Securing Proof-of-Stake Nakamoto Consensus Under Bandwidth Constraint

Joachim Neu
jneu@stanford.edu

Srivatsan Sridhar
svatsan@stanford.edu

Lei Yang
leiy@csail.mit.edu

David Tse
dntse@stanford.edu

Mohammad Alizadeh
alizadeh@csail.mit.edu

ABSTRACT

Satoshi Nakamoto's Proof-of-Work (PoW) longest chain (LC) protocol was a breakthrough for Internet-scale open-participation consensus. Many Proof-of-Stake (PoS) variants of Nakamoto's protocol such as Ouroboros or Snow White aim to preserve the advantages of LC by mimicking PoW LC closely, while mitigating downsides of PoW by using PoS for Sybil resistance. Previous works have proven these PoS LC protocols secure assuming all network messages are delivered within a bounded delay. However, this assumption is not compatible with PoS when considering bandwidth constraints in the underlying communication network. This is because PoS enables the adversary to reuse block production opportunities and spam the network with equivocating blocks, which is impossible in PoW. The bandwidth constraint necessitates that nodes choose carefully which blocks to spend their limited download budget on. We show that 'download along the longest header chain', a natural download rule for PoW LC, emulated by PoS variants, is insecure for PoS LC. Instead, we propose 'download towards the freshest block' and prove that PoS LC with this download rule is secure in bandwidth constrained networks. Our result can be viewed as a first step towards the co-design of consensus and network layer protocols.

1 INTRODUCTION

Consensus. In the state machine replication (SMR) formulation of the consensus problem, a group of *nodes* aim to order *transactions* received from the environment into a common *ledger*. For this purpose, nodes exchange messages and perform computations as prescribed by the consensus protocol. Consensus is made non-trivial by an adversary who has some control over message delays and who controls a certain fraction of nodes and can cause them to deviate from the protocol in an arbitrary (*Byzantine*) manner in a concerted effort to disturb consensus. *Secure* consensus is achieved if the resulting transaction ledgers across different honest nodes and points in time are *consistent* so that it is meaningful to speak of *the* single common ledger (which is *safe*), and if that ledger is *live* in the sense that every transaction gets assigned a position in the ledger soon after it is input to honest nodes for the first time.

Nakamoto's Longest Chain Protocol. In their seminal Bitcoin white-paper [28], Satoshi Nakamoto describes the *longest chain* (LC) consensus protocol. In this protocol, honest nodes broadcast blocks to each other. A block contains a list of transactions, a nonce, and a reference to a parent block, resulting in chains of blocks up to a root

genesis block that is common knowledge. A block is *valid* if a cryptographic hash of it is smaller than a certain fixed threshold, and if the transactions it contains have been legitimized by the owners of the affected assets and are consistent with respect to transactions preceding it as ordered in the same block and its ancestor blocks. Every node adds valid blocks it receives to its local copy of the block tree. Nodes also aim to produce new blocks. For this purpose they bundle recently received transactions together with a reference to the block at the tip of the longest chain in their local block tree and use brute force search to determine a nonce such that the resulting block is valid (*i.e.*, the hash inequality is satisfied). Newfound valid blocks are broadcast to other nodes, completing the process. Each node outputs as ledger the transactions as ordered in the prefix of the block that is k-deep in the longest chain of its local block tree.

Besides being remarkably simple, Nakamoto's LC consensus protocol has two outstanding properties. First, it enables consensus in a *permissionless* setting by way of using *proof-of-work* (PoW) puzzles as a Sybil resistance mechanism [16, 25]. The bottleneck to block production is finding nonces which lead to valid blocks which satisfy the hash inequality, and as long as the majority of hash power at every point in time is controlled by honest nodes, honest nodes output a secure ledger [19, 30]. Second, the LC can tolerate *dynamic participation* in the sense that the ledger remains secure even as the total hash power participating in the protocol as well as its distribution among participants varies over time.

Proof-of-Stake Longest Chain. A drawback of Nakamoto's PoW LC is the high electricity consumption and as a result a tendency for centralization of nodes at places of relatively low electricity cost. Because of the qualities of LC and to overcome the drawbacks of PoW, protocols such as Ouroboros [3, 12, 26] and Sleepy Consensus [11, 31] preserve the operating principle of the LC but replace PoW with *proof-of-stake* (PoS) lotteries, where a party is assigned random block production opportunities in proportion to the amount of stake it holds in the system, thus effectively substituting 'one CPU, one vote' by 'one coin, one vote'. For this purpose, nodes use synchronized clocks to count time slots of a predetermined duration. For every time slot, nodes evaluate a block production lottery associated with their cryptographic identity. For instance in [3, 12], nodes get to produce a new valid block if the output of a *verifiable random function* (VRF) is below a threshold proportional to the node's stake.

Longest Chain Under Bandwidth Constraint. While PoS LC behaves in some aspects similar to PoW LC, it differs drastically in others. For instance, in PoS, block production opportunities can be 'reused' in the sense that when a node is eligible to produce a

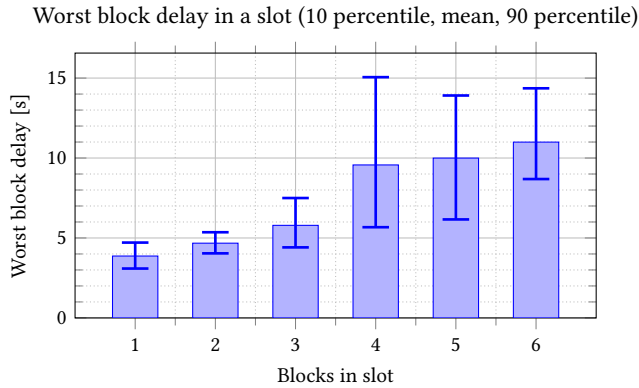Worst block delay in a slot (10 percentile, mean, 90 percentile)

Figure 1: The block propagation delay measured in experiments with Cardano's Ouroboros implementation increases as the number of blocks produced and broadcast in the network per slot is increased, showing that network delay cannot be treated as independent of network load. Details of the experimental setup are given in Appendix B.

Honest block download rate at victim node

Figure 2: The rate at which a victim node downloads honest blocks when there is no attack and when the node is under a spamming attack. "Actual" represents the actual growth rate of the honest chain during the same period. Note that the download rate at the node keeps up with the honest chain growth rate when there is no attack, but fails when under a spamming attack. Details of the experimental setup are given in Appendix C.

block in a certain time slot, it can in fact create many equivocating but equally valid blocks for the same time slot, each potentially with a different set of transactions and/or attached to a different parent block. This problem arises because block production 'lottery tickets' in PoS can not depend on the proposed block's transactions. Otherwise an adversary could increase its chances to produce a block by trying various sets of transactions (*grinding*). Similarly the PoS lotteries can not depend on the parent block, as the adversary could extend several chains at once to increase their chance of block production (*nothing-at-stake* attack [4]). In PoW however, each block production opportunity corresponds to a unique block (a combination of transaction set, parent block and nonce), thus the rate of block production opportunities simultaneously bounds the rate at which new valid blocks can be created.

Previous analysis [12, 15, 31] shows that this difference is immaterial in the synchronous network model where message propagation delay between honest nodes is controlled by the adversary, but below a known upper bound Δ. Under such a network model, PoS LC and PoW LC behave the same in terms of security, transaction throughput and confirmation latency. This model, however, is over-idealized in that it assumes a fixed delay upper bound for every single message, irrespective of whether few or many messages are being transmitted simultaneously. The model does not capture notions of capacity and congestion which have a significant impact on the behavior of real networks. In fact, an increase in network delay with increasing network load (via increased block size) has been demonstrated previously for Bitcoin [14]. Similarly, increasing the network load (via increasing the number of blocks per slot) leads to increased network delay in our experiments (see Figure 1) with Cardano's Ouroboros implementation—a PoS protocol. Once we enrich the network model to capture such phenomena, the difference in the behavior of PoW LC and PoS LC with respect to reuse of block production opportunities strikes. The possibility of producing (infinitely) many equivocating valid blocks per block production opportunity opens up new adversarial strategies
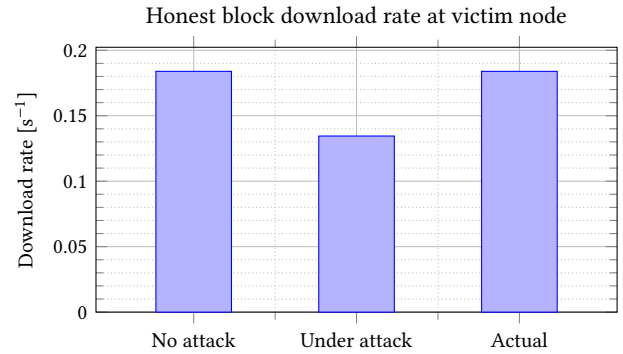
in which the adversary aims to exhaust limited network resources with useless 'spam' in an attempt to disturb consensus. This protrudes in another experiment (see Figure 2) where nodes run PoS LC with our implementation of Cardano's block download logic as per [22]. Adversarial spamming (through block equivocations) causes significant network traffic at the victim node, leaving insufficient bandwidth for the victim to download honest blocks. As a result, the victim is unable to keep up with the growth of the honest chain. In turn, the victim node could be easily fooled by a longer chain from the adversary, potentially resulting in a safety violation.

We model a bandwidth constrained network as follows. Recall that blocks in Nakamoto consensus consist of a list of transactions as *block content*, and the information pertaining to the PoS/PoW lottery and the block tree structure (reference to parent block) as *block header*. Since a block's header is small compared to its content, we assume that block headers propagate with a known delay upper bound Δ_h between honest nodes. At any point after obtaining a block header, a node can request the corresponding block content from the network. Since a block's content is large, every honest node can only download a limited number of blocks' contents per time slot. This model is inspired by the peer-to-peer network designs used for blockchain protocols. For instance, in the Cardano network [9, 10], each node advertises its block header chain to its peers, which in turn decide based on the block headers which block contents to fetch. Without a carefully designed *download rule* for the protocol to determine which blocks honest nodes should spend their scarce bandwidth on, we will see that consensus cannot be achieved with PoS LC.

The 'Download Along The Longest Header Chain' Rule. Given that in LC, honest nodes extend the longest chain, a natural download rule is 'download along the longest header chain', *i.e.*, based on the block tree structure obtained from block headers, a node identifies the longest (header) chain, and prioritizes downloading the blocks
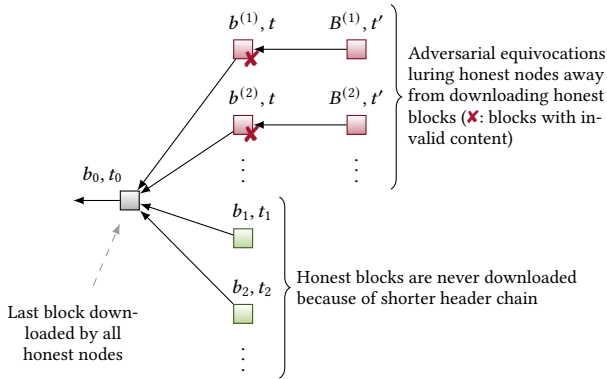
Figure 3: In PoS LC with 'download along the longest header chain' rule, an adversary can stall consensus indefinitely if it has two block production opportunities $t < t'$ at which it creates infinitely many equivocating chains $b_0 \leftarrow b^{(i)} \leftarrow B^{(i)}$ where $b^{(i)}$ have invalid content. The blocks of later honest block production opportunities $... > t_2 > t_1 > t' > t$ are never downloaded by other honest nodes, because they prioritize the longer adversarial header chains, wasting their bandwidth downloading each $b^{(i)}$ only to discard it immediately thereafter because of invalid content.

along that chain. Indeed, Bitcoin does exactly that [1]. Cardano's Ouroboros implementation also follows this paradigm in broad strokes [9, 10, 13] for chain selection [24] and block downloads [23]. As long as the block production rate is low relative to the download bandwidth, this (and other rules that ensure that nodes download a block at most once) work well for PoW LC, simply because the number of distinct blocks is limited by the number of block production opportunities.

