A Large-Scale Proof-of-Stake Blockchain in the Open Setting*

(or, How to Mimic Nakamoto's Design via Proof-of-Stake)

Lei Fan† Jonathan Katz‡ Hong-Sheng Zhou§

Abstract

Bitcoin and blockchain technologies have proven to be a phenomenal success. The underlying techniques hold huge promise to change the future of financial transactions, and eventually the way people and companies compute, collaborate and interact. At the same time, the current Bitcoin-like proof-of-work based blockchain systems are facing many challenges. For example, a huge amount of energy/electricity is needed to maintain the Bitcoin blockchain.

We propose a new approach to constructing energy-efficient blockchain protocols. More concretely, we develop proof-of-stake based, large-scale blockchain protocols in the open network setting. Our contributions are as follows:

- We for the first time formally investigate "greedy" strategies for proof-of-stake based protocols. In a proof-of-work based system, players follow a "simple" strategy by extending only the longest chain. However, in a proof-of-stake based system, players may make attempts to extend a set of chains, and we call such strategies, *greedy*. We have two major findings.

  Players with greedy strategies can extend the blockchain faster. Our first finding is an interesting upper bound of extending blockchain: greedy players can generate blockchain at most 2.7 times faster than playing the simple strategy.

  Our second finding is a design of a novel greedy strategy called "D-distance-greedy" strategy, which enables us to construct a class of secure proof-of-stake blockchain protocols, against an *arbitrary* adversary.
- We for the first time provide a strategy to allow players to register to the system during the protocol execution.
- Our protocols are the first natural mimic of Bitcoin blockchain but via proof-of-stake mechanism. Our design is very simple, using only standard hash functions and unique digital signatures, which makes our design very appealing in practice.
- Finally, as a side contribution, we for the first time identify a new security property called *chain soundness* for proof-of-stake based protocols, which captures the intuition of ensuring new players can join the protocol execution securely.

*All results in this paper have been submitted and presented in public. In particular, the presentation slides and video can be available online at Stanford Blockchain Conference https://cyber.stanford.edu/sbc19

†Shanghai Jiao Tong University

‡George Mason University

§Virginia Commonwealth University

Contents

1 Introduction

Bitcoin and proof-of-work mechanism. Cryptocurrencies like Bitcoin [43] have proven to be a phenomenal success. The system was designed and implemented by an unknown researcher, under the name Satoshi Nakamoto nine years ago. The underlying techniques hold huge promise to change the future of financial transactions, and eventually the way people and companies compute, collaborate, and interact. At the heart of these cryptocurrency systems are distributed *blockchain* protocols, jointly executed by a large-scale peer-to-peer network of nodes called *miners* via the so-called *proof-of-work* mechanism [26, 3]. These blockchain protocols implement a highly trustworthy, append-only, and always-available public ledger, which can then be used to implement a global payment system (as in Bitcoin) or a global computer (as in Ethereum [13]). Nakamoto's design has unique features:

- **open network:** the Bitcoin blockchain protocol can be executed in an *open* network setting in which all miners are allowed to join/leave the protocol execution at any moment they want; miners are encouraged/incentivized to invest certain amount of computing power to join the effort of maintaining the blockchain;
- **large-scale network:** the protocol has very low communication complexity and *can scale* to a large network of nodes.

From proof-of-work to proof-of-stake. Bitcoin system has "wasted" a huge amount of computing resources over the past several years. It is definitely desirable to utilize alternative resources such as *coins (also called stakes)* to secure a blockchain. If successful, the new system will be "green" in the sense that it does not require a huge amount of computing power to back up its security. Attempts have been made: proof-of-stake (PoS) mechanisms have been widely discussed in the cryptocurrency community (e.g., [2, 39, 55, 8]). In a nutshell, in a proof-of-stake based blockchain protocol, players are expected to prove ownership of a certain number of coins/stakes; only the players that can provide such a proof are allowed to participate in the process of maintaining the blockchain.

Blockchain protocols with provable security. In the past years, the security of Bitcoin-like protocols has been investigated. For example, Garay et al. [29] took the *provable security* approach and investigated Nakamoto's blockchain in a cryptographic framework (please also see [47], Nakamoto's protocol can be showed to achieve chain soundness property.

1.1 Our construction

1.1.1 Warm-up: Nakamoto's design and proof-of-work (PoW) based core-chain

We first briefly review Nakamoto's design ideas [43]. The blockchain in Bitcoin consists of a chain of ordered blocks $B_1, B_2, B_3, \ldots$, and PoW-players (i.e., miners) in each round (or time slot) attempt to extend the blockchain with a new block by solving proof-of-work puzzles [26, 3]. The *puzzle* for each miner is defined by (1) the "context", i.e., the latest block in the longest blockchain in the miner's view, and (2) the "payload", i.e., the set of valid transactions to be included in the new block; and a valid *puzzle solution* to the problem is defined by a hash inequality. More concretely, assume the longest blockchain for a miner consists of $B_1, B_2, \ldots, B_i$, and B_i is the latest block. The miner now attempts to find a valid puzzle solution *nonce* which can satisfy the following hash inequality: $\mathsf{H}(\mathsf{hash}(B_i), payload, nonce) < \mathtt{T}$, where $\mathsf{H}(\cdot)$ and $\mathsf{hash}(\cdot)$ are two hash functions, $payload$ denotes the set of valid transactions to be included in the new block, and $\mathtt{T}$ denotes the target of proof-of-work puzzle difficulty (which specifies how difficult to identify a puzzle solution by making a hash query attempt). In the case that a new valid solution, *nonce*, is identified, such a solution can be used for defining a new valid block B_{i+1} as follows: $B_{i+1} := \langle h_i, payload, nonce\rangle$, where $h_i := \mathsf{hash}(B_i)$. Then the new block B_{i+1} will be revealed by the miner, and broadcasted to the network and then accepted by the remaining miners in the system. (The above description is oversimplified.)

We may consider a further simplified version of the above blockchain protocol, called *Bitcoin core-chain protocol*. In the core-chain protocol, the payload will be ignored, and now puzzle is based on $\mathsf{H}(\mathsf{hash}(B_i), nonce) < \mathtt{T}$, and the new block B_{i+1} is defined as $B_{i+1} := \langle h_i, nonce \rangle$. (We often call the blocks in a blockchain protocol, *blocks*, while the blocks in a core-chain protocol, *block-cores*.)

We note that, mimicking Nakamoto's footprint via proof-of-stake mechanism is non-trivial, and we have to address many technical challenges. To make our presentation more accessible, we start with the basic version of our proof-of-stake based core-chain protocol, Π^{core}, and then present the improved versions $\Pi^{\mathrm{core}\circ}$ and $\Pi^{\mathrm{core}\bullet}$ which deal with adversarial greedy strategies and adaptive stake registrations, respectively. These eventually allow us to develop a full-fledged proof-of-stake blockchain protocol Π^{main}. Next, we illustrate our key ideas step by step.

1.1.2 Step 1, Π^{core}: Proof-of-stake (PoS) based core-chain, the basic version

We intend to mimic Nakamoto's design. Our proof-of-stake (PoS) based protocol will be maintained by PoS-players (i.e., stakeholders); We first consider the basic strategy that all players attempt to extend the longest chain with a new block. Similar to that in the PoW-based protocol, a winning PoS-player is chosen with some probability but using a different hash inequality. More concretely, assume the longest core-chain for a PoS-player consists of the following ordered block-cores, $B_1, B_2, \ldots, B_i$; let $\mathtt{round}$ denote the current time step (or round number); consider a *unique* digital signature scheme [40] $(\mathsf{uKeyGen}, \mathsf{uKeyVer}, \mathsf{uSign}, \mathsf{uVerify})$, and assume the PoS-player holds the signing-verification key pair $(\mathrm{SK}, \mathrm{PK})$. If the PoS-player is chosen, then the following hash inequality holds: $\mathsf{H}(\mathsf{hash}(B_i), \mathtt{round}, \mathrm{PK}, \sigma) < \mathtt{T}$, where $\sigma := \mathsf{uSign}_{\mathrm{SK}}(h_i, \mathtt{round})$, and $h_i := \mathsf{hash}(B_i)$. The new block-core B_{i+1} is defined as $B_{i+1} := \langle h_i, \mathtt{round}, \mathrm{PK}, \sigma \rangle$.

Our design is very similar to Nakamoto's: the context here consists of the latest block-core in the longest core-chain, and the payload in the core-chain is empty; the puzzle solution consists of the current time, a PoS-player's verification key and his signature of the context. When the adversary (1) follows the basic strategy, i.e., extending the single longest chain, and (2) has all stakes registered without being aware of the state of protocol execution, then our protocol can be viewed as a proof-of-stake analogy of Nakamoto's, and the security properties i.e., chain growth, chain quality, and common prefix (cf [29, 47]) can be demonstrated.

Achieving chain soundness. Chain soundness property ensures that new players can join the system securely. Note that, a new player is not aware of the current state of the protocol execution (because the player did not participate in the protocol execution). In our design, each (new or existing) honest player takes the longest chain as the best chain. We can show that the adversary now cannot generate a longer chain privately to "confuse" the new players (otherwise the adversary can also confuse the existing honest players, which will violate the common prefix property). See Section 3 for more details.

Theorem 1.1 (informal)**.** *Consider core-chain protocol Π^{core} where all players follow the basic strategy of extending the longest chain; in addition, all players have their stakes registered without being aware of the state of the protocol execution. If more than 51% stakes are honest, then the protocol Π^{core} can achieve chain growth, chain quality, common prefix and chain soundness properties.*

1.1.3 Step 2, $\Pi^{\mathrm{core}\circ}$: Securing the core-chain against an arbitrary adversary

For simplicity, in the protocol Π^{core} above, we focus on the setting that all players follow the basic strategy to extend the core-chain. That is, each player will make attempts to extend the *single* best chain (i.e., the longest chain) in his/her local view. We note that, this basic strategy has been widely adopted in the proof-of-work setting; there, extending a chain is expensive in the sense that it requires significant amount of computing power; it will be extremely costly to extend multiple chains simultaneously. However, in the proof-of-stake setting, it is very cheap to extend a chain. The proof-of-stake players may follow a *full-greedy* strategy to extend all chains, expecting to obtain additional advantage for extending the best chain. This introduces difficulty for security analysis. Furthermore, it is extremely challenging to defend against an adversary who may play an

arbitrary strategy in the protocol execution. For example, instead of playing the full-greedy strategy to extend *all* chains, the adversary may on purpose extend only the weaker/shorter chains; this may create a scenario that, two (or multiple) chains may take turns to be the best chain, and the common prefix property may not be achieved.

Understanding the power of greedy strategies To address the above difficulty, our key observation is that, *honest players should also play a carefully designed greedy strategy*. We have two major contributions for understanding the greedy strategies. First, we focus on full-greedy strategy, and demonstrate a very interesting upper bound: the full-greedy strategy will allow a PoS player to improve his/her chance of extending chains with a factor at most e where $e \approx 2.718$[1]. This upper bound allows us to develop secure core-chain protocols against full-greedy adversaries.

Defending against an arbitrary adversary Our second major contribution is, we for the first time shed light on addressing arbitrary adversarial attacks. As discussed before, an arbitrary adversary could break the common prefix property if the blockchain protocol is not carefully designed. In supplementary material C, we describe a protocol in which honest players follow a "height-greedy" strategy. It turns out that, this protocol is secure against a restricted adversary who may play fully greedy strategy. However, this protocol becomes insecure in the presence of an arbitrary adversary.

To defend against this powerful adversary, a carefully designed greedy strategy is needed. We introduce a simple but novel, "D-distance-greedy" strategy. A D-distance-greedy player will make attempts to extend a *set of chains*; these chains have the following unique property: after removing the last D blocks, those chains are prefix of the best chain. Note that, by following the D-distance-greedy strategy, the honest miners extend the chain set that share the same prefix. The D-distance-greedy strategy can have common prefix property enabled, which can effectively defend against an arbitrary adversary. If majority of stakes are honest , then the protocol can achieve the security properties. We can have the following theorem. See Section 4 (and also Supplementary material C) for more details.

Theorem 1.2 (informal)**.** *Consider core-chain protocol* $\Pi^{\text{core}\circ}$ *where honest players follow the* 2*-distance-greedy strategy (resp.,* 0*-distance-greedy strategy) while adversarial players follow an arbitrary strategy; in addition, all players have their stakes registered without being aware of the state of the protocol execution. If more than* 63% *stakes (resp.,* 73% *stakes) are honest, then the protocol* $\Pi^{\text{core}\circ}$ *can achieve chain growth, chain quality, common prefix and chain soundness properties.*

1.1.4 Step 3, $\Pi^{\text{core}\bullet}$: Securing the core-chain further, against adaptive stake registration

The protocol $\Pi^{\text{core}\circ}$ above is expected to be executed in a less realistic setting where all players must have their stakes registered without being aware of the state of the protocol execution. Recall that the hash inequality $\mathsf{H}(\mathit{context}, \mathit{solution}) < \mathtt{T}$ is used in the process of extending the chains. In reality, an adversary may have a stake registered *based on the state* of the protocol execution. More concretely, the adversary can play a "rejection re-sampling" strategy to generate keys, and then have his/her stake registered adaptively: the adversary first runs the key generation algorithm to obtain a key-pair $(\mathtt{PK}, \mathtt{SK})$, and then checks if the corresponding $(\mathtt{PK}, \sigma)$ is a valid solution to the hash inequality; if not, the adversary re-samples a new key-pair. This adaptive stake registration strategy enables the adversary (to be selected) to extend the chains with much higher probability. To address this concern, we introduce new ideas to our protocol design: to extend the chains with new blocks, a player must have his/her stake registered a specified number of rounds earlier. See Section 5 for more details.

Theorem 1.3 (informal)**.** *Consider core-chain protocol* $\Pi^{\text{core}\bullet}$ *where honest players follow the* 2*-distance-greedy strategy (resp.,* 0*-distance-greedy strategy) while adversarial players follow an arbitrary strategy; If more than* 63% *stakes (resp.,* 73% *stakes) are honest, then the protocol* $\Pi^{\text{core}\bullet}$ *can achieve chain growth, chain quality, common prefix and chain soundness properties.*

[1]Note, e is Euler's number, or the base of the natural logarithm.

1.1.5 Step 4, Π^{main}: From the core-chain to a blockchain

In this step, we will "upgrade" the core-chain protocol to a regular blockchain protocol so that payload (e.g., the transactions) can be included. Intuitively, the core-chain can be viewed as a (biased) randomness beacon; we can use the beacon to select a PoS-player to generate a new block so that the blockchain can be extended. More concretely, once a new block-core B_{i+1} is generated by a PoS-player (in the core-chain protocol), then the PoS-player is selected for generating the new block $\tilde{B}_{i+1}$, in the following format: $\tilde{B}_{i+1} = \langle \mathsf{hash}(\tilde{B}_i), B_{i+1}, \tilde{X}_{i+1}, \tilde{\text{PK}}, \tilde{\sigma} \rangle$, where $\tilde{\sigma} \leftarrow \mathsf{Sign}_{\tilde{\text{SK}}}(\tilde{h}_i, B_{i+1}, \tilde{X}_i)$, $\tilde{X}_{i+1}$ is payload and $\tilde{h}_i := \mathsf{hash}(\tilde{B}_i)$, and $B_{i+1} := \langle h_i, \mathtt{round}, \text{PK}, \sigma \rangle$. Here the PoS-player holds two pairs of keys, i.e., (SK, PK) of the unique signature scheme $(\mathsf{uKeyGen}, \mathsf{uSign}, \mathsf{uVerify})$, and $(\tilde{\text{SK}}, \tilde{\text{PK}})$ of a regular[2] digital signature scheme $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. Now we attach each block to the core-chain via the corresponding block-core; we can reduce the security of the blockchain protocol to the security of the core-chain protocol. Please also see Figure 1 for a pictorial illustration.

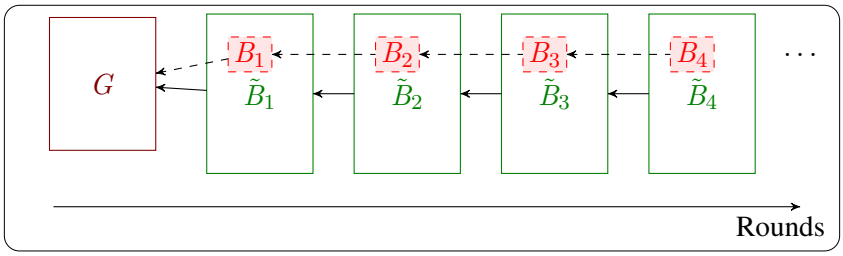

Figure 1: Blockchain structure

Blockchain $\tilde{\mathcal{C}}$ consists of initial setup information (i.e., genesis block) G, and then an ordered sequence of blocks $\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \ldots$. Here, each block $\tilde{B}_i$ consists of a block-core B_i and additional information. A core-chain $\mathcal{C}$ consists of the initial setup information G and the ordered sequence of block-cores $B_1, B_2, B_3, \ldots$.

Finally, we have the theorem; See Section 6 for more details.

Theorem 1.4 (informal)**.** *Consider core-chain protocol Π^{main} where honest players follow the 2-distance-greedy strategy (resp., 0-distance-greedy strategy) while adversarial players follow an arbitrary strategy. If more than 63% stakes (resp., 73% stakes) are honest, then the protocol Π^{main} can achieve chain growth, chain quality, common prefix and chain soundness properties.*

1.1.6 Discussions and extensions

Our design can tolerate many well-known rational attacks such as nothing at stake attacks, selfish mining attacks and more; see Section 7 for more details. Our design can also be extended in multiple directions, including enabling adaptive difficulty adjustment and supporting light clients as in the original Bitcoin [43], and incentivizing the system via better strategies (e.g., [49]), managing transactions in blockchain more effectively (e.g., [51]), and more; see Section 8 for details.

1.2 Related work

1.2.1 Cryptocurrency and proof-of-work

Anonymous digital currency was introduced by Chaum [18] in the early 1980s. The first decentralized currency system, Bitcoin [43], was launched about 30 years later, by incentivizing a set of players to solve moderately-hard cryptographic puzzles (also called proof-of-work puzzles [26, 3]). After that, many cryptocurrency sys-

[2] To achieve adaptive security, this regular digital signature scheme will be replaced by a forward-secure digital signature scheme [6]. We remark that, the core-chain protocols, i.e., the protocols without having payload included, in previous steps are adaptively secure.

tems were created based on proof-of-work puzzles (e.g., LiteCoin [1], Ethereum [13, 56]). Please refer to the online course [44] and the survey [12].

The security of Bitcoin system has been analyzed in the rational setting, e.g., [28, 27, 45, 34, 52, 53], and also in the cryptographic setting, e.g., [29, 47, 54, 35, 36, 30, 5]. Several important cryptographic properties, *common prefix* [29, 47], *chain quality* [29], and *chain growth* [35], have been considered for proof-of-work protocols.

1.2.2 Proof-of-stake

Using coins/stakes to construct cryptocurrency has been intensively considered. Since the inception of the idea in an online forum [10], several proof-of-stake proposals have been introduced and/or implemented (e.g., [2, 39, 55, 14, 8]). We remark that these proposals are *ad hoc* without formal security, and it is not clear how to formally prove the security of these proposals. Very recently, several provably secure proof-of-stake based blockchain proposals (e.g., [20, 37, 23, 50]) have been developed. We provide more details below.

Bitcoin-like projects. The Sleepy protocol [50], is very efficient but it is designed for the closed setting which means new players are not allowed to join the system during the execution. In follow-up work, Snow White [23], new players are allowed to join the system; but these new players need to contact a group of honest majority players.

Ouroboros Praos [24] is concurrent and independent work of ours. We note that the protocol in Ouroboros Praos cannot allow new players to join the protocol execution securely, and some trusted advice must be provided. In their very recent follow-up work [4], a fix to Ouroboros Praos is provided, with the same goal of allowing new players to join the protocol execution securely as in this paper. In [24, 4], the protocol execution consists of multiple epoch, and in each epoch multiple blocks will be generated. Note that, for each player, he can predict all blocks he will generate in the current epoch. This means, their protocols are more vulnerable to selfish mining attacks (than ours). In Praos/Genesis, protocol execution is divided into epochs, and a "chunk" of $O(\kappa)$ blocks are generated in each epoch; our protocol, as in Bitcoin, only 1 block is generated in each epoch. Our protocol can achieve faster confirmation than Praos/Genesis; For the same reason, new players can join in the execution more flexibly, at any block, instead of at any chunk as in Genesis.

We remark that, Bitcoin protocol follows the "per block" approach. Different from the "per epoch" approaches [23, 24, 4], above, ours is the first "per block" solution with provable security. Note that, in "per epoch" proof-of-stake protocols, the randomness used for choosing the current block generator has been published an epoch, i.e., many blocks ago, while in "per epoch" proof-of-stake here, the randomness used for choosing the current block generator is only published one block ago. In some sense, our scheme can achieve better block unpredictability.

BFT-like projects. Algorand [20, 31, 19] presents an interesting alternative solution. We note that, Algorand protocol is not Bitcoin-like: in Algorand, an improved but *multi-round* Byzantine Agreement (BA) sub-protocol is used so that multiple players jointly generate a block; in contrast, in Bitcoin-like protocols (e.g., [2, 23, 24] and our protocols), no any form of BA will be involved and a single player will be chosen to generate a block. We notice that Algorand protocol has been simulated / carried out in certain idealized network environment [31]; however it is not clear how practical Algorand protocol can be when it is executed in a large-scale (i.e., 10,000 nodes) *real-world* open network. On the other hand, Bitcoin-like protocols are more suitable to be executed in the real world network environment (in which network delay is non-trivial).

1.2.3 Additional related work

Combining proof-of-work and proof-of-stake. The idea of combining proof-of-work and proof-of-stake has been studied in [38, 22, 9, 25, 21]. Duong et al [25] provided a provably secure and scalable blockchain

via proof-of-work/proof-of-stake in the open setting.

Additional alternative mechanisms. Alternative consensus techniques via different resources have been considered. For example, the physical storage resource is used in [46, 41]. A hybrid proposal of utilizing both computing and space resources, called proof-of-space-time was introduced in [42]. Recently, blockchain protocols via trusted hardware have also been proposed [33, 57].

1.3 Organization

The remaining of the paper is organized as follows. In Section 2, we introduce an analysis framework for proof-of-stake protocols.

In Section 3, we construct the basic version of our proof-of-stake based core-chain protocol, and then provide the security analysis. In Section A and in Section B, the security analysis details, and some instantiation of the building block are provided.

In Section 4, we investigate greedy strategies, and develop a modified proof-of-stake based core-chain protocol to defend against an arbitrary adversary. Then we provide more details for the analysis of the modified core-chain protocol against an arbitrary adversary. In Section C, we develop modified proof-of-stake based core-chain protocols to defend against full-greedy adversaries

In Section 5, we improve the modified core-chain protocol further so that it can be executed in the real-world setting where the players are allowed to register their key-pairs adaptively.

In Section 6, we upgrade the core-chain protocol to a full-fledged blockchain protocol.

Finally in Section 7, we provide discussions on rational attacks on our design, and in Section 8, many extensions are provided. In Section D, we provide the basic functionalities and primitives for our constructions.

2 Model

In order to study the security of Bitcoin-like proof-of-work based protocols, Garay et al. [29] proposed a cryptographic framework and showed that (a simplified version of) Bitcoin protocol can achieve several important security properties. Then, Pass et al. [48] strengthened Garay et al.'s analysis by considering a more realistic communication network (i.e., partially synchronous network) in which messages from honest players can be delayed with a bounded number of rounds. Below we define a framework for analyzing proof-of-stake based blockchain protocols. We note that we take many formulation ideas from the previous framework [29, 48].

2.1 Blockchain protocol executions

The execution of proof-of-stake blockchain protocol. Following Canetti's formulation of the "real world" executions [15, 16], we present an abstract model for proof-of-stake (PoS) blockchain protocol Π in the $\{\mathcal{F}_{\mathsf{Setup}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid model, where $\mathcal{F}_{\mathsf{NET}}$ denotes the partially synchronous network communication functionality (see Supplemental material D.1), and $\mathcal{F}_{\mathsf{Setup}}$ denotes the setup functionality (which will be explained soon), for the PoS-players. We consider the execution of blockchain protocol Π that is directed by an environment $\mathcal{Z}(1^{\kappa})$ (where κ is a security parameter), which activates a set $\mathcal{P}$ of PoS-players. The environment $\mathcal{Z}$ can "manage" protocol players through an adversary $\mathcal{A}$ that can dynamically corrupt honest parties. More concretely, the $\{\mathcal{F}_{\mathsf{Setup}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid execution proceeds as follows. Each party in the execution is initialized with an initial state including all initial public information e.g., a genesis block. The environment $\mathcal{Z}$ first actives the adversary $\mathcal{A}$ and provides instructions for the adversary. The execution proceeds in rounds, and in each round, a protocol party could be activated by the environment or the functionalities.

In each round, each PoS-player $P \in \mathcal{P}$, with a local state $state$ (note that $state$ originally includes the initial state), proceeds as follows. When PoS-player P is activated by the environment $\mathcal{Z}$ by $(\textsc{Input-Stake}, P, x)$

where x is the input from the environment, and potentially P receives subroutine output message $(\text{MESSAGE}, P', m)$ for any $P' \in \mathcal{P}$, from $\mathcal{F}_{\mathsf{NET}}$, the PoS-player P interacts with the functionality $\mathcal{F}_{\mathsf{Setup}}$ and receives some output y from $\mathcal{F}_{\mathsf{Setup}}$.

Next, the PoS-player P executes the protocol Π on input its local state $state$, the value y received from the functionality $\mathcal{F}_{\mathsf{Setup}}$, an input from the environment x, and the message m received from the functionality $\mathcal{F}_{\mathsf{NET}}$; and then P obtains an updated local state $state$ and an outgoing message m', i.e.,$\{state, m'\} \leftarrow \Pi(state, x, y, m)$. After that, P sends $(\text{BROADCAST}, m')$ to $\mathcal{F}_{\mathsf{NET}}$ and then returns $(\text{RETURN-STAKE}, P)$ to the environment $\mathcal{Z}$.

At any round r of the execution, $\mathcal{Z}$ can send message $(\text{CORRUPT}, P)$, where $P \in \mathcal{P}$, to adversary $\mathcal{A}$. Then $\mathcal{A}$ will have access to the party's local state and control P.

