Accountable Liveness

Andrew Lewis-Pye
London School of Economics
a.lewis7@lse.ac.uk

Joachim Neu
a16z Crypto Research
jneu@a16z.com

Tim Roughgarden
Columbia University & a16z Crypto Research
tim.roughgarden@gmail.com

Luca Zanolini
Ethereum Foundation
luca.zanolini@ethereum.org

Abstract

Safety and liveness are the two classical security properties of consensus protocols. Recent works have strengthened safety with *accountability*: should any safety violation occur, a sizable fraction of adversary nodes can be proven to be protocol violators. This paper studies to what extent analogous accountability guarantees are achievable for *liveness*. To reveal the full complexity of this question, we introduce an interpolation between the classical synchronous and partially-synchronous models that we call the *x-partially-synchronous network model* in which, intuitively, at most an x fraction of the time steps in any sufficiently long interval are asynchronous (and, as with a partially-synchronous network, all time steps are synchronous following the passage of an unknown "global stablization time"). We prove a precise characterization of the parameter regime in which accountable liveness is achievable: if and only if $x < 1/2$ and $f < n/2$, where n denotes the number of nodes and f the number of nodes controlled by an adversary. We further refine the problem statement and our analysis by parameterizing by the number of violating nodes identified following a liveness violation, and provide evidence that the guarantees achieved by our protocol are near-optimal (as a function of x and f). Our results provide rigorous foundations for liveness-accountability heuristics such as the "inactivity leaks" employed in Ethereum.

1 Introduction

The atomic broadcast variant of Byzantine-fault tolerant (BFT) consensus is a fundamental problem in distributed computing, where a set of *n nodes* must agree on a total ordering of input *transactions* into an output *log*. This problem has received renewed attention recently in the context of cryptocurrencies and blockchains because it is a fundamental primitive for such systems. Specifically, each node in the protocol successively produces an output log of transactions, and the protocol must guarantee the two key security properties: *safety*—ensuring that logs remain consistent across nodes and across time—and *liveness*—ensuring that every input transaction is eventually included in the logs of nodes. Despite some *f adversary* nodes acting arbitrarily (subject to being computationally bounded) in an effort to undermine consensus, these properties should hold for all non-adversary (a.k.a. *honest*) nodes. The typical guarantees for classical protocols state that if the fraction of adversary nodes is below a given threshold, the protocol remains *safe* and *live*.

However, if the fraction of adversary nodes exceeds this threshold, the protocol's security properties no longer hold. What then? If there is a security violation, can we at least determine which nodes caused it? In data-center applications of consensus, this is useful for

identifying and remedying faulty machines; and it becomes even more important in proof-of-stake blockchain applications of consensus, where self-interested nodes may deviate from the protocol for financial gain and may be willing to accept consensus security violations as collateral damage. Here, identifying such adversary nodes allows the system to confiscate their stake as a form of punishment, thereby incentivizing honest behavior, and to compensate damages from the violations. Such aims are often referred to as *crypto-economic security*.

Towards this goal, the literature has recently introduced the notion of *accountable safety* [14, 45], which is a strengthening [37, 38] of safety that stipulates: if any two nodes at any two points in time ever have inconsistent output logs, *i.e.*, a safety violation occurs, then a substantial fraction of nodes can be identified as having provably violated the protocol. Specifically, if there is ever a safety violation between two honest nodes, then from their respective transcripts of the protocol's execution, a *certificate of guilt* can be extracted for a set of nodes such that (a) no honest node is ever falsely accused of having contributed to the safety violation, and (b) the set of identified guilty nodes is guaranteed to be fairly large.

1.1 Scope & Contributions

This paper initiates the systematic study of *accountability for liveness* (see Secs. E and 7 for a discussion of related work). More precisely, we investigate under what circumstances, and through which protocol techniques, the following guarantee can be achieved: should a liveness violation occur, then a certificate of guilt is eventually produced such that (a) no honest node is ever falsely accused, and (b) the set of identified (guilty) nodes is guaranteed to be fairly large. Note that, technically, traditional (*eventual*) liveness is such a weak guarantee that violations cannot be assessed by any one point in the execution [4] (as pending transactions may still get confirmed in the future). We thus consider a slightly stronger (*timely-*)liveness notion where transactions have to be confirmed within a deadline.

Intuitively, because liveness violations would seem to typically involve the unexpected *absence* of messages (*e.g.*, votes) rather than the unexpected *presence* of messages (*e.g.*, double-voting to cause a safety violation), one might speculate that certificates of guilt ("proofs of misbehavior") should be more difficult to guarantee for liveness violations than for safety violations. We prove that accountable liveness is indeed harder to achieve than accountable safety, in two senses.

First, accountable safety is achievable in partial synchrony; indeed, any partially-synchronous protocol can be augmented to guarantee accountable safety without imposing any additional timing assumptions [20, 39]. (This statement assumes a computationally

bounded adversary and secure digital signatures, assumptions we also make here.) In contrast, we prove that accountable liveness *cannot* be attained in a partially-synchronous network without additional timing assumptions (at least as long as the protocol is required to preserve safety under partial synchrony if all nodes are honest). On the other hand, accountable liveness is vacuous in a (fully-)synchronous network, where (at least in some variants [13, 23, 30, 31, 47]) liveness can be guaranteed even if almost all nodes are adversarial. To characterize the possibility–impossibility frontier, we introduce the *x-partially-synchronous* network model in which, over any sufficiently long time interval, the network is asynchronous for at most an x fraction of the time. This model generalizes both synchronous networks (when $x = 0$) and partially-synchronous networks (when $x = 1$), and may be of independent interest in the context of recent work on fine-grained network models (*e.g.*, [26]). Our goal is thus to design consensus protocols that satisfy standard security properties (safety and liveness) under partial synchrony but, *additionally*, achieve accountable liveness in the *x-partially-synchronous* model (for x as large as possible).

Second, accountable safety is achievable without any assumptions on the number of adversarial nodes—even if all but two of the nodes are adversarial, the honest nodes' transcripts will provide a certificate of guilt following any safety violation.[1] By contrast, we show that no protocol offering standard guarantees in partial synchrony (as described above) can also provide accountable liveness in the case that there is an adversarial majority (even under synchrony). The interesting parameter regime is therefore an adversary that controls at least one-third of the nodes (as otherwise, standard protocols cannot suffer liveness violations and accountability is irrelevant) and less than one-half of the nodes. Our protocols guarantee accountable liveness in this parameter regime (in *x*-partial-synchrony with $x < 1/2$).

1.2 Results

We study atomic broadcast consensus protocols that, roughly speaking (see Sec. 2.4 for a formal statement), guarantee two sets of requirements:

(1) As a *baseline*, the protocol should satisfy the standard security and performance properties required of protocols for partial synchrony. Specifically, it should be safe and live up to $\lfloor (n-1)/3 \rfloor$ adversary nodes, with expected confirmation latency during synchrony on the order of the network delay bound Δ that is guaranteed to hold after the adversary-chosen "global stabilization time" (GST) of the partially-synchronous model.

(2) *Additionally*, for specified parameters x and $\tau_{\max}^{\mathrm{AL}}$, if the network happens to be *x*-partially-synchronous and there happen to be $f \leq \tau_{\max}^{\mathrm{AL}}$ adversary nodes, then the protocol should be accountably live, *i.e.*, produce certificates of guilt for "many" adversary nodes whenever a transaction is not confirmed in a timely fashion.

On the positive, achievability side, in Secs. 3 to 5, we present an accountably live protocol. Specifically, this protocol achieves:

Theorem (Informal version of Thm. 1). *For any $x < 1/2$ and $n/3 < \tau_{\max}^{\mathrm{AL}} < n/2$, the protocol of Algs. 1, 3 and 4 is accountably live when run in x-partial-synchrony with $f \leq \tau_{\max}^{\mathrm{AL}}$ adversary nodes, and identifies arbitrarily close to $\tau_{\max}^{\mathrm{AL}} - \left\lfloor \frac{(1+x)(\tau_{\max}^{\mathrm{AL}} - n/3)}{1-x} \right\rfloor$ adversary nodes when there is a liveness violation.*

In addition, the protocol of Algs. 1, 3 and 4 satisfies the aforementioned point (1), namely the standard security and performance properties required of protocols for partial synchrony (cf. Lems. 1 and 2). We use the Tendermint protocol [11] as the starting point for our construction, but our techniques also readily apply to other PBFT-style protocols like HotStuff [51] or Streamlet [18].

On the negative, impossibility side, we show in Sec. 6 that the restrictions on x and $\tau_{\max}^{\mathrm{AL}}$ in the aforementioned theorem are fundamental for accountable liveness:

Theorem (Informal version of Thm. 2). *No protocol can simultaneously provide safety under partial synchrony even when $f = 0$, and accountable liveness with $\tau_{\max}^{\mathrm{AL}} \geq n/2$ under synchrony.*

Theorem (Informal version of Thm. 3 and Cor. 1). *No optimally-resilient protocol can be accountably live under x-partial-synchrony for $x \geq 1/2$ (and thus also not under partial synchrony).*

Here, "optimally-resilient" means that the numbers of adversary nodes the protocol can tolerate while remaining safe or live, respectively, are maximal.[2]

Finally, one can ask how good the protocol of Secs. 3 to 5 is, in terms of the number of adversary nodes it can identify. We show for classical PBFT-style protocols (including but not limited to PBFT [17], Tendermint [11], HotStuff [51], CasperFFG [14, 21], and Streamlet [18]) an upper bound on the number of adversary nodes that can be identified, and we conjecture this bound to hold for all (also non-PBFT-style) protocols:

Theorem (Informal version of Thms. 4 and 5 and Conj. 1). *For every $k \geq 3$: Classical PBFT-style protocols (conjecture: all protocols!) that remain safe and live under $\lfloor (n-1)/3 \rfloor$ adversary nodes in partial synchrony, cannot be accountably live under (Δ, g, x)-partial-synchrony for $x \geq 1/k$ and guarantee to identify $n/3 - \left\lfloor \frac{\tau_{\max}^{\mathrm{AL}} - n/3}{k-2} \right\rfloor$ (or more) adversary nodes when liveness is violated.*

Fig. 1 plots the number of adversary nodes the scheme of Secs. 3 to 5 achieves to identify in case of a liveness violation (——), and the aforementioned upper bound (··········), for varying $x < 1/2$ and $n/3 < \tau_{\max}^{\mathrm{AL}} < n/2$. (Recall that $\tau_{\max}^{\mathrm{AL}} \leq n/3$ is uninteresting because, by goal (1), the protocol is then guaranteed to be live, so accountability is trivial.) Fig. 1 shows that our impossibility result matches our achievability result closely. Remarkably, they are tight for $x = 1/3$.

1.3 Discussion

Comparison to Synchronous Atomic Broadcast Protocols. The definition of *x*-partial-synchrony for $x < 1$ and with respect to a delay bound Δ implies synchrony with respect to a delay bound

---

[1]We note that [45] establishes lower bounds on the size of the adversary for which accountable safety can be provided for certain protocols. Nevertheless, appropriately designed protocols can provide accountability even if all but two nodes are adversarial (see [12, 20, 38, 39], for example).

[2]We focus on optimally-resilient protocols since these are the most relevant, and because this also simplifies our analysis by ruling out certain complexities. For example, if we were not to focus on optimally-resilient protocols, a protocol might claim resilience that is sub-optimal and then appear to achieve non-trivial accountability simply by actually ruling out liveness violations beyond the claimed resilience.
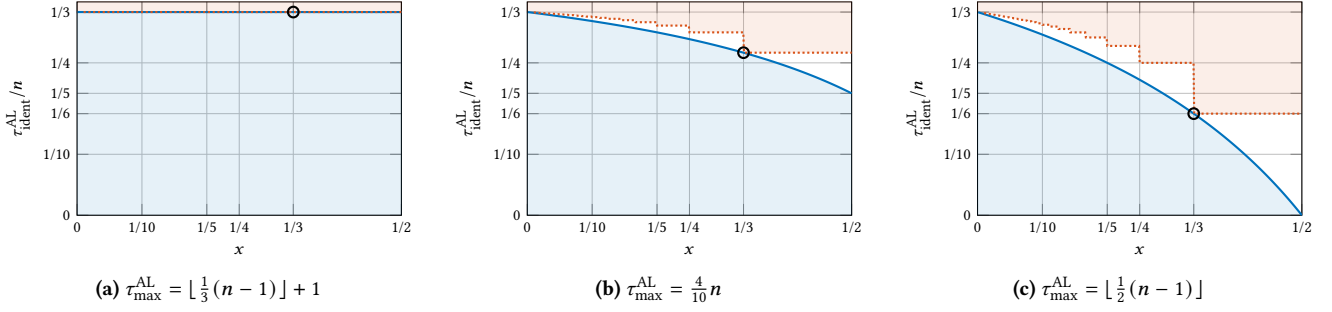
(a) $\tau_{\max}^{\mathrm{AL}} = \lfloor \frac{1}{3}(n-1) \rfloor + 1$ **(b)** $\tau_{\max}^{\mathrm{AL}} = \frac{4}{10}n$ **(c)** $\tau_{\max}^{\mathrm{AL}} = \lfloor \frac{1}{2}(n-1) \rfloor$

Figure 1: Illustration of key results: Impossibility of accountable liveness for $\tau_{\max}^{\mathrm{AL}} \geq \frac{1}{2}n$, provided in Thm. 2, and for $x \geq 1/2$, provided in Thm. 3, determine choices of $\tau_{\max}^{\mathrm{AL}}$ and range of x plotted. Plots show fraction of identified adversary nodes ($\tau_{\mathrm{ident}}^{\mathrm{AL}}/n$) achieved (——) by the scheme of Secs. 3 to 5 (Thm. 1 with $\delta_x \approx 0$), and the $\tau_{\mathrm{ident}}^{\mathrm{AL}}/n$ upper bound (········) of Thms. 4 and 5. Achievability and impossibility are tight (◯) for $x = 1/3$.

Δ^* that can be arbitrarily larger than Δ; intuitively, Δ^* scales with the "sufficiently large interval length" under which the x-partially-synchronous condition is guaranteed to kick in. (See Sec. 2.5 for a more detailed discussion.) In principle, therefore, one could employ a *synchronous* atomic broadcast protocol, such as a variant of the Dolev–Strong protocol [13, 23, 30, 31, 46, 47], with delay bound Δ^*. However, such a protocol would always incur large confirmation latency (scaling with Δ^*), and thus would not achieve our requirement of expected confirmation latency scaling with Δ when the network becomes synchronous. Such a protocol would also fail our requirement that it should be safe and live up to $\lfloor (n-1)/3 \rfloor$ adversary nodes in partial synchrony.

Add-On Features. Known techniques, similar to those used for accountable safety, can be used to handle the reconfiguration process needed to remove adversary nodes once identified for violating liveness guarantees—such as manual intervention or externally triggered reconfiguration procedures [48]. A more automated strategy, benefiting from the large implied synchrony bound Δ^* mentioned above, is described in [34], where the authors propose a wrapper that runs an accountably-safe state-machine replication protocol until a safety violation is detected. Upon detecting such a violation, the wrapper initiates a recovery procedure—requiring message delays to remain bounded by Δ^* during this recovery—to use certificates of guilt to reliably identify misbehaving nodes, remove them safely, and then restart the protocol without these adversarial nodes. The same recovery procedure can be used with the certificates of guilt produced by liveness accountability in this work.

1.4 Technical Overview

Once the model is formally defined in Sec. 2, our first task towards establishing Thm. 1 is to describe a modification of Tendermint consensus that allows nodes to *blame* others for a lack of liveness. Recall that an execution of Tendermint is partitioned into *views*. Any node that does not see progress in a particular view will now blame a number of other nodes for lack of progress in that view. (Δ, g, x)-partial-synchrony ensures that (over a sufficiently long window) at least a $(1-x)$ fraction of views will be synchronous (*i.e.*, message delivery will be reliable in those views), and for synchronous views we aim to ensure that the outcome of the blaming process has

certain useful properties. Specifically, the process of blaming other nodes is designed so that:

(i) No honest node is blamed by any honest node in any synchronous view. Since the honest nodes are a majority, this means that no honest node is "majority blamed" in any synchronous view.

(ii) In each synchronous view with an honest leader that does not confirm new transactions, there are at least $n/3$ adversary nodes that are blamed by all honest nodes (and so which are "majority blamed" in that view).

A weakness of (ii) above is that an adversarial leader may prevent progress in a synchronous view, removing the need for $n/3$ adversarial nodes to be majority blamed. For this reason, in Sec. 4.2 we introduce the use of *super-views*: each super-view consists of a number of views, with the number chosen so that (with random leader selection) each super-view is likely to have at least one honest leader. We can then consider an analogous notion of blame for super-views, which satisfies a corresponding version of (i) above, and which satisfies a version of (ii) with the weaker requirement that the super-view should have at least one honest leader.

Given this setup, we must then extract an "adjudication rule" for identifying guilty parties in the event of an attack on liveness. To demonstrate the basic idea behind the adjudication rule, consider (for now) a simplified setup in which we suppose all super-views have at least one honest leader. Suppose we see a sequence of super-views $\mathcal{U}$, all of which fail to make progress with liveness, and that we know a majority of super-views in $\mathcal{U}$ are synchronous (the number of super-views in $\mathcal{U}$ will be function of our formal definition of (Δ, g, x)-partial-synchrony in Sec. 2). In this section, we now describe an approach which suffices to identify at least *one* adversary node in the case that there is a strict majority of honest nodes. Later, in Sec. 5, we show how to generalize the method to identify a greater number of adversary nodes when a greater proportion of super-views are synchronous and/or a smaller number of nodes are adversary, and we also drop the assumption that every super-view has at least one honest leader.

A Trivial Heuristic. If we can find a node that is majority blamed in a majority of super-views in $\mathcal{U}$, then we can identify that node as adversary. This holds since an honest node is never majority blamed

in a synchronous super-view and a majority of super-views in $\mathcal{U}$ are synchronous. The following is therefore a *sufficient* condition for being able to identify at least one adversary node: adversary nodes are on average majority blamed in over half of the super-views in $\mathcal{U}$. This argument has one-sided error (no false positives, but may have false negatives).

Extending the Trivial Heuristic. The basic idea is that, if one chooses the *right* subset of super-views in $\mathcal{U}$ (efficiently computable and guaranteed to contain all synchronous super-views in $\mathcal{U}$), then the heuristic above remains *sound* and is also *complete* (*i.e.*, applies whenever none of the views in $\mathcal{U}$ confirm new transactions).

The Details. Recall that $\mathcal{U}$ is a set of consecutive super-views. We assume that a majority of the super-views in $\mathcal{U}$ are synchronous.

(1) Let $\mathcal{P}_u^{\mathrm{A}}$ denote the set of nodes that are majority blamed in super-view $u \in \mathcal{U}$.
(2) Form an undirected graph G with $|\mathcal{U}|$ vertices, each corresponding to a super-view in $\mathcal{U}$, and an undirected edge (u, u') whenever $|\mathcal{P}_u^{\mathrm{A}} \cap \mathcal{P}_{u'}^{\mathrm{A}}| \geq n/6$.
(3) Let $\mathcal{U}'$ denote the subset of super-views u such that:
    (a) $|\mathcal{P}_u^{\mathrm{A}}| \geq n/3$; and
    (b) $\deg_G(u) \geq |\mathcal{U}|/2$.

Proof of Soundness. The heuristic never outputs an honest node:

- If u is synchronous, $|\mathcal{P}_u^{\mathrm{A}}| \geq n/3$.
- If u, u' are both synchronous, then $|\mathcal{P}_u^{\mathrm{A}} \cap \mathcal{P}_{u'}^{\mathrm{A}}| \geq n/6$ (since there are $< n/2$ adversary nodes).
- In the graph G above, every vertex corresponding to a synchronous super-view has degree $\geq |\mathcal{U}|/2$ (by previous point and the assumption that a majority of super-views are synchronous).
- Every synchronous super-view belongs to $\mathcal{U}'$ (by first and third points above).
- A majority of super-views in $\mathcal{U}'$ are synchronous (by the previous point and the assumption that a majority of super-views in $\mathcal{U}$ are synchronous).
- No honest node can be majority blamed in a majority of the super-views in $\mathcal{U}'$ (because no honest node is majority blamed in any synchronous super-view).