Unfortunately, as illustrated in Figure 3, this download rule fails for PoS LC in that the resulting protocol is not secure, even if the block production rate is low and the adversary controls a small minority of the stake. The reason is that the adversary can use consecutive adversarial block production opportunities (at t and t' in Figure 3) to produce infinitely many equivocating chains ($b_0 \leftarrow b^{(i)} \leftarrow B^{(i)}$ in Figure 3). To avoid honest nodes building on these equivocating chains, the adversary fills $b^{(i)}$ with invalid content, which honest nodes can only detect after they have already wasted their scarce bandwidth to download it. As a result, honest nodes produce blocks off b_0 in their block production opportunities ($b_1, b_2, ...$ at $t_1 < t_2 < ...$ in Figure 3), but these are never downloaded by other honest nodes because the adversarial header chains are longer and thus of higher download priority. In practice, implementations such as Cardano's of the Ouroboros protocol attempt to mitigate such attacks using heuristics, *e.g.*, rejecting peers that propagate invalid or equivocating blocks [9, 10, 13]; to which an adversary may respond by creating Sybil peers (recall that there is no relation between consensus stakeholders and peers in the underlying communication network). While particular instantiations may or may not be susceptible to variants of this attack based on implementation details and heuristics, this discussion suggests fundamental flaws with the 'download along the longest header

chain' philosophy and prompts a search for a conceptually simple and provably secure alternative.

In the attack described above, we observe that even though new honest blocks are being proposed, the highest download priority is given to the older adversarial blocks. If nodes downloaded the fresher blocks proposed in more recent time slots $t_1, t_2, ...$ instead, then this attack would not succeed. This observation motivates the following download rule.

The 'Download Towards The Freshest Block' Rule. We propose a simple download rule for PoS LC, 'download towards the freshest block', *i.e.*, at every instant an honest node identifies the block from the most recent block production opportunity based on the header information, and downloads any missing blocks in the prefix and including that freshest block. This rule ensures that every now and then when an honest node proposes a block, other honest nodes make progress towards downloading that block and its prefix. This way, honest nodes have a chance to produce blocks extending it in the future, and align their block production efforts toward a particular chain, which is arguably the key stepping stone of prior security analysis techniques [15, 31] on which we build. In particular, our download rule avoids the attack of Figure 3 in which no honest block will ever be downloaded.

Our 'download towards the freshest block' rule for PoS LC has another interesting property in comparison to the 'download along the longest header chain' rule of PoW LC. 'Download along the longest header chain' uses no information about the timing of blocks, which is natural as such information is not reliably available in PoW LC. Yet, this rule does not work when simply copied for PoS LC. In contrast, 'download towards the freshest block' works for PoS LC and leverages the block production timing information which is particular to PoS LC but not reliably available in PoW LC.

Duration Of Time Slots. Our analysis suggests another modification which is unique to PoS. In PoW LC, the rate at which block production lotteries are evaluated depends on the total hash power in the network. In PoS however, new lotteries can only be evaluated once every time slot, and the duration of the time slot can be chosen by the protocol designer. This difference impacts the relationship between network delay and security. In PoW, a centralized adversary can quickly run lotteries for the next block immediately after mining one block, while honest nodes must wait for each block to be propagated and downloaded by the other nodes. This allows an adversary with less than 50% of the hash power to build a longer chain faster than the honest network. In PoS with the 'download towards the freshest block' rule, we can try to bridge this gap by increasing the slot duration so that all honest nodes download the freshest block and its prefix by the end of the time slot (with high probability). While the honest nodes are busy downloading the freshest block, the adversary does not get any more block production opportunities, and hence does not gain any advantage due to their centralization. With our analysis, we suggest that to achieve security under bandwidth constraint, rather than further reduce the block production probability, we should increase the time slot duration to allow honest nodes to download the freshest block.

Joachim Neu, Srivatsan Sridhar, Lei Yang, David Tse, and Mohammad Alizadeh

Our Contributions. We prove that LC PoS with our drop-in 'download towards the freshest block' rule is secure in synchronous networks with bandwidth constraints in which the adversary can spam the network with equivocating blocks at an arbitrary rate, the adversary can withhold blocks, and the adversary can release blocks with invalid content that honest nodes discard after downloading. Our analysis extends and refines proof techniques for PoS LC in synchronous networks without bandwidth constraint [31]. More broadly, our result can be viewed as a first step towards the co-design of consensus and network layer protocols, in the following manner. Our rule can also be shown to provide security for traditional propose-and-vote-style consensus protocols such as Streamlet [8]. Moreover, spamming becomes an even more serious concern in high throughput systems such as [5, 17, 38] where honest messages already consume substantial bandwidth, leaving little margin to deal with spam. Our download rule and analysis can be used to carefully design such protocols and analyze their security.

Related Work. Network-level attacks on Bitcoin have bee studied in [2, 6]. Eclipse attacks on peer-to-peer networks, where an adversary uses several IP addresses to occupy all connections maintained by a victim node, have been studied in [21, 33–35] and in the context of Bitcoin in [7]. The temporary shutdown of a PoS LC protocol Solana [37] in September 2021, was reportedly due to an increase in the transaction load in the network, and "lack of prioritization of network-critical messaging caused the network to start forking" [27]. These examples indicate that network-level threats are a serious concern and raise the question of careful co-design of consensus and network layer protocols.

We point to [5] and [18] which identify the need for careful modelling of bandwidth constraints in the context of high throughput systems. For PoW, [5, 14, 36] note that the network delay increases with the size of the message (*i.e.*, block size in this case). In this model, it is assumed that as long as the network load is less than the bandwidth, every message is downloaded within a given delay bound which depends on the message size but is independent of the total network load.

In the PoS context, [18] extends this by modelling the inbox of each node as a queue. Each message undergoes a propagation delay before being added to the recipient's inbox queue. The recipient can retrieve messages from their queue at a rate limited by their bandwidth, resulting in a queuing delay. However, the security result [18, Theorem 1] still assumes a bounded network delay. This assumption is only shown to hold when the adversary does not corrupt any nodes and does not send or delay any messages [18, Theorem 3], and therefore the security claim does not hold for all adversarial strategies. In particular, this excludes adversaries that can spam the network using equivocating blocks and cause attacks such as in Figure 3. The model we use is a variant of that in [18] with the difference that nodes can inspect a small segment ('block header') at the beginning of every message in their queue and decide based on that which message ('block content') to prioritize for download (subject to the bandwidth constraint). This modification allows us to prove security against a general adversary, even with unbounded equivocations.

Although our work is the first to prove security of PoS LC under bandwidth constraints, our analysis uses tools from several years

---

Algorithm 1 Idealized PoS LC consensus protocol $\Pi^{\rho,\tau,T_{\text{conf}}}$ (helper functions/procedures: Appendix A, $\mathcal{F}_{\text{headertree}}$: Algorithm 2)

1: **on** INIT(genesisHeaderChain, genesisTxs)
2:     $h\mathcal{T}, dC \leftarrow \{\text{genesisHeaderChain}\}$, genesisHeaderChain $\quad\triangleright$ *Initialize header tree and longest downloaded chain with genesis block*
3:     blkTxs[dC] $\leftarrow$ genesisTxs $\quad\triangleright$ *Default of blkTxs entries:* unknown
4: **on** RECEIVEDHEADERCHAIN(C)
5:     **assert** $\mathcal{F}_{\text{headertree}}$.VERIFY($C$) = true $\quad\triangleright$ *Verify block production lottery and header block chain structure, see Algorithm 2*
6:     $h\mathcal{T} \leftarrow h\mathcal{T} \cup \text{prefixChainsOf}(C)$
7:     $\mathcal{Z}$.BROADCASTHEADERCHAIN(C)
8: **on** RECEIVEDCONTENT(C, txs) $\quad\triangleright$ *Callbacks are queued until $C \in h\mathcal{T}$ and blkTxs[C'] $\notin \{\text{unknown}, \text{invalid}\}$ for all prefix chains $C' \leq C$*
9:     **assert** $C[-1]$.txsHash = Hash(txs) $\quad\triangleright$ *Verify txs match header*
10:     **if** txsAreSemanticallyValidWrtPrefixesOf(C, txs) $\quad\triangleright$ *Store and relay txs for block if txs are valid, otherwise mark block as* invalid
11:         blkTxs[C] $\leftarrow$ txs
12:         $\mathcal{Z}$.UPLOADCONTENT(C, txs)
13:     **else**
14:         blkTxs[C] $\leftarrow$ invalid
15:     $\mathcal{T}' \leftarrow h\mathcal{T} \setminus \{C' \in h\mathcal{T} \mid \text{blkTxs}[C'] \in \{\text{unknown}, \text{invalid}\}\}$
16:     $dC \leftarrow \text{longestChain}(\mathcal{T}')$ $\quad\triangleright$ *Update longest downloaded chain*
17: **on** SCHEDULECONTENTDOWNLOAD() $\quad\triangleright$ *Called when download idle*
18:     $\mathcal{T}' \leftarrow \{C \in h\mathcal{T} \mid \forall C' \leq C : \text{blkTxs}[C'] \neq \text{invalid}\}$ $\quad\triangleright$ *Discard chains with invalid blocks*
19:     $C \leftarrow \arg\max_{C' \in \mathcal{T}'} C'$.time $\quad\triangleright$ *Find freshest block*
20:     $C \leftarrow \arg\min_{C' \in \{C'' \leq C \mid \text{blkTxs}[C''] = \text{unknown}\}} |C'|$ $\quad\triangleright$ *First block with missing content in prefix of freshest block*
21:     $\mathcal{Z}$.REQUESTCONTENT(C) $\quad\triangleright$ *If the requested content is available, callback* RECEIVEDCONTENT(.) *will be triggered by $\mathcal{Z}$*
22: **for** time slots $t \leftarrow 1, ..., T_h$ of duration τ
23:     txs $\leftarrow \mathcal{Z}$.RECEIVEPENDINGTXSSEMANTICALLYVALIDWRT(dC)
24:     **if** $C' \neq \perp$ with $C' \leftarrow \mathcal{F}_{\text{headertree}}$.EXTEND($t, dC,$ txs) $\quad\triangleright$ *Check eligibility to produce a new block, and if so do so, see Algorithm 2*
25:         $\mathcal{Z}$.UPLOADCONTENT(C', txs)
26:         $\mathcal{Z}$.BROADCASTHEADERCHAIN(C')
27:     $\mathcal{Z}$.OUTPUTLEDGER($dC^{\lceil T_{\text{conf}}}$) $\quad\triangleright$ *Ledger of node i at time t:* LOG_i^t

---

of security analysis for LC protocols, including [4, 12, 15, 19, 20, 30–32]. We use the idea of loners from [32] (also called convergence opportunities in [31]), and the concept of pivots from [31] (similar to Nakamoto blocks from [15]).

Outline. We recapitulate the details of Nakamoto's LC protocol and introduce our formal model for bandwidth constrained networks in Section 2. We provide an overview of the security argument of PoS LC with the 'download towards the freshest block' rule under bandwidth constraint in Section 3. Technical details of the security analysis are provided in Section 4.

