Proof of Stake versus Proof of Work

White Paper

BitFury Group

Sep 13, 2015 (Version 1.0)

Abstract

Proof of stake is a consensus mechanism for digital currencies that is an alternative to proof of work used in Bitcoin. The main declared advantages of proof of stake approaches are the absence of expensive computations and hence a lower entry barrier for block generation rewards. In this report, we examine the pros and cons of both consensus systems and show that existing implementations of proof of stake are vulnerable to attacks which are highly unlikely in Bitcoin and proof of work approaches in general.

Version History

Version	Date	Change description
1.0	Sep 13, 2015	Initial version

The underlying database structure for transactions of Bitcoin and other digital currencies is a decentralized ledger, called the blockchain, which stores the entire transaction history. The name stems from the fact that transactions are bundled into blocks; each block in the blockchain (except for the first i.e. genesis block) references a previous block. Each node participating in the Bitcoin network has its own copy of the blockchain, which is synchronized with other nodes using a peer-to-peer protocol[1]. Any implementation of digital currency must have a way to secure its blockchain against attacks. For example, an attacker may spend some money and then reverse the spending transaction by broadcasting his own version of the blockchain, which does not include this transaction; as security of the blockchain does not rely on a single authority, users have no prior knowledge as to which version of the ledger is valid.

In Bitcoin, the security of the network relies on a *proof of work* (PoW) algorithm in the form of block mining. Each node that wants to participate in mining is required to solve a computationally difficult problem to ensure the validity of the newly mined block; solutions are rewarded with bitcoins. The protocol is fair in the sense that a miner with p fraction of the total computational power can win the reward and create a block with the probability p. An attacker is required to solve the same tasks as the rest of the Bitcoin network; i.e., an attack on Bitcoin will only be successful if the attacker can bring to bear significant computational resources.

Operation of the Bitcoin protocol is such that security of the network is supported by physically scarce resources:

- specialized hardware needed to run computations, and
- electricity spent to power the hardware.

This makes Bitcoin inefficient from a resource standpoint. To increase their share of rewards, Bitcoin miners are compelled to participate in an arms race to continuously deploy more resources in mining. While this makes the cost of an attack on Bitcoin prohibitively high, the ecological unfriendliness of the Bitcoin protocol has resulted in proposals to build similar systems that are much less resource intensive.

One possible decentralized ledger implementation with security not based on expensive computations relies on *proof of stake* (PoS) algorithms. The idea behind proof of stake is simple: instead of mining power, the probability to create a block and receive the associated reward is proportional to a user's ownership stake in the system. An individual stakeholder who has p fraction of the total number of coins in circulation creates a new block with p probability.

The rationale behind proof of stake is the following: users with the highest stakes in the system have the most interest to maintain a secure network, as they will suffer the most if the reputation and price of the cryptocurrency would diminish because of the attacks. To mount a successful attack, an

[1] Strictly speaking, there exist Bitcoin nodes that do not store the entire blockchain, but rather rely on simplified payment verification [1]. We don't consider these nodes in the following research, as they do not contribute to the security of the network.

outside attacker would need to acquire most of the currency, which would be prohibitively expensive for a popular system.

We study whether or not proof of stake consensus is more susceptible to attacks than proof of work in the following sections.

1 Theory

A cryptocurrency system such as Bitcoin possesses state information enabling users to derive their balances. For Bitcoin, the system state is a collection of unspent transaction outputs (UTXOs), with each output cryptographically locked, requiring the user to provide a proof of ownership to spend a UTXO. Instead of explicitly storing the balance, Bitcoin and other digital currencies maintain a complete history of users' transactions, which can be used to infer the current state and all past states of the system.

Transactions represent atomic changes to the system state. Transactions are grouped in blocks; blocks form an ordered collection called the blockchain. To organize the blockchain, each block contains a reference to the previous one. The first block in the chain is the *genesis block*; it does not have a previous block and is usually hardcoded into the protocol. New blocks are discovered according the specified set of rules set by the cryptocurrency protocol. An important function of these rules is to protect against attacks on the blockchain and to reach consensus in case multiple instances of the blockchain appear. Proof of work and proof of stake refer to two kinds of restrictions on valid blocks and imply two different consensus mechanisms.

Definition 1. We say that a user works *on top* of block B if he attempts to discover a block that references B as the previous block.

Note that B uniquely determines the entire blockchain so we can denote a blockchain with its latest block and say that blocks are discovered on top of blockchain B.

A cryptocurrency system can be viewed as a distributed database, with copies of the database belonging to infrastructure providers for the currency communicating via a peer-to-peer Internet protocol. In terms of the CAP (consistency, availability and partition-tolerance) theorem [2], cryptocurrency systems are available (every request receives a response) and partition-tolerant (the service still performs even if some nodes fail), but are not consistent. From time to time, different users of the system will see different states of the system as current. In some cases, the inconsistency corresponds to the situation when a new block hash been discovered but has not yet been relayed to all users of the system. To obtain eventual consistency [3], a sound consensus protocol should impose the following requirement:

Condition 1. A user who discovered a block should be encouraged to broadcast it over the network immediately and not hold it for himself.

In other cases, system inconsistency is caused by the blockchain splitting into several branches (Fig. 1), There are various causes of blockchain branching (*forking*):

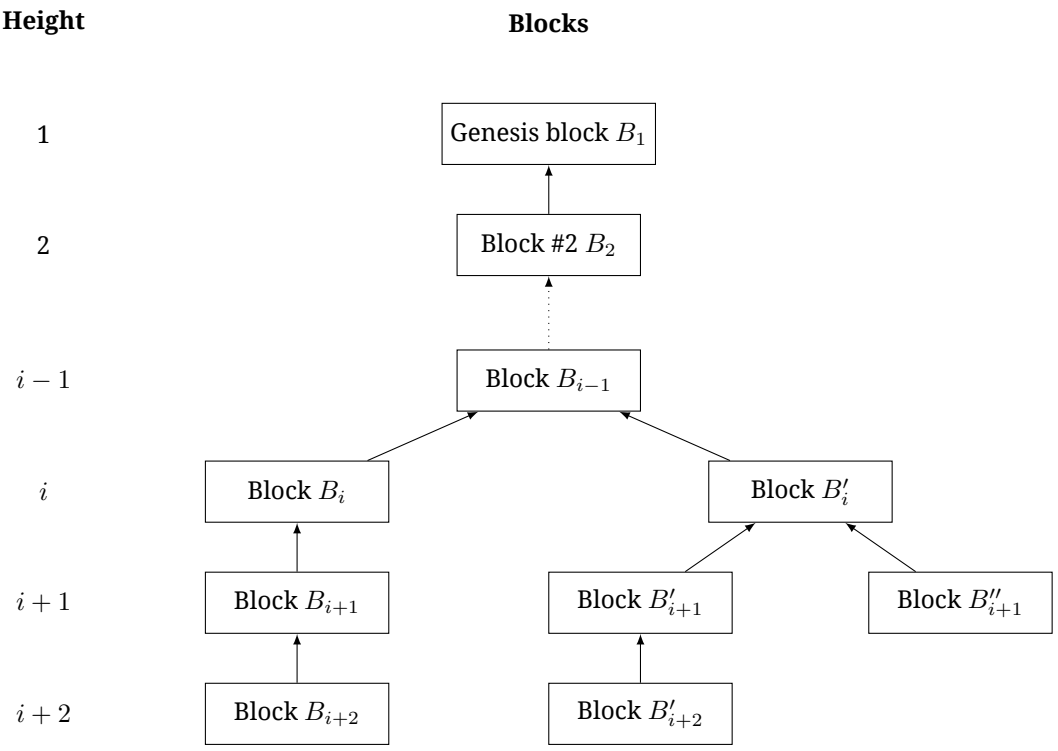

Figure 1: Multiple blockchains in a cryptocurrency system. With a "good" consensus protocol, rational users should mine on top of either B_{i+2} or B'_{i+2}

- Two users discover new blocks at about the same time
- An attacker attempts to reverse completed transactions by forking the blockchain.