Proof of Completeness. To show that the heuristic is guaranteed to output a (necessarily adversary) node, we argue as follows:

- Consider a 0–1 matrix M with rows indexed by adversary nodes p and columns indexed by super-views u of $\mathcal{U}'$. Define $M_{p,u} = 1$ if $p \in \mathcal{P}_u^{\mathrm{A}}$ and $M_{p,u} = 0$ otherwise.
- For every synchronous super-view in $\mathcal{U}'$, the corresponding column sum is at least $n/3$.
- By properties (a) and (b) of super-views in $\mathcal{U}'$, for every asynchronous super-view in $\mathcal{U}'$, the corresponding column sum is at least $n/6$. This is because, by (b), every asynchronous super-view $u \in \mathcal{U}'$ must have overlap $|\mathcal{P}_u^{\mathrm{A}} \cap \mathcal{P}_{u'}^{\mathrm{A}}| \geq n/6$ with some synchronous super-view u, but in synchronous super-views only adversary nodes are majority blamed, so the overlap counts into the column sum of u in M despite M having rows only for adversary nodes.
- Because over half the super-views of $\mathcal{U}'$ are synchronous (see above), the sum of M's entries is $\geq \frac{|\mathcal{U}'|}{2} \cdot \frac{n}{3} + \frac{|\mathcal{U}'|}{2} \cdot \frac{n}{6} = \frac{n|\mathcal{U}'|}{4}$.

- Because there are less than $n/2$ rows (by assumption on the number of adversary nodes), the average row sum is $> \frac{|\mathcal{U}'|}{2}$.
- Thus, there exists a row with more than $\frac{|\mathcal{U}'|}{2}$ 1's—a adversary node that is majority blamed in a majority of the views in $\mathcal{U}'$.

Outline. Sec. 2 introduces the x-partial-synchrony model and our notion of accountable liveness, and also covers various preliminaries. Sec. 3 reviews the starting point of our protocol, a variant of Tendermint consensus. This protocol subsequently serves as our running example, in Secs. 4 and 5, for how to achieve accountable liveness. Sec. 6 proves the two impossibility results for accountable liveness that are mentioned above. Sec. 7 discusses related work. Sec. 8 concludes. Sec. E discusses additional related works.

2 Model & Preliminaries

We model protocol execution and define consensus security properties in a standard way, except for our use of x-partially-synchronous networks (Sec. 2.1) and our definition of accountable liveness (Sec. 2.3). There are n nodes, denoted $p \in \mathcal{P}$, each of which has a cryptographic identity (public/secret key pair for signatures) that is commonly known (*public-key infrastructure, PKI*). There is an *environment* that, over time, inputs *transactions* to the nodes, and to which, over time, each node outputs a *log* of transactions. The nodes' objective is *atomic broadcast*, *i.e.*, to reach agreement on an ordering of their input transactions into their output logs. For this purpose, nodes can send each other messages over a *network*. A computationally bounded *adversary* seeks to disrupt consensus and for this purpose can corrupt nodes and delay network messages. For ease of exposition, we treat cryptographic signatures as *ideal*, *i.e.*, we assume that the adversary cannot forge signatures of honest nodes.

Time proceeds in discrete *rounds* and, for simplicity, nodes are assumed to have synchronized clocks. In each round, each node receives messages from the network and possibly transactions from the environment. The node then updates its internal state, produces messages to send to other nodes via the network, and outputs a log of confirmed transactions to the environment. The adversary chooses f nodes, denoted $\mathcal{P}_{\mathrm{a}} \subseteq \mathcal{P}$, to corrupt at the beginning of the execution and in particular before any protocol randomness is drawn (*static corruption*). The adversary learns the internal state of *adversary* nodes, and can make them deviate from the protocol arbitrarily for the entire execution (*permanent Byzantine faults*). Non-adversary *honest* nodes ($\mathcal{P}_{\mathrm{h}} \subseteq \mathcal{P}$) follow the protocol.

2.1 x-Partially-Synchronous Networks

When nodes send messages to each other, the messages are *delayed* by the adversary subject to certain constraints. We propose the x-*partially-synchronous network model* as an interpolation between two classic network models, synchronous [42] and partially-synchronous [24] (also called *eventually-synchronous*) networks.

Specifically, under synchrony, there is a known delay upper-bound of $\Delta > 0$ rounds. It will be convenient to view the synchronous network model as follows. When a node instructs the network to send a message to another node, the message is enqueued in the recipient's pending message queue together with a *countdown* initialized to Δ. The adversary can decrease the countdown at will. The countdown also decreases by 1 with each round. Once the

countdown hits 0, the message is delivered to the recipient at the beginning of the next round. Partially-synchronous networks extend the above with a *global stabilization time* GST $< \infty$, a round which the adversary chooses adaptively. The countdown of pending messages is guaranteed to decrement each round only after GST.

Our *x-partially-synchronous setting* inherits the assumptions of partial synchrony. Furthermore, for every round before GST, the adversary adaptively chooses whether the round is synchronous (*i.e.*, all countdowns decrement) or not. (All rounds after GST are synchronous.) Call an interval Δ-*synchronous* if each of its rounds is synchronous in this sense. The restriction on the adversary is parameterized by (in addition to Δ) a known function g and a known value $x \in [0, 1]$. The restriction is then that, for every partition of the execution into *periods* of length Δ' rounds, and for every interval of $g(\Delta')$ periods, at least a $(1-x)$ fraction of the periods are Δ-synchronous. For example, $g(1)$ specifies the minimum length of an interval (in rounds) for which it is guaranteed that at most an x fraction of the interval's rounds are asynchronous. We generally assume that g grows unboundedly with Δ'—the longer the period of synchrony Δ' desired, the longer one may have to wait for it. For every function g, the x-partially-synchronous model generalizes the synchronous ($x = 0$) and partially synchronous ($x = 1$) models. Where precision about the network parameters is required, we write (Δ, g, x)-*partial-synchrony*. Since x is the dominant quantitative parameter affecting accountable liveness, while Δ and g are often clear from context, we also often write x-*partial-synchrony*. Sec. 2.5 discusses the plausibility of the x-partially-synchronous model from a practical perspective.

2.2 Atomic Broadcast

For logs, *i.e.*, transaction sequences, we write $\Lambda \preceq \Lambda'$ iff Λ is a prefix of or equal to Λ', and $\Lambda \asymp \Lambda'$ ("Λ is consistent with Λ'") iff $\Lambda \preceq \Lambda'$ or $\Lambda' \preceq \Lambda$. We follow the usual definition of atomic broadcast:

Definition 1. A protocol Π where node p at round t *confirms* the output log Λ_t^p, achieves *atomic broadcast* with *safety resilience* τ^S and *liveness resilience* τ^L, iff in every execution satisfying the desired network model:

- **Safety**: If $f \leq \tau^S$, then: $\forall p, q \in \mathcal{P}_h : \forall t, t' : \Lambda_t^p \asymp \Lambda_{t'}^q$.
- **Liveness**: If $f \leq \tau^L$, then: for every transaction tx input to all $\mathcal{P}_h$ by t_0^{tx}: $\exists t_1^{tx} < \infty : \forall t \geq t_1^{tx} : \forall p \in \mathcal{P}_h : tx \in \Lambda_t^p$.

For the minimum t_0^{*tx}, t_1^{*tx} that satisfy the liveness condition, called the *input round* and *confirmation round* of tx, respectively, we define $T_{conf}^{tx} \triangleq t_1^{*tx} - t_0^{*tx}$ as the *confirmation latency* of tx.

We say a protocol satisfying Def. 1 is τ^S-*safe* and τ^L-*live*.

2.3 Accountable Liveness

In addition to the parameters (Δ, g, x) of the x-partially-synchronous model, the notion of accountable liveness is parametric in *period length* Δ', *accountable-liveness resilience* τ_{max}^{AL}, *sensitivity* τ_{ident}^{AL}, and *failure probability* ε^{AL}, which will become clear below. Given these parameters, we build up the definition of accountable liveness in three steps. First, we define what it means for a *timely-liveness violation* to occur. Then, we define what it means for a protocol message to constitute a *certificate of guilt*. Finally, we define what it means for an atomic broadcast protocol to be *accountably live*.

Traditional (*eventual*) liveness (as in Def. 1) is a weaker property in some sense, and a stronger property in another sense, than what we need. Specifically, traditional liveness is so weak that violations of it cannot be assessed by any one point in time [4] (any transaction in question may still get confirmed in the future). But (Δ, g, x)-partial-synchrony with $x < 1$ allows to demand a more ambitious liveness guarantee where transactions need to be confirmed within $\Delta'g(\Delta')$ rounds. On the other hand, traditional liveness is strong in that it requires *all* honest nodes to confirm a transaction. For our purposes, a slightly weaker variant suffices, that requires only *all except "a few"* honest nodes to confirm. This is particularly meaningful under (Δ, g, x)-partial-synchrony with $x < 1$, because that setting implies a (large) delay upper-bound of $\Delta^* \triangleq \Delta'g(\Delta')$ so that if *some* honest node has confirmed a transaction, then *all* honest nodes will have confirmed that transaction Δ^* later.

Concretely, we say a protocol with maximum liveness resilience τ^L satisfies *timely-liveness* if all transactions are confirmed within $\Delta'g(\Delta')$ rounds by all except τ^L honest nodes. We concern ourselves with accountability for timely-liveness violations:

Definition 2. For a protocol with maximum liveness resilience τ^L, and for period length Δ', an execution has a *timely-liveness violation* at round t iff there exists a transaction tx with input round $t_0^{tx} \leq t - \Delta'g(\Delta')$ that (*strictly*) more than τ^L honest nodes have not confirmed by t, *i.e.*, $\exists tx$ input to all $\mathcal{P}_h$ by $t_0^{tx} \leq t - \Delta'g(\Delta')$: $\exists \mathcal{P}' \subseteq \mathcal{P}_h : (|\mathcal{P}'| > \tau^L) \wedge (\forall p \in \mathcal{P}' : tx \notin \Lambda_t^p)$.

Accountability is carried out through certificates of guilt:[3]

Definition 3. A protocol message m constitutes a *certificate of guilt* for a node p iff m is never obtained by any node in any (Δ, g, x)-partially-synchronous execution with $f \leq \tau_{max}^{AL}$ in which $p \in \mathcal{P}_h$.

Intuitively, accountable liveness means that, whenever an input transaction remains unconfirmed for "sufficiently long" by "many" honest nodes, "eye-witness evidence" (*e.g.*, many attestations to the unexpected absence of certain messages) emerges that accuses at least τ_{ident}^{AL} nodes of misbehavior. This eye-witness evidence is guaranteed to implicate only adversarial nodes provided the network was (Δ, g, x)-partially-synchronous and $f \leq \tau_{max}^{AL}$.

Definition 4. An atomic broadcast protocol Π is *accountably live* in (Δ, g, x)-partially-synchronous networks with *period length* Δ', *accountable-liveness resilience* τ_{max}^{AL}, *sensitivity* τ_{ident}^{AL}, and *failure probability* ε^{AL}, iff for any fixed round t: For every adversary with $f \leq \tau_{max}^{AL}$ in (Δ, g, x)-partially-synchronous networks, with probability at least $(1 - \varepsilon^{AL})$, if a timely-liveness violation occurs at round t, then, eventually, some honest node broadcasts a certificate of guilt for some set $\mathcal{P}' \subseteq \mathcal{P}$ of nodes with $|\mathcal{P}'| \geq \tau_{ident}^{AL}$.

For simplicity, and as this captures the crux of the problem, we state our (accountable) liveness claims for liveness violations that occur at a *fixed round*. (Accountable) liveness properties for *all*

---

[3]Note that Def. 3 is a *conditional* variant (where only (Δ, g, x)-partially-synchronous executions with $f \leq \tau_{max}^{AL}$ are considered) of the notion of a certificate of guilt that appeared in the accountable-safety literature [34, p. 5] (where no such restrictions are imposed on the executions considered). As shown in Secs. 1.1 and 1.2 and Thms. 2 and 3, the two restrictions on the executions in Def. 3, namely on network timing and on a maximum adversary strength, are necessary for the notion of accountable liveness to be achievable. Sec. 2.5 discusses that for suitably chosen parameters, restricting to x-partially-synchronous executions may practically not be very severe.

transactions *over an entire appropriately-bounded execution horizon* are readily obtained with a union bound.

2.4 Overall Protocol Design Goal

We now formally state our protocol design objective. We aim to design atomic broadcast protocols that guarantee two requirements:

(1) As a *baseline*, for any given τ, Δ, when run among $n = 3\tau + 1$ nodes in a *partially-synchronous* network (*i.e.*, x-partially-synchronous with $x = 1$), the protocol should be ($\tau^{\mathrm{S}} = \tau$)-safe and ($\tau^{\mathrm{L}} = \tau$)-live, and guarantee expected confirmation latency $O(\Delta)$, independent of g, for transactions input after GST. In other words, the protocol should satisfy the standard security and performance properties required of protocols for partial synchrony.

(2) *Additionally*, for any given $g, x, \tau_{\max}^{\mathrm{AL}}, \varepsilon^{\mathrm{AL}}$, the protocol should be accountably live for some $\Delta', \tau_{\mathrm{ident}}^{\mathrm{AL}}$ if the network happens to be (Δ, g, x)-partially-synchronous and if $f \leq \tau_{\max}^{\mathrm{AL}}$. Note that $\tau_{\max}^{\mathrm{AL}} > \tau^{\mathrm{L}}, \varepsilon^{\mathrm{AL}} < 1, \tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$ constitutes the non-trivial regime. Smaller Δ' and larger $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ are better, since it shortens the duration of non-confirmation after which accountability is required, and increases the number of adversary nodes identified.

2.5 Discussion

Recall that under (Δ, g, x)-partial-synchrony, the assumptions of partial synchrony hold regarding Δ and GST. In addition, there is a known function g and a known value $x \in [0, 1]$, such that, before GST, for any partition of time into periods of Δ' rounds, any interval of $g(\Delta')$ periods has at most x fraction of the periods not be Δ-synchronous. Is such an assumption practically plausible? Consider that typical global round-trip times in the Internet are in the order of hundreds of milliseconds, and that Internet connectivity outages lasting longer than hours are exceedingly rare. Furthermore, network service-level agreements commonly promise at most a certain latency for at least a certain fraction of every long-enough period of time. This leads us to believe that with x of tens of percentage points, Δ of seconds, Δ' of tens of seconds, and $\Delta' g(\Delta')$ of tens of hours, x-partial-synchrony is at least a plausible assumption. It does not appear much less plausible than assumptions of Internet delay upper-bounds of seconds, implicit in systems like Ethereum [15] or Cardano [32]. At the same time, the longer periods of synchrony one demands, the longer one plausibly has to wait, justifying that $g(\Delta')$ should grow unboundedly with Δ'.

Note that x-partial-synchrony for $x < 1$ implies a large delay upper-bound of $\Delta^* \triangleq \Delta' g(\Delta') \gg \Delta$. The reader may then ask, why not run a synchronous atomic broadcast protocol with that delay bound Δ^*? Given that some such protocols are safe and live even if almost all nodes are adversary [13, 23, 30, 31, 47], they would trivially be accountably live as well, since there are no timely-liveness violations (let alone liveness violations) to begin with. However, such protocols suffer from high latency in the order of $\Delta^* \gg \Delta$, and thus do not satisfy the target expected confirmation latency of $O(\Delta)$ after GST. Furthermore, such protocols do not guarantee safety under partial synchrony. Thus, such protocols do not satisfy the baseline goal (1) set out in Sec. 2.4.

The reader may wonder why we focus on the atomic broadcast variant of consensus, rather than, for instance, on the state-machine replication variant, which in addition to nodes also models the system's clients [47]. This is for two reasons: (1) Our x-partially-synchronous network model incorporates periods of asynchrony, and earlier works suggest [47] that while there is a considerable difference between state-machine replication and atomic broadcast under synchrony, this is not the case as soon as safety during periods of asynchrony is required. (2) Accountable liveness inherently arises from the interplay of nodes and their network delay (cf. Sec. 1.1). Unlike in accountable safety, clients play no particular role in accountable liveness, so, for simplicity, we leave them aside.

We focus exclusively on accountability of the liveness property here, and otherwise stick to "regular" unaccountable safety. This is because accountability of the safety property has been studied extensively already in earlier works [12, 14, 15, 37, 38, 45], and those techniques are orthogonal and can readily be applied independently to make an accountably live protocol accountably safe.

3 Consensus Protocol

A variant of the Tendermint consensus protocol [11] (inspired by [12, Sec. 9.1]) is provided in Alg. 1, described as pseudo-code from the perspective of any honest node p. Like earlier Tendermint versions, the protocol proceeds in *views*. Each view v is associated with a randomly selected *leader* node L_v, known to all nodes. Conceptually, each view consists of a proposal by the leader (Alg. 1, ln. 10), followed by rounds of voting (Alg. 1, lns. 14, 18 and 22).

There are two main differences compared to traditional Tendermint [11]: More time is allotted (highlighted in orange in Alg. 1) for each proposal and voting phase, and there is an extra third round of voting (highlighted in green in Alg. 1).

Let us discuss the first main difference. For simplicity, the extra delay before a view's proposal allows to analyze each view's liveness in isolation: if the network is synchronous *for the duration of that view v* (synchronous view), *i.e.*, messages sent during $12\Delta v$ to $12\Delta(v + 1) - \Delta$ arrive within Δ time, and L_v is honest, and "enough" votes are cast, then a new consensus decision is reached (by all honest nodes). Furthermore, for liveness alone, simple Δ delay would suffice before the proposal, after the proposal, and after each of the votes. Our delays of $2\Delta, 2\Delta, 3\Delta, 3\Delta, 2\Delta$, respectively, are chosen to enable accountable liveness, as will become clear in Sec. 4.

Regarding the second main difference, note that the extra third round of voting is different from the earlier two, in that nodes do not vote for a block, but indicate whether all transactions recently seen as pending have been confirmed. These VoteLive votes are used in Secs. 4 and 5 to detect liveness violations and trigger the production of certificates of guilt. The VoteLive votes are produced but not consumed in Alg. 1, and thus have no influence on proposing, voting, or confirming in Alg. 1, and can therefore be neglected in the traditional safety and liveness analyses of Alg. 1 (Lems. 1 and 2).

Regarding notation, for any protocol state variable $\mathfrak{X}$, we denote by $\mathfrak{X}_t^p$ the state as viewed by (honest) node p at time t. If we set $t = \infty$, we mean the state as viewed at the end of the execution. We may omit the node if clear from context, such as in Alg. 1. Specifically, we denote by $\mathfrak{M}_t^p$ the set of message received from the network, upon verification and stripping of the messages' signatures (Alg. 1, lns. 4 and 7), as viewed by node p at time t, and for any t including ∞, we define $\mathfrak{M}_t^{\cup} \triangleq \bigcup_{p \in \mathcal{P}_{\mathrm{h}}} \mathfrak{M}_t^p$ to denote the union of $\mathfrak{M}_t^p$ across

Algorithm 1 Tendermint consensus variant, code for node p (based on [11], [12, Sec. 9.1], with extra delay highlighted in orange, and extra round of liveness voting highlighted in green)

1   Blocks:
- $b \triangleq \text{Block}(p, v, b_{-1}, Q, \text{txs})$ consists of: creator p, view v, parent block b_{-1}, quorum certificate Q, transactions txs.
- Genesis block: $b_0 \triangleq \text{Block}(0, 0, \bot, \emptyset, \emptyset)$.
- Validity:

$$\text{valid}(\mathfrak{M}, b) \triangleq \quad (b = b_0)$$
$$\vee \quad ((b \in \mathfrak{M})$$
$$\wedge \text{valid}(\mathfrak{M}, b.Q, b.b_{-1}, 1)$$
$$\wedge (b.p = L_{b.v})$$
$$\wedge (b.v > b.b_{-1}.v)) \tag{1}$$

2   Votes:
- $w \triangleq \text{Vote}(p, b, s)$ consists of: creator p, target block b, vote stage s.
- Validity:

$$\text{valid}(\mathfrak{M}, w) \triangleq (w \in \mathfrak{M}) \wedge \text{valid}(\mathfrak{M}, w.b) \tag{2}$$

- $w' \triangleq \text{VoteLive}(p, v)$ consists of: creator p, view v.
- Validity:

$$\text{valid}(\mathfrak{M}, w') \triangleq (w' \in \mathfrak{M}) \tag{3}$$

3   Quorum certificates (QCs):
- Validity:

$$\text{valid}(\mathfrak{M}, Q, b, s) \triangleq \quad ((Q = \emptyset)$$
$$\wedge (b = b_0))$$
$$\vee \quad ((\forall w \in Q : \text{valid}(\mathfrak{M}, w) \wedge (w.b = b) \wedge (w.s = s))$$
$$\wedge (|\{w.p \mid w \in Q\}| > 2n/3)) \tag{4}$$

- Notation: If $\text{valid}(\mathfrak{M}, Q, b, s)$, then $v(Q) \triangleq b.v$.

