FairPoS: Input Fairness in Proof-of-Stake with Adaptive Security

James Hsin-yu Chiang[1], Bernardo David[2], Ittay Eyal[3], Tiantian Gong[4]

[1] Technical University of Denmark, Denmark
jchi@dtu.dk
[2] IT University of Copenhagen, Denmark
bernardo@bmdavid.com
[3] Technion, IC3, Haifa, Israel
ittay@technion.ac.il
[4] Purdue University, West Lafayette, USA
tg@purdue.edu

Abstract. We present "FairPoS", the first blockchain protocol that achieves input fairness with adaptive security. Here, we introduce a novel notion of "input fairness": the adversary cannot learn the plain-text of any finalized client input *before* it is include in a block in the chain's common-prefix. Should input fairness hold, input ordering attacks which depend on the knowledge of plain-text of client inputs are thwarted. In FairPoS, input fairness with adaptive security is achieved by means of the delay encryption scheme of DeFeo *et al.* [9], a recent cryptographic primitive related to time-lock puzzles, allowing *all* client inputs in a given round to be encrypted under the same key, which can only be extracted after enough time has elapsed. In contrast, alternative proposals that prevent input order attacks by encrypting user inputs are not adaptively secure as they rely on small static committees to perform distributed key generation and threshold decryption for efficiency's sake. Such small committees are easily corrupted by an adaptive adversary with a corruption budget applicable over a large set of participants in a permissionless blockchain system. The key extraction task in delay encryption can, in principle, be performed by any party and is secure upon adaptive corruption, as no secret key material is learned. However, the key extraction requires highly specialized hardware in practice. Thus, FairPoS requires resource-rich, staking parties to insert extracted keys to blocks which enables light-clients to decrypt past inputs. Note that naive application of key extraction can result in chain stalls lasting the *entire* key extraction period. In FairPoS, this is addressed by a novel *longest-extendable-chain* rule. We formally prove that FairPoS achieves input fairness and the original security of Ouroborous Praos against an adaptive adversary.

1 Introduction

Blockchain protocols permit the elected leader of each round to produce a block containing an ordered list of transactions chosen by its leader. This ordering privilege is exploited in front-running [12], where adversarial inputs can be interleaved with honest inputs to extract financial value from the honest victim in

applications such as automatic market makers [4]. Such behaviour financially penalizes the honest user, but also generates excess demand for block-space since front-running attacks [4] always require additional inputs from the adversary, thereby inflicting block congestion, as acutely observed on Avalanche [2].

We introduce FairPoS which achieves a novel notion of *input fairness* whilst retaining the asymptotic security of Ouroborous Praos [13]. Informally, input fairness ensures that the plain-text content of any finalized input (in the common-prefix) could not have been observed by the adversary prior to its finalization. FairPoS achieves this by encrypting inputs with a delay encryption scheme by DeFeo et al. [9], which is similar to time-lock puzzles [28], but allows all client inputs of a given round to be encrypted under the same key, thereby requiring only a single key extraction for each block. The *extraction* procedure to recover the decryption key is parameterized to run in at least time d, and can be be performed by any party with access to specialized hardware to perform extractions in d time. This preserves adaptive-security, as no relevant key material is learned upon corruption of an honest party.

Still, it is not practical for light-clients or non-staking parties to perform key extraction. Firstly, we expect only resource-rich participants to have access to the specialized hardware [1] necessary to perform extractions in exactly d time. Secondly, any party joining the protocol would need to perform key extractions for all blocks beginning from genesis, which becomes increasingly impractical at higher chain lengths. In FairPoS, staking parties must insert the extracted keys from past inputs into blocks within a fixed schedule, thus ensuring decryption keys are made publicly available in lock-step with chain growth. We note that adversarial delay of blocks propagation can impede chain growth if honest parties cannot finish key extractions on time due to delayed arrival. In FairPoS, such attacks are addressed with a novel *longest-extractable-chain* rule, which asserts a notion of *timeliness* on the arrival of blocks, ensuring that honest leaders can complete the key extraction on schedule.

Comparison to related work. Recent proposals achieve a similar notion with distributed key generation and threshold encryption committees [5,26,27], where a small subset of protocol participants jointly generate a public key, to which user inputs are encrypted in each round. Inputs are subsequently decrypted by the same committee when the inputs are finalized. This approach is not secure against an adaptive adversary, unlike the underlying permissionless blockchains.

Protocols such as Bitcoin [16] and Ouroborous Praos [13] achieve security against an adversary that can adaptively corrupt parties in a large participant set as the protocol execution progresses; performing any additional interactive cryptographic protocol over such a large set of intermittently available parties is not practical, so instead, committees assigned to the protocol task can be elected from the larger global set of parties executing the blockchain protocol [18]. For the *interactive* task of performing distributed key generation and threshold decryption, however, the *adaptive* adversary can easily identify the active committee parties once the first protocol message is sent. Upon corruption it will learn

secret key material, which cannot be erased by parties as it must be maintained as interactive protocol state[5].

Another line of research [23,22,21,10] proposes a notion of *fair input ordering*. A block leader will order inputs based on their order of arrival. However, fair ordering is only meaningful in a setting with a secure connection between client and round leaders: in a peer-to-peer gossip network setting common in massively distributed permissionless blockchain protocols, the receipt-order of messages is adversarially controlled, rendering the notion of fair ordering highly impractical. A secure connection with the next block leader implies public knowledge of its identity, contradicting adaptive security again.

Overview. In Section 2, we introduce Delay Encryption and an abstract model of an Ouroborous Praos execution (δ-PoS). In Section 3, we then define our proposed notion of Input Fairness and extend the PoS model with delay encryption and a novel "longest-extendable-chain" selection rule to obtain a formal model of FairPoS. In Section 4, we prove that FairPoS achieves input fairness whilst maintaining the asymptotic security of Ouroborous Praos (PoS) against an adaptive adversary. Proofs of stated theorems and lemmas are provided in Appendix D.

2 Preliminaries

2.1 Delay Encryption

The delay encryption (DE) scheme by De Feo *et al.* [9] consists of the following four algorithms: A global DE.Setup parameterized with a security parameter $\lambda \in \{0,1\}^*$ and delay parameter d generates public encryption (DE.pk) and extraction (DE.ek) keys. In each round, a public session $\mathsf{id} \in \{0,1\}^*$ is sampled, and DE.Encaps can be used to generate a pair (c,k) of a ciphertext c and a key k corresponding to id and the encryption key DE.pk. The DE.Extract algorithm runs in at least d time, and returns a session key idk, with which the DE.Decaps algorithm can compute a key k from ciphertext c for all (c,k) generated with the same session id and public paramaters from a given setup.

1. $\mathsf{DE.Setup}(\lambda, d) \to (\mathsf{DE.ek}, \mathsf{DE.pk})$
2. $\mathsf{DE.Encaps}(\mathsf{DE.pk}, \mathsf{id}) \to (c,k)$
3. $\mathsf{DE.Extract}(\mathsf{DE.ek}, \mathsf{id}) \to \mathsf{idk}$
4. $\mathsf{DE.Decaps}(\mathsf{DE.pk}, \mathsf{id}, \mathsf{idk}, c) \to k$

Delay encryption is an isogeny-based delay protocol, and similar to [14] is built from isogeny walks in graphs of pairing friendly supersingular elliptic curves. In implementations [14], such isogeny evaluations occupy memory space in the terabytes. Parties performing *Extract* are expected to deploy specialized FPGA hardware [1] in order to achieved the parameterized extraction time.

[5] We note a line of work which achieves adaptive security in large scale cryptographic protocols via anonymous committees [7,19,11,17,15]. However, the efficiency of such approaches remains impractical. We consider this as an orthogonal line of research.

2.2 Longest-chain PoS model and security

We present a model of *longest-chain proof-of-stake* protocols, formalized by the Ouroborous line of work [25,13,3] and subsequent improvements [8,24]. We adopt modelling approach in [25,13,3,24], where the PoS protocol is modelled by two orthogonal components: the first describes the *leader election process* and the second part models the views of blockchain trees which result from a protocol execution *induced by a given leader schedule*.

Idealized leader elections. Time in PoS is divided into units named slots, each capturing the duration of a single protocol round. In a given round, a party with relative stake $\alpha \in (0, 1]$ becomes a slot leader for a given slot with probability

$$\phi(\alpha) = 1 - (1 - f)^{\alpha}$$

where parameter *active slot coefficient* f is the probability that a leader holding all stake will be elected leader in given slot: importantly, $\phi(\alpha)$ is maintained even if share α is split amongst multiple, virtual parties (eq. 2 in [13]). Let a characteristic string w be defined as a sequence of leader election results, where an election result at slot t is defined as follows.

$$w_t = \begin{cases} 0 & \text{a single honest leader} \\ 1 & \text{multiple honest leaders / adversarial leader} \\ \perp & \text{no leader} \end{cases}$$

In PoS [13], leader election is modelled by sampling characteristic strings from an idealized, *dominant distribution* $\mathcal{D}_{\alpha}^{f}$ that is strictly *more adversarial* than the *true* setting where the *adaptive adversary* corrupts up to $(1 - \alpha)$ of the stake during the protocol execution (Theorem 8 in [13]). Thus, any security that holds in PoS executions induced by characteristic strings sampled from $\mathcal{D}_{\alpha}^{f}$ must also hold in the true protocol execution against an adaptive adversary dynamically corrupting up to $1 - \alpha$ stake.

Definition 1 (Dominant distribution $\mathcal{D}_{\alpha}^{f}$ (Definition 11 in [13])). *For an adaptive adversary corrupting up to $1-\alpha$ stake fraction and active slot coefficient $f \in [0, 1)$, the dominant distribution $\mathcal{D}_{\alpha}^{f}$ is defined by the following probabilities:*

$$p_{\perp} = 1 - f \qquad p_0 = \phi(\alpha) \cdot (1 - f) \qquad p_1 = 1 - p_{\perp} - p_1 \tag{1}$$

Definition 2 (Blocks, chains, trees and branches). *A block $B = (\mathsf{sl}, \mathsf{st}, \mathsf{d}, \mathsf{ldr})$ generated at slot sl contains state $\mathsf{st} \in \{0, 1\}^*$, data $\mathsf{d} \in \{0, 1\}^*$ and party ldr that generated B and was the leader of slot sl. A chain is a sequence of blocks $B_0, ..., B_n$ associated with a strictly increasing sequence of slots, where B_0 is the genesis block, and the state of B_i is $H(B_{i-1})$, $H(\cdot)$ denoting a collision-resistant hash function. We write $\mathcal{C}.\mathsf{tip}$ to denote the block at the tip of chain $\mathcal{C}$ and $\mathcal{C}_j$ to denote a block $B \in \mathcal{C}$ such that $B.\mathsf{sl} = j$. If such a block does not exist, $\mathcal{C}_j = \perp$. Let $\mathcal{C}^{\lceil k}$ denote the chain obtained from $\mathcal{C}$ by removing the last k blocks. Multiple chains form a tree if their blocks share state. A branch $\mathcal{B}$ in a tree $\mathcal{T}$ is a chain*

which ends with a leaf block. We write $\mathcal{C} \preceq \mathcal{B}$ to indicate $\mathcal{C}$ is a prefix of $\mathcal{B}$. When quantifying over all chains in a tree, $\forall \mathcal{C} \in \mathcal{T}$, we quantify over all prefixes of all tree branches. Let $\mathsf{len}(\mathcal{C})$ denote the number of blocks in chain $\mathcal{C}$.