Let $\mathsf{EXEC}^{\mathcal{F}_{\mathsf{Setup}}, \mathcal{F}_{\mathsf{NET}}}_{\Pi, \mathcal{A}, \mathcal{Z}}$ be a random variable denoting the joint $\mathtt{VIEW}$ of all parties (i.e., all their inputs, random coins and messages received) in the above $\{\mathcal{F}_{\mathsf{Setup}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid execution; note that this joint view fully determines the execution. Whenever $\mathcal{F}_{\mathsf{Setup}}, \mathcal{F}_{\mathsf{NET}}$ are clear from context we often write $\mathsf{EXEC}_{\Pi, \mathcal{A}, \mathcal{Z}}$.

Remark 2.1. *For simplicity, we focus on the idealized "flat" model where all PoS-players have the same number of stakes. Note that, in the reality, each different honest PoS-player may have a different amount of stake. In addition for simplicity, we focus on the idealized "static difficulty" model where the number of PoS-players that who have stakes, is fixed during the course of the protocol execution. That means, if some new PoS-players join the system, then the same number of PoS-players will leave the system. In Supplemental material 8, we will discuss how to extend our main results in the idealized flat, static difficulty model to the more realistic non-flat, adaptive difficulty setting.*

Remark 2.2 (Player joining and leaving)**.** *Protocol players are allowed to join the protocol execution* $\mathsf{EXEC}^{\mathcal{F}_{\mathsf{Setup}}, \mathcal{F}_{\mathsf{NET}}}_{\Pi, \mathcal{A}, \mathcal{Z}}$. *More explicitly, PoS-players, during their first interactions with the setup functionality* $\mathcal{F}_{\mathsf{Setup}}$*, can have themselves registered.*

However, it is very subtle to have PoS-players unregistered *when they decide to leave the protocol execution. In the current version of our modeling, we assume that when (honest) PoS-players leave the protocol execution, they will* erase *their own local internal information. That means, the protocol execution is in the non-erasure model (except for these already unregistered players). For this reason, we currently* disabled *the* STAKE-UNREGISTER *command in the* $\mathcal{F}_{\mathsf{Setup}}$ *(e.g.,* $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ *in Section 5) but keep the* STAKE-REGISTER *command; we note that the* STAKE-UNREGISTER *comand can be added back if certain sophisticated mechanisms are introduced.*

2.2 Security properties

2.2.1 Blockchain basics

A *blockchain* $\mathcal{C}$ consists of a sequence of ℓ concatenated blocks $B_0 \| B_1 \| B_2 \| \cdots \| B_\ell$, where $\ell \geq 0$ and B_0 is the initial block (genesis block). We use $\mathsf{len}(\mathcal{C})$ to denote *blockchain length*, i.e., the number of blocks in blockchain $\mathcal{C}$; and here $\mathsf{len}(\mathcal{C}) = \ell$. We use sub blockchain (or subchain) for referring to segment of a chain; here for example, $\mathcal{C}[1, \ell]$ refers to an entire blockchain, whereas $\mathcal{C}[j, m]$, with $j \geq 1$ and $m \leq \ell$ would refer to a sub blockchain $B_j \| \cdots \| B_m$. We use $\mathcal{C}[i]$ to denote the i-th block B_i in blockchain $\mathcal{C}$. If blockchain $\mathcal{C}$ is a prefix of another blockchain $\mathcal{C}'$, we write $\mathcal{C} \preceq \mathcal{C}'$. If a chain $\mathcal{C}$ is truncated the last κ blocks, we write $\mathcal{C}[\neg\kappa]$.

Notation. For some $\mathcal{A}, \mathcal{Z}$, consider some $\mathtt{VIEW}$ in the support of $\mathsf{EXEC}^{(\Pi^V, \mathcal{C})}(\mathcal{A}, \mathcal{Z}, \kappa)$. We use the notation $|\mathtt{VIEW}|$ to denote the number of rounds in the execution, $\mathtt{VIEW}^r$ to denote the prefix of $\mathtt{VIEW}$ up until round r, $state_i(\mathtt{VIEW})$ denote the local state of player i in $\mathtt{VIEW}$, $\mathcal{C}_i(\mathtt{VIEW}) = \mathcal{C}(state_i(\mathtt{VIEW}))$ and $\mathcal{C}^r_i(\mathtt{VIEW}) = \mathcal{C}_i(\mathtt{VIEW}^r)$.

2.2.2 Chain growth, common prefix, and chain quality

Previously, several fundamental security properties for proof-of-work blockchain protocols have been defined: *common prefix property* [29, 48], *chain quality property* [29], and *chain growth property* [35]. Intuitively, the chain growth property states that the chains of honest players should grow linearly to the number of rounds. The common prefix property indicates the consistency of any two honest chains except the last κ blocks. The chain quality property, aims at indicating the number of honest blocks' contributions that are contained in a sufficiently long and continuous part of an honest chain. Specifically, for parameters $\ell \in \mathbb{N}$ and $\mu \in (0, 1)$, the ratio of honest input contributions in a continuous part of an honest chain has a lower bounded μ. We follow the same path to define the security properties for proof-of-stake blockchain protocols. The definitions for these properties are formally given as follows.

Definition 2.3 (Chain growth)**.** *Consider a blockchain protocol Π with a set $\mathcal{P}$ of players. The chain growth property with parameter $g \in \mathbb{R}$, states: for any honest player P' with local chain $\mathcal{C}'$ at round r', and honest player P'' with local chain $\mathcal{C}''$ at round r'', where $P', P'' \in \mathcal{P}$ and $r'' > r'$, in the execution $\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}}$, it holds that $\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') \geq g(r'' - r')$.*

Definition 2.4 (Common prefix)**.** *Consider a blockchain protocol Π with a set $\mathcal{P}$ of players. The common prefix property states the following: for any honest player P' adopting local chain $\mathcal{C}'$ at round r', and honest player P adopting local chain $\mathcal{C}$ at round r, in the execution $\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}}$, where $P', P \in \mathcal{P}$ and $r \leq r'$, it holds that $\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'$.*

Definition 2.5 (Chain quality)**.** *Consider a blockchain protocol Π with a set $\mathcal{P}$ of players. The chain quality property with parameters μ, ℓ, where $\mu \in \mathbb{R}$ and $\ell \in \mathbb{N}$, states: for any honest player $P \in \mathcal{P}$, with local chain $\mathcal{C}$ in round r, in $\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}}$, it holds, for large enough ℓ consecutive blocks of $\mathcal{C}$, the ratio of honest blocks is at least μ.*

2.2.3 New property: Chain soundness

We here introduce a new security property, *chain soundness*, which is critical for blockchain protocols in the open setting. A good protocol in the open network environment, should ensure honest new players to join the system securely. Intuitively, the protocol can help the new players to obtain a blockchain which is compatible with the local chain of an existing honest player in some recent rounds. While this property is not needed for protocols in the *closed* setting where new players are not allowed, it is important for blockchains in the open network environments. Without this security requirement, unsatisfactory protocols could be introduced to the system. The chain soundness property can be described as follows.

Definition 2.6 (Chain soundness)**.** *Consider a blockchain protocol Π with a set $\mathcal{P}$ of players. Consider a new player $P \in \mathcal{P}$ with best local chain $\mathcal{C}$ in round r, in $\mathsf{EXEC}_{\Pi,\mathcal{A},\mathcal{Z}}$. The* chain soundness *property states the following: for the new player P and any existing players P' with best local chain $\mathcal{C}'$ at round r, it holds that $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}$ and $\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'$.*

We remark that, the chain soundness property is different from the common prefix property above. In the latter, only existing players are considered. Several previous protocols (e.g., [37, 23]) can achieve common prefix but not chain soundness.

3 Proof-of-stake core-chain, the basic version

In this section, we describe the basic version of our design. Here, we consider a restricted adversary who (1) extends at most one chain in each time window, and (2) has its stakes registered independent of the state of protocol execution.

Each protocol player first has its stakes registered independent of the state of protocol execution. Then it extends the longest chain in its local view. More concretely, assume the longest core-chain for a PoS-player

consists of the following ordered block-cores, $B_1, B_2, \ldots, B_i$; let round denote the current time (or round number); consider a *unique* digital signature scheme $(\mathsf{uKeyGen}, \mathsf{uKeyVer}, \mathsf{uSign}, \mathsf{uVerify})$, and assume the PoS-player holds the signing-verification key pair $(\mathrm{SK}, \mathrm{PK})$. If the PoS-player is chosen, then the following hash inequality holds: $\mathsf{H}(\mathsf{hash}(B_i), \mathtt{round}, \mathrm{PK}, \sigma) < \mathtt{T}$, where $\sigma := \mathsf{uSign}_{\mathrm{SK}}(h_i, \mathtt{round})$, and $h_i := \mathsf{hash}(B_i)$. The new block-core B_{i+1} is defined as $B_{i+1} := \langle h_i, \mathtt{round}, \mathrm{PK}, \sigma \rangle$.

Next we will provide a formal description for our protocol. We use a setup functionality $\mathcal{F}_{\mathsf{rCERT}}$ in Section 3.1 to capture the hash inequality and the block-core signing/verification. This setup functionality can be implemented by using hash function $\mathsf{H}(\cdot)$ and a unique signature scheme $(\mathsf{uKeyGen}, \mathsf{uKeyVer}, \mathsf{uSign}, \mathsf{uVerify})$. Finally, we note that in this section, the adversary will follow the basic strategy in the sense that all the accounts are registered previously before the protocol execution.

3.1 Setup functionality $\mathcal{F}_{\mathsf{rCERT}}$

Resource certification functionality $\mathcal{F}_{\mathsf{rCERT}}$. The functionality consists of several phases, "Stake Resource Registration", "Stake Election", and "Block Verification". In this version of resource certificate functionality, the "Stake Resource Registration" phase is disabled, and we have all PoS-player P registered initially. (Jumping ahead, in Section 5, a strengthened version of resource certification functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ in Figure 10, will be introduced; there, at any time step, a PoS-player P can send a register command to functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ for registration.) For simplicity, we assume a registered PoS-player P is granted *one unit of the stake*, and he can request the functionality for leader election once in each execution round.

Firstly, we introduce "Stake Resource Registration" phase. In this phase, a player P sends the comand $(\textsc{Elect}, P, \langle h^{\mathrm{prev}}, \mathtt{round} \rangle)$ to the functionality. If the player is qualified for election, the functionality computes the unique signature $\sigma := \mathsf{uSign}(\mathsf{sk}_P, \langle h^{\mathrm{prev}}, \mathtt{round} \rangle)$. The functionality then with probability p selects this party as the leader and notifies the player whether he is selected or not.In this phase, We remark that, the elected party P can obtain only a *single* signature for a h^{prev} in one round; in previous certificate or digital signature functionalities (see [17]), multiple signatures are allowed to be generated for the same value. Then the functionality generate a unique id h for the player P as the identity of a new block. With the id h, the functionality can distinguish every valid block. The functionality store a record of the form $\langle h^{\mathrm{prev}}, \mathtt{round}, P, \sigma, h, 1 \rangle$ for the new block. Here, h^{prev} is the id of the previous block. So that the functionality can and trace the order of of blocks. The functionality return $(\textsc{Elected}, P, h, \sigma, \mathsf{b})$ to P and adversary where b is used to indicate if P is elected in this round.

Secondly, the verification process of $\mathcal{F}_{\mathsf{rCERT}}$ proceeds as follows. Upon receiving a verification request, the functionality will check if the signature is valid by sending a request to adversary. Then the functionality will also check there is a valid record of this block been recorded. This would ensure the completeness, unforgeability, and guarantees consistency properties of the block.

In Supplementary material B, we implement functionality $\mathcal{F}_{\mathsf{rCERT}}$ in the $\mathcal{F}_{\mathsf{RO}}$-hybrid model.

3.2 Our core-chain protocol

We now describe the core-chain protocol Π^{core}. Each PoS-player P, once activated by the environment on $(\textsc{Input-Stake}, P)$ at round round, and received a core-chain set $\mathbb{C}$ from $\mathcal{F}_{\mathsf{NET}}$, the party P finds the best valid core-chain $\mathcal{C}_{\mathrm{best}}$ by running the subroutine BestCore (in Figure 4), and then updates its local core-chain $\mathcal{C} := \mathcal{C}_{\mathrm{best}}$.

Let ℓ be the length of core-chain $\mathcal{C}$. In our design, only the elected PoS-players are allowed to generate new block-cores (to extend the core-chain). Now, each registered PoS-player P will work on the right "context" which consists of the *latest block-core in the longest core-chain and the current time*; formally $\mathit{context} := \langle h^{\mathrm{prev}}, \mathtt{round} \rangle$ where $\mathcal{C}[\ell]$ is the latest block-core in the longest core-chain $\mathcal{C}$, and h^{prev} is the identity returned by the functionality $\mathcal{F}_{\mathsf{rCERT}}$ for $\mathcal{C}[\ell]$, and round denotes the current time. The PoS-player P may query $\mathcal{F}_{\mathsf{rCERT}}$ by command $(\textsc{Elect}, P, \mathit{context}, \mathcal{C})$ to see if he is selected to extend $\mathcal{C}$. If the PoS-player P is

FUNCTIONALITY $\mathcal{F}_{\mathsf{rCERT}}$

The functionality interacts with a set $\mathcal{P}$ of parties, an an adversary. The functionality is parameterized by a difficulty parameter p, a security parameter κ, and a unique digital signature scheme $\Sigma = ((\mathsf{uKeyGen}, \mathsf{uKeyVer}, \mathsf{uSign}, \mathsf{uVerify}))$.
Initially, a set $\mathcal{P}_0$ of distinct players are registered, where $\mathcal{P}_0 \subseteq \mathcal{P}$; That is, for each $P \in \mathcal{P}_0$, compute $(\mathsf{sk}_P, \mathsf{pk}_P) \leftarrow \mathsf{uKeyGen}(1^\kappa)$, record the tuple $(P, \mathsf{pk}_P, \mathsf{sk}_P)$, send (P, pk_P) to the adversary, and party P.

Stake Resource Registration.

1. (The command STAKE-REGISTER is disabled, and all stake registration must be completed during initialization. In the strengthened version of the functionality in Figure 10, this command will be enabled for supporting regular stake registration.)

2. Upon receiving message $(\text{RETRIEVE}, P)$ from party $P' \in \mathcal{P}$, if there is a recorded tuple $(P, \mathsf{sk}_P, \mathsf{pk}_P)$, then output $(\text{RETRIEVED}, P, \mathsf{pk}_P)$ to P'. Else output $(\text{RETRIEVED}, P, \perp)$ to P'.

Stake Election:
For each round,

1. Set $\mathtt{B}_P := 0$ for every registered party $P \in \mathcal{P}_0$.

2. Upon receiving $(\text{ELECT}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle)$ from party P, do:

   - Set $\mathsf{b} := 0$. (the party P is not elected by default)
   - If $(P, \mathsf{sk}_P, \mathsf{pk}_P)$ has been recorded, and $\mathtt{B}_P = 0$ (P has been registered and been granted with one unit of stake), do:
     compute $\sigma := \mathsf{uSign}(\mathsf{sk}_P, \langle h^{\text{prev}}, \mathtt{round}\rangle)$;
     verify that $\mathsf{uVerify}(\mathsf{pk}_P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma) = 1$;
     if verified, then choose a random tag h;
     set $\mathsf{b} := 1$ with probability p; and set $\mathsf{b} := 0$ otherwise. (the party P is elected with probability p);
     record $\langle h^{\text{prev}}, \mathtt{round}, P, \sigma, h, \mathsf{b}\rangle$.

3. Set $\mathtt{B}_P := 1$

4. Send $(\text{ELECTED}, P, \sigma, h, \mathsf{b})$ to party P and the adversary.

Block Verification:

1. Upon receiving $(\text{CORE-VERIFY}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma, h)$ from $P' \in \mathcal{P}$,

   - If $\langle h^{\text{prev}}, \mathtt{round}, P, \sigma, h, 1\rangle$ is recorded, then set $f := 1$.

2. Output $(\text{CORE-VERIFIED}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma, h, f)$ to the party P'.

Figure 2: Resource certification functionality $\mathcal{F}_{\mathsf{rCERT}}$.

selected (with certain probability p), he would receive a message $(\text{ELECTED}, P, h, \sigma, \mathsf{b})$ from $\mathcal{F}_{\mathsf{rCERT}}$ such that $\mathsf{b} = 1$. Once receiving the signature σ from the functionality, the PoS-player P defines a new block-core $B := \langle\langle h^{\text{prev}}, h, \mathtt{round}\rangle, P, \sigma\rangle$, updates his local core-chain $\mathcal{C}$ and then broadcasts the local core-chain to the network. Please refer to Figure 3 for more details of our core-chain protocol.

Note that here PoS-players have access to the functionality $\mathcal{F}_{\mathsf{rCERT}}$. The players need to register to the functionality $\mathcal{F}_{\mathsf{rCERT}}$ before querying the functionality.

The best core-chain strategy. Our proof-of-stake core-chain protocol Π^{core} uses the subroutine BestCore to single out the best valid core-chain from a set of core-chains. Now we describe the rules of selecting the best

PROTOCOL Π^{core}

Initially, a set $\mathcal{P}_0$ of players are registered to the functionality $\mathcal{F}_{\mathsf{rCERT}}$, where $\mathcal{P}_0 \subseteq \mathcal{P}$.
Set $\mathcal{C} := \emptyset$, and $state := \emptyset$.

Upon receiving message (INPUT-STAKE, P) from the environment $\mathcal{Z}$ at round $\mathtt{round}$, the PoS-player $P \in \mathcal{P}$, with local state $state$, proceeds as follows.

1. *Select the best local PoS core-chain:*

   Let $\mathbb{C}$ be the set of core-chains collected from $\mathcal{F}_{\mathsf{NET}}$.

   Compute $\mathcal{C}_{\text{best}} := \mathsf{BestCore}(\mathbb{C} \cup \{\mathcal{C}\}, \mathtt{round})$, and set $\mathcal{C} := \mathcal{C}_{\text{best}}$, and $\ell := \mathsf{len}(\mathcal{C}_{\text{best}})$

2. *Attempt to extend PoS core-chain:*

   Parse $\mathcal{C}[\ell]$ as $\langle\langle h_\ell^{\text{prev}}, \mathtt{round}_\ell, P_\ell, \sigma_\ell\rangle, h_\ell\rangle$.

   - *Stake election:* Send (ELECT, $P, \langle h_\ell, \mathtt{round}\rangle$) to functionality $\mathcal{F}_{\mathsf{rCERT}}$, and receive (ELECTED, $P, h_{\ell+1}, \sigma, \mathsf{b}$) from $\mathcal{F}_{\mathsf{rCERT}}$.
   - *If* $\mathsf{b} = 1$*, generate a new block-core:* Set the new block-core $B := \langle\langle h_\ell, \mathtt{round}, P, \sigma\rangle, h_{\ell+1}\rangle$, and set $\mathcal{C} := \mathcal{C} \| B$, and $state := state \cup \{\mathcal{C}\}$, and then send (BROADCAST, $\mathcal{C}$) to $\mathcal{F}_{\mathsf{NET}}$.

   Return (RETURN-Stake, P) to the environment $\mathcal{Z}$.

Figure 3: Our proof-of-stake core-chain protocol Π^{core} in the $\{\mathcal{F}_{\mathsf{rCERT}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid model. (See Figure 4 for the subroutine BestCore.)

core-chain. Roughly speaking, a core-chain is the best one if it is the *current longest valid* core-chain. The BestCore subroutine takes as input, a core-chain set $\mathbb{C}'$ and the current time information $\mathtt{round}'$. Intuitively, the subroutine validates all $\mathcal{C} \in \mathbb{C}'$, then finds the valid longest core-chain.

SUBROUTINE BestCore

The subroutine BestCore has the access to $\mathcal{F}_{\mathsf{rCERT}}$.
Input: $(\mathbb{C}', \mathtt{round}')$.
Output: $\mathcal{C}_{\text{best}}$.

For every chain $\mathcal{C} \in \mathbb{C}'$, and proceed as follows.

1. Set $\ell := \mathsf{len}(\mathcal{C})$.
2. For i from ℓ down to 1, verify block-core $\mathcal{C}[i]$, as follows.
   - Parse $\mathcal{C}[i]$ into $\langle\langle h_i^{\text{prev}}, \mathtt{round}_i, P_i, \sigma_i\rangle, h_i\rangle$.
     Parse $\mathcal{C}[i-1]$ into $\langle\langle h_{i-1}^{\text{prev}}, \mathtt{round}_{i-1}, P_{i-1}, \sigma_{i-1}\rangle, h_{i-1}\rangle$.
   - If $\mathtt{round}_i < \mathtt{round}'$ and $\mathtt{round}_{i-1} < \mathtt{round}_i$, execute:
     - If $h_i^{\text{prev}} \neq h_{i-1}$, remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.
     - Else send (CORE-VERIFY, $P_i, \langle h_i^{\text{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i$) to $\mathcal{F}_{\mathsf{rCERT}}$. Upon receiving message (CORE-VERIFIED, $P_i, \langle h_i^{\text{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i, f_i$) from $\mathcal{F}_{\mathsf{rCERT}}$, if $f_i = 0$ remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.

     Otherwise, remove the core-chain $\mathcal{C}$ from $\mathbb{C}'$.

Set $\mathcal{C}_{\text{best}}$ be the longest core-chain in $\mathbb{C}'$. Then output $\mathcal{C}_{\text{best}}$.

Figure 4: The core-chain set validation subroutine BestCore.

In more detail, BestCore proceeds as follows. On input the current set of core-chains $\mathbb{C}'$ and the current

time information $\texttt{round}'$, and for each core-chain $\mathcal{C}$, the subroutine then evaluates every block-core of the core-chain $\mathcal{C}$ sequentially. Let ℓ be the length of $\mathcal{C}$. Starting from the head of $\mathcal{C}$, for every block-core $\mathcal{C}[i]$, for all $i \in [\ell]$, in the core-chain $\mathcal{C}$, the $\mathsf{BestCore}$ subroutine (1) ensures that $\mathcal{C}[i]$ is linked to the previous block-core $\mathcal{C}[i-1]$ correctly, and (2) tests if the signature generated by that PoS-player can be verified (by interacting with $\mathcal{F}_{\mathsf{rCERT}}$). After the validation, the best valid core-chain is the longest one. (See Figure 4 for more details.)

3.3 Security analysis, high-level ideas

We are now ready to state our theorem for our core-chain protocol Π^{core} in the presence of an adversary who extends blockchain via the basic strategy (i.e., extending a single chain only).

Theorem 3.1 (Theorem 1.1, restated)**.** *Consider core-chain protocol Π^{core} in the presence of restricted adversary, i.e., all honest players follow the simple strategy of extending the longest chain while the adversary extends at most one chain in each time window; in addition, both honest and malicious players have their stake registered independent of the state in the protocol execution. Let α and β be the effective expected number of blocks generated by honest and malicious players in a round respectively. If $\alpha = \lambda\beta$, $\lambda > 1$, then the protocol Π^{core} can achieve chain growth, chain quality, common prefix and chain soundness properties.*

3.3.1 Proof ideas

We describe the high-level proof ideas. The proof details can be found in Supplementary material A.

In our design, malicious players cannot stop the honest players from being selected to generate new block-cores. This means the chain growth property can be achieved. Furthermore, the portion of block-cores from malicious players are bounded by the proportion of stakes they control. Recall that, in our design, we assume that the honest players control more stakes than the malicious players. For the same core-chain, the malicious players cannot contribute more block-cores than the honest players do. This means the chain quality property can be achieved. Finally, we assume that, in each round, the probability that the stakeholders find a new block-core B is very small. This means, in most of the rounds, no new block-core will be generated and be broadcast, and all of the honest players will extend the same core-chain. Note that, even all of malicious players try to extend a different core-chain, the growth rate of the malicious core-chain is still lower than that of the public core-chain. This will allow us to prove the common prefix property. Our core-chain protocol will be executed in a setting that the adversary can delay the messages from honest players up to certain say Δ, number of rounds. The honest players may be misled to work on a wrong core-chain during the delayed rounds. As a result, the effort from the honest players is wasted during these delayed rounds. Our analysis will also take into account of the network delay.

Here we assume all players have their stakes registered without being aware of the state of the protocol execution. In Section 5, we will discuss how to eliminate this assumption, and show how to improve the core-chain protocol so that it can be executed in a more realistic setting.

Chain growth. In order to calculate the chain growth rate, we consider the worst case for the honest players. The best strategy for the malicious players is to delay all of the messages from the honest players to discount the stakes of honest players. We use α to denote the discounted number of block-cores that honest players can generate. We have $\alpha = \frac{\alpha_0}{1+\Delta\alpha_0}$. (The calculation steps can be found in Supplementary material A.1.) We use a hybrid execution to formalize the worst delay setting in the formal proof. In the hybrid execution, the malicious players contribute nothing to the chain growth and delay all honest messages to decrease the chain growth rate. In the real execution, the probability that an honest player is chosen will not be lower than that in the hybrid execution. The message from malicious players will not decrease the chain growth that contributed by honest players. Therefore, the chain growth rate is not worse than that in the hybrid execution.

Lemma 3.2 (Chain growth)**.** *Consider an execution of core-chain protocol* Π^{core}*, where an honest PoS-player* P' *is with best local core-chain* $\mathcal{C}'$ *in round* r'*, and an honest PoS-player* P'' *with best local core-chain* $\mathcal{C}''$ *in round* r''*, and* $r'' > r'$*. Then we have* $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$ *where* $t = r'' - r'$*,* $g = (1-\delta)\alpha$*, and* $\delta > 0$*.*

Chain quality. In order to reduce the chain quality, the best strategy for malicious parties is to generate as many block-cores as they can. When the honest players generate and broadcast a new block-core, they will try to send out another one to compete with the honest one. We focus on the worst case that the malicious players win all of the competition. During any t consecutive rounds, the chain growth rate is αt on average. The malicious players will contribute βt block-cores. The chain quality will remain at least $1 - \frac{\beta}{\alpha}$.

