Introduction to Proof-of-Stake

Simas Šakenis

January 24, 2022

Abstract

The central technical challenge of any cryptocurrency system is maintaining an extendable public ledger in a decentralized fashion. The participating entities have to be able to agree on which version of the ledger is correct at any point in time, without delegating trust to any single authority. Such an agreement is made possible by consensus mechanisms. Despite being the most widely used to date, the proof-of-work consensus mechanism has turned out to yield undesirable economic incentives (e.g., encouraged consumption of power) which makes it unsustainable. The most popular among the proposed alternatives circumventing this issue is proof-of-stake, a consensus mechanism recently adopted by the second-largest cryptocurrency, Ethereum. The goal of this work is to give a formal introduction to proof-of-stake. We proceed by presenting a cryptographic framework in which both proof-of-work and proof-of-stake are formalized, comparing the practical sustainability of the two consensus mechanisms, stating the key security properties of proof-of-stake, and surveying some open problems.

1 Introduction

A cryptocurrency system is an interactive protocol between a set of participants that enables them to exchange a virtual resource [Tschorsch and Scheuermann, 2016]. Let $P_1, P_2, P_3, \ldots$ denote the participants of the protocol (the number of participants is not fixed because anyone is free to join or leave the protocol at any time). Each participant P_i is associated with a unique identity, which is the public-private key pair known to P_i that we denote by $(\text{PK}_i, \text{SK}_i)$. The virtual resource being exchanged is digital coins.

At any given point in time, the state of the cryptocurrency system is fully described by a list of transactions recorded up to that point. We denote the stream of transactions by $X_1, X_2, X_3, \ldots$. A transaction is a transfer or an assignment of ownership of digital coins. A transaction may or may not have a sender. If a transaction does not have a sender, it is said to "mine" digital coins, as it assigns ownership to coins that previously have not been owned by anyone (which makes it analogous to mining a natural resource such as gold). The initial state of the cryptocurrency system consists of some number n of such sender-less transactions endowing each of the initial participants with some number of digital coins (n

is the number of initial participants). All subsequent transactions are voluntarily created by participants whenever they decide to exchange digital coins.

The cryptocurrency protocol allows participants to maintain the list of transactions without a central authority, thereby keeping the system decentralized. The list of transactions is represented as a list (or chain) of blocks $B_0, B_1, B_2, \ldots$, called a *blockchain*, where each block contains some finite list of transactions and a hash of the previous block. Block B_0, called the *genesis block*, contains the initialization transactions $X_1, X_2, \ldots, X_n$. Each of the subsequent blocks contains the voluntary transactions created on demand by the participants and is added by one of the participants, selected via the interactive protocol.

The execution of the cryptocurrency protocol proceeds in rounds. In each round, each one of the participants receives some (possibly empty) set of messages, uses them (along with its current state) to determine the longest valid chain of blocks known to it at this point, checks if it is allowed to add a new block at the current round, and, if so, extends the chain with a new block and sends it to all other participants.

Protocols based on proof-of-work and proof-of-stake differ only in how the participant that is allowed to add a new block is determined. As we explain in this paper, an important practical advantage of proof-of-stake-based cryptocurrency protocols is that they do not lead to environmentally unsustainable economic incentives that result from proof-of-work-based schemes. At the same time, however, proof-of-stake-based protocols have a number of problems and possible attack scenarios that are not applicable to proof-of-work-based protocols.

2 Framework

We now describe some of the constituent parts of the framework sketched out in the introduction in more detail and then present a more rigorous definition of the protocol. Our framework is inspired by formalizations from [Fan et al., 2021] and [Thomsen and Spitters, 2020].

2.1 Transactions

A transaction X_j is a string of bits representing a (possibly empty) list of inputs and a (possibly empty) list of outputs. That is,

$$X_j = (x_1^j \| \ldots \| x_{\mathsf{numIn}_j}^j) \| (y_1^j \| \ldots \| y_{\mathsf{numOut}_j}^j), \tag{2.1}$$

where numIn_j or numOut_j might be zero. Each input consists of a hash of some previous transaction, a digital signature of this hash, and an index of one of the outputs of that transaction. We denote the hash function by H and the signing and verification algorithms of the digital signature scheme by S and V. The k-th input of transaction X_j is then defined as

$$x_k^j = H(X_{j_k}) \| S_{\mathrm{SK}_i}(H(X_{j_k})) \| \mathsf{idxOut}_k, \tag{2.2}$$

where SK_i is the private key of the sender (i.e., the source of a transaction). Each output consists of a public key of the recipient and a positive integer indicating the number of digital coins transferred to the recipient. That is, the k-th output of transaction X_j is defined as

$$y_k^j = \text{PK}_i \| v_k, \tag{2.3}$$

where PK_i is the public key of the recipient.

In order for the transaction X_j to be valid, the sum of the values from the previous transaction outputs referenced by $x_1^j, \ldots, x_{\mathsf{numIn}_j}^j$ has to be greater than or equal to the sum of the values from the outputs $y_1^j, \ldots, y_{\mathsf{numOut}_j}^j$. The difference between these two sums, called the *transaction fee*, specifies the number of digital coins that the creator of the block containing transaction X_j is rewarded with for including it in the block.

The following is a concrete instantiation of this notation for one example transaction.