δ-PoS execution model

The evolution of a PoS execution state $\Gamma_t = \left(\{\mathcal{T}^{(i)}, \mathbf{m}^{(i)}\}_{i\in\mathcal{H}}, \mathcal{T}^{\mathcal{A}}\right)$ in a single protocol round $\Gamma_t \xrightarrow{w_{t+1}} \Gamma_{t+1}$ induced by environment $\mathcal{Z}$, adversary $\mathcal{A}$ and characteristic string w occurs as follows.

For local tree views $\{\mathcal{T}^{(i)}\}_{i\in\mathcal{H}}$ and $\mathcal{T}^{\mathcal{A}}$:

T0. $\mathcal{T}_0^{(i)}$ consists of the genesis block.

T1. If $w_{t+1} = 0$, a single honest party runs **Extend** on the longest chain $\mathcal{C}$ in $\mathcal{T}^{(i)}$, upon which the extended chain is added to $\mathcal{T}^{(i)}$ and $\mathcal{T}^{\mathcal{A}}$.

T2. At any time during the round, adversary $\mathcal{A}$ may:

- **a)** Run **Extend** on a chain in $\mathcal{T}^{\mathcal{A}}$ for slot $t+1$ if $w_{t+1} = 1$, upon which the updated chain is added to $\mathcal{T}^{\mathcal{A}}$.
- **b)** Update any honest tree view $\mathcal{T}^{(i)}$ with a chain from $\mathcal{T}^{\mathcal{A}}$ or an honest message queue $\{\mathbf{m}^{(i)}\}_{i\in\mathcal{H}}$ (see **M2**).

For honest message queues $\{\mathbf{m}^{(i)}\}_{i\in\mathcal{H}}$:

M1. For a chain $\mathcal{C}'$ extended by honest party i during the round (**T2**), the entry $(\mathcal{C}', \mathcal{H})$ is added to local message queue $\mathbf{m}^{(i)}$.

M2. At any time during the round, adversary $\mathcal{A}$ may deliver a chain from an entry $(\mathcal{C}, \mathcal{P}) \in \mathbf{m}^{(i)}$ to a subset of honest users $\mathcal{I} \subseteq \mathcal{H}$:

- **a)** The entry $(\mathcal{C}, \mathcal{P})$ in $\mathbf{m}^{(i)}$ is updated to $(\mathcal{C}, \mathcal{P}')$, where $\mathcal{P}' = \mathcal{P} \backslash \mathcal{I}$.
- **b)** Each $(\mathcal{C}, \mathcal{P}) \in \mathbf{m}^{(i)}$ must be delivered to all honest parties $\mathcal{H}$ by slot $\mathcal{C}.\mathsf{sl} + \delta$, and is removed from $\mathbf{m}^{(i)}$ when delivery is completed.

Extend. To extend $\mathcal{C}$, the party i generates $B = (t+1, H(\mathcal{C}.\mathsf{tip}), d, i)$ containing an *ordering* of inputs $d = \{\mathsf{in}_i\}_{i\in[m]}$ provided by $\mathcal{Z}$ for slot t.

Fig. 1. δ-PoS model induced by environment $\mathcal{Z}$ and characteristic string w.

A model of δ-PoS exections. As in [16,25,13,3], we model the execution of PoS initiated upon the activation of an environment $\mathcal{Z}$, which spawns both honest parties $\mathcal{H}$ and an adversary $\mathcal{A}$. Upon each activation by the environment, each party executes the protocol according to Figure 1, which precisely models the adversarial powers to influence the round-wise evolution of block tree structures in the local view parties as the full PoS protocol in [13], but omits details such as block proofs, signatures or individual leader election procedures such as evaluation of verifiable random functions. A given characteristic string w *induces* executions of our δ-PoS model that generate local tree structures identical to those resulting from a full PoS [13] protocol execution that *induces* a leader election sequence consistent with w and activates the same parties and adversarial actions.

Let the protocol execution state Γ_t in slot t, consist of honest party states, including the local block tree view $\mathcal{T}^{(i)}$ and the outbound message queue $\mathbf{m}^{(i)}$ for each honest party $i \in \mathcal{H}$. Further, let Γ_t include the blockchain tree view $\mathcal{T}^{\mathcal{A}}$ of the adversary.

$$\Gamma_t = \left(\{\mathcal{T}^{(i)}, \mathbf{m}^{(i)}\}_{i \in \mathcal{H}}, \mathcal{T}^{\mathcal{A}}\right) \tag{2}$$

The outbound message queue $\mathbf{m}^{(i)} = \{(\mathcal{C}, \mathcal{P}), ...\}$ in Γ_t is the set of broadcast, yet undelivered chains previously sent by honest party i. For each entry $(\mathcal{C}, \mathcal{P}) \in \mathbf{m}^{(i)}$, $\mathcal{C}$ was initially broadcast and added to the local message queue at slot $\mathcal{C}.\mathsf{sl} \leq t$. Each entry in $\mathbf{m}^{(i)}$ consists of a chain $\mathcal{C}$ and honest party subset $\mathcal{P} \subset \mathcal{H}$, which has yet to receive the message. $\mathcal{A}$ is required to deliver all honestly broadcast chains with a delay of no more than δ slots. The model executes round-wise beginning from initial state Γ_0, where the tree views of all parties consist only of the genesis block.

In each round from slot t to $t+1$, the leader is implied by by $w_{t+1} \in \{0, 1, \perp\}$. For a uniquely honest slot, the environment $\mathcal{Z}$ is permitted to activate any honest party to extend the longest chain in its local view, where the inputs for insertion in the block are provided by $\mathcal{Z}$. We interpret $w_{\mathsf{slot}} = 1$ as a strictly adversarial slot, since the adversary could affect the structure of local trees views in the same way as multiple honest leaders: namely, by producing multiple blocks associated with the same slot.

PoS Security. The seminal work on formalizing the Bitcoin backbone protocol [16] proved *liveness* and *persistence* of longest-chain proof-of-work (PoW) protocols in terms of common-prefix, chain growth and chain quality properties, which are also achieved for PoS in Ouroboros Praos [13]. We restate these below and formally prove them for FairPoS in Section 4.

Definition 3 (Common prefix, k-CP; with parameter $k \in \mathbb{N}$). *The chains $\mathcal{C}_1$, $\mathcal{C}_2$ possessed by two honest parties at the onset of the slots $t_1 < t_2$ are such that $\mathcal{C}_1^{\lceil k} \preceq \mathcal{C}_2$, where $\mathcal{C}_1^{\lceil k}$ denotes the chain $\mathcal{C}_1^{\lceil k}$ obtained by removing the last k blocks from $\mathcal{C}_1$, and $\preceq$ denotes the prefix relation.*

Definition 4 (Chain growth, (τ, s)-CG; with parameter $\tau \in (0, 1]$ and $s \in \mathbb{N}$). *Consider the chains $\mathcal{C}_1$, $\mathcal{C}_2$ possessed by two honest parties at the onset of two slots t_1, t_2 with t_2 at least s slots ahead of t_1. Then it holds that $\mathsf{len}(\mathcal{C}_2) - \mathsf{len}(\mathcal{C}_1) \geq \tau \cdot s$. We call τ the speed coefficient.*

Definition 5 (Chain quality, (μ, k)-CQ; with parameters $\mu \in (0, 1]$ and $k \in \mathbb{N}$). *Consider any portion of length at least k of the chain possessed by an honest party at the onset of a round; the ratio of blocks originating from the adversary is at most $1 - \mu$. We call μ the chain quality coefficient.*

3 The FairPoS protocol

We introduce the FairPoS model in parts. In Section 3.1, we formally define a novel notion of *input fairness*, for clients sending transactions to a FairPoS

execution. In order for an encrypted input to reach the k-common-prefix, the duration implied by the delay parameter d must be sufficiently long.

In Section 3.2, we introduce the formal model of FairPoS, which extends δ-PoS with key extraction and a novel *longest-extractable-chain* selection rule. Here, encrypted inputs are generated as in Section 3.1 and given by the environment $\mathcal{Z}$ to parties executing the protocol. We first describe a naive application of key extraction, demonstrating adversarial chain stalls up to $d-1$ slots, thereby motivating the design of the *longest-extendable-chain* selection rule in the FairPoS model, which mitigates such adversarial impedance and ensures honest chain growth *independent* of chosen delay encryption parameter d.

A security analysis of FairPoS follows in Section 4, where the precise relationship between input fairness, chain growth, common-prefix and chain quality properties is formalized.

3.1 Input fairness & input encryption

Definition 6 (Input fairness, IF). *Consider the chain $\mathcal{C}$ possessed by an honest party at the onset of a round, where k-CP holds true. Input fairness holds if for all blocks $B \in \mathcal{C}^{\lceil k}$: 1. the adversary cannot decrypt an encrypted input in B before B is in the common-prefix; 2. encrypted inputs in B are eventually decrypted by all honest parties.*

Input fairness is conditioned on k-common-prefix property in FairPoS. Intuitively, the extraction delay d in FairPoS must be parameterized, such that the encrypted input can reach the common-prefix before d time passes. For simplicity, we denote d as time in slots. Note that input fairness permits an encrypted input to *not* become finalized and decrypted by the adversary: we argue this outcome is acceptable as the client transaction is not executed and thus cannot be exploited in any input ordering attacks. This is consistent with [5,26,27].

We sketch the input encryption procedure for FairPoS shown in Figure 2, where the environment $\mathcal{Z}$ provides the plain-text input for a party to encrypt and sign. For a block B, inputs are encrypted to a session id which is set to the chain tip that B is extending, such that $\mathsf{id} = \mathcal{C}.\mathsf{tip}$ and $\mathcal{C}.\mathsf{tip} = B.\mathsf{st}$. To ensure that a slot leader cannot insert an encrypted input to a *later* block, potentially deferring its insertion until the key extraction is completed, we ensure that the input is *bound* to a child block of $\mathcal{C}.\mathsf{tip}$ with a signature.

In the adaptive corruption setting, we deploy an efficient key evolving signature scheme (KES) [6,20] as used in [13]. Such schemes evolve secret key material *forward* with each signature, thereby erasing any information that could be used to generate verifying signatures of past rounds (See Definition 10). An adaptive adversary could always corrupt a user who has just *broadcast* a newly delay encrypted and signed input; with static key material only, the adversary would learn the signature key and generate verifying signatures of the delay encrypted input to insert it into a block, potentially *after* decrypting the encrypted input.

For public verification of such signatures, we assume the presence of logical accounts for all parties, each associated with a public signature verification key inferred from the chain tip.

FairPoS input encryption procedure

Let $\mathsf{KES} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Update})$ and $\mathsf{SKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ denote a key evolving signature scheme and a symmetric-key encryption scheme. Let the genesis block of a chain contain a delay encryption parameter $\mathsf{DE.pk}$ and a chain tip imply account keys $\{\mathsf{KES.vk}_i\}_{i\in[n]}$.