4   $\mathfrak{M} \leftarrow \{b_0\}$ //

5   $\mathfrak{Q} \leftarrow \emptyset$

6   At all times, re-broadcast all messages and transactions received from the network or as input.

7   At all times, add to $\mathfrak{M}$ any message (block or vote) received from the network, upon verification and stripping of the message's signature (ensuring that message m was indeed created by $m.p$), and any transaction received from the network or as input. Denote the set of transactions in $\mathfrak{M}$ as $\mathfrak{M}t$. Record when elements are added to $\mathfrak{M}$ and $\mathfrak{M}t$ "as of" time t as $\mathfrak{M}_t$ and $\mathfrak{M}t_t$.

8   At all times, confirm as log Λ_t^p the sequence of transactions on the path from b_0 to b iff $\exists Q_1, Q_2 \subseteq \mathfrak{M} : \text{valid}(\mathfrak{M}, Q_1, b, 1) \wedge \text{valid}(\mathfrak{M}, Q_2, b, 2)$.

9   **for** $v = 1, 2, 3, \ldots$

10   **at** $t = 12\Delta v + 2\Delta$

11   **if** $L_v = p$

12         $(b, Q) \leftarrow \arg\max_{(b,Q):\text{valid}(\mathfrak{M}, Q, b, 1)} b.v$

13   Sign and broadcast $\text{Block}(p \leftarrow p, v \leftarrow v, b_{-1} \leftarrow b, Q \leftarrow Q, \text{txs} \leftarrow \{\text{tx} \in \mathfrak{M}t \mid \text{tx} \notin b_0.\text{txs}\|\ldots\|b.\text{txs}, \text{ for the path from } b_0 \text{ to } b\})$

14   **at** $t = 12\Delta v + 4\Delta$

15   **if** $\exists b : \text{valid}(\mathfrak{M}, b) \wedge (b.v = v)$ // $\text{valid}(\mathfrak{M}, b) \implies (b.p = L_v)$. Proceed only with one b satisfying the condition.

16   **if** $v(\mathfrak{Q}) \le v(b.Q)$

17   Sign and broadcast $\text{Vote}(p \leftarrow p, b \leftarrow b, s \leftarrow 1)$

18   **at** $t = 12\Delta v + 7\Delta$

19   **if** $\exists b, Q : (b.v = v) \wedge \text{valid}(\mathfrak{M}, Q, b, 1)$ // Proceed only with one (b, Q) satisfying the condition.

20         $\mathfrak{Q} \leftarrow Q$

21   Sign and broadcast $\text{Vote}(p \leftarrow p, b \leftarrow b, s \leftarrow 2)$

22   **at** $t = 12\Delta v + 10\Delta$

23   **if** $\mathfrak{M}t_{12\Delta v} \subseteq \Lambda_{12\Delta v + 10\Delta}^p$

24   Sign and broadcast $\text{VoteLive}(p \leftarrow p, v \leftarrow v)$

all honest nodes. By t^- we mean the time *just before* any honest node executes its code for time t. By t^+ we mean the time *just after* all honest nodes have executed their code for time t.

Lemma 1. *Assuming $n > 3f$, Alg. 1 is safe in partial synchrony (i.e., $\tau^S = \lfloor (n-1)/3 \rfloor$).*

Lemma 2. *Assuming $n > 3f$, Alg. 1 is live in partial synchrony (i.e., $\tau^L = \lfloor (n-1)/3 \rfloor$), with expected confirmation latency $O(\Delta)$ after GST.*

The proofs are analogous to those for earlier Tendermint variants [11] [12, Sec. 9.1] and are therefore relegated to Sec. A.

Note that Lems. 1 and 2 are both under partial synchrony and under the assumption $n > 3f$, as is part of our design goal. In fact, Lems. 1 and 2 show that Alg. 1 satisfies goal (1) in Sec. 2.4. In Sec. 4, we analyze what we can learn from *individual* synchronous views. In Sec. 5, we finally leverage the x-partially-synchronous model, and the *aggregate* combinatorial structure of synchronous views it implies, to arrive at accountable liveness.

4   Blame Accounting

In Secs. 4 and 5, we describe how certificates of guilt are produced so as to render the consensus protocol of Alg. 1 accountably live according to Def. 4. A high-level overview of the accountability process, which nodes run in parallel to Alg. 1, is given in Fig. 2.

At all times, nodes take note of potential timely-liveness violations (Fig. 2, step (2)), based on the lack of $2n/3$-quorums of VoteLive messages for the most recent views $\mathcal{V}$ of Alg. 1 of the recent $\Delta' g(\Delta')$ rounds (final determination of Δ' and $\mathcal{V}$ is made in Sec. 4.2 and Sec. 5; for now, think of them as 12Δ and as the most recent $g(\Delta')$ views in Alg. 1, respectively). Honest nodes sign and broadcast their *transcripts* of Alg. 1, which means their $\mathfrak{M}$, on a continuously ongoing basis.[4]

Assuming x-partial-synchrony, these transcripts propagate to all honest nodes within $\Delta' g(\Delta')$ time (Fig. 2, step (3)). Honest nodes discard equivocating, invalidly-signed, and syntactically-malformed transcripts, and use a default of $\bot$ for missing transcripts, so that for each node and time they retain exactly one transcript. If an honest node has taken note of a potential timely-liveness violation, the node applies a function ψ that is the centerpiece of liveness accountability developed in Secs. 4 and 5. Namely, ψ is applied to the sanitized transcripts, to obtain a set $\mathcal{P}'$ of seemingly guilty nodes, for which the node then signs and broadcasts accusations.

Again assuming x-partial-synchrony, these accusations propagate to all honest nodes within $\Delta' g(\Delta')$ time (Fig. 2, step (4)). If a node was accused for a particular point in time by a majority of nodes, then that constitutes a certificate of guilt for that node.

Note that in the above process, the adversary may cook up or alter its transcripts, subject to not being able to forge signatures. Honest nodes do not attempt to filter transcripts *semantically*, i.e., based on the information contained—rather, ψ will take care of that. Note that when ψ is invoked by an honest node, it is guaranteed that the proper transcripts of all honest nodes are provided to it. Secs. 4 and 5 are all about how to design ψ such that, under those circumstances, ψ never outputs an honest node, and ψ outputs a "large" set of adversary nodes whenever a transaction has remained unconfirmed for more than $\Delta' g(\Delta')$ time (Fig. 2, step (1)).

The ψ we construct proceeds in two steps: (1) *Blame accounting* (Sec. 4): First, ψ uses the $\{\mathfrak{M}_t^p\}$ to count how frequently a node p did not see a vote from node p' that p' "ought to" have sent, where

---

[4]There are various ways to reduce the communication overhead of this step, which we leave to future work, to retain simplicity, and since our primary focus here is to show the achievability of liveness accountability, not its most efficient implementation.
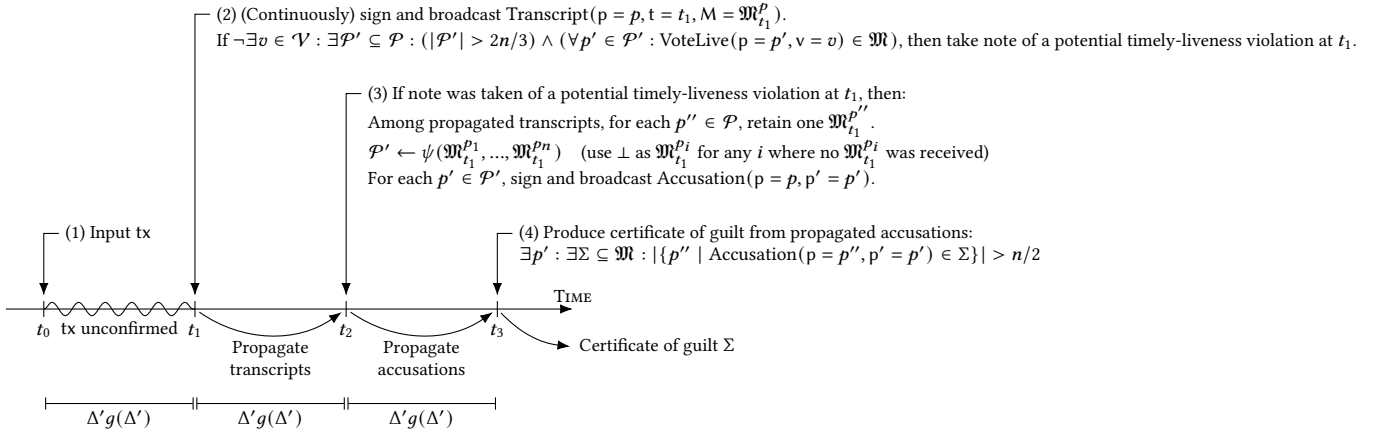
(2) (Continuously) sign and broadcast $\text{Transcript}(\mathsf{p} = p, \mathsf{t} = t_1, \mathsf{M} = \mathfrak{M}_{t_1}^p)$.
If $\neg \exists v \in \mathcal{V} : \exists \mathcal{P}' \subseteq \mathcal{P} : (|\mathcal{P}'| > 2n/3) \land (\forall p' \in \mathcal{P}' : \text{VoteLive}(\mathsf{p} = p', \mathsf{v} = v) \in \mathfrak{M})$, then take note of a potential timely-liveness violation at t_1.

(3) If note was taken of a potential timely-liveness violation at t_1, then:
Among propagated transcripts, for each $p'' \in \mathcal{P}$, retain one $\mathfrak{M}_{t_1}^{p''}$.
$\mathcal{P}' \leftarrow \psi(\mathfrak{M}_{t_1}^{P_1}, ..., \mathfrak{M}_{t_1}^{P_n})$ (use $\perp$ as $\mathfrak{M}_{t_1}^{P_i}$ for any i where no $\mathfrak{M}_{t_1}^{P_i}$ was received)
For each $p' \in \mathcal{P}'$, sign and broadcast $\text{Accusation}(\mathsf{p} = p, \mathsf{p}' = p')$.

(4) Produce certificate of guilt from propagated accusations:
$\exists p' : \exists \Sigma \subseteq \mathfrak{M} : |\{p'' \mid \text{Accusation}(\mathsf{p} = p'', \mathsf{p}' = p') \in \Sigma\}| > n/2$

(1) Input tx

Figure 2: Overview of how certificates of guilt for Alg. 1 are produced, from the perspective of a node p: (2) Nodes continuously share their transcripts of Alg. 1 with each other. If p has not received VoteLive from $> 2n/3$ nodes for the most recent views $\mathcal{V}$ of Alg. 1 of the last $\Delta' g(\Delta')$ rounds ($\mathcal{V}$ and Δ' are determined in Sec. 4.2 and Sec. 5), then p takes note of a potential timely-liveness violation. (3) Each node retains one transcript for each other node. Since honest transcripts are unique and guaranteed to propagate within $\Delta' g(\Delta')$ rounds, honest nodes always have the correct transcript for other honest nodes. If p has taken note of a potential timely-liveness violation, then: A function ψ, the centerpiece of liveness accountability developed in Secs. 4 and 5, is used to identify guilty nodes (and never mis-identifies honest nodes). Accusations for these nodes are shared among honest nodes. (4) Accusations against a particular node by a majority of nodes constitute a certificate of guilt. (1) If a transaction remains unconfirmed for $\Delta' g(\Delta')$ rounds, then $2\Delta' g(\Delta')$ rounds later a certificate of guilt for "many" adversary nodes is produced.

"ought to" is determined based on $\mathfrak{M}_t^p$ and the assumption that the network was synchronous. We say that p *blames* p' for unexplained missing votes that may have led to a timely-liveness violation. Note that these blame counts in isolation may be inaccurate, but form the basis for the next step. (2) *Adjudication rule* (Sec. 5): Second, ψ leverages the assumptions that, roughly, "not too many" nodes are adversary and that the network is "often synchronous" (x-partial-synchrony), and as a result, adversary nodes draw more blame than honest nodes, and an *adjudication rule* can be used to reliably identify adversary nodes based on the blame counts.

4.1 Simple Blame Accounting

Simple *blame accounting* for Alg. 1 is provided in Alg. 2. It is invoked with a set of views $\mathcal{V}$. The final choice of $\mathcal{V}$ is made in Sec. 4.2 and Sec. 5. For the remainder of Sec. 4.1, think of $\mathcal{V}$ as the set of the $g(\Delta')$ most recent views of Alg. 1, with $\Delta' = 12\Delta$. This choice and x-partial-synchrony gives us that at most x fraction of views in $\mathcal{V}$ are not synchronous (i.e., violate the network-delay upper-bound Δ). In the following, we observe how Alg. 2 behaves for individual views $v \in \mathcal{V}$, irrespective of the choice of $\mathcal{V}$.

Lemma 3. *With Algs. 1 and 2, for every $v \in \mathcal{V}$, if the network is synchronous for view v, and L_v is honest, then: unless the block b proposed by L_v is confirmed by all honest nodes by $12\Delta v + 9\Delta$ and all honest nodes see VoteLive messages for v from $> 2n/3$ nodes by $12\Delta v + 12\Delta$, there is a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$, such that for every $p \in \mathcal{P}_h$, for all $p' \in \mathcal{P}'$, $\text{Blame}_{p,v,p'} = 1$ in Alg. 2.*

Intuitively, this means that if conditions are "very good" during view v, but there is (or appears to be) no liveness, then there is a large set $\mathcal{P}'$ of nodes that all honest nodes blame simultaneously.

Algorithm 2 Blame accounting for Alg. 1, given views $\mathcal{V}$

1: $\mathfrak{M}_t^p \leftarrow \mathfrak{M}$ of Alg. 1, ln. 4, for node p and "receive-time-annotated", i.e., supports access to $\mathfrak{M}$ in the view of p of Alg. 1 "as of" any time t
2: $\forall p \in \mathcal{P}, v \in \mathcal{V}, p' \in \mathcal{P} : \text{Blame}_{p,v,p'} \leftarrow 0$ // Default: no blame
3: **for** $p \in \mathcal{P}, v \in \mathcal{V}, p' \in \mathcal{P}$
4:   **if** $\exists b \in \mathfrak{M}_{12\Delta v+3\Delta}^p : \text{valid}(\mathfrak{M}_{12\Delta v+3\Delta}^p, b) \land (b.v = v)$ // Assuming synchrony, if L_v has sent a valid proposal b for view v in time ...
5:     **if** $\neg \exists Q \subseteq \mathfrak{M}_{12\Delta v+1\Delta}^p, b' \in \mathfrak{M}_{12\Delta v+1\Delta}^p : \text{valid}(\mathfrak{M}_{12\Delta v+1\Delta}^p, Q, b', 1) \land (v(Q) > v(b.Q))$ // ... and p' cannot possibly have had an excuse not to stage-1 vote for the proposal ...
6:       **if** $\neg \exists w \in \mathfrak{M}_{12\Delta v+5\Delta}^p, b'' \in \mathfrak{M}_{12\Delta v+5\Delta}^p : \text{valid}(\mathfrak{M}_{12\Delta v+5\Delta}^p, w) \land (w.p = p') \land (w.b = b'') \land (w.s = 1) \land (b''.v = v)$ // ... and yet p' did not cast any valid stage-1 vote for any valid proposal for view v ...
7:         $\text{Blame}_{p,v,p'} \leftarrow 1$ // ... then p' deserves blame!
8:   **if** $\exists b \in \mathfrak{M}_{12\Delta v+6\Delta}^p, Q \subseteq \mathfrak{M}_{12\Delta v+6\Delta}^p : (b.v = v) \land \text{valid}(\mathfrak{M}_{12\Delta v+6\Delta}^p, Q, b, 1)$ // Assuming synchrony, if there was a valid proposal b for v, and there was a valid stage-1 QC for b ...
9:     **if** $\neg \exists w \in \mathfrak{M}_{12\Delta v+8\Delta}^p, b' \in \mathfrak{M}_{12\Delta v+8\Delta}^p, Q' \subseteq \mathfrak{M}_{12\Delta v+8\Delta}^p : \text{valid}(\mathfrak{M}_{12\Delta v+8\Delta}^p, w) \land (w.p = p') \land (w.b = b') \land (w.s = 2) \land (b.v = v) \land \text{valid}(\mathfrak{M}_{12\Delta v+8\Delta}^p, Q, b, 1)$ // ... and yet p' did not cast any valid stage-2 vote for any valid proposal b' for view v for which there was a valid stage-1 QC ...
10:      $\text{Blame}_{p,v,p'} \leftarrow 1$ // ... then p' deserves blame!
11: **if** $\mathfrak{M}t_{12\Delta v+1}^p \subseteq \Lambda_{12\Delta v+9\Delta}^p$ // Assuming synchrony, if all transactions pending at the beginning of the view were confirmed within the view ...
12:    **if** $\neg \exists w' \in \mathfrak{M}_{12\Delta v+11\Delta}^p : (w'.p = p') \land (w'.v = v)$ // ... and yet p' did not cast a VoteLive vote for view v ...
13:      $\text{Blame}_{p,v,p'} \leftarrow 1$ // ... then p' deserves blame!

Lemma 4. *With Algs. 1 and 2, for every $v \in \mathcal{V}$, if the network is synchronous for view v, then for every $p \in \mathcal{P}_h$, for every $p' \in \mathcal{P}_h$, $\text{Blame}_{p,v,p'} = 0$ in Alg. 2.*

Intuitively, this means that if conditions are "good" during view v, then honest nodes do not blame each other.

Before we proceed to the proofs of Lems. 3 and 4, let us explain the intuition behind the design of Alg. 2 and its relation to Alg. 1. Recall that compared to traditional Tendermint variants [11], [12, Sec. 9.1], which have Δ delay after the proposal and after each round of voting, Alg. 1 also has delay *before* the proposal. This pedagogical simplification, which can be discharged, allows to treat the liveness of each view independently; it entails that the timeframes of the decision processes of different views do not overlap: If the network is synchronous for a view and the view's leader is honest, then the delay before the proposal ensures that the proposal is such that it is viewed as admissible (Alg. 1, ln. 16) to vote for by all honest nodes. Subsequent delays after the proposal and after each round of voting ensure that, assuming synchrony, all honestly produced messages reach all honest nodes in time to proceed with the next protocol phase and to ultimately lead to a new confirmed block.

Alg. 1 increases the phase-separating delays from Δ, Δ, Δ, Δ to 2Δ, 2Δ, 3Δ, 3Δ, 2Δ. The key reason for increasing the delays is: Assuming synchrony, if an honest node p sees a message m by time t, then it knows that all nodes must see m by time $t + \Delta$, due to p re-broadcasting m at t. Vice versa, if p does not see m by time t, it knows that no honest node can have seen m by time $t - \Delta$. Thus, an honest node's $\mathfrak{M}_t^p$ serves as an inner and outer bound (in the sense of subsets) on the message sets of any other (purportedly honest) node shortly after or before, respectively. Thus, from $\mathfrak{M}_t^p$ of an honest node p, assuming synchronous view v, one can infer (part of) the internal state of another (purportedly honest) node p' as of $t - \Delta$, to the extent that the state affects the sending/not-sending of messages, which is observable in $\mathfrak{M}_t^p$. For instance, Alg. 2, ln. 5 infers the "highest" (by view) lock $\mathfrak{Q}^{p'}$ any honest p' could have had by $12\Delta v$ (and, since locks are updated only at Alg. 1, ln. 20, by $12\Delta v + 4\Delta$, which is critical for the check of Alg. 1, ln. 16 to pass).

One can also, assuming synchronous view v, infer from $\mathfrak{M}_t^p$ for honest p what another (purportedly honest) p' "ought to" do at $t + \Delta$, to the extent that such action is affected by the receipt/non-receipt of messages. For instance, Alg. 2, ln. 4 infers that p' receives a valid timely proposal for view v (Alg. 1, ln. 15) and thus "ought to" stage-1 vote for it in Alg. 1, ln. 17.

