdsourcing process, e.g., task posting, task receiving and reward assignment. Miners are introduced into the system, which record past transactions in the blockchain and validate a new block by the consensus protocol. Taking advantage of blockchain, the users can finish a crowdsourcing process and store the encrypted solutions in the data plane, which runs locally on users' local personal computer without relying on central entities. As such, the requesters and workers reach an agreement on underlying blockchain which is used to achieve consensus on each task state. In addition, the posting tasks and solutions are signed by digital signature, which guarantees the integrity of transactions and avoid the repudiation between the requesters and workers. Further, the proposed framework is evaluated on Ethereum by implementing a software prototype with real dataset. Nevertheless, the authors in [178] only implement the simple and basic process of crowdsourcing, which is not appropriate for current real complex systems.

5) Smart Home: Home appliances that upload a large quantity of home and personal data to centralized databases controlled by smart device manufacturers or retailers may be exposed to serious privacy problems [179]. Therefore, the blockchain technology can provide a transparent and secured alternative framework for private data management in smart home. Also, all the historical traceable data records are also available in blockchain for potential smart home applications. However, applying blockchain to IoT has one key challenge that needs to be addressed, i.e., high computing resource and processing delay requirements due to the use of PoW. To solve this problem, the authors in [180] investigate an optimized blockchain that eliminates the overhead associated with the classic blockchain while maintaining the security and privacy benefits. In particular, the proposed blockchain requires no mining and thus saves computing resources and incurs no additional delays in processing generated transactions. In the framework, there are mainly three layers, namely, smart home, overlay network and cloud storage. Each home has a local private immutable ledger that is similar to the blockchain but managed centrally. High resource devices jointly create a distributed overlay to implement the blockchain that guarantees the end-to-end security and privacy. Communications among different entities in different layers are regarded as transactions that are bundled into blocks. The verified blocks are added into the blockchain without incurring PoW, which significantly decreases the processing overhead. It is demonstrated that the proposed method in [180] has lower packet and processing overhead, and the security design is validated against several cyber-attacks. Further, the proposed architecture is application-agnostic for diverse IoT-based cases in addition to smart home case. However, a comprehensive analysis on consensus algorithm in the proposed framework is needed for extensive evaluation.

6) Vehicular Communication: Vehicular Communication Systems (VCSs) support message exchange among vehicles and that between vehicles and infrastructures, where the security is a main concern. Usually, this can be addressed through secure group communication, therefore, secure key management scheme is considered as an important technique for network security [181]. In [182], the authors present a framework for providing secure key management in VCSs, where the security managers are considered as blockchain miners which capture the vehicle departure information, encapsulate block to transport keys and execute rekeying to vehicles. In particular, the proposed framework mainly consists of two layers. The first layer is a novel network topology based on decentralized blockchain structure. The blockchain is used to simplify the distributed key transfer procedure in

VCSs for better efficiency. In this layer, the third-party central managers are removed, and the key transfer process is verified and authenticated by the security managers. These processes considered as transactions are recorded in blocks and shared within the same security domain for security managers to create public ledgers. With the assistance of this simplified structure, message exchange in security domains can be accelerated since the message is sent to the destination directly rather than passing through central managers. The second layer leverages the dynamic transaction collection period to reduce the key transfer time in vehicles handover. Through extensive simulations, it is shown in [182] that the proposed framework with the blockchain structure performs better than the existing one with a central manager.

7) Smart Grid: The smart grid is expected to provide complicated consumption monitoring and energy trading, which suffers from the security concerns, i.e., lack of privacy and anonymity. With the help of blockchain, the transaction security is provided in decentralized smart grid energy trading, relying on the cryptographic techniques instead of the trusted authorities. The study in [183] adopts and implements a proof-of-concept for decentralized energy trading system where all nodes jointly vote on validity of generated transactions by checking the history of publicly accessible distributed ledger. In particular, the nodes in smart grid can communicate through anonymous communication channel and trade energy ownership using distributed smart contracts, thus creating a dynamic market of energy trade. Such a market-based energy trade removes the dependency of a central energy provider and leads to an anonymous and decentralized environment. It is shown in [183] that the proposed decentralized smart grid energy trading framework is more reliable with higher security against common attacks, e.g., DDoS attack and double-spending attack, compared with traditional centralized trading solutions. However, since the proposed system in [183] is built upon the Bitcoin systems, the high computing power requirements incurred by PoW is difficult to satisfy in real smart grid environment.

A recent research [184] has studied Plug-in Hybrid Electric Vehicles (PHEVs) in smart girds, where PHEVs can share their surplus electricity through e-transportation meshes. However, this poses challenges in high cost, weak security, and the complex process of electricity trading. To achieve trustful, secure and decentralized electricity trading, the authors in [184] explore a promising consortium blockchain technology to audit and verify transaction records among PHEVs. Recall that the consortium blockchain is a specific blockchain with authorized nodes to maintain distributed shared databases. Energy transaction records among PHEVs are uploaded to the authorized nodes after encryption. The authorized nodes run an algorithm to audit and record the transactions into the publicly accessible shared ledger. It is indicated that the consortium blockchain has high potential to support decentralized electricity trading system with moderate cost. Based on [184], the authors in [185] employ consortium blockchain technology to develop a unified and secure P2P energy trading system in Industrial Internet of Things (IIoT), named energy blockchain. Similar to that in [184], the energy blockchain is established on the pre-selected authorized nodes to publicly audit and share transaction records in general energy trading scenarios, i.e., the consensus process is performed by pre-selected authorized nodes. The performance evaluation shows that the proposed energy blockchain in [185] supports fast P2P energy trading in IIoT. Additionally, it can also be applied in the vehicle-to-grid and energy harvesting networks.

B. Open Issues for Distributed Consensus in Blockchain Networks

1) Scalability and Efficiency Tradeoff: For blockchain networks, it is never exaggerated to reiterate the appropriate trade-off between scalability and efficiency, since they are the two opposite end of the consensus performance axis. Apart from the hybrid protocols discussed in Section V-B, it is worth noting that various approaches have been continuously proposed in both the research community and the industry. For example, the cryptocurrency communities tend to propose incremental improvement that emphasize backward compatibility to existing consensus protocol or network realization. These proposals include adjusting parameters of existing networks [91], e.g., SegWit for Bitcoin [186], and introducing hub-and-spoke topology among nodes, e.g., Lightning network [187], which thus requires lower frequency of block validation. However, due to the lack of theoretical supports, these proposals suffer from potential vulnerabilities or limitation in meeting the Byzantine agreement conditions.