Gen: Upon (GEN), set $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^k, T)$, return vk.

Sign: Upon $(\text{SIGN}, \mathcal{C}, \mathsf{in})$

1. Let $\mathsf{id} = \mathcal{C}.\mathsf{tip}$. Assert $\mathsf{vk} \in \mathsf{accts}(\mathcal{C}.\mathsf{tip})$. Set $\mathsf{pk} \leftarrow \mathsf{DE.pk}(\mathcal{C}_0)$
2. Compute $(c, k) \leftarrow \mathsf{DE.Encaps}(\mathsf{pk}, \mathsf{id})$.
3. Encrypt input with key k: $m \leftarrow \mathsf{SKE.Enc}_k(\mathsf{in})$.
4. Generate $\sigma \leftarrow \mathsf{KES.Sign}_{\mathsf{sk}}(c \,|\, m \,|\, \mathsf{id})$.
5. Set $\mathsf{sk} \leftarrow \mathsf{Update}(\mathsf{KES.sk})$, thereby erasing signing previous key.
6. Return (c, m, σ).

Fig. 2. Input encryption in FairPoS

3.2 The (d, δ, Δ)-FairPoS protocol

A key contribution of FairPoS is to achieve security that is independent of the chosen delay encryption parameter d. Towards this, we reintroduce Δ-monotonicity from [13], which states that an honest chain tip can always be extended by the first honest leader following Δ slots, a key property required to realize security in longest-chain proof-of-stake.

Definition 7 (Δ-Monotonicity, from [13]). *Let $\mathcal{T} = \bigcup_{i\in\mathcal{H}} \mathcal{T}^{(i)}$ be an honest tree rooted in genesis resulting from an execution of protocol π in a δ-synchronous network: it consists of all chains broadcast by honest parties. Further, let $\mathsf{depth}(i)$ denote the length of the chain extended by the uniquely honest leader of slot i. $\mathcal{T}$ exhibits the Δ-monotonicity property if*

$$\textit{For all uniquely honest slots } (i, j) \textit{ s.t. } j \geq i + \Delta : \mathsf{depth}(j) > \mathsf{depth}(i)$$

In PoS executed in a δ-synchronous setting , δ-monotonicity is trivially achieved: any honest block tip must arrive in the view of other honest party after δ slots, and is thus considered as a chain candidate for extension by any honest leader applying the longest chain selection rule.

Towards achieving Δ-monotonicity when extending PoS with key extraction with delay d, such that d can independently be parameterized from Δ, we first illustrate a naive key extraction application where this is not achieved to motivate the final FairPoS design.

Here, we must introduce a formal notion of *receipt delay*, which, informally, quantifies how far a local key extraction process is "behind schedule" due to adversarial delays.

Definition 8 (Receipt delay). *Let $\boldsymbol{r}^{(i)}(B) : \mathcal{B} \to \mathbb{Z}_0$ be the delay in slots between $B.\mathsf{sl}$ and the local arrival of block B from the view of the party (i). If B is an empty-block ($\perp$), we define the receipt delay to be $\perp$.*

When the extraction window is defined as a slot interval preceding genesis, where slot indices are "negative", let the receipt delay is defined as $\perp$. We allow a receipt delay of $\perp$ to be interpreted as a receipt delay of 0.

Naive key extraction causing honest chain stall. Let an extraction schedule be defined as $D = d + \delta$, where d is the delay encryption parameter in slots, and δ be the maximum network delay δ. We initially assume a leader election sequence with no empty slots.

For an honest leader of slot t extending chain $\mathcal{C}$ (according to the longest-chain rule), let $\mathcal{C}_{t-D}$ denotes the block in $\mathcal{C}$ with a key extraction due in the honest block of slot t extending $\mathcal{C}$. In order to ensure that *current and future* slot leaders can extend a chain $\mathcal{C}$, the following must hold.

1. The honest slot leader must receive $\mathcal{C}_{t-D}$ *on time* (latest at slot $t - D + \delta$).
2. The honest slot leader must ensure that blocks in $\mathcal{C}$, which are extracted by *future* honest leaders, must also arrive *on time*, e.g. $(\mathcal{C}_{t-D+1}, ..., \mathcal{C}_{t-1})$.

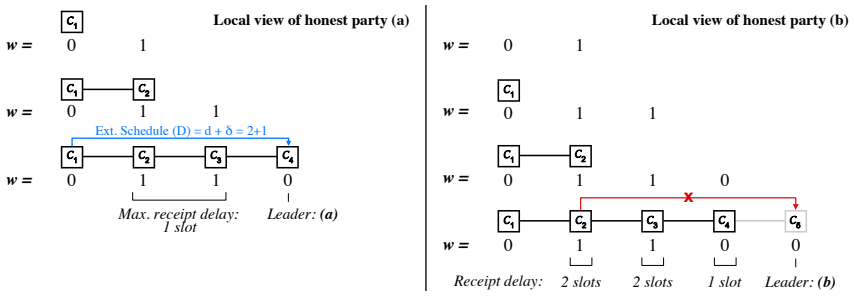

Fig. 3. Example: Naive application of key extraction

Let the maximum network delay of a single slot be $\delta = 1$ in Figure 3. Consider honest party (a)'s view shown in the figure below, who is the leader of slot 4. Party (a) asserts that:

1. It can extract $\mathcal{C}_1$ by slot $\mathcal{C}_4$ given extraction schedule $D = 2 + 1 = 3$.
2. Blocks $\mathcal{C}_2$ and $\mathcal{C}_3$ were received in party (a)'s view with a *receipt delay* of no more than $\delta = 1$ slot.

Recall, condition (2) is intended to ensure that future honest leaders will be able to perform extractions of $\mathcal{C}_2, \mathcal{C}_3$ on time. However, the adversarial blocks $\mathcal{C}_{2,3}$ are only forwarded to party (a) with the maximally permitted delay $\delta = 1$, who then rebroadcasts (see M1 in Figure 1) to other honest parties:

$$\mathcal{A} \xrightarrow{\mathcal{C}_{2,3}} \mathcal{P}_{(a)} \xrightarrow{\mathcal{C}_{2,3}} \{\mathcal{P}_i\}_{i \in \mathcal{H}} \tag{3}$$

This inflicts an *additional* receipt delay on $\mathcal{C}_{2,3}$ in the view of other honest parties, such as party (b), the leader of slot 5. In the view of party (b),

1. It *cannot* extract $\mathcal{C}_2$ by slot 5 as the extraction of $\mathcal{C}_2$ is only 1/3 complete.
2. Blocks $\mathcal{C}_3$ and $\mathcal{C}_4$ were received in party (b)'s view with a *receipt delay* that *exceeds* 1 slot.

Thus, we have a honest chain stall: any honest party other than party (a) cannot extend honest chain tip $\mathcal{C}_4$ until slot 6, inflicting a chain stall of $d-1$.

Key extraction and longest-extendable-chains. We provide an informal overview to key extraction in FairPoS. Consider the view of honest party (a) shown in Figure 4, who is the leader of slot t. Let the protocol be executed in a 1-synchronous setting. Honest party (a), considering the extension of the chain $\mathcal{C}$, asserts the following:

1. It has completed the extraction of blocks in the *extraction window* with an *extraction schedule* $D = 12$ slots.
2. All blocks in the chain with *pending extractions* must have been received in party (a)'s view with maximum receipt delays shown in Figure 4.

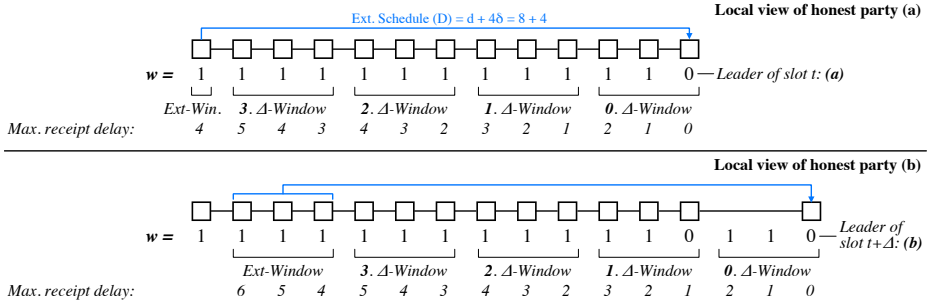

Fig. 4. Key extraction in (d, δ, Δ)-FairPoS

Maximally permitted receipt delays are specific to each Δ-window, which divides the D slots of the extraction schedule. For the n'th Δ-window, the maximum permitted receipt delay is $n \cdot \delta = n \cdot 1$ for the first slot to the right of the window. Each subsequent slot in the n'th window is permitted an additional receipt delay of 1 slot. Observe that Δ-monotonicity holds for Δ=3: consider party (b), which is the leader of slot $t+\Delta = t+3$. Even if the adversary can induce an additional receipt delay of $\delta = 1$ in the view of party (b) as shown in Figure 4, party (b) will be able to assert conditions (1) and (2) above. This is because each block in the n'th Δ-window of party (a) is now in the $n+1$'th Δ-window of party (b)'s view and is permitted an additional delay of $\delta = 1$.

Extraction window. We define an extraction window as the the slots for which associated blocks have key extractions that are due in the current slot. This accounts for the possibility of a *gap* between the chain tip and the current slot, where no extracted keys could have been inserted.

Let the FairPoS protocol be parameterized with extraction schedule D. We define $\mathsf{gap2tip}(t, \mathcal{C})$ as the number of *empty* slots between slot t and $\mathcal{C}.\mathsf{tip.sl}$.

$$\mathsf{gap2tip}(t, \mathcal{C}) = t - \mathsf{tip}(\mathcal{C}).\mathsf{sl} \quad \text{for} \quad t > \mathsf{tip}(\mathcal{C}).\mathsf{sl}$$

Then, the extraction window of current slot t and chain $\mathcal{C}$ with extraction schedule D can be defined as the range of slots, which are at least D slots in the past and are associated with blocks in $\mathcal{C}$ *without* key extractions already inserted in $\mathcal{C}$ due to the absence of blocks.

$$\mathsf{extWin}_D(t, \mathcal{C}) = (t - D - \mathsf{gap2tip}(t, \mathcal{C}) : t - D\,] \tag{4}$$

Extendable chains. Let (d, δ, Δ)-FairPoS be parameterized by delay encryption parameter $d > 0$, maximum network delay $\delta \geq 0$ and desired monotonicity parameter $\Delta > \delta$ (Definition 7), such that d divides $\Delta - \delta$. If this holds, then we can define n, the number of Δ-windows which divide extraction schedule D.

$$D = n\Delta \quad \text{where} \quad n = d/(\Delta - \delta) \quad \text{s.t.} \quad n \in \mathbb{Z} \tag{5}$$

The delay extraction schedule D can be expanded to $D = n\Delta = d + n\delta$, as the right equality is implied by $n = d/(\Delta - \delta)$ from Equation 5.