Example 2.1. Suppose that the first non-sender-less transaction in the protocol is a transaction of 5 digital coins from participant P_2 to participant P_4, and that the initial balance of participant P_2 is 12 digital coins (i.e., X_2 is a sender-less transaction whose first and only output indicates 12 digital coins being sent to P_2). Then,

$$X_{n+1} = \left(\left(x_1^{n+1} \right), \left(y_1^{n+1}, y_2^{n+1} \right) \right), \tag{2.4}$$

where

$$x_1^{n+1} = H(X_2) \| S_{\text{SK}_2}(H(X_2)) \| 1, \tag{2.5}$$
$$y_1^{n+1} = \text{PK}_4 \| 5, \tag{2.6}$$
$$y_2^{n+1} = \text{PK}_2 \| 6. \tag{2.7}$$

That is, P_2 transfers 5 of its 12 digital coins to P_4, transfers 6 of the remaining digital coins back to itself, and leaves 1 coin as a fee that will be claimed by the participant who extends the blockchain with a block containing this transaction.

2.2 Blocks

A block B_ℓ is a string of bits consisting of a hash of the previous block, a concatenation of a finite list of transactions, and a special protocol-specific string. That is,

$$B_\ell = H(B_{\ell-1}) \| (X_{\ell_1} \| \ldots \| X_{\ell_{\mathsf{numTxs}}}) \| \mathsf{specialStr}, \tag{2.8}$$

for any $\ell \in \{1, 2, \ldots\}$, where $\ell_1, \ldots, \ell_{\mathsf{numTxs}}$ are the indices of the transactions included in block B_ℓ, and

$$B_0 = X_1 \| \ldots \| X_n. \tag{2.9}$$

The last transaction in each non-genesis block is a sender-less transaction rewarding the participant who created the block with some number of digital coins. In order for the block to be valid, that number has to be less than or equal to the protocol-defined block creation reward plus the sum of the fees from all other transactions included in the block. Note that

a block creator is free to choose which transactions to include in a block. This means that, under free-market conditions, transaction fees will be determined by the market equilibrium resulting from the demand of transaction makers and the supply of block creators.

A participant is allowed to add a block to a blockchain (and claim the block creation reward) if the hash of the created block is smaller than some threshold value. In a proof-of-work-based protocol, the threshold value is fixed and participants hash the block with different random values of $\mathsf{specialStr}$ as fast as possible until the first "winning value" is found. The more hash function evaluations a given participant can perform per round, the larger its probability of winning. In a proof-of-stake-based protocol, each participant has a different threshold value that is proportional to the number of coins owned by that participant, but participants can only evaluate a single block hash per round since the $\mathsf{specialStr}$ value is fixed in any given round. To use a lottery analogy, in proof-of-work each participant buys a certain number of lottery tickets with equal winning probability in each round where the number of tickets depends on the participant's hashing rate (i.e., computational power), whereas in proof-of-stake each participant buys a single lottery ticket in each round but the winning probability of the ticket differs depending on the participant's digital coin balance (i.e., stake). We describe this formally in the following two paragraphs.

In the proof-of-work consensus mechanism, $\mathsf{specialStr}$ is a random number (a nonce) chosen by the participant. Let $\mathsf{hashRate}(P_i)$ denote the number of evaluations of $H(\cdot)$ that participant P_i is able to compute within a duration of a single round. Let $\mathsf{lotteryPoW}(P_i, t, C)$ be a function that involves choosing $\mathsf{hashRate}(P_i)$ different nonce values $r_1^t, r_2^t, \ldots, r_{\mathsf{hashRate}(P_i)}^t$ and checking whether, for at least one r_j^t, setting $B_\ell = H(B_{\ell-1})\|(X_{\ell_1}\| \ldots \|X_{\ell_{\mathsf{numTxs}}})\|r_j^t$ would yield $H(B_\ell) < T$, where $C = B_0\|B_1\| \ldots \|B_{\ell-1}$. If such an r_j^t exists, $\mathsf{lotteryPoW}(P_i, t, C)$ returns B_ℓ instantiated with $\mathsf{specialStr} = r_j^t$. Otherwise, $\mathsf{lotteryPoW}(P_i, t, C)$ returns $\emptyset$. In the proof-of-work-based protocol, participant P_i is allowed to extend the blockchain C in round t if and only if $\mathsf{lotteryPoW}(P_i, t, C)$ returns a block.

In the proof-of-stake consensus mechanism, the value $\mathsf{specialStr}$ for participant P_i at round t is fixed and given by $\mathsf{specialStr} = t\|\mathrm{PK}_i\|S_{\mathrm{SK}_i}(H(B_{\ell-1})\|t)$. Let $\mathsf{stake}(P_i, C)$ denote the number of digital coins owned by participant P_i after accounting for all transactions recorded in a blockchain $C = B_0\|B_1\| \ldots \|B_{\ell-1}$. Let $\mathsf{lotteryPoS}(P_i, t, C)$ be a function that involves computing $B_\ell = H(B_{\ell-1})\|(X_{\ell_1}\| \ldots \|X_{\ell_{\mathsf{numTxs}}})\|t\|\mathrm{PK}_i\|S_{\mathrm{SK}_i}(H(B_{\ell-1})\|t)$ and checking whether $H(B_\ell) < \mathsf{stake}(P_i, C)T$, where $C = B_0\|B_1\| \ldots, B_{\ell-1}$. If so, $\mathsf{lotteryPoS}(P_i, t, C)$ returns B_ℓ. Otherwise, $\mathsf{lotteryPoS}(P_i, t, C)$ returns $\emptyset$. In the proof-of-stake-based protocol, participant P_i is allowed to extend the blockchain C in round t if and only if $\mathsf{lotteryPoS}(P_i, t, C)$ returns a block.

2.3 Blockchain validation

In this section, we specify the conditions under which a blockchain is considered valid. Informally, a blockchain is valid if the genesis block is valid and each new block references the previous one, solves the consensus puzzle (i.e., hashes to a number smaller than the relevant threshold), and contains only valid transactions. We now specify these conditions separately and in more detail.