An alternative design is to extend a existing, main blockchain network with multiple "sidechains" [188]. Sidechains are independent alter-chain networks "pegged" onto the main blockchain. The two-way peg mechanism of sidechains guarantee that the tokens can be freely moved between sidechains with atomic transfer. Such design introduce parallelism into the existing network and thus is able to increase the transaction throughput by adding more chains onto the main chain. Again, it is worth noting that the pegged sidechains inherit the same vulnerabilities of each alter-chain, and greatly increase the complexity of the system. Additional risks such as fraudulent token transfer are also introduced by the new mechanism.

Another way of network design focuses on the data organization for transactions. Therefore, instead of changing the block/transaction proposal mechanisms, such methodology indirectly addresses the scalability problem by changing the manner of transaction validation in the consensus layer. As we have briefly introduced in Section II-B, the typical examples of nonlinear topology for block organization include the protocol of Greedy Heaviest-Observed Sub-Tree (GHOST) [27], [39] and the block-DAG-based protocol SPECTRE [28], [189]. These proposed mechanisms are both extended from the standard PoW-based Nakamoto protocol. While the GHOST protocol forms a block tree that gradually converges to a single chain of blocks with the "greatest block weight", block DAG-based protocols no longer require the consensus nodes to reconcile their local views of the blockchain. Therefore, multiple blocks can be appended to the block DAG concurrently and more frequently than with the Nakamoto protocol. Compared with the heuristic scalability proposals [186]–[188], the block DAG-based protocol is better support by theoretically proved consensus properties such as probabilistic validity, agreement and weak liveness. However, since the block DAG is essentially an unordered representation of blocks, to provide the consensus properties, the consensus nodes have to go through complicated pairwise ordering between blocks by recursively tracing back to their ancestors in the DAG. For realization of large-scale networks, this will be resource consuming in terms of both computation and storage.

Again, it is worth noting that the engineering society also proposes several DAG-based for validated transaction storage, e.g., Byteball [190] and IoTA Tangle [26], [191]. One major difference of IoTA Tangle from the other DAG-based proposal lie in that it discards the concept of block as a package of transactions, and requires nodes to directly publish transactions onto the transaction DAG. A node is enforced by IoTA Tangle to approve more than two transactions by linking them as parents in the DAG to get its own new transaction weighted for ordering. Although analysis of incentive compatibility is provided in a framework of noncooperative game in [191], theoretical study of the consensus properties are yet missing for IoTA Tangle.

2) Cost of Decentralization: Although advocates of permissionles blockchain networks cherish the properties such as trustlessness and self-organization, decentralization with blockchain networks are not "at no cost", even with the problem of throughput-scalability addressed. One typical example is to realize the distributed, general-purpose VM atop blockchain consensus, since it requires every consensus node to execute the same smart contract locally to verify the state transition. Also, with the append-only blockchain structure, it is seemingly inevitable for the consensus process to concentrate to a few powerful nodes due to the limitation on a node's local storage.

Inspired by the infrasturctures of distributed databases and clouds, the concept of "sharding" [91] is also introduced into the blockchain networks [192]–[194]. Sharding a blockchain network is essentially partition the consensus nodes into subgroups, such that the nodes may only keep track of the transaction subset that is relevant to their own services. It is worth noting that sharding usually consist of two processes, namely, the sharding of the nodes as transaction validators and the sharding of the transactions. With the former process, multiple BFT committees can be constructed in parallel by extending a PoW-BFT hybrid protocol (see the protocol "Elastico" in [192]) and help improve the transaction throughput. With the latter process, microblocks are generated within each sub-blockchains by the corresponding sub-group of nodes, and the sub-blockchains are anchored periodically with the checkpoint blocks [193], [194] (cf. Casper [150]). However, sharding protocols are only able to reduce the storage burden of consensus nodes linearly as the number of shards increases.

Considering that historical data such as spent transactions become redundant as the blockchain grows, it is plausible to seek an approach for "pruning" the blockchain data without undermining its immutability. For cryptocurrency networks, hard forks such as SegWit [186] can be considered an manural pruning process, since only the block header carried forward by the fork is needed. However, it is better expected that the out-of-date blocks "have the right to be forgotten" [195]. Until recently, how to enable pruning in a blockchain generally remain an open question. A handful experimental proposal of pruning can be found in [196], [197]. In [196], a service-specified pruning method is proposed to allow the service-related transaction to have a expiry time. A transaction is separated into two parts, the transaction summary and the payload data. While syncing with the network, a node no longer needs to down load the payload data for future block validation. For Bitcoin-like networks, the concept of "account tree" in the form of a Merkle tree is proposed in [197] to replace the UTXO. Instead of keeping transaction records, a node only need to keep the Merkle root of the account tree in a block. In this sense, the nodes in the proposed networks in [196], [197] behave exactly as the Lightweight nodes in the Bitcoin network.

VII. Conclusions

In this paper, we have provided an comprehensive survey on the recent development of blockchain technologies, with a specific emphasis on the designing methodologies and related studies of (permissionless) distributed consensus protocols. We have provided in the survey a succinct overview of the implementation stacks for blockchain networks, from where we started our in-depth investigation into the design of consensus protocols and their impact on the emerging applications of blockchain networks. We have examined the influence of the blockchain consensus protocols from the perspective of three different interested parties, namely, the deployers of blockchain networks, the consensus participants (i.e., the consensus nodes) in the blockchain networks and the users of blockchain networks.

We have provided a thorough review of the blockchain consensus protocols including BFT-based protocols, Nakamoto protocols, virtual mining and hybrid protocols, for which we highlighted the link of permissionless consensus protocols to the tranditional Byzantine agreement protocols and their distinctive characteristics. We also highlighted the necessity of incentive compatibility in the protocol design, especially for the permissionless blockchain networks. We have provided an extensive survey on the studies regarding the incentive mechanism embedded in the blockchain protocols. From a game-theoretic perspective, we have also investigated their influence on the strategy adoption of the consensus participants in the blockchain network.

Based on our comprehensive survey of the protocol design and the consequent influence of the blockchain networks, we have provided an outlook on the emerging applications of blockchain networks in different areas. Our focus has been put upon how traditional problems, especially in the ares of wireless networking, can be reshaped with the introduction of blockchain networks. We hope this survey will serve as an efficient guideline for further understanding about blockchain technologies and for potential research directions that may lead to exciting outcomes in related areas.