Lemma 3.3 (Chain quality)**.** *Assume that* $\alpha = \lambda\beta$*, and* $\lambda > 1$*. Consider an execution of core-chain protocol* Π^{core}*, where an honest PoS-player is with core-chain* $\mathcal{C}$*. If among* ℓ *consecutive block-cores in chain* $\mathcal{C}$*, there are* ℓ_{good} *block-cores that are generated by honest PoS-players, then we have* $\Pr\left[\frac{\ell_{\text{good}}}{\ell} \geq \mu\right] \geq 1 - e^{-\Omega(\ell)}$ *where* $\mu = 1 - (1+\delta)\frac{1}{\lambda}$*, and* $\delta > 0$*.*

Common prefix. We assume $\alpha + \beta \ll 1$. This guarantees that the honest players will work on the same best core-chain in most rounds. We also assume the majority of PoS-players are honest, so that other chain will not grow as fast as the longest chain. Together, we have that the public best chain is longer than any other core-chains after a sufficient long time period. All of the honest players will converge on the best public chain with high probability except the last several block-cores.

Lemma 3.4 (Common prefix)**.** *Assume that* $\alpha = \lambda\beta$*, and* $\lambda > 1$*. Consider an execution of core-chain protocol* Π^{core}*, where two honest PoS-players,* P *in round* r *and* P' *in round* r'*, with the local best core-chains* $\mathcal{C}$ *and* $\mathcal{C}'$*, respectively, where* $r' \geq r$*. Then we have* $\Pr\left[\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'\right] \geq 1 - e^{-\Omega(\kappa)}$*.*

Chain soundness. A new player will take the longest core-chain as the best chain. As we discussed above, the honest players can generate the longest chain except the latest several block-cores. This means the new player can choose the best core-chain as the existing players do in the protocol execution.

Lemma 3.5 (Chain soundness)**.** *Consider for every round,* $\alpha = \lambda\beta$*,* $\lambda > 1$*, and* $\delta > 0$*. Consider core-chain protocol* Π^{core}*. Consider two honest PoS-players,* P' *and* P'' *in round* r*, with the local best core-chains* $\mathcal{C}'$ *and* $\mathcal{C}''$*, respectively, where* P' *is a new player and* P'' *is an existing player in round* r*. Then we have* $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ *and* $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$*.*

4 Greedy strategies, defend against an arbitrary adversary

In the previous section (Section 3), we consider a restricted adversary who (1) extends at most one chain in each round, and (2) has its stakes registered independent of the state of protocol execution. In this section, we will remove the restriction condition (1); our goal here is to design a core-chain protocol which is secure against a more realistic adversary who can extend the chains in an arbitrary way, as long as the adversary has his stakes registered independent of the state of protocol execution.

4.1 Greedy strategies

We remark that, in a proof-of-stake protocol, extending chains can be "very cheap". Thus, a proof-of-stake player may take a *greedy* strategy to extend the core-chains in a protocol execution: instead of extending a single chain, the player makes attempts to extend *a set of chains*, expecting to extend the best chain faster. Next, we will formally study greedy strategies.

Distance-greedy strategies We first introduce *distance-greedy strategies* for honest protocol players. Consider a blockchain protocol execution. In each player's local view, there are multiple chains, which can be viewed as a tree. More concretely, the genesis block is the root of the tree, and each path from the root to a leaf is essentially a chain. The tree will "grow": the length of each existing chain may increase, and new chains may be created, round after round.

Before giving the formal definition for distance-greedy strategies, we define the "distance" between two chains in a tree.

Definition 4.1 (Distance between two chains)**.** *Consider two chains $\mathcal{C}$ with length ℓ, and $\mathcal{C}'$ with length ℓ', respectively. Let d be a non-negative integer. We say the distance of chain $\mathcal{C}'$ from chain $\mathcal{C}$ is d, if d is the smallest non-negative integer so that $\mathcal{C}'[1, \ell' - d] \preceq \mathcal{C}$, and we write* $\mathsf{distance}(\mathcal{C} \to \mathcal{C}') = d$.

Remark 4.2. *We note that the distance of $\mathcal{C}$ from $\mathcal{C}'$ is different from the distance of $\mathcal{C}'$ from $\mathcal{C}$, and it is very possible that* $\mathsf{distance}(\mathcal{C}' \to \mathcal{C}) \neq \mathsf{distance}(\mathcal{C} \to \mathcal{C}')$. *For example, in Figure 5, the distance of $\mathcal{C}$ from $\mathcal{C}'$ is 4, i.e.* $\mathsf{distance}(\mathcal{C}' \to \mathcal{C}) = 4$, *while the distance of $\mathcal{C}'$ from $\mathcal{C}$ is 2, i.e.,* $\mathsf{distance}(\mathcal{C} \to \mathcal{C}') = 2$. *We also note that the distance of $\mathcal{C}$ from itself is always 0, i.e.,* $\mathsf{distance}(\mathcal{C} \to \mathcal{C}) = 0$.

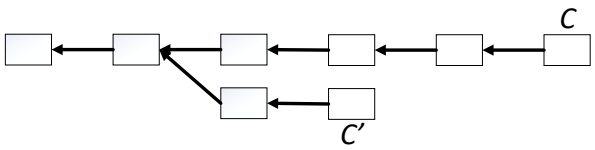

Figure 5: The distance between two chains $\mathcal{C}$ and $\mathcal{C}'$. (The distance from $\mathcal{C}'$ to $\mathcal{C}$ is 2; the distance from $\mathcal{C}$ to $\mathcal{C}'$ is 4).

We are ready to introduce the distance-greedy strategies. Intuitively, a player who plays a distance-greedy strategy will make attempts to extend a *set of chains* in which all chains have the following properties: (1) the chain should be very "close" to the best chain, i.e., the distance from the chain to the best chain must be small; (2) the chain should not fall behind the best chain too much, i.e., the length of the chain must be big. More formally, we have the following definition.

Definition 4.3 ($(\mathtt{D}, \mathtt{F})$-distance-greedy strategy)**.** *Consider a blockchain protocol execution. Let P be a player of the protocol execution, and T be a tree which consists of chains with the same genesis block, in player P's local view. Let $\mathcal{C}_{\text{best}}$ be the longest chain at round r, where $\ell = \mathsf{len}(\mathcal{C}_{\text{best}})$. Consider parameters $\mathtt{D}$ and $\mathtt{F}$. Define a chain set $\mathbb{C}_{\text{best}}$ as*

$$\mathbb{C}_{\text{best}} = \left\{\mathcal{C} \;\middle|\; \mathsf{distance}(\mathcal{C}_{\text{best}} \to \mathcal{C}) \leq \mathtt{D} \bigwedge \mathsf{len}(\mathcal{C}) \geq \ell - \mathtt{F}\right\}$$

We say the player is $(\mathtt{D}, \mathtt{F})$-distance-greedy *if, for all r, the player makes attempts to extend all chains $\mathcal{C} \in \mathbb{C}_{\text{best}}$.*

Remark 4.4 ($\mathtt{D}$-distance-greedy strategy)**.** *In the above definition, $\mathtt{F} \leq \mathtt{D}$. In the remaining of the paper, for simplicity we set $\mathtt{F} := \mathtt{D}$, and define the best chain set as*

$$\mathbb{C}_{\text{best}} = \left\{\mathcal{C} \;\middle|\; \mathsf{distance}(\mathcal{C}_{\text{best}} \to \mathcal{C}) \leq \mathtt{D}\right\}$$

and call it $\mathtt{D}$-distance-greedy strategy. In Figures 6 and 7, pictorial illustration for the examples of 1-distance-greedy and 2-distance-greedy strategies can be found respectively. In both figures, the blocks, which are marked with a blue "$\oslash$" symbol on the top right, are not extended by the honest player in the current round.

We also note that many useful variants of the above distance greedy strategies can be designed. A fundamental rule here is that, honest players will only extend the chains which are close to the best extension position.

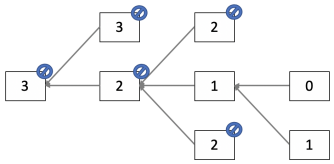

Figure 6: 1-distance-greedy

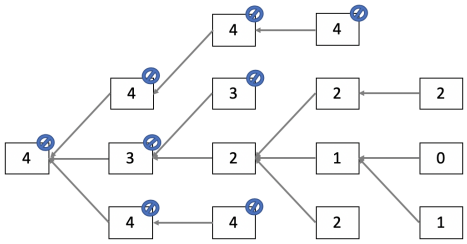

Figure 7: 2-distance-greedy

Adversarial strategies Now we turn to consider the adversarial strategies. An adversary may consider different strategies to break the security of the blockchain system. For example, a restricted adversary, as we discussed in the previous section, may follow the basic strategy by extending only a single chain in each round; the basic strategy can be viewed as the 0-distance-greedy strategy. Another restricted adversary may follow the *full-greedy strategy* to extend the all chains; here the full-greedy strategy is essentially the ℓ-distance-greedy where ℓ is the length of the longest chain.

In this section, we will defend against a much more powerful adversary who, instead of playing a restricted adversarial strategy, may extend the chains in an arbitrary manner. We note that, it is non-trivial to defend against an arbitrary adversary. A core-chain protocol secure against the full-greedy adversary may become insecure when the adversary plays an arbitrary strategy. In Appendix C, we present a concrete example by constructing a core-chain protocol which is secure against the full-greedy adversary, but insecure against an arbitrary adversary.

4.2 The modified core-chain protocol $\Pi^{\text{core}\circ}$

Next, we present a new core-chain protocol $\Pi^{\text{core}\circ}$ with the goal of defending against adversaries who play any arbitrary strategies. This new protocol is based on the core-chain protocol Π^{core} in Section 3 but now the players follow the $\mathtt{D}$-distance-greedy strategy. Details can be found in Figure 8.

We note that, the subroutine BestCore in the previous protocol, has also been modified into subroutine $\mathtt{D}$-BestCore$^{\circ}$; now the modified subroutine $\mathtt{D}$-BestCore$^{\circ}$, instead of returning the single longest chain, will output *a set of chains* including the longest chain, and several chains that are very close to, and slightly (i.e., $\mathtt{D}$ blocks) shorter than the longest chain. Intuitively, the bigger the greedy parameter $\mathtt{D}$ is, the better the chance that the player extend the set of chains. However, the (computation and storage) complexity of the protocol is proportional to the greedy parameter $\mathtt{D}$. In practice, we can choose $\mathtt{D} = 2$.

In our protocol execution, when there are two or more number of longest chains, i.e., they are with the same length in the honest player's local view, the honest player will adopt the earliest chain in his view.

4.3 Security analysis

Stake amplification In our core-chain protocol $\Pi^{\text{core}\circ}$, the honest players extend the chains by using the $\mathtt{D}$-distance-greedy strategy. Here every honest player will take a single chain as the best chain (and via the $\mathtt{D}$-distance-greedy strategy, the honest player makes efforts to extend the best set of chains). The advantage of following the $\mathtt{D}$-distance-greedy strategy is that, the honest player can amplify his stake, and extend the chains faster. We introduce *amplification ratio*.

Definition 4.5 (Amplification ratio)**.** *Consider a PoS blockchain protocol. Let N_0 be the average increased length of the longest chain that extended by a group of players P, follow the 0-distance-greedy strategy. Let $N_{\mathtt{D}}$ be the average increased length of the longest chain that extended by the same group of players P, follow the $\mathtt{D}$-distance-greedy strategy. We define the amplification ratio for following the $\mathtt{D}$-distance-greedy strategy as $\mathtt{A}^{\circ}_{\mathtt{D}} = \frac{N_{\mathtt{D}}}{N_0}$*

PROTOCOL $\Pi^{\text{core}\circ}$

Initially, a set $\mathcal{P}_0$ of players are registered to the functionality $\mathcal{F}_{\mathsf{rCERT}}$, where $\mathcal{P}_0 \subseteq \mathcal{P}$.
Initially, for each $P \in \mathcal{P}$, set $\mathcal{C} := \emptyset$, and $state := \emptyset$.

Upon receiving message $(\text{INPUT-STAKE}, P)$ from the environment $\mathcal{Z}$ at round $\mathtt{round}$, the PoS-player $P \in \mathcal{P}$, with local state $state$, proceeds as follows.

1. *Select the best local PoS core-chain:*

   Let $\mathbb{C}$ be the set of core-chains collected from $\mathcal{F}_{\mathsf{NET}}$.

   Compute $\mathbb{C}_{\text{best}} := \mathtt{D\text{-}BestCore}^{\circ}(\mathbb{C} \cup \{\mathcal{C}\}, \mathtt{round})$. (D-BestCore$^{\circ}$ will return a best chain set)

2. *Attempt to extend PoS core-chain:*

   For each $\mathcal{C} \in \mathbb{C}_{\text{best}}$, do: (Each player will try to extend all chains in the best chain set)

   - Set $\ell := \mathsf{len}(\mathcal{C})$
   - Parse $\mathcal{C}[\ell]$ as $\langle\langle h_\ell^{\text{prev}}, \mathtt{round}_\ell, P_\ell, \sigma_\ell\rangle, h_\ell\rangle$.
   - *Stake election:* Send $(\text{ELECT}, P, \langle h_\ell, \mathtt{round}\rangle)$ to functionality $\mathcal{F}_{\mathsf{rCERT}}$, and receive $(\text{ELECTED}, P, h_{\ell+1}, \sigma, \mathsf{b})$ from $\mathcal{F}_{\mathsf{rCERT}}$.
   - *If* $\mathsf{b} = 1$, *generate a new block-core:* Set the new block-core $B := \langle\langle h_\ell, \mathtt{round}, P, \sigma\rangle, h_{\ell+1}\rangle$, and set $\mathcal{C} := \mathcal{C}\|B$, and $state := state \cup \{\mathcal{C}\}$, and then send $(\text{BROADCAST}, \mathcal{C})$ to $\mathcal{F}_{\mathsf{NET}}$.

   Return $(\text{RETURN-Stake}, P)$ to the environment $\mathcal{Z}$.

Figure 8: Our proof-of-stake core-chain protocol $\Pi^{\text{core}\circ}$ in the $\{\mathcal{F}_{\mathsf{rCERT}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid model. (See Figure 9 for the subroutine D-BestCore$^{\circ}$.)

Jumping ahead, in the remaining of this section, we will show that the amplification ratio for following the fully-greedy strategy is $\mathtt{A}^{\circ}_{\text{fully}} = 2.718$, and the amplification ratio for following the 2-distance-greedy strategy is $\mathtt{A}^{\circ}_{2} = 1.65$. In addition, by definition, $\mathtt{A}^{\circ}_{0} = 1$.

Virtual chains We now bring a new perspective for our blockchain analysis, by introducing the notion of *virtual chains*. Based on the definition of D-distance-greedy strategy (as in Definition 4.3), the best set of chains are bounded by the longest chain and two small constants D and F. This definition will ensure that all chains in the set will be very "close" to the longest chain. We thus can view this set of chains as a single *virtual* chain. We note that, the chain growth rate for the virtual chain will become larger, since extending the virtual chain is equivalent to extending the set of chains. Now, instead of having the honest players make the attempts to extend the best set of chains, we let the honest players make attempts to extend the single best virtual chain with the amplified growth rate.

The adversary may also participate in the protocol execution by playing certain greedy strategy or even an arbitrary strategy. Depending on the strategies the adversary plays, equivalently the adversary is making efforts to extend a single or multiple virtual chains. In the remaining of the paper, we will focus on the analysis of virtual chains; if the context is clear, we often ignore the word "virtual" for simplicity.

Theorem statement In previous section, the security properties of protocol Π^{core} have been proven under the assumption of honest majority of stakes based on α and β. Now, we can prove the security properties of the modified core-chain protocol $\Pi^{\text{core}\circ}$ but under the assumption of honest majority of *effective stakes* based on α° and β°. Here $\alpha^{\circ} = 1.6\alpha$ and $\beta^{\circ} = 2.718\beta$. We note that here $\alpha^{\circ} \ll 1$ and $\beta^{\circ} \ll 1$ (since $\alpha \ll 1$ and $\beta \ll 1$). We have:

Theorem 4.6 (Theorem 1.2, restated)**.** *Consider an execution of core-chain protocol $\Pi^{\text{core}\circ}$: all honest players follow the 2-distance-greedy strategy while adversarial players could follow any arbitrary strategy; in addition, all players have their stakes registered without being aware of the state of the protocol execution. If $\alpha^{\circ} = \lambda\beta^{\circ}$,*

SUBROUTINE D-BestCore$^{\circ}$

The subroutine D-BestCore$^{\circ}$ has the access to $\mathcal{F}_{\mathsf{rCERT}}$.
Input: $(\mathbb{C}', \mathtt{round}')$.
Output: $\mathbb{C}_{\mathrm{best}}$.

Verify the chain set
For every chain $\mathcal{C} \in \mathbb{C}'$, proceed as follows.

1. Set $\ell := \mathsf{len}(\mathcal{C})$.
2. For i from ℓ down to 1, verify block-core $\mathcal{C}[i]$, as follows.
   - Parse $\mathcal{C}[i]$ into $\langle\langle h_i^{\mathrm{prev}}, \mathtt{round}_i, P_i, \sigma_i\rangle, h_i\rangle$.
     Parse $\mathcal{C}[i-1]$ into $\langle\langle h_{i-1}^{\mathrm{prev}}, \mathtt{round}_{i-1}, P_{i-1}, \sigma_{i-1}\rangle, h_{i-1}\rangle$.
   - If $\mathtt{round}_i < \mathtt{round}'$ and $\mathtt{round}_{i-1} < \mathtt{round}_i$, execute:
     - If $h_i^{\mathrm{prev}} \neq h_{i-1}$, remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.
     - Else send $(\textsc{Core-Verify}, P_i, \langle h_i^{\mathrm{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i)$ to $\mathcal{F}_{\mathsf{rCERT}}$. Upon receiving message $(\textsc{Core-Verified}, P_i, \langle h_i^{\mathrm{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i, f_i)$ from $\mathcal{F}_{\mathsf{rCERT}}$, if $f_i = 0$ remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.

     Otherwise, remove the core-chain $\mathcal{C}$ from $\mathbb{C}'$.

Identify the best chain
Set $\mathcal{C}_{\mathrm{best}}$ as the longest core-chain in $\mathbb{C}'$.

Identify the best chain set
Set $\mathbb{C}_{\mathrm{best}} := \emptyset$.
For any chain $\mathcal{C} \in \mathbb{C}'$

- if $\mathsf{distance}(\mathcal{C}_{\mathrm{best}} \rightarrow \mathcal{C}) \leq \mathtt{D}$ then set $\mathbb{C}_{\mathrm{best}} := \mathbb{C}_{\mathrm{best}} \cup \{\mathcal{C}\}$. (D is a small constant, $\mathtt{D} = 2$ typically.)

Then return $\mathbb{C}_{\mathrm{best}}$ as the output.

Figure 9: The core-chain set validation subroutine D-BestCore$^{\circ}$.

$\lambda > 1$, *then the protocol* $\Pi^{\mathrm{core}\circ}$ *can achieve chain growth, chain quality, common prefix and chain soundness properties.*

To prove the main theorem here, we first show a very interesting lemma (i.e., Lemma 4.7) that the amplification ratio for following any arbitrary strategy is bounded by a factor e (the base of natural logarithm). Intuitively, if protocol players follow any arbitrary strategy and extend all chains, one of the relatively shorter chains will become the longest chain with certain probability; that means, the longest chain will be extended faster. However, we note that, the shorter the chain is, the lower the probability of being extended into the longest chain is; collectively, the longest chain will strictly increase but will be bounded by a constant factor.

Lemma 4.7. *Consider core-chain protocol* $\Pi^{\mathrm{core}\circ}$. *Assume that malicious players can generate a new block with probability* β *in a round. Assume that the malicious players could follow any arbitrary strategy to extend a core-chain* $\mathcal{C}'$ *at round* r' *into* $\mathcal{C}''$ *at round* r'', *where* $r'' > r'$. *Then we have* $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') < e\beta \cdot t\right] \geq 1 - e^{-\Omega(t)}$ *where* $t = r'' - r'$ *and* e *is the base of natural logarithm.*

Proof. Let $f(t, l)$ be the number of chains with length $l + \mathsf{len}(\mathcal{C}')$ at round $t + r'$, where $0 \leq t \leq r'' - r'$. The initial conditions are as follows:

- $f(t, 0) := 1$ for all $t \geq 0$;

it means that, at $t \geq 0$, there is only 1 chain (i.e. $\mathcal{C}'$) which is 0 block longer than $\mathcal{C}'$;

- $f(0, l) := 0$ for all $l > 0$;
  it means that, at $t = 0$, there is no chain which is l blocks longer than $\mathcal{C}'$;

We first discuss the average case. Recall that malicious players can generate a new block with probability β in a round. We have $f(t, l) := f(t-1, l) + f(t-1, l-1)\beta$ on average. Note that, if we view β as a variable, the coefficients of $f(t, l)$ can be viewed as a polynomial, and they will follow the Pascal's Triangle. That is, we have $f(t, l) = \binom{t}{l}\beta^l$.

We now develop a simplified form of $f(t, l)$. We note that, in a round there is at most one block that can be extended from an existing block on average. Let k be the number of rounds for generating one block on average, and we have $k > 1$, and $t = kl$. Now we have:

$$\begin{aligned}
f(t,l) &= \frac{t!}{(l!)(t-l)!}\beta^l \\
&\approx \frac{\sqrt{2\pi t}t^t}{(\sqrt{2\pi l}l^l)\cdot(\sqrt{2\pi(t-l)}(t-l)^{t-l})}\cdot\beta^l \\
&= \frac{\sqrt{2\pi kl}(kl)^{kl}}{(\sqrt{2\pi l}l^l)\cdot(\sqrt{2\pi(kl-l)}(kl-l)^{kl-l})}\cdot\beta^l \\
&= \sqrt{\frac{kl}{2\pi l(kl-l)}}\cdot\frac{(kl)^{kl}}{l^l(kl-l)^{kl-l}}\cdot\beta^l \\
&= \sqrt{\frac{k}{2\pi(k-1)l}}\left[\frac{k^k}{(k-1)^{k-1}}\cdot\beta\right]^l
\end{aligned}$$

The second approximate equality above is based on Stirling's approximation.

Note that, $g(k) = \left(1+\frac{1}{k}\right)^k$ is a monotone increasing function and $\lim_{k\to\infty} g(k) = e$. We now have $\left(\frac{k}{k-1}\right)^{k-1} < e$. That is :

$$f(t,l) < \sqrt{\frac{k}{2\pi(k-1)l}}(ke\beta)^l \tag{1}$$

We note that the chains will always increase, and we will have at least one chain with length $l+\mathsf{len}(\mathcal{C}')$; that means, we have $f(t, l) \geq 1$. From Equation 1, we have $\sqrt{\frac{k}{2\pi(k-1)l}}(ke\beta)^l \geq 1$; we can then obtain $ke\beta > 1$. Otherwise, $f(t, l) \ll 1$ for large enough l. Recall that $t = kl$; we now have $l < e\beta \cdot t$.

Next, we turn to the worst case discussion. With Chernoff bound, for any $\delta > 0$, we have $\Pr\left[l > (1+\delta)e\beta \cdot t\right] \leq -e^{\Omega(t)}$. Here $l = \mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}')$. We have $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') < e\beta \cdot t\right] \geq 1 - e^{-\Omega(t)}$ This concludes the proof. □

Defining β°. We use β° to denote the equivalent expected number of blocks that malicious players can extend a chain in a round. From Lemma 4.7, we have $\beta^\circ = e\beta$. That is, the amplification ratio of malicious players is bounded by e, the base of natural logarithm .

Defining α°. We next show lemmas stating that, honest players, by following the 2-distance-greedy strategy, can obtain extra advantage for extending the public chain.

Lemma 4.8. *Consider an execution of core-chain protocol $\Pi^{\text{core}\circ}$. Let $\mathcal{C}_{\text{best}}$ be the best chain at round r, and set $\ell := \mathsf{len}(\mathcal{C}_{\text{best}})$. Assume malicious players do not help honest players to extend any chain. Consider any chain $\mathcal{C}$ in round r where $\mathsf{len}(\mathcal{C}) = \ell - m$, and $\mathsf{distance}(\mathcal{C}_{\text{best}} \to \mathcal{C}) = d$, and $m + d >$ D. Assume the honest*

players follow the $\mathtt{D}$*-distance-greedy strategy to extend* $\mathcal{C}$ *at round* r *to* $\mathcal{C}'$ *at round* r' *where* $r' \geq r$*. Then* $\mathcal{C}'$ *cannot be selected as the best chain at round* r'*.*

Proof. Assume all players follow $\mathtt{D}$-distance-greedy strategy, and they extend chain $\mathcal{C}'$ from chain $\mathcal{C}$ at round $r' \geq r$, such that $\mathsf{len}(\mathcal{C}') = \ell - 1$. Hence, the $\mathsf{distance}(\mathcal{C}'_{\text{best}} \to \mathcal{C}') = d + \ell - 1 - (\ell - m) > \mathtt{D}$, where $\mathcal{C}'_{\text{best}}$ is the best chain at round r'.
Following $\Pi^{\text{core}\circ}$, honest players will not extend $\mathcal{C}'$. Thus, for any chain $\mathcal{C}'$, extended from $\mathcal{C}$, we have, $\mathsf{len}(\mathcal{C}') < \ell$. Therefore $\mathcal{C}'$ cannot be selected as the best chain. □

Lemma 4.9. *Consider core-chain protocol* $\Pi^{\text{core}\circ}$ *where greedy parameter is* $\mathtt{D} = 2$*. Assume malicious players do not help honest players to extend any chain. Let* t *be the number of rounds for extending the longest chain with* $\mathtt{D} + 1$ *new blocks. Then we have* $t \approx \frac{1.8}{\alpha}$ *on average.*

Proof. Consider that at round r', the longest chain is with length $l + 3$. This chain is extended by some players during the t rounds from the chains with length l. Note that, following the $\mathtt{D}$ distance greedy strategy, at a round $r + i$, where $1 \leq i \leq t$, there are $z_1 = \left(\sum_{i=1}^{j} 1 \cdot \alpha\right)$ number of chains with length $l + 1$. After this, we can compute the number of chains with length $l + 2$ as $z_2 = \left(\sum_{j=1}^{k} \left(\sum_{i=1}^{j} 1 \cdot \alpha\right) \cdot \alpha\right)$ at round $r + j$ where $i \leq j \leq t$. Let z_3 denote the number of chains with length $l + 3$.

We have,

$$z_3 = \sum_{k=1}^{t} \left(\sum_{j=1}^{k} \left(\sum_{i=1}^{j} 1 \cdot \alpha \right) \cdot \alpha \right) \cdot \alpha \tag{2}$$

Then we have $z_3 = \frac{(\alpha t)^3}{6}(1 + \frac{3}{2t} + \frac{1}{2t^2}) > \frac{(\alpha t)^3}{6}$.