Since there is Δ time lag in the backward and forward direction in the arguments above, respectively, this explains why most of the Δ delays in the original Tendermint protocol turn into 2Δ delays in Alg. 1 when liveness accountability is sought. The 3Δ delay between the first and second round of voting in Alg. 1 is needed because there is "an extra round of indirection" between one honest node seeing a stage-1 QC and all honest nodes seeing it. Specifically, the check of Alg. 2, ln. 8 needs to be Δ after the check of Alg. 2, ln. 6 (to ensure that if Alg. 2, ln. 8 fails, there is a large set of nodes that are blamed by *all* honest nodes in Alg. 2, ln. 6), which in turn must be Δ after Alg. 1, ln. 17, and Alg. 2, ln. 8 needs to be Δ before Alg. 1, ln. 21 (to ensure that if Alg. 2, ln. 8 passes, all nodes "ought to" stage-2 vote in Alg. 1, ln. 21). The case of block confirmation (based on stage-1 and stage-2 QCs) and a quorum of VoteLive votes from $> 2n/3$ nodes is analogous; hence the 3Δ delay there as well.

Proofs of Lems. 3 and 4 are relegated to Secs. B.1 and B.2.

Algorithm 3 Refined blame accounting for Alg. 1, given super-views $\{\mathcal{V}_u\}_{u \in \mathcal{U}}$ (changes over Alg. 2 highlighted in orange)

```
1   𝔐ₜᵖ ← 𝔐 of Alg. 1, ln. 4, for node p and "receive-time-annotated", i.e., supports
    access to 𝔐 in the view of p of Alg. 1 "as of" any time t
2   ∀p ∈ 𝒫, u ∈ 𝒰, p' ∈ 𝒫 : Blame_{p,u,p'} ← 0
3   for p ∈ 𝒫, u ∈ 𝒰, v ∈ 𝒱ᵤ, p' ∈ 𝒫
4       ... same as Alg. 2, ln. 4 ...
5           ... same as Alg. 2, ln. 5 ...
6               ... same as Alg. 2, ln. 6 ...
7                   Blame_{p,u,p'} ← 1
8       ... same as Alg. 2, ln. 8 ...
9           ... same as Alg. 2, ln. 9 ...
10              Blame_{p,u,p'} ← 1
11      ... same as Alg. 2, ln. 11 ...
12          ... same as Alg. 2, ln. 12 ...
13              Blame_{p,u,p'} ← 1
```

4.2 Refined Blame Accounting

Looking at Lems. 3 and 4, the blame accounting of Alg. 2 has a drawback. Based on the network model, views of Alg. 1 can be categorized into three groups. Either (a) the network is asynchronous for the view, or (b) the network is synchronous for the view but the view's leader L_v is adversary, or (c) the network is synchronous for the view and the view's leader L_v is honest. Lems. 3 and 4 make no guarantee for (a)—which seems unavoidable. Lem. 4 protects honest nodes from blame during (b) and (c)—which seems the best we can hope for. But Lem. 3 ensures blame for adversary nodes, if there is no liveness, only during (c). During (b), the adversary "gets away" without blame.

The refined blame accounting of Alg. 3 improves this. It assumes a partition $\{\mathcal{V}_u\}_{u \in \mathcal{U}}$ of views into *super-views*. If the network is synchronous for a super-view $u \in \mathcal{U}$, then honest nodes are protected from blame, and as long as *any* view $v \in \mathcal{V}_u$ has an honest leader L_v, "many" adversary nodes are blamed by all honest nodes for the super-view u if there is no liveness. Foreshadowing Sec. 5, $\{\mathcal{V}_u\}_{u \in \mathcal{U}}$ is chosen as follows: For $\Delta' = 12\Delta K_{\text{views}}$ for some fixed parameter K_{views}, choose the $g(\Delta')$ most recent periods of length Δ' of the execution, aligned to view boundaries in Alg. 1, as super-views $\mathcal{U}$. Each of these periods u corresponds to K_{views} views of Alg. 1, which make $\mathcal{V}_u$. Finally, $\mathcal{V} \triangleq \bigcup_{u \in \mathcal{U}} \mathcal{V}_u$. Note that even for moderate K_{views}, *most* super-views u have at least *some* view $v \in \mathcal{V}_u$ with honest L_v, due to the random choice of L_v. This and x-partial-synchrony will give us that almost $(1 - x)$ fraction of super-views in $\mathcal{U}$ are synchronous *and* have some honest leader. Revisiting the earlier categorization of views, for super-views, this eliminates the problematic group (b) and leaves us to deal only with (a) and (c). In the following, we observe how Alg. 3 behaves for individual super-views $u \in \mathcal{U}$, irrespective of the choice of $\{\mathcal{V}_u\}_{u \in \mathcal{U}}$. Changes in Lems. 5 and 6 over Lems. 3 and 4 are highlighted in orange.

Lemma 5. *With Algs. 1 and 3, for every super-view $u \in \mathcal{U}$, if the network is synchronous for all views $\mathcal{V}_u$, and L_v is honest for some view $v \in \mathcal{V}_u$, then: unless the block b proposed by L_v is confirmed by all honest nodes by $12\Delta v + 9\Delta$ and all honest nodes see* VoteLive *messages for v from $> 2n/3$ nodes by $12\Delta v + 12\Delta$, there is a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$, such that for every $p \in \mathcal{P}_h$, for all $p' \in \mathcal{P}'$, $\text{Blame}_{p,u,p'} = 1$ in Alg. 3.*

Algorithm 4 Critical-Subsets adjudication rule for Algs. 1 and 3, given super-views $\{\mathcal{V}_u\}_{u \in \mathcal{U}}$, $\tau_{\max}^{\mathrm{AL}} < n/2$, $x < 1/2$

1: Compute $\mathrm{Blame}_{p,u,p'}$ for all $p \in \mathcal{P}$, $p' \in \mathcal{P}$, $u \in \mathcal{U}$ using Alg. 3
2: $\forall u \in \mathcal{U} : \mathcal{P}_u^{\mathrm{A}} \leftarrow \{p' \in \mathcal{P} \mid \sum_{p \in \mathcal{P}} \mathrm{Blame}_{p,u,p'} \geq n - \tau_{\max}^{\mathrm{AL}}\}$
3: Undirected graph $G \leftarrow (\mathcal{U}, \{\{u, u'\} \mid |\mathcal{P}_u^{\mathrm{A}} \cap \mathcal{P}_{u'}^{\mathrm{A}}| \geq 2n/3 - \tau_{\max}^{\mathrm{AL}}\})$
4: $\mathcal{U}' \leftarrow \{u \in \mathcal{U} \mid (|\mathcal{P}_u^{\mathrm{A}}| \geq n/3) \wedge (\deg_G(u) > (x + \delta_x)|\mathcal{U}|)\}$
5: **return** $\{p \in \mathcal{P} \mid \exists \mathcal{U}'' \subseteq \mathcal{U}' : (\forall u \in \mathcal{U}'' : p \in \mathcal{P}_u^{\mathrm{A}}) \wedge (|\mathcal{U}''| > (x + \delta_x)|\mathcal{U}'|)\}$

PROOF. Follows the same steps as the proof of Lem. 3. □

Lemma 6. *With Algs. 1 and 3, for every* super-view $u \in \mathcal{U}$, *if the network is synchronous for* all views $\mathcal{V}_u$, *then for every* $p \in \mathcal{P}_{\mathrm{h}}$, *for every* $p' \in \mathcal{P}_{\mathrm{h}}$, $\mathrm{Blame}_{p,u,p'} = 0$ *in Alg. 3.*

PROOF. Follows directly from Lem. 4. □

5 Adjudication Rule

In Sec. 1.4, we described an adjudication rule that suffices to determine at least one adversary node in the event of a timely-liveness violation. A precise version of this rule is specified in Alg. 4. We now tackle the analysis for this rule in the general case for parameters x, $\tau_{\max}^{\mathrm{AL}}$, and when, due to the random leader election, not all super-views may have some honest leader. Recall that $\{\mathcal{V}_u\}_{u \in \mathcal{U}}$ are chosen as follows: For $\Delta' = 12\Delta K_{\mathrm{views}}$ for some fixed parameter K_{views}, we choose the $g(\Delta')$ most recent periods of length Δ' of the execution, aligned to view boundaries in Alg. 1, as super-views $\mathcal{U}$. Each of these periods u corresponds to K_{views} views of Alg. 1, which make up $\mathcal{V}_u$. Also, $\mathcal{V} \triangleq \bigcup_{u \in \mathcal{U}} \mathcal{V}_u$. A super-view is *synchronous* if the network respects a message-delay upper-bound Δ during view v of Alg. 1 for all $v \in \mathcal{V}_u$.

Note that even with moderate K_{views}, *most* super-views u have at least *some* view $v \in \mathcal{V}_u$ with honest L_v, due to the random choice of L_v. This and x-partial-synchrony give us that, for a fixed point in time where ψ is invoked, and appropriate choice of δ_x, except with some small probability, at least $(1 - x - \delta_x)$ fraction of super-views in $\mathcal{U}$ are synchronous *and* have some honest leader; vice versa, at most $(x + \delta_x)$ fraction of super-views in $\mathcal{U}$ are either not synchronous *or* have no honest leader. Denote by $\mathcal{U}_{\mathrm{s}}$ the subset of $\mathcal{U}$ containing all synchronous super-views, by $\mathcal{U}_{\mathrm{hl}}$ the subset of $\mathcal{U}$ containing all super-views with some honest leader, and by $\mathcal{U}_{\mathrm{shl}} \triangleq \mathcal{U}_{\mathrm{s}} \cap \mathcal{U}_{\mathrm{hl}}$. Complementation of these sets is taken with respect to $\mathcal{U}$. Note that with this notation we expect $|\mathcal{U}_{\mathrm{shl}}| \geq (1 - x - \delta_x)|\mathcal{U}|$, which Lem. 7 establishes based on a well-known Chernoff bound.

Lemma 7. *For a fixed round, and fixed $\delta_x > 0$, assuming (Δ, g, x)-partial-synchrony and $f \leq \tau_{\max}^{\mathrm{AL}} < n/2$, and choosing $K_{\mathrm{views}} = \lceil \log_2(\frac{2}{\delta_x}) \rceil$, except with probability $\exp(-\delta_x g(\Delta')/6)$, we get $|\mathcal{U}_{\mathrm{shl}}| \geq (1 - x - \delta_x)|\mathcal{U}|$.*

The proof is relegated to Sec. C.1.

Since we consider the regime where $x < 1/2$, we may choose $\delta_x < 1/2 - x$ by choosing $K_{\mathrm{views}} = \lceil \log_2(\frac{4}{1-2x}) \rceil + 1$, to ensure $1 - x - \delta_x > x + \delta_x$, which we subsequently make use of.

Remark 1. *For Lems. 8 to 10, we assume that some honest node took note of a potential timely-liveness violation in Fig. 2, that (Δ, g, x)-partial-synchrony holds, and that $1 - x - \delta_x > x + \delta_x$.*

We first show that $\mathcal{U}_{\mathrm{shl}} \subseteq \mathcal{U}'$:

Lemma 8. *For a fixed round, under the assumptions of Rem. 1, except with probability $\exp(-\delta_x g(\Delta')/6)$, for $\mathcal{U}'$ of Alg. 4, $\mathcal{U}_{\mathrm{shl}} \subseteq \mathcal{U}'$.*

The proof is relegated to Sec. C.2.
We then show that Alg. 4 never outputs an honest node:

Lemma 9 (Soundness of ψ). *For a fixed round, under the assumptions of Rem. 1, except with probability $\exp(-\delta_x g(\Delta')/6)$, no $p \in \mathcal{P}_{\mathrm{h}}$ is returned by Alg. 4.*

The proof is relegated to Sec. C.3.
Finally, we show that Alg. 4 outputs a certain number of (necessarily adversary) nodes:

Lemma 10 (Completeness of ψ). *For a fixed round, under the assumptions of Rem. 1, except with probability $\exp(-\delta_x g(\Delta')/6)$, Alg. 4 returns $\mathcal{P}'$ with $|\mathcal{P}'| \geq f - \left\lfloor \frac{(f - n/3) + (x + \delta_x)(\tau_{\max}^{\mathrm{AL}} - n/3)}{1 - x - \delta_x} \right\rfloor$.*

The proof is relegated to Sec. C.4.

Remark 2. It is easy to verify that when Alg. 4 is used in the context of ψ in Fig. 2, that $\mathcal{P}'$ obtained from ψ when input all honest transcripts and $\perp$ for all adversary transcripts, satisfies the bound on $|\mathcal{P}'|$ of Lem. 10, and is a subset of any $\mathcal{P}''$ obtained from ψ when input all honest transcripts and any other adversary-produced transcripts. This property is crucial to ensuring that $\mathcal{P}'$ is accused by *all* honest nodes in the context of Fig. 2, so that subsequently a certificate of guilt for $\mathcal{P}'$ is formed.

We combine the above results to assert accountable liveness:

Theorem 1. *For any given $g, x < 1/2, n/3 < \tau_{\max}^{\mathrm{AL}} < n/2, \varepsilon^{\mathrm{AL}}$, the atomic broadcast protocol of Alg. 1 together with Algs. 3 and 4, with design parameter δ_x, when instantiated with $K_{\mathrm{views}} = \lceil \log_2(\frac{2}{\delta_x}) \rceil + C$ and $\Delta' = 12\Delta K_{\mathrm{views}}$, where C is chosen such that $\exp(-\delta_x g(\Delta')/6) \leq \varepsilon^{\mathrm{AL}}$, is accountably live for $\tau_{\mathrm{ident}}^{\mathrm{AL}} = \tau_{\max}^{\mathrm{AL}} - \left\lfloor \frac{(1 + x + \delta_x)(\tau_{\max}^{\mathrm{AL}} - n/3)}{1 - x - \delta_x} \right\rfloor$, when run in (Δ, g, x)-partial-synchrony with $f \leq \tau_{\max}^{\mathrm{AL}}$.*

The proof is relegated to Sec. C.5.
Thm. 1 shows that the protocol consisting of the combination of Algs. 1, 3 and 4 satisfies goal (2) in Sec. 2.4. Together with Lems. 1 and 2, this completes the picture that the protocol satisfies the goals laid out in Sec. 2.4.

It is important to observe that, as required by Def. 4, $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ of Thm. 1 provides a lower bound *across* all executions with $f \leq \tau_{\max}^{\mathrm{AL}}$, on the number of identifiable adversary nodes. How does the number of identifiable adversary nodes depend on the actual number of adversary nodes f? This is equivalent to asking for a variant of Def. 4 with a function $\tau_{\mathrm{ident}}^{\mathrm{AL}}(f)$ instead of a constant $\tau_{\mathrm{ident}}^{\mathrm{AL}}$. The answer is provided by Lem. 10.

6 Impossibility Results

It is well-known [24] that atomic broadcast protocols that are τ^{S}-safe and τ^{L}-live under partial synchrony, must satisfy $n > 2\tau^{\mathrm{L}} + \tau^{\mathrm{S}}$. Since protocols with these properties are the baseline laid out in our design goals in Sec. 2.4, we focus subsequently on protocols that achieve that bound in an optimal fashion:

Definition 5. An atomic broadcast protocol with safety resilience τ^{S} and liveness resilience τ^{L} is *optimally resilient* iff $n = 2\tau^{\mathrm{L}} + \tau^{\mathrm{S}} + 1$.

6.1 Impossibility for $\tau_{\max}^{\mathrm{AL}} \geq n/2$

A perhaps intuitive consequence of the requirements that "enough" honest nodes alone should be able to produce certificates of guilt, while "few" adversary nodes alone should not, is that non-trivial accountable liveness is impossible for $\tau_{\max}^{\mathrm{AL}} \geq n/2$:

Theorem 2. No atomic broadcast protocol can be 0-safe under partial synchrony and achieve non-trivial accountable liveness with $\tau_{\max}^{\mathrm{AL}} \geq n/2$ and $\tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$ under (Δ, g, x)-partial-synchrony (even if $x = 0$, *i.e.*, even under synchrony).

The proof is relegated to Sec. D.1. (Note that for any $\tau > 0$, τ-safety implies $(\tau - 1)$-safety, so 0-safety is the weakest non-trivial such threshold safety property.)

6.2 Impossibility for $x \geq 1/2$

For the scheme of Secs. 4 and 5, it is intuitive that non-trivial accountable liveness cannot be achieved if the network is "more asynchronous than synchronous", since honest nodes can then be framed by the adversary more often than they can reliably detect adversary behavior. Perhaps interestingly, this impossibility holds for *all* consensus protocols and *all* liveness accountability mechanisms:

Theorem 3. No optimally-resilient atomic broadcast protocol with $\tau^{\mathrm{L}} > 0$ that is 0-safe under partial synchrony can achieve non-trivial accountable liveness with $\tau_{\max}^{\mathrm{AL}} > \tau^{\mathrm{L}}$ and $\tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$ under (Δ, g, x)-partial-synchrony for $x \geq 1/2$.

The proof is relegated to Sec. D.2.

Corollary 1. No optimally-resilient atomic broadcast protocol with $\tau^{\mathrm{L}} > 0$ that is 0-safe under partial synchrony can achieve non-trivial accountable liveness with $\tau_{\max}^{\mathrm{AL}} > \tau^{\mathrm{L}}$ and $\tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$ under partial synchrony.

PROOF. Follows from Thm. 3 and the fact that partial synchrony is a special case of x-partial-synchrony with $x = 1$. □

6.3 Impossibility for $\tau_{\mathrm{ident}}^{\mathrm{AL}}$

Thms. 2 and 3 show that accountable liveness can at best be expected for the region circumscribed by $x < 1/2$ and $\tau_{\max}^{\mathrm{AL}} < n/2$. Furthermore, only $\tau_{\max}^{\mathrm{AL}} \geq n/3$ is of interest, as a protocol for $n = 3\tau + 1$ with $\tau = \tau^{\mathrm{L}} = \tau^{\mathrm{S}}$ otherwise achieves trivial accountable liveness (since the protocol then is always live when $f \leq \tau_{\max}^{\mathrm{AL}}$). In fact, the atomic broadcast consensus protocol of Alg. 1 together with the accountability mechanism of Fig. 2 and Algs. 3 and 4 achieves non-trivial accountable liveness with *some* $\tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$ for the interior of the aforementioned region. But is the value of $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ achieved by this combined scheme "good"? Could one hope to achieve even higher $\tau_{\mathrm{ident}}^{\mathrm{AL}}$, either with a different consensus protocol and/or with a different accountability mechanism?

Let us address this question in four steps. First, a simple argument shows that *no* combination of consensus protocol and accountability mechanism can achieve $\tau_{\mathrm{ident}}^{\mathrm{AL}} > \tau^{\mathrm{L}} + 1$. Second, we show that for the consensus protocol of Alg. 1, *every* accountability mechanism has to satisfy $\tau_{\mathrm{ident}}^{\mathrm{AL}} < (\tau^{\mathrm{L}} + 2) - \left\lfloor \frac{\tau_{\max}^{\mathrm{AL}} - (\tau^{\mathrm{L}} + 1)}{k-2} \right\rfloor$ for every $k \geq 3$ and $x \geq 1/k$. Third, the proof of the aforementioned bound reveals that the bound holds for a large class of PBFT-style consensus protocols, *irrespective* of the liveness accountability mechanism. As can be

seen from Fig. 1 (—— vs. ·········), the bound closely matches the $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ achieved by the combined scheme of Algs. 1, 3 and 4 and Fig. 2, and is even tight for $x = 1/3$. In this sense, the combined scheme of Algs. 1, 3 and 4 and Fig. 2 is near optimal. Finally, we explain why we conjecture that the bound holds for *every* combination of consensus protocol (not just PBFT-style) and liveness accountability mechanism that together satisfy the design goals set out in Sec. 2.4.

For the first step, optimal resilience (Def. 5, cf. Lem. 12 in Sec. D) implies that for any protocol for $n = 3\tau + 1$ with $\tau = \tau^{\mathrm{L}} = \tau^{\mathrm{S}}$, an adversary only needs to let $\tau^{\mathrm{L}} + 1$ of its f nodes behave adversarily (in particular, let them crash) to cause a liveness violation, while the remaining $f - (\tau^{\mathrm{L}} + 1)$ adversary nodes can behave indistinguishably from honest nodes. Thus, we cannot hope to guarantee to detect more than $\tau^{\mathrm{L}} + 1$ guilty nodes in case of any liveness violation. Ergo, no more than $\tau_{\mathrm{ident}}^{\mathrm{AL}} = \tau^{\mathrm{L}} + 1$ can be achieved by any protocol.

For the second step:

Theorem 4. For every $k \geq 3$: Alg. 1, irrespective of the liveness accountability mechanism, cannot achieve accountable liveness for any $\tau_{\max}^{\mathrm{AL}} > \tau^{\mathrm{L}}$ under (Δ, g, x)-partial-synchrony for $x \geq 1/k$, unless $\tau_{\mathrm{ident}}^{\mathrm{AL}} < (\tau^{\mathrm{L}} + 2) - \left\lfloor \frac{\tau_{\max}^{\mathrm{AL}} - (\tau^{\mathrm{L}} + 1)}{k-2} \right\rfloor$.