REFERENCES

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” *Self-published Paper*, May 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] D. T. T. Anh, M. Zhang, B. C. Ooi, and G. Chen, “Untangling blockchain: A data processing view of blockchain systems,” *IEEE Transactions on Knowledge and Data Engineering*, vol. PP, no. 99, pp. 1–1, 2018.

[3] F. Tschorsch and B. Scheuermann, “Bitcoin and beyond: A technical survey on decentralized digital currencies,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 2084–2123, third quarter 2016.

[4] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten, “Sok: Research perspectives and challenges for bitcoin and cryptocurrencies,” in *2015 IEEE Symposium on Security and Privacy*, San Jose, CA, May 2015, pp. 104–121.

[5] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, “Hawk: The blockchain model of cryptography and privacy-preserving smart contracts,” in *2016 IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, May 2016, pp. 839–858.

[6] K. Yeow, A. Gani, R. W. Ahmad, J. J. P. C. Rodrigues, and K. Ko, “Decentralized consensus for edge-centric internet of things: A review, taxonomy, and research issues,” *IEEE Access*, vol. 6, pp. 1513–1524, 2018.

[7] K. Christidis and M. Devetsikiotis, “Blockchains and smart contracts for the internet of things,” *IEEE Access*, vol. 4, pp. 2292–2303, May 2016.

[8] F. Glaser, “Pervasive decentralisation of digital infrastructures: A framework for blockchain enabled system and use case analysis,” in *Proceedings of the 50th Hawaii International Conference on System Sciences*, Waikoloa, HI, Jan. 2017.

[9] N. Kshetri, “Can blockchain strengthen the internet of things?” *IT Professional*, vol. 19, no. 4, pp. 68–72, Aug. 2017.

[10] R. C. Merkle, “A digital signature based on a conventional encryption function,” in *Advances in Cryptology - CRYPTO ’87: Conference on the Theory and Applications of Cryptographic Techniques*, C. Pomerance, Ed., Santa Barbara, CA, Aug. 1987, pp. 369–378.

[11] A. Mohr, “A survey of zero-knowledge proofs with applications to cryptography,” Southern Illinois University, Carbondale, Tech. Rep., 2007.

[12] O. Goldreich, “Zero-knowledge twenty years after its invention.” *IACR Cryptology ePrint Archive*, vol. 2002, p. 186, 2002.

[13] M. Raynal, “Communication and agreement abstractions for fault-tolerant asynchronous distributed systems,” *Synthesis Lectures on Distributed Computing Theory*, vol. 1, no. 1, pp. 1–273, May 2010.

[14] F. B. Schneider, “Implementing fault-tolerant services using the state machine approach: A tutorial,” *ACM Computing Surveys*, vol. 22, no. 4, pp. 299–319, Dec. 1990.

[15] S. Bano, A. Sonnino, M. Al-Bassam, S. Azouvi, P. McCorry, S. Meiklejohn, and G. Danezis, “Consensus in the age of blockchains,” *arXiv preprint arXiv:1711.03936*, 2017.

[16] N. Bozic, G. Pujolle, and S. Secci, “Securing virtual machine orchestration with blockchains,” in *2017 1st Cyber Security in Networking Conference (CSNet)*, Rio de Janeiro, Brazil, Oct. 2017, pp. 1–8.

[17] D. Chatzopoulos, M. Ahmadi, S. Kosta, and P. Hui, “Flopcoin: A cryptocurrency for computation offloading,” *IEEE Transactions on Mobile Computing*, vol. PP, no. 99, pp. 1–1, 2017.

[18] M. Castro and B. Liskov, “Practical byzantine fault tolerance and proactive recovery,” *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, Nov. 2002.

[19] M. Vukolić, “The quest for scalable blockchain fabric: Proof-of-work vs. bft replication,” in *Open Problems in Network Security: IFIP WG 11.4 International Workshop*, Zurich, Switzerland, Oct. 2015, pp. 112–125.

[20] Z. Zheng, S. Xie, H.-N. Dai, and H. Wang, “Blockchain challenges and opportunities: A survey,” School of Data and Computer Science, Sun Yat-sen University, Tech. Rep., 2016.

[21] M. Conti, C. Lal, S. Ruj *et al.*, “A survey on security and privacy issues of bitcoin,” *arXiv preprint arXiv:1706.00916*, 2017.

[22] N. Atzei, M. Bartoletti, and T. Cimoli, “A survey of attacks on ethereum smart contracts (sok),” in *Principles of Security and Trust: 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software*, Uppsala, Sweden, Apr. 2017, pp. 164–186.

[23] K. Christidis and M. Devetsikiotis, “Blockchains and smart contracts for the internet of things,” *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[24] S. Ghosh, *Distributed systems: an algorithmic approach*. CRC press, 2014.

[25] J. Garay, A. Kiayias, and N. Leonardos, “The bitcoin backbone protocol: Analysis and applications,” in *Advances in Cryptology - EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Part II*, Sofia, Bulgaria, Apr. 2015, pp. 281–310.

[26] S. Popov, “The tangle version 1.3,” IOTA Foundation, Tech. Rep., Oct. 2017.

[27] Y. Sompolinsky and A. Zohar, “Secure high-rate transaction processing in bitcoin,” in *19th International Conference on Financial Cryptography and Data Security*, San Juan, Puerto Rico, Jan. 2015, pp. 507–527.

[28] Y. Sompolinsky, Y. Lewenberg, and A. Zohar, “Spectre: A fast and scalable cryptocurrency protocol.” *IACR Cryptology ePrint Archive*, vol. 2016, p. 1159, 2016.

[29] P. Maymounkov and D. Mazières, “Kademlia: A peer-to-peer information system based on the xor metric,” in *Peer-to-Peer Systems: First International Workshop*, Cambridge, MA, Mar. 2002, pp. 53–65.

[30] Ethereum Foundation, “Ethereum wire protocol v.5,” https://github.com/ethereum/wiki/wiki/Ethereum-Wire-Protocol, accessed: 2017-11-15.

[31] ——, “Whisper protocol v.5,” https://github.com/ethereum/go-ethereum/wiki/Whisper-
accessed: 2017-11-15.

[32] https://github.com/telehash/telehash.github.io, accessed: 2017-11-15.