In order to discourage deliberate branching, a sound consensus protocol should add the following requirement:

Condition 2. A user should be discouraged from discovering blocks on top of intermediate chains. More precisely, if there is a known block B' referencing the block B, the user should have no reason to build on B.

With the situation depicted in Figure 1, only three blocks can be used as a base for extending the blockchain according to Conditions 1–2: B_{i+2}, B''_{i+1}, and B'_{i+2}. There are further restrictions imposed on valid blockchains; in most cases, these conditions amount to selecting a chain with the maximum number of blocks (which excludes B''_{i+1} from blocks a rational user could build on his own chain). The goal of consensus rules is to assure selection of a single chain; however, it is usually the case that some rules depend on the user. E.g., if there are multiple Bitcoin blockchains with the same length, users should select the one they received first. Thus, there is no way to enforce the selection of the same blockchain for all users (as it is very difficult or impossible to ensure that user-specific rules are followed).

In order for the system to be eventually consistent, its consensus protocol should satisfy the following third requirement:

Condition 3. Consensus rules should be constructed in a way that results in resolving blockchain forks, i.e. one of the competing branches should take over all other branches in a reasonable amount of time.

We will use separate terms for discovering blocks using proof of work and proof of stake algorithms:

Definition 2. The process of solving a computational challenge imposed by a proof of work protocol is called (block) *mining*.

Definition 3. The process of solving a computational challenge imposed by a proof of stake protocol is called (block) *minting*.

1.1 Proof of Work

Consider Bitcoin as an example of a cryptocurrency system secured with a proof of work algorithm. Each block in Bitcoin consists of two parts:

- block header of key parameters, including block creation time, reference to the previous block and the Merkle tree root [4] of the block of transactions
- block list of transactions.

To reference a specific block, its header is hashed twice with the SHA-256 function [5]; the resulting integer value belongs to the interval $[0, 2^{256} - 1]$. To account for different possible implementations, we will use a generic hashing function $\text{hash}(\cdot)$ with a variable number of arguments and range $[0, M]$. For example, arguments of the function can be treated as binary strings and merged together to form a single argument that can be passed to the SHA-256 hashing function.

The block reference is used in the proof of work protocol; in order for a block to be considered valid, its reference must not exceed a certain threshold:

$$\text{hash}(B) \leqslant M/D, \tag{1}$$

where $D \in [1, M]$ is the target difficulty. There is no known way to find B satisfying (1) other than iterating through all possible variables in the block header repeatedly. The higher the value of D, the more iterations are needed to find a valid block; the expected number of operations is exactly D.

The time period $T(r)$ for a miner with hardware capable of performing r operations per second to find a valid block is distributed exponentially with the rate r/D (see Appendix A):

$$P\{T(r) \leqslant t\} = 1 - \exp(-rt/D).$$

Consider n Bitcoin miners with hash rates r_1, r_2, ..., r_n. The period of time to find a block T is equal to the minimum value of random variables $T(r_i)$ assuming that the miner publishes a found block and

it reaches other miners immediately[2]. According to the properties of the exponential distribution, T is also distributed exponentially:

$$P\{T \stackrel{\text{def}}{=} \min(T_1, \ldots, T_n) \leqslant t\} = 1 - \exp\Big(-\frac{t}{D}\sum_{i=1}^{n} r_i\Big);$$

$$P\{T = T_i\} = \frac{r_i}{\sum_{j=1}^{n} r_j}.$$

The last equation shows that the mining is fair: a miner with a share of mining power p has the same probability p to solve a block before other miners. It can be shown that proof of work as used in Bitcoin satisfies Conditions 1–3.

1.2 Proof of Stake

In proof of stake algorithms, inequality (1) is modified to depend on the user's ownership of the particular PoS protocol cryptocurrency and not on block properties. Consider a user with address A and balance $\text{bal}(A)$. A commonly used proof of stake algorithm uses a condition as

$$\text{hash}(\text{hash}(B_{prev}), A, t) \leqslant \text{bal}(A)M/D, \tag{2}$$

where

- B_{prev} denotes the block the user is building on,
- t is the current UTC timestamp.

For various reasons, some cryptocurrencies use modified versions of (2) which we discuss in the corresponding sections.

Unlike (1), the only variable that the user can change is the timestamp t in the left part of equation (2). The address balance is locked by the protocol; e.g., the protocol may calculate the balance based on funds that did not move for a day. Alternatively, a PoS cryptocurrency may use unspent transaction outputs as Bitcoin does; in this case, the balance is naturally locked. A proof of stake protocol puts restrictions on possible values of t. For example, if t must not differ from the UTC time on network nodes by more than an hour, then a user can attempt no more than 7200 values of t. Thus, there are no expensive computations involved in proof of stake.

Together with an address A and a timestamp t satisfying (2), a user must provide a proof of ownership of the address. To achieve this, the user can sign the newly minted block with his signature; in order to produce a valid signature, one must have a private key corresponding to the address A.

The time to find a block for address A is exponentially distributed with rate $\text{bal}(A)/D$ (see Appendix A). Consequently, the (2) implementation of proof of stake is fair: the probability to generate a valid block is equal to the ratio of user's balance of funds to the total amount of currency in circulation. The time to find a block for the entire network is distributed exponentially with rate $\sum_a \text{bal}(a)/D$.

[2]There is selfish mining attack that is based on hiding mined blocks [6]. We don't consider it here and assume that all miners are honest.

Thus, if the monetary supply of the currency $\sum_a \text{bal}(a)$ is fixed or grows at a predictable rate, the difficulty D should be known in advance:

$$D = \frac{1}{T_{ex}} \sum_a \text{bal}(a),$$

with T_{ex} denoting the expected time between blocks. In practice, D needs to be adjusted based on recent blocks because not all currency owners participate in block minting.

1.3 Delegated Proof of Stake

Delegated proof of stake (DPoS) is a generic term describing an evolution of the basic PoS consensus protocols. DPoS is utilized in BitShares (Section 2.5), as well as proposed algorithms such as Slasher and Tendermint (Section 4). In these protocols, blocks are minted by a predetermined set of users of the system (*delegates*), who are rewarded for their duty and are punished for malicious behavior (such as participation in double-spending attacks). In DPoS algorithms, delegates participate in two separate processes:

- building a block of transactions
- verifying the validity of the generated block by digitally signing it.

While a block is created by a single user, to be considered valid, it typically needs to be signed by more than one delegate.

The list of users eligible for signing blocks is changed periodically using certain rules; for example, in Slasher, delegates for each block are selected based on stake and blockchain history. The set of delegates for each block is typically small; a notable exception is Tendermint, for which each block can be signed by any of the users of the system. In some DPoS versions, a delegate needs to show commitment by depositing his funds into a time-locked security account (which is confiscated in case of malicious behavior); this version of DPoS is often referred to as *deposit-based proof of stake*.

Delegated proof of stake does not use the conditions of (2). Stake is factored into DPoS with one of the following methods:

- Delegates may be elected based on their stake in the system
- Delegates may receive votes from all users of the system with voting power depending on a voter's stake
- Delegates' votes on valid blocks may have power proportional to the size of their security deposit.

Overall, delegated proof of stake is less standardized than basic PoS; we detail existing implementations in the corresponding sections.

1.4 Objectivity of Consensus Protocols

One of the tasks of a consensus protocol is to provide a way for newcomers to determine the current state of the system based on information received from peer nodes. This task is not trivial as some of the nodes can belong to a party performing a Sybil attack [7].

Definition 4 ([8])**.** A consensus protocol is *objective* if a new node can independently arrive to the same current state as the rest of the network based solely on protocol rules (e.g., a definition of the genesis block) and messages propagated across the system (e.g., a set of all blocks).