2 PROTOCOL AND MODEL

Model Main Features. For ease of exposition, we assume a static set of N active *nodes*, each with a cryptographic identity corresponding to one unit of stake. Our analysis can be easily extended to the case of heterogeneous and dynamic stake using tools from [11, 12]. Nodes' cryptographic identities are common knowledge. We are interested in the large system regime $N \rightarrow \infty$. A *static* adversary $\mathcal{A}$ chooses a set of nodes (up to a fraction β of all nodes,

---

Algorithm 2 Idealized functionality $\mathcal{F}_{\text{headertree}}$: block production lottery and header block chain structure (cf. Appendix A)

1: **on** INIT(genesisHeaderChain)
2:     $\mathcal{T} \leftarrow \{\text{genesisHeaderChain}\}$
3: **on** ISLEADER(P, t) **from** $\mathcal{A}$ or $\mathcal{F}_{\text{headertree}}$
4:     **if** lottery$[P, t] = \bot$
5:         lottery$[P, t] \xleftarrow{\$}$ (true with probability ρ/N, else false)
6:     **return** lottery$[P, t]$
7: **on** EXTEND(t', C, txs) **from** party P **at** time slot t
8:     $C' \leftarrow C \,\|\, \text{newBlock}(\text{time} = t', \text{party} = P, \text{txsHash} = \text{Hash}(\text{txs}))$
9:     **if** $C \in \mathcal{T} \land$ ISLEADER$(P, t') = \text{true} \land C.\text{time} < t' \le t$
10:         $\mathcal{T} \leftarrow \mathcal{T} \cup \{C'\}$
11:         **return** C'
12:     **return** $\bot$
13: **on** VERIFY(C)
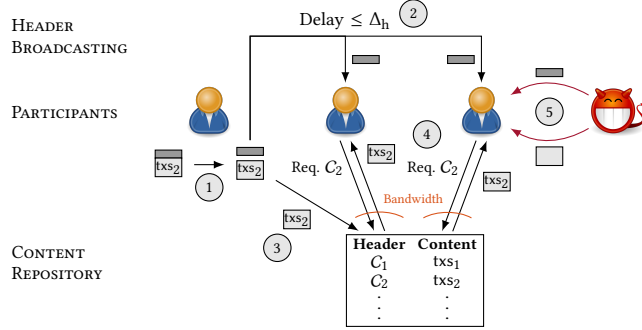14:   **return** $C \in \mathcal{T}$

Figure 4: In our model, block headers are propagated with a known delay upper bound Δ_{h}, while block content is subject to a bandwidth constraint. ① An honest node produces a new valid block, consisting of header and content. ② Block headers are broadcast and arrive at honest nodes within at most Δ_{h} delay. ③ Block content is submitted to an idealized 'repository'. ④ A hash of the corresponding block content is included in the block header. Upon request, the content of a certain block is obtained from the 'repository', subject to a constraint on the rate of downloaded block contents. ⑤ An adversary can push block headers and block content to honest nodes independent of delay and bandwidth constraints.

where β is common knowledge) to corrupt before the randomness of the protocol is drawn and the execution commences. Uncorrupted *honest* nodes follow the protocol as specified at all times, corrupted *adversarial* nodes deviate from the protocol in an arbitrary *Byzantine* manner coordinated by the adversary in an attempt to inhibit consensus. Thus, for simplicity, we assume that all nodes are always *awake*. Our analysis builds on techniques from 'The Sleepy Model of Consensus' [31] and the refined machinery therein can be used to extend our analysis to the setting of asleep/awake honest nodes in a straightforward manner.

Protocol Main Features. Pseudocode of an idealized PoS LC Nakamoto consensus protocol employing the 'download towards the freshest block' rule is provided in Algorithm 1 (cf. [31, Figure 3]).

Implementation details of the block production lottery and the handling of the blockchain data structure are abstracted away in the idealized functionality $\mathcal{F}_{\text{headertree}}$ provided in Algorithm 2 (cf. [31, Figure 2]). An index of the helper functions used in the pseudocode is provided in Appendix A. With specific implementations of $\mathcal{F}_{\text{headertree}}$ a variety of PoS LC protocols can be modelled such as protocols from the Ouroboros family [3, 12, 26] and the Sleepy Consensus [11, 31] family. In the main loop of the PoS LC protocol (Algorithm 1, lines 22ff.) the node attempts in every time slot (which is of duration τ) to produce a new block containing transactions txs and extend the longest downloaded chain (denoted dC) in the node's local view. If successful, the block content txs and the resulting new block header chain C' are provided to the environment $\mathcal{Z}$ for dissemination to all nodes.

Dissemination of Block Headers and Contents. As illustrated in Figure 4, block header chains broadcast via $\mathcal{Z}$.BROADCASTHEADERCHAIN$(.)$ are delivered by the environment $\mathcal{Z}$ to every node with a delay determined by $\mathcal{A}$, up to a delay upper bound Δ_{h} that is common knowledge. For simplicity, we assume that block headers broadcast by honest nodes arrive at all other nodes by the beginning of the next time slot and the content may be downloaded thereafter, *i.e.* $\Delta_{\text{h}} < \tau$ (*lock step*). Once an honest node receives a new valid block header chain (Algorithm 1, lines 4ff.), the node adds it to its local header tree h$\mathcal{T}$. Block content uploaded via $\mathcal{Z}$.UPLOADCONTENT$(.)$ is kept by $\mathcal{Z}$ in an idealized repository. Honest nodes can request the content for a particular header via $\mathcal{Z}$.REQUESTCONTENT$(.)$. If available, the content requested from the repository will be delivered by $\mathcal{Z}$ to the honest node (Algorithm 1, lines 8ff.). To model the network's bandwidth constraint of C blocks per second, $\mathcal{Z}$ delivers at most $\bar{C} = C\tau$ block contents requested from the repository to each honest node per time slot.[1] Upon verifying that the content matches the hash in the block header and that the txs are valid with respect to the ledger determined by the block's prefix, the node adds txs to its local view. Otherwise, the block is marked as invalid, to prevent downloading any of its descendants in the future. Finally, the node updates its longest downloaded chain.

'Download Towards The Freshest Block' Rule. Whenever no block content download is in progress, the 'download towards the freshest block' rule is used to determine which block's content to request from $\mathcal{Z}$ next (Algorithm 1, lines 17ff.). For this purpose, the header tree h$\mathcal{T}$ is pruned by invalid blocks and their descendants. Then, the first unknown block in the prefix of the freshest block is requested. The callback RECEIVEDCONTENT$(.)$ is triggered if and only if the requested block content is found in $\mathcal{Z}$'s repository.

Adversarial Strategies And Powers. Adversarial strategies and powers include but are not limited to: reusing block production opportunities to produce multiple blocks (*equivocations*, by calling $\mathcal{F}_{\text{headertree}}$.EXTEND$(.)$ multiple times each with a different txs or a different chain C); extending any chain using past block production opportunities as long as the purported block production time slots along any chain are strictly increasing; releasing block headers late or selectively to honest nodes; proactively pushing block headers or block content to honest nodes irrespective of

---

[1]Unlike [18], we do not model the upload bandwidth because honest nodes only send very few messages in our protocol.

delay or bandwidth constraints (by triggering the node's respective RECEIVEDHEADERCHAIN(.) or RECEIVEDCONTENT(.) callback); withholding the content of blocks; including invalid txs in blocks; breaking ties in chain selection and the download rule.

3 HIGH LEVEL SECURITY ARGUMENT

We define two variables T and γ which will be determined after the analysis. Intuitively, T acts as an upper bound on the delay for downloading a block—a replacement for Δ in the traditional analysis of [31] and [15]. However, unlike Δ in [15, 31], not all blocks are guaranteed to be downloaded within a delay of T slots, and the upper bound is only probabilistic. This is similar to the probabilistic delay bounds used in [18]. Note that Δ in the traditional analysis is the delay for propagation of the entire block, but Δ_h in our model is the propagation delay for only the smaller headers. The variable γ determines the minimum ratio of the bandwidth to the rate of block production opportunities. Clearly, we require at least $\gamma > 1$. Our analysis reveals a finite value of γ which guarantees security even in the presence of unbounded equivocations. Our proof is divided into two sections. In Section 4.3, we prove that the protocol $\Pi^{\rho,\tau,T_{\text{conf}}}$ achieves safety and liveness whenever the random sequence specifying the block production opportunities satisfies certain properties. Next, in Section 4.4, we show that these properties hold throughout the execution horizon with high probability.

We begin by identifying special time slots, called T-loners (inspired by [32]), where a single honest block production opportunity is isolated by at least T slots on both sides from any other block production opportunities. Such slots correspond vaguely to convergence opportunities in [31]. For T time slots after a T-loner, the honest block proposed in the T-loner is the freshest block and hence honest nodes download towards it. If each honest node has at most $\bar{C}T$ blocks to download in the prefix of the freshest block, this block will be downloaded within T slots. A sufficient condition for this is that the freshest block is a descendent of a recent ($< \gamma T$ slots old) block which has been downloaded by every honest node, and the number of block production opportunities between the two blocks is less than $\bar{C}T$ (Proposition 1). This is sufficient in spite of equivocations because the blocks in the prefix of the freshest block must be from distinct time slots, so the number of blocks to be downloaded is bounded by the number of slots with block production opportunities. If the block from a T-loner is downloaded within T slots, the block from the successive T-loner (being honest) must be at a greater height. Therefore under the above sufficient condition, the rate of T-loners gives a lower bound on the downloaded chain growth rate of honest nodes (Proposition 2). In Lemma 3, we prove that if $\bar{C}$ is sufficiently larger than $\gamma\rho$ (where ρ is the rate of block production opportunities per slot, see Algorithm 2 line 4), then the number of block production opportunities in any interval of γT slots is less than $\bar{C}T$ with high probability.

Inspired by [31], we define a T-pivot as a slot such that in every interval containing the T-pivot, there are more T-loners than block production opportunities that do not correspond to T-loners (or there are no block production opportunities that do not correspond to T-loners). In Lemma 2, we show that if the block production rate is small enough and the adversary controls only a minority of the total stake, then in any interval of $(\gamma - 1)T$ slots, there is a slot that

is both a T-loner and a T-pivot, with high probability. Like in [31], this proof involves showing that any given slot satisfies the pivot conditions with a positive probability, and then bootstrapping over large intervals. We also define a *great block* as an honest block from a slot t that has the property that it stays in the longest downloaded chain of every honest node after slot $t + T$.

Next, we show that great blocks appear often. For this, we show that the block proposed in the first slot that is both a T-loner and a T-pivot (pivot-loner) after a great block, is also a great block (Lemma 4). From this, we conclude that every block produced in a pivot-loner slot is also a great block (since the genesis block itself is a great block). Like earlier works [15, 31], we prove Lemma 4 by contradiction by supposing that the block from the pivot-loner is displaced from some honest node's longest downloaded chain. In contrast to [15, 31], we require a recursion to prove the result. This is because in our model, we only claim that blocks from T-loners are downloaded within T slots if they are proposed within γT slots after the great block. For the recursion, we identify that within $(\gamma - 1)T$ slots after the first pivot-loner, there will be another pivot-loner, and then for γT more slots, all honest nodes can download the blocks corresponding to T-loners within T slots. This recursion ends when we identify a pivot-loner whose block is displaced from some honest node's longest downloaded chain within γT slots after its proposal. At this point, we can guarantee that the blocks from all T-loners around this pivot-loner are downloaded in T slots, allowing us to recover the proof from [31] to show a contradiction. See Figure 5 for an illustration of this recursion.