Let $\mathsf{validGenesis}(B)$ be a binary function that returns 1 if and only if B is a valid genesis block. Specifically, $\mathsf{validGenesis}(B)$ returns 1 if and only if B can be parsed as a concatenation of sender-less transactions.

Let $\mathsf{solvesPuzzlePoW}(B, C)$ and $\mathsf{solvesPuzzlePoS}(B, C)$ be binary functions that return 1 if and only if B solves the block creation puzzle in proof-of-work and proof-of-stake, respectively. Specifically, $\mathsf{solvesPuzzlePoW}(B, C)$ returns 1 if and only if $H(B) < T$, while $\mathsf{solvesPuzzlePoS}(B, C)$ parses B as $h\|(X_{\ell_1}\| \ldots \|X_{\ell_{\mathsf{numTxs}}})\|t\|\text{PK}\|\sigma$ and returns 1 if and only if $H(B) < \mathsf{stake}(P_i, C)T$, where P_i is the participant with public key PK, and $V_{\text{PK}}(h\|t, \sigma) = 1$ (i.e., the owner of the public key PK is the unique participant that could have signed the concatenation of the previous block hash and the round index).

Let $\mathsf{validTransaction}(X, C)$ be a binary function that returns 1 if and only if X is a valid transaction with respect to the blockchain C. The function first verifies that the sender owns all of the previous transaction outputs referenced in the inputs of X. That is, after parsing X as $(x_1\| \ldots \|x_{\mathsf{numIn}})\|(y_1\| \ldots \|y_{\mathsf{numOut}})$, the function checks whether, for each $k \in \{1, 2, \ldots, \mathsf{numIn}\}$ and $x_k = h\|\sigma\|\mathsf{idxOut}$, h is a hash of some transaction X' that is included in one of the blocks in C, idxOut is an integer between 1 and the number of outputs of X', the idxOut-th output of X' is not referenced by the inputs of any other transaction included in any of the blocks in C, and whether, parsing the idxOut-th output of X' as $\text{PK}\|v$, we have that $V_{\text{PK}}(h, \sigma) = 1$ (i.e., the sender proves that it knows the private key corresponding to the public key to which the referenced output is addressed). The function then verifies that, for each $k \in \{1, 2, \ldots, \mathsf{numOut}\}$, y_k parses as a concatenation of some public key and some integer output value. The return value is 1 if and only if all of these checks are passed.

Let $\mathsf{validReward}(B)$ be a binary function that returns 1 if and only if the reward claimed by the block creator in the last transaction in block B does not exceed the valid amount. Specifically, $\mathsf{validReward}(B)$ returns 1 if and only if the last transaction included in B is a sender-less transaction that transfers some number y of digital coins to one of the participants such that y is less than or equal to the sum of the protocol-defined block-creation reward and the transaction fees from all of the transactions in B.

Let $\mathsf{validBlockchainPoW}(C)$ and $\mathsf{validBlockchainPoS}(B, C)$ be binary functions that return 1 if and only if C is a valid blockchain in proof-of-work and proof-of-stake, respectively. Specifically, the functions parse C as $B_0\|B_1\| \ldots \|B_\ell$ and return 1 if and only if all of the following conditions are satisfied:

1. $\mathsf{validGenesis}(B_0) = 1$,
2. for every $j \in \{1, 2, \ldots, \ell\}$ and block $B_j = h_j\|(X_{j_1}\| \ldots \|X_{j_{\mathsf{numTxs}}})\|\mathsf{specialStr}_j$:
   (a) $h_j = H(B_{j-1})$,
   (b) $\mathsf{solvesPuzzle}(B_j, B_0\| \ldots \|B_{j-1}) = 1$,
   (c) for every $k \in \{1, 2, \ldots, \mathsf{numTxs} - 1\}$, $\mathsf{validTransaction}(X_{j_k}, B_0\| \ldots \|B_{j-1}) = 1$,
   (d) $\mathsf{validReward}(B_j) = 1$,

where $\mathsf{solvesPuzzle}$ is either $\mathsf{solvesPuzzlePoW}$ or $\mathsf{solvesPuzzlePoS}$, correspondingly.

2.4 Protocol

We are now in a position to present the formal definition of a cryptocurrency protocol.

Definition 2.2 (Protocol $\Pi(\kappa, T, E, \mathsf{lotteryFn}, \mathsf{validBlockchainFn})$)**.** For any $\kappa \in \{0, 1, 2, \ldots\}$, $T \in \{1, 2, \ldots, 2^\kappa\}$, tuple $E = (e_1, \ldots, e_n)$ with $e_j \in \{0, 1, 2, \ldots\}$ for each $j \in \{1, 2, \ldots, n\}$, function $\mathsf{lotteryFn}$ mapping a participant, a round index, and a blockchain either to a block or to $\emptyset$, and function $\mathsf{validBlockchainFn}$ mapping a blockchain either to 0 or to 1, $\Pi(\kappa, T, E, \mathsf{lotteryFn}, \mathsf{validBlockchainFn})$ is an interactive protocol in which each party P_i proceeds as follows: constructs the genesis block B_0 containing n sender-less transactions $X_1, \ldots, X_n$ where X_j assigns ownership of e_j digital coins to P_j, initializes C_{curr} to B_0, and, for each round $t \in \{1, 2, 3, \ldots\}$,

1. receives a (possibly empty) list of blockchains $C_1, \ldots, C_k$ broadcasted by other parties in the end of the previous round;
2. evaluates $\mathsf{validBlockchainFn}(C_j)$ for every $j \in \{1, 2, \ldots, k\}$ and updates C_{curr} to be the longest blockchain from $\{C_{\mathsf{curr}}\} \cup \{C_j : j \in \{1, 2, \ldots, k\}$ and $\mathsf{validBlockchainFn}(C_j) = 1\}$ (where the length of the blockchain is the number of blocks it contains);
3. evaluates $\mathsf{lotteryFn}(P_i, t, C_{\mathsf{curr}})$ and if it returns a block B, updates C_{curr} to $C_{\mathsf{curr}} \| B$ and broadcasts the updated value of C_{curr} to all other parties.