If $\frac{(\alpha t)^3}{6} = 1$ then $z_3 > 1$. We have that there is a chain with length $l+3$ on average. That is, $\alpha t = \sqrt[3]{6} \approx 1.8$, and $t \approx \frac{1.8}{\alpha}$. □

Note that, now honest players by following 2-distance greedy strategy, in $t \approx \frac{1.8}{\alpha}$ rounds, we have a blockchain with length $l + 3$. If honest players follow the basic strategy by extending only the longest chain, we have a blockchian with length $l + 3$ in $t' \approx \frac{3}{\alpha}$ rounds. That means the honest stake application is now $\frac{t'}{t} = \frac{3}{1.8} = 1.65$

If we use α° to denote the equivalent expected number of blocks that the honest players will extend a chain in a round, we have $\alpha^\circ = 1.65\alpha$. That is, the amplification ratio (of extending chains by honest players) is $\mathtt{A}_2^\circ = 1.65$.

This lemma can be easily extended to the cases where $\mathtt{D} > 2$ by defining $z_4, z_5, \ldots$ similarly. For $\mathtt{D} = 3$, the amplification ratio is $\frac{4}{\sqrt[4]{24}} \approx 1.81$; For $\mathtt{D} = 4$, the amplification ratio is $\frac{5}{\sqrt[5]{120}} \approx 1.92$; For $\mathtt{D} = 5$, the amplification ratio is $\frac{6}{\sqrt[6]{720}} \approx 2$; For $\mathtt{D} = 6$, the amplification ratio is $\frac{7}{\sqrt[7]{5040}} \approx 2.07$; and in general we have $\mathtt{D} = n$, the amplification ratio is $\frac{n}{\sqrt[n]{n!}}$.

Lemma 4.10. *Consider core-chain protocol* $\Pi^{\text{core}\circ}$ *in the presence of an arbitrary adversary. Consider a core-chain* $\mathcal{C}'$ *in round* r'*; this core-chain is extended to chain* $\mathcal{C}''$ *in round* r''*, where* $r'' > r'$*. Let* $t = r'' - r'$*. Let* X *be the number of blocks that are generated by malicious players during the* t *rounds. Then for any* $\delta > 0$*, we have* $\Pr[X < (1 + \delta)\beta^\circ t] > 1 - e^{-\Omega(t)}$.

Proof sketch. First, consider the case that all the honest players follow the general-greedy strategy. From Lemma 4.7, we have $\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') < (1 + \delta)e(\alpha + \beta)t$ with overwhelming probability. Furthermore, the honest players, playing the same strategy as that by malicious players, will contribute more than $\frac{\alpha}{\alpha+\beta}(\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}'))(1 - \delta)$ blocks with overwhelming probability. We use Y to denote the number of blocks that are contributed by malicious players. We have $Y < (1 + \delta)e\beta t$ blocks with overwhelming probability.

Second, consider the case that the honest players follow the D-greedy strategy where D is a small constant such as 2. In this case the chain will grow slower than that in the previous case, on average. As a result, in any round, the malicious players will have less opportunity to extend the chain than that in the previous case. Therefore, we have $X < Y$. Putting these together, we have $\Pr[X < (1+\delta)\beta^\circ t] > 1 - e^{-\Omega(t)}$. □

Lemma 4.11. *Consider core-chain protocol $\Pi^{\text{core}\circ}$ in the presence of an adversary, $\alpha^\circ = \lambda\beta^\circ$, where $\lambda > 1$. Let $\mathcal{C}_{\text{best}}$ is the longest chain at round r, and set $l := \mathsf{len}(\mathcal{C}_{\text{best}})$. Consider chain $\mathcal{C}$ at round r, where its length is $(\ell - m)$ and $\mathsf{distance}(\mathcal{C}_{\text{best}} \to \mathcal{C}) > \mathtt{D}$. Assume chain $\mathcal{C}'$ at round r', where $r' > r$, is extended from chain $\mathcal{C}$. Then we have $\Pr\left[\mathcal{C}' = \mathcal{C}'_{\text{best}}\right] \leq e^{-\Omega(m)}$, where $\mathcal{C}'_{\text{best}}$ is the best chain at round r'.*

Proof. Assume that $\mathcal{C}'$, extended from $\mathcal{C}$, is the best chain at round r'. Let r^* where $(r < r^* \leq r')$, be the first round that $\mathcal{C}^*$ is extended from $\mathcal{C}$ by the adversary is the best chain. Let $t = r^* - r$, since the adversary extend at least $\mathtt{F}'$ blocks from round r to round r^*, $t = \Omega(m)$.
We have that,

$$\Pr\left[\mathsf{len}(\mathcal{C}^*) > \ell - m + (1+\delta)(\beta^\circ t)\right] \leq e^{-\Omega(m)}$$

$$\Pr\left[l^* < \ell + (1-\delta')(\alpha^\circ t)\right] \leq e^{-\Omega(m)}$$

Since $\alpha^\circ > \beta^\circ$ and δ, δ' are small, using union bound, we have,

$$\Pr\left[\mathsf{len}(\mathcal{C}^*) > l^*\right] < e^{-\Omega(m)}$$

Hence,

$$\Pr\left[\mathcal{C}' = \mathcal{C}'_{\text{best}}\right] \leq \Pr\left[\mathsf{len}(\mathcal{C}^*) > l^*\right] < e^{-\Omega(m)}$$

□

4.3.1 Honest majority for ensuring security of distance-greedy strategies

If we assume $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, we can obtain the following result (see Table 1): when honest players follow the 0-distance-greedy (or, 0-distance-greedy, 2-distance-greedy, 6-distance-greedy, respectively) strategy, and malicious players follow the 0-distance-greedy (or, arbitrary, arbitrary, respectively) strategy, to ensure the security of the core-chain protocol, 51% (or, 73%, 63%, 57% respectively) majority of stakes must be honest.

Table 1: Honest majority for ensuring security

Honest Players	Malicious Players	Honest Majority (ensuring security)
0-distance-greedy	0-distance-greedy	51%
0-distance-greedy	arbitrary	73%
2-distance-greedy	arbitrary	63%
6-distance-greedy	arbitrary	57%

4.3.2 Security properties in the presence of an arbitrary adversary

Based on the above discussions, we have $\alpha^\circ = \mathtt{A}_2^\circ\alpha \ll 1$ and $\beta^\circ = e\beta \ll 1$, where $\mathtt{A}_2^\circ = 1.6$. Here we assume, $\alpha \ll 1$ and $\beta \ll 1$. We have $\alpha^\circ \ll 1$ and $\beta^\circ \ll 1$. Then similarly, the security properties will still hold if we change the previous assumption of honest majority of stakes based on α and β, into the assumption of honest majority of *effective stakes* based on α° and β°.

Chain growth. Honest players will extend blockchain faster if they follow, not the basic strategy, but the D-greedy strategy, where $\mathtt{D} > 0$, and the chain growth rate will increase. From Lemma 3.2, we have:

Corollary 4.12 (Chain growth)**.** *Consider core-chain protocol $\Pi^{\text{core}\circ}$ in the presence of an arbitrary adversary. Consider an honest PoS-player P' with best local PoS core-chain $\mathcal{C}'$ in round r', and an honest PoS-player P'' with best local core-chain $\mathcal{C}''$ in round r'', where $r'' > r'$. Then we have $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$ where $t = r'' - r'$, $g = (1-\delta)\alpha^\circ$, and $\delta > 0$.*

Chain quality. A D-distance-greedy adversary can extend a chain faster than basic adversary, when $\mathtt{D} > 0$. Intuitively, this will reduce the chain quality. However, from Lemma 4.10, the number of blocks from malicious players on any chain is bounded. If we assume the honest players extend chains faster than the malicious players, the chain quality property will still hold as in Lemma 3.3.

Corollary 4.13 (Chain quality)**.** *Assume $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider an execution of core-chain protocol $\Pi^{\text{core}\circ}$ with an arbitrary adversary. Consider an honest PoS-player with PoS core-chain $\mathcal{C}$. Consider that ℓ consecutive block-cores of $\mathcal{C}$, where ℓ_{good} block-cores are generated by honest PoS-players. Then we have $\Pr\left[\frac{\ell_{good}}{\ell} \geq \mu\right] \geq 1 - e^{-\Omega(\ell)}$ where $\mu = 1 - (1+\delta)\frac{1}{\lambda}$.*

Common prefix. As discussed before, in our protocol, the honest players extend the chains via the D-distance-greedy strategy. No matter what strategy the adversary takes, the longest chain, say $\mathcal{C}$, will grow with the rate as stated in Corollary 4.12. In addition, in the protocol each honest player will take the longest chain in his view as the best chain, and then make efforts to extend the best set of chains defined by the longest chain and the D-distance-greedy strategy. Note that, following the D-distance-greedy strategy, the honest player can amplify his chain growth rate. As we discussed before, in our analysis, we will view the set of chains as a *virtual* chain. Equivalently, each honest player will extend a single virtual chain with the amplified chain growth rate in each round. We remark that, this will not harm the following analysis of common prefix property: the definition of D-distance-greedy strategy will enforce all chains in the set are very close to the longest chain.

Next, to facilitate our discussions, we introduce a notation for calculating the number of rounds of support from honest players, for a chain.

Definition 4.14. *Consider a chain $\mathcal{C}$ in a protocol execution. If there is at least one honest player has made attempts to extend $\mathcal{C}$ at round r, then we say chain $\mathcal{C}$ receives 1 round of support from honest players, and we write $\mathtt{support}_{\mathcal{C}}(r) = 1$. Otherwise, if no honest player made any attempts to extend $\mathcal{C}$ at round r, then we write $\mathtt{support}_{\mathcal{C}}(r) = 0$.*

Similarly, for a time window, from round r to round r', where $r' > r$, the total number of rounds of support that chain $\mathcal{C}$ receives from honest players can be written as $\mathtt{support}_{\mathcal{C}}(r, r') = \sum_{i=r}^{r'} \mathtt{support}_{\mathcal{C}}(i)$.

For a protocol execution with chain $\mathcal{C}$, it is easy to see that $\mathtt{support}_{\mathcal{C}}(r, r') \leq r' - r$. In the setting that there is no network delay, all honest players will have the same chain as the best chain in each round. This means in each round, all the honest players will make attempts to extend the *same* chain. Consider two divergent chains, $\mathcal{C}$ and $\hat{\mathcal{C}}$, in the protocol execution, We have $\mathtt{support}_{\mathcal{C}}(r, r') + \mathtt{support}_{\hat{\mathcal{C}}}(r, r') \leq r' - r$

In the setting there is network delay, honest players may make attempts to extend different chains. More explicitly, when a new block is generated, since the adversary is allowed to delay the new block for Δ rounds, honest players may make attempts to extend different chains for Δ rounds. In any successive $r' - r$ rounds, the longest chain will increase $(\alpha^\circ + \beta^\circ)(r' - r)$ blocks on average. That is the honest players will make attempts to extend multiple chains for $(\alpha^\circ + \beta^\circ)\Delta(r' - r)$ round on average. Recall that we assume $\alpha^\circ + \beta^\circ \ll 1$. In the setting with network delay Δ, two divergent chains, $\mathcal{C}$ and $\hat{\mathcal{C}}$, we have $\mathtt{support}_{\mathcal{C}}(r, r') + \mathtt{support}_{\hat{\mathcal{C}}}(r, r') \leq (1 + c)(r' - r)$, where $c = (\alpha^\circ + \beta^\circ)\Delta$ is a small constant.

Claim 4.15. *Consider an execution of core-chain protocol $\Pi^{\text{core}\circ}$. Consider two divergent chains, $\mathcal{C}$ and $\mathcal{C}'$ from round r to round r'. Then at least one chain that receives support from honest players, less than $\frac{1+c}{2}(r' - r)$ rounds.*

Proof. Consider the chains $\mathcal{C}$ and $\mathcal{C}'$, we have $\texttt{support}_{\mathcal{C}}(r, r') + \texttt{support}_{\mathcal{C}'}(r, r') \leq (1+c)(r'-r)$. Without loss of generality, let $\texttt{support}_{\mathcal{C}}(r, r') \leq \texttt{support}_{\mathcal{C}'}(r, r')$. Then we immediately have $\texttt{support}_{\mathcal{C}}(r, r') \leq \frac{1+c}{2}(r'-r)$. □

Lemma 4.16. *Assume $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\circ}$ with an arbitrary adversary. Consider a core-chain $\mathcal{C}$ for an honest player P at round r. This chain $\mathcal{C}$ has been extended for successive rounds from round r to round r'. Then the probability that $\mathcal{C}$ is the best chain for player P at round r', is less than $e^{-\Omega(u)}$, where $u = (r'-r) - \texttt{support}_{\mathcal{C}}(r, r')$.*

Proof. Let $\hat{\mathcal{C}}$ be the longest chain for each round in the protocol execution. During the time window, from round r to round r', the chain $\mathcal{C}$ receives supports for $\texttt{support}_{\mathcal{C}}(r, r')$ number of rounds, and during these rounds, chain $\mathcal{C}$ is the longest chain, i.e., $\hat{\mathcal{C}} = \mathcal{C}$. The chain $\mathcal{C}$ does not receive supports from honest players for u rounds, where $u = (r'-r) - \texttt{support}_{\mathcal{C}}(r, r')$, and during these u rounds, $\mathcal{C}$ is shorter than the longest chain $\hat{\mathcal{C}}$, i.e., $\mathsf{len}(\hat{\mathcal{C}}) > \mathsf{len}(\mathcal{C})$.

We note that, during these u rounds, honest players make attempts to extend the chain $\hat{\mathcal{C}}$, while the adversary may make attempts to extend chain $\mathcal{C}$. During the u rounds, chain $\mathcal{C}$ can be extended for $\beta^\circ u$ blocks on average while $\hat{\mathcal{C}}$ will be extended by honest players for $\alpha^\circ u$ blocks on average.

Let L be the number of blocks that $\mathcal{C}$ is extended in the u rounds. Let δ be any small constant, with the Chernoff bounds, we have

$$\Pr[L < (1+\delta)\beta^\circ u] > 1 - e^{-\Omega(u)}$$

Let $\hat{L}$ be the number of blocks that $\hat{\mathcal{C}}$ is extended in the u rounds. Let $\hat{\delta}$ be any small constant, with the Chernoff bounds, we have

$$\Pr[\hat{L} > (1-\delta)\alpha^\circ u] > 1 - e^{-\Omega(u)}$$

From the assumption, we have $\lambda > 1$ and $\alpha^\circ = \lambda\beta^\circ$. Putting all together and by choosing δ and $\hat{\delta}$, we have that $\Pr[\hat{L} > L] > 1 - e^{-\Omega(u)}$.

Note that chain $\hat{\mathcal{C}}$ is longer than, or identical to the chain $\mathcal{C}$ at round r. Now we have that the chain $\hat{\mathcal{C}}$ is longer than $\mathcal{C}$ at round r' with probability no less than $1 - e^{-\Omega(u)}$. This concludes the proof. □

Lemma 4.17 (Common prefix)**.** *Assume $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider an execution of core-chain protocol $\Pi^{\text{core}\circ}$ with an arbitrary adversary. Consider two honest players, P in round r with the local best core-chain $\mathcal{C}$, and $\hat{P}$ in round r' with the local best core-chain $\hat{\mathcal{C}}$, respectively, where $r' \geq r$. Then we have $\Pr\left[\mathcal{C}[\neg\kappa] \preceq \hat{\mathcal{C}}\right] \geq 1 - e^{-\Omega(\kappa)}$.*

Proof. Consider chain $\hat{\mathcal{C}}$, and We use $\hat{\mathcal{C}}^r$ to denote the portion of the chain $\hat{\mathcal{C}}$ from the beginning to round r. Let r_0 be the round that the last common block of chains $\mathcal{C}$ and $\hat{\mathcal{C}}^r$, is generated. Set $t = r - r_0$.

Let $u = (r-r_0) - \texttt{support}_{\mathcal{C}}(r_0, r)$ and $\hat{u} = (r-r_0) - \texttt{support}_{\hat{\mathcal{C}}^r}(r_0, r)$. From the assumption, $\mathcal{C}$ is the best chain at round r for player P, and $\hat{\mathcal{C}}$ is the best chain at round r' for player $\hat{P}$. With the lemma 4.16, we have that $\mathcal{C}$ is the best chain with probability no more than $e^{-\Omega(u)}$ and $\hat{\mathcal{C}}$ is the best chain with probability no more than $e^{-\Omega(\hat{u})}$.

From the Claim 4.15 we have $u > \frac{1-c}{2}(r-r_0)$ or $\hat{u} > \frac{1-c}{2}(r-r_0)$. Without loss of generality, we assume $u > \frac{1-c}{2}(r-r_0)$. If $r - r_0 > \Theta(\kappa)$, we have $\mathcal{C}$ is the best chain with probability no more than $e^{-\Omega(\kappa)}$. This contradicts with that $\mathcal{C}$ is the best chain for player P. Thus we have, $r - r_0 \leq \Theta(\kappa)$ with probability no less than $1 - e^{-\Omega(\kappa)}$. The divergent length of $\mathcal{C}$ from $\hat{\mathcal{C}}$ is less than $(\alpha^\circ + \beta^\circ)\Theta(\kappa) = \Theta(\kappa)$ blocks with probability no less than $1 - e^{-\Omega(\kappa)}$. This completes the proof. □

Chain soundness. As in Lemma 3.5, the protocol can achieve chain soundness property. Otherwise the common prefix property will be violated.

Corollary 4.18 (Chain soundness)**.** *Assume for every round, $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider an execution of core-chain protocol $\Pi^{\text{core}\circ}$ with an arbitrary adversary. Consider two honest players, P' and P'' in round r, with the local best core-chains $\mathcal{C}'$ and $\mathcal{C}''$, respectively, where P' is a new player and P'' is an existing player in round r. Then we have $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ and $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$.*

5 Defending against an adaptive-registration adversary

In previous sections (Sections 3 and 4), we assume that all players have their stakes registered, before they join the protocol execution. Let's provide a brief explanation below. Note that the process of extending the chains is based on the hash inequality $\mathsf{H}(\mathit{context}, \mathit{solution}) < \mathtt{T}$, where $\mathit{solution}$ is in the form of (PK, σ). Since the adversary now knows the $\mathit{context}$, he can play a "rejection re-sampling" strategy to generate their keys adaptively. More concretely, the adversary first runs key generation algorithm to obtain a key-pair (PK, SK), and then check if the corresponding (PK, σ) is a valid solution to the hash inequality; if not, the adversary will repeat the process. By adopting this strategy, malicious players can significantly increase the probability that they are chosen to extend the chains. In this section, we propose new ideas to address the rejection re-sampling attack; as a result, we can allow the players to have their stakes registered *during the protocol execution*.

5.1 Setup functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$

Our key idea to defend against the rejection re-sampling attack, is that, the players must have had their stakes registered, a certain number of rounds before they are allowed to participate in the process of extending the chains. To achieve this goal, in our core-chain protocol design, we need use a modified resource certification functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ in which all registration information must be carefully recorded. We remark that, it is non-trivial to formalize the registration bookkeeping. In a real world blockchain protocol, the registration information will be stored in a block of the blockchain. Let $\mathbb{C}$ be all chains in the protocol execution, and let $\mathcal{T}$ be a tree to store the information for the players in the set $\mathcal{P}$. The root of $\mathcal{T}$ is corresponding to the genesis block of the blockchain. A node on the tree is mapped from a block on a blockchain. When a player is selected for generating a block, a corresponding node will be recorded in the tree. It is easy to see that a `path` in the tree from root to a node, corresponds to a chain in the chain set $\mathbb{C}$ in the real world protocol execution. The details of the modified functionality can be found in Figure 10.

As defined in $\mathcal{F}_{\mathsf{rCERT}}$ earlier, the modified functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ consists of three phases, "Stake Resource Registration", "Stake Election" and "Stake Verification". At any time, a PoS-player P can send a registration command $(\text{STAKE-REGISTER}, P, \mathcal{C})$ to functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ for registration where $\mathcal{C}$ is the specified blockchain that P to be registered. The functionality check that if P is registered on chain $\mathcal{C}$ or not. Please note that, P will be registered in the new generated leaf node of $\mathcal{C}$ and will also be mapped on the tree $\mathcal{T}$. If P is not registered on $\mathcal{C}$, the functionality generate a key pair $(\mathsf{sk}_P, \mathsf{pk}_P) \leftarrow \mathsf{uKeyGen}(1^\kappa)$, and records (P, pk_P). Then, for each execution round, a registered PoS-player P is granted *one unit of the stake*, and he can then request the functionality for leader election in this round. Just like in $\mathcal{F}_{\mathsf{rCERT}}$, this is processed in "Stake Election" phase. The difference here is that a player will be verified more carefully to prevent adaptive registration. Intuitively, only a player who registers on the prefix of a chain can be elected to extend a block on this chain. This verification is abstracted as subroutine $\mathsf{StakeVerify}_\eta$ (in Figure 11).

Finally, the block verification request $(\text{CORE-VERIFY}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma, h, \mathcal{C}, i)$ will proceed as follows. In the request, the functionality will verify if the i-th block on the $\mathcal{C}$ is valid. Upon receiving a verification request, the functionality would check if there is a corresponding path on the $\mathcal{T}$ with $\mathcal{C}$. The functionality then verifies if the signature is valid for the block. Then the functionality will check if the player is registered

FUNCTIONALITY $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$

The functionality interacts with a set $\mathcal{P}$ of parties, and an adversary. The functionality is parameterized by a difficulty parameter p, a security parameter κ, a stake registration bound η, a tree $\mathcal{T}$, and a unique digital signature scheme $\Sigma = (\mathsf{uKeyGen}, \mathsf{uSign}, \mathsf{uVerify})$.
Initially, a set $\mathcal{P}_0$ of distinct players are registered, where $\mathcal{P}_0 \subseteq \mathcal{P}$; That is, for each $P \in \mathcal{P}_0$, compute $(\mathsf{sk}_P, \mathsf{pk}_P) \leftarrow \mathsf{uKeyGen}(1^\kappa)$, record the tuple $(P, \mathsf{pk}_P, \mathsf{sk}_P)$, send (P, pk_P) to the adversary, and party P. The records (P, pk_P) are stored at the root of the tree $\mathcal{T}$

Stake Resource Registration.

1. Upon receiving a message $(\text{STAKE-REGISTER}, P, \mathcal{C})$ from party $P \in \mathcal{P}$, retrieve $\mathtt{path}$ corresponding with $\mathcal{C}$. If there is an entry (P, pk_P) with location $\mathtt{path}'$ in the tree $\mathcal{T}$, and $\mathtt{path}' \prec \mathtt{path}$, then ignore the message. Otherwise, compute $(\mathsf{sk}_P, \mathsf{pk}_P) \leftarrow \mathsf{uKeyGen}(1^\kappa)$, record the tuple $(P, \mathsf{pk}_P, \mathsf{sk}_P)$, send $(\text{STAKE-REGISTERED}, P, \mathsf{pk}_P)$ to the adversary, and party P. Record (P, pk_P) with location $\mathtt{path}$ in the tree $\mathcal{T}$. (the party P registered.)

2. Upon receiving message $(\text{RETRIEVE}, P)$ from party $P' \in \mathcal{P}$, if there is a recorded tuple $(P, \mathsf{sk}_P, \mathsf{pk}_P)$, then output $(\text{RETRIEVED}, P, \mathsf{pk}_P)$ to P'. Else output $(\text{RETRIEVED}, P, \perp)$ to P'.

The STAKE-REGISTERED will take effect only after a block is generated on the $\mathcal{C}$.

Stake Election:
For each round,

1. Set $\mathtt{B}_P := 0$ for every registered party $P \in \mathcal{P}$.
2. Upon receiving $(\text{ELECT}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \mathcal{C})$ from party P, proceed:
3. Retrieve $\mathtt{path}$ corresponding with $\mathcal{C}$ and set $l := \mathsf{len}(\mathcal{C})$.
4. Set $\mathsf{b} := 0$. (the party P is not elected by default)
   (a) If $\mathsf{StakeVerify}_\eta(\mathtt{path}, P, l) = 1$ and $\mathtt{B}_P = 0$
   - compute $\sigma := \mathsf{uSign}(\mathsf{sk}_P, \langle h^{\text{prev}}, \mathtt{round}\rangle)$,
     and verify that $\mathsf{uVerify}(\mathsf{pk}_P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma) = 1$.
     if verified, then choose a random tag h;
     set $\mathsf{b} := 1$ with probability p; and set $\mathsf{b} := 0$ otherwise.
     (the party P is elected with probability p);
     record $\langle h^{\text{prev}}, \mathtt{round}, P, \sigma, h, \mathsf{b}\rangle$.
5. Set $\mathtt{B}_P := 1$
6. Output $(\text{ELECTED}, P, h, \sigma, \mathsf{b})$ to P and the adversary.

Block Verification:

1. Upon receiving $(\text{CORE-VERIFY}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma, h, \mathcal{C}, i)$ from party $P' \in \mathcal{P}$,
   retrieve $\mathtt{path}$ corresponding with $\mathcal{C}$.
2. If $\mathsf{StakeVerify}_\eta(\mathtt{path}, P, i) = 1$,
   - If $\mathtt{path}[i] = \langle h^{\text{prev}}, \mathtt{round}, P, \sigma, h, 1\rangle$, then set $f := 1$.
3. Output $(\text{CORE-VERIFIED}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma, h, f)$ to the party P'.

Figure 10: Resource certification functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$.

long enough before the block is generated. We use the parameter η to denote that the account is registered η blocks before.

SUBROUTINE $\mathsf{StakeVerify}_{\eta}$

This subroutine verifies if a player is valid with input $\langle \mathtt{path}, P, \ell \rangle$.
Set $f := 0$

1. If there exists a record of the form (P, pk_P) on the $\mathtt{path}$ which is located on the j-th block where $j < \ell - \eta$,
   set $f := 1$.

Output f.

Figure 11: Stake verification subroutine $\mathsf{StakeVerify}_{\eta}$.

5.2 The modified core-chain protocol $\Pi^{\text{core}\bullet}$

In order to prevent the malicious players from having their stakes registered adaptively, the protocol is modified as:

- A player is qualified to be elected to sign a block in the functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ (see Figure 10), only if his key pair has been registered at least η blocks earlier.
- For two divergent chains, $\mathcal{C}$ and $\mathcal{C}'$, if $\max(\mathsf{distance}(\mathcal{C}' \to \mathcal{C}), \mathsf{distance}(\mathcal{C} \to \mathcal{C}')) > \eta$, then whichever chain that generates the η blocks first is the better one.