Proof of work consensus is an example of an objective protocol; as long as a new node is connected to at least one "honest" user, it will choose the valid blockchain, as it has higher cumulative computational difficulty. Proof of stake, on the other hand, is not objective. Indeed, consider an attacker with substantial computing power. Provided the attacker's fork is long enough, difficulty within it is adjusted to reflect the situation in which only accounts controlled by the attacker are active; this allows the attacker to generate a chain longer than the valid blockchain. While long range forks would be rejected by existing users of the system (e.g., by introducing a rule that limits the length of a possible fork), newcomers without prior knowledge of the current state would still choose the attacker's blockchain.

Definition 5. A consensus protocol is *weakly subjective* if a node needs a recent state in addition to protocol rules and messages propagated across the system to independently determine the current state of the system.

In the case of proof of stake, if there is a rule that disallows forks with a branching point more than N blocks in the past, it suffices to read the contents of a block with depth N or less to reliably determine the current state of the system. A newcomer can access this block from a trusted source (e.g., a website dedicated to the currency in question). While this method could undermine the security and decentralization aspects of a PoS system, in *Proof of stake: How I learned to love weak subjectivity* [8], it is asserted that weak subjectivity is a good way to combine computer-driven and social-driven security.

2 Implementations

There are currently several noteworthy cryptocurrencies based on proof of stake consensus (Table 1):

- Peercoin, or PPCoin (peercoin.net)
- Nxt (nxt.org)
- BlackCoin (blackcoin.co)
- Novacoin (novacoin.org).

Delegated proof of stake is used by BitShares, a digital asset (bitshares.org). Developers of Ethereum (ethereum.org) plan to change consensus to delegated proof of stake in the future; as of now, Ethereum exclusively relies on proof of work.

2.1 Peercoin

Peercoin, or PPCoin, is a digital currency founded in 2012 by anonymous developer Sunny King. Peercoin uses a combination of proof of work and proof of stake [10]:

Table 1: Market statistics of PoS digital currencies. For market cap and 24h volume, we used median values taken over a period of one month (July 24 to August 23, 2015) from CoinMarketCap [9]

Name	Supply, million	Market cap, $ million	24h volume, $ thousand	Remarks
Peercoin (PPCoin)	22.6	10.8	83	Hybrid PoW / PoS
NXT	1000	10.7	27	
BlackCoin	75.0	2.2	15	Hybrid PoW / PoS
Novacoin	1.14	1.4	20	Hybrid PoW / PoS
BitShares	2511	11.7	55	Delegated PoS

- Proof of work is used for the distribution of new coins.
- Proof of stake is used to secure transactions, i.e., as a primary means of generating blocks of transactions.

The structure of the Peercoin blockchain is very similar to Bitcoin: just like Bitcoin, Peercoin operates based on unspent transaction outputs. New blocks are found every 10 minutes on average; transactions consist of inputs referencing UTXOs and outputs.

Peercoin uses the same proof of work algorithm based on the SHA-256 hashing function as Bitcoin; however, mining rewards depend not on the block height but rather on the target difficulty. When the difficulty increases, the reward goes down; a 16x rise in difficulty halves the block reward. As the mining reward approaches zero, there is less incentive to use PoW compared to PoS. In this regard, Peercoin is energy efficient in the long run.

Proof of stake in Peercoin is based on a notion of *coin age*. Coin age was introduced in Bitcoin to prioritize transactions [11].

Definition 6. *Coin age* of an unspent transaction output is its value multiplied by the time period after it was created. A transaction spending a previously unspent output *consumes*, or *destroys*, its coin age.

Coin age is also used in Peercoin to determine the valid blockchain, which is the blockchain with the highest total consumed coin age in all transactions in all blocks (cf. Bitcoin mechanism using total expected work spent on building the chain).

The possibility to create a block using an unspent output U in Peercoin is determined by the condition

$$\text{hash}(\text{hash}(B_{prev}), U, t) \leqslant \text{bal}(U)\,\text{age}(U)M/D, \tag{3}$$

with $\text{age}(U)$ denoting time after U was created. The PoS variation (3) means that, instead of an amount of currency, the probability to create a block is proportional to the total coin age of the UTXOs belonging to a user.

The first transaction in the minted block is a special coinstake transaction that proves that the "lucky" unspent output belongs to a user that minted the block. The first input of the coinstake transaction must be an unspent output U satisfying (3); in most cases, the user spends U by sending its value back to himself.

The stated reason behind the PoS implementation (3) is to result in a more even distribution of minted block owners. Indeed, minting a block lessens the probability for the same user to mint the following block, as the total coin age of his UTXOs decreases. The result is that users with small stakes are still able to mint a block provided they wait long enough.

To provide stability for the blockchain, Peercoin uses periodically broadcasted checkpoints. This type of broadcasting introduces a centralization point. While checkpoints are present in the Bitcoin protocol [12], they can be switched off and are hardcoded into the protocol rather than broadcasted; accordingly, Bitcoin's checkpoint mechanism does not imply a risk of centralization.

2.2 Nxt

Nxt is a pure proof of stake cryptocurrency that began in 2013. Unlike Peercoin, Nxt does not use proof of work to create new coins; the entire available supply of 1 billion coins (NXTs) was present in the system from the genesis block. Thus, the only incentive to mint blocks is to collect transaction fees.

Transactions in Nxt are radically different from Bitcoin ones, which in turn results in other protocol differences [13]:

- Nxt transactions don't have locking and unlocking scripts. Instead, the Nxt protocol uses accounts to encode a sender and a recipient of a transaction. This restriction severely limits the capabilities of the system to handle smart contracts compared to Bitcoin.
- There are various types of transactions, e.g. special transactions for messaging, colored coins, digital goods, etc.
- There is a hardcoded limit on the number of transactions in a block: no more than 255. On the other hand, the mean time between blocks is 1 minute.
- All transactions have an expiration time (24 hours by default). Transactions that are not included into a block when the expiration time passes are removed from the memory pool and must be resent.

Nxt uses a modified version of proof of stake consensus. Instead of (2), the inequality

$$\text{hash}(\text{hash}(B_{prev}), A) \leqslant t\,\text{bal}(A)M/D \tag{4}$$

must hold true for a user's account A in order to be able to mint a block. In (4), B_{prev} is a block used to extend the blockchain, and t is the time that has passed since B_{prev} was created. Thus, after a new block is propagated across the network, the chance to satisfy the PoS condition for any given user is initially low and grows with time. The hashing function in (4) is calculated by

1. using a special field (generation signature) of B_{prev}
2. signing the generation signature with the account's public key
3. using the first 8 bytes of the resulting signature.

Because public keys corresponding to accounts are public knowledge, anyone can check whether (4) holds for any user account A. As in Bitcoin, the valid blockchain in Nxt is determined as a chain with maximum expected total difficulty D.

Note that unlike (2), (4) does not include *any* variables that user A could influence. Therefore, PoS in Nxt has the following interesting property: it is possible to predict with reasonable accuracy the minter of the next block. To do this, one just has to find

$$\hat{A} = \arg\min_{A} \frac{\text{hash}(\text{hash}(B_{prev}), A)}{\text{bal}(A)}$$

for all user accounts.

Nxt has a different underlying probabilistic model compared to Bitcoin. It is favorable for big players: if a user holds a substantial amount of currency, his probability to mint a block is higher than his stake [14] (see Appendix B).

2.3 Novacoin

Novacoin is a hybrid proof of work / proof of stake cryptocurrency quite similar in principle to Peercoin. The main differences are:

- Novacoin uses a more conservative reward formula for mining: the reward is halved every 64x multiple of difficulty [15] (compared to 16x in Peercoin).
- A slightly different method based on *coin day weight* (instead of coin age) is used in proof of stake (3). Furthermore, the coinstake reward for a proof of stake block depends on the consumed coin age and PoS difficulty [16].