We define the m'th "Δ-window" of current slot t as the following interval, consistent with Figure 4.

$$\Delta\mathsf{Win}_D(t, m) = (t - (m+1)\Delta : t - m\Delta] \quad \text{for} \quad m \in [0 : \frac{D}{\Delta}) \tag{6}$$

We formalize *chain extendability*. A chain $\mathcal{C}$ is extendable by leader of slot t with local receipt delay view $\mathbf{r}^{(i)}$ if the conditions ① and ② in Equation 7 hold.

$$\begin{aligned} \mathsf{ext}(t, \mathcal{C}, \mathbf{r}^{(i)}) &= \begin{cases} 1 & ① \wedge ② \\ 0 & \text{otherwise} \end{cases} \\ ① \; \forall m \in [0 : \tfrac{D}{\Delta}) : \forall j \in \Delta\mathsf{Win}_D(t, m) &: \mathbf{r}^{(i)}(\mathcal{C}_j) \leq m\delta + (t - m\Delta - j) \\ ② \qquad \forall j \in \mathsf{extWin}_D(t, \mathcal{C}) &: \mathbf{r}^{(i)}(\mathcal{C}_j) \leq \tfrac{D}{\Delta}\delta + (t - D - j) \end{aligned} \tag{7}$$

Observe that extendability holds for the respective chain in the view of parties (a) and (b) in Figure 4 .

Theorem 1. *(Δ-Monotonicity of FairPoS) Every protocol execution of (d, δ, Δ)-FairPoS results in an honest tree $\mathcal{T}$ that exhibits the Δ-monotonicity property.*

Longest-extendable-chain selection. In FairPoS, an honest slot leader choses to extend the *longest extendable chain* in its local tree view $\mathcal{T}^{(i)}$.

$$\mathsf{maxExtChain}(t, \mathcal{T}^{(i)}, \mathbf{r}^{(i)}) = \underset{\mathcal{C} \in \mathcal{T}^{(i)} : \mathsf{ext}(t, \mathcal{C}, \mathbf{r}^{(i)})}{\arg\max} \mathsf{len}(\mathcal{C}) \tag{8}$$

The formal model of (d, δ, Δ)-FairPoS is stated in Figure 5, which extends the δ-PoS in Figure 1 with the longest-extendable-chain selection rule, key extractions and the insertion of delay encrypted inputs, generated by the procedure in Figure 2. The protocol execution state of (d, δ, Δ)-FairPoS is extended with local receipt delays:

$$\Gamma_t = \left(\{\mathcal{T}^{(i)}, \mathbf{m}^{(i)}, \mathbf{r}^{(i)}\}_{i \in \mathcal{H}}, \mathcal{T}^{\mathcal{A}}\right) \tag{9}$$

(d, δ, Δ)-FairPoS execution model

The evolution of a (d, δ, Δ)-FairPoS execution state $\Gamma_t = \left(\{\mathcal{T}^{(i)}, \mathbf{m}^{(i)}, \mathbf{r}^{(i)}\}_{i \in \mathcal{H}}, \mathcal{T}^{\mathcal{A}}\right)$ in a single execution round induced by environment $\mathcal{Z}$, adversary $\mathcal{A}$ and characteristic string w adheres to **T1-T3** and **M1-M2** of PoS execution model (Figure 1) with the following differences:

T2' differs from **T2** of PoS in the following:
- *Longest-extendable-chain selection* replaces *longest-chain-selection*.
- **Extend'** procedure replaces **Extend** for honest parties and $\mathcal{A}$.

The receipt delays $\{\mathbf{r}^{(i)}\} i \in \mathcal{H}$ in Γ_t evolve as follows:

R1: For each chain $\mathcal{C}$ delivered to honest party i by $\mathcal{A}$ at slot $t+1$: for each newly seen block $\mathcal{C}_j \in \mathcal{C}$, party i records $\mathbf{r}^{(i)}(\mathcal{C}_j) \leftarrow t + 1 - \mathcal{C}_j.\mathsf{sl}$.

Each honest party and the adversary must perform:

E1: In each round, exactly a single Extract step on each block in its local view for which key extraction is pending.

Extend': Party i generates $B = (t, H(\mathcal{C}.\mathsf{tip}), d_{\mathsf{ext}} | d_{\mathsf{ins}}, P_i)$, where:
- $d_{\mathsf{ext}} = (\mathsf{idk}, \mathsf{idk}', ...)$ are extractions of blocks in $\mathcal{C}$ associated with $\mathsf{extWin}_D(t, \mathcal{C})$.
- $d_{\mathsf{ins}} = \{(c_j, m_j, \sigma_j)\}_{j \in [n]}$ is an *ordering* of encrypted inputs from $\mathcal{Z}$ for slot t. When party i receives d_{ins}, it asserts for $\mathsf{id} = \mathcal{C}.\mathsf{tip}$ and each entry in d_{ins}:
  - $\exists \mathsf{vk}_j \in \mathsf{accts}(\mathcal{C}.\mathsf{tip}) : 1 \leftarrow \mathsf{KES.Verify}_{\mathsf{vk}_j}((c_j | m_j | \mathsf{id}), \sigma_j)$.

Then, the party adds $\mathcal{C}' = \mathcal{C} | B$ to its local view.

Fig. 5. FairPoS execution

Views in initial state Γ_0 contain a genesis block which includes public parameters $\mathsf{DE.pk}, \mathsf{DE.ek}$. Importantly, we *require* the adversary $\mathcal{A}$ and each honest party to perform exactly one extraction step for each pending key extraction in each round of the execution. Thus, no party or adversary gains a time advantage in extracting session keys from blocks (See E1 in Figure 5).

4 FairPoS security

4.1 Common-prefix in FairPoS

Demonstrating k-common-prefix in FairPoS is accomplished by formally relating the tree views generated in an execution of (d, δ, Δ)-FairPoS with those that could have resulted from δ-PoS.

Let the *honest tree* of protocol execution state $\Gamma_t = \left(\{\mathcal{T}^{(i)}, \mathbf{m}^{(i)}, \mathbf{r}^{(i)}\}_{i \in \mathcal{H}}, \mathcal{T}^{\mathcal{A}}\right)$ be given be the union of honest tree views at slot t:

$$\mathcal{T}^{\mathcal{H}}(\Gamma_t) = \bigcup_{i \in \mathcal{H}} \mathcal{T}^{(i)} \tag{10}$$

In the analysis of PoS [13,24], the branching structure of the honest tree informs us about events where a local chain was abandoned for a longer, alternative

branch according to the *longest chain* selection rule. Informally, the common-prefix property is violated if the k-common-prefix shared between prior and newly adopted chains if their shared prefix is "too short". We begin our analysis with the (viable) branches which could have been adopted by honest parties in the first place.

Viable branches in PoS. A viable branch $\mathcal{B}$ in a tree $\mathcal{T}$ must exceed all honest chains-tips generated more than δ slots prior to $\mathcal{B}.\mathsf{tip.slot}$ in length: this property arises from the honest application of the *longest chain rule*. The honest leader of $\mathcal{B}.\mathsf{tip.slot}$ must have received any honest chain-tips generated δ slots prior and considered them as *alternative candidates* for extension. Therefore, the resulting, *viable* branch $\mathcal{B}$ must exceed these in length. For branch $\mathcal{B} \in \mathcal{T}$, we formalize this set of alternative chain candidates as

$$\mathsf{altChns}_\delta(\mathcal{B}, \mathcal{T}) = \{\, \mathcal{C} \subseteq \mathcal{T} \mid \mathcal{C}.\mathsf{tip.ldr} \in \mathcal{H} \;\wedge\; \mathcal{B}.\mathsf{tip.sl} > \mathcal{C}.\mathsf{tip.sl} + \delta \,\} \tag{11}$$

For a PoS protocol execution state $\Gamma_t = \left(\{\mathcal{T}^{(i)}, \mathbf{m}^{(i)}\}_{i\in\mathcal{H}}, \mathcal{T}^{\mathcal{A}}\right)$, the set of well-formed, viable branches is formalized as the set of branches with lengths exceeding their honest, alternative chains.

$$\mathsf{viableBranches}_\delta^{\mathsf{PoS}}(\Gamma_t) = \{\forall \mathcal{B} \in \mathcal{T}^{\mathcal{H}}(\Gamma_t) \mid \mathsf{len}(\mathcal{B}) > \underset{\mathcal{C} \,\in\, \mathsf{altChns}_\delta(\mathcal{B}, \mathcal{T}^{\mathcal{H}}(\Gamma_t))}{\arg\max} \mathsf{len}(\mathcal{C}) \;\} \tag{12}$$

Viable chains in FairPoS. The notion of viable branches must be strengthened for FairPoS since the *longest-extendable-chain* rule introduces additional constraints for the adoption of a chain in the local honest tree view. Let the *extendable prefix* of a branch $\mathcal{B}$ in the view of honest parties at slot t be defined as the "longest extendable prefix" of a branch.

$$\mathsf{extPrefix}(t, \mathcal{B}, \{\mathbf{r}^{(i)}\}_{i\in\mathcal{H}}) = \underset{\mathcal{C} \preceq \mathcal{B} \,:\, \exists i \in \mathcal{H} \,:\, \mathsf{ext}(t, \mathcal{C}, \mathbf{r}^{(i)})}{\arg\max} \mathsf{len}(\mathcal{C}) \tag{13}$$

For a (d, δ, Δ)-FairPoS state, let the set of viable chains be defined as the extendable prefixes (Equation 13) of branches in the honest tree with lengths which exceed those of its alternative chains (Equation 11) generated Δ slots prior: by the Δ-monotonicity property (Theorem 1), these chains must have been extendable by the leader that generated the respective prefix and considered as candidates for extension.

$$\begin{aligned}\mathsf{viableChains}_\Delta^{\mathsf{FairPoS}}(\Gamma_t) = \{\, \mathcal{C} \subseteq \mathcal{T}^{\mathcal{H}}(\Gamma_t) \mid \exists \mathcal{B} \in \mathcal{T}^{\mathcal{H}}(\Gamma_t) : \mathcal{C} = \mathsf{extPrefix}(t, \mathcal{B}, \{\mathbf{r}^{(i)}\}_{i\in\mathcal{H}}) \\ \wedge\; \mathsf{len}(\mathcal{C}) > \underset{\mathcal{C}' \,\in\, \mathsf{altChns}_\Delta(\mathcal{C}, \mathcal{T}^{\mathcal{H}}(\Gamma_t))}{\arg\max} \mathsf{len}(\mathcal{C}')\end{aligned} \tag{14}$$

We restate the *divergence* notion from [13,24] which formally describes the *magnitude* of branching caused by the switching between viable chains.