Now malicious players cannot register a biased key pair for the following η blocks to increase the probability he will be elected. However, there is still an issue that the malicious players may register a biased key pair for a round, η blocks later. We will show this issue can be resolved if we further improve the best chain strategy. The intuition is that if the malicious players prepare a biased key pair for a public chain, then the honest player will win some blocks among the η blocks with high probability. The malicious players cannot predict the signature of honest players, so he cannot predict the input of the blocks η blocks after. This means that the malicious players cannot get advantage for η blocks after if the chain is public. If the malicious players decide to extend a hidden blockchain, he can prepare a biased player for a block η blocks after. However, he will lose the chain growth competition for the first η blocks. Hence, we modify the $\mathtt{D}$-$\mathsf{BestCore}^{\circ}$ into $(\mathtt{D}, \eta)$-$\mathsf{BestCore}^{\bullet}$ as in Figure 13. Finally, we note that the security proof can be found in Supplementary material 5.3.

PROTOCOL $\Pi^{\text{core}\bullet}$

Initially, a set $\mathcal{P}_0$ of players are registered to $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$.

Upon receiving message $(\text{REGISTER-STAKE}, P)$ from the environment $\mathcal{Z}$ at round $\mathtt{round}$,
Let $\mathbb{C}$ be the set of core-chains collected from $\mathcal{F}_{\mathsf{NET}}$.
For each $\mathcal{C} \in \mathbb{C}$, send $(\text{STAKE-REGISTER}, P, \mathcal{C})$ to functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$.

Upon receiving message $(\text{INPUT-STAKE}, P)$ from the environment $\mathcal{Z}$ at round $\mathtt{round}$, the PoS-player $P \in \mathcal{P}$, proceeds as follows.

1. *Select the best local PoS core-chain:*

   Let $\mathbb{C}$ be the set of core-chains collected from $\mathcal{F}_{\mathsf{NET}}$.

   Compute $\mathbb{C}_{\text{best}} := (\mathtt{D}, \eta)\text{-}\mathsf{BestCore}^{\bullet}(\mathbb{C}, \mathtt{round})$. ($(\mathtt{D}, \eta)$-$\mathsf{BestCore}^{\bullet}$ will return a best chain set)

2. *Attempt to extend PoS core-chain:*

   For each $\mathcal{C} \in \mathbb{C}_{\text{best}}$, do: (Each player will try to extend all chains in the best chain set)

   - Set $\ell := \mathsf{len}(\mathcal{C})$.
   - Parse $\mathcal{C}[\ell]$ as $\langle\langle h_\ell^{\text{prev}}, \mathtt{round}_\ell, P_\ell, \sigma_\ell\rangle, h_\ell\rangle$.
   - *Stake election:* Send $(\text{ELECT}, P, \langle h_\ell, \mathtt{round}\rangle, \mathcal{C})$ to functionality $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$, and receive $(\text{ELECTED}, P, h_{\ell+1}, \sigma, \mathsf{b})$ from $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$.
   - *If* $\mathsf{b} = 1$, *generate a new block-core:* Set the new block-core $B := \langle\langle h_\ell, \mathtt{round}, P, \sigma\rangle, h_{\ell+1}\rangle$, and set $\mathcal{C} := \mathcal{C} \| B$, and $state := state \cup \{\mathcal{C}\}$, and then send $(\text{BROADCAST}, \mathcal{C})$ to $\mathcal{F}_{\mathsf{NET}}$.

   Return $(\text{RETURN-Stake}, P)$ to the environment $\mathcal{Z}$.

Figure 12: Our proof-of-stake core-chain protocol $\Pi^{\text{core}\bullet}$ in the $\{\mathcal{F}^{\bullet}_{\mathsf{rCERT}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid model. (See Figure 13 for the subroutine $(\mathtt{D}, \eta)$-$\mathsf{BestCore}^{\bullet}$.)

5.3 Security analysis

In previous section, the security properties of protocol $\Pi^{\text{core}\circ}$ have be proven under the assumption of honest majority of *effective stakes* based on α° and β°. Now, under the same assumption, we will show that our modified core-chain protocol $\Pi^{\text{core}\bullet}$ can also achieve the same properties. Please note that our new adversary is stronger since there is no restriction on how players are registered with respect to the protocol execution. More concretely, we have the following theorem:

Theorem 5.1 (Theorem 1.4, restated)**.** *Consider core-chain protocol $\Pi^{\text{core}\bullet}$ where honest players follow the 2-greedy strategy while adversarial players follow the general-greedy strategy. If $\alpha^{\circ} = \lambda\beta^{\circ}$, $\lambda > 1$, then the protocol $\Pi^{\text{core}\bullet}$ can achieve chain growth, chain quality, common prefix and chain soundness properties.*

5.3.1 Important lemmas

In our modified protocol $\Pi^{\text{core}\bullet}$, malicious players cannot register key pairs so that they can extend the chains immediately. What the malicious players can do, however, is to register biased key pairs now, and then try to extend the chains many rounds later. We will prove that malicious players cannot obtain additional advantage by playing this strategy. First, we will show that the probability that malicious players are able to predict the latest block of the best public chain is negligible.

Lemma 5.2. *Let chain $\mathcal{C}$ be the best valid public chain with length ℓ in round r. Suppose the length of best valid public chain $\mathcal{C}'$ will be $\ell + \eta$ in round $r + t$. The probability that the malicious players predict the last block on chain $\mathcal{C}'$ in round r is $e^{-\Omega(\eta)}$ at most in round r.*

Proof. From chain quality property, we know that the honest players will contribute blocks in the last η blocks with probability no less than $1 - e^{-\Omega(\eta)}$. Blocks generated by honest players are unpredictable for malicious

SUBROUTINE $(\mathtt{D}, \eta)$-BestCore$^\bullet$

The subroutine $(\mathtt{D}, \eta)$-BestCore$^\bullet$ has the access to $\mathcal{F}^\bullet_{\mathsf{rCERT}}$.
Input: $(\mathbb{C}', \mathtt{round}')$.
Output: $\mathbb{C}_{\mathrm{best}}$.

Verify the chain set
For every chain $\mathcal{C} \in \mathbb{C}'$, and proceed as follows.

- Set $\ell := \mathsf{len}(\mathcal{C})$.
- For i from ℓ down to 1, verify block-core $\mathcal{C}[i]$, as follows.
  - Parse $\mathcal{C}[i]$ as $\langle\langle h_i^{\mathrm{prev}}, \mathtt{round}_i, P_i, \sigma_i\rangle, h_i\rangle$.
    Parse $\mathcal{C}[i-1]$ as $\langle\langle h_{i-1}^{\mathrm{prev}}, \mathtt{round}_{i-1}, P_{i-1}, \sigma_{i-1}\rangle, h_{i-1}\rangle$.
  - If $\mathtt{round}_i < \mathtt{round}'$ and $\mathtt{round}_{i-1} < \mathtt{round}_i$, execute:
    - If $h_i^{\mathrm{prev}} \neq h_{i-1}$, remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.
    - Else send $(\textsc{Core-Verify}, P_i, \langle h_i^{\mathrm{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i, \mathcal{C}, i)$ to $\mathcal{F}^\bullet_{\mathsf{rCERT}}$.
      Upon receiving message $(\textsc{Core-Verified}, P_i, \langle h_i^{\mathrm{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i, f_i)$ from $\mathcal{F}^\bullet_{\mathsf{rCERT}}$, if $f_i = 0$ remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.

    Otherwise, remove the core-chain $\mathcal{C}$ from $\mathbb{C}'$.

Identify the best chain
Set $\mathcal{C}_{\mathrm{best}} := \emptyset$
For every chain $\mathcal{C} \in \mathbb{C}'$ do:

- If $\mathcal{C}_{\mathrm{best}} = \emptyset$ then $\mathcal{C}_{\mathrm{best}} := \mathcal{C}$
- Set $d := \mathsf{distance}(\mathcal{C} \to \mathcal{C}_{\mathrm{best}})$. Set $d' := \mathsf{distance}(\mathcal{C}_{\mathrm{best}} \to \mathcal{C})$
  - If $d < \eta$ or $d' < \eta$
    - If $d < d'$ then $\mathcal{C}_{\mathrm{best}} := \mathcal{C}$
  - If $d \geq \eta$ and $d' \geq \eta$
    - If block $\mathcal{C}[\eta]$ is generated earlier than block $\mathcal{C}_{\mathrm{best}}[\eta]$, then set $\mathcal{C}_{\mathrm{best}} := \mathcal{C}$.

Identify the best chain set
Set $\mathbb{C}_{\mathrm{best}} := \emptyset$.
For any chain $\mathcal{C} \in \mathbb{C}'$

- If $\mathsf{distance}(\mathcal{C}_{\mathrm{best}} \to \mathcal{C}) \leq \mathtt{D}$ then set $\mathbb{C}_{\mathrm{best}} := \mathbb{C}_{\mathrm{best}} \cup \{\mathcal{C}\}$.

Then return $\mathbb{C}_{\mathrm{best}}$ as the output.

Figure 13: The core-chain set validation subroutine $(\mathtt{D}, \eta)$-BestCore$^\bullet$.

players. We have that the malicious players cannot predict any block from honest players before it is published. Furthermore, he cannot predict the last block of $\mathcal{C}'$ in round r if there is a honest block on chain $\mathcal{C}'$ at last. We get the conclusion that the malicious players predict the last block of chain $\mathcal{C}'$ in round r is at most $e^{-\Omega(\eta)}$. □

If malicious players cannot predict the last block of the best chain, then he cannot have a biased key pair registered so that the corresponding stakeholder can be chosen in a future round with much higher probability. From Lemma 5.2, we conclude that a malicious player, by playing adaptive key registration strategy, cannot improve the probability that he is chosen for extending the public chain. Next, we will show that the malicious

players cannot gain advantage, by playing this adaptive strategy, for extending a private (hidden) chain.

Lemma 5.3. *Assume the malicious players fork a hidden chain $\mathcal{C}$ from a block during t rounds. Let $\ell_{hidden} < \eta$ be the length of the forked hidden chain $\mathcal{C}$. Assume the length of public best chain increase ℓ_{public} blocks during this t rounds. We have* $\Pr[\ell_{public} > \ell_{hidden}] > 1 - e^{-\Omega(\ell_{hidden})}$.

Proof. From the modified protocol, we also know that the adaptive key generation will not affect the first η blocks. It is said that the adaptive strategy will not help the malicious players to shorten the rounds for the first η blocks. The probability that $\ell_{hidden} > \ell_{public}$ equals to the probability that the malicious players break common prefix property. From Corollary 4.17, we have $\Pr[\ell_{public} > \ell_{hidden}] > 1 - e^{-\Omega(\ell_{hidden})}$. □

From the modified protocol, we know that if there are two divergent chains, the honest players will compare the first η different blocks. Suppose η is large enough. From Lemma 5.3, we know that the hidden chain will not be accepted with overwhelming probability, even the adversary follows the adaptive key registration strategy. That means, the adaptive key registration strategy is not helpful for extending the hidden chain.

5.3.2 Security properties in the presence of an adaptive registration

Chain growth. Honest players in protocol $\Pi^{\text{core}\bullet}$ will extend the chains in the same way as that in protocol $\Pi^{\text{core}\circ}$. From Corollary 4.12 we have:

Corollary 5.4 (Chain growth)**.** *Consider core-chain protocol $\Pi^{\text{core}\bullet}$ in the presence of an arbitrary adversary who is allowed to register to the system adaptively. Consider an honest PoS-player P' with best local PoS core-chain $\mathcal{C}'$ in round r', and an honest PoS-player P'' with best local core-chain $\mathcal{C}''$ in round r'', where $r'' > r'$. Then we have* $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$, *where $t = r'' - r'$, $g = (1-\delta)\alpha^\circ$, and $\delta > 0$.*

Chain quality. From Lemma 5.3 and 5.2, the adversary cannot obtain additional advantage by playing the adaptive strategy. That is, they cannot produce more blocks by adaptively selecting key-pairs and having their stakes registrated. From Corollary 4.13 we have:

Corollary 5.5 (Chain quality)**.** *Consider $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\bullet}$ in the presence of an arbitrary adversary who is allowed to register to the system adaptively. Consider an honest PoS-player with PoS core-chain $\mathcal{C}$. Consider that ℓ consecutive block-cores of $\mathcal{C}$, where ℓ_{good} block-cores are generated by honest PoS-players. Then we have* $\Pr\left[\frac{\ell_{good}}{\ell} \geq \mu\right] \geq 1 - e^{-\Omega(\ell)}$, *where $\mu = 1 - (1+\delta)\frac{1}{\lambda}$.*

Common prefix. Again, from Lemma 5.3 and Lemma 5.2, the adversary cannot obtain extra benefit by playing the adaptive strategy. They cannot produce more blocks by adaptively selecting key pairs. From Corollary 4.17 we have:

Corollary 5.6 (Common prefix)**.** *Consider $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\bullet}$ in the presence of an arbitrary adversary who is allowed to register to the system adaptively. Consider two honest PoS-players, P in round r and P' in round r', with the local best PoS core-chains $\mathcal{C}$, $\mathcal{C}'$, respectively, where $r' \geq r$. Then we have* $\Pr\left[\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'\right] \geq 1 - e^{-\Omega(\kappa)}$.

Chain soundness. As in Corollary 4.18, we can achieve the chain soundness property because we use *the longest chain is the best chain* policy. Otherwise, if a new player accepts a chain which is not the best public chain, this chain will also be accepted by existing players. The common prefix property will be violated.

Corollary 5.7 (Chain soundness)**.** *Consider for every round, $\alpha^\circ = \lambda\beta^\circ$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\bullet}$ in the presence of an arbitrary adversary who is allowed to register to the system adaptively. Consider two honest PoS-players, P' and P'' in round r, with the local best core-chains $\mathcal{C}'$ and $\mathcal{C}''$, respectively, where P' is a new player and P'' is an existing player in round r. Then we have $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ and $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$.*

6 From core-chain to blockchain

In this section, we start to extend the core-chain protocol in Section 5 to a blockchain protocol in which payload (transactions) will be included. We want to emphasize that the payload cannot be included into the core block directly. If the payload is in the core block, the malicious players may try to brute-force different payloads to obtain the solution that satisfies the hash inequality. Furthermore, the scheme must guarantee that a malicious player cannot change the payload he signed before.

Intuitively, the core-chain can be viewed as a (biased) random beacon; we can use the beacon to select a PoS-player to generate a new block with payload. The block with payload will also be linked as a hash chain which is called main blockchain. More concretely, once a new block-core B_{i+1} is generated by a PoS-player (in the blockchain protocol), then the PoS-player is selected to generate the new block $\tilde{B}_{i+1}$, in the following format $\tilde{B}_{i+1} = \langle \mathsf{hash}(\tilde{B}_i), B_{i+1}, \tilde{X}_{i+1}, \tilde{\mathrm{PK}}, \tilde{\sigma} \rangle$ where $\tilde{\sigma} \leftarrow \mathsf{Sign}_{\tilde{\mathrm{SK}}}(\tilde{h}_i, B_{i+1}, \tilde{X}_i)$ and $\tilde{h}_i := \mathsf{hash}(\tilde{B}_i)$, and $B_{i+1} := \langle h_i, \mathtt{round}, \mathrm{PK}, \sigma \rangle$. $\tilde{X}_{i+1}$ is payload. Here we note that in our blockchain protocol design, the PoS-player holds two combined pairs of keys, $(\mathrm{SK}, \mathrm{PK})$ of the strengthened unique signature scheme $(\mathsf{uKeyGen}, \mathsf{uSign}, \mathsf{uVerify})$, and $(\tilde{\mathrm{SK}}, \tilde{\mathrm{PK}})$ of a regular[3] digital signature scheme $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$. Now the blocks in the main blockchain are "glued" with the block-cores in the blockchain, and we can reduce the security of the blockchain protocol to the security of the blockchain protocol.

In the formal description of our blockchain protocol below, we will use a slightly augmented setup functionality $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ to capture the hash inequality and the block and block-core signing/verification. Similarly, this setup functionality can be implemented by using hash function $\mathsf{H}(\cdot)$ and a digital signature scheme.

6.1 Setup functionality $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$

In our blockchain protocol design, we will use the setup functionality, $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ (in Figure 14), which is an augmented version of the resource certification functionality in Section 5.1. The first two phases, "Stake Resource Registration" and "Stake Election", remain the same. A new phase, "Signature Generation", is defined to be utilized to sign a main block. And "Block Verification" is extended to verify main block.

6.2 Main blockchain protocol

We now describe our PoS based blockchain protocol Π^{main}. The blockchain protocol can be viewed as an augmented version of the core-chain protocol in Section 5.2, and now it uses the augmented resource certificate functionality $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ as setup functionality. The functionality $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ has the same phases as of $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$. The difference is $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ will provide extra service for main block signature and verification. The details can be found in Section 6.1.

As in the core-chain protocol, for each PoS-player P, once activated by the environment on $(\textsc{Input-Stake}, P)$ at a round, and received a blockchain set $\tilde{\mathbb{C}}$ from $\mathcal{F}_{\mathsf{NET}}$, the party P finds the best valid blockchain $\tilde{\mathcal{C}}_{\mathrm{best}}$ by running the subroutine $\mathsf{BestMain}$ (in Figure 16), and then updates its local blockchain $\tilde{\mathcal{C}} := \tilde{\mathcal{C}}_{\mathrm{best}}$. Note that, the i-th block in blockchain $\tilde{\mathcal{C}}$, is in the following format $\tilde{B}_i := \langle \langle \tilde{h}_{i-1}, B_i, \tilde{X}_i \rangle, P_i, \tilde{\sigma}_i \rangle$. That means, from $\tilde{B}_i$, we can obtain the i-th block-core B_i. We thus can derive the core-chain $\mathcal{C}$ from the blockchain $\tilde{\mathcal{C}}$. If the PoS-player P is selected (with certain probability p), he can query the functionality to generate a signature σ for $context := \langle h_\ell, \mathtt{round}_{\ell+1} \rangle$. Then he defines a new block-core $B_{\ell+1} := \langle \langle h_\ell, \mathtt{round}_\ell \rangle, P, \sigma \rangle$, updates his local core-chain $\mathcal{C}$. Once the new block-core $B_{\ell+1}$ is generated, the PoS-player P can query the functionality to generate a signature $\tilde{\sigma}$ for $msg := \langle \tilde{h}_\ell, B_{\ell+1}, \tilde{X}_{\ell+1} \rangle$. Then he can define a new block $\tilde{B}_{\ell+1} := \langle \langle \tilde{h}_\ell, B_{\ell+1}, \tilde{X}_{\ell+1} \rangle, P, \tilde{\sigma} \rangle$, and update his local blockchain $\tilde{\mathcal{C}}$. He then broadcasts the local blockchain to the network. Please refer to Figure 15 for more details of our blockchain protocol. In the verification phase,

[3] We note that, to achieve adaptive security, this regular signature scheme will be replaced by a forward-secure digital signature scheme [6]. We further note that, our constructions in previous sections are adaptively secure, and they do not require forward-secure digital signature scheme.

FUNCTIONALITY $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$

The functionality performs same functions with $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$ when receiving commands for **Stake Resource Registration**, **Stake Election**, and **Block Verification**.

In addition:

Signature Generation:

Upon receiving $(\text{BLOCK-SIGN}, P, msg)$ from a party P,
compute $\tilde{\sigma} \leftarrow \mathsf{Sign}(msg)$, verify that no entry $(P, msg, \tilde{\sigma}, 0)$ is recorded. If it is, then output an error message (ERROR) to P and halt. Else, output $(\text{BLOCK-SIGNED}, P, msg, \tilde{\sigma})$ to P and the adversary, and record the entry $(P, msg, \tilde{\sigma}, 1)$.

Block Verification:
Upon receiving $(\text{BLOCK-VERIFY}, P, msg, \tilde{\sigma})$ from a party $P' \in \mathcal{P}$,

1. If there exists a record of the form $(P, \cdot)$, (the party P is elected)
   - If $(P, msg, \tilde{\sigma}, 1)$ is recorded, then set $f := 1$.
   - Else, if P is not corrupted, and no entry $(P, msg, \tilde{\sigma}', 1)$ for any $\tilde{\sigma}'$ is recorded, then set $f := 0$ and record the entry $(P, msg, \tilde{\sigma}, f)$.
   - Else, if there is an entry $(P, msg, \tilde{\sigma}, f')$, then set $f := f'$.

   Output $(\text{BLOCK-VERIFIED}, P, msg, f)$ to the party P'.

2. Otherwise, if there is no record of the form $(P, \cdot)$, (the party P is not elected)
   set $f := 0$ and output $(\text{BLOCK-VERIFIED}, P, msg, f)$ to the party P'.

Figure 14: Augmented resource certification functionality $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$.

PROTOCOL Π^{main}

Π^{main} performs same as $\Pi^{\text{core}\bullet}$ by calling subroutine BestMain and $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ instead of $(\mathtt{D}, \eta)$-BestCore$^{\bullet}$ and $\mathcal{F}^{\bullet}_{\mathsf{rCERT}}$.
In addition, in *Stake election* :
If the player is elected to generate a block after generating a core-block he will:

- *Attempt to extend PoS blockchain:*
  - *Generate a new block:*
    If the new block-core B is obtained, then compute $\tilde{h}_\ell := \mathsf{hash}(\tilde{\mathcal{C}}[\ell])$, send $(\text{SIGN}, P, \langle \tilde{h}_\ell, B, \tilde{X} \rangle)$ to $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$, and receive $(\text{SIGNED}, P, \langle \tilde{h}_\ell, B, \tilde{X} \rangle, \tilde{\sigma})$ from $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$.
    set the new block $\tilde{B} := \langle \langle \tilde{h}_\ell, B, \tilde{X} \rangle, P, \tilde{\sigma} \rangle$, and set $\tilde{\mathcal{C}} := \tilde{\mathcal{C}} \| \tilde{B}$, and $state := state \cup \{\tilde{\mathcal{C}}\}$, and then send $(\text{BROADCAST}, \tilde{\mathcal{C}})$ to $\mathcal{F}_{\mathsf{NET}}$.

Figure 15: Our proof-of-stake blockchain protocol Π^{main} in the $\{\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid model. (See Figure 16 for the subroutine BestMain.)

the functionality $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$ will verify if the main-chain is valid after the core-chain is verified to ensure the payload is correct.

In Section 5, our proof-of-stake core-chain protocol $\Pi^{\text{core}\bullet}$ uses the subroutine $(\mathtt{D}, \eta)$-BestCore$^{\bullet}$ to single out the best valid core-chain from a set of core-chains. Here we describe a similar strategy, subroutine BestMain, to single out the best blockchain from a set of blockchains. The subroutine BestMain here is a slightly augmented version of the subroutine $(\mathtt{D}, \eta)$-BestCore$^{\bullet}$ in our core-chain protocol. Intuitively, a blockchain is the best one if it is the *current longest valid* blockchain. The BestMain subroutine takes as input, a blockchain set $\tilde{\mathbb{C}}'$ and the current time information $\mathtt{round}'$. Intuitively, the subroutine validates all $\tilde{\mathcal{C}} \in \tilde{\mathbb{C}}'$, then finds the valid longest blockchain.

In more detail, BestMain proceeds as follows. On input the current set of blockchains $\tilde{\mathbb{C}}'$ and the current

SUBROUTINE BestMain

The subroutine BestMain has access to $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$, and with input $(\tilde{\mathbb{C}}', \texttt{round})$.
For every chain $\tilde{\mathcal{C}} \in \tilde{\mathbb{C}}'$, and proceed as follows.

1. Set $\ell := \mathsf{len}(\tilde{\mathcal{C}})$. Derive $\mathcal{C}$ from $\tilde{\mathcal{C}}$.
2. Perform same as in $(\mathtt{D}, \eta)$-BestCore$^{\bullet}$.
   In addition verify each block on $\tilde{\mathcal{C}}$ by query $(\text{BLOCK-VERIFY}, P_i, \langle \tilde{h}_{i-1}, B_i, \tilde{X}_i \rangle, \tilde{\sigma}_i)$ to $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$.

Perform same best chain strategy with $\tilde{\mathcal{C}}$ as in $(\mathtt{D}, \eta)$-BestCore$^{\bullet}$.

Figure 16: The chain set validation subroutine BestMain.

time information round, and for each blockchain $\tilde{\mathcal{C}}$, the subroutine first unfolds the blockchain $\tilde{\mathcal{C}}$ into the corresponding core-chain $\mathcal{C}$; the subroutine then evaluates every block-core of the core-chain $\mathcal{C}$, and then every block of the blockchain $\tilde{\mathcal{C}}$, sequentially. Let ℓ be the length of $\tilde{\mathcal{C}}$. (ℓ is also the length of the corresponding core-chain $\mathcal{C}$.) Starting from the head of $\mathcal{C}$, for every block-core $\mathcal{C}[i]$, for all $i \in [\ell]$, in the core-chain $\mathcal{C}$, the BestMain subroutine (1) ensures that $\mathcal{C}[i]$ is linked to the previous block-core $\mathcal{C}[i-1]$ correctly, and (2) tests if the signature generated by that PoS-player can be verified (by interacting with $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$). Then for every block-core $\tilde{\mathcal{C}}[i]$, for all $i \in [\ell]$, in the blockchain $\tilde{\mathcal{C}}$, the BestMain subroutine (1) ensures that $\tilde{\mathcal{C}}[i]$ is linked to the previous block $\tilde{\mathcal{C}}[i-1]$ correctly, and (2) tests if the signature generated by that PoS-player can be verified (by interacting with $\tilde{\mathcal{F}}^{\bullet}_{\mathsf{rCERT}}$). After the validation, the best valid blockchain set is selected. Please refer to Figure 16 for more details. The security analysis can be found in Section 6.3.

6.3 Analysis of blockchain protocol

As mentioned before, our blockchain protocol Π^{main} can be viewed as an augmented version of the core-chain protocol $\Pi^{\text{core}\bullet}$ in Section 5; each security property of our blockchain protocol can be reduced to the corresponding property of the core-chain protocol. We note that, as in the core-chain protocol $\Pi^{\text{core}\bullet}$, the security properties hold under the assumption of honest majority of effective stakes based on α° and β°.