Coin day weight in Novacoin is calculated similarly to coin age, with a shift of 30 days and a cap of 90 coin days. For example, the coin day weight of a UTXO with the value of 1 Novacoin changes with time in the following way:

Age, days	1	10	30	60	90	120	200
Coin day weight	0	0	0	30	60	90	90

2.4 BlackCoin

BlackCoin is a hybrid cryptocurrency which used proof of work for the initial distribution of wealth and now relies solely on proof of stake. BlackCoin is the only one of considered currencies that uses the unmodified proof of stake approach (2), according to which the user's stake is computed as his share of total coin supply [17]. New blocks are minted every 64 seconds on average; minters are paid with 1% yearly interest (which means BlackCoin is an inflationary currency).

2.5 BitShares

BitShares is a polymorphic digital asset which can be used to create fungible assets pegged to particular markets (e.g., US dollars) [18]. It is similar to a colored coins system, with BitShares acting

as an internal currency (like XCP for Counterparty). One of the innovative features of BitShares is its delegated proof of stake consensus algorithm. The founders of BitShares are now developing BitShares 2.0, an enterprise-grade financial smart contract platform utilizing several technologies from BitShares including DPoS.

Delegated proof of stake in BitShares relies on the concept of *witnesses*. Stakeholders can select an arbitrary number of witnesses to generate blocks. Each stakeholder has a number of votes equal to the amount of BitShares he possesses; votes can be distributed among witnesses in an arbitrary way. Besides witness candidates, a user also selects a number of witnesses he estimates is sufficient for decentralization. Naturally, a user cannot vote for more witnesses than he believes is necessary.

After results of voting are tallied up, the top N witnesses are selected. N is defined as the minimal number satisfying at least 50% of stakeholders' votes. Selected witnesses produce blocks every two seconds in turns. After each of N witnesses has had his turn, the list of witnesses is shuffled so that the order of block minters constantly changes.

Similar to witnesses, users of the system elect *delegates*, who have the privilege to change network parameters, including transaction fees, block sizes and intervals, as well as witness rewards. To make changes to the network protocol, delegates co-sign a special account (so-called genesis account). After the majority of delegates have approved a proposed change, stakeholders then have a two-week period when they can recall their delegates and cancel the changes. Unlike witnesses, delegates are not rewarded for their efforts.

3 Attacks and Problems

This section describes attacks that are possible in each PoS-based consensus as well as some protocol-specific attacks. Several attack descriptions are taken from *Cryptocurrencies without proof of work* [19].

The majority of problems with PoS protocols arise from the fact that a protocol is not aware of anything except for its blockchain [20]. In a proof of work system, there is an external factor, namely the amount of computational work involved to find a solution to (1). With proof of stake, there is nothing physical "anchoring" the blockchain in reality; thus, one can intuitively see PoS consensus as more prone to attacks.

3.1 Nothing at Stake Problem

The naïve proof of stake algorithm relying solely on (2) has a serious problem: if there is a fork of the blockchain (whether accidental or deliberate), the rational behavior for all users of the network is to mint blocks on both branches. With a proof of work algorithm, such behavior is irrational. By splitting the resources on multiple branches, a miner diminishes the probability to find a block on each of them; the optimal strategy in a PoW system is always to mine on a single branch. In contrast, in a proof of stake algorithm, the probability to find a block does not decrease if the user is attempting to mint on multiple blockchain branches. Thus, the rational behavior in a PoS system is to mint blocks on top of

all branches that a user is aware of. Worse than that, naïve PoS does not satisfy Condition 2: under some circumstances, it may be appropriate for a user to build on top of an intermediate blockchain. See Section 3.4 for an example.

The problem makes it easy to perform double-spending or other sorts of attacks relying on forking the blockchain. As long as users of the system believe the attack may succeed, they will support it by minting on top of the attacker's blockchain. While forking attacks will be opposed by users with a large amount of currency who will fear to lose their money, if the currency is evenly distributed among many users, the attack is more likely to succeed.

There are several proposed solutions to "nothing at stake" problem; they are covered in Section 4. Delegated PoS algorithms are generally not affected by this problem, as they have stakes in the form of block minters' funds; these funds would be confiscated if a minter partakes in an attack on the system.

3.2 Initial Distribution Problem

In a proof of stake system, there is always a concern that the initial holders of coins will not have an incentive to release their coins to third parties, as the coin balance directly contributes to their wealth. In Bitcoin and other PoW systems, early adopters of the technology are at the same position as the rest of the users: in order to mine coins, they need to improve their hardware continuously and optimize resource consumption. In a proof of stake system, a user that acquired 10% of the coins when the system was just launched (e.g., for $1,000) is at an advantage compared to users with the same funds when the system has gained popularity and $1,000 translates to 0.01% of all coins.

In order to deal with this situation, PoS implementations utilize additional algorithms to distribute the initial wealth; e.g., both Peercoin and Ethereum use proof of work algorithms to create new coins (see Section 4).

3.3 Long Range Attack

In a system with PoS consensus, an attacker possessing enough computational power can attempt to build an alternative blockchain starting from the very first block. In PoW-type systems, this attack is prevented by the enormous amount of computational power needed to build the blockchain from scratch; however, this task is within the realm of possibility with proof of stake. As the attacker is able to move coins freely in the blockchain he is building, he has a much higher dimensionality of the search space; therefore, this kind of attack may be preferable to building an alternative blockchain with a recent branching point. The attack persists in delegated proof of stake, where it can be performed by a majority of delegates.

To prevent a long range attack, the protocol can specify the maximum allowed depth of a branching point. For example, in Nxt a user may not accept the alternative blockchain if it differs from the existing blockchain in more than the last 720 blocks (which corresponds to 12 hours). The restriction, however, does not solve the problem for new users. When a new user connects to the network, he sees multiple blockchains with no prior knowledge of their authenticity. If the attacker's blockchain

is preferable to the valid blockchain in terms of the protocol, new users will adopt it instead. One way to know which is the right blockchain is to download it from a trusted source; however, this makes the system semi-centralized and not completely trustless. Nevertheless, blockchain downloads from trusted sources are used in virtually all existing proof of stake systems.

3.4 Bribe Attack

In this attack, an attacker attempts to double-spend his funds in the following way:

1. Buy some goods or services
2. Wait until the payment transaction is considered confirmed by the merchant
3. Announce a reward for building on top of a truncated blockchain that does not include the payment transaction. For example, if merchant waited for six confirmations, the attacker will start with the blockchain without the six latest blocks. The attacker may offer a larger reward for users that mint only on top of the attacker's blockchain (without this, the attacker's blockchain would never catch up to the correct one).
4. The attacker may continue paying bribes even when the lengths of their blockchain and the correct blockchain become equal in order to gain support of most stakeholders.

Users participating in the attack lose nothing if the attack fails; for the attacker, the attack is profitable as long as the total amount of paid bribes is less than the value of the goods. To compare, in a PoW system, a similar attack would require the attacker to bribe the majority of miners. Furthermore, in this case miners lose resources spent on computations if the attack fails, so the amount of bribes most probably would be prohibitively high. An attack is not feasible in a delegated PoS system for reasons described in Section 3.1.

3.5 Coin Age Accumulation Attack

This attack is specific to Peercoin and other systems using coin age instead of wealth as a measure of user's stake.

In the first versions of Peercoin protocol, coin age was uncapped. This means that by waiting long enough, an attacker could potentially accumulate enough coin age to effectively overtake the network. For example, an attacker owning 5% of all coins could split his money into multiple outputs and wait until the age of his UTXOs would become 10 times more than average. After that, the attacker could mint multiple blocks in sequence with high probability (provided his individual UTXOs are sufficiently small) to perform double-spending or other malicious activity. If there were multiple users attempting this attack, it would lead to deterioration of the network.

In later versions of the Peercoin protocol, the age of a UTXO is capped at 90 days. Similarly, the coin age parameter is capped in Novacoin and BlackCoin. While capping makes an accumulation attack significantly less likely, it also diminishes the benefits of using coin age instead of stake in (3).