The proof is relegated to Sec. D.3.

For the third step, we note that the specifics of Alg. 1 enter into the proof of Thm. 4 only to argue that there is a timely-liveness violation in one of the executions $E_{k,i}$ (cf. Fig. 4) considered in the proof. The commonly considered classical PBFT-style consensus protocols like Alg. 1, PBFT [17], Tendermint [11], HotStuff [51], CasperFFG [14, 21], or Streamlet [18], all exhibit timely-liveness violations in these executions. Intuitively, this is because the protocols never collect enough ($> 2n/3$) votes to form quorum certificates, before a view change is triggered due to lack of progress. They thus cannot confirm blocks. More formally, this is because the protocols satisfy the following property:

Definition 6. We say a consensus protocol is *now-or-never* iff there exists Δ'' dividing $\Delta' g(\Delta')/k$ such that if the protocol execution is partitioned into intervals of length Δ'', and for each interval, any $\lceil n/3 \rceil$ nodes are partitioned off temporarily until the end of the interval, while all other communication is next-round-delay, then the protocol does not confirm any transaction.

This condition is in fact sufficient to prove an analogue of Thm. 4:

Theorem 5. For every $k \geq 3$: A now-or-never consensus protocol, irrespective of the liveness accountability mechanism, cannot achieve accountable liveness for any $\tau_{\max}^{\mathrm{AL}} > \tau^{\mathrm{L}}$ under (Δ, g, x)-partial-synchrony for $x \geq 1/k$, unless $\tau_{\mathrm{ident}}^{\mathrm{AL}} < (\tau^{\mathrm{L}} + 2) - \left\lfloor \frac{\tau_{\max}^{\mathrm{AL}} - (\tau^{\mathrm{L}} + 1)}{k-2} \right\rfloor$.

The proof is relegated to Sec. D.4.

Now to the final fourth step: Why does this proof not carry over to general atomic broadcast protocols? The argument for Thms. 4 and 5 hinges on the assumption that one of the $E_{k,i}$ (cf. Fig. 4) has a timely-liveness violation. An asynchronous protocol, for instance, may be able to leverage the limited rounds of message exchange allowed at the boundaries of the $\mathcal{T}_i$ or $\mathcal{T}^{(v)}$ to achieve timely-liveness occasionally, independently of any Δ'' and in particular for $\Delta'' = \Delta' g(\Delta')/k$. Nonetheless, we conjecture that even a randomized asynchronous protocol would have a constant probability for a timely-liveness violation in one of the $E_{k,i}$, because

only a small constant number of full communication rounds can be completed.

Conjecture 1. *For every* $k \geq 3$: *Every atomic broadcast consensus protocol*, *irrespective of the liveness accountability mechanism*, *cannot achieve accountable liveness for any* $\tau_{\max}^{\mathrm{AL}} > \tau^{\mathrm{L}}$ *under* (Δ, g, x)-*partial-synchrony for* $x \geq 1/k$, *unless* $\tau_{\mathrm{ident}}^{\mathrm{AL}} < (\tau^{\mathrm{L}}+2) - \left\lfloor \frac{\tau_{\max}^{\mathrm{AL}} - (\tau^{\mathrm{L}}+1)}{k-2} \right\rfloor$.

It is important to observe that, as stated, Thms. 4 and 5 and Conj. 1 assert a bound on $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ as used in Def. 4, *i.e.*, where $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ is no more than the lowest number of guilty nodes identified across *all executions with* $f \leq \tau_{\max}^{\mathrm{AL}}$. The reasoning for Thm. 4 identifies worst-case executions where $f \approx \tau_{\max}^{\mathrm{AL}}$ nodes act adversarily and the number of guilty nodes identified is minimized. One may be interested in a variant of Def. 4 where $\tau_{\mathrm{ident}}^{\mathrm{AL}}$ is a function of the actual number of adversary nodes f, and $\tau_{\mathrm{ident}}^{\mathrm{AL}}(f)$ is no more than the lowest number of guilty nodes identified across *all executions with* f *adversary nodes*. Indeed, the steps of the argument for Thm. 4 go through with f instead of $\tau_{\max}^{\mathrm{AL}}$ to obtain $\tau_{\mathrm{ident}}^{\mathrm{AL}}(f) < (\tau^{\mathrm{L}}+2) - \left\lfloor \frac{f - (\tau^{\mathrm{L}}+1)}{k-2} \right\rfloor$.

7 Related Work

Accountable Liveness. It appears the term "accountable liveness" was first introduced by Tas *et al.* [49, Appendix F], who provided a preliminary definition in the context of the Babylon chain. Their work examines how accountability might extend to liveness violations in *synchronous* PoS protocols, and establishes a theorem analogous to our Thm. 2 for their framework (and for SMR rather than atomic broadcast protocols). While the technical details of their proof necessarily differ from our proof of Thm. 2, the underlying reason both theorems hold is the same: if a *certificate* implicating adversarial nodes can be constructed by an honest minority in any instance of an attack on liveness carried out by an adversarial majority, then the adversary is also able to construct certificates implicating honest parties, leading to a failure of soundness.

An informal notion of accountable liveness appears in the Pod Network project [2], but only in the setting of partial ordering consensus—and it remains documented solely in a blog post rather than in their formal publication [5]. In contrast, we provide a thorough analysis of accountable liveness in the context of total ordering consensus, establishing both the necessary and sufficient conditions for realizing it in our x-partially-synchronous network model.

Ethereum [1] addresses liveness violations in its consensus protocol, Gasper [15]—which combines LMD-GHOST for chain selection and Casper FFG [14] for finality—through *inactivity leaks* [14], a mechanism that gradually reduces the stake of non-participating nodes. The goal is that the remaining active nodes can eventually form a supermajority of the *effective* stake, restoring finality despite prolonged outages. This approach lacks a formal specification, and its guarantees remain unclear. The approach should be seen as a heuristic solution rather than a rigorously defined mechanism.

Timing Models. In addition to the classical synchronous and partially-synchronous network models [17, 23–25], recent work has introduced *granular synchrony* [26], a model that represents the network as a graph with communication links spanning fully synchronous, partially synchronous, and asynchronous behaviors.

This approach bridges the gap between classical assumptions by capturing the heterogeneous and dynamic nature of real-world networks. Notably, granular synchrony unifies existing timing models as specific instances within its broader framework. Lewis-Pye and Roughgarden [34] propose a timing model that serves as an interpolation between the synchronous and partially-synchronous settings. In addition to the standard parameters of the partially-synchronous model, i.e., Δ and GST, they introduce an additional parameter, Δ^*, with $\Delta^* \geq \Delta$ and may or may not bound message delays *before* GST. Our x-partial-synchrony model also provides a (different) interpolation between synchrony and partial synchrony, and may be of independent interest in the context of the recent interest in more fine-grained network models.

For further discussion of additional related works, see Sec. E, where we discuss papers on accountable safety, slashing, recovery procedures, and responsive and network-adaptive protocols.

8 Discussion & Conclusion

In this paper, we introduced the notion of accountability for liveness in the context of atomic broadcast. By proposing the x-partially-synchronous network model, we demonstrated how to identify adversarial protocol violators using the ψ function, which combines blame accounting (Sec. 4) with an adjudication rule (Sec. 5). We also proved that accountable liveness becomes impossible to achieve if the network model or adversary strength deviate beyond certain thresholds (Sec. 6)—underscoring the intrinsic trade-offs and additional assumptions required for liveness accountability.

Beyond these theoretical foundations, our work serves as a starting point for automating responses to liveness attacks in blockchains. Notably, Ethereum already implements an automated response to major liveness issues. Ethereum's consensus protocol, Gasper, addresses liveness violations through a mechanism called *inactivity leaks*, originally introduced to prevent Casper FFG [14] from stalling indefinitely when more than one-third of nodes fail to participate. The techniques presented in this paper provide a complementary and more general approach: our methods enable the detection and formal identification of adversarial nodes through certificates of guilt. Once such a liveness failure is detected, offending nodes can be pinpointed and slashed—an approach akin to Ethereum's inactivity leaks but with precise and stronger accountability guarantees.

Acknowledgments

References

[1] 2024. Ethereum.org: The complete guide to Ethereum. https://ethereum.org/
[2] 2025. pod – how it works. https://pod.network/how-it-works
[3] Ittai Abraham, Dahlia Malkhi, Kartik Nayak, Ling Ren, and Maofan Yin. 2020. Sync HotStuff: Simple and Practical Synchronous State Machine Replication. In *SP*. IEEE, 106–118.
[4] Bowen Alpern and Fred B. Schneider. 1985. Defining Liveness. *Inf. Process. Lett.* 21, 4 (1985), 181–185.
[5] Orestis Alpos, Bernardo David, Jakov Mitrovski, Odysseas Sofikitis, and Dionysis Zindros. 2025. Pod: An Optimal-Latency, Censorship-Free, and Accountable Generalized Consensus Layer. arXiv:2501.14931v2 [cs.DC]
[6] Sarah Azouvi and Marko Vukolic. 2022. Pikachu: Securing PoS Blockchains from Long-Range Attacks by Checkpointing into Bitcoin PoW using Taproot. In *ConsensusDay@CCS*. ACM, 53–65.

[7] Erica Blum, Jonathan Katz, and Julian Loss. 2019. Synchronous Consensus with Optimal Asynchronous Fallback Guarantees. In *TCC (1) (Lecture Notes in Computer Science, Vol. 11891)*. Springer, 131–150.

[8] Erica Blum, Jonathan Katz, and Julian Loss. 2020. Network-Agnostic State Machine Replication. arXiv:2002.03437v3 [cs.CR]

[9] Erica Blum, Chen-Da Liu Zhang, and Julian Loss. 2020. Always Have a Backup Plan: Fully Secure Synchronous MPC with Asynchronous Fallback. In *CRYPTO (2) (Lecture Notes in Computer Science, Vol. 12171)*. Springer, 707–731.

[10] Ethan Buchman, Rachid Guerraoui, Jovan Komatovic, Zarko Milosevic, Dragos-Adrian Seredinschi, and Josef Widder. 2022. Revisiting Tendermint: Design Tradeoffs, Accountability, and Practical Use. In *DSN (Supplements)*. IEEE, 11–14.

[11] Ethan Buchman, Jae Kwon, and Zarko Milosevic. 2018. The latest gossip on BFT consensus. arXiv:1807.04938v3 [cs.DC]

[12] Eric Budish, Andrew Lewis-Pye, and Tim Roughgarden. 2024. The Economic Limits of Permissionless Consensus. In *EC*. ACM, 704–731.

[13] Vitalik Buterin. 2018. *A Guide to 99% Fault Tolerant Consensus*. https://vitalik.eth.limo/general/2018/08/07/99_fault_tolerant.html

[14] Vitalik Buterin and Virgil Griffith. 2017. Casper the Friendly Finality Gadget. arXiv:1710.09437v4 [cs.CR]

[15] Vitalik Buterin, Diego Hernandez, Thor Kamphefner, Khiem Pham, Zhi Qiao, Danny Ryan, Juhyeok Sin, Ying Wang, and Yan X Zhang. 2020. Combining GHOST and Casper. arXiv:2003.03052v3 [cs.CR]

[16] Christian Cachin, Rachid Guerraoui, and Luís E. T. Rodrigues. 2011. *Introduction to Reliable and Secure Distributed Programming (2. ed.)*. Springer.

[17] Miguel Castro and Barbara Liskov. 1999. Practical Byzantine Fault Tolerance. In *OSDI*. USENIX Association, 173–186.

[18] Benjamin Y. Chan and Elaine Shi. 2020. Streamlet: Textbook Streamlined Blockchains. In *AFT*. ACM, 1–11.

[19] Pierre Civit, Seth Gilbert, and Vincent Gramoli. 2021. Polygraph: Accountable Byzantine Agreement. In *ICDCS*. IEEE, 403–413.

[20] Pierre Civit, Seth Gilbert, Vincent Gramoli, Rachid Guerraoui, and Jovan Komatovic. 2023. As easy as ABC: Optimal (A)ccountable (B)yzantine (C)onsensus is easy! *J. Parallel Distributed Comput.* 181 (2023), 104743.

[21] Francesco D'Amato. 2022. *Casper FFG as a full protocol and its relationship with Streamlet*. https://ethresear.ch/t/casper-ffg-as-a-full-protocol-and-its-relationship-with-streamlet/13803

[22] Evangelos Deirmentzoglou, Georgios Papakyriakopoulos, and Constantinos Patsakis. 2019. A Survey on Long-Range Attacks for Proof of Stake Protocols. *IEEE Access* 7 (2019), 28712–28725.

[23] Danny Dolev and H. Raymond Strong. 1983. Authenticated Algorithms for Byzantine Agreement. *SIAM J. Comput.* 12, 4 (1983), 656–666.

[24] Cynthia Dwork, Nancy A. Lynch, and Larry J. Stockmeyer. 1988. Consensus in the presence of partial synchrony. *J. ACM* 35, 2 (1988), 288–323.

[25] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. 1985. Impossibility of Distributed Consensus with One Faulty Process. *J. ACM* 32, 2 (1985), 374–382.

[26] Neil Giridharan, Ittai Abraham, Natacha Crooks, Kartik Nayak, and Ling Ren. 2024. Granular Synchrony. In *DISC (LIPIcs, Vol. 319)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 30:1–30:22.

[27] Tiantian Gong, Gustavo Franco Camilo, Kartik Nayak, Andrew Lewis-Pye, and Aniket Kate. 2025. Recover from Excessive Faults in Partially-Synchronous BFT SMR. Cryptology ePrint Archive, Paper 2025/083. https://eprint.iacr.org/2025/083

[28] Andreas Haeberlen, Petr Kouznetsov, and Peter Druschel. 2007. PeerReview: practical accountability for distributed systems. In *SOSP*. ACM, 175–188.

[29] Andreas Haeberlen and Petr Kuznetsov. 2009. The Fault Detection Problem. In *OPODIS (Lecture Notes in Computer Science, Vol. 5923)*. Springer, 99–114.

[30] Ruomu Hou and Haifeng Yu. 2023. Optimistic Fast Confirmation While Tolerating Malicious Majority in Blockchains. In *SP*. IEEE, 2481–2498.

[31] Ruomu Hou, Haifeng Yu, and Prateek Saxena. 2022. Using Throughput-Centric Byzantine Broadcast to Tolerate Malicious Majority in Blockchains. In *SP*. IEEE, 1263–1280.

[32] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. 2017. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In *CRYPTO (1) (Lecture Notes in Computer Science, Vol. 10401)*. Springer, 357–388.

[33] Andrew Lewis-Pye and Tim Roughgarden. 2023. Permissionless Consensus. arXiv:2304.14701v5 [cs.DC]

[34] Andrew Lewis-Pye and Tim Roughgarden. 2025. Beyond Optimal Fault Tolerance. arXiv:2501.06044v4 [cs.DC]

[35] Atsuki Momose and Ling Ren. 2021. Multi-Threshold Byzantine Fault Tolerance. In *CCS*. ACM, 1686–1699.

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

[37] Joachim Neu, Ertem Nusret Tas, and David Tse. 2020. Snap-and-Chat Protocols: System Aspects. arXiv:2010.10447v1 [cs.CR]

[38] Joachim Neu, Ertem Nusret Tas, and David Tse. 2022. The Availability-Accountability Dilemma and Its Resolution via Accountability Gadgets. In *Financial Cryptography (Lecture Notes in Computer Science, Vol. 13411)*. Springer, 541–559.

[39] Joachim Neu, Ertem Nusret Tas, and David Tse. 2024. Short Paper: Accountable Safety Implies Finality. In *FC (1) (Lecture Notes in Computer Science, Vol. 14744)*. Springer, 41–50.

[40] Rafael Pass and Elaine Shi. 2017. The Sleepy Model of Consensus. In *ASIACRYPT (2) (Lecture Notes in Computer Science, Vol. 10625)*. Springer, 380–409.

[41] Rafael Pass and Elaine Shi. 2018. Thunderella: Blockchains with Optimistic Instant Confirmation. In *EUROCRYPT (2) (Lecture Notes in Computer Science, Vol. 10821)*. Springer, 3–33.

[42] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. 1980. Reaching Agreement in the Presence of Faults. *J. ACM* 27, 2 (1980), 228–234.

[43] Alejandro Ranchal-Pedrosa and Vincent Gramoli. 2024. ZLB: A Blockchain to Tolerate Colluding Majorities. In *DSN*. IEEE, 209–222.

[44] Alex Shamis, Peter R. Pietzuch, Burcu Canakci, Miguel Castro, Cédric Fournet, Edward Ashton, Amaury Chamayou, Sylvan Clebsch, Antoine Delignat-Lavaud, Matthew Kerner, Julien Maffre, Olga Vrousgou, Christoph M. Wintersteiger, Manuel Costa, and Mark Russinovich. 2022. IA-CCF: Individual Accountability for Permissioned Ledgers. In *NSDI*. USENIX Association, 467–491.

[45] Peiyao Sheng, Gerui Wang, Kartik Nayak, Sreeram Kannan, and Pramod Viswanath. 2021. BFT Protocol Forensics. In *CCS*. ACM, 1722–1743.

[46] Elaine Shi. 2020. Foundations of Distributed Consensus and Blockchains. https://www.distributedconsensus.net Book manuscript.

[47] Srivatsan Sridhar, Ertem Nusret Tas, Joachim Neu, Dionysis Zindros, and David Tse. 2024. Consensus Under Adversary Majority Done Right. Cryptology ePrint Archive, Paper 2024/1799. https://eprint.iacr.org/2024/1799

[48] Srivatsan Sridhar, Dionysis Zindros, and David Tse. 2023. Better Safe than Sorry: Recovering after Adversarial Majority. arXiv:2310.06338v2 [cs.CR]

[49] Ertem Nusret Tas, David Tse, Fangyu Gai, Sreeram Kannan, Mohammad Ali Maddah-Ali, and Fisher Yu. 2023. Bitcoin-Enhanced Proof-of-Stake Security: Possibilities and Impossibilities. In *SP*. IEEE, 126–145.

[50] Ertem Nusret Tas, David Tse, Fisher Yu, and Sreeram Kannan. 2022. Babylon: Reusing Bitcoin Mining to Enhance Proof-of-Stake Security. arXiv:2201.07946v1 [cs.CR]

[51] Maofan Yin, Dahlia Malkhi, Michael K. Reiter, Guy Golan-Gueta, and Ittai Abraham. 2019. HotStuff: BFT Consensus with Linearity and Responsiveness. In *PODC*. ACM, 347–356.

A Addendum Consensus Protocol

Remark 3. Note that $\text{valid}(\mathfrak{M}, Q, b, s)$ implies $\text{valid}(\mathfrak{M}, b)$ because $\text{valid}(\mathfrak{M}, w)$ implies $\text{valid}(\mathfrak{M}, w.b)$.

Remark 4. Note that, through the parent block relation in Alg. 1, ln. 1, valid blocks form a tree rooted at the genesis block b_0. As a result, the path from b_0 to any valid block b exists and is unique. Block b is an *ancestor* of block b', denoted as $b \preceq b'$, iff b is on the path from b_0 to b'. Ancestors of any valid block are from strictly increasing views.

Remark 5. Assuming $f \leq 2n/3$ (which holds in particular if we assume $n > 3f$), by Alg. 1, ln. 3, an honest node's vote is required to form any valid quorum certificate (QC). Thus, by Alg. 1, for any view v, no honest node can see any valid QC for v before $12\Delta v + 4\Delta$. It follows that in every honest node p's view, $\mathsf{v}(\mathfrak{Q}_t^p)$ as a function of t is non-decreasing. To see this, suppose $t' = 12\Delta v' + 7\Delta$ for view v' is a time where p updates $\mathfrak{Q}^p$ such that $v'' \triangleq \mathsf{v}(\mathfrak{Q}_t^p) > \mathsf{v}(\mathfrak{Q}_{t'^+}^p) = v'$ for some time t with $t < t'^+$. Then p saw a valid QC for view v'' at time $t < 12\Delta v' + 7\Delta < 12\Delta v'' + 4\Delta$, a contradiction to the aforementioned.