Definition 9 (Divergence). *For two chains $\mathcal{C}_1$ and $\mathcal{C}_2$, define their divergence to be the quantity*

$$\mathsf{div}(\mathcal{C}_1, \mathcal{C}_2) = \mathsf{min}(\mathsf{len}(\mathcal{C}_1), \mathsf{len}(\mathcal{C}_2)) - \mathsf{len}(\mathcal{C}_1 \cap \mathcal{C}_2)$$

where $\mathcal{C}_1 \cap \mathcal{C}_2$ denotes the common prefix of $\mathcal{C}_1$ and $\mathcal{C}_2$. We extend this notion of divergence to the protocol execution state Γ resulting from the execution of protocol π induced by characteristic w in the δ-synchronous setting: here, the maximum divergence over any two viable chains is quantified.

$$\mathsf{div}_\delta^\pi(\Gamma) = \mathsf{max}_{\mathcal{C}_1,\mathcal{C}_2 \in \mathsf{viableBranches}_\delta^\pi(\Gamma)} \mathsf{div}(\mathcal{C}_1, \mathcal{C}_2)$$

Finally, we define the divergence of a characteristic string w to be the maximum divergence observable over all states which could have resulting from protocol executions induced by w. More formally, let $\mathsf{exec}_\delta^\pi(\Gamma_0, w)$ denote all possible executions of π beginning with state Γ_0 which could have been induced by w in δ-synchronous network. Then the divergence of a characteristic string w is defined as:

$$\begin{gathered} \mathsf{div}_\delta^\pi(w) = \mathsf{max}_{\Gamma \in \mathsf{reachable}_\delta^\pi(\Gamma_0, w)} \mathsf{div}_\delta^\pi(\Gamma) \\ \textit{where } \mathsf{reachable}_\delta^\pi(\Gamma_0, w) = \{\Gamma \mid \exists \lambda \in \mathsf{exec}_\delta^\pi(\Gamma_0, w) : \Gamma_0 \xrightarrow{\lambda} \Gamma\} \end{gathered} \tag{15}$$

For PoS, the probability that that the divergence exceeds k-blocks over an execution of R, is given by the following theorem from [13].

Theorem 2. *(PoS Divergence, Theorem 4 in [13]) Let active slot coefficient $f \in (0, 1]$ and α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for some $\epsilon > 0$. Further, let w be a string drawn from $\{0, 1, \perp\}^R$ according to D_α^f. Then we have $\Pr_{w \leftarrow\$ D_\alpha^f}[\mathit{div}_\Delta^{\mathit{PoS}}(w) \geq k] \leq \exp(\ln(R) - \Omega(k - \Delta))$.*

Towards demonstrating k-common prefix in FairPoS, we first present a central theorem, which states that for all executions of (d, δ, Δ)-FairPoS in the δ-synchronous setting, there exists an execution of PoS in the Δ-synchronous setting, such that viable chains of the d, δ, Δ-FairPoS honest tree are *equivalent* ($\equiv$) to the viable branches of the PoS honest tree: here, we define the equivalence of chains such that only their structural properties are considered, formally stated in Definition 11.

Theorem 3. (Equivalent trees) For any (d, δ, Δ)-FairPoS execution λ induced by a charactistic string $w \in \{0, 1, \perp\}^*$, $\Gamma_0 \rightarrow^\lambda \Gamma$, there exists a Δ-PoS execution λ' induced by same w, $\Gamma_0' \rightarrow^{\lambda'} \Gamma'$, such that the viable chains in Γ are equivalent to the viable branches in Γ'.

$$\begin{gathered} \forall w \in \{0,1,\perp\}^* : \forall \lambda \in \mathsf{exec}_\delta^{\mathsf{FairPoS}}(\Gamma_0, w), \Gamma_0 \xrightarrow{\lambda} \Gamma : \exists \lambda' \in \mathsf{exec}_\Delta^{\mathsf{PoS}}(\Gamma_0', w), \Gamma_0 \xrightarrow{\lambda'} \Gamma' : \\ \mathsf{viableChains}_\delta^{\mathsf{FairPoS}}(\Gamma) \equiv \mathsf{viableBranches}_\Delta^{\mathsf{PoS}}(\Gamma') \end{gathered}$$

Since divergence is defined over viable chains and viable branches, we can infer Corollary 1 from Theorem 3.

Corollary 1. $\forall w \in \{0, 1, \perp\}^* : \mathit{div}_\delta^{\mathit{FairPoS}}(w) \leq \mathit{div}_\Delta^{\mathit{PoS}}(w)$

This allows us to infer k-common-prefix from bounding the probability of the event $\mathsf{div}_\Delta^{\mathsf{PoS}}(w) > k$ for $w \leftarrow\$ \mathcal{D}_\alpha^f$ in PoS [13].

Theorem 4. (k-Common prefix in FairPoS) *Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0,1]$ and some $\epsilon > 0$. The probability that $\mathcal{A}$ makes the protocol violate the k-common-prefix property in a δ-synchronous environment throughout a period of R slots is no more than* $\exp(\ln R + \Delta - \Omega(k))$.

4.2 Chain growth, chain quality and input fairness

Both chain growth and chain quality of FairPoS can be derived from Δ-monotonicity of FairPoS (Theorem 1) and probabilities bounding security failure from PoS [13].

Theorem 5. ((τ, s)-Chain growth in (d, δ, Δ)-FairPoS) Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0,1]$ and some $\epsilon > 0$. Then the probability that $\mathcal{A}$ makes the protocol violate the chain growth property with parameters $s \geq 4\Delta$ and $\tau = c\alpha/4$ throughout a period of R slots, is no more than $\exp(-c\alpha s/(20\Delta) + lnR\Delta + O(1))$, where c denotes the constant $c := c(f, \Delta) = f(1-f)^{\Delta}$.

Theorem 6. ((μ, k)-Chain quality in (d, δ, Δ)-FairPoS) Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0,1]$ and some $\epsilon > 0$. Then the probability that $\mathcal{A}$ makes FairPoS violate the chain quality property with parameters k and $\mu = 1/k$ throughout a period of R slots, is no more than $\exp(\ln R - \Omega(k))$.

Input fairness is obtained from chain growth, common prefix and chain quality. Informally, given time d and chain growth rate τ, we can determine common-prefix and chain quality parameters such that a decrypted input must have sufficient time (d slots) to reach finalization or lie in an abandoned chain.

Lemma 1. (Input fairness from CG, CP and CQ in (d, δ, Δ)-FairPoS) If for an execution of (d, δ, Δ)-FairPoS, (τ, d)-chain growth, $(d\tau(\tau - \delta/(\Delta - \delta)) - 1)$-common prefix, and $(1/(D+1), D+1)$-chain quality hold, where $D = d\Delta/(\Delta - \delta)$, then input fairness is implied.

Theorem 7. (Input fairness in (d, δ, Δ)-FairPoS) Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0,1]$ and some $\epsilon > 0$. Then, the probability that $\mathcal{A}$ makes the FairPoS violate the input fairness property falls exponentially with d.

We refer to Appendix C for a discussion of (d, δ, Δ)-FairPoS security for specific parameterizations of d, δ, Δ.

5 Conclusion

We contribute FairPoS, the first longest-chain, proof-of-stake protocol achieving input fairness against an adaptive adversary. When adopting the leader election procedure from Ouroborous Praos [13] or one that induces leader sequences (*i.e.* characteristic strings) consistent with the *dominant distribution* (Definition 1), FairPoS achieves input fairness in addition to the common-prefix, chain-growth and chain-quality properties of PoS [13]. We note that Kiayas *et al.* [24] provide tighter bounds for k-common prefix in PoS. Applying this updated analysis framework to security of FairPoS is planned as future work.

6 Acknowledgements

We thank Stefan Dziembowski and Sebastian Faust for exploratory discussions on mitigating front-running at the consensus protocol level. We thank Luca De Feo for his insights on isogeny-based cryptography[6] and his views on deploying Delay Encryption in practice.

[6] At the crypt@b-it 2022 summer school on cryptography.

References

1. Alliance, V.: VDF Alliance Official Wiki. https://supranational.atlassian.net/wiki/spaces/VA/overview (2022)
2. Avalanche: Apricot Phase Four: Snowman++ and Reduced C-Chain Transaction Fees. https://medium.com/avalancheavax/apricot-phase-four-snowman-and-reduced-c-chain-transaction-fees-1e1f67b42ecf (2021)
3. Badertscher, C., Gaži, P., Kiayias, A., Russell, A., Zikas, V.: Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 913–930 (2018), https://doi.org/10.1007/978-3-319-78375-8_3
4. Bartoletti, M., Chiang, J.H.y., Lluch-Lafuente, A.: Maximizing extractable value from automated market makers. arXiv preprint arXiv:2106.01870 (2021), https://arxiv.org/pdf/2106.01870
5. Bebel, J., Ojha, D.: Ferveo: Threshold Decryption for Mempool Privacy in BFT networks. Cryptology ePrint Archive (2022), https://eprint.iacr.org/2022/898
6. Bellare, M., Miner, S.K.: A forward-secure digital signature scheme. In: CRYPTO'99 (Aug 1999)
7. Benhamouda, F., Gentry, C., Gorbunov, S., Halevi, S., Krawczyk, H., Lin, C., Rabin, T., Reyzin, L.: Can a public blockchain keep a secret? In: TCC 2020, Part I (Nov 2020)
8. Blum, E., Kiayias, A., Moore, C., Quader, S., Russell, A.: Linear consistency for proof-of-stake blockchains. arXiv preprint arXiv:1911.10187 (2019), https://arxiv.org/abs/1911.10187
9. Burdges, J., Feo, L.D.: Delay encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 302–326. Springer (2021), https://doi.org/10.1007/978-3-030-77870-5_11
10. Cachin, C., Mićić, J., Steinhauer, N.: Quick Order Fairness. arXiv preprint arXiv:2112.06615 (2021), https://arxiv.org/abs/2112.06615
11. Cascudo, I., David, B., Garms, L., Konring, A.: YOLO YOSO: Fast and simple encryption and secret sharing in the YOSO model. Cryptology ePrint Archive, Report 2022/242 (2022), https://eprint.iacr.org/2022/242
12. Daian, P., Goldfeder, S., Kell, T., Li, Y., Zhao, X., Bentov, I., Breidenbach, L., Juels, A.: Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In: IEEE Symposium on Security and Privacy. pp. 910–927. IEEE (2020). https://doi.org/10.1109/SP40000.2020.00040, https://doi.org/10.1109/SP40000.2020.00040
13. David, B., Gaži, P., Kiayias, A., Russell, A.: Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 66–98. Springer (2018), https://doi.org/10.1007/978-3-319-78375-8_3
14. De Feo, L., Masson, S., Petit, C., Sanso, A.: Verifiable delay functions from supersingular isogenies and pairings. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 248–277. Springer (2019), https://doi.org/10.1007/978-3-030-34578-5_10
15. Erwig, A., Faust, S., Riahi, S.: Large-scale non-interactive threshold cryptosystems through anonymity. Cryptology ePrint Archive, Report 2021/1290 (2021), https://eprint.iacr.org/2021/1290