3.6 Precomputing Attack

Let A be the minter of a certain block B_h at height h; i.e., A satisfies (2) with the parameters corresponding to B_h. If A possesses substantial computing power, he can influence the hash of block B_h in order to be able to mint the next block B_{h+1}, e.g. by adding a new transaction into B_h. To reserve B_{h+1} for himself, A scans all accounts and calculates whether condition (2) holds for each allowed timestamp t. If the hash of B_h is "bad", i.e. computations show that the next block will be minted by another user, the attacker would change the parameters of the inserted transaction and try again. The attacker can build a long chain of blocks to collect more fees than is fair and attempt to double-spend (by building his chain of blocks in secret and then releasing them all at once to override the correct blockchain with a transaction that the attacker wants to reverse).

The effectiveness of a precomputing attack depends on the attacker's stake and the total number of accounts or UTXOs in the system. In a PoW system, this attack is virtually impossible, as it takes exponentially more work to generate a block with a "good" hash than to generate a valid block. Likewise, in a delegated PoS system, a sequence of block signers cannot be influenced by the properties of the newest block; therefore, this type of consensus is resistant to precomputing attacks.

3.7 Comparison with Proof of Work

PoW-based currencies have certain exposure to double-spend attacks; however, the chance of a successful double-spend progressively decreases as a payment transaction gains confirmations and heavily depends on the amount of mining power an attacker possesses [21]. To mitigate the risk of double spending, merchants commonly wait for a certain number of confirmations (e.g., 6). Additionally, there exist mechanisms to decrease the risk in fast payments [22].

Types of attacks common to both proof of work and proof of stake are denial of service (DoS) and Sybil attacks. A DoS attack is aimed to disrupt the normal operation of the cryptocurrency network by flooding the nodes. For example, an attacker can flood the network with many low-value transactions, as was the case in a large-scale flood attack on the Bitcoin network, which took place on July 2015 [23]. In a Sybil attack, the attacker disrupts the network by creating a number of misbehaving nodes. The susceptibility of the network to DoS and Sybil attacks depends on not only the type of consensus utilized by the network but also the details of the network protocol. There are no inherent properties that would make PoS less susceptible to these types of attacks compared to PoW.

One of the few attack vectors that is specific to proof of work consensus is selfish mining [6]. In selfish mining, an attacker selectively reveals mined blocks in order to waste computational resources of honest miners. As there are no expensive resources involved in block generation in the case of PoS consensus, the attack is ineffective for PoS currencies. On the other hand, there is no evidence that a selfish mining attack has ever been successfully performed in Bitcoin [24], and some research argues that the attack description is based on faulty assumptions [25].

See Table 2 for a comparison of proof of work and proof of stake in terms of vulnerability. For PoW consensus, a degree of susceptibility to attacks can be predicted simply based on the total hash rate of

the system. In the case of PoS systems, there is no equivalent measure of the network "health status":

- if a stake is distributed evenly among many users, the system is prone to attacks that are based on a blockchain fork
- if there are users with large stakes, they can disrupt the operation of the network (e.g., by censoring transactions).

Table 2: Vulnerability of proof of work and proof of stake consensus mechanisms to attack vectors

	Vulnerability		
Attack type	PoW	PoS	Delegated PoS
Short range attack (e.g., bribe)	−	+	−
Long range attack	−	+	+[3]
Coin age accumulation attack	−	maybe[4]	−
Precomputing attack	−	+	−
Denial of service	+	+	+
Sybil attack	+	+	+
Selfish mining	maybe[5]	−	−

3.8 Attack Costs

3.8.1 Bribe Attack

Consider the following double-spending attack scenario:

1. The attacker performs a spending transaction he wants to reverse later.
2. Immediately after the transaction, the attacker starts to build an alternative chain based on the block prior to the one containing the transaction. The attacker builds on the alternative chain in secret.
3. After the transaction gains the necessary number of confirmations (e.g., 6) and the attacker's chain is longer than the valid chain, the attacker publishes it whole. The attacker's chain is accepted as the new valid blockchain, and the transaction is reversed.

To perform a successful attack with 100% probability, the attacker needs to control more than 50% of the resources used to secure the system (computational power in case of PoW; liquidity in case of PoS) for the duration of the attack. Thus, in the case of proof of stake, the attacker does not need to *own* more than a half of the currency – he only needs to gain the privilege of accessing 50% of the currency in circulation for a few hours, e.g., by paying bribes as described in Section 3.4. The bribe amount is

[3] Can be prevented by using social-driven security in addition to protocol rules
[4] Depends on the type of a proof of stake condition
[5] It is unclear whether selfish mining attacks are realistically possible

proportional to the rewards that participating users would lose from not building on top of the valid blockchain (recall that if bribed users build on top of both the attacker's and the valid chain, attacker's blockchain would never catch up). Suppose

- minting rewards result from transaction fees,
- transaction throughput is 100,000 transactions per day (approximately equal to Bitcoin throughput),
- transaction fee is $0.10 (higher than in Bitcoin currently).

Then, the daily minting reward would be $10,000; if the attack lasts for an hour, the attacker would need to pay less than $500 in bribes.

Compare this with cost of an attack in Bitcoin. The reward associated with each block is 25 bitcoins, or about $5,000. In case of economic equilibrium, the price to mine a block is close to this number; assume it equals $4,000. If an attack needs to last for 6 blocks, the attacker would need to pay at least for 7 blocks in order to override the valid blockchain (which amounts to $28,000). This is more than 50 times higher than in the case of proof of stake. There is another consideration that makes an attack on proof of work less reliable and more costly. The attack becomes obvious after the attacker's blockchain is published because the reorganization of 6 last blocks is statistically a very rare event. As there is a relatively small number of miners in the Bitcoin ecosystem, participating in the attack would damage their reputation. In case of a PoS system, there is no reputation to speak of because most stakeholders are likely anonymous.

Proof of stake systems may introduce slightly inflationary economics to reward minters. Consider a system with a $3 billion market capitalization (slightly less than Bitcoin) and a 1% yearly inflationary rate. The daily minting reward of the system is

$$\$3 \cdot 10^9 \cdot 0.01/365 \approx \$82 \cdot 10^3.$$

This number implies that the cost of 1 hour attack is close to $3,400 – still significantly less than for Bitcoin.

In order for the attack to be profitable, the attacker would need to double-spend a large amount of money. Accordingly, the counterparty could demand more confirmations for the attacker's spending transaction. In both PoW and PoS, the direct cost of the attack grows linearly with the required number of confirmations; thus, the ratio of costs in the two cases should roughly remain the same. However, as the attack on a PoW system continues, it becomes increasingly obvious for all participants of the system because the attack drains the hash rate on the valid blockchain. If there is a relatively small number of miners (as it is the case in Bitcoin), users might identify miners participating in the attack even before it is finished; if attackers are mining pools, their participants would have an incentive to switch to other pools or temporarily suspend their operations.

In the case that an attack on a proof of stake currency is performed multiple times, the currency exchange rate drops as a response; this would make it easier for the attacker to gather more resources

for future attacks. This is not so for proof of work currencies, as mining costs (electricity, equipment, etc.) do not depend on the currency exchange rate.

3.8.2 Accumulation Attack

Consider coin age accumulation attack described in Section 3.5. This type of attack is impossible in PoW systems; it only affects proof of stake systems, for which the total consumed coin age determines the valid blockchain. We will consider Peercoin as an example susceptible to an accumulation attack.

Suppose the attacker wants to reverse a transaction. To accomplish this, he would create a forked blockchain, for which he destroys all his coin age by moving his funds to new unspent outputs (which still belong to the attacker). The attacker's blockchain could outweigh the valid blockchain in terms of consumed coin age, provided that the attack would be performed fast enough and the attacker would have enough coin age at his disposal.