Theorem 6.1. *Consider blockchain protocol Π^{main} where honest players follow the 2-distance-greedy strategy while adversarial players follow the fully-greedy strategy. If $\alpha^{\circ} = \lambda\beta^{\circ}$, $\lambda > 1$, then the protocol Π^{main} can achieve chain growth, chain quality, common prefix and chain soundness properties.*

Chain growth. Chain growth property comes from that of the core-chain protocol. If a PoS-player is chosen to generate a block-core in the core-chain, the PoS-player is also be chosen to generate the corresponding block in the blockchain. That means, when the core-chain is extended with a new block-core, the corresponding blockchain is also extended with a new block. More formally, we have the following statement.

Corollary 6.2 (Chain growth)**.** *Consider the blockchain protocol Π^{main}. Consider $\alpha^{\circ} = \lambda\beta^{\circ}$, $\lambda > 1$, and $\delta > 0$. Consider an honest PoS-player with the best PoS blockchain $\tilde{\mathcal{C}}$ in round r, and local PoS blockchain $\tilde{\mathcal{C}}'$ in round r', where $r' > r$. Then we have $\Pr\left[\mathsf{len}(\tilde{\mathcal{C}}') - \mathsf{len}(\tilde{\mathcal{C}}) \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$, where $t = r' - r$, $g = (1-\delta)\alpha^{\circ}$.*

Proof. From the protocol, we know that every PoS blockchain $\tilde{\mathcal{C}}$ is associated with a PoS core-chain $\mathcal{C}$. Each valid block-core B has a corresponding block $\tilde{B}$. We have, $\mathsf{len}(\tilde{\mathcal{C}}') = \mathsf{len}(\mathcal{C}')$ and $\mathsf{len}(\tilde{\mathcal{C}}) = \mathsf{len}(\mathcal{C})$. That means, $\mathsf{len}(\tilde{\mathcal{C}}') - \mathsf{len}(\tilde{\mathcal{C}}) = \mathsf{len}(\mathcal{C}') - \mathsf{len}(\mathcal{C})$. From Corollary 5.4, we have $\Pr\left[\mathsf{len}(\tilde{\mathcal{C}}') - \mathsf{len}(\tilde{\mathcal{C}}) \geq g \cdot t\right] = \Pr\left[\mathsf{len}(\mathcal{C}') - \mathsf{len}(\mathcal{C}) \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$. This completes the proof. □

Chain quality. Similarly, if an honest player contributes a block-core to the core-chain, he also contributes a block to the blockchain. More formally, we have the following statement.

Corollary 6.3 (Chain quality)**.** *Consider the blockchain protocol* Π^{main}*. Consider* $\alpha^\circ = \lambda\beta^\circ$*,* $\lambda > 1$*, and* $\delta > 0$*. Consider an honest PoS-player with the best PoS blockchain* $\tilde{\mathcal{C}}$*. Consider any* ℓ *consecutive blocks on* $\tilde{\mathcal{C}}$*, including* ℓ_{good} *blocks are generated by honest PoS-players. Then we have* $\Pr[\frac{\ell_{good}}{\ell} \geq \mu] \geq 1 - e^{-\Omega(\ell)}$ *where* $\mu = 1 - (1+\delta)\frac{1}{\lambda}$*.*

Proof. From the algorithms, we know that every PoS blockchain $\tilde{\mathcal{C}}$ is associated with a PoS core-chain $\mathcal{C}$. Let $\ell_{good}^{\text{core}}$ be the number of block-cores from honest stakeholders on core-chain $\mathcal{C}$. Let $\ell_{good}^{\text{main}}$ be the number of blocks from honest stakeholders on blockchain $\tilde{\mathcal{C}}$. Recall that both block-core $\mathcal{C}[i]$ and the corresponding block $\tilde{\mathcal{C}}[i]$ are signed by the same stakeholder. We have $\ell_{good}^{\text{core}} = \ell_{good}^{\text{main}} = \ell_{good}$. We also have that $\mathsf{len}(\mathcal{C}) = \mathsf{len}(\tilde{\mathcal{C}}) = \ell$. From Corollary 5.5 we have $\Pr[\frac{\ell_{good}}{\ell} \geq \mu] \geq 1 - e^{-\Omega(\ell)}$, where $\mu = 1 - (1+\delta)\frac{1}{\lambda}$. □

Common prefix. Our analysis is based on the common prefix analysis of core-chain. The core-chain can achieve common prefix as we discussed. The opportunity for malicious players to destroy common prefix probability is to generate different blockchain for the same core-chain. For the malicious players can sign different blocks for one block-core, this will allow him to fork the blockchain. So the malicious players can fork the blockchain when they are chosen to generate block. However, with the property of hash function, the malicious players can not generate two blocks with same hash value. When an honest player is chosen to extend a block, he will only support one blockchain. Then all of the honest players will converge on one blockchain.

Corollary 6.4 (Common prefix)**.** *Consider the blockchain protocol* Π^{main}*. Consider* $\alpha^\circ = \lambda\beta^\circ$*,* $\lambda > 1$*, and* $\delta > 0$*. Consider two honest PoS-players,* P *in round* r *and* P' *in round* r'*, with the local best PoS blockchains* $\tilde{\mathcal{C}}$*,* $\tilde{\mathcal{C}}'$*, respectively, where* $r' \geq r$*. Then we have* $\Pr\left[\tilde{\mathcal{C}}[\neg\kappa] \preceq \tilde{\mathcal{C}}'\right] \geq 1 - e^{-\Omega(\kappa)}$*.*

Proof. As we discussed, $\tilde{\mathcal{C}}$ and $\tilde{\mathcal{C}}'$ are associated with core-chains $\mathcal{C}$ and $\mathcal{C}'$ respectively. From Corollary 5.6 we know that $\Pr\left[\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'\right] \geq 1 - e^{-\Omega(\kappa)}$.

Based on the assumption that $\alpha^\circ = \lambda\beta^\circ$ and $\lambda > 1$, we can have that the malicious players are not able to generate more than $\Theta(\kappa)$ blocks before an honest player is chosen to generate block with high probability. All of the honest players will converge on the same chain. Put them together, we have $\Pr\left[\tilde{\mathcal{C}}[\neg\kappa] \preceq \tilde{\mathcal{C}}'\right] \geq 1 - e^{-\Omega(\kappa)}$. □

Chain soundness. A new player will accept a blockchain (in which the corresponding core-chain is included). The proof idea for achieving chain soundness property of our blockchain protocol directly follows that for the core-chain protocol. We have the following statement.

Corollary 6.5 (Chain soundness)**.** *Consider the blockchain protocol* Π^{main}*. Consider* $\alpha^\circ = \lambda\beta^\circ$*,* $\lambda > 1$*, and* $\delta > 0$*. There are two honest PoS-players,* P' *and* P'' *in round* r*, with the local best PoS blockchains* $\tilde{\mathcal{C}}'$ *and* $\tilde{\mathcal{C}}''$*, respectively. Let* P' *be a new player and* P'' *be an existing player in round* r*. Then we have* $\tilde{\mathcal{C}}'[\neg\kappa] \preceq \tilde{\mathcal{C}}''$ *and* $\tilde{\mathcal{C}}''[\neg\kappa] \preceq \tilde{\mathcal{C}}'$*.*

Proof. Blockchains $\tilde{\mathcal{C}}'$ and $\tilde{\mathcal{C}}''$ are associated with core-chains $\mathcal{C}'$ and $\mathcal{C}''$ respectively. From Lemma 3.5 we know that $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ and $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$. We immediately have $\tilde{\mathcal{C}}'[\neg\kappa] \preceq \tilde{\mathcal{C}}''$ and $\tilde{\mathcal{C}}''[\neg\kappa] \preceq \tilde{\mathcal{C}}'$. □

7 Discussions of rational attacks

Protocols in previous sections have been rigorously analyzed in cryptographic setting. In this section, we briefly discuss how our proof-of-stake protocols defend against rational attacks.

Nothing at stake attack. In a "nothing at stake" attack, rational players/attackers extend all potential best blockchains to increase the probability to be rewarded. To be noticed, a rational player will not try to extend blockchain from a block which will be the longest chain with very low probability even a "PoS" mining operation is cheap. This is because that store and update the whole chain is heavy. As discussed in the Introduction, in our design honest players will also extend multiple blockchains, which effectively defeat nothing at stake attack. More concretely, in the modified core-chain protocol $\Pi^{\text{core}\circ}$ in Section 4, honest players follow the D-distance-greedy strategy.This means potential longest chains are already supposed to be extended in our protocol.

The players who will try to follow the full-greedy strategy are malicious (not rational) nothing at stake attackers. We have proved (in Lemma 4.7) that the amplification ratio for nothing at stake attackers (following the full-greedy strategy) is bounded. That means, even in the presence of a full-greedy adversary (i.e., nothing at stake attacker), our protocol can still achieve desired security properties.

Selfish mining. In "selfish mining", a rational attacker may not publish his valid puzzle solution immediately to the rest of the network. This greatly endangers the fairness of blockchain. In general, PoS protocols are more vulnerable to selfish mining attacks: selfish miners in PoS can predict when they should generate two or more blocks on a private hidden chain easily.

Our design can effectively weaken the affect of selfish mining. As shown in Section 4, in the modified core-chain protocol where players follow the D-distance-greedy strategy (meaning players will try to extend a set of blockchains instead of the longest one), a block generated by an honest player will still be extended even there is a longer chain with one more block from selfish miners. The greater D is, the lower the probability that selfish miners succeed.

Stake grinding. In "stake grinding", an attacker/player may attempt to bias the randomness in his own favor with the goal of increasing the probability that he can be selected to generate next block. In our construction, there is no randomness as input to generate a block. We use unique signature scheme to sign a block and the transaction data and other randomness are removed from core blockchain. Our scheme is immune to stake grinding attack.

Long-range attack. In "long-range attack", an adversary may generate a private blockchain, starting from a very "old" block or even from the genesis block. In general PoS blockchain, no physical resources will be burnt, the adversary may generate his own chain by trying much more times than he is supposed to do in the honest public blockchain. As a result, the adversary may be able to easily generate more blocks to fork a chain which is longer than the public chain. In our construction, there is no randomness can be tried to generate a block except the round number. The blocks that adversary can generate in a time period is proportional to the stakes he controls. This design is immune to long-range attack.

8 Extensions

Our design is a natural mimic of Nakamoto's but via proof-of-stake. We can easily "borrow" many ideas in Nakamoto's white paper (and in follow-up papers) to our design. In this section, we discuss a few of them.

Blockchain with adaptive difficulty adjustment. In Bitcoin, in order to maintain a steady chain growth rate, the system adjusts the PoW hash target difficulty adaptively. The smaller the target, the lower the probability to get a valid PoW block by a hash function query, and vice versa. Our scheme can be extended to support adaptive difficulty easily. As in Nakamoto's system, the target difficulty is adjusted every $\mathtt{m}$ blocks for some integer $\mathtt{m}$. The time span of difficulty adjustment is called an *epoch*; and let $\mathtt{t}$ be the expected time of an epoch. Let t_i be the actual time span of the i-th epoch, and $\mathtt{T}_i$ be the target difficulty in the i-th epoch. We have the target difficulty in the $(i+1)$-th epoch as follows:

$$\mathtt{T}_{i+1} = \frac{t_i}{\mathtt{t}}\mathtt{T}_i$$

From the equation above we can observe that, if $t_i > \mathtt{t}$ then $\mathtt{T}_{i+1} > \mathtt{T}_i$ and vice-versa. In the case that $t_i > \mathtt{t}$, the stakeholders spend longer time to obtain m blocks; it means the system requires more time than expected for the i-th epoch; thus, the target difficulty should be increased so that the stakeholders can find new blocks faster in the next epoch. This *negative feedback* mechanism makes the system stable. To extend a PoS blockchain, we modify the hash inequality as $\mathsf{H}(\mathsf{hash}(B_i), \mathtt{round}, \mathrm{PK}, \sigma) < \mathtt{T}_i$. A player will test if he is qualified to sign a PoS-block based on the current target difficulty $\mathtt{T}_i$.

Blockchain in the non-flat model. Our ideas in previous sections are described in the "flat" model, where all PoS-players are assumed to hold the same number of stakes (and they are selected as the winning player with the same probability in each round). In reality, PoS-players have different amounts of stake. We next discuss how to extend our design ideas properly into this more realistic "non-flat" model. Consider a PoS-player, with verification-signing key pair $(\mathrm{PK}, \mathrm{SK})$, holding v number of stakes. Let $\mathtt{T}_j$ denote the target difficulty in the current epoch, i.e., the j-th epoch. We change the hash inequality as follows:

$$\mathsf{H}(\mathsf{hash}(B_i), \mathtt{round}, \mathrm{PK}, \sigma) < v\mathtt{T}_i$$

Now we argue that the winning probability of a PoS-player for generating a new block-core is proportional to the amount of stake he controls. We assume the total amount of stakes in the whole system is n; consider hash function $\mathsf{H} : \{0,1\}^* \mapsto \{0,1\}^\kappa$. We assume $np \ll 1$, where $p = \frac{\mathtt{T}_i}{2^\kappa}$. Now the PoS-player can play different strategies. If the PoS-player puts his v coins in one account, the probability that he is selected to sign a PoS block is vp. If the PoS-player puts his v coins in v accounts and every account has one stake, the probability that an account is selected to sign a PoS block is p. The outputs of hash function are independent for different verification keys. The total probability that the PoS-player is selected is $1-(1-p)^v \approx vp$. That is, the probability a stakeholder is selected in the non-flat model is (approximately) equal to the accumulated probability that he distributes the stakes to different accounts as in the flat-model. For a PoS-player, the probability that he is selected only depends on the total amount of stakes he controls.

Synchronization. In our construction, in the hash inequality $\mathsf{H}(\mathsf{hash}(B_i), \mathtt{round}, \mathrm{PK}, \sigma) < \mathtt{T}_i$, each player relies on his local clock to set the value $\mathtt{round}$. In our security analysis in previous sections, for simplicity, we assume these local clocks can be perfectly synchronized via a global clock. In our future work, we will improve our security analysis so that our protocol can be based on a "relaxed" global clock (in the sense that, players' local clocks may deviate from the "idealized" clock slightly). We note that, this "relaxed" global clock can be instantiated via the Network Time Protocol (NTP). Typically, NTP can synchronize players within tens of milliseconds over the public Internet.

Other considerations. We can also mimic Nakamoto's design and incentivize the players to participate in the protocol by collecting the "rewards". We note that new ideas (e.g., [49]) can be adopted. To extend our design idea to a full-fledged blockchain protocol, we also need to use authenticated data structure to more effectively manage the transactions. Instead of straightforwardly including the entire "payload" $\tilde{X}_i$ in the block $\tilde{B}_i$ (as in Section 6, and in [29, 48]), we can store a Merkle root in $\tilde{B}_i$. New ideas (e.g., [51]) can also be used in our design.

References

[1] Litecoin. 2011. https://litecoin.org.

[2] NXT whitepaper. 2014. https://www.dropbox.com/s/cbuwrorf672c0yy/NxtWhitepaper_v122_rev4.pdf.

[3] A. Back. Hashcash — A denial of service counter-measure. 2002. http://hashcash.org/papers/hashcash.pdf.

[4] C. Badertscher, P. Gazi, A. Kiayias, A. Russell, and V. Zikas. Ouroboros Genesis: Composable Proof-of-Stake Blockchains with Dynamic Availability. In *CCS*, 2018. https://eprint.iacr.org/2018/378.

[5] C. Badertscher, U. Maurer, D. Tschudi, and V. Zikas. Bitcoin as a transaction ledger: A composable treatment. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 324–356. Springer, Heidelberg, Aug. 2017.

[6] M. Bellare and S. K. Miner. A forward-secure digital signature scheme. In M. J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 431–448. Springer, Heidelberg, Aug. 1999.

[7] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73. ACM, 1993.

[8] I. Bentov, A. Gabizon, and A. Mizrahi. Currencies without proof of work. In *Bitcoin Workshop*, 2016.

[9] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld. Proof of activity: Extending bitcoin's proof of work via proof of stake. In *SIGMETRICS Perform. Eval. Rev.*, pages 34–37. ACM, 2014.

[10] Bitcointalk. Proof of stake instead of proof of work. July 2011. Online post by QuantumMechanic, available at https://bitcointalk.org/index.php?topic=27787.0.

[11] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In C. Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, Dec. 2001.

[12] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. SoK: Research perspectives and challenges for bitcoin and cryptocurrencies. In *2015 IEEE Symposium on Security and Privacy*, pages 104–121. IEEE Computer Society Press, May 2015.

[13] V. Buterin. A Next-Generation Smart Contract and Decentralized Application Platform. 2014. https://github.com/ethereum/wiki/wiki/White-Paper.

[14] V. Buterin. Understanding serenity, part 2: Casper. 2015. https://blog.ethereum.org/2015/12/28/understanding-serenity-part-2-casper/.

[15] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, Jan. 2000.

[16] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, 2000. http://eprint.iacr.org/2000/067.

[17] R. Canetti. Universally composable signatures, certification and authentication. Cryptology ePrint Archive, Report 2003/239, 2003. http://eprint.iacr.org/2003/239.

[18] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982.

[19] J. Chen, S. Gorbunov, S. Micali, and G. Vlachos. Algorand agreement: Super fast and partition resilient byzantine agreement. 2018. https://eprint.iacr.org/2018/377.

[20] J. Chen and S. Micali. Algorand. In *arXiv:1607.01341*, May 2017. http://arxiv.org/abs/1607.01341.

[21] A. Chepurnoy, T. Duong, L. Fan, and H.-S. Zhou. Twinscoin: A cryptocurrency via proof-of-work and proof-of-stake. In *Cryptology ePrint Archive, Report 2017/232*, 2017. https://eprint.iacr.org/2017/232.

[22] CryptoManiac. Proof of stake. novacoin wiki. 2014. https://github.com/novacoin-project/novacoin/wiki/Proof-of-stake/.

[23] P. Daian, R. Pass, and E. Shi. Snow white: Robustly reconfigurable consensus and applications to provably secure proofs of stake. In *FC*, 2019. http://eprint.iacr.org/2016/919.

[24] B. David, P. Gazi, A. Kiayias, and A. Russell. Ouroboros Praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In *EUROCRYPT*, 2018. http://eprint.iacr.org/2017/573.

[25] T. Duong, L. Fan, and H.-S. Zhou. 2-hop blockchain: Combining proof-of-work and proof-of-stake securely. In *Cryptology ePrint Archive, Report 2016/716*, 2016. https://eprint.iacr.org/2016/716.

[26] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In E. F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 139–147. Springer, Heidelberg, Aug. 1993.

[27] I. Eyal. The miner's dilemma. In *2015 IEEE Symposium on Security and Privacy*, pages 89–103. IEEE Computer Society Press, May 2015.

[28] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In N. Christin and R. Safavi-Naini, editors, *FC 2014*, volume 8437 of *LNCS*, pages 436–454. Springer, Heidelberg, Mar. 2014.

[29] J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 281–310. Springer, Heidelberg, Apr. 2015.

[30] J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol with chains of variable difficulty. In *CRYPTO*, 2017. https://eprint.iacr.org/2016/1048.

[31] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *SOSP*, 2017. https://eprint.iacr.org/2017/454.

[32] D. Hofheinz and J. Müller-Quade. Universally composable commitments using random oracles. In M. Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 58–76. Springer, Heidelberg, Feb. 2004.

[33] Intel. Proof of Elapsed Time (PoET). 2016. https://intelledger.github.io/introduction.html.

[34] A. Kiayias, E. Koutsoupias, M. Kyropoulou, and Y. Tselekounis. Blockchain mining games. In *Proceedings of the 2016 ACM Conference on Economics and Computation (EC)*, pages 365–382, 2016.

[35] A. Kiayias and G. Panagiotakos. Speed-security tradeoffs in blockchain protocols. Cryptology ePrint Archive, Report 2015/1019, 2015. http://eprint.iacr.org/2015/1019.

[36] A. Kiayias and G. Panagiotakos. On trees, chains and fast transactions in the blockchain. Cryptology ePrint Archive, Report 2016/545, 2016. http://eprint.iacr.org/2016/545.

[37] A. Kiayias, A. Russell, B. David, and R. Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *CRYPTO*, 2017. http://eprint.iacr.org/2016/889.

[38] S. King and S. Nadal. PPCoin: Peer-to-peer crypto-currency with proof-of-stake. 2012. https://peercoin.net/assets/paper/peercoin-paper.pdf.

[39] J. Kwon. Tendermint: Consensus without mining. 2014. https://tendermint.com/static/docs/tendermint.pdf.

[40] A. Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 597–612. Springer, Heidelberg, Aug. 2002.

[41] A. Miller, A. Juels, E. Shi, B. Parno, and J. Katz. Permacoin: Repurposing bitcoin work for data preservation. In *2014 IEEE Symposium on Security and Privacy*, pages 475–490. IEEE Computer Society Press, May 2014.

[42] T. Moran and I. Orlov. Proofs of space-time and rational proofs of storage. Cryptology ePrint Archive, Report 2016/035, 2016. http://eprint.iacr.org/2016/035.

[43] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoin.org/bitcoin.pdf.

[44] A. Narayanan, J. Bonneau, , E. W. Felten, A. Miller, and S. Goldfeder. Bitcoin and cryptocurrency technology. 2015. https://www.coursera.org/learn/cryptocurrency.

[45] K. Nayak, S. Kumar, A. Miller, and E. Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. Cryptology ePrint Archive, Report 2015/796, 2015. http://eprint.iacr.org/2015/796.

[46] S. Park, K. Pietrzak, A. Kwon, J. Alwen, G. Fuchsbauer, and P. Gaži. Spacemint: A cryptocurrency based on proofs of space. Cryptology ePrint Archive, Report 2015/528, 2015. http://eprint.iacr.org/2015/528.

[47] R. Pass, L. Seeman, and a. shelat. Analysis of the blockchain protocol in asynchronous networks. In J. Coron and J. B. Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 643–673. Springer, Heidelberg, Apr. / May 2017.

[48] R. Pass, L. Seeman, and A. Shelat. Analysis of the blockchain protocol in asynchronous networks. In *EUROCRYPT*, 2017. https://eprint.iacr.org/2016/454.

[49] R. Pass and E. Shi. FruitChains: A fair blockchain. In E. M. Schiller and A. A. Schwarzmann, editors, *36th ACM PODC*, pages 315–324. ACM, July 2017.

[50] R. Pass and E. Shi. The sleepy model of consensus. In *ASIACRYPT*, 2017. http://eprint.iacr.org/2016/918.

[51] L. Reyzin, D. Meshkov, A. Chepurnoy, and S. Ivanov. Improving authenticated dynamic dictionaries, with applications to cryptocurrencies. In A. Kiayias, editor, *FC 2017*, volume 10322 of *LNCS*, pages 376–392. Springer, Heidelberg, Apr. 2017.

[52] A. Sapirshtein, Y. Sompolinsky, and A. Zohar. Optimal selfish mining strategies in bitcoin. In J. Grossklags and B. Preneel, editors, *FC 2016*, volume 9603 of *LNCS*, pages 515–532. Springer, Heidelberg, Feb. 2016.

[53] O. Schrijvers, J. Bonneau, D. Boneh, and T. Roughgarden. Incentive compatibility of bitcoin mining pool reward functions. In J. Grossklags and B. Preneel, editors, *FC 2016*, volume 9603 of *LNCS*, pages 477–498. Springer, Heidelberg, Feb. 2016.

[54] Y. Sompolinsky and A. Zohar. Secure high-rate transaction processing in bitcoin. In R. Böhme and T. Okamoto, editors, *FC 2015*, volume 8975 of *LNCS*, pages 507–527. Springer, Heidelberg, Jan. 2015.

[55] P. Vasin. Blackcoin's proof-of-stake protocol v2. 2014. http://blackcoin.co/blackcoin-pos-protocol-v2-whitepaper.pdf.

[56] G. Wood. Ethereum: A secure decentralized transaction ledger. 2014. http://gavwood.com/paper.pdf.

[57] F. Zhang, I. Eyal, R. Escriva, A. Juels, and R. van Renesse. REM: Resource-Efficient Mining for Blockchains. In *USENIX Security*, 2017. https://eprint.iacr.org/2017/179.

A Security analysis for the basic version of core-chain protocol

Our core-chain protocol Π^{core} is in the "flat, static difficulty" model in which each PoS-player holds a unit of stake and the total number of stakeholders is fixed. Let n be the total number of stakeholders in the protocol. Let p denote the probability that a stakeholder is qualified to extend the core-chain in a round. Let ρ denote the ratio of malicious stake. Let $\alpha_0 = (1-\rho)np$ be the expected number of honest stakeholders that are qualified in a round to extend the longest core-chain. Let $\beta_0 = \rho np$ be the expected number of malicious stakeholders that are qualified in a round to extend any chosen core-chain. Let α and β be the effective counterparts, respectively in the network delay setting. Here we assume $np \ll 1$. This means the expected number of stakeholders that are qualified to extend a core-chain in a round is much less than 1. Additionally, we assume that $\alpha_0 = \lambda\beta_0$ where $\lambda \in (1, \infty)$.

We are now ready to state our theorem for our core-chain protocol Π^{core} in the presence of an adversary who extends blockchain via the basic strategy (i.e., extending a single chain).

Theorem A.1 (Theorem 1.1, restated)**.** *Consider core-chain protocol Π^{core} where all players follow the simple strategy of extending the longest chain; in addition, all players have their stake registered independent of the state in the protocol execution. Let α and β be the effective expected number of blocks generated by honest and malicious players in a round respectively. If $\alpha = \lambda\beta$, $\lambda > 1$, then the protocol Π^{core} can achieve chain growth, chain quality, common prefix and chain soundness properties.*

Basic terms Before giving the details of the security analysis, we define two terms, *public chain* and *honest successful round*, as follows.

Definition A.2 (Public chain)**.** *Consider a round r. We say a chain $\mathcal{C}$ is a* public chain in round r *if such chain $\mathcal{C}$ is known by all honest players in round r.*

Definition A.3 (Honest successful round)**.** *We say a round r is an* honest successful *round, if in the round r, at least one honest PoS-player is selected to extend the core-chain.*

Let p_{good} be the probability that a round can be an honest successful round. We have $p_{good} = 1 - (1 - p)^{(1-\rho)n}$. In the case that $np \ll 1$, we have $p_{good} \approx p(1-\rho)n$. That is $p_{good} \approx \alpha_0$. In the following sections, we assume the probability that a round is honest successful round is α_0 directly.