[33] JSON-RPC Working Group, “Json-rpc 2.0 specification,” 2012.

[34] D. Schwartz, N. Youngs, and A. Britto, “The ripple protocol consensus algorithm,” *Ripple Labs Inc White Paper*, vol. 5, 2014.

[35] I. Bentov, A. Gabizon, and A. Mizrahi, “Cryptocurrencies without proof of work,” in *International Conference on Financial Cryptography and Data Security*, Christ Church, Barbados, Feb. 2016, pp. 142–157.

[36] C. Cachin, “Architecture of the hyperledger blockchain fabric,” in *Workshop on Distributed Cryptocurrencies and Consensus Ledgers (DCCL 2016)*, Chicago, IL, Jul. 2016.

[37] V. Buterin, “Ethereum: A next-generation smart contract and decentralized application platform,” Ethereum Foundation, Tech. Rep., 2014. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper

[38] Protocol Labs, “Filecoin: A decentralized storage network,” Protocol Labs, Tech. Rep., Aug. 2017.

[39] A. Kiayias and G. Panagiotakos, “On trees, chains and fast transactions in the blockchain.” *IACR Cryptology ePrint Archive*, vol. 545, 2016.

[40] J. Backman, S. Yrjl, K. Valtanen, and O. Mmmel, “Blockchain network slice broker in 5g: Slice leasing in factory of the future use case,” in *2017 Internet of Things Business Models, Users, and Networks*, Copenhagen, Denmark, Nov. 2017, pp. 1–8.

[41] A. Mackenzie, S. Noether, and Monero Core Team, “Improving obfuscation in the cryptonote protocol,” Monero Research Lab, Tech. Rep., Jan. 2015.

[42] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox, “Zcash protocol specification,” Zerocoin Electric Coin Company, Tech. Rep., Dec. 2017.

[43] C. Decker and R. Wattenhofer, “Information propagation in the bitcoin network,” in *Proceedings of IEEE International Conference on Peer-to-Peer Computing*, Trento, Italy, Sep. 2013, pp. 1–10.

[44] A. Biryukov, D. Khovratovich, and I. Pustogarov, “Deanonymisation of clients in bitcoin p2p network,” in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’14, Scottsdale, AZ, USA, 2014, pp. 15–29.

[45] A. E. Gencer, S. Basu, I. Eyal, R. van Renesse, and E. G. Sirer, “Decentralization in bitcoin and ethereum networks,” *arXiv preprint arXiv:1801.03998*, 2018.

[46] A. M. Antonopoulos, *Mastering Bitcoin: unlocking digital cryptocurrencies*. ” O’Reilly Media, Inc.”, 2014.

[47] J. R. Douceur, “The sybil attack,” in *First International Workshop on Peer-to-Peer Systems*, Cambridge, MA, Mar. 2002, pp. 251–260.

[48] V. Buterin, “Bitcoin network shaken by blockchain fork,” *Bitcoin Mangazine*, vol. 12, Mar. 2013. [Online]. Available: https://bitcoinmagazine.com/articles/bitcoin-network-shaken-by-blockchain-fork-1363

[49] C. Cachin, K. Kursawe, F. Petzold, and V. Shoup, “Secure and efficient asynchronous broadcast protocols,” in *Advances in Cryptology – CRYPTO 2001: 21st Annual International Cryptology Conference*. Santa Barbara, CA: Springer Berlin Heidelberg, Aug. 2001, pp. 524–541.

[50] M. Correia, N. F. Neves, and P. Verssimo, “From consensus to atomic broadcast: Time-free byzantine-resistant protocols without signatures,” *The Computer Journal*, vol. 49, no. 1, pp. 82–96, Jan. 2006.

[51] C. Cachin and M. Vukolić, “Blockchains consensus protocols in the wild,” *arXiv preprint arXiv:1707.01873*, 2017.

[52] A. Miller and J. J. LaViola Jr, "Anonymous byzantine consensus from moderately-hard puzzles: A model for bitcoin," University of Central Florida, Computer Science, Tech. Rep., Apr. 2014. [Online]. Available: http://nakamotoinstitute.org/research/anonymous-byzantine-consensus

[53] F. Sun and P. Duan, "Solving byzantine problems in synchronized systems using bitcoin," *Self-published Paper*, Sep. 2014. [Online]. Available: https://allquantor.at/blockchainbib/pdf/sun2014solving.pdf

[54] S. Liu, P. Viotti, C. Cachin, V. Quéma, and M. Vukolic, "Xft: Practical fault tolerance beyond crashes." in *12th USENIX Symposium on Operating Systems Design and Implementation*, Savannah, GA, Nov. 2016, pp. 485–500.

[55] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, *Algorithmic game theory*. Cambridge University Press Cambridge, 2007, vol. 1.

[56] M. Babaioff, S. Dobzinski, S. Oren, and A. Zohar, "On bitcoin and red balloons," in *Proceedings of the 13th ACM Conference on Electronic Commerce*, ser. EC '12. New York, NY: ACM, Jun. 2012, pp. 56–73.

[57] S. Athey, I. Parashkevov, V. Sarukkai, and J. Xia, "Bitcoin pricing, adoption, and usage: Theory and evidence," Stanford Institute for Economic Policy Research, Tech. Rep. Working Paper No. 3469, Aug. 2016.

[58] A. Gervais, G. O. Karame, V. Capkun, and S. Capkun, "Is bitcoin a decentralized currency?" *IEEE Security Privacy*, vol. 12, no. 3, pp. 54–60, May 2014.

[59] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable (revised selected papers)," in *Financial Cryptography and Data Security: 18th International Conference*, Christ Church, Barbados, Mar. 2014, pp. 436–454.

[60] O. Ersoy, Z. Ren, Z. Erkin, and R. L. Lagendijk, "Information propagation on permissionless blockchains," *arXiv preprint arXiv:1712.07564*, 2017.

[61] J. Debus, "Consensus methods in blockchain systems," Frankfurt School of Finance & Management, Blockchain Center, Tech. Rep., May 2017.

[62] J. A. Kroll, I. C. Davey, and E. 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)*, vol. 2013, Washington, D.C., Jun. 2013.

[63] M. Rosenfeld, "Analysis of bitcoin pooled mining reward systems," *arXiv preprint arXiv:1112.4980*, 2011.