Assume the attacker has x share of the total coin age of the system S; our goal is to determine x high enough to accomplish the attack with probability p. Let r_d denote the coin age destruction rate of the system – the speed with which coin age is destroyed in transactions (e.g., transactions in Bitcoin consume ≈5 million bitcoin days each day [26], which implies $r_d \approx 5 \cdot 10^6$ bitcoins). The attacker's block should satisfy such a proof of stake condition (3), which means the attacker would need to spend time T generating it. T is an exponentially distributed random variable with mean t_0/x, where t_0 is the expected time between blocks (in Peercoin, $t_0 = 10$ minutes). In order for the attacker's chain to succeed with probability p, the coin age destroyed by the honest part of the network (Tr_d) needs to be lower than the coin age destroyed by the attacker:

$$P\{Tr_d < xS\} = p.$$

Accounting for cumulative distribution function of the exponential distribution, we obtain

$$\begin{aligned} 1 - \exp\Big(-\frac{x}{t_0} \cdot \frac{xS}{r_d}\Big) &= p; \\ x &= \sqrt{-\log(1-p)\frac{t_0 r_d}{S}}. \end{aligned} \tag{5}$$

In order to calculate a quantitative value of attacker's share x, we need to determine relationship between the destruction rate r_d and accumulated coin age of the system S. Assuming the accumulated coin age grows with time, $r_d < B$, where B is the total amount of currency in circulation. The above inequality holds for Peercoin, where $r_d \approx 10^7$ peercoins and $B \approx 2.3 \cdot 10^7$ peercoins [27]. We will assume $r_d \approx B$. Coin age is capped at 90 days in Peercoin. Thus, the effective coin age of any unspent output used in condition (3) cannot exceed 90 days; correspondingly, $S \leqslant B \cdot 90$ days. We will use a value rather close to maximum, $S \approx B \cdot 60$ days. This estimate is supported by the low transaction throughput of the Peercoin network [28].

The estimates for r_d and S give us

$$x = \sqrt{-\log(1-p)\frac{10\text{ min} \cdot B}{B \cdot 60\text{ days}}} = \sqrt{-\log(1-p)\frac{1}{144 \cdot 60}}. \tag{6}$$

For an attack to succeed with probability $p = 0.5$, the attacker needs to have $x \approx 0.9\%$ of all coin age; for $p = 0.9$, (6) yields $x \approx 1.6\%$.

Just as in other types of attacks, the attacker does not need to *possess* x share of coin age; he may bribe other users to participate in the attack. As participants have a diminished probability to collect transaction fees until their coin age falls back to normal (roughly 60 days), the total bribe amount should be comparable to

$$x \cdot \langle \text{daily minting reward} \rangle \cdot 60 \text{ days}/2.$$

For a system with $10,000 daily minting reward and $x = 1.6\%$, the bribe amount is nearly $4,800. Under these assumptions, the attacker would be able to override the blockchain history for $xS/r_d \approx 1$ day.

3.8.3 Long Range Attack

Short range attacks described earlier in this section are made expensive in the case of delegated proof of stake, so we need to consider the cost of long range attacks as well. For proof of work systems, the cost of a long range attack is prohibitively high. For example, an attack on Bitcoin lasting for 1,000 blocks would require $4 million at very least (and, unlike a short range attack, it would be highly visible as observed network hash rate would drop in half for an extended time).

In earlier versions of proof of stake, the cost of a long range attack would be much lower; as we showed in the previous section, a one day long attack may cost about $5,000 in a system where a valid blockchain is determined based on total destroyed coin age. In delegated PoS, an attack typically requires collusion by 2/3 of delegates; its cost is difficult to assess, as delegated PoS protocols use differing methods to select, reward and punish delegates.

4 Modifications and Hypothetical Algorithms

4.1 Block References in Transactions

In several proof of stake systems (e.g. in BitShares, Chains of Activity protocol proposed in *Cryptocurrencies without proof of work* [19], and in the *Transactions as Proof of Stake* algorithm [29]), each transaction can optionally include a hash of a recent block known to a transaction sender. This change makes it impossible to include such a transaction into any chain not containing the referenced block. Therefore, an attacker creating an alternative blockchain and users supporting the attack by minting on top of attacker's chain cannot collect transaction fees for a majority of transactions.

The modification relies on a somewhat optimistic assumption that each user is able to detect the correct blockchain at every moment; it is not effective if a minting reward is not the result of transaction fees.

4.2 Hybrid PoW / PoS Consensus

Some cryptocurrencies implementing proof of stake deal with the initial distribution problem (Section 3.2) by using a limited version of proof of work to increase currency supply; examples are Peercoin and Novacoin. Hybrid consensus operates by enabling two types of valid blocks:

- PoW blocks satisfying (1)
- PoS blocks satisfying (2) or a similar condition.

A primary alternative is a completely premined currency (such as Nxt). A solution with PoW and PoS blocks separated in time is used for Ethereum. Currently, Ethereum exclusively relies on proof of work; PoS consensus will be introduced in the future when the system becomes more mature.

Hybrid PoW / PoS consensus is secure against long range attacks provided there is enough hashing power in the network. Inclusion of PoW blocks into a blockchain can protect against other attacks described in Section 3, too. However, as proof of stake is usually introduced as an alternative to proof of work, combined solutions eventually abandon PoW completely or diminish its role.

4.3 Tendermint

Tendermint [30] is a proposed concept of a blockchain secured by a modified proof of stake consensus. In Tendermint, each block is cryptographically signed by *validators*. A validator is a user committing to his duty by locking his coins in a *bonding transaction*; the voting power of each validator is proportional to the amount of locked currency. After finishing his validator duty, a user unlocks his funds by posting an *unbonding transaction*; the funds are then unlocked with a certain delay (*unbonding period*).

The block is considered valid if it signed by validators holding at least $2/3$ of all voting power. Thus, a fork can occur only if there is a group of validators with $1/3$ total voting power signing both of competing chains. If there is a fork, validators signing multiple chains can be punished by a user publishing an *evidence transaction* containing a proof of their maliciousness, i.e., digital signatures of blocks at the same height on both chains. The evidence transaction destroys all locked funds of misbehaving validators. This logic protects against short-range attacks (e.g., bribe attacks described in Section 3.4). However, it does not prevent long-range attacks: validators with $2/3$ of the voting power can conspire to publish a blockchain fork after all their funds are unlocked. To avoid long-range attacks, a mechanism prohibiting long-range forks can be introduced (as in Nxt).

4.4 Slasher

Slasher is a hybrid PoS / PoW algorithm outlined by Vitalik Buterin, one of Ethereum's architects [31]. Unlike other hybrid algorithms, Slasher exclusively uses proof of work to generate blocks; each block is validated with *both* PoW and PoS.

Blocks are mined with proof of work (1). However, instead of including a coinbase transaction as a reward, a miner specifies a value hash(n) for some big integer value n. The number serves as a proof

of mining; the miner can claim his reward by issuing a special transaction uncovering n. The mining reward is locked for 100 blocks and is a limited time offer; the reward for the block at height h can be redeemed in a transaction recorded in one of the blocks $h + 100$, $h + 101$, ..., $h + 900$. The mean time interval between blocks is 30 seconds.

To form a valid block, it has to be cryptographically signed. Signers of a block at height h are selected at random using a condition similar to (2):

$$\text{Signers}(h) = \{A : \text{hash}(n(h-2000), n(h-1999), ..., n(h-1901), A) \leqslant 64M\,\text{bal}(A)/B\}, \qquad (7)$$

where $B = \sum_a \text{bal}(a)$ is the total number of coins in circulation, and $n(i)$ is a proof of mining used in block i. Equation (7) means that there are 64 signers of each block on average, and the probability to become a signer is proportional to a user's amount of currency. The number of signatures is used to determine the valid blockchain among several choices. By signing a block, signers gain a reward locked for the next 1,000 blocks. The signing reward is higher than the mining reward in order to discourage an arms race among miners.