Letting $\mathsf{lotteryFn} = \mathsf{lotteryPoW}$ and $\mathsf{validBlockchainFn} = \mathsf{validBlockchainPoW}$ in the above definition yields a cryptocurrency protocol with proof-of-work consensus mechanism, whereas letting $\mathsf{lotteryFn} = \mathsf{lotteryPoS}$ and $\mathsf{validBlockchainFn} = \mathsf{validBlockchainPoS}$ in the above definition yields a cryptocurrency protocol with proof-of-state consensus mechanism. We have thus achieved our goal of obtaining a common framework in which both consensus mechanisms can be formalized. In the rest of the paper, we will write Π_{PoW} as a shorthand for a proof-of-work-based protocol $\Pi(\kappa, T, E, \mathsf{lotteryPoW}, \mathsf{validBlockchainPoW})$ and Π_{PoS} as a shorthand for a proof-of-stake-based protocol $\Pi(\kappa, T, E, \mathsf{lotteryPoS}, \mathsf{validBlockchainPoS})$.

3 Proof-of-Work versus Proof-of-Stake

We are now ready to derive the result revealing the main difference between proof-of-work and proof-of-stake – namely, the block creation probabilities in Π_{PoW} and Π_{PoS}. The following two theorems establish the result.

Theorem 3.1 (Proof-of-work lottery)**.** *In Π_{PoW}, if the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle and all participants start working on some blockchain C at the same time (i.e., set C_{curr} to C in the same round), then the probability that P_i will be allowed to extend C is $\mathsf{hashRate}(P_i) / (\mathsf{hashRate}(P_1) + \cdots + \mathsf{hashRate}(P_n))$.*

Proof. Suppose that the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle and that all participants start working on some blockchain C at the same time. Let Y_i be a random variable representing the number of rounds it would take participant P_i to find a block extending C if P_i continued working on C until this happens. Since, under the random oracle model, evaluations of the hash function H are independent, Y_i represents

the time until the first success in independent Bernoulli trials with probability of success given by $\Pr[\mathsf{lotteryPoW}(P_i, t, C) \neq \emptyset]$. By Lemma A.4, assuming T is small enough the probability of success is approximately $\mathsf{hashRate}(P_i)T/2^\kappa$, and so, by the definition of the Geometric distribution, the distribution of Y_i is approximately $\mathrm{Geom}(\mathsf{hashRate}(P_i)T/2^\kappa)$. Then, by Lemma A.1, assuming T is small enough the distribution of Y_i is approximately $\mathrm{Expo}(\mathsf{hashRate}(P_i)T/2^\kappa)$. Finally, since P_i will be allowed to extend C if and only if P_i is the first participant to find a valid block extending C, which is equivalent to the event that $Y_i = \min\{Y_1, \ldots, Y_n\}$, we get that P_i will be allowed to extend C with probability

$$\Pr[Y_i = \min\{Y_1, \ldots, Y_n\}] = \frac{\mathsf{hashRate}(P_i)T/2^\kappa}{\mathsf{hashRate}(P_1)T/2^\kappa + \cdots + \mathsf{hashRate}(P_n)T/2^\kappa} \tag{3.1}$$

$$= \frac{\mathsf{hashRate}(P_i)}{\mathsf{hashRate}(P_1) + \cdots + \mathsf{hashRate}(P_n)}, \tag{3.2}$$

where (3.1) follows from Lemma A.2. □

Theorem 3.2 (Proof-of-stake lottery)**.** *In Π_{PoS}, if the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle and all participants start working on some blockchain C at the same time (i.e., set C_{curr} to C in the same round), then the probability that P_i will be allowed to extend C is $\mathsf{stake}(P_i, C)/\left(\mathsf{stake}(P_1, C) + \cdots + \mathsf{stake}(P_n, C)\right)$.*

Proof. Suppose that the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle and that all participants start working on some blockchain C at the same time. Let Y_i be a random variable representing the number of rounds it would take participant P_i to find a block extending C if P_i continued working on C until this happens. Since, under the random oracle model, evaluations of the hash function H are independent, Y_i represents the time until the first success in independent Bernoulli trials with probability of success given by

$$\Pr[\mathsf{lotteryPoS}(P_i, t, C) \neq \emptyset] = \Pr[H(B_\ell) < \mathsf{stake}(P_i, C)T] = \mathsf{stake}(P_i, C)T/2^\kappa, \tag{3.3}$$

where the last equality follows from Lemma A.3. Thus, by the definition of the Geometric distribution, the distribution of Y_i is $\mathrm{Geom}(\mathsf{stake}(P_i, C)T/2^\kappa)$. Then, by Lemma A.1, assuming T is small enough the distribution of Y_i is approximately $\mathrm{Expo}(\mathsf{stake}(P_i, C)T/2^\kappa)$. Finally, since P_i will be allowed to extend C if and only if P_i is the first participant to find a valid block extending C, which is equivalent to the event that $Y_i = \min\{Y_1, \ldots, Y_n\}$, we get that P_i will be allowed to extend the C with probability

$$\Pr[Y_i = \min\{Y_1, \ldots, Y_n\}] = \frac{\mathsf{stake}(P_i, C)T/2^\kappa}{\mathsf{stake}(P_1, C)T/2^\kappa + \cdots + \mathsf{stake}(P_n, C)T/2^\kappa} \tag{3.4}$$

$$= \frac{\mathsf{stake}(P_i, C)}{\mathsf{stake}(P_1, C) + \cdots + \mathsf{stake}(P_n, C)}, \tag{3.5}$$

where (3.4) follows from Lemma A.2. □

Theorems 3.1 and 3.2 reveal the core difference between proof-of-work and proof-of-stake: in proof-of-work-based consensus, the voting power (i.e., the ability to add the next block

to a blockchain) of a participant is tied to the fraction of computational resources owned by that participant (i.e., "one-CPU-one-vote" [Nakamoto, 2008]), whereas in proof-of-stake-based consensus, the voting power of a participant is tied to the fraction of the digital currency owned by that participant (i.e., "one-coin-one-vote"). This has important practical implications in the real world.