Finally, the property of a great block ensures that honest transactions will be included in this block or its prefix, and the transactions in the prefix of this block remain in every honest node's ledger forever. Therefore, the presence of a great block leads to safety and liveness of the ledger output by the consensus protocol. Combining the above arguments, we have that a great block grants safety and liveness, every block from a pivot-loner is a great block, and a pivot-loner occurs in every interval of $(\gamma - 1)T$ slots. This ensures that if we wait for γT slots to confirm a transaction, all honest nodes download a block from a pivot-loner with high probability, and hence we get safety. We also get liveness with a latency of γT (Lemma 1).

Finally, we set the analysis variables T and γ, and the protocol parameters $\tau, \rho, T_{\text{conf}}$ such that we achieve safety and liveness over an arbitrarily long time horizon for a given adversarial fraction $\beta < \frac{1}{2}$, except with negligible probability (Theorem 1). With κ as the security parameter, our analysis sets $T = 1$ slot, and $\gamma = \text{poly}(\kappa)$ Similar to the result in the bounded delay models in [12, 15, 31], we must limit the block production rate as $\rho < \frac{1-2\beta}{4(1-\beta)}$. We require that each honest node can download $\bar{C} = \text{poly}(\kappa)$ blocks per time slot. Under a given bandwidth constraint that does not scale with the security parameter, this requires setting the slot size $\tau = \text{poly}(\kappa)$. Finally, transactions can be confirmed after waiting for $T_{\text{conf}} = \text{poly}(\kappa)$ slots.
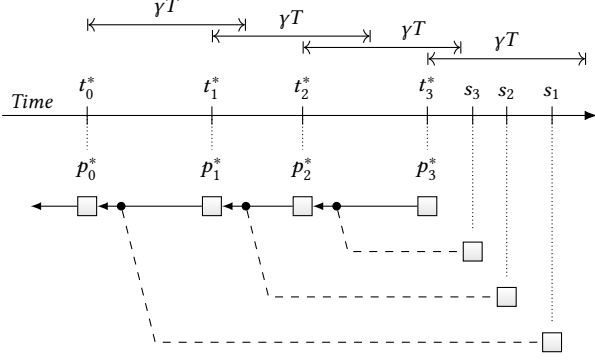
Figure 5: Illustration of the recursion in the proof of Lemma 4. The great block p_0^* is proposed in slot t_0^*. The sequence $t_1^*, t_2^*, \ldots$ denotes the successive pivot-loners after t_0^*. Slot s_j denotes the first slot in which an honest node's longest chain does not contain p_j^*. Note that p_j^* is a descendant of p_{j-1}^* and hence, $s_j \leq s_{j-1}$. Also note that each new pivot-loner arrives within γT slots of the previous one and the recursion stops when $s_3 < t_3^* + \gamma T$. The intervals of γT duration guarantee that all T-loners in the interval are downloaded within T slots.

4 SECURITY PROOF DETAILS

4.1 Definitions

The PoS LC protocol $\Pi^{\rho, \tau, T_{\text{conf}}}$ has three parameters. The length of each time slot is τ seconds, the average number of block production opportunities per time slot is ρ, and the confirmation latency is T_{conf} slots. The network has the following additional parameters. Each honest node has a download bandwidth of C block contents per second (for convenience, we fix the size of the block content). For ease of analysis, we will deal with $\bar{C} = C\tau$ such that each honest node can download the content of $\bar{C}$ blocks in one time slot. The adversary controls β fraction of the stake. We denote by κ the security parameter. We define additional variables T and γ which are unknown to the nodes in the protocol, and will be specified after the analysis.

Define the random variables H_t (and A_t) which are the number of honest (and adversarial) block production opportunities in slot t. An execution $\mathcal{E}^{\rho, \beta, T_{\text{h}}}$ of time horizon T_{h} is specified by the sequence $\{H_t, A_t\}_{t \leq T_{\text{h}}}$. We consider the regime where the number of nodes $N \to \infty$ and each of them holds one coin of stake. In this setting, by the Poisson approximation to a binomial random variable, we have $H_t \overset{\text{i.i.d.}}{\sim} \text{Poisson}((1 - \beta)\rho)$ and $A_t \overset{\text{i.i.d.}}{\sim} \text{Poisson}(\beta\rho)$, independent of each other.

Then we define the following constants:

$$p \triangleq \Pr[A_t + H_t > 0] \quad = 1 - e^{-\rho}, \tag{1}$$

$$p_{\text{H}} \triangleq \Pr[H_t = 1, A_t = 0] = (1 - \beta)\rho e^{-\rho}, \tag{2}$$

$$p_{\text{L}} \triangleq p_{\text{H}}(1 - 2pT). \tag{3}$$

Denote by $dC_i(t)$ the longest fully downloaded chain of an honest node i at the end of slot t. Let $|b|$ denote the height of a block b. We will also use the same notation $|C|$ to denote the length of

a chain C. Define $L_i(t) = |dC_i(t)|$. At the end of each slot, honest node i outputs the ledger $\text{LOG}_i^t = dC_i(t)^{\lceil T_{\text{conf}}}$, which consists of ordered transactions contained in all blocks in $dC_i(t)$ with time slot up to $t - T_{\text{conf}}$.

A consensus protocol is *secure* if it satisfies:

- *Safety:* For all time slots t, t' and honest nodes i and j, $\text{LOG}_i^t \preceq \text{LOG}_j^{t'}$ or $\text{LOG}_j^{t'} \preceq \text{LOG}_i^t$.
- *Liveness with parameter* T_{live}: If a transaction tx is received by all honest nodes before slot t, then for all honest nodes i and $t' \geq t + T_{\text{live}}$, $\text{tx} \in \text{LOG}_i^{t'}$.

Definition 1. A T-*loner* is a slot t such that $H_t = 1$, $A_t = 0$, and $\forall t' \in [t - T, t + T] \setminus \{t\} : A_{t'} = H_{t'} = 0$. Define the predicate $\text{Loner}_T(t)$ as true iff t is a T-loner.

Denote by $\mathcal{B}(r, s]$ the number of slots with block production opportunities in the interval $(r, s]$. Denote by $\mathcal{L}(r, s]$ the number of T-loners in the interval $(r, s]$ Denote by $\mathcal{N}(r, s]$ the number of block production opportunities in $(r, s]$ that are not T-loners.

$$\mathcal{B}(r, s] \triangleq \sum_{t=r+1}^{s} \mathbb{1}\{A_t + H_t > 0\}, \tag{4}$$

$$\mathcal{L}(r, s] \triangleq \sum_{t=r+1}^{s} \mathbb{1}\{\text{Loner}_T(t)\}, \tag{5}$$

$$\mathcal{N}(r, s] \triangleq \sum_{t=r+1}^{s} \mathbb{1}\{A_t + H_t > 0 \wedge \neg\text{Loner}_T(t)\}$$
$$= \mathcal{B}(r, s] - \mathcal{L}(r, s]. \tag{6}$$

Definition 2. T-*pivot* is a slot t such that

$$\forall(r, s] \ni t : \quad (\mathcal{L}(r, s] > \mathcal{N}(r, s]) \vee (\mathcal{N}(r, s] = 0). \tag{7}$$

The predicate $\text{Pivot}_T(t)$ is true iff t is a T-pivot.

We will say that a slot t is a T-pivot-loner iff $\text{Pivot}_T(t) \wedge \text{Loner}_T(t)$. The prefix T may be omitted when the value of T is clear.

Definition 3. An execution $\mathcal{E}^{\rho, \beta, T_{\text{h}}}$ satisfies $\text{FrequentPivots}_{T, \gamma}$ iff $\forall t \leq T_{\text{h}} - (\gamma - 1)T$:

$$\exists t' \in (t, t + (\gamma - 1)T] : \quad \text{Pivot}_T(t') \wedge \text{Loner}_T(t'). \tag{8}$$

Definition 4. An execution $\mathcal{E}^{\rho, \beta, T_{\text{h}}}$ satisfies $\text{FewBlockOpps}_{\bar{C}, T, \gamma}$ iff

$$\forall t \leq T_{\text{h}} - \gamma T : \quad \mathcal{B}(t, t + \gamma T] \leq \bar{C}T. \tag{9}$$

Definition 5. Define the predicate $\text{ChainGrowth}_{(r, s]}$ to be true iff for all T-loners $t_1, \ldots, t_m \in (r, s]$, the block proposed in t_j is

(1) downloaded by all honest nodes by the end of slot $t_j + T$,
(2) at a height greater than the block proposed in t_{j-1} (for $j > 1$).

Definition 6. A great block is an honest block b from slot t such that $\forall$ honest nodes i: $\forall t' \geq t + T : b \in dC_i(t')$.

Note that the genesis block, which is defined to be from slot 0 and 'downloaded' by honest nodes from the start, is a great block.

4.2 Proof Overview

Lemma 1. *If the execution $\mathcal{E}^{\rho, \beta, T_{\text{h}}}$ satisfies $\text{FrequentPivots}_{T, \gamma}$ and $\text{FewBlockOpps}_{\bar{C}, T, \gamma}$, then the protocol $\Pi^{\rho, \tau, T_{\text{conf}}}$ with $T_{\text{conf}} = \gamma T$ achieves safety and liveness with $T_{\text{live}} = 2\gamma T$ in the execution $\mathcal{E}^{\rho, \beta, T_{\text{h}}}$.*

Lemma 1 is proved in Section 4.3.

Lemma 2. *If* $2p_H(1 - 2pT) = p(1 + \epsilon_1)$ *for some* $0 < \epsilon_1 < 1$, *and* $w \geq \max\left\{\frac{2\ln(T_h)}{\alpha_1 p}, \frac{2T\ln(T_h)}{p_L}, 2\right\}$,

$$\Pr\left[\mathcal{E}^{\rho,\beta,T_h} : \neg\mathsf{FrequentPivots}_{T,\gamma}\right]$$

$$\leq T_h \exp(-\alpha_3(\gamma - 1)T) - T_h \exp\left(-\frac{wp_L}{2T}\right) - T_h^2 \exp(-\alpha_1 pw), \quad (10)$$

where $\alpha_1 = \eta\epsilon_1^2$ *for some* $\eta > 0$ *and* $\alpha_3 = \frac{1}{4}w^{-1-6/\alpha_1}$.

Lemma 2 is proved in Section 4.4.

Lemma 3. *If* $\bar{C} = \gamma p(1 + \epsilon_2)$ *for some* $\epsilon_2 > 0$, *then*

$$\Pr\left[\mathcal{E}^{\rho,\beta,T_h} : \neg\mathsf{FewBlockOpps}_{\bar{C},T,\gamma}\right] \leq T_h \exp(-\alpha_2\gamma pT), \quad (11)$$

where $\alpha_2 = \frac{\epsilon_2^2}{\epsilon_2 + 2}$.

Lemma 3 is proved in Section 4.4.

Theorem 1. *The protocol* $\Pi^{\rho,\tau,T_{\mathrm{conf}}}$ *with parameters* $\rho = \frac{1-2\beta-\epsilon_1}{4(1-\beta)}$, $\tau = \Omega\left((\kappa + \ln T_h)^m\right)$ *and* $T_{\mathrm{conf}} = \Omega\left((\kappa + \ln T_h)^m\right)$, *achieves safety and liveness with* $T_{\mathrm{live}} = \Omega\left((\kappa + \ln T_h)^m\right)$ *(where* $m = 2 + \frac{6}{\eta\epsilon_1^2}$), *against an adversary with* $\beta < \frac{1}{2}$ *fraction of stake, over a time horizon* T_h, *except with probability* $\mathrm{negl}(\kappa)$.