A.1 Analysis with bounded delay

We assume that the malicious parties can delay messages up to Δ number of rounds. (This is guaranteed by $\mathcal{F}_{\mathsf{NET}}$.) When an honest PoS-player is qualified to generate a new PoS block-core, he will broadcast it to the network and expect all parties to receive it. The honest player may not obtain the best PoS core-chain and thus work on a different PoS core-chain. If an honest player generates a new PoS block-core during the delay time and later receives a better PoS block-core from the network, then this generated PoS block-core will become useless and thus this honest player's effort during the time window is wasted. In this subsection, we provide a formal analysis for our core-chain protocol in the presence of the network delay.

Hybrid expriment To analyze the best strategy of the adversary, and the worst scenario that may happen to the honest players, we consider the following notations.

Let $\mathsf{REAL}(\omega) = \mathsf{EXEC}_{\Pi^{\mathrm{core}},\mathcal{A},\mathcal{Z}}(\omega)$ denote the typical execution of Π^{core} where

- ω is the randomness in the execution,
- Messages of honest players may be delayed by $\mathcal{F}_{\mathsf{NET}}$ in at most Δ rounds.

Let $\mathsf{HYB}^r(\omega) = \mathsf{EXEC}^r_{\Pi^{\mathrm{core}},\mathcal{A},\mathcal{Z}}(\omega)$ denote the hybrid execution as in real execution except that after round r, $\mathsf{HYB}^r(\omega)$ has the following modifications from $\mathsf{REAL}(\omega)$:

- The randomness is fixed to ω as in $\mathsf{HYB}^r(\omega)$,
- $\mathcal{F}_{\mathsf{NET}}$ delays all messages generated by *honest* PoS-players to exact Δ rounds,
- Remove all new messages sent by the adversary to honest players, and delay currently undelivered messages from corrupted parties to the maximum of Δ rounds,
- Whenever some message is being delayed, no *honest* PoS-players query the functionality $\mathcal{F}_{\mathsf{rCERT}}$ until the message is delivered.

In $\mathsf{REAL}(\omega)$, the number of honest successful rounds is not less than that in the $\mathsf{HYB}^r(\omega)$. The following lemma shows that the chain growth rate in the real execution is not lower than that in the hybrid execution.

In order to distinguish core-chain in $\mathsf{HYB}^r(\omega)$ with in $\mathsf{REAL}(\omega)$ executions, we use $\mathcal{C}_{\mathrm{hybrid}}$ to denote it.

Claim A.4. *Consider two executions $\mathsf{REAL}(\omega)$ and $\mathsf{HYB}^r(\omega)$ for all ω, r. For any honest PoS-player P at round r', where $r' > r$, let $\mathcal{C}'$ denote the PoS core-chain of P at round r' in the execution $\mathsf{REAL}(\omega)$ and $\mathcal{C}'_{\mathrm{hybrid}}$ denote the PoS core-chain of P at round r' in the execution $\mathsf{HYB}^r(\omega)$. Then we have $\mathsf{len}(\mathcal{C}') \geq \mathsf{len}(\mathcal{C}'_{\mathrm{hybrid}})$.*

Proof. We prove this lemma by induction. We consider the initial state before round r. From the definition of hybrid experiment, all players have same $\mathtt{VIEW}$ at round r. We have $\mathsf{len}(\mathcal{C}) \geq \mathsf{len}(\mathcal{C}_{\mathrm{hybrid}})$. We suppose it holds for all players before round $s-1$. The only case that $\mathsf{len}(\mathcal{C}^s) < \mathsf{len}(\mathcal{C}^s_{\mathrm{hybrid}})$ is the player P received a new core-chain to extend $\mathcal{C}^s_{\mathrm{hybrid}}$ at round s in $\mathsf{HYB}^r(\omega)$. According to the definition of hybrid experiment, this extended PoS block-core must be generated at round $s-\Delta$ by an honest player P_*, that makes $\mathsf{len}(\mathcal{C}^s_{\mathrm{hybrid}}) = \mathsf{len}(\mathcal{C}^{s-\Delta}_{\mathrm{hybrid}}) + 1$. At the same time, the player P_* must succeed to extend PoS block-core at round $s-\Delta$ in $\mathsf{REAL}(\omega)$. This extension will make $\mathcal{C}^{s-\Delta}_*$ increase by one block. For player P_* is honest, P must have received the extension at (or before) round r'. Putting them together, we have $\mathsf{len}(\mathcal{C}') \geq \mathsf{len}(\mathcal{C}'^{\Delta})$. □

Analysis in the worst delay setting Note that, the malicious players can delay the messages for at most Δ rounds. As a consequence, some efforts from honest players may be wasted. Below we develop a lemma for the "discount" version of honest players' efforts in the execution of $\mathsf{HYB}^r(\omega)$.

Claim A.5. *Consider $\mathsf{HYB}^r(\omega)$ where the adversary is allowed to delay messages for at most Δ rounds. Let $\alpha_0 > 0$ be the expected number of honest stakeholders that are chosen in a round. Let α be the actual probability that a round $s > r$ is an honest successful round. Then we have that $\alpha = \frac{\alpha_0}{1+\Delta\alpha_0}$.*

Proof. In $\mathsf{HYB}^r(\omega)$, if round r', where $r' > r$, is an *honest successful round*, then no PoS-players will query functionality $\mathcal{F}_{\mathsf{rCERT}}$ in the next Δ rounds. Now, assume in $\mathsf{HYB}^r(\omega)$, there are c number of honest successful rounds, from round r to round $(r+t)$, where $t > 0$. We then have the number of actual working rounds for honest stakeholders will remain $t - \Delta c$. For each round, the probability that it is an honest successful round is α_0. We have $\alpha_0(t - \Delta c) = c$. This implies that $c = \frac{\alpha_0 t}{1+\Delta\alpha_0}$. We then have $\alpha = \frac{\alpha_0}{1+\Delta\alpha_0}$. □

Let $\mathtt{VIEW}^r$ denote the $\mathtt{VIEW}$ at round r in $\mathsf{REAL}(\omega)$ where $r > 0$. Let $\mathsf{len}(\mathtt{VIEW}^r)$ denote the length of the best public PoS core-chain in $\mathtt{VIEW}^r$. The following lemma demonstrates that each successful round would contribute one PoS block-core to the best public PoS core-chain after Δ rounds in an execution of $\mathsf{HYB}^r(\omega)$.

Claim A.6. *Consider* $\mathsf{HYB}^r(\omega)$. *For any honest successful round* s, *where* $s > r$, *it holds that* $\mathsf{len}(\mathtt{VIEW}^{s+\Delta}) - \mathsf{len}(\mathtt{VIEW}^s) \geq 1$.

Proof. By Definition A.3, there is at least one honest PoS-player producing a PoS block-core at round s. Let $\mathcal{C}^s_{\text{hybrid}}$ be the PoS core-chain that is extended by the PoS-player at round s. We have $\mathsf{len}(\mathcal{C}^s_{\text{hybrid}}) \geq \mathsf{len}(\mathtt{VIEW}^s)$. At the end of round s the honest player will broadcast the extended chain with length $\mathsf{len}(\mathcal{C}^s_{\text{hybrid}})+1$. At the end of round $s+\Delta$, all honest players will receive the extended core-chain, we have $\mathsf{len}(\mathtt{VIEW}^{s+\Delta}) \geq \mathsf{len}(\mathcal{C}^{s+\Delta}_{\text{hybrid}}) = \mathsf{len}(\mathcal{C}^s_{\text{hybrid}}) + 1$. Putting them together, we have $\mathsf{len}(\mathtt{VIEW}^{s+\Delta}) - \mathsf{len}(\mathtt{VIEW}^s) \geq 1$. □

Corollary A.7. *Consider* $\mathsf{HYB}^r(\omega)$. *Assume there are* h *number of honest successful rounds from round* r *to round* $r+t$ *where* $t > 0$. *Then it holds that* $\mathsf{len}(\mathtt{VIEW}^{r+t+\Delta}) - \mathsf{len}(\mathtt{VIEW}^r) \geq h$.

Proof. Let r_k be the kth honest successful round where $r < \mathtt{round}_k < r+t$ and $1 \leq k \leq h$. From Claim A.6, we have $\mathsf{len}(\mathtt{VIEW}^{\mathtt{round}_k+\Delta}) - \mathsf{len}(\mathtt{VIEW}^{\mathtt{round}_k}) \geq 1$. Then we have $\mathsf{len}(\mathtt{VIEW}^{r+t}) - \mathsf{len}(\mathtt{VIEW}^r) \geq \sum_{i=1}^{h}(\mathsf{len}(\mathtt{VIEW}^{\mathtt{round}_k+\Delta}) - \mathsf{len}(\mathtt{VIEW}^{\mathtt{round}_k})) \geq h$. □

A.2 Achieving chain growth property

We here demonstrate that our core-chain protocol satisfies the growth property (Definition 2.3). The concrete statement to be proved can be found in Lemma 3.2.

Claim A.8. *Consider* $\mathsf{HYB}^r(\omega)$, *and* $\delta > 0$. *Let* X *be the number of honest successful rounds from round* r *to round* $r+t$, *where* $t > 0$. *Then we have* $\Pr[X > (1-\delta)\alpha t] > 1 - e^{-\Omega(t)}$.

Proof. Based on Claim A.5, we have that, on average, there are αt number of honest successful rounds in any t consecutive rounds. By Chernoff bound, we have $\Pr[X \leq (1-\delta)\alpha t] \leq e^{-\delta^2\alpha t/2}$. Thus, we have $\Pr[X > (1-\delta)\alpha t] > 1 - e^{-\delta^2\alpha t/2} = 1 - e^{-\Omega(t)}$. □

Claim A.9. *Consider* $\mathsf{HYB}^r(\omega)$ *and* $\delta > 0$. *Consider an honest PoS-player* P *with the best PoS core-chain* $\mathcal{C}_{\text{hybrid}}$ *in round* r, *and an honest PoS-player* P' *with the best PoS core-chain* $\mathcal{C}'_{\text{hybrid}}$ *in round* r', *respectively, where* $r' - r \gg \Delta$. *Then we have*

$$\Pr\left[\mathsf{len}(\mathcal{C}'_{\text{hybrid}}) - \mathsf{len}(\mathcal{C}_{\text{hybrid}}) \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$$

where $t = r' - r$ *and* $g = (1-\delta)\alpha$.

Proof. First, we note that $\mathcal{C}_{\text{hybrid}}$ will be received by all honest players no later than round $r+\Delta$ because player P is honest. We have $\mathsf{len}(\mathcal{C}_{\text{hybrid}}) \leq \mathsf{len}(\mathtt{VIEW}^{r+\Delta})$. Now we consider the chain growth from round $r+\Delta$ to round r'. For $t \gg \Delta$, we have $t \approx t - \Delta$ for simplicity. From Claim A.8, in any t consecutive rounds the number of honest successful round is more than $(1-\delta)\alpha t$ with the probability at least $1 - e^{-\Omega(t)}$. Together with Claim A.6 and Corollary A.7, we have $\mathsf{len}(\mathtt{VIEW}^{r'}) - \mathsf{len}(\mathtt{VIEW}^{r+\Delta}) \geq (1-\delta)\alpha t$. Chain $\mathcal{C}'_{\text{hybrid}}$ is an valid PoS core-chain accepted by an honest PoS-playerP' at round r'. We have $\mathsf{len}(\mathcal{C}'_{\text{hybrid}}) \geq \mathsf{len}(\mathtt{VIEW}^{r'})$.

Putting these together, we get $\mathsf{len}(\mathcal{C}'_{\mathrm{hybrid}}) - \mathsf{len}(\mathcal{C}_{\mathrm{hybrid}}) \geq \mathsf{len}(\mathtt{VIEW}^{r'}) - \mathsf{len}(\mathtt{VIEW}^{r+\Delta}) \geq (1-\delta)\alpha t$ with probability at least $1 - e^{-\Omega(t)}$. The corresponding growth rate is $g = (1-\delta)\alpha$. □

Reminder of Lemma 3.2. *Consider an execution of core-chain protocol Π^{core}, where an honest PoS-player P' is with best local core-chain $\mathcal{C}'$ in round r', and an honest PoS-player P'' with best local core-chain $\mathcal{C}''$ in round r'', and $r'' > r'$. Then we have $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$ where $t = r'' - r'$, $g = (1-\delta)\alpha$, and $\delta > 0$.*

Proof. In order to distinguish the notation clearly, we use $\mathcal{C}'_{\mathrm{hybrid}}$ and $\mathcal{C}''_{\mathrm{hybrid}}$ to denote the PoS core-chains of the best core-chains of P at round r' and r'' in the execution of $\mathsf{HYB}^r(\omega)$. From Claim A.9, we have $\Pr[\mathsf{len}(\mathcal{C}''_{\mathrm{hybrid}}) \geq \mathsf{len}(\mathcal{C}'_{\mathrm{hybrid}}) + g \cdot t] \geq 1 - e^{-\Omega(t)}$ where $t = r'' - r'$, in $\mathsf{HYB}^r(\omega)$. We now turn to the core-chain growth property in $\mathsf{EXEC}_{\Pi^{\mathrm{core}},\mathcal{A},\mathcal{Z}}$. From the definition of hybrid execution, we know that all honest players have same initial status at round r'. We have $\mathsf{len}(\mathcal{C}') = \mathsf{len}(\mathcal{C}'_{\mathrm{hybrid}})$. By Claim A.4, we have $\mathsf{len}(\mathcal{C}'') \geq \mathsf{len}(\mathcal{C}''_{\mathrm{hybrid}})$. It follows that,

$$\begin{aligned} &\Pr[\mathsf{len}(\mathcal{C}'') \geq \mathsf{len}(\mathcal{C}') + g \cdot t] \\ \geq\ &\Pr[\mathsf{len}(\mathcal{C}''_{\mathrm{hybrid}}) \geq \mathsf{len}(\mathcal{C}'_{\mathrm{hybrid}}) + g \cdot t] \\ \geq\ &1 - e^{-\Omega(t)} \end{aligned} \tag{3}$$

where $g = (1-\delta)\alpha$. This completes the proof. □

A.3 Achieving chain quality property

The chain-quality property (Definition 2.5) ensures that the ratio of blocks from honest players in a continuous part of longest core-chain has a lower bound. We show that the number of block-cores produced by the adversarial miners is bounded by the number of their stakes. We demonstrate that the ratio of honest PoS block-cores in an honest player's PoS core-chain is under a suitable lower bound in a sufficient number of rounds with an overwhelming probability.

We note that the discussion of chain-quality property is only for the best chain. From the chain growth property, we know that network delay will discount the effective honest stakes. In the following proof, when we calculate the length of chain we use the α in stead of α_0.

First, we will build the relationship between length of a core-chain and the number of rounds.

Claim A.10. *Consider $\mathsf{REAL}(\omega)$, and $\delta > 0$. Let Z be the number of rounds in which ℓ consecutive block-cores are generated. Then we have $\Pr[Z > (1-\delta)c\ell] > 1 - e^{-\Omega(\ell)}$ where $c = \frac{1}{\alpha+\beta}$.*

Proof. All players can extend $\alpha + \beta$ number of PoS block-cores in a round on average. In order to generate ℓ block-cores, it will consume $\frac{\ell}{\alpha+\beta}$ rounds on average. Let $c = \frac{1}{\alpha+\beta}$, and Z be the number of rounds which generate the ℓ consecutive PoS block-cores. For any $\delta > 0$, by using Chernoff bounds, we have $\Pr[Z \leq (1-\delta)c\ell] \leq e^{-\delta^2 c\ell/3}$. That is, $\Pr[Z > (1-\delta)c\ell] > 1 - e^{-\delta^2 c\ell/3} = 1 - e^{-\Omega(\ell)}$. This completes the proof. □

Now we consider the contribution from honest players in any consecutive block-cores. If the adversarial players want to contribute more PoS block-cores on the core-chain, they will try to generate more PoS block-cores and beat the PoS block-cores from honest players in the competition. Thus, the worst case is the adversarial players make use of all the stakes to generate PoS block-cores and win all of the competition. First, we will prove the core-chain quality property in any t consecutive rounds.

Claim A.11. *Consider $\mathsf{REAL}(\omega)$, and an honest PoS-player P with PoS core-chain $\mathcal{C}$. Consider ℓ consecutive PoS block-cores of $\mathcal{C}$ that are generated from round r to round $r + t$. Assume $\alpha = \lambda\beta$ where $\lambda > 1$. Then we have $\Pr[\mu \geq 1 - (1+\delta)\frac{1}{\lambda}] > 1 - e^{-\Omega(t)}$ for any $\delta > 0$, where μ is the ratio of honest block-cores of the PoS core-chain $\mathcal{C}$.*

Proof. Consider the ℓ consecutive PoS block-cores of $\mathcal{C}$ that are generated from round r to round $r+t$. From Lemma 3.2, we have $\Pr[\ell \geq (1-\delta^*)\alpha \cdot t] \geq 1-e^{-\Omega(t)}$ for any $\delta^* > 0$. Let Y be the number of valid malicious PoS block-cores which are actually generated in t rounds to extend a core-chain. By Chernoff bound, we have

$$\Pr[Y < (1+\delta')\beta \cdot t] > 1 - e^{-\Omega(t)}$$

We then have

$$\Pr\left[\mu \geq \frac{\ell - Y}{\ell}\right] > 1 - e^{-\Omega(t)}$$

That is, By picking δ^* and δ' sufficiently small, we have

$$\Pr\left[\mu \geq 1 - (1+\delta)\frac{1}{\lambda}\right] > 1 - e^{-\Omega(t)}$$

for any $\delta > 0$. This completes the proof. □

Now we are ready to prove the core-chain quality property for consecutive block-cores on a core-chain.

Reminder of Lemma 3.3. *Assume that $\alpha = \lambda\beta$, and $\lambda > 1$. Consider an execution of core-chain protocol Π^{core}, where an honest PoS-player is with core-chain $\mathcal{C}$. If among ℓ consecutive block-cores in chain $\mathcal{C}$, there are ℓ_{good} block-cores that are generated by honest PoS-players, then we have $\Pr\left[\frac{\ell_{\text{good}}}{\ell} \geq \mu\right] \geq 1 - e^{-\Omega(\ell)}$ where $\mu = 1 - (1+\delta)\frac{1}{\lambda}$, and $\delta > 0$.*

Proof. Let t be the number of rounds that the ℓ block-cores are generated. From Claim A.10, we have $\Pr[t > (1-\delta)c\ell] > 1 - e^{-\Omega(\ell)}$. From Claim A.11, the ratio of honest PoS block-cores in t consecutive rounds with ℓ PoS block-cores is $\mu \geq 1-(1+\delta)\frac{1}{\lambda}$ with probability at least $1-e^{-\Omega(t)}$. Putting them together, the probability is at least $1 - e^{-\Omega(\ell)}$. This completes the proof. □

A.4 Achieving common prefix property

We now turn to proving the common prefix property (Definition 2.4) for the core-chain protocol Π^{core}. The concrete statement can be found in Lemma 3.4.

Intuitively, from the assumption that honest players control more resource than the malicious ones, we can see that if the malicious parties maintain a hidden, forked core-chain, and try to extend it by themselves, then the hidden core-chain will be shorter than the public core-chain. From the assumption $\alpha + \beta \ll 1$, we see that in most rounds no new block will be generated. This means all honest players will have the same view in most rounds. Since all honest players will extend the same public chain, the public chain will be the longest one.

Actually, we don't need to assume the strategy of malicious players. The common prefix property holds for all adversary model. We only give the proof idea in this section. The formal proof can be found in the section 4.3.

Reminder of Lemma 3.4. *Assume that $\alpha = \lambda\beta$, and $\lambda > 1$. Consider an execution of core-chain protocol Π^{core}, where two honest PoS-players, P in round r and P' in round r', with the local best core-chains $\mathcal{C}$ and $\mathcal{C}'$, respectively, where $r' \geq r$. Then we have $\Pr[\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'] \geq 1 - e^{-\Omega(\kappa)}$.*

The main proof ideas are as follows:

Considering from round r to r', there are $r'-r$ rounds totally. Let $\mathcal{C}'$ be the best chain in round r'. We will argue that the malicious players can not fork a chain $\mathcal{C}$ from round r with almost same length of $\mathcal{C}'$. For the network delay Δ is small, in most rounds all of the honest players will take the same best chain. This means that in most rounds the honest players will try to extend the same chain. That is only one chain will be extended in a round by honest players. Putting these together, at least one chain of $\mathcal{C}'$ and $\mathcal{C}$ will not be extended in more

than $\frac{r'-r}{2}$ rounds. WLOG, we assume it is $\mathcal{C}$. In these rounds, the best chain will be extended by honest layers and $\mathcal{C}$ will be extended only by malicious players.

We assume $\alpha = \lambda\beta$, where $\lambda > 1$. If $r' - r$ is long, the malicious player can not keep $\mathcal{C}$ be extended with same growing rate with best chain.

Definition A.12 (Divergent length)**.** *Given two different core-chain $\mathcal{C}'$ and $\mathcal{C}''$. Let B be the last common block on $\mathcal{C}'$ and $\mathcal{C}''$. Let ℓ' be the length from B to the end of $\mathcal{C}'$ and ℓ'' be the length from B to the end of $\mathcal{C}''$. The divergent length of $\mathcal{C}'$ and $\mathcal{C}''$ is $\ell = max\{\ell', \ell''\}$.*

Claim A.13. *Let $\alpha = \lambda\beta$, $\lambda > 1$ and $(\alpha + \beta)\Delta \ll 1$, exists $\delta > 0$. Consider $\mathsf{REAL}(\omega)$. Let $\mathcal{C}$ be the best public core-chain in round r. Let $\mathcal{C}'$ be another valid core-chain which is different with $\mathcal{C}$. Let ℓ be the divergent length of $\mathcal{C}$ and $\mathcal{C}'$. We have $\Pr[\mathsf{len}(\mathcal{C}) - \mathsf{len}(\mathcal{C}') > (1-\delta)\ell] > 1 - e^{-\Omega(\ell)}$.*

Proof. Suppose the last common block-core of $\mathcal{C}$ and $\mathcal{C}'$ is generated in round $s = r - t$. From Claim A.10, we have $t > (1-\delta)\frac{\ell}{\alpha+\beta}$ with probability no less than $1 - e^{-\Omega(\ell)}$. Let $X = \mathsf{len}(\mathcal{C}) - \mathsf{len}(\mathcal{C}^s)$ be the length growth of best public core-chain in the t rounds, with Lemma 3.2, we have $X > (1-\delta)\alpha t$ with probability no less than $1 - e^{-\Omega(t)}$ During the t rounds, all the players will generate $(\alpha + \beta)t$ block-cores which are longer than core-chain $\mathcal{C}^s$ on average. With the network delay, this will confuse the honest players $(\alpha+\beta)\Delta t$ rounds on average. That is the honest players may contribute to other core-chain during the confusing rounds. Let Y be the block-cores that the honest players contribute during the confusing rounds. We have $Y = (\alpha + \beta)\Delta t\alpha$ on average. For $(\alpha+\beta)\Delta \ll 1$, we have $Y \ll X$. Let Z be the number of block-cores that malicious players can extend for a core-chain during the t rounds. We have $Z = \beta t$ on average. With Chernoff bounds, we have $Z < (1+\delta)\beta t$ with probability no less than $1 - e^{-\Omega(t)}$. Putting these together, we have $\Pr[X - (Y + Z) > (1-\delta)\frac{\lambda-1}{\lambda+1}\ell] > 1 - e^{-\Omega(t)} = 1 - e^{-\Omega(\ell)}$. For $\lambda > 1$, we have $\Pr[\mathsf{len}(\mathcal{C}) - \mathsf{len}(\mathcal{C}') > (1-\delta)\ell] = \Pr[X - (Y+Z) > (1-\delta)\ell] > 1 - e^{-\Omega(\ell)}$. This completes the proof. □

A.5 Achieving chain soundness property

We now turn to proving the chain soundness property for the core-chain protocol Π^{core}. The concrete statement can be found in Lemma 3.5, and will be restated below. Before providing the formal proof, we here give some informal proof ideas. As in PoW-based blockchain protocols, in our Π^{core}, all players follow the longest chain. That is, the longest chain is the best chain. The malicious players cannot create a chain which grows faster than the best public chain. Therefore, the malicious players cannot mislead new players by providing them a longer chain.

Reminder of Lemma 3.5. *Consider for every round, $\alpha = \lambda\beta$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol Π^{core}. Consider two honest PoS-players, P' and P'' in round r, with the local best core-chains $\mathcal{C}'$ and $\mathcal{C}''$, respectively, where P' is a new player and P'' is an existing player in round r. Then we have $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ and $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$.*

Proof. Let $\mathcal{C}$ be the best public chain in round r. This implies that both P' and P'' have already received the public best $\mathcal{C}$. Let ℓ' be the divergent length of $\mathcal{C}'$ and $\mathcal{C}$. From Claim A.13, we have $\Pr[\mathsf{len}(\mathcal{C}) - \mathsf{len}(\mathcal{C}') \geq (1-\delta)\ell'] > 1 - e^{-\Omega(\ell')}$. If $\mathcal{C}'[\neg\kappa] \not\preceq \mathcal{C}$, we have $\mathsf{len}(\mathcal{C}) > \mathsf{len}(\mathcal{C}')$ with probability no less than $1 - e^{-\Omega(\kappa)}$. This contradicts the fact that P_i already took $\mathcal{C}'$ as the best chain. Therefore, we have $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}$. Similarly, we can have $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}$. Note that $\mathsf{len}(\mathcal{C}') \geq \mathsf{len}(\mathcal{C})$, this means that the divergent length of $\mathcal{C}$ is shorter than $\mathcal{C}'$. We now have $\mathcal{C}[\neg\kappa] \preceq \mathcal{C}'$. Similarly, we can have $\mathcal{C}[\neg\kappa] \preceq \mathcal{C}''$. Putting these together, we obtain $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ and $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$ which completes the proof. □

B Implementing $\mathcal{F}_{\mathsf{rCERT}}$ in the $\mathcal{F}_{\mathsf{RO}}$-hybrid model

We implement functionality $\mathcal{F}_{\mathsf{rCERT}}$ in the $\mathcal{F}_{\mathsf{RO}}$-hybrid model; see Figure 17. Here $\mathcal{F}_{\mathsf{RO}}$ is the random oracle functionality; see Appendix D.2.

PROTOCOL π_{rCERT}

The protocol is parameterized by a difficulty parameter p and a security parameter κ, and a unique digital signature scheme $\Sigma = (\mathsf{uKeyGen}, \mathsf{uSign}, \mathsf{uVerify})$.