First, note that it is necessary to tie the voting power to some economically scarce resource in order for the cryptocurrency system to be functional. If this was not the case, a malicious participant could gain unlimited voting power and break the protocol. For instance, if any owner of a public-private key pair would be given equal chance of adding the next block (i.e., "one-identity-one-vote"), then some adversary could create an arbitrarily large number of (fake) identities and shift all of the voting power in the system over to itself. As long as the resource giving participants voting power is scarce, competition for it will take place and such artificial manipulations of the power balance will be impossible.

The economically scarce resource to which voting power is tied in proof-of-work-based protocols is computing power. This includes both access to the hardware for performing the computations (i.e., evaluating hash functions) as well as the capacity to keep this hardware in continuous operation. As such, the proof-of-work consensus mechanism induces competition for resources in the real-world – namely, the materials, construction equipment, and human labor involved in the production of specialized hardware and, most importantly, the electricity consumed while using this hardware to run the computations. The resulting hardware arms race and the incentive for power consumption are the main properties that make proof-of-work-based protocols practically unattractive. For instance, it has been estimated that, in 2019, a single Bitcoin transaction consumed about as much electricity as 21 average U.S. households in one day [Xiao et al., 2020]. As a result, proof-of-work is now considered by many to be environmentally unsustainable.

The economically scarce resource to which voting power is tied in proof-of-stake-based protocols is digital coins. Since this is a virtual resource, competition for it does not directly induce any real-world costs, which makes proof-of-stake-based cryptocurrency protocols much more sustainable than proof-of-work-based ones. Checking if a participant is allowed to extend the blockchain at any given round involves just a single evaluation of a hash function when proof-of-stake is used, which means that no special hardware is required in order to participate in block creation. Mainly due to its much more environmentally-friendly incentive structure, the proof-of-stake consensus mechanism has gained a lot of popularity in recent years. In 2020, it was announced that proof-of-stake is going to be adopted by Ethereum [ETH, 2020], a cryptocurrency with the second-largest market capitalization.

It is worth noting that in both consensus mechanisms it is possible for participants to collude. In a proof-of-work-based protocol, a subset of participants may organize to ensure that they are only hashing distinct block candidates as a group in each round. The hashing rate of such a coalition is then the sum of the hashing rates of its members, and so, by Theorem 3.1, the group's voting power (i.e., the probability that one of its members will extend the current blockchain) is the sum of the voting powers of the colluding individuals. In a proof-of-stake-based protocol, an analogous outcome can be achieved if a subset of participants agree to transfer their digital coins to some "joint account" (i.e., a new participant with a

public-private key pair known to everyone in the coalition). The coin balance in this new account is then equal to the sum of the coin balances of the colluding individuals, and so, by Theorem 3.2, the group's voting power is the sum of the voting powers of its members.

4 Security

We now turn to the security properties of proof-of-stake, starting with two important features that follow directly from the security of the digital signature scheme.

Theorem 4.1. *If the digital signature scheme (S, V) used in Π_{PoS} is CMA secure, then, for every polynomial-time participant P_{i^*}, the probability that P_{i^*} can send digital coins owned by some other participant P_j is negligible.*

Proof. We prove the contrapositive. Suppose that there exists a polynomial-time participant P_{i^*} that can send digital coins owned by some other participant P_j with non-negligible probability. Since digital coins are sent only if the transaction sending them is included in one of the blocks in a valid blockchain and, by the definition of $\mathsf{validBlockchain}$, a blockchain is valid only if all of the transactions included in all of the blocks comprising it are valid, this means that, with non-negligible probability, P_{i^*} is able to create a transaction X^* such that, for some C with $\mathsf{validBlockchain}(C) = 1$, $\mathsf{validTransaction}(X^*, C) = 1$ but the k^*-th input of X^* references a previous transaction output that was addressed to some other participant P_j. Let A be an adversary that simulates P_{i^*}, parses the k^*-th input of the transaction X^* created by P_{i^*} as $h\|\sigma\|\mathsf{idxOut}$, and outputs (h, σ). Clearly, given that P_{i^*} is efficient, A must also be efficient. Also, by the definition of $\mathsf{validTransaction}$, if X^* is valid with respect to some blockchain C, then $V_{\text{PK}_j}(h, \sigma) = 1$. It follows that A is an efficient adversary that can forge the digital signature corresponding to the private key SK_j with non-negligible probability. This implies that the digital signature scheme (S, V) is not CMA secure. □

Theorem 4.2. *If the digital signature scheme (S, V) used in Π_{PoS} is CMA secure, then, for every polynomial-time participant P_{i^*}, every valid blockchain C, and every round t, the probability that P_{i^*} can extend C at round t if it is not allowed to is negligible.*