16. Garay, J., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: Analysis and applications. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 281–310. Springer (2015), https://doi.org/10.1007/978-3-662-46803-6_10
17. Gentry, C., Halevi, S., Krawczyk, H., Magri, B., Nielsen, J.B., Rabin, T., Yakoubov, S.: YOSO: You only speak once - secure MPC with stateless ephemeral roles. In: CRYPTO 2021, Part II (Aug 2021)
18. Gilad, Y., Hemo, R., Micali, S., Vlachos, G., Zeldovich, N.: Algorand: Scaling byzantine agreements for cryptocurrencies. In: Proceedings of the 26th symposium on operating systems principles. pp. 51–68 (2017), https://doi.org/10.1145/3132747.3132757
19. Goyal, V., Kothapalli, A., Masserova, E., Parno, B., Song, Y.: Storing and retrieving secrets on a blockchain. In: Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part I (2022). https://doi.org/10.1007/978-3-030-97121-2_10
20. Itkis, G., Reyzin, L.: Forward-secure signatures with optimal signing and verifying. In: CRYPTO 2001 (Aug 2001)
21. Kelkar, M., Deb, S., Kannan, S.: Order-fair consensus in the permissionless setting. In: Proceedings of the 9th ACM on ASIA Public-Key Cryptography Workshop. pp. 3–14 (2022), https://doi.org/10.1145/3494105.3526239
22. Kelkar, M., Deb, S., Long, S., Juels, A., Kannan, S.: Themis: Fast, Strong Order-Fairness in Byzantine Consensus (2021), https://eprint.iacr.org/2021/1465
23. Kelkar, M., Zhang, F., Goldfeder, S., Juels, A.: Order-fairness for byzantine consensus. In: Annual International Cryptology Conference. pp. 451–480. Springer (2020), https://doi.org/10.1007/978-3-030-56877-1_16
24. Kiayias, A., Quader, S., Russell, A.: Consistency of proof-of-stake blockchains with concurrent honest slot leaders. In: 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS). pp. 776–786. IEEE (2020), https://doi.org/10.1109/ICDCS47774.2020.00065
25. Kiayias, A., Russell, A., David, B., Oliynykov, R.: Ouroboros: A provably secure proof-of-stake blockchain protocol. In: Annual international cryptology conference. pp. 357–388. Springer (2017), https://doi.org/10.1007/978-3-319-63688-7_12
26. Malkhi, D., Szalachowski, P.: Maximal Extractable Value (MEV) Protection on a DAG. arXiv e-prints pp. arXiv–2208 (2022), https://arxiv.org/abs/2208.00940
27. Momeni, P.: Fairblock: Preventing blockchain front-running with minimal overheads. Master's thesis, University of Waterloo (2022), https://eprint.iacr.org/2022/1066
28. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto (1996), http://bitsavers.trailing-edge.com/pdf/mit/lcs/tr/MIT-LCS-TR-684.pdf

A Key evolving signature schemes

We present the formal definitions of key evolving signature scheme of [6,20]

Definition 10. *A key evolving signature scheme* $\mathsf{KES} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Update})$ *is a tuple of algorithms such that:*

1. $\mathsf{Gen}(1^k, T)$ *is a probabilistic key generation algorithm that takes as input a security parameter* 1^k *and the total number of periods* T*, outputting a key pair* $(\mathsf{KES.sk}_1, \mathsf{KES.vk})$*, where* $\mathsf{KES.vk}$ *is the verification key and* $\mathsf{KES.sk}_1$ *is the initial signing key (we assume that the period* j *to which a signing key* $\mathsf{KES.sk}_j$ *corresponds is encoded in the signing key itself).*
2. $\mathsf{Sign}_{\mathsf{KES.sk}_j}(m)$ *is a probabilistic signing algorithm that takes as input a secret key* $\mathsf{KES.sk}_j$ *for the time period* $j \leq T$ *and a message* m*, outputting a signature* σ_j *on* m *for time period* j *(we assume that the period* j *for which a signature* σ_j *was generated is encoded in the signature itself).*
3. $\mathsf{Verify}_{\mathsf{KES.vk}}(m, \sigma_j)$ *is a deterministic verification algorithm that takes as input a public key* $\mathsf{KES.vk}$*, a message* m *and a signature* σ_j*, outputting* 1 *if* σ_j *is a valid signature on message* m *for time period* j *and* 0 *otherwise.*
4. $\mathsf{Update}(\mathsf{KES.sk}_j)$ *is a probabilistic secret key update algorithm that takes as input a secret key* $\mathsf{KES.sk}_j$ *for the current time period* j *and outputs a new secret key* $\mathsf{KES.sk}_{j+1}$ *for time period* $j+1$*. We define* $\mathsf{KES.sk}_{T+1}$ *as the empty string and set it as the output of* $\mathsf{Update}(\mathsf{KES.sk}_T)$*.*

Correctness: *for every key pair* $(\mathsf{KES.sk}_1, \mathsf{KES.vk}) \leftarrow \mathsf{Gen}(1^k, T)$*, every message* m *and every time period* $j \leq T$*,* $\mathsf{Verify}_{\mathsf{KES.vk}}(m, \mathsf{Sign}_{\mathsf{KES.sk}_j}(m)) = 1$*.*

B Chain equivalence

Definition 11 (Equivalence $\equiv$). *Two chains* $\mathcal{C}_0$ *and* $\mathcal{C}_1$ *are equivalent,* $\mathcal{C}_0 \equiv \mathcal{C}_1$*, if* $\mathcal{C}'_0 = \mathcal{C}'_1$*, where* $\mathcal{C}'$ *is obtained from* $\mathcal{C}$ *with the following procedure:*
- *Let* $\mathcal{C} = (B_0, ..., B_R)$ *and set* $B'_0 \leftarrow (0, \epsilon, \epsilon, 0)$*,* $\mathcal{C}' \leftarrow B'_0$*,*
- *For* $k \in [1, R]$*:*
  - $B'_k \leftarrow (B_k.\mathsf{sl}, H(B'_{k-1}), \epsilon, B_k.\mathsf{ldr})$
  - $\mathcal{C}' \leftarrow \mathcal{C}'|B'_k$

We lift this definition of equivalence to chain sets (or trees), where each chain in one set has exactly one equivalent chain in the other.

C Protocol parameterizations

We illustrate parameterizations of (d, δ, Δ)-FairPoS in Table 1 for discussion. Observe that (d, δ, Δ)-FairPoS can be parameterized with any d, δ and Δ such that Equation 5 holds. Delay parameter d can always be parameterized sufficiently long for any k-common prefix, since the probability that k-common prefix is

violated falls exponentially with decreasing parameter Δ, which is chosen independently of d.

Note that a smaller Δ will increase security, as the Δ-monotonicity property will ensure faster honest chain growth. The extractions schedule D is a function of d, δ and Δ (Equation 5). Then, let the difference between the extraction schedule and delay parameter "$D-d$" be interpreted as the "time gap" between the moment session keys are extracted by parties and the slot when these will appear on chain. A shorter "$D-d$" interval implies that light-clients which are not performing key extractions will observe the chain-state sooner, as decryption keys are posted to the chain within a shorter time period.

d	δ	Δ	n	D	$D-d$
30	6	8	15	120	90
60	6	8	30	240	180
60	6	21	4	84	24
90	6	21	6	126	36

Table 1. Parameterizations of (d, δ, Δ)-FairPoS

We observe following trade-offs in Table 1: for a smaller Δ, which increases the security of FairPoS, we obtain a larger $D-d$, implying a larger lag in chain-state observability for non-extracting parties. This is intuitive, as a shorter Δ parameter implies that honest parties must "converge" sooner on extendable chains. This is achieved by permitting additional "slack" or "$D-d$" in the extraction schedule. We argue that this is necessary trade-off which can be acceptable in practice, as parties which need to decrypt inputs as soon as possible are likely to perform key extractions (e.g. DeFi market arbitrageurs). Even at larger "$D-d$" intervals, key extraction remains critical for practicality: newly joining parties will immediately obtain keys to decrypt the entire input history beginning from genesis from the blockchain itself.

D Proofs

Theorem 1. *(Δ-Monotonicity of FairPoS) Every protocol execution of (d, δ, Δ)-FairPoS results in an honest tree $\mathcal{T}$ that exhibits the Δ-monotonicity property.*

Proof (Theorem 1). Δ-monotonicity holds if every chain-tip that is honestly generated at slot i iconsidered a candidate for extension by the first honest leader at or following slot $i+\Delta$. In other words, the proof obligation is to demonstrate honest chain extendability (Equation 7) holds for every honest chain-tip, Δ slots following its generation. We prove this by induction:

Base case (Genesis). The genesis block at slot 0 is trivially extendable at any slot $\mathsf{sl} \geq \Delta$. Recall that we permit negative chain indices, e.g. $\mathcal{C}_{-j}$ for

$j \in \mathbb{Z}$, which denote "empty blocks", for which the local receipt delay $\mathbf{r}^{(i)}(\mathcal{C}_{-j})$ is always $\perp$, and by the definition of receipt delay (Definition 8) is interpreted as 0. Thus, when extending genesis, every "empty block" in the extraction window and the "m"th Δ-window will not violate the receipt delay bounds imposed by the *extendable-chain* criteria in Equation 7.

Inductive case (Slot i). A chain $\mathcal{C}$ honestly extended at slot i must fulfill the receipt delay constraints in Equation 7 in the view of the slot leader. This honest leader will have rebroadcast every block in $\mathcal{C}$ upon arrival: thus, in the worst case, all other honest parties will have received blocks in $\mathcal{C}$ with an additional receipt delay of δ slots compared to the leader of slot i. The following must hold for $\mathcal{C}$ in the view of any honest leader of slot $j \geq i + \Delta$:

- For $m \in [0 : n)$: each block of $\mathcal{C}$ in the "m"-th Δ-window (Equation 6) in the view of leader of slot i is now in the "$m+1$"-th Δ-window in the view of the honest leader of slot j, it is permitted an additional *worst-case* additional delay of δ by the extendable chain definition (Equation 7).
- Blocks in the "$n-1$"th window of $\mathcal{C}$ in the view of leader i are in the extraction window of leader of slot j, and are also permitted an additional *worst-case* delay of δ (Equation 7).

□

Theorem 3. (Equivalent trees) For any (d, δ, Δ)-FairPoS execution λ induced by a charactistic string $w \in \{0, 1, \perp\}^*$, $\Gamma_0 \rightarrow^{\lambda} \Gamma$, there exists a Δ-PoS execution λ' induced by same w, $\Gamma'_0 \rightarrow^{\lambda'} \Gamma'$, such that the viable chains in Γ are equivalent to the viable branches in Γ'.

$$\forall w \in \{0,1,\perp\}^* : \forall \lambda \in \mathsf{exec}_{\delta}^{\mathsf{FairPoS}}(\Gamma_0, w), \Gamma_0 \xrightarrow{\lambda} \Gamma : \exists \lambda' \in \mathsf{exec}_{\Delta}^{\mathsf{PoS}}(\Gamma'_0, w), \Gamma_0 \xrightarrow{\lambda'} \Gamma' : \\ \mathsf{viableChains}_{\delta}^{\mathsf{FairPoS}}(\Gamma) \equiv \mathsf{viableBranches}_{\Delta}^{\mathsf{PoS}}(\Gamma')$$

Proof (Theorem 3). Let $\mathsf{exec}_{\delta}^{\pi}(\Gamma, w_t)$ denote all possible *single round* executions of π in a δ-synchronous setting induced by $w_t \in \{0, 1, \perp\}$ at slot t, beginning with protocol state Γ. Further, we define the *honest extendable tree* of FairPoS state Γ_t as the union of the extendable prefixes (Equation 13) of all tree branches in the view of honest parties.

$$\mathcal{T}_{\mathsf{ext}}^{\mathcal{H}}(\Gamma_t) = \bigcup_{\mathcal{C} \in \mathcal{T}^{\mathcal{H}} : \exists \mathcal{B} \in \mathcal{T}^{\mathcal{H}} : \mathcal{C} = \mathsf{extPrefix}(t, \mathcal{B}, \{\mathbf{r}^{(i)}\}_{i \in \mathcal{H}})} \mathcal{C}$$

For a FairPoS state Γ_t and PoS state Γ'_t where $\mathcal{T}_{\mathsf{ext}}^{\mathcal{H}}(\Gamma_t) \equiv \mathcal{T}^{\mathcal{H}}(\Gamma'_t)$ we observe that viable branches of Γ_t and viable chains of Γ'_t converge by definition.

$$\mathsf{viableChains}_{\delta}^{\mathsf{FairPoS}}(\Gamma_t) \equiv \mathsf{viableBranches}_{\Delta}^{\mathsf{PoS}}(\Gamma'_t)$$

We prove the theorem round-wise, by induction.