PROOF. From Lemma 1, safety and liveness holds except with probability

$$\Pr\left[\mathcal{E}^{\rho,\beta,T_h} : \neg\mathsf{FrequentPivots}_{T,\gamma} \vee \neg\mathsf{FewBlockOpps}_{\bar{C},T,\gamma}\right]$$

$$\leq T_h e^{-\alpha_3(\gamma-1)T} + T_h e^{-wp_L/2T} + T_h^2 e^{-\alpha_1 pw} + T_h e^{-\alpha_2\gamma pT} \quad (12)$$

by Lemma 2, Lemma 3 and a union bound. Let $\kappa' = \kappa + \ln T_h$. Let's set $T = 1$, *i.e.* the prefix of the honest block in every 1-loner is downloaded within 1 slot. Then we pick $\epsilon_1 > 0$ and set ρ such that $2p_H(1 - 2pT) = p(1 + \epsilon_1)$. For small ρ, $p_H \approx (1 - \beta)\rho$ and $p \approx \rho$. Therefore we choose ρ such that $\rho = \frac{1-2\beta-\epsilon_1}{4(1-\beta)}$ (note that $\epsilon_1 < 1 - 2\beta$). Next, we pick w such that $w \geq \max\left\{\frac{2\ln(T_h)+\Omega(\kappa)}{\alpha_1 p}, \frac{2T\ln(T_h)+\Omega(\kappa)}{p_L}, 2\right\}$. This ensures that the probabilities $T_h e^{-wp_L/2T}$ corresponding to not finding a T-loner in some interval of size w, and $T_h^2 e^{-\alpha_1 pw}$ corresponding to having more non-loners than loners in some interval of size at least w, are $\mathrm{negl}(\kappa)$. This results in $w = c_1\kappa'$ where c_1 depends on β and ϵ_1.

Next, we pick γ so that the probabilities $T_h e^{-\alpha_3(\gamma-1)T}$ corresponding to the pivot condition not holding for some interval smaller than w for all slots in an interval of $(\gamma - 1)T$ slots, and $T_h e^{-\alpha_2\gamma pT}$ corresponding to $\neg\mathsf{FewBlockOpps}_{\bar{C},T,\gamma}$, are $\mathrm{negl}(\kappa)$. Therefore we get $\gamma \geq \max\left\{1 + \frac{\ln(T_h)+\Omega(\kappa)}{T}4w^{1+6/\alpha_1}, \frac{\ln(T_h)+\Omega(\kappa)}{\alpha_2 pT}\right\} \approx c_2\kappa'^{2+6/\alpha_1}$. Finally, we require the bandwidth $\bar{C} = \gamma\rho(1 + \epsilon_2) \approx c_3\kappa'^{2+6/\alpha_1}$. We defined $\bar{C}$ as the bandwidth constraint in terms of blocks per slot. In order to interpret this in terms of bandwidth in blocks per second, we set $\tau \approx \frac{c_3}{C}\kappa'^{2+6/\alpha_1}$, allowing us to download $\bar{C}$ blocks per slot with a constant capacity C bytes per second. Finally, $T_{\mathrm{conf}} = \gamma T = c_4\kappa'^{2+6/\alpha_1}$ time slots and $T_{\mathrm{live}} = 2\gamma T$ accordingly. Note that T_{conf} corresponds to $\gamma T\tau = c_5\kappa'^{4+12/\alpha_1}$ in units of real time. Appendix D shows the optimality of τ and T_{conf} up to the order of the polynomial, and how one should choose ϵ_1. □

4.3 Combinatorial Analysis

The key difference between analysis in the synchronous model and the bandwidth constrained model is that in the bandwidth constrained case, not every block is guaranteed to be downloaded within a small delay irrespective of the network load. Below, we will identify a sufficient condition under which we can ensure that the unique honest block from a T-loner is downloaded within T slots, and show that this leads to a lower bound on the downloaded chain growth rate.

Proposition 1. *(Sufficient Condition for Chain Growth) Suppose that* $\mathcal{E}^{\rho,\beta,T_h}$ *satisfies* $\mathsf{FewBlockOpps}_{\bar{C},T,\gamma}$. *Further, suppose that an honest block* b_0 *proposed in slot* t_0 *is such that* $\forall i, \forall t' \in [t_0+T, s], b_0 \in \mathrm{d}C_i(t')$. *Then,* $\mathsf{ChainGrowth}_{(t_0+T,s']}$ *where* $s' = \min\{s, t_0 + \gamma T\}$.

PROOF. Let $t_1, ..., t_m$ be the T-loners in $(t_0 + T, s']$. Since honest node i proposes in slot t on the tip of $\mathrm{d}C_i(t - 1)$, the blocks in slots $t_1, ..., t_m$ are descendants of b_0.

Let b_j be the block from t_j for $1 \leq j \leq m$. Note that each T-loner t_j requires that there are no other blocks produced in slots $[t_j, t_j + T]$. The header of b_j is received by all honest nodes at the end of t_j. Due to the downloading rule, during the time $(t_j, t_j + T]$, all honest nodes download the chain containing b_j. Note that block production times must be strictly increasing in valid chains. Then, using $\mathcal{B}(t_0, t_0 + \gamma T] \leq \bar{C}T$, there are less than $\bar{C}T$ blocks after b_0 on the chain containing b_j. Furthermore, since T-loners are honest blocks and honest nodes only propose on their downloaded chain, the prefix of each T-loner can be downloaded (*i.e.*, does not contain invalid or missing blocks). Thus, the blocks proposed at T-loners $t_1, ..., t_m$ are downloaded by all honest nodes before the end of slot $t_1 + T, ..., t_m + T$, respectively. Further, since honest nodes propose on their longest chain and $t_{j+1} > t_j + T$, the height of b_{j+1} is more than the height of b_j. □

Proposition 2. *Let* t_0 *be such that* $\mathsf{ChainGrowth}_{(t_0+T,s]}$ *holds with* T-*loners* $t_1, ..., t_m \in (t_0 + T, s]$. *Then,*

$$L_{\min}(t) - L_{\min}(t_j + T) \geq \mathcal{L}(t_j + T, t - T] \quad (13)$$

for all $0 \leq j < m$ *and* $t_j + T < t \leq s + T$.

PROOF. Let the block proposed in slot t_j be b_j. Fix a j such that $0 \leq j < m$. Since honest nodes propose on their longest downloaded chain, $|b_{j+1}| \geq L_{\min}(t_j + T) + 1$. Since $\mathsf{ChainGrowth}_{(t_0+T,s]}$, the honest blocks from all T-loners in $(t_j + T, t - T]$ are at different heights and are downloaded before time t. Then we conclude that $L_{\min}(t) \geq |b_{j+1}| + \mathcal{L}(t_j + T, t - T] - 1$. □

Lemma 4 (Fresh Block Lemma). *Suppose that* $\mathcal{E}^{\rho,\beta,T_h}$ *satisfies both* $\mathsf{FrequentPivots}_{T,\gamma}$ *and* $\mathsf{FewBlockOpps}_{\bar{C},T,\gamma}$. *Further, suppose that a great block* p_0^* *occurs in time slot* t_0^* *and let* t_1^* *be the first slot such that* $t_1^* > t_0^*$ *and* $\mathsf{Loner}_T(t_1^*) \wedge \mathsf{Pivot}_T(t_1^*)$. *Then, the block* p_1^* *in slot* t_1^* *is a great block.*

PROOF. For the sake of contradiction, let s_1 be the first slot such that $s_1 \geq t_1^* + T$ and $p_1^* \notin \mathrm{d}C_i(s_1)$ for some honest node i.

If $s_1 < t_1^* + \gamma T$, we may use Proposition 1 to show that T-loners in the interval $(t_0 + T, s_1]$ can be downloaded soon. This case will be treated in the end. But if this is not the case, we will use a recursion to find another pivot-loner that satisfies this condition.

For this, first assume that $s_1 \geq t_1^* + \gamma T$. If so, then due to FrequentPivots$_{T,\gamma}$, there exists another pivot-loner t_2^* such that $t_1^* < t_2^* \leq s_1 - T$. Note that the unique honest block p_2^* from t_2^* must be a descendant of p_1^* because p_1^* is in all honest nodes' longest chains before t_2^*. Further, if s_2 is the first time such that $s_2 \geq t_2^* + T$ and $p_2^* \notin dC_i(s_2)$ for some i, then $s_2 \leq s_1$ because p_2^* is a descendant of p_1^*.

Now, further if $s_2 \geq t_2^* + \gamma T$, then there exists another pivot-loner t_3^* such that $t_2^* < t_3^* \leq s_2 - T$. Just as in the case above, p_3^* is a descendant of p_2^* and therefore $s_3 \leq s_2$. This recursion will terminate when we get a pivot-loner t_m^* such that $s_m < t_m^* + \gamma T$. To see that the recursion must terminate, we note that $s_j \leq s_{j-1}$ and $t_j^* > t_{j-1}^*$, so $s_j - t_j^* < s_{j-1} - t_{j-1}^*$ for all j. The time slots defined above are illustrated in Figure 5. The scenarios in which the pivot-loner blocks may be de-confirmed are shown in Figure 6.

Now we show a contradiction for the case when $s_m < t_m^* + \gamma T$ for some $m \geq 1$. Let $C' = dC_i(s_m)$ for some i such that $p_m^* \notin C'$. Let h_m' be the last honest block corresponding to a T-loner on C', or the great block p_0^*, whichever has a greater height. Let h_m' be produced in the slot r_m. We know that $r_m \notin [t_m^* - T, t_m^* + T]$ since t_m^* is a T-loner. If $r_m > t_m^* + T$, then $p_m^* \in dC_{i'}(r_m - 1)$ for some i' since honest nodes mine on their longest downloaded chain. This is a contradiction because we assumed that s_m is the first slot such that $s_m \geq t_m^* + T$ and $p_m^* \notin dC_i(s_m)$ for some i. So, we conclude that $r_m < t_m^* - T$. All blocks in C' extending h_m' are from block production opportunities that are not T-loners. Moreover, these opportunities are from slots $(r_m + T, s_m]$ because r_m is a T-loner. So,

$$|C'| \leq |h_m'| + \mathcal{N}(r_m + T, s_m) \tag{14}$$

Since $s_m < t_m^* + \gamma T$, we have ChainGrowth$_{(t_m^* + T, s_m]}$ (Proposition 1). This means that all T-loners in $[t_m^* + T, s_m]$ are at different heights and are downloaded within T slots. However, since we also know that p_{m-1}^* is in all longest chains at least until t_m^* and $t_m^* < t_{m-1}^* + (\gamma - 1)T$, ChainGrowth$_{(t_{m-1}^* + T, t_m^*]}$ is also true. Similarly, for all $1 \leq j < m$, p_{j-1}^* is in all longest chains until $t_j^* < t_{j-1}^* + (\gamma - 1)T$, hence ChainGrowth$_{(t_{j-1}^* + T, t_j^*]}$ for all $1 \leq j < m$. Combining all these intervals and noting that there are no block production opportunities in $(t_{j-1}^*, t_{j-1}^* + T]$ since t_{j-1}^* is a T-loner, we have that ChainGrowth$_{(t_0^* + T, s_m]}$. In other words, all T-loners in $(t_0^* + T, s_m]$ lead to chain growth. By definition of h_m', we have $r_m \geq t_0^*$. Therefore, from Proposition 2,

$$L_{\min}(s_m) \geq L_{\min}(r_m + T) + \mathcal{L}(r_m + T, s_m - T) \tag{15}$$

Note that $L_{\min}(s_m) \leq L_i(s_m) \ \forall i$ and $|C'| = L_i(s_m)$ for some i. Also note that h_m' is from a T-loner and is a descendant of p_0^* (so $r_m > t_0^* + T$), or the same as p_0^*. Since we showed ChainGrowth$_{(t_0^* + T, s_m]}$, we have $L_{\min}(r_m + T) \geq |h_m'|$. Using the above observations with (14) and (15), we get

$$\mathcal{L}(r_m + T, s_m - T) \leq \mathcal{N}(r_m + T, s_m) \tag{16}$$

Note that there is a difference of T in the size of the two intervals above. We can close this gap as follows. Suppose that $\mathcal{L}(s_m - T, s_m) > 0$. Due to the definition of loners, we have $\mathcal{N}(s_m - T, s_m) = 0$. Then, (16) implies that

$$\mathcal{L}(r_m + T, s_m - T) \leq \mathcal{N}(r_m + T, s_m - T) \tag{17}$$

Now, we are only concerned about the case that $\mathcal{L}(s_m - T, s_m) = 0$. In this case, (16) gives

$$\mathcal{L}(r_m + T, s_m) \leq \mathcal{N}(s_m + T, s_m) \tag{18}$$

Either (17) or (18) holds, with $r_m + T < t_m^*$ and $s_m - T \geq t_m^*$. Since t_m^* is a T-pivot, this is a contradiction. □

Note that since the genesis block is a great block to begin with, Lemma 4 actually shows that the block from every T-pivot-loner is a great block. Therefore, under FrequentPivots$_{T,\gamma}$, every interval of $(\gamma - 1)T$ slots brings at least one great block.