Proof. We prove the contrapositive. Suppose that there exists a polynomial-time participant P_{i^*}, a valid blockchain C, and a round t, such that, with some non-negligible probability, P_{i^*} can come up a block B^* such that $\mathsf{solvesPuzzle}(B^*, C) = 1$ even though

$$H(H(B_{\ell-1})\|(X_{\ell_1}\|\ldots\|X_{\ell_{\mathsf{numTxs}}})\|t\|\text{PK}_{i^*}\|S_{\text{SK}_{i^*}}(H(B_{\ell-1})\|t)) \geq \mathsf{stake}(P_{i^*}, C)T, \tag{4.1}$$

where $C = B_0\|B_1\|\ldots\|B_\ell$. Let A be an adversary that simulates P_{i^*}, parses the block B^* created by P_{i^*} as $h\|(X_{\ell_1}\|\ldots\|X_{\ell_{\mathsf{numTxs}}})\|t\|\text{PK}\|\sigma$, and outputs $(h\|t, \sigma)$. Clearly, given that P_{i^*} is efficient, A must also be efficient. Also, by the definition of $\mathsf{solvesPuzzle}$, if B^* is such that $\mathsf{solvesPuzzle}(B^*, C) = 1$, then $V_{\text{PK}}(h, \sigma) = 1$. It follows that A is an efficient adversary that can forge the digital signature corresponding to the private key SK owned by the participant whose public key is PK with some non-negligible probability. This implies that the digital signature scheme (S, V) is not CMA secure. □

We now state a fundamental result about the security of proof-of-stake-based cryptocurrency systems established by Fan et al. [2021]. The following are the three security properties that are shown to be achievable.

Definition 4.3 (Chain growth)**.** The *chain growth* property with parameter $g \in \mathbb{R}$ holds in protocol Π_{PoS} if, for any honest participant P_{i_1} with $C_{\mathsf{curr}} = C_1$ at round t_1, and honest participant P_{i_2} with $C_{\mathsf{curr}} = C_2$ at round t_2, with $t_1 < t_2$, the difference between the number of blocks in C_2 and the number of blocks in C_1 is at least $g(t_2 - t_1)$.

Definition 4.4 (Common prefix)**.** The *common prefix* property with parameter $k \in \mathbb{N}$ holds in protocol Π_{PoS} if, for any honest participant P_{i_1} with $C_{\mathsf{curr}} = C_1$ at round t_1, and honest participant P_{i_2} with $C_{\mathsf{curr}} = C_2$ at round t_2, with $t_1 < t_2$, C_2 starts with $C_1[\neg k]$, where $C_1[\neg k]$ denotes the blockchain obtained from C_1 by removing the last k blocks.

Definition 4.5 (Chain quality)**.** The *chain quality* property with parameter $\mu \in \mathbb{R}$ and $\ell \in \mathbb{N}$ holds in protocol Π_{PoS} if, for any honest participant P_i with $C_{\mathsf{curr}} = C$ at round t and large enough ℓ consecutive blocks in C, the ratio of the honest blocks is at least μ.

The result proved in Fan et al. [2021] is then given by the following theorem.

Theorem 4.6. *If the hash function H used in protocol Π_{PoS} is modeled as a random oracle, the digital signature scheme (S, V) used in the protocol is CMA secure, and the honest participants hold at least 73% of the stake (i.e., own at least 73% of the digital coins), then, for any strategy used by the malicious participants, Π_{PoS} can achieve chain growth, common prefix, and chain quality properties.*

5 Open Problems

In this section, we sketch out a few of the main vulnerabilities of proof-of-stake-based cryptocurrencies and explain why none of them apply proof-of-work-based protocols [Nguyen et al., 2019, BitFury Group, 2015]. This shows that, despite its sustainability advantages, proof-of-stake can not yet be considered as definitively superior to proof-of-work.

5.1 Nothing-at-Stake

One of the main shortcomings of the proof-of-stake consensus mechanism, known as the *nothing-at-stake problem*, stems from the same property that gives it its core practical advantage – namely, the fact that the resource used to determine the voting power of participants (i.e., digital coins) is a virtual one. Since the work that participants have to perform in order to compete for block creation has negligible real-world costs, it is irrational to only attempt to extend a single blockchain in any given round if there is non-zero probability that some other blockchain is going to surpass the current one. That is, in any given round t, an economically rational participant P_i would check if it is allowed to extend *any* of the blockchains that it knows about at that point instead of only the longest one, thereby violating the protocol. This is because the cost of attempting to extend a blockchain in a proof-of-work-based protocol is the cost of computing a single hash, which we assume is negligible (hence the name *nothing-at-stake*), while the expected benefit of doing so is the

block creation reward times the probability that the corresponding blockchain is going to outgrow all of the other ones at some point in the future, which is non-negligible for at least some (and possibly all) of the blockchains known at round t. We thus get that the protocol specification is at odds with the rational incentives of participants, which makes it unsafe to assume that the majority of the stakeholders are going to behave honestly.

The proof-of-work consensus mechanism does not lead to this problem because the opportunity cost of attempting to extend any given blockchain in a round is non-zero. That is, if a participant distributes its hashing power between multiple blockchains in a single round, then the probability that it is going to be allowed to extend any of them (and, in turn, the participant's expected reward) becomes smaller. The rational strategy is therefore to pick the blockchain that is most likely to be the longest one in the future and use all of one's hashing power to try to extend it. Given that all other participants are governed by the same incentives, the rational thing to do at any given round is to work on the longest currently known blockchain.