Base step ($0 \rightarrow 1$). For the first round executed on the genesis block (slot 0) and given a characterstic string w we must prove:

$$\begin{gathered}\forall\, r \in \mathsf{exec}_{\delta}^{\mathsf{FairPoS}}(\Gamma_0, w_1)\,,\ \Gamma_0 \xrightarrow{r} \Gamma_1 : \exists\, r' \in \mathsf{exec}_{\Delta}^{\mathsf{PoS}}(\Gamma'_0, w_1)\,,\ \Gamma'_0 \xrightarrow{r'} \Gamma'_1 \\ \mathcal{T}_{\mathsf{ext}}^{\mathcal{H}}(\Gamma_1) \equiv \mathcal{T}^{\mathcal{H}}(\Gamma'_1)\end{gathered} \tag{16}$$

We describe the translation from r to r'. If w_1 is honest, then the elected honest parties will generate a block extending genesis in the rounds of both protocols (genesis is immediately extendable, as no key extractions are due in the block associated at slot 1). If w_1 is dishonest, then whatever the blocks the adversary generates in r is performed by the adversary in r': resulting extendable honest tree in Γ_1 and honest tree in Γ'_1 must be equivalent.

Induction step ($t \rightarrow t+1$). We must prove

$$\begin{gathered}\forall\, r \in \mathsf{exec}_{\delta}^{\mathsf{FairPoS}}(\Gamma_t, w_{t+1})\,,\ \Gamma_t \xrightarrow{r} \Gamma_{t+1} : \exists\, r' \in \mathsf{exec}_{\Delta}^{\mathsf{PoS}}(\Gamma'_t, w_{t+1})\,,\ \Gamma'_t \xrightarrow{r'} \Gamma'_{t+1} \\ \mathcal{T}_{\mathsf{ext}}^{\mathcal{H}}(\Gamma_{t+1}) \equiv \mathcal{T}^{\mathcal{H}}(\Gamma'_{t+1})\end{gathered} \tag{17}$$

By induction hypothesis it holds that this holds that $\mathcal{T}_{\mathsf{ext}}^{\mathcal{H}}(\Gamma_t) \equiv \mathcal{T}^{\mathcal{H}}(\Gamma'_t)$. For any honest or adversarial action in round r, we illustrate the respective action in round r', before arguing the equivalence of extendable honest trees in Γ_{t+1} and honest trees in Γ'_{t+1}.
Honest actions in $\mathsf{FairPoS}$ round r are also performed in PoS round r':

H1 If w_{t+1} is a uniquely honest slot, the longest (extendable) chains will be equivalent in the honest party's view, and the honest leader(s) will extend the equivalent chains of both protocol executions in rounds r and r' respectively. (By induction hypothesis, the the honest extendable tree in Γ_t is equivalent to the honest tree in Γ'_t). A newly extended chain from honest party i is added to the message queue $\mathbf{m}^{(i)}$ and $\mathcal{T}^{\mathcal{A}}$ in both r and r'.

Adversarial actions include dishonest block generation (T2 in Figure 1) and the delivery of messages (M2 in Figure 1). The translation of adversarial actions from r round to the r' is described.

A1 If the adversary adds dishonest blocks to $\mathcal{T}^{\mathcal{A}}$ at any adversarial slot $t' \leq t+1$ in the round r, this action is performed in r' round.
A2 If the adversary delivers a $\mathcal{C}$ with adversarial chain tip from $\mathcal{T}^{\mathcal{A}}$ to $\mathcal{T}^{(i)}$ in r:
  a. If $\mathcal{C}$ is extendable by party i in state Γ_{t+1} following r, $\mathcal{A}$ delivers equivalent $\mathcal{C}'$ from $\mathcal{T}^{\mathcal{A}\prime}$ to $\mathcal{T}^{(i)\prime}$ in the r' round: $\mathcal{C}, \mathcal{C}'$ must both exist in the adversarial tree views in states Γ_t, Γ'_t due to **A1**.
  b. Else if $\mathcal{C}$ *remains* unextendable in state Γ_{t+1} by party i, equivalent $\mathcal{C}'$ is not added to the equivalent honest tree $\mathcal{T}^{(i)\prime}$ in r'.
A3 If the adversary delivers a chain $\mathcal{C}$ from message queue $\mathbf{m}^{(i)}$ to an honest tree view $\mathcal{T}^{(i)}$ in r:
  a. If $\mathcal{C}$ is extendable by party i in state Γ_{t+1} following r, $\mathcal{A}$ delivers equivalent $\mathcal{C}'$ to $\mathcal{T}^{(i)\prime}$ in the r' round.

b Else if $\mathcal{C}$ is not extendable by party i in state Γ_{t+1}, its equivalent $\mathcal{C}'$ is not added to the honest tree in round r': since $\Delta > \delta$, it is always possible for $\mathcal{A}$ defer an honest chain delivery for a longer delay in PoS (Δ-synchrony) than in the ΔPoS (δ-synchrony) execution.

A4 If any $\mathcal{C} \in \mathcal{T}^{(i)}$ in an honest tree view *becomes* extendable in slot $t+1$ following round r in the view of party i.

a. If $\mathcal{C}$ and equivalent $\mathcal{C}'$ feature an honest chain tip it must be present in an honest message queue $\mathbf{m}^{(j)\prime}$ of the PoS execution (**H1**) and will be delivered to party i in round r'. An honest chain tip in a FairPoS execution can remain unextendable for up to Δ slots (Theorem 1), which is exactly the maximum message delay permitted for messages in $\mathbf{m}^{(j)\prime}$ in the PoS Δ-synchronous setting.

b. If $\mathcal{C}$ and equivalent $\mathcal{C}'$ feature an adversarial chain tip, $\mathcal{C}'$ cannot be in an honest tree view nor an honest message queue in the PoS execution prior to slot $t+1$: **A2** requires an adversarial chain tip in FairPoS to be extendable before it is introduced to an honest tree in PoS. Instead, any $\mathcal{C}'$ with adversarial tip must be in $\mathcal{T}^{\mathcal{A}}$, $\mathcal{T}^{\mathcal{A}\prime}$ of both executions (**A1**) at the beginning of the round, and $\mathcal{C}'$ can therefore still be delivered to $\mathcal{T}^{(i)\prime}$ from $\mathcal{T}^{\mathcal{A}\prime}$ by the adversary in round r'.

The extendable honest tree in Γ_{t+1} and honest tree in Γ'_{t+1} must be equivalent since for an addition of a *newly extendable chain* (**A2**(a), **A3**(a), **A4**) to the honest tree in FairPoS round r, the equivalent chain in PoS is added to the honest tree in round r'. □

Theorem 4. (k-Common prefix in FairPoS) *Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0,1]$ and some $\epsilon > 0$. The probability that $\mathcal{A}$ makes the protocol violate the k-common-prefix property in a δ-synchronous environment throughout a period of R slots is no more than* $\exp(\ln R + \Delta - \Omega(k))$.

Proof. (Theorem 4) Let w be drawn from dominant distribution $\mathcal{D}^f_\alpha$ (Equation 1), with honest stake α and parameter f satisfying $\alpha(1-f)\Delta = (1+\epsilon)/2$ for some $\epsilon > 0$. From Corollary 1 and Theorem 2 we infer the following:

$$\Pr_{w \leftarrow \mathcal{D}^f_\alpha}[\mathsf{div}^{\mathsf{FairPoS}}_\delta(w) \geq k] \leq \Pr_{w \leftarrow \mathcal{D}^f_\alpha}[\mathsf{div}^{\mathsf{PoS}}_\Delta(w) \geq k] \leq \exp(\ln R + \Delta - \Omega(k)) \quad (18)$$

From Corollary 1, $\mathsf{div}^{\mathsf{FairPoS}}_\delta(w) \geq k \implies \mathsf{div}^{\mathsf{PoS}}_\Delta(w) \geq k$, implying the left equality in Equation 18. The right inequality is inferred from Theorem 2.

Theorem 5. ((τ, s)-Chain growth in (d, δ, Δ)-FairPoS) Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0,1]$ and some $\epsilon > 0$. Then the probability that $\mathcal{A}$ makes the protocol violate the chain growth property with parameters $s \geq 4\Delta$ and $\tau = c\alpha/4$ throughout a period of R slots, is no more than $\exp(-c\alpha s/(20\Delta) + lnR\Delta + O(1))$, where c denotes the constant $c := c(f, \Delta) = f(1-f)^\Delta$.

Proof (Theorem 5). The proof of chain-growth in FairPoS closely follows that of Ouroborous Praos, as both analyses assume the dominant distribution (Definition 1) to model leader elections during protocol executions. Thus, we reproduce the main proof argument of Theorem 6 in [13] for the convenience of the reader, and refer to [13] for the derivation of the exact probability bounds, which are directly inferred from the dominant distribution.

Recall that the definition of chain growth requires that if the longest chain possessed by an honest party at the onset of some slot sl_1 is $\mathcal{C}_1$, and the longest chain possessed by a (potentially different) honest party at the onset of slot $\mathsf{sl}_2 \geq \mathsf{sl}_1 + s$ is $\mathcal{C}_2$, then $\mathsf{len}(\mathcal{C}_2) - \mathsf{len}(\mathcal{C}_1) \geq \tau s$.

Let a uniquely honest slot (0) be Δ-right-isolated if it is followed with Δ consequent slots which are empty ($\perp$). Let $\mathcal{C}$ be a chain with a tip that is generated in a Δ-right-isolated uniquely honest slot. Then the next slot leader will necessarily consider $\mathcal{C}$ candidate for extension since

- Chain $\mathcal{C}$ must be have arrived in the view of all honest parties after δ slots in a δ-synchronous setting, where $\delta < \Delta$ (Equation 5).
- Chain $\mathcal{C}$ must be extractable after Δ slots (Theorem 1).

Thus, in the view of all honest parties, chain $\mathcal{C}$ must be a candidate for extension according to the longest-extractable-chain rule (Equation 8) in FairPoS.