4.3.1 Proof of Lemma 1.

PROOF. Let $T_{\text{conf}} = \gamma T$. First, we prove safety by contradiction. Suppose that $C = dC_i(t)$ for some honest node i. Suppose that $C' = dC_j(t')$ for some honest node j and $t' \geq t$ such that $dC_i(t)^{\lceil T_{\text{conf}}} \not\preceq dC_j(t')^{\lceil T_{\text{conf}}}$.

Consider all the loners $t_1, ..., t_m \in (t - \gamma T, t - T]$ with block b_j proposed in slot t_j. From the assumption that $dC_i(t)^{\lceil T_{\text{conf}}} \not\preceq dC_j(t')^{\lceil T_{\text{conf}}}$, we can see that (i) if $b_j \in dC_i(t)$, then $b_j \notin dC_j(t')$, (ii) if $b_j \in dC_j(t')$, then $b_j \notin dC_i(t)$, and (iii) otherwise, $b_j \notin dC_i(t) \cup dC_j(t')$. For each $j = 1, ..., m$, either $b_j \notin dC_i(t)$ or $b_j \notin dC_j(t')$. This means that for all $j = 1, ..., m$, b_j is not a great block. Due to Lemma 4, this also means that there are no T-pivots in the interval $(t - \gamma T, t - T]$, which is a contradiction to FrequentPivots$_{T,\gamma}$.

We next prove liveness. Assume a transaction tx is received by all honest nodes before time t. We know that there exists a pivot-loner t^* in the interval $(t, t + (\gamma - 1)T]$. The honest block b^* from t^* or its prefix must contain tx since tx is seen by all honest nodes at time $t < t^*$. Moreover, b^* is also a great block, i.e., $b^* \in dC_i(t')$ for all honest nodes i and $t' \geq t^* + T$. Therefore, tx $\in$ LOG$_i^{t'}$ for all $t' \geq t^* + T_{\text{conf}}$, which is at most $t + 2\gamma T$. □

4.4 Probabilistic Analysis

4.4.1 Proof of Lemma 3.

PROOF. $\mathcal{B}(t, t + \gamma T]$ has a Binomial$(\gamma T, p)$ distribution. By applying a Chernoff bound, we get for any given t,

$$\Pr\left[\mathcal{B}(t, t + \gamma T] > \bar{C}T\right] \leq \exp\left(-\frac{(\bar{C}T - \gamma p T)^2}{(2 + \epsilon_2)\gamma p T}\right) \leq \exp\left(-\frac{\epsilon_2^2 \gamma p T}{2 + \epsilon_2}\right).$$

By using a union bound, we derive

$$\Pr\left[\forall t\colon \mathcal{B}(t, t + \gamma T] > \bar{C}T\right] \leq \exp\left(-\frac{\epsilon_2^2 \gamma p T}{2 + \epsilon_2} + \ln(T_h)\right).$$

□

4.4.2 Proof of Lemma 2.

Definition 7 (Pivot condition). PivotCondition$_{(r,s]}$ iff

$$(\mathcal{L}(r, s] > \mathcal{N}(r, s]) \vee (\mathcal{N}(r, s] = 0). \tag{19}$$

Note that Pivot$_T(t)$ iff $\forall (r, s] \ni t$, PivotCondition$_{(r,s]}$.

Definition 8 (Weak Pivot). Time slot t satisfies WeakPivot$_{w,T}(t)$ iff

$$\forall (r, s] \ni t, s - r < w\colon \text{PivotCondition}_{(r,s]}. \tag{20}$$
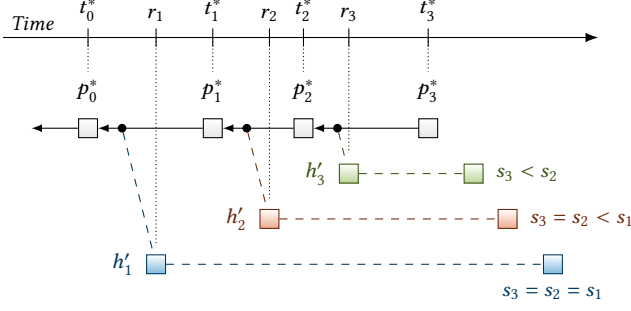
Figure 6: An example of the recursion used in the proof of Lemma 4, where $m = 3$. The great block p_0^* is proposed in slot t_0^*. The sequence $t_1^*, t_2^*, \dots$ denotes the successive pivot-loners after t_0^*. Slot s_j denotes the first slot in which an honest node's longest chain does not contain p_j^*. The block h_j' is the last honest block corresponding to a T-loner on the chain that displaces p_j^*, and that T-loner occurs in slot r_j. Since p_3^* is a descendant of p_2^*, p_3^* may be displaced earlier than p_2^* in which case $s_3 < s_2$ (green). Otherwise, p_3^* is displaced at the same time as p_2^* is displaced in which case $s_3 = s_2$. In turn, p_2^* may be displaced earlier than p_1^* so that $s_2 < s_1$ (orange), or all three pivots are displaced at once, then $s_3 = s_2 = s_1$ (blue).

Proposition 3. *If $2p_H(1 - 2pT) = p(1 + \epsilon_1)$ for some $0 < \epsilon_1 < 1$,*

$$\forall (r, s) \colon \quad \Pr\left[\neg \mathsf{PivotCondition}_{(r,s)}\right] \le \exp\left(-\alpha_1 p(s - r)\right), \quad (21)$$

with $\alpha_1 = \eta \epsilon_1^2$ for some constant η.

PROOF. By a simple Chernoff bound, for $\epsilon > 0$,

$$\Pr\left[\mathcal{B}(r, s] \ge p(s - r)(1 + \epsilon)\right] \le \exp\left(-\frac{\epsilon^2 p(s - r)}{2 + \epsilon}\right). \quad (22)$$

Let $H(r, s] \triangleq \sum_{t=r+1}^{s} \mathbb{1}\{H_t = 1, A_t = 0\}$. Again by a Chernoff bound $\forall r, s$ and $0 < \epsilon' < 1$,

$$\Pr\left[H(r, s] \le p_H(s - r)(1 - \epsilon')\right] \le \exp\left(-\frac{\epsilon'^2 p_H(s - r)}{2}\right). \quad (23)$$

Now, let $N = p_H(s - r)(1 - \epsilon')$. For $j = 1, \dots, N$, let $Y_j = 0$ if the j-th uniquely honest time slot is followed by T empty slots, and 1 otherwise. Then, the Y_j's are mutually independent random variables and $\Pr\left[Y_j = 1\right] = 1 - (1 - p)^T$. Then, for $0 < \epsilon' < 1$,

$$\Pr\left[\sum_{j=1}^{N} Y_j \ge (1 - (1 - p)^T)N + \epsilon' N\right] \le \exp\left(-\frac{\epsilon'^2 N}{3(1 - (1 - p)^T)}\right). \quad (24)$$

Similarly, let $Z_j = 0$ if the j-th uniquely honest time slot is preceded by T empty slots, and 1 otherwise. Then, the Z_j's are mutually independent random variables and $\Pr\left[Z_j = 1\right] = 1 - (1 - p)^T$. Then,

$$\Pr\left[\sum_{j=1}^{N} Z_j \ge (1 - (1 - p)^T)N + \epsilon' N\right] \le \exp\left(-\frac{\epsilon'^2 N}{3(1 - (1 - p)^T)}\right). \quad (25)$$

Then, when none of the above four bad events occur,

$$\mathcal{L}(r, s] > N - \sum_{i=1}^{N} Y_j - \sum_{i=1}^{N} Z_j$$

$$\ge N(1 - 2(1 - (1 - p)^T) - 2\epsilon')$$
$$\ge p_H(s - r)(1 - \epsilon')(1 - 2pT - 2\epsilon')$$
$$\ge p_H(1 - 2pT)(s - r)(1 - \epsilon'')$$
$$\mathcal{B}(r, s] < p(s - r)(1 + \epsilon).$$

If we choose $\frac{1 + \epsilon}{1 - \epsilon''} < 1 + \epsilon_1$, then $2p_H(1 - 2pT) = p(1 + \epsilon_1)$ implies that

$$2\mathcal{L}(r, s] > \mathcal{B}(r, s]$$
$$\implies \mathcal{L}(r, s] > \mathcal{B}(r, s] - \mathcal{L}(r, s] = \mathcal{N}(r, s]$$

except with probability upper bounded by $\exp\left(-\eta \epsilon_1^2 p(s - r)\right)$ for some constant η. □

Proposition 4. *If $2p_H(1 - 2pT) = p(1 + \epsilon_1)$, then for an execution horizon T_h and $w \ge \frac{2 \ln(T_h)}{\alpha_1 p}$,*

$$\Pr\left[\exists (r, s], s - r \ge w \colon \neg \mathsf{PivotCondition}_{(r,s)}\right]$$
$$\le \exp\left(-\alpha_1 p w + 2 \ln(T_h)\right). \quad (26)$$

PROOF. Using a union bound and Proposition 3,

$$\Pr\left[\exists (r, s], s - r \ge w \colon \neg \mathsf{PivotCondition}_{(r,s)}\right]$$
$$\le \sum_{(r,s], s - r \ge w} \Pr\left[\neg \mathsf{PivotCondition}_{(r,s)}\right]$$
$$\le T_h^2 \exp(-\alpha_1 p w)$$
$$= \exp(2 \ln(T_h) - \alpha_1 p w).$$

□

Proposition 5. *If $2p_H(1 - 2pT) = p(1 + \epsilon_1)$, then for a time horizon T_h and $w \ge \frac{2 \ln(T_h)}{\alpha_1 p}$,*

$$\Pr\left[\exists t \colon \mathsf{WeakPivot}_{w,T}(t) \wedge \neg \mathsf{Pivot}_T(t)\right]$$
$$\le \exp\left(-\alpha_1 p w + 2 \ln(T_h)\right). \quad (27)$$