5.2 Bribing Attack

The nothing-at-stake problem in proof-of-stake-based cryptocurrency protocols creates a possibility for the following attack scenario, known as the *bribing attack*. Suppose that an attacker makes a cryptocurrency transaction to purchase a good in the real-world, consumes it, and broadcasts to all other participants an alternative (initially shorter) blockchain that does not contain the attacker's original transaction but contains new transactions transferring digital coins from the attacker to all participants that have not made transactions since the attacker's original one. Since this strictly increases the expected benefit of attempting to extend this alternative blockchain for the recipients of those new transactions (assuming they have not made transactions recently, they will not lose any digital coins if the alternative chain becomes the main one), the attacker is able to effectively bribe some number of participants and thereby consolidate block creation effort on the malicious blockchain. Since the block creators participating in this scheme do not lose anything if the attack fails (trying to extend an additional blockchain in each round incurs negligible cost in proof-of-stake), they are going to accept the bribes. If the attack succeeds the attacker, will regain ownership of the digital coins originally spent on the real-world good while losing the (attacker-chosen) amount paid to other participants as bribes. As long as the sum of the bribes is smaller than the value of the attacker's original purchase, the attack will be profitable.

The bribing attack is not viable in a proof-of-work-based cryptocurrency protocol for the same reason that the nothing-at-stake problem does not arise in proof-of-work. Since working on the attacker's blockchain would reduce the participant's chances of extending the main blockchain in each round, participants will be inclined to refuse to work on the malicious blockchain. The bribe amounts that the attacker would have to pay to the other participant to make it rational for them to participate in the scheme will be prohibitively high if the proof-of-work consensus mechanism is deployed. For this reason, the attack is only feasible in proof-of-stake.

5.3 Friction of Circulation

The fact that, in a proof-of-stake-based protocol, the participant's block creation probability at any given round is directly proportional to their stake at that round implies that participants are incentivized to refrain from spending digital coins. This can be especially problematic when a proof-of-stake-based cryptocurrency system is in its early stages. The initial owners of the digital coins will be at an advantage compared to the participants that join later (i.e., whose initial endowments are zero) because, as the cryptocurrency gains popularity, the real-world currency equivalent of each participant's endowment is going to increase. That is, by just saving the digital coins, a participant will be able to retain its level of voting power without investing any real-world resources while newcomers will be forced to spend more and more to acquire the same number of coins (and thus the same level of voting power) as time goes on. Because of this, the digital coins may tend to get stuck in the hands of early participants, which would hamper the overall usability of the system.

Since proof-of-work-based protocols do not associate voting power to participants' wealth in the system, the problem does not arise. No matter how early participants invest real-world resources to acquire some level of voting power in the system, they will need to continually invest more resources to keep up with the competition in order to retain that level of voting power. The equivalent of the proof-of-stake case would be if hardware purchased at one point in time would automatically improve without any cost of maintenance. Since this is clearly not the case, the problem is specific to proof-of-stake.

6 Conclusion

In this paper, we gave a formal, self-contained introduction to the proof-of-stake consensus mechanism. We presented a framework revealing the core difference between proof-of-work and proof-of-stake (captured by the respective block-creation probabilities), outlined the practical implication of this difference, discussed the security of proof-of-stake, and surveyed some of the open problems specific to proof-of-stake-based protocols. While proof-of-stake does have a major advantage over proof-of-work (namely, in being more environmentally sustainable), we conclude that it is not yet clear whether the proof-of-stake consensus mechanism is a definitively superior alternative to proof-of-work.

References

Ethereum 2.0 specifications. `https://github.com/ethereum/consensus-specs`, 2020.

BitFury Group. Proof of stake versus proof of work white paper. 09 2015. URL `https://bitfury.com/content/downloads/pos-vs-pow-1.0.2.pdf`.

Lei Fan, Jonathan Katz, Phuc Thai, and Hong-Sheng Zhou. A permissionless proof-of-stake blockchain with best-possible unpredictability. Cryptology ePrint Archive, Report 2021/660, 2021. `https://ia.cr/2021/660`.

Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. URL http://www.bitcoin.org/bitcoin.pdf.

Cong Nguyen, Hoang Dinh Thai, Diep Nguyen, Dusit Niyato, Huynh Nguyen, and Eryk Dutkiewicz. Proof-of-stake consensus mechanisms for future blockchain networks: Fundamentals, applications and opportunities. *IEEE Access*, PP:1–1, 06 2019. doi: 10.1109/ACCESS.2019.2925010.

Søren Eller Thomsen and Bas Spitters. Formalizing nakamoto-style proof of stake. Cryptology ePrint Archive, Report 2020/917, 2020. https://ia.cr/2020/917.

Florian Tschorsch and Björn Scheuermann. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys Tutorials*, 18(3):2084–2123, 2016. doi: 10.1109/COMST.2016.2535718.

Yang Xiao, Ning Zhang, Wenjing Lou, and Y. Thomas Hou. A survey of distributed consensus protocols for blockchain networks. *IEEE Communications Surveys & Tutorials*, 22: 1432–1465, 2020.

A Appendix

The following are several useful lemmas referenced in Section 3.

Lemma A.1. *If $X \sim \text{Geom}(p)$, then the distribution of X approaches $\text{Expo}(p)$ as $p \to 0^+$.*

Proof. First, note that

$$\lim_{p\to 0^+} \ln\left((1-p)^{x/p}\right) = \lim_{p\to 0^+} \frac{x\ln(1-p)}{p} \tag{A.1}$$

$$= x\left(\lim_{p\to 0^+} \frac{\ln(1-p)}{p}\right) \tag{A.2}$$

$$= x\left(\lim_{p\to 0^+} \frac{\frac{d}{dp}\left(\ln(1-p)\right)}{\frac{d}{dp}(p)}\right) \tag{A.3}$$

$$= x\left(\lim_{p\to 0^+} \frac{\frac{-1}{1-p}}{1}\right) \tag{A.4}$$

$$= -x, \tag{A.5}$$

where (A.3) follows from L'Hôpital's rule, which implies that

$$(1-p)^{x/p} = e^{\ln\left((1-p)^{x/p}\right)} \to e^{-x} \quad \text{as } p \to 0^+. \tag{A.6}$$