[64] J. Zou, B. Ye, L. Qu, Y. Wang, M. A. Orgun, and L. Li, "A proof-of-trust consensus protocol for enhancing accountability in crowdsourcing services," *IEEE Transactions on Services Computing*, pp. 1–1, 2018.

[65] C. Cachin, "Yet another visit to paxos," *IBM Research, Zurich, Switzerland, Tech. Rep. RZ3754*, 2009.

[66] J. Sousa, A. Bessani, and M. Vukolić, "A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform," *arXiv preprint arXiv:1709.06921*, 2017.

[67] H. Kopp, C. Bösch, and F. Kargl, "Koppercoin – a distributed file storage with financial incentives," in *12th International Conference on Information Security Practice and Experience*, Zhangjiajie, China, Nov. 2016, pp. 79–93.

[68] M. Jakobsson and A. Juels, "Proofs of work and bread pudding protocols (extended abstract)," in *Secure Information Networks: Communications and Multimedia Security IFIP TC6/TC11 Joint Working Conference on Communications and Multimedia Security (CMS'99)*, Leuven, Belgium, Sep. 1999, pp. 258–272.

[69] J. Aspnes, C. Jackson, and A. Krishnamurthy, "Exposing computationally challenged byzantine impostors," Yale University, Tech. Rep. YALEU/DCS/TR-1332, 2005.

[70] J. Alwen and B. Tackmann, "Moderately hard functions: Definition, instantiations, and applications," in *Proceedings of the 15th International Conference on Theory of Cryptography: Part I*, Baltimore, MD, Nov. 2017, pp. 493–526.

[71] S. Micali, M. Rabin, and S. Vadhan, "Verifiable random functions," in *40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*, New York City, NY, Oct. 1999, pp. 120–130.

[72] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "Snarks for c: Verifying program executions succinctly and in zero knowledge," in *Advances in Cryptology – CRYPTO 2013: 33rd Annual Cryptology Conference*, Santa Barbara, CA, Aug. 2013, pp. 90–108.

[73] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *Self-published Paper*, Aug. 2012. [Online]. Available: https://peercoin.net/assets/paper/peercoin-paper.pdf

[74] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Advances in Cryptology – CRYPTO 2017: 37th Annual International Cryptology Conference*, Santa Barbara, CA, Aug. 2017, pp. 357–388.

[75] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, "Proof of activity: Extending bitcoin's proof of work via proof of stake (extended abstract)," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 3, pp. 34–37, Dec. 2014.

[76] M. Milutinovic, W. He, H. Wu, and M. Kanwal, "Proof of luck: An efficient blockchain consensus protocol," in *Proceedings of the 1st Workshop on System Software for Trusted Execution*, ser. SysTEX '16, Trento, Italy, Dec. 2016, pp. 2:1–2:6.

[77] A. Biryukov and D. Khovratovich, "Equihash: Asymmetric proof-of-work based on the generalized birthday problem," *Ledger Journal*, vol. 2, pp. 1–30, Apr. 2017.

[78] A. Miller, A. Kosba, J. Katz, and E. Shi, "Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. Denver, CO: ACM, Oct. 2015, pp. 680–691.

[79] T. Moran and I. Orlov, "Rational proofs of space-time," Bar-Ilan University Cyber Center, Tech. Rep., 2017.

[80] M. Ball, A. Rosen, M. Sabin, and P. N. Vasudevan, "Proofs of useful work," *IACR Cryptology ePrint Archive*, p. 203, Feb. 2017.

[81] S. Al-Kuwari, J. H. Davenport, and R. J. Bradford, "Cryptographic hash functions: Recent design trends and security notions," *IACR Cryptology ePrint Archive*, 2011.

[82] J. A. Garay, A. Kiayias, and G. Panagiotakos, "Proofs of work for blockchain protocols," *IACR Cryptology ePrint Archive, Report 2017/775*, Aug. 2017.

[83] D. Kraft, "Difficulty control for blockchain-based consensus systems," *Peer-to-Peer Networking and Applications*, vol. 9, no. 2, pp. 397–413, Mar 2016.

[84] K. Saito and H. Yamada, "What's so different about blockchain? – blockchain is a probabilistic state machine," in *2016 IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, Nara, Japan, Jun. 2016, pp. 168–175.

[85] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol with chains of variable difficulty," in *Advances in Cryptology – CRYPTO 2017*, Santa Barbara, CA, Aug. 2017, pp. 291–323.

[86] L. Fan and H.-S. Zhou, "A scalable proof-of-stake blockchain in the open setting," *Cryptology ePrint Archive, Report 2017/656*, 2017. [Online]. Available: https://eprint.iacr.org/2017/656

[87] M. B. Taylor, "The evolution of bitcoin hardware," *Computer*, vol. 50, no. 9, pp. 58–66, 2017.

[88] www.coinmarketcap.com, https://coinmarketcap.com/coins/views/all/, accessed: 2017-11-15.

[89] A. Kiayias and G. Panagiotakos, "Speed-security tradeoffs in blockchain protocols." *IACR Cryptology ePrint Archive*, p. 1019, Oct. 2016.

[90] R. Pass, L. Seeman, and A. Shelat, "Analysis of the blockchain protocol in asynchronous networks," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Paris, France, May 2017, pp. 643–673.

[91] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. Gün Sirer, D. Song, and R. Wattenhofer, "On scaling decentralized blockchains," in *Financial Cryptography and Data Security: International Workshops on BITCOIN, VOTING and WAHC*, Christ Church, Barbados, Feb. 2016, pp. 106–125.

[92] P. R. Rizun, "Subchains: A technique to scale bitcoin and improve the user experience," *Ledger*, vol. 1, pp. 38–52, 2016.

[93] X. Liu, W. Wang, D. Niyato, N. Zhao, and P. Wang, "Evolutionary game for mining pool selection in blockchain networks," *IEEE Wireless Communications Letters*, pp. 1–1, 2018.

[94] P. R. Rizun, "A transaction fee market exists without a block size limit," *Self-published Paper*, Aug. 2015.

[95] M. Ghosh, M. Richardson, B. Ford, and R. Jansen, "A torpath to torcoin: proof-of-bandwidth altcoins for compensating relays," U.S. Naval Research Laboratory, Washington, DC, Tech. Rep., 2014.

[96] F. Zhang, I. Eyal, R. Escriva, A. Juels, and R. V. Renesse, "REM: Resource-efficient mining for blockchains," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1427–1444.