PROOF. If some t is a weak pivot (with $w \ge \frac{2 \ln(T_h)}{\alpha_1 p}$) and t is not a pivot, then $\exists (r, s] \ni t$ with $s - r \ge w$ such that $\neg \mathsf{PivotCondition}_{(r,s)}$. But the probability for this is bounded accordingly by Proposition 4. □

Proposition 6. *If $2p_H(1 - 2pT) = p(1 + \epsilon_1)$, then for time horizon T_h and $w \ge 2$,*

$$\forall t \colon \quad \Pr\left[\mathsf{WeakPivot}_{w,T}(t)\right] \ge \frac{1}{2}(1 - p)^{\frac{6 \ln(w)}{\alpha_1 p}}. \quad (28)$$

PROOF. For $v \triangleq \frac{2 \ln(w) + \ln(2)}{\alpha_1 p}$, consider the events

$$E_1 \triangleq \{\mathcal{N}(t - v, t + v] = 0\}, \quad (29)$$
$$E_2 \triangleq \{\forall (r, s] \ni t, v \le s - r \le w \colon \mathsf{PivotCondition}_{(r,s)}\}. \quad (30)$$

Note that, $E_1 \cap E_2 \subseteq \{\mathsf{WeakPivot}_{w,T}(t)\}$. Further, we note that that $E_1' \subseteq E_1$ where

$$E_1' \triangleq \{\mathcal{B}(t - v, t + v] = 0\}, \quad \Pr[E_1] \ge \Pr\left[E_1'\right] = (1 - p)^{2v}.$$

Now, using a union bound and Proposition 3 as in the proof of Proposition 4,

$$\Pr\left[\neg E_2\right] \le w^2 \exp(-\alpha_1 p v) = \frac{1}{2}. \quad (31)$$

Given the choice of v, and that $\Pr[E_2 \mid E_1] \ge \Pr[E_2]$,

$$\Pr\left[\mathsf{WeakPivot}_{w,T}(t)\right] \ge \Pr[E_1 \cap E_2]$$

10

$$\geq \Pr[E_1]\Pr[E_2]$$
$$\geq \frac{1}{2}(1-p)^{2v}$$
$$= \frac{1}{2}(1-p)^{\frac{4\ln(w)+2\ln(2)}{\alpha_1 p}}$$
$$\geq \frac{1}{2}(1-p)^{\frac{6\ln(w)}{\alpha_1 p}}.$$

$\square$

Proposition 7. *For a time horizon T_h and any w,*

$$\forall t: \Pr\left[\exists t' \in (t, t+w]: \mathsf{Loner}_T(t')\right] \geq 1 - \exp\left(-\frac{wp_\mathrm{L}}{2T}\right). \quad (32)$$

Proof. Let $k = \frac{w}{2T}$ and $t_j = t + (2j+1)T$ for $j = 0, ..., k-1$. The events $\mathsf{Loner}_T(t_j)$ are mutually independent. Denote

$$E_j \triangleq \{\mathsf{Loner}_T(t_j)\} \qquad \text{for } j = 0, ..., (k-1) \quad (33)$$
$$E(t, t+w] \triangleq \{\exists t' \in (t, t+w]: \mathsf{Loner}_T(t')\}. \quad (34)$$

We have $E(t, t+w] \supseteq \bigcup_{j=0}^{k-1} E_j$. Then

$$\Pr\left[\neg E(t, t+w]\right] \leq \Pr\left[\bigcap_{j=0}^{k-1} \neg E_j\right]$$
$$\leq \prod_{j=0}^{k-1} \Pr\left[\neg E_j\right]$$
$$\leq \left(1 - p_\mathrm{H}(1-p)^{2T}\right)^k$$
$$\leq \exp\left(-\frac{w}{2T}p_\mathrm{H}(1-p)^{2T}\right)$$
$$\leq \exp\left(-\frac{wp_\mathrm{L}}{2T}\right).$$

$\square$

Proposition 8. *If $2p_\mathrm{H}(1 - 2pT) = p(1 + \epsilon_1)$, then for horizon T_h and any w,*

$$\forall t: \Pr\left[\exists t' \in (t, t+(\gamma-1)T]: \mathsf{WeakPivot}_{w,T}(t') \wedge \mathsf{Loner}_T(t')\right]$$
$$\geq 1 - \exp(-\alpha_3(\gamma-1)T) - \exp\left(-\frac{wp_\mathrm{L}}{2T}\right), \quad (35)$$

with $\alpha_3 \triangleq \frac{1}{4}w^{-1-6/\alpha_1} \geq 0$.

Proof. Let k be the largest integer such that $(\gamma-1)T \geq 2wk$. For $i = 0, ..., (k-1)$, define

$$E_i \triangleq \{t + (2i+1)w \text{ is } w\text{-pivot}\} \quad (36)$$
$$E_i' \triangleq \{\exists t' \in [t + (2i+1)w, t + (2i+2)w]: \mathsf{Loner}_T(t')\} \quad (37)$$
$$E \triangleq \{\exists t' \in (t, t+(\gamma-1)T]: \mathsf{WeakPivot}_{w,T}(t')\}. \quad (38)$$

Let $t_i = t + (2i-1)w$, and suppose that E_i occurs for some i. If t_i is also a loner, then E_i' occurs as well. If t_i is not a T-loner, let t' be the first slot such that $t' > t_i$ and $H_t + A_t > 0$. Then, t' is a T-loner because otherwise we would have $\mathcal{N}(t_i - 1, t'] \geq \mathcal{L}(t_i - 1, t']$ which is a contradiction. Then, we show that t' is also a weak pivot. For every $(r, s] \ni t_i, t'$ such that $s - r \leq w$, we know PivotCondition because t_i is a weak pivot. However for all $(r, s] \ni t'$ such that $(r, s] \not\ni t_i$ and $s - r \leq w$, we also have PivotCondition because $\mathcal{N}(t_i, t'] = 0$.

Thus, we have $\bigcup_{i=0}^{k-1}(E_i \cap E_i') \subseteq E$, and by construction $\mathcal{E}_i$ are independent. Hence,

$$\Pr[E] \geq \Pr\left[\bigcup_{i=0}^{k-1}(E_i \cap E_i')\right] = 1 - \Pr\left[\bigcap_{i=0}^{k-1}(\neg E_i \cup \neg E_i')\right]$$
$$= 1 - \Pr\left[\left(\bigcap_{i=0}^{k-1}\neg E_i\right) \cup \left(\bigcap_{i=0}^{k-1}\neg E_i'\right)\right]$$
$$\geq 1 - \Pr\left[\bigcap_{i=0}^{k-1}\neg E_i\right] - \Pr\left[\bigcap_{i=0}^{k-1}\neg E_i'\right]$$
$$\geq 1 - \left(1 - \frac{1}{2}(1-p)^{\frac{6\ln(w)}{\alpha_1 p}}\right)^k - \exp\left(-\frac{wp_\mathrm{L}}{2T}\right)$$
$$\geq 1 - \left(1 - \frac{1}{2}e^{-\frac{6\ln(w)}{\alpha_1}}\right)^k - \exp\left(-\frac{wp_\mathrm{L}}{2T}\right)$$
$$\geq 1 - \exp(-\alpha_3(\gamma-1)T) - \exp\left(-\frac{wp_\mathrm{L}}{2T}\right), \quad (39)$$

where we have used Proposition 6. $\square$

Proposition 9. *If $2p_\mathrm{H}(1 - 2pT) = p(1 + \epsilon_1)$, then for horizon T_h and $w \geq \max\left\{\frac{2\ln(T_\mathrm{h})}{\alpha_1 p}, \frac{2T\ln(T_\mathrm{h})}{p_\mathrm{L}}, 2\right\}$,*

$$\Pr\left[\forall t: \exists t' \in (t, t+(\gamma-1)T]: \mathsf{WeakPivot}_{w,T}(t') \wedge \mathsf{Loner}_T(t')\right]$$
$$\geq 1 - \exp(-\alpha_3(\gamma-1)T + \ln(T_\mathrm{h})) - \exp\left(-\frac{wp_\mathrm{L}}{2T} + \ln(T_\mathrm{h})\right). \quad (40)$$

Proof. By a union bound over all T_h possible time slots, and using Proposition 8. $\square$

Proof of Lemma 2. Finally, to prove Lemma 2, let

$$E_1 \triangleq \{\forall t: \exists t' \in (t, t+(\gamma-1)T]: \mathsf{WeakPivot}_{w,T}(t') \wedge \mathsf{Loner}_T(t')\}$$
$$E_2 \triangleq \{\forall t: \mathsf{WeakPivot}_{w,T}(t) \Rightarrow \mathsf{Pivot}_T(t)\}$$
$$E \triangleq \{\forall t: \exists t' \in (t, t+(\gamma-1)T]: \mathsf{Pivot}_T(t') \wedge \mathsf{Loner}_T(t')\}.$$

Note that $E_1 \cap E_2 \subseteq E$. Then we apply a union bound on the probabilities from Propositions 9 and 5. $\square$

ACKNOWLEDGMENT

REFERENCES

[1] 2020. *Bitcoin Developer Guide – P2P Network – Initial Block Download – Headers-First.* https://developer.bitcoin.org/devguide/p2p_network.html#headers-first
[2] Maria Apostolaki, Aviv Zohar, and Laurent Vanbever. 2017. Hijacking Bitcoin: Routing Attacks on Cryptocurrencies. In *2017 IEEE Symposium on Security and Privacy (SP)*. 375–392. https://doi.org/10.1109/SP.2017.29
[3] Christian Badertscher, Peter Gaži, Aggelos Kiayias, Alexander Russell, and Vassilis Zikas. 2018. Ouroboros Genesis: Composable proof-of-stake blockchains with dynamic availability. In *Conference on Computer and Communications Security (CCS '18)*. ACM, 913–930.
[4] Vivek Bagaria, Amir Dembo, Sreeram Kannan, Sewoong Oh, David Tse, Pramod Viswanath, Xuechao Wang, and Ofer Zeitouni. 2019. Proof-of-Stake Longest Chain Protocols: Security vs Predictability. *arXiv preprint arXiv:1910.02218* (2019).
[5] Vivek Bagaria, Sreeram Kannan, David Tse, Giulia Fanti, and Pramod Viswanath. 2019. Prism: Deconstructing the Blockchain to Approach Physical Limits. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*. ACM, 585–602.
[6] Tong Cao, Jiangshan Yu, Jérémie Decouchant, and Paulo Esteves-Verissimo. 2018. Revisiting Network-Level Attacks on Blockchain Network. (2018). https://orbilu.uni.lu/bitstream/10993/38142/1/bcrb18-cao.pdf

[7] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. 2003. Secure Routing for Structured Peer-to-Peer Overlay Networks. *SIGOPS Oper. Syst. Rev.* 36, SI (Dec. 2003), 299–314. https://doi.org/10.1145/844128.844156

[8] Benjamin Y. Chan and Elaine Shi. 2020. Streamlet: Textbook Streamlined Blockchains. In *Advances in Financial Technologies (AFT '20)*. ACM, 1–11.

[9] Duncan Coutts, Neil David, Marcin Szamotulski, and Peter Thompson. 2020. *Introduction to the design of the Data Diffusion and Networking for Cardano Shelley*. Technical Report. IOHK. Version 1.9.

[10] Duncan Coutts, Alex Vieth, Neil Davies, Marcin Szamotulski, Karl Knutsson, Marc Fontaine, and Armando Santos. 2021. *The Shelley Networking Protocol*. Technical Report. IOHK. Version 1.2.0, Revision 49.

[11] Phil Daian, Rafael Pass, and Elaine Shi. 2019. Snow White: Robustly Reconfigurable Consensus and Applications to Provably Secure Proof of Stake. In *Financial Cryptography and Data Security (FC '19)*. Springer, 23–41.