To fight blockchain forks, Slasher utilizes a mechanism similar to Tendermint. When a user sees multiple blocks with the same height signed by a same signer, the user can publish a transaction with these signatures. If this transaction is included into a block before the signer's reward is unlocked, 33% of the reward is sent to the user who discovered malicious behavior, and the rest is destroyed.

Slasher's proof of stake implementation strongly discourages users from short-range forks. Indeed, vulnerability of PoS to short-range attacks stems from the following observation. In a PoS model (2), the chances to mint a block on top of each of the competing chains are independent, as they depend on a hash of the latest block in a chain. Users have an incentive to mint on top of each chain because this increases their expected earnings. In Slasher, however, the probability to become a signer of a block is decided with a large time lag and is the same for all branches provided they are relatively short. Thus, users have no incentive to support a fork of the main chain, as they know in advance whether they will become signers of future blocks. It does not make sense to support multiple branches unless a user believes the fork will last for at least 2,000 blocks.

As proof of work is used to generate blocks, long-range attacks on Slasher require substantial computational resources. This puts Slasher at a significant advantage compared to delegated proof of stake consensus models that do not utilize proof of work (e.g. BitShares).

4.5 Casper

Casper is a conceptual proof of stake algorithm proposed by Ethereum developer Vlad Zamfir [32]. Similar to Tendermint, Casper utilizes the concept of block validators. A validator bets on blocks by assigning each of them a probability; the sum of the probabilities for blocks at the same height must equal 100%. To select one block at each height, validators may need to perform several betting rounds until at least $2/3$ of validators bet on a certain block with a high probability (e.g., 99%). To resist majority attacks, Casper punishes deviations from the protocol by withholding generation rewards

and locked funds from misbehaving validators.

5 Conclusion

Currently, there are several digital currencies implementing some form of proof of stake consensus including Peercoin, Nxt, Novacoin, BlackCoin and BitShares. However, pure proof of stake approaches pose substantial security threats that cannot be recreated in proof of work systems (including Bitcoin). These problems are inherent to proof of stake algorithms, as proof of stake consensus is not anchored in the physical world (cf. with hashing equipment in proof of work).

That is why virtually all of currencies relying on proof of stake use additional mechanisms to address security issues. For example, the initial distribution problem can be solved by using a time-constrained version of proof of work; double-spend attacks can be prevented by including information about recent blocks into transactions. Still, these improvements can be seen as ad-hoc and incomplete. Unlike proof of work, proof of stake consensus is not objective; the state of a PoS system cannot be reliably determined by new users based solely on protocol rules and a list of blocks and other network messages obtained from peers. In order to prevent long range forks of the blockchain, a proof of stake system needs to implement weak subjectivity by combining protocol rules with social-driven security. The social component of PoS systems weakens their decentralization and mathematical soundness. As Vitalik Buterin puts it [20], "[A]ll "pure" proof-of-stake systems are ultimately permanent nobilities where the members of the genesis block allocation always have the ultimate say. No matter what happens ten million blocks down the road, the genesis block members can always come together and launch an alternate fork with an alternate transaction history and have that fork take over."

A recent development in proof of stake are delegated systems. While these systems solve several major problems with the straightforward PoS implementations, they are not yet widespread, making it difficult to evaluate their security. Nevertheless, delegated PoS solves the "nothing at stake" problem and prevents short range attacks on the system.

Appendix A Distribution of Block Generation Time

Suppose all valid blocks in a blockchain protocol need to satisfy a condition

$$U \leqslant \theta \leqslant 1, \tag{8}$$

where $U \sim [0, 1]$ is a uniformly distributed random variable generated by hashing certain data and normalizing the obtained value. Due to the properties of hashing functions, both proof of work (1) and proof of stake (2) are particular cases of (8):

- in case of PoW, $\theta = 1/D$
- in case of PoS, $\theta = \text{bal}(A)/D$, with $\text{bal}(A)$ denoting user's stake in the system.

To generate a block, a user needs to find data resulting in U satisfying (8). Let N denote the number of data combinations a user needs to evaluate before finding a valid block. In proof of work, the search space is large, so the user can iterate through r combinations per second, where r is determined by user's mining equipment and is theoretically unlimited. In the case of proof of stake (2), the search space is small, so we can assume $r = 1$. The time T it takes for the user to find a valid block is related to N: $T = N/r$. Consider the cumulative probability distribution

$$P\{T \leqslant t\} = P\{N \leqslant rt\} = 1 - P\{N > rt\} = 1 - (1-\theta)^{rt} = 1 - \exp(\log(1-\theta)rt).$$

Assuming $\theta \ll 1$ (as it is usually the case),

$$\log(1-\theta) \approx -\theta; \quad P\{T \leqslant t\} \approx 1 - \exp(-\theta rt).$$

Thus, T is distributed exponentially with rate θr. In case of PoW, this rate is equal to r/D, and in case of PoS it equals $\mathrm{bal}(A)/D$.

Appendix B Probability Distribution of Block Minters in Nxt

As was mentioned in Section 2.2, Nxt is described by a probability model fundamentally different from other cryptocurrencies. One can intuitively see this from PoS (4) used in Nxt: it uses the time parameter in a different way than proof of work (1) or generic proof of stake (2). The analysis of (4) performed in the Nxt white paper [14] is not entirely convincing, so we decided to perform our own research on the subject.

Suppose there are n Nxt users with relative balances

$$b_1 \geqslant b_2 \geqslant \ldots \geqslant b_n > 0, \quad \sum_{i=1}^{n} b_i = 1.$$

Proof of stake (4) assigns each user a uniformly distributed variable $U_i \sim [0, 1/b_i]$. A user mints the next block if the variable assigned to him is less than all other variables. Thus, the probability to discover a block for i-th user is

$$p_i = P\{U_i = \min_j U_j\}.$$

By integrating over the probability space, we obtain

$$p_i = \prod_{j=1}^{n} \frac{1}{b_j} \int_0^{1/b_i} dx_i \underbrace{\int_{x_i}^{1/b_1} dx_1 \int_{x_i}^{1/b_2} dx_2 \ldots \int_{x_i}^{1/b_n} dx_n}_{n-1 \text{ integrals over } x_j, j \neq i}.$$

After simple transformations,

$$p_i = b_i \int_0^{1/b_1} \prod_{j \neq i} (1 - b_j x)\, dx,$$

or, after expanding the product,

$$p_i = b_i \sum_{j=0}^{n-1} \frac{(-1)^j B_j^{(i)}}{(j+1) b_1^{j+1}}, \tag{9}$$

where $B_j^{(i)}$ is a sum over products of j-tuples from balances b that don't include b_i. (For example, if $n = 4$,

$$\begin{aligned} B_1^{(0)} &= 1; \\ B_1^{(1)} &= b_2 + b_3 + b_4; \\ B_1^{(2)} &= b_2 b_3 + b_2 b_4 + b_3 b_4; \\ B_1^{(3)} &= b_2 b_3 b_4, \end{aligned}$$

and so on.) Expression (9) is difficult to analyze in general; we consider several important cases.

Equal stakes. Assume $b_i = 1/n$. In this case,

$$p_i = \frac{1}{n} \int_0^n \Big(1 - \frac{x}{n}\Big)^{n-1} dx = \frac{1}{n}\Big(1 - \frac{x}{n}\Big)^n \Big|_n^0 = \frac{1}{n},$$

that is, probability distribution is fair.

One big stake. Assume

$$b_1 = a; \quad \forall i = 2, \ldots, n \quad b_i = b = \frac{1-a}{n-1}.$$

i.e., there is a single user with a big stake a, and many users with small equal stakes b.

$$p_1 = a \int_0^{1/a} (1 - bx)^{n-1} dx = a \frac{1 - (1 - b/a)^n}{nb}.$$

Accordingly, the difference between the actual and the fair probabilities to mint a block for the user with a big stake is

$$p_1 - a = \frac{a}{nb}(1 - nb - (1 - b/a)^n).$$

Recall that b can be expressed using a and n:

$$p_1 - a = \frac{n-1}{n} \frac{a}{1-a} \left[1 - \frac{n(1-a)}{n-1} - \Big(1 - \frac{1-a}{a(n-1)}\Big)^n\right].$$

When there are many users,

$$\lim_{n\to\infty} p_1 = a + \frac{a}{1-a}\big[a - e^{(a-1)/a}\big].$$

This means a user with a single big stake is at an advantage: his percentage of blocks is strictly higher than his stake in the system (see Fig. 2). Therefore, Nxt is not a fair system.