[97] S. Park, K. Pietrzak, J. Alwen, G. Fuchsbauer, and P. Gazi, "Spacecoin: A cryptocurrency based on proofs of space," MIT, Tech. Rep., Jun. 2015.

[98] J. Blocki and H.-S. Zhou, "Designing proof of human-work puzzles for cryptocurrency and beyond," in *14th International Conference on Theory of Cryptography*, Beijing, China, Oct. 2016, pp. 517–546.

[99] S. King, "Primecoin: Cryptocurrency with prime number proof-of-work," *Self-published Paper*, Jul. 2013. [Online]. Available: http://primecoin.io/bin/primecoin-paper.pdf

[100] J. Andersen and E. Weisstein, "Cunningham chain. from mathworld–a wolfram web resource," 2005. [Online]. Available: http://mathworld.wolfram.com/CunninghamChain.html

[101] A. Shoker, "Sustainable blockchain through proof of exercise," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, Cambridge, MA, Oct. 2017, pp. 1–9.

[102] M. Ball, A. Rosen, M. Sabin, and P. N. Vasudevan, "Average-case fine-grained hardness," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC 2017, Montreal, Canada, 2017, pp. 483–496.

[103] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proceedings on Advances in cryptology—CRYPTO '86*, Santa Barbara, California, USA, Aug. 1986, pp. 186–194.

[104] S. Johnson, V. Scarlata, C. Rozas, E. Brickell, and F. Mckeen, "Intel® software guard extensions: Epid provisioning and attestation services," Intel, Tech. Rep., 2016.

[105] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz, "Permacoin: Repurposing bitcoin work for data preservation," in *2014 IEEE Symposium on Security and Privacy*, San Jose, CA, May 2014, pp. 475–490.

[106] A. Juels and B. S. Kaliski, Jr., "Pors: Proofs of retrievability for large files," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*. New York, NY, USA: ACM, Oct. 2007, pp. 584–597.

[107] S. Wilkinson, "Storj a peer-to-peer cloud storage network," Storj Labs Inc., Tech. Rep., Dec. 2014. [Online]. Available: https://storj.io/storj.pdf

[108] D. Vorick and L. Champine, "Sia: Simple decentralized storage," Nebulous Inc., Tech. Rep., Nov. 2014. [Online]. Available: https://coss.io/documents/white-papers/siacoin.pdf

[109] D. Wagner, "A generalized birthday problem," in *Advances in Cryptology — CRYPTO 2002*, Santa Barbara, California, Aug. 2002, pp. 288–304.

[110] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger (eip-150 revision)," *Ethereum Project Yellow Paper*, vol. 151, 2017.

[111] P. Daian, I. Eyal, A. Juels, and E. G. Sirer, "(short paper) piecework: Generalized outsourcing control for proofs of work," in *Financial Cryptography and Data Security: FC 2017 International Workshops on WAHC, BITCOIN, VOTING, WTSC, and TA*, Sliema, Malta, Apr. 2017, pp. 182–190.

[112] S. Dziembowski, S. Faust, V. Kolmogorov, and K. Pietrzak, "Proofs of space," in *Advances in Cryptology – CRYPTO 2015: 35th Annual Cryptology Conference*, Santa Barbara, CA, Aug. 2015, pp. 585–605.

[113] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford, "Captcha: Using hard ai problems for security," in *Advances in Cryptology – EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques*, Warsaw, Poland, May 2003, pp. 294–311.

[114] D. Hofheinz, T. Jager, D. Khurana, A. Sahai, B. Waters, and M. Zhandry, "How to generate and use universal samplers," in *Advances in Cryptology – ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security*, Hanoi, Vietnam, Dec. 2016, pp. 715–744.

[115] N. T. Courtois, "On the longest chain rule and programmed self-destruction of crypto currencies," *arXiv preprint arXiv:1405.0534*, 2014.

[116] R. Recabarren and B. Carbunar, "Hardening stratum, the bitcoin pool mining protocol," *arXiv preprint arXiv:1703.06545*, 2017.

[117] A. Laszka, B. Johnson, and J. Grossklags, "When bitcoin mining pools run dry," in *Financial Cryptography and Data Security: FC 2015 International Workshops on BITCOIN, WAHC and Wearable*, San Juan, Puerto Rico, Jan. 2015, pp. 63–77.

[118] M. Maschler, E. Solan, and S. Zamir, *Game Theory*. Cambridge University Press, 2013.

[119] I. Abraham, D. Malkhi, K. Nayak, L. Ren, and A. Spiegelman, "Solidus: An incentive-compatible cryptocurrency based on permissionless byzantine consensus," *arXiv preprint arXiv:1612.02916*, 2016.

[120] C. Decker, J. Seidel, and R. Wattenhofer, "Bitcoin meets strong consistency," in *Proceedings of the 17th International Conference on Distributed Computing and Networking*, ser. ICDCN '16, Singapore, 2016, pp. 13:1–13:10.

[121] A. Stone, "An examination of single transaction blocks and their effect on network throughput and block size," *Self-published Paper*, Jun. 2015. [Online]. Available: http://ensocoin.org/resources/1txn.pdf

[122] K. Baqer, D. Y. Huang, D. McCoy, and N. Weaver, "Stressing out: Bitcoin "stress testing"," in *Financial Cryptography and Data Security: International Workshops on BITCOIN, VOTING and WAHC*, Christ Church, Barbados, Feb. 2016, pp. 3–18.

[123] G. Pappalardo, T. Di Matteo, G. Caldarelli, and T. Aste, "Blockchain inefficiency in the bitcoin peers network," *arXiv preprint arXiv:1704.01414*, 2017.

[124] M. Möser and R. Böhme, *Trends, Tips, Tolls: A Longitudinal Study of Bitcoin Transaction Fees*, San Juan, Puerto Rico, Jan. 2015, pp. 19–33.

[125] N. Houy, "The economics of bitcoin transaction fees," GATE Groupe d'Analyse et de Théorie Économique Lyon-St Étienne, Tech. Rep., Feb. 2014.

[126] S. Feng, W. Wang, Z. Xiong, D. Niyato, P. Wang, and S. Wang, "On cyber risk management of blockchain networks: A game theoretic approach," *arXiv preprint arXiv:1804.10412*, 2018.

[127] N. Dimitri, "Bitcoin mining as a contest," *Ledger Journal*, vol. 2, pp. 31–37, Apr. 2017.