Now, let $\hat{\mathsf{sl}}_1, ..., \hat{\mathsf{sl}}_h$ be the increasing sequence of all Δ-right-isolated uniquely honest slots among the slots in $T := \{\mathsf{sl}_1 + \Delta, \mathsf{sl}_1 + \Delta + 1, ..., \mathsf{sl}_2 - \Delta\}$. Observe that since $\hat{\mathsf{sl}}_1 \geq \mathsf{sl}_1 + \Delta$, the leader of $\hat{\mathsf{sl}}_1$ will append a block to a chain that is at least as long as $\mathcal{C}_1$, since $\mathcal{C}_1$ will be known to him and will be considered in the longest-extractable-chain selection rule. Therefore, the chain that the leader of $\hat{\mathsf{sl}}_1$ diffuses will be at least 1 block longer than $\mathcal{C}_1$. Analogously, the leader of every $\hat{\mathsf{sl}}_i$ will diffuse a chain that is at least 1 block longer than the chain diffused by the leader of $\hat{\mathsf{sl}}_{i-1}$ since $\hat{\mathsf{sl}}_{i-1}$ is Δ-right-isolated. Finally, the chain diffused by the leader of $\hat{\mathsf{sl}}_h$ will be known to all parties at slot sl_2 and hence $\mathsf{len}(C2)$ will be at least as long as this chain. It follows that $\mathsf{len}(\mathcal{C}_2) - \mathsf{len}(\mathcal{C}_1) \geq h$.

It remains to bound the number h of Δ-right-isolated uniquely honest slots among the slots with indices in T . To make our notation more flexible, let $H_T(x)$ denote the number of Δ-right-isolated uniquely honest slots among the slots from T in $x \in \{0, 1, \perp\}^R$. From Theorem 6 in [13] we have for $c = f(1-\Delta)^{\Delta}$:

$$\Pr_{x \leftarrow \mathcal{D}_\alpha^f}[H_T(x) < c\alpha s/4] = \Delta \cdot e^{-\frac{c\alpha(s-3\Delta)}{20\Delta}}$$

Applying the union bound over R slots, we conclude that the probability that there is a chain growth violation with parameters s and $\tau = c\alpha/4$ is no more than

$$R\Delta \exp(-c\alpha(s-3\Delta)/(20\Delta)) = \exp(-c\alpha(s-3\Delta)/(20\Delta) + \ln R\Delta)$$

□

Theorem 6. ((μ, k)-Chain quality in (d, δ, Δ)-FairPoS) Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1-f)\Delta = (1+\epsilon)/2$ for active slot coefficient $f \in (0, 1]$ and some $\epsilon > 0$. Then the probability that $\mathcal{A}$ makes FairPoS violate the chain quality property with parameters k and $\mu = 1/k$ throughout a period of R slots, is no more than $\exp(\ln R - \Omega(k))$.

Proof (Theorem 6). The proof of chain-growth in FairPoS closely follows that of Ouroborous Praos, as both analyses assume the dominant distribution (Definition 1) to model leader elections during protocol executions. Thus, for the convenience of the reader, we restate and adapt Lemma 4 in [13] and its main proof argument, from which Theorem 6 follows.

Lemma 2. *((Adapted from Lemma 4 in [13])) Let $k, \Delta \in \mathbb{N}$ and $\epsilon \in (0, 1)$. Let A be an adaptive adversary against the protocol (d, δ, Δ)-FairPoS corrupting up to $(1-\alpha)$ stake for some $\alpha > 0$ satisfying $\alpha(1-f)^{\Delta} = (1+\epsilon)/2$. Let $B_1, ..., B_k$ be a sequence of consecutive blocks in a chain $\mathcal{C}$ possessed by an honest party. Then at least one block B_i was created in a Δ-right-isolated uniquely honest slot, except with probability $\exp(-\Omega(k))$.*

For convenience, let us call a slot good if it is Δ-right-isolated uniquely honest, and bad if it is neither empty nor good. Moreover, we call a block good (resp. bad) if it comes from a good (resp. bad) slot.

Towards contradiction, assume that all blocks $B_1, ..., B_k$ are bad. Let G_1 denote the latest good block preceding B_1 in $\mathcal{C}$, and G_2 the earliest good block appearing after B_k in $\mathcal{C}$ (or the last block of $\mathcal{C}$, if there is no good one). Note that all blocks between G_1 and G_2 are bad.

Let $\hat{\mathsf{sl}}_1$ (resp. $\hat{\mathsf{sl}}_2$) denote the good slot in which G_1 (resp. G_2) was created (if G_2 is not good, let $\hat{\mathsf{sl}}_2$ be the current slot). Denote by T the continuous sequence of slots between $\hat{\mathsf{sl}}_1$ and $\hat{\mathsf{sl}}_2$, excluding $\hat{\mathsf{sl}}_1$ and including $\hat{\mathsf{sl}}_2$. As we argued in the proof of Theorem 5, in each good slot in T the (unique) leader creates a block that has depth increased by at least 1 compared to the block from the previous good slot. Therefore, we have $\mathsf{depth}(G_2) \geq \mathsf{depth}(G_1) + g$, where g is the number of good slots in T . However, in chain $\mathcal{C}$ we have $\mathsf{depth}(G_2) \leq \mathsf{depth}(G_1) + b$, where b is the number of bad slots in the same sequence T . These two conditions can only be satisfied at the same time if $g \leq b$, we will now show that this is very unlikely.

We can bound $\Pr_{x \leftarrow \$ \mathcal{D}_{\alpha}^{f}}[g(x) \leq b(x)]$ as follows: we know that $\alpha(1-f)^{\Delta} = (1+\epsilon)/2$ and this implies that good slots are sampled with higher probability than bad slots. Therefore, $\Pr_{x \leftarrow \$ \mathcal{D}_{\alpha}^{f}}[g(x) \leq b(x)]$ falls exponentially with k. Lemma 2 directly implies Theorem 6. □

Lemma 1. (Input fairness from CG, CP and CQ in (d, δ, Δ)-FairPoS) If for an execution of (d, δ, Δ)-FairPoS, (τ, d)-chain growth, $(d\tau(\tau - \delta/(\Delta - \delta)) - 1)$-common prefix, and $(1/(D+1), D+1)$-chain quality hold, where $D = d\Delta/(\Delta - \delta)$, then input fairness is implied.

Proof (Lemma 1). The proof requires us to infer input fairness from chain growth and common-prefix parameters as stated in the Lemma 1.

More informally, we frame the proof obligation as follows. At the onset of a given slot t, we are given a time budget of $\sim d$ slots, and must show that chain-growth will result in the sufficient growth in the length of the chain possessed by any honest party, such that for any encrypted input inserted at a block in slot t, it will reach the k-common-prefix within the given time budget, unless it becomes part of an abandoned branch.

Let $\mathcal{C}$ be the chain extended by an honest party at honest slot t. For simplicity, let us first assume that all slots are uniquely honest. An adversary begins the extraction of $\mathsf{id} = \mathcal{C}.\mathsf{tip}$ at the onset of its generation. Then, there remains $d-1$ slots between the encryption of the input at slot t and its decryption, since the input at t is encrypted with the parent block as session id. A chain growth rate of τ implies that the longest chain possessed by any honest party after $d-1$ time must increase by $k = \tau(d-1)$. Thus, in this naive scenario, input fairness would be implied by (τ, d)-CG and $(\tau(d-1))$-CP.

In the case that slots are *not all uniquely honest*, the adversary must be permitted a head-start in extracting the session key idk from any block: let us denote this the *time advantage*, which comes from two properties of the chain possessed by the honest user at the onset of slot t:

1. *Leading empty slots* between $\mathcal{C}.\mathsf{tip.sl}$ and current slot t. These empty slots represent a head-start the adversary has in decrypting inputs inserted in a child block of $h(\mathcal{C}.\mathsf{tip})$ generated at slot t.
2. *Leading adversarial block span* in $\mathcal{C}$ including $\mathcal{C}.\mathsf{tip}$, allowing it to generate blocks immediately after the extraction period d instead of waiting the full extraction schedule $D = d + n\delta$, as an honest party would. In the worst case, such a adversarial block span always leads up to $\mathcal{C}.\mathsf{tip}$, which is also adversarially generated, permitting the adversary an additional head-start in decrypting inputs.

For the (1) *leading empty slots*, the (τ, d)-CG property gives us the maximum number of slots in d, in which *no blocks* were generated for $\mathcal{C}$, namely $d(1-\tau)$. For the (2) *leading adversarial block span*, we quantify the *time advantage* gained from the leading adversarial block span as follows: for every D consecutive adversarial blocks, the adversary gains an additional $n\delta$ *time advantage*, since it does not have to wait the entire extraction schedule $D = d + n\delta$, where $n = D/\Delta = d/(\Delta - \delta)$. Note that we are granted $(1/(D+1), D+1)$-chain quality in Lemma 1, where $D = n\Delta = d\Delta/(n-\delta)$ as in Equation 5. We assume the worst case, namely, that all D slots leading up to t are indeed adversarial, and thus permit the adversary the maximum possible extraction time-advantage of $n\delta$ slots.

Thus the total time advantage in producing the adversarial block $\mathcal{C}.\mathsf{tip}$ obtained from (1) and (2) is given by $t_{\mathsf{adv}} = d(1-\tau)+n\delta = d(1-\tau+\delta/(\Delta-\delta))$. Thus we require a *contracted* common prefix parameter $k = \tau(d-t_{\mathsf{adv}})-1$, in order for the honest input at slot t to either reach the common prefix before the adversary

(with *time advantage*) can complete the key extraction of $\mathsf{id} = h(\mathcal{C}.\mathsf{tip})$, or not join the common prefix at all. Rewriting gives us $k = d\tau(\tau - \delta/(\Delta - \delta)) - 1$ □

Theorem 7. (Input fairness in (d, δ, Δ)-FairPoS) Let $\mathcal{A}$ be an an adaptive adversary against the protocol (d, δ, Δ)-FairPoS that corrupts up to $(1-\alpha)$ stake, where α be such that $\alpha(1 - f)\Delta = (1 + \epsilon)/2$ for active slot coefficient $f \in (0, 1]$ and some $\epsilon > 0$. Then, the probability that $\mathcal{A}$ makes the FairPoS violate the input fairness property falls exponentially with d.

Proof (Theorem 7). With Lemma 1 we obtain input fairness from (τ, d)-CG and $(d\tau(\tau - \delta/(\Delta - \delta)) - 1)$-CP. Further, for an execution of (d, δ, Δ)-FairPoS for R slots,

- (τ, d)-CG with parameters $d \geq 4\Delta$ and $\tau = c\alpha/4$ is violated with probability no more than $\exp(-d\alpha c/(20\Delta) + lnR\Delta + O(1))$, where c denotes the constant $c := c(f, \Delta) = f(1 - f)^{\Delta}$ (Theorem 5).
- k-CP with parameter $k = d\tau(\tau - \delta/(\Delta - \delta)) - 1$ is violated with probability no more than $\exp(\ln R + \Delta - \Omega(k))$ (Theorem 4).
- $(1/(D + 1), D + 1)$-CQ with parameter $D = d\Delta/(\Delta - \delta)$ is violated with probability no more than $\exp(\ln R + \Omega(D))$ (Theorem 6).

Probabilities above decline exponentially with increasing d. □