We then get that

$$\Pr[pX \le x] = \Pr[X \le x/p] \tag{A.7}$$

$$= 1 - (1-p)^{\lfloor x/p \rfloor} \tag{A.8}$$

$$= 1 - (1-p)^{\lfloor x/p \rfloor - x/p}(1-p)^{x/p} \tag{A.9}$$

$$\to 1 - e^{-x} \quad \text{as } p \to 0^+, \tag{A.10}$$

where (A.8) follows from the definition of the CDF of a $\text{Geom}(p)$ random variable and (A.10) follows from (A.6). This shows that the CDF of a random variable pX approaches the CDF of the $\text{Expo}(1)$ random variable as $p \to 0^+$. It follows that the distribution of X approaches $\text{Expo}(p)$ as $p \to 0^+$. □

Lemma A.2. *If $X_1, X_2, \ldots, X_n$ are independent random variables with $X_j \sim \text{Expo}(\lambda_j)$ for each $j \in \{1, 2, \ldots, n\}$, then $\Pr[X_j = \min\{X_1, \ldots, X_n\}] = \lambda_j/(\lambda_1 + \cdots + \lambda_n)$.*

Proof. Letting $X' \sim \text{Expo}(\lambda_1 + \cdots + \lambda_n)$, we have that

$$\Pr[X_j = \min\{X_1, \ldots, X_n\}] = \int_0^\infty \Pr[X_j = x \text{ and } X_k > x \text{ for all } k \neq j]dx \tag{A.11}$$

$$= \int_0^\infty \Pr[X_j = x] \left(\prod_{k \neq j} \Pr[X_k > x] \right) dx \tag{A.12}$$

$$= \int_0^\infty \lambda_j e^{-\lambda_j x} \left(\prod_{k \neq j} e^{-\lambda_k x} \right) dx \tag{A.13}$$

$$= \int_0^\infty \lambda_j \left(\prod_{k=1}^{n} e^{-\lambda_k x} \right) dx \tag{A.14}$$

$$= \lambda_j \int_0^\infty e^{-(\lambda_1 + \cdots + \lambda_n)x} dx \tag{A.15}$$

$$= \lambda_j \int_0^\infty \Pr[X' > x]dx \tag{A.16}$$

$$= \lambda_j E[X'] \tag{A.17}$$

$$= \frac{\lambda_j}{\lambda_1 + \cdots + \lambda_n}, \tag{A.18}$$

where (A.12) follows from the fact that X_j are independent, (A.13) follows from the definitions of the PDF and the CDF of an $\text{Expo}(\lambda)$ random variable, (A.16) follows from the definition of X' (and the definition of $\text{Expo}(\lambda)$ CDF), (A.17) follows from the Darth Vader rule, and (A.18) follows from the fact that the expected value of an $\text{Expo}(\lambda)$ random variable is $1/\lambda$. □

Lemma A.3. *If the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle and $T \in \{1, 2, \ldots, 2^\kappa\}$, then, for any $x \in \{0,1\}^*$, $\Pr[H(x) < T] = T/2^\kappa$.*

Proof. Suppose that the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle. Let T be any number from $\{1, 2, \ldots, 2^\kappa\}$ and let x be any string of bits. In the random oracle model, evaluating $H(x)$ is equivalent to sampling a uniformly random number from $\{0, 1, \ldots, 2^\kappa - 1\}$. Since there are exactly T numbers in $\{0, 1, \ldots, 2^\kappa - 1\}$ that are smaller than T, we get that $\Pr[H(x) < T] = T/2^\kappa$. □

Lemma A.4. *If the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle, then, for any $i \in \{1, 2, \ldots, n\}$, $t \in \{1, 2, 3, \ldots\}$, any blockchain C, and small enough T, $\Pr[\mathit{lotteryPoW}(P_i, t, C) \neq \emptyset] \approx \mathit{hashRate}(P_i)T/2^\kappa$.*

Proof. Suppose that the hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$ is modeled as a random oracle. Let i be any number from $\{1, 2, \ldots, n\}$, let t be any number from $\{1, 2, 3, \ldots\}$, and let C by any blockchain. Let $r_1^t, \ldots, r_{\mathsf{hashRate}(P_i)}^t$ be the nonce values that are used in $\mathsf{lotteryPoW}(P_i, t, C)$ to instantiate B_ℓ and, for every $j \in \{1, 2, \ldots, \mathsf{hashRate}(P_i)\}$, let x_j be

the bit string representing a block B_ℓ instantiated with nonce r_j^t. Then,

$$
\begin{aligned}
&\Pr[\mathsf{lotteryPoW}(P_i, t, C) \neq \emptyset] \\
&= 1 - \Pr[\mathsf{lotteryPoW}(P_i, t, C) = \emptyset] && \text{(A.19)} \\
&= 1 - \Pr[H(x_j) \geq T \text{ for all } j \in \{1, 2, \ldots, \mathsf{hashRate}(P_i)\}] && \text{(A.20)} \\
&= 1 - \prod_{j=1}^{\mathsf{hashRate}(P_i)} (1 - \Pr[H(x_j) < T]) && \text{(A.21)} \\
&= 1 - (1 - T/2^\kappa)^{\mathsf{hashRate}(P_i)} && \text{(A.22)} \\
&\approx \mathsf{hashRate}(P_i) T/2^\kappa, && \text{(A.23)}
\end{aligned}
$$

where (A.21) follows from the fact that H is modeled as a random oracle (hence evaluations with different inputs are independent), (A.22) follows from Lemma A.3, and (A.23) follows from the binomial approximation given that T is small enough. □