A.1 Proof of Lem. 1

PROOF OF LEM. 1. Towards a contradiction, suppose honest p confirms b (*i.e.*, p outputs as its log Λ_t^p the sequence of transactions on the path from b_0 to b at some time t, as per Alg. 1, ln. 8), and honest p' confirms b', but neither $b \preceq b'$ nor $b' \preceq b$. Without loss of generality, assume $b.\mathsf{v} \leq b'.\mathsf{v}$. Let b'' be the block such that $b'' \preceq b'$ and $b''.\mathsf{v}$ is minimal, subject to the constraint $b''.\mathsf{v} \geq b.\mathsf{v}$. Note that b'' exists because b' is a candidate, and b'' is unique due to Rem. 4. Note that $b'' \neq b$ because otherwise $b \preceq b'$, which would be a

contradiction to earlier assumptions. Note that because p confirms b, $\exists Q_1, Q_2 : \text{valid}(\mathfrak{M}_\infty^\cup, Q_1, b, 1) \land \text{valid}(\mathfrak{M}_\infty^\cup, Q_2, b, 2)$. Because p' confirms b', p' views b' as valid, and thus all its ancestors have valid stage-1 QCs, *i.e.*, $\exists Q_3 : \text{valid}(\mathfrak{M}_\infty^\cup, Q_3, b'', 1)$.

Suppose $b''.v = b.v$. Due to the assumption $n > 3f$, and quorum intersection, some honest node has contributed to both Q_1 and Q_3, a contradiction because according to Alg. 1 honest nodes send only one stage-1 vote per view.

Suppose $b''.v > b.v$. Let $Q_0 \triangleq b''.Q$. Note that $v(Q_0) < b''.v$ due to Rem. 4. Note that it must be that $v(Q_0) < b.v$ because b'' was chosen with minimal $b''.v$ (subject to $b''.v \geq b.v$). Due to the assumption $n > 3f$, and quorum intersection, some honest node p has contributed to both Q_2 and Q_3. Thus, $v(\mathfrak{Q}_{12\Delta(b.v+1)}^p) = b.v$, by Alg. 1, ln. 20. Yet, p allegedly stage-1 votes for b'' in a later $b''.v \geq b.v+1$, *i.e.*, after $12\Delta b''.v \geq 12\Delta(b.v+1)$, specifically at $12\Delta b''.v+4\Delta$. This is even though $v(b''.Q) = v(Q_0) < b.v = v(\mathfrak{Q}_{12\Delta(b.v+1)}^p) \leq v(\mathfrak{Q}_{12\Delta b''.v}^p) \leq v(\mathfrak{Q}_{12\Delta b''.v+4\Delta}^p)$ (recall that by Rem. 5, $v(\mathfrak{Q}_t^p)$ is non-decreasing in t), a contradiction to Alg. 1, ln. 16. □

A.2 Proof of Lem. 2

PROOF OF LEM. 2. Suppose the leader L_v is honest for some view v with $12\Delta v \geq \text{GST}$. Because the network is synchronous, $\mathfrak{M}_{12\Delta v+2\Delta}^{L_v} \supseteq \mathfrak{M}_{12\Delta v}^p$ for any honest node p. Furthermore, honest nodes do not update their $\mathfrak{Q}$ in the time frame from $(12\Delta(v-1)+7\Delta)^+$ to $(12\Delta v + 7\Delta)^-$, and thus also not in the time frame from $12\Delta v$ to $12\Delta v + 4\Delta$. As a result, Q chosen by L_v at $12\Delta v + 2\Delta$ in Alg. 1, ln. 12 is such that for every honest node p, $v(\mathfrak{Q}_{12\Delta v+4\Delta}^p) \leq v(Q)$ in Alg. 1, ln. 16. Furthermore, by synchrony, $\mathfrak{M}_{12\Delta v+4\Delta}^p \supseteq \mathfrak{M}_{12\Delta v+2\Delta}^{L_v}$. Therefore, b, Q chosen by L_v at $12\Delta v + 2\Delta$ in Alg. 1, ln. 12 are viewed as valid by all honest nodes at $12\Delta v + 4\Delta$, and so is the resulting block b^* produced by L_v at $12\Delta v+2\Delta$ in Alg. 1, ln. 13, which by synchrony all honest nodes receive by $12\Delta v+4\Delta$. Thus, the condition of Alg. 1, ln. 15 is satisfied in all honest views, and all honest nodes will stage-1 vote for b^* in Alg. 1, ln. 17. By synchrony, all honest votes propagate to all honest nodes by $12\Delta v + 7\Delta$. Thus, the condition of Alg. 1, ln. 19 is satisfied in all honest views. Furthermore, since L_v is honest and produces only one block in view v, b^* is the only block that satisfies Alg. 1, ln. 19 in any honest view. As a result, all honest nodes will stage-2 vote for b^* at $12\Delta v + 7\Delta$. Again by synchrony, these votes will propagate to all honest nodes by $12\Delta v + 9\Delta$, who will thus confirm b^* and with it all pending transactions that any honest node has seen and re-broadcast by $12\Delta v + \Delta$, and which thus (by synchrony) the honest L_v included in b^* at $12\Delta v + 2\Delta$. □

B Addendum Blame Accounting

B.1 Proof of Lem. 3

PROOF OF LEM. 3. Assume the network is synchronous for view v and L_v is honest and proposes b. We consider two cases: (A) Some honest node does not view b as confirmed by $12\Delta v + 9\Delta$. (B) Every honest node views b as confirmed by $12\Delta v + 9\Delta$, but some honest nodes does not see VoteLive messages for v from $> 2n/3$ nodes by $12\Delta v + 12\Delta$. We show that in both cases there is a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$ such that for every honest node p, for all $p \in \mathcal{P}'$, $\text{Blame}_{p,v,p'} = 1$ in Alg. 2.

First, case (A). Let q be an honest node that does not view b as confirmed by $12\Delta v + 9\Delta$. Since the network is synchronous and L_v is honest, the condition of Alg. 2, ln. 4 is met for every honest node p. The condition of Alg. 2, ln. 5 must be met for every honest node because L_v is honest and chooses (b, Q) according to Alg. 1, ln. 12, and the network is synchronous. Since q does not view b as confirmed by $12\Delta v + 9\Delta$, but b is viewed as valid by every honest node after $12\Delta v+3\Delta$ due to network synchrony and L_v being honest, either (a) there must be a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$ such that q has not seen a valid stage-1 vote for b from any $p' \in \mathcal{P}'$ by $12\Delta v + 9\Delta$, and thus no honest node has seen a valid stage-1 vote for b from any $p' \in \mathcal{P}'$ by $12\Delta v + 8\Delta$ (and thus also not by $12\Delta v + 6\Delta$ or by $12\Delta v + 5\Delta$), or (b) there must be a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$ such that q has not seen a valid stage-2 vote for b from any $p' \in \mathcal{P}'$ by $12\Delta v + 9\Delta$, and thus no honest node has seen a valid stage-2 vote for b from any $p' \in \mathcal{P}'$ by $12\Delta v + 8\Delta$.

Suppose (a) holds. Then, since L_v is honest and b is its unique proposal for view v, the condition of Alg. 2, ln. 6 is met for every honest node p for all $p' \in \mathcal{P}'$. Then, for every honest node p, $\text{Blame}_{p,v,p'} \leftarrow 1$ in Alg. 2, ln. 7 for all those $p' \in \mathcal{P}'$, as desired.

Suppose (b) holds. Suppose there is some honest node r for which the condition of Alg. 2, ln. 8 is not satisfied. Then, by synchrony and since L_v is honest, there must be a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$ such that no honest node has seen a valid stage-1 vote for b from any $p' \in \mathcal{P}'$ by $12\Delta v + 5\Delta$, and we are done by the same argument as for (a). We thus assume that the condition of Alg. 2, ln. 8 holds for every honest node. Then, by (b), there must be a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$ such that the condition of Alg. 2, ln. 9 is met for all $p' \in \mathcal{P}'$ for every honest node p, so that $\text{Blame}_{p,v,p'} \leftarrow 1$ in Alg. 2, ln. 10 for all $p' \in \mathcal{P}'$, as desired.

Now consider case (B). Suppose q is an honest node that does not see VoteLive messages for v from $> 2n/3$ nodes by $12\Delta v + 12\Delta$, but all honest nodes view b as confirmed by $12\Delta v + 9\Delta$. By synchrony and since L_v is honest and since all honest nodes view b as confirmed by $12\Delta v + 9\Delta$ by assumption, for every honest node p, $\mathfrak{M}t_{12\Delta v+1}^p \subseteq \mathfrak{M}t_{12\Delta v+2}^{L_v} \subseteq \Lambda_{12\Delta v+9\Delta}^p$. (Note that a node r can determine the confirmed output log of another node s as per any round t based on the transcript of s.) Thus, Alg. 2, ln. 11 is met for every honest node p. Since q does not see VoteLive messages for v from $> 2n/3$ nodes by $12\Delta v + 12\Delta$, by synchrony, there must be a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq n/3$ such that no honest node has seen a VoteLive messages for v from any $p' \in \mathcal{P}'$ by $12\Delta v + 11\Delta$. Thus, for every honest node p and every $p' \in \mathcal{P}'$, the condition of Alg. 2, ln. 12 holds, so that $\text{Blame}_{p,v,p'} \leftarrow 1$ in Alg. 2, ln. 13, as desired. □

B.2 Proof of Lem. 4

PROOF OF LEM. 4. Suppose, for contradiction, that the network is synchronous for view v but for some honest p and for some honest p', $\text{Blame}_{p,v,p'} = 1$ in Alg. 2. Then, $\text{Blame}_{p,v,p'} \leftarrow 1$ either (a) in Alg. 2, ln. 7, (b) in Alg. 2, ln. 10, or (c) in Alg. 2, ln. 13.

Suppose (a) holds. Then, it must be that the conditions of Alg. 2, ln. 4 and Alg. 2, ln. 5 are met for p. But then, by the assumption of synchrony, in the view of p', the conditions of Alg. 1, ln. 15 and Alg. 1, ln. 16 were met, so that p' cast a valid stage-1 vote for a block proposed in view v, and thus, again by synchrony, the condition of

Alg. 2, ln. 6 cannot be met for p, a contradiction to Blame$_{p,v,p'} \leftarrow 1$ in Alg. 2, ln. 7.

Suppose (b) holds. Then, it must be that the condition of Alg. 2, ln. 8 is met for p. But then, by the assumption of synchrony, the condition of Alg. 1, ln. 19 was met in the view of p', so that p' cast a valid stage-2 vote for a valid block b by $12\Delta v + 7\Delta$, which, again by synchrony, p must have seen by $12\Delta v + 8\Delta$, so that the condition of Alg. 2, ln. 9 cannot be met for p, a contradiction to Blame$_{p,v,p'} \leftarrow 1$ in Alg. 2, ln. 10.

Suppose (c) holds. Then, it must be that the condition of Alg. 2, ln. 11 is met for p. But then, by the assumption of synchrony, $\mathfrak{Mt}_{12\Delta v}^{p'} \subseteq \mathfrak{Mt}_{12\Delta v+1}^{p} \subseteq \Lambda_{12\Delta v+9\Delta}^{p} \subseteq \Lambda_{12\Delta v+10\Delta}^{p'}$, so that the condition of Alg. 1, ln. 23 was met in the view of p', and p' cast a VoteLive vote for view v by $12\Delta v + 10\Delta$, which, again by synchrony, p must have seen by $12\Delta v + 11\Delta$, so that the condition of Alg. 2, ln. 12 cannot be met for p, a contradiction to Blame$_{p,v,p'} \leftarrow 1$ in Alg. 2, ln. 13. $\square$

C Addendum Adjudication Rule

A well-known Chernoff bound is used in the proof of Lem. 7:

Proposition 1 (Chernoff bound). *Let $Z_1, ..., Z_m$ be independent Bernoulli random variables with $p_i \triangleq \Pr[Z_i = 1]$. Let $\mu \triangleq \mathrm{E}\left[\sum_{i=1}^{m} Z_i\right]$. Then, $\forall c \geq 0$:*

$$\Pr\left[\left(\sum_{i=1}^{m} Z_i\right) \geq (1+c)\mu\right] \leq \exp\left(\frac{-c^2\mu}{2+c}\right). \quad (5)$$

A counting lemma is used in the proof of Lem. 10:

Lemma 11. *Let Ω be some set with $|\Omega| < \infty$, $h : \Omega \to \{0, ..., m\}$ be some function, for some $m \in \mathbb{N}$, and $c < \frac{1}{|\Omega|}\sum_{\omega\in\Omega} h(\omega) \triangleq \mu$. Then, $|\{\omega \mid h(\omega) \leq c\}| \leq \lfloor |\Omega|\frac{m-\mu}{m-c}\rfloor$.*

PROOF. Let $\Omega_- \triangleq \{\omega \mid h(\omega) \leq c\}$, $\Omega_+ \triangleq \{\omega \mid h(\omega) > c\}$. Then, $\mu = \frac{1}{|\Omega|}\sum_{\omega\in\Omega} h(\omega) = \frac{1}{|\Omega|}\left(\sum_{\omega\in\Omega_-} h(\omega) + \sum_{\omega\in\Omega_+} h(\omega)\right)$. Since $h(\omega) \leq c$ for all $\omega \in \Omega_-$, and $h(\omega) \leq m$ for all $\omega \in \Omega_+$, and also $|\Omega| = |\Omega_-| + |\Omega_+|$, $|\Omega|\mu \leq |\Omega_-|c + (|\Omega| - |\Omega_-|)m$. So, rearranging, $|\Omega|\mu - |\Omega|m \leq |\Omega_-|(c - m)$. Finally, multiplying both sides by (-1), and dividing, $|\Omega|\frac{m-\mu}{m-c} \geq |\Omega_-|$. Since $|\Omega_-| \in \mathbb{N}$, $|\{\omega \mid h(\omega) \leq c\}| = |\Omega_-| \leq \lfloor|\Omega|\frac{m-\mu}{m-c}\rfloor$. $\square$

C.1 Proof of Lem. 7

PROOF OF LEM. 7. For each super-view $u \in \mathcal{U}$, let Z_u be the random variable with $Z_u = 0$ if $\exists v \in \mathcal{V}_u : L_v \in \mathcal{P}_h$, and $Z_u = 1$ otherwise, i.e., if $\forall v \in \mathcal{V}_u : L_v \in \mathcal{P}_a$. Since $|\mathcal{P}_a| = f \leq \tau_{\max}^{AL} < n/2$, and leaders per view are chosen independently and uniformly at random, and each super-view has K_{views} views, $\mathrm{E}[Z_u] = \Pr[Z_u = 1] \leq 2^{-K_{views}}$. Then, from Prop. 1,

$$\Pr\left[\left(\sum_{u\in\mathcal{U}} Z_i\right) \geq 2 \cdot 2^{-K_{views}}g(\Delta')\right] \leq \exp\left(-2^{-K_{views}}g(\Delta')/3\right). \quad (6)$$

For any target $\delta_x > 0$, by choosing $K_{views} = \lceil\log_2(\frac{2}{\delta_x})\rceil$, except with probability $\exp(-\delta_x g(\Delta')/6)$, at most δ_x fraction of super-views in $\mathcal{U}$ will not have *some* honest leader: $|\mathcal{U}_{hl}^c| \leq \delta_x|\mathcal{U}|$.

Furthermore, due to (Δ, g, x)-partial-synchrony and our choice of Δ', we get $|\mathcal{U}_s^c| \leq x|\mathcal{U}|$. Thus, from a union bound, $|\mathcal{U}_{shl}| = |\mathcal{U}_s \cap \mathcal{U}_{hl}| \geq |\mathcal{U}| - |\mathcal{U}_s^c| - |\mathcal{U}_{hl}^c| = (1 - x - \delta_x)|\mathcal{U}|$. $\square$

C.2 Proof of Lem. 8

PROOF OF LEM. 8. Let $u \in \mathcal{U}_{shl}$. Assuming the blame counts are based on all honest transcripts (which they are, as we have argued at the beginning of Sec. 4 and in Fig. 2), and using the assumption that some honest node took note of a potential timely-liveness violation, and using $|\mathcal{P}_a| \leq \tau_{\max}^{AL}$, so that $|\mathcal{P}_h| \geq n - \tau_{\max}^{AL}$, by Lem. 5, $|\mathcal{P}_u^A| \geq n/3$.

Let $u, u' \in \mathcal{U}_{shl}$, so per the above, $|\mathcal{P}_u^A| \geq n/3$, $|\mathcal{P}_{u'}^A| \geq n/3$. Due to Lem. 6, $\mathcal{P}_u^A \subseteq \mathcal{P}_a$, $\mathcal{P}_{u'}^A \subseteq \mathcal{P}_a$. Since $|\mathcal{P}_a| \leq \tau_{\max}^{AL}$, by an intersection argument, $|\mathcal{P}_u^A \cap \mathcal{P}_{u'}^A| \geq 2n/3 - \tau_{\max}^{AL}$.

Since any two $u, u' \in \mathcal{U}_{shl}$ have an edge in G of Alg. 4 per the above, and $|\mathcal{U}_{shl}| \geq (1 - x - \delta_x)|\mathcal{U}| > (x + \delta_x)|\mathcal{U}|$ due to x-partial-synchrony and $1 - x - \delta_x > x + \delta_x$ by assumption, for every $u \in \mathcal{U}_{shl}$, $\deg_G(u) > (x + \delta_x)|\mathcal{U}|$.

Thus, for every $u \in \mathcal{U}_{shl}$, $|\mathcal{P}_u^A| \geq n/3$ and $\deg_G(u) > (x+\delta_x)|\mathcal{U}|$, so $u \in \mathcal{U}'$, as desired. $\square$

C.3 Proof of Lem. 9

PROOF OF LEM. 9. Recall that $\mathcal{U}_{shl} \subseteq \mathcal{U}'$ by Lem. 8, $\mathcal{U}' \subseteq \mathcal{U}$ by definition in Alg. 4, and $|\mathcal{U}_{shl}| \geq (1-x-\delta_x)|\mathcal{U}|$. Thus, $|\mathcal{U}'\cap\mathcal{U}_{shl}| = |\mathcal{U}_{shl}| \geq (1 - x - \delta_x)|\mathcal{U}| \geq (1 - x - \delta_x)|\mathcal{U}'|$, i.e., $\mathcal{U}_{shl}$ makes up at least $(1 - x - \delta_x)$ fraction of $\mathcal{U}'$, i.e., at most $x + \delta_x$ fraction of $\mathcal{U}'$ are not in $\mathcal{U}_{shl}$.

Suppose for contradiction that $p \in \mathcal{P}_h$ is returned by Alg. 4. Then there is a subset $\mathcal{U}''$ of $\mathcal{U}'$ such that $p \in \mathcal{P}_u^A$ for all $u \in \mathcal{U}''$ and $|\mathcal{U}''| > (x + \delta_x)|\mathcal{U}'|$. From the latter, there must be $u \in \mathcal{U}'' \cap \mathcal{U}_{shl}$ so that $p \in \mathcal{P}_u^A$, i.e., p is blamed by at least $n - \tau_{\max}^{AL}$ nodes in u, but, together with $\tau_{\max}^{AL} < n/2$, Lem. 6 rules such $u \in \mathcal{U}_{shl}$ out. This is the desired contradiction. $\square$

C.4 Proof of Lem. 10

PROOF OF LEM. 10. Consider a $|\mathcal{P}_a| \times |\mathcal{U}'|$ matrix M with rows indexed by $p \in \mathcal{P}_a$ and columns indexed by $u \in \mathcal{U}'$ of Alg. 4. Let $M_{p,u} = 1$ if $p \in \mathcal{P}_u^A$, and 0 otherwise. Recall that $\mathcal{U}_{shl} \subseteq \mathcal{U}'$ (Lem. 8). Consider $u \in \mathcal{U}' \cap \mathcal{U}_{shl} = \mathcal{U}_{shl}$. By Lems. 5 and 6, $\sum_{p\in\mathcal{P}_a} M_{p,u} \geq n/3$. Now consider $u \in \mathcal{U}' \cap \mathcal{U}_{shl}^c$. By definition of $\mathcal{U}'$ in Alg. 4, $\deg_G(u) > (x + \delta_x)|\mathcal{U}|$. Since $\mathcal{U}_{shl}$ makes up at least $(1 - x - \delta_x)$ fraction of $\mathcal{U}'$ (proof of Lem. 9), i.e., at most $x+\delta_x$ fraction of $\mathcal{U}'$ are not in $\mathcal{U}_{shl}$, there is some $u' \in \mathcal{U}_{shl}$ such that $\{u, u'\} \in G$ and thus $|\mathcal{P}_u^A \cap \mathcal{P}_{u'}^A| \geq 2n/3 - \tau_{\max}^{AL}$. By Lem. 6, $\mathcal{P}_{u'}^A \subseteq \mathcal{P}_a$, thus $\mathcal{P}_u^A \cap \mathcal{P}_{u'}^A \subseteq \mathcal{P}_a$, thus $|\mathcal{P}_u^A \cap \mathcal{P}_a| \geq 2n/3 - \tau_{\max}^{AL}$, thus $\sum_{p\in\mathcal{P}_a} M_{p,u} \geq 2n/3 - \tau_{\max}^{AL}$.