[128] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Optimal pricing-based edge computing resource management in mobile blockchain," in *2018 IEEE International Conference on Communications (ICC)*, Kansas City, Kansas, May 2018.

[129] Y. Jiao, P. Wang, D. Niyato, and Z. Xiong, "Social welfare maximization auction in edge computing resource allocation for mobile blockchain," in *2018 IEEE International Conference on Communications (ICC)*, Kansas City, Kansas, May 2018.

[130] N. Houy, "The bitcoin mining game," *Ledger Journal*, vol. 1, no. 13, pp. 53 – 68, 2016.

[131] K. Nayak, S. Kumar, A. Miller, and E. Shi, "Stubborn mining: Generalizing selfish mining and combining with an eclipse attack," in *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, Saarbrücken, Germany, Mar. 2016, pp. 305–320.

[132] M. Carlsten, "The impact of transaction fees on bitcoin mining strategies," Master's thesis, Princeton University, 2016.

[133] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *Financial Cryptography and Data Security: 20th International Conference, Revised Selected Papers*, Christ Church, Barbados, Feb. 2017, pp. 515–532.

[134] Y. Sompolinsky and A. Zohar, "Bitcoin's security model revisited," *arXiv preprint arXiv:1605.09193*, 2016.

[135] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. Vienna, Austria: ACM, Oct. 2016, pp. 3–16.

[136] J. Gbel, H. Keeler, A. Krzesinski, and P. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," *Performance Evaluation*, vol. 104, no. Supplement C, pp. 23 – 41, 2016.

[137] J. Beccuti and C. Jaag, "The bitcoin mining game: On the optimality of honesty in proof-of-work consensus mechanism," Swiss Economics, Working Papers 0060, Aug. 2017. [Online]. Available: https://ideas.repec.org/p/chc/wpaper/0060.html

[138] D. K. Tosh, S. Shetty, X. Liang, C. A. Kamhoua, K. A. Kwiat, and L. Njilla, "Security implications of blockchain cloud with analysis of block withholding attack," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, Madrid, Spain, May 2017, pp. 458–467.

[139] L. Luu, R. Saha, I. Parameshwaran, P. Saxena, and A. Hobor, "On power splitting games in distributed computation: The case of bitcoin pooled mining," in *2015 IEEE 28th Computer Security Foundations Symposium*, Verona, Italy, Jul. 2015, pp. 397–411.

[140] I. Eyal, "The miner's dilemma," in *2015 IEEE Symposium on Security and Privacy*, San Jose, CA, May 2015, pp. 89–103.

[141] S. Bag, S. Ruj, and K. Sakurai, "Bitcoin block withholding attack: Analysis and mitigation," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1967–1978, Aug. 2017.

[142] S. Bag and K. Sakurai, "Yet another note on block withholding attack on bitcoin mining pools," in *19th International Conference on Information Security*, Honolulu, HI, Sep. 2016, pp. 167–180.

[143] "Slushpool," Dec. 2017. [Online]. Available: https://slushpool.com/home/

[144] A. Kiayias, I. Konstantinou, A. Russell, B. David, and R. Oliynykov, "A provably secure proof-of-stake blockchain protocol," *IACR Cryptology ePrint Archive*, vol. 2016, p. 889, 2016.

[145] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poet)," in *19th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, Boston, MA, Nov. 2017, pp. 282–297.

[146] "Slimcoin: A peer-to-peer crypto-currency with proof-of-burn," www.slimcoin.org, Tech. Rep., May 2014. [Online]. Available: https://github.com/slimcoin-project/slimcoin-project.github.io/blob/master/whitepaperSLM.pdf

[147] L. Ren, "Proof of stake velocity: Building the social currency of the digital age," *Self-published Paper*, Apr. 2014. [Online]. Available: https://coss.io/documents/white-papers/reddcoin.pdf

[148] I. Bentov, R. Pass, and E. Shi, "Snow white: Provably secure proofs of stake," *IACR Cryptology ePrint Archive*, vol. 2016, p. 919, Sep. 2016.

[149] B. David, P. Gaži, A. Kiayias, and A. Russell, "Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain," in *Advances in Cryptology – EUROCRYPT 2018*, Tel Aviv, Israel, Apr. 2018, pp. 66–98.

[150] V. Buterin and V. Griffith, "Casper the friendly finality gadget," *arXiv preprint arXiv:1710.09437*, 2017.

[151] W. Li, S. Andreina, J.-M. Bohli, and G. Karame, "Securing proof-of-stake blockchain protocols," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology: ESORICS 2017 International Workshops*, Oslo, Norway, Sep. 2017, pp. 297–315.

[152] A. Poelstra, "Distributed consensus from proof of stake is impossible," *Self-published Paper*, May 2014. [Online]. Available: https://download.wpsoftware.net/bitcoin/old-pos.pdf

[153] N. Houy, "It will cost you nothing to kill a proof-of-stake crypto-currency," GATE Groupe d'Analyse et de Théorie Économique Lyon-St Étienne, Tech. Rep., 2014.

[154] I. Eyal, A. E. Gencer, E. G. Sirer, and R. V. Renesse, "Bitcoin-ng: A scalable blockchain protocol," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. Santa Clara, CA: USENIX Association, Mar. 2016, pp. 45–59.

[155] R. Pass and E. Shi, "Hybrid Consensus: Efficient Consensus in the Permissionless Model," in *31st International Symposium on Distributed Computing (DISC 2017)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), A. W. Richa, Ed., vol. 91. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Oct. 2017, pp. 39:1–39:16.

[156] M. K. Reiter, "A secure group membership protocol," *IEEE Transactions on Software Engineering*, vol. 22, no. 1, pp. 31–42, Jan. 1996.

[157] E. K. Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford, "Enhancing bitcoin security and performance with strong consistency via collective signing," in *25th USENIX Security Symposium (USENIX Security 16)*, Austin, TX, Aug. 2016, pp. 279–296.

[158] J. Kwon, "Tendermint: Consensus without mining (draft)," *Self-published Paper*, fall 2014. [Online]. Available: https://tendermint.com/static/docs/tendermint.pdf

[159] "Proof of authority chains," Jan. 2018. [Online]. Available: https://github.com/paritytech/parity

[160] D. Larimer, "Delegated proof-of-stake (dpos)," Bitshare whitepaper, Tech. Rep., 2014.

[161] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*. Shanghai, China: ACM, Oct. 2017, pp. 51–68.