Initially, a set $\mathcal{P}_0$ of distinct players are registered, where $\mathcal{P}_0 \subseteq \mathcal{P}$; That is, for each $P \in \mathcal{P}_0$, compute $(\mathsf{sk}_P, \mathsf{pk}_P) \leftarrow \mathsf{uKeyGen}(1^\kappa)$, record the tuple $(P, \mathsf{pk}_P, \mathsf{sk}_P)$.

Stake Election: For each round, each registered party P sets $\mathtt{B}_P := 0$, then proceeds as follows. Upon receiving $(\text{ELECT}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle)$ from the environment $\mathcal{Z}$,

1. If $\mathtt{B}_P = 0$, compute $\sigma := \mathsf{uSign}(\mathsf{sk}_P, \langle h^{\text{prev}}, \mathtt{round}\rangle)$; send $(h^{\text{prev}}, \mathtt{round}, \mathsf{pk}_P, \sigma)$ to the functionality $\mathcal{F}_{\mathsf{RO}}$ and receives h.
   - If $h > \mathtt{T}$ where $\mathtt{T} = p \cdot 2^\kappa$, then set $\mathsf{b} := 0$.
   - Else, set $\mathsf{b} := 1$.

   Set $\mathtt{B}_P := 1$

   Send $(\text{ELECTED}, P, \sigma, h, \mathsf{b})$ to the adversary and the environment.
2. Otherwise, ignore the message.

Block Verification: Upon receiving $(\text{CORE-VERIFY}, P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma, h)$ from the environment $\mathcal{Z}$,

- If $\mathsf{uVerify}(\mathsf{pk}_P, \langle h^{\text{prev}}, \mathtt{round}\rangle, \sigma) = 1$, send $(h^{\text{prev}}, \mathtt{round}, \mathsf{pk}_P, \sigma)$ to the functionality $\mathcal{F}_{\mathsf{RO}}$ and receives h.
  - If $h > \mathtt{T}$ where $\mathtt{T} = p \cdot 2^\kappa$, then set $f := 0$.
  - Else, set $f := 1$.

  send $(\text{CORE-VERIFIED}, (P, \langle h^{\text{prev}}, \mathtt{round}\rangle), \sigma, f)$ to the environment.

Figure 17: Resource certification protocol π_{rCERT}.

C Defending against a full-greedy adversary

In Section 4, we studied greedy strategies: distance-greedy strategies for honest players. We demonstrated that a core-chain protocol using distance-greedy strategies can be proven to be secure against an arbitrary strategy.

In this section, we focus on a much more restricted adversary, the full-greedy adversary. Correspondingly, a relatively simpler strategy called "F-height-greedy strategy" for a non-negative integer F, will be introduced to defend against the full-greedy adversary.

C.1 Height-greedy strategies

Intuitively, a F-height-greedy player will make attempts to extend all chains with big heights; if the height of a chain does not fall behind the longest chain for F blocks, then the chain will be extended. Apparently, when $\mathtt{F} = 0$, only the longest chain will be extended. Let ℓ be the height of the longest chain; when $\mathtt{F} = \ell$, all chains will be extended, and the ℓ-height-greedy strategy is essentially the *full-greedy* strategy.

Definition C.1 (F-height-greedy strategy)**.** *Consider a blockchain protocol execution. Let P be a player of the protocol execution, and T be a tree which consists of chains with the same genesis block, in P's local view.*

Let ℓ be the length of the longest chain at round r. We say the player is $\mathtt{F}$-height-greedy *if, for all r, the player makes attempts to extend all chains with length at least $(\ell - \mathtt{F})$, where $0 \leq \mathtt{F} \leq \ell$.*

If $\mathtt{F} = \ell$, we say the player follows the full-greedy *strategy; if $\mathtt{F} = 0$, we say the player follows the* basic *(greedy) strategy.*

In Figures 18, 19 and 20, different greedy strategies are illustrated; the blocks in the red rectangles will be extended. More concretely, in Figure 18, the players follow the basic strategy (0-height-greedy strategy), i.e., the players only make attempts to extend the longest chain; In Figure 19, players follow the 1-height-greedy strategy; that is, the players attempt to extend the longest chain as well as the chains with only one block behind.

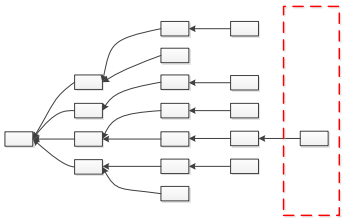

Figure 18: 0-height-greedy

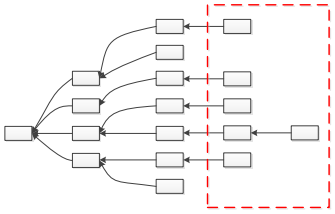

Figure 19: 1-height-greedy

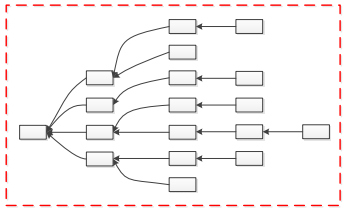

Figure 20: Full-greedy

Next, we introduce stake amplification when a player follows a greedy strategy.

Definition C.2 (Amplification ratio for height-greedy strategies)**.** *Consider a PoS blockchain protocol. Let N_0 be the number of blocks that a group of players P contribute to extend a blockchain in a fixed number of rounds if they follow the 0-height-greedy strategy on average. Let $N_{\mathtt{F}}$ be the number of blocks that the same group of players P contribute to extend a blockchain in the same time period if they follow the $\mathtt{F}$-height-greedy strategy on average. We define the amplification ratio for following the $\mathtt{F}$-height-greedy strategy as $\mathtt{A}^{\star}_{\mathtt{F}} = \frac{N_{\mathtt{F}}}{N_0}$*

Jumping ahead, in next subsections, we will show that the amplification ratio for following the full-greedy strategy is $\mathtt{A}^{\star}_{\text{fully}} = 2.718$, and the amplification ratio for following the 2-height-greedy strategy is $\mathtt{A}^{\star}_{2} = 2.1$. In addition, by definition, $\mathtt{A}^{\star}_{0} = 1$.

C.2 The modified core-chain protocol $\Pi^{\text{core}\star}$

Next, we introduce a new core-chain protocol $\Pi^{\text{core}\star}$ where players follow the $\mathtt{F}$-height-greedy strategy. Details can be found in Figure 21.

Intuitively, the bigger the greedy parameter $\mathtt{F}$ is, the better the chance that the player extend the set of chains. However, the (computation and storage) complexity of the protocol is proportional to the greedy parameter $\mathtt{F}$. In practice, we can choose $\mathtt{F} = 2$ (and the size of set $\mathbb{C}_{\text{best}}$ is 10 on average).

PROTOCOL $\Pi^{\text{core}\star}$

Initially, a set $\mathcal{P}_0$ of players are registered to the functionality $\mathcal{F}_{\mathsf{rCERT}}$, where $\mathcal{P}_0 \subseteq \mathcal{P}$.
Initially, for each $P \in \mathcal{P}$, set $\mathcal{C} := \emptyset$, and $state := \emptyset$.

Upon receiving message (INPUT-STAKE, P) from the environment $\mathcal{Z}$ at round $\mathtt{round}$, the PoS-player $P \in \mathcal{P}$, with local state $state$, proceeds as follows.

1. *Select the best local PoS core-chain:*

   Let $\mathbb{C}$ be the set of core-chains collected from $\mathcal{F}_{\mathsf{NET}}$.

   Compute $\mathbb{C}_{\text{best}} := \mathtt{F\text{-}BestCore}^{\star}(\mathbb{C} \cup \{\mathcal{C}\}, \mathtt{round})$. (F-BestCore* will return a best chain set)

2. *Attempt to extend PoS core-chain:*

   For each $\mathcal{C} \in \mathbb{C}_{\text{best}}$, do: (Each player will extend all chains in the best chain set)

   - Set $\ell := \mathsf{len}(\mathcal{C})$.
   - Parse $\mathcal{C}[\ell]$ as $\langle\langle h_\ell^{\text{prev}}, \mathtt{round}_\ell, P_\ell, \sigma_\ell\rangle, h_\ell\rangle$.
   - *Stake election:* Send (ELECT, $P, \langle h_\ell, \mathtt{round}\rangle$) to functionality $\mathcal{F}_{\mathsf{rCERT}}$, and receive (ELECTED, $P, h_{\ell+1}, \sigma, \mathsf{b}$) from $\mathcal{F}_{\mathsf{rCERT}}$.
   - *If* $\mathsf{b} = 1$, *generate a new block-core:* Set the new block-core $B := \langle\langle h_\ell, \mathtt{round}, P, \sigma\rangle, h_{\ell+1}\rangle$, and set $\mathcal{C} := \mathcal{C} \| B$, and $state := state \cup \{\mathcal{C}\}$, and then send (BROADCAST, $\mathcal{C}$) to $\mathcal{F}_{\mathsf{NET}}$.

   Return (RETURN-Stake, P) to the environment $\mathcal{Z}$.

Figure 21: Our proof-of-stake core-chain protocol $\Pi^{\text{core}\star}$ in the $\{\mathcal{F}_{\mathsf{rCERT}}, \mathcal{F}_{\mathsf{NET}}\}$-hybrid model. (See Figure 22 for the subroutine F-BestCore*.)

SUBROUTINE F-BestCore*

The subroutine F-BestCore* is allowed to access to the functionality $\mathcal{F}_{\mathsf{rCERT}}$, and with input $(\mathbb{C}', \mathtt{round}')$.
For every chain $\mathcal{C} \in \mathbb{C}'$, and proceed as follows.

1. Set $\ell := \mathsf{len}(\mathcal{C})$.
2. For i from ℓ down to 1, verify block-core $\mathcal{C}[i]$, as follows.
   - Parse $\mathcal{C}[i]$ into $\langle\langle h_i^{\text{prev}}, \mathtt{round}_i, P_i, \sigma_i\rangle, h_i\rangle$.
     Parse $\mathcal{C}[i-1]$ into $\langle\langle h_{i-1}^{\text{prev}}, \mathtt{round}_{i-1}, P_{i-1}, \sigma_{i-1}\rangle, h_{i-1}\rangle$.
   - If $\mathtt{round}_i < \mathtt{round}'$ and $\mathtt{round}_{i-1} < \mathtt{round}_i$, execute:
     - If $h_i^{\text{prev}} \neq h_{i-1}$, remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.
     - Else send (CORE-VERIFY, $P_i, \langle h_i^{\text{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i$) to $\mathcal{F}_{\mathsf{rCERT}}$.
       Upon receiving message (CORE-VERIFIED, $P_i, \langle h_i^{\text{prev}}, \mathtt{round}_i\rangle, \sigma_i, h_i, f_i$) from $\mathcal{F}_{\mathsf{rCERT}}$, if $f_i = 0$ remove this core-chain $\mathcal{C}$ from $\mathbb{C}'$.

     Otherwise, remove the core-chain $\mathcal{C}$ from $\mathbb{C}'$.

Let $\mathcal{C}_{\text{best}}$ be the longest core-chain in $\mathbb{C}'$ and $\ell := \mathsf{len}(\mathcal{C}_{\text{best}})$.
Set $\mathbb{C}_{\text{best}} := \emptyset$.
For any chain $\mathcal{C} \in \mathbb{C}'$, do:

- if $\mathsf{len}(\mathcal{C}) \geq \ell - \mathtt{F}$, then set $\mathbb{C}_{\text{best}} := \mathbb{C}_{\text{best}} \cup \{\mathcal{C}\}$. ($\mathtt{F}$ is a small constant, $\mathtt{F} = 2$ typically.)

Then return $\mathbb{C}_{\text{best}}$ as the output.

Figure 22: The core-chain set validation subroutine F-BestCore*.

C.3 Security analysis

Remark C.3. *We first remark that the protocol we constructed here is* not *secure against an arbitrary adversary. Intuitively, an arbitrary adversary can play the "balancing attacks" by always supporting the shorter chains with the goal of creating two or multiple chains with long fork. In the remaining of this section, we will deal with a much more restricted adversary who play full-greedy strategy.*

Note that the security properties of protocol Π^{core} have be proven under the assumption of honest majority of stakes based on α and β. Now, we can prove the security properties of the modified core-chain protocol $\Pi^{\text{core}\star}$ but under the assumption of honest majority of *effective stakes* based on $\alpha^\star$ and $\beta^\star$. Here $\alpha^\star = 2.1\alpha$ and $\beta^\star = 2.718\beta$. We note that here $\alpha^\star \ll 1$ and $\beta^\star \ll 1$ (since $\alpha \ll 1$ and $\beta \ll 1$).

Theorem C.4. *Consider core-chain protocol $\Pi^{\text{core}\star}$ where honest players follow the 2-height-greedy strategy while adversarial players follow the full-greedy strategy; in addition, all players have their stake registered without being aware of the state of the protocol execution. If $\alpha^\star = \lambda\beta^\star$, $\lambda > 1$, then the protocol $\Pi^{\text{core}\star}$ can achieve chain growth, chain quality, common prefix and chain soundness properties.*

Defining $\beta^\star$. We use $\beta^\star$ to denote the equivalent expected number of blocks that malicious players can extend a chain in a round. We note that the full-greedy of height-greedy is the same as the full-greedy of distance-greedy, hence, $\beta^\star = \beta^\circ = e\beta$.

Defining $\alpha^\star$. We next show a lemma stating that, honest players, by following the 2-height-greedy strategy, can obtain extra advantage for extending the public chain.

Lemma C.5. *Consider core-chain protocol $\Pi^{\text{core}\star}$ where greedy parameter is $\mathtt{F} = 2$. Assume malicious players do not help honest players to extend any chain. Let t be the number of rounds for extending the longest chain with one new block. Then we have $t \approx \frac{1}{2.1\alpha}$ on average.*

Proof. Assume that the length of the longest chain at round r is l, and the length of the longest chain at round r' is $l+1$. That is $t = r' - r$. We further assume, in round r, the number of chains with length $l-1$ is x, and the number of chains with length $l-2$ is y . (We also assume only one chain, i.e., the longest chain, is with length l.)

For simplicity, we assume the chain will grow with a steady rate and we here only investigate the average case. In this case with the above assumption, in round r' the length of the longest chain will increase to $l+1$, and the number of chains with length l will be x and the number of chains with length $l-1$ will be y.

Consider the longest chain ,with length $l+1$ in round r', it is extended by some players during the t rounds from a chain with length l. At round r there is 1 chain with length l and at round r' there are x chains with length l. On average, we have $\frac{1+x}{2}\alpha t = 1$. With similar arguments, considering the chain with length l in round r', they are extended from a chain with length l. On average, we have $\frac{x+y}{2}\alpha t = x - 1$. Finally, considering the chain with length $l-1$ in round r', they are extended from a chain with length l. On average, we have $y\alpha t = y - x$. Putting these together, we have:

$$\begin{cases} \frac{1+x}{2}\alpha t = 1 \\ \frac{x+y}{2}\alpha t = x - 1 \\ y\alpha t = y - x \end{cases} \tag{4}$$

That is $\alpha t \approx 0.48$. We get $t \approx \frac{1}{2.1\alpha}$. □

This lemma shows that, if the honest players follow the 2-height-greedy strategy they will extend the longest chain with 1 block in $t \approx \frac{1}{2.1\alpha}$ rounds on average. If we use $\alpha^\star$ to denote the equivalent expected number of blocks that the honest players will extend a chain in a round, we have $\alpha^\star = 2.1\alpha$. That is, the amplification ratio (of extending chains by honest players) is $\mathtt{A}_2^\star = 2.1$.

C.3.1 Honest majority for ensuring security of greedy strategy

We can obtain the following result (see Table 2): when honest players follow the 0-height-greedy (or, 0-height-greedy, 2-height-greedy, respectively) strategy, and malicious players follow the 0-height-greedy (or, fully-greedy, fully-greedy, respectively) strategy, to ensure the security of the core-chain protocol, 51% (or, 73%, 57%, respectively) majority of stakes must be honest.

Table 2: Honest majority for ensuring security

Honest Players	Malicious Players	Honest Majority (ensuring security)
0-height-greedy	0-height-greedy	51%
0-height-greedy	full-greedy	73%
2-height-greedy	full-greedy	57%

C.3.2 Security properties in the presence of the full-greedy adversary

Based on the above discussions, we have $\alpha^\star = \mathtt{A}_2^\star\alpha \ll 1$ and $\beta^\star = e\beta \ll 1$, where $\mathtt{A}_2^\star = 2.1$. Here we assume, $\alpha \ll 1$ and $\beta \ll 1$. We have $\alpha^\star \ll 1$ and $\beta^\star \ll 1$. Then similarly, the security properties will still hold if we change the previous assumption of honest majority of stakes based on α and β, into the assumption of honest majority of *effective stakes* based on $\alpha^\star$ and $\beta^\star$.

Chain growth. Honest players will extend blockchain faster if they follow, not the basic strategy, but the F-height-greedy strategy, where $\mathtt{F} > 0$, and the chain growth rate will increase. We have:

Corollary C.6 (Chain growth)**.** *Consider core-chain protocol $\Pi^{\text{core}\star}$ in the presence of the full-greedy adversary. Consider an honest player P' with best local core-chain $\mathcal{C}'$ in round r', and an honest player P'' with best local core-chain $\mathcal{C}''$ in round r'', where $r'' > r'$. Then we have $\Pr\left[\mathsf{len}(\mathcal{C}'') - \mathsf{len}(\mathcal{C}') \geq g \cdot t\right] \geq 1 - e^{-\Omega(t)}$ where $t = r'' - r'$, $g = (1-\delta)\alpha^\star$, and $\delta > 0$.*

Chain quality. A F-height-greedy adversary can extend a chain faster than basic adversary, when $\mathtt{F} > 0$. Intuitively, this will reduce the chain quality. However, from Lemma 4.10, the number of blocks from malicious players on any chain is bounded. If we assume the honest players extend chains faster than the malicious players, the chain quality property will still hold as in Lemma 3.3.

Corollary C.7 (Chain quality)**.** *Consider $\alpha^\star = \lambda\beta^\star$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\star}$ with a full-greedy adversary. Consider an honest player with core-chain $\mathcal{C}$. Consider that ℓ consecutive block-cores of $\mathcal{C}$, where ℓ_{good} block-cores are generated by honest players. Then we have $\Pr\left[\frac{\ell_{good}}{\ell} \geq \mu\right] \geq 1 - e^{-\Omega(\ell)}$ where $\mu = 1 - (1+\delta)\frac{1}{\lambda}$.*

Common prefix. We note that, if the full-greedy adversary cannot extend chains faster than the F-height-greedy honest players, he cannot generated a longer forked chain to violate the common prefix property. From Lemma 3.4 we have:

Corollary C.8 (Common prefix)**.** *Consider $\alpha^\star = \lambda\beta^\star$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\star}$ with the full-greedy adversary. Consider two honest players, P in round r and P' in round r', with the local best core-chains $\mathcal{C}$, $\mathcal{C}'$, respectively, where $r' \geq r$. Then we have $\Pr\left[\mathcal{C}[1,\ell] \preceq \mathcal{C}'\right] \geq 1 - e^{-\Omega(\kappa)}$ where $\ell = \mathsf{len}(\mathcal{C}) - \Theta(\kappa)$.*

Chain soundness. As in Lemma 3.5, the protocol can achieve chain soundness property. Otherwise the common prefix property will be violated.

Corollary C.9 (Chain soundness)**.** *Assume for every round, $\alpha^\star = \lambda\beta^\star$, $\lambda > 1$, and $\delta > 0$. Consider core-chain protocol $\Pi^{\text{core}\star}$ with the full-greedy adversary. Consider two honest players, P' and P'' in round r, with the local best core-chains $\mathcal{C}'$ and $\mathcal{C}''$, respectively, where P' is a new player and P'' is an existing player in round r. Then we have $\mathcal{C}'[\neg\kappa] \preceq \mathcal{C}''$ and $\mathcal{C}''[\neg\kappa] \preceq \mathcal{C}'$.*

D Supplementary material: Basic Primitives/Functionalities

D.1 Network communication $\mathcal{F}_{\mathsf{NET}}$

The underlying communication for blockchain protocols are formulated via a functionality $\mathcal{F}_{\mathsf{NET}}$ which captures the atomic unauthenticated "send-to-all" broadcast in a semi-synchronous communication setting. The functionality is parameterized by an upper bound Δ on the network latency, and interacts with players under the direction of the adversary. More concretely, the functionality proceeds as follows. Whenever it receives a message from a player, it would contact the adversary to ask the adversary to specify the delivery time for the message. Note that, if the specified delivery time exceeds the delay upper bound Δ, the functionality would not follow the adversary's instruction, and only delay the message to a maximum number of Δ rounds. That said, no messages are delayed more than Δ rounds. In addition, the adversary could read all messages sent by all honest players before deciding his strategy; the adversary may "spoof" the source of a message they transmit and impersonate the (honest) sender of the message. The functionality $\mathcal{F}_{\mathsf{NET}}$ is formally described in Figure 23.

FUNCTIONALITY $\mathcal{F}_{\mathsf{NET}}$

The functionality is parametrized by Δ, and interacts with a set $\mathcal{P}$ of PoS-players, and the adversary.

- Upon receiving $(\text{BROADCAST}, m)$ from a party P at round r where $P \in \mathcal{P}$, send $(\text{BROADCAST}, m)$ to $\mathcal{S}$ and record (P, m, b, r) where $b = 0$.
- Upon receiving (DELAY, m, P', t) from $\mathcal{S}$ where $P' \in \mathcal{P}$ (here, the adversary can "spoof" the source of the message), then
  - If there is a record (P, m, b, r) such that $b = 0$ and $t \leq \Delta$, then send $(\text{MESSAGE}, P', m)$ to all other PoS-players at round $r + t$ and reset $b := 1$.
  - Else, if $t > \Delta$, send $(\text{MESSAGE}, P', m)$ to all other PoS-players at round $r + \Delta$ and reset $b := 1$.
  - Else, ignore the message.

Figure 23: Network functionality $\mathcal{F}_{\mathsf{NET}}$.

D.2 Random Oracle Functionality $\mathcal{F}_{\mathsf{RO}}$

The random oracle model (e.g., [7]) captures an idealization of a hash function. We here present the random oracle functionality $\mathcal{F}_{\mathsf{RO}}$ that has been defined in [32].

D.3 Unique signature scheme.

Unique signature scheme was introduced in [40], which consists of four algorithms, a randomized key generation algorithm uKeyGen, a deterministic key verification algorithm uKeyVer, a deterministic signing algorithm uSign, and a deterministic verification algorithm uVerify. We expect for each verification key there exists only one signing key. We also expect for each pair of message and verification key, there exists only one signature. We have the following definition.

FUNCTIONALITY $\mathcal{F}_{\mathsf{RO}}$

The functionality $\mathcal{F}_{\mathsf{RO}}$ is parameterized by a security parameter κ, and interacts with a set $\mathcal{P}$ of parties, and an adversary. The functionality keeps a list L (which is initially empty) of pairs of bitstrings.

1. Upon receiving a value (m) (with $m \in \{0,1\}^*$) from some party $P \in \mathcal{P}$ or from the adversary, proceed as follows.
   - If there is a pair $(m, \tilde{h})$ for some $\tilde{h} \in \{0,1\}^\kappa$ in the list L,set $h := \tilde{h}$.
   - if there is no such pair, choose uniformly $h \in \{0,1\}^\kappa$ and store the pair (m, h) in L.

   Once h is set, reply to the requesting party with (h).

Figure 24: Random oracle functionality $\mathcal{F}_{\mathsf{RO}}$.

Definition D.1. *We say* $(\mathsf{uKeyGen}, \mathsf{uKeyVer}, \mathsf{uSign}, \mathsf{uVerify})$ *is a unique signature scheme, if it satisfies:*

Correctness of key generation: Honestly generated key pair can always be verified. More formally, it holds that

$$\Pr\left[(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{uKeyGen}(1^\kappa) \ : \ \mathsf{uKeyVer}(\mathrm{PK}, \mathrm{SK}) = 1\right] \geq 1 - \mathsf{negl}(\kappa)$$

Uniqueness of signing key: There does not exist two different valid signing keys for a verification key. More formally, for all PPT *adversary* $\mathcal{A}$*, it holds that*

$$\Pr\left[\begin{array}{l}(\mathrm{PK}, \mathrm{SK}_1, \mathrm{SK}_2) \leftarrow \mathcal{A}(1^\kappa) \ : \\ \mathsf{uKeyVer}(\mathrm{PK}, \mathrm{SK}_1) = 1 \wedge \mathsf{uKeyVer}(\mathrm{PK}, \mathrm{SK}_2) = 1 \wedge \mathrm{SK}_1 \neq \mathrm{SK}_2\end{array}\right] \leq \mathsf{negl}(\kappa)$$

Correctness of signature generation: For any message x*, it holds that*

$$\Pr\left[\begin{array}{l}(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{uKeyGen}(1^\kappa); \sigma := \mathsf{uSign}(\mathrm{SK}, x) \\ \ : \ \mathsf{uVerify}(\mathrm{PK}, x, \sigma) = 1\end{array}\right] \geq 1 - \mathsf{negl}(\kappa)$$

Uniqueness of signature generation: For all PPT *adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{l}(\mathrm{PK}, x, \sigma_1, \sigma_2) \leftarrow \mathcal{A}(1^\kappa) \ : \\ \mathsf{uVerify}(\mathrm{PK}, x, \sigma_1) = 1 \wedge \mathsf{uVerify}(\mathrm{PK}, x, \sigma_2) = 1 \wedge \sigma_1 \neq \sigma_2\end{array}\right] \leq \mathsf{negl}(\kappa)$$

Unforgeability of signature generation: For all PPT *adversary* $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{l}(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{uKeyGen}(1^\kappa); (x, \sigma) \leftarrow \mathcal{A}^{\mathsf{uSign}(\mathrm{SK}, \cdot)}(1^\kappa) \ : \\ \mathsf{uVerify}(\mathrm{PK}, x, \sigma) = 1 \wedge (x, \sigma) \notin Q\end{array}\right] \leq \mathsf{negl}(\kappa)$$

where Q *is the history of queries that the adversary* $\mathcal{A}$ *made to signing oracle* $\mathsf{uSign}(\mathrm{SK}, \cdot)$*.*

Remark D.2 (Instantiations for the unique signature scheme)**.** *Efficient instantiations can be found in literature. For example, the well-known BLS signature [11] can be a good candidate.*