To make Nxt fair, the Nxt white paper [14] proposes a simple modification based on the following observation: if $U \sim [0, 1]$, then $-\log U/b$ is an exponentially distributed random variable with rate b. Thus, if we change proof of stake (4) to

$$\log M - \log \mathrm{hash}(\mathrm{hash}(B_{prev}), A) \leqslant t\, \mathrm{bal}(A)/D,$$

the probabilities to mint a block would be distributed fairly. However, the proposal remains unimplemented.

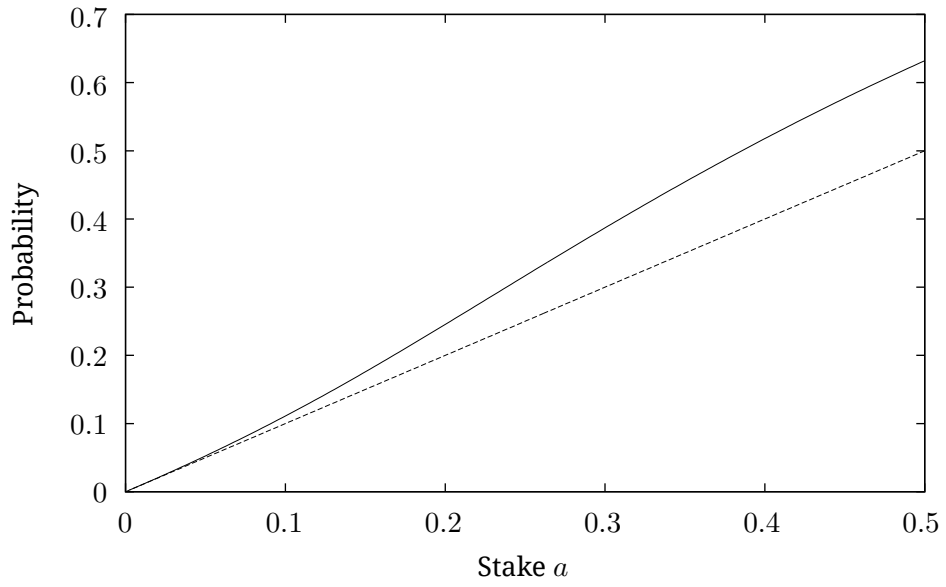

Figure 2: Probability to mint a block in Nxt depending on size of a stake

References

[1] Thin client security. In: Bitcoin Wiki
URL: https://en.bitcoin.it/wiki/Thin_Client_Security

[2] *Seth Gilbert, Nancy Lynch* (2002). Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. In: ACM SIGACT News, Vol. 33 (2), pp. 51–59.

[3] *Werner Vogels* (2008). Eventually consistent - revisited
URL: http://www.allthingsdistributed.com/2008/12/eventually_consistent.html

[4] Merkle tree. In: English Wikipedia
URL: https://en.wikipedia.org/wiki/Merkle_tree

[5] SHA-2. In: English Wikipedia
URL: https://en.wikipedia.org/wiki/SHA-2

[6] *Ittay Eyal, Emil Gün Sirer* (2013). Majority is not enough: Bitcoin mining is vulnerable
URL: http://arxiv.org/pdf/1311.0243v5

[7] Sybil attack. In: English Wikipedia
URL: https://en.wikipedia.org/wiki/Sybil_attack

[8] *Vitalik Buterin* (2014). Proof of stake: How I learned to love weak subjectivity. In: Ethereum Blog
URL: https://blog.ethereum.org/2014/11/25/proof-stake-learned-love-weak-subjectivity/

[9] Crypto-currency market capitalizations
URL: http://coinmarketcap.com/

[10] *Sunny King, Scott Nadal* (2012). PPCoin: peer-to-peer crypto-currency with proof-of-stake
URL: http://www.peercoin.net/assets/paper/peercoin-paper.pdf

[11] Transaction fees. In: Bitcoin Wiki
URL: https://en.bitcoin.it/wiki/Transaction_fees

[12] Checkpoint lock-in. In: Bitcoin Wiki
URL: https://en.bitcoin.it/wiki/Checkpoint_Lockin

[13] Nxt whitepaper. In: Nxt Wiki
URL: https://wiki.nxtcrypto.org/wiki/Whitepaper:Nxt

[14] *mthcl* (2014). The math of Nxt forging
URL: http://www.docdroid.net/e29h/forging0-5-1.pdf.html

[15] Proof of work. In: Novacoin Project Wiki
URL: https://github.com/novacoin-project/novacoin/wiki/Proof-of-work

[16] Proof of stake. In: Novacoin Project Wiki
URL: https://github.com/novacoin-project/novacoin/wiki/Proof-of-stake

[17] *Pavel Vasin* (2014). BlackCoin's proof-of-stake protocol v2
URL: http://blackcoin.co/blackcoin-pos-protocol-v2-whitepaper.pdf

[18] *Daniel Larimer, Charles Hoskinson, Stan Larimer* (2014). BitShares: a peer-to-peer polymorphic digital asset exchange
URL: http://scribd.com/doc/173481633/BitShares-White-Paper

[19] *Iddo Bentov, Ariel Gabizon, Alex Mizrahi* (2013). Cryptocurrencies without proof of work
URL: http://www.cs.technion.ac.il/~idddo/CoA.pdf

[20] *Vitalik Buterin* (2014). On stake. In: Ethereum Blog
URL: https://blog.ethereum.org/2014/07/05/stake/

[21] *Meni Rosenfeld* (2012). Analysis of hashrate-based double-spending
URL: https://bitcoil.co.il/Doublespend.pdf

[22] *Ghassan O. Karame, Elli Androulaki, Srdjan Capkun* (2012). Two bitcoins at the price of one? Double-spending attacks on fast payments in Bitcoin
URL: http://eprint.iacr.org/2012/248.pdf

[23] July 2015 flood attack. In: Bitcoin Wiki
URL: https://en.bitcoin.it/wiki/July_2015_flood_attack

[24] *Matt Springer* (2014). Is Bitcoin currently experiencing a selfish miner attack? In: ScienceBlogs
URL: http://scienceblogs.com/builtonfacts/2014/01/11/is-bitcoin-currently-experiencing-a-selfish-miner-attack/

[25] *Ed Felten* (2013). Game theory and Bitcoin. In: Freedom to Tinker
URL: https://freedom-to-tinker.com/blog/felten/game-theory-and-bitcoin/

[26] Bitcoin days destroyed. Blockchain.info
URL: https://blockchain.info/charts/bitcoin-days-destroyed

[27] Peercoin charts live. blockr.io (retrieved on Sep 07, 2015)
URL: http://ppc.blockr.io/charts

[28] Peercoin block explorer. blockr.io (retrieved on Sep 07, 2015)
URL: http://ppc.blockr.io/

[29] *Daniel Larimer* (2013). Transactions as proof-of-stake
URL: http://bravenewcoin.com/assets/Uploads/TransactionsAsProofOfStake10.pdf

[30] *Jae Kwon* (2015). Tendermint: consensus without mining
URL: http://tendermint.com/docs/tendermint.pdf

[31] *Vitalik Buterin* (2014). Slasher: a punitive proof-of-stake algorithm. In: Ethereum Blog
URL: https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/

[32] *Vlad Zamfir* (2015). Introducing Casper "the friendly ghost". In: Ethereum Blog
URL: https://blog.ethereum.org/2015/08/01/introducing-casper-friendly-ghost/