[162] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach," in *2018 IEEE International Conference on Communications (ICC)*, Kansas City, Kansas, May 2018.

[163] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing: Challenges and applications," *arXiv preprint arXiv:1711.05938*, 2017.

[164] K. Suankaewmanee, D. T. Hoang, D. Niyato, S. Sawadsitang, P. Wang, and Z. Han, "Performance analysis and application of mobile blockchain," in *IEEE International Conference on Computing, Networking and Communications*, Maui, Hawaii, Mar. 2018.

[165] N. Fotiou and G. C. Polyzos, "Decentralized name-based security for content distribution using blockchains," in *2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, San Francisco, CA, Apr. 2016, pp. 415–420.

[166] C. Fang, H. Yao, Z. Wang, W. Wu, X. Jin, and F. R. Yu, "A survey of mobile information-centric networking: Research issues and challenges," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2018.

[167] A. Lewko and B. Waters, "Unbounded hibe and attribute-based encryption," in *Advances in Cryptology – EUROCRYPT 2011*, Tallinn, Estonia, May 2011, pp. 547–567.

[168] T.-H. Chang and D. Svetinovic, "Data analysis of digital currency networks: Namecoin case study," in *2016 21st International Conference on Engineering of Complex Computer Systems (ICECCS)*, Dubai, Nov. 2016, pp. 122–125.

[169] T. Jin, X. Zhang, Y. Liu, and K. Lei, "Blockndn: A bitcoin blockchain decentralized system over named data networking," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, Milan, Italy, Jul. 2017, pp. 75–80.

[170] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability," in *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, Madrid, Spain, May 2017, pp. 468–477.

[171] N. Kaaniche and M. Laurent, "A blockchain-based data usage auditing architecture with enhanced privacy and availability," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, Cambridge, MA, Oct. 2017, pp. 1–5.

[172] N. Herbaut and N. Negru, "A model for collaborative blockchain-based video delivery relying on advanced network services chains," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 70–76, 2017.

[173] W. Wang, D. Niyato, P. Wang, and A. Leshem, "Decentralized caching for content delivery based on blockchain: A game theoretic perspective," in *IEEE ICC 2018 Next Generation Networking and Internet Symposium*, Kansas City, USA, May 2018.

[174] F. R. Yu, M. Huang, and H. Tang, "Biologically inspired consensus-based spectrum sensing in mobile ad hoc networks with cognitive radios," *IEEE Network*, vol. 24, no. 3, pp. 26–30, May 2010.

[175] S. Raju, S. Boddepalli, S. Gampa, Q. Yan, and J. Deogun, "Identity management using blockchain for cognitive cellular networks," in *2017 IEEE International Conference on Communications (ICC)*, Paris, France, May 2017, pp. 1–6.

[176] S. Raju, S. Boddepalli, N. Choudhury, Q. Yan, and J. S. Deogun, "Design and analysis of elastic handoff in cognitive cellular networks," in *2017 IEEE International Conference on Communications (ICC)*, Paris, France, May 2017.

[177] K. Kotobi and S. G. Bilén, "Blockchain-enabled spectrum access in cognitive radio networks," in *Wireless Telecommunications Symposium (WTS), 2017*, Chicago, IL, Apr. 2017, pp. 1–6.

[178] M. Li, J. Weng, A. Yang, and W. Lu, "Crowdbc: A blockchain-based decentralized framework for crowdsourcing," Online, available at: https://eprint.iacr.org/2017/444.pdf.

[179] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for iot security and privacy: The case study of a smart home," in *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, Kona, Big Island, Hawaii, Mar. 2017, pp. 618–623.

[180] ——, "Lsb: A lightweight scalable blockchain for iot security and privacy," *arXiv preprint arXiv:1712.02969*, 2017.

[181] S. Rafaeli and D. Hutchison, "A survey of key management for secure group communication," *ACM Computing Surveys (CSUR)*, vol. 35, no. 3, pp. 309–329, 2003.

[182] A. Lei, H. Cruickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, "Blockchain-based dynamic key management for heterogeneous intelligent transportation systems," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1832–1843, 2017.

[183] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2016.

[184] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, pp. 1–1, 2017.

[185] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2017.

[186] E. Lombrozo, J. Lau, and P. Wuille, "Segregated witness (consensus layer)," Tech. Rep., Dec. 2015. [Online]. Available: https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki

[187] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," Lightning Labs, Tech. Rep., Nov. 2016.

[188] A. Back, M. Corallo, L. Dashjr, M. Friedenbach, G. Maxwell, A. Miller, A. Poelstra, J. Timón, and P. Wuille, "Enabling blockchain innovations with pegged sidechains," Blockstream Inc., Tech. Rep. [Online]. Available: http://kevinriggen.com/files/sidechains.pdf

[189] Y. Lewenberg, Y. Sompolinsky, and A. Zohar, "Inclusive block chain protocols," in *Financial Cryptography and Data Security*, San Juan, Puerto Rico, Jan. 2015, pp. 528–547.

[190] A. Churyumov, "Byteball: a decentralized system for storage and transfer of value," 2017.

[191] S. Popov, O. Saa, and P. Finardi, "Equilibria in the tangle," *arXiv preprint arXiv:1712.05385*, 2017.

[192] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. Vienna, Austria: ACM, Oct. 2016, pp. 17–30.

[193] A. E. Gencer, R. van Renesse, and E. G. Sirer, "Short paper: Service-oriented sharding for blockchains," in *Financial Cryptography and Data Security*, Sliema, Malta, Apr. 2017, pp. 393–401.

[194] Z. Ren and Z. Erkin, "A scale-out blockchain for value transfer with spontaneous sharding," *arXiv preprint arXiv:1801.02531*, 2018.

[195] R. Dennis, G. Owenson, and B. Aziz, "A temporal blockchain: A formal analysis," in *2016 International Conference on Collaboration Technologies and Systems (CTS)*, Orlando, FL, Oct. 2016, pp. 430–437.

[196] J. Sidhu, "Syscoin: A peer-to-peer electronic cash system with blockchain-based services for e-business," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, Vancouver, BC, Jul. 2017, pp. 1–6.

[197] J. Bruce, "The mini-blockchain scheme rev 3," *Self-published Paper*, Mar. 2017. [Online]. Available: http://cryptonite.info/files/mbc-scheme-rev3.pdf