Since $|\mathcal{U}' \cap \mathcal{U}_{shl}| \geq (1 - x - \delta_x)|\mathcal{U}'|$ (proof of Lem. 9), and thus $|\mathcal{U}' \cap \mathcal{U}_{shl}^c| \leq (x + \delta_x)|\mathcal{U}'|$, and $2n/3 - \tau_{\max}^{AL} \leq n/3$ due to $n/3 \leq \tau_{\max}^{AL} < n/2$, $\sum_{p\in\mathcal{P}_a}\sum_{u\in\mathcal{U}'} M_{p,u} \geq (1-x-\delta_x)|\mathcal{U}'|n/3 + (x + \delta_x)|\mathcal{U}'|(2n/3 - \tau_{\max}^{AL}) = |\mathcal{U}'|(n/3 + (x + \delta_x)(n/3 - \tau_{\max}^{AL}))$.

Since there are $f = |\mathcal{P}_a|$ rows in M, the average row weight μ of M is $\mu \triangleq |\mathcal{U}'|(n/3 + (x + \delta_x)(n/3 - \tau_{\max}^{AL}))/f$. Note that for every $f \leq \tau_{\max}^{AL}$, $\mu > (x + \delta_x)|\mathcal{U}'|$, since $x + \delta_x < 1/2$. Consider the function $h : \mathcal{P}_a \to \{0, ..., |\mathcal{U}'|\}, p \mapsto \sum_{u\in\mathcal{U}'} M_{p,u}$. Using Lem. 11, $|\{p \mid h(p) \leq (x + \delta_x)|\mathcal{U}'|\}| \leq \left\lfloor\frac{f - n/3 - (x+\delta_x)(n/3 - \tau_{\max}^{AL})}{1 - x - \delta_x}\right\rfloor$.

Thus, the number of rows in M with weight more than $(x + \delta_x)|\mathcal{U}'|$ is at least $f - \left\lfloor\frac{f - n/3 - (x+\delta_x)(n/3 - \tau_{\max}^{AL})}{1 - x - \delta_x}\right\rfloor$. The p corresponding to these rows satisfy the return condition of Alg. 4, $\exists \mathcal{U}'' \subseteq \mathcal{U}' : (\forall u \in \mathcal{U}'' : p \in \mathcal{P}_u^A) \wedge (|\mathcal{U}''| > (x + \delta_x)|\mathcal{U}'|)$, where we recall that $M_{p,u} = 1$ if $p \in \mathcal{P}_u^A$, and 0 otherwise. $\square$

C.5 Proof of Thm. 1

PROOF OF THM. 1. *Soundness.* Suppose a certificate of guilt is produced for node p and time t, according to the process of Fig. 2 and Secs. 4 and 5. And towards a contradiction, suppose $p \in \mathcal{P}_h$. Because $f \leq \tau_{\max}^{\mathrm{AL}} < n/2$, an honest node p' has accused p. Thus, p' obtained $p \in \mathcal{P}'$ for $\mathcal{P}'$ returned by ψ. Since p' is honest, p' must have taken note of a potential timely-liveness violation in Fig. 2. But by Lem. 9, under the given circumstances, no $p \in \mathcal{P}_h$ is returned by Alg. 4. This is a contradiction, so $p \in \mathcal{P}_a$.

Completeness. Suppose an execution with a timely-liveness violation at t. Then more than $\lfloor (n-1)/3 \rfloor$ honest nodes have not voted VoteLive for any of the views in $\mathcal{V}$, and thus all honest nodes have taken note of a potential timely-liveness violation at t in Fig. 2. Thus, all honest nodes apply ψ, and by Lem. 10 and Rem. 2, and considering that $f \leq \tau_{\max}^{\mathrm{AL}}$, there is a set $\mathcal{P}'$ with $|\mathcal{P}'| \geq \tau_{\max}^{\mathrm{AL}} - \left\lfloor \frac{(1+x+\delta_x)(\tau_{\max}^{\mathrm{AL}} - n/3)}{1-x-\delta_x} \right\rfloor$ so that for every honest node, Alg. 4 as part of ψ returns a superset of $\mathcal{P}'$. All honest nodes obtain (a superset of) $\mathcal{P}'$ from ψ by time $t + \Delta' g(\Delta')$, according to the process of Fig. 2 and Secs. 4 and 5, and accuse all $\mathcal{P}'$. By $t + 2\Delta' g(\Delta')$, these accusations have propagated to all honest nodes, and since honest nodes are a majority, they have produced a certificate of guilt for $\mathcal{P}'$. □

D Addendum Impossibility Results

A key step in establishing the impossibility of accountable liveness is to obtain executions in which timely-liveness must be violated for "reasonable" protocols. In this context, a useful consequence of optimal resilience (cf. Def. 5) is that the liveness guarantee for such protocols is tight in the sense that there must exist an execution with $\tau^L + 1$ *crash faults* in which timely-liveness is violated:

Lemma 12. *If an atomic broadcast protocol is τ^S-safe under partial synchrony and either $\tau^S > 0$ or n is even, then, under synchrony, some execution with $\lceil (n-\tau^S)/2 \rceil$ crash faults violates timely-liveness.*

PROOF. The argument proceeds analogously to the classical "split-brain converse" [24]. Towards a contradiction, suppose Π satisfies the conditions of Lem. 12, *i.e.*, Π is τ^S-safe under partial synchrony, but under (Δ, g, x)-partial-synchrony Π preserves timely-liveness (in fact it suffices to consider traditional liveness) for all except at most $(\lceil (n-\tau^S)/2 \rceil - 1)$ honest nodes in all executions with $\lceil (n-\tau^S)/2 \rceil$ crash faults. Note that $\tau^L \leq (\lceil (n-\tau^S)/2 \rceil - 1)$ is the maximum liveness resilience that can be expected due to Def. 5.

Note that we can partition $\mathcal{P}$ into $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ with $|\mathcal{P}_1| = |\mathcal{P}_3| = \lceil (n-\tau^S)/2 \rceil$ and $|\mathcal{P}_2| \leq \tau^S$. Let $\mathrm{tx}_1, \mathrm{tx}_2$ be two high-entropy transactions, so that Π cannot guess them, and they cannot have been hard-coded into Π.

Execution E_1: The nodes of $\mathcal{P}_1$ crash. The nodes of $\mathcal{P}_2 \cup \mathcal{P}_3$ are honest. The network is synchronous. The environment inputs tx_1 into all nodes, and does not input tx_2 into any node. Since Π is assumed to preserve liveness for all except at most $(\lceil (n-\tau^S)/2 \rceil - 1)$ honest nodes in all executions with $\lceil (n-\tau^S)/2 \rceil$ crash faults under synchrony and $|\mathcal{P}_1| = \lceil (n-\tau^S)/2 \rceil$ and $|\mathcal{P}_3| = \lceil (n-\tau^S)/2 \rceil > \lceil (n-\tau^S)/2 \rceil - 1$, by some time t_1, some (honest) node $p_3 \in \mathcal{P}_3$ will output a log containing tx_1 and not containing tx_2.

Execution E_2: The nodes of $\mathcal{P}_3$ crash. The nodes of $\mathcal{P}_1 \cup \mathcal{P}_2$ are honest. The network is synchronous. The environment inputs tx_2 into all nodes, and does not input tx_1 into any node. Since Π is assumed to preserve liveness for all except at most $(\lceil (n-\tau^S)/2 \rceil - 1)$ honest nodes in all executions with $\lceil (n-\tau^S)/2 \rceil$ crash faults under synchrony and $|\mathcal{P}_3| = \lceil (n-\tau^S)/2 \rceil$ and $|\mathcal{P}_1| = \lceil (n-\tau^S)/2 \rceil > \lceil (n-\tau^S)/2 \rceil - 1$, by some time t_2, some (honest) node $p_1 \in \mathcal{P}_1$ will output a log containing tx_2 and not containing tx_1.

Execution E_3: Before round $\max(t_1, t_2)$, communication among the nodes of $\mathcal{P}_1 \cup \mathcal{P}_2$ is synchronous, and communication among the nodes of $\mathcal{P}_2 \cup \mathcal{P}_3$ is synchronous, but any nodes $p \in \mathcal{P}_1, q \in \mathcal{P}_3$ cannot communicate (asynchrony). Such a delay satisfies partial synchrony with GST $= \max(t_1, t_2)$. The nodes of $\mathcal{P}_1 \cup \mathcal{P}_3$ are honest. The nodes of $\mathcal{P}_2$ behave like the "split-brain" adversary: To the nodes of $\mathcal{P}_1$, they behave like the nodes $\mathcal{P}_2$ in E_2. To the nodes of $\mathcal{P}_3$, they behave like the nodes $\mathcal{P}_2$ in E_1. The environment inputs tx_1 into all nodes of $\mathcal{P}_2 \cup \mathcal{P}_3$, and tx_2 to all nodes of $\mathcal{P}_1 \cup \mathcal{P}_2$. Since E_3 is indistinguishable from E_1 until round $\max(t_1, t_2)$ for the nodes of $\mathcal{P}_3$, and E_3 is indistinguishable from E_2 until round $\max(t_1, t_2)$ for the nodes of $\mathcal{P}_1$, (honest) node $p_3 \in \mathcal{P}_3$ (see E_1) will output a log containing tx_1 and not containing tx_2, and (honest) node $p_1 \in \mathcal{P}_1$ (see E_2) will output a log containing tx_2 and not containing tx_1. This is a safety violation that contradicts the assumption that Π is τ^S-safe under partial synchrony, since $|\mathcal{P}_2| \leq \tau^S$. □

D.1 Proof of Thm. 2

PROOF OF THM. 2. Towards a contradiction, suppose Π satisfies the conditions of Thm. 2 for period length Δ' and is accountably live with appropriate parameters. Pick any even n.

Execution E_1: Consider an execution under synchrony where the crash of the nodes in $\mathcal{P}_1$ with $|\mathcal{P}_1| = n/2$ causes a timely-liveness violation. Such an execution exists according to Lem. 12. Let $\mathcal{P}_2 \triangleq \mathcal{P} \setminus \mathcal{P}_1$ behave honestly. Due to $\tau_{\max}^{\mathrm{AL}} \geq n/2, \tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$, Π purportedly being accountably live, and the timely-liveness violation, honest nodes will eventually produce a certificate of guilt for one of the nodes in $\mathcal{P}_1$.

Execution E_2: The roles of $\mathcal{P}_1$ and $\mathcal{P}_2$ are swapped compared to E_1: $\mathcal{P}_1$ are now honest, and $\mathcal{P}_2$ are adversary. The adversary nodes do not communicate with the honest nodes, but otherwise behave like the honest nodes in E_1.

Because the views (on the protocol execution) of the adversary nodes in E_2 are identical to those of the honest nodes in E_1, the adversary nodes will eventually produce a certificate of guilt for one of the nodes in $\mathcal{P}_1$. But all $\mathcal{P}_1$ are honest in E_2. This is a contradiction to the definition of accountable liveness, as desired. □

D.2 Proof of Thm. 3

PROOF OF THM. 3. Without loss of generality, we consider $x = 1/2$. If $\tau_{\max}^{\mathrm{AL}} \geq n/2$, we are done by Thm. 2. So we assume $\tau_{\max}^{\mathrm{AL}} < n/2$. Towards a contradiction, suppose protocol Π satisfies the conditions of Thm. 3 for some period length Δ'. We require the mild regularity condition that $\Delta' g(\Delta')$ is an even number.

We proceed in two steps: (1) We describe three executions E_A, E_B, and E_C that are indistinguishable in terms of which and when messages are received by honest nodes. We assume that timely-liveness is violated (with a particular set of nodes that do not confirm) in E_A (and due to indistinguishability also in E_B and E_C). We then show that if Π is accountably live with $\tau_{\mathrm{ident}}^{\mathrm{AL}} > 0$, then a certificate of guilt
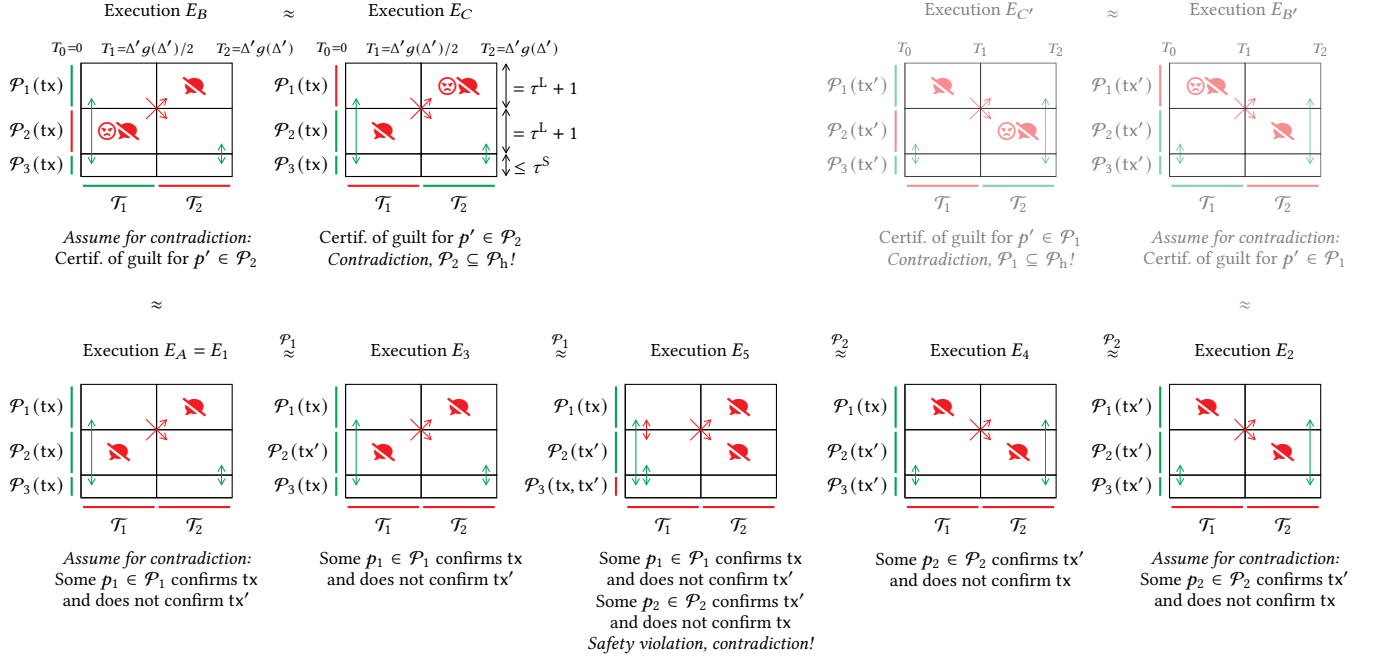
Figure 3: Illustration of indistinguishable executions $E_B, E_C, E_A = E_1, E_2, E_3, E_4, E_5$ **used in the proof of Thm. 3. Nodes (with their respective transaction input at round** 0**) are indicated on the vertical axis, time (from the beginning of round** 0 **to the beginning of round** $\Delta' g(\Delta')$**) is indicated on the horizontal axis. Green/red bars at the edge of the rectangles indicate honest/adversary nodes and synchronous/asynchronous rounds, respectively. Green/red arrows indicate communication between groups of nodes that does/doesn't occur. Red "**🕸**" indicates a network partition, "**☹🕸**" indicates an adversary-emulated network partition.**

must eventually be produced in E_B for an adversary node, but since E_B and E_C are indistinguishable, a certificate of guilt is eventually produced in E_C for the same node, but that node is honest in E_C. This is the contradiction, and thus such Π cannot exist. (2) We show, using five executions E_1, E_2, E_3, E_4, E_5, where $E_A = E_1$, that due to the protocol being optimally-resilient, timely-liveness must indeed be violated (with a particular set of nodes that do not confirm) in E_A, discharging the earlier assumption used in (1).

We start with (1). Partition $\mathcal{P}$ into $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ with $|\mathcal{P}_1| = \tau^L + 1 \leq \tau^{AL}_{max}, |\mathcal{P}_2| = \tau^L + 1 \leq \tau^{AL}_{max}, |\mathcal{P}_3| = n - 2(\tau^L + 1) \leq \tau^S$. See Fig. 3, which illustrates all subsequently detailed executions from the beginning of round 0 to the beginning of round $\Delta' g(\Delta')$. For easy reference to sets of rounds of the executions, let $T_i \triangleq i\Delta' g(\Delta')/2$ for $i \in \{0, 1, 2\}$, $\mathcal{T}_i \triangleq [T_{i-1}, T_i)$ for $i \in \{1, 2\}$, and $\mathcal{T} \triangleq \mathcal{T}_1 \cup \mathcal{T}_2$.

Execution E_A (cf. Fig. 3, "Execution $E_A = E_1$"): All nodes are input tx at time 0. No other transactions are input. All nodes are honest. The network is asynchronous during $\mathcal{T}$ (see Fig. 3), and next-round-delay afterwards for all nodes (not shown in Fig. 3). (By "next-round-delay" we mean every message arrives in the next round of the model. Recall, the network delay bound is Δ rounds.) During asynchrony, communication between any nodes $p \in \mathcal{P}_1, q \in \mathcal{P}_2$ is delayed until T_2. During $\mathcal{T}_1$, communication in $\mathcal{P}_1 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_1. During $\mathcal{T}_2$, communication in $\mathcal{P}_2 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_2.

Note that we are *not invoking accountable liveness* in E_A, and therefore E_A does not have to be x-partially-synchronous, and we may (and do) assume that the network is asynchronous for the entire $\mathcal{T}$. We only assume (for the purposes of part (1), and this assumption is subsequently discharged in part (2)) that there is a timely-liveness violation in E_A. More specifically, we assume that tx is not confirmed by any honest node $p_1 \in \mathcal{P}_1$ by the end of $\mathcal{T}$ in E_A.

Execution E_B (cf. Fig. 3): All nodes are input tx at time 0. No other transactions are input. Nodes in $\mathcal{P}_1 \cup \mathcal{P}_3$ are honest. Nodes in $\mathcal{P}_2$ are adversary. The network is asynchronous during $\mathcal{T}_2$, and otherwise has next-round-delay for all nodes. During $\mathcal{T}$, adversary nodes in $\mathcal{P}_2$ delay the sending to and receiving from nodes in $\mathcal{P}_1$ until T_2. During $\mathcal{T}_1$, adversary nodes in $\mathcal{P}_2$ also delay the sending to and receiving from nodes in $\mathcal{P}_3$ until T_1. During $\mathcal{T}_2$, adversary nodes in $\mathcal{P}_2$ communicate with nodes in $\mathcal{P}_3$ using next-round-delay (and the network asynchrony allows this). During network asynchrony, communication between any node $p \in \mathcal{P}_1$ and any node outside $\mathcal{P}_1$ is delayed until T_2. After T_2, the adversary nodes behave honestly.

Execution E_C (cf. Fig. 3): All nodes are input tx at time 0. No other transactions are input. Nodes in $\mathcal{P}_2 \cup \mathcal{P}_3$ are honest. Nodes in $\mathcal{P}_1$ are adversary. The network is asynchronous during $\mathcal{T}_1$, and otherwise has next-round-delay for all nodes. During $\mathcal{T}$, adversary nodes in $\mathcal{P}_1$ delay the sending to and receiving from nodes in $\mathcal{P}_2$ until T_2. During $\mathcal{T}_2$, adversary nodes in $\mathcal{P}_1$ also delay the sending to and receiving from nodes in $\mathcal{P}_3$ until T_2. During $\mathcal{T}_1$, adversary nodes

in $\mathcal{P}_1$ communicate with nodes in $\mathcal{P}_3$ using next-round-delay (and the network asynchrony allows this). During network asynchrony, communication between any node $p \in \mathcal{P}_2$ and any node outside $\mathcal{P}_2$ is delayed until T_1. After T_2, the adversary nodes behave honestly.