[12] Bernardo David, Peter Gazi, Aggelos Kiayias, and Alexander Russell. 2018. Ouroboros Praos: An Adaptively-Secure, Semi-synchronous Proof-of-Stake Blockchain. In *EUROCRYPT 2018*. Springer, 66–98.

[13] Edsko de Vries, Thomas Winant, and Duncan Coutts. 2020. *The Cardano Consensus and Storage Layer*. https://github.com/input-output-hk/ouroboros-network/tree/master/ouroboros-consensus/docs/report

[14] Christian Decker and Roger Wattenhofer. 2013. Information propagation in the Bitcoin network. In *P2P*. IEEE, 1–10.

[15] Amir Dembo, Sreeram Kannan, Ertem Nusret Tas, David Tse, Pramod Viswanath, Xuechao Wang, and Ofer Zeitouni. 2020. Everything is a Race and Nakamoto Always Wins. In *Conference on Computer and Communications Security (CCS '20)*. ACM, 859–878.

[16] Cynthia Dwork and Moni Naor. 1992. Pricing via Processing or Combatting Junk Mail. In *CRYPTO (Lecture Notes in Computer Science, Vol. 740)*. Springer, 139–147.

[17] Matthias Fitzi, Peter Gaži, Aggelos Kiayias, and Alexander Russell. 2018. Parallel Chains: Improving Throughput and Latency of Blockchain Protocols via Parallel Composition. Cryptology ePrint Archive, Report 1119.

[18] Matthias Fitzi, Peter Gaži, Aggelos Kiayias, and Alexander Russell. 2020. Proof-of-Stake Blockchain Protocols with Near-Optimal Throughput. Cryptology ePrint Archive, Report 2020/037. https://eprint.iacr.org/2020/037.

[19] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. 2015. The Bitcoin backbone protocol: Analysis and applications. In *EUROCRYPT 2015*. Springer, 281–310.

[20] Peter Gaži, Aggelos Kiayias, and Alexander Russell. 2020. Tight Consistency Bounds for Bitcoin. (2020). https://eprint.iacr.org/2020/661.

[21] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. 2015. Eclipse Attacks on Bitcoin's Peer-to-Peer Network. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, Washington, D.C., 129–144. https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/heilman

[22] IOHK. 2020. *input-output-hk/ouroboros-network*. https://github.com/input-output-hk/ouroboros-network/blob/master/ouroboros-network/src/Ouroboros/Network

[23] IOHK. 2020. *input-output-hk/ouroboros-network*. https://github.com/input-output-hk/ouroboros-network/blob/master/ouroboros-network/src/Ouroboros/Network/BlockFetch/Decision.hs#L162

[24] IOHK. 2021. *input-output-hk/ouroboros-network*. https://github.com/input-output-hk/ouroboros-network/blob/master/ouroboros-consensus-shelley/src/Ouroboros/Consensus/Shelley/Protocol.hs#L281

[25] Markus Jakobsson and Ari Juels. 1999. Proofs of Work and Bread Pudding Protocols. In *Communications and Multimedia Security (IFIP Conference Proceedings, Vol. 152)*. Kluwer, 258–272.

[26] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. 2017. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *CRYPTO 2017*. Springer, 357–388.

[27] Michael McSweeney. 2021. *Solana experiences transaction stoppage as developers report 'intermittent instability'*. Retrieved 2021-11-21 from https://www.theblockcrypto.com/linked/117624/solana-experiences-transaction-stoppage-as-developers-report-intermittent-instability

[28] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf.

[29] Ravi Netravali, Anirudh Sivaraman, Somak Das, Ameesh Goyal, Keith Winstein, James Mickens, and Hari Balakrishnan. 2015. Mahimahi: Accurate record-and-replay for HTTP. In *2015 USENIX Annual Technical Conference (USENIX ATC 15)*. 417–429.

[30] R Pass, L Seeman, and A Shelat. 2017. Analysis of the blockchain protocol in asynchronous networks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*.

[31] Rafael Pass and Elaine Shi. 2017. The Sleepy Model of Consensus. In *ASIACRYPT 2017*. Springer, 380–409.

[32] Ling Ren. 2019. Analysis of Nakamoto Consensus. *IACR Cryptol. ePrint Arch.* (2019), 943.

[33] Atul Singh, Miguel Castro, Peter Druschel, and Antony Rowstron. 2004. Defending against Eclipse Attacks on Overlay Networks. In *Proceedings of the 11th Workshop on ACM SIGOPS European Workshop* (Leuven, Belgium) *(EW 11)*. Association for Computing Machinery, New York, NY, USA, 21–es. https://doi.org/10.1145/1133572.1133613

[34] Atul Singh, Tsuen-Wan Ngan, Peter Druschel, and Dan S. Wallach. 2006. Eclipse Attacks on Overlay Networks: Threats and Defenses. In *INFOCOM*. IEEE.

[35] Emil Sit and Robert Morris. 2002. Security Considerations for Peer-to-Peer Distributed Hash Tables. In *Peer-to-Peer Systems*, Peter Druschel, Frans Kaashoek, and Antony Rowstron (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 261–269.

[36] Yonatan Sompolinsky and Aviv Zohar. 2015. Secure high-rate transaction processing in bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 507–527.

[37] Anatoly Yakovenko. 2018. Solana: A new architecture for a high performance blockchain v0.8.13. Retrieved 2021-11-21 from https://solana.com/solana-whitepaper.pdf

[38] Haifeng Yu, Ivica Nikolić, Ruomu Hou, and Prateek Saxena. 2020. OHIE: Blockchain Scaling Made Simple. In *2020 IEEE Symposium on Security and Privacy (SP)*. 90–105. https://doi.org/10.1109/SP40000.2020.00008

A HELPER FUNCTIONS FOR ALGORITHMS 1 AND 2

- Hash(txs): Cryptographic hash function to produce a binding commitment to txs (modelled as a random oracle)
- $C' \preceq C$: Relation describing that C' is a prefix of C
- $C \| C'$: Concatenation of C and C'
- prefixChainsOf(C): Set of prefixes of C
- longestChain($\mathcal{T}$): Determine longest chain among set $\mathcal{T}$ of chains. Ties are broken by the adversary.
- txsAreSemanticallyValidWrtPrefixesOf(C, txs): Verifies for each transaction in txs that the transaction is semantically valid with respect to and properly authorized by the owner of the underlying assets as determined by the transaction's prefix in the ledger resulting from appending txs to the transactions as ordered in C (assumes that content of all blocks in C is known to the node)
- newBlock(time $= t$, party $= P$, txsHash $=$ Hash(txs)): Produces a new block header with the given parameters
- $\mathcal{Z}$.broadcastHeaderChain(C): Broadcasts header chain C via $\mathcal{Z}$ to other nodes
- $\mathcal{Z}$.uploadContent(C, txs): Uploads content txs for the block identified by chain C into the block content repository of $\mathcal{Z}$ ($\mathcal{Z}$ only stores the content txs if its hash matches the transaction hash in C)
- $\mathcal{Z}$.requestContent(C): Requests content associated with block identified by chain C from the repository of $\mathcal{Z}$
- $\mathcal{Z}$.receivePendingTxsSemanticallyValidWrt(C): Retrieves a set of pending transactions that are not included in but semantically valid (see above) with respect to C
- $\mathcal{Z}$.outputLedger(C): Declares C as the node's ledger to the $\mathcal{Z}$ (this constitutes LOG_i^t for which consistency and liveness are required for a secure consensus protocol)

B EXPERIMENTAL SETUP DETAILS FOR FIGURE 1

For the experiment demonstrating network congestion, we start 17 Cardano nodes in 17 AWS data centers across the globe and connect them into a fully-connected graph. We point out that the Cardano block fetch logic includes an optimization to only download blocks that have larger heights than the locally-adopted longest chain. As a result, a node may not eventually download every block whose header it sees. To demonstrate network congestion in the absence

of a suitable download rule, we modify the code to disable this optimization and ensure that every node eventually downloads all blocks. To show congestion, we configure a variable number (N) of nodes to mine blocks at the same time, and measure the time for all 17 nodes to download all N blocks.

C EXPERIMENTAL SETUP DETAILS FOR FIGURE 2

To demonstrate the attack, we implement our PoS LC node that includes a replica of Cardano's chain sync, block fetch, and fetch prioritization logic as per [22] in 500 lines of Golang code. We implement all essential features of Cardano's network stack, including protocol multiplexing that removes head-of-line blocking.

We set up one PoS LC node as the victim of the attack, and connect it to one honest node and three adversarial nodes. The victim node does not mine blocks. The honest node represents the mining power of the whole network, and produces blocks at 0.2 block/s. The adversarial nodes forge an unlimited number of spam blocks by equivocating, and advertise the spam blocks to the victim through the chain sync protocol. Each honest and adversarial block is 1 MB. To simulate a limited network condition, we use Mahimahi [29] to impose an ingress capacity of 4 Mbps at the victim node, and a one-way propagation delay of 50 ms to and from the victim node.

D PROOF OF THEOREM 1 CONTD.

We will now see that $\tau = \frac{c_3}{C}\kappa'^{2+6/\alpha_1}$ and $T_{\text{conf}} = c_4\kappa'^{2+6/\alpha_1}$ are optimal in their polynomial degree. Given T and ϵ_1, we set $\rho T = \frac{1-2\beta-\epsilon_1}{4(1-\beta)}$. As in the main proof, we set

$$w = c_1'\frac{T\kappa'}{\rho} = c_1''\frac{T^2\kappa'}{1-2\beta-\epsilon_1}. \tag{41}$$

Let $v = 6/\alpha_1 = 6/\eta\epsilon_1^2$. Subsequently, we set

$$\gamma \approx c_2'\frac{\kappa'}{T}w^{1+v} = c_2''\frac{\kappa'^{2+v}T^{1+2v}}{(1-2\beta-\epsilon_1)^{1+v}}. \tag{42}$$

Then,

$$\tau = \frac{\bar{C}}{C} = \rho\gamma(1+\epsilon_2) \approx c_3'\frac{\kappa'^{2+v}T^{2v}}{(1-2\beta-\epsilon_1)^v} \tag{43}$$

and

$$T_{\text{conf}} = \gamma T = c_4'\frac{\kappa'^{2+v}T^{2+2v}}{(1-2\beta-\epsilon_1)^{1+v}}. \tag{44}$$

Clearly, τ and T_{conf} increase with T. Therefore, for a fixed ϵ_1, these are minimized by setting $T = 1$, and this shows the optimality of the order of the polynomial in terms of ϵ_1 (or v).

Finally, we can also find the optimal setting for ϵ_1. By choosing a larger ϵ_1 close to $1-2\beta$, v is minimized, minimizing the order of the polynomial, $2+v$. We choose $\epsilon_1 = 1-2\beta-\Theta(\kappa'^{-1})$. Note that choosing ϵ_1 larger than this does not change the order of the polynomial significantly but decreases the denominator in (43) and (44). On the other hand, choosing $\epsilon_1 = 1-2\beta-\Theta(1)$ would significantly increase the polynomial degree while increasing the denominator only by a constant factor. By setting $\epsilon_1 = 1-2\beta-\Theta(\kappa'^{-1})$, we get $\rho = \Theta(\kappa'^{-1})$. Denoting the minimum value of v as $v^* = \frac{6}{\eta(1-2\beta)^2}$, $\tau = \Theta\left(\kappa'^{2+2v^*}\right)$ and $T_{\text{conf}} = \Theta\left(\kappa'^{3+2v^*}\right)$.