Note that E_A, E_B, E_C are indistinguishable to honest nodes because they receive the same messages at the same rounds during each of the executions. Recall that we have assumed a timely-liveness violation in E_A, where for every $p \in \mathcal{P}_1$, tx $\notin \Lambda^p_{\Delta'g(\Delta')-1}$, i.e., p has not confirmed tx by the end of $\mathcal{T}$ in E_A. So there must be a timely-liveness violation in E_B, and eventually a message is produced in E_B that identifies an adversary node $p' \in \mathcal{P}_2$ as guilty for the timely-liveness violation, because Π is assumed to be accountably live with $\tau^{\text{AL}}_{\text{ident}} > 0$. Then, due to E_B and E_C being indistinguishable in terms of which and when messages are produced, eventually a message is produced in E_C that identifies $p' \in \mathcal{P}_2$ as guilty for a timely-liveness violation. (Note that there was no timely-liveness violation in E_C because $\mathcal{P}_1$ are adversary in E_C. Regardless, due to the indistinguishability of E_B and E_C, honest nodes go along with the adversary nodes to eventually produce a certificate of guilt for $p' \in \mathcal{P}_2$ in E_C.) But that is a contradiction because $\mathcal{P}_2$ are honest in E_C.

We now proceed to (2), showing that E_A must indeed have a timely-liveness violation (namely where none of $\mathcal{P}_1$ confirms) if Π is optimally-resilient. We consider five executions E_1, E_2, E_3, E_4, E_5, where $E_A = E_1$, see Fig. 3.

Execution E_1 (cf. Fig. 3, "Execution $E_A = E_1$"): All nodes are input tx at time 0. No other transactions are input. All nodes are honest. The network is asynchronous during $\mathcal{T}$, and next-round-delay afterwards for all nodes. During asynchrony, communication between any nodes $p \in \mathcal{P}_1, q \in \mathcal{P}_2$ is delayed until T_2. During $\mathcal{T}_1$, communication in $\mathcal{P}_1 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_1. During $\mathcal{T}_2$, communication in $\mathcal{P}_2 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_2.

Execution E_2 (cf. Fig. 3): All nodes are input tx′ at time 0. No other transactions are input. All nodes are honest. The network is asynchronous during $\mathcal{T}$, and next-round-delay afterwards for all nodes. During asynchrony, communication between any nodes $p \in \mathcal{P}_1, q \in \mathcal{P}_2$ is delayed until T_2. During $\mathcal{T}_1$, communication in $\mathcal{P}_2 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_1. During $\mathcal{T}_2$, communication in $\mathcal{P}_1 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_2.

Execution E_3 (cf. Fig. 3): Same as E_1, except nodes $\mathcal{P}_2$ are input tx′ at time 0.

Execution E_4 (cf. Fig. 3): Same as E_2, except nodes $\mathcal{P}_1$ are input tx at time 0.

Execution E_5 (cf. Fig. 3): Nodes $\mathcal{P}_1$ are input tx, nodes $\mathcal{P}_2$ are input tx′, and nodes $\mathcal{P}_3$ are input tx, tx′ at time 0. Nodes $\mathcal{P}_1 \cup \mathcal{P}_2$ are honest. Nodes $\mathcal{P}_3$ are adversary. The network is asynchronous during $\mathcal{T}$, and next-round-delay afterwards for all nodes. During asynchrony, communication between any nodes $p \in \mathcal{P}_1, q \in \mathcal{P}_2$ is delayed until T_2. During $\mathcal{T}_1$, communication in $\mathcal{P}_2 \cup \mathcal{P}_3$ is next-round-delay and communication in $\mathcal{P}_1 \cup \mathcal{P}_3$ is next-round-delay. All other communication is delayed until T_2. Nodes $\mathcal{P}_3$ perform a split-brain attack, behaving to $\mathcal{P}_1$ like in E_3 and to $\mathcal{P}_2$ like in E_4.

Towards a contradiction, assume that there is no timely-liveness violation in E_1 and in E_2 where none of $\mathcal{P}_1$ and none of $\mathcal{P}_2$ confirm

tx and tx′, respectively, i.e., by time $\Delta'g(\Delta')$, some honest node $p_1 \in \mathcal{P}_1$ has confirmed tx and not confirmed tx′ in E_1 (because they had no way of knowing tx′ which is assumed to be high-entropy), and some honest node $p_2 \in \mathcal{P}_2$ has confirmed tx′ and not confirmed tx in E_2 (because they had no way of knowing tx which is assumed to be high-entropy). (If either E_1 or E_2 has a timely-liveness violation where either none of $\mathcal{P}_1$ or none of $\mathcal{P}_2$ confirm, we can take that as E_A and adjust (1) accordingly and we are done—which is hinted in Fig. 3 with executions $E_{B'}, E_{C'}$.) Note that from the perspective of the nodes $\mathcal{P}_1$, E_1 and E_3 are indistinguishable until T_2, thus $p_1 \in \mathcal{P}_1$ confirms tx in E_3 and does not confirm tx′ in E_3 by time $\Delta'g(\Delta')$. Note that from the perspective of the nodes $\mathcal{P}_2$, E_2 and E_4 are indistinguishable until T_2, thus $p_2 \in \mathcal{P}_2$ confirms tx′ in E_4 and does not confirm tx in E_4 by time $\Delta'g(\Delta')$. Note also that from the perspective of the nodes $\mathcal{P}_1$, E_3 and E_5 are indistinguishable until T_2, and from the perspective of the nodes $\mathcal{P}_2$, E_4 and E_5 are indistinguishable until T_2. Thus, $p_1 \in \mathcal{P}_1$ confirms tx in E_5 and does not confirm tx′ in E_5 by time $\Delta'g(\Delta')$. Likewise, $p_2 \in \mathcal{P}_2$ confirms tx′ in E_5 and does not confirm tx in E_5 by time $\Delta'g(\Delta')$. Since both p_1 and p_2 are honest in E_5, this constitutes a safety violation, even though $|\mathcal{P}_3| \leq \tau^{\text{S}}$ in E_5, giving the required contradiction. Thus, there must be a timely-liveness violation in either E_1 or E_2 where either none of $\mathcal{P}_1$ or none of $\mathcal{P}_2$ confirm, and we can use that as E_A in (1). □

D.3 Proof of Thm. 4

Proof of Thm. 4. Towards a contradiction, suppose for some $k \geq 3$, Alg. 1 with some liveness accountability mechanism satisfies the conditions of Thm. 4 for $x = 1/k$ (which is without loss of generality) with $(\tau^{\text{AL}}_{\text{ident}} - 1) \geq (\tau^{\text{L}} + 1) - \left\lfloor \frac{\tau^{\text{AL}}_{\max} - (\tau^{\text{L}} + 1)}{k-2} \right\rfloor$. Then $\mathcal{P}$ can be partitioned into $\mathcal{P}_0, \mathcal{P}_1, ..., \mathcal{P}_k, \mathcal{P}_{k+1}$, such that $|\mathcal{P}_0| = \tau^{\text{AL}}_{\text{ident}} - 1$, $|\mathcal{P}_i| = \left\lfloor \frac{\tau^{\text{AL}}_{\max} - (\tau^{\text{L}} + 1)}{k-2} \right\rfloor$ for $i \in \{1, ..., k\}$, $|\mathcal{P}_{k+1}| = n - \sum_{i=0}^{k} |\mathcal{P}_i|$. Note that for every $i \in \{1, ..., k\}$, $|\mathcal{P}_0| + |\mathcal{P}_i| \geq \tau^{\text{L}} + 1 > \tau^{\text{L}}$, and $|\mathcal{P}_0| + \sum_{i=0}^{k-1} |\mathcal{P}_i| \leq \tau^{\text{AL}}_{\max}$, $|\mathcal{P}_0| < \tau^{\text{AL}}_{\text{ident}}$.

We require the mild regularity condition that k divides $\Delta'g(\Delta')$. Partition the first $\Delta'g(\Delta')$ rounds into k equally-sized intervals $\mathcal{T}_i$ for $i \in \{1, ..., k\}$: Let $T_i \triangleq i\Delta'g(\Delta')/k$ for $i \in \{0, ..., k\}$, $\mathcal{T}_i \triangleq [T_{i-1}, T_i)$ for $i \in \{1, ..., k\}$, and $\mathcal{T} \triangleq \bigcup_{i=1}^{k} \mathcal{T}_i$.

Now consider the executions $E_{k,i}$ for $i \in \{1, ..., k\}$ (see Fig. 4 for an illustration), where in the i-th execution the following holds: $\mathcal{P}_0$ are crashed, $\mathcal{P}_{k+1}$ are honest, $\mathcal{P}_i$ are honest, and all other nodes are adversary. All nodes are input a high-entropy transaction tx at round 0. Furthermore, $\mathcal{T}_{k-i+1}$ is asynchronous, the remaining rounds are next-round-delay. During asynchrony, messages to, from, and within $\mathcal{P}_i$ are delayed until the end of asynchrony; all other messages have next-round-delay. For every $j \in \{1, ..., k\} \setminus \{i\}$, the non-crashed adversary nodes in $\mathcal{P}_j$ behave during $\mathcal{T}_{k-j+1}$ as if messages to, from, and within $\mathcal{P}_j$ are delayed until the end of $\mathcal{T}_{k-j+1}$, i.e., they delay processing incoming messages and sending messages until the end of $\mathcal{T}_{k-j+1}$. Other than the extra delay, adversary nodes follow the protocol. Observe that all executions $E_{k,i}$ for $i \in \{1, ..., k\}$ are indistinguishable from the perspective of honest nodes in terms of which messages they receive and when.

Observe that in every execution $E_{k,i}$, according to Alg. 1, tx is not confirmed by any of the honest nodes by T_k, i.e., by the beginning
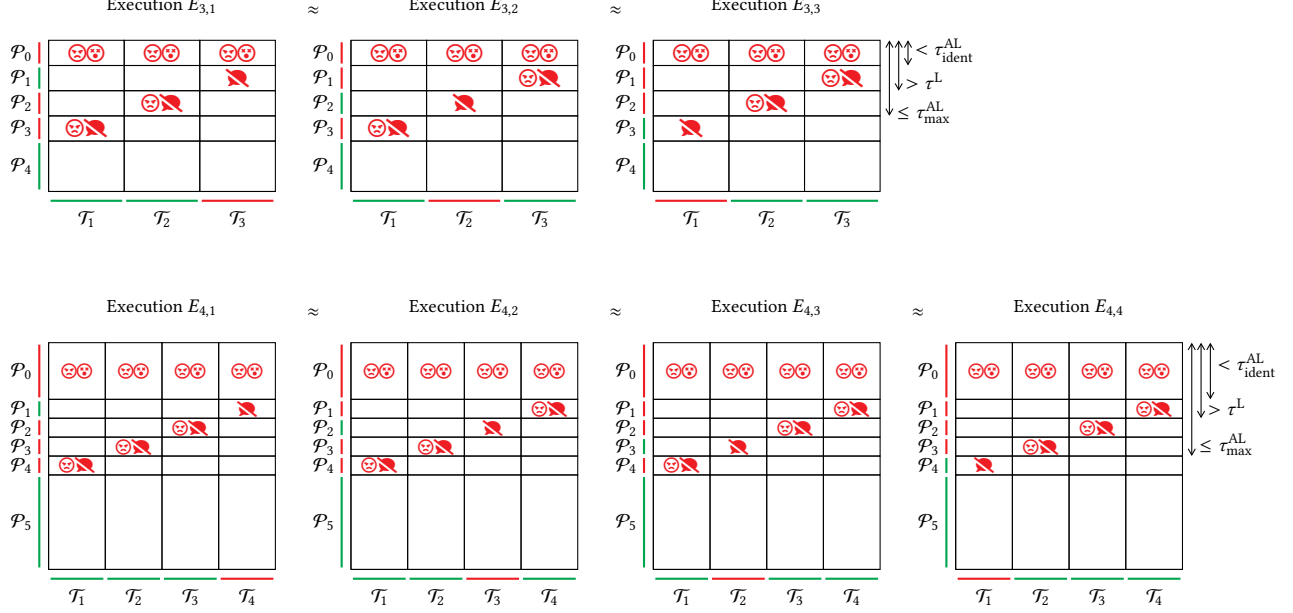
Figure 4: Illustration of indistinguishable executions used in Thm. 4, for $k = 3$ and $k = 4$. See caption of Fig. 3 for legend. Red "☹☹" indicates adversary-emulated crashed nodes.

of round $\Delta' g(\Delta')$. This is because for every view v before $\Delta' g(\Delta')$, the following holds (let $t_v \triangleq 12\Delta v$): The block b_v proposed by L_v at $t_v + 2\Delta$ (Alg. 1, ln. 13) reaches, by $t_v + 4\Delta$, only L_v (if for some $j \in \{1, ..., k\}$, $t_v \in \mathcal{T}_{k-j+1}$ and $L_v \in \mathcal{P}_0 \cup \mathcal{P}_j$) or less than $n - \tau^{\mathrm{L}}$ nodes (otherwise). In any case, b_v reaches at most $2n/3$ nodes (since $\tau^{\mathrm{L}} = \lfloor (n-1)/3 \rfloor$ for Alg. 1, cf. Lem. 2) by $t_v + 4\Delta$. But no node votes stage-1 for b_v after $t_v + 4\Delta$ (Alg. 1, ln. 17). As a result, at most $2n/3$ stage-1 votes are ever produced for b_v according to Alg. 1, and thus no stage-1 QC is ever produced for b_v, and thus b_v is never confirmed (Alg. 1, ln. 8). This implies a timely-liveness violation.

Recall that Alg. 1 was assumed to be equipped with a liveness accountability mechanism that would render it accountably live with sensitivity $\tau^{\mathrm{AL}}_{\mathrm{ident}}$, so in all of the executions $E_{k,i}$ (for our purposes here it suffices that this is the case in one of the executions), eventually, a proof of guilt is produced for $\tau^{\mathrm{AL}}_{\mathrm{ident}}$ adversary nodes. Since $|\mathcal{P}_0| = \tau^{\mathrm{AL}}_{\mathrm{ident}} - 1 < \tau^{\mathrm{AL}}_{\mathrm{ident}}$, it follows that there must be a proof of guilt for some node not in $\mathcal{P}_0$. But then by indistinguishability of the executions, there is an execution among the $E_{k,i}$ where that node is honest, yet a certificate of guilt is produced for that node. This is the desired contradiction. □

D.4 Proof of Thm. 5

PROOF OF THM. 5. We follow the steps of the proof of Thm. 4. Only the network delay (both real and adversary-emulated) in the executions $E_{k,i}$ is slightly adjusted to leverage the now-or-never property.

Partition the first $\Delta' g(\Delta')$ rounds into equally-sized intervals $\mathcal{T}^{(v)}$ of length Δ''. This is possible because Δ'' divides $\Delta' g(\Delta')/k$ by assumption. Consider the executions $E_{k,i}$ of the proof of Thm. 4. The adversary nodes in $\mathcal{P}_0$ no longer crash. During asynchrony, within each $\mathcal{T}^{(v)}$, messages to, from, and within $\mathcal{P}_i \cup \mathcal{P}_0$ are delayed

until the end of $\mathcal{T}^{(v)}$. For every $j \in \{1, ..., k\} \setminus \{i\}$, the non-crashed adversary nodes in $\mathcal{P}_j \cup \mathcal{P}_0$ behave during $\mathcal{T}_{k-j+1}$, within each $\mathcal{T}^{(v)}$, as if messages to, from, and within $\mathcal{P}_j \cup \mathcal{P}_0$ are delayed until the end of $\mathcal{T}^{(v)}$. Observe that all executions $E_{k,i}$ for $i \in \{1, ..., k\}$ are still indistinguishable from the perspective of honest nodes in terms of which messages they receive and when.

In every execution $E_{k,i}$ and every $\mathcal{T}^{(v)}$, $\lceil n/3 \rceil$ nodes are partitioned off temporarily until the end of $\mathcal{T}^{(v)}$. Thus, by the now-or-never property, tx is not confirmed by any of the honest nodes by T_k, i.e., by the beginning of round $\Delta' g(\Delta')$. This implies a timely-liveness violation. The rest of the proof proceeds as in the proof of Thm. 4. □

E Additional Related Work

We survey additional related works beyond those discussed in Sec. 7. Achieving consensus among distributed nodes is a decades-old problem, traditionally defined by two fundamental properties: *safety* and *liveness* [16, 17]. Safety ensures that honest nodes never diverge on decided values, while liveness guarantees that decisions are eventually reached, all despite possible asynchrony and up to a threshold of adversary nodes.

Accountable Safety. With the advent of blockchain technology, and in particular proof-of-stake (PoS) protocols, a stronger notion of safety called *accountable safety* has emerged [10, 14, 19, 28, 29, 37–39, 43–45]. While accountable safety keeps the traditional requirement that decisions remain consistent under an adversarial threshold, it also allows to *identify* specific misbehaving nodes in the event of a safety violation. This property is especially powerful in PoS settings, where such identification can trigger financial penalties

(*slashing* [14]) on adversary nodes, thus creating strong economic incentives to follow the protocol [12, 43].

A key challenge arises when extending accountable safety to systems with *dynamic participation* [33]. A recent work [38] formalizes an availability-accountability dilemma, proving that no protocol can remain fully accountably-safe while also guaranteeing liveness if the active set of nodes fluctuates like in the sleepy model [40]. Neu *et al.* design an accountability gadget that checkpoints a longest-chain protocol and can be combined with any BFT protocol providing accountable safety under static participation, thus addressing this dilemma in practical settings.

Slashing. While existing PoS protocols with accountable safety can identify adversary nodes, they do not always ensure that those nodes' stakes are actually slashed—thereby falling short of providing slashable safety. In particular, adversary nodes can exploit posterior corruption attacks by reusing previously held stakes after withdrawal, making slashing ineffective [22]. To address this gap, Tas *et al.* [49, 50] and Azouvi and Vukolić [6] propose leveraging Bitcoin [36] as a *checkpointing* mechanism, anchoring critical PoS states within Bitcoin's immutable ledger. This design prevents adversary nodes from rewriting or invalidating older blocks—thereby circumventing long-range attacks—once the stake is withdrawn. Simultaneously, if a safety violation is detected in recent PoS blocks, the protocol can identify and slash the responsible nodes.

Recovery Mechanisms. Another important aspect of accountable safety involves *recovery* following a slashable event. In blockchain systems like Ethereum, for instance, a major slashing often relies on social consensus to coordinate recovery—potentially including hard forks to stabilize the network. To formalize and automate this process, Lewis-Pye and Roughgarden [34] propose a *wrapper* approach that runs an execution of an accountably-safe, optimally-resilient SMR protocol until a consistency violation occurs. When this occurs, the wrapper initiates a *recovery procedure* to reach consensus on a set of adversary nodes for which proof of misbehavior exists, along with a long initial segment of the log generated by, below which no consistency violations have been detected. The wrapper then restarts the protocol with the adversary nodes removed. Gong *et al.* [27] further explore recovery under alive-but-corrupt nodes and partial synchrony. These works mark initial steps toward automating post-slashing recovery.

Responsive and Network Adaptive Protocols. Responsive synchronous protocols [3, 41] operate under synchrony assumptions but can improve their latency if actual message delivery happens to be faster than the network's delay bound. For instance, Thunderella [41] achieves near-instant confirmation via an asynchronous fast path under favorable conditions but falls back to a slow synchronous path if responsiveness fails, ensuring safety and liveness. These protocols leverage a *fast path* to achieve low latency when the network is responsive while falling back to a conservatively timed execution when conditions degrade. While such protocols do not achieve full partial-synchrony safety (i.e., safety under arbitrarily long message delays), for any $x < 1$, the implied synchronous model allows them to operate at the Δ^* time scale when network conditions are not optimal, benefiting from the fast path whenever possible. On the other hand these protocols have a tradeoff between resilience on the responsive fast path and the synchronous fallback. A sequence of works [7–9] explored whether it is possible to design Byzantine-fault tolerant protocols that tolerate more than one-third Byzantine faults under synchrony, while still ensuring resilience to a certain fraction of faults—ideally up to one-third—in asynchrony or partial synchrony (the fast path). Their results establish that a BFT protocol can simultaneously tolerate $f_a < \frac{n}{3}$ faults under asynchrony and $f_a \leq f_s < \frac{n}{2}$ faults under synchrony, if and only if the condition $2f_s + f_a < n$ holds. Momose and Ren [35] further demonstrate that it is possible to separate fault tolerance thresholds for different timing models and for safety and liveness. Specifically, they show that safety under synchrony can be improved while still preserving other fault thresholds, including liveness under synchrony and both safety and liveness under